OULUN SEUDUN
AMMATTIKORKEAKOULU

MIKROKONTROLLERIN INTEGROIMINEN LINUX-KÄYTTÖJÄRJESTELMÄÄN

Ari Väyrynen
2011
Oulun seudun ammattikorkeakoulu

MIKROKONTROLLERIN INTEGROIMINEN LINUX-KÄYTTÖJÄRJESTELMÄÄN

Ari Väyrynen
Opinnäytetyö
30.3.2011
Tietotekniikan koulutusohjelma
Oulun seudun ammattikorkeakoulu

OULUN SEUDUN AMMATTIKORKEAKOULU TIIVISTELMÄ

Koulutusohjelma	Opinnäytetyö	Sivuja + Liitteitä
Tietotekniikan koulutusohjelma	Opinnäytetyö	32 + 10
Suuntautumisvaihtoehto	Aika	
Sulautetut järjestelmät	30.3.2011	
Työn tilaaja	Työn tekijä	
OAMK Tekniikan yksikkö	Ari Väyrynen	

Työn nimi

Mikrokontrollerin integroiminen Linux-käyttöjärjestelmään

Asiasanat

AVR, Linux, Ubuntu, DS18x20, mikrokontrollerit, Arduino

Opinnäytetyössä selvitettiin Arduino Duemilanove 328 -mikrokontrolleri-alustaisen tuotteen käyttöönottoa Linux-käyttöjärjestelmäympäristössä. Arduino on avoimeen lähdekoodiin perustuva elektroniikka-alusta. Sen vahvuutena ovat halpuus, joustavuus, helppokäyttöisyys sekä koodin ja ohjeiden esteetön jakaminen. Lisäksi tutkimuksessa selvitettiin Arduinon soveltamista käytännön tarpeisiin esim. lämpötilan seuraamiseen. Teoriaosuudessa käsiteltiin mikrokontrollerin ominaisuuksia ja sen ohjelmointia C-ohjelmointikielellä.

Arduino-kehitysalustan käytön havainnollistamiseksi opinnäytetyössä toteutettiin esimerkkisovellus ja kytkentä, jossa Arduinon digitaaliseen tuloon on kytketty 1-Wire-väylän kautta DS18x20-lämpötila-antureita. Arduino-kehitysalusta integroitiin USB-portin kautta Linux-tietokoneeseen, jossa lämpötilatiedon jatkojalostaminen tapahtui Web-sivuille kuviksi PHP-ohjelmointikielellä ja Gnuplot-komentoriviohjelmalla.

Opinnäytetyön tuloksena syntyi Home Temperature Monitor -niminen esimerkkisovellus, jossa toteutettuja ratkaisuja voidaan käyttää esimerkiksi ohjelmien jatkokehityksessä ja myös opetustilanteissa.

SISÄLTÖ

SANASTO

API	Application Programming Interface, ohjelmointirajapinta.
AVR	Alf (Egil Bogen) and Vegard (Wollan) 's Risc processor architecture, Atmelin valmistama 8-bittinen mikrokontrolleriperhe.
Bluetooth	Avoin standardi laitteiden langattomaan kommunikointiin lähietäisyydellä.
Ethernet	Pakettipohjainen lähiverkkoratkaisu.
Glogaali	Kaikille näkyvä.
GPRS	General Packet Radio Service, GSM-verkossa toimiva pakettikytkentäinen tiedonsiirtopalvelu.
GPS	Global Positioning System, satelliittipaikannusjärjestelmä.
Luokka	Olioiden määrittelyssä hyväksikäytettävä kokonaisuus.
Metodi	Tarkoittaa menetelmää, tapaa suorittaa määrämuotoisesti askel askeleelta edistyvä toimintoketju, jossa saavutetaan tavoiteltu tehtävä tai päämäärä.
PHP	Hypertext Preprocessor on Perlin kaltainen ohjelmointikieli.
PWM	Pulse-Width Modulation, pulssinleveysmodulaatio.
RFID	Radio Frequency IDentification, radiotaajuinen etätunnistusmenetelmä tiedon etälukuun ja -tallentamiseen.
RISC	Reduced Instruction Set Computing, prosessorin suunnittelustrategia.
Servo	Asemointiin tarkoitettu toimilaitteen ohjauspiiri.
Xbee	Standardi, kehitetty vähän virtaa kuluttaville langattomille verkoille.

1 JOHDANTO

Sulautetut järjestelmät vaikuttavat meidän kaikkien jokapäiväisessä elämässä, vaikka emme aina sitä itse edes huomaa. Harvoin tulee ajatelleeksi, kun soittaa kännykällä tai ajaa autolla, että samalla käyttää tietokonetta. Komponenttien jatkuva halpeneminen ja pienentyminen sekä niiden laskentatehon kasvu ovat mahdollistaneet sen, että mikro-ohjaimella varustettu laitteisto pystyy tarjoamaan yhä enemmän samoja palveluita kuin perinteiset henkilökohtaiset tietokoneet. Tämä on johtanut siihen, että sulautetut järjestelmät ovatkin jo kauan olleet toimintojensa perusteella älykkäitä laitteita eivätkä vain perinteiseen kommunikointiin kykeneviä laitteita.

Sisällön tuottaminen näille järjestelmille onkin muuttunut hyvin paljon samanlaiseksi kuin perinteisissä tietokoneissa. Ohjelmointikielet ovat ajan myötä tulleet lähemmäksi toisiaan molemmissa järjestelmissä. On jopa kevennettyjä versioita perinteisten tietokoneiden käyttöjärjestelmistä, jolloin sisällön tuottaminen tapahtuu ainoastaan laskentatehon rajoitusten mukaan.

Tänä päivänä markkinoille tulee jatkuvasti erittäin edullisia ja tehokkaita kehitysalustoja niin harrastelijoiden kuin oppilaitosten käyttöön. Tässä opinnäytetyössä perehdytään ennalta valittuun Arduino-nimiseen kehitysalustaan, joka toimii markkinoilla open source -projektina ja on esimerkkinä muille alan valmistajille. Avoin lähdekoodi mahdollistaa uusien innovaatioiden kehittämisen rakentamalla olemassa olevan päälle ilman, että kaikkea tarvitsee tehdä itse alusta asti uudelleen.

Opinnäytetyön tavoitteena on tuottaa lämpötilaa mittaava esimerkkisovellus, joka integroi Arduino-kehitysalustan ja Linux-pohjaiset sovellukset ja järjestelmät. Esimerkkisovellus mahdollistaa lämpötilan seuraamisen Internetin kautta.

2 SOVELLUKSEN KEHITYSYMPÄRISTÖ

Tässä luvussa esitetään lyhyesti sulautetut järjestelmät, Arduino-kehitysalustan perusominaisuudet, 1-Wire-väylä sekä käytettyjen rajapintojen toiminta.

2.1 Sulautettu järjestelmä

Sulautettu järjestelmä (engl. embedded system) on tiettyyn tarkoitukseen tehty tietokonejärjestelmä. Sille on tyypillistä, että käyttäjän ei tarvitse olla tietoinen laitteen sisällä olevasta tietokoneesta, vaikka hän voikin sen olemassaolon helposti päätellä. Tavallisimmin pieni sulautettu järjestelmä toimii ilman käyttöjärjestelmää ja kiintolevyä ja ne on toteutettu mikro-ohjaimen avulla. Lisäksi järjestelmät voivat olla mobiileja (matkapuhelimet, GPS-laitteet) ja akkukäyttöisiä, mikä asettaa tiukkoja vaatimuksia energian kulutukselle. Suuremmat sulautetut järjestelmät toimivat tehokkailla prosessoreilla ja ne koostuvat useista laitteista, käyttöjärjestelmästä sekä monimutkaisista ohjelmistoista. Tällaisia ovat lentokoneiden valvontajärjestelmät, ydinvoimaloiden turvajärjestelmät ja avaruussukkuloiden lentojärjestelmät. Tämän kaltaiset sovellutukset alkavat jo muistuttaa normaaleja tietokonejärjestelmiä. Sulautetun järjestelmän eräs määritelmähän on puhua laitteesta, joka ei enää ulospäin vaikuta tietokoneelta. (Karvinen - Karvinen 2009, 9.)

Perinteisesti sulautetut järjestelmät ovat olleet suljettuja, eli niin sanottu kolmas osapuoli ei ole voinut tarjota niihin ohjelmiaan. Joissakin uudemmissa järjestelmissä on mahdollistettu tällaisten sovelluksien lisääminen järjestelmään, esimerkiksi matkapuhelinmalleissa ja tallentavissa digi-tv-sovittimissa. Sen sijaan avoimen lähdekoodin käyttö on nopeasti yleistymässä sulautettujen järjestelmien kehityksessä. (Turunen 2010.)

2.2 Mikro-ohjaimen ohjelmointi C-kielellä

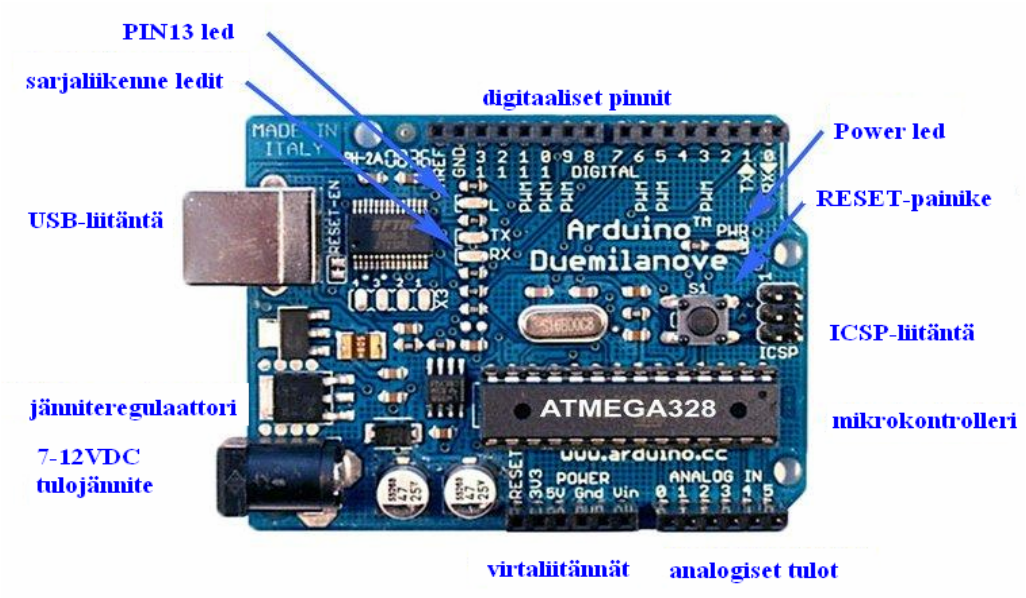
C-kielestä tuli 1990-luvulla johtava mikrotietokoneiden ohjelmointikieli. 2000-luvulla se on tullut mikro-ohjainympäristöön. Ennen sulautetut järjestelmät ohjelmoitiin laiteläheisellä konekielellä eli Assembly-kielellä. Mikäli ohjelma ei ole erityisen nopeuskriittinen, on syytä käyttää ns. korkeamman tason ohjelmointikieltä. Eri valmistajien julkaisemia C-kääntäjiä on saatavilla myös mikro-ohjaimien ohjelmointiin. Flash-muistitekniikan kehittyessä ja valmistuskustannusten pudotessa ne integroitiin osaksi mikro-ohjaimia niiden ohjelmamuistiksi. C-kieli ja flash-muisti yhdessä helpottavat oleellisesti sulautettujen järjestelmien ohjelmistojen testaus- ja kehitystyötä. C-kieltä voidaan pitää sulautettujen järjestelmien yleiskäyttökielenä. Vaikka mikrokontrollerialustaisten ytimien teho ym. ominaisuudet ovat parantuneet, on itse ohjelmoijan opittava myös, miten mikro-ohjain toimii ja miten se liitetään ulkoiseen maailmaan. Mikro-ohjaimet sisältävät vielä rajoituksia niin muisti- kuin suorituskykyresursseille. Nämä seikat saa selville piirivalmistajan julkistamista datalehdistä. (Vahtera 2008, 2.)

Pienten sulautettujen systeemien C-kieli on paljon suppeampi kuin standardin mukainen ANSI C. Tämä johtuu laitteiston eli raudan vajavaisuudesta. Esimerkiksi tiedostojen käsittelyyn liittyvät kirjastot ja käskyt yleensä puuttuvat, koska tavallisimmin pieni sulautettu järjestelmä toimii ilman käyttöjärjestelmää ja kiintolevyä. Korkeamman tason kielet eivät kokonaan poista Assembly-kielen osaamisen tarvetta, koska laiteläheisyys tuo kääntäjäkohtaisia erikoisuuksia, jotka eivät ole oikeaa C-kieltä, vaan kääntäjän valmistajan tekemiä bittikomentoja, joilla päästään manipuloimaan prosessorin ja liitäntäpiirien rekistereitä. (Vahtera 2008, 2.)

2.3 Arduino Duemilanove 328 -kehitysalusta

Ohjelmoinnin ja C-kielen hyvälläkään ymmärtämisellä ja osaamisella sulautetuihin järjestelmiin ei koodia synny. Tämän vuoksi on tärkeää tutustua Arduino Duemilanove 328:n (kuva 1) ATmega328-mikrokontrollerin valmista-

jan julkaisemaan datalehtiseen http://www.atmel.com/dyn/resources /prod_documents /8271S.pdf. Arduino-kehitysalustojen ytiminä ovat Atmelin AVR-mikrokontrollerit. Arduinossa on 14 digitaalista IO-nastaa, joista 6:ta voidaan käyttää PWM-ulostuloina, 6 analogista sisääntuloa, 16 MHz:n kide, USB-liitäntä, virtaliitäntä, ICSP-liitin ja reset-painike. Arduino sisältää alkulatausohjelmiston. Näin ollen erillistä ohjelmointilaitteistoa tai -kaapelia (esim. ICSP) ei tarvita, joten ainoa investointi Arduino-levyn lisäksi on USB-kaapeli. Valmistaja toimittaa Arduino-kehitysalustan täysin käyttövalmiina, joten käyttäjän tarvitsee vain joko kytkeä se USB-liitäntään tai liittää paristo tai muuntaja virtaliittimeen. (Hardware 2010.)



KUVA 1. Arduino Duemilanoven keskeiset liitännät (Hardware 2010)

Arduinon tekniikka pohjautuu Amtelin AVR-mikrokotrolleriperheeseen, joten kaikissa AVR-ohjaimissa on samanlainen rautarakenne. Arduino-kehitysalustan rautaa voi laajentaa helposti lisäämällä siihen valmiita lisälaitteita, esimerkiksi Ethernet, RFID, Xbee, GPRS Quadband, MicroSD 2GB, GPS, BlueTooth -moduuleita. Arduino-korttien tarkemmat tiedot löytyvät valmistajan www-sivuilta http://arduino.cc/en/Main/Hardware. (Hardware 2010.)

Arduino Duemilanove -kortin teknisiä ominaisuuksia lyhyesti:

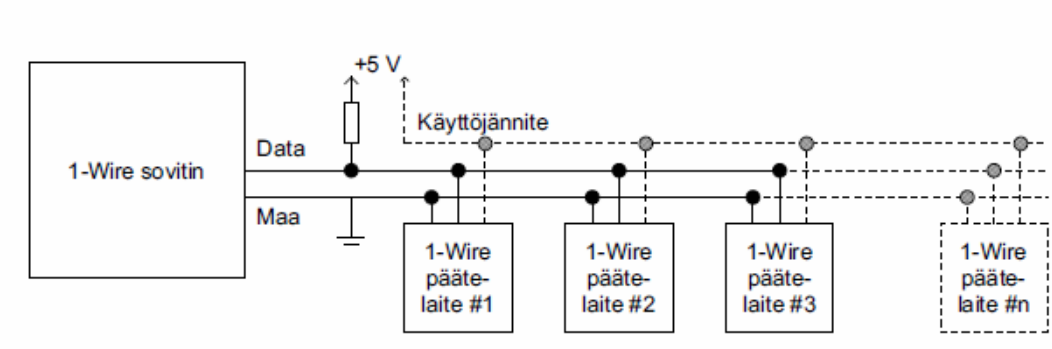
- ATMEGA328-mikrokontrolleri
    - valmiina Arduino-alkulatausohjelma (bootloader) (2 KB)
    - ohjelmille Flash-koodimuistia 32 KB
    - SRAM-työmuistia 2 KB
    - EEPROM-parametrimuistia 1 KB
- 14 digitaalista I/O-liitäntää
    - 40 mA / liitäntä
    - 6 kpl PWM-lähtöjä
- 6 kpl analogista tuloa
- 16 MHz:n kellotaajuus (ulkoinen kide)
- reset-painike
- USB-ohjelmointiliitäntä (FT232RL)
    - B-tyypin runkoliitin
    - +5 V:n käyttöjännite kortille
    - ohjelmien lataus
    - toimii myös sarjaliikenneyhteytenä tietokoneelle
- ICSP-ohjelmointiliitäntä
    - piirin perinteistä sarjaohjelmointia varten
    - 6 pinniä ilman sarjavastuksia
- DC-virtaliitin (2,1 mm)
    - tulojännite 7–12 VDC
    - MC33269D-piirillä +5 VDC:n käyttöjännite kortille (mikäli USB ei kytkettynä)
    - FTDI-piirillä +3,3 V 50 mA
- kortin mitat 68 x 53 mm (2,7" x 2,1").

Arduinon ohjelmointiin tarkoitettu kehitysympäristö toimii niin Windows-, Mac OS- kuin Linux-käyttöjärjestelmäympärisössä. Kehitysympäristön voi ladata suoraan valmistajan www-sivuilta http://arduino.cc/en/Main/Software. Koska

kyseessä on avoimeen lähdekoodiin perustuva alusta, löytyy internetistä valmiita ohjelmakoodin pätkiä. Myös itse kehitysympäristö sisältää valmiita ohjelmointikirjastoja, joita kannattaa hyödyntää omissa projekteissa, kuten tässäkin tutkimuksessa on tehty.

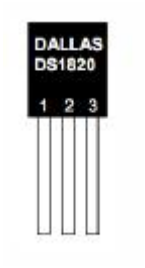
2.4 1-Wire-väylä ja DS18x20-lämpötila-anturi

1-Wire-väylän on suunnitellut amerikkalainen Dallas Semiconductor Corporation (Dallas-Maxim). 1-Wire-väylä on tarkoitettu sellaisille laitteille, joiden pitää antaa merkkisignaalia tai jotain arvoa. 1-Wire-väylä (kuva 2) on tiedonsiirtoväylä, jossa väyläohjain kommunikoi yhtä datalinjaa ja yhteistä maatasoa käyttäen yhden tai useamman päätelaitteen kanssa. (Vahtera 2008, 7.)



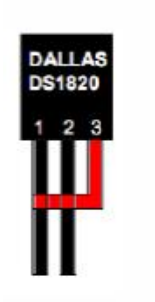
KUVA 2. Periaatekuva 1-Wire-väylästä (Vahtera 2008, 7)

Päätelaitteita on markkinoilla runsaasti DS18x20-antureiden (kuva 3) lisäksi. Päätelaitteet voivat ottaa tarvitsemansa energian datalinjasta (parasite mode), joten ne eivät välttämättä tarvitse erillistä omaa käyttöjännitteen syöttöä. Miltei kaikissa päätelaitteissa on kuitenkin valmius erillisen jännitteensyöttöön käyttöön. Päätelaitteet kytketään 1-Wire-väylään rinnan. (Vahtera 2008, 7.)



KUVA 3. DALLAS DS1820 -lämpötila-anturi (DS18S20 High-Precision 1-Wire Digital Thermometer. 2010)

Kuvassa 4 on esitetty, miten 3-nastaisesta piiristä saadaan tehtyä 2-nastaisen (DS18S20 High-Precision 1-Wire Digital Thermometer. 2010).



KUVA 4. DS1820-anturin muuttaminen parasite modeen (DS18S20 High-Precision 1-Wire Digital Thermometer. 2010)

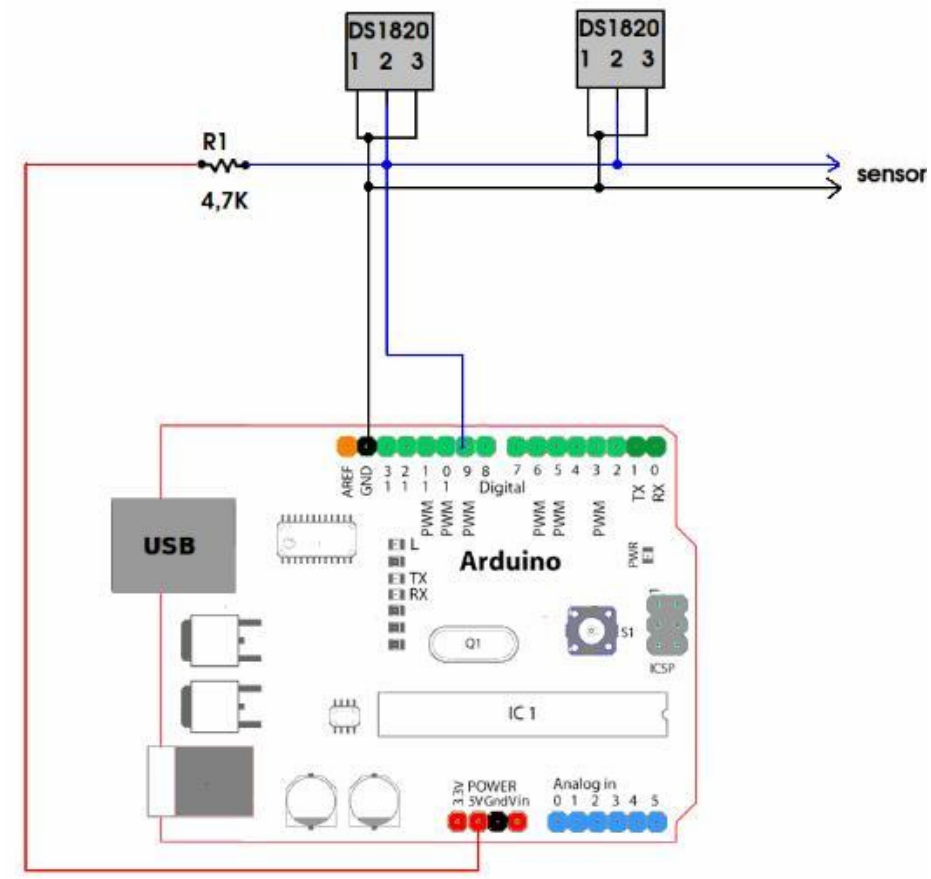
Päätelaitteet tunnistetaan komponentin sisään kirjoitetun osoitteen perusteella. Jokaisella päätelaitteella on 8-bittinen perhekoodi, 48-bittinen yksilöllinen koodi ja 8-bittinen tarkistussumma. Näistä saatua 64-bittistä tunnistetta kutsutaan päätelaitteen ID-koodiksi. Kaikki toiminteet voidaan kohdistaa ID-koodia hyväksikäyttäen tietylle päätelaitteelle. Vastaavasti 1-Wire-standardi määrittää tietyt algoritmit päätelaitteiden ID-koodien selvittämiseen.1-Wire-väylä on synkroninen sarjaväylä, jossa liikennöinti tapahtuu yhdellä kertaa aina yhteen suuntaan, joko päätelaitteelta sovittimelle tai päinvastoin. Sovittin tai mikro-ohjain (isäntäkone) voi lähettää myös sellaisia komentoja, jotka kohdistuvat samanaikaisesti kaikille väylällä oleville päätelaitteille (orja). (Vahtera 2008, 7.)

Väylän liikennöinti perustuu sovittimen väylälle antamiin kellopulsseihin. Tieto kulkee väylällä yksi bitti kerrallaan. Väylän tiedonsiirtokapasiteetti on 15,4 kbit/s (standard) tai 125 kbit/s (overdrive). (Vahtera 2008, 7.)

Kuten aikaisemmin mainittiin, päätelaitteet voivat ottaa käyttöenergiansa väylän datalinjasta. Tämän vuoksi datalinja on kytketty käyttöjännitteeseen (+5 V) n. 5 kilo-ohmin vastuksen kautta. Tällöin päätelaitteet voivat ottaa virran sisäisestä kondensaattoristaan korkeintaan 1 µA:n virralla. Jos piirin virran tarve on tätä suurempi, on käytettävä erillistä käyttöjännitesyöttöä. (Vahtera 2008, 7.)

Mikro-ohjaimen (isäntäkone) ja anturin (orja) välinen viestintä tapahtuu seuraavasti: Väylää hallitseva laite (isäntäkone) asettaa väylän alatilaan 480–960 mikrosekunnin (nollauspulssi) ajaksi ja sen jälkeen se jää odottamaan orjan vastausta. Orja vastaa siihen vetämällä oman väylän alatilaan 15–60 mikrosekunnin päästä saatuaan isännältä merkin. Orja pitää väylää alatilassa 60–240 mikrosekuntia. Näin se ilmoittaa isäntäkoneelle olevansa käytettävissä. (Using a UART to Implement a 1-Wire Bus Master. 2010)

Kuvassa 5 on esitetty esimerkkikytkentä DS18x20-lämpötila-antureiden kytkemiseksi 1-Wire-väylän kautta Arduinon digitaaliseen I/O-liitäntään 9.



KUVA 5. Arduinon kytkeminen 1-Wire-väylään (Sensores de temperatura DS18x20. 2010)

2.5 Ubuntu

Ubuntu on vapaista ohjelmistoista (avoimesta lähdekoodista) koostuva Linux-käyttöjärjestelmä, joka rakentuu Debian-projektin tekemälle työlle. Ubuntu sisältää kaikki peruskäyttöön tarvittavat ohjelmat, kuten tekstinkäsittelyn, taulukkolaskennan, nettiselaimen ja pikaviestimen. Lisää ohjelmia on helppo asentaa asennustyökalulla. (Esittely. 2010.)

Linux on Ubuntun käyttämä käyttöjärjestelmän ydin ja tärkeä osa tietokoneen toimintaa. Se tarjoaa yhteyden ohjelmien ja laitteiston välille. Linuxin laittoi alulle vuonna 1991 suomalainen opiskelija Linus Torvalds. Se oli itsenäisesti kehitetty UNIX-ydin, jonka oli tarkoitus ottaa kaikki hyöty irti siihen aikaan uudesta i386-arkkitehtuurista. Alkujaan Linuxia saattoi ajaa vain

i386-järjestelmillä. Linuxin kehittämiseen on osallistunut ihmisiä joka puolelta maailmaa, ja heidän ansiostaan Linux toimii nykyään käytännössä kaikilla nykyaikaisilla arkkitehtuureilla. (Esittely. 2010.)

Linuxilla on ollut teknologisen merkityksen lisäksi myös ideologista arvoa. Vapaiden ohjelmistojen ympärille on kehittynyt suuri yhteisö, joka panostaa vapaiden ohjelmistojen kehittämiseen niin hyväksi kuin mahdollista. (Esittely. 2010.)

2.6 Lämpötila-anturin ohjaus Arduinolla

Yleisesti rajapinta sisältää joukon metodien otsikoita. Jos luokka toteuttaa rajapinnan, pitää siinä olla rajapinnan kaikkien metodien määrittely. Rajapinta on siis tavallaan sopimus siitä, mitä kaikkia metodeja luokka ainakin sisältää.

Arduino-kehitysalustaan löytyy linkistä http://milesburton.com zip-paketti TCL 3.6.0 - Version 3.6.0 (1/10/2010), joka sisältää 1-Wire-väylän ohjaukseen OneWire-luokkakirjaston (OneWire.cpp, OneWire.h) sekä DS18x20-lämpötila-anturin ohjaukseen tarkoitetun DallasTemperature-luokkakirjaston (DallasTemperature.cpp, DallasTemperature.h). Zip-paketti sisältää myös Arduino-projekteihin valmiita monipuolisia esimerkkitapauksia (sketchbook). Luokkien metodit ja selitykset löytyvät readme.txt-tiedostosta.(Dallas Temperature Control Library. 2010.)

Kuvassa 6 on esimerkkikoodi Arduino-kehitysalustalle 1-Wiren käytöstä. Ohjelman aluksi määritetään globaalit muuttujat ja vakiot, jonka jälkeen suoritetaan setup()-funktio kerran. Siinä tehdään kertaluotoiset asetukset. Funktiota setup() kutsutaan automaattisesti ja se ei ota vastaan mitään parametrejä eikä myöskään palauta parametrejä. Funktio loop() suoritetaan automaattisesti. Tämä on se funktio, jota toistetaan loputtomasti, kunnes Arduino sammutetaan. Funktio ei palauta arvoa, eikä se saa parametrejä. Tässä ole-

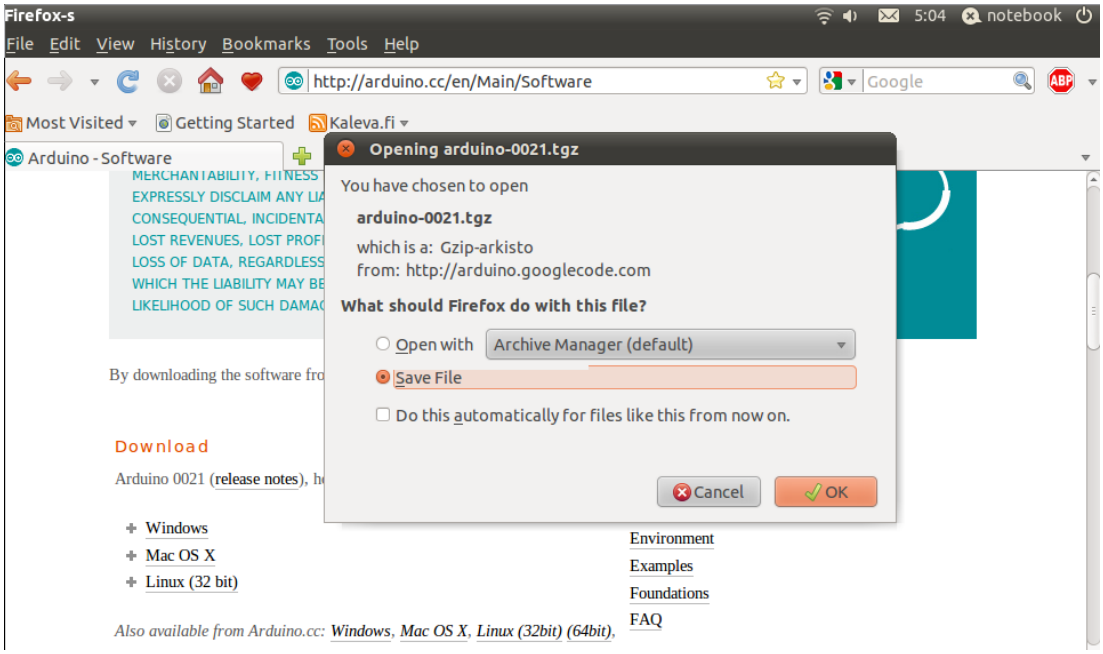
va koodinpätkä on tarkoitettu muun muassa piireille DS18B20, DS1822, DS18S20 ja DS1820. (Dallas Temperature Control Library. 2010.)

```
#include <OneWire.h>
#include <DallasTemperature.h>

// Data wire is plugged into pin 9 on the Arduino
#define ONE_WIRE_BUS 9

// Setup a oneWire instance to communicate with any OneWire devices
(not just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

void setup(void)
{
  // start serial port
  Serial.begin(9600);
  Serial.println("Dallas Temperature IC Control Library Demo");

  // Start up the library
  sensors.begin();
}


void loop(void)
{
  // call sensors.requestTemperatures() to issue a global tempera-
ture
  // request to all devices on the bus
  Serial.print("Requesting temperatures...");
  sensors.requestTemperatures(); // Send the command to get tempera-
tures
  Serial.println("DONE");

  Serial.print("Temperature for Device 1 is: ");
  Serial.print(sensors.getTempCByIndex(0)); // Why "byIndex"? You
can have more than one IC on the same bus. 0 refers to the first IC
on the wire
}
```

KUVA 6. Simple.pde-esimerkkikoodi Arduino-kehitysalustaan (Dallas Temperature Control Library. 2010)

3 ARDUINON KEHITYSYMPÄRISTÖN ASENNUS

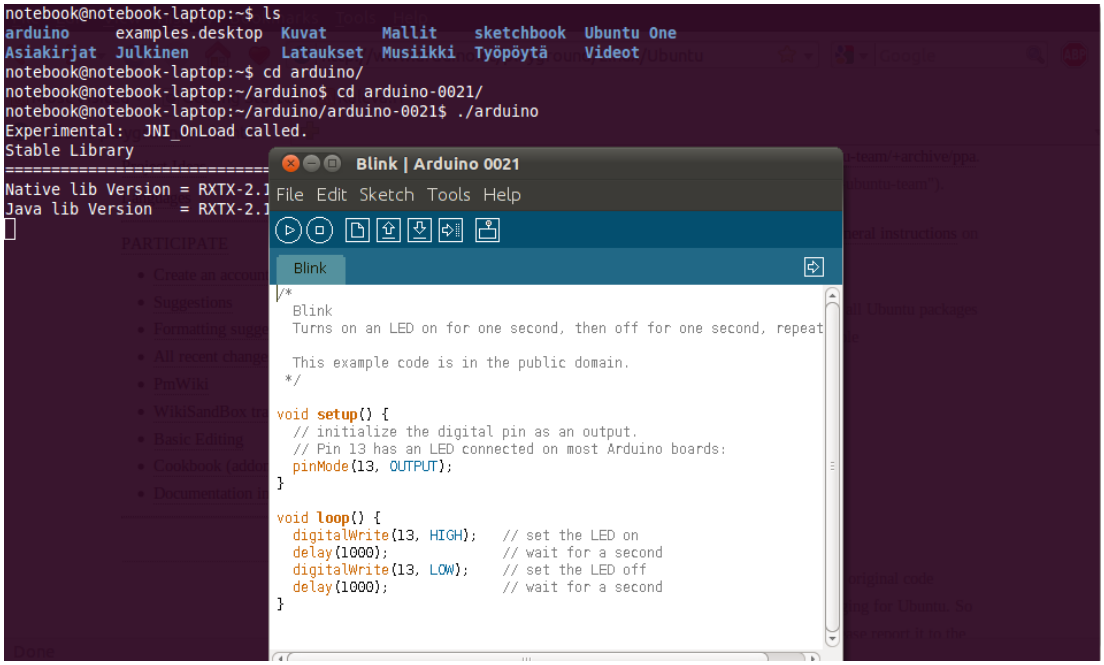
Arduinon kehitysympäristön voi ladata Arduinon viralliselta kotisivuilta http://arduino.cc/en/Main/Software. Tässä työssä valittiin Linux-käyttöjärjestelmäversio eli Linux (32 bit) -versio (kuva 7).



KUVA 7. Arduino-kehitysympäristön lataus

Avataan Linuxin (Ubuntu) komentokehote ja mennään hakemistoon, minne tallennettiin ladattu tiedosto. Paketti puretaan komenolla ”sudo tar -xvzf arduino-0021.tgz” ja mennään syntyneeseen hakemistoon komennolla ”cd arduino-0021/”. Tämän jälkeen voidaan käynnistää Arduinon kehitysympäristön komennolla ”./arduino”. Jos tulee virheilmoituksia, annetaan seuraavat komennot järjestyksessä ”sudo add-apt-repository ppa:arduino-ubuntu-team”, ”sudo apt-get update” ja ”sudo apt-get install arduino” ja vastataan kehotteisiin kyllä.

Arduinon kehitysympäristön testaus suoritetaan seuraavasti: käynnistä Arduino komenolla ./arduino ja valitaan File: Examples: 1.Basics: ja josta valitaan Blink. Työtilaan ilmestyy koodia (kuva 8).



KUVA 8. Arduinon kehitysympäristön työtila

Valitaan kehitysympäristön valikosta USB-portti: Tools: Serial Port: /dev/ttyUSB0. Käännetään ohjelma painamalla Compile/Verify ja siirretään ohjelma Arduinoon painamalla työkalupalkista Upload . Arduinon sarjaportin punainen ja vihreä ledi vilkkuvat. Kun oranssi ledi (PIN13 led) Arduinon piirilevyllä alkaa vilkkua, kehitysympäristö toimii.

4 SOVELLUKSEN TOTEUTTAMINEN

Tässä luvussa käydään vaiheittain läpi tavoitteena olleen esimerkkisovelluksen toteutus.

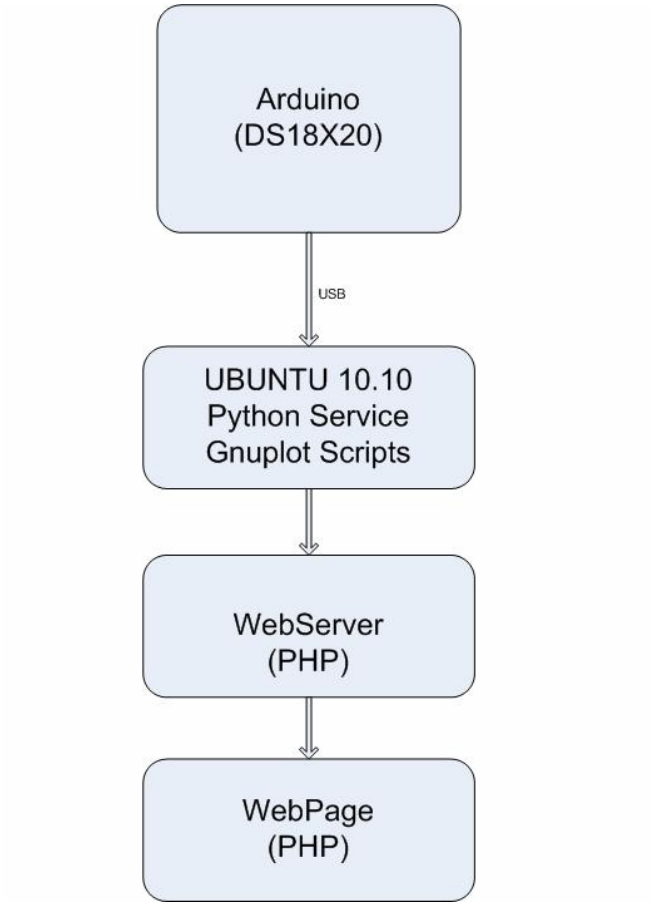
4.1 Toteutuksen lähtökohdat

Ohjelmointityössä yleisesti ei pyritä jokaisessa uudessa sovelluksessa toteuttamaan kaikkia perusratkaisuja uudelleen, vaan käytetään hyväksi valmiita ratkaisuja ja esimerkkejä ja sovelletaan niitä. Näin voidaan vähentää työskentelyaikaa huomattavasti, kun osa ratkaisuista voidaan toteuttaa valmiiden mallien pohjalta. Siksi työhön otettiin pohjaksi Arduinon valmis koodipohja, jossa on toteutettu 1-Wire-väylän ohjaus, ja vastaavasti Web-sivujen pohjana käytettiin valmista PHP:llä koodattua pohjaa, joita muokattiin tarpeen mukaan. Kun kysymyksessä on lämpötilatieto, haluttiin se myös tallentaa ja jatkojalostaa Gnuplot-ohjelman avulla haluttuun graafiseen muotoon.

4.2 Home Temperature Monitor

Päätarkoituksena toteutuksessa oli ajatus, että esimerkiksi kodin lämpötilaa voidaan seurata poissa ollessa. Tietoturva-asioihin ei tässä työssä oteta kantaa, koska lämpötilaa ei pyritä ohjaamaan, vaan ainoastaan seuraamaan.

Projekti (kuva 9) perustui Arduino-kehitysalustaan ja Dallas DS18x20 -lämpötila-antureihin. Lämpötilatiedon jalostaminen tapahtuu Web-serverissä PHP-ohjelmointikielellä kuviksi, ja Python-ohjelmointikieltä käytetään tiedonhakuun lukemalla USB-porttia ja tallentamalla tuleva tieto tekstitiedostoon temp.txt. Lisäksi sarjaportista tulevaa tietoa tallennetaan data-tiedostoon, jotta lämpötilaprofiilin kehitystä voidaan seurata ajan kuluessa. Tämä tieto käsiteltiin Linuxista löytyvällä Gnuplot-ohjelmalla kuviksi.



KUVA 9. Home Temperature Monitorin lohkokaavioesitys

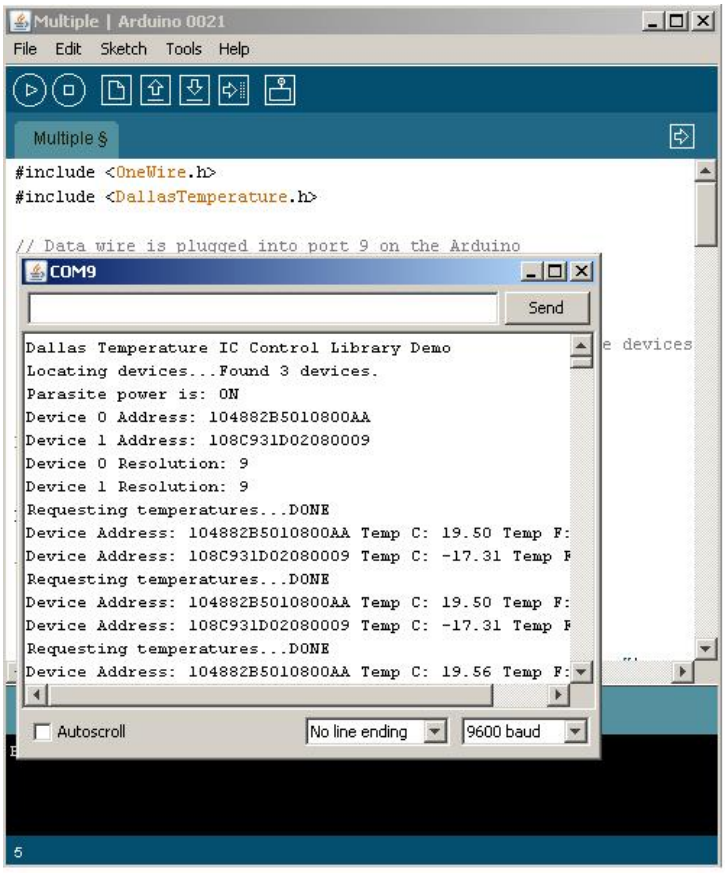
4.3 Dallas Temperature Control Libraryn asentaminen

Dallas Temperature Control Libraryn voi ladata oheisesta linkistä:

http://milesburton.com/index.php?title=Dallas_Temperature_Control_Library.

Dokumentin kirjoitushetkellä versio oli TCL 3.6.0 (1/10/2010). Paketti puretaan ja siitä kopioidaan hakemistot DallasTemperature ja OneWire Arduinon kehitysympäristön alle ~/arduino-0021/libraries/ -hakemistoon. Tämän jälkeen voidaan käynnistää Arduinon kehitysympäristön ja avata valmis sketchbook komennolla File/Examples/DallasTemperature/Multilpe.pde. Seuraavaksi määritellään kehitysympäristön valikosta, mihin Linuxin USB-porttiin Arduino-kehitysalusta on kytketty. Valitsemalla valikosta

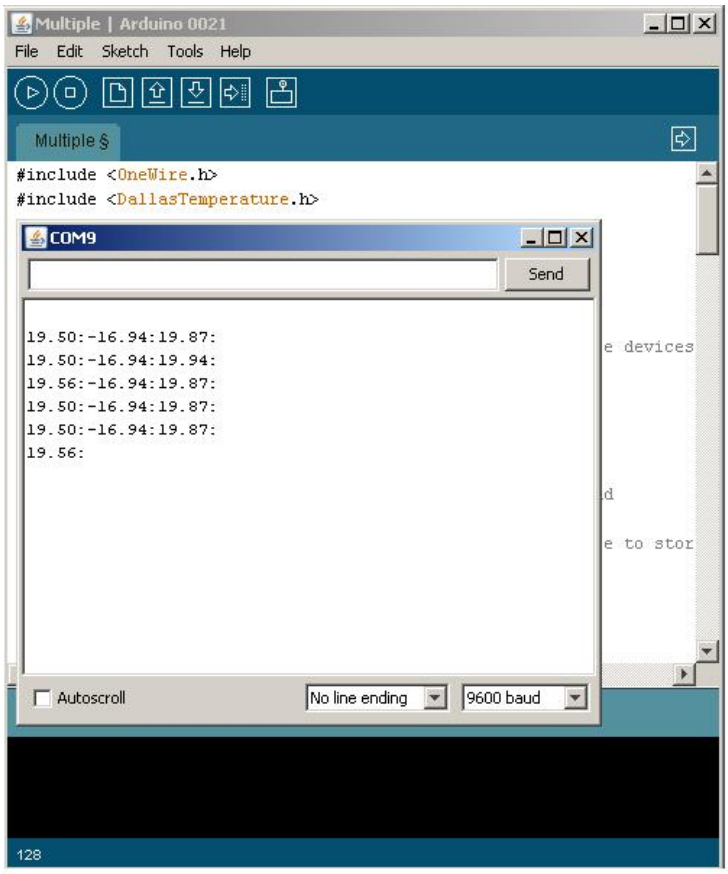
Tools/Serial Port voidaan oikea portti valita, esimerkiksi /dev/ttyUSB1. Käännetään ohjelma painamalla Compile/Verify ja siirretään ohjelma Arduinoon painamalla työkalupalkista Upload . Arduinon sarjaportin punainen ja vihreä ledi vilkkuvat. Kun ohjelma on ladattu Arduinoon, voidaan avata kehitysympäristön valikosta Tools/Serial Monitor-valikko ja katsoa, lähettääkö Arduino sarjaportin kautta dataa (kuva 10). Kannattaa huomioida, että aina kun avataan yhteys Arduinon USB-porttiin, Arduino itse uudelleen käynnistää (resetoi) sisältämänsä ohjelman. Tämä voi tietyissä tilanteissa aiheuttaa ongelmia, mutta se voidaan estää kytkemällä +5 V ylösvetovastuksen kautta Arduinon reset-nastaan.



KUVA 10. Arduinon sarjaportin monitorointi kehitysympäristössä

Tämän jälkeen multiple.pde:n sisältämää koodia voidaan muuttaa siten, että Arduinon lähettämän datan formaatti on muodoltaan ”19.50:-16.94:19.87: ” (liite 1).

Uudelleen kääntämisen ja lataamisen jälkeen Arduino lähettää dataa sarjaportista (kuva 11).



KUVA 11. Arduinon sarjaportin monitorointi

4.4 Linuxin sarjaportin lukeminen Python-ohjelmalla

Arduinon USB-portista tulevan tiedon lukemiseen voidaan Linuxissa käyttää pientä Python-ohjelmointikielellä tehtyä ohjelmaa (kuva 12). Esimerkkiohjelma lukee kerran minuutissa sarjaportista tulevaa tietoa ja tallentaa sen joko temp.txt- tai data-tiedostoon. Home Temperature Monitor -ohjelma käyttää

temp.txt:ssa olevaa tietoa ja Gnuplot-ohjelma käyttää data-nimistä tiedostoa, johon on leimattu päivämäärä ja kelloaika.

```
#! /usr/bin/python

# 2010 Ari Vayrynen aja2.py

import os
import serial
import threading
import time
import cmd
import sys



# Read data from Arduino serial port a line at a time
# and dump to file with timestamps
class Arduino(threading.Thread):
  def run(self):
    f = open('/var/www/data', 'a')#file for gnuplot
    f1 = open('/var/www/home_monitor/temp.txt','w')#file for php
    # Port may vary from /dev/ttyUSB1
    self.ser = serial.Serial('/dev/ttyUSB1', 9600)
    self.ser.flushInput()
    old_timestamp = None
    while 1:
      data = self.ser.readline().strip()


      if data:
            timestamp = time.strftime("%m/%d/%Y %H:%M",
time.localtime())
            if timestamp != old_timestamp:
              # Only log once per minute
          print >>f, timestamp, data.strip()
          f.flush()
          f1 = open('/var/www/picaxe/temp.txt','w')
          print >>f1, data.strip()
              f1.close()
          old_timestamp = timestamp


def main():
   try:
     ard = Arduino()
     ard.start()
   except KeyboardInterrupt:
     print '^C received, shutting down server'

if __name__ == '__main__':
  main()
```

KUVA 12. Python-koodi lämpötilatietojen lukemiseen USB-portista

Python-koodia voidaan ajaa terminaali-ikkunassa komennolla ”sudo python aja2.py”, mutta ennen tätä sarjaportin käsittelyyn pitää antaa oikeudet komennolla ”sudo chmod 777 /dev/ttyUSB1”. Komennot voi tallentaa Linuxin /etc/crontab-nimiseen tiedostoon lisäämällä rivit

```
*/10 *    * * *   root    chmod 777 /dev/ttyUSB1
```

ja

```
*/6  *    * * *   root    python /etc/aja2.py.
```

Tämä tehdään siksi ettei haluta käynnistää palveluita käsin esimerkiksi koneen uudelleenkäynnistyksen jälkeen. Temp.txt- ja data-tiedostojen sisältöjen formaatin pitäisi näyttää kuvan 13 mukaiselta.

```
20.62:-2.06:20.94: //temp.txt-tiedoston sisältö

12/16/2010 16:10 21.44:-8.44:21.87: //data-tiedoston sisältö
12/16/2010 16:11 21.44:-8.38:21.87:
12/16/2010 16:12 21.44:-8.44:21.87:
12/16/2010 16:13 21.37:-8.44:21.81:
12/16/2010 16:14 21.37:-8.50:21.81:
12/16/2010 16:15 21.37:-8.44:21.81:
12/16/2010 16:16 21.37:-8.50:21.87:
12/16/2010 16:17 21.44:-8.50:21.87:
```

KUVA 13. Tiedostojen temp.txt ja data esimerkkisisältö

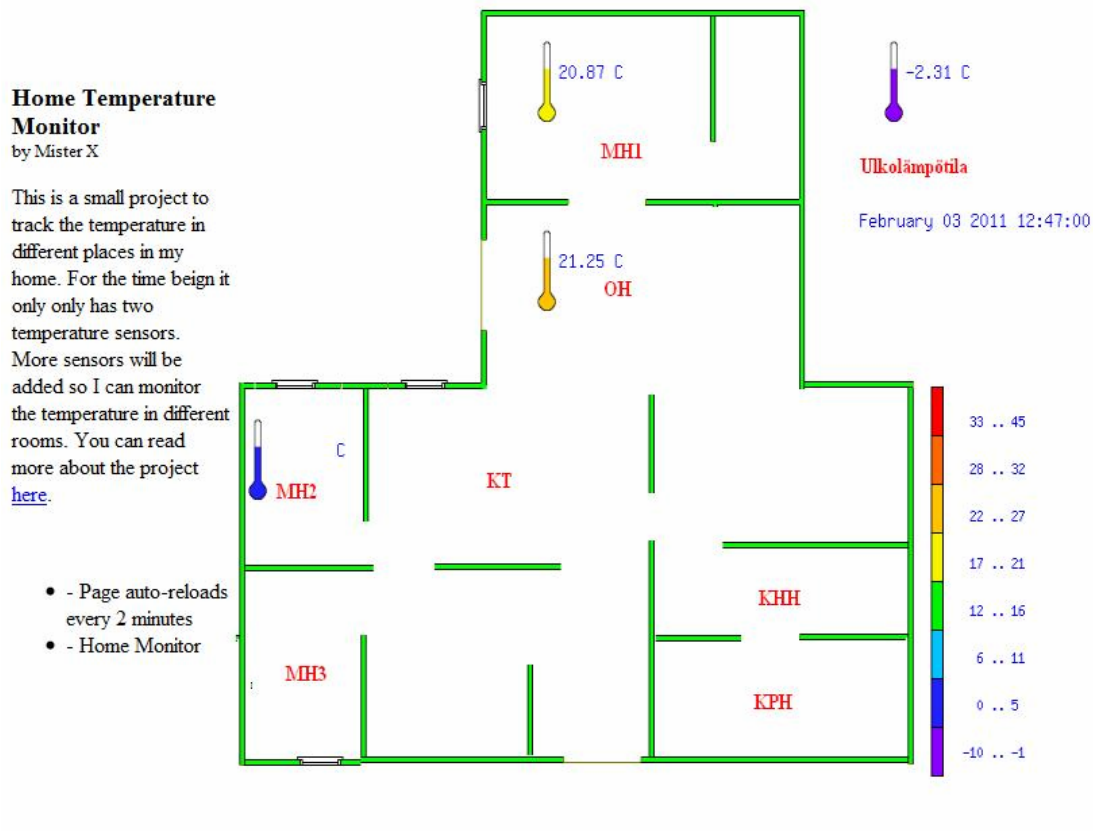
4.5 Lämpötilatietojen näyttäminen Web-sivulla

Web-sivun pohjana voidaan käyttää PHP-kielellä ohjelmoitua home_image.phps-nimistä tiedostoa. Se löytyy seuraavan linkin alta:

http://steliosm.net/site/?Projects:Home_Monitor.

Muutoksia koodiin tarvitsee tehdä lähinnä lisäämällä lämpötila-antureita, muokkaamalla talon pohjakuvaa ja laittamalla tiedostojen polut oikein (liite 2). Kuvatiedosto Images.zip tulee purkaa …/www/home_monitor/images-hakemistoon, jotta php-koodi löytää ne. Itse home_monitor.html-tiedosto voi olla myös …/www/home_monitor-hakemistossa ja sen sisältö voisi olla liitteen 3 kaltainen. Home_monitor-hakemiston oikeudet tulisi tarkistaa ja ne voisivat olla esimerkiksi muotoa chmod 755 -R.

Järjestelmän testaamisen voi suorittaa käynnistämällä Web-selaimen ja kirjoittamalla selaimen osoiteriville osoitteeksi joko http://127.0.0.1 /home_monitor/home_monitor.html tai http://localhost/home_monitor/. Näyttöön tulisi ilmestyä seuraavanlainen kuvan 14 mukainen näkymä.



KUVA 14. Valmis Home Temperature Monitor -sovellus

4.6 Lämpötilatiedon muuttaminen kuviksi

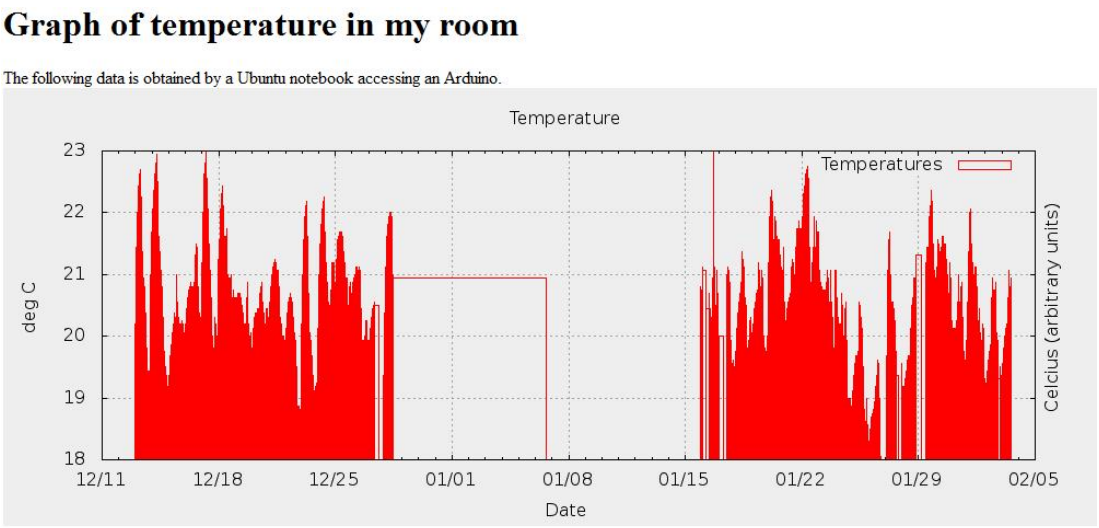
Gnuplot on kätevä komentoriviohjelma 2- ja 3-ulotteisten kuvaajien luomiseen funktioista tai annetuista pisteistä. Gnuplot toimii useimmissa käyttöjärjestelmissä, kuten Linuxissa, Windowsissa ja Mac OS X:ssä. Ohjelmiston kehityksen aloittivat Thomas Williams ja Colin Kelley vuonna 1986 ja myöhemmin kehittäjiä on tullut lisää. Gnuplot osaa tulostaa kuvaajia monissa tiedostomuodoissa (PNG, EPS, SVG,..). Lisäksi Gnuplotilla voi tuottaa LaTeX-kuvauskieltä, jolloin se tulostaa kuvaajan tex-tiedostomuodossa, jonka voi suoraan yhdistää LaTeX-dokumenttiin. (Kotz 1991.)

Kuvassa 15 on esitetty, miten luodaan Gnuplotin avulla kuva ”graph.png” data-tiedoston sisällöstä. Tallennetaan skripti esimerkiksi nimellä sisa.sh …/etc/-hakemiston alle ja ajetaan se terminaalissa komenolla ”root gnuplot /etc/sisa.sh”. Lisäämällä /etc/crontab-tiedostoon komento ” */7 * * * * root gnuplot /etc/sisa.sh” muodostuu kuva /var/www/graph-hakemistoon joka seitsemäs minuutti.

```
set   autoscale                          # scale axes automatically
set  xtic auto                            # set xtics automatically
set  ytic auto
set contour base
set nosurface
set view 0,0
set xdata time
set format x '%m/%d'
set yrange [18.00:23.00]
set timefmt '%m/%d/%Y %H:%M'
set grid xtics ytics
set terminal png xeeeeee x000000 size 1000, 400
set output '/var/www/graph/graph.png'
set xlabel 'Date'
set ylabel 'deg C'
set y2label 'Celcius (arbitrary units)'
set title 'Temperature'
plot '/var/www/data' using 1:3 title 'Temperatures' with boxes
#linespoints
```

KUVA 15. Gnuplot-skripti

Kuvassa 16 on Gnuplot-ohjelmalla luotu kuva data-tiedoston sisällöstä, joka linkitetään näkymään Web-sivulla. Tempsensor.html-tiedoston sisältö voisi olla kuvan 17 kaltainen.



KUVA 16. Graph.png-kuva

```
<HTML>

<head>
<link rel="stylesheet" type="text/css" href="/plug.css">
<title>Graph of temperature in my room</title>
</head>
<body>
<h1>Graph of temperature in my room</h1>
The following data is obtained by a Ubuntu notebook accessing an Ar-
duino.
<img src="../graph/graph.png">
<br>
Note:
<p>
For more details see <a
href="http://xxx.xxx.com">http://xxx.xxx.com</a>.

</body>

</HTML>
```

KUVA 17. Tempsensor.htm-tiedoston sisältö

5 TESTAUS

Arduino-kehitysalustan ohjelmistojen ja kytkentöjen kehittämiseen ja testaukseen voi käyttää Infology Pty Ltd:n valmistamaa VirtualBreadboard Version 4.1.8.0 -simulaattoria (löytyy linkistä http://www.virtualbreadboard.net/). Toistaiseksi se toimii ainoastaan Windows-ympäristössä. Simulaattorin käytöllä voi välttää esimerkiksi vääristä kytkennöistä johtuvat piirien tuhoutumiset ja vikaantumiset. Yleensä ohjelmia ja kytkentöjä kehitettäessä niitä kannattaa testata simulaattorissa niin kauan, että suurimmat ongelmat toiminnassa on saatu selvitettyä. Tähän syynä on suhteellisen pitkä aika, joka ominaisuuksien varmentamiseen kuluu esimerkiksi Arduinon omaa kehitysympäristöä käytettäessä.

Tässä opinnäytetyössä simulaattoria käytettiin lähinnä Arduino-kehitysalustan koekytkentöjen toiminnan ymmärtämiseksi ja opiskeluun. Simulaattorilla tehtiin esimerkkikytkentä, jossa Arduino-kehitysalustan analogiset tulot saivat jännitejakokytkennästä tarvittavan informaation ja niiden perusteella voitiin Arduinon digitaalisella PWM-lähdöllä ohjata servomoottoria 180-astetta. Ohjelmistojen testauksen yhteydessä käytännölliseksi tuli myös ohjelmistokehitystyökalujen ohjelmointivirheiden etsintätyökalu eli ns. debuggeri. Laitetestaukseen käytin koululta saamaani Arduino-korttia.

Home Temperature Monitor -sovelluksen testaaminen suoritettiin projektin edetessä vaihe vaiheelta, koska projektin kokonaisuus hahmottui pala palalta ja opiskelijan tietotaiton kehittyessä. Kehittelyn ja testauksen aikana sovelluksen eri lohkojen kriteereiksi muodostuivat varmatoimisuus ja nopeus.

6 YHTEENVETO

Sulautettujen järjestelmien C-ohjelmointi on hyvin haasteellista verrattuna ohjelmointityöhön toisilla kielillä. Vaikka ohjelmointi onkin käytännössä pitkälti samanlaista kuin perinteinen C-kielen ohjelmointi, luovat sulautettu järjestelmä -arkkitehtuurin tuomat rajoitukset erilaisia ongelmia.

Opinnäytetyö oli henkilökohtaisesta näkökulmasta katsottuna innostavan haastava ja motivoiva. Oma ohjelmointikokemukseni rajoittuu opintojen yhteydessä tehtyihin harjoitustöihin, joten itsenäinen työskentely ja laitteistojen ja ohjelmistojen soveltaminen toisiinsa oli todella mielenkiintoista. Suurimmat haasteet työssä syntyivät siitä, mitä koodausta käyttäisin Linuxissa. Testasin Arduinon ohjaamista sarjaportin kautta kehittämällä pieniä Qt- ja Web-sovelluksia ja käyttämällä mm. C-, Python- ja PHP-koodausta. Haastavuutta lisäsi se, että nykyinen kehitystahti sulautettujen laitteiden soveltamisessa on hyvin nopeaa. Tahdin nopeudesta johtuukin, ettei paperilla saatava materiaali yleensä ole enää uusinta tietoa. Sähköisessä muodossa materiaalia on saatavilla runsaasti, mutta se on yleensä ottaen englanniksi.

Vaikka sulautettujen järjestelmien kehittyminen on nopeaa, pystyin turvautumaan sovelluskehittäjien keskustelupalstoilla käytyihin ongelmanratkaisukeskusteluihin. Oli mielenkiintoista huomata, kuinka avuliaasti kehittäjäyhteisö tarttui pieniltäkin tuntuvien ongelmien ratkaisuun. Tällaisia yhteisöjä ovat mm. Arduinon sovelluskehittäjien keskustelupalstat sekä Arduino Finlandin Facebook-yhteisö.

Opinnäytetyön yhteydessä syntyi Home Temperature Monitor -niminen esimerkkisovellus, joka mahdollistaa kodin lämpötilan seuraamisen Internetin kautta. Lisäksi esimerkkisovellus tukee käytännönläheistä oppimista laitteistojen ja ohjelmistojen suhteen, joten tässäkin mielessä Arduino-kehitysalustaa voi suositella harrastelijoiden ja oppilaitosten käyttöön.

Opinnäytetyön edetessä tein itselleni merkintöjä sovellusmahdollisuuksista jatkokehitystä varten. Kuitenkin projektin loppuvaiheessa huomasin listan kasvaneen niin suureksi, ettei työhön varattu aika olisi millään riittänyt niiden ominaisuuksien lisäämiseen, puhumattakaan täysin uusista toiminnoista. Yksi jatkokehitysmahdollisuus voisi olla lämpötilan ja valaistuksen ohjaaminen Internetin kautta.

LÄHTEET

1-wire Communication Through Software. 2010. Maxim-IC. Saatavissa: http://pdfserv.maxim-ic.com/en/an/AN126.pdf. Hakupäivä 11.1.2011.

Dallas Temperature Control Library. 2010. Saatavissa: http://milesburton.com/index.php/Dallas_Temperature_Control_Library. Hakupäivä 7.10.2010.

DS18S20 High-Precision 1-Wire Digital Thermometer. 2010. Maxim-IC. Saatavissa: http://datasheets.maxim-ic.com/en/ds/DS18S20.pdf. Hakupäivä 11.1.2011.

Esittely. 2010. Saatavissa: http://wiki.ubuntu-fi.org/Esittely. Hakupäivä 27.10.2010.

Hardware. 2010. Saatavissa: http://www.arduino.cc/en/Main/hardware. Hakupäivä 2.12.2010.

Karvinen, Tero - Karvinen, Kimmo 2009. Sulautetut: Opi rakentamaan robotteja ja muita sulautettuja järjestelmiä. Helsinki: Readme.fi.

Kotz, David 1991. LaTeX and the GNUPLOT Plotting Program. Saatavissa: http://www.fnal.gov/docs/products/gnuplot/tutorial/. Hakupäivä 11.12.2010.

Sensores de temperatura DS18x20. 2010. LSDios Arduino. Saatavissa: http://sites.google.com/site/lsdiosarduino/Proyectos-Arduino/sensores-de-temperatura-ds18x20. Hakupäivä 11.12.2010.

Turunen, Ari 2010. Ohjelmistopalvelut pilvestä. Saatavissa: http://www.csc.fi/csc/julkaisut/tieteentietotekniikka/2010/3/pilvi. Hakupäivä 7.10.2010.

Using a UART to Implement a 1-Wire Bus Master. 2010. Maxim-IC. Saatavissa: http://pdfserv.maxim-ic.com/en/an/AN214.pdf. Hakupäivä 11.1.2011.

Vahtera, Pentti 2008. Mikro-ohjaimen ohjelmointi C-kielellä 2. Saatavissa: http://www.microsalo.com/Kirja. Hakupäivä 9.10.2010.

```
# Multiple.pde
#include <OneWire.h>
#include <DallasTemperature.h>

// Data wire is plugged into port 9 on the Arduino
#define ONE_WIRE_BUS 9
#define TEMPERATURE_PRECISION 9

// Setup a oneWire instance to communicate with any OneWire devices (not
just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

int numberOfDevices; // Number of temperature devices found

DeviceAddress tempDeviceAddress; // We'll use this variable to store a
found device
address

void setup(void)
{
  // start serial port
  Serial.begin(9600);


  // Start up the library
  sensors.begin();

  // Grab a count of devices on the wire
  numberOfDevices = sensors.getDeviceCount();

     // Loop through each device, print out address
  for(int i=0;i<numberOfDevices; i++)
  {
    // Search the wire for address
    if(sensors.getAddress(tempDeviceAddress, i))
       {


           // set the resolution to 9 bit (Each Dallas/Maxim device is capable of
several different resolutions)
```

```
 sensors.setResolution(tempDeviceAddress, TEMPERATURE_PRECISION);


      }else{
            //Serial.print("Found ghost device at ");
            //Serial.print(i, DEC);
            //Serial.print(" but could not detect address. Check power and ca-
bling");
      }
  }

}

// function to print the temperature for a device
void printTemperature(DeviceAddress deviceAddress)
{


  // method 2 - faster
  float tempC = sensors.getTempC(deviceAddress);
  //Serial.print("Temp C: ");
  Serial.print(tempC);
  Serial.print(":");

}

void loop(void)
{
  // call sensors.requestTemperatures() to issue a global temperature
  // request to all devices on the bus
  delay(1000);  //1 second loop
  Serial.println("");
  //Serial.print("Requesting temperatures...");
  sensors.requestTemperatures(); // Send the command to get temperatures



  // Loop through each device, print out temperature data
  for(int i=0;i<numberOfDevices; i++)
  {
    // Search the wire for address
    if(sensors.getAddress(tempDeviceAddress, i))
      {

            // It responds almost immediately. Let's print out the data
            printTemperature(tempDeviceAddress); // Use a simple function to
print out the data
```

```
      }
      //else ghost device! Check your power requirements and cabling

  }
}

// function to print a device address
void printAddress(DeviceAddress deviceAddress)
{
  for (uint8_t i = 0; i < 8; i++)
  {
    if (deviceAddress[i] < 16) Serial.print("0");
    Serial.print(deviceAddress[i], HEX);
  }
}
```

HOME_IMAGE.PHP-ESIMERKKIOHJELMA


```
<?php
// Home_image.php
//http://php.net/manual/en/function.fgets.php
$handle = fopen("/var/www/home_monitor/temp.txt", "r");//Reading a file line
by line
if ($handle){
		while (($buffer = fgets($handle, 4096)) !==false){
				//echo $buffer;
				$temps = explode (':', $buffer);
//ulko:MH1:OH:MH2

		}
		if (!feof($handle)){
				echo "Error: unexpected fgets() fail\n";
		}
		fclose ($handle);
}

//$temps = explode (':', $buffer); //ulko:MH1:OH:MH2


$home_img = "images/home.png";
$legend_img = "images/legend.png";

$img_home = @imagecreatefrompng($home_img);
$img_legend = @imagecreatefrompng($legend_img);


if ($temps[3] <= -1)//MH2
{
 $img_therm_3 = @imagecreatefrompng("images/therm_1.png");
} elseif ($temps[3] <= 5) {
 $img_therm_3 = @imagecreatefrompng("images/therm_2.png");
} elseif ($temps[3] <= 11) {
 $img_therm_3 = @imagecreatefrompng("images/therm_3.png");
} elseif ($temps[3] <= 16) {
 $img_therm_3 = @imagecreatefrompng("images/therm_4.png");
} elseif ($temps[3] <= 21) {
 $img_therm_3 = @imagecreatefrompng("images/therm_5.png");
} elseif ($temps[3] <= 27) {
 $img_therm_3 = @imagecreatefrompng("images/therm_6.png");
```

```
} elseif ($temps[3] <= 32) {
 $img_therm_3 = @imagecreatefrompng("images/therm_7.png");

} elseif ($temps[3]) {
 $img_therm_3 = @imagecreatefrompng("images/therm_8.png");
}
if ($temps[2] <= -1)//OH
{
 $img_therm_2 = @imagecreatefrompng("images/therm_1.png");
} elseif ($temps[2] <= 5) {
 $img_therm_2 = @imagecreatefrompng("images/therm_2.png");
} elseif ($temps[2] <= 11) {
 $img_therm_2 = @imagecreatefrompng("images/therm_3.png");
} elseif ($temps[2] <= 16) {
 $img_therm_2 = @imagecreatefrompng("images/therm_4.png");
} elseif ($temps[2] <= 21) {
 $img_therm_2 = @imagecreatefrompng("images/therm_5.png");
} elseif ($temps[2] <= 27) {
 $img_therm_2 = @imagecreatefrompng("images/therm_6.png");
} elseif ($temps[2] <= 32) {
 $img_therm_2 = @imagecreatefrompng("images/therm_7.png");
} elseif ($temps[2]) {
 $img_therm_2 = @imagecreatefrompng("images/therm_8.png");
}



if ($temps[1] <= -1) //MH1
{
 $img_therm_0 = @imagecreatefrompng("images/therm_1.png");
} elseif ($temps[1] <= 5) {
 $img_therm_0 = @imagecreatefrompng("images/therm_2.png");
} elseif ($temps[1] <= 11) {
 $img_therm_0 = @imagecreatefrompng("images/therm_3.png");
} elseif ($temps[1] <= 16) {
 $img_therm_0 = @imagecreatefrompng("images/therm_4.png");
} elseif ($temps[1] <= 21) {
 $img_therm_0 = @imagecreatefrompng("images/therm_5.png");
} elseif ($temps[1] <= 27) {
 $img_therm_0 = @imagecreatefrompng("images/therm_6.png");
} elseif ($temps[1] <= 32) {
 $img_therm_0 = @imagecreatefrompng("images/therm_7.png");
} elseif ($temps[1]) {
 $img_therm_0 = @imagecreatefrompng("images/therm_8.png");
}
```

```
if ($temps[0] <= -1) //OUT
{
 $img_therm_1 = @imagecreatefrompng("images/therm_1.png");
} elseif ($temps[0] <= 5) {
 $img_therm_1 = @imagecreatefrompng("images/therm_2.png");
} elseif ($temps[0] <= 11) {
 $img_therm_1 = @imagecreatefrompng("images/therm_3.png");
} elseif ($temps[0] <= 16) {
 $img_therm_1 = @imagecreatefrompng("images/therm_4.png");
} elseif ($temps[0] <= 21) {
 $img_therm_1 = @imagecreatefrompng("images/therm_5.png");
} elseif ($temps[0] <= 27) {
 $img_therm_1 = @imagecreatefrompng("images/therm_6.png");
} elseif ($temps[0] <= 32) {
 $img_therm_1 = @imagecreatefrompng("images/therm_7.png");
} elseif ($temps[0]) {
 $img_therm_1 = @imagecreatefrompng("images/therm_8.png");
}



imagecopymerge ($img_home, $img_therm_1, 260, 20, 1, 1, 19, 69, 100); //
MH1
imagecopymerge ($img_home, $img_therm_0, 560, 20, 1, 1, 19, 69, 100); //
Out
imagecopymerge ($img_home, $img_therm_2, 260, 160, 1, 1, 19, 69, 100); //
OH
imagecopymerge ($img_home, $img_therm_3, 10, 300, 1, 1, 19, 69, 100); //
MH2



$textcolor = imagecolorallocate($img_home, 0, 0, 255);
imagestring($img_home, 4, 80, 320, trim($temps[3])." C", $textcolor); // MH2
imagestring($img_home, 4, 280, 180, trim($temps[2])." C", $textcolor); // OH
imagestring($img_home, 4, 280, 40, trim($temps[0])." C", $textcolor); // MH1
imagestring($img_home, 4, 580, 40, trim($temps[1])." C", $textcolor); // OUT



imagestring($img_home, 4, 540, 150, date("F d Y H:i:s", filec-
time("/var/www/home_monitor/temp.txt")), $textcolor);
imagecopymerge ($img_home, $img_legend, 600, 280, 0, 0, 15, 291, 100);
imagestring($img_home, 2, 630, 300, " 33 .. 45", $textcolor);
```

```
imagestring($img_home, 2, 630, 335, " 28 .. 32", $textcolor);
imagestring($img_home, 2, 630, 370, " 22 .. 27", $textcolor);
imagestring($img_home, 2, 630, 405, " 17 .. 21", $textcolor);
imagestring($img_home, 2, 630, 440, " 12 .. 16", $textcolor);
imagestring($img_home, 2, 630, 475, "  6 .. 11", $textcolor);
imagestring($img_home, 2, 630, 510, "  0 .. 5", $textcolor);
imagestring($img_home, 2, 630, 545, "-10 .. -1", $textcolor);

header("Content-Type: image/png");
imagepng($img_home);
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
← ------ Home_monitor.html----------→
<html>
<head>


  <meta http-equiv="refresh" content="600">



  <link href="style.css" rel="stylesheet" type="text/css">



  <title>::HomeTemperatureMonitor::</title>

  <script language="JavaScript" src="overlib_mini.js"><!-- overLIB (c) Erik
Bosrup --></script>
</head>


<body bgcolor="#ffffff">


<div id="overDiv" style="position: absolute; visibility: hidden; z-index:
1000;"></div>


<center>

<table align="center" width="85%">


 <tbody>

    <tr>


  <td valign="top" width="50%">
  <br>

      <br>
```

```
<p>


  <big><font color="#000000"><b>Home Temperature Moni-
tor</b></font></big>
  <br>

     <small>by Mister X </small>
  <br>

     </p>


     <p>This is a small project to track the temperature in differ-ent places in
my home. For the time beign it only only has two tem-perature sensors. More
sensors will be added so I can monitor the temperature in different rooms.
You can read more about the project <a
href="http://xxxx.xxx.com/">here</a>.<br>

     </p>
     <p><br>

     <ul>


  <li>- Page auto-reloads every 2 minutes
  <meta http-equiv="refresh" content="120" >
  </li>

       <li>- Home Monitor
  </li>


     </ul>

  </td>

  <td rowspan="2" valign="top" width="40%">
  <img src="home_image.php" alt="Home Temperature Monitor" use-
map="#Home_Map" align="middle" border="0" height="625" width="650">

  </td>

 </tr>
```

```
</tbody>
</table>



</center>


</body>
</html>
                init();

                setup();

                for (;;)
                                loop();

                return 0;
}}
```