Simo Kastell

I/O-modulin ohjelmointi ja sen liittäminen LabVIEW-ohjelmaan

Opinnäytetyö
Syksy 2010
Tekniikan yksikkö
Tietotekniikan koulutusohjelma
Sulautetut järjestelmät

Seinäjoen ammattikorkeakoulu
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

OPINNÄYTETYÖN TIIVISTELMÄ

Koulutusyksikkö: Tekniikan yksikkö
Koulutusohjelma: Tietotekniikan koulutusohjelma
Suuntautumisvaihtoehto: Sulautettujen järjestelmien suuntautumisvaihtoehto

Tekijä: Simo Kastell

Työn nimi: I/O-modulin ohjelmointi ja sen liittäminen LabVIEW-ohjelmaan

Ohjaaja: Heikki Palomäki

Vuosi: 2010 Sivumäärä: 38 Liitteiden lukumäärä: 2

Tämän opinnäytetyön tarkoituksena oli ohjelmoida ATmega32U4-mikrokontrolleriin pohjautuva I/O-moduli ja saada se toimimaan LabVIEW:n kanssa. Toimivuus testattiin toteuttamalla esimerkkisovellus.

I/O-modulin ohjelmointiin käytettiin pohjana Atmelin USB CDC -ohjelmapakettia, jonka avulla I/O-moduli näkyy tietokoneelle virtuaalisarjaporttina. I/O-moduli ohjelmoitiin käyttäen C-ohjelmointikieltä.

I/O-modulia varten oli suunniteltava viestintäprotokolla, jonka avulla mikrokontrolleri ja LabVIEW pystyisivät siirtämään tietoa keskenään. Mikrokontrollerin koodin sujuvan toiminnan kannalta oli järkevää suunnitella tilakone. I/O-modulia varten suunniteltiin ja ohjelmoitiin digitaaliset sisääntulot ja uloslähdöt, analogiset sisääntulot ja uloslähdöt sekä keskeytykset.

LabVIEW:llä tehtiin subVI-aliohjelmat, joiden avulla on yksinkertaista tehdä sovelluksia. Opinnäytetyötä varten tehdyn esimerkkisovelluksen on tarkoitus esitellä I/O-modulin ominaisuuksia. Esimerkkisovellus oli myös tärkeä osa työn testausta.

Työlle asetetut tavoitteet saavutettiin. I/O-moduli saatiin ohjelmoitua ja esimerkkisovellusta käyttäen todettiin sen toimivuus LabVIEW:n kanssa.

Avainsanat: ATmega32U4, LabVIEW, tilakone, I/O-moduli

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology
Degree programme: Information Technology
Specialisation: Embedded Systems

Author: Simo Kastell

Title of the thesis: Programming of the I/O-module and its integration with the LabVIEW

Supervisor: Heikki Palomäki

Year: 2010 Number of pages: 38 Number of appendices: 2

The purpose of this thesis was to program an I/O module based on the ATmega32U4 microchip and to make it work with the LabVIEW program. An example program was made for testing purposes.

Atmel USB CDC software was used as base on which the programming of the I/O module was built on. The USB CDC software makes the I/O module show up as a virtual serial port when connected to a PC. C programming language was used to program the I/O module.

A communications protocol needed to be designed for the I/O module. The protocol allows the data exchange between the microcontroller and LabVIEW. It was also necessary to design a finite-state machine to ensure the smooth operation of the microcontroller code. Digital inputs and outputs, analog inputs and outputs as well as the needed interrupts were developed and programmed for the I/O module.

LabVIEW was used to create subVI subroutines, which make it simple to develop applications. The intention of the example program presented in the thesis is to demonstrate the features of the I/O module. The example program also had an important role in the testing phase of the thesis.

The objectives of this work were accomplished. The I/O module was programmed successfully and its correct functionality was determined using the example program in LabVIEW.

Keywords: ATmega32U4, LabVIEW, finite-state machine, I/O module

SISÄLLYS

KÄYTETYT TERMIT JA LYHENTEET

ADC	Analog-to-digital converter, AD-muunnin.
CDC	Communication device class, laitetason määritelmä viestintälaitteiden tunnistusta varten.
I/O	Input/output, sisääntulo/uloslähtö.
ISR	Interrupt service routine, keskeytyspalvelurutiini.
PWM	Pulse-width modulation, pulssinleveysmodulaatio.
RS-232	Recommended standard 232, standardi sarjaliikenteen siirtoa varten.
SAR-muunnin	Successive Approximation Register, Peräkkäisaproksimaatioon perustuva AD-muunnin.
subVI	LabVIEW-ohjelmassa käytettävä aliohjelma.
UART	Universal asynchronous receiver/transmitter, sarjaliikennepiiri.
VI	Virtual instrument, virtuaali-instrumentti on LabVIEW:llä ohjelmoitu ohjelma.
VISA	Virtual instrument software architecture, LabVIEW:ssä käytettävä ohjelmointirajapinta erilaisten väylien ohjaukseen.

(Bishop 2007.)

KUVIO- JA TAULUKKOLUETTELO

1 JOHDANTO

1.1 Työn tausta

I/O-moduli on laite, johon voidaan syöttää tai josta voidaan saada ulos tietoa. I/O-moduli on tarkoitettu välikappaleeksi ohjaamaan toisia laitteita tai saamaan niiden tilasta tietoa. I/O-laitteet ovat hyvin tärkeässä asemassa automaatiossa ja sulautetuissa järjestelmissä. I/O-modulin käyttökohteita voisivat olla esimerkiksi tiedonkeruu, ohjaus, prosessivalvomo, opetus ja testaus.

Työn aihe ja I/O-moduli on saatu Seinäjoen ammattikorkeakoululta. Työssä käytettävä I/O-moduli pohjautuu ATmega32U4-mikrokontrolleriin. I/O-moduli on USB-liitännällä varustettu.

Aihe on varsin mielenkiintoinen, eivätkä I/O-modulin käyttömahdollisuudet rajoitu ainoastaan LabVIEW:n pariin. Työ tullaan toteuttamaan menetelmillä, jotka mahdollistavat suuren joustavuuden I/O-modulin käyttöympäristölle. Tässä opinnäytetyössä tullaan kuitenkin keskittymään I/O-modulin käyttöönottamiseen LabVIEW-ympäristössä.

Työn tekijän henkilökohtaisena motiivina on oppia kunnolla ATmega32U4:n käyttö ja sen kehitysmahdollisuudet. Työssä käytetään mikrokontrollerin ominaisuuksia varsin monipuolisesti, joten työ on mainio tapa tutustua ja oppia käyttämään AVR-ympäristöä.

1.2 Työn tavoite

I/O-moduli vaatii toimiakseen ensin ohjelmoinnin. Tämän opinnäytetyön tavoitteena on ohjelmoida toimiva I/O-moduli ATmega32U4-mikrokontrolleriin pohjautuvas-

ta piirikortista siten, että se toimii LabVIEW-ympäristössä. LabVIEW:iin ohjelmoidaan subVI-aliohjelmat, joiden avulla on yksinkertaista tehdä sovelluksia I/O-modulille. Lopuksi tehdään vielä esimerkkisovellus LabVIEW-ohjelmalla, jonka avulla I/O-modulin toimintaa voidaan esitellä.

Tavoitteena on tehdä I/O-modulista mahdollisimman helppokäyttöinen sovellusten kehittämistä varten. Tämä tarkoittaa sitä että I/O-modulia ohjatakseen tarvitsee vain käyttää valmiita subVI-kuvakkeita. Tietysti on osattava lisäksi hieman LabVIEW-ohjelmointia.

1.3 Työn rakenne

Sulautettujen järjestelmien ohjelmoinnissa on ensiarvoisen tärkeää tuntea ohjelmoitava laitteisto. Näin ollen ensimmäiseksi työssä tutkitaan niitä ATmega32U4:n ja I/O-modulin ominaisuuksia, jotka ovat merkityksellisiä työn toteutuksen kannalta.

Toiseksi esitellään työssä käytetty ohjelmointiympäristö. Tässä osiossa käsitellään työssä käytettyjä ohjelmistoja ja ohjelmia.

Työn eri vaiheiden suunnitelua ja toteutusta tarkastellaan seuraavissa kahdessa osiossa, joissa käydään läpi myös lähestymistapoja, joita työn eri vaiheet vaativat.

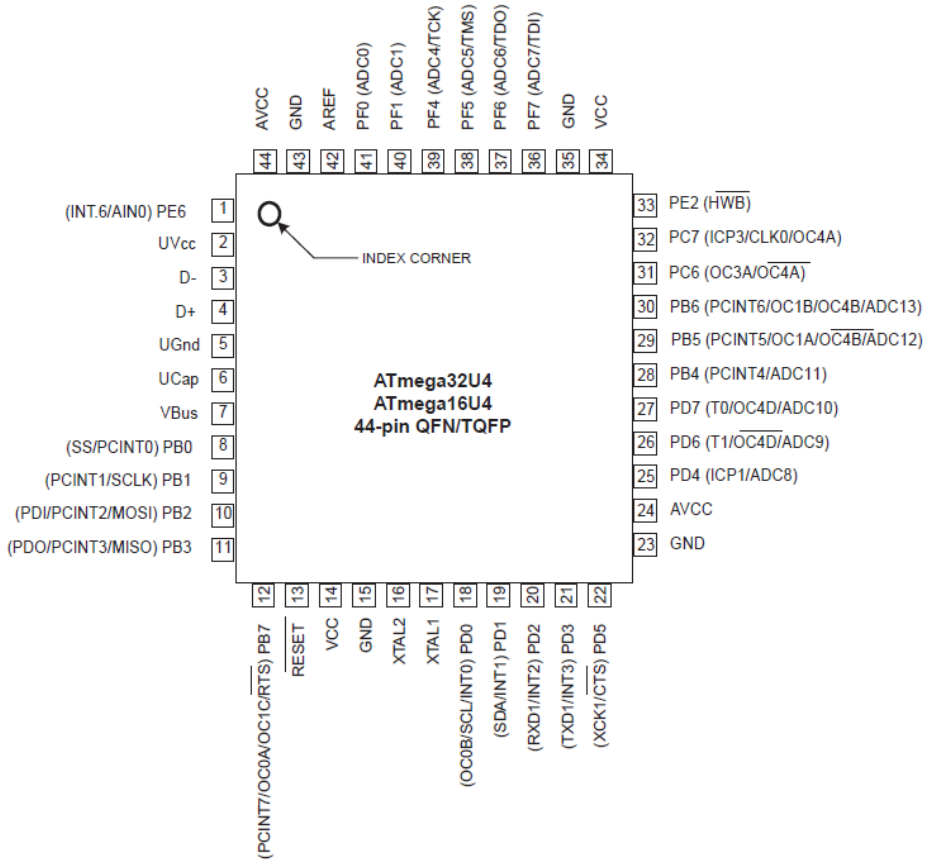
Lopuksi tutustutaan työssä käytettyihin testausmenetelmiin ja esitellään vielä työn tulokset ja arviointi.

2 ATMEGA32U4-MIKROKONTROLLERI

2.1 Yleistä

Työssä käytetty ATmega32U4-mikrokontrolleri on Atmelin AVR-mikrokontrolleriperheeseen kuuluva 8-bittinen mikrokontrolleri. Mikrokontrollerin mahdollinen käyttöjännitealue on 2,7 - 5,5 V. Työssä käytettävän I/O-modulin käyttöjännite on USB-väylän kautta saatava 5 V. ATmega32U4:n nastojen suositeltu maksimi virransyöttö ja vastaanottokestoisuus on 20 mA. (Atmel Corporation 2009, 381.)

Kuvasta 1 näkee mikrokontrollerin kaikki 44 nastaa, joista 26 on ohjelmoitavia I/O-nastoja. Ohjelmoitavat nastat on jaettu portteihin B, C, D, E ja F. Osa nastoista sisältää myös erikoisominaisuuksia kuten PWM:n, ADC:n ja ulkoiset keskeytykset.



Kuva 1. ATmega32U4-mikrokontrollerin nastajärjestys. (Atmel Corporation 2009, 3.)

2.2 Ominaisuudet

Työn toteutuksen kannalta oli tärkeää tuntea miten mikrokontrollerin eri osat toimii. Mikrokontrollerin tärkeimmät ominaisuudet työn kannalta olivat I/O-liitännät, ajastin/laskuri, PWM, AD-muunnin ja keskeytykset.

2.2.1 I/O-liitännät

Mikrokontrollerin porttien I/O-nastat ovat kaksisuuntaisia ja ylösvetovastuksilla varustettuja (Atmel Corporation 2009, 64). Mikrokontrollerin jokainen portti sisältää kolme I/O-rekisteriä, joilla porttien käyttäytymistä voi muuttaa. Rekisterien avulla voidaan muuttaa nastojen tiloja joko sisääntuloiksi tai uloslähdöiksi. (Gadre 2000, 45.)

Portin suuntarekisterillä (DDR) asetetaan portin nastojen suunnat. Nastan bitti asetetaan tilaan 0, jos nastasta halutaan sisääntulo. Vastaavasti nastan bitti asetetaan tilaan 1, jos se halutaan uloslähdöksi. (Atmel Corporation 2009, 65-66.)

Portin tietorekisterillä (PORT) voidaan aktivoida sisäinen ylösvetovastus. Ylösvetovastusta käytetään muuttamaan portin nastan tilaa. (Atmel Corporation 2009, 65-66.)

Portin tulorekisterin (PIN) lukemalla voidaan tarkastaa portin nastan tilan. Tämä rekisteri on riippumaton DDR:n tilasta. (Atmel Corporation 2009, 66.)

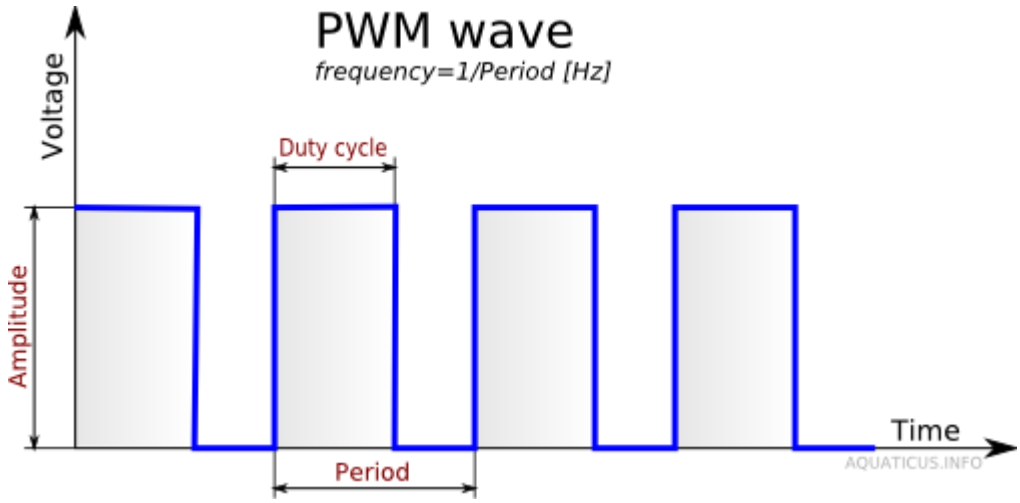
2.2.2 Ajastin/laskuri ja PWM

Ajastin/laskurit ovat todennäköisesti yleisimmin käytetty ominaisuus mikrokontrollereissa. Ne ovat hyvin monikäyttöisiä ja niiden avulla voi esimerkiksi mitata aikavälejä, säätää pulssinleveyttä, mitata taajuuksia tai välittää uloslähtösignaaleja. Teknisesti ajastin/laskuri on vain ylöspäin laskeva binäärilaskuri. Laskuri laskee

valinnan mukaan joko mikrokontrollerin kellosta tai ulkoisesta lähteestä tulevia pulsseja. (Barnett, Cox & O'Cull 2007, 109-110.)

PWM eli pulssinleveysmodulaatio on yksi keino toteuttaa tiedon muunnos digitaalisesta analogiseksi. PWM toimii siten, että kanttiaallon pulssisuhdetta muuttamalla saadaan aikaiseksi vaihtelevaa tasajännitettä. (Barnett ym. 2007, 123.)

Kuva 2 selventää miten pulssinleveysmodulaatio toimii. Kun pulssisuhdetta muunnetaan mikrokontrollerilla, niin uloslähtöjännitteen kanttiaallon leveys muuttuu. Kun jännite on korkealla pitemmän aikaa, saadaan mikrokontrollerista ulos suurempaa jännitettä. Vastaavasti jos pulssisuhde on pieni, niin uloslähtöjännite on pienempi.



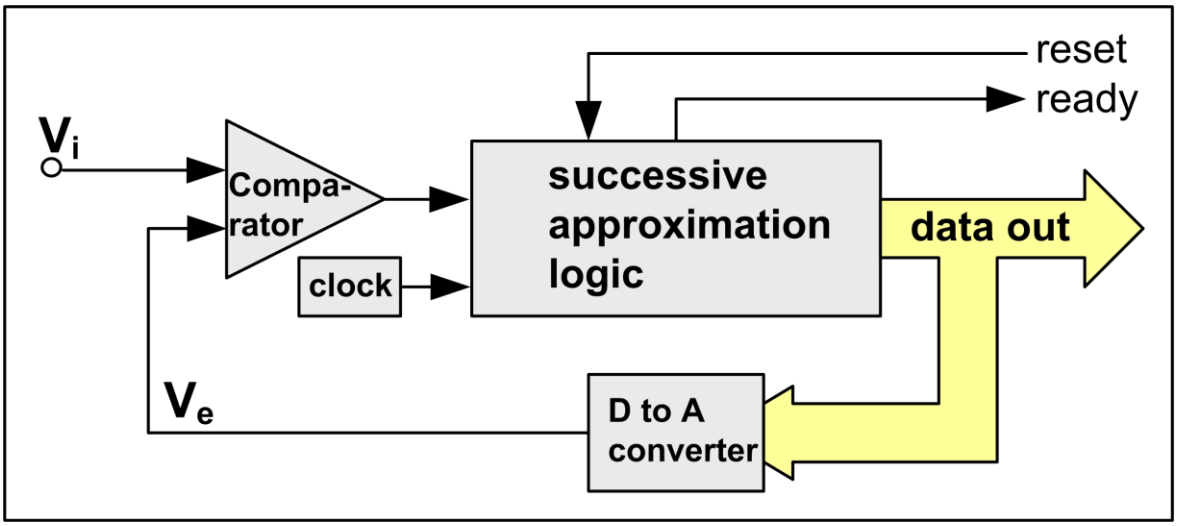
Kuva 2. PWM-kanttiaallon toiminta. (Aquaticus PWM Guide 2010.)

2.2.3 AD-muunnin

AD-muunnin on laite, joka muuntaa analogisen sisääntulojännitteen digitaaliseksi numeroarvoksi (Barnett ym. 2005, 141). ATmega32U4:n AD-muunnin on 10-bittinen. Tämä merkitsee sitä, että muunnettava analoginen jännite voidaan esittää 1024 numeron tarkkuudella.

Erityyppisiä AD-muuntimia on useita, mutta tässä työssä keskitytään peräkkäisaproksimaatioon (SAR) perustuvaan AD-muuntimeen. Peräkkäisaproksimaatioon perustuva AD-muunnos on tekniikka, jolla ATmega32U4:n AD-muunnin toimii. (Vahtera 2008, 2.)

SAR-muuntimen toiminta perustuu siihen, että muunnettava jännite V_i viedään ensin vertailijaan, jonka kautta se siirretään muuntimeen. Muunnin muuttaa analogisen jännitteen digitaaliseksi tiedoksi. Digitaalinen tieto käännetään uudelleen analogiseksi jännitteeksi V_e ja se viedään vertailijaan. Vertailijassa verrataan jännitteitä V_i ja V_e, kunnes ne saavat saman arvon ja muunnos voidaan lopettaa. Kuva 3 selventää SAR-muuntimen toimintaperiaatetta. (Vahtera 2008, 2.)



Kuva 3. SAR-muuntimen toimintaperiaate. (Vahtera 2008.)

2.2.4 Keskeytykset

Keskeytykset ovat pääasiallisesti laitteistolta tulevia kutsuja. Keskeytys toimii siten, että keskeytys katkaisee normaalin ohjelman suorituksen ja siirtyy tämän jälkeen keskeytyspalvelurutiiniin (ISR). ISR:ssä tapahtuu ohjelmoitu toiminto, kun keskeytyskutsu on tullut. Keskeytykset ovat hyödyllisiä tilanteissa, joissa prosessorin täytyy vastata heti. Keskeytysten käyttö myös nopeuttaa laitteiston toimintaa tilanteissa, joissa muuten joutuisi kiertokyselyn avulla tutkimaan tapahtuuko mitään. (Barnett ym. 2007, 97.)

Seuraavassa listassa on selvennetty miten keskeytys toimii vaihe vaiheelta:

1. Oheislaite keskeyttää mikrokontrollerin ohjelman.

2. Nykyinen käsky mikrokontrollerissa suoritetaan loppuun.

3. Seuraavan käskyn osoite tallennetaan pinomuistiin.

4. ISR:n osoite ladataan ohjelmalaskuriin.

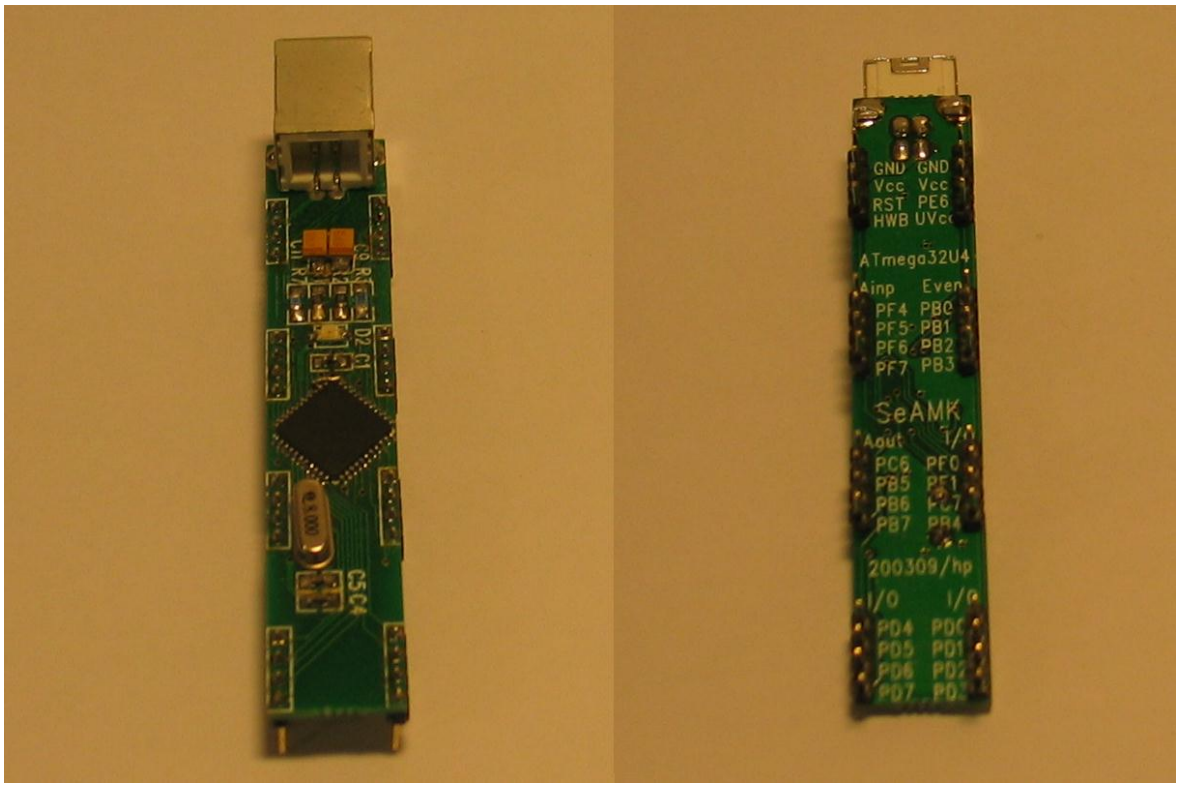
5. Mikrokontrolleri suorittaa ISR-koodin.

6. Mikrokontrolleri lataa ohjelmalaskuriin pinoon tallenetun arvon ja normaalin ohjelman suoritus voi jatkua.

(Gadre 2000, 53.)

2.3 I/O-moduli

I/O-modulin piirikortti saatiin valmiina Seinäjoen ammattikorkeakoulun puolesta. I/O-modulin mikrokontrolleri toimii 8 Mhz:n kellotaajuudella, koska piirikortilla on 8 Mhz:n kide ja mikrokontollerin esijakajaksi on säädetty 1. I/O-moduli on USB-väyläinen, mikä vaikuttaa sen helppokäyttöisyyteen. Ohjelmointia varten I/O-modulissa on käytettävissä 25 I/O-nastaa. Työssä käytetään kuitenkin vain 24 I/O-nastaa, sillä ylimääräinen nasta rikkoisi kortin harmonian parittomuudellaan. I/O-moduli on kuvattu ylä- ja alapuolelta kuvassa 4.



Kuva 4. I/O-moduli.

Nastojen fyysinen sijainti I/O-modulissa on merkittävää, sillä eri nastoilla on erilaisia erikoisominaisuuksia. I/O-modulissa nastat onkin järjestetty nimettyihin ryhmiin niiden käyttötarkoitukset huomioiden seuraavan listan mukaisesti:

- 12 nastaa digitaaliseen I/O-käyttöön
- 4 nastaa analogisille uloslähdöille
- 4 nastaa analogisille sisääntuloille
- 4 nastaa keskeytyksille.

3 OHJELMOINTIYMPÄRISTÖ

3.1 Ohjelmistojen valinta

Tässä opinnäytetyössä käytettävien ohjelmistojen valinta perustuu valtaosin siihen, että Seinäjoen ammattikorkeakoulun opinnoissa on käytetty samoja ohjelmistoja. Suurin osa ohjelmistoista on ilmaisia. LabVIEW on kuitenkin kaupallinen ohjelmisto ja näin ollen maksullinen.

3.2 WinAVR

WinAVR on avoimeen lähdekoodiin perustuva työkalupaketti Atmelin AVR-mikrokontrollerien ohjelmointiin. WinAVR:n mukana tulee työssä käytettävä GNU GCC -kääntäjä. (Vahtera 2003, 51.) Työssä käytetään WinAVR:n vanhempaa versiota 20090313, koska uusimmalla versiolla oli ongelmia kääntää tämän työn koodin joitakin osia.

3.3 AVRstudio

Mikrokontrollerin ohelmointityökaluna työssä käytetään Atmelin AVRstudiota, joka on integroitu ohjelmien kehitysympäristö. Kehitysympäristönä se sisältää tarvittavat työkalut projektinhallintaan, koodin kirjoittamiseen ja debuggaukseen. AVRstudio sisältää assembler-kääntäjän, mutta myös tuen ulkoiselle C-kääntäjälle. (Vahtera 2003, 75-76.)

AVRstudiota käytetään tässä työssä pääasiallisesti ohjelmointi- ja debuggaustyökaluna. Ohjelmointikielenä käytetään C-kieltä. Koodi käännetään HEX-tiedostoksi AVRstudion kautta käyttäen WinAVR:n GCC-kääntäjää.

3.4 Flip

HEX-tiedosto on siirrettävä mikrokontrollerin flash-muistiin. Atmelin Flip eli FLexible In-system Programmer on tarkoitettu ohjelmien siirtämiseksi mikrokontrolleriin. Ohjelmien siirtämiseen voi käyttää RS-232-, USB- tai CAN-väylää. (Atmel Corporation 2010, 1.) Tässä työssä käytetään USB-väylää, sillä se on ainut mahdollisuus, jonka työssä käytettävä I/O-moduli tarjoaa.

3.5 LabVIEW

National Instrumentsin LabVIEW-ohjelmassa käytetään graafista ohjelmointikieltä, jossa ohjelmointiin käytetään pääasiassa kuvakkeita tekstin sijaan. LabVIEW:n avulla tehdään ohjelmia, joita kutsutaan virtuaali-instrumenteiksi. (Bishop 2007, 4.) LabVIEW-ohjelmointi muistuttaa hyvin paljon vuokaavion tekemistä, joten ohjelman kulkua on helppo ymmärtää.

LabVIEW-ohjelmointi tapahtuu etupaneeli- ja lohkokaavioikkunoissa. Etupaneelissa rakennetaan ohjelman käyttöliittymä ja lohkokaavioikkunassa ohjelmoidaan johdotukset ja muut ohjelman kannalta merkittävät käskyt, rakenteet ja silmukat.

3.6 USB CDC

USB-väylä on nykyisin valtaosin korvannut RS-232-väylän. Sen seurauksena UART-rajapintaa käyttävien sovellusten täytyy siirtyä käyttämään USB-väylää. USB-väylän käyttöön siirtyminen voi johtaa ei-toivottuihin muutoksiin itse laitteessa, kuin myös tietokoneessa. Atmelin ratkaisu kehitystyön helpottamiseen on USB CDC -virtuaalisarjaportti. (Atmel Corporation 2007, 1.)

USB CDC -virtuaalisarjaportti on laiteohjelmisto mikrokontrolleriin, jonka avulla mikrokontrolleri tunnistuu virtuaalisarjaporttina tietokoneelle. Tämä vaatii CDC-paketin mukana tulevan INF-ajuritiedoston asentamisen tietokoneelle. (Atmel Corporation 2007, 2-3.)

4 SUUNNITTELU

4.1 Yleistä

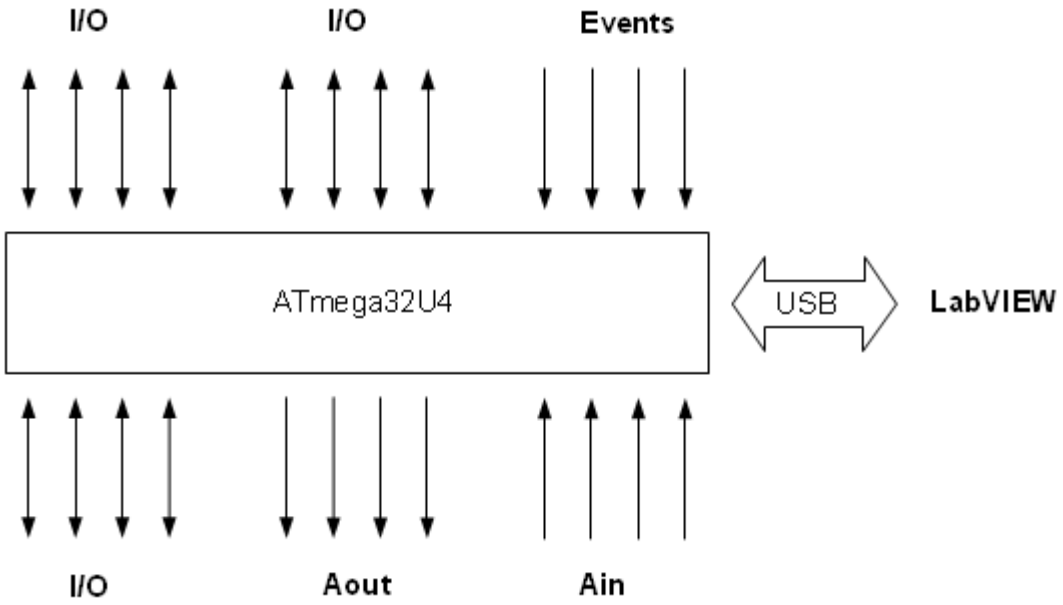
Ensimmäisenä suunniteluvaiheena oli tärkeintä määritellä mitä I/O-modulin toivottiin tekevän ja rakentaa yksinkertainen lohkokaavio I/O-modulin toiminnoista. Seuraavaksi piti suunnitella viestintäprotokolla LabVIEW:in ja I/O-modulin välille. Ohjelmakoodin jouhevan suorituksen kannalta oli järkevää suunnitella tilakone mikrokontrolleriin.

4.2 Työn määrittelyvaihe

Ensimmäinen vaihe suunnittelussa oli määritellä työn ominaisuudet. Koska työssä on käytössä elektroniikaltaan valmis I/O-moduli, ominaisuudet oli järkevä suunnitella tämän ehdoilla. Alkuasetelmien mukaan piti siis suunnitella mikrokontrolleriin ohjelma, jonka oli osattava digitaaliset I/O-liitännät, PWM, AD-muunnos ja keskeytykset.

Mikrokontrollerin piti pystyä keskustelemaan LabVIEW-ohjelman kanssa. LabVIEW:ssä on tuki sarjaporttilaitteita varten. Niinpä I/O-modulissa oli sopivaa käyttää Atmelin CDC-virtuaalisarjaportti ohjelmapakettia oman koodin pohjana.

Kuvan 5 lohkokaaviosta näkee selkeästi I/O-modulin toimintalohkot. Mikrokontrolleri on USB-väylän kautta yhteydessä tietokoneeseen ja LabVIEW-ohjelmaan. Mikrokontrollerin I/O-liitännät ovat vaihtoehtoisesti joko sisääntuloja tai uloslähtöjä. Analoginen uloslähtö (Aout) on PWM-ohjausta varten. Analoginen sisääntulo (Ain) on AD-muunnosta varten ja Events on keskeytyksille. Nuolet osoittavat mihin suuntaan tietoa on mahdollista siirtää.



Kuva 5. Lohkokaavio I/O-modulin toiminnasta.

4.3 Viestintäprotokolla

Mikrokontrollerin ja LabVIEW:n välille oli suunniteltava viestintäprotokolla. Viestintäprotokollan tarkoitus on luoda yhtenäinen komentokanta, jonka avulla tunnistetaan mitä lähetettävä ja vastaanotettava data on. Viestintäprotokollassa määritellään, mitä tapahtuu kun määrätyt bitit ovat päällä tai pois.

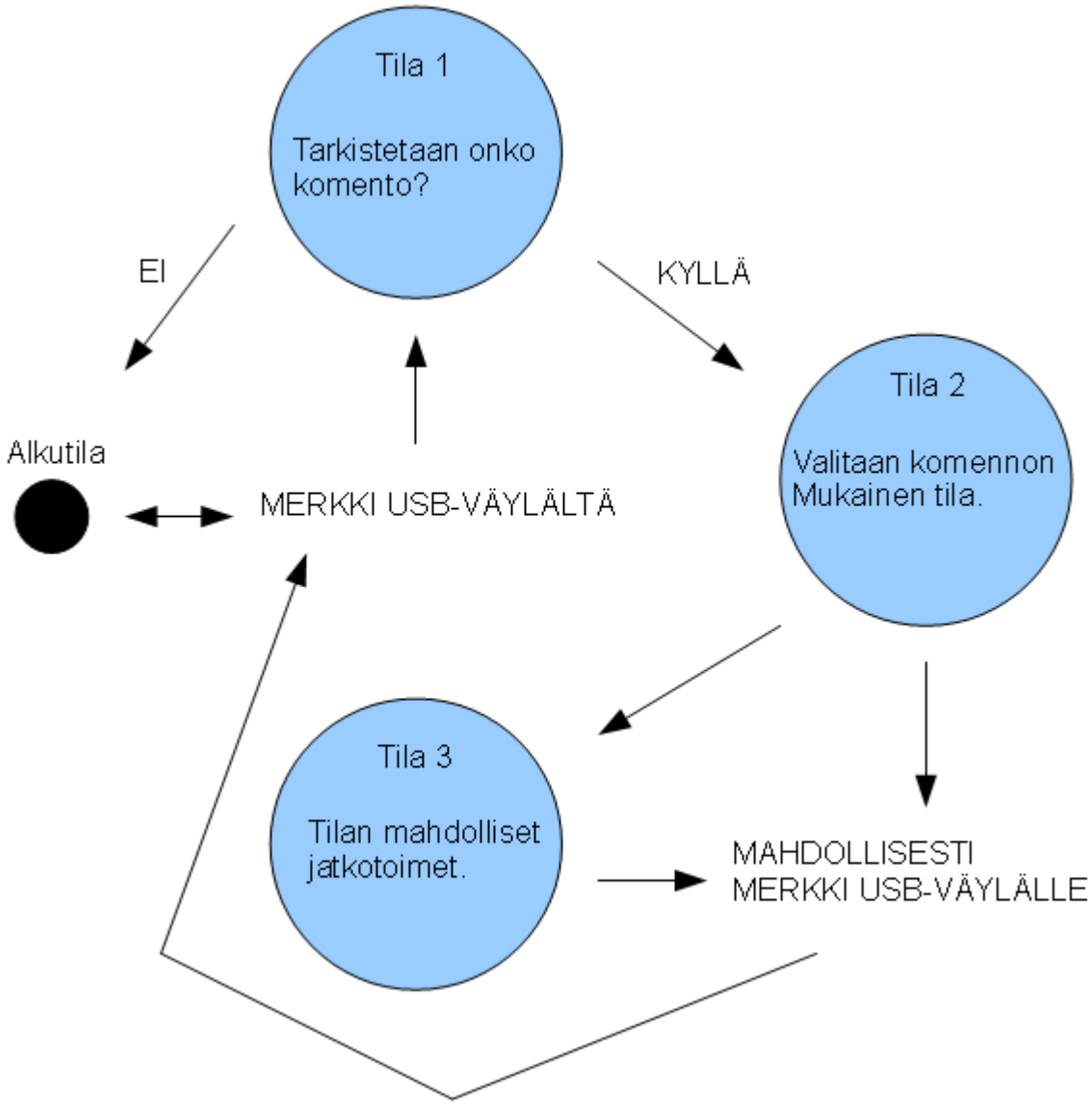
Työn viestintäprotokolla on suunniteltu lähettämään ja vastaanottamaan tietoa tavun kerrallaan. Tämä on sarjaporttiliikenteen tekninen rajoitus (TAL Technologies, Inc. 2010). Lähetettävien ja vastaanotettavien tavujen sisältö on suunniteltu toimaan tilakoneessa. Viestit on jaettu komentoon, porttiin/arvoon, bitin numeroon ja bitin arvoon. Viestintäprotokollan viestien sisältöä ja toimintaa selitetään tarkemmin työn toteutuksen yhteydessä, jossa esitellään I/O-modulin liitäntöjen tiloja. Viestintäprotokollan tiedot on kokonaisuudessaan mukana liitteenä 1.

4.4 Tilakone

Tilakone on yleinen käsite, jolla on monia käyttötapoja. Yleisesti ottaen tilakone (FSM) on laite, jolla on sisääntuloja ja uloslähtöjä. Tilakone muuttaa tilan arvoa sisääntulojen perusteella. Tilakoneen uloslähdöt riippuvat tilan arvosta. Tilakone koostuu näin ollen neljästä elementistä: joukosta äärellisiä tiloja, sisääntuloista, siirtymisehdoista seuraavaan tilaan ja uloslähdöistä. (Gadre 2000, 152.)

Tilakoneen etuja on, että yleensä yhden tilan käsittelyyn menee hyvin vähän aikaa. Käytännössä tämä tarkoittaa, että mikrokontrollerille jää enemmän aikaa suorittaa tehtäviään, esimerkiksi analogisen tiedon keruuta, sarjakomentojen käsittelyä ja matemaattisten tehtävien suoritusta. (Barnett ym. 2007, 75.)

Ohjelma oli järkevää suunnitella toimimaan tilakoneessa siten, että tilojen välissä ohjelma voi suorittaa CDC-rutiineja. Ellei ohjelma pääse suorittamaan CDC-ruutiineja, se voi jumiintua. Tilakoneeseen suunniteltiin viestintäprotokollan mukaisesti 3 toimintatilaa. Kuten kuvan 6 tilakaaviosta käy ilmi, ensimmäisen tilan aikana tarkastetaan onko USB-väylää pitkin tullut komento. Jos komentoa ei ole tullut, palataan ohjelman alkutilaan. Jos komento on tullut, valitaan komennon mukainen tila ja suoritetaan sen tehtävät. Jos tila ei vaadi jatkotoimenpiteitä, lähetetään mahdolliset tiedot USB-väylään. Tilan vaatiessa jatkotoimia siirrytään niiden suoritukseen ja lähetetään tiedot USB-väylään, jonka jälkeen tarkastetaan onko tullut uutta merkkiä. Tilojen siirtymisehdot ja toiminnat käydään tarkemmin läpi toteutusosion yhteydessä.



Kuva 6. Tilakaavio I/O-modulin ohjelmakoodin toiminnasta.

5 TOTEUTUS

5.1 Yleistä

Työn pääasialliset osa-alueet ovat mikrokontrollerin ohjelmointi ja LabVIEW-ohjelmointi. Ensimmäinen vaihe oli toteuttaa mikrokontrollerin ohjelmointi. Seuraava vaihe oli tutustua LabVIEW-ohjelmistoon ja ohjelmoida subVI-aliohjelmat, siten että mikrokontrollerin ja LabVIEW:in välinen vuorovaikutus olisi mahdollista. Lopuksi tehtiin vielä LabVIEW:llä esimerkkisovellus subVI-aliohjelmien avulla.

5.2 Mikrokontrollerin ohjelmointi

Seuraava vaihe oli ohjelmoida ATmega32U4-mikrokontrolleri. Prosessi aloitettiin lataamalla Atmelin verkkosivulta oikea CDC-ohjelmapaketti. Siirtämällä CDC-ohjelmapaketin HEX-tiedosto mikrokontrolleriin, I/O-modulin USB-portti näkyy virtuaalisarjaporttina tietokoneelle. CDC-ohjelmapakettia käytettiin oman koodin alustana.

I/O-modulin ominaisuudet ja tekstit määrittelivät, mitä ohjelmoitiin. Täytyi siis ohjelmoida digitaaliset uloslähdöt, digitaaliset sisääntulot, analogiset uloslähdöt, analogiset sisääntulot ja keskeytykset. Taulukossa 1 on nimetty piirikortin I/O-linjat niiden tarkoituksen mukaan, sekä taulukoitu ominaisuudet, joita linjoilta vaadittiin.

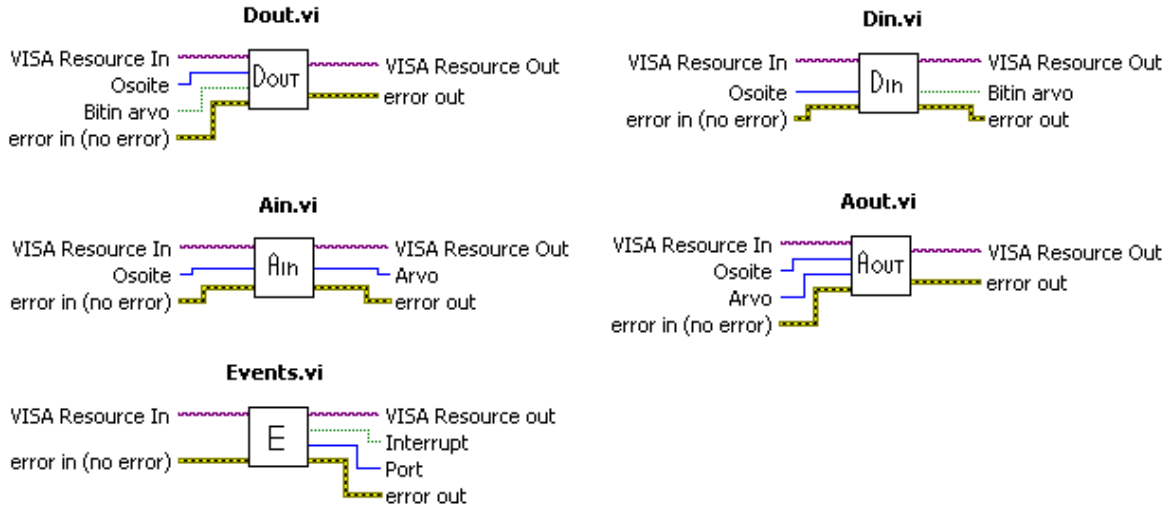
Taulukko 1. Piirikortilla olevien I/O-linjojen ominaisuudet.

Nimi	Linjojen määrä	Linjan bittisyys	Linjan tilat	Toteutus
Ain	4	10	0-1023	ADC
Aout	4	10	0-1023	PWM
Events	4	1	2	Keskeytys
I/O	12	1	2	Bitti on/off

5.3 LabVIEW-ohjelmointi

LabVIEW-ohjelmointi toteutettiin tekemällä I/O-liitännille omat subVI-aliohjelmat. SubVI on virtuaali-instrumentti, jota voi käyttää toisessa virtuaali-instrumentissa. Virtuaali-instrumentti kutsuu subVI-aliohjelmaa, eli se toimii kuten aliohjelma normaalissa ohjelmoinnissa. SubVI-aliohjelmien käyttö on tehokas ohjelmointikeino, joka mahdollistaa tietyn koodin käytön eri tilanteissa. (Bishop 2007, 197.)

Työssä tehtiin omat subVI-aliohjelmat digitaalisille uloslähdöille, digitaalisille sisääntuloille, analogisille uloslähdöille, analogisille sisääntuloille ja keskeytyksille. Ohjelmointi LabVIEW:ssä tapahtui asettelemalla sisääntulot ja uloslähdöt loogiseen järjestykseen etupaneeli-ikkunassa. Etupaneeli toimii myös ohjelman käyttöliittymänä. Lohkokaavio-ikkunassa ohjelmoitiin rakenteet, koodit ja tehtävät sekä tehtiin tarvittavat johdotukset eri osien välillä. Seuraavaksi jokaiselle subVI-aliohjelmalle tehtiin oma kuvake, jonka liittimiin sisääntulot ja uloslähdöt laitettiin paikoilleen. Valmiit subVI-kuvakkeet on esitelty kuvassa 7. Lopuksi tehtiin vielä esimerkkisovellus, jota esitellään toteutus-osion lopussa tarkemmin.



Kuva 7. Valmiit subVI-kuvakkeet seliteteksteineen.

Valmiiden subVI-kuvakkeiden käyttö selkeyttää huomattavasti sovellusten tekoa LabVIEW-ympäristössä. Ohjelmoinnissa käytettiin LabVIEW:n VISA I/O -kieltä, joka osaa käsitellä myös sarjaportteja.

5.4 I/O-modulin toiminta

Seuraavaksi tarkennetaan ohjelmoidun I/O-modulin eri tilojen toimintaa. I/O-modulin toimintatilojen LabVIEW-lohkokaaviot ovat liitteessä 2.

5.4.1 Digitaaliset I/O-liitännät

Ensimmäisen tilan aikana vastaanotetaan LabVIEW:ltä tavu, joka sisältää komennon ja liitäntöjen porttinumeron. Komennolla valitaan digitaalinen uloslähtö tai sisääntulo. Porttinumerolla valitaan käytettävä portti.

Toisen tilan aikana vastaanotetaan bitin numero ja bitin arvo. Bitin numerolla valitaan, mikä nasta on kyseessä. Bitin arvolla valitaan, saako valittu nasta arvon 0 vai 1. Portteja ja nastoja ohjelmoitaessa pitää huomioida että käytettävät portit ja nastat ovat seuraavan listan mukaiset:

- Portin B nasta 4
- Portin C nasta 7
- Portin D nastat 0-7
- Portin F nastat 0 ja 1.

Kun tarvittavat tavut on saatu lähetettyä mikrokontrollerille, ohjataan nastoja tarpeen mukaan päälle ja pois, jos kyseessä on digitaalinen uloslähtö. Digitaalisen sisääntulon ollessa kyseessä luetaan portin nastan tila ja lähetetään tilan komento- ja osoitetiedot USB-väylän kautta LabVIEW:iin.

5.4.2 Analogiset uloslähdöt

Ensimmäinen vastaanotettava tavu LabVIEW:ltä sisältää komennon käyttää PWM-lähtöä ja kanavaosoitteen, jolla oikea portti valitaan. Käytettävät PWM-lähdöt työssä ovat portin C nasta 6 ja portin B nastat 5, 6 ja 7.

Koska työssä käytetään 10-bittistä PWM-ohjausta, on PWM-ohjausarvo jaettava kahteen tavuun. Tilakoneen toisessa tilassa vastaanotetaan LabVIEW:ltä ohjausarvon 3 ylintä bittiä. Tilakoneen kolmannessa tilassa vastaanotetaan loput 7 bittiä. Mikrokontrollerissa yhdistetään saadut tavut yhdeksi 10-bittiseksi PWM-ohjausarvoksi.

5.4.3 Analogiset sisääntulot

Ensimmäinen vastaanotettava tavu LabVIEW:ltä sisältää komennon käyttää AD-muunninta ja kanavaosoitteen valittua nastaa varten. Käytössä on portin F nastat 4, 5, 6 ja 7. Saatuaan komennon ja kanavaosoitteen, AD-muuntimen alustukset suoritetaan mikrokontrollerissa ja AD-muunnos alkaa.

Muunnoksen jälkeen lähetetään tiedot LabVIEW:iin. Ensin lähetetään tunnistekomento ja kanavaosoite, jonka jälkeen lähetetään mittaustulokset kahdessa eri tavussa. Mittaustulos tallentuu kahteen eri rekisteriin, joista ensin on luettava ADCL:n tiedot ja vasta sitten ADCH:n tiedot (Atmel Corporation 2009, 312-313). Jos kyseistä järjestystä ei noudateta, mittausdata menee sekaisin.

5.4.4 Keskeytykset

Keskeytykset toteutettiin kahdella tavalla. Ensin käsitellään reaaliaikainen ISR-menetelmän totetus, jonka jälkeen käsitellään kiertokyselymentelmää. Kiertokyselymenetelmä toteutettiin varalle, sillä keskeytysvektorin kanssa oli aluksi ongelmia.

Keskeytyksiä varten oli alustettava rekisterit ja ohjelmoitava keskeytysvektori. Keskeytyksen tullessa ohjelman suoritus keskeytyy ja siirrytään keskeytysvektoriin. Keskeytysvektorissa tarkistetaan case-rakenteen avulla minkä nastan tila on muuttunut. Tässä työssä laitettiin keskeytys tunnistumaan jännitteen laskevalla reunalla, esimerkiksi kytkintä painaessa. Kytkimen painalluksen tullessa keskeytysvektorista lähetetään LabVIEW:lle komento ja osoite, joka sisältää kytketyn

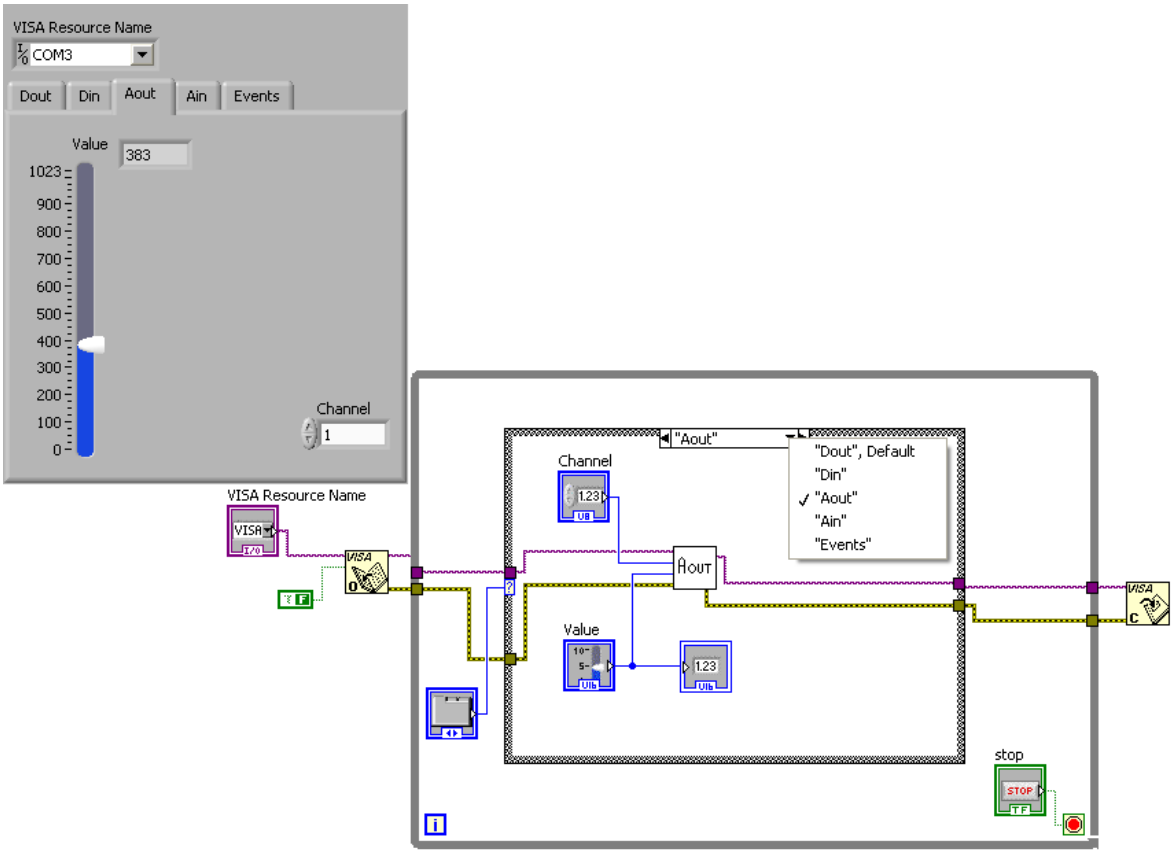
nastan tiedon. LabVIEW:iin ohjelmoidulla subVI-aliohjelmalla tarkistetaan virtuaalisarjaportin liikenne. Keskeytyksen tullessa aktivoidaan subVI-aliohjelman suoritus.

Keskeytysjärjestelmä toteutettiin myös kiertokyselymenetelmällä. Kiertokysely käynnistyy vastaanotettuaan ensin komennon, joka aktivoi kiertokyselyn. Jos ehto komennon lähetyksestä toteutuu, kiertokysely tarkistaa jatkuvasti ohjelman kierron aikana onko nappia painettu. Kiertokysely on varsin hidas menetelmä, jota kannattaa käyttää vain pakon alla.

5.5 Esimerkkisovellus

Tämän työn esimerkkisovellus on tehty lähinnä apuvälineeksi I/O-modulin testausta silmällä pitäen. Esimerkkisovelluksen avulla voi käyttää kaikkia I/O-moduliin ohjelmoituja ominaisuuksia.

Esimerkkisovellus pohjautuu LabVIEW:llä toteutettuun case-rakenteeseen ja rakenteen tilat on sidottu välilehden valitsimeen. Rakenteen eri tiloihin on laitettu jokaiseen oma työssä ohjelmoitu subVI-aliohjelma. Välilehden käyttäminen tekee esimerkkisovelluksesta todella selkeän käyttää. Välilehdillä yksinkertaisesti vain aktivoidaan käytettävä subVI. Esimerkkisovelluksen etupaneeli ja lohkokaavio on esitelty kuvassa 8.



Kuva 8. Esimerkkisovelluksen etupaneeli ja lohkokaavio.

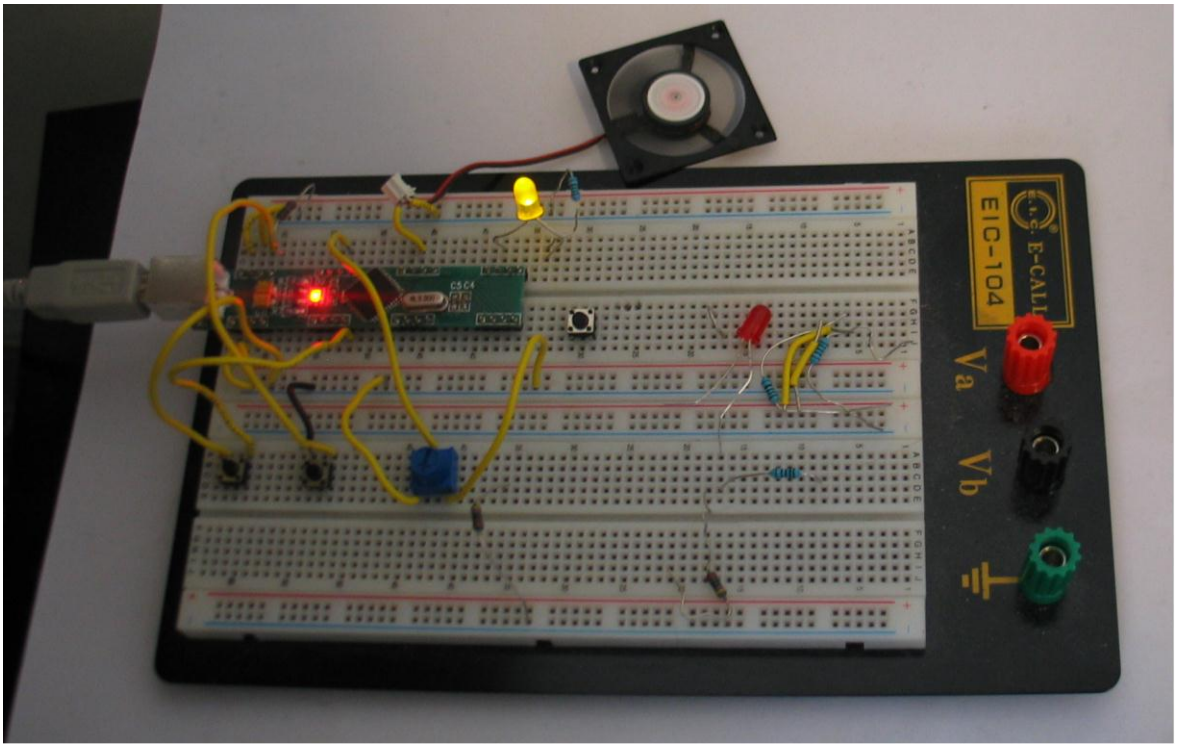
6 TESTAUS

6.1 Yleistä

Toimivan ohjelmiston kannalta testaustyö on tärkeä osa-alue. Työn eri vaiheissa käytettettiin erilaisia testausmenetelmiä. Testauksen apuna käytettiin koekytkentälautaa, Docklight-ohjelmaa, oskiloskooppia ja yleismittaria. Docklight on sarjaliikenteen simulointiin ja testaukseen tarkoitettu työkaluohjelma.

6.2 Koekytkentälauta

I/O-moduli oli kytketty koekytkentälautaan testien ajaksi. Koekytkentälauta kaikessa yksinkertaisuudessaan mahdollistaa erilaisten johdotusten ja laitteiden kiinnittämisen I/O-moduuliin.



Kuva 9. I/O-moduli kiinnitettynä koekytkentälautaan.

Koekytkentälautaan oli kiinnittetty mikrokontrollerin lisäksi nappeja, ledejä, säädettävä vastus ja tuuletin. Nämä osat mahdollistivat jo riittävän monipuolisen testausympäristön I/O-modulin toimivuuden toteamista varten.

6.3 Mikrokontrollerin koodin testaus

Mikrokontrollerin koodia jouduttiin testaamaan jatkuvasti ongelmien ilmetessä. AVRstudion debuggeri oli hyvä apuväline. Debuggeri on ohjelmavirheiden etsimistä varten. Debuggaus-tilassa ohjelmaa suoritetaan askel kerrallaan, joten se on kätevä keino nähdä mitä ohjelman suorituksessa todella tapahtuu.

Kun koodista oli saatu riittävän virheetön versio toteutettua, niin sarjaporttikomentojen toiminnan testausta suoritettiin Docklight-apuohjelmalla ja koekytkentälevyn avulla. Yleismittarilla ja oskiloskoopilla mitattiin I/O-modulin nastojen tiloja ja PWM-toteutuksen kanttiaaltoa. I/O-modulin toiminnan näytettyä hyvältä, oli aika siirtyä LabVIEW-toteutukseen. Lopputestaus I/O-modulille suoritettiin LabVIEW-esimerkkisovelluksen avulla.

6.4 LabVIEW-testaus

Testaus LabVIEW:ssä aloitettiin järjestelmällisesti testaamalla yksi subVI-aliohjelma kerrallaan. Tällä menetelmällä vältyttiin mahdollisilta muiden subVI-aliohjelmien aiheuttamilta ongelmilta. Keskeytysten vastaanotto osoittautui aluksi pienoiseksi ongelmaksi. Keskeytysmenetelmä toteutettiin tämän johdosta kahdella eri menetelmällä. Ongelmista päästiin kuitenkin eroon, joten I/O-moduliin otettiin käyttöön lopulta vaihtoehdoista parempi ISR-versio.

Seuraava vaihe oli testata useampaa subVI-aliohjelmaa samaan aikaan. Testauksen aikana huomattiin selvää viivettä, jos aktivoitiin kaksi subVI-aliohjelmaa samaan aikaan. Aluksi tämän luultiin johtuvan siitä että virtuaalinen sarjaportti meni tukkoon liiallisesta liikenteestä. Todellinen syy kuitenkin paljastui ohjelmointivirheeksi, joka saatiin korjattua.

Lopuksi testattiin vielä LabVIEW-esimerkkisovelluksella ja koekytkentälevyyn kiinnitetyillä komponenteilla I/O-modulin toimintaa kokonaisuutena.

7 TULOKSET JA ARVIOINTI

7.1 Tulokset

Tämän opinnäytetyön tarkoituksena oli ohjelmoida I/O-moduli ja liittää se LabVIEW:iin. Opinnäytetyön tehtävät onnistuttiin toteuttamaan. Tulokseksi saatiin LabVIEW:llä toimiva I/O-moduli.

Mikrokontrollerin koodia varten suunniteltiin tilakone ja viestintäprotokolla. Mikrokontrollerin koodi toimii testien perusteella virheettömästi. Mikrokontrolleri lähettää ja vastaanottaa sarjaporttikomennot tarkalleen niin kuin oli suunniteltukin.

LabVIEW-osio toimii kuten pitääkin. Tehtyjen subVI-kuvakkeiden avulla on huomattavan helppoa kehittää sovelluksia I/O-modulille. Esimerkkisovelluksella testatessa I/O-moduli toimi LabVIEW:ssä kuten oli tarkoitus.

7.2 Arviointi

Työ oli varsin monipuolinen sisältäen C-ohjelmointia, LabVIEW-ohjelmointia ja hieman elektroniikan tuntemustakin. Työ olikin varsin opettavainen monipuolisuutensa vuoksi. Työlle asetetut tavoitteet saavutettiin onnistuneesti.

Työn haastavin vaihe oli LabVIEW-ohjelmointi, sillä ohjelma ei ollut työn tekijälle entuudestaan kovin tuttu. LabVIEW-toteutuksen aikana olikin varmasti eniten ongelmia. Kaikki huomatut ongelmat saatiin kuitenkin ratkaistua. Luultavasti tällä osa-alueella olisikin työn kannalta eniten parannettavan varaa.

Opettavaisinta työssä oli itse mikrokontrollerin ohjelmoiminen. Mikrokontrollerin ohjelmoinnissa käytettiin hyväksi kaikkia tärkeimpiä mikrokontrollerin ominaisuuksia. Ohjelmointi vaati taustatyötä mikrokontrollerin teknisistä ominaisuuksista, tär-

keimpänä ajastin/laskurin toiminta. Kaiken kaikkiaan työ oli tekijälle todella opettavainen kokonaisuus.

7.3 Jatkokehitys

Opinnäytetyön tavoitteet saavutettiin, mutta aina löytyy mahdollisuuksia jatkokehitykselle. I/O-moduliin ohjelmoitu koodi toimii tilakoneessa, joten koodi on todella joustava ominaisuuksien lisäilyjen kannalta.

Työssä ei muutettu I/O-modulin fyysisiä komponentteja. Tällä saralla olisikin mahdollista kehittää I/O-modulia. Tälläisenään I/O-modulista ei saa uloslähtöjännitteitä kuin 5 V ja maksimi virransyöttö sekä vastaanottokestoisuus on 20 mA. Näillä arvoilla on mahdotonta ohjailla kovin isoja laitteita. Kehitysmahdollisuutena erinomainen, eikä edes kovin työläs, voisi olla juurikin I/O-modulin elektroniikan muokkaus siten että se soveltuu vaikkapa teollisuuskäyttöön.

Seinäjoen ammattikorkeakoulun LabVIEW-kurssilla on käytetty National Instrumentsin valmistamaa I/O-modulia. Yksi jatkokehitysmahdollisuus olisikin tehdä I/O-modulista ja LabVIEW:stä opiskelijoille opetuspaketti.

LÄHTEET

Atmel Corporation. 2009. ATmega16U4/32U4 Preliminary. [Pdf-julkaisu]. San Jose, CA, USA. [Viitattu 4.11.2010]. Saatavana: http://www.atmel.com/dyn/resources/prod_documents/doc7766.pdf

Atmel Corporation. 2007. AVR280: USB Host CDC Demonstration. [Pdf-julkaisu]. San Jose, USA. [Viitattu 4.11.2010]. Saatavana: http://www.atmel.com/dyn/resources/prod_documents/doc7727.pdf

Atmel Coporation. 2010. FLIP 3.4.2 Release Notes. [Txt-tiedosto]. San Jose, CA, USA. [Viitattu 4.11.2010]. Saatavana: http://www.atmel.com/dyn/resources/prod_documents/FLIP_3_4_2_Release_Notes.txt

Aquaticus ROV. 2010. Aquaticus PWM guide. [Verkkosivu]. [Viitattu 4.11. 2010]. Saatavana: http://aquaticus.info/pwm

Barnett, R., O'Cull, L. & Cox, S. 2007. Embedded C Programming and The Atmel AVR. Kanada: Thomson Delmar Learning.

Bishop, R.H. 2007. LabVIEW 8 Student Edition. Upper Saddle River, NJ, USA: Pearson Prentice Hall.

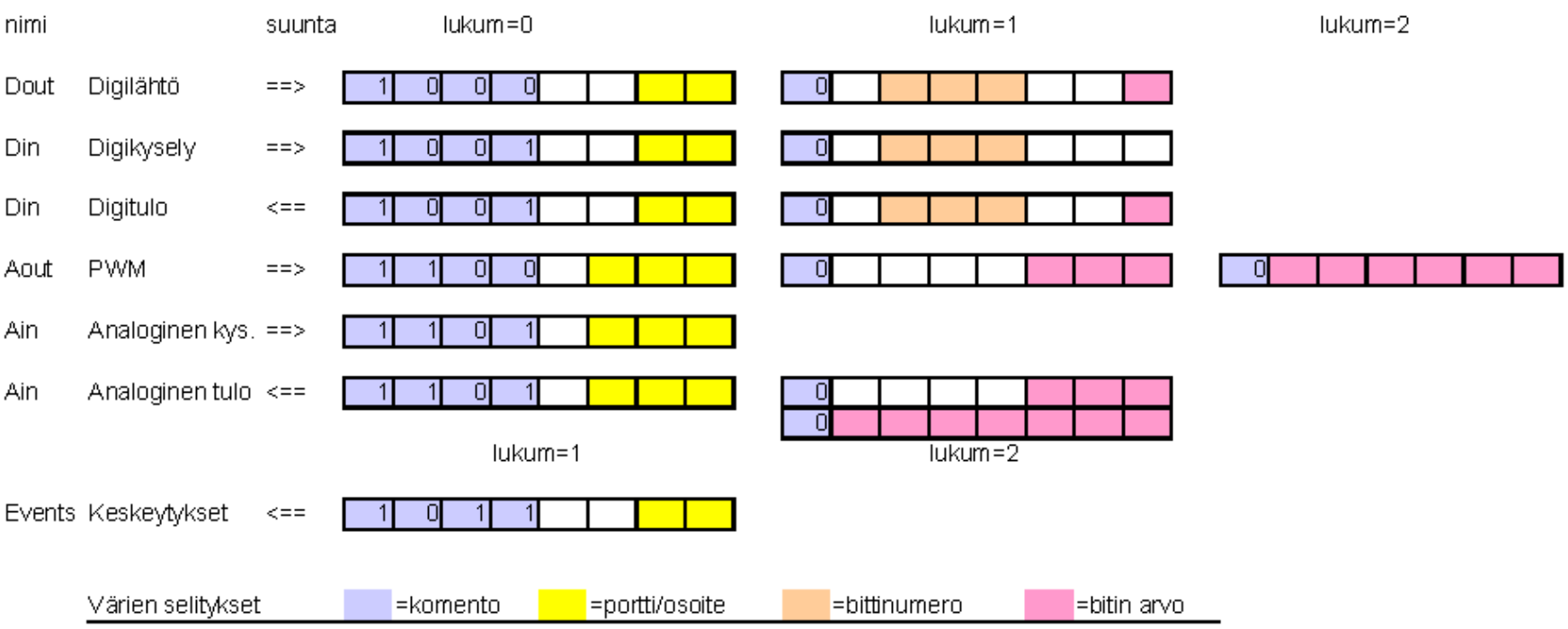
Gadre, D.V. 2000. Programming & Customizing the AVR Microcontroller. Blacklick, OH, USA: McGraw-Hill Professional Publishing.

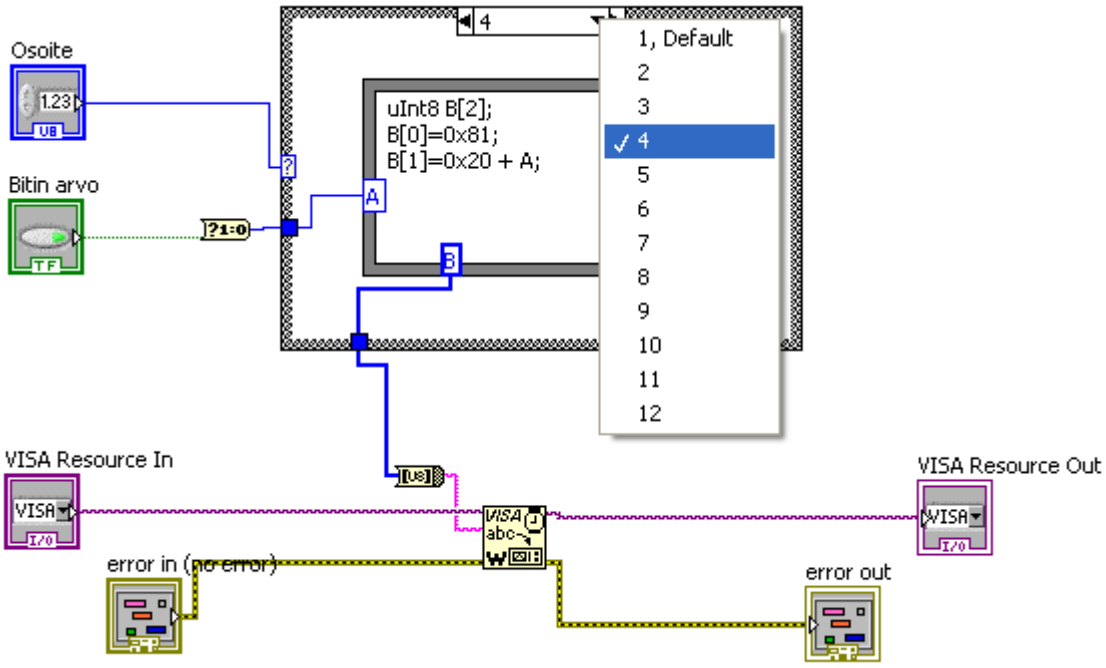
Tal Technologies, Inc. 2010. Introduction to Serial Communications. [www-julkaisu]. Philadelphia, PA, USA: [Viitattu 4.11.2010]. Saatavana: http://www.taltech.com/resources/intro-sc.html

Vahtera, Pentti. 2003. Mikro-ohjaimen ohjelmointi C-kielellä. Helsinki: WSOY.

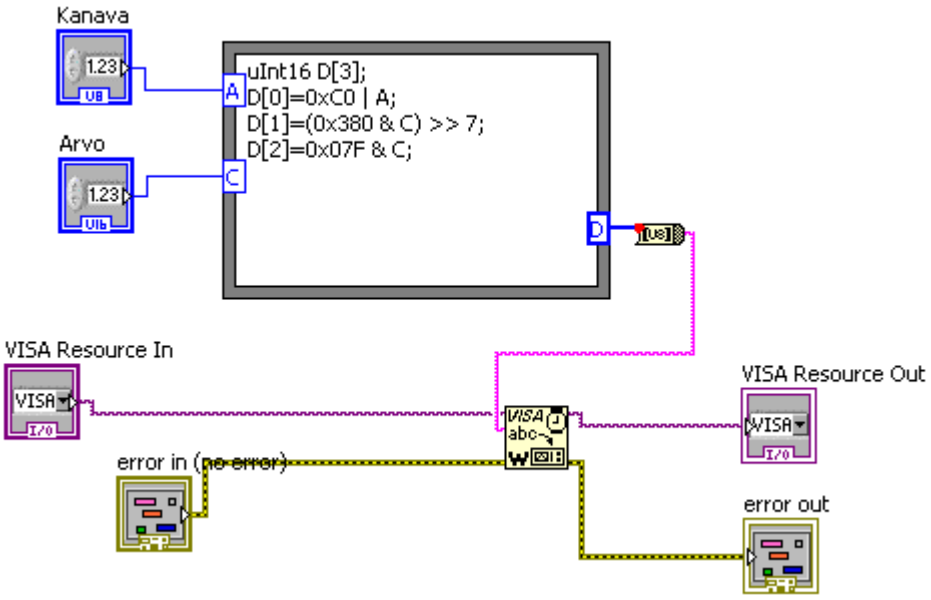
Vahtera, Pentti. 2008. AVR_rauta. ADC-ohjelmointia. [Pdf-julkaisu]. Salo: [Viitattu 4.11.2010]. Saatavana: http://www.microsalo.com/Kirja/Kirja_29092008.rar

LIITE 1: Viestintäprotokolla

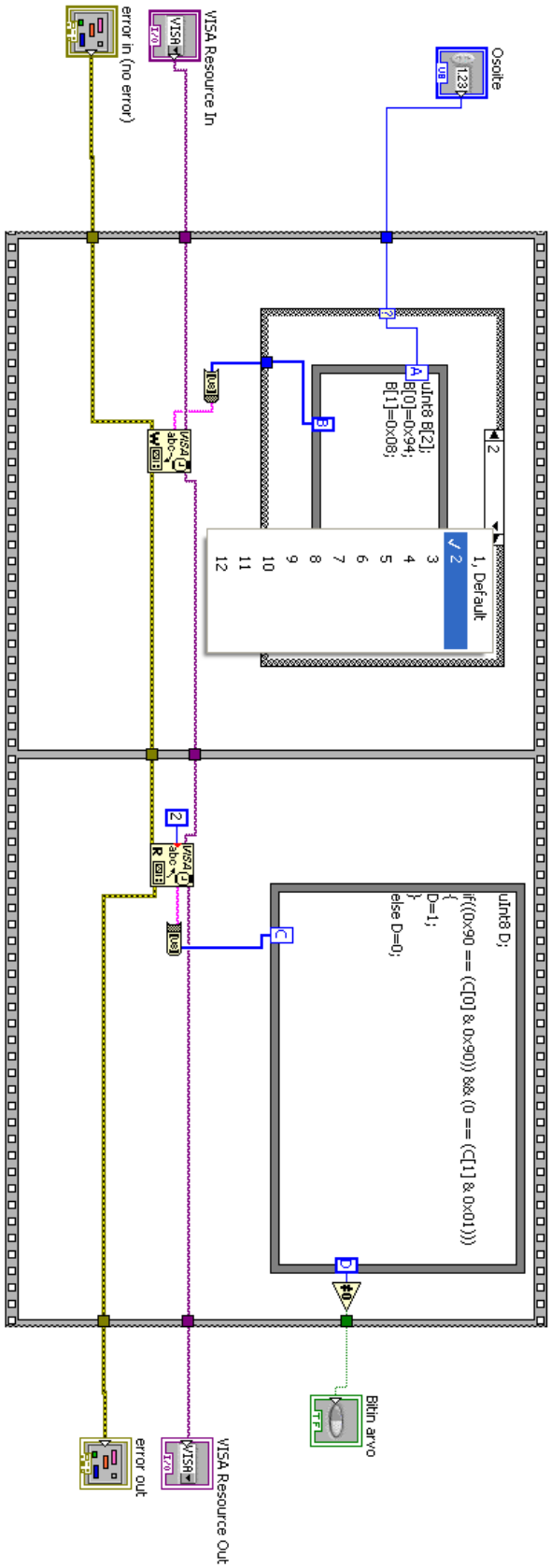




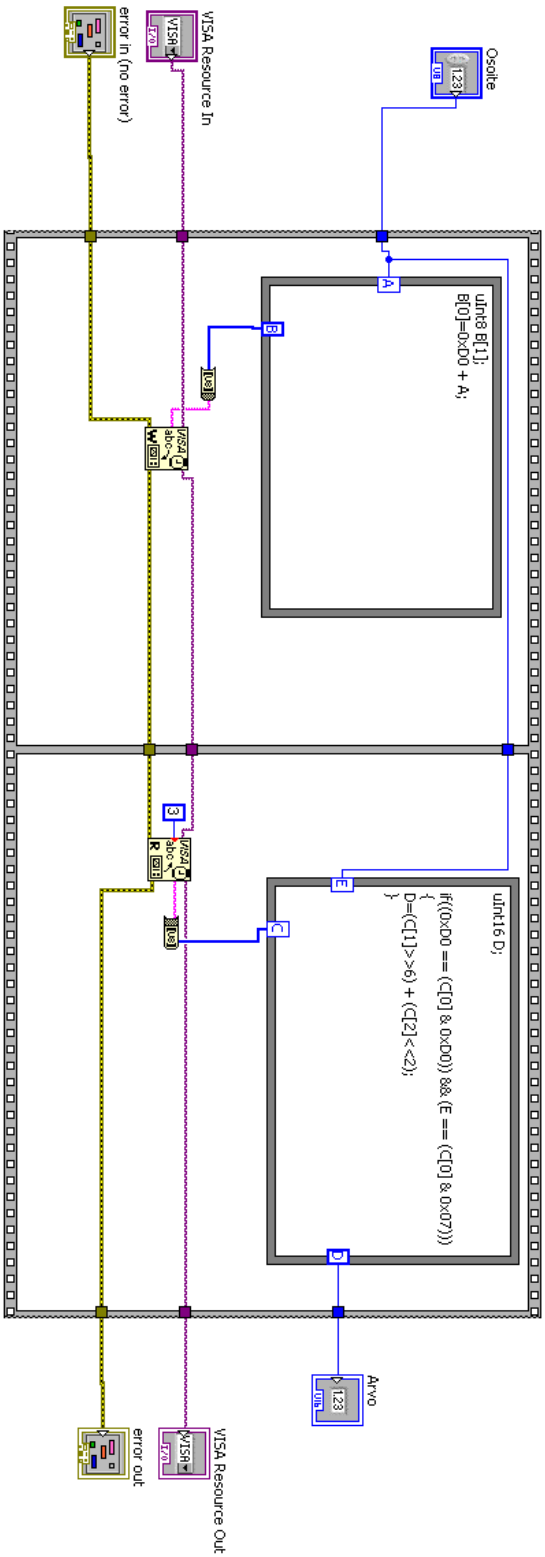
Digitaalisen uloslähdön lohkokaavio.



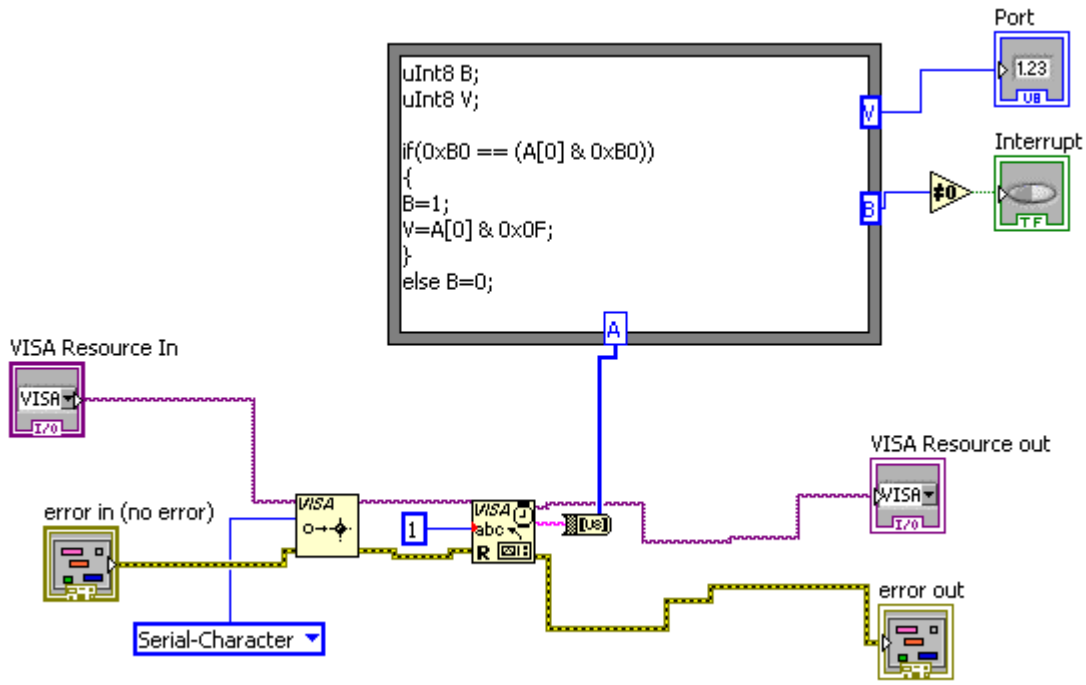
Analogisen uloslähdön lohkokaavio.



Digitaalisen sisääntulon lohkokaavio.



Analogisen uloslähdön lohkokaavio.



Keskeytysten lohkokaavio.