TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Tietokonetekniikan suuntautumisvaihtoehto

Joni Nurminen

ETÄOHJATTAVA VALVONTAKAMERA

Työn valvoja: Yliopettaja Kai Poutanen

Tampere 22.04.2008

Tekijä: Joni Nurminen
Työn nimi: Etäohjattava valvontakamera

Päivämäärä: 22.04.2008
Sivumäärä: 32 sivua ja 19 liitesivua
Hakusanat: Attiny2313, web-kamera, valvontalaite, askelmoottori
Koulutusohjelma: Tietotekniikka
Suuntautumisvaihtoehto: Tietokonetekniikka

Työn valvoja: Yliopettaja Kai Poutanen

Työssä suunnitellaan ja toteutetaan web-sivujen kautta ohjattavissa oleva valvonta/seurantajärjestelmä. Internetissä olevalta hallintasivulta pystyy ohjaamaan jalustalle rakennettua web-kameraa sekä pysty että vaakasuuntaan, mutta kumminkin vain rajakytkimien sallimissa rajoissa. Hallintasivu suojataan kirjautumisella, jossa käyttäjältä kysytään tunnusta ja salasanaa.

Hallintasivulla käyttäjä voi itse määrätä kameran pyörittämiseen tarvittavien askelien määrän. Käyttäjä voi myös sulkea/laittaa päälle sivulta ohjattavaa kameraa sekä sen käyttämiä sovelluksia. Tilatiedot päivittyvät ruudulle heti käyttäjän nähtäväksi. Käyttäjä voi halutessaan laittaa liiketunnistuksen päälle. Jos kuvaan tulee tällöin liikettä, tallentuvat kuvat automaattisesti palvelimelle. Tämän mahdollistaa skriptikielellä koodattu ohjelma. Käyttäjä pystyy myös hallinnoimaan tallennettuja kuvia. Kuvien nimeäminen toteutettiin ”aikaleimalla”, joten tallentuneen kuvan ottoajankohta selviää nimestä, päivän ja sekunnin tarkkuudella.

Mikrokontrolleri ohjelmoitiin C-kielellä käyttäen AVR-GCC-kääntäjää, sekä Avrdude-ohjelmointiohjelmaa. Kytkentä piirrettiin Eagle-ohjelmistolla. Käyttöjärjestelmänä oli Fedora 6 ja palvelinohjelmana toimi Lighttpd.

Author:	Joni Nurminen
Title:	Remote controlled Surveillance Camera
Date:	22.04.2008
Number of pages:	32 pages and 19 appendices
Key words:	Attiny2313, Webcam, Step Motor, Security system
Program:	Information Technology
Specialisation:	Computer Engineering
Supervisor:	Senior Lecturer Kai Poutanen

The purpose of this thesis is to plan and realise a surveillance system controlled via a web-page interface. The webcam can be turned in the vertical or horizontal direction through the interface, but only in the area that border sensors will allow. The control page is secured with login system, where the user types in an username and a password.

On the control page the user can control how the webcam will turn and how many steps it will run. The user can also turn the webcam on and off via the web interface. Information about the states will immediately update to screen. If the user wants, he/she can turn the motion detector online, if the camera detects motion, it will automatically save the frames. Naming of the pictures is done with "timestamps", so it is possible to find out when the picture was taken.

The camera interface is coded by using PHP and XHTML programming languages. The micro controller is programmed by using C language and with the Avrdude programming program. The operating system is Fedora 6 and the www-server in use was Lightppd.

SISÄLLYSLUETTELO

KÄYTETYT TERMIT JA LYHENTEET

Fedora 6	RPM-pohjainen Linux-käyttöjärjestelmä
Konsoli	Vastaava kuin terminaali, pääte tai pääte-emulaattori
PHP	*Hypertext Preprocessor*, dynaamisiin web-palvelimiin tarkoitettu ohjelmointikieli
Xhtml	*eXtensible Hypertext Markup Language*, HTML:stä kehitetty www-sivujen merkintäkieli
MySQL	SQL-tietokannan hallintajärjestelmä
Moduuli	Linuxin ytimeen (kernel) ladattavia osia, kuten laiteajureita
Gedit	Vapaan ohjelmistolisenssin alla toimiva tehokas tekstieditori
MakeFile	Aputiedosto, jolla helpotetaan esim. C-ohjelmien kääntämistä
Yum	Fedorassa käytetty komentorivipohjainen paketinhallintatyökalu
Rpmfind	Web-palvelu, josta voi hakea RPM-paketteja hakusanoilla
Bash	Useimpien Linux-jakeluiden oletuskomentotulkki
Skripti	Skriptikielellä (komentosarjakieli) kirjoitettu lyhyt koodin pätkä
ATtiny	Atmelin yksi mikrokontrollerityyppi

1 JOHDANTO

Tarkoituksena on suunnitella ja toteuttaa monipuolinen ja käyttäjäystävällinen Internet-sivun kautta ohjattava valvontalaite. Internet on väärällään kaikenlaisia askelmoottoreilla toteutettuja sovelluksia. Sieltä löytyy sälekaihtimiensulkijaa ja kameran kääntelijää. Pääosa web-kameran ohjausyksiköistä pystyi kääntämään kameraa vain vaakatasossa ja silloinkin sen sai käännettyä niin, että johdot menivät solmuun.

Tästä lähti sitten idea toteuttaa täysin oma ohjausyksikkö kameralle, jolla pystyy valvomaan 360 asteen aluetta niin vaaka kuin pystysuunnassakin. Tällaisen valvontayksikön voisi kiinnittää vaikka kattoon tarvitsematta pelätä, että se jumittuisi käytössä omiin johtoihinsa. Kameraa voi ohjata web-pohjaiselta hallintasivulta vaikka kännykällä tai pda:lla. Halutessaan käyttäjä voi jättää valvontayksikön kuvaamaan tiettyä kohdetta ja tallentamaan palvelimelle kuvat, joissa on tapahtunut liikettä. Käyttäjä voi sitten selata kännykällä kameran tallentamia kuvia vaikka ulkomailta ja selvittää sitten kuvista, kuka ne omenat varasti mummon omenapuista.

Vastaavanlaisia käyttökohteita valvontayksikölle löytyy varmasti. Tässä opinnäytetyössä toteutetaan etäohjattava valvontayksikkö täydellisenä komponenttien valinnasta lähtien lopulliseen käyttöön.

2 LAITTEISTO

2.1 Palvelin

Palvelimena toimii PC-tietokone, jossa on AMD Athlon 1300+ -prosessori ja muistia 512 Mt. Käyttöjärjestelmänä on Fedora 6, joka on ilmainen Linux-jakeluversio.

Linux-pohjainen käyttöjärjestelmä on hyvä valinta, koska se on ilmainen ja siihen on saatavilla lähes vastaavat ohjelmistot kuin Windowsissakin on, mutta ilmaiseksi. Internetistä löytyy paljon oppaita ja ohjeita Linuxin ja sen ohjelmistojen käyttöön. Linuxilla sarjaportin käsittely on helppoa ja sarjaporttiin kirjoittaminen onnistuu yhdellä konsolikoodirivillä. Esimerkkinä ’a’-kirjaimen lähetys sarjaporttiin *echo a > /dev/ttyS0*

WWW-palvelimeksi valittiin Ligthttpd:n, joka on kevyempi vaihtoehto Apachen httpd-palvelimista. Sovelluksen käyttöliittymä ei tarvitse toimiakseen Apachen suurta ominaisuuskirjoa. Lighttpd löytyy Fedoran pakettienhallinnasta, joten asennus tapahtuu helposti kirjoittamalla konsoliin komento *yum install lighttpd*

Käyttöliittymä on suojattu kirjautumissovelluksella, joka käyttää PHP:n sessioneja ja MySQL-tietokantaa toimintaansa. PHP on ohjelmointikieli, jota käytetään erityisesti Web-palvelinympäristöissä dynaamisten web-sivujen luonnissa. PHP-tulkiksi asennettiin paketinhallinnasta löytyvä PHP5-versio.

MySQL on tehokas SQL-tietokannan hallintajärjestelmä. MySQL-tietokannan päälle tehtävässä ohjelmoinnissa käytettiin PHP-ohjelmointikieltä. PHP:n ja MySQL-tietokannan asennus tapahtuu konsolikomennolla
yum install php php-mysql mysql mysql-server

2.2 Web-kamera

Laitteen web-kamerana toimii Logitech QuickCam Pro 3000, joka oli lojunut hyllyssä käyttämättömänä jo pitkän aikaa. Kamera hyödyntää USB-liitäntää ja kuvan laatu on kohtalaisen hyvä. 640x480-resoluutio on tarpeeksi tarkka tähän työhön. Tarkennuksen voi tehdä pyörittämällä linssiä tarvittavaan suuntaan (kuva 1).

Kuva 1. Logitech QuickCam Pro 3000

2.3 Kameran asennus

Linux-järjestelmien ehkä suurin ongelma on laitteistoajurien huono saatavuus. Ajureiden asentaminenkaan ei aina ole helppoa, ja hommaan, joka saadaan tehdyksi Windowsissa napin painalluksella, saattaa Linuxissa vierähtää useita tunteja.

Kameran toimintakuntoon saattamiseksi Linuxin kernelissä pitää olla tuki Video4Linuxille ja usb:lle. Konsoliin kirjoitettu komento *lsusb* listaa löydetyt usb-laitteet. Jos järjestelmä on tunnistanut kameran, niin listauksesta pitäisi löytyä
Bus 002 Device 008: ID 046d:08b0 Logitech, Inc. QuickCam 3000 Pro [pwc].

Vakiokernelissä on mukana muutamia ajureita Video4Linuxille, ja ne voi tarkistaa polusta */usr/src/kernels/2.6.22.14-72.fc6-i686/drivers/media/video/*.

Uusimmista kerneleistä löytyy pwc-ajuri, joka on Nemosoftin kehittämä Philipsin ja Loqitechin web-kameroille. Komento *dmesg | less* listaa kernelissä jo esikäännetyt moduulit, jotka käynnistyvät aina bootin yhteydessä.

Listauksesta löytyi seuraavaa:

pwc: Philips webcam module version 10.0.13 loaded
pwc: Supports Philips PCA645/646, PCVC675/680/690, PCVC720[40]/730/740/750 & PCVC830/840
pwc: Also supports the Askey VC010, various Logitech Quickcams, Samsung MPC-C10 and MPC-C30
pwc: Logitech QuickCam Pro 3000 USB webcam detected
pwc: Registered as /dev/video0

Listauksesta voi lukea, että moduulin versio 10.0.13 on ladattu. Ajuri tukee Philipsin web-kameroita sekä useita Logitechin kameroita. Logitetechin Quickcam pro 3000 on tunnistettu ja rekisteröity laitteeksi */dev/video0.*

Kuvankaappausohjelmaksi asennettiin Xawtv, joka on Linuxin yksi suosituimpia television katseluohjelmia. Xawtv-paketin mukana tulee useita ohjelmia, mutta se mitä paketista tarvittiin, oli webcam-ohjelma. Tällä ohjelmalla voi lähettää etäpalvelimelle säännöllisin väliajoin kuvakaappauksen tv-lähetyksestä ftp:llä tai ssh:lla.

Rpmfind palvelulla löydettiin Xawtv:n versio 3.9.5, joka asennettiin koneelle.
Asennus loi *.webcamrc* -nimisen confiq-tiedoston, jonka sisältöä muutettiin Gedit nimisellä tekstieditorilla. Komento *gedit /srv/www/lighttpd/.webcamrc*
avaa tiedoston editoitavaan muotoon.

```
Delay = 7
[ftp]
dir  = /srv/www/lighttpd/serial/kamera/
file = webcam.jpg
tmp  = uploading.jpg
passive = 1
debug = 0
auto = 0
local = 1
ssh = 0
```

Dir-kohtaan laitettiin polku, jonne webkameran still-kuvat lähetetään. Poluksi muutettiin Lighttpd-palvelimen hakemisto. Delaylla määritetään, monenko sekunnin välein still-kuva lähetetään palvelimelle /2/.

Komennolla *webcam* & käynnistään ohjelma. Sen lopussa oleva &-merkki määrää ohjelman tausta-ajoon. Ohjelma pyörii taustalla ikiloopissa ja lähettää saadun kuvan 7 sekunnin välein määrätyn polun päässä olevaan kansioon ja tässä tapauksessa omalle paikallispalvelimelle.

3 MIKRO-OHJAINKORTTI

3.1 Komponentit

Ohjaimeksi valittiin Atmelin AVR-tuoteperheeseen kuuluva 8-bittinen Attiny2313-prosessori. Tähän päädyttiin, koska Internetistä löytyi helposti valtava määrä tietoa tästä prosessorista, mutta pääsyynä valintaan kuitenkin oli se, että Linuxiin oli tarjolla avoimeen lähdekoodiin perustuvia ohjelmia prosessorin ohjelmoimiseksi. Attiny2313-mikrokontrollerin ominaisuuksia:

-2 KB Flash-muistia
-128 B SRAM- käyttömuistia
-128 B EEPROM-muistia
-Yksi 8-bittinen ajastin/laskuri
-Yksi 16-bittinen ajastin/laskuri
-Vahtiajastin
-SPI-sarjaliitäntä piirin ohjelmointia varten
-Sarjaliitäntä
-18 I/O-liitäntää
-Kaksi virransäästötilaa (Idle ja Power Down) /5/

Kiteeksi valittiin 12 MHz:n kide, jonka molemmat nastat kytkettiin prosessorin datalehden suosituksen mukaan 27 pF keraamikondensaattoreilla maahan./5, s. 26/.

Prosessorin käyttöjännitteen vakavointiosassa käytettiin L7805CV-jänniteregulaattoria suotokondensaattoreineen. Regulaattorissa on vakiona sisäänrakennettu oikosulku- ja ylilämpösuojaus. Regulaattorin virtakestoisuus on 1 A. Jänniteregulaattori pyrkii aina pitämään lähtöjännitteen tietyllä tasolla. L7805CV-tyyppinumerosta, viimeinen numero (5) kertoo lähtöjännitteen tason suuruuden. /7/

On suositeltavaa käyttää regulaattorin kanssa suotokondensaattoreita, jotka pyrkivät suodattamaan jännitteen muutoksia ja estämään regulaattorin värähtely.

Tulopuolelle laitettiin ensin 1N5408-tyypin diodi estämään virrankulku, jos jännitelähteen napaisuus on väärinpäin. Diodin virtakesto on 3 A. Tulopuolelle laitettiin vielä 4,7 uF/35 V:n tantaalikondensaattori etusuodattimeksi.

Lähtöpuolelle laitettiin kaksi suotokondensaattoria. 4,7 uF/35 V:n tantaalikondensaattori sekä yksi 100 nF muovikondensaattori suodattamaan regulaattorin lähtöpuolta. /7, s. 4/

Tarvittiin vielä yksi jänniteregulaattori suotokondensaattoreineen hoitamaan jännitteen vakavointi askelmoottorien ohjaukseen. Tähän tehtävään valittiin L7808CV-jänniteregulaattori, jonka lähtöjännite on 8 V.

Moottorien ohjauspiireiksi valittiin L239B-tyypin kaksois-h-siltapiirit. Piirin ohjausjännite on 5 V, ja piirissä on 4 kanavaa, joista saa ulos 1 A:n jokaisesta. /4/

Tietokoneen ja mikro-ohjainkortin saamiseksi kommunikoimaan keskenään tarvittiin vielä sarjaliitin sekä TTL/RS-muunnin, jolla sovitetaan lähtevän datan jännitetasot (0 ja 5 V) sarjaliitännän jännitetasojen (-12 V ja 12 V) kanssa ja toisinpäin.

Sarjaliittimeksi valittiin tavallinen 9-nastainen D-liitin (piirilevymalli). TTL-/RS-muuntimeksi valittiin ST232ACN-piiri. Piirin datalehden mukaan, muunnin tarvitsee vielä viisi 0,1 nF:n tantaalikondensaattoria toimiakseen./6, s. 4/

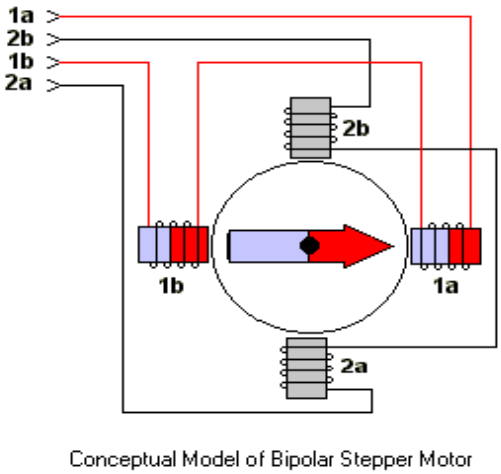
Moottorien ja anturien liittämiseksi mikro-ohjainkorttiin tarvittiin vielä 14 ruuviterminaalia. Raja-antureiksi hankittiin neljä SS-5GL2-tyypin kytkintä (kuva 2). Jännitteen tuomiseksi kortille tarvittiin 2,1 mm:n urospuolinen DC-runkoliitin.

Kuva 2. Rajakytkin

3.2 Askelmoottorit /8/

Askelmoottoreita käytetään paljon kohteissa, joissa tarvitaan tarkkaa paikoitusta. Moottoreita löytyy vaikka tietokoneiden levyke-, kovalevy- ja CD-Rom-asemista, joissa luku/kirjoituspää pitää saada tarkasti paikannettua levykkeen uralle. Kytkennällisesti askelmoottorit on jaettavissa kahteen ryhmään: Bipolaariseen ja Unipolaariseen. Unipolaarit moottorit tunnistaa siitä, että niistä lähtee 5, 6 tai 8 johtoa. Bipolaarimoottorista lähtee vain 4 johtoa.

Bipolaarimoottorissa on 2 käämiä, jotka ovat identtisiä toisilleen, mutta eivät ole sähköisesti kosketuksissa toisiinsa. Kun käämiin (staattori) tuodaan jännite, syntyy magneettikenttä, joka pyrkii kääntämään roottoria sitä kohti. Vaihtamalla seuraava kela jännitteiseksi ja jatkamalla tätä tietyllä sekvenssillä saadaan roottori pyörimään (kuva 3).



Kuva 3. Bipolaarimoottori /8/

Bipolaarimoottoria voidaan ajaa kolmella eri tavalla: Wave-ajo, Hi-torque-ajo sekä Half-step-ajo. Wave-ajo, jonka virrankulutus on pienempi kuin muissa, koska vain yhteen vaiheeseen tuodaan kerralla jännite. Tässä tapauksessa jännite ohjataan seuraavalla sekvenssillä:

1. *0001*
2. *0010*
3. *0100*
4. *1000*

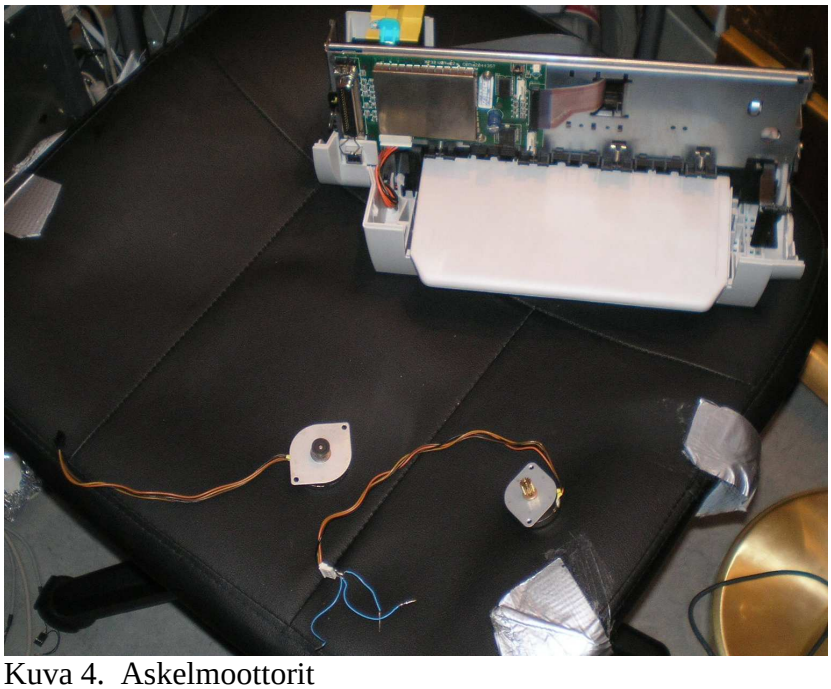
Hi-torque-ajossa tuodaan kahteen vierekkäiseen vaiheeseen jännite, joten roottori pysähtyy vaiheiden väliin. Tällä saadaan suuri pitomomentti ja vääntövoima, mutta virrankulutus vastaavasti kasvaa. Ohjaussekvenssi:

1 .*0011*
2. *0110*
3. *1100*
4. *1001*

Half-step-ajossa roottori pysäytetään vuoron perään sekä vaiheiden väliin, että vaiheeseen. Näin normaaliin askelpituuteen saadaan vielä väliaskel, joten askelmäärä kaksinkertaistuu. Ohjauksessa on huono pito-/vääntömomentti. Ohjaussekvenssi:

1. *0001*
2. *0011*
3. *0010*
4. *0110*
5. *0100*
6. *1100*
7. *1000*
8. *1001*

Vanhasta Canonin mustesuihkutulostimesta irrotettiin tarvittavat 2 askelmoottoria työtä varten (kuva 4).



Kuva 4. Askelmoottorit

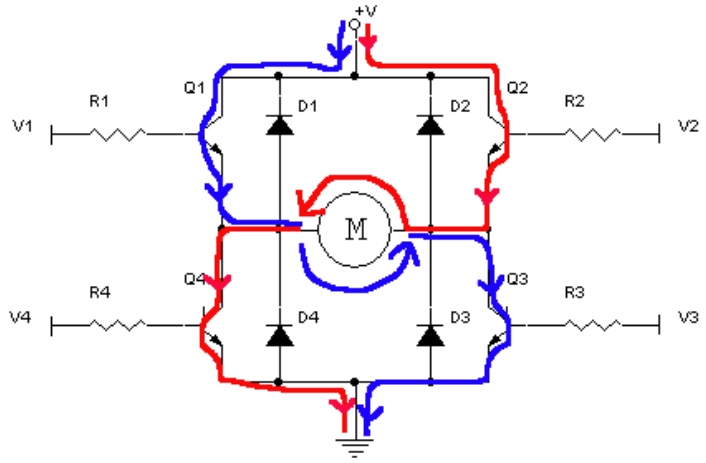
Moottorille menevien johtojen määrästä (4) voi päätellä, että moottorit ovat bipolaarisia. Bipolaarimoottoreissa on kaksi käämiä, joista yhdestä tulee aina kaksi johtoa. Käämien tunnistaminen tapahtuu yleismittarilla mittaamalla käämin resistanssi. Askelmoottorien käämien resistanssiksi mitattiin 9,5 Ohmia, jolloin 8 V-jännitteellä käämien ottamaksi virraksi tulee 0,84 A

Koska moottorin on pystyttävä pitämään itsensä paikallaan, jouduttiin mitoittamaan moottorin ohjausjännitteeksi vain 8 V. Kun moottori pitää itseään paikallaan, nousee virrankulutus maksimiinsa, joka on 8 voltilla 0,84 A. Ohjauspiirit pystyvät antamaan vain 1 A kanavaa kohden. Jatkuva 0,84 A:n syöttäminen kuumentaa ohjauspiirejä, joten niiden jäähdytys on hoidettava hyvin.

Askelmoottorin ohjaaminen vaatii tasajännitteen, jota katkotaan jokaiselle vaiheelle jaksollisesti. Askelmoottorin kytkeminen pelkästään suoraan AC- tai DC-jännitteelle ei saa moottoria pyörimään.

Bipolaarinen moottorinohjaus mahdollistaa virran kulkemisen molempiin suuntiin moottorin käämissä, ja näin saadaan parempi vääntövoima, mutta ohjauspiiri tulee monimutkaisemmaksi. Bipolaarimoottorin ohjaukseen tarvitaankin h-siltaa. Tämä mahdollistaa moottorin ajon molempiin suuntiin.

H-sillan toimintaperiaate moottorin pyörittämiseen on hyvin yksinkertainen. Transistorit toimivat kytkiminä. +V:hen tuodaan moottorin ohjaukseen haluttu jännite, ja kun esimerkiksi transistorien Q2 ja Q4 kannalle tuodaan kynnysjännitettä (0,7 V) suurempi jännite, niin transistorit alkavat johtaa ja moottori alkaa pyöriä. Vastaavasti jännitteen tuonti kannoille Q1 ja Q3 laittaa moottorin pyörimään toiseen suuntaan. Diodien tehtävä on suojata transistoreita jännitepiikeiltä. Vastuksilla R1->R4 rajoitetaan transistorien kannoille tuleva virta sopivan suuruiseksi (kuva 5).

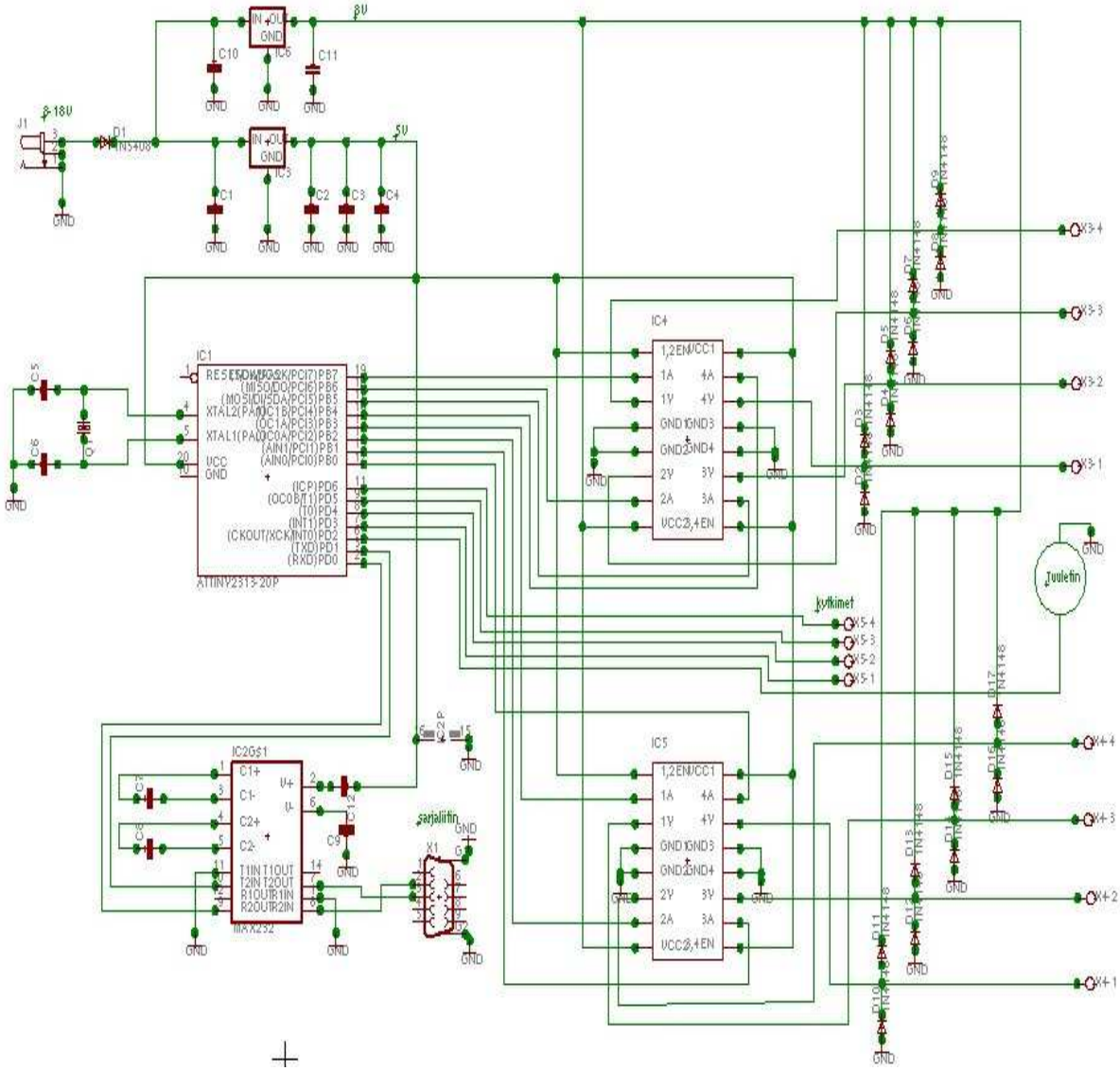


Kuva 5. H-silta

Ohjauspiiri olisi helppo rakentaa transistoreilla, vastuksilla ja diodeilla, mutta kahden moottorin ohjaukseen tarvitsisi 8 transistoria, jotka veisivät jo itsessään suuren tilan levystä. Niinpä päätettiin käyttää ohjaukseen L293B-kaksois-h-siltaa. Yhdellä piirillä voi ohjata vain yhtä moottoria, joten piirejä tarvittiin kaksi. L293B-piirissä on neljä kanavaa, joista jokainen voi antaa ulos jatkuvana 1 A ja hetkellisesti 2 A. Piirissä on myös sisäinen ylilämpösuojaus. B-mallissa ei ole sisäistä diodisuojausta, joten se tehtiin levylle erikseen 16 diodilla.

3.3 Kytkentä

Kytkennän piirtämiseen käytettiin Cadsoftin EAGLE-ohjelmistoa. Sen ilmaisversio riitti kattamaan tarvittavat ominaisuudet. EAGLE on saatavilla Windows, Linux ja jopa Mac-käyttöjärjestelmiin (kuva 6).

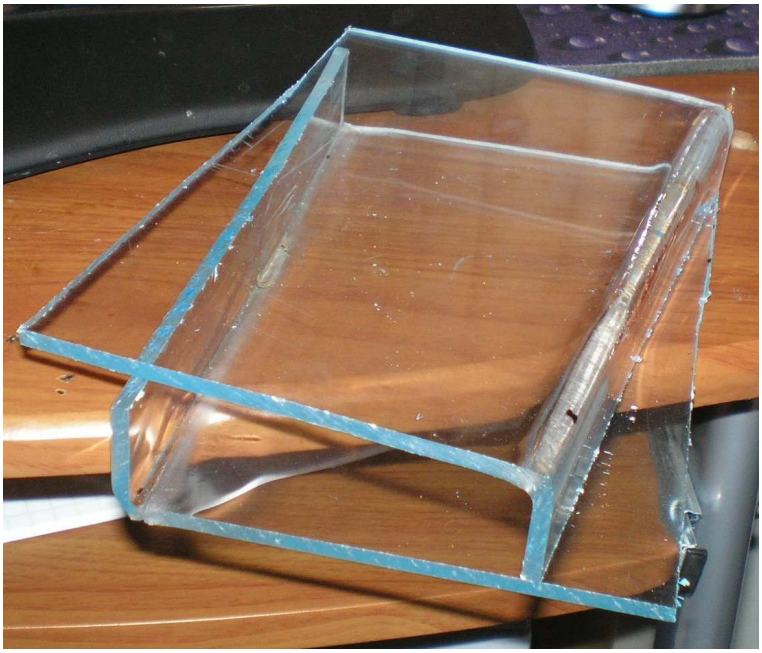


Kuva 6. Ohjainkortin kytkentä

3.4 Kotelo

Projektin ehkä haastavin osuus oli kotelon suunnittelu ja rakennus. Kameran piti liikkua sulavasti vaaka- ja pystysuunnassa. Mikro-ohjainkortin pitää mahtua kotelon sisälle ja johdotuksien läpiviennille piti keksiä sopivat kohdat, että ne eivät haittaisi kameran liikkumista.

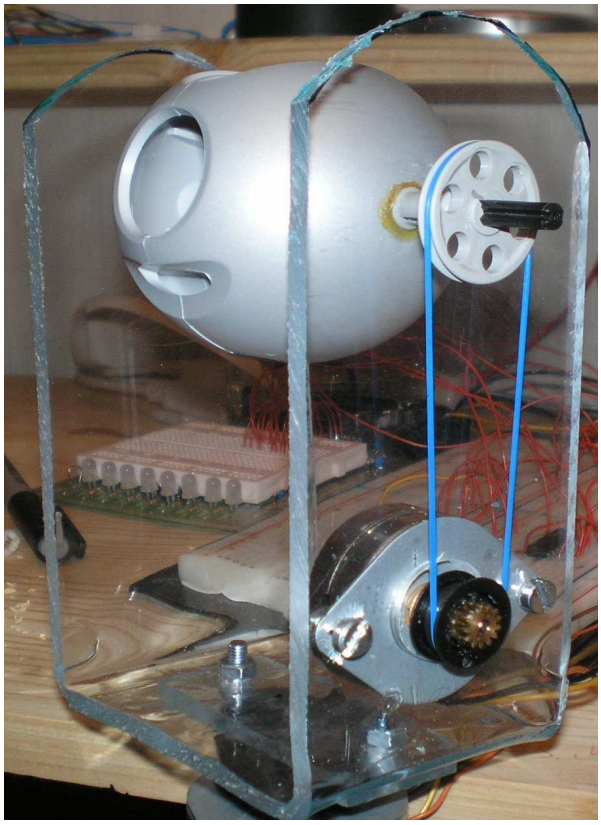
Kotelo materiaaliksi valittiin työstämisen helppouden vuoksi pleksi. Sitä pystyy helposti muotoilemaan kuumailmapuhaltimella, ja vastaavasti osat saa helposti kiinni toisiinsa kuumaliimalla (kuva 7).



Kuva 7. Kotelo

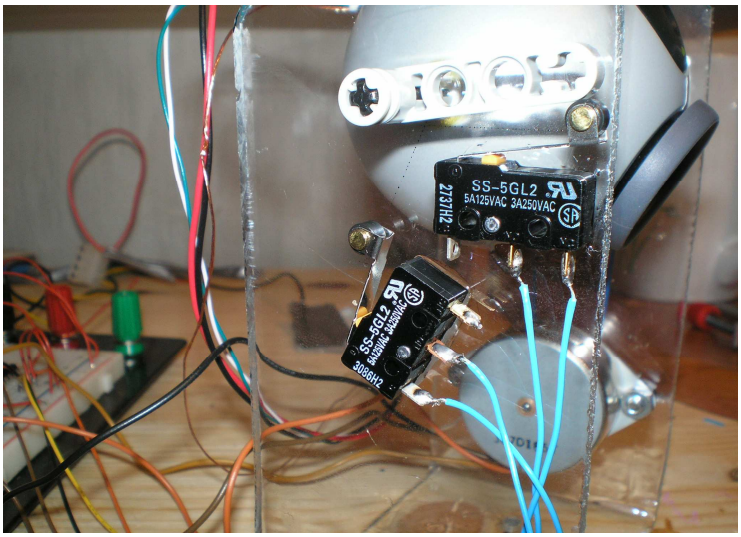
Seuraavaksi rakennettiin runko web-kameralle. Ensiksi purettiin web-kamera osiin ja lisättiin kylkiin kiinnikkeet, joihin saa akselit kiinni. Tarvittavat osat löytyivät jo puretusta mustesuihkutulostimesta sekä tekniikkalegoista.

Rungon taivutuksen jälkeen porattiin rungon alaosaan reikä askelmoottorin akselia varten. Yläosaan porattiin reiät, joiden läpi tulee kameran akselit. Voimansiirto askelmoottorilta kameraa pyörittävään akseliin tapahtuu kuminauhalla. Kuminauha on hihnapyörillä kiinni akseleiden päissä (kuva 8).



Kuva 8. Runko

Kameran ympäripyörimisen ja johtojen sotkeutumisen estämiseksi, asennettiin rajakytkimet rungon toiselle puolelle. Kytkimien avulla saadaan myös kameran asennosta tietoa (kuva 9).



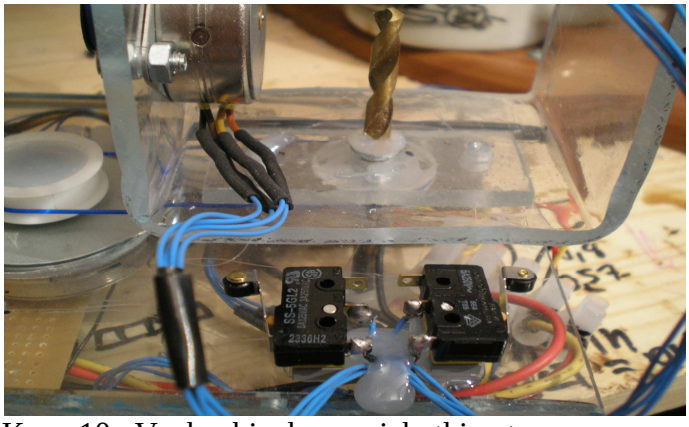
Kuva 9. Rajakytkimet

Kytkimen toinen pää on kytketty suoraan prosessorin nastaan, joka on alustettu tuloksi. Nastan lähtöarvo on alustettu loogiseksi ykköseksi (5 V). Vastaavasti kytkimen toinen pää on vedetty suoraan maahan. Aisan osuessa anturiin nastan jännite putoaa nollaan ja tila muuttuu loogiseksi nollaksi. Tilan muutoksia voi helposti lukea ohjemallisesti.

Seuraavaksi liitettiin web-kameran runko koteloon. Runko sijoitettiin kotelon reunaan, koska rungon akseli pitää viedä koko kotelon läpi tukevuuden varmistamiseksi. Akselina toimii kestävä 5 mm kovametalliporanterä. Akseli on tuettu kuumaliimalla kiinnitetyillä prikoilla, jokaisesta kohdasta, joten väljyyttä ei ole.

Askelmoottori, joka pyörittää kameraa vaakatasossa, sai sijoituspaikakseen kotelon keskikohdan. Voimansiirto moottorin akselista tapahtuu myös hihnapyörillä ja kuminauhalla. Moottorin viereen asennettiin vanhasta kannettavasta purettu tuuletin. Sen tehtävänä on jäähdyttää L293B-piirejä ja kierrättää kuumaa ilmaa pois kotelosta. Tuuletin toimii 5 V jännitteelle ja sen virran kulutukseksi on ilmoitettu 0,07 A. Tuulettimen ohjaus otettiin suoraan prosessorilta.

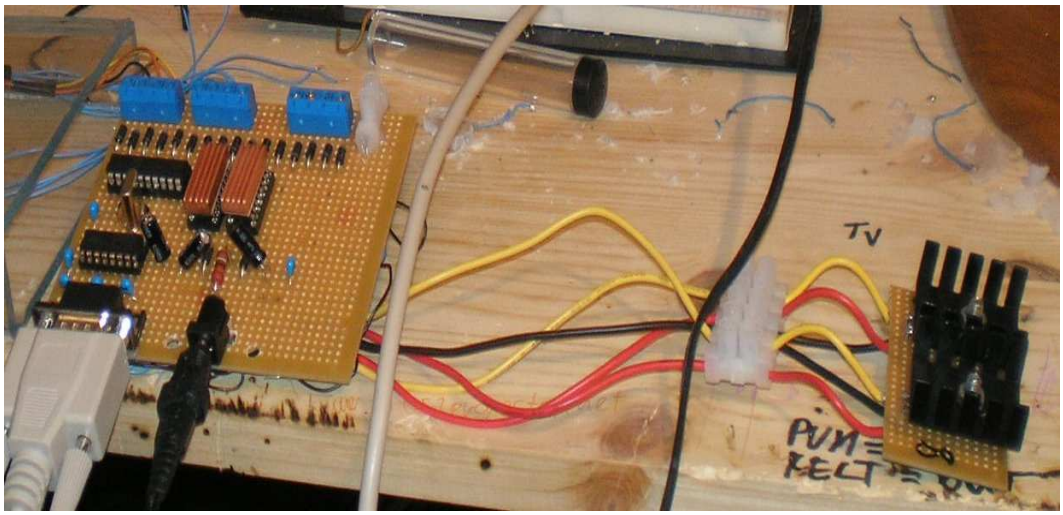
Seuraavaksi kiinnitettiin koteloon kaksi raja-anturia valvomaan kameran rungon pyörimistä vaakatasossa. Anturien toimintaperiaate on täysin sama kuin pystytason anturikytkennässäkin (kuva 10).



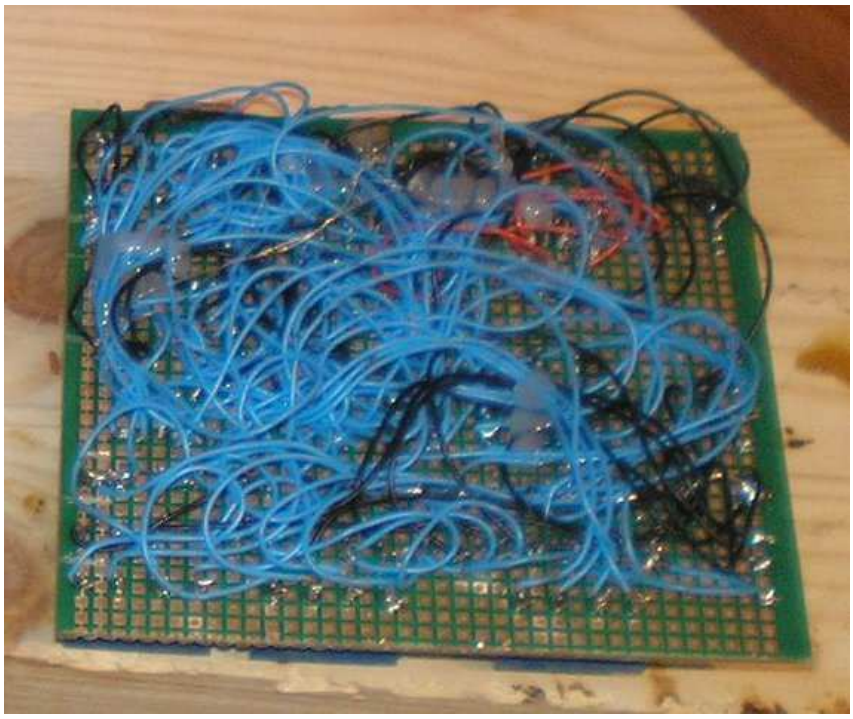
Kuva 10. Vaakaohjauksen rajakytkimet

3.5 Piirilevy

Ensiksi testattiin kytkennän toimintaa kytkentäalustalla, ja kun kytkentä saatiin toimimaan halutulla tavalla, koottiin lopullinen kytkentä reikälevylle. Valmis kotelo asetti vaatimukseksi, että levy ei saanut olla isompi kuin 10 cm x 10cm, koska koteloon kiinnitetty moottori vei tilaa jo sisäpuolelta. Komponenttien sijoittelussa piti olla tarkka. Tilan puutteen takia päätettiin sijoittaa molemmat jänniteregulaattorit erilliselle levylle jäähdytysripoineen ja yhdistää ne sitten riviliittimillä itse päälevyyn (kuva 11 ja 12).

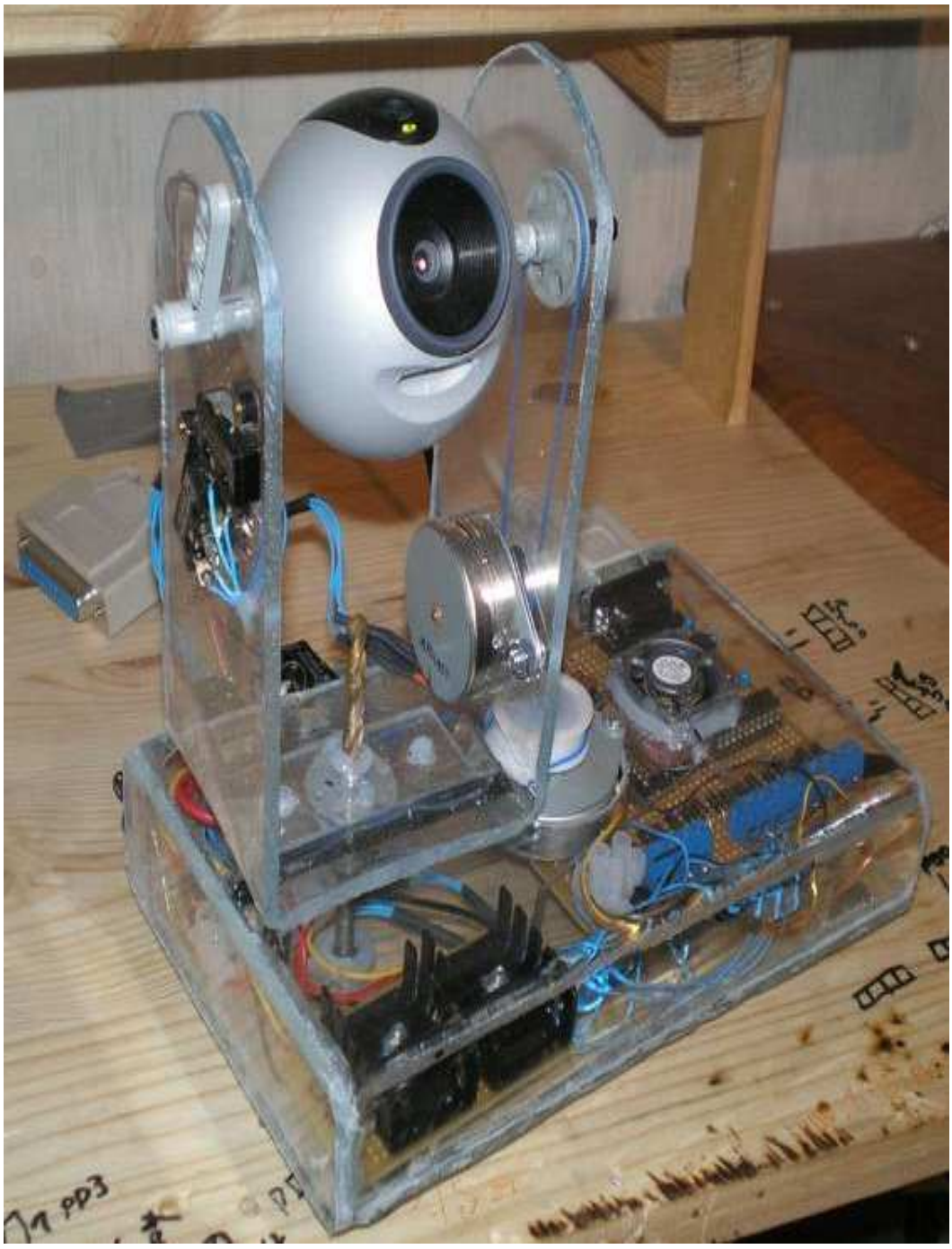


Kuva 11. Valmis kytkentä levyllä



Kuva 12. Johdotukset

Kameran liikkumisen edistämiseksi kuorittiin johdon eristettä noin 30 cm:n matkalta. Johto kiinnitettiin tukevasti metallilevyllä ja kahdella ruuvilla kotelon takaosaan. Kotelon takaosaan porattiin reikä jännitteensyötölle sekä jyrsittiin sopiva kolo sarjaliitintä varten. Piirilevy on helposti vedettävissä kotelon kyljestä sen verran irti, että prosessorin saa irrotettua ohjelmoitavaksi (kuva 13).



Kuva 13. Järjestelmä toimintakunnossa

4 PROSESSORIN OHJELMOINTI

4.1 Laitteisto

Prosessorin ohjelmointi suoritettiin EXB2313-kehityskortilla. Ohjelmointiin tarvittiin vielä ohjelmointikaapeli, jonka rakennusohjeet löytyivät helposti Internetistä. Dontronics DT006 -tyypin puskuroitu eli aktiivijohto rakennettiin ohjelmointia varten.

4.2 Ohjelmisto

Prosessorin ohjelmoimiseksi tarvitsi asentaa vielä muutama ohjelmisto. Kaikki käytetyt ohjelmat perustuvat avoimeen lähdekoodiin, joten ne ovat täysin ilmaisia. Ohjelmat löytyvät suoraan Fedoran pakettienhallinnasta ja ne on siksi helppo asentaa. Ohjelmointiin käytettävät ohjelmat ovat komentorivipohjaisia.

- avr-gcc-kääntäjä
- avr-libc, C-kirjasto Atmelin AVR- mikro-ohjaimille
- avrdude, ohjelmointiohjelma
- avr-objcopy, muuttaa käännetyn ohjelman hex-tiedostoksi

Lähdekoodin siirtämiseksi prosessorille päätettiin työn helpottamiseksi koodata oma Makefile. Makefile on aputiedosto, jolla helpotetaan ohjelmien kääntämistä. Tällä korvataan pitkien komentojen kirjoittaminen komentoriville.

4.3 Makefile /10/

Makefile koostuu makroista, kohteista, riippuvuuksista sekä säännöistä. Makro määritellään merkinnällä **NIMI=sisältö**, ja sen jälkeen kaikki **$(NIMI)**-viittaukset korvautuvat makron sisällöllä.

Makefilen kääntäminen tapahtuu *make*-komennolla. Seuraavana on esitelty prosessorin ohjelmointiin tehdyn Makefilen toimintamalli.

Aluksi määritetään makrot. Makromäärittelyistä on se hyöty, että MakeFilen muuttaminen tulee helpommaksi. Esimerkiksi jos vaikka halutaan kääntää tiedostot debug optiolla *-d,* niin optio lisätään CFLAGS-makroon sen sijaan, että se kirjoitettaisiin erikseen jokaiseen sääntöön. Alla on listattuna MakeFilessä käytetyt makrot.

CC, avr-gcc-kääntäjän sijainti
CFLAGS, käännöksessä tarvittavat optiot
OBJ2HEX, avr-objcopy-ohjelman sijainti
AVRDUDE, avrdude-ohjelman sijainti
TARGET, haluttu nimi tiedostolle

RM, sisältää *rm -f* unix-komennon, joka on tiedoston poistokomento force-optiolla.

Seuraavaksi merkitään kohdetiedosto ja riippuvuustiedosto sekä niille säännöt. Kohteen merkinnän jälkeen tulee aina kaksoispistemerkintä, jonka jälkeen kirjoitetaan riippuvuudet. Säännöt voivat olla mitä tahansa unix-komentoja. Jotta sääntö tulkittaisiin säännöksi, on sen alussa oltava täsmälleen yksi tabulaattorilyönti. Välilyönnit tuottavat virheilmoituksen.

Ensimmäinen kohde Makefilessä on *program*, josta ohjelman rakentaminen aloitetaan. Ensin tarkistetaan, onko kohteelle riippuvuuksia. Sen jälkeen tarkistetaan, ovatko ne myös kohteina Makefilessa. Jos kohteita löytyy, rakennetaan ne ensin oikeassa järjestyksessään.

Makefilessä on kohde *clean*, jolla ei ole riippuvuustiedostoja. Se ei tarvitse minkään muiden sääntöjen suorittamista, vaan sen omat säännöt suoritetaan aina. Kohde *clean* on Makefilen lopussa, ja se tuhoaa käännöksessä syntyneet objekti- ja hex-tiedostot. Seuraava make-komento siis alkaa aina tyhjältä pöydältä.

Lopullisen Makefilen koodi:

```
CC=/usr/bin/avr-gcc
CFLAGS=-g -Os -Wall -mcall-prologues -mmcu=attiny2313 -std=gnu99
OBJ2HEX=/usr/bin/avr-objcopy
AVRDUDE=/usr/bin/avrdude
TARGET=oma_serial
RM=rm -f

program : $(TARGET).hex
$(AVRDUDE) -p attiny2313 -P /dev/parport0 -c stk200 -u -U flash:w:$(TARGET).hex

%.obj : %.o
$(CC) $(CFLAGS) $< -o $@

%.hex : %.obj
$(OBJ2HEX) -R .eeprom -O ihex $< $@

clean :
$(RM)*.hex *.obj *.o
```

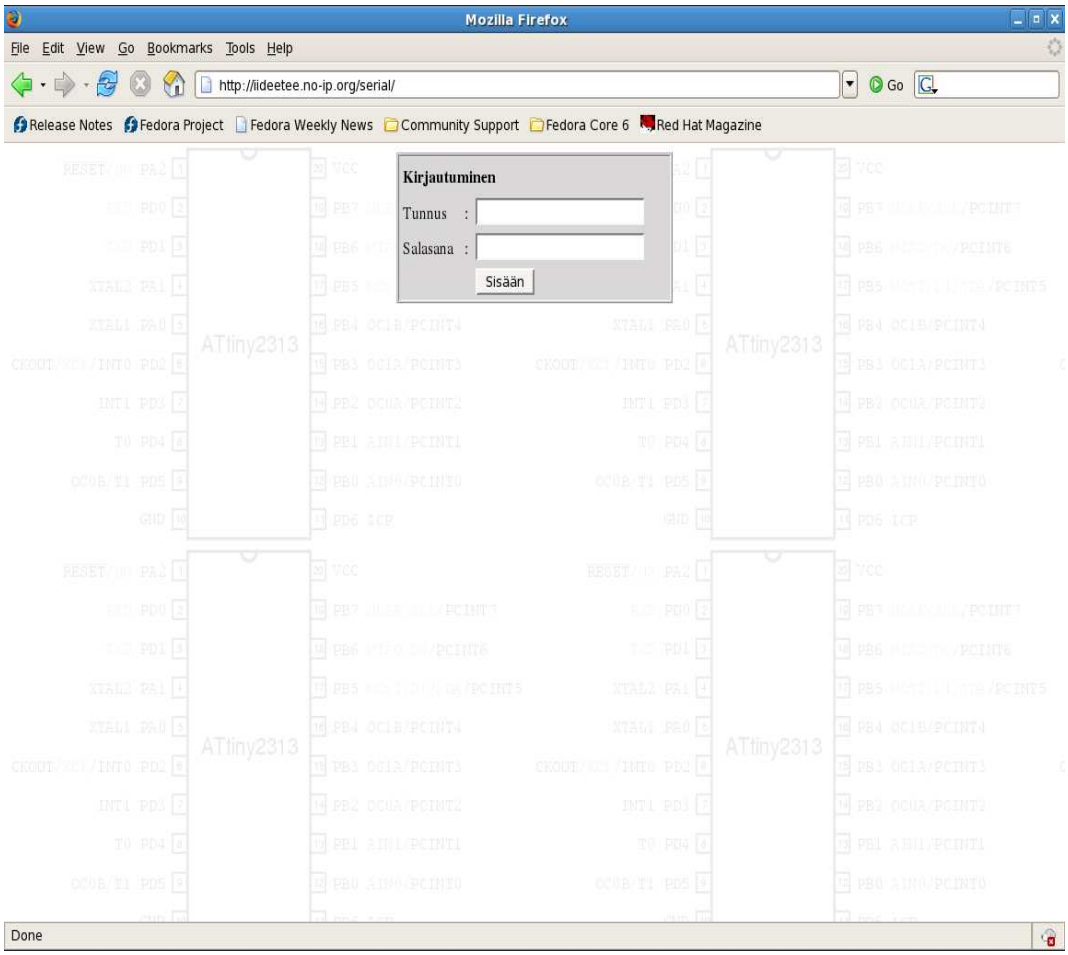
5 KÄYTTÖLIITTYMÄ

Käyttöliittymä toteutettiin PHP- ja XHTML-kielillä. Kirjautumisen tunnus- ja salasanakysely hallintasivulle tehtiin MySQL-tietokantakyselyllä. Hallintasivujen kautta ajettavat käskyt toteutettiin Linuxin Shell-skripteillä.

5.1 Kirjautuminen

Yhdistettäessä palvelimelle näytetään käyttäjälle ensiksi XHTML:llä tehty sisäänkirjautumislomake. Lomakekenttiin annetut tiedot (tunnus ja salasana) lähetetään tarkistettavaksi POST-metodia käyttäen koodissa määrätylle PHP-sivulle. Salasanan ja käyttäjätunnuksen ollessa oikeita käyttäjä ohjautuu hallintasivulle. Väärä tunnus tai salasana saa käyttäjän ohjautumaan virheilmoitussivulle (kuva 14).

Käyttäjän kirjautuessa onnistuneesti hallintasivulle käynnistyy sessio. Sessiomuuttujiin rekisteröidään annettu käyttäjänimi ja salasana. Sessio on tekniikka, jonka avulla voidaan säilyttää tietoa yhden session ajan tai niin kauan, kunnes käyttäjä sulkee selaimen. Sessiot eroavat evästeistä sillä tavalla, että sessiotieto säilytetään palvelimella, eikä sitä välitetä selaimelle. Sessiotekniikkaa käyttämällä käyttäjän ei tarvitse aina kirjoittaa salasanoja uudestaan kirjautuakseen sivulle, vaan voi asioida välillä muillakin sivuilla. Jokaisessa PHP-sivussa, missä käytetään sessioita, täytyy koodin alussa kutsua ensin *session_start()*-funktiota. Funktio tutkii, löytyykö jo käynnissä oleva sessiotunniste evästeistä. /11/

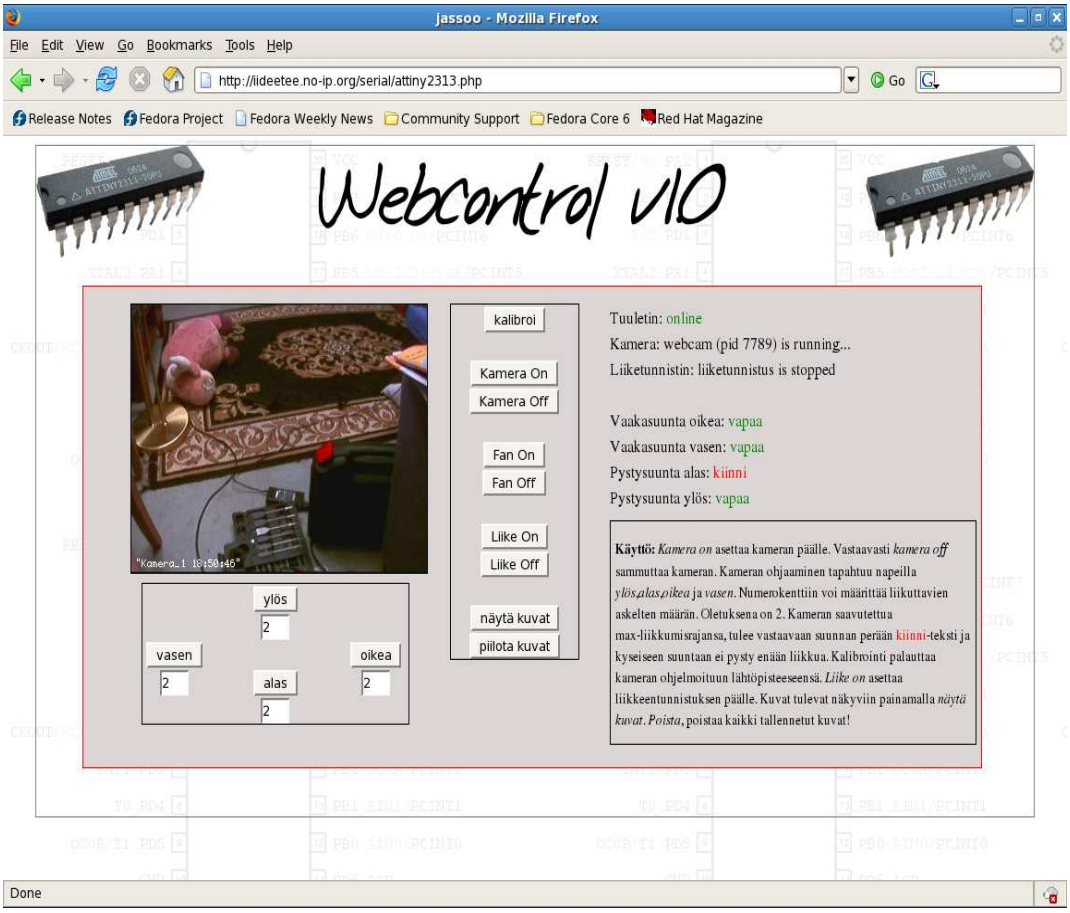


Kuva 14. Kirjautumislomake

5.2 Hallintasivu

Hallintasivulta ohjataan itse laitteistoa. Mahdollisia toimintoja on kameran kalibrointi, jossa kamera hakee ohjelmallisesti määrätyn kohdan automaattisesti. Kameran pystyy sulkemaan ja laittamaan päälle hallintasivulta. Kotelon ilmankierrosta huolehtivan tuulettimen voi myös sulkea ja laittaa päälle. Ohjelmallisesti toimivan liiketunnistuksen tilaa voi ohjata sekä selata tai poistaa mahdollisia liiketunnistuksen tallentamia kuvia (kuva 15).

Kameraa pystyy ohjaamaan vaaka- sekä pystytasossa, ja käyttäjän on mahdollista määrätä liikuttavien askelien määrä väliltä 0-9. Oletusarvona on 2 askelta.



Kuva 15. Hallintasivu

5.3 Tilatiedot

Sivusto kertoo reaaliajassa toimintojen tilan. Käännettäessä kameraa niin, että se saavuttaa rajakytkimen, vastaavaan suunnan teksti *vapaa* muuttuu tekstiksi *kiinni* ja käämit muuttuvat jännitteettömiksi. Tila muuttuu heti takaisin *vapaaksi*, kun anturi jälleen vapautuu. Tuulettimen tilatieto päivittyy myös suoraan hallintasivulle.

Tämän mahdollistaa BASH-tulkille kirjoitettu sarjaportin ”nuuskinta”-skripti. Prosessori on ohjelmoitu lähettämään sarjaporttiin I/O-porttien tilat aina, kun kysely tulee.

```
#/bin/bash
while true          # Ikuinen silmukka
do
# Luetaan unix-komennolla read sarjaportista merkkejä
# muuttujaan RIVI, kunnes tulee rivinvaihtomerkki.
read RIVI < /dev/ttyS0
# Päivitetään muuttujan RIVI sisältö ilman rivinvaihtomerkkiä
# Kirjoitetaan sarjaportista tullut merkki tiedostoon serial.txt
echo -n $RIVI > /home/iideetee/Desktop/lighttpd/serial/serial.txt
done
```

Vastaanotettu ASCII-merkki käännetään PHP-koodissa binäärimuotoon, josta bitit erotetaan vielä omiin lokeroihinsa. Jokaisen lokeron bitti kuvaa yhden I/O:n tilaa, joten tilat on tästä helppo lukea hallintasivun näytölle.

Jotta skripti toimisi, pitää sarjaportti ensiksi alustaa samoilla arvoilla kuin mikro-ohjainkortin sarjaportti. Sarjaportin alustus tehdään komennolla
stty -F /dev/ttyS0 9600 cs8 -parenb -cstopb -icanon

Stty on komento, millä voidaan muuttaa terminaalin pääteasetuksia, jossa optio –F kertoo, että luku tapahtuu */dev/ttyS0* -laitteesta näppäimistöltä luvun sijaan. Siirtonopeus on 9600 bit/s. Databittien määrää kuvaa *cs8*, joka saa vielä lisäoptioina pariteettitarkistuksen (pois), yksi stop-bitti, sekä ei-kanonisen tulostuksen *-parenb -cstopb -icanon*. /12/

Kameran ja liiketunnistuksen käyttöä varten tehtiin erilliset käynnistysohjelmat */etc/init.d* -hakemistoon. Kaikki Fedoran palvelujen käynnistysskriptit löytyvät kyseisestä hakemistosta. Käynnistysskriptit tunnistavat käytännössä aina parametrit *start, stop, restart* ja *status.* Käynnistysohjelma sisältää linkin käynnistettävään ohjelmaan.

Kameran ja liiketunnistuksen tilatiedot hallintasivulle tulevat PHP-koodissa ajetun kyselyn kautta. Tilakysely tapahtuu PHP:n *system*-funktiolla. Funktiolla pystyy suorittamaan komentorivipohjaisia käskyjä, sekä tulostamaan tuloksen näytölle. /13/
PHP-koodissa komento:
system('sudo /etc/init.d/liiketunnistus status');

System on itse funktio, jonka sisältö suoritetaan. *Sudo* antaa oikeuden komentojen suorittamiseen pääkäyttäjänä Linux-järjestelmissä. Ajettava komento on */etc/init.d/liiketunnistus status*. Komennon suoritus tulostaa hallintasivun näytölle liiketunnistus skriptin tilan. Jos skripti on käynnissä, näytölle tulostuu *Liiketunnistin: liiketunnistus (pid 23938) is running...*

Pid on prosessin yksilöllinen id-numero. Vastaavasti *stop*-komento funktion lopussa suorittaa *kill*-komennon, joka saa attribuuttina prosessin pid-numeron ja näytölle tulostuu *Liiketunnistin: liiketunnistus is stopped*

5.4 Liiketunnistus

Liiketunnistukseen on saatavilla ilmaisia tai maksullisia ohjelmia. Päätettiin tehdä itse oma skripti liikkeentunnistukseen. Ohjelma tehtiin BASH-tulkille tarkoitetulla skriptillä. Ohjelma tarvitsee toimiakseen ImageMagick-ohjelmiston. Ohjelmisto löytyy Fedoran pakettienhallinnasta ja on ilmainen. Asennus tapahtuu komennolla *yum install imageMagick*

ImageMagick on ohjelmisto, jolla voi tehdä kuvatiedostolle periaatteessa mitä vain. Skriptissä käytettiin ohjelmiston *compare*-käskyä. /9/

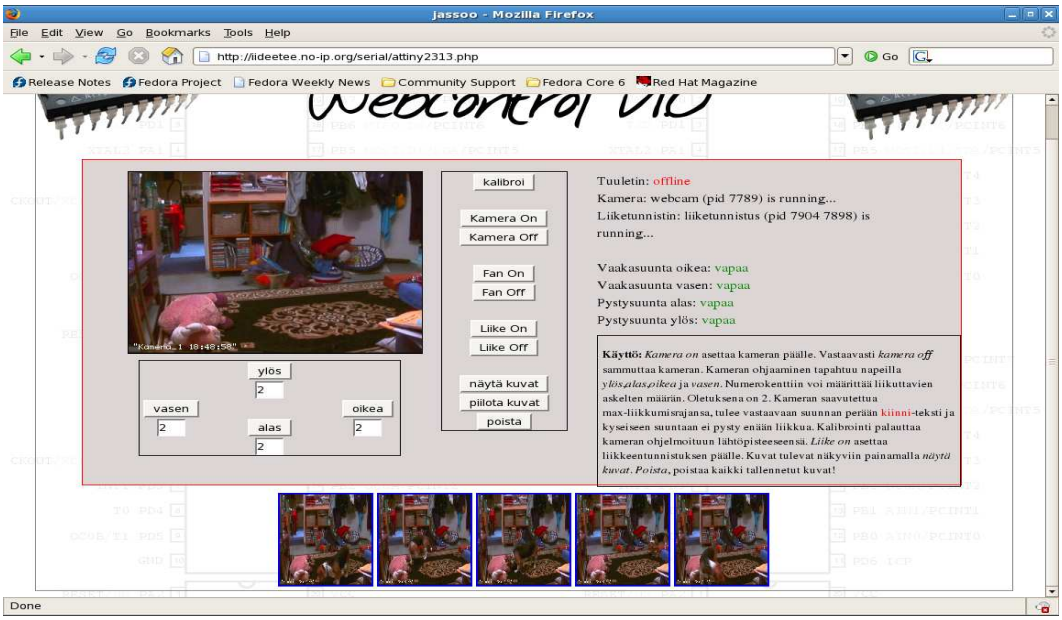
```
#!/bin/sh
POLKU=/home/iideetee/Desktop/lighttpd/serial/kamera
while(true)
do
cp $POLKU/webcam.jpg $POLKU/tmp/webcam2.jpg
sleep 10
compare -metric PSNR $POLKU/webcam.jpg $POLKU/tmp/webcam2.jpg null
>               tulos
tulos2=`head -c 4 tulos`
tulos2=${tulos2/./}
if test $tulos2 -lt 300
then
aika=`date +%_T_%F`
cp $POLKU/webcam.jpg $POLKU/save/$aika.jpg
fi
done
```

Edellinen koodi alkaa pyöriä ikiloopissa, kun liiketunnistus laitetaan päälle hallintasivulta. Ensiksi kopioidaan *kamera*-kansiosta webkameralta tullut kuva *webcam.jpg tmp*-kansioon ja nimetään se *webcam2.jpg*:ksi. Tämän jälkeen odotetaan noin 7 sekuntia, jotta uusi kuva ehtisi tulla kameralta *kamera*-kansioon. Seuraavaksi verrataan molempien kuvien signaalikohinasuhteen huippuarvoa ja kirjoitetaan saatu tulos *tulos*-nimiseen tiedostoon.

Kirjoitetaan *tulos2*-muuttujaan saadusta PSNR-luvusta ensimmäiset 3 lukua ja poistetaan erottimena toimiva piste. PSNR-luku on sitä isompi mitä parempi signaalikohinasuhde on. Kameralla otetun kahden peräkkäisen kuvan PSNR oli noin 31-33 dB, ja muutamankin pixelin muutos kuvassa sai luvun putoamaan. Testien jälkeen päädyttiin käyttämään vertailuarvona 30 dB.

Jos kahden peräkkäisen kuvan PSNR-suhde on pienempi kuin 30 dB, kirjoitetaan *aika*-nimiseen muuttujaan päivämäärä ja tarkka aika sekunnilleen.

Tämän jälkeen se kuva, missä oli ollut muutosta edelliseen kuvaan, kopioidaan *save*-nimiseen kansioon ja kuvalle annetaan nimeksi vertailuhetkellä saatu aikaleima. Näin kuvia on helppo etsiä päivämäärän ja kellonajan mukaan (kuva 16).



Kuva 16. Tallennetut kuvat

Hallintasivun *näytä kuvat* -nappia painettaessa aukeaa thumbnail-galleria kuvista, joita liiketunnistusohjelma on tallentanut. Gallerian saa vastaavasti pois näkyvistä *piilota kuvat* -napilla. Klikattaessa pikkukuvaa aukeaa se täydessä koossa uudelle sivulle (kuva 17). *Poista*-nappi poistaa kaikki tallennetut kuvat palvelimen kiintolevyltä.

Kuva 17. Suurennettu kuva

6 PROSESSORIN OHJELMA

Ohjelman alkuun tehdään esikäännökseen tarvittavat määritykset sekä otetaan käyttöön tarvittavat kirjastot.

```
#define __AVR_ATtiny2313__
#define F_CPU 12000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
```

Mikrocontrollerin tyypin määritys on pakollinen, koska sen perusteella valitaan *avr/io.h*:n prosessorikohtaiset määritykset. F_CPU:hun laitetaan kellotaajuus hertzeissä. Tämän avulla *util/delay.h*-kirjaston viivefunktion viive saadaan laskettua oikean kestoiseksi.

6.1 Sarjaliikenne /5 ; 1/

Attiny2313:ssa on sisäänrakennettu UART (Universal Asynchronous Receiver Transmitter) sarjaliikennekommunikointiin. Datan lähetyksen ja vastaanoton sallimiseksi pitää initUART-funktion avulla tehdä alla olevat määritykset.

```
void InitUART (unsigned char baudrate)
{
UBRRL = baudrate;
UBRRH = (baudrate >> 8);
/* sallitaan lähetys ja vastaanotto */
UCSRB = (1 << RXEN) | (1 << TXEN);
/* 8 data bittiä, 1 stop bitti ja ei pariteettia */
UCSRC = (1 << UCSZ1) | (1 << UCSZ0);
}
```

InitUART-funktiossa tuodaan parametrinä pääohjelmassa määritetty *baudrate.* Operoidessa asynkronisessa normaalimoodissa *baudrate* lasketaan kaavalla:

$$UBRR = \frac{fosc}{16(BAUD)} - 1 \qquad (1)$$

Kaavassa *fosc* on oskillaattorin kellotaajuus, BAUD on bittinopeus ja UBRR on 12-bittinen rekisteri, joka jakaa oskillaattorikellon UBRRL ja UBRRH-rekistereihin. UBRRH sisältää neljä eniten merkitsevää bittiä ja UBRRH loput 8 vähiten merkitsevää bittiä.

Käyttämällä 12 MHz kidettä, ja kun baudinopeudeksi halutaan 9600 bit/s, kaavalla saadaan UBRR-arvoksi 77.125, josta käytettävä arvo on 77.

Kaavaa käyttämällä käänteisesti saadaan uusi BAUD-arvo kun UBRR-arvoksi sijoitetaan 77.

$$BAUD = \frac{fosc}{16(UBRR + 1)} \qquad (2)$$

Bittinopeudeksi saadaan 9615,38 bit/s
Jakamalla saatu bittinopeus halutulla nopeudella (9600bps), saadaan virheprosentti.

$$Virheprosentti = \left(\frac{baudrate}{baudrate_{haluttu}} - 1 \right) * 100\% \qquad (3)$$

Edellä lasketuilla arvoilla virheprosentiksi saadaan 0.16 %. /5, s. 138/

Lähetys ja vastaanotto saadaan päälle laitattamalla *UCSRB*-rekisteristä *RXEN* ja *TXEN* bitit ykkösiksi.

UCSRB = (1 << RXEN) | (1 << TXEN);

UCSRC-rekisteriin määritetään käytettävä kehysmuoto. Laittamalla *UCSZ1*- ja *UCSZ0*-bitit ykkösiksi saadaan kehysmuoto koostumaan 8 databitistä. UMSEL-bitin ollessa 0 käytetään asynkronista moodia. Bittien *UPM0* ja *UPM1* ollessa samanaikaisesti nollia pariteettimoodi on pois päältä. *USBS*-bitin ollessa nolla, käytetään yhtä stop-bittiä (taulukko 1).

Bit	7	6	5	4	3	2	1	0	
	–	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL	UCSRC
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	1	1	0	

Taulukko 1. USART hallinta- ja tilarekisteri

Pääohjelmassa alustetaan DDRB-suuntarekisteri lähtötilaan (1), joten prosessorin nastat PB0-PB7 on lähtöinä. PORTB:n nastoilla ohjataan askelmoottoreita. DDRD-suuntarekisteri on vain 7 bittinen. Sen kolmanneksi viimeinen bitti PD2 alustetaan lähdöksi ja muut tuloiksi. PD2 nasta on kytketty suoraan kotelon tuulettimeen. PD3-PD6-nastat, jotka on alustettu tuloiksi, on kytketty suoraan kotelon neljään eri rajakytkimeen.

PORTD:n nastat (PD3-PD6), jotka on kytketty kytkimiin, alustetaan loogisiksi ykkösiksi (5 V). Raja-kytkimen toinen pää on kytketty maahan, joten kytkimen mennessä pohjaan, putoaa portin jännite nollaan volttiin ja tulo luetaan loogiseksi nollaksi.

Datatavun lähetys ja vastaanotto on toteutettu kahdella erillisellä aliohjelmalla, joita kutsutaan tarvittaessa. Alla on ohjelmassa käytetyt aliohjelmat datan vastaanottamiseen ja lähettämiseen.

```
unsigned char ReceiveByte (void)
{
while (!(UCSRA & (1 << RXC)));
return UDR;
}

void TransmitByte (unsigned char data2)
{
while (!(UCSRA & (1 << UDRE)));
UDR = data2;
}
```

ReceiveByte-vastaanottoaliohjelmassa odotetaan datan tuloa, kunnes RXC-lippu (Receive Complete) nousee. Tämän jälkeen palautetaan saatu merkki takaisin pääohjelmaan, jossa switch-case rakenteella toteutettu ohjelma suorittaa saatua merkkiä vastaavan tapahtuman.

Vastaavasti lähetysaliohjelmassa *TransmitByte* odotetaan lähetyspuskurin tyhjenemistä seuraamalla UDRE-lipun (USART Data Register Empty) tilaa. UDRE-bitti nousee 1-tilaan kun UDR-rekisteri on tyhjä ja se on valmis vastaanottamaan uuden merkin. Bitti nollaantuu, kun uusi merkki on kirjoitettu UDR-rekisteriin.

Ohjelmassa lähetysaliohjelmalla lähetetään porttien tilatietoa sarjaliikenneväylää pitkin pc:lle, josta se erillisten ohjelmien avulla muutetaan luettavaan muotoon web-sivustolle.

6.2 Moottorien ajo

Askelmoottorien ajo on toteutettu aliohjelmilla, joita sitten ikiloopissa olevalla switch-case rakenteella kutsutaan vastaanotetun merkin mukaan. Hallintasivulta käskettäessä kameraa siirtymään yhden askeleen ylöspäin lähetetään ASCII-merkki 'c' sarjaporttiin. Prosessorin vastaanottoaliohjelma lukee merkin ja lähettää sen pääohjelmaan, jossa switch-case rakenne hyppää suorittamaan kohtaa *case 'c*'.

```
case 'c'        :
if(PIND != 0x6B && PIND != 0x6F)
{
ylos();
TransmitByte(PIND);
TransmitByte(0x0A);
break;
}
```

```
TransmitByte(PIND);
TransmitByte(0x0A);
PORTB = 0x00 ;
break;
```

Ensiksi tarkistetaan, onko rajakytkin pohjassa, jos ei ole, ajetaan yksi askel ylöspäin ja lähetetään PORTD:n inputnastojen tilat sekä rivinvaihtomerkki sarjaportin välityksellä pc:lle. Tämän jälkeen hypätään case-rakenteesta pois. Vastaavasti jos raja-kytkin on jo pohjassa, lähetetään inputnastojen tilatiedot PC:lle, jotta hallintasivulle saadaan käyttäjälle ilmoitus rajan saavuttamisesta.

Tämän jälkeen kytketään askelmoottorien käämit jännitteettömäksi, jolloin ohjaaminen rajakytkimen suuntaan ei enään onnistu.

YHTEENVETO

Valvontalaite saatiin toimimaan loppujen lopuksi halutulla tavalla. Työ oli erittäin haastava, mutta antoisa. Koska aikaisempaa kokemusta ATMEL:in AVR-prosessoreista ei ollut, nousi Internet datalehtien ja ohjeiden haussa tärkeään asemaan. Työtä tehdessä tuli tutuksi monia uusia vapaan lähdekoodin ohjelmistoja, joista ei ollut aikaisempaa kokemusta. Näistä tärkeimpiä Eagle, kiCAD ja Avrdude.

Laitteen kustannukset nousivat melko suuriksi, koska mitään työkaluja ei ollut valmiina, vaan ne piti ostaa. Itse piirilevyn komponentit tulivat maksamaan noin 25 euroa, mutta muut osat löytyivät vanhoista tulostimista. Laitteisto olisi ollut paljon helpompi rakentaa, jos käytetyt askelmoottorit olisivat olleet unipolaarisia. Näin ollen ohjaukseen olisi tarvittu vain halpa Darlington-piiri (ULN2003). Koska löydetyt moottorit olivat bipolaarisia, tuli koko projekti hankalammaksi toteuttaa, koska moottorien pyörimiseksi molempiin suuntiin tarvittiin h-silta.

Itse rakennetulla h-sillalla olisi saanut ulos isompia virtoja, mutta tilan ahtauden takia käytettiin valmiita piirejä. Ongelmaksi koitui piirien kuumeneminen. Moottorin pitäessä kameraa paikallaan, eli kun yksi vaihe on jännitteen alaisena kokoajan, alkaa ohjauspiiri tietenkin kuumeta. Mittaamalla saatiin käämin kuluttamaksi virraksi pitotilassa 0,84 A. Ohjauspiiri pystyy syöttämään yhtäjaksoisesti 1 A, mutta jäähdytys on hoidettava jotenkin. Päädyttiin liimaamaan passiivinen kuparisiili piirien päälle ja liittämään tuuletin kotelon päälle kierrättämään ilmaa sisäpuolella.

Ongelmaa ei olisi ollut, mikäli ohjaus olisi toiminut niin, että kamera vain siirtyisi johonkin kohtaan ja sen jälkeen käämit laitettaisiin jännitteettömäksi.

Tässä tapauksessa kameran tarkka paikallistaminen olisi mahdotonta. Laitteen voimansiirto olisi tulevaisuudessa tarkoitus muuttaa hihna- tai ketjuvetoiseksi. Näin estetään kuminauhan venymisessä syntyvä paikan vääristymä.

Hallintasivun kautta pystyisi määrittämään suunnan, jota kuvataan, esim. ovi, ikkuna tai eteinen. Kameran saisi ohjelmallisesti aina kuvaamaan haluttua paikkaa, vaikka jollain tietyllä sekvenssillä.

Lisäämällä kytkentään Ethernet-piiri ja kirjoittamalla mikrokontrollerille reaaliaikakäyttöjärjestelmä, saadaan laite toimimaan itsenäisesti. Tällöin laite saadaan toimimaan palvelimena, kun vielä lisätään kytkentään USB-väylä, saadaan laitteeseen kytkettyä WLAN-kortti.

LÄHTEET

1. HowTO: Serial communication beetween PC and Atmel attiny2313. [www-sivu]. [Viitattu 12.2.2008] http://www.windmeadow.com/node/25

2. How to set up a linux webcam server. [www-sivu]. [Viitattu 12.2.2008] http://nulldigital.net/articles/how_to_set_up_a_webcam_server.htm

4. L293B datasheet [www-sivu]. [Viitattu 12.2.2008] http://pdf1.alldatasheet.com/datasheet-pdf/view/22430/STMICROELECTRONICS/L293B.html 25 s.

5. ATTINY2313 datasheet [www-sivu]. [Viitattu 12.2.2008] http://pdf1.alldatasheet.com/datasheet-pdf/view/80317/ATMEL/ATTINY2313/L293B.html 211 s.

6. ST232ACN datasheet [www-sivu]. [Viitattu 12.2.2008] http://pdf1.alldatasheet.com/datasheet-pdf/view/23684/STMICROELECTRONICS/ST232ACN.html 10 s.

7. L7805CV datasheet [www-sivu]. [Viitattu 12.2.2008] http://pdf1.alldatasheet.com/datasheet-pdf/view/22634/STMICROELECTRONICS/L7805CV.html 34 s.

8. Stepperworld [www-sivu]. [Viitattu 12.2.2008] http://www.stepperworld.com/Tutorials/pgBipolarTutorial.htm

9. ImageMagick [www-sivu]. [Viitattu 12.2.2008] http://imagemagick.org/script/command-line-options.php

10. Make-a tutorial www-sivu]. [Viitattu 12.2.2008] http://www.eng.hawaii.edu/Tutor/Make/

11. PHP Tutorial-Session [www-sivu]. [Viitattu 12.2.2008] http://www.tizag.com/phpT/phpsessions.php

12. Unix Manual Page for stty
[www-sivu]. [Viitattu 12.2.2008]
http://www.scit.wlv.ac.uk/cgi-bin/mansec?1+stty

13. PHP: system-Manual
[www-sivu]. [Viitattu 12.2.2008]
http://fi.php.net/system

LIITTEET

1. Prosessorin C-kielinen koodi, 7 sivua

2. Kirjautumiskyselyn Xhtml-koodi, 1 sivu

3. Kirjautumiskyselyn tarkistuksen PHP-koodi, 1 sivu

4. Hallintasivun Xhtml/PHP-koodi, 7 sivua

5. Hallintasivun tyylitiedosto, 3 sivua

```
/ * PROSESSORIN C-KIELINEN OHJELMA */

#define __AVR_ATtiny2313__
#define F_CPU 12000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>


void InitUART (unsigned char baudrate);
unsigned char ReceiveByte (void);
void TransmitByte (unsigned char data2);
void eteen ();
void taakse ();
void ylos ();
void alas ();
void viive ();


int main (void)
{
  unsigned char data;

  int luku = 0;  /*luku muuttujien alustukset */
  int luku2 = 0;

  DDRB=0xFF;  /*11111111 portti B:n kaikki suuntarekisterit lähtötilaan (1) */
  PORTB=0x00;  /* käämit jännitteettömäksi alkutilassa */

  DDRD=0x04; /*0 0000100   PD2lähdöksi(1), PD3-PD6 tuloiksi(0) */
  PORTD=0x78; /*0 1111000  PD3-PD6 pull-upit ylös */


 InitUART (77); /* UBRR= fosc / (16(BAUD-1))*/
 while (1)
  {


   data = ReceiveByte();
   switch (data)
   {



          case 'e' :
                        PORTD |= (1 << PD2);       /* tuuletin päälle */
                        viive();
```

```
                    TransmitByte(PIND); /* lähetetään input nastojen tilat */
                    TransmitByte(0x0A); /* lähetetään rivinvaihtomerkki */
                    break;

            case 'r' :
                    PORTD &= (0 << PD2);     /* tuuletin pois päältä */
                    PORTD=0x78;
                    TransmitByte(PIND);
                    TransmitByte(0x0A);
                    break;


            case 'w'        :               /* virrat pois käämeistä */

                    PORTB = 0x00;
                    break;

            case 'x' :
                    if(PIND != 0x73 && PIND != 0x77) /* jos ei osu etu-anturiin ..
*/
                            {
                    eteen();        /* yksi askel eteen */

                    TransmitByte(PIND);
                    TransmitByte(0x0A);
                    break;
                    }


                    TransmitByte(PIND); /* lähetetään I/O:den tilat sarjaporttiin */
                    TransmitByte(0x0A);
                    PORTB = 0x00 ;      /* ja käämit jännitteettömäksi */
                    break;

            case 'z' :
                    if(PIND != 0x3B && PIND != 0x3F) /* jos ei osu taka-anturiin ..
*/
                    {
                    taakse();       /* ykse askel taakse */

                    TransmitByte(PIND);
                    TransmitByte(0x0A);
                    break;
                    }


                    TransmitByte(PIND); /* lähetetään I/O:den tilat sarjaporttiin */
                    TransmitByte(0x0A);
```

```
                        PORTB = 0x00 ;      /* ja käämit jännitteettömäksi */
                        break;
        case 'c' :
                        if(PIND != 0x6B && PIND != 0x6F) /* jos ei osu yläanturiin ..
*/
                        {
                        ylos(); /* yksi askel ylos */

                        TransmitByte(PIND);
                        TransmitByte(0x0A);
                        break;
                        }


                        TransmitByte(PIND); /* lähetetään I/O:den tilat sarjaporttiin */
                        TransmitByte(0x0A);
                        PORTB = 0x00 ;      /* ja käämit jännitteettömäksi */

                        break;

        case 'v' :
                        if(PIND != 0x5B && PIND != 0x5F) /* jos ei osu ala-anturiin ..
*/
                        {
                        alas();         /* yksi askel alas */
                        TransmitByte(PIND);
                        TransmitByte(0x0A);
                        break;
                        }


                        TransmitByte(PIND); /* lähetetään I/O:den tilat sarjaporttiin */
                        TransmitByte(0x0A);
                        PORTB = 0x00 ;      /* ja käämit jännitteettömäksi */
                        break;

        case 'm'        :
/* pystysuunnan kalibrointi, ajetaan ensin alas niin kauan kunnes rajkytkin tulee vastaan.
Tämän jälkeen ajetaan ylös niin kauan kunnes törmätään anturiin ja lasketaan samalla
matkaan kuluvien askelien määrä. */
while (PIND != 0x5B && PIND != 0x5F ) /* pind6 0 1011011 0x5B*/
                        {
                        alas();
                        }
                        while (PIND != 0x6B && PIND != 0x6F) /* pind5 0 1101011
0x6B*/
                        {
                        luku++;
```

```
                    ylos();
                    }
                    for(int i=0;i<(luku-5);i++) /* ajetaan kamera paikkaan,
laskettujen                         askelten lukumäärä-5. */
                        {
                        alas();
                        }
                    luku = 0; /* alustetaan luku nollaksi */


                    /* vaakasuunnan kalibroini. Vastaava toiminta kuin pystysuunnan
                    kalibroinnissakin */
                    while (PIND != 0x73 && PIND != 0x77) /* pind4 0 1110011 */
                    {
                    eteen();
                    }
                    while (PIND != 0x3B && PIND != 0x3F) /* pind3 0 0111011 */
                    {
                    luku2++;
                    taakse();
                    }
                    for(int j=0;j<((luku2/2)+3);j++)
                        {
                        eteen();
                        }
                    PORTB = 0x00; /* käämit jännitteettömäksi */
                    luku2 = 0;
                    break;



            }

    }


}

/* ----------------aliohjelmat------------------- */

void ylos()
{
for(int i=0;i<1;i++)

                    {
                    PORTB = 0x10 ; /*00010000*/
                    _delay_ms(50);
```

```
                        PORTB = 0x40 ; /*01000000*/
                        _delay_ms(50);
                        PORTB = 0x20 ; /*00100000*/
                        _delay_ms(50);
                        PORTB = 0x80 ; /*10000000*/
                        _delay_ms(50);
                        }
                        PORTB = 0x80;
}

void alas()
{
for(int i=0;i<1;i++)

                        {
                        PORTB = 0x80 ; /*10000000*/
                        _delay_ms(50);
                        PORTB = 0x20 ; /*00100000*/
                        _delay_ms(50);
                        PORTB = 0x40 ; /*01000000*/
                        _delay_ms(50);
                        PORTB = 0x10 ; /*00010000*/
                        _delay_ms(50);
                        }
                        PORTB = 0x10 ;

}

void taakse()
{
for(int i=0;i<1;i++)

                        {
                        PORTB = 0x01 ; /*00000001*/
                        _delay_ms(50);
                        PORTB = 0x04 ; /*00000100*/
                        _delay_ms(50);
                        PORTB = 0x02 ; /*00000010*/
                        _delay_ms(50);
                        PORTB = 0x08 ; /*00001000*/
                        _delay_ms(50);
                        }
                        PORTB = 0x08;


}

void eteen()
```

```
{
for(int i=0;i<1;i++)

                        {
                        PORTB = 0x08 ; /*00001000*/
                        _delay_ms(50);
                        PORTB = 0x02 ; /*00000010*/
                        _delay_ms(50);
                        PORTB = 0x04 ; /*00000100*/
                        _delay_ms(50);
                        PORTB = 0x01 ; /*00000001*/
                        _delay_ms(50);
                        }
                        PORTB = 0x01;


}
void viive()
{
for(int i=0;i<5;i++)
    {
    _delay_ms(50);
    }
}

void InitUART (unsigned char baudrate)
{

  UBRRL = baudrate;
  UBRRH = (baudrate >> 8);

  /* sallitaan lähetys ja vastaanotto */
  UCSRB = (1 << RXEN) | (1 << TXEN);

  /* 8 data bittiä, 1 stop bitti ja ei pariteettia */
  UCSRC = (1 << UCSZ1) | (1 << UCSZ0);

}


unsigned char ReceiveByte (void)
{

  /* odotetaan datan tuloa */
  while (!(UCSRA & (1 << RXC)));


  return UDR; /* palautetaan saatu merkki pääohjelmaan */
```

```
}

void TransmitByte (unsigned char data2)
{

  while (!(UCSRA & (1 << UDRE)));  /* kunnes datarekisteri on tyhjä, lippu nousee */

  /* laitetaan data bufferiin ja lähetetään se */
  UDR = data2;
}
```

```
// KÄYTTÖLIITTYMÄ
// KIRJAUTUMISKYSELY

<html>
<head>
<style type="text/css">
body
{
background-image: url("http://iideetee.no-ip.org/led/ATtiny2313.png");
}
</style>
</head>
<body>

<table width="300" border="1" align="center" cellpadding="0" cellspacing="1" bgcolor="#D9D7D7">
<tr>
<form name="form" method="post" action="validate.php">
<td>
<table width="100%" border="0" cellpadding="3" cellspacing="1" bgcolor="#D9D7D7">
<tr>
<td colspan="3"><strong>Kirjautuminen</strong></td>
</tr>
<tr>
<td width="78">Tunnus</td>
<td width="6">:</td>
<td width="294"><input name="username" type="text" id="username"></td>
</tr>
<tr>
<td>Salasana</td>
<td>:</td>
<td><input name="password" type="text" id="password"></td>
</tr>
<tr>
<td> </td>
<td> </td>
<td><input type="submit" name="Submit" value="Sis&auml;&auml;n"></td>
</tr>
</table>
</td>
</form>
</tr>
</table>

</body>
</html>
```

```
// KIRJAUTUMISEN TARKISTUS

<?php
session_start(); // käynnistetään sessioni
$host="localhost"; // Hostin nimi
$name="****"; // Mysql:n käyttäjänimi
$passu="*******"; // Mysql:n salasana
$db_name="login"; // Tietokannan nimi
$tbl_name="login"; // "Tablen" nimi

// Yhdistetään serverille ja valitaan tietokanta.
mysql_connect("$host", "$name", "$passu")or die("cannot connect");
mysql_select_db("$db_name")or die("cannot select DB");

// Tuodaan annettu käyttäjänimi ja salasana kirjautumiskyselystä.
$username = $_POST['username'];
$password = $_POST['password'];

// Suoritetaan mysql-käsky, jossa tarkistetaan löytyykö annettu salasana ja käyttäjänimi tietokannasta.
$sql = "SELECT * FROM $tbl_name WHERE username='$username' and password='$password'";
$result = mysql_query($sql); // suoritetaan kysely tietokannasta ja sijoitetaan tulos muuttujaan.

// Mysql_num_row, laskee rivit
$count = mysql_num_rows($result);

// Tulos oikea jos, $myusername and $mypassword, table on yhdellä rivillä.
if($count == 1)
{

// Rekisteröidään $myusername, $mypassword ja ohjataan sivulle "attiny2313.php"
$_SESSION['username'] = $username;
$_SESSION['password'] = $password;

echo $_SESSION['username'];
header("location:attiny2313.php");
}
else // jos $myusername ja $mypassword ei matsannu, annetaan virheilmoitus
{
echo "V&auml;&auml;r&auml; <b> tunnus </b> tai <b> salasana</b><br />";
echo "<a href=\"index.php\">Palaa arpoon uudestaan</a>";
//header("location:index.php");
}
?>
```

```
// HALLINTASIVU

<?php
session_start();
if(!isset($_SESSION['username']))
{

 header("Location:index.php");
}

?>

<?php


$ylos = $_POST['ylos'];
$alas = $_POST['alas'];
$oikea = $_POST['oikea'];
$vasen = $_POST['vasen'];
$ota_kuva = $_POST['ota_kuva'];
$On = $_POST['Camera_On'];
$Off = $_POST['Camera_Off'];
$nayta = $_POST['nayta'];
$piilota = $_POST['piilota'];
$Liike_On = $_POST['Liike_On'];
$Liike_Off = $_POST['Liike_Off'];
$kalibroi = $_POST['kalibroi'];
$poista = $_POST['poista'];
$Fan_On = $_POST['Fan_On'];
$Fan_Off = $_POST['Fan_Off'];
$RS232_On = $_POST['RS232_On'];
$RS232_Off = $_POST['RS232_Off'];

/* kirjota data sarjaporttiin*/


    if(isset($Fan_On))
    {
    $ttyS0 = fopen("/dev/ttyS0" , "w");
    fwrite($ttyS0 , "e");
    fclose($ttyS0);
    tiedustele();
    }

    if(isset($Fan_Off))
    {
    $ttyS0 = fopen("/dev/ttyS0" , "w");
```

```
fwrite($ttyS0 , "r");
fclose($ttyS0);
tiedustele();
}

if(isset($poista))
{
system('cd kamera/save ; sudo rm *.jpg');
}

if(isset($Liike_On))
{
system('sudo /etc/init.d/liiketunnistus start > /dev/null &');
}

if(isset($Liike_Off))
{
system('sudo /etc/init.d/liiketunnistus stop > /dev/null &');
}

if(isset($On))
{
system('sudo /etc/init.d/oma start > /dev/null &');
}

if(isset($Off))
{
system('sudo /etc/init.d/oma stop > /dev/null &');
$ttyS0 = fopen("/dev/ttyS0" , "w");
fwrite($ttyS0 , "w");
fclose($ttyS0);
}

if(isset($kalibroi))
{
$ttyS0 = fopen("/dev/ttyS0" , "w");
fwrite($ttyS0 , "m");
fclose($ttyS0);
}

if(isset($ylos))
{
$ttyS0 = fopen("/dev/ttyS0" , "w");
        for($i=0;$i<$_POST['luku'];$i++)
        {
        usleep(200);
        fwrite($ttyS0 , "c");
        }
```

```
    fclose($ttyS0);
    }

    if(isset($alas))
    {
    $ttyS0 = fopen("/dev/ttyS0" , "w");
    for($i=0;$i<$_POST['luku3'];$i++)
            {
            usleep(200);
            fwrite($ttyS0 , "v");
            }
    fclose($ttyS0);
    }

    if(isset($oikea))
    {
    $ttyS0 = fopen("/dev/ttyS0" , "w");
            for($i=0;$i<$_POST['luku1'];$i++)
            {
            usleep(200);
            fwrite($ttyS0 , "z");
            }
    fclose($ttyS0);
    }

    if(isset($vasen))
    {
    $ttyS0 = fopen("/dev/ttyS0" , "w");
            for($i=0;$i<$_POST['luku2'];$i++)
            {
            usleep(200);
            fwrite($ttyS0 , "x");
            }
    fclose($ttyS0);
    }
tiedustele();


function tiedustele()
{
global $taulu; /* taulu, taulukko globaaliksi */
     usleep(500000); /* ehtii uusi ascii tuleen data muuttujaan */

$data = fopen("/srv/www/lighttpd/serial/serial.txt", "r");
while (!feof($data))
    {

   $rivi = fgets($data, 1024); /* luetaan asciina tullut data muuttujaan */
```

```
    $rivi = decbin(hexdec(bin2hex($rivi))); /*muutetaan ensin ascii hexaksi, sitten hexa -> deciksi ja lopuksi dec->binääriksi*/
    $rivi_8bit = substr("00000000",0,8 - strlen($rivi)) . $rivi; /*muutetaan binääri näkymään 8 bittisenä*/

    }
fclose($data);

$taulu = str_split($rivi_8bit);  /* jaetaan bitit taulukon indexeihin */

}
?>


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>jassoo</title>
<link href="tyylit.css" rel="stylesheet" type="text/css" media="screen" />
</head>
<body>

<div id="kokoalue">

            <div id="ylaosa">
            <div id="kuva1"></div>
            <div id="kuva2"></div>
            </div>



<div id="keskialue">


    <div id="kamera">
     <!-- Javakoodi kuvan automaattiseen päivitykseen -->

            <IMG src="http://iideetee.no-ip.org/serial/kamera/webcam.jpg" border="1" name="refresh">
    <SCRIPT language="JavaScript" type="text/javascript">

     var t = 7 // interval in seconds
     image = "http://iideetee.no-ip.org/serial/kamera/webcam.jpg"
     function Start()
      {
     tmp = new Date();
```

```
tmp = "?"+tmp.getTime()
document.images["refresh"].src = image+tmp
setTimeout("Start()", t*1000)
 }
Start();
</SCRIPT>

</div>

<div id="ohjaus">
<form action=" " method="post">
<div id="ylos"><input type="submit" name="ylos" value="yl&ouml;s" /><input
type="text" name="luku" size="1" maxlength="1" value="2" /></div>
<div id="oikea"><input type="submit" name="oikea" value="oikea" /><input
type="text" name="luku1" size="1" maxlength="1" value="2"/></div>
<div id="vasen"><input type="submit" name="vasen" value="vasen" /><input
type="text" name="luku2" size="1" maxlength="1" value="2"/></div>
<div id="alas"><input type="submit" name="alas" value="alas" /><input
type="text" name="luku3" size="1" maxlength="1" value="2"/></div>
</form>
</div>

<div id="valikko">
<form action=" " method="post">
<div id="kalibroi"><input type="submit" name="kalibroi" value="kalibroi"
/></div><br />
<!--        <div id="RS232_On"><input type="submit" name="RS232_On"
value="RS232 On" /></div>
<div id="RS232_Off"><input type="submit" name="RS232_Off" value="RS232 Off"
/></div><br />  -->
<div id="Camera_On"><input type="submit" name="Camera_On" value="Kamera
On" /></div>
<div id="Camera_Off"><input type="submit" name="Camera_Off" value="Kamera
Off" /></div><br />
<div id="Fan_On"><input type="submit" name="Fan_On" value="Fan On"
/></div>
<div id="Fan_Off"><input type="submit" name="Fan_Off" value="Fan Off"
/></div><br />
<div id="liike_On"><input type="submit" name="Liike_On" value="Liike On"
/></div>
<div id="liike_Off"><input type="submit" name="Liike_Off" value="Liike Off"
/></div><br />
<div id="nayta"><input type="submit" name="nayta" value="n&auml;yt&auml;
kuvat" /></div>
<div id="piilota"><input type="submit" name="piilota" value="piilota kuvat"
/></div>

<?php
```

```
            if(isset($nayta))
            {
    echo "<div id=\"poista\"><input type=\"submit\" name=\"poista\" value=\"poista\"
/></div>";
            }
    ?>

    </form>
    </div>
    <div id="kuva">
    <?php

    echo "<span class=\"tila\">Tuuletin: </span>";
    if($taulu[5] == 0)
    echo "<span class=\"offline\">offline</span><br />";
    if($taulu[5] == 1)
    echo "<span class=\"online\">online</span><br />";

    echo "<span class=\"tila\">Kamera: </span>";
    system('sudo /etc/init.d/oma status');
    echo "<br />";
    echo "<span class=\"tila\">Liiketunnistin: </span>";
    system('sudo /etc/init.d/liiketunnistus status');
    echo "<br />";
/*  echo "<span class=\"tila\">Sarjaportin luku: </span>";
    system('sudo /etc/init.d/serial status');
    echo "<br />";  */
    echo "<br />";
    echo "<span class=\"tila\">Vaakasuunta oikea: </span>";
    if($taulu[1] == 1)
    echo "<span class=\"online\">vapaa</span><br />";
    if($taulu[1] == 0)
    echo "<span class=\"offline\">kiinni</span><br />";

    echo "<span class=\"tila\">Vaakasuunta vasen: </span>";
    if($taulu[4] == 1)
    echo "<span class=\"online\">vapaa</span><br />";
    if($taulu[4] == 0)
    echo "<span class=\"offline\">kiinni</span><br />";

    echo "<span class=\"tila\">Pystysuunta alas: </span>";
    if($taulu[2] == 1)
    echo "<span class=\"online\">vapaa</span><br />";
    if($taulu[2] == 0)
    echo "<span class=\"offline\">kiinni</span><br />";

    echo "<span class=\"tila\">Pystysuunta yl&ouml;s: </span>";
    if($taulu[3] == 1)
```

```
    echo "<span class=\"online\">vapaa</span><br />";
    if($taulu[3] == 0)
    echo "<span class=\"offline\">kiinni</span><br />";
    ?>

    <div id="ohje">
    <p><b>Käyttö:</b> <i>Kamera on</i> asettaa kameran p&auml;&auml;lle.
Vastaavasti <i>kamera off</i> sammuttaa kameran. Kameran ohjaaminen tapahtuu
napeilla <i>yl&ouml;s</i>,<i>alas</i>,<i>oikea</i> ja <i>vasen</i>. Numerokenttiin
voi m&auml;&auml;ritt&auml;&auml; liikuttavien askelten m&auml;&auml;r&auml;n.
Oletuksena on 2. Kameran saavutettua max-liikkumisrajansa, tulee vastaavaan suunnan
perään <span class=red>kiinni</span>-teksti ja kyseiseen suuntaan ei pysty
en&auml;&auml;n liikkua. Kalibrointi palauttaa kameran ohjelmoituun
l&auml;ht&ouml;pisteeseens&auml;. <i>Liike on</i> asettaa liikkeentunnistuksen
p&auml;&auml;lle. Kuvat tulevat n&auml;kyviin painamalla <i>n&auml;yt&auml;
kuvat</i>. <i>Poista</i>, poistaa kaikki tallennetut kuvat!</p>
    </div>
    </div>
</div>
    <div id="fotot">
    <?php
    if(isset($nayta))
    {
            $kansio = opendir("kamera/save");
                        if ($kansio == false)

                                echo "Kansioo ei löydy!";
                        else

                        while (($file = readdir($kansio)) !== false)
                        {

                        if($file != "." && $file != "..")
                        {
    echo "<a href = \"kamera/save/". $file ."\">";
    echo "<img src = \"kamera/save/". $file ."\" width=\"100\" height=\"120\"
alt=\"kuva\"/></a>";
    echo " ";
                        }
                    }
                    closedir($kansio);
    }
    ?>
</div>
</div>
</body>
</html>
```

TYYLITIEDOSTO(css)

```
body
{
background-image: url("osat/ATtiny2313.png");
}
#kokoalue
{
margin-left: auto;
margin-right: auto;
width: 95%;
height: auto;
min-height: 600px;
border: 1px solid gray;
}
#ylaosa
{
margin-left: auto;
margin-right: auto;
margin-top: 0px;
height:115px;
width: 100%;
background-image: url("osat/webcontrol.png");
background-repeat: no-repeat;
background-position: center;

}
#kuva1
{
margin-top:0px;
height: 100px;
width: 182px;
background-image: url("osat/piiri2.png");
float: left;
}
#kuva2
{
margin-top:0px;
height: 100px;
width: 182px;
background-image: url("osat/piiri2.png");
float: right;
}

#keskialue
{
border:3px solid black;
```

```
width: 90%;
height: 430px;
background-color: #dcd5d5;
margin-left:50px;
margin-top:10px;
border: 1px solid red;
}

#kamera
{
position: absolute;
left: 12%;
width: 320px;
top: 150px;
}
#ohjaus
{
position: absolute;
left: 13%;
width: 25%;
top: 400px;
border: 1px solid black;
}
#vasen
{
text-align: center;
width: 70px;
float: left;
}
#oikea
{
text-align: center;
width: 70px;
float: right;
}
#ylos
{
text-align: center;
width: 70px;
margin-top: 0px;
margin-left: auto;
margin-right: auto;
}
#alas
{
text-align: center;
width: 70px;
margin-top: 25px;
```

```
margin-left: auto;
margin-right: auto;
}
#valikko
{
position: absolute;
left: 42%;
width: 12%;
top: 150px;
text-align: center;
border: 1px solid black;
}
#kuva
{
position: absolute;
left: 57%;
width: 320px;
top: 150px;
}
#fotot
{
margin-top: 10px;
border: none;
text-align: center;
}
.offline
{
color: red;
}
.online
{
color: green;
}
#ohje
{
width: 395px;
margin-top:10px;
margin-left:0px;
border: 1px solid black;
}
#ohje p
{
font-size: smaller;
}
.red
{
color:red;
}
```