Markus Markkanen

Proview-prosessinohjausjärjestelmä

Esittely ja käyttöönotto

Opinnäytetyö
Kevät 2011
Tekniikan yksikkö
Automaatiotekniikan koulutusohjelma
Koneautomaatio

Seinäjoen ammattikorkeakoulu
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: Tekniikan yksikkö

Koulutusohjelma: Automaatiotekniikan koulutusohjelma

Suuntautumisvaihtoehto: Koneautomaatio

Tekijä: Markus Markkanen

Työn nimi: Proview-prosessinohjausjärjestelmä

Ohjaaja: Niko Ristimäki

Vuosi: 2011 Sivumäärä: 55 Liitteiden lukumäärä: 1

Tämän työn lähtökohtana oli tutkia Proview-prosessinohjausjärjestelmän sopivuutta paikalliskäyttöön tarkoitettua biomassan pyrolysointilaitosta varten. Proview on avoimen lähdekoodin ohjelmisto, jonka avulla voi tavallisella tietokoneella ajaa ohjelmoitavaa logiikkaohjelmistoa sekä valvoa, kerätä tietoa ja hallita prosessia tietoverkon välityksellä.

Tutkimus tehtiin asentamalla tietokoneelle Ubuntu-Linux-käyttöjärjestelmä ja Proview-ohjelmisto. Ohjelman toimintaa kokeiltiin PLC-kierron osalta askeltavalla perusohjelmalla, johon valittiin yleisiä logiikkakomponentteja. Koekytkentä ulkomaailmaan tehtiin käyttäen avoimeen laitteistoon pohjautuvaa Arduino-mikrokontrolleria. Tähän työhön kirjattiin ohjelmiston käyttöönotossa tarvittavat perustoimenpiteet, joilla ohjelmistoon voidaan kytkeä fyysinen I/O-liitäntä ja PLC-ohjelmoinnissa pääsee alkuun. Työssä käsitellään myös sellaisen tiedon perusvaatimuksia, jotka antavat eväitä monimutkaisempien Proview-käyttöjen luomiseen.

Tämän työn tuloksena havaittiin, että Proview ohjelmisto sopii hyvin lähtökohtana olleen biomassan pyrolysointilaitoksen ohjaus- ja hallintajärjestelmän perustaksi.

Tietolähteinä käytettiin pääosin Ubuntu- ja Proview-ohjelmistojen sisäistä ohjeistusta. Arduino-mikrokontrollerista löytyi tietoa Arduino-kehityssivustolta. Hyväksi tietolähteeksi ongelmatapauksissa havaittiin Proview-ohjelmiston keskustelupalsta.

Avainsanat: Proview, PLC, SCADA, ohjelmoitava logiikka, Arduino

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology

Degree programme: Automation Technology

Specialisation: Machine Automation

Author: Markus Markkanen

Title of the thesis: Proview Process Control System

Supervisor: Niko Ristimäki

Year: 2011 Number of pages: 55 Number of appendices: 1

The aim of this thesis was to investigate the possibility of using Proview Process Control System to control the operation of a small scale biomass pyrolysis plant. Proview, which is an Open Source Process Control System, can be used for designing and running SoftPLC (Software Programmable Logic Controller) in an everyday computer environment. Furthermore, it can be used across the same computer network for process supervision and data acquisition from different nodes which are separated from a computer running Proview as a programmable controller.

First the Ubuntu-Linux operating system, and then the Proview software in Ubuntu were installed onto a computer. The main functions of the PLC-program were thereafter tested with the Sequential Function Chart-type Grafcet language. The real world input/output was tested by using Arduino, an open source hardware microcontroller. Sources of information for this thesis were primarily internal manuals included in Ubuntu and Proview software. Information about the Arduino microcontroller hardware was obtained from the Arduino homepage.

The result of this thesis is that Proview software is capable to perform as a suitable PLC- and operator environment in a biomass pyrolysis plant. The necessary steps for getting the Proview software running, and sources of information and techniques that are essential to the system designer for effective usage of the Proview system resources are explained in this thesis.

Keywords: Proview, PLC, SCADA, programmable logic, Arduino

SISÄLTÖ

Kuvaluettelo

Käytetyt termit ja lyhenteet

Attribuutti Luokassa oliolle määritelty tietomuuttuja.

Avoin laitteisto Laitteen valmistamiseen tarkoitetut suunnitelmat, esim. piirikaaviot ovat julkisia. Suunnitelmiin saa tehdä muutoksia ja laitteita saa myydä kaupallisesti mutta muunnellut suunnitelmat tulee julkaista samalla tavoin avoimesti kuin alkuperäiset.

Avoin lähdekoodi Avoimella lähdekoodilla tarkoitetaan ohjelmistojen julkaisutapaa, jossa ohjelmistosta on saatavilla lähdekoodi ja joka on julkaistu jollain avoimen lähdekoodin lisenssillä. Lähdekoodia voi muokata kuka vaan ja käyttää omiin, jopa kaupallisiin tarkoituksiin. Koodiin tehdyt muutokset tulee kuitenkin myös julkaista avoimen lähdekoodin lisenssillä.

Linux-jakelu Ohjelmistokokoelma joka muodostaa yleensä kokonaisen käyttöjärjestelmän Linux-ytimen päälle.

Linux-ydin Ydin eli kerneli määrittelee käyttöjärjestelmän rakenteen ja ominaisuudet. Kerneli ohjailee mm. laiteajureita ja virtuaalimuistin määrää. Käyttöjärjestelmä käyttää laitteistoa kernelin välityksellä. (Bovet & Cesati, 2005, 8-34.)

Luokka Luokka kuvailee minkälaisena tiettyyn luokkaan kuuluva olio näyttäytyy. Saman luokan olioilla on yhteisiä piirteitä.

Olio Olio muodostuu tiedoista jotka määrittelevät olion tilaa tai sen ominaisuuksia.

PLC Programmable Logic Controller. Ohjelmoitava logiikka, jota käytetään automaatioprosessien ohjaamisessa.

Pääte-emulaattori Merkkipohjainen käyttöliittymä tiedostojärjestelmän selailuun ja järjestelmän ohjailuun yksittäisillä käskyillä.

SCADA SCADA (Supervisory Control And Data Acquisition) on tietokoneohjelmisto, jolla voi valvoa ohjelmoitavaa logiikkaa ja jossa on kehittyneet tiedoston talletus- ja välitysmahdollisuudet. Tässä työssä käsitelty Proview-ohjelmisto sopii kyseisen määritelmän piiriin. (Bailey & Wright. 2003. 1-4.)

SoftPLC Ohjelmisto, jolla voidaan ajaa logiikkaohjelmistoa tietokoneen omalla suorittimella peruskäyttöjärjestelmän alaisuudessa. Käyttöjärjestelmän tulee sisältää reaaliaikalaajennos eli varata logiikkaohjelmalle prosessoriaikaa tasaisin aikavälein.

Solmu Solmu on tietue, jonka avulla rakennetaan linkitettyjä tietorakenteita. Solmu sisältää tietoa ja yhden tai useamman linkin muihin solmuihin.

1 LÄHTÖKOHTA

Tämän työn lähtökohtana oli tutkia sopivaa prosessinohjausjärjestelmää paikalliskäyttöön tarkoitettua biomassan pyrolysointilaitetta varten. Biomassaa pyrolysoidaan kerralla n. 1 m³, jonka jälkeen ylimääräinen hiilituhka kuljetetaan ruuvikuljettimen avulla ulos varastolavalle. Ennen pyrolyysia biomassaa kuivataan vesi-ilma-lämmönvaihtimen avulla. Lämpöenergiaa otetaan talteen uuteen kuivausilmaan ilma-ilma-lämmönvaihtimessa käyttäen hyväksi poistuvan ilman kondensointia.

2 OHJAINLAITTEISTOLTA VAADITTAVAT RESURSSIT

Laitteeseen sopivalla ohjausjärjestelmällä tulisi olla mahdollista ohjata analogisia ohjainlaitteita, kuten säädettävää vesipumppua ja puhallinta sekä ilmaa ja palokaasuja ohjaavia säätöventtiilejä. Digitaalisia ohjauksia relelähtöineen tulisi olla saatavilla mm. ruuvikuljettimelle ja biomassan syöttöluukulle. Lämpöä mittaavia antureita järjestelmässä tarvitaan esimerkiksi ilman, veden ja pyrolyysireaktion lämpötilojen mittaamiseen laitteiston eri kohdissa. Laitteiston lämpömittareina käytetään pääosin K-tyypin termopareja omilla A/D-muuntimillaan, lisäksi ulkoilman lämpötilaa mittaa analogista signaalia antava yksittäinen piiri. Analogisia tuloja tarvitaan vielä takaisinkytkentään säätöventtiilejä ohjattaessa. Loppukäyttäjä ohjaa laitteistoa kosketusnäytöltä, joten ohjauksessa säästetään niiltä osin muutama fyysinen painikenappi. Ainoiksi fyysisiksi käyttäjän tuloiksi, jotka ohjataan I/O-kortin kautta, jää syöttöluukun ylös/alas-napit sekä hätäseis-nappi. Kaiken kaikkiaan projektissa tarvitaan analogista ja digitaalista I/O:a sekä lähtöinä että tuloina.

Järjestelmän ohjelmoinnin tulisi olla helppoa, varsinkin laitoksen kokeiluvaiheessa ohjelmakierron muuttaminen tulisi olla yksinkertaista. Ohjelmakierron tulee olla reaaliaikainen ja vaadittavaksi kiertonopeudeksi arvioidaan 20 ms.

Laitteen loppukäyttäjää ajatellen järjestelmällä pitäisi pystyä esittämään prosessi mahdollisimman selkeässä muodossa, vastaten kuvaa reaalimaailmasta. Biomassan pinnankorkeus, lämpötilat järjestelmän eri osissa, ruuvikuljettimen toiminta jne. tulisi sisällyttää kosketusnäytöllä esitettyyn kuvaajaan, josta voi yhdellä silmäyksellä nähdä toiminnan eri tiloissa sekä mahdolliset hälytykset. Prosessia tulee pystyä seuraamaan ja hallitsemaan verkon välityksellä niin, että prosessin alkamisajankohtaa voi muuttaa ja pyrolyysiprosessin sekä kiertoveden lämpötilan kehitystä voi seurata. Tämä tarkoittaa myös sitä, että prosessista on saatavilla kerättyä tietoa kuvaajia varten.

Lähtökohtien perusteella päädyttiin Proview-ohjelmistoon. Lisähyötynä ohjelmiston käyttöönotossa on sen halpuus. Itse ohjelmisto ei maksa mitään ja ulkoisen I/O-

liitännän valinnassa voi vapaasti etsiä sopivimman, sitoutumatta yhden valmistajan tuotteisiin.

Tässä työssä käydään läpi Proview-ohjelmiston perusasennus. Askeltavaa logiikkaohjelmaa havainnoidaan yksinkertaisella esimerkillä, kuitenkin käyden läpi käytetyn Grafcet-ohjelmointikielen perusosat. Kosketuskäyttöliittymän avulla käytettävästä sisäisestä I/O-liitännästä tehdään esimerkkitapaus ja fyysisiä I/O-liitäntöjä käsitellään Arduino-mikrokontrollerin avulla.

3 PC-KÄYTÖN REAALIAIKAISUUS

PC-käytön heikkoutena PLC-laitteistoon verrattuna on ollut sen reaaliaikaisuuden puute. Reaaliaikaisessa logiikassa PLC-ohjelman kierto määrättynä ajankohtana on varmennettu suoritinkäskyjen avulla. Aikaisemmin useimpien tietokoneiden käyttöjärjestelmissä ei ollut komponentteja reaaliaikaisuuden varmentamiseen, vaan sen ohjelmat varasivat prosessoriaikaa muilla perusteilla. Ohjelmoitavassa logiikkakäytössä tällainen toiminta on riskialtista kun esimerkiksi samaan aikaan ajettava, tietoliikennettä käsittelevä komponentti, voisi jumittaa PLC-kierron sattumanvaraisesti. (Yaghmour, Masters, Ben-Jossef & Gerum, P. 2008. 351-364.)

Reaaliaikaisuuden vaatimus on joissakin tietokonepohjaisissa logiikkakäytöissä ratkaistu asentamalla PC-koneelle erillinen lisäkortti, jonka prosessorilla on ajettu reaaliaikaista käyttöjärjestelmää, jossa logiikkaohjelmaa ajetaan. Lisäkortti taas on ollut yhteydessä PC:n ei-reaaliaikaiseen käyttöjärjestelmään. Nykyisin eräissä tietokoneen käyttöjärjestelmäytimissä on otettu reaaliaikaisuus huomioon niin, että logiikkaohjelmaosa suorittaa ohjelmakierron määrättynä ajankohtana, riippumatta prosessoria kuormittavasta mahdollisesta toisesta ohjelmasta. (Yaghmour ym. 2008. 365.)

Tässä työssä esiteltävän Proview-prosessinohjausjärjestelmän logiikkaosa on reaaliaikainen. Sen logiikkakierron reaaliaikaisuus varmennetaan Linux-käyttöjärjestelmäytimen keskeytysajastimella. Ohjelmiston pohjana käytettävä Ubuntu-Linux-jakelu määrittää oletuksena prosessorin keskeytysajastimeksi 250 Hz, joka mahdollistaa ohjelman minimikiertoajaksi 4 ms. Asennuksessa Linux-ytimelle määritetyn keskeytysajastimen arvon voi tarkistaa komentamalla päätteellä: `grep CONFIG_HZ /boot/config-<käytettävän ytimen numero>`. Keskeytysajastimen kierron voi säätää vieläkin nopeammaksi kokoamalla oman Linux-ytimen ja näin saaduksi Proview-ohjelman kiertonopeudeksi on kehittäjien parissa saatu 100 ns. (Sjöfors 2010).

Keskeytysajastimen tihentämisellä on varjopuolensa. Mitä useammin prosessori joutuu käsittelemään keskeytyksiä, sitä hitaammin se käsittelee muita ohjelmia,

joten suorittimen tehovaatimus kasvaa. Mainittakoon, että käytettäessä nykyaikaista minikannettavaa, tässä työssä esiteltävillä kevyillä logiikkaohjelmilla, käytönaikaisen koneen hidastumisen logiikkakierron kanssa huomaa ainoastaan heikentyneenä videotiedoston toistona. Näin voidaan päätellä, että ainakaan peruskäytössä koneen tehovaatimus ei ole merkittävä. Isommissa järjestelmissä logiikkakäyttö voi olla hajautettua. Monimutkaiset logiikkaohjelmat voi eriyttää omille koneilleen, ja suorittaa samalla käytön hallintaa sekä valvontaa toisilla koneilla.

4 PROVIEW-OHJELMISTON ESITTELY

Proview on avoimen lähdekoodin lisenssillä julkaistu Linux-ytimeen perustuva ohjelmistopohjainen PLC-järjestelmä, jolla voi toteuttaa hajautettua prosessinohjausta ja -valvontaa. Ohjelmisto on kehitetty alkujaan ruotsalaisen terästehtaan SSAB:n tuotantojärjestelmien ohjaamista varten. Nykyään terästehtaalla on yli 400 eri Proview-ohjelmistolla ohjattua prosessia, ja n. 50 henkilöä kehittämässä sekä ylläpitämässä tuotannon Proview-järjestelmää. (Westman 2007.)

Proview-ohjelmistossa käytettävä GNU GPL on vapaiden ohjelmistojen julkaisuun tarkoitettu lisenssi, joka varmistaa että levitettävästä ohjelmasta on aina saatavilla myös lähdekoodi. Kuka tahansa voi käyttää tai muokata ohjelmaa, ja sitä voi myös käyttää kaupallisiin tarkoituksiin. Muutosten tekijä kuitenkin velvoitetaan julkaisemaan ohjelmaa edelleen lähdekoodin kanssa, jolloin paranneltu ohjelma palautuu kehittäjä- ja käyttäjäyhteisön eduksi. (GNU 2010.)

Proview voi kommunikoida toisten tietokoneiden kanssa Ethernetin välityksellä, tai käyttää erilaisia sarjaliikenneprotokollia. Erilaisia toimilaite-I/O-protokollia on useita. Yleisemmin käytetyistä mm. Profibus, Modbus ja Profinet ovat suoraan saatavilla Proview-järjestelmäkokoonpanossa. Eräille USB-I/O-korteille on valmiita ohjelmakirjastoja saatavilla. Proview-ohjelmiston sivustolta on ladattavissa ohjeet omien I/O-kirjastojen luontia varten, joten oman I/O-kirjaston lisääminen on mahdollista, jos tietoliikenneprotokolla on tiedossa. (Proview 2011.)

Proview-ohjelmistossa käytettävä Grafcet-ohjelmointikieli ei ole yhtenevä standardin IEC61131-3 kanssa, mutta kielestä löytyy monia vastaavia tai samanlaisia funktioita. (Sjöfors 2011.) Epäyhteneväisyys tosin estää keskinäiset ohjelmien siirrot muiden PLC-ohjelmien kesken.

Proview-ohjelmiston yksi vahvoista puolista, mutta samalla myös kompastuskivi, on sen oliosuuntautunut lähestymistapa, joka ei ole läheinen muihin markkinoilla oleviin PLC-järjestelmiin nähden. Oliosuuntautuneen ohjelmiston käyttöönotto ei ole suoraviivaista, vaan vaatii usein aikaa ja kärsivällisyyttä omaksua, samalla tavoin kuin olio-ohjelmoinnin opettelu. Järjestelmän omaksumisen jälkeen taas

etuna on nopea kehitystyö ja ohjelmiston parempi hallittavuus isoinakin kokonaisuuksina.

5 PROVIEW-OHJELMISTON ASENNUS

5.1 Asennuksesta yleisesti

Tässä työssä Proview-ohjelman asennus ja käyttö suoritetaan niin, että perustana olevaa käyttöjärjestelmää tai sen tärkeiden osien muokkausta ei tarvitse hallita. Kuitenkin Proview-ohjelmiston tehokas käyttö hajautettuna logiikka- ja prosessinvalvontajärjestelmänä vaatii jonkin verran Linux-ympäristön tuntemusta. Käytettävästä Linux-jakelusta olisi hyvä tietää ainakin tiedostojärjestelmän rakenne ja komentorivin käyttö pääte-emulaattorilta.

Nykyisin monien Linux-ydintä käyttävien käyttöjärjestelmien automatisoidut ohjelmien ja lisälaitteiden asennusohjelmat ovat tervetulleita ominaisuuksia sujuvan työpöytäkäytön takaamiseksi, mutta vähentävät samalla alustana olevan järjestelmän perustuntemusta. Syvällisempään Linux-tietämykseen pääsee käsiksi esimerkiksi lukemalla netistä löytyviä Linux-oppaita. Yksi helppolukuisimmista oppaista on *Introduction to Linux – A Hands on Guide* (Garrels 2008), joka opastaa yleisesti Linux-jakeluiden tiedostojärjestelmään ja komentorivin käyttöön. Kaikkia asioita oppaasta tuskin kannattaa lähteä heti opiskelemaan, vaan pitäytyä aluksi perusteissa, sillä monessa asiassa, kuten esimerkiksi tulostinlaitteiden säädössä, komentorivikäyttö tuskin antaa lisäarvoa graafisiin asennus- ja käyttöohjelmiin verrattuna.

Proview-ohjelmiston oliosuuntautunut lähestymistapa saattaa vaikeuttaa ohjelmistossa alkuunpääsyä. Käyttöä ei kannata opiskella suoraviivaisesti lukemalla käyttöohjeet. Parempi tapa käytön opiskeluun on tehdä toisinaan ohjelmia ja palata ohjeistuksen pariin aika-ajoin. Näin kokonaiskäsitys ohjelmiston olio-ajattelutavasta kehittyy asteittain. Tässä työssä ei käsitellä tarkoituksellisesti esimerkiksi asetuksissa tarvittavia osasia oliomallin ja -nimien avulla. Oliomallina ohjelman ja sen osaset voi mieltää, kun ohjelmiston rakenteeseen saa käytön avulla tuntumaa.

5.2 Ohjelmiston asennus

Tässä ohjeistuksessa oletetaan että koneelle on asennettu Ubuntu-Linux-käyttöjärjestelmä graafisella käyttöliittymällä. Ubuntun voi ladata Ubuntu Suomi -sivustolta (Ubuntu Suomi 2011), jolta löytyy myös asennusohjeet.

Proview-ohjelmiston sivuilta löytyy latauslinkki Proview-ohjelmapakettiin ja sen komponentteihin (Proview 2011). Valitaan ohjelmapaketti pwrXX (XX on senhetkinen ohjelmistonumero esim. pwr48), jonka asennus sisältää prosessiohjelman kehitysympäristön ja ohjelmiston käytönaikaisen hallintaohjelmiston sekä tallennusympäristön. Ohjelmapaketista valitaan asennettavalle käyttöjärjestelmälle sopiva versio esimerkiksi 32-bittiselle Ubuntulle pwr48_4.8.1-2ubuntu_i386.deb.

Ohjelmisto asennetaan latauksen jälkeen kaksoisnapsauttamalla ohjelmapaketin kuvaketta, jolloin Ubuntun sovellusvalikoima avautuu ja kysyy pääkäyttäjän salasanaa asennuksen varmistamiseksi. Asennuksen jälkeen käyttöjärjestelmä käynnistetään uudelleen. Käyttöjärjestelmään on nyt luotu uusi käyttäjätunnus pwrp. Kirjaudutaan ulos ja vaihdetaan käyttäjäksi pwrp. Salasana on sama kuin tunnus eli pwrp.

Paketissa asentuvat samanaikaisesti kehitysympäristö ja käyttäjäympäristö. Peruskäyttäjä ei saa automaattisesti järjestelmän pääkäyttäjäoikeuksia. Annetaan käyttäjätunnukselle pwrp mahdollisuus järjestelmän tärkeiden osien muokkaukseen. Valitaan järjestelmän hallintaa varten Järjestelmäasetukset → Käyttäjät ja ryhmät, josta valitaan tunnus pwrp. Vaihdetaan tunnuksen tyypiksi ylläpitäjä. Vaihdetaan myös tunnukselle pwrp vahvempi salasana.

Seuraavaksi vaihdetaan järjestelmän alueelliset muotoilut, joihin kuuluu mm. desimaalipisteen käyttö ja ajannäyttöasetukset. Valitaan Järjestelmäasetukset → Kieliasetukset → Alueelliset muotoilut -välilehti. Valitaan kieliasetus English (United States). HUOM! Alueelliset muotoilut -asetusten manuaalinen vaihto jäänee ohjelmiston seuraavassa versiossa pois. Ominaisuus on kehittäjien tiedossa ja sitä pidetään ohjelmointivirheenä.

Näin järjestelmään on asennettu vaadittavat komponentit, jotta logiikkaohjelmia voi kehittää ja ajaa samalla koneella.

6 PROVIEW-OHJELMISTON KÄYTTÖ

6.1 Projektin kirjaaminen ja käytettävien asemien rekisteröinti

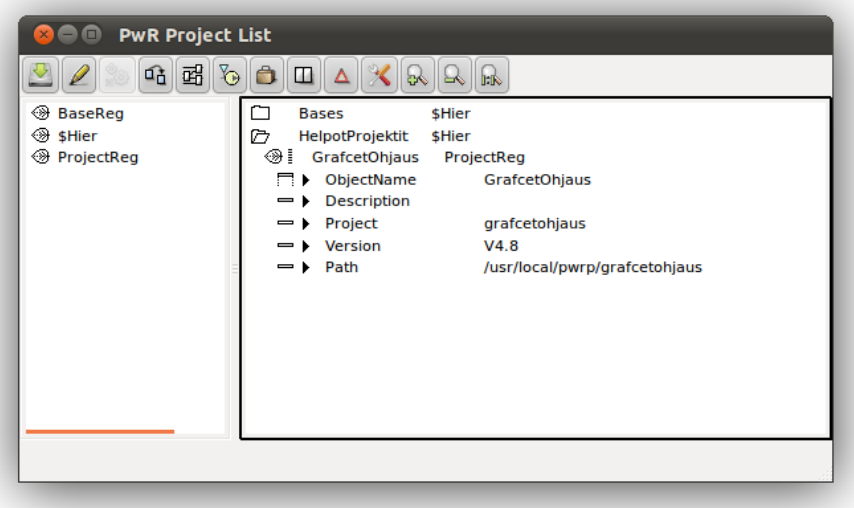
Proview-ohjelma käynnistetään kaksoisnapsauttamalla työpöydällä olevaa pikakuvaketta. Aluksi ohjelma avaa käyttäjän hyväksyttäväksi ohjelman levityksessä käytettävän GNU GPL -lisenssin. Käyttäjälisenssi hyväksytään, jolloin ruudulle jää kaksi ohjelmaikkunaa, tapahtumaikkuna ja projektilista. Development Console eli tapahtumaikkuna on kurkistus ohjelmiston sisäiseen maailmaan. Tapahtumaikkunaan ohjelma kirjaa käytön aikana ohjelmalle merkittävät tapahtumat. Tapahtumaikkunaa voi käyttää esim. virhetilanteiden analysointiin, jolloin ikkunaan kirjautuu virhe ja sen aiheuttanut asia. PwR Project List on projektilista, jolla käyttäjä aloittaa ohjelman käytön, joko valitsemalla aiemmin luodun projektin tai kirjaamalla uuden.

Luodaan aluksi uusi projekti. Valitaan muokkaustila napsauttamalla pikakuvaketta tai valitsemalla näppäimistöltä [Ctrl+E]. Valitaan pohja uudelle hierarkialle valitsemalla ikkunan vasemmasta puoliskosta kohde $Hier, sen tekstin kohdalta, jonka jälkeen hiiren keskinapilla tai rullalla napsautetaan ikkunan oikean puoliskon Bases-sanan päällä, jolloin sen alapuolelle syntyy uusi hierarkiapohja.

Proview-valikkojen muokkauksessa on huomioitava kohteiden valintakohdan merkitys. Tässä $Hier-kohdetta ei voi valita sen edessä olevan lehden päältä. Myöskään uutta hierarkiakohdetta ei luoda napsauttamalla Bases-kohteen kansion päällä, jolloin uudesta kohteesta tulisi hierarkiassa edellisen kohteen lapsi.

Annetaan uudelle $Hier-kohteelle nimi napsauttamalla sitä ja muokkaamalla kohteen ominaisuuksia. Kohteen valikossa liikutaan nuolinäppäimillä. Valitaan nuoli oikealle - nuoli alas - nuoli oikealle, jolloin ObjectName-kentän arvolle voi antaa nimen. Annetaan kentässä value-kohteen nimeksi HelpotProjektit. Luodaan projektihakemistolle vielä ensimmäinen projekti. Valitaan ikkunan vasemmasta paneelista ProjectReg, jonka jälkeen oikean paneelin HelpotProjektit-lehtikuvakkeen päällä (hiiren keskinappia napsauttamalla) luodaan sille lapsi. Annetaan ProjectReg-kohteelle edellisen esimerkin mukaan, nuolinäppäimillä

valikossa liikkuen, nimeksi GrafcetOhjaus (kuva 1). Lopuksi tallennetaan projekti valitsemalla [Ctrl+S], jolloin ohjelma avaa vielä varmistusikkunan toimenpiteelle. Hyväksytään projektin luominen ja poistutaan muokkaustilasta valitsemalla [Ctrl+E].

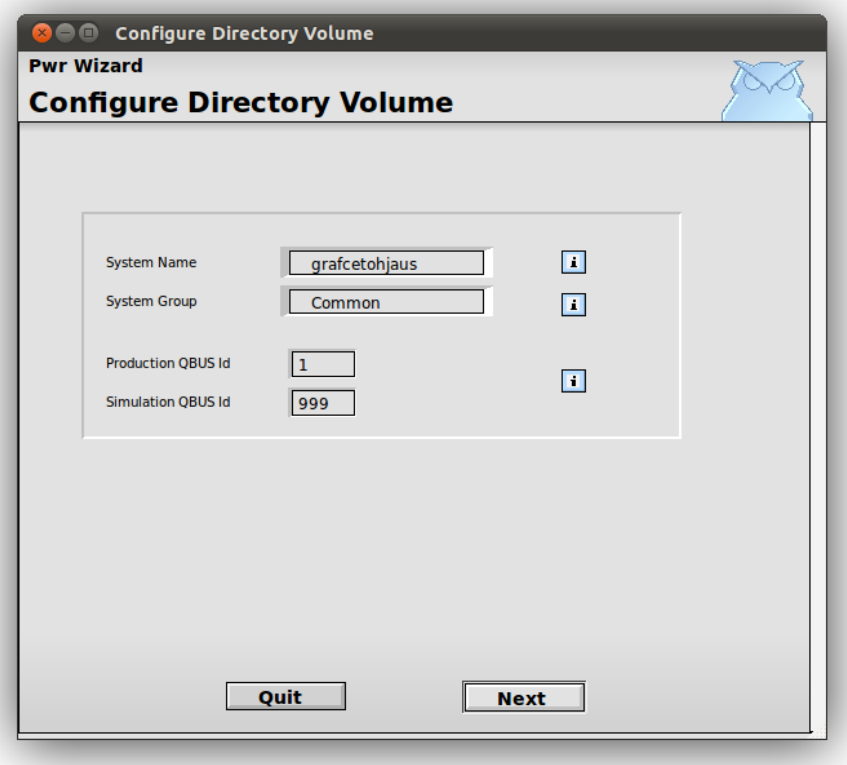


Kuva 1: Projektiluettelo

Avataan juuri luotu projekti napsauttamalla GrafcetOhjaus-kohdassa hiiren oikealla napilla ja valitsemalla valikosta Open Project. Ruudulle avautuu asennusvelho, jolla määritellään järjestelmään kuuluvat niteet (volume) ja solmut (node) hakemistoniteeseen Directory Volume, (kuva 2).

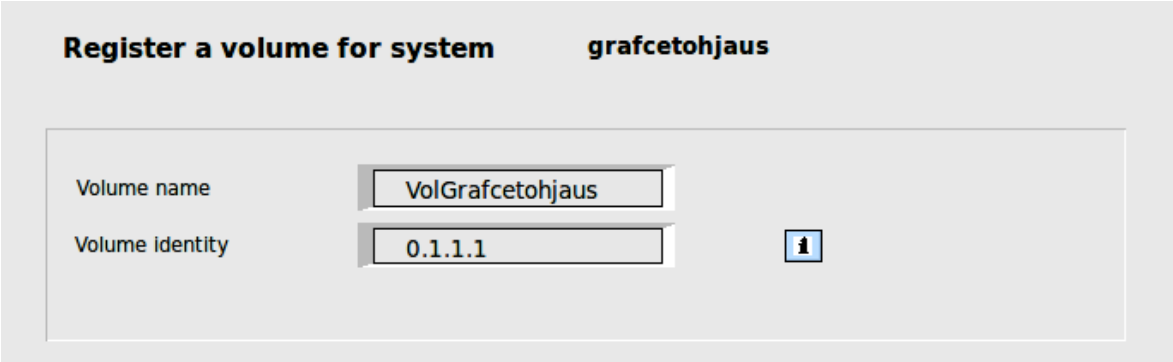
Automaatiojärjestelmää ja sen kokoonpanoa suunniteltaessa on yleensä selvää, mihin solmuun tietyt logiikassa käytettävät komponentit (eli logiikkaohjelmassa käytettävät osat kuten logiikkaohjelma, AND-lohko, digitaalinen syöte, analoginen syöte ym.) kuuluvat ohjelman ajon aikana. Proview-ympäristössä käytettävät komponentit kerätään niteisiin (Volume), joka on niille eräänlainen säiliö. Niteellä on nimi ja identiteetti, johon komponentit järjestetään puumalliseen tietueeseen.

Proview-järjestelmässä on erilaisia niteitä, josta osa toimii ajonaikaisina oliovarastoina. Kaksi useimmin käytettävää nideluokkaa ovat Root Volume ja Directory Volume, jotka seuraavassa alustetaan asennusvelhon avulla. Root Volume -nidettä muokataan vielä alustuksen jälkeenkin ja sen muokkaaminen logiikkaohjelman luomisen aikana on käyttäjälle tavallisinta.



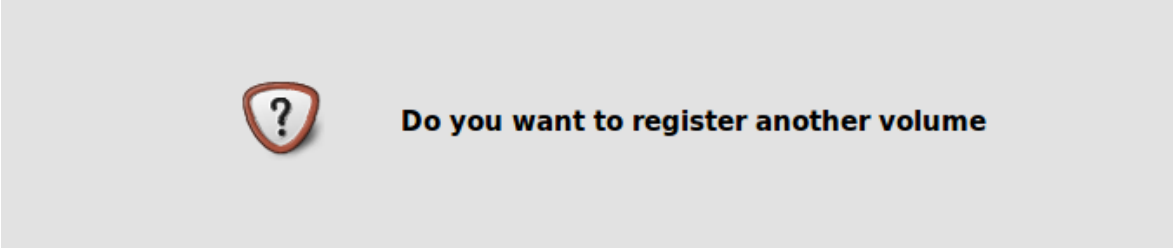
Kuva 2: Hakemistoniteen määrittelyä, kommunikaatioväylien numerointi

Asennusvelhon ensimmäisellä sivulla (kuva 2) määritellään järjestelmän nimi ja järjestelmäryhmä. Järjestelmän nimi System Name on yleensä yhtenevä projektin nimen kanssa. Järjestelmäryhmä System Group on jäsenmääre, jolla järjestelmään liitetään käyttäjät käyttäjätietokannassa. Häiriöttömän toimivuuden takaamiseksi verkko jaetaan tuotanto- ja simulaatioväylään, jotka tässä ovat 1 ja 999. Tässä käyttöliittymäkokeilussa ne tosin jäävät samalle tietokoneelle eikä tuotantoväylää edes avata. Asennusvelhossa olevat i-merkityt napit avaavat niiden vieressä olevaa kenttää vastaavan ohjesivun. Siirrytään seuraavalle sivulle valitsemalla Next.



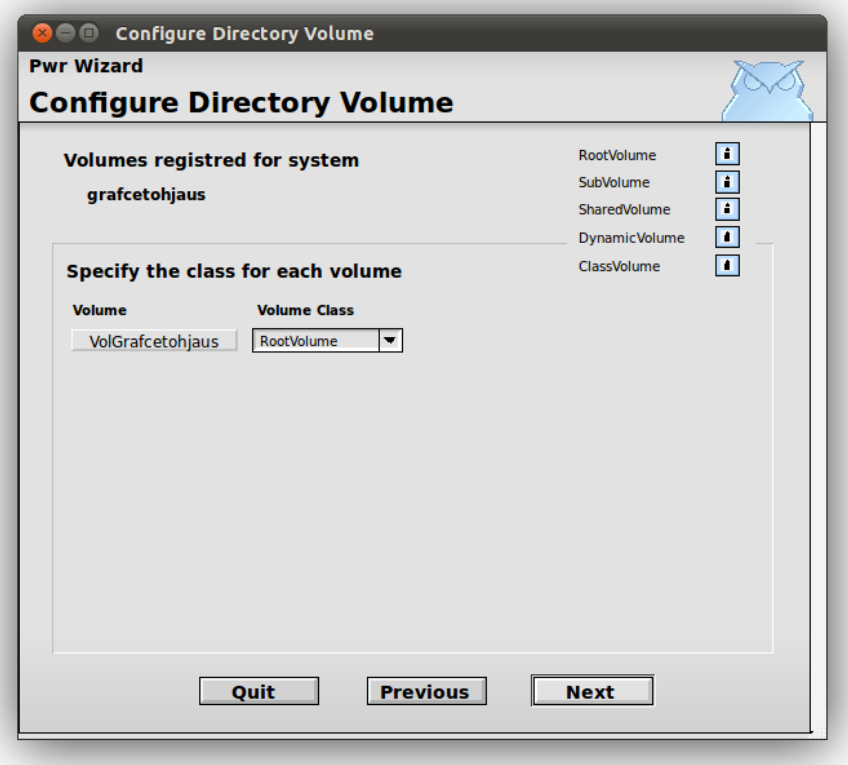
Kuva 3: Kirjataan nide VolGrafcetohjaus järjestelmälle grafcetohjaus

Kuvassa 3 rekisteröidään juurinide (Root Volume) järjestelmään. Jokaisen projektin jokaisella niteellä Proview-tietoverkossa tulee olla yksilöllinen nimi ja identiteettinumero. Juurinide VolGrafcetohjaus rekisteröityy yleiseen GlobalVolumeList-luetteloon. Valitaan Next.



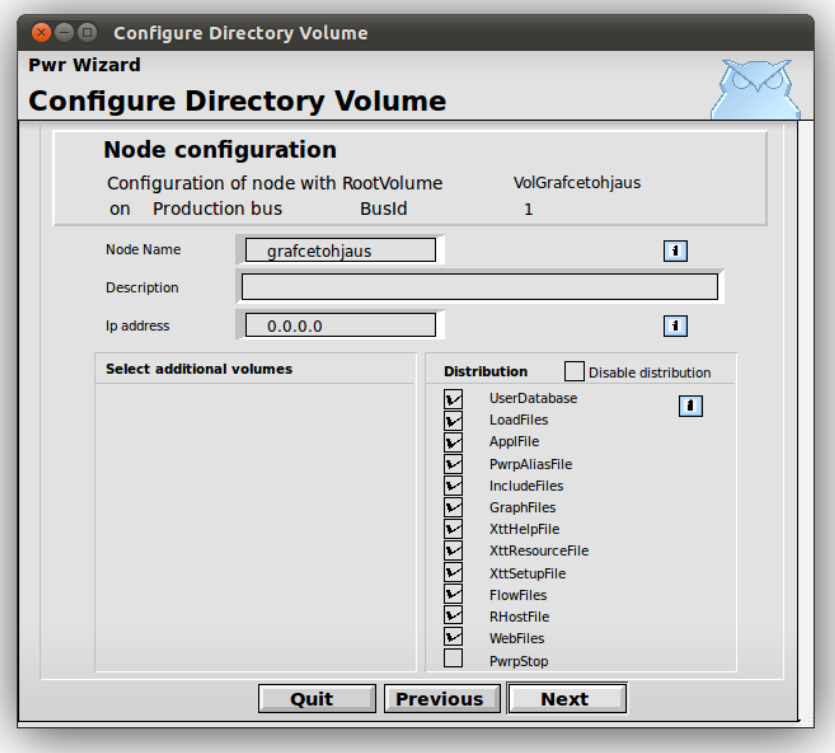
Kuva 4: Haluatko rekisteröidä uuden niteen?

Seuraavaksi asennusvelho tarjoaa uuden niteen rekisteröimistä (kuva 4). Jos järjestelmään kuuluisi lisää solmuja, esim. tiedonkeräysasema (storage station), tässä kohtaa voisi niteiden kirjaamista jatkaa. Ei rekisteröidä enempää niteitä, joten valitaan No.



Kuva 5: Järjestelmälle rekisteröidyt niteet

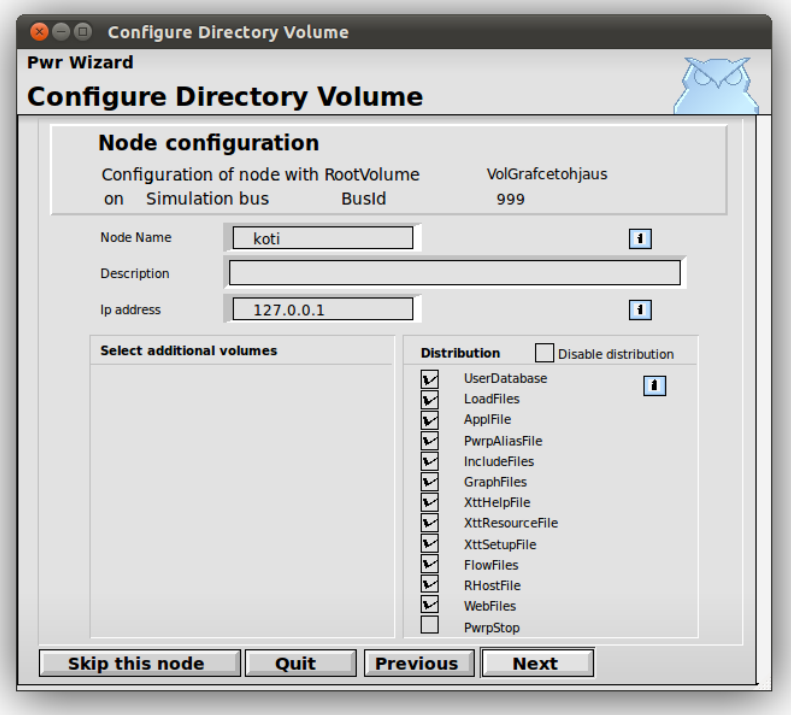
Erilaisille niteille voi määritellä niteen luokan. Kuvassa 5 nähdään, että ollaan määrittelemässä vain yhtä nidettä VolGrafcetohjaus (niteen Directory Volume lisäksi). Tässä työssä ei siis määritellä enempää niteitä, koska tarkoitus on valvoa prosessia samalta koneelta, jossa prosessi myös ajetaan. Ei tehdä muutoksia vaan jatketaan eteenpäin painamalla Next.



Kuva 6: Tuotantoväylän osoitteen määrittely

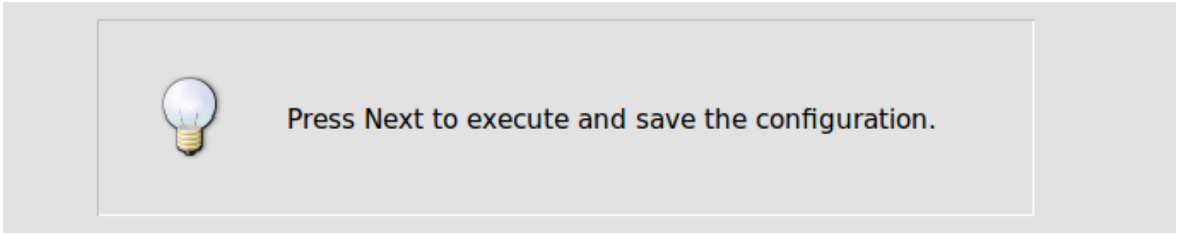
Hajautetussa valvontajärjestelmässä tuotantoväylälle annetaan sopiva IP-osoite (kuva 6). Jakelu (Distribution) -kenttä määrittelee minkälaisia tiedostoja jaetaan tuotantoväylällä tuotantoasemalle sekä valvonta- ja käyttöasemille.

Tätä solmua ei tulla tässä työssä tarvitsemaan, koska asemaa ajetaan simulaation ja myöhemmin asemaan liitetyn mikro-ohjainkortin avulla. Jätetään IP-osoitekenttä sekä muut valinnat alkuarvoonsa ja painetaan Next.



Kuva 7: Simulaatioaseman solmun määrittely

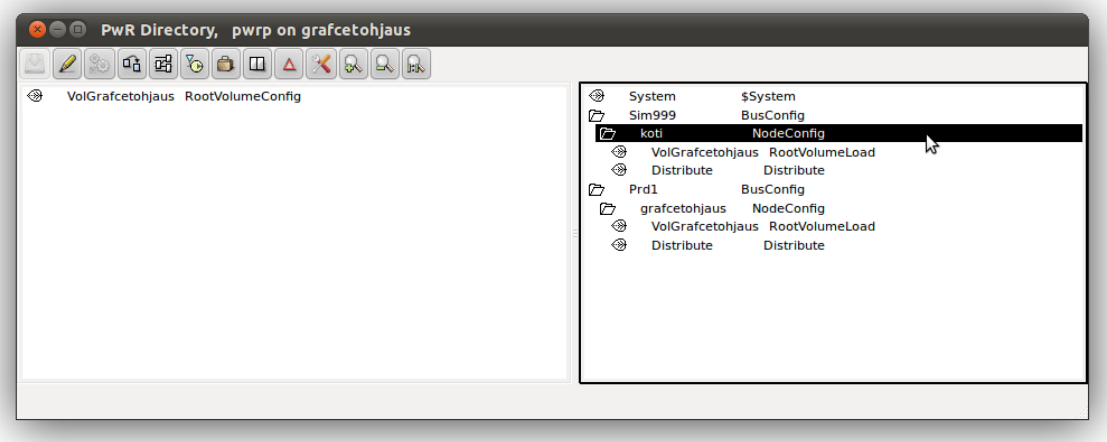
Tällä simulaatioasemalla ajetaan myöhemmin käytönaikaista prosessiympäristöä (kuva 7). Jätetään IP-osoitteeksi oletus 127.0.0.1, joka on yleinen oman koneen (localhost) osoite ja valitaan Next.



Kuva 8: Hyväksy muutokset ja tallenna kokoonpano

Ohjelma kysyy vielä varmistusta muutoksiin (kuva 8). Hyväksytään kokoonpano valitsemalla Next.

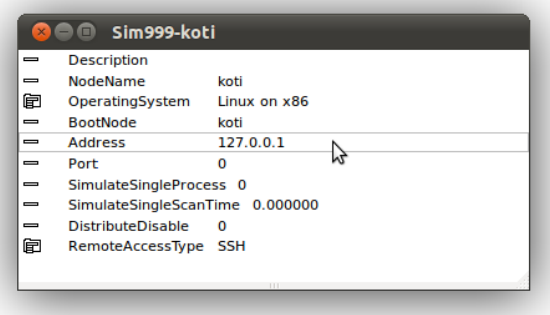
Ennen kuin jatketaan asennusvelhon käyttöä, tarkastellaan asennusvelhon tähän asti luomaa Directory Volume -kokoonpanoa (kuva 9).



Kuva 9: Konfiguroitu hakemistonide Directory Volume

Ikkuna on jaettu kahteen paneeliin. Vasemmalla puolella on konfiguroidut niteet ja oikealla puolella konfiguroidut solmut. Väyläkonfiguraation (BusConfig) alla lapsina sijaitsevat RootVolumeLoad ja Distribute. RootVolumeLoad määrittelee, mikä nide ladataan Proview-ajonaikaisen ympäristön käynnistyksessä. Distribute sisältää asennusvelhon aikana valitun tiedostonjakelun (ks. kuva 7).

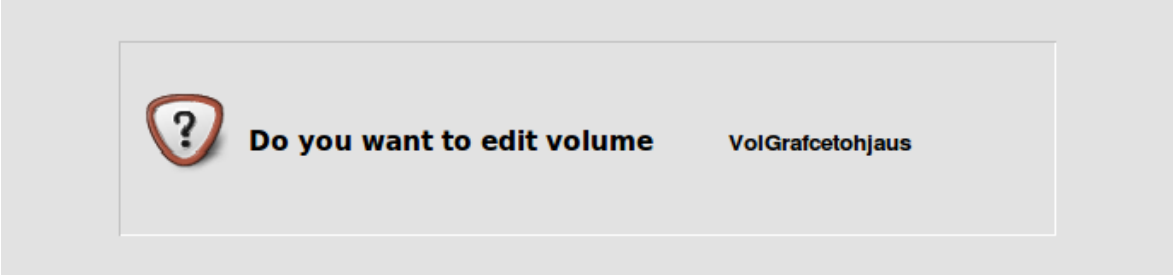
Tarkastellaan solmukonfiguraatiota: Tässä ”koti” on järjestelmän peruskäyttäjä. Valitaan oikeasta paneelista peruskäyttäjän kohdalta alue (ks. kuva 9) ja napsautetaan hiiren oikeaa nappia. Valitaan Open Object, jolloin voidaan tarkastella solmun asetuksia. Voidaan havaita, että Address-kenttä on aikaisemmin valittu 127.0.0.1 (kuva 10). Suljetaan simulaatiosolmun määrittelyikkuna.



Kuva 10: Simulaatiosolmun määrittelyikkuna

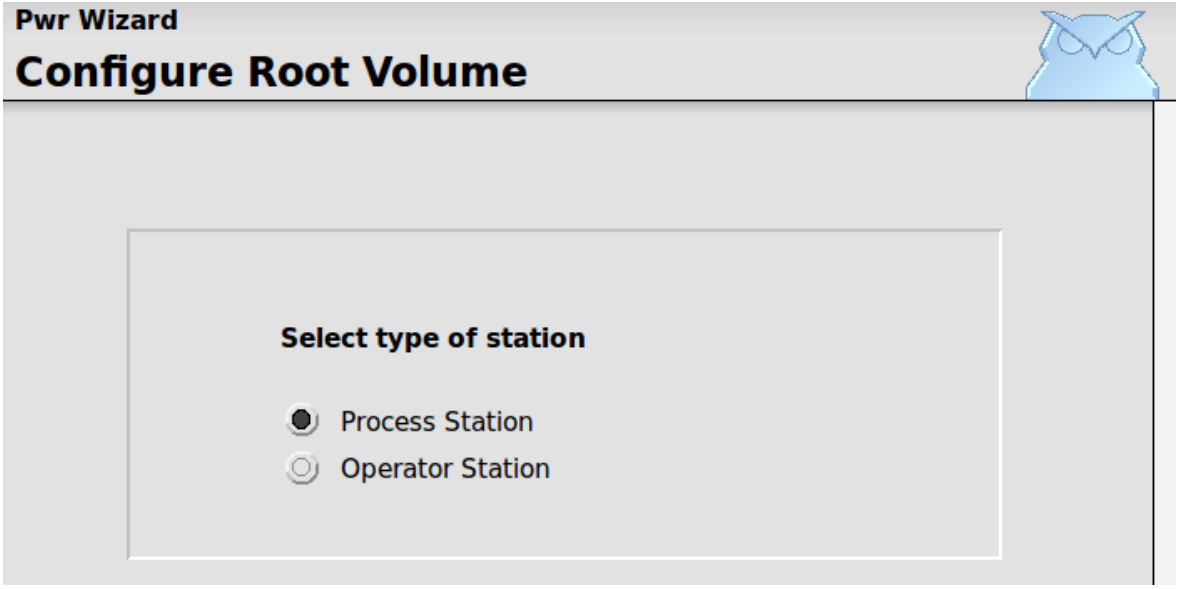
Asetuksia voi myöhemmin muuttaa tai niteitä tai solmuja lisätä valitsemalla hakemistoniteen kohdalla muokkaustilan [Ctrl+E].

Palataan asennusvelhoon (kuva 11).



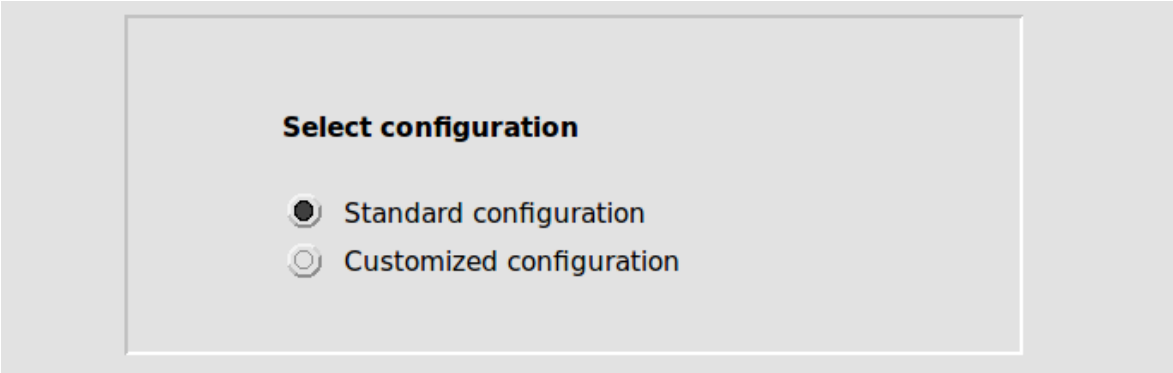
Kuva 11: Haluatko muokata juurinidettä VolGrafcetohjaus?

Vastataan kysymykseen myöntävästi Yes. Asennusvelho avaa taustalle tyhjän ikkunan juuriniteelle ja jatkaa kysymyksillä (kuva 12).



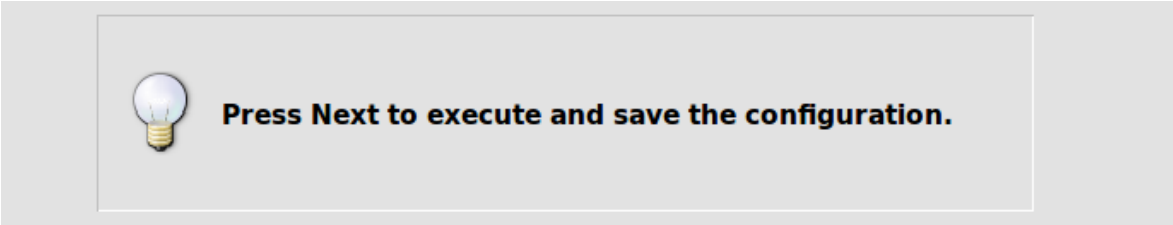
Kuva 12: Juuriniteen asematyypin valinta

Asennusvelho kysyy aseman lajia. Valitaan prosessiasema (Process Station). Prosessiasema sisältää käyttäjäaseman (Operator Station) ja tallennusaseman (Storage Station) toiminnallisuuden, joten järjestelmää voidaan käyttää samalta koneelta. Jatketaan painamalla Next.



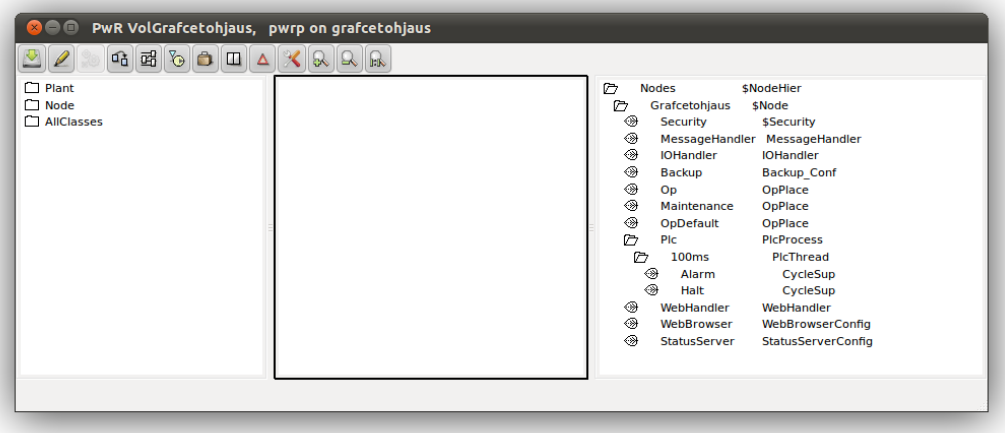
Kuva 13: Juuriniteen kokoonpanon valinta

Valitaan juuriniteelle peruskokoonpano (kuva 13) ja painetaan Next.



Kuva 14: Hyväksy ja tallenna juuriniteen kokoonpano

Kokoonpanon voi vielä hyväksyä tai perua (kuva 14). Hyväksytään se painamalla Next. Juuriniteen hallintaikkuna avautuu muokkaustilassa (kuva 15).



Kuva 15: Juuriniteen hallintaikkuna

HUOM! Projektin, jossa samalla koneella on sekä kehitysympäristö että valvonta- ja ajonaikainen ympäristö, saa yksinkertaisimmillaan käyttöön naputtelemalla

asennusvelhon aikana toistuvasti Enter-näppäintä, jolloin kaikkiin järjestelmän osiin valitaan perusasetukset.

Suljetaan lopuksi ohjelma ja sen kaikki avoinna olevat ikkunat.

6.2 PLC-ohjelman laatiminen

Laaditaan ohjelma, joka vuorotellen kytkee kolmea eri digitaalista lähtöä vuorotellen, ja pitää niitä päällä 3 sekuntia kerrallaan. Digitaalilähtöjä indikoivat tässä kolme eriväristä lamppua: punainen, keltainen ja vihreä. Ohjelma kiertää viisi kertaa, jonka jälkeen se pysähtyy. Ohjelman kierto alkaa painikkeella Startti ja sen voi keskeyttää painikkeella Stoppi. Startti-painike aloittaa ohjelman kierron jälleen sen keskeytyskohdasta. Ohjelman kierron voi alustaa Nollaus-painikkeesta, jolloin ohjelmalaskuri nollataan. Ohjelman kierron voi jälleen aloittaa Startti-painikkeella, jolloin se alkaa ensimmäisestä lampusta.

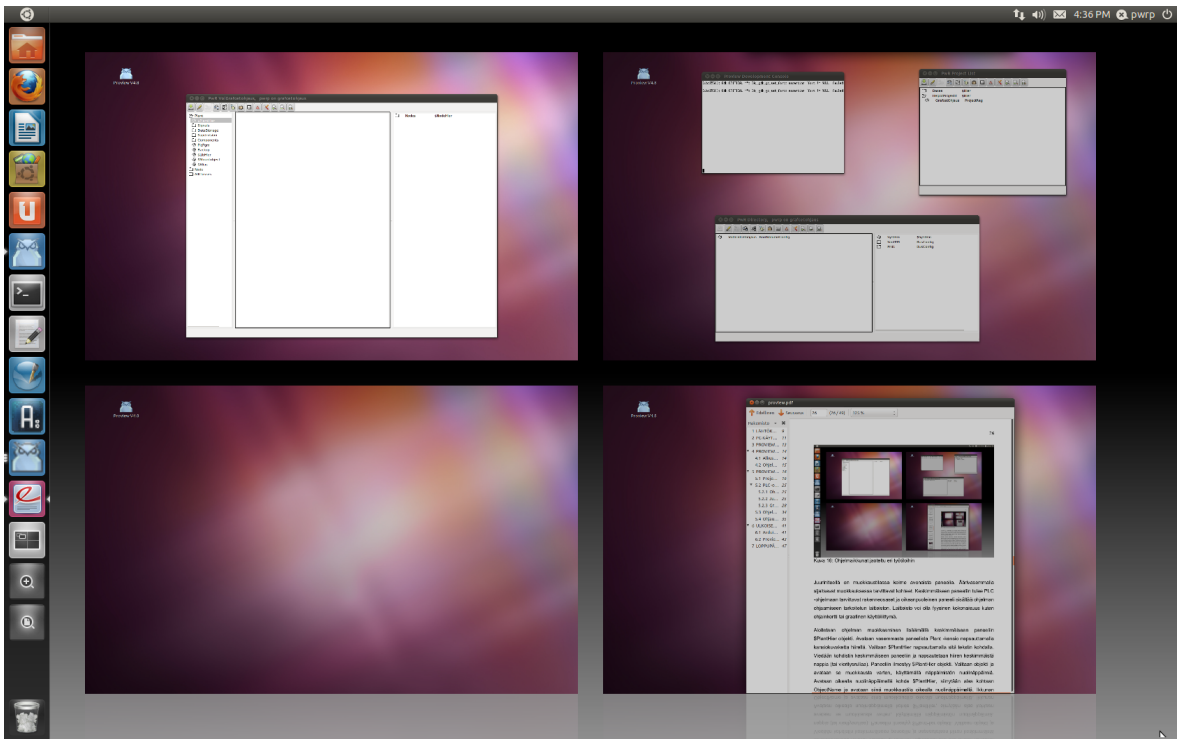
Ohjelmalla on kolme digitaalista tuloa(Di): Startti, Stoppi ja Nollaus.

Ohjelmalla on kolme digitaalista lähtöä(Do): Punainen, Keltainen ja Vihreä.

Ohjelman sisäisenä muuttujana(Dv) on Käynnissä-arvo.

6.2.1 Juuriniteen muotoilu

Avataan Proview-ohjelma. Avataan HelpotProjektit-kansiosta GrafcetOhjaus-projekti napsauttamalla sitä hiiren oikealla napilla ja valitsemalla Open Project, jolloin hakemistonide avautuu. Valitaan hakemistoniteen vasemmasta paneelista kohde VolGrafcetohjaus, napsautetaan sitä hiiren oikealla napilla ja valitaan Open Volume. Esillä on nyt juurinide VolGrafcetohjaus (kuva 15). Ohjelman muut ikkunat kannattaa siirtää toiseen työtilaan juuriniteen muokkauksen ajaksi. Myöhemmin PLC-ohjelman vuorottaisen simuloinnin ja muokkauksen ajaksi ohjelmaikkunoita kannattaa sijoitella eri työtiloihin, jotta työpöytäympäristö pysyy selkeänä (kuva 16).



Kuva 16: Ohjelmaikkunat jaoteltu eri työtiloihin

Aloitetaan työskentely juuriniteellä ja otetaan muokkaustila käyttöön valitsemalla [Ctrl+E].

Juuriniteellä on muokkaustilassa kolme avonaista paneelia. Äärivasemmalla sijaitsevat muokkauksessa tarvittavat kohteet. Keskimmäiseen paneeliin tulee PLC-ohjelmaan tarvittavat rakenneosaset ja oikeanpuoleinen paneeli sisältää ohjelman ohjaamiseen tarkoitetun laitteiston. Laitteisto voi olla fyysinen kokonaisuus, kuten ohjainkortti tai graafinen käyttöliittymä.

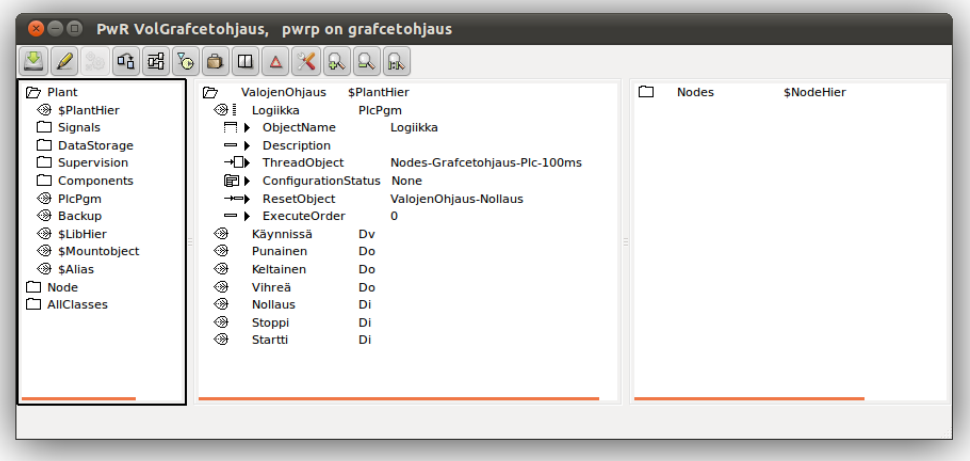
Aloitetaan ohjelman muokkaaminen lisäämällä keskimmäiseen paneeliin $PlantHier-objekti. Avataan vasemmasta paneelista Plant-kansio napsauttamalla kansiokuvaketta hiirellä. Valitaan $PlantHier napsauttamalla sitä tekstin kohdalla. Viedään kohdistin keskimmäiseen paneeliin ja napsautetaan hiiren keskimmäistä nappia (tai vieritysrullaa). Paneeliin ilmestyy $PlantHier-objekti. Valitaan objekti ja avataan se muokkausta varten käyttämällä näppäimistön nuolinäppäimiä. Avataan oikealla nuolinäppäimellä kohde $PlantHier, siirrytään alas kohtaan ObjectName ja avataan siinä muokkaustila oikealla nuolinäppäimellä. Ikkunan alaosaan ilmestyy value-kenttä, johon kirjoitetaan hierarkialle nimi ValojenOhjaus.

Lisätään hierarkian alle signaalit tuloille ja lähdöille. Avataan vasemmasta paneelista Plant-kansion alta Signals-kansio. Valitaan kohde Di (Digital Input), viedään kohdistin keskimmäiseen paneeliin kohteen ValojenOhjaus vasemmalla puolella olevan lehden päälle, ja painetaan hiiren keskinappia. $PlantHier-kohde muuttuu kansioksi, ja juuri lisätty Di hierarkiassa sen lapseksi. Annetaan Di-objektille nimeksi Startti. Lisätään vielä kaksi digitaalituloa Stoppi ja Nollaus. Lisätään hierarkiaan digitaalilähdöt Punainen, Keltainen ja Vihreä. Digitaalilähdöt löytyvät vasemmasta paneelista nimellä Do (Digital output). Lopuksi lisätään digitaaliarvo Dv (Digital value). Digitaaliarvoa tarvitaan logiikkaohjelmassa sisäisesti, ilmaisemaan käynnissä olevaa työkiertoa. Annetaan kohteelle Dv nimeksi Käynnissä.

Lisätään kohde PLC-ohjelmalle. Kohde löytyy vasemmasta paneelista Plant-kansiosta nimellä PlcPgm. Lisätään se lapseksi kohteen ValojenOhjaus alle ja annetaan nimeksi Logiikka.

Annetaan logiikkaohjelmalle Reset-arvo, jonka avulla Grafcet-logiikan työkierron voi tarvittaessa nollata. Muokataan juuri lisätyn kohteen Logiikka, ResetObject-ominaisuutta. Annetaan sille arvo ValojenOhjaus-Nollaus. Kohteen arvo tulee suoraviivaisesti $PlantHier-puusta, jonka alla on lapsena tulo Nollaus. (kuva 17).

Tässä on puussa käytetty vain yhtä $PlantHier-objektia. Hierarkiaobjekteja kannattaa käyttää enemmän isommissa projekteissa, esimerkiksi laittamalla tulot ja lähdöt eri kansioihin. Tallennetaan tila [Ctrl+S] ja poistutaan muokkaustilasta [Ctrl+E].

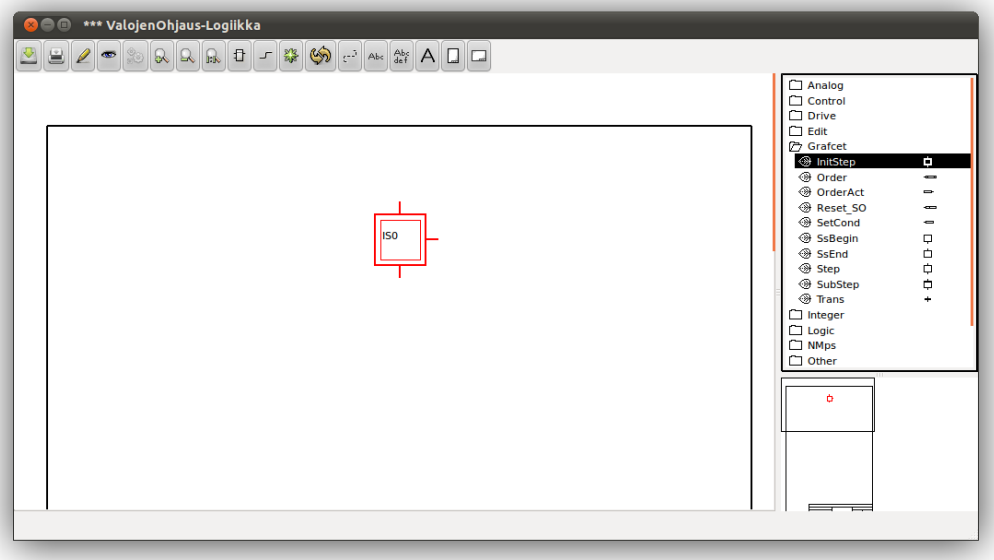


Kuva 17: Logiikkaohjelman loogiset tulot ja lähdöt

6.2.2 Graafinen PLC-ohjelman muotoilu

Aloitetaan logiikkaohjelman muotoilu. Valitaan juuriniteestä kohde Logiikka ja napsautetaan sen päällä hiiren oikealla napilla. Avautuvasta valikosta valitaan Open Program. Valitaan avautuvassa ohjelmaikkunassa muokkaustila [Ctrl+E].

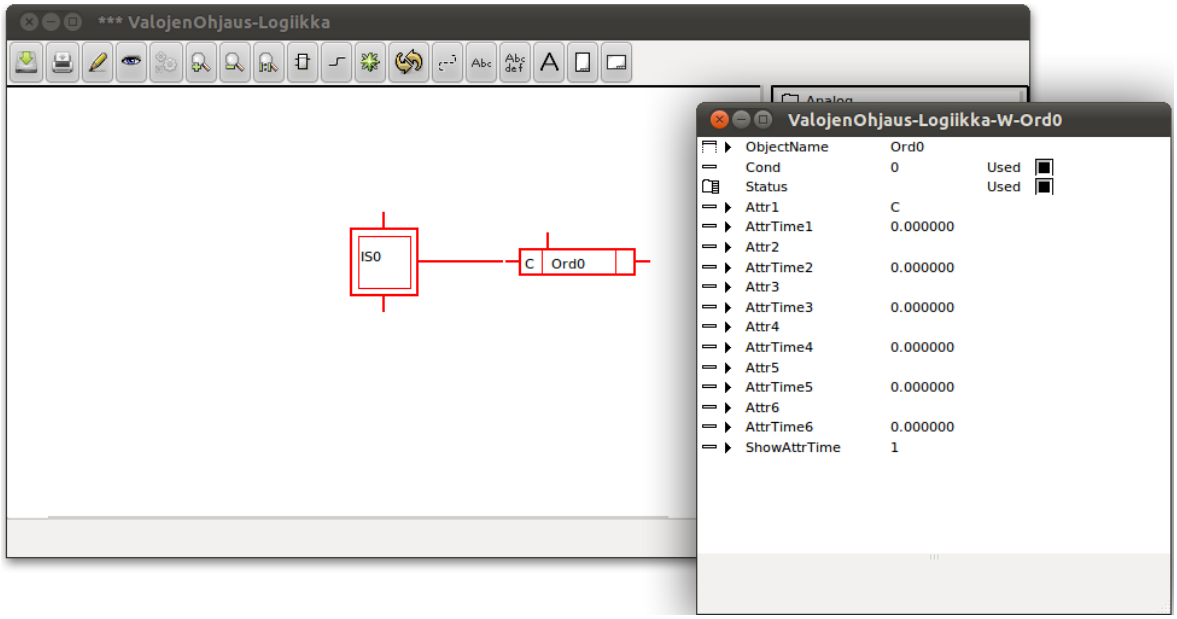
Oikeanpuoleisessa paneelissa sijaitsevat logiikkaosaset ohjelmaa varten. Valitaan työkierron aloituskappale kansiosta Grafcet. Valitaan kohde InitStep, mennään piirtoalueelle ja napsautetaan paikalleen (kuva 18). Hiiren nappia ei pidetä pohjassa piirtoalueelle siirryttäessä. InitStep on ensimmäinen käynnistyvä lohko työkierrossa, ja on heti päällä automaatio-ohjelman käynnistyttyä. Sen, ja kaikki muutkin lohkot työkierrossa, pysäyttää samanaikaisesti ResetObject, joka määriteltiin logiikkaohjelmalle aikaisemmin.



Kuva 18: Työkierron aloituskappale

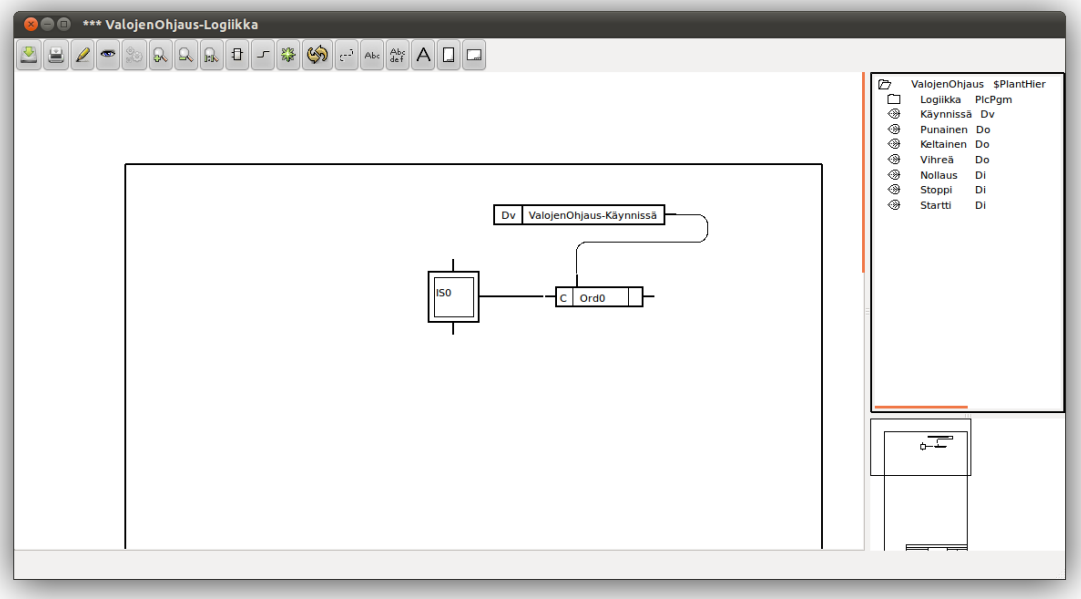
Viedään kohdistin InitStep-kohteen oikeanpuoleisen johdon päälle, tartutaan sitä hiiren keskinapilla, raahataan sitä nappi pohjassa vähän matkaa oikealle ja päästetään irti. Aloituslohkon viereen ilmestyy Ord0-niminen objekti. Napsautetaan sitä hiiren oikealla napilla ja valitaan HelpClass. Avautuvasta ohjeesta saa tietoa Order-luokan ominaisuuksista.

Napsautetaan Ord0-objektia uudelleen oikealla napilla ja valitaan ObjectEditor. Editorissa voi muokata kohteen attribuutteja. Painetaan ensin Cond-kohdan oikealla puolella oleva nappi pohjaan. Napsautetaan sitten ensimmäistä attribuuttia ja valitaan muokkaustila näppäimistön oikealla nuolinäppäimellä (kuva 19). Annetaan arvoksi C ja suljetaan editori. Näin Order-objektille on saatu ehdollinen toimivuus. Ord0 johtaa, kun sen yläpuoliseen johtimeen tuodaan totuusarvo eli 1.



Kuva 19: Order-objektin muokkaus

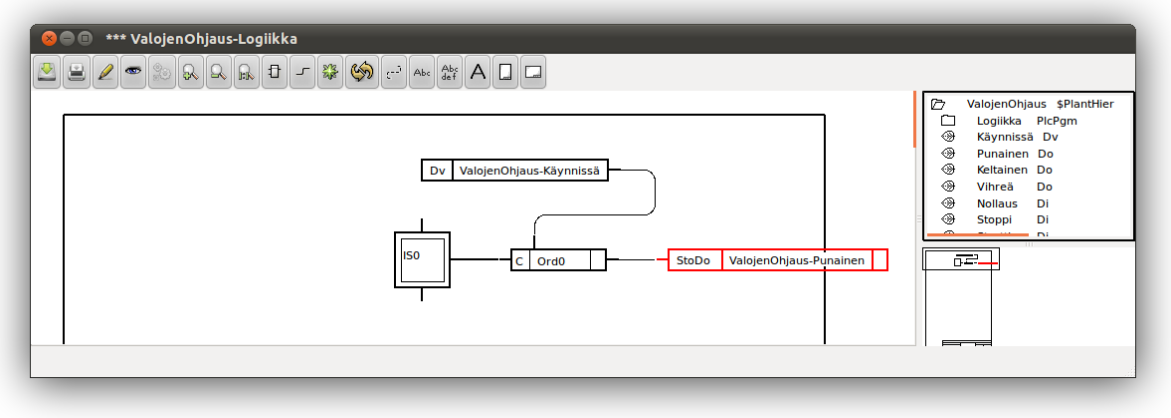
Valmistellaan seuraavaksi Ord0-objektille tulo *käynnissä*. Valitaan oikeanpuoleisesta paneelista digitaaliarvon haku GetDv (Get Digital value), joka löytyy polun Signals-Digital alta. Asetetaan se Ord0-objektin yläpuolelle. Yhdistetään se viemällä kohdistin Dv-objektin johtimen päälle, tarttumalla siihen hiiren keskinapilla ja raahaamalla se Ord0-objektin ylimmäiseen johtimeen. Liitetään vielä Dv-objekti logiikassa käytettyyn arvoon seuraavasti: Viedään hiiren osoitin yläpalkissa olevan tähdenmuotoisen kuvion päälle, jolloin opastusteksti ilmestyy kertoen kuvion toiminnan olevan *Show Plant Hierarchy.* Napsautetaan sitä, jolloin oikeanpuoleiseen paneeliin ilmestyy joukko juuriniteessä aikaisemmin määriteltyjä I/O-arvoja. Valitaan kohta Käynnissä *Dv*. Viedään kohdistin logiikkapiirustuksessa olevan Dv-kohteen päälle ja valitaan hiiren oikealla napilla valikosta Connect. Näin kuvan digitaaliarvo on liitetty logiikan arvoon Käynnissä (kuva 20).



Kuva 20: Käynnissä-arvo liitettynä ehdolliseen Order-objektiin

Haetaan ensimmäinen lähtö ja liitetään se Ord0-objektin oikeanpuoleiseen johtimeen. Vaihdetaan oikeanpuoleiseen paneeliin jälleen komponenttilista napsauttamalla yläpalkin pikanäppäintä Show object palette. Valitaan polusta Signals-Digital, StoDo-objekti. StoDo (Store Digital output) kytkee digitaalista ulosmenoa, mutta ei aseta sitä pysyvästi. Jos sen tilalle valittaisiin SetDo (Set Digital output), digitaalinen ulostulo jäisi päälle SetDo-komponentin sammuttamisen jälkeenkin. Sen voisi sammuttaa toisaalla ohjelman kierrossa komponentilla ResDo (Reset Digital output).

Viedään StoDo-objekti käskyn Ord0 oikealle puolelle. Vaihdetaan oikeanpuoleiseen paneeliin taas I/O-lista tähden muotoisella pikanäppäimellä ja kiinnitetään Punainen Do-lähtö StoDo-objektiin. Yhdistetään vielä Ord0 ja StoDo (kuva 21).



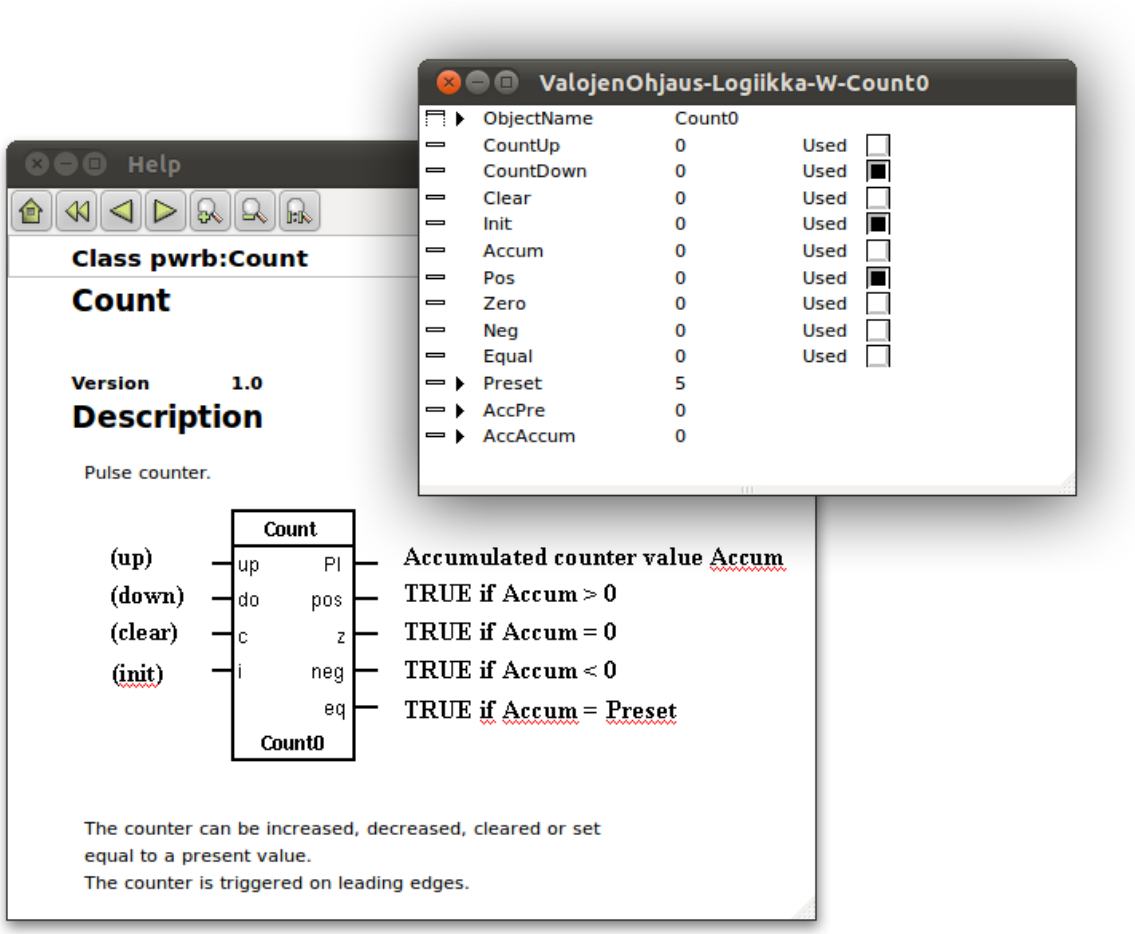
Kuva 21: Lähtö yhdistetty Order-objektiin

Muodostetaan ensimmäiselle Grafcet-lohkolle logiikka seuraavaan lohkoon siirtymistä varten. Jatketaan ensimmäisen lohkon alapuolelta lähtevää johdinta, jolloin sen alapuolelle ilmestyy siirtymäehto (Transition condition) T0. Valitaan komponenttipaneelista Logic-kansiosta Wait-logiikkapala ja kytketään se samaan Ord0-lähtöön StoDo-objektin kanssa. Muokataan Wait-palan arvoja ja asetetaan TimerTime arvoon 3, joka määrittelee tässä 3 sekunnin odotusaikaa, ennen kuin logiikkapala muuttuu johtavaksi. Kytketään lopuksi Wait-objekti siirtymäehdon T0 vasemmanpuoleiseen johtimeen katkoviivalla. Katkoviiva ilmaisee takaisinkytkentäjohtimen, joka on ohjelman oikean suoritusjärjestyksen kannalta välttämätön. Takaisinkytkentäjohdin löytyy yläpalkin Feedback connection -napista.

Digitaalituloa ja -lähtöä ilmaisevien komponenttien tunnukset ovat pitkiä ja mahtuvat huonosti piirrokseen. Muokataan tunnukset lyhyemmiksi. Valitaan lähtö StoDo ValojenOhjaus-Punainen. Siirrytään kohteen muokkaustilaan ja vaihdetaan DoObjectSegments arvosta 2 arvoon 1. Tuloksena komponenttiin jää vain lähdön tyyppi ja nimi. Tehdään sama kohteelle Dv ValojenOhjaus-Käynnissä. Näin on saatu Grafcet-työkierron ensimmäinen lohko valmiiksi.

Jatketaan Grafcet-työkierron laatimista loppuun asti (kuva 23). Valmistellaan työlohkot keltaiselle ja punaiselle lähdölle. Viimeiseen työlohkoon lisätään laskurikomponentti, jonka asetukset nähdään kuvassa 22. Asetukset on helpoin tehdä lukemalla komponentin ohjesivua samanaikaisesti (valitsemalla HelpClass hiiren oikean napin valikosta). Lisätään vielä setti-resetti Käynnissä-arvon

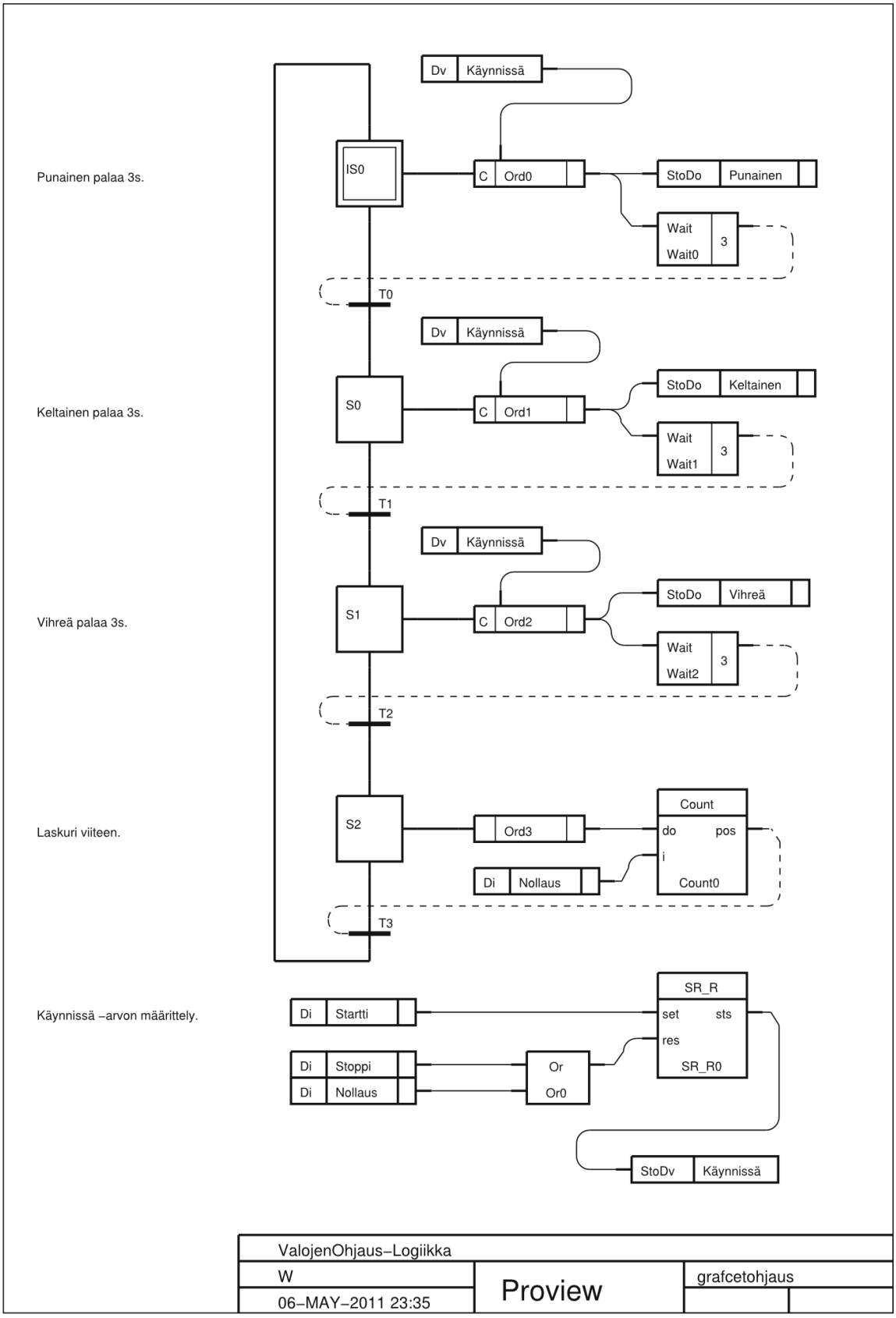
määrittämiseksi. Koko työkierron nollaus määriteltiin jo juuriniteen konfiguroinnissa attribuutissa ResetObject (kuva 17).



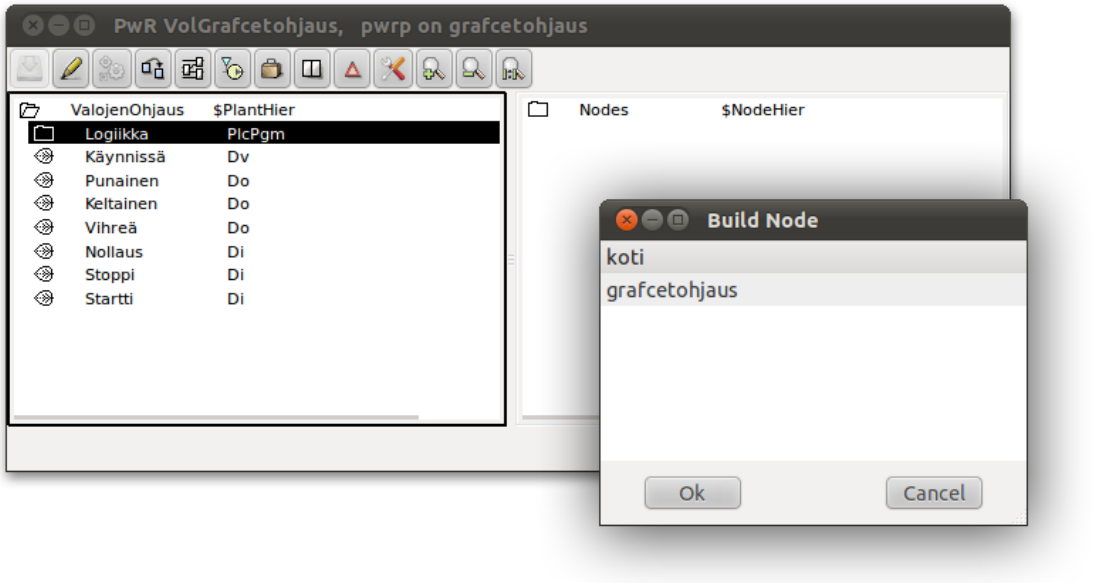
Kuva 22: Laskurin asetusten teko

Ohjelmalohkojen käsittely, ensimmäisestä alkaen, on seuraavanlainen: Vuorossa oleva ohjelmalohko IS0 on päällä. Ehdollinen Ord0-komponentti johtaa, kun sille tuodaan signaali yläpuoliseen johtimeen. Punaisen valon lähtö tulee päälle samanaikaisesti Wait0-komponentin kanssa. Kun Wait0-komponentti kolmen sekunnin kuluttua johtaa, siirtymäehto T0 aukeaa. Ohjelman suoritus siirtyy alaspäin seuraavaan lohkoon, jolloin lohko IS0 sekä sen komponentit sammuvat. Ohjelmakierto jatkuu, kunnes alaspäin laskeva laskuri on vähentynyt nollaan. Laskurin voi alustaa Nollaus-painikkeella.

Tallennetaan työ. Ohjelman oikeellisuuden voi tarkistaa valitsemalla ylävalikosta Build Program. Jos ohjelmassa on virheitä, aukeaa niistä ilmoittava ikkuna.



Kuva 23: Valmis työkierto



Kuva 24: Logiikkaohjelman kääntäminen

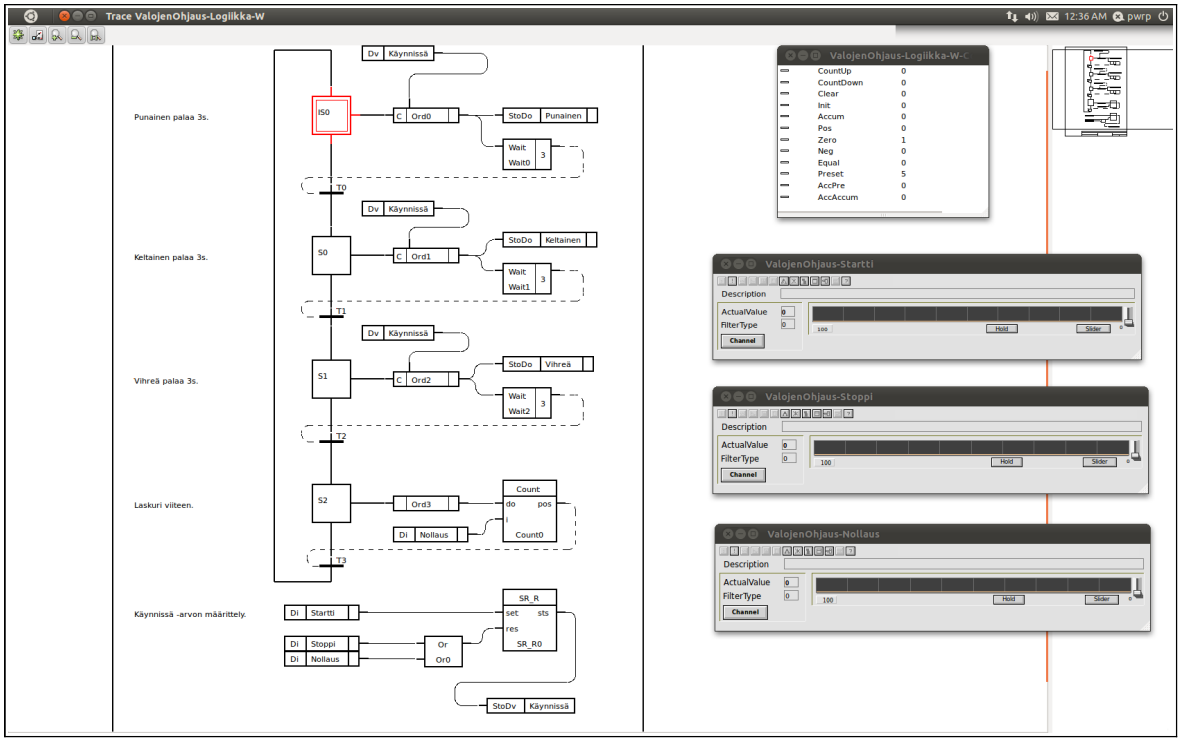
Siirrytään juuriniteeseen VolGrafcetohjaus. Käännetään PLC-ohjelma painamalla napista Build Node. Valintaikkuna, joka sisältää kaksi vaihtoehtoista solmua, aukeaa. Aikaisemmin määriteltiin (kuva 6) tuotantoväylän solmuksi grafcetohjaus, jota ei käynnistetä. Valitaan solmu **koti**.

6.3 Ohjelman ajo simuloimalla

Ajetaan ohjelma ilman graafista käyttöliittymää. Valitaan ruudun ylälaidasta Tools-kohdan alta Runtime Monitor. Käynnistetään simulaatio valitsemalla Start Runtime. Ikkunasta Runtime Monitor löytyy nappi Start Runtime Navigator. Avataan sillä ajonaikainen tarkasteluikkuna Xtt. Valitaan polku Database – ValojenOhjaus - Logiikka. Siirretään kohdistin kohteen Logiikka päälle ja valitaan hiiren oikealla napilla avautuvasta valikosta Open Plc. Avautuvassa ikkunassa näkyy aiemmin laadittu logiikkaohjelma ja työkierron aloituslohko IS0 on päällä.

Valitaan ruudun ylälaidasta Mode-Simulate. Nyt voidaan valita simulaatiossa tarvittaville tuloille omat ohjailukuvaajat. Napsautetaan kohdassa Di Nollaus hiiren oikealla napilla ja avautuvasta valikosta Object Graph. Valitaan avautuneen ikkunan otsikkopalkissa hiiren oikealla napilla avautuvasta valikosta Aina Päällimmäisenä, jotta ikkuna ei häviäisi muiden ikkunoiden alle. Tehdään

sama tuloille Di Startti ja Di Stoppi. Valitaan vielä laskurin Count0 päällä, hiiren oikealla napilla OpenObject, jolloin saadaan näkyville laskurin sisäiset tiedot (kuva 25).



Kuva 25: Simulaatio käynnissä

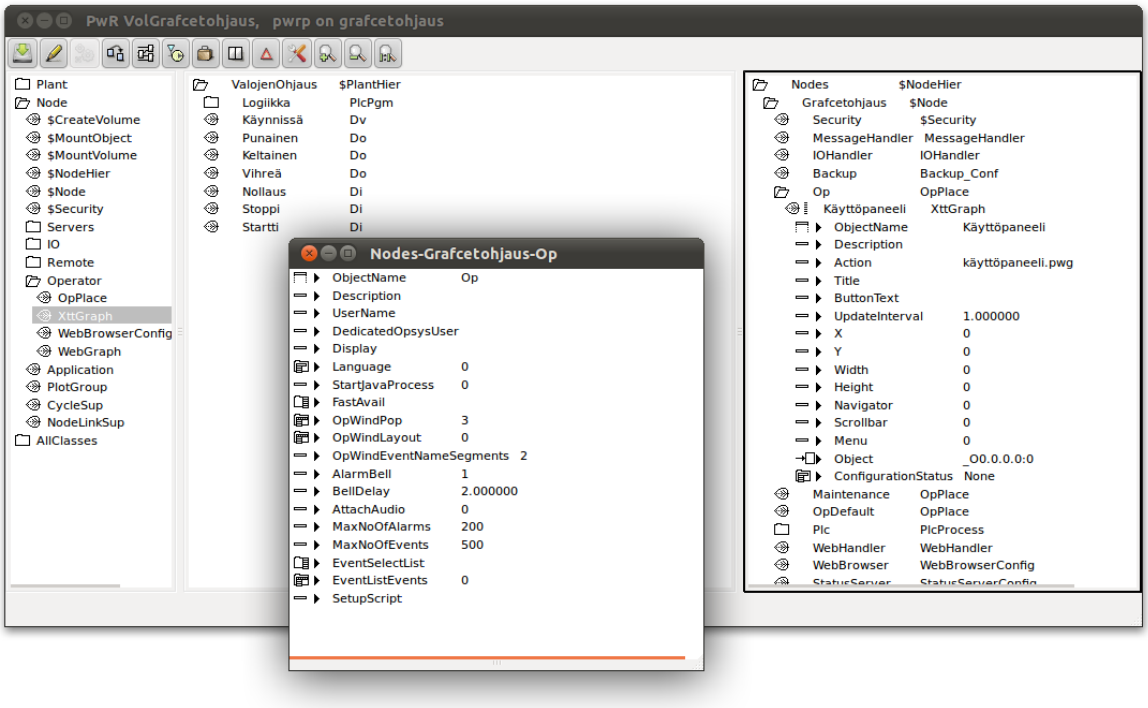
Kokeillaan ohjelmaa nollaamalla ensin laskuri kytkemällä nollaustulon ohjailukuvaajasta liukukytkin käyttöasentoon napsauttamalla Slider-nappia. Nyt voidaan liukukytkimellä valita tulo päälle tai pois. Kokeillaan samaa Startti- ja Stoppi- kuvaajilla. Logiikkaohjelman punaisen värin kierrosta sen eri osissa voidaan havainnoida ohjelman toimintaa.

Simulaatiota suljettaessa paras ratkaisu on valita Runtime Monitor -ikkunasta Reset Runtime. Se sulkee simulaation aikana avatut tarkasteluikkunat, simulaatioikkunan ja ajonaikaisen hallintaikkunan Xtt samalla kertaa.

6.4 Ohjaus graafisella käyttöliittymällä

Proview-ohjelmisto sisältää graafisen käyttöliittymäeditorin, jolla voi tehdä havainnollisia kuvia automaation käyttöä ja valvontaa varten. Aloitetaan

muokkaamalla juurinidettä VolGrafcetohjaus. Avataan editointitila [Ctrl+E]. Valitaan vasemmanpuoleisesta paneelista polun Node-Operator alta, kohde XttGraph. Viedään kohdistin oikeanpuoleiseen paneeliin polulle Nodes-Grafcetohjaus ja asetetaan se lapseksi objektin Op alle, napsauttamalla sen lehteä hiiren keskinapilla. Valitaan XttGaph-objekti ja avataan se näppäimistön oikealla nuolinäppäimellä. Annetaan sille nimeksi Käyttöpaneeli. Jotta käyttöpaneelin voisi myöhemmin avata suoraan operaattori-ikkunasta, lisätään se linkkinä OpPlace-objektiin. Tässä vaiheessa se on helpointa tehdä valitsemalla Käyttöpaneeli-objekti jonka jälkeen napsautetaan Op-objektin päällä hiiren oikeaa nappia. Valitaan avautuneesta valikosta polku Connect - FastAvail-Row 1 - Column 1. Jos myöhemmin on tarvetta muokata mitä tahansa objektia, sen voi tehdä valitsemalla hiiren oikealla näppäimellä objektin, ja valitsemalla avautuvasta valikosta kohdan Open Object. Kuvassa 26 on objekti Op avattuna erilliseen ikkunaan.



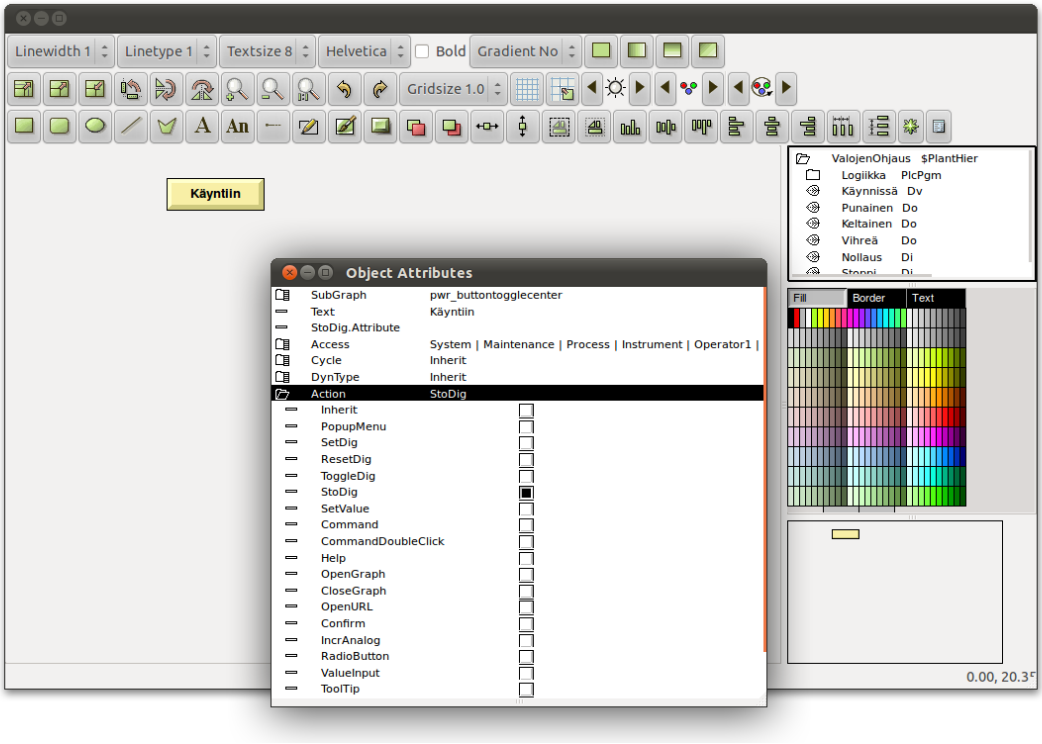
Kuva 26: Objekti avattuna

Tallennetaan [Ctrl+S] ja poistutaan muokkaustilasta [Ctrl+E].

Avataan seuraavaksi graafinen muokkain objektille Käyttöpaneeli. Napsautetaan hiiren oikealla napilla Käyttöpaneeli-objektia ja valitaan avautuvasta valikosta Open Ge.

Luodaan kolme painonappia Start, Stop ja Nollaus. Oikeanpuoleisesta valikosta valitaan Pushbuttons-kansiosta ButtonToggleCenter-kohde. Viedään kohdistin vasemmanpuoleiselle ruudulle ja napsautetaan se siihen kiinni hiiren keskinapilla. Tuplaklikataan painonappia, jolloin päästään muokkaamaan kohteen attribuuttien arvoja. Annetaan Text-kohteelle, arvo Käyntiin.

Nappi toimii perusasetuksilla siten, että kertapainalluksesta se jää pohjaan ja nousee ylös toisella painalluskerralla. Tehdään napista sellainen, että se pysyy pohjassa vain painalluksen ajan, ja palautuu heti ylös kun siitä irrottaa otteen. Muokataan attribuuttia Action poistamalla Inherit ja valitsemalla tilalle StoDig (kuva 27).



Kuva 27: Painonapin muokkausta

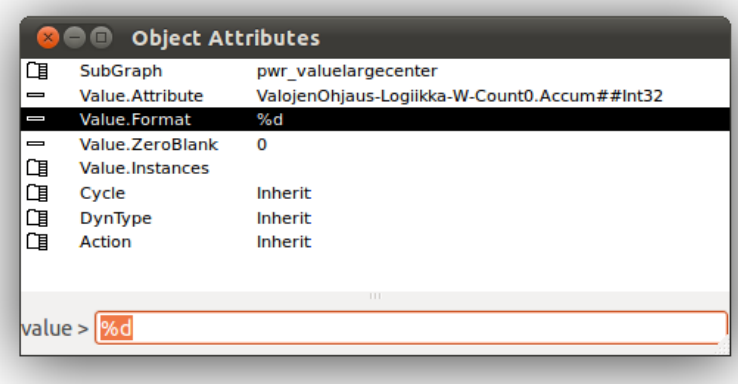
Lisätään kaksi muuta samanlaista painonappia ja annetaan nimeksi Pysäytys ja Nollaus.

Lisätään merkkilamppu. Valitaan oikeasta paneelista Indicators-kansiosta IndRound-merkkivalo. Siirretään se muokkauspöydälle painonappien viereen. Valitaan merkkilampulle toinen väri napsauttamalla sitä, jonka jälkeen kohdistin

viedään oikeanpuoleisen väripaletin päälle ja napsautetaan siellä punaista. Lisätään vielä kaksi lamppua, keltainen ja vihreä.

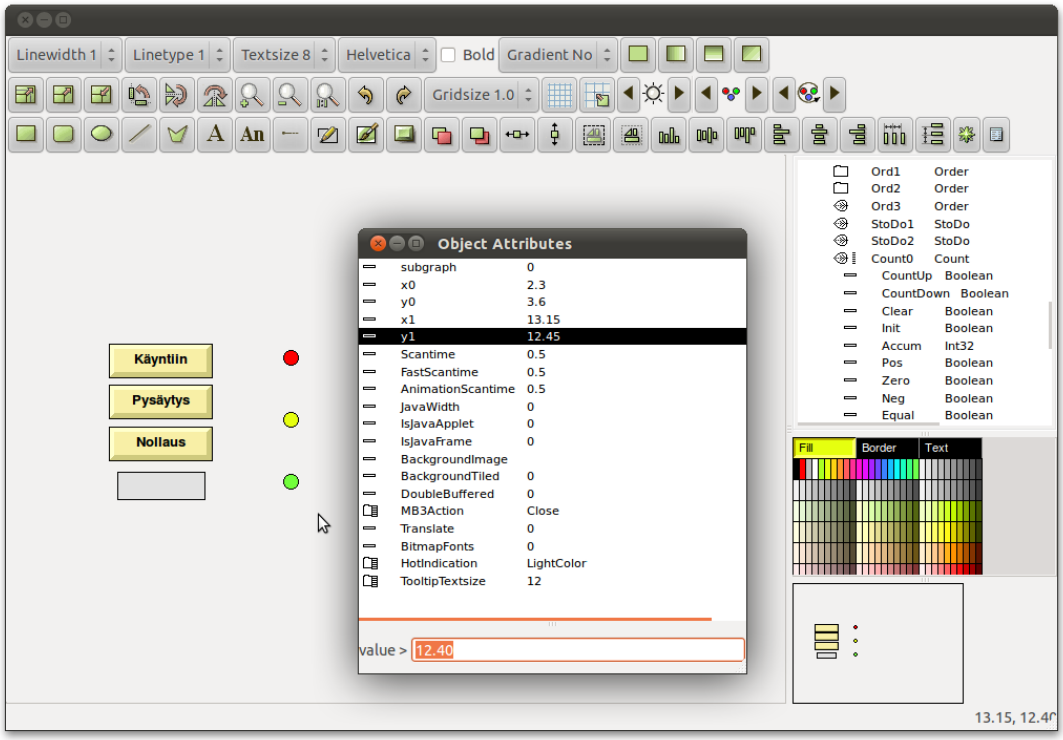
Kytketään napit ja lamput logiikkaohjelman vastaaviin tuloihin ja lähtöihin. Valitaan tähdenmuotoinen pikavalintanappi View Plant Hierarchy. Oikeanpuoleiseen paneeliin ilmestyy logiikkaohjelman kansio, josta löytyy ohjelmassa käytettävät tulot ja lähdöt. Valitaan valikosta Startti ja työpöydältä Käyntiin-nappi. Valitaan sitten ruudun ylävalikosta Functions - Connect. Yhdistämisen pikavalintana voi käyttää yhdistelmää [Ctrl+Q]. Varmistetaan vielä yhdistäminen valitsemalla Käyntiin-napin attribuuttivalikko tuplaklikkaamalla kuvaketta, ja katsomalla StoDig.Attribute-määritteen vieressä olevaa kenttää. Kenttään on ilmestynyt polku logiikkaohjelman tulolle. Yhdistetään loput painikkeet ja lamput.

Lisätään käyttöpaneeliin indikaattori tehdyille työkierroksille. Palataan komponenttivalikkoon View Plant Hierarchy -napin kanssa. Valitaan Values-kansiosta komponentti ValueLargeCenter ja asetetaan työpöydälle painikkeiden jatkoksi. Valitaan taas oikeanpuoleiseen paneeliin logiikkaohjelman lähdöt ja tulot. Haetaan ValojenOhjaus–Logiikka–W -polusta, laskuri Count0. Avataan se oikealla nuolinäppäimellä ja valitaan arvo Accum. Yhdistetään se indikaattorin kanssa näppäimillä [Ctrl+Q]. Muokataan vielä ValueLargeCenter-komponenttia lisäämällä sen Value.Format-kenttään formaattimerkintä %d (kuva 28). Merkintä kertoo komponentille esitettävän näyttöformaatin, joka tässä on kokonaisluku. Formatoidusta tulostuksesta löytyy netistä runsaasti tietoa hakusanalla printf.



Kuva 28: Laskurin numeronäyttöformaatin asettaminen

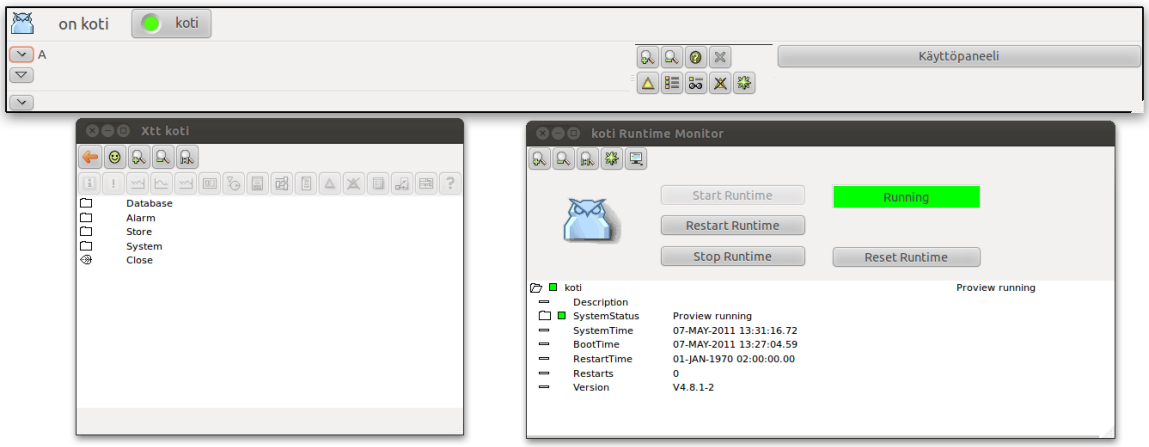
Asetetaan käyttöpaneelin koko. Valitaan ruudun yläreunasta File - Graph attributes. Avautuneeseen ikkunaan, kohdalle x0 ja y0, asetetaan vasen yläraja, sekä kohdille x1 ja y1 oikea alaraja. Liikuttelemalla kohdistinta työpöydällä näkyy oikeassa alareunassa sen sijainti x- ja y-suunnassa. Asetetaan kohdistin painikerivin vasempaan ylänurkkaan ja merkitään sijainti ikkunaan. Sama tehdään oikeaan alareunaan (kuva 29).



Kuva 29: Asetetaan käyttöpaneelin koko

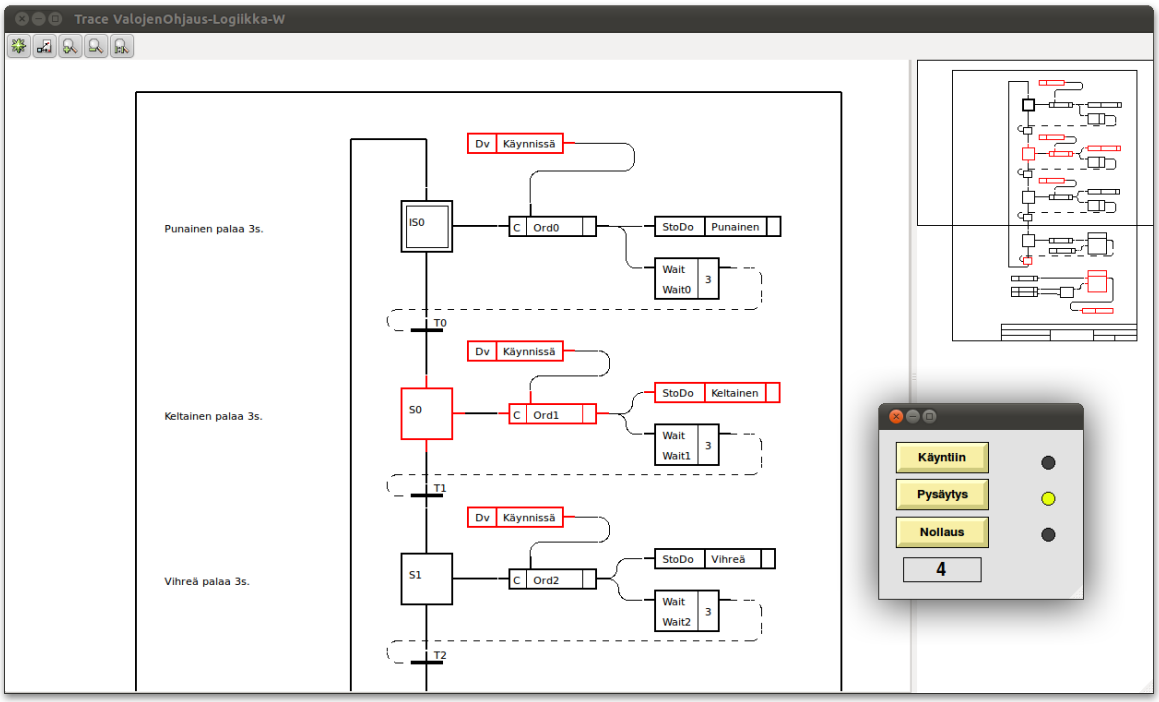
Tallennetaan [Ctrl+S] ja suljetaan graafinen muokkain. Palataan juuriniteen hallintaikkunaan ja käännetään ohjelma (kuva 24). Avataan ajonaikainen ympäristö valitsemalla ruudun ylälaidasta Tools - Runtime Monitor. Käynnistetään ajonaikainen ympäristö valitsemalla Start Runtime. Runtime Monitor -ikkunasta valitaan kuvake Start Operator Environment, jolloin avautuu kysymysikkuna Select Operator Place. Valitaan ikkunasta operaattoripaikaksi Nodes-Grafcetohjaus-Op. Näytön yläreunaan ilmestyy operaattorin valvontaikkuna (kuva 30). Valvontaikkunan paikkaa voi halutessaan muuttaa tarttumalla siihen hiirellä Alt-näppäin pohjaan painettuna.

Avataan edellä luotu käyttöpaneeli napsauttamalla sen käynnistysnappia valvontaikkunan oikeassa reunassa. Asetetaan käyttöpaneeli olemaan jatkuvasti näkyvillä napsauttamalla sen yläreunaa hiiren oikealla napilla, sekä valitsemalla avautuvasta valikosta Aina päällimmäisenä. Avataan logiikkaohjelman tarkasteluikkuna valitsemalla Xtt-ikkunasta polku Database – ValojenOhjaus - Logiikka ja valitsemalla hiiren oikealla napilla Open Plc.



Kuva 30: Ajonaikaiset valvonta-apuvälineet

Logiikkaohjelmaa voi nyt ajaa ja valvoa yksinkertaisen käyttöpaneelin avulla (kuva 31).



Kuva 31: Ohjaus käyttöpaneelin avulla

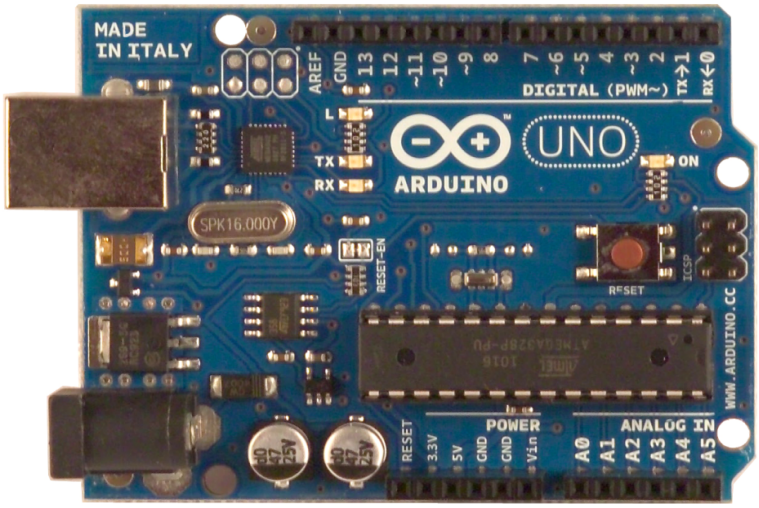
7 ULKOISEN I/O:N LIITTÄMINEN OHJELMISTOON

7.1 Arduino Uno -mikrokontrolleri

Tässä työssä käytetään Arduino Uno -mikrokontrolleria logiikkaohjelman liittämiseksi ulkomaailmaan (kuva 32). Arduino on avoin laitteisto (Arduino 2011), jonka kytkentäkaavio (liite 1) on vapaasti saatavilla ja muokattavissa. Kommunikointi tietokoneen ja mikrokontrollerin välillä tehdään USB-liitäntää käyttäen. Mikrokontrollerin USB-liitännästä huolehtii Atmega8U2-sarjaliitäntämuunnin, jonka ajurit ovat avointa lähdekoodia. Sarjaliitäntämuunninta ohjaa 16 megahertsin kide. Johtuen osittain USB-sarjaliikenteen, ja toisaalta mikrokontrollerin hitaudesta, mahdollinen PLC-ohjelman kiertoaika on alimmillaan 10 ms.

Arduino Uno sisältää 14 digitaalista I/O:a, joista kuusi voidaan määrittää PWM-lähdöksi. Käytettäessä mikrokontrolleria USB-liitännän välityksellä kaksi digitaalista I/O:a käytetään USB-kommunikointiin, jolloin käyttöön jää 14 digitaalista I/O:a. Arduinoon voi liittää kuusi analogista tuloa. (Margolis. 2011. 134.)

Käyttöjännite Arduino Unossa on 5V. Käytettäessä mikrokontrolleria USB-väylän kanssa, se saa virtansa suoraan tietokoneen USB-liitännästä, jolloin ulkopuolinen jännitelähde on tarpeeton. Mikrokontrollerin ohjelmakierrosta huolehtii 16 megahertsin keraaminen resonaattori.



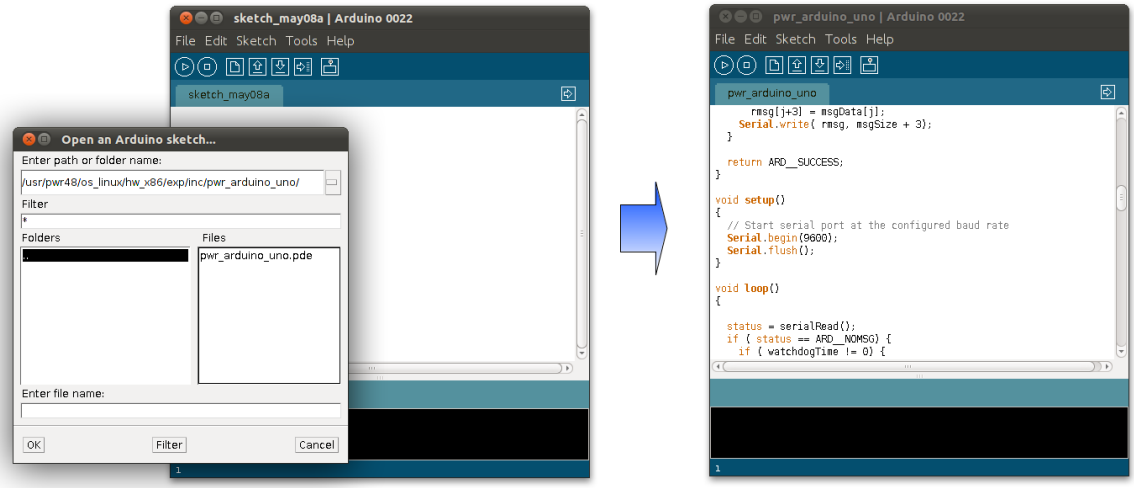
Kuva 32: Arduino Uno -mikrokontrolleri

7.2 Mikrokontrollerin asetukset Proview-ohjelmistoa varten

Jotta Arduino Uno -mikrokontrolleri saadaan kommunikoimaan Proview-ohjelmiston kanssa, on siihen ladattava ohjelmisto, joka kerää I/O-tiedot ja lähettää ne edelleen USB-väylän kautta tietokoneelle. Ohjelmiston lataukseen käytetään Arduino-mikrokontrolleriperheelle muokattua ohjelmistoa. Asennetaan käyttöjärjestelmään Ubuntun sovellusvalikon kautta Arduino-integroitu kehitysympäristö -niminen ohjelma.

Kytketään Arduino USB-kaapelilla kiinni tietokoneeseen. Avataan ohjelma ja tarkastetaan kohdasta Tools-Serial Port, että tietokone on löytänyt mikrokontrollerin, joka näkyy kohteena /dev/ttyACM0. Valitaan File – Open ja haetaan polulta /usr/pwr48/os_linux/hw_x86/exp/inc/ kohde pwr_arduino_uno.pde (kuva 33). Valitaan OK, jolloin saadaan mikrokontrolleriin ladattava ohjelmakoodi kehitysohjelmaan näkyville. Tarkastellaan ohjelmaa ja haetaan ohjelmasta rivi, joka määrittelee sarjaliikennenopeuden tietokoneen ja Arduinon välillä. Rivillä lukee Serial.begin(9600);. Vaihdetaan liikenteen arvoksi 19200 bittiä/sekunti, joten riville tulee Serial.begin(19200);.

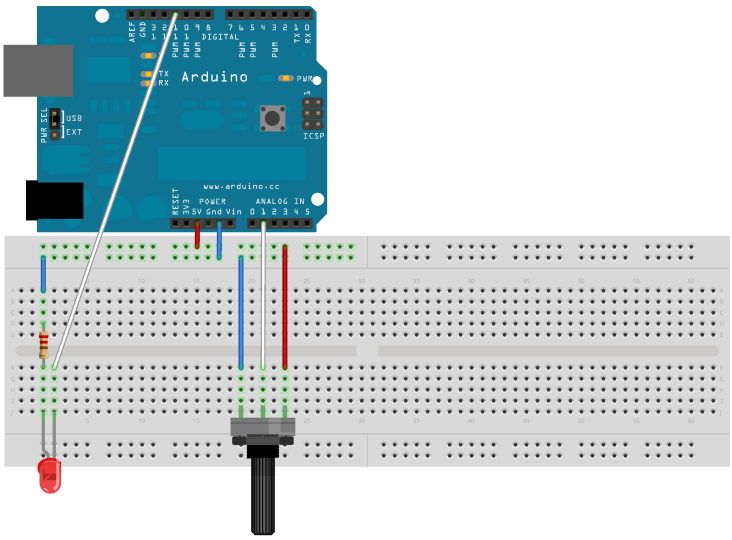
Valitaan toimintavalikosta Verify, jolloin ohjelma kääntää koodin mikrokontrollerille sopivaksi. Ladataan ohjelma Arduinolle valitsemalla Upload.



Kuva 33: Arduino-kehitysympäristö

7.3 Proview-ohjelman konfigurointi analogia-I/O:lle

Tehdään Arduinolle kuvan 34 kaltainen kytkentä. Tarkoituksena on ohjata LED-valoa potentiometrillä. Kytkennässä potentiometriltä tulee analoginen signaali Arduinon analogiatuloon A1. LED-valoa ohjataan pulssinleveysmodulaatio (PWM) -lähdöllä 11. LED-valoa suojaa ylivirralta 220 ohmin vastus.



Kuva 34: I/O:n asettelu koekytkentälevylle (Kuva on tehty Frizing-ohjelmalla)

Avataan Proview ja luodaan uusi projekti kansioon HelpotProjektit. Annetaan projektille nimeksi AnalogiaKokeilu. Avataan projekti ja annetaan asennusvelhon luoda tarvittavat niteet perusasetuksilla, naputtelemalla kysymyksiin Enter. Muokataan juurinidettä VolAnalogiakokeilu luomalla $PlantHier-hierarkia nimellä LampunHimmennys. Lisätään hierarkiaan Ai-analogiatulo, nimellä Himmennin ja Ao-analogialähtö, nimellä Lamppu, sekä PlcPgm-ohjelmaobjekti, nimellä Logiikka.

Muokataan seuraavaksi juuriniteen ”fyysistä” puolta. Kohteena on oikeanpuoleisen paneelin Nodes hierarkia, joka kertoo vasemmanpuoleisen logiikkahierarkian kytkennästä ulkopuolisiin laitteisiin. Luvussa 6.4 tietokoneruudulle logiikan ajonaikana ladattava käyttöikkuna määriteltiin oikeanpuoleiseen paneeliin, joten sitäkin pidetään ulkopuolisena liitäntänä.

Aloitetaan hakemalla vasemmanpuoleisesta paneelista polusta Node–IO–USB-Arduino kohde Arduino_USB ja lisätään se lapseksi kohteen Analogiakokeilu alle.

Annetaan sille nimeksi USB. Valitaan kohdassa Process mikrokontrolleria käsitteleväksi prosessiksi Plc. Kohtaan ThreadObject syötetään Nodes-Analogiakokeilu-Plc-100ms, joka on suora polku hierarkiassa PLC-työkiertoon. Tässä prosessin kierroksi on annettu 100 ms, mutta sitä voi tarvittaessa muokata PlcThread-kohteessa (hiiren oikea nappi-Open Object-ScanTime). Työkiertoa voi myös tarkkailla ja hallita Alarm- ja Halt-objekteilla.

Lisätään Arduino_USB-kohteen lapseksi Arduino_Uno, joka kuvaa tässä yksittäistä Arduino-mikrokontrolleria. Laitteita voi olla enemmänkin, joten annetaan laitteelle nimeksi Uno1. Laitteen osoitteen USB-liityntänä määrittelee Device-kenttä, jolle annetaan arvoksi /dev/ttyACM0. Vaihdetaan kenttään BaudRate arvo 19200.

Arduino-lähdöstä puuttuu vielä määrittelyt I/O-kanaville. Analogiatulo ja -lähtö löytyvät polulta Node-IO-Channels, nimillä ChanAi ja ChanAo. Lisätään ne hierarkiaan Uno1-kortin alle ja annetaan nimet Säätöpotikka ja PunainenLedi.

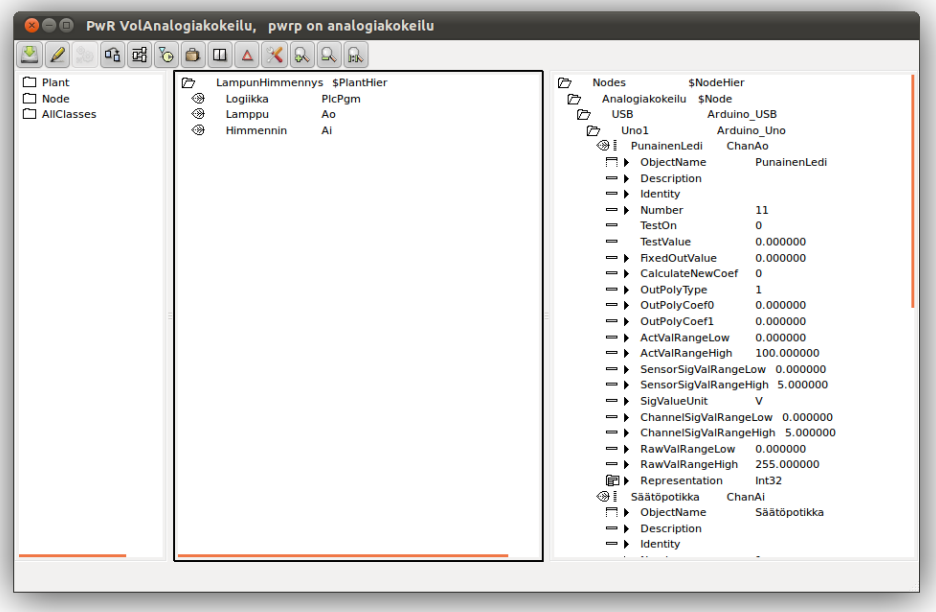
Muokataan ominaisuuksia seuraaviksi:

	Säätöpotikka	PunainenLedi
ActValRangeLow	0	0
ActValRangeHigh	100	100
RawValRangeLow	0	0
RawValRangeHigh	1023	255
ChannelSigValRangeLow	0	0
ChannelSigValRangeHigh	5	5
SensorSigValueRangeLow	0	0
SensorSigValueRangeHigh	5	5

Arduino sisältää 10 bittisen A/D-muuntimen, joten analogiatulot arvotetaan välille 0 - 1023. PWM-lähdöille voidaan antaa arvoja 8 bitin tarkkuudella. Arduinon lähtö- ja tulosignaalijännite arvotetaan välille 0 - 5 V. Lisätään vielä lähdön ja tulon numerot kohdalle Number, jotka koekytkentäpiirustuksen mukaan ovat Säätöpotikka-tulolle 11 ja PunainenLedi-lähdölle 1 (kuva 35).

Jotta PLC-hierarkialle saadaan tieto valitusta I/O:sta, yhdistetään fyysisen maailman, eli Arduino-mikrokontrollerin sekä ohjelmalogiikan tulot ja lähdöt.

Valitaan oikeanpuoleisesta paneelista lähtö PunainenLedi, viedään kohdistin keskimmäiseen paneeliin kohdalle Lamppu ja hiiren oikealla näppäimellä aukeavasta valikosta valitaan Connect Channel. Tehdään sama analogiatulon säätöpotikka ja PLC-hierarkian Himmennin kanssa. Tallennetaan ja poistutaan editointitilasta [Ctrl+S] [Ctrl+E].

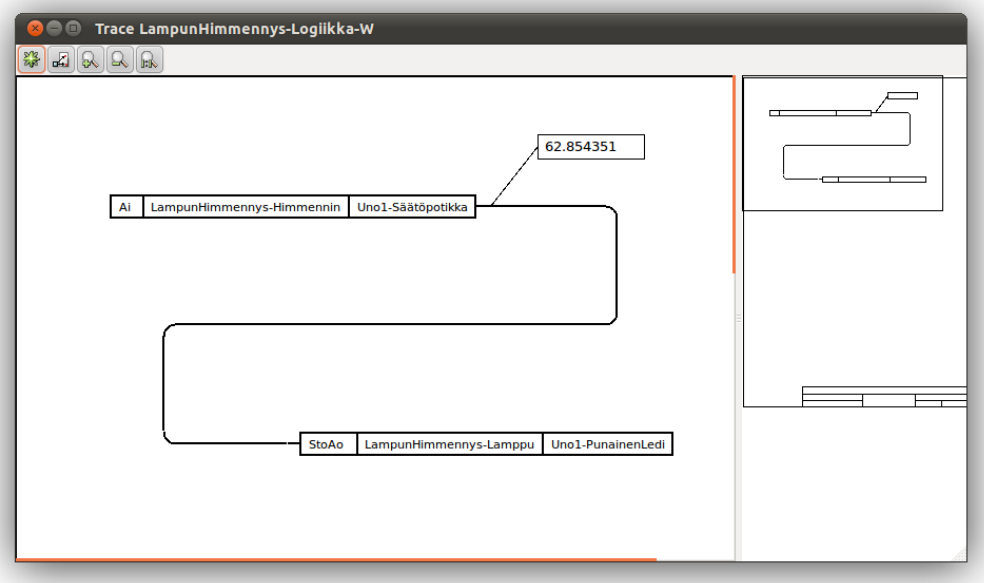


Kuva 35: Analogiatulon ja -lähdön määrittely

Tehdään yksinkertainen logiikkaohjelma himmentimelle. Avataan ohjelmoitavan logiikan muokkausohjelma valitsemalla Logiikka-ohjelman, kohdalla oikealla hiirinäppäimellä Open Program ja avataan editointitila [Ctrl+E]. Lisätään analogiasignaalit tulolle ja lähdölle valitsemalla GetAi ja StoAo polulta Signals-Analog. Yhdistetään signaalit johtimella. Vaihdetaan oikeanpuolimmainen paletti LampunHimmennys-hierarkiaan ja yhdistetään signaalit vastaaviin tuloihin ja lähtöihin. Tallennetaan ja suljetaan editori. Käännetään ohjelma valitsemalla Build Node.

Kokeillaan ohjelmaa käynnistämällä Runtime Monitor. Potentiometriä kääntämällä voi nyt säätää lampun kirkkautta. Avataan Runtime Navigator ja polusta Database-LampunHimmennys avataan logiikka oikealla hiirinäppäimellä Open Plc. Lisätään johtimeen signaalinäyttö seuraavasti: Tartutaan Ai-tuloon hiiren keskinapilla ja liikutetaan hiirtä nappi pohjassa. Päästetään halutussa kohdassa irti hiiren

keskinapista jolloin paikkaan ilmestyy signaalin vahvuutta osoittava numerosarja (kuva 36). Signaali on määritelty nyt välillä 0 - 100. Tuloa ja lähtöä voi skaalata tai kääntää päinvastaiseksi muuttamalla signaalien määrittelyarvoja juuriniteessä.



Kuva 36: Analogiasignaali

8 LOPPUPÄÄTELMÄT

Lähtökohtana olleeseen biomassan pyrolysointilaitteeseen Proview-järjestelmä soveltuu hyvin. Analogisten ja digitaalisten liitäntöjen määrällä ei ole käytännössä ylärajaa, joten liitäntäkortteja (esim. Arduino) voi järjestelmään yhdistää tarvittava määrä. Lisäksi käyttäjän ohjauspainikkeita ei ole välttämätöntä toteuttaa fyysisillä käyttökytkimillä, vaan ne voi toteuttaa kosketusnäyttöön, johon voi myös ohjelmoida prosessin fyysistä mallia esittävän kuvaajan.

Logiikkaohjelman kierron muuttaminen on yksinkertaista. Logiikkaohjelmia voi tarvittaessa laatia varastoon ja kokeilla pyrolyysilaitteiston ohjausta paikan päällä eri asetuksilla. Proview-ohjelmiston kiertonopeus on pyrolyysilaitteelle riittävä, jopa käytettäessä tässä esitettyä ”hidasta” USB-liitäntäistä korttia.

Proview-järjestelmän käyttöönottoa haittasi jonkin verran ohjeistuksen epäselvyys. Suurin osa ohjeista on käännetty ruotsin kielestä, joka näkyy englanninkielisessä ohjeistuksessa. Lisäksi monimutkaisia asioita oli esitetty joissakin tapauksissa turhan ylimalkaisilla esimerkeillä ja esimerkkikuvia ei ole kaikissa tapauksissa selitetty kokonaan. Käyttöönotossa oli suurena apuna Proview-nettifoorumi, josta löytyi vanhoja keskusteluja selailemalla apu kaikkiin tässä työssä kohdattuihin ongelmiin. Nettifoorumi on aktiivinen ja esitettyihin kysymyksiin saa pitävän vastauksen usein jo samana päivänä.

Proview-ohjelmisto ja sen ohjeistus ovat jatkuvan kehitystyön alla. Ohjelmiston ja erilaisten lisäliitäntöjen käyttöönottoon voidaan siis odottaa päivityksiä tulevaisuudessa. Jos erikoisempia liitäntöjä ohjelmoi itse, ne kannattaa palauttaa ohjelmiston aktiivikehittäjien tietoon, jolloin ohjelmisto paranee entisestään.

Proview-ohjelmisto on monipuolinen ja hyvin muokattavissa oleva. Ohjelmiston oliosuuntautuneisuudesta johtuen, sen käyttöönotto vaatii jonkin verran enemmän työtä kuin markkinoilla olevien vastaavien ohjelmistojen. Ohjelman idean sisäistämisen jälkeen järjestelmän käyttö on vastaavasti helpompaa.

Yksi järjestelmän eduista on sen hinta. Ohjelmisto ei maksa mitään ja PC-rauta on nykypäivänä halpaa. Suurin hintakulu Proview-järjestelmässä lienee henkilöstön kouluttaminen sen konfigurointiin ja perusohjelmointiin.

Globaalina näkökohtana Proview ja muut avoimen lähdekoodin ohjelmistot voivat osaltaan auttaa kehitysmaita parantamaan taloudellista kilpailukykyä tarjoamalla tekniikan apuvälineitä, joihin niillä ei muuten olisi varaa. Kalliit PLC-ohjelmistot ja -rauta ovat usein kehittyvien maiden pienyritysten tavoittamattomissa kun taas työvoimakustannukset ovat pienet. Tällaisissa tapauksissa avoimen lähdekoodin ohjelmistot voivat olla väylä teknologiseen kehittymiseen.

LÄHTEET

Arduino. 2011. Mikrokontrollerin kehityssivusto ja keskustelufoorumi. [Verkkosivu]. [Viitattu 9.5.2011]. Saatavana: http://www.arduino.cc/

Bailey, D. & Wright, E. 2003. Practical SCADA for Industry. Great Britain. Newnes. Elsevier.

Bovet, D. & Cesati, M. 2005. Understanding the Linux Kernel. USA. O'Reilly Media.

Garrels, M. 2008. Introduction to Linux - A Hands on Guide. [Verkkojulkaisu]. [Viitattu 1.5.2011]. Saatavana: http://ldp.tonnikala.org/LDP/intro-linux/intro-linux.pdf

GNU. 2010. Avoimen lähdekoodin lisenssi. [Verkkojulkaisu]. [Viitattu 23.5.2011]. Saatavana: http://www.gnu.org/licenses/licenses.html

Margolis, M. 2011. Arduino Cookbook. USA. O'Reilly Media.

Proview. 2011. Kehityssivusto, ohjeistus ja keskustelufoorumi. [Verkkosivu]. [Viitattu 1.5.2011]. Saatavana: http://www.proview.se/

Sjöfors, C. (Nimimerkillä claes). 2010. Proview keskustelufoorumi. [Verkkosivu]. [Viitattu 23.5.2011]. Saatavana: http://www.proview.se/ → Forum → Aihe: How to think as a ProView programmer?

Sjöfors, C. (Nimimerkillä claes). 2011. Proview keskustelufoorumi. [Verkkosivu]. [Viitattu 23.5.2011]. Saatavana: http://www.proview.se/ → Forum → Aihe: IEC61121-3

Ubuntu Suomi. 2011. [Verkkosivu]. [Viitattu 1.5.2011]. Saatavana: http://www.ubuntu-fi.org/

Westman, M. 2007. Hembygge styr järnverket. SSAB i Oxelösund kör Linux och system i öppen källkod. [Verkkojulkaisu]. [Viitattu 23.5.2011]. Saatavana: http://www.nyteknik.se/nyheter/automation/article255976.ece

Yaghmour K., Masters J., Ben-Jossef G. & Gerum P. 2008. Building Embedded Linux Systems. USA. O'Reilly Media.

LIITE 1. Arduino-mikrokontrollerin piirikaavio