Esa-Pekka Kovalainen
Jari Porkka

HOMEeDRIVE

Mikrokontrollerilla toteutettu kodin ohjausjärjestelmä

HOMEeDRIVE

Mikrokontrollerilla toteutettu kodin ohjausjärjestelmä

Esa-Pekka Kovalainen
Jari Porkka
Opinnäytetyö
Kevät 2012
Tietotekniikan koulutusohjelma
Oulun seudun ammattikorkeakoulu

ALKULAUSE

Opinnäytetyö on tehty ubiHOME -laboratoriossa Oulun seudun ammattikorkeakoulun Raahen yksikössä. Työ on tehty syksyn 2011 ja kevään 2012 aikana. Tilaajana toimi ubiHOME-laboratorio ja työn valvojana ja ohjaajana toimi Juha Räty. Kiitämme Juha Rätyä työn tarjoamisesta ja kannustavasta ja ammattimaisesta ohjauksesta työn aikana.

Kempeleessä 9.4.2012

Jari Porkka

Esa-Pekka Kovalainen

TIIVISTELMÄ

Oulun seudun ammattikorkeakoulu
Tietotekniikan koulutusohjelma

Tekijät: Esa-Pekka Kovalainen, Jari Porkka
Opinnäytetyön nimi: HOMEeDrive
Työn ohjaaja: Juha Räty
Työn valmistumislukukausi ja -vuosi: Kevät 2012 Sivumäärä:34

Opinnäytetyön tarkoituksena oli kehittää laite tai järjestelmä, jonka avulla voitaisiin ohjata erilaisia ja erillisiä kodin elektroniikkaa ja sähkölaitteita keskitetysti yhdellä laitteella. Laitteen tulisi olla helposti muunneltava ja laajennettava vastaamaan muuttuviin tarpeisiin. Työn tilaajana toimi ubiHOME -projekti, jonka laboratoriossa kehitetään erilaisia älykkäitä järjestelmiä joita voidaan soveltaa kotiympäristön rakentamiseen helpottamaan uusien teknologioiden ja innovaatioiden soveltamista.

Aluksi tutkittiin erilaisten mikroprosessori alustojen soveltuvuutta laitteen ytimeksi. Tutkittiin PIC 16F877:ään pohjautuvaa teknologiaa, joka olisikin pääosin soveltunut järjestelmän rakentamiseen ja toteutukseen. Päädyimme kuitenkin rakentamaan järjestelmän Arduinon prosessorille, koska totesimme sen kaikilta ominaisuuksiltaan ylivoimaisesti parhaaksi laajennettavuutensa, ominaisuuksiensa ja hintansa suhteen.

Tuloksena saatiin järjestelmä, jolla voidaan ohjata Pc:ltä ASHII -koodeilla, IR -kaukosäätimellä ja puhelimella tai tabletilla bluetoohin kautta. Tässä vaiheessa toteutimme päälle – pois - ohjauksia, mutta koska jatkokehitysmahdollisuudet ovat lähes rajattomat, voidaan järjestelmällä kontrolloida ja ohjata lähes mitä tahansa kodin sähköjärjestelmää. Arduino pohjautuu avoimelle lähdekoodille ja samoin puhelimessa ja tabletissa käytetty Java- tai xml- koodi.

Asiasanat: Arduino, mikrokontrolleri, ohjausjärjestelmä

ABSTRACT

Oulu University of Applied Sciences
Information Technology

Authors: Esa-Pekka Kovalainen, Jari Porkka
Title of thesis: HOMEeDRIVE
Supervisor: Juha Räty
Term and year of completion: Spring 2012 Number of pages:34

Purpose of this Bachelor's thesis was to develop a device or system that could be used to control different and separate home electronics and electrical equipment on a centralized single device. Our client was a ubiHOME project, which develops in the laboratory a variety of intelligent systems that can be used to make new technologies and innovations easier to apply in a home environment.

Initially, we studied the various options of microprocessor platforms compatible with the device core. It was also studied a PIC 16F877 based technology, which would be the most suitable system for the construction and implementation. However, we decided to construct a system to an Arduino substrate, in an ATMega328 processor because we found that it had the best expandability, features and price ratio.

The result was a system that can be controlled from a PC using a USB port with phase, ASCII codes, an IR remote control and a phone or a tablet through Bluetooth. At this phase, we only carried out on – off controls, the analog sensor data reading and processing, and dimming. As further development possibilities are almost endless, the system can control almost any home electrical system or device. Arduino is based both on an open source code and also on a Java and XML code that are used in mobile phones and tablets.

Keywords:Arduino,Microcontroller,Controsystem

Sisällys

1 JOHDANTO 6

2 MÄÄRITELMÄ 8

2.1 Mikrokontrolleri Arduino UNO 9

2.2 Android mobiilialusta 10

3 TOIMINTAYMPÄRISTÖ 11

3.1 Mikrokontrolleri Arduino UNO ja siihen liittyvät laajennukset 11

3.2 Arduino -ohjelmointi 14

3.3 Android -alusta 15

3.4 LG-ohjelmoitava IR-kaukosäädin 18

3.5 Infrapunaohjaus 18

3.6 Bluetooth 19

4 TOTEUTUS 20

4.1 BIC16F877 20

4.2 Arduino 21

4.3 Käyttöliittymä 25

5 TESTAUS 28

5.1 Testaussuunnitelma 28

*5.2*Testauksen toteutus 29

6 JATKOKEHITYS 30

7 YHTEEVETO 31

LÄHDELUETTELO 32

LIITTEET 34

1 JOHDANTO

Opinnäytetyön aihe on saatu Oulun seudun ammattikorkeakoulun Raahen yksikön ubiHome -projektista. Projektin tarkoitus on kehittää kotiympäristöön uuteen teknologiaan pohjautuvia ratkaisuja helpottamaan esim. vanhuksien kotona asumisessa ja uusien innovaatioiden löytämisessä kotiympäristöön Opinnäytetyö on tehty syksyn 2011 ja kevään 2012 aikana.

Työn kehittely lähti liikkeelle tarpeesta kehittää laite tai järjestelmä, jolla voitaisiin yhdellä laitteella ohjata mahdollisimman monta kodin sähkölaitetta. Aluksi ajatuksena oli kotiteatterin ja siihen liittyvän ympäristön, kuten verhojen, valkokankaan, videotykin yms. laitteiden ohjaus keskitetysti. Aiheesta oli tehty yksi opinnäytetyö, mutta se oli jäänyt keskeneräiseksi ainakin kauko-ohjaimella ohjattavuuden osalta.

Ratkaisua lähdettiin hakemaan aikaisemmassa työssä käytetyn PIC 16F877-mikrokontrollerin kautta ja tutkimmekin ja osittain rakensimme siihen pohjautuvan ratkaisun. Se on yleisesti käytetty alusta, joka on edullinen ja suhteellisen helppo ohjelmoida esimerkiksi c-kielellä.

Merkittävä osa taustatyöstä oli löytää ja selvittää mikrokontrollerialusta, joka sopisi parhaiten työhön. Tehtävänasettelun lähtökohtana oli, että työ pohjautuu valmiina löytyvälle ohjelmoitavalle mikrokontrollerille ja tulee täyttämään sulautetun järjestelmän tunnusmerkit.

Tutkimme rinnalla samalla myös muita kontrollereita ja löysimme Arduinon alustan, jossa kiinnitti huomiotamme Linuxiin pohjautuva avoimeen lähdekoodiin perustuva mikrokontrollerialusta, komponenttien edullisuus ja valmis tuoteperhe, josta löytyi mm. valmis bluetooth -alusta, joka voitiin suoraan liittää mikrokontrolleriin.

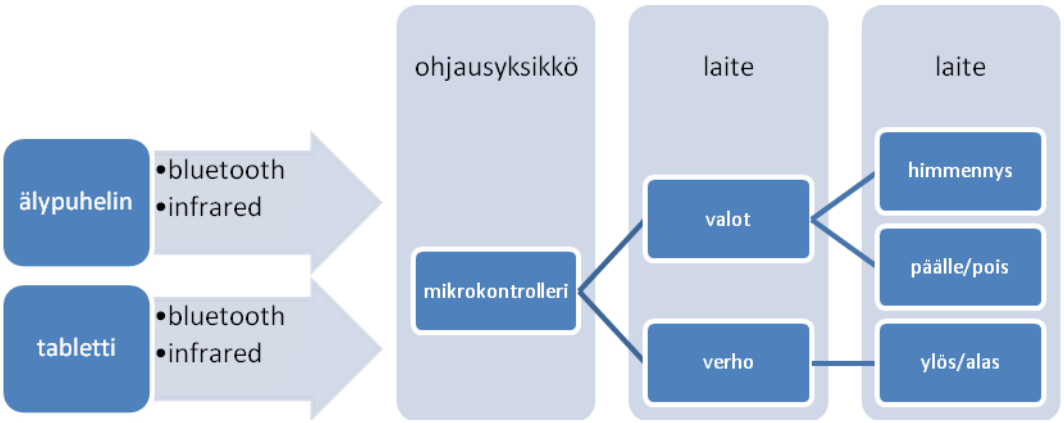
Ohjaimen langattomuus ja pelkkä IR -ohjaimen kantosäde ei ollut riittävä, joten päätimme tutkia, miten ohjaus voitaisiin integroida mobiilille päätelaitteelle, kuten tabletille ja älypuhelimelle ja hyödyntää laitteissa olevaa bluetooth -teknologiaa. Ratkaisua lähdimme hakemaan Android–alustasta, koska se on alkuperin suunniteltu mobiililaitteille. Myös se että molemmat alustat pohjautuvat avoimeen lähdekoodiin, oli merkittävässä roolissa, kun päätös ympäristöstä, jolle kokonaisuus lähdettäisiin rakentamaan, tehtiin. Toinen merkittävä tekijä oli taloudelliset

näkökohdat, koska järjestelmä on tarkoitus tuotteistaa. Molemmat alustat palvelevat tätä tavoitetta.

2 MÄÄRITELMÄ

Tämän opinnäytetyön tavoitteena on kehittää ohjausjärjestelmä, jolla voidaan ohjata keskitetysti erilaisia ja erillisiä kodin sähköjärjestelmiä. Lähtökohtana oli rakentaa järjestelmä, jolla voidaan yhdellä laitteella keskitetysti ohjata mikrokontrolleriavusteisesti kotiteatterijärjestelmää, valoja ja verhojen moottoreita. Mikrokontrollerialustana käytetään Arduino–alustaa, johon on saatavilla paljon erilaisia lisäkomponentteja, kuten suoraan siihen sopiva IR ja bluetooth -alusta, jota työssä hyödynnetään.

Järjestelmän tulee olla sellainen, että käyttäjä ei tarvitse erityisiä taitoja kyetäkseen käyttämään järjestelmää. Myös uusien laitteiden lisäämisen tulee olla helppoa.



Kuva 1. Hahmotelma järjestelmästä

Ohjaus tulee voida suorittaa kaukosäätimellä, älypuhelimella tai tabletilla. Kaukosäädin ohjaa Infrared -ohjauksella, älypuhelin ja tabletti bluetoothin kautta. Tabletin ja älypuhelimen tulee käyttää sellaista käyttöjärjestelmää, että se on helposti muunneltavissa uusia toimintoja ja laitekokonaisuuksia varten.

Työ toteutetaan tässä vaiheessa, siten että rakennetaan yksinkertaiset päälle/pois- ja himmennysohjaukset Android -käyttöliittymään, jolla ohjataan mikrokontrolleria, joka ohjaa, ledejä, jotka simuloivat aitoa ohjausympäristöä. Lähtökohtana on, että ratkaisut perustuvat avoimeen koodiin, mikrokontrollerialustat ovat edullisia ja helposti laajennettavissa.

2.1 Mikrokontrolleri Arduino UNO

Arduino on Atmelin prosessoreilla toteutettu mikrokontrolleri tuoteperhe, johon kuuluu useita alustoja ja laajennusmahdollisuuksia kuten bluetooth, rf, ir, gps –loggeri, moottoriohjauksia ja huomattava määrä muita valmiita laajennuksia, joilla voidaan rakentaa, tarpeesta riippuen, erilaisia ohjauskokonaisuuksia Alustan laatu on ammattikäyttö tasoinen, mutta hinnaltaan harrastelijoidenkin saavutettavissa. Alustan ohjelmointiin on saatavilla ilmainen kehitystyökalu, joka on ladattavissa Arduinon sivustolta.

2.2 Android mobiilialusta

Android on avoimeen lähdekoodiin perustuva, lähinnä mobiililaitteille tarkoitettu käyttöjärjestelmä, joka on Googlen julkaisema. Se sisältää käyttöjärjestelmän ja erilaisia sovelluksia. Android pohjautuu GPL–lisensoituun Linux- käyttöjärjestelmän ytimeen. Android -käyttöjärjestelmän perustana on Java-ohjelmointirajapinta. Google on luonut oman ohjelmistokehitysympäristön SDK :n (Software Development Kit). Kehitysympäristö sisältää tarvittava työkalut mm. kääntäjän, debuggerin ja kaikki ohjelmistokehityksessä tarvittavat työkalut.

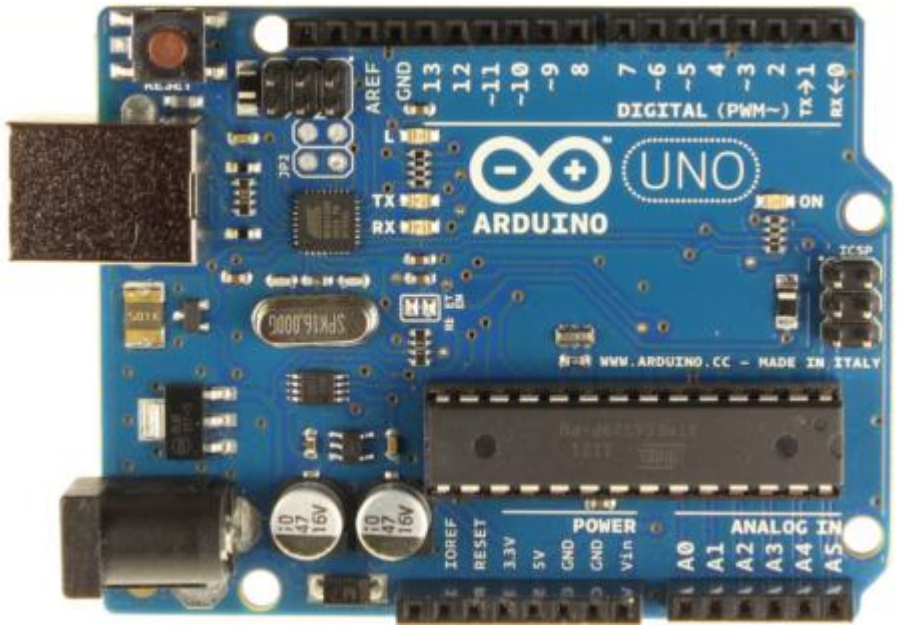
Laitteiden virtuaalitestausta varten on Dalvikin emulaattori (Dalvik Virtual Macine). Eclipse (Eclipse Foundation) -ohjelmointialusta, joka pohjautuu myöskin avoimeen koodiin, on ottanut paikkansa Android kehitysalustana. Ympäristö on rakennettu Java -ohjelmointikielellä, johon ohjelmointiin se on alun perin kehitetty. Nykyisin siinä on laajennukset myös muihin tunnettuihin kieliin, kuten esimerkiksi C++:aan.

Eclipseen voi liittää Dalvikin emulaattorilaajennuksen, jolla voi testata luotuja sovelluksia virtuaaliympäristössä. Arduino ja Android perustuvat avoimeen lähdekoodiin, kehitystyökalut ovat ilmaisia ja molempien taustalla on laaja kehittäjäyhteisö.

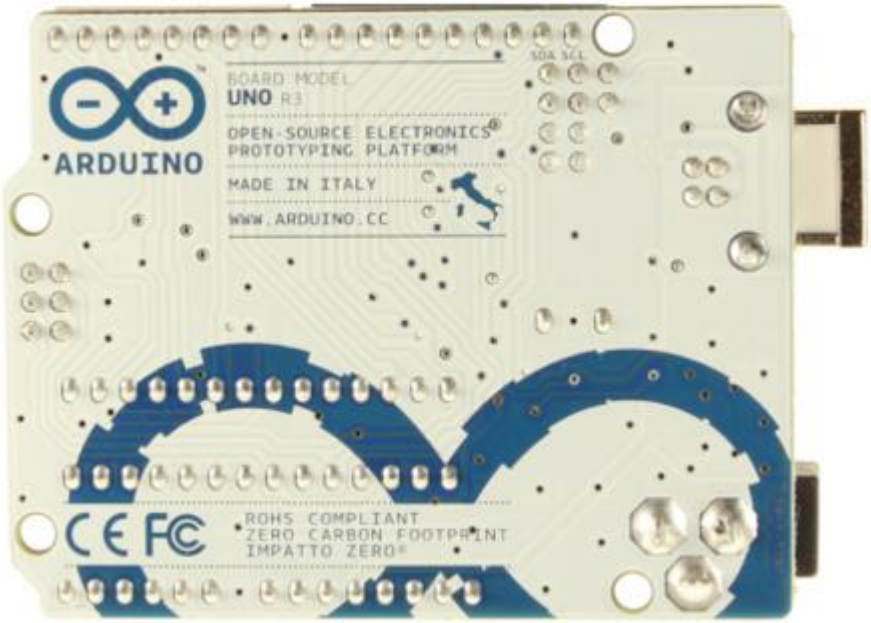
3 TOIMINTAYMPÄRISTÖ

Opinnäytetyössä kehitetään ohjausjärjestelmä, jolla ohjataan kodin sähkölaitteita yhdellä ohjaimella keskitetysti. Tämä järjestelmä on tarkoitettu ensisijaisesti käytettäväksi kotiympäristössä. Pohjana on Arduino–mikrokontrolleri, johon tehtävänannossa määritellyt ohjauskäskyt ohjelmoidaan. Ohjausväylänä käytetään bluetooth- ja infrared -väyliä ja myös usb–väylän käyttö on mahdollinen.

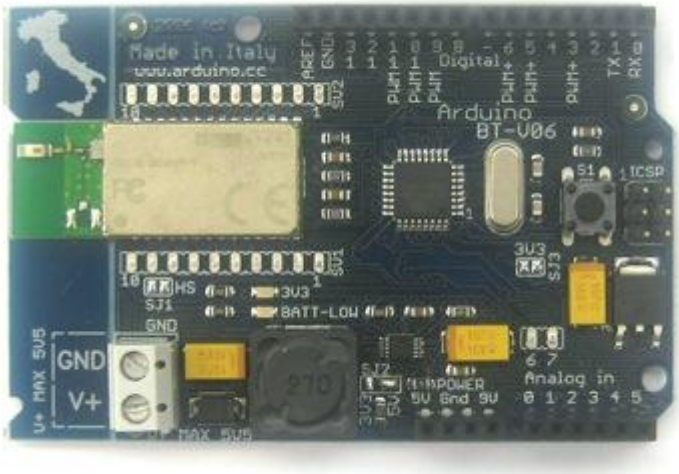
3.1 Mikrokontrolleri Arduino UNO ja siihen liitettävät laajennusalustat



Kuva 2. Arduino UNO R3 -mikrokontrolleri, edestä



Kuva 3. Arduino UNO R3 -mikrokontrolleri takaa



Kuva 4. Arduino bluetooth -adapteri

Kuva 5. RN-42 bluetooth -adapteri

Kuva 6. Ifrared -adapteri

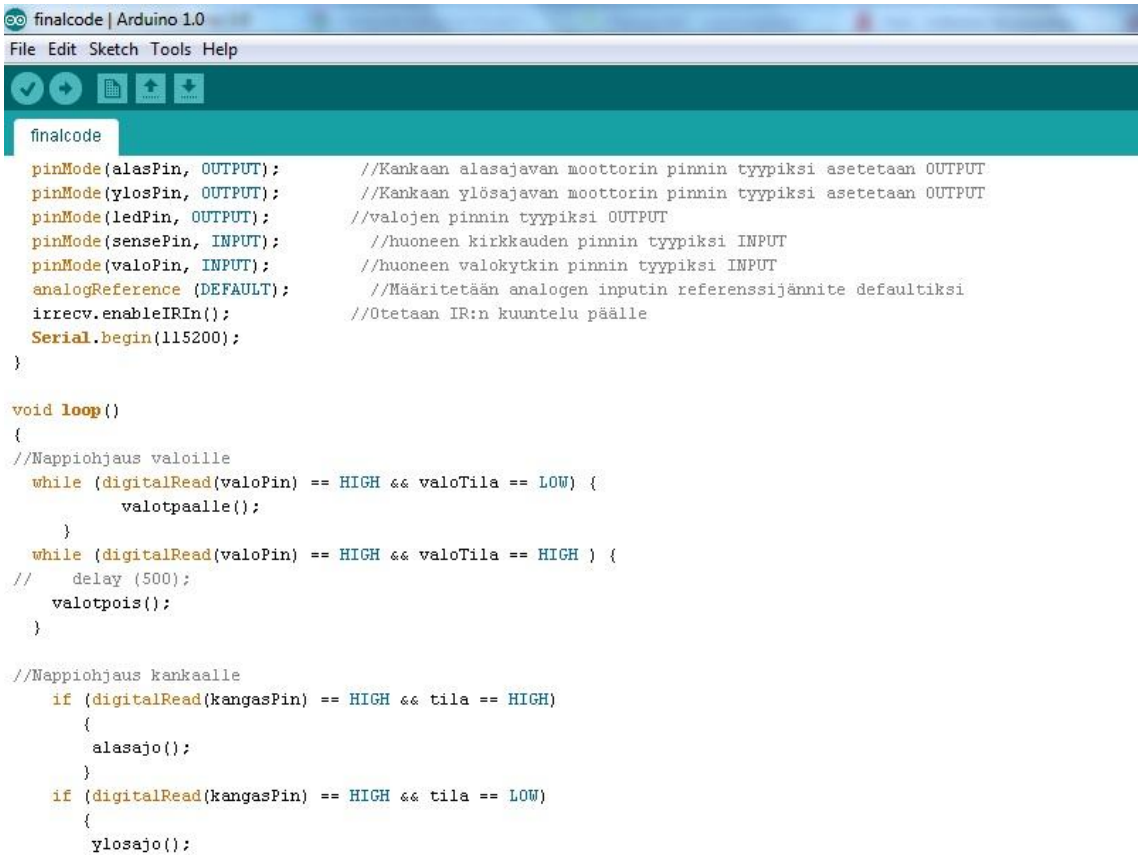
3.2 Arduino ohjelmointialusta

Ohjelmisto

Arduinossa on vain kaksi pääfunktiota:

setup()– funktio, joka alustaa laitteen asetukset.

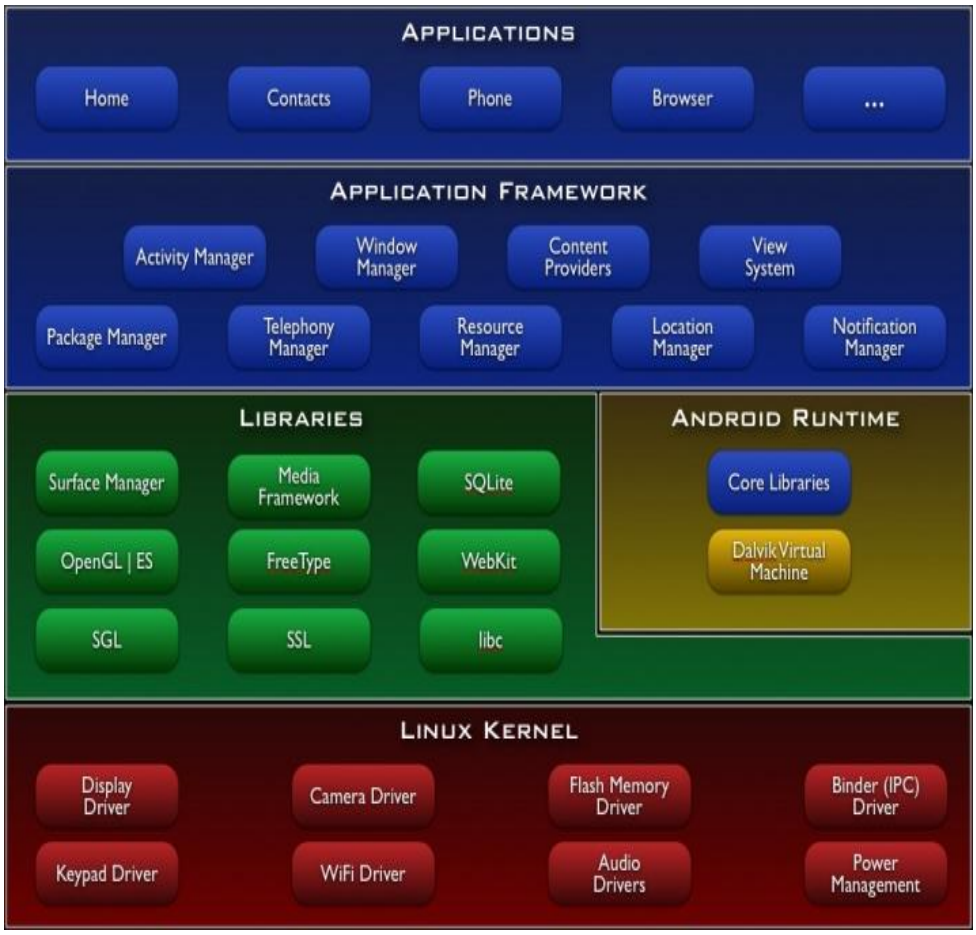
loop()– funktio, jota toistetaan virran sammuttamiseen asti.



Kuva 7. Näkymä Arduino -editorista ja työssä käytettyä koodia

3.3 Android

Alla olevassa kuvassa on esitetty Androidin arkkitehtuuri



Kuva 8. Adroid -arkkitehtuuri
http://developer.android.com/guide/basics/what-is-android.html

Android -alusta sisältää kaikki mobiiliohjelmoinnissa tarvittavat elementit. Se on ohjelmistopino, joka sisältää mm. Linuxin, Bionic-C-kirjaston ja järjestelmäkirjaston. Linux on ytimenä.

Tiedon tallentamiseen käytetään SQLite -relaatiotietokantaa. Android tukee mm. GSM-, CDMA-, UMTS-, Bluetooth-, Wi-Fi- ja WiMAX -teknologioita.

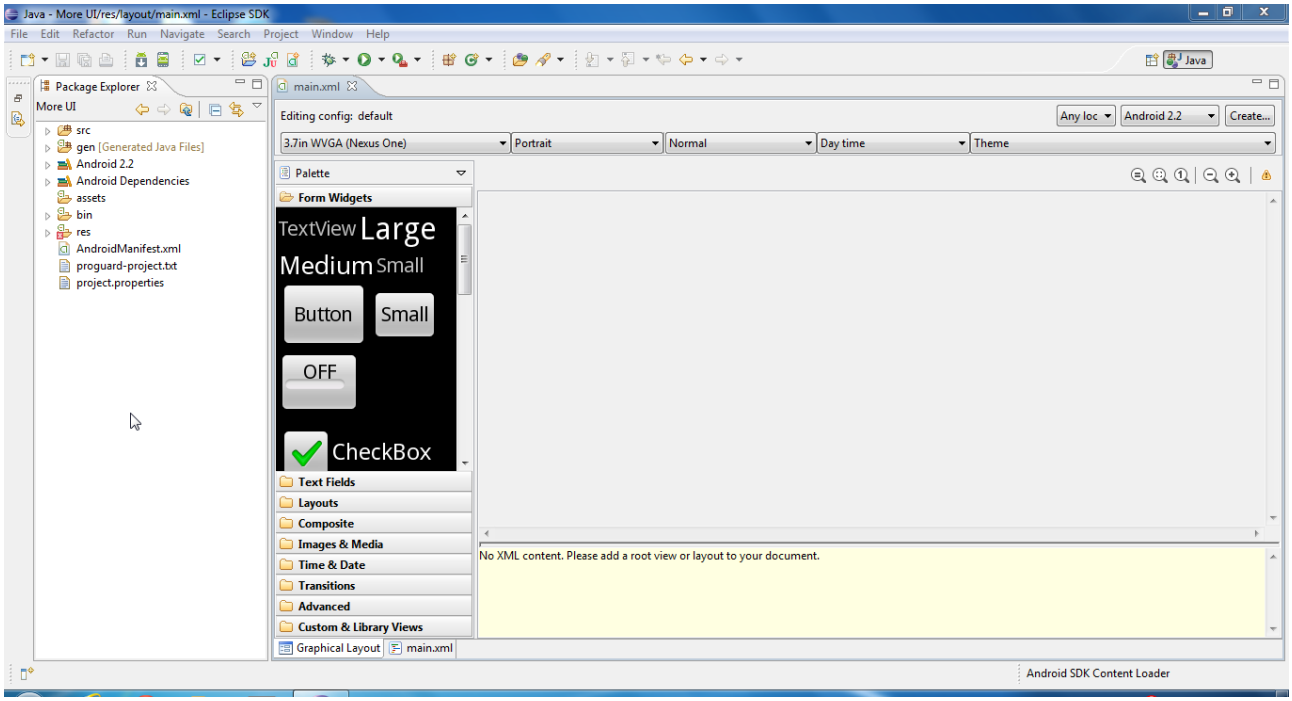
Android on alun perin julkistettu 2007 Open Handset Alliancen perustamisen myötä. Se koostuu 84 teleoperaattorista, ohjelmisto- ja laitteistovalmistajasta. Google on julkaissut suuren osan Android -koodista avoimen koodin ja vapaan ohjelmiston Apache -lisenssillä.
Androidin versio 1.1 julkistettiin 9.2 2009, se on siis varsin uusi alusta. Viimeisin versio on 4.0. Julkaistu19.10.2011.Lähde:fi.wikipedia.org/wiki/Android.
http://developer.android.com/guide/basics/what-is-android.html, http://www.android.com/

Asennetaan uusin versio JDK :sta (Java developmen kit)
http://www.oracle.com/technetwork/java/javase/downloads/index.html
Tämä on perustyökalupakki Java–pohjaiseen ohjelmointiin.

Ladataan Eclipse -ohjelmointiympäristö osoitteesta
http://www.eclipse.org/
Lisätään samalta sivulta Eclipseen Java Development Tools
Tämän jälkeen käynnistetään Eclipse ja tarkistetaan plash–ruudusta, että on asennettu viimeisin versio. Tällä hetkellä se on (02.05.2012) Indico ja näyttää seuraavalta:

Kuva 9. Eclipse

Ja itse editointi ympäristö seuraavalta:



Kuva 10. Näkymä editorista

Käyttöliittymä rakennettaan widgeteistä. Ne ovat käyttöliittymäelementtejä joita yhditelemällä kootaan käyttöliittymä. XML-kieli on rakenteellinen kuvauskieli, joka auttaa jäsentämään laajoja tietomassoja selkeämmin. XML:n kehittäjä on World Wide Web Consortium.

Rakenteinen tieto käsittää mm. taulukkolaskennan tiedostot, osoitekirjat, konfiguraatiotiedostot, sähköisen kaupankäynnin viestit sekä tekniset piirustukset. XML on joukko sääntöjä rakenteisen tiedon esittämiseen tarkoitettujen tekstiformaattien suunnitteluun. (XML:n voi nähdä myös ohjeena tai yhteisenä käytäntönä.) XML ei ole ohjelmointikieli eikä soveltajan tarvitse osata ohjelmoida pystyäkseen käyttämään sitä. XML:n avulla tietokoneiden on helppoa tuottaa ja lukea tietoa täsmällisessä muodossa. XML välttää suunnittelun tyypilliset sudenkuopat: se on laajennettavissa, järjestelmäriippumaton ja se tukee kansainvälistämistä ja lokalisointia. XML on myös täysin unicode-yhteensopiva. www.w3c.tut.fi/index.html

3.4 LG ohjelmoitava ir -kaukosäädin

Opetettava kaukosäädin ir–ohjaukseen

Kuva 11. IR–kaukosäädin

3.5 Infrapuna–ohjaus

Infrapunasäteily on sähkömagneettista säteilyä, jonka aallonpituus on suurempi kuin näkyvän valon. Infrapunasäteilyä hyödynnetään mm. kaukosäätimissä, joilla välitetään ohjauksia erilaisille laitteille. Säteilylähteenä on ifrapunadiodi. Sen teho on pieni ja kantomatka vain muutamia metrejä ja välissä ei saa olla esteitä. IrDA 1.0- ja 1.1-standardien mukaiset laitteet toimivat vain yhden metrin etäisyydellä. 1.1-standardilla saavutetaan kuitenkin jopa 4 Mbit/s tiedonsiirtonopeus. Uusilla tekniikoilla nopeus tulee kohoamaan jopa 50 Mbit/s, joka on käyttökelpoinen nopeus raskaammassakin tiedonsiirrossa.
Lähde:(wikipedia.org/wiki/infrapunasäteily)

3.6 Bluetooth

Bluetooth on radioteknologiaan perustuva tiedonsiirtomenetelmä ja sen kantomatka on n. 10 metriä 2.0 -versiolla. Bluetooth on luotettavampi kuin infrapuna–teknologia, koska se toimii ilman näköyhteyttä. Nykyisin käytössä oleva 2.0 versio siirtää tietoa 3.0 Mbit/s. 2,5 mW teholla siis tiedonsiirtoetäisyys on 10 m ja nostamalla teho 100 mW: iin nousee kantama jopa 100 m:iin, jolloin se kilpailee jo lähiverkon kanssa. Se on ohjauslaitteena varsin käyttökelpoinen monissakin eri järjestelmissä, eteenkin kotiympäristössä. Kun pian laitteisiin tuleva standardi 3.0 saadaan käyttöön, tulee se mullistamaan lähitiedon siirron, sillä sen ytimenä toimii 802.11 PAL (Protocol Adaption Layer), eli 802.11 WLAN yhteyden hyödyntäminen. Siirtonopeus kasvaa jopa 24 Mbit/s. Lähde: (wikipedia.org/wiki/bluetooth)

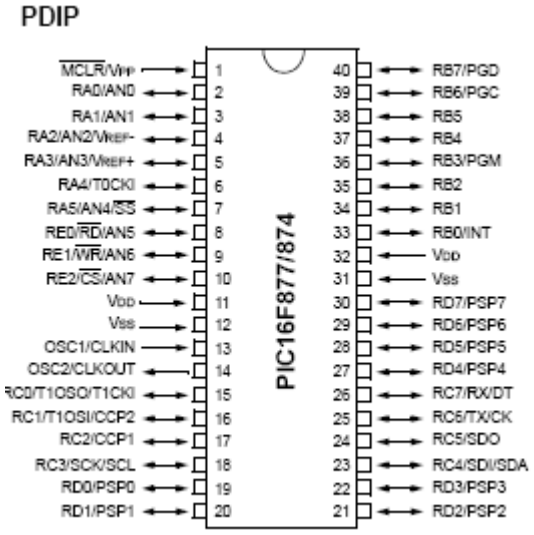
4 TOTEUTUS

4.1 BIC 16F877

Kun ratkaisua lähdettiin hakemaan, millä alustalla työ toteutettaisiin, oli vahvasti esillä PIC 16F877-mikrokontrolleri, joka on edullinen ja paljon käytetty, luotettava alusta. Myös ohjelmointityökalut siihen ovat verkosta saatavilla. Ohjelmointikielenä tulisi käyttää C–kieltä tai symbolista konekieltä.

PIC16F877 on RISC(eng. Reduced Instruction Set Computer), joka on yleinen prosessoriarkkitehtuuri. Siinä käskykannat pidetään yksinkertaisina ja ne suoriutuvat nopeasti tehtävistä.(Haltsonen, Rautiainen, Tietokonetekniikka).

PIC F877 on 8-bittinen ja sisältää 35 käskyä. Se sisältää 8 Kb Flash-muistia, 256b EPROM-(Erasable Programmable Read Only Memory) ja 368b- datamuistia. Nämä ominaisuudet ovat täysin riittävät opinnäytetyön tarpeisiin.



Kuva 12. PIC16F877–liitännät

Koska tehtävän annossa oli määritelty, että mikrokontrolleria tulisi kyetä ohjaamaan IR– ja bluetooth -ohjauksella ja sen tulisi kyetä myös välittämään samoilla medioilla annettuja käskyjä, tutkimme, miten ja millaisilla kontrolleriin liitettävillä komponenteilla se olisi mahdollista.
Aikaisemmassa (Tero Koivunen, 2009) tehdyssä opinnäytetyössä oli aihetta tutkittu ja ei ollut hyvin toimivaa ratkaisua löydetty. Työssä oli käytetty erillistä IR–vastaanotinta, joka oli kytkentälangoilla kytketty PIC:iin. Nämä molemmat vaativat myös erillisen virtalähteen. Lisäksi toimintaetäisyys oli jäänyt vaatimattomaksi, noin kahteen metriin.
http://ww1.microchip.com/

4.2 Arduino

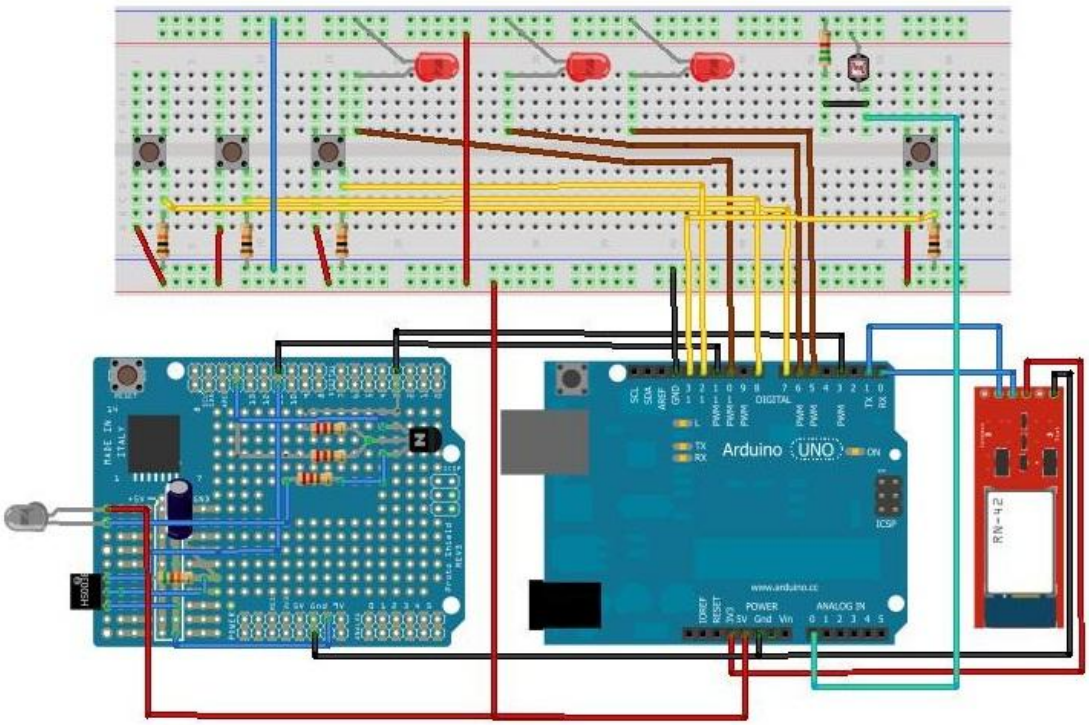
Internetin elektroniikka harrastajien sivuja selailemalla huomiomme kiinnittyi Arduino–alustaan. Harrastajien ja ammattilaisten piirissä se oli saanut lähes poikkeuksetta kiittäviä arvosteluja ja myös valmis laaja tuoteperhe herätti kiinnostuksen.

Arduino -alustat ovat yleisesti ottaen helppoja ohjelmoitavia, sillä useista versioista löytyy FTDI :n USB-sarjamuunninpiiri. Erillistä ohjelmointikaapelia ei tarvita, joten ainoa investointi on USB-kaapeli.
Seuraavassa taulukossa vertailu alustoista:

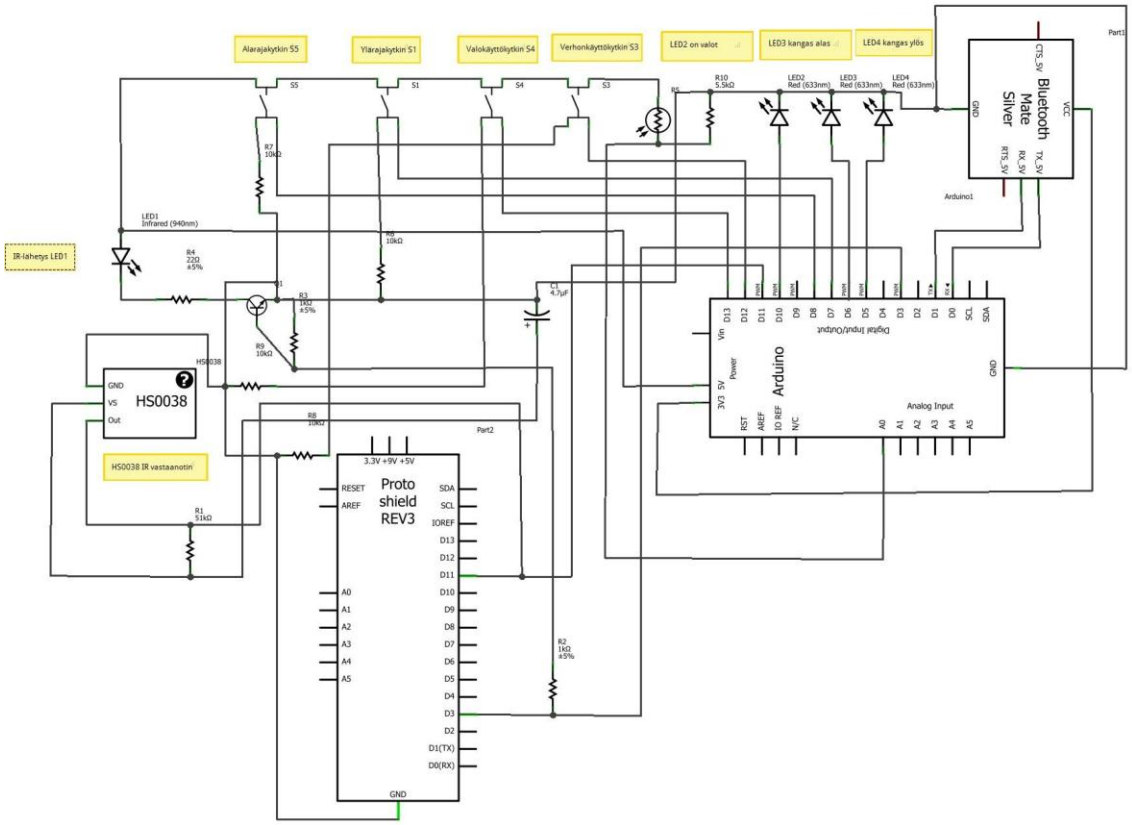
Taulukko 1 Vertailu taulukko PIC16F887 ja Arduino UNO

	PIC	UNO
	8-BIT	8-BIT
Arkkitehtuuri	RISC	ATmega328
flash	8-kt	32 kt
Datamuisti	368 t	2 kt
EEPROM datamuisti	256 t	1 kt
Digitalisia liitäntöjä	14 kpl / A / B / C / D / E	14 digitaalista in/out
	A = 16 bit, B,C,D=8bit, E=3bit	joista 6 kpl pwm
Analogisia liitäntöjä		6 kpl input
Max	20 MHz	16MHz
Ohjelmointi		

Selvitystyön tuloksena saatujen tietojen pohjalta päätimme, että Aduino on ehdottomasti se alusta, jolle prototyyppi ohjausjärjestelmästä rakennetaan. Tilattuamme komponentit rakensimme seuraavanlaisen demonstraation:



Kuva 13. Alustojen kytkentä



Kuva 14. Alustojen kytkentäkaavio

Alustalle ohjelmoitiin seuraavat toiminnot:

Toiminnot

1. Kangas ylös
2. Kangas alas
3. Valot
4. Valojen kirkastus
5. Valojen himmennys

Anturit

1. Valoisuus (fotoresistori)
2. Kankaan raja-anturit (ylä- ja alakuolo)

Ohjaustavat

1. Bluetooth
2. Infrapuna
3. Kankaan ohjaus napista (1 nappi, vuorottelee ylös/alas)
4. Valojen ohjaus (1 nappi, himmennettävissä)

Toiminta

1. Järjestelmää voidaan ohjata alustalla olevista napeista IR-säätimellä tai bluetoothin välityksellä. Bluetooth -ohjaus onnistuu PC-, tablet- tai älypuhelimilla. Bluetooth -ohjaus toimii bluetoothin välityksellä ASCII-koodeilla.

2. IR-kaukosäätimeksi valitsimme LG:n kaukosäätimen, jonka lähettämiä IR-koodeja opetimme ohjelmalle.

3. Kaukosäätimellä voidaan ohjata kangas alas tai ylös. Kangas pysähtyy automaattisesti ääriasentoon, kun rajakytkin antaa ohjelmalle keskeytyksen.

4. Kaukosäätimellä voidaan sytyttää ja sammuttaa valot huoneen valoisuuden mukaan sopivaksi (kun on hämärää, tulee kirkkaammat ja kun on kirkasta, tulee himmeämmät)

5. Bluetoothin välityksellä käyttäjä voi ohjata kankaan alas tai ylös. Kangas pysähtyy automaattisesti ääriasentoon, kun rajakytkin antaa ohjelmalle keskeytyksen.

6. Bluetoothin välityksellä käyttäjä voi sytyttää ja sammuttaa valot huoneen valoisuuden mukaan sopivaksi (kun on hämärää, tulee kirkkaammat ja kun on kirkasta, tulee himmeämmät)

7. Alustalla olevilla kankaan ohjausnapilla voidaan kangas laskea/nostaa yhtä näppäintä painamalla. Kangas pysähtyy automaattisesti oikeaan kohtaan.

8. Alustalla olevalla näppäimellä voidaan portaattomasti säätää valoisuuden voimakkuutta ja sammuttaa valot.

   Koodaus toteutettiin Arduinon omalla editorilla, koodiesimerkit liitteissä.

4.3 Käyttöliittymä

Käyttöliittymää pohdittiin aluksi uuden Nokian Windows Phone -käyttöjärjestelmän, Qt–kehitysalustan ja Androit–käyttöjärjestelmän välillä.
Windows Phone -käyttöjärjestelmä sulkeutui pois, koska siitä ei ollut käyttökokemusta ja oletuksena oli, että jos rakentamamme ohjausjärjestelmä tuotteistetaan, ei ole mitään käsitystä millaisia lisenssi- yms. maksuja käyttöjärjestelmän hyödyntämisestä tulee. Qt on toimiva järjestelmä ja se skaalautuu monelle alustalle, mutta se on ilmeisesti jäämässä jonkinlaiseen marginaaliin laitemaailmassa, varsinkin mobiililaitteiden osalta.

Käyttöliittymä Android laitteille luotiin Eclipsen editorilla. Siihen rakennettiin
yksinkertaiset On / Off -painikkeet, joilla mikrokontrolleria ohjattiin bluetoothin välityksellä. On /Off -painikkeet saatiin toimimaan moitteettomasti mutta käyttöliittymään tarkoitettu liukukytkin ei toiminut (valojen himmennys). Sekin toiminto toimi ASCII -koodeilla ohjattuna suoraan näppäimistöltä, mutta käyttöliittymän kautta se ei toiminut.

Eclipcellä luodut widgedeiksi, kutsutut käyttöliittymä elementit pohjautuvat XML–koodiin. XML on tarkoitettu rakenteisen tiedon esittämiseen.

Seuraavana käyttöliittymän layout:

Kuva 15. Käyttöliittymä

Koodiesimerkki :

```
<Button
                android:id="@+id/button1"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="LIGHT" />
```

Tämä koodinpätkä lisäämällä ohjelmaan, se parittaa laitteet, etsii siis bluetooth- laitteet, joihin voidaan yhdistää.

```
T
Set<BluetoothDevice> pairedDevices =
mBluetoothAdapter.getBondedDevices();
// If there are paired devices
if (pairedDevices.size() > 0) {
    // Loop through paired devices
    for (BluetoothDevice device : pairedDevices) {
        // Add the name and address to an array adapter to show in a
ListView
        mArrayAdapter.add(device.getName() + "\n" +
device.getAddress());
    }
}
```

```
Set<BluetoothDevice> pairedDevices =
mBluetoothAdapter.getBondedDevices();
// If there are paired devices
if (pairedDevices.size() > 0) {
    // Loop through paired devices
    for (BluetoothDevice device : pairedDevices) {
        // Add the name and address to an array adapter to show in a
ListView
        mArrayAdapter.add(device.getName() + "\n" +
device.getAddress());
    }
}
```

Lähde:http://developer.android.com/guide/topics/wireless/bluetooth.html

5 Testaus

5.1 Testaussuunnitelma

Sovittiin, että testaus etenee sitä mukaan, kun eri työvaiheet valmistuvat. Aluksi tietenkin varmistetaan, että komponentit ovat oikeat, varmistetaan elektroniikka ja mikrokontrollerialustan toiminta, liitokset, alustan kuormitus.

Ohjelmisto ja editoritestaus:

Testataan editorit sekä Arduinon että Androidin valmiilla koodeilla.
Testataan luodut koodit editorilla, Eclipse näyttää osan virheistä debuggauksessa.

Valmiiden koodien testaus laiteohjauksessa:
Testataan valmistuneet koodit osissa ja jatketaan siihen asti, kun on valmis.

Lopuksi testaan koko järjestelmän toiminta ohjaus ohjaukselta ja vaihe vaiheelta

5.2 Testauksen toteutus ja tulokset

Koska demonstraatio koottiin elektroniikan testaus-alustalle, aiheuttivat jotkin liitokset häiriötä komponenttien toiminnalle. Ne korjattiin pudistamalla liitokset huolellisesti ja käyttämällä elektroniikan puhdistusainetta. Apuna käytettiin myös yleismittaria kun etsittiin viallisia komponentteja. Mikrokontrolleri ja elektroniikka- alusta saatiin toimimaan aika vähällä vaivalla.

Androidin ja Arduinon ohjelmointi-editorien toiminta testattiin valmiilla koodeilla ja luotu koodi testattiin heti lyhyissä pätkissä ja virheet haettiin esiin saman tien.
ASCII -koodeilla ohjaus toimi näppäimistön kautta oikein, kuten myös infrapuna-ohjaimen kautta.

Kun testattiin Android -laitteilla, tabletilla ja puhelimella, muut toiminnot toimivat oikein, mutta liukukytkin, jolla ohjataan valojen kirkkautta ei toiminut. Syytä ei tässä vaiheessa saatu selville. Kysymyksessä voi olla joko laiteperäinen vika tai se on koodissa.

Testauksen lopputuloksena todettiin, että laite toimii tehtävänannon mukaisesti paitsi Android -käyttöliittymän bluetooth -himmennyksen osalta. Toiset käyttöliittymät toimivat kaikilta osin oikein.

6 JATKOKEHITYSMAHDOLLISUUDET

Löydettyämme Arduino -alustan huomasimme pian, että sen käyttömahdollisuudet ovat monipuoliset. Syntyi idea kodin käyttöliittymästä, jolla mikrokontrolleripohjaisesti ohjataan koko taloa, kerätään lokia esim. kosteiden tilojen antureiden toiminnasta, energian kulutuksesta yms. ja jolla voidaan ohjata toimintoja energiaa säästävämpään suuntaan. Arduinoon voidaan helposti liittää massamuistiyksikkö, jolla tiedon keruuta ja erilaisia ohjaustoimintoja voidaan toteuttaa. Arduino on laaja tuoteperhe, joka jo itsessään antaa kattavat mahdollisuudet toteuttaa erilaisia produktioita. Myös avoimeen koodiin perustuva laajan kehittäjäjoukon tuki on käytössä, kaikkea ei tarvitse ratkaista itse. Alustojen edulliset hinnat antavat tavalliselle kehittäjälle hyvät mahdollisuudet toteuttaa innovaatioita.

Käyttöliittymän pohjana oleva Android alusta on kehityskelpoinen ja kovaa vauhtia laajeneva ympäristö maailmalla. Lukuisat laitevalmistajat, jotka käyttävät Androidin käyttöjärjestelmää, pitävät huolen, että laitteet pysyvät edullisina ja niiden päälle voi rakentaa jatkossakin hyvin toimivia järjestelmiä.

Jatkokehitysprojektina olemme käynnistäneet prosessin, jossa rakennetaan omakotitaloon järjestelmä, jolla ohjataan talon valaistus, lukitus, murtohälytysjärjestelmä ja joitakin muitakin kohteita, jotka ovat vielä kehitysvaiheessa. Järjestelmä pohjautuu Arduinoon ja Androidiin.

Kun bluetoothin standardi 3.0 otetaan käyttöön, antaa se järjestelmälle uusia ulottuvuuksia. Myös dadasähkön käyttö voisi tulla kysymykseen ja se on meillä jatkokehitystutkimuksen kohteena.

7 YHTEENVETO

Luotiin mikroprosessori -pohjainen ohjausjärjestelmä, jolla voidaan ohjata ja hallita kodin sähkölaitteita. Järjestelmä soveltuu, ja voidaan räätälöidä käyttäjän tarpeiden mukaiseksi. Se voi toimia täysin itsenäisenä sulautettuna järjestelmänä tai käyttäjän ohjausten mukaan tai niiden yhdistelmänä.

Opinnäytetyön tehtävänannon tavoitteet saavutettiin. Työn tarkoituksena oli kehittää ohjausjärjestelmä, jolla voidaan ohjata keskitetysti kodin sähkölaitteita.
Työssä käyttämämme Android -alusta mahdollistaa käyttöliittymän tekemisen lähes kaikille avoimen koodin laitteille. Sen päälle voidaan rakentaa ohjaukset yhdistäen olemassa olevan laiteympäristön uuteen käyttöliittymään.
Arduino pohjautuu samaan käytettävyysfilosofiaan kuin Android. Helppokäyttöisyys, laajennettavuus ja edullisuus ovat keskeinen anti näissä ympäristöissä.

Tämän projektin huonona puolena oli, että se meinasi lähteä rönsyilemään, kun alustojen tarjoamat mahdollisuudet alkoivat perehtymisvaiheessa selvitä. Tehtävä alkoi viedä mukanaan, toi paljon uusia ideoita ja avarsi uusia näkökulmia kotiympäristön teknisistä ratkaisuista. Tämä työ on ollut eräänlainen yhteenveto koulutuksestamme. Työssä yhdistyy sulautetun järjestelmän suunnittelu ja toteutus, käyttöliittymäsuunnittelu ja toteutus, mobiiliohjelmointi ja laitesuunnittelu. Kaiken kaikkiaan antoisa ja opettavainen projekti.

LÄHDELUETTELO

. Lähde 1: Android developer

http://developer.android.com/guide/basics/what-is-android.html

(noutopäivä 1.12.2011)

Lähde 2:Wikipedia

(noutopäivä 2.12.2011)

fi.wikipedia.org/wiki/Android.

Lähde 2: Android

http://www.android.com/

(noutopäivä 2.12.2011)

Lähde 3: Oracle

http://www.oracle.com/technetwork/java/javase/downloads/index.html

(noutopäivä 2.12.2011)

Lähde 4: Eclipse

http://www.eclipse.org/

(noutopäivä 2.12.2011)

Lähde 5: w3c

www.w3c.tut.fi/index.html

Lähde 6: Wikipedia.org

(wikipedia.org/wiki/bluetooth)

(noutopäivä 12.1.2012)

Lähde 7:Android

http://developer.android.com/guide/topics/wireless/bluetooth.html

(noutopäivä12.1.2012)

Lähde 8: Veikko Tapanisen Android luennot

wiki.raahe.oamk.fi/bin/view.pl/Public/R2670DP_2012

Lähde 9: Microchip

http://ww1.microchip.com/

(noutopäivä 11.11.2011)

Lähde 10: Reto Meier, Professional Android 2 Development
http://ezp.oamk.fi:2048/login?url=http://site.ebrary.com/lib/oamk/docDetail.action?docID=10366621

Lähde 11: Seppo Haltsonen / Esko T Rautiainen Tietokonetekniikka

LIITTEET

Liite 1. Arduino UNO Dadalehti

Liite 2.PIC 16F887 Datalehti

Liite 3. Arduino koodit

Liite 4. Android koodit

Liite 1.

Arduino UNO Datalehti

Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode. Revision 3 of the board has the following new features:

- *1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.*
- *Stronger RESET circuit.*
- *Atmega 16U2 replace the 8U2.*

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino,

moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Schematic & Reference Design

EAGLE files: arduino-uno-Rev3-reference-design.zip (NOTE: works with Eagle 6.0 and newer)
Schematic: arduino-uno-Rev3-schematic.pdf
Note: The Arduino reference design can use an Atmega8, 168, or 328, Current models use an ATmega328, but an Atmega8 is shown in the schematic for reference. The pin configuration is identical on all three processors.

Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- ***VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.*
- ***5V.**This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.*
- ***3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.*
- ***GND.** Ground pins.*

Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead()functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal

pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- ***Serial: 0 (RX) and 1 (TX).*** *Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.*
- ***External Interrupts: 2 and 3.*** *These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.*
- ***PWM: 3, 5, 6, 9, 10, and 11.*** *Provide 8-bit PWM output with the analogWrite() function.*
- ***SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).*** *These pins support SPI communication using the SPI library.*
- ***LED: 13.*** *There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.*

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- ***TWI: A4 or SDA pin and A5 or SCL pin.*** *Support TWI communication using the Wire library.*

There are a couple of other pins on the board:

- ***AREF.*** *Reference voltage for the analog inputs. Used with analogReference().*
- ***Reset.*** *Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.*

See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the ***Tools > Board*** *menu (according to the microcontroller on your board). For details, see the reference and tutorials.*

The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- *On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.*
- *On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.*

You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of theATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

LIITE 2

PIC 16F887 datalehti

Devices Included in this Data Sheet:
High-Performance RISC CPU:
• Only 35 single-word instructions to learn
• All single-cycle instructions except for program branches, which are two-cycle
• Operating speed: DC – 20 MHz clock input DC – 200 ns instruction cycle
• Up to 8K x 14 words of Flash Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
• Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers
Peripheral Features:
• Timer0: 8-bit timer/counter with 8-bit prescaler
• Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
• Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
• Two Capture, Compare, PWM modules
- Capture is 16-bit, max. resolution is 12.5 ns
- Compare is 16-bit, max. resolution is 200 ns
- PWM max. resolution is 10-bit
• Synchronous Serial Port (SSP) with SPI™ (Master mode) and I2C™ (Master/Slave)
• Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
• Parallel Slave Port (PSP) – 8 bits wide with external RD, WR and CS controls (40/44-pin only)
• Brown-out detection circuitry for Brown-out Reset (BOR)
Analog Features:
• 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
• Brown-out Reset (BOR)
• Analog Comparator module with:
- Two analog comparators
- Programmable on-chip voltage reference (VREF) module
- Programmable input multiplexing from device inputs and internal voltage reference
- Comparator outputs are externally accessible
Special Microcontroller Features:
• 100,000 erase/write cycle Enhanced Flash program memory typical
• 1,000,000 erase/write cycle Data EEPROM memory typical
• Data EEPROM Retention > 40 years

• Self-reprogrammable under software control
• In-Circuit Serial Programming™ (ICSP™) via two pins
• Single-supply 5V In-Circuit Serial Programming
• Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
• Programmable code protection
• Power saving Sleep mode
• Selectable oscillator options
• In-Circuit Debug (ICD) via two pins

CMOS Technology:

• Low-power, high-speed Flash/EEPROM technology
• Fully static design
• Wide operating voltage range (2.0V to 5.5V)
• Commercial and Industrial temperature ranges
• Low-power consumption
• PIC16F873A
• PIC16F874A
• PIC16F876A
• PIC16F877A

Liitteisiin sijoitetaan sellainen aineisto, joka tuntuu tarpeelliselta, mutta ei sovi tekstiin sisällytettäväksi.

Devices Included in this Data Sheet:

High-Performance RISC CPU:

• Only 35 single-word instructions to learn
• All single-cycle instructions except for program branches, which are two-cycle
• Operating speed: DC – 20 MHz clock input DC – 200 ns instruction cycle
• Up to 8K x 14 words of Flash Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
• Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

• Timer0: 8-bit timer/counter with 8-bit prescaler
• Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
• Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
• Two Capture, Compare, PWM modules
- Capture is 16-bit, max. resolution is 12.5 ns

- Compare is 16-bit, max. resolution is 200 ns
- PWM max. resolution is 10-bit
• Synchronous Serial Port (SSP) with SPI™ (Master mode) and I2C™ (Master/Slave)
• Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
• Parallel Slave Port (PSP) – 8 bits wide with external RD, WR and CS controls (40/44-pin only)
• Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

• 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
• Brown-out Reset (BOR)
• Analog Comparator module with:
- Two analog comparators
- Programmable on-chip voltage reference (VREF) module
- Programmable input multiplexing from device inputs and internal voltage reference
- Comparator outputs are externally accessible

Special Microcontroller Features:

• 100,000 erase/write cycle Enhanced Flash program memory typical
• 1,000,000 erase/write cycle Data EEPROM memory typical
• Data EEPROM Retention > 40 years
• Self-reprogrammable under software control
• In-Circuit Serial Programming™ (ICSP™) via two pins
• Single-supply 5V In-Circuit Serial Programming
• Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
• Programmable code protection
• Power saving Sleep mode
• Selectable oscillator options
• In-Circuit Debug (ICD) via two pins

CMOS Technology:

• Low-power, high-speed Flash/EEPROM technology
• Fully static design
• Wide operating voltage range (2.0V to 5.5V)
• Commercial and Industrial temperature ranges
• Low-power consumption
• PIC16F873A
• PIC16F874A
• PIC16F876A
• PIC16F877A

LIITE 3.

```
/* Insinöörityö Esa-Pekka Kovalainen & Jari Porkka 4.4.2012
Arduino alustalle tehty kodin viihde-elektroniikan ohjausjärjestelmä.*/

#include <IRremote.h>

#define NUM1 0x20DF8877 // 1 LG kaukosäätimessa
#define NUM2 0x20DF48B7 // 2 LG kaukosäätimessa
```

```
#define NUM3 0x20DFC837 // 3 LG kaukosäätimessa
#define NUM4 0x20DF28D7 // 4 LG kaukosäätimessa
#define ALAS 0x36163AC5  //alas LG kaukosäätimessa
#define YLOS 0x36167A85  //yläs LG kaukosäätimessa
#define VAS 0x36161AE5  //vasen LG kaukosäätimessa
#define OIK 0x3616FA05  //oikea LG kaukosäätimessa

int sensePin = 0;              // fotoresitori kytketty pinni 2
int alasPin = 5;              // moottorin ohjaus alas pinni nro 5
int ylosPin = 6;              // moottorin ohjaus ylös pinni nro 6
int ylarajaPin = 7;            // kankaan ylärajakytkimen pinni nro 7
int alarajaPin = 8;            // kankaan alarajakytkimen pinni nro 8
int ledPin = 10;               // valot kytketty pinni 10
int RECV_PIN = 11;             //Määritetään IR:n pinniksi 11
int kangasPin = 12;             // kankaan nappiohjauksen pinni nro 12
int valoPin = 13;              // valojen nappiohjauksen pinni nro 13
int tila = HIGH;              // jos HIGH kangas ylhäällä jos LOW kangas alhaalla
int ohjaus = LOW;             // ohjaus tila HIGH kun kankaan ohjauksen painike painettu
int serport = 0;
int kirkkaus = 0;
int valoTila = 0;
int valoViive = 0;

IRrecv irrecv(RECV_PIN);           //IR kuuntelut RECV_PIN:Sta
decode_results results;            //dekoodataan IR:n lukema koodi kirjastosta

void config_bt()
{
  Serial.print("BT CONFIG \n");
  Serial.println("SET CONTROL BAUD 115200, 8n1");
}


void setup()
{
  pinMode(ylarajaPin, INPUT);     //ylärajapinnin tyypiksi asetetaan INPUT
  pinMode(alarajaPin, INPUT);     //alarajapinnin tyypiksi asetetaan INPUT
  pinMode(kangasPin, INPUT);       //kaskypinnin tyypiksi asetetaan INPUT
  pinMode(alasPin, OUTPUT);       //Kankaan alasajavan moottorin pinnin tyypiksi asetetaan
OUTPUT
  pinMode(ylosPin, OUTPUT);       //Kankaan ylösajavan moottorin pinnin tyypiksi asetetaan
OUTPUT
  pinMode(ledPin, OUTPUT);       //valojen pinnin tyypiksi OUTPUT
  pinMode(sensePin, INPUT);       //huoneen kirkkauden pinnin tyypiksi INPUT
  pinMode(valoPin, INPUT);       //huoneen valokytkin pinnin tyypiksi INPUT
  analogReference (DEFAULT);       //Määritetään analogen inputin referenssijännite defaultiksi
  irrecv.enableIRIn();          //Otetaan IR:n kuuntelu päälle
  Serial.begin(115200);
}

void loop()
```

```
{
//Nappiohjaus valoille
  while (digitalRead(valoPin) == HIGH && valoTila == LOW) {
        valotpaalle();
    }
  while (digitalRead(valoPin) == HIGH && valoTila == HIGH ) {
//    delay (500);
   valotpois();
  }

//Nappiohjaus kankaalle
   if (digitalRead(kangasPin) == HIGH && tila == HIGH)
     {
      alasajo();
     }
   if (digitalRead(kangasPin) == HIGH && tila == LOW)
     {
      ylosajo();
     }

//bluetooth alkaa
   if( Serial.available() )
    {
     bluetooth();
    }
//infrapuna alkaa
   if (irrecv.decode(&results))
    {
     irohj();
    }
   }
//valkokankaan alasajon aliohjelma
void alasajo()
    {
      while (digitalRead(alarajaPin) == LOW)
      {
        digitalWrite(alasPin, HIGH);
      }
      digitalWrite(alasPin, LOW);
      tila = LOW;
      ohjaus = LOW;
      Serial.println("Alhaalla");
   }
//valkokankaan ylösajon aliohjelma
void ylosajo()
   {
    while (digitalRead(ylarajaPin) == LOW)
      {
        digitalWrite(ylosPin, HIGH);
      }
      digitalWrite(ylosPin, LOW);
```

```
      tila = HIGH;
      ohjaus = LOW;
      Serial.println("Ylhaalla");
   }

 void bluetooth()
     {
       serport = Serial.read();      //luetaan set-muuttujaan tieto sarjaportila (bt)
        if( serport == '1' && tila == HIGH)
          {
         alasajo();
         Serial.flush();
         }
        if( serport == '2' && tila == LOW )
         {
           ylosajo();
           Serial.flush();
         }
        if( serport == '3')
         {
           valotautoPaalle();
           Serial.flush();
         }
        if( serport == '4')
         {
           valotpois();
           Serial.flush();
         }

   }
//IR-ohjauksen aliohjelma
 void irohj()
     {
       if (results.value == NUM1 && tila == HIGH)
       {
          alasajo();
        }
       if (results.value == NUM2 && tila == LOW)
       {
          ylosajo();
        }
       if (results.value == NUM3)
       {
          valotautoPaalle();
        }
       if (results.value == NUM4)
       {
          valotpois();
        }

    irrecv.resume();    //Jatketaan IR:n kuuntelua
```

```
    }
//automaattisten valojen aliohjelma
void valotautoPaalle()
   {
     int val = analogRead(sensePin);          //luetaan tilan valaistus fotoresistorilla
     val = constrain(val, 850, 980);          //raja-arvot tilan valaistukselle
     int kirkkaus = map(val, 850, 980, 255, 0);  //raja-arvot valaistuksen suuruudelle
     analogWrite(ledPin, kirkkaus);
   }

//valojen nappiohjauksen aliohjelma
void valotpaalle() {
       if (kirkkaus < 249) {
       kirkkaus = kirkkaus + 10;
       analogWrite(ledPin, kirkkaus);
       delay (100);
       }
       else if (kirkkaus > 249) {
       delay (2000);
       valoTila = HIGH;
       }
  }
//valojen sammutus aliohjelma
void valotpois() {
          kirkkaus = 0;
          analogWrite(ledPin, kirkkaus);
          valoTila = LOW;
          delay (2000);

}
```

LIITE 4

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello" />
```

```
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.31" >

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_weight="0.22"
            android:orientation="vertical" >


            <Button
                android:id="@+id/button1"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="LIGHT" />

            <ToggleButton
                android:id="@+id/toggleButton2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="ToggleButton" />



            <Button
                android:id="@+id/button2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="LIGHT" />



            <ToggleButton
                android:id="@+id/toggleButton1"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="ToggleButton" />

            <Button
                android:id="@+id/button3"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_weight="0.22"
                android:text="AV" />

Liite
</LinearLayout>

    </LinearLayout>


    <ToggleButton
        android:id="@+id/toggleButton3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```
        android:text="ToggleButton" />


    <Button
        android:id="@+id/button4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Button" />

</LinearLayout>
```