Opinnäytetyö (AMK)

Elektroniikan koulutusohjelma

Elektroniikkasuunnittelu

2012

Miikka Salokannel

METSÄKONEEN HAKKUUPÄÄN SIMULAATTORI

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

Turun ammattikorkeakoulu

Elektroniikan koulutusohjelma | Elektroniikkasuunnittelu

Toukokuu 2012 | 29 sivua

Ohjaaja: TkT Timo Tolmunen

Miikka Salokannel

METSÄKONEEN HAKKUUPÄÄN SIMULAATTORI

Tämän opinnäytetyön tarkoituksena oli suunnitella ja rakentaa metsäkoneen hakkuupään ohjaus- ja mittausjärjestelmälle simulaattori. Suunnittelun lähtökohtana oli simulaattorin käyttäminen järjestelmän kehittäneen yrityksen tuotekehityksessä sovelluksen testausvälineenä. Simulaattorin toteutuksen tuli myös mahdollistaa ohjaus- ja mittausjärjestelmän esittelyn eri tilaisuuksissa.

Simulaattori koostuu neljästä moduulista, joista kolme on ohjaus- ja mittausjärjestelmän moduuleita. Neljäs moduuli on simuloitavat signaalit tuottava moduuli, johon on toteutettu CoDeSys sovelluskehitys- ja ohjelmointityökalulla sovellus.

Jokaista ohjaus- ja mittausjärjestelmän lähtöä kuvataan LEDillä, joka syttyy lähdön aktivoituessa. Jokaista tuloa kuvataan omalla painokytkimellä, joka painettaessa aktivoi tulon. Myös rotaattorin liikkeitä ohjaavat joystickit on korvattu pelkillä painokytkimillä tilan säästämisen vuoksi. Simuloivan moduulin sovellus tuottaa hakkupäissä olevien kaksitila-anturien NPN- ja PNP-tyyppiset signaalit sekä kaksikanavaisten pulssianturien tuottamat pulssijonot pituuden, halkaisijan ja sahausliikkeen kuvaamiseksi.

Simulaattorin mekaniikka toteutettiin muovisalkkuun, joka mahdollistaa simulaattorin helpon kuljettamisen sekä antaa mekaanisen suojan simulaattorille.

Lopputuloksena saatiin kompakti simulaattori, jolla pystytään testaamaan ohjaus- ja mittausjärjestelmän sovellusta sekä esittelemään järjestelmän ominaisuuksia ja fyysistä rakennetta.

ASIASANAT:

simulointi, mallintaminen, metsäkoneet

BACHELOR´S THESIS | ABSTRACT
TURKU UNIVERSITY OF APPLIED SCIENCES

Degree Proramme in Electronics | Electronics design

May 2012| 29 pages

Instructor: Timo Tolmunen D.Sc (Tech.)

Miikka Salokannel

HARVESTER HEAD SIMULATOR

This bachelor's thesis purpose was to design and build harvester head simulator for company that develops logging systems. Simulator is mainly used as a testing tool for logging system application at research and development department. Simulator structure should also make possible demonstrations of the logging system at different occasions.

Simulator consist four modules and three of them are logging system modules. Fourth module is simulating module, where is the application executed with CoDeSys software tool.

Every output in logging system is represented with LED, which lights when output is activated. Every input in logging system is represented with push-button, which activates input when it is pushed. Also joysticks, which control rotator movements, are represented with push-buttons to make simulator more compact. Simulating module generates harvester heads typical sensor signals like switching sensors NPN- and PNP-signals and incremental encoders 2-channel quadrature coded signals for length, diameter and saw-state measuring.

Simulator mechanics were implemented in plastic briefcase so that simulator is easy to transport and briefcase gives mechanical shield for simulator.

Result was a compact simulator, which is capable to test logging systems application and also to demonstrate system features and physical appearance.

KEYWORDS:

simulation, modeling, forest machinery

SISÄLTÖ

LIITTEET

LYHENNELUETTELO

CAN	Ajoneuvoteollisuuden kehittämä ja käyttämä dataväylä laitteiden väliseen kommunikointiin. Väylä koostuu kahdesta parikierretystä johdosta, joista käytetään nimiä CAN-high ja CAN-low (Controller Area Network)
CoDeSys	Saksalaisen 3S-Smart Sofware Solutions kehittämä laitteistoriippumaton IEC 61131-3 standardin PLC-laitteiden sovelluskehitys- ja ohjelmointityökalu (Controller Development System)
IEE	Ammatillinen yhteisö, joka omistautunut tukemaan teollisuuden insinöörin ammattia ja yksilöitä sekä on mukana kehittämässä laatua ja tuottavuutta (Institute of Industrial Engineers)
I/O	Tulot ja lähdöt (Input/Output)
LDC	Technion Oy:n kehittämän ohjaus- ja mittausjärjestelmän näyttömoduuli (Logger Display Controller)
LHC	Technion Oy:n kehittämän ohjaus- ja mittausjärjestelmän hakkuupään moduuli (Logger Head Controller)
LHI	Technion Oy:n kehittämän ohjaus- ja mittausjärjestelmän liityntäyksikkö (Logger Harvester Interface)
PLC	Laajalti teollisuuden automaatiosovelluksissa ja ajoneuvoteollisuuden ohjausjärjestelmissä käytetty mikroprosessori-pohjainen laite, jossa on sovelluskohtainen määrä tulo- ja lähtöportteja ohjaamassa koneen toimintaa. PLC-laitteet ovat vahvasti korvanneet aiemmin käytettyä releteknіikkaa (Programmable Logic Controller)
RS-232	Sarjamuotoinen kommunikointiväylä (Recommend Standard 232)
TCC1010	Technion Oy:n kehittämä PLC-laite työkone- ja teollisuusympäristöön (Technion Can Controller)

1 JOHDANTO

Metsäkoneiden kehitys on alkanut Pohjois-Amerikassa jo 1800-luvun lopulla, mutta Eurooppaan ja pohjoismaihin nämä metsäkoneet saapuivat kuitenkin vasta 1900-luvun alussa. Metsäkoneiden ja hakkuupäiden kehitys jatkui nopeasti kaatokouran prototyypistä harvestereihin. Samalla syntyivät ensimmäiset reletekniikkaan perustuvat mittausjärjestelmät, jotka muuttuivat myöhemmin mikroprosessoripohjaisiksi. Ensimmäinen harvesterin virtuaalisimulaattori nähtiin 1990-luvulla.

Nykyisin metsäkoneiden ohjaus- ja mittausjärjestelmien kehittäjillä on omia simulaattoreita, joiden laajuus vaihtelee käyttökohteen mukaan toimintoja esittelevästä demolaitteesta metsäkoneenkuljettajan koulutussimulaattoriin. Simulaattorit ovat hyvin spesifisiä ja soveltuvat käytettäväksi vain ohjaus- ja mittausjärjestelmän kehittäjän järjestelmässä.

Tämän opinnäytetyön lähtökohtana oli suunnitella ja toteuttaa Technion Oy:n kehittämälle metsäkoneen hakkuupään ohjaus- ja mittausjärjestelmälle simulaattori, jota on mahdollista käyttää tuotekehityksessä sovelluksen testausvälineenä. Simulaattorin toteutuksen tulisi myös mahdollistaa järjestelmän esittelyn asiakkaille. Technion Oy:n aikaisempi simulointitapa on hyvin paikkasidonnainen ja simulaattorin kopioitavuus on hankalaa. Nämä ovat aiheuttaneet tarpeen kompaktille simulaattorille.

Opinnäytetyössä käsitellään simulaattorin spesifikaatiot, jotka ovat yrityksen ohjelmistosuunnittelijan määrittelemiä ja perustuvat eri hakkuupäiden valmistajien tapoihin toteuttaa ohjaus- ja mittaustoiminnot hakkuupäässä. Spesifikaatioiden pohjalta on suunniteltu ja toteutettu simulaattorin mekaaninen ja sähköinen toteutus sekä sovellus, jolla simuloidaan hakkuupään liikkeitä.

2 MALLINTAMINEN JA SIMULOINTI

Simulointi ja mallintaminen ovat termeinä melko uusia, mutta tieteessä on jo pitkään käytetty todellisuuden ilmiöiden ja järjestelmien kuvaamiseen niin matemaattista mallinnusta kuin laboratoriossa tehtävää simulointia. Molemmista termeistä puhutaan usein samassa yhteydessä, koska asiat ovat hyvin lähellä toisiaan. Esimerkiksi fyysisten mallien tekemistä ei voida kutsua simuloinniksi, mutta matemaattisten mallien tekemistä voidaan. Samalla tavalla tietokonesimulaatiota ei voida kutsua mallintamiseksi, mutta matemaattista simulointia voidaan. Järjestelmästä tehtävä mallinnus on fyysinen tai matemaattinen kopio itse järjestelmästä, jossa on järjestelmän kaikki ominaisuudet ja toiminnot. Simulaatio on taas prosessi, joka simuloi laboratoriossa tai tietokoneella varsinaisen tilanteen niin järjestelmän kaltaisesti kuin mahdollista. Voidaan siis sanoa, että mallintaminen on staattinen kuvaus järjestelmästä ja simulointi on kuvaus ajan vaikutuksesta järjestelmään. [1, 2]

Simuloinnin ja mallintamisen käyttämiseen voi olla monia eri syitä. Vuonna 1998 IEE (Institute of Industrial Engineers) listasi oman näkemyksenä simulointiin ja mallintamiseen liittyvistä eduista ja haitoista. Listan perusteella monien on helppo hyväksyä syyt, miksi simulointia ja mallintamista tulisi käyttää osana tutkimuksia ja harjoittelua. Etuina IEE näki mm. seuraavat asiat [1]:

- Jokaisen ehdotetun vaihtoehdon testaaminen mahdollistaa oikean valinnan tekemisen kuitenkaan kuluttamatta ylimääräisiä resursseja.
- Ajan kulkuun vaikuttaminen mahdollistaa tapahtumien hidastamisen tai nopeuttamisen perusteellisten tutkimuksien helpottamiseksi.
- Syy-seuraussuhteita voidaan ymmärtää paremmin uudelleen rakentamalla järjestelmä, seuraamalla sen tapahtumia tarkkaan ja hallinnoimalla järjestelmää itse.
- Toimintatapojen ja -menetelmien eri mahdollisuuksien tutkiminen on mahdollista häiritsemättä tai keskeyttämättä varsinaista järjestelmää.
- Auttaa ymmärtämään monimutkaisten järjestelmien muuttujien välisen yhteyden, joka mahdollistaa ongelmien ratkaisun.

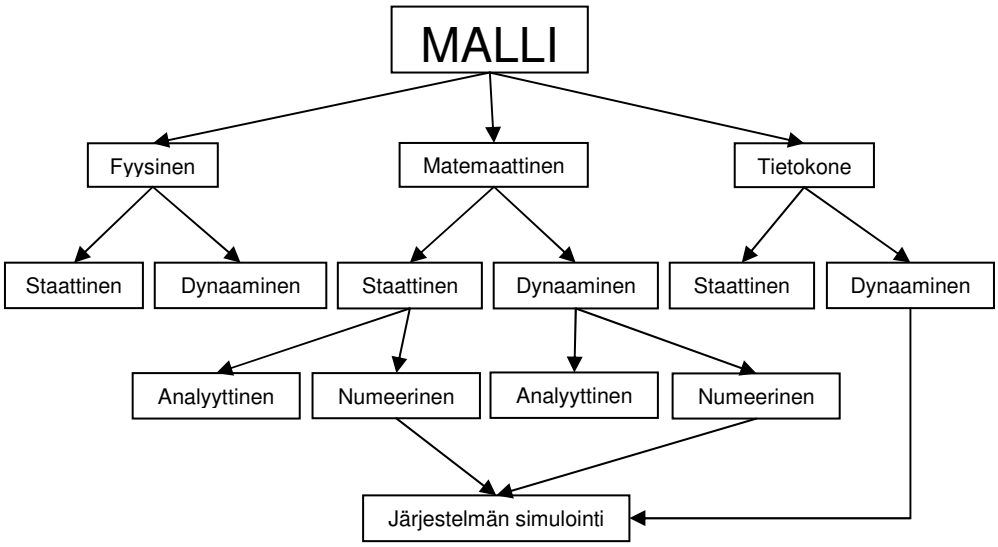
- Prosessin viiveiden, tietojen ja materiaalien tarkastelu auttaa tunnistamaan, onko järjestelmässä havaittu rajoite ilmiö vai itse aiheutettu.
- Saadaan parempi ymmärrys järjestelmästä seuraamalla sen toimintaa kuin ennustamalla, miten sen tulisi toimia.
- Suunnitelman visualisointi käyttämällä animaatioita helpottaa järjestelmän oikean toiminnan seuraamista.
- Auttaa muutoksen valmistelussa vastaamalla jo suunnitteluvaiheessa ”Mitä jos” -kysymykseen.
- Simuloitu tutkimus maksaa huomattavasti vähemmän kuin oikean järjestelmän muuttaminen tai muokkaaminen.
- Laadukkaampi harjoittelu voidaan tehdä vähemmin kustannuksin ja häiriöin verrattuna käytännön harjoitteluun.

Muita etuja syntyy myös silloin, kun kyseessä on toteutukseltaan vaarallinen järjestelmä tai ilmiö. Vanhimpia käyttökohteita ovat sodankäyntiin liittyvät mallintamiset ja simuloinnit, joita käytettiin antiikin Roomassa taistelujen tekniikoiden ja strategioiden harjoitteluun. Silloin jo ymmärrettiin tietynlainen simuloinnin tarve harjoittelun muodossa, jotta ei menetettäisi sotilaita ennen varsinaista taistelua. Tämän päivän esimerkkinä vastaavasta vaarallisesta toteutuksesta toimii lentäjien koulutus, jossa lentokonesimulaattori on merkittävässä roolissa harjoittelun alkuvaiheessa. Nykyään myös eettisesti arveluttavissa tutkimuksissa, kuten ihmisen tai eläimen anatomian tutkiminen, mallintamisen ja simuloinnin käyttämisen mahdollisuus on suuri etu. [1, 3, 4]

Simuloinnin ja mallintamisen haittoja IEE listasi selkeästi vähemmän. Listalta löytyi esimerkiksi erikoiskoulutus, jota tarvitaan monimutkaisten mallien tekemiseen. Myös tuloksien analysointi on vaikeaa, kun tulokset ovat havaintoja järjestelmän vuorovaikutuksesta tai satunnaisuudesta. Simulointi ja mallintaminen voi olla myös kallista sekä aikaa vievää, kun sitä käytetään väärin tilanteessa, jossa analyyttinen ratkaisu olisi paras. [1]

2.1 Mallintaminen

Mallit voidaan jakaa kuvan 1 mukaan kolmeen ryhmään: fyysiset mallit, matemaattiset mallit ja tietokonemallit. Jokainen ryhmä voidaan vielä jakaa staattisiin ja dynaamisiin malleihin. [2]



Kuva 1. Mallien ryhmät. [2]

Fyysiset mallit ovat tyypillisesti pienoismalleja järjestelmistä, jotka ovat ajasta riippumattomia. Esimerkiksi kaupoista löytyvät pienoismallit autoista, helikoptereista ja lentokoneista ovat staattisia fyysisiä malleja. Dynaaminen fyysinen malli on taas järjestelmä, joka on riippuvainen ajasta. Autot, helikopterit ja lentokoneet testataan tyypillisesti tuulitunnelissa niiden aerodynaamisten ominaisuuksien selvittämiseksi sekä varmentamiseksi. Näissä testeissä kokeillaan eri ilmavirran voimakkuuksia, jotta voidaan mitata antureiden avulla paineprofiileja kyseessä olevalle laitteelle. Ilmavirran voimakkuus muuttuu ajan funktiona ja on näin esimerkki dynaamisesta fyysisestä mallista. [1, 2]

Suurin osa järjestelmien fyysisistä malleista voidaan muuntaa helposti matemaattisiksi malleiksi. Kun järjestelmästä tehdään matemaattinen yhtälö, sitä voidaan kutsua matemaattiseksi malliksi. Näin ovat tutkijat tehneet jo hyvin pitkään tutkiessaan esimerkiksi luonnonilmiöitä ja yrittäessään selittää niiden syntyä sekä käyttäytymistä. Staattinen matemaattinen malli on yhtälö, joka ei

sisällä aikaa muuttujana. Kun yhtälöön lisätään aika muuttujaksi, yhtälö muuttuu dynaamiseksi matemaattiseksi malliksi. [2]

Mallintamisen ja simuloinnin käsite on muuttunut täysin tietokoneiden aikakaudella. Nykyään kaikki stokastiset ja jatkuvat matemaattiset mallit voivat saada numeerisia arvoja, kun käytetään apuna numeerisia ratkaisutapoja käyttävää tietokonetta. Tätä ratkaisumallia kutsutaan tietokonemallintamiseksi. [2]

2.2 Simulointi

Simulaation merkityksestä käsitteenä on monia eri variaatioita. Simulaatiota voidaan pitää tieteen kolmantena työkaluna teorian ja kokeellisen toiminnan ohella. Toisaalta simulaatiota pidetään tietokoneen avulla tehtävänä numeerisena ratkaisumenetelmänä, joka sisältää määritellyn ajanjakson aikana tapahtuvia matemaattisia ja loogisia malleja. Tämän takia numeerista laskentaa kutsutaan helposti simulaatioksi, vaikka numeerinen laskenta antaa vain yhden muuttujan variaatiot suhteessa muihin, eikä koko tapahtumaa huomioiden myös ajan. Simulaatio voi kuitenkin myös olla ”viimeinen vaihtoehto”, jota tulisi käyttää kaikkien muiden tapojen epäonnistuessa. Kuitenkin yhteinen tekijä kaikissa määrittelyissä on aika. [1, 2]

Simulaatiot voidaan jakaa kahteen ryhmään: diskreetit tapahtumasimulaatiot ja jatkuvat simulaatiot. Molemmilla on erilainen tapa lisätä aika mukaan malliin, jota simuloidaan. Diskreetissä simulaatiossa tiettyjen tapahtumien esiintyminen peräkkäin muuttaa mallin tilasta toiseen ajan kuluessa, kun taas jatkuva simulaatio kuvaa mallin käyttäytymisen katkeamattomana sarjana ajasta riippumattomia tapahtumia. Simulaatioiden eroa voidaan kuvata auton ja liikennevalojen mallilla. Liikennevalo voi esiintyä kolmessa eri tilassa: punainen, keltainen tai vihreä. Normaalisti se vaihtaa tilaa, kun sisäisessä ajastimessa on kulunut tietty aika, joka laukaisee tilan muutoksen. Auto taas ei voi vaihtaa liikennevalojen tapaan nopeuttaan välittömästi tilasta toiseen. Sen täytyy käydä läpi kiihdytys, joka määritellään nopeuden muutos ajan funktiona. Tämä

nopeuden muutos on selkeästi enemmän jatkuva-aikainen muutos kuin yksittäinen välittömästi tapahtuva muutos. [1]

2.2.1 Diskreetti tapahtumasimulointi

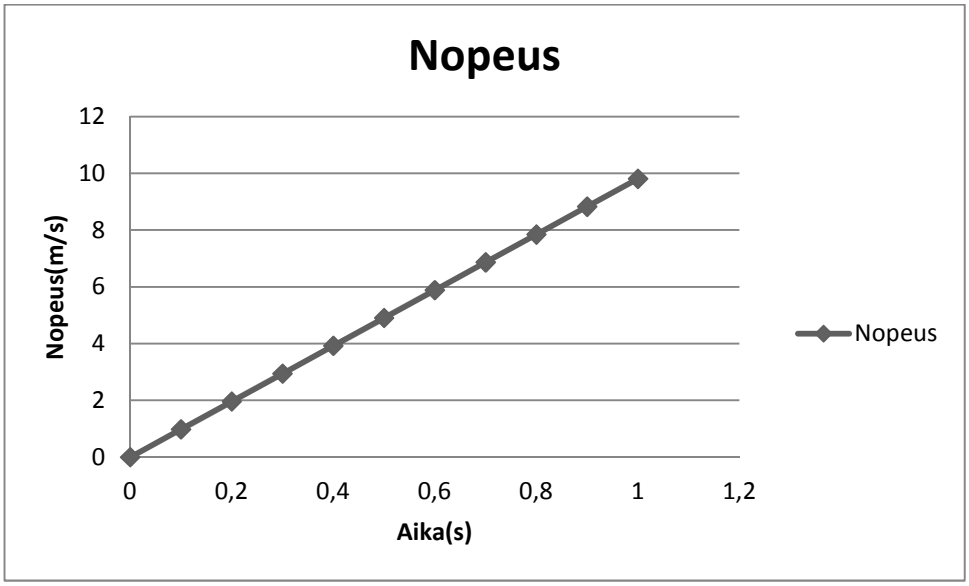
Diskreetti tapahtumasimulointi määritellään variaatioina mallissa, jonka aiheuttaa mallissa aikajärjestyksessä esiintyvät tapahtumat. Tapahtumat ovat luonteeltaan äkillisiä ja ne voivat aiheuttaa variaatioita tai muutoksia järjestelmän tiloissa. Järjestelmän tila määritellään yhtenä tai useampana muuttujana, jotka kuvaavat järjestelmää jokaisena ajan hetkenä. Näitä muuttujia kutsutaan tilamuuttujiksi. [1]

Luvussa 2.2 esillä ollut liikennevalomalli toimii esimerkkinä diskreetti tapahtumapohjaisesta simulaatiosta. Liikennevalojen tila voidaan esittää jokaisena ajan hetkenä aktiivisena olevana valona. Tilamuuttuja on silloin valon väri, joka esittää tilamuuttujan tilaa. Sen arvo kuvaa jokaisena ajan hetkenä täysin liikennevalojen tilan. Tapahtumat liikennevaloissa ovat valon värin muutokset punaiseen, keltaiseen ja vihreään. Nämä tapahtumat voivat esiintyä halutussa järjestyksessä ja muutoksen voi laukaista tietyn aikamääreen kuluminen tai esimerkiksi anturi, joka tunnistaa ajoneuvon saapumisen liikennevaloihin. [1]

Toinen osa diskreettiä tapahtumasimuloitua järjestelmää on kello. Kello pitää kirjaa simulointiajasta, ja se voi toimia tapahtuman laukaisevana tekijänä järjestelmässä, kuten liikennevalomallissa. Simulointiaika voi olla sama asia kuin seinäkelloaika tai reaaliaika. Simulaation käydessä seinäkelloaikaa, aika kuluu niin kuin ihmiset normaalisti ymmärtävät ajan kulumisen. Toinen vaihtoehto on se, että aika kuluu nopeammin tai hitaammin verrattuna seinäkelloaikaan. Näistä tärkeämpi ominaisuus simuloinnissa on ajan kulun nopeuttamisen mahdollisuus, jolloin järjestelmän analysointi, joka normaalisti vie reaaliajassa tunteja, viikkoja, kuukausia tai vuosia, voidaan toteuttaa supistetussa ajassa. [1]

2.2.2 Jatkuva simulointi

Edellisessä luvussa esitetty diskreetti tapahtumasimulointi perustuu aikajärjestyksessä esiintyviin tapahtumiin, joka muuttaa mallin tilaa aina, kun tapahtuma esiintyy. Jatkuva simulointi määritellään taas tilamuuttujilla, jotka muuttuvat jatkuvasti ajan suhteen. Tilamuuttujat ovat järjestelmän parametreja, jotka kuvaavat järjestelmän käyttäytymistä. Esimerkkinä toimii pallon liike, kun se pudotetaan usean sadan metrin korkeudesta ja siihen vaikuttaa vain maan vetovoima. Yksi tilamuuttujista, joka kuvaa pallon liikettä, on nopeus. Pallon nopeus ennen pudotusta on nolla. Kun pallo pudotetaan, siihen alkaa vaikuttaa maan vetovoima, jonka seurauksena sen nopeus alkaa kiihtyä. Pallon nopeus kasvaa tasaisesti ilman katkoja ajan funktiona, joka on kuvattu kuvassa 2. [1]



Kuva 2. Pudotetun pallon nopeus ajan funktiona. [1]

2.3 Tulosten vahvistaminen, todentaminen ja hyväksyminen

Edellisessä luvussa kuvattu pallon pudotus on hyvin yksinkertaistettu esimerkki jatkuvasta simuloinnista, joka antaisi käsityksen pallon äärettömästä nopeuden kiihtymisestä. Näin ei kuitenkaan ole todellisuudessa ja siksi hyvin oleellinen osa mallintamista ja simulointia on niiden vahvistaminen, todentaminen sekä hyväksyminen. Nämä kolme asiaa ovat perusedellytys luotettavalle ja uskottavalle mallintamiselle sekä simuloinnille. [1]

Vahvistaminen käsitetään laadunhallinnassa testausprosessina, joka määrittää sen, että vastaako tuote sen määrittelyitä tai onko tuote yhteensopiva siihen sovellettavien määräysten kanssa. Myös mallintamisessa ja simuloinnissa vahvistaminen määritellään prosessina, joka arvioi, onko malli määrittelyiden mukainen. Vahvistaminen vastaa tyypillisesti seuraaviin kysymyksiin [1, 5]:

- Vastaako toteutettu malli konseptuaalista mallia?
- Tyydyttääkö konseptuaalinen malli aiotun käytön tarpeen?
- Tuottaako toteutettu malli tulokset oikeassa muodossa ja silloin, kun niitä tarvitaan?

Laadunhallinnassa todentaminen käsitetään testausprosessina, joka määrittää sen, että täyttääkö tuote suunnitellun asiakkaan tai käyttäjän vaatimukset. Mallintamisessa ja simuloinnissa todentaminen on prosessi, joka määrittelee sen, kuinka tarkasti malli vastaa todellista kohdetta, prosessia tai ilmiötä. Todentamisen aikana tyypillisesti haetaan vastausta seuraaviin kysymyksiin [1, 5]:

- Vastaako konseptuaalinen malli jäljiteltävää kohdetta?
- Kuinka lähellä mallin tulokset ovat jäljiteltävän kohteen käyttäytymistä?
- Kuinka paljon ja kuinka tarkasti muuttujia tulee huomioida, jotta tulokset ovat luotettavia ja käyttökelpoisia?

Hyväksymistä pidetään usein samana asiana kuin vahvistaminen ja todentaminen, vaikka se on hyvin erityyppinen prosessi. Vahvistaminen sekä todentaminen ovat pohjimmiltaan testausprosesseja ja ovat luonteeltaan teknisiä. Hyväksyminen on taas päätösprosessi, joka ei ole luonteeltaan tekninen, mutta voidaan antaa teknisenä tietona. Hyväksynnän mallin käyttämiselle antaa vastuullinen taho. Hyväksyntä tulisi antaa aina vain tiettyyn tarkoitukseen, koska ”mihin käyttöön tahansa” -hyväksyttyjä voidaan käyttää tarkoituksiin, joihin niitä ei ole todennettu. Hyväksynnän tekevä taho päättää hyväksymisestä vahvistamisessa ja todentamisessa esille tulleiden asioiden pohjalta. Hyväksymisprosessin aikana vastataan tyypillisesti seuraaviin kysymyksiin [1, 5]:

- Ovatko mallin kyvyt ja suunnitellun käyttösovelluksen vaatimukset yhtenevät?
- Näyttääkö vahvistamisesta ja todentamisesta saadut tulokset sen, että malli kykenee tuottamaan hyödyllisen tarkkoja tuloksia, kun sitä käytetään suunniteltuun käyttösovellukseen?
- Mitkä ovat seuraukset, jos käytetään epätarkkaa mallia suunniteltuun käyttösovellukseen?

Vahvistaminen, todentaminen ja hyväksyminen ovat hyvin lähellä toisiaan ja siksi niiden erot ovat usein esitetty kysymyksillä; vahvistaminen vastaa kysymykseen ”Tehtiinkö malli oikein?”, todentaminen vastaa kysymykseen ”Tehtiinkö oikea malli?” ja hyväksyminen vastaa kysymykseen ”Onko malli oikea käyttösovellukseen?”. [1, 5]

3 METSÄKONEEN HAKKUUPÄÄ

Metsäkoneiden kehitys voidaan katsoa alkaneeksi 1800-luvun lopussa, jolloin Pohjois-Amerikassa rakennettiin tiettävästi ensimmäinen tukkien kuljetukseen tarkoitettu höyryveturi. Pohjoismaihin nämä metsäkoneet rantautuivat vasta 1910-luvulla. Tästä kehitys jatkui nopeasti siten, että Euroopan ensimmäisen kaatokouran prototyypin rakensi 1960-luvulla Garpenbergin metsäyliopisto. Vuonna 1972 Sakari Pinomäki sai valmiiksi ensimmäisen pohjoismaiden harvesterin ja pian tämän jälkeen ensimmäiset reletekniikkaan perustuvat mittausautomaatit ilmestyivät. Ensimmäinen mikroprosessoritekniikkaan perustuva mittausautomaatti kehitettiin jo 1970-luvun lopussa, mutta ne löivät itsensä läpi vasta 10 vuotta myöhemmin. Vasta 1990-luvun alussa Suomi luopui käsin mittauksesta ja saman vuosikymmenen aikana nähtiin jo kävelevä metsäkone ja ensimmäinen virtuaalisimulaattori. [6]

Puun kaataminen voidaan tehdä nykyään kolmella eri tavalla: manuaalisesti käsin esimerkiksi kirveellä tai sahalla, motorisoidusti käsin kuten moottorisahalla tai koneellisesti esimerkiksi harvesterilla. Nykyään koneellinen kaataminen on näistä suosituin sen turvallisuuden, nopeuden ja tehokkuuden vuoksi. Suomessa tehdään noin 95 % hakkuista hakkuukoneilla. Koneet vaihtelevat pienestä itse modifioidusta traktorista suuriin harvestereihin. Toiset koneet kykenevät vain puun kaatamiseen ja toiset koneet voivat esimerkiksi kaatamisen lisäksi karsia, mitata ja katkaista puun haluttuihin pituuksiin. Itse kone voi liikkua pyörillä tai teloilla ja se on varustettu puomilla. Puomin päässä on hakkuupää, joka käsittelee puun. [7, 8]

Puunkorjuumenetelmiä on useita, mutta kolme yleisintä niistä ovat puu-, runko- ja tavaralajimenetelmä. Puumenetelmässä puu kaadetaan ja kuljetetaan sellaisenaan tienvarteen odottamaan jatkokuljetusta. Runkomenetelmän ero puumenetelmään on vain puun karsinta, joka tehdään kaatamisen jälkeen. Tavaralajimenetelmässä puu kaadetaan, minkä jälkeen aloitetaan sen karsinta. Samanaikaisesti mittalaitteet mittaavat rungon pituutta sekä halkaisijaa, joiden tietojen pohjalta runko katkotaan tehtaiden tarpeiden mukaisiin pituuksiin. Näin rungon jalostusarvo optimoidaan jo metsässä ja päästään samalla eroon

ylimääräisistä rungon käsittelykerroista. Tämän vuoksi tavaralajimenetelmä on nykyään kasvavassa roolissa puunkorjuumenetelmissä. [8, 9]

Hakkuupäät alkoivat saada nykypäivän muotonsa 1980-luvun alussa, kun ruotsalainen SP-maskiner kehitti ensimmäisenä maailmassa yksioteharvesterin. Tavaralajimenetelmässä puunkorjuu on useimmin tehty juuri yksiote-harvesterilla. Kuvassa 3 on esitetty Kesla Oyj:n valmistaman rullasyöttöisen RH30-harvesterikouran rakenne esimerkkinä yksiote-harvesterista. [6, 7, 10]

Kuva 3. Yksioteharvesterin rakenne; syöttörullat(1), takakarsintaveitset(2), etukarsintaveitset(3), saha(4), mittalaitteet(5), rotaattori(6), kallistussylinteri(7), puomikaapeli(8) ja venttiililohko(9). [11]

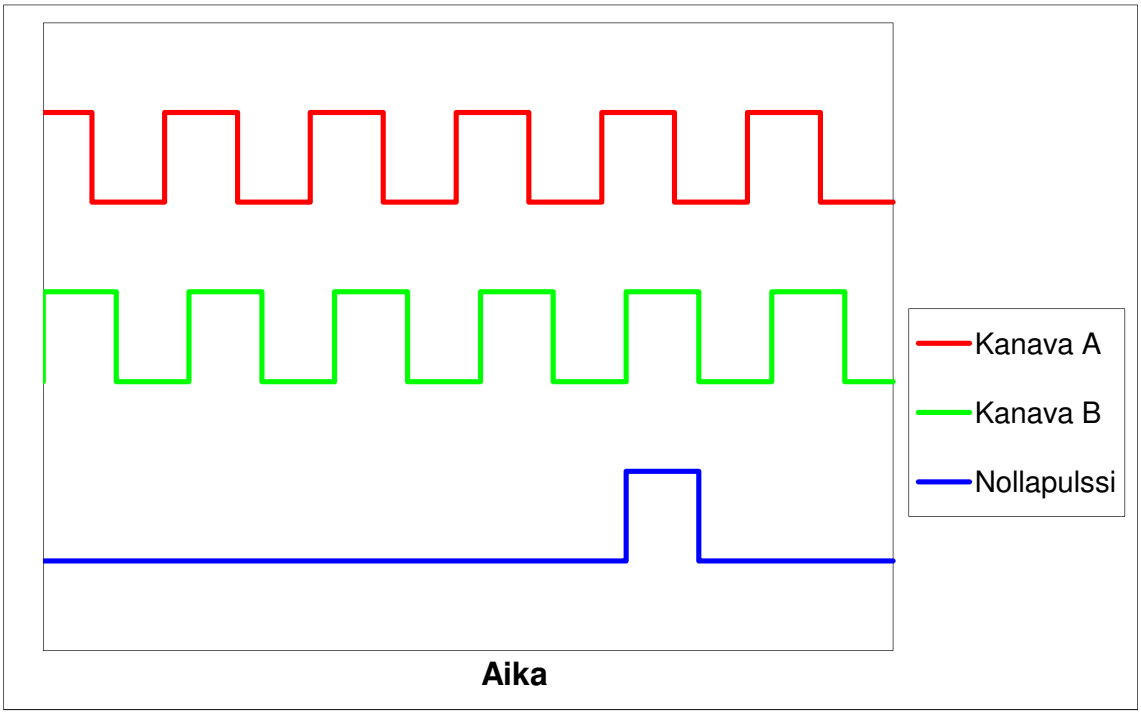
3.1 Rakenne

Hakkuupää on hydraulijärjestelmä, joka toimii hydrostaattisella paineella. Paineen tuottavat hydraulipumput, jotka saavat energiansa dieselmoottorista. Hydrauliöljy johdetaan letkujen avulla hakkuupäähän, jossa on tyypillisesti erilaisia hydraulisylintereitä, -pumppuja, -moottoreita ja -venttiileitä. Hydrauliosien avulla saadaan toteutettua hakkuupään perustoiminnot kuten puun kaato, syöttö kourassa, katkaisu ja karsinta. [7]

Anturin tehtävä on mitata ja muuntaa jokin fysikaalinen suure toiseksi halutuksi suureeksi. Anturin lähdön haluttu suure on tyypillisesti sähköinen signaali, mutta se voi myös olla esimerkiksi paine- tai liikesignaali. Mitattavia suureita hakkuupäässä ovat esimerkiksi puun halkaisija sekä pituus, sahan asento ja venttiilien paine. [12, 13]

Puun pituuden mittaus toteutetaan tyypillisesti hakkuupäissä mittapyörällä, joka koskettaa puun runkoa ja pyörii puun liikkuessa. Mittapyörän akseliin on kiinnitetty pulssianturi, joka tuottaa pulssijonon lähtöihinsä, kun sen akseli pyörii. Pulssianturityyppejä on kuitenkin monia erilaisia. Ne jaetaan toimintaperiaatteensa mukaan kahteen ryhmään: inkrementtianturit ja absoluuttikulma-anturit. Inkrementtianturi laskee muutoksen 0-pisteestä, jolloin se tulee käyttää 0-pisteessä joka kerta, kun järjestelmä käynnistetään. Absoluuttikulma-anturi tietää virrankatkaisusta huolimatta oman asemansa koko ajan, jolloin se ei vaadi 0-pisteeessä käyntiä. Sekä inkrementtianturit että absoluuttikulma-anturit tuottavat nollapulssin myös silloin, kun anturin akseli on kiertänyt täyden kierroksen. Molemmat pulssianturityypit voidaan vielä edelleen jakaa toteutustavan mukaan optisiin, magneettisiin ja kapasitiivisiin pulssiantureihin. [7, 12…14]

Edellä mainittujen kohtien lisäksi pulssiantureita on mahdollista saada yksi- tai kaksikanavaisena. Yksikanavainen pulssianturi tuottaa nimensä mukaisesti yhden pulssijonon, josta saadaan pituustieto pulssien määrästä sekä nopeustieto pulssien taajuudesta. Kaksikanavainen pulssianturi tuottaa kaksi pulssijonoa, joilla on 90°:n vaihe-ero. Tämä mahdollistaa yksinkertaisen logiikkakytkennän avulla pulssianturin pyörimissuunnan selvittämisen, seuraamalla kummalta kanavalta pulssi saapuu ensin. Hakkuupäässä käytetään tyypillisesti juuri kaksikanavaisia optisia pulssiantureita, koska puun liikesuunta voi olla eteenpäin tai taaksepäin. Kuvassa 4 on esitetty kaksikanavaisen pulssianturin tuottamat pulssijonot sekä nollapulssi, joka on tahdistettu kanavalle B. [7, 12…14]



Kuva 4. Pulssianturin tuottamat pulssijonot ja nollapulssi. [18]

Puun halkaisijan mittaus on tyypillisesti toteutettu hakkuupäässä potentiometrillä, joka on sijoitettu karsintaveitsiin tai syöttörulliin. Resistanssi muuttuu, kun hakkuupäätä avataan tai suljetaan. Potentiometrin yli olevaa jännitettä mitataan ohjausmoduulin analogiatulolla, josta saadaan laskettua halkaisija. Tämä toteutustapa vaatii potentiometrin säännöllistä kalibrointia, jotta tarkkuus säilyisi riittävällä tasolla. Halkaisijan mittaus voidaan myös toteuttaa vastaavilla pulssiantureilla kuten pituuden mittaus. Tällöin yksi pulssi vastaa esimerkiksi hakkuupään 1 mm:n avautumaa, jolloin hakkuupään maksimiavautuman ollessa 600 mm syntyy koko liikeradan aikana 600 pulssia. Koska halkaisijaa mittaava pulssianturi on tyypillisesti inkrementtianturi, se nollataan joka käynnistyksen yhteydessä hakkuunpään ollessa täysin avoinna. [7]

Sahan toimintaa seurataan tyypillisesti kahdella anturilla: induktiivianturilla ja pulssianturilla. Induktiivianturilla toteutetaan saha kotona -toiminto, jolla varmistetaan sahan alkuasennossa oleminen. Tämä suojaa sahaa vaurioitumiselta estämällä puun syötön eteen ja taakse sahausliikkeen aikana. Induktiiviantureita käytetään hakkuupään valmistajasta riippuen NPN- tai PNP-

tyyppisiä. Pulssianturilla toteutetaan sahan asento -tieto, jolla voidaan seurata sahausliikettä. Sen toimintatapa on samankaltainen kuin pulssiantureilla toteutetun halkaisijan mittaus. Anturi tuottaa kiinteän pulssimäärän liikeradan aikana ja sahan liikeradan pituus on vähintään yhtä paljon kuin hakkuupään maksimiavautuma. Sahausliikettä jatketaan tyypillisesti vain siihen asti kuin puun halkaisijan suhteen on tarvetta. Näin säästetään sahaa fyysisesti, mahdollisesti käytettävää kantokäsittelyn torjunta-ainetta sekä nopeutetaan hakkuupään toimintaa. [7]

3.2 Liikkeet ja toiminnot

Kun puun käsittely aloitetaan, ensimmäinen vaihe on puun lähestyminen. Metsäkoneen puomin ja hakkuupään välissä on rotaattori, joka mahdollistaa hakkuupään pyörimisen ja tekee puusta kiinniottamisesta helppoa. Rotaattori käännetään haluttuun asentoon ja samaan aikaan kallistussylinteri on ylös nostettuna. Kallistussylinteri mahdollistaa hakkuupään asennon muuttamisen vaaka- ja pystyasennon välillä. Tällä tavoin hakkuupäällä on mahdollista käsitellä niin pystyssä olevia puita kuin kaatuneita tai kaadettuja. Puuta lähestyttäessä tulee edellä mainittujen lisäksi olla syöttörullat sekä karsintaveitset avattuina. [7]

Seuraavassa vaiheessa puuhun tartutaan sulkemalla syöttörullat ja karsintaveitset. Puun kaato aloitetaan sahaamalla puu poikki, minkä jälkeen kallistussylinteri käännetään automaattisesti ala-asentoon. Mikäli hakkuupäässä on kantokäsittelyn mahdollisuus, jossa kantoon suihkutetaan torjunta-ainetta juurikäävän torjumiseksi, torjunta-aine levittyy kantoon sahan kautta sahauksen aikana. Puun kaadon jälkeen voidaan aloittaa sen käsittely. [7]

Syöttörullat liikuttavat puuta eteenpäin ja taaksepäin. Samanaikaisesti mittalaitteet mittaavat pituutta ja halkaisijaa sekä karsintaveitset karsivat puun. Puu katkaistaan sahalla haluttuihin pituuksiin anturien antamien mittatietojen perusteella. Sahatut tukit merkitään tyypillisesti värimerkein kokonsa mukaan. Hakkuupään alaosaan on sijoitettu suuttimet, jotka maalaavat raidan tukin päähän sen tippuessa sahauksen jälkeen. Värejä on yleensä vähintään kahta

erilaista, jotta saadaan merkittyä useampaa eri kokoa eri väriyhdistelmin. Kun koko puu on käsitelty, avataan syöttörullat sekä karsintaveitset, nostetaan kalistussylinteri takaisin ylös ja voidaan aloittaa seuraavan puun käsittely. [7]

3.3 Ohjaus- ja mittausjärjestelmä

Ohjaus- ja mittausjärjestelmät ovat nykyaikaisen metsän harvennuksen tärkeimpiä työkaluja. Nämä kaksi järjestelmää ovat yleensä yhdistetty yhdeksi järjestelmäksi, johon kuuluu vähintään pääyksikkö ja hakkuupään ohjausmoduuli. Kuvassa 5 on esitetty Technion Oy:n kehittämä ohjaus- ja mittausjärjestelmä Logger, joka sisältää pääyksikön LDC (Logger Display Controller), hakkuupään ohjausmoduulin LHC (Logger Head Controller) sekä kytkentämoduulin LHI (Logger Harvester Interface). Näistä kytkentämoduuli on Technion Oy:n innovaatio, joka mahdollistaa järjestelmän helpon toteuttamisen kaivinkonekäyttöön. [7, 15]

Kuva 5. Esimerkki metsäkoneen ohjaus- ja mittausjärjestelmästä. [15]

Järjestelmän pääyksikkö ohjaa hakkuupään liikkeitä, mutta laskee myös mittalaitteiden mittaamien halkaisijan ja pituuden avulla sahatun tavaran tilavuuden sekä tallentaa tarvittavat tiedot. Tietojen tallentaminen on tärkeää seuraavista syistä [7, 15]:

- Metsän omistajille ja harvennuksen tekevälle urakoitsijalle voidaan maksaa tallennettujen tietojen perusteella.
- Metsäyhtiöt voivat saada reaaliaikaisen tiedon heidän metsien hakkuutilanteesta ja suunnitella tulevat toiminnot tarkemmin.
- Ohjausjärjestelmä voi ennustaa puun muodon aikaisempien puiden tietojen perusteella. Tämä yhdistettynä tarkkaan mittaukseen, mahdollistaa rungon korkean hyödyntämisasteen ja sitä kautta tuottaa tehtaille mitoiltaan optimaalisen raakamateriaalin.
- Mittatiedot mahdollistavat myös koneen kuljettajan tuottavuuden seurannan.

Nykypäivän metsäkoneessa on erillinen hakkuupään ohjausmoduuli, joka kommunikoi pääyksikön kanssa CAN-väylän (Controller Area Network) kautta. Ennen CAN-väylän käyttöä, jokaisen hakkuupään venttiilin ohjauslinjat tuotiin puomia pitkin hakkuupäähän. Hakkuupäässä ei ollut erillistä hakkuupään ohjausmoduulia ja tämän vuoksi puomissa kulki suuri määrä johtoja, jotka olivat poikki mennessään kalliitta ja työläitä vaihtaa. Käyttämällä erillistä ohjausmoduulia ja CAN-väylää, koneen ja hakkuupään välillä kulkee vain neljä johtoa: käyttöjännite, maa sekä CAN-väylän high- ja low-johto. [7, 15]

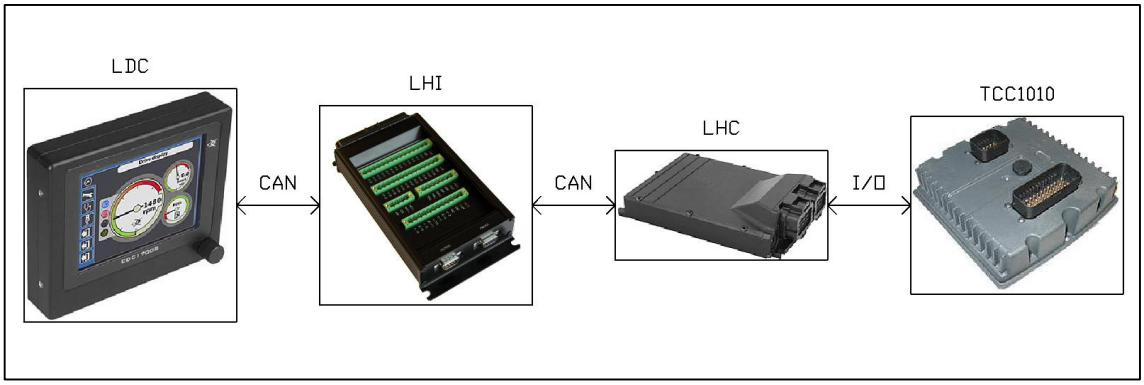
Hakkuupään ohjausmoduulin tehtävä on analysoida pääyksiköltä saapuvat CAN-viestit ja antaa niiden perusteella tarvittavat ohjaukset hakkuupään venttiileille. Ohjausmoduuli myös kerää ja välittää antureilta saatavat tiedot CAN-väylää pitkin pääyksikölle sekä tarvittaessa suorittaa välilaskentaa ja signaalien muunnosta ennen viestien välitystä. [7, 15]

4 SIMULAATTORIN SPESIFIKAATIOT

Simulaattori toteutetaan Technion Oy:n kehittämälle metsäkoneen hakkuupään Logger-mittalaitteelle. Simulaattoria käytetään pääasiassa tuotekehityksessä sovelluskehityksen testilaitteena. Toisena käyttökohteena on Logger-järjestelmän esittely niin messuilla kuin asiakastapaamisissa. [16]

4.1 Mekaaniset ja sähköiset spesifikaatiot

Simulaattori koostuu kuvan 6 mukaisesti neljästä moduulista. LDC, LHI ja LHC ovat Logger-järjestelmän moduuleita, joiden toimintaa simuloidaan. Technion Oy:n kehittämä TCC1010 (Technion Can Controller) PLC-moduuli (Programmable Logic Controller) mallintaa ja simuloi hakkuupään toimintaa tuottamalla järjestelmälle luvussa 4.2 määriteltävät anturisignaalit. [16]



Kuva 6. Simulaattorin moduulit. [16]

Metsäkoneiden tyypillinen käyttöjännite on 24 V, joka tulee olla myös simulaattorin antama käyttöjännite Logger-järjestelmälle. Simulaattorin tulee toimia verkkosähköllä ja se tulee olla kytkettävissä verkkosähköön mahdollisimman standardilla ja helposti saatavalla johdolla. [16]

Ohjaus- ja mittausjärjestelmän moduulien välinen kommunikointi tapahtuu CAN-väylän kautta. LHC-moduulin sekä TCC1010-moduulin välinen kommunikointi tapahtuu suoraan I/O-rajapinnan (Input/Output) kautta. Jokaista Logger-järjestelmän lähtöä kuvataan omalla LEDillä, joka palaa lähdön ollessa

aktiivinen. Jokaista järjestelmän tuloa kohden on yksi kytkin, jota painamalla voidaan aktivoida tulo. Tuloja on sekä PNP-tyyppisiä että NPN-tyyppisiä. [16]

Simulaattorin mekaaninen toteutus tulee olla mahdollisimman kompaktin kokoinen, jotta kuljettaminen olisi helppoa. Jokainen tulo ja lähtö tulee olla merkittynä kyseessä olevan toiminnon kuvauksella, kuten esimerkiksi syöttö eteen, syöttö taakse, etuveitset auki jne. Toimintojen kuvaukset eivät saa olla kiinteitä, jotta niiden vaihtaminen paikasta toiseen olisi helppoa. [16]

4.2 Sovelluksen spesifikaatiot

Simulaattorin sovellus toteutetaan TCC1010 PLC-laitteelle CoDeSys-sovelluskehitystyökalulla (Controller Development System). Sovelluksen tulee simuloida seuraavien antureiden toimintaa [16]:

- puun pituuden mittaus (pulssianturi)
- puun halkaisijan mittaus (pulssianturi)
- sahan asento (pulssianturi)
- saha-kotona (kaksitila-anturi)
- latvasaha-kotona (kaksitila-anturi).

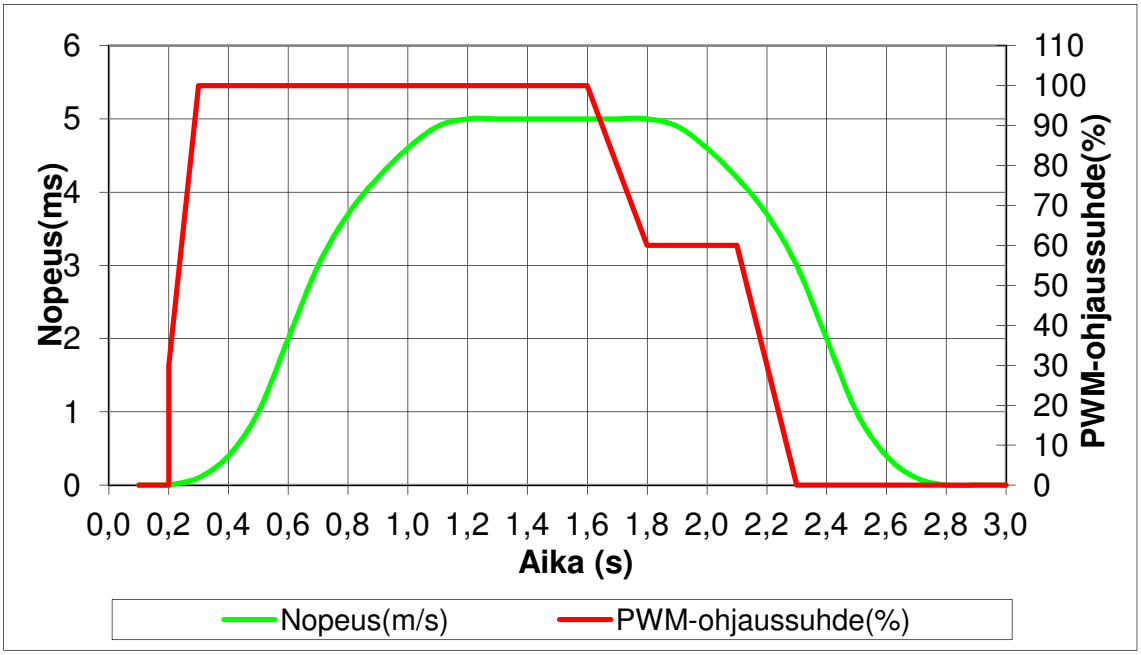
Sovellukseen määritellään ennalta viisi eri runkoprofiilia. Runkoprofiileilla simuloidaan järjestelmälle eripaksuisia ja -pituisia puita. Runkoprofiili tulee olla valittavissa erillisellä kytkimellä. Sovelluksen tulee tunnistaa uuden rungon käsittelyn aloitus seuraavien liikkeiden sarjasta [16]:

1. etu- ja takakarsimaveitset auki
2. syöttörullat auki
3. kallistussylinteri ylös.

Kun seuraavassa vaiheessa, edellä mainittujen liikkeiden jälkeen, etu- ja takakarsimaveitset sekä syöttörullat suljetaan, ohjaus- ja mittausjärjestelmän tulee nähdä valittuna olevan rungon halkaisija hakkuupäässä. Tätä seuraa puun käsittely, mikä päättyy puun pituuden loppuessa tai aukaistaessa etu- ja takakarsimaveitset sekä syöttörullat. [16]

4.2.1 Pituus

Hakkuupään puun syöttönopeutta ohjataan PWM-signaalin pulssi suhteella siten, että 100 %:n ohjaussuhteella puun maksimi syöttönopeus on 5 m/s. Kuvassa 7 on esitetty esimerkki PWM-ohjauksen vaikutuksesta puun syöttönopeuteen. [16]



Kuva 7. PWM-ohjauksen vaikutus puun syötön nopeuteen. [16]

Puun liikettä ja pituuden mittausta kuvataan LHC-moduulille kaksikanavaisena pulssianturina, joka tuottaa pulssin 5 mm välein. Pituuspulssien maksimi tulotaajuus on 1 kHz. [16]

4.2.2 Halkaisija

Puun halkaisijaa kuvataan LHC-moduulille kaksikanavaisena pulssianturina, joka tuottaa pulssin 1 mm:n välein hakkuupäätä avattaessa. Puun syötön ollessa käynnissä tulee halkaisijan mittatieto saapua LHC-moduulille 100 mm:n välein. Puun maksimihalkaisija on 600 mm ja halkaisijan pulssitieto on nollattava jokaisen käynnistykseen yhteydessä hakkuupään ollessa täysin auki. [16]

4.2.3 Saha

Sahan asentotieto kuvataan kaksikanavaisena pulssianturina, joka tuottaa 100 pulssia sahan koko liikeradan aikana. Sahausliike kestää alkuasennosta ääriasentoon 2 s. Paluuliike kestää ääriasennosta alkuasentoon 1 s:n. Pulssitieto on nollattava jokaisen käynnistyksen yhteydessä sahan ollessa alkuasennossa. [16]

Saha kotona -toiminto sekä latvasaha kotona -toiminto kuvataan kaksitilatoimintoina, jotka tulee olla valittavissa PNP- tai NPN-tyyppiseksi. Sahan ollessa alkuasennossa, tulee toiminnon olla aktiivisena. [16]

5 HAKKUUPÄÄN SIMULAATTORI

Hakkuupäiden valmistajia on useita ja jokaiselta valmistajalta on vielä saatavilla useampi eri malli hakkuupäitä. Tämä simulaattori on toteutettu Technion Oy:n yhden asiakkaan ja yhden kouratyypin ohjaus- ja mittausjärjestelmän sovelluksen pohjalta. Tämä on määrännyt esimerkiksi I/O-rajapinnan kytkennät, tulojen ja lähtöjen määrät sekä pulssianturien käyttökohdat.

Simulaattoria on kuitenkin tarkoitus käyttää kaikkien asiakkaiden sovelluksien testaamiseen. Siksi simulaattorin mekaanisessa ja sähköisessä toteutuksessa on pyritty mahdollisimman yksinkertaiseen ja yleiskäyttöiseen toteutukseen, jotta sen muunneltavuus olisi helppoa. Käytetyt komponentit ovat valittu mahdollisimman standardin mukaisiksi, jotta niitä saa helposti hankittua varaosiksi ja vastaavia korvaavia komponentteja löytyisi paljon. Muunneltavuus on pyritty ottamaan huomioon myös sovelluksen toteutuksessa mahdollisuuksien mukaan.

5.1 Mekaaninen toteutus

Simulaattorin mekaniikka on toteutettu sisämitoiltaan 559 mm pitkään, 432 mm leveään ja 203 mm korkeaan muovisalkkuun. Muovisalkku täyttää MIL-STD-810F-, MIL-STD-C-4150J- ja MIL-STD-648C -standardit, jotka määrittelevät salkun kestävän UV-säteilyä, liuottimia, korroosiota sekä sienikasvustoa. Lisäksi standardit määrittelevät salkun tiiveyden IP67-luokaksi, jolloin salkku on täysin pölytiivis sekä kestää 1 m:iin asti upottamisen veteen. Salkku antaa näin riittävän suojan simulaattorille kuljetusta sekä säilytystä varten.

Simulaattorin rakenne on jaettu kolmeen kerrokseen liitteen 1 mukaisesti. Salkun pohjan alempaan kerrokseen on sijoitettu kaikki ne komponentit, joiden ei ole tarkoitus näkyä esiteltäessä ohjaus- ja mittausjärjestelmää. Tällaisia komponentteja ovat virtalähde, DIN-kisko, riviliittimet, simuloiva moduuli sekä kaikki johdot moduulien, kytkimien ja LEDien välillä. DIN-kiskon käytöllä on pyritty minimoimaan ruuvikiinnitysten määrä, jotta komponenttien vaihtaminen ja lisääminen olisi mahdollisimman helppoa.

Pohjan alemman kerroksen päälle on rakennettu liitteen 2 mukainen metallilevy, josta muodostuu pohjan ylempi kerros. Levyyn on kiinnitetty LEDit, kytkimet sekä LHC- ja LHI-moduuli. Hakkuupään moduulista lähtevät johdot kulkevat läpiviennin kautta alempaan kerrokseen, jossa on toteutettu varsinaiset kytkennät. LHI-moduulista lähtevät johdot kulkevat sekä näytölle että alempaan kerrokseen. Tulojen ja erojen

Kanteen on kiinnitetty ainoastaan LDC-moduuli, josta johdot kulkevat LHI-moduulille. Kanteen on kiinnitetty mahdollisimman vähän komponentteja, jotta sen paino jäisi alhaiseksi. Tämä estää salkun kaatumisen, kun se on ollessa avattuna.

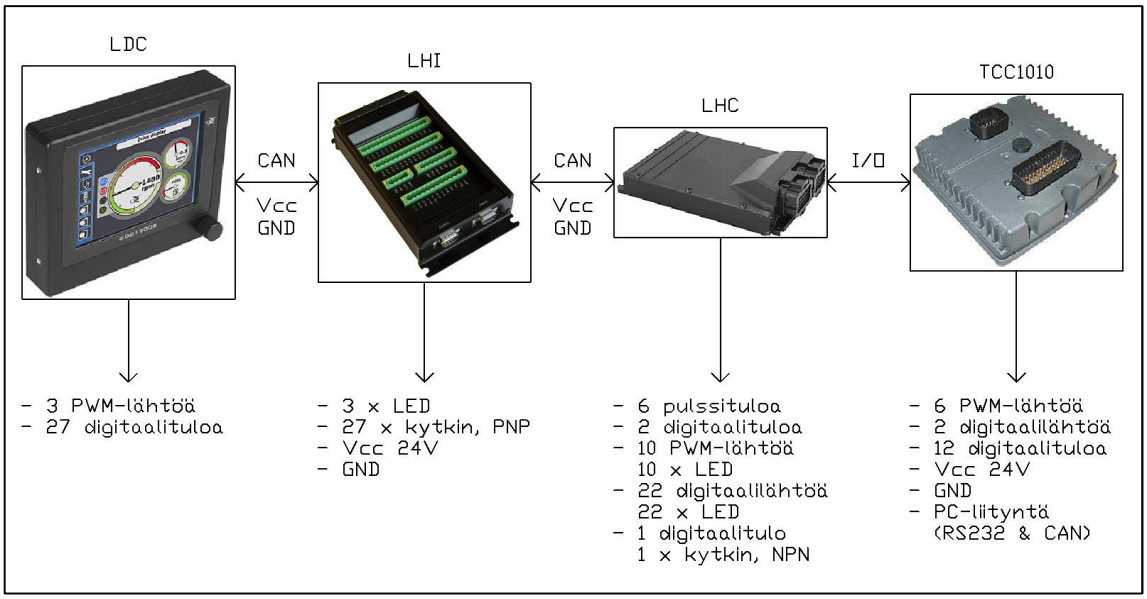
5.2 Sähköinen toteutus

Käyttöjännite on toteutettu Lambda DPP50-24 -teholähteellä, jonka maksimiteho on 50 W ja lähtöjännite 24 V. Teholähde saa virran verkkosähköstä, johon se on kytketty IEC 60320 -standardin C13-liittimen kautta. Liittimessä on 10 A:n sulake oikosulun varalta sekä kytkin, joka toimii koko simulaattorin päävirtakytkimenä.

Moduulien välisten suorien johdotusten sijaan, salkun pohjan alempaan kerrokseen on rakennettu DIN-kisko, johon on kiinnitetty 70 kpl:ta riviliittimiä. Riviliittimien tarkoituksena on toimia pisteenä, jossa kytkennät haarautuvat kytkimille, LEDeille ja moduuleille. Esimerkiksi teholähteeltä lähtevät käyttöjännite ja maapotentiaali on kytketty riviliittimiin, josta ne ovat jaettu simulaattorin jokaiselle moduulille. Koska riviliittimet ovat ruuviliittimillä toimivia, kytkentöjen muuttaminen on nopeaa, eikä vaadi erityistyökaluja. Esimerkkejä erityistyökaluista ovat moduulien liittimien kontaktien vaihtamiseen tarvittavat kontaktipuristimet tai viallisten kytkimien ja LEDien johtojen juottamiseen tarvittava juotin.

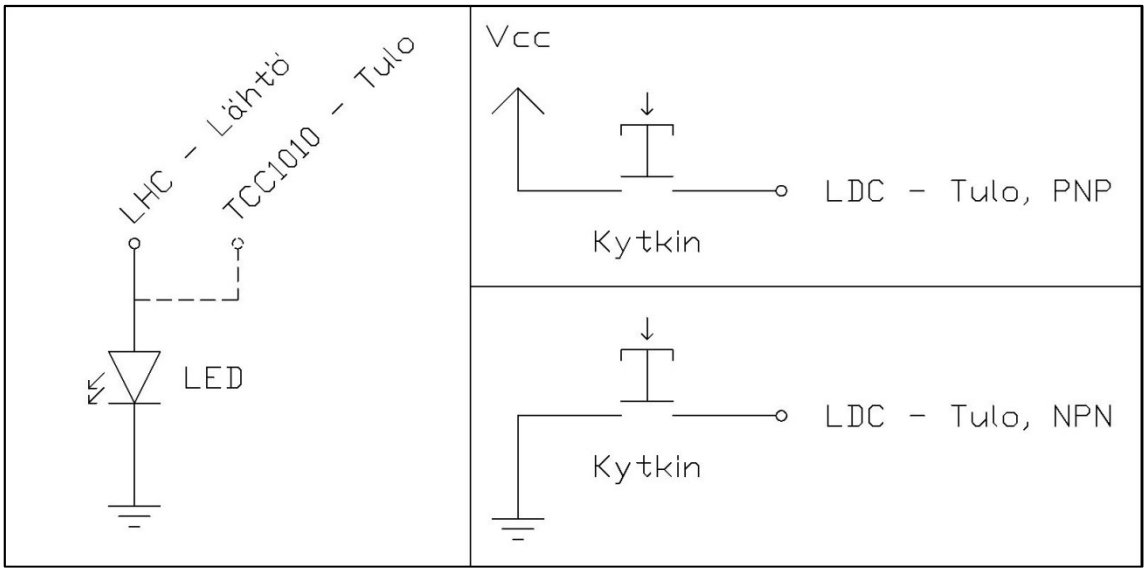
Kyseessä olevan sovelluksen tulojen ja lähtöjen määrä on esitetty kuvassa 8, mutta varsinaisessa toteutuksessa kytkimien kokonaismäärä on 30 kpl:ta ja

LEDien 40 kpl:ta. Ylimääräiset kytkimet ja LEDit ovat varattu muiden hakkuupään valmistajien I/O-määrien vaihteluiden vuoksi.



Kuva 8. Simulaattorin sähköinen toteutus.

Lähtöjen aktivoitumista kuvaaviksi LEDeiksi on valittu valmiiksi johdot ja etuvastuksen sisältävät LEDit, jotka ovat tarkoitettu 24 V:n käyttöjännitteelle. Tämä ratkaisu vähentää kytkentöjen tekemistä sekä mahdollisen viallisen LEDin vaihtaminen tulee mahdollisimman helpoksi. Jokainen ohjaus- ja mittausjärjestelmän lähtö on kytketty kuvan 9 mukaisesti LEDin kautta maapotentiaaliin. Poikkeuksena ovat lähdöt, jotka ovat kytketty myös TCC1010-moduulin tuloihin, jotta simuloiva sovellus saa tarvitsemansa tilatiedon näistä lähdöistä. Kyseessä olevat lähdöt ovat tarkemmin kuvattu luvussa 5.3.



Kuva 9. Lähtöjen ja tulojen kytkennät.

Tulot aktivoivat kytkimet toimivat vain painettaessa, jolloin kytkin kytkee kuvan 9 mukaisesti tulon joko käyttöjännitteeseen tai maapotentiaaliin riippuen tulon tyypistä.

5.3 Sovellus

Simulaattorin sovelluksen alustana toimii TCC1010-moduuli. Moduulissa on 20 I/O-pistettä ja jokaisella I/O-pisteellä on oma toiminto. Toiminnot ovat esitetty taulukossa 1.

Taulukko 1. TCC1010-moduulin I/O-rajapinta.

TCC1010			
Pinni	**Funktio**	**Johdotus - Kuvaus**	
X2-24	DO	LHC X1-1	LATVASAHA KOTONA (PNP tai NPN)
X2-1	DO	LHC X1-8	SAHA KOTONA (PNP tai NPN)
X2-2	DI	LHC X2-33	ETUVEITSET AUKI
X2-3	DI	LHC X2-32	ETUVEITSET KIINNI
X2-25	DI	LHC X2-31	TAKAVEITSET AUKI
X2-26	DI	LHC X2-30	TAKAVEITSET KIINNI
X2-4	PWM	LHC X1-7	PITUUS KANAVA A
X2-5	PWM	LHC X1-6	PITUUS KANAVA B
X2-6	PWM	LHC X1-5	SAHAN ASENTO KANAVA A
X2-7	PWM	LHC X1-4	SAHAN ASENTO KANAVA B
X2-8	PWM	LHC X1-3	HALKAISIJA KANAVA B
X2-9	PWM	LHC X1-2	HALKAISIJA KANAVA A
X2-10	DI	LHC X2-27	SYÖTTÖ ETEEN
X2-11	DI	LHC X2-26	SYÖTTÖ TAAKSE
X2-12	DI	LHC X2-25	SYÖTTÖRULLAT KIINNI
X2-33	DI	LHC X2-24	SYÖTTÖRULLAT AUKI
X2-34	DI	LHC X2-23	KALLISTUSSYLINTERI ALAS
X2-35	DI	LHC X2-22	KALLISTUSSYLINTERI YLÖS
X1-11	DI	LHC X2-3	SAHA ULOS
X1-13	DI	LHC X2-17	LATVASAHA ULOS
X1-1	VCC	TEHOLÄHDE 24V	
X1-2	CAN1 HI	LHC X1-26	
X1-3	CAN1 LO	LHC X1-19	
X1-4	RS232 TX	CoDeSys-LIITYNTÄ (PC)	
X1-5	GND	TEHOLÄHDE GND	
X1-8	RS232 RX	CoDeSys-LIITYNTÄ (PC)	

Simulaattorin sovellus jakautuu 5 osaan: pääohjelmaan `PLC_PRG` ja funktioihin `LENGTH`, `DIAMETER`, `SAWS_STATES` sekä `HEAD_STATE`. Näiden lisäksi on määritelty 5 tietuetta: `IO_CONFIGURATION`, `LENGTH_PARAMETERS`, `DIAMETER_PARAMETERS`, `SAWS_PARAMETERS` ja `LOGS_PROFILES`.

`IO_CONFIGURATION`-tietueessa määritellään hakkuupään toimintojen I/O-pisteet, joiden pohjalta sovellus tietää, mikä toiminto on missäkin I/O-pisteessä. `LOGS_PROFILES`-tietueessa määritellään runkoprofiilit, joita sovellus simuloi hakkuupään moduulille. Muissa tietueissa määritellään jokaisen funktion

parametrit kootusti, kuten esimerkiksi DIAMETER_PARAMETERS-tietueessa on määritetty kaikki seuraavat halkaisijan simulointiin liittyvät asiat:

- halkaisijapulssilähtöjen kanava
- yhden pulssin vastaavuus millimetreinä.
- mittauksen tiheys millimetreinä
- hakkuupään maksimiavautuma.

Tietueiden käyttö helpottaa ohjelmakoodin myöhempää muuttamista, koska kaikki simulointiin liittyvät asetukset on kerätty yhteen. Tietueissa esiintyviä asetuksia käytetään myös useammassa paikassa, jolloin muutos tarvitsee tehdä vain tietueeseen. Kaikki tietueet on esitetty tarkemmin liitteessä 8.

Pääohjelma PLC_PRG on esitetty liitteessä 3. Sen tehtävä on kutsua liitteissä 4 - 7 esitettyjä funktioita hakkuupään tilanteen mukaan. HEAD_STATE-funktion tehtävä on seurata ja päivittää seuraavien hakkuupään toimintojen tilatietoja:

- etuveitset auki ja kiinni
- takaveitset auki ja kiinni
- syöttörullat auki ja kiinni
- kallistussylinteri alas ja ylös.

Funktio myös palauttaa pääohjelmalle tiedon uuden rungon aloituksesta, jolloin pääohjelma alustaa LENGTH- ja DIAMETER-funktiot. Toisena tietona HEAD_STATE-funktio palauttaa pääohjelmalle tiedon, onko hakkuupäässä runko käsiteltävänä. Mikäli hakkuupäässä on runko, pääohjelma kutsuu LENGTH- ja DIAMETER-funktioita. Näiden funktioiden tehtävä on tuottaa pituus- ja halkaisijapulssit hakkuupään anturituloihin.

Pääohjelma seuraa myös SAHA ULOS- sekä LATVASAHA ULOS -tulojen tilaa, ja kutsuu SAWS_STATES-funktiota tulojen aktivoituessa. SAWS_STATES-funktion tehtävä on tuottaa sahan liikepulssit sekä ohjata SAHA KOTONA- ja LATVASAHA KOTONA -antureiden toimintaa.

6 POHDINTA

Opinnäytetyö aloitettiin määrittelemällä simulaattorin vaatimukset ja ominaisuudet yrityksen kahden ohjelmistosuunnittelijan kanssa. Oma tieto ja kokemus metsäkoneista sekä niiden hakkuupäistä oli hyvin vähäinen, joten järjestelmiin tutustuminen vaati huomattavasti aikaa ennen kuin kykeni aloittamaan varsinaisen suunnittelun. Toinen haaste työssä oli ohjelmointiosuus, jossa CoDeSys-sovelluskehitystyökalu oli ennestään tuttu, mutta kokemuksena oli vain hyvin yksinkertaisten sovelluksien tekeminen. Tässä työssä suurin osa simulaattorin ominaisuuksista ja toiminnoista tehtäisiin sovelluksen avulla.

Simulaattorin toteutuksen suunnittelu aloitettiin sähköisistä ja mekaanisista ominaisuuksista, koska ne olivat yksinkertaisimmat ja selkeimmät osuudet suunnittelussa. Mekaniikasta tehtiin kaksi versiota, koska ensimmäisessä versiossa olisi joutunut työstämään liikaa salkkua ja tiiveyden säilyttäminen olisi ollut tällöin huomattavasti vaativampaa.

Ohjelmiston toteutus osoittautui ennakoidusti vaativimmaksi osuudeksi työstä. Sovelluksen tekemistä kuitenkin helpotti sovelluksen alustana olevan moduulin sähköisten ominaisuuksien ennestään tunteminen. Sovelluksen perus-ominaisuudet, kuten puun pituuden ja halkaisijan sekä sahan asennon pulssijonojen tuottaminen, toimivat suunnitellun mukaisesti. Myös sahojen alkuasentoa kuvaavien anturien simulointi toimii suunnitellusti. Simulaattorin käyttökokemukset ovat tämän työn kirjoitushetkellä hyvin vähäiset. Mekaniikan rakentaminen on vienyt odotettua kauemmin aikaa, jolloin toteutetut sähköiset toiminnot ja ominaisuudet on testattu yksittäisinä toimintoina.

Selkein kehityskohde simulaattorissa on puiden runkoprofiilit. Nykyisessä toteutuksessa rungot ovat kuvattu lineaarisesti tyvestä latvaan kaventuvina, jollaisia ne eivät ole todellisuudessa. Tämän toteutus toisi realistisuutta lisää simulointiin, mutta siitä on toistaiseksi luovuttu aika syistä.

7 YHTEENVETO

Tämän opinnäytetyön tarkoituksena oli toteuttaa metsäkoneen hakkuupään simulaattori, jolla voitaisiin testata ohjaus- ja mittausjärjestelmän sovellusta. Vastaavanlaisia simulaattoreita löytyy lähes jokaiselta eri ohjaus- ja mittausjärjestelmän valmistajalta, mutta ne, joista löytyy kirjallisuutta, ovat tyypillisesti paljon laajempia kokonaisuuksia. Kyseessä voi olla esimerkiksi virtuaalisimulaattori, jonka käyttötarkoitus on kouluttaa metsäkoneen kuljettajaa käyttämään järjestelmää mahdollisimman realistisessa ympäristössä ennen oikean metsäkoneen käyttöä.

Jokaisen valmistajan simulaattori on spesifi ja soveltuu käytettäväksi vain ohjaus- ja mittausjärjestelmän valmistajan laitteissa. Tässä työssä tämä korostui vielä enemmän, koska simuloiva moduuli ja sen sovellus on toteutettu yrityksen omista tuotteista.

Tämän työn lopputuloksena syntyi simulaattori, joka kykenee testaamaan ohjaus- ja mittausjärjestelmän sovellusta simuloimalla hakkuupäässä tyypillisesti esiintyvien antureiden signaalit. Näitä antureita ovat pulssianturit, jotka tuottavat puun pituuden ja halkaisijan mittatiedon sekä sahan asentotiedon. Tämän lisäksi simulaattori simuloi tyypillisesti induktiiviantureilla toteutetut saha kotona -tiedot niin sahan kuin latvasahan osalta. Nämä anturit voidaan ohjelmallisesti valita joko NPN- tai PNP-tyyppisiksi.

Tässä työssä toteutetun simulaattorin vahvuus on kompakti koko, joka mahdollistaa helpon liikuteltavuuden. Toisaalta se myös rajoittaa tämän simulaattorin jatkokehittämistä virtuaalisimulaattoriksi, koska realistisista käyttökokemuksesta on luovuttu esimerkiksi rotaattorin ohjaus joystickien pois jättämisellä. Toisaalta sovelluksella toteutetut anturisignaalit ovat helposti muokattavissa sekä vaihdettavissa toisentyyppisiksi, mikä antaa mahdollisuuden käyttää niitä myös hyvin toisenlaisessakin toteutuksessa.

LÄHTEET

[1] Banks, C. Principles of modeling and simulation: A Multidisciplinary approach. Hoboken, NJ, USA: Wiley, 2009.

[2] Singh, V. System modeling and simulation. Daryaganj, Delhi, IND: New age international, 2009.

[3] Kortelainen, J. VTT, ”Simulointi osana moniteknisen tuotteen tuoteprosessia”, [www-dokumentti]. Saatavilla: http://oci.oulu.fi/monidigi/Julkaisut (Luettu: 6.7.2011).

[4] Virtanen, L & Valli, T. Tampereen yliopisto, ”Simulointi ja WWW”, [www-dokumentti]. Saatavilla: http://www.cs.uta.fi/ipopp/www/ipopp97/valli-virtanen (Luettu: 25.7.2011).

[5] Sargent, R. Syracuse University, “Verification and validation of simulation models”, [www-dokumentti]. Saatavilla: http://surface.syr.edu (Luettu: 11.2.2012).

[6] Konttinen, H. & Drushka, K. Metsäkoneiden maailmanhistoria. Helsinki: Otava, 1997.

[7] Heikkilä, M. Englanninkielinen opintomateriaali harvesterien kaatopäistä. Tampereen Ammattikorkeakoulu, 2005. Saatavilla: http://www.theseus.fi (Luettu: 11.2.2012).

[8] Forest.fi, ”Sanasto”, [www-dokumentti]. Saatavilla: http://www.forest.fi (Luettu: 12.2.2012).

[9] Salokivi, J. Oppimateriaali puutavaralajimenetelmästä John Deere Forestry Oy:lle. Tampereen ammattikorkeakoulu, 2008. Saatavilla: http://www.theseus.fi (Luettu: 12.2.2012).

[10] SP-Maskiner, ”Company”, [www-dokumentti]. Saatavilla: http://www.spmaskiner.se (Luettu: 14.8.2011).

[11] Kesla Oyj, ”Tuotteet”, [www-dokumentti]. Saatavilla: http://www.kesla.fi/rh-heads (Luettu: 19.2.2012).

[12] Nyce, D. Linear position sensors: Theory and application. Hoboken, NJ, USA: Wiley, 2004.

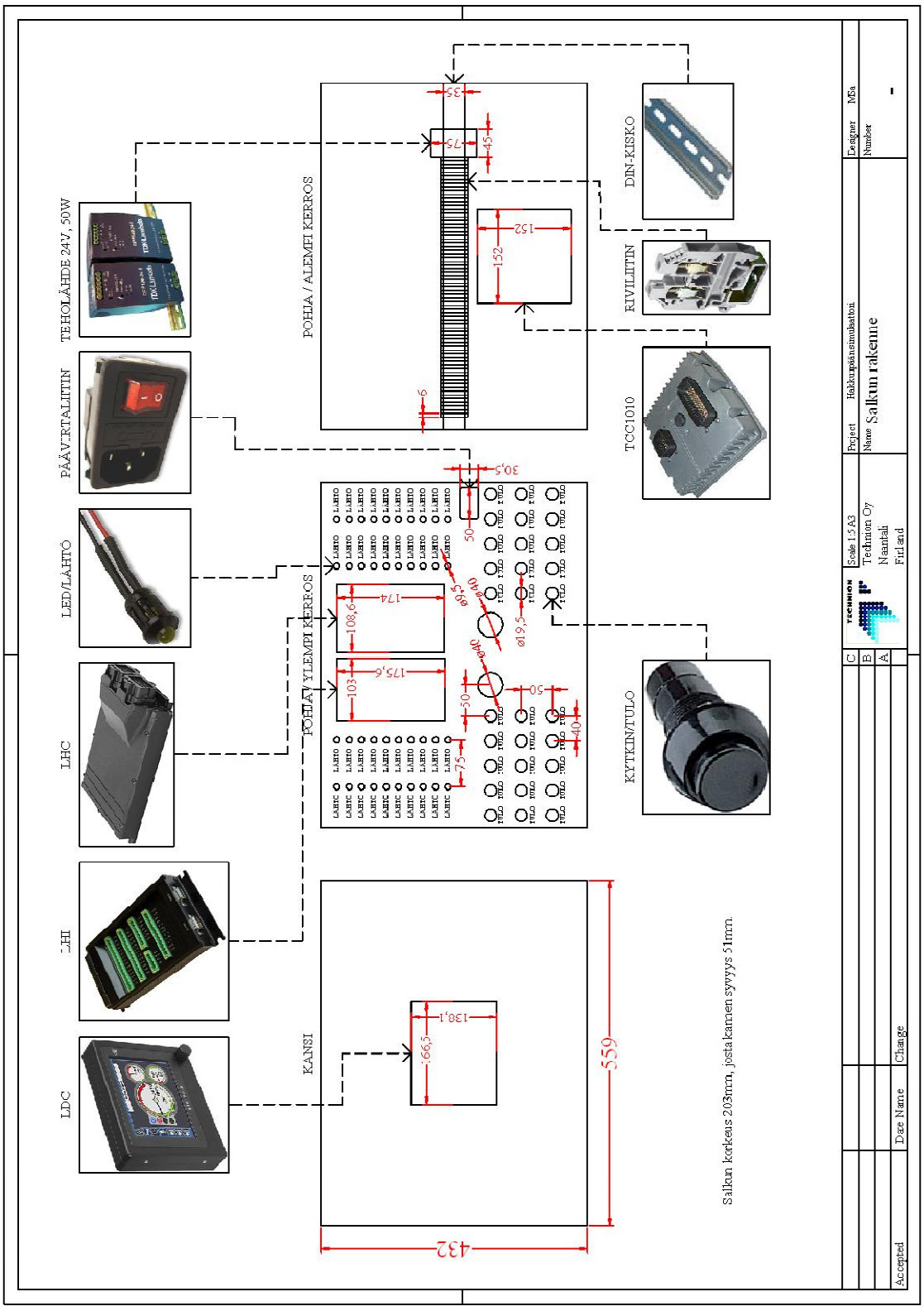
[13] Honkanen, H. Kajaanin ammattikorkeakoulu, ”Anturit”, [www-dokumentti]. Saatavilla: http://gallia.kajak.fi/opmateriaalit/yleinen/honHar/ma/ELE_A%20N%20T%20U%20R%20I%20T.pdf (Luettu: 19.6.2011).

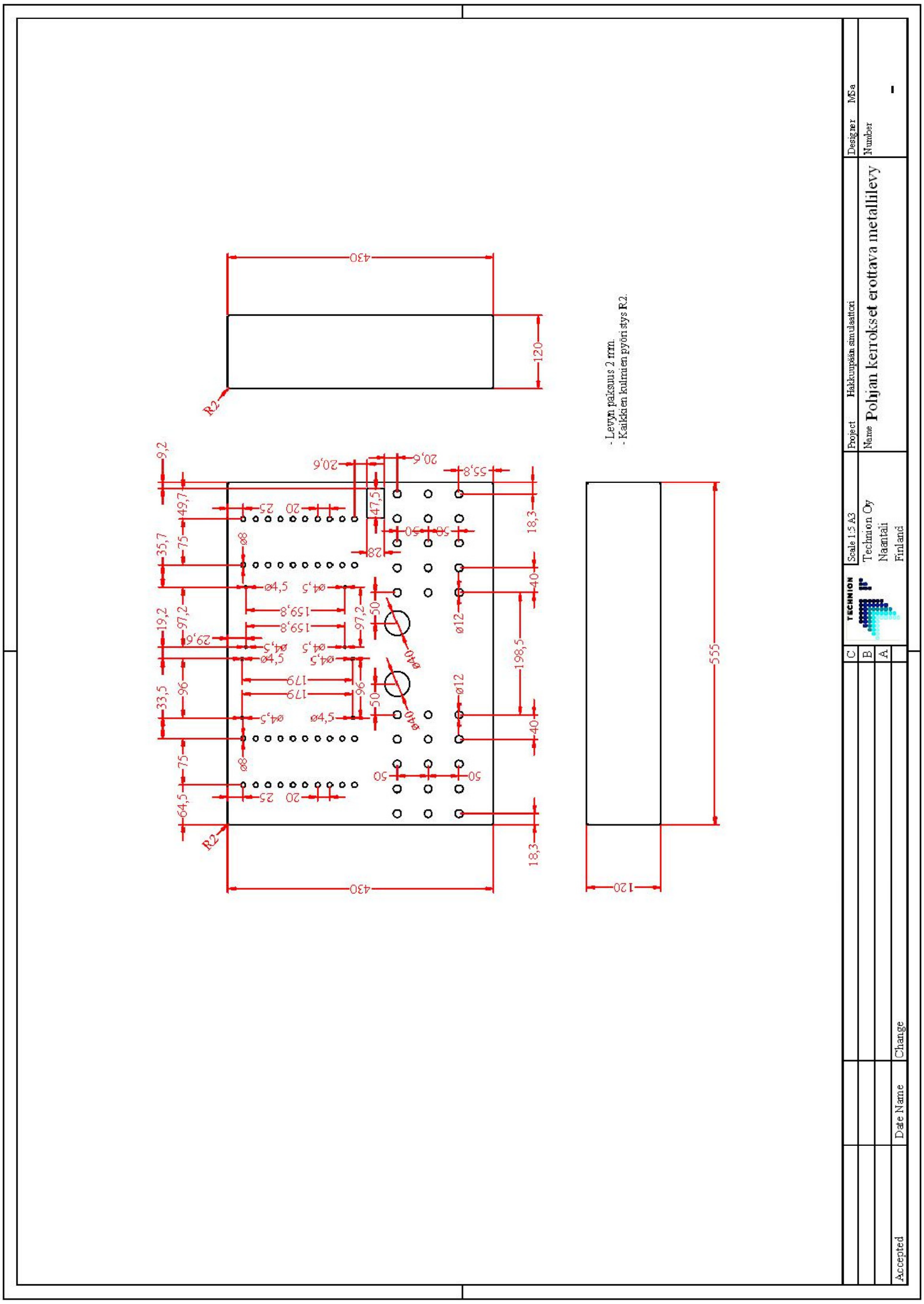
[14] SKS Group Oy, ”Tekel pulssianturit”, [www-dokumentti]. Saatavilla: http://www.sks.fi (Luettu: 3.3.2012).

[15] Technion Oy, ”Metsäkoneet”, [www-dokumentti]. Saatavilla: http://www.technion.fi (Luettu: 3.3.2012).

[16] Mäkinen, J. & Lehtinen, T. Keskustelu simulaattorin spesifikaatioista. 3.5.2011.

[17] 3S-Smart Software Solutions GmbH, “CoDeSys newcomer”, [www-dokumentti]. Saatavilla: http://www.3s-software.com (Luettu: 3.3.2012).


TEHOLÄHDE 24V, 50W
PÄÄVIRTALIITIN
LED/LÄHTÖ
LHC
LDC
POHJA / ALEMPI KERROS
DIN-KISKO
RIVILIITIN
TCC1010
KYTKIN/TULO
POHJA / YLEMPI KERROS
KANSI
Salkun korkeus 203mm, josta kannen syvyys 51mm
559
432
Scale 1:5 A3
Technion Oy
Nastola
Finland
Project Hakkuripäänsimulaattori
Name Salkun rakenne
Designer MSa
Number -
Accepted
Date Name
Change


- Levyn paksuus 2 mm.
- Kaikkien kulmien pyöristys R2.
Project Hakkuupään simulaattori
Name Pohjan kerrokset erottava metallilevy
Designer MSa
Number
Scale 1:5 A3
Technion Oy
Naantali
Finland
Accepted
Date Name
Change

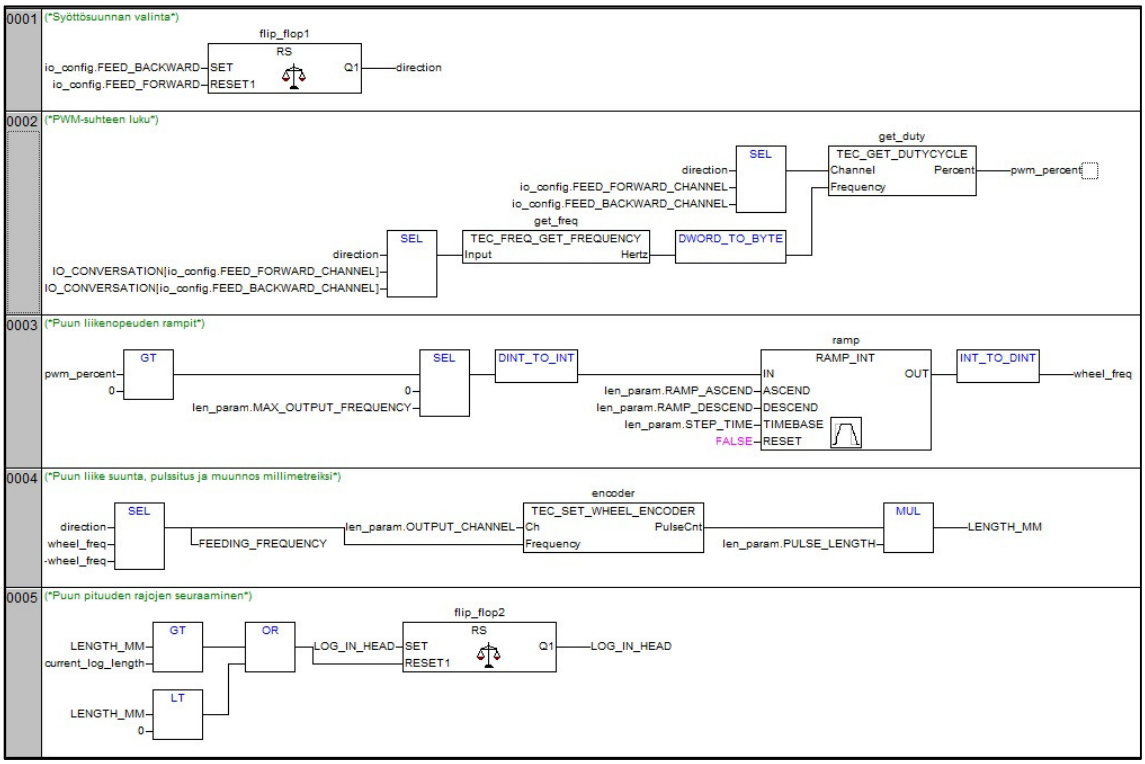
```
0001 PROGRAM PLC_PRG
0002 VAR
0003     leng: LENGTH;
0004     saws: SAWS_STATES;
0005     diam: DIAMETER;
0006     head: HEAD_STATE;
0007     pituus_mm: DINT;
0008     halkaisija_mm: DINT;
0009     saha_kotona: BOOL;
0010     sahan_asento: DINT;
0011     pituus_alustus: BOOL;
0012     halkaisija_alustus: BOOL;
0013     latvasaha_kotona: BOOL;
0014 END_VAR
```

```
0001 head();                                  (*kutsutaan HEAD_STATE-funktiota*)
0002 IF(NEW_LOG = TRUE) THEN                  (*uutta runkoa aloitettaessa*)
0003     pituus_alustus := INI(leng, TRUE);   (*alustetaan pituus-funktio*)
0004     halkaisija_alustus := INI(diam, TRUE); (*alustetaan halkaisija-funktio*)
0005 END_IF
0006
0007 IF (LOG_IN_HEAD = TRUE) THEN                                  (*puun ollessa hakkuupäässä*)
0008     leng();                                                   (*kutsutaan LENGTH-funktiota*)
0009     pituus_mm := leng.LENGTH_MM;                              (*apumuuttuja*)
0010     diam(FEEDING_FREQUENCY := leng.FEEDING_FREQUENCY);        (*kutsutaan DIAMETER-funktiota ja viedään LENGTH-funktion pulssien taajuus DIAMETER-funktioon*)
0011     halkaisija_mm := diam.DIAMETER_MM;                        (*apumuuttuja*)
0012 END_IF
0013
0014 IF((saws_param.SAW_OUT = TRUE) OR (saws_param.HEAD_SAW_OUT)) THEN   (*sahauksen aloitus*)
0015     saws();                                                         (*kutsutaan SAWS-funktiota*)
0016     saha_kotona := saws_param.SAW_HOME;                             (*apumuuttuja*)
0017     latvasaha_kotona := saws_param.HEAD_SAW_HOME;                   (*apumuuttuja*)
0018 END_IF
```

```
FUNCTION_BLOCK HEAD_STATE
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
    back_knives_state: BOOL := TRUE;
    feeding_rollers_state: BOOL := TRUE;
    front_knives_state: BOOL := TRUE;
    tilt_state: BOOL := FALSE;
END_VAR
```

```
    IF io_config.BACK_DELIMBING_KNIVES_CLOSE > AI_trigger_limit THEN
        back_knives_state := FALSE;
    ELSIF io_config.BACK_DELIMBING_KNIVES_OPEN > AI_trigger_limit THEN
        back_knives_state := TRUE;
    ELSE
        back_knives_state := back_knives_state;
    END_IF

    IF io_config.FEEDING_ROLLERS_CLOSE = TRUE THEN
        feeding_rollers_state := FALSE;
    ELSIF io_config.FEEDING_ROLLERS_OPEN = TRUE THEN
        feeding_rollers_state := TRUE;
    ELSE
        feeding_rollers_state := feeding_rollers_state;
    END_IF

    IF io_config.FRONT_DELIMBING_KNIVES_CLOSE > AI_trigger_limit THEN
        front_knives_state := FALSE;
    ELSIF io_config.FRONT_DELIMBING_KNIVES_OPEN > AI_trigger_limit THEN
        front_knives_state := TRUE;
    ELSE
        front_knives_state := front_knives_state;
    END_IF

    IF io_config.TILT_DOWN = TRUE THEN
        tilt_state := FALSE;
    ELSIF io_config.TILT_UP = TRUE THEN
        tilt_state := TRUE;
    ELSE
        tilt_state := tilt_state;
    END_IF

    IF ((back_knives_state = FALSE) AND (feeding_rollers_state = FALSE) AND (front_knives_state = FALSE)  AND (NEW_LOG = TRUE)) THEN
        LOG_IN_HEAD := TRUE;
        NEW_LOG := FALSE;
    END_IF

    IF ((back_knives_state = TRUE) AND (feeding_rollers_state = TRUE) AND (front_knives_state = TRUE) AND (tilt_state = TRUE)) THEN
        NEW_LOG := TRUE;
    END_IF

    RETURN;
```

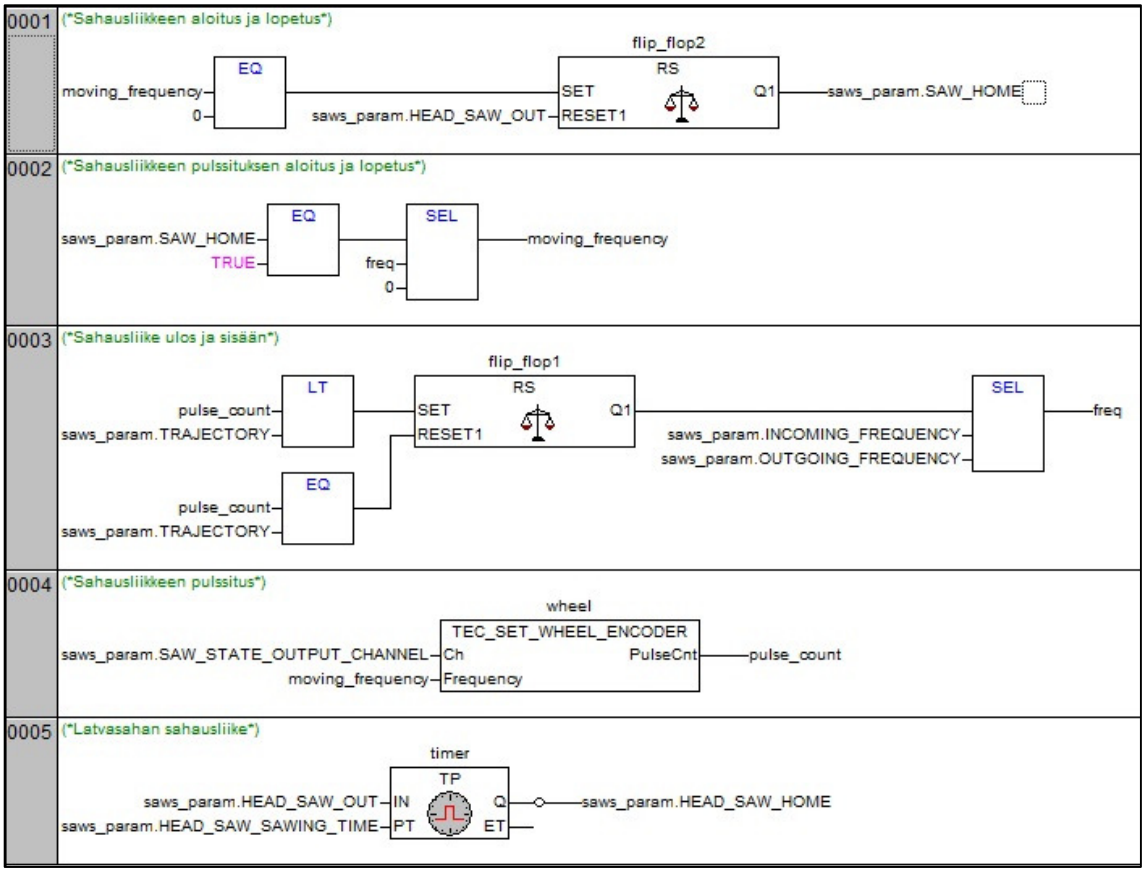
```
0001 FUNCTION_BLOCK LENGTH
0002 VAR_INPUT
0003 END_VAR
0004 VAR_OUTPUT
0005     FEEDING_FREQUENCY: DINT;
0006     LENGTH_MM: DINT;
0007 END_VAR
0008 VAR
0009     get_freq: TEC_FREQ_GET_FREQUENCY;
0010     get_duty: TEC_GET_DUTYCYCLE;
0011     encoder: TEC_SET_WHEEL_ENCODER;
0012     ramp: RAMP_INT;
0013     pwm_percent: DWORD;
0014     wheel_freq: DINT;
0015     direction: BOOL;
0016     flip_flop1: RS;
0017     flip_flop2: RS;
0018 END_VAR
```



```
0001 FUNCTION_BLOCK DIAMETER
0002 VAR_INPUT
0003     FEEDING_FREQUENCY: DINT;
0004 END_VAR
0005 VAR_OUTPUT
0006     DIAMETER_MM: DINT;
0007 END_VAR
0008 VAR
0009     wheel: TEC_SET_WHEEL_ENCODER;
0010     freq: DINT;
0011     pulse_count: DINT;
0012 END_VAR
```

```
0001 (*Halkaisija pulssien taajuus*)
     dia_param.MEASUREMENT_INTERVAL ─┐
                                    DIV ── FEEDING_FREQUENCY ──┐
     len_param.PULSE_LENGTH ────────┘                          DIV ── freq
                                                               ┘

0002 (*Puun halkaisija pulssitus*)
                                        wheel
                                  TEC_SET_WHEEL_ENCODER
     dia_param.OUTPUT_CHANNEL ──  Ch                PulseCnt ── pulse_count
     freq ──────────────────────  Frequency

0003 (*Halkaisijan muunnos millimetreiksi*)
     logs_prof.LOG_DIAMETER[current_log_diameter] ─┐
                                                  SUB ── DIAMETER_MM
     pulse_count ──────────────────────────────────┘
```

```
0001 FUNCTION_BLOCK SAWS_STATES
0002 VAR_INPUT
0003 END_VAR
0004 VAR_OUTPUT
0005 END_VAR
0006 VAR
0007     wheel: TEC_SET_WHEEL_ENCODER;
0008     pulse_count: DINT;
0009     freq: DINT;
0010     moving_frequency: DINT;
0011     selection: DINT;
0012     flip_flop1: RS;
0013     flip_flop2: RS;
0014     timer: TP
0015 END_VAR
```



```
0001 TYPE IO_CONFIGURATION :
0002 STRUCT
0003     FEED_FORWARD_CHANNEL: BYTE := 6;                    (*Eteensyötto-tulon kanava*)
0004     FEED_BACKWARD_CHANNEL: BYTE := 7;                   (*Taaksesyöttö-tulon kanava*)
0005     FEED_FORWARD AT %IX0.6: BOOL;                       (*Eteensyöttö-tulon osoite*)
0006     FEED_BACKWARD AT %IX0.7: BOOL;                      (*Taaksesyöttö-tulon osoite*)
0007     FRONT_DELIMBING_KNIVES_OPEN AT %IW5 : WORD;         (*Etuveitset auki -tulon osoite*)
0008     FRONT_DELIMBING_KNIVES_CLOSE AT %IW6 : WORD;        (*Etuveitset kiinni -tulon osoite*)
0009     BACK_DELIMBING_KNIVES_OPEN AT %IW8 : WORD;          (*Takaveitset auki -tulon osoite*)
0010     BACK_DELIMBING_KNIVES_CLOSE AT %IW9 : WORD;         (*Takaveitset kiinni -tulon osoite*)
0011     TILT_UP AT %IX0.11 : BOOL;                          (*Kallistussylinteri ylös -tulon osoite*)
0012     TILT_DOWN AT %IX0.10 : BOOL;                        (*Kallistussylinteri alas -tulon osoite*)
0013     FEEDING_ROLLERS_OPEN AT %IX0.9 : BOOL;              (*Syöttörullat auki -tulon osoite*)
0014     FEEDING_ROLLERS_CLOSE AT %IX0.8 : BOOL;             (*Syöttörullat kiinni -tulon osoite*)
0015 END_STRUCT
0016 END_TYPE
0017 (*
0018 PWM duty cycle capturing inputs
0019 0 => DI1 = X2-4
0020 1 => DI2 = X2-5
0021 2 => DI3 = X2-6
0022 3 => DI4 = X2-7
0023 4 => DI5 = X2-8
0024 5 => DI6 = X2-9
0025 6 => DI7 = X2-10
0026 7 => DI8 = X2-11
0027 8 => DI9 = X2-12
0028 9 => DI10 = X2-33
0029 10 => DI11 = X2-34
0030 11 => DI12 = X2-35
0031 12 => DI13 = X1-11
0032 13 => DI14 = X1-13
0033 *)
```

```
0001 TYPE LOGS_PROFILES :
0002 STRUCT
0003     LOG_LENGTH: ARRAY [0..4] OF DINT := 10000,15000,20000,25000,30000;   (*Puun rungon pituus vaihtoehdot*)
0004     LOG_DIAMETER: ARRAY [0..4] OF DINT := 200,300,400,500,600;           (*Puun rungon halkaisija vaihtoehdot*)
0005 END_STRUCT
0006 END_TYPE
0007
```

```
0001 TYPE LENGTH_PARAMETERS :
0002 STRUCT
0003     OUTPUT_CHANNEL: BYTE := 1;                (*pituuspulssien lähdöt*)
0004     PULSE_LENGTH: DINT := 5;                  (*yhden pituuspulssin vastaavuus millimetreinä*)
0005     RAMP_ASCEND: INT := 100;                  (*nousevan rampin askel pulsseina*)
0006     RAMP_DESCEND: INT := 100;                 (*laskevan rampin askel pulsseina*)
0007     STEP_TIME: TIME := t#200ms;               (*rampin askeleen kesto*)
0008     MAX_OUTPUT_FREQUENCY: DINT := 1000;       (*pulssituksen maksimitaajuus*)
0009 END_STRUCT
0010 END_TYPE
0011 (*
0012 ENCODER OUTPUT CHANNELS (BYTE):
0013 1 = A = DO4   =X2-4,   B=DO5   =X2-5
0014 2 = A = DO6   =X2-6,   B=DO7   =X2-7
0015 3 = A = DO8   =X2-8,   B=DO9   =X2-9
0016  4 = A = DO10 =X2-10, B=DO11 =X2-11
0017 *)
0018
```

```
0001 TYPE DIAMETER_PARAMETERS :
0002 STRUCT
0003     OUTPUT_CHANNEL: BYTE := 3;                (*halkaisijapulssien lähtöjen valinta*)
0004     PULSE_LENGTH: DINT := 1;                  (*yhden halkasijapulssin vastaavuus millimetreinä*)
0005     MEASUREMENT_INTERVAL: DINT := 100;        (*mittausväli millimetreinä*)
0006     MAX_DIAMETER: DINT := 600;                (*hakkuupään maksimiavautuma millimetreinä*)
0007 END_STRUCT
0008 END_TYPE
0009 (*
0010 ENCODER OUTPUT CHANNELS (BYTE):
0011 1 = A = DO4 =X2-4, B=DO5 = X2-5
0012 2 = A = DO6 =X2-6, B=DO7 = X2-7
0013 3 = A = DO8 =X2-8, B=DO9 = X2-9
0014 4 = A = DO10 =X2-10, B=DO11 = X2-11
0015 *)
0016
```

```
0001 TYPE SAWS_PARAMETERS :
0002 STRUCT
0003     SAW_STATE_OUTPUT_CHANNEL: BYTE := 2;      (*sahapulssien lähdöt*)
0004     SAW_OUT AT %IX0.12 : BOOL;                (*sahalaippa ulos -tulo*)
0005     SAW_HOME AT %QX2.1 : BOOL := FALSE;       (*saha kotona -lähtö*)
0006     OUTGOING_FREQUENCY: DINT := 50;           (*sahalaipan ulosmeno taajuus(pulssia/sekunti)*)
0007     INCOMING_FREQUENCY: DINT := -100;         (*sahalaipan sisaantulo taajuus, aina negatiivinen(pulssia/sekunti)*)
0008     TRAJECTORY: DINT := 100;                  (*sahalaipan koko liikerata pulsseina*)
0009     HEAD_SAW_HOME AT %QX2.0 : BOOL := FALSE;  (*latvasaha kotona -lähtö*)
0010     HEAD_SAW_OUT AT %IX0.13: BOOL;            (*latvasaha ulos -tulo*)
0011     HEAD_SAW_SAWING_TIME: TIME := t#1000ms;
0012 END_STRUCT
0013 END_TYPE
0014 (*
0015 ENCODER OUTPUT CHANNELS (BYTE):        SAW HOME SENSORS (BOOL) :
0016 1 => A = DO4 = X2-4,      B=DO5  = X2-5      DOL1 => X2-1 = NPN
0017 2 => A = DO6 = X2-6,      B=DO7  = X2-7      DO1 => X2-1 = PNP
0018 3 => A = DO8 = X2-8,      B=DO9  = X2-9      DOL2 => X2-24 = NPN
0019 4 => A = DO10 = X2-10,    B=DO11 = X2-11     DO13 => X2-24 = PNP
0020 *)
```