

KARELIA-AMMATTIKORKEAKOULU
Sähkötekniikan koulutusohjelma

Niko Mikkonen

GSM-OHJAUS

Opinnäytetyö
Huhtikuu 2015



OPINNÄYTETYÖ
Huhtikuu 2015
Sähkötekniikan koulutusohjelma

Karjalankatu 3
80200 JOENSUU
p. (013) 260 6800

Tekijä(t)
Niko Mikkonen

Nimeke
GSM-ohjaus

Toimeksiantaja
Karelia-amk

Tiivistelmä

Opinnäytetyön tarkoituksena oli tutkia erilaisia etähallintamahdollisuuksia, keskittyen GSM-ohjaukseen. Aluksi tarkasteltiin kauko-ohjauksen historiaa, josta siirryttiin vanhan sotateollisuuden kautta nykypäivän jokapäiväisiin käyttökohteisiin.

Pääpaino opinnäytetyöllä oli auton polttoainekäyttöisten lisälämmittimien GSM-ohjauksella. Aluksi tarkasteltiin valmistajien virallisia lisäosia, josta siirryttiin tutkimaan erilaisia mahdollisia itse rakennettavaan ohjaimen soveltuvia osia ja laitteita. Haluttujen laitteiden löytämisen jälkeen siirryttiin suunnittelemaan mikropiirin ympärille GSM-puhelimeen perustuvaa etähallintalaitetta.

Laitteen suunnittelun jälkeen tilattiin komponentit ja näistä kasattiin GSM-ohjattava rele, joka soveltuu autojen Eberspächer-lisälämmittimien etäkäynnistykseen tekstiviestillä. Laitteeseen tehtiin ohjelma, joka keskustelee puhelimen sarjaväylän kanssa ja ohjaa täten relettä.

Lopuksi suunniteltiin mahdollista jatkokehitystä ja laitteen laajentamista. Loppusanoissa pohdittiin myös omaa oppimista ja etäohjauksen tulevaisuutta.

Kieli
suomi

Sivuja 37
Liitteet 2
Liitesivumäärä 3

Asiasanat
GSM, kauko-ohjaus, ohjelmointi



THESIS
April 2015
Degree Programme in Electrical Engineering

Karjalankatu 3
FI 80200 JOENSUU
FINLAND
Tel. 358-13-260-6800

Author(s)

Niko Mikkonen

Title
GSM-control

Commissioned by
Karelia UAS

Abstract

The purpose of this thesis was to study different ways of remote controlling with the focus on GSM-control. At first the history of remote controlling was studied, followed by a look back to old military technology and finally coming to everyday appliances.

The main focus of the thesis was the GSM-controlling of additional fuel operated car heater. At first the official manufacturer's add-ons were studied and after that a look for suitable parts and components for a DIY controller. After the proper parts were found, a microchip based controller with a GSM phone was designed.

After the planning of the device, the parts were ordered and with these a GSM-controlled relay was built, which is suitable for controlling in-car Eberspächer heaters with SMS messages. A program which can communicate with a phone through serial port and thus controls a relay was designed.

In the end, a possible expansion of the device was planned. Finally, own learning and the future of remote controlling were discussed.

Language
Finnish

Pages 37
Appendices 2
Pages of Appendices 3

Keywords

GSM, remote control, programming

Sisältö

1	Johdanto.....	6
2	Etäohjauksesta	6
2.1	Historiaa.....	6
2.2	Nykyiset käyttökohteet.....	7
3	Valmistajan viralliset lisäosat	8
3.1	Eberspächer.....	8
3.2	Webasto.....	11
4	Laitteiden valinta.....	12
4.1	GSM-laitteet.....	12
4.1.1	Nokia 3210.....	12
4.1.2	Nokia 3310.....	13
4.1.3	Samsung Galaxy Spica.....	13
4.1.4	Shield.....	13
4.2	Ohjauslaite.....	14
4.2.1	Arduino	14
4.2.2	ATmega	15
4.2.3	PC.....	15
4.2.4	Raspberry Pi	16
5	Valitut laitteet.....	16
5.1	3310:n edut ja haitat	16
5.2	Arduinon/ATmegan edut ja haitat.....	17
5.3	Valmiit komponentit.....	17
5.4	Komponenttien ja laitteiden käyttölämpötila	18
6	Piirikaavio	18
6.1	Piirilevy	18
6.2	Kaavio.....	19
7	Komponenttien hankinta.....	22
7.1	Hinta	22
7.2	Tilaaminen ja takuut.....	23
8	Ohjelmointi ja prototyyppi	24
8.1	Käytettävät ohjelmat	24
8.1.1	Arduino IDE.....	24
8.1.2	AVRDUDE	24
8.2	Puhelimen muokkaus.....	25

8.3	Arduino prototyyppi	26
8.4	FBUS-protokolla.....	27
8.5	Koodin tekeminen	28
9	Laitteen rakentaminen	30
9.1	Arduinon ja piirin erot	30
9.2	Ohjelmointikanta	30
9.3	Testaus ja kasaus	31
9.4	Valmis laite.....	32
10	Asennus autoon.....	34
10.1	Asennuspaikka.....	34
10.2	Liittäminen lämmityslaitteeseen	34
10.2.1	Moduulikellokytkin	34
10.2.2	Easystart T.....	35
10.2.3	Minikellokytkin	36
11	Pohdinta	37
	Lähteet.....	38

Liitteet

Liite 1	FBUS testikoodi
Liite 2	ATmega8 lähdekoodi

1 Johdanto

Autooni on asennettu Eberspächer-merkkinen polttoainekäyttöinen lisälämmitin erityisen kylmien talvien varalle. Tarve etäkäynnistykseen tuli, kun totesin auton lisälämmittimen ajastimen olevan riittämätön omiin käyttötarkoituksiin. Jos jonain päivänä oli kylmää ja lisälämmitintä tarvitsi käyttää, niin täytyi tietää tarkka kelonaika, milloin autoa käytettiin. Joskus aikataulut luistivat ja auto lämpeni tarpeettoman pitkään, kuluttaen näin turhan paljon akkua ja polttoainetta.

Olin jo aiemmin ollut kiinnostunut elektroniikasta ja GSM-ohjauksesta muissa laitteissa, joten oli luonnollista kokeilla rakentaa ohjainta itse. Vaihtoehtoisesti olisi voinut hankkia Eberspächerin alkuperäisen GSM-moduulin, mutta tämä oli aivan liian kallis.

Tässä opinnäytetyössä olen tutustunut erilaisiin etäohjausmahdollisuuksiin ja vertailut näitten hyviä ja huonoja puolia. Pääpaino on GSM-ohjaimilla, mutta myös muunlaisella tekniikalla toteutettuja ratkaisuja on tarkasteltu. Opinnäytetyö alkaa yleisellä tietoudella etäohjauksesta ja jatkuu oman ohjaimen valmistamisella vaiheittain, jota painotetaan enemmän. Pysin myös rakentamaan laitteen mahdollisimman edullisesti.

2 Etäohjauksesta

2.1 Historiaa

Tarve elektroniikan käytölle muualtakin kuin vain suoraan laitteesta käsin on ollut olemassa varmasti yhtä pitkään kuin itse elektroniikkakin. Ensimmäisiä kokeiluja etäohjauksen saralla onkin Nikola Teslan, ehkä sähkötekniikan tärkeimmän edistäjän vuonna 1898 pieneen veneeseen kehittämä radiokauko-ohjain (Sarkar 2006, 276). Useat henkilöt kehittivät radio-ohjausta 1900-luvun alkupuolella ja

tämä herättikin nopeasti sotateollisuuden mielenkiinnon. Ensimmäisessä maailmansodassa Saksa käytti räjähteillä lastattuja kauko-ohjattavia veneitä vihollisen laivaliikennettä vastaan (Karau 2003, 91). Toisessa maailmansodassa Neuvostoliitto käytti radiokauko-ohjattuja panssarivaunuja Suomen joukkoja vastaan. Nämä saattoivat olla aseistettuja tai räjähteillä lastattuja, kantamaa näissä oli n. 500–1500 m (Lichagin 2004).

Ehkä yleisin kauko-ohjauksen muoto on kuitenkin television kaukosäädin, jonka keksi Eugene Polley, Zenith Electronics -yhtiössä työskennellyt insinööri vuonna 1955. Polleyn keksintö perustui näkyvään valoon, jota näytettiin television reunoissa oleviin valoherkkiin kennoihin (The Telegraph, 2012). Näkyvä valo aiheutti kuitenkin ongelmia auringon kanssa ja Polley kehitteli vuonna 1956 parannetun version, Zenith Space Commandin, joka perustui ultraääneen (Farhi, 2007). Nykyäänkin käytetty infrapuna syrjäytti ultraäänikaukosäätimet 80-luvun alkupuolella.

2.2 Nykyiset käyttökohteet

Nykyään etäohjausta on nähtävissä joka puolella, eikä siihen välttämättä kiinnitetä enää edes huomiota. Jokaisen television mukana tulee kaukosäädin, eikä TV:n kaikkia ominaisuuksia välttämättä voikaan käyttää ilman sitä. Lähietäisyydeltä voidaan myös ohjata esimerkiksi huoneen valaistusta tai kytkeä päälle ja pois melkein mitä tahansa sähkölaitetta pistorasiaan liitettävien ohjaimien avulla.

GSM-ohjaus puolestaan on yleistynyt kohteissa, joissa viiveellä ei ole väliä, kuten lämmitysten kytkennässä tai lämpötilatiedoissa. Laitteet voivat toimia puheluiden tai tekstiviestien välityksellä. Puhelinohjauksessa saattaa olla esimerkiksi ohjaimen muistiin tallennettu ääniviesti, joka antaa laitteeseen soittaessa mahdollisia komentoja, jotka syötetään numeroita painamalla. Tekstiviestiohjauksessa taas laitteeseen voidaan lähettää viestinä komento, jonka laite toteuttaa ja mahdollisesti lähettää vastausviestin. Kodin valvontalaitteet ovat myös yksi hyvä esi-

merkki. Laitteen havaitessa liikettä voidaan käyttäjälle lähettää tekstiviestivaroitus. Jotkin järjestelmät mahdollistavat myös puhelun vastaanottamisen ja tätä kautta kohteen kuuntelemisen puhelimen välityksellä.

Laajat internetyhteydet ovat mahdollistaneet suurempaa, viiveettömämpää ohjausta. Esimerkki tällaisesta ovat talon valvontalaitteet, kuten web-kamerat, joita voi asentaa esimerkiksi kesämökille tai taloon. Tällaiset laitteet tuovat turvallisuuden tunnetta, koska melkein mistä tahansa voi tarkastaa tilanteen kohteessa.

3 Valmistajan viralliset lisäosat

3.1 Eberspächer

Polttoainekäyttöiset lisälämmittimet toimivat normaalisti omalla yksinkertaisella kellokytkimellään, joiden toimintaa voidaan yleensä myös hallita etänä valmistajalta erikseen hankittavilla lisäosilla. Eberspächer tarjoaa kahdenlaista etämoduulia: Easystart R+ (kuva 1) ja Easystart Call (kuva 2).

Easystart R+ on radiotaajuudella toimiva kaukosäädin, johon on integroituna normaalit kellokytkimen toiminnot, eli sillä pystyy ohjelmoimaan ajastuksia tai käynnistämään lämmityksen suoraan. Koska R+ toimii radiotaajuudella, on sen kantama rajoittunut, ideaaliolosuhteissa esteettömässä maastossa noin 1 km. Kantamatkaa rajoittavat erityisesti kaupunkiolosuhteissa rakennukset yms. esteet. Laitteen hinta on 299 € (Eeperi, 2015).



Kuva 1. Eberspächer Easystart R+. (Auto-Standheizung 2015).

Easystart Call puolestaan on Eberspächerin valmistama virallinen GSM-lisäosa. Sitä on mahdollista ohjata tekstiviesteillä tai soittamalla laitteeseen liitettyyn normaaliin GSM-liittymään. Callia ei luonnollisestikaan koske R+:n kantomatkarajoitukset, vaan se toimii kaikkialla, jossa vain on matkapuhelinverkko. Vastaanoton parantamiseksi laitteeseen sisältyy myös ulkoinen antenni. Hintaa ohjaimelle kertyy 345 € (Autoextra, 2015).



Kuva 2. Easystart Call. (Kfz-braun_info 2015).

Easystart Callin käytön helpottamiseksi on myös saatavilla Android- ja iOS-sovellukset (kuva 3). Kumpikin käyttää kuitenkin ohjaukseen tekstiviestejä, joten kustannukset ovat samat kuin normaalissakin SMS-ohjauksessa.



Kuva 3. Easystart Call Android-käyttöliittymä. (Google Play 2015).

3.2 Webasto

Myös Eberspächerin kilpailijalla, Webastolla on oma tuotesarjansa lämmityslaitteensa etäohjaukseen. Ohjaimet ovat vastaavanlaisia kuin Eberin, sisältäen kaksi radiokauko-ohjainta ja GSM-vastaanottimen.

Telestart T91 radio-ohjain on yksinkertainen painokytin, jolla lämmitys saadaan käynnistettyä ja sammutettua. Telestart T100 HTM -sarjan ohjaimessa on lisäksi näyttö, joka mahdollistaa myös ajastusten hallinnan etänä. Molempien ohjainten

käyttösäde on esteettömässä ympäristössä n. 1 km, sama kuin Eberspächerin vastaavissa tuotteissa (Webasto, 2012).

4 Laitteiden valinta

4.1 GSM-laitteet

Oikeanlaisen GSM-laitteen valinta oli tärkeässä asemassa ohjaimen suunnittelun kannalta. Valintaan vaikutti erityisesti helppo yhdistettävyyys ohjauslaitteeseen ja myös koko projektin teemana ollut edullisuus.

GSM-laitteeksi harkitsin kahta erilaista vaihtoehtoa, puhelinta ja ohjaukseen tarkoitettua piiriä. Kumpaankin ratkaisuun liittyi luonnollisestikin hyviä ja huonoja puolia.

4.1.1 Nokia 3210

Olin jo aiemmin testannut vanhaa Nokia 3210 -matkapuhelimen ohjausta Arduinolla, mutta ainakaan tämä kyseinen yksilö ei suostunut yhteistyöhön, vaan antoi vaan satunnaisia arvoja sarjaliitäntänsä kautta oikeista synkronointikomennoista huolimatta.

Kyseisen mallin komentoja ei myöskään ole kovin hyvin dokumentoitu ja näiden arvaaminen on käytännössä lähes mahdotonta. Tämä 3210 oli myös muuten epäluotettava ja saattoi sammua yllättäen.

4.1.2 Nokia 3310

Nokia 3310 on valmistajan ehkä maailmanlaajuisesti tunnetuin matkapuhelinmalli, joka on kestävyydellään saanut jo osittain legendan omaisen maineen. Puhelin on hyvin yksinkertainen ja sen FBUS-protokolla on myös kohtuullisen laajasti dokumentoitu. Puhelin ei tue AT-komentoja, koska se ei sisällä varsinaista modeemia. Nämä olisivat osaltaan helpottaneet ohjelmointia, mutta myös FBUS:n kanssa tulee toimeen.

Nokia 3310 on vuonna 2000 julkaistu 3210:n korvaaja. Sitä on myyty maailmanlaajuisesti 126 miljoonaa kappaletta ja se onkin yksi maailman suosituimpia puhelinmalleja. Siitä on julkaistu myös useita ominaisuuksiltaan vaihtelevia versioita, joihin kuuluvat mm. 3330 ja 3350, joihin lisättiin WAP-toiminnot. (Stinson, 2015)

4.1.3 Samsung Galaxy Spica

Harkitsin myös vanhaa Android-pohjaista puhelinta, Samsung Galaxy Spicaa. Tämä malli edelsi nykyistä Galaxy S -sarjaa. Ohjelmointi kyseiselle alustalle ei ole tuttua. Vaikka tämä ei välttämättä olisikaan itsessään ongelma, on kyseinen älypuhelin kokemuksiin perustuen vakaudeltaan paljon huonompi kuin perinteisen mallinen puhelin.

Spican käyttämiseen olisi tarvinnut esim. Bluetooth-adapterin, jonka avulla olisi luotu langaton yhteys käytettävän ohjaimen ja puhelimen välillä. Langaton yhteys tällaisessa käytössä olisi tuonut varmasti ylimääräistä epäluotettavuutta, eikä tämänkään takia olisi ollut hyvä vaihtoehto.

4.1.4 Shield

Arduino-laitteille on tehty useita kohtuullisen edullisia shieldeiksi kutsuttuja moduuleja, eli suoraan Arduinon päälle asennettavia lisäosia. On olemassa myös

GSM-shieldejä (kuva 4), jotka ovat helppoja asentaa ja saada toimimaan, kaikki moduulien toiminnot on myös huolellisesti dokumentoitu. Vaikka hinta ei olekaan järin korkealla, on se silti suurempi hankinta, kuin jokin vanha puhelin, varsinkin jos tällainen sattuu jo valmiiksi löytymään.



Kuva 4. Arduino GSM-shield. (Arduino 2015b)

Shieldin sovittaminen lopulliseen itsenäiseen, ei Arduinoon perustuvaan laitteeseen olisi ollut kömpelöä, koska se on, kuten aiemmin on mainittu, suunniteltu asennettavaksi juotoksettomasti suoraan Arduinon päälle. Tämä olisi tuonut laitteeseen myös ylimääräistä kokoa ja shieldin juotoksia olisi saattanut joutua purkamaan.

4.2 Ohjauslaite

4.2.1 Arduino

Arduino on viime vuosina kasvattanut suosiotaan niin elektroniikan harrastajien, kuin ammattilaistenkin keskuudessa. Se on yksinkertainen ja silti monipuolinen ohjelmointialusta. Arduinosta on olemassa monia eri versiota, 1,85 cm x 4,3 cm, 14-liitäntäisestä Nanosta, 5,3 cm x 10,2 cm, 52-liitäntäiseen Megaan, mutta lähes kaikki Arduinot perustuvat Atmelin Atmega-sarjan mikrokontrollereihin (Arduino 2015d).

Arduino onkin yksinkertaisimmillaan juotokseton Atmega-liitäntäalusta, joka tarjoaa valmiit jänniteregulaattorit, USB-liitäntän ja helpon liitettävyyden valmiiden johtoliittimien ansiosta. Ohjelmointiin käytetään Arduino-kieltä, joka on sarja C- ja C++-funktiota. Arduinoa voi myös ohjelmoida suoraan C- ja C++-kielillä (Arduino 2015a).

4.2.2 ATmega

ATmega, jonka ympärille myös Arduino on rakennettu, on Atmelin AVR-sarjaan perustuva, vuodesta 1996 asti kehitetty 8-bittinen mikrokontrolleri (Robot Technology, 2015). Niitä on saatavana montaa eri mallia suuresti vaihtelevilla liittimäärillä ja suorittimen nopeuksilla, sekä useilla muistipiirivaihtoehdoilla.

Atmel tarjoaa AVR-sarjassaan myös monia muita mikropiirejä. ATtiny on pieneen tilaan sopiva, 4–28-pinninen piiri. Koostaan huolimatta sillä on mahdollista suorittaa monimutkaisiakin toimia, joita rajoittaa ainoastaan valittu pinnimäärä ja 0,5–16 kt:n muistikoko (Atmel 2015b). Nykyään Atmel tarjoaa myös monipuolisiin sovelluksiin sopivia 32-bittisiä mikropiirejä.

4.2.3 PC

Ohjaus olisi myös mahdollista toteuttaa lähes millä tahansa PC-tietokoneella sarjaliitännän kautta. Esimerkiksi ITX-standardin mukaisia emolevyjä on saatavilla montaa eri kokoa, perusmalli Mini-ITX on 17 cm x 17 cm ja pienin Mobile-ITX 6 cm x 6 cm (VIA 2015).

PC-pohjaisuus tuo mukanaan suuren muokattavuuden, mutta tietokoneet ovat samalla myös turhan monipuolisia tällaiseen käyttökohteeseen. Monipuolisuus tuo mukanaan myös korkeamman virran tarpeen, joka ei autokäytössä ole välttämättä hyvä asia, varsinkin jos auto on käyttämättä pitemmän aikaa. Laitteiden hinta on myös turhan korkea verrattuna muihin vaihtoehtoihin.

4.2.4 Raspberry Pi

Raspberry Pi eli RasPi on ARM-prosessoriteknologiaan perustuva pieni, noin maksukortin kokoinen tietokone. Se on tullut nopeasti todella suosituksi vaihtoehdoksi lähes kaikkiin ”älyä” vaativiin harrastajaprojekteihin ja onkin siksi herättänyt monien kilpailijoiden huomion. RasPin edullisuus mahdollistaa sen käytön pysyvääkin sijoittamista vaativissa kohteissa (Raspberry Pi 2015).

Kuten PC, myös RasPi olisi ollut tähän käyttökohteeseen turhankin monipuolinen ja sen ominaisuudet olisivat menneet suurelta osin hukkaan. Vaikka Raspberry Pi onkin varsin edullinen, ei se pärjää silti hinnassa Arduinolle/Atmegalle.

5 Valitut laitteet

5.1 3310:n edut ja haitat

Nokia 3310:n maine tulee lähinnä sen hyvästä kestävyydestä, joka autokäyttöä ajatellen on hyvä asia. Puhelimessa on myös hyvä akunkesto, tällä ei tosin ole väliä jos akun korvaa jänniteregulaattorilla. 3310 on nykyään aika edullinen, niitä saa jopa ilmaiseksi ja saatavuus on kohtuullisen hyvä puhelimen suosion takia.

Haittapuolena on lähinnä puhelimen ikä, joka saattaa aiheuttaa ongelmia mm. muuten hyvän kestävyuden kanssa. Puhelimen FBUS-liitäntään käyvää kaapelia on myös vaikea löytää, joten se täytyy tehdä joko itse tai juottaa johdot suoraan emolevylle, kuten tässä tapauksessa tein.

5.2 Arduinon/ATmegan edut ja haitat

Atmega-piirit ovat äärimmäisen suosittuja harrastelijoiden keskuudessa ja tämän takia melkein kaikkeen mahdolliseen löytyy ohjeita, vinkkejä ja ohjelmointiratkaisuja. Koodia on todella helppo kirjoittaa ja testata, koska sen kääntäminen on nopeaa ja sen saa lähetettyä heti Arduinoon.

ATmegan valitsin sen Arduino yhteensopivuuden takia. Tarkemmaksi malliksi valikoitui ATmega8, Atmelin hieman vanhempi malli, jota on myös käytetty varhaisissa Arduinoissa. ATmega8 oli myös laajasti saatavilla ja sen hinta oli Kiinasta hankittaessa alhainen.

5.3 Valmiit komponentit

Jotkut ohjaimen liittyvistä osista on edullisempaa ja helpompaa ostaa valmiiksi kasattuina. Tällaisia olivat tässä projektissa jännitteensäädin ja ohjaimen virtapiiristä erotettu rele. Kumpikaan osista ei ole erityisen monimutkainen, mutta valmiin hankkiminen oli ainakin tässä tapauksessa edullisempaa kuin itse kasaaminen. Rele on mikropiirin virtapiiristä valmiiksi optoerottimella eristetty, ettei releen aukaisu tai sulkeminen aiheuta mikropiirin toiminnan häiriintymistä, jota oli havaittavissa aiemmin testaamissani laitteissa relettä suoraan ohjattaessa. Rele vie valmistajan mukaan päällä ollessaan 15–20 mA:n jatkuvan virran, Arduino suosittelee jatkuvaksi syöttövirraksi 20 mA pinniä kohden, tämän takia onkin hyvä ottaa piiriltä kuormaa pois käyttämällä kyseistä menetelmää (Arduino 2015c).

Varteenotettavia lisälaitteita olisi ollut myös kellopiiri ja nestekidenäyttö. Näitä kumpaakaan tosin ei välttämättä tarvinnut ja päätinkin siksi jättää ne pois.

5.4 Komponenttien ja laitteiden käyttölämpötila

Kaikilla komponenteilla on valmistajan suosittelema käyttölämpötila, näistä turvarajoista poikkeaminen saattaa johtaa mm. lyhempään käyttöikään ja toimintahäiriöihin. Esim. vanhemmat LCD-näytöt, kuten valitussa 3310-puhelimessa, hidastuvat pakkasolosuhteissa huomattavasti, mutta ainakin omien kokemusten mukaan kestävät kylmän sään rikkoontumatta. Käytettävän ATmega8-piirin ilmoitettu lämpötila-alue on $-40\text{ °C} - 80\text{ °C}$, joka sopii Suomen vaihteleviin sääolosuhteisiin hyvin. Vaikka eri valmistajien suositukset vaihtelevatkin, ovat ne yleensä hyvin samankaltaisia, esimerkkinä Alteran suositukset (Altera 2015):

- Kuluttajataso: $0\text{ °C} - 85\text{ °C}$
- Teollisuus: $-40\text{ °C} - 100\text{ °C}$
- Autoteollisuus: $-40\text{ °C} - 125\text{ °C}$
- Laajennettu: $-40\text{ °C} - 125\text{ °C}$
- Sotilaskäyttöön tarkoitettu: $-55\text{ °C} - 125\text{ °C}$

Kaikkien hankittujen komponenttien käyttölämpötiloja ei ollut ilmoitettu, tällöin oletetaan niiden olevan alinta, eli kuluttajatasoa. Tämän takia niiden käyttäytymistä varsinkin kylmissä olosuhteissa on vaikea arvioida, mutta koska kyseessä on omaan käyttöön tuleva laite, en kokenut tätä ongelmaksi.

6 Piirikaavio

6.1 Piirilevy

Piirilevynä päädyin käyttämään valmista reikälevyä, jota käytetään yleensä pieniin projekteihin. Liitokset pisteiden välillä tehdään yleensä käyttäen hyppylankoja, itse käytin kuitenkin suurimmaksi osaksi tinasiltoja, käyttäen johtoja joissain kohdissa. Reikälevylle rakennettaessa joutuu laitteen suunnittelemaan yksipuoleiseksi ja tämä lisää kokoa hieman. Kaksipuoleiselle piirilevylle tehtäessä laitteesta saisi mahdollisesti pienemmän, koska johtimille käytössä oleva pinta-ala

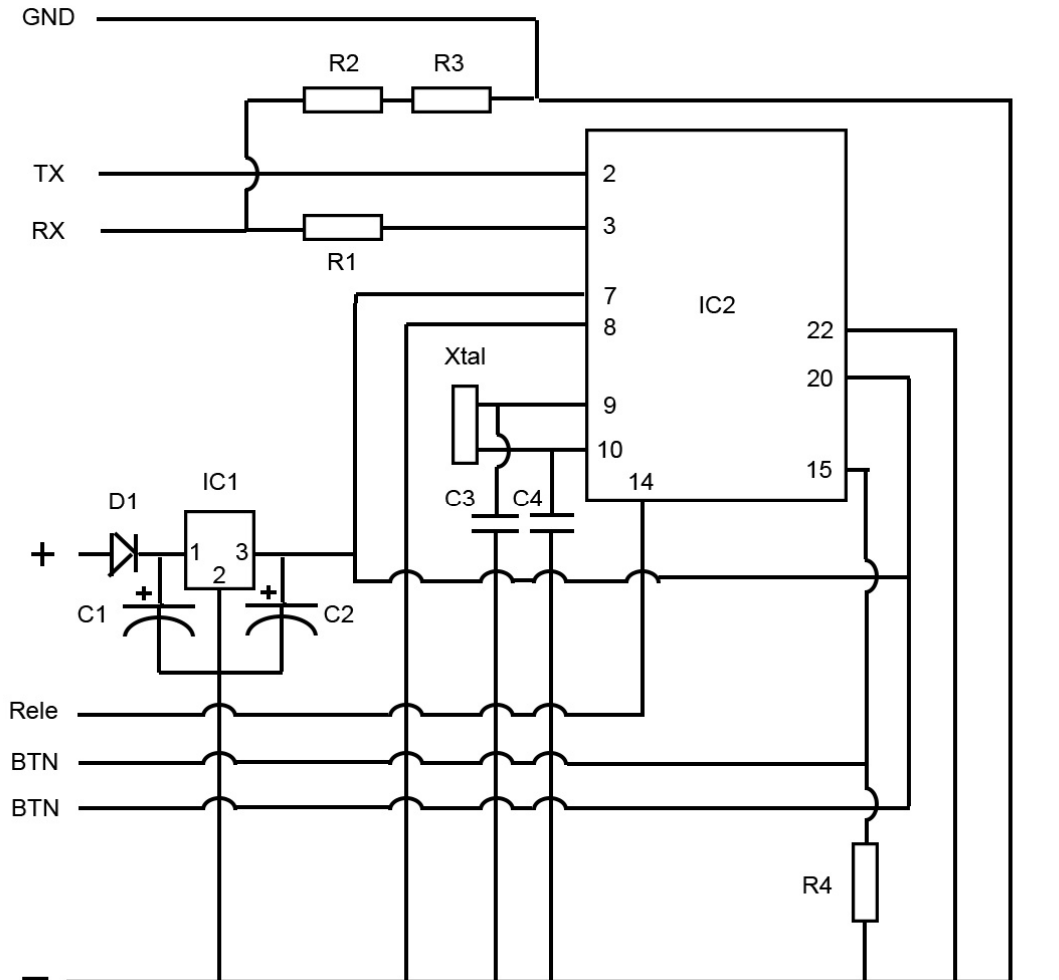
kaksinkertaistuisi. Tämän valmistaminen olisi tosin turhan monimutkaista, koska levyn joutuisi tekemään esim. syövyttämällä, johon tarvitsisi työhön sopivia tarvikkeita.

Elektroniikan peruskomponentit ovat yleensä suoraan sopivia levyn reikäjakoon (0,1" eli 2,54 mm), mikä helpotti juottamista. Vaihtoehtoina oli erillisillä ja jonoittain toisiinsa liitetyillä rei'illä varustetut mallit, näistä päädyin ensimmäiseen helpomman suunnittelun takia, vaikka tinasiltoja joutuukin tekemään enemmän.

6.2 Kaavio

Aloitin valitsemalla tarvittavat komponentit ja piirtämällä näitten pohjalta piirikaavion. Kaavion tein käsin piirto-ohjelmalla ja se on nähtävänä kuvassa 5. Komponentit ovat seuraavat:

- D1 = 1N4004
- C1, C2 = 10 μ F 50 V
- C3, C4 = 22 pF 50 V
- R1, R2, R3 = 220 Ω
- R4 = 10 k Ω
- IC1 = L7805
- IC2 = ATmega8
- Xtal = 16 MHz



Kuva 5. Piirikaavio.

D1 suojaa laitetta rajoittamalla virran suuntaa. IC1 on 5 V:n jänniteregulaattori, syöttöjännitteeksi käy n. 7–30 V, tarkkaa yläarvoa ilman lisjäähdytystä on hie- man vaikea määrittää, mutta lämpeneminen ei ole ongelma ainakaan auton 12 V:n sähköjärjestelmää käytettäessä. Xtal on 16 MHz:n kide, jonka avulla AT- mega8 tahdistuu kyseiseen kellotaajuuteen. C3 ja C4 ovat ATmega8:n tietoleh- dessä annetut kondensaattorin suositusarvot kyseistä kidettä käytettäessä (At- mel 2015a). Pinniin nro 14 liitetään 5 V:n rele. BTN-paikkoihin liitetään painonappi manuaalista sammutusta varten. Vastus R4 huolehtii, ettei sisääntulopinni 15 jää ”kellumaan”, jolloin sen tila vaihtelisi satunnaisesti.

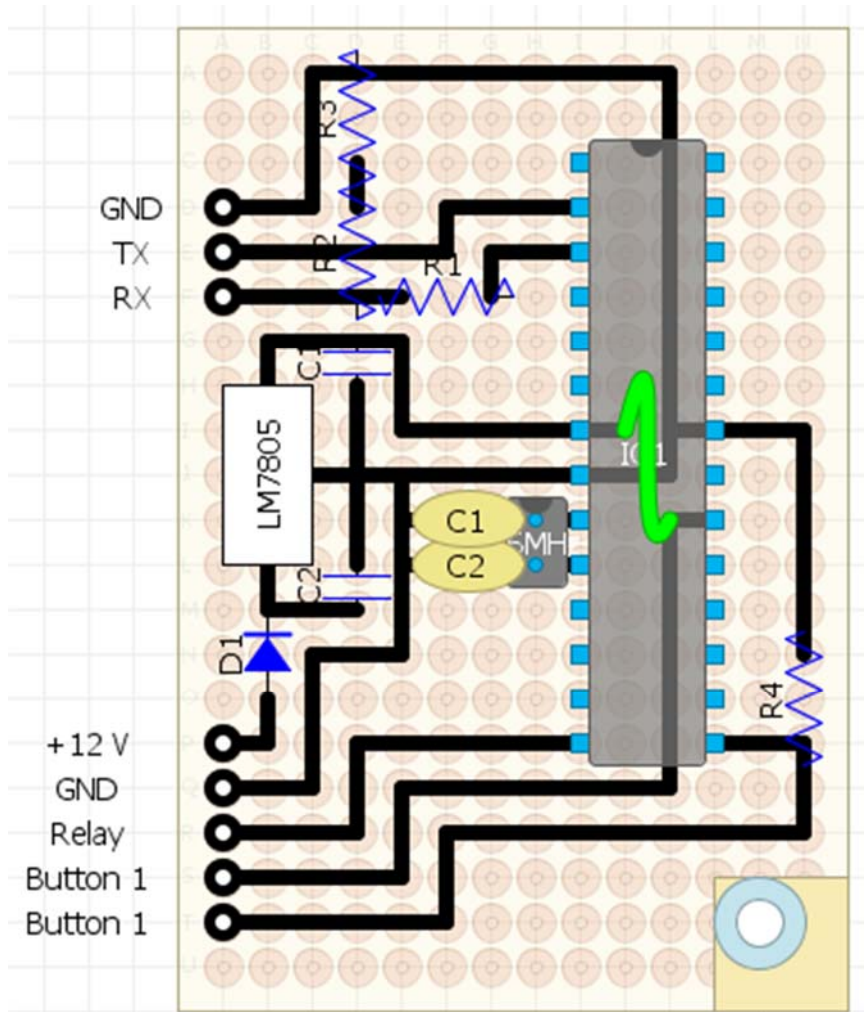
GND-, TX- ja RX-paikkoihin liitetään puhelimen FBUS-väylä. RX-pinniin tulevaa jännitettä on rajoitettu jännitteenjakopiirillä. Jakaja hyödyntää jännitteenjako- sääntöä:

$$U_1 = U (R_1 / (R_1 + R_2)) \quad (1)$$

missä U_1 = lähtevä jännite (V)
 U = tuleva jännite (V)
 R_1 = ensimmäinen vastus (Ω)
 R_2 = toinen vastus (Ω)

Käytin kolmea 220 Ω :n vastusta R_1 , R_2 ja R_3 , eli kaavaan soveltaen $U = 5$ V, $R_1 = 2 * 220 \Omega$ ja $R_2 = 220 \Omega$. Tämä pudottaa jännitteen sopivalle 3,3 V:n tasolle. ATmegaan sisään tulevaa jännitettä ei tarvitse muuttaa.

Seuraavaksi tein mallin laitteesta juottamista varten (kuva 6). Piirtämiseen käytin DIYLC nimistä ohjelmaa, jonka avulla voi nopeasti piirtää piiri- ja reikälevymalleja. Malli auttaa huomattavasti komponenttien sijoittelussa ja tinasiltojen tekemisessä.



Kuva 6. GSM-ohjaimen DIYLC-malli.

7 Komponenttien hankinta

7.1 Hinta

Projektiin joutui hankkimaan useita komponentteja ja laitteita. Osat ovat Suomessa aika kalliita mm. arvonlisäverojen ja useiden välikäsiensä takia. Osat oli siis edullisinta hankkia Kiinasta, eBay-nettihuutokaupan avulla. Komponentit ovat lähestulkoon saman laatuista kuin Suomesta hankitut, valmiissa laitteissa saattaa kuitenkin olla esim. heikkoja juotoksia. Hinnaltaan komponentit ovat lähes poikkeuksetta ainakin puolet Suomen vastaavan verrattuna. Komponenttien hintoja:

- 5 kpl L7805, 1,01 €
- 10 kpl 16 MHz oskillaattori, 0,94 €
- 10 kpl 10 uF kondensaattori, 0,94 €
- 50 kpl 22 pF kondensaattori, 1,51 €
- 10 kpl DIP-28 kantoja, 1,49 €
- 2 kpl reikälevy, 2,82 €
- 1 kpl 5 V rele, 1,29 €
- 3 kpl ATmega8, 4,20 €

Laitteen hinnaksi komponenttien osalta tulee siis vähän yli 6 €. Hintaa kuitenkin nostavat vielä lisämateriaalit, kuten kotelo ja liittimet, mutta silti hinnaksi jää alle 15 €.

7.2 Tilaaminen ja takuut

Ebaysta tilaaminen hoituu nopeasti PayPalin avulla. Tuotteiden toimitusajat vaihtelevat paljon myyjän, kuljetusyhtiön ja esim. Kiinan juhlapyhien mukaan. Nopeimmillaan tavarat saapuvat noin viikossa, pisimmillään saattaa kestää jopa 2 kuukautta. Projektin komponenteilla ja laitteilla ei ollut hirvittävän kiire, joten tämä ei lopulta muodostunut kovinkaan suureksi ongelmaksi.

Kiinan tuotteilla ei yleensä ole takuuta ja jos onkin, se ei toimi ilmaiseksi, vaan asiakkaan täytyy maksaa postikulut edestakaisin Kiinan ja Suomen välillä. Toimitusaika saattaa näissä tapauksissa olla hinnan ohella ongelmana ja siksi takuumahdollisuuksia käytetäänkin harvoin tuotteiden edullisuuden takia. GSM-ohjainprojektissa tavara kuitenkin toimi pääosin hyvin ja komponentit olivat suhteellisen edullisia, vaikka niitä olisikin joutunut tilaamaan lisää. Ainoastaan aiemmin mainittu valmiiksi kasattu jännitteensäädin ei toiminut, joten korvasin tämän L7805-regulaattorilla.

8 Ohjelmointi ja prototyyppi

8.1 Käytettävät ohjelmat

Mikrokontrollerin ohjelmointiin voidaan käyttää lähes mitä tahansa ohjelmointityökalua. Koska Arduino on paljon käytetty mikrokontrolleri, on sitä varten saatavilla lisäosa moneen ohjelmointiympäristöön.

ATmegaa varten on puolestaan useita sitä varten suunniteltuja työkaluja. Tällainen on esimerkiksi oikeiden sulakeasetuksien selvittämiseksi tehty sulakelaskuri.

8.1.1 Arduino IDE

Arduino IDE on Arduino-mikrokontrollerin ohjelmointiin suunniteltu yksinkertainen ohjelmointityökalu. Siihen on saatavilla useita kirjastoja, joilla voidaan helposti laajentaa Arduinon mahdollisia toimintoja. Ohjelma käyttää C++ pohjaista kieltä Atmelin AVR Studion tapaan. IDE osaa kääntää ohjelman Arduinolle sopivaan muotoon ja ladata sen suoraan kontrolleriin USB-liitännän kautta. Tämän mahdollistaa Arduinon sisältämä bootloader-ohjelmisto, jonka ansiosta erillistä ohjelmointikaapelia ei tarvita.

Aiemmin mainittu AVR Studio on Atmelin oma AVR-sarjan piireille tarkoitettu ohjelmointiympäristö. Se on monipuolisempi kuin Arduino IDE, mutta näin pieneen projektiin ominaisuuksilleen turhankin laaja.

8.1.2 AVRDUDE

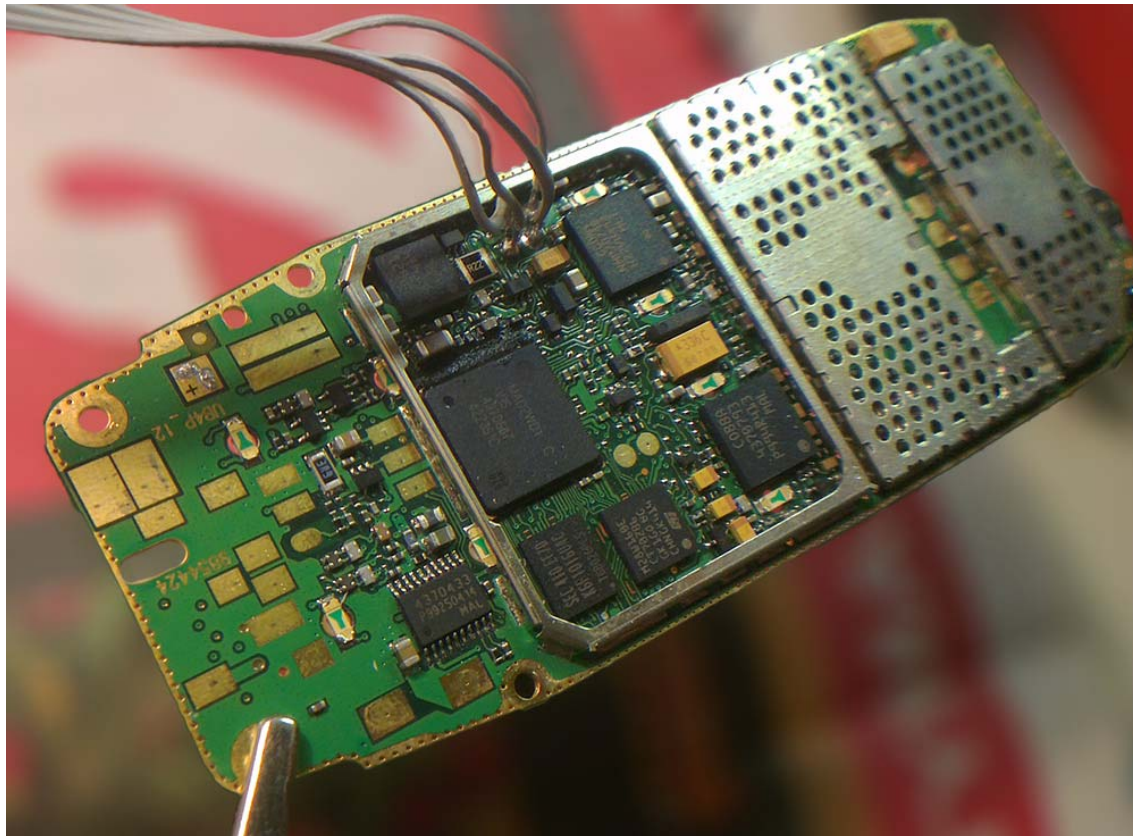
AVRDUDE on ATmega-piirien kanssa käytettävä komentorivipohjainen hallintaohjelma. Se mahdollistaa mm. valmiiksi käännettyjen ohjelmien lataamisen ja ehkä tärkeimpänä ominaisuutena sulakeasetusten määrittämisen, joita tarvitsee

muuttaa esim. kellotaajuuksia vaihdettaessa. Ohjelmaa käytetään usein tässäkin projektissa käytetyn USBASP-ohjelmointilaitteen kanssa.

AVRDUDEn on tehnyt Brian S. Dean henkilökohtaisena projektinaan AVR-sarjan mikropiirien ohjelmointiin. Kasvaneen kiinnostuksen takia hän julkaisi AVRDUDEn vapaaseen levitykseen (Savannah 2015). Ohjelma onkin suosittu AVR-ohjelmoijien keskuudessa.

8.2 Puhelimen muokkaus

Puhelinta piti muokata projektiin sopivaksi, eli siihen täytyi juottaa johdot sisäiseen FBUS-liittimeen (kuva 7). Tämän olisi voinut välttää hankkimalla alkuperäisen liitinjohdon, mutta nämä alkavat olla aika harvinaisia. Puhelin oli helppo purkaa ja johdot sai asennettua liittimiin. Kylkeen tehty lovi mahdollisti johtojen vetämisen ulos puhelimen kyljestä, akun alta (kuva 8).



Kuva 7. Puhelimen emolevy, johdot juotettuna paikalleen.

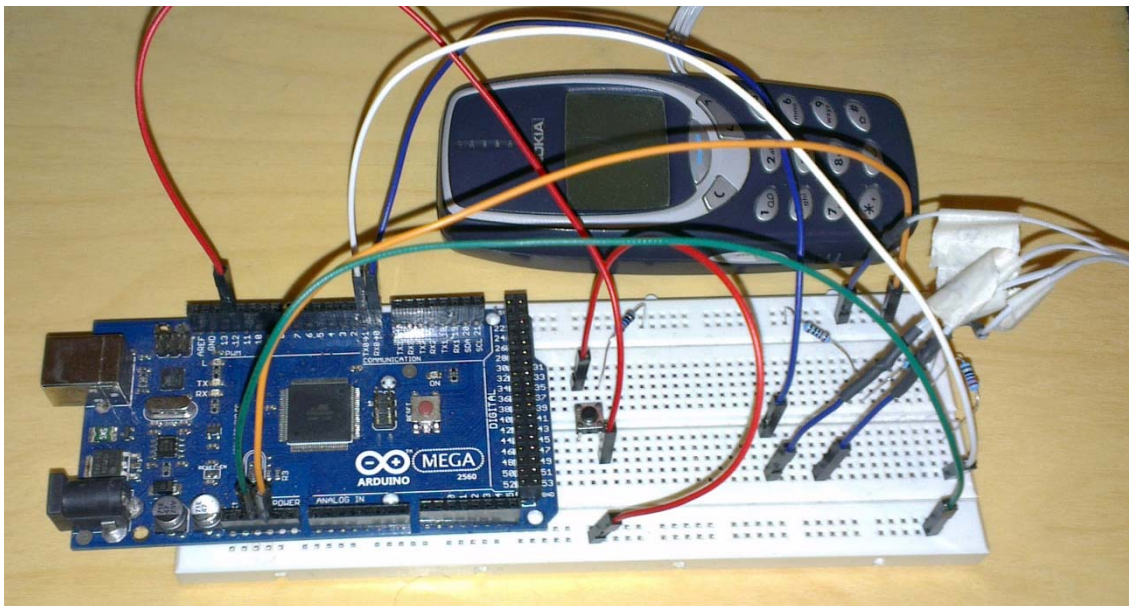


Kuva 8. Puhelin ilman takakuorta ja akkua.

Juotin johtimen myös MBUS-liittimeen, joka on Nokian toinen, vanhempi väylätyyppi FBUS:n rinnalla. MBUS on kaksisuuntainen liitäntä, eli lähetys ja vastaanotto menevät samaa johdinta pitkin, se on kuitenkin hitaampi kuin FBUS (Tuominen 1999). Tätä liitintä ei kuitenkaan tarvitse käyttää, koska samat asiat toimivat FBUS-väylän kautta nopeammin.

8.3 Arduino prototyyppi

Arduinoon rakennettuun prototyyppiin (kuva 9) ei mennyt kovinkaan paljon osia, vaan siihen tarvitsi vastuksia, johtoa ja painonapin. Kuvassa oikealla näkyy jännitteen jakaja, joka alentaa Arduinon viiden voltin käyttöjännitteen n. 3,3 V:iin, joka on sopivalla tasolla verrattuna 3310:n käyttämään jännitteeseen.



Kuva 9. Arduino prototyyppi.

Kuvassa 9 näkyy myös alkuperäinen Arduino Mega 2560. Kuvasta voi nähdä Arduinon keskellä olevan ATmega piirin, sekä muut osat, kuten kiteen ja jännitteen-säätimen, näitä tarvitaan luonnollisesti myös itsenäisessä, rakennettavassa GSM-ohjaimessa.

8.4 FBUS-protokolla

Vanhojen Nokia-puhelimien käyttämä FBUS-väylä on kohtuullisen hyvin dokumentoitu, mutta silti hieman hankalasti ohjelmoitava sarjaliitântä. Yhteys laitteiden välillä synkronoidaan lähettämällä puhelimelle 128 0x55-heksadesimaalia, eli U-kirjainta. Tämän jälkeen yhteys on luotu ja laitteet voivat keskustella keskenään. Nopeudeltaan FBUS on 115200 bittiä sekunnissa, tätä tietoa tarvitaan sarjayhteyttä luotaessa (Peacock 2010).

Puhelin lähettää useista tapahtumista tiedon väylän kautta, esim. vastaanotettu tekstiviesti tulee kokonaisuudessaan mm. lähettäjän ja viestikeskukseen numeron kanssa. Vastaamatta jääneestä puhelusta tulee myös viesti, mutta toisaalta *100# koodilla (DNA prepaid liittymän saldokysely) ei tule mitään, eli kaikkea kommunikointia ei FBUS kuitenkaan näytä.

8.5 Koodin tekeminen

Ohjelmointi alkoi testaamalla FBUS:n ja Arduinon välisen yhteyden toimivuutta InsideGadgets-sivulta löytyneellä koodilla, joka kysyi puhelimelta rauta- ja ohjelmistotietoja (liite 1) (InsideGadgets, 2013). Koodista näkyi myös kommunikaation perusperiaatteet, eli lähinnä liitännän synkronointi. Puhelimen vastaukset näkyivät Arduino IDE:n serial monitorissa (sarjaliikenteen seuraaminen) heksadesimaaliarvoina. Kokeilin myös näyttää arvoja ASCII-merkkeinä ja tämä toimikin jossain määrin, mutta kaikki arvot eivät ole varsinaisia merkkejä vaan muuta dataa, esim. checksumeja. Heksadesimaaleja pystyi tulkitsemaan Wayne Peacockin EmbedTronicsiin kirjoittaman mainion artikkelin avulla ja tästä saattoi päätellä, että liitäntä toimii oikein.

Seuraavaksi oli vuorossa tekstiviestin vastaanottaminen, tässäkin edellä mainittu artikkeli auttoi. Viestin vastaanottaessaan puhelin lähettää sen FBUS-väylään, tähän liikenteeseen sisältyy myös tietoa viestin sijoituksesta puhelimen muistiin ja puhelinnumeroista. Koska kaiken vastaanotetun tiedon käsitteleminen vaatisi paljon koodia, pyrin erottelemaan tiedosta pelkän viestin. SMS-viesti pakataan 7-bittisestä ASCII-tekstimuodosta 8-bittiseksi tilan säästämiseksi. Pakkaus yksinkertaisesti yhdistää bitit peräkkäin ja lisää loppuksi perään nollia pitääkseen arvon 8-bittisenä, esimerkiksi sana "nokiatesti" pakattuna seuraavassa taulukossa (taulukko 1) värikoodattuna luettavuuden helpottamiseksi. Binäärilukuja katsottaessa on syytä muistaa lukusuunnan olevan oikealta vasemmalle.

Taulukko 1. SMS-viestin pakkaus.

AS-CII	HEX	BIN 7-Bit	BIN 8-Bit	HEX
n	6E	1101110	11101110	EE
o	6F	1101111	11110111	F7
k	6B	1101011	00111010	3A
i	69	1101001	00011101	1D
a	61	1100001	10100110	A6
t	74	1110100	10010111	97
e	65	1100101	11100111	E7
s	73	1110011	11110100	F4
t	74	1110100	00110100	34
i	69	1101001	-	-

Ohjelma lukee sarjaväylästä tavuja heksadesimaalina ja oikean kohdalle sattuesssa laittaa sen muistiin. Tämän jälkeen luetaan seuraava arvo ja lisätään se edelliseen. Jos tästä summasta tulee oikea arvo, jatketaan edelleen samalla tavalla, kunnes haluttu arvo ja tietty tavujen määrä on saavutettu. Jos luku on eri kuin haluttu, tyhjennetään "checksum", eli yhteenlaskettujen arvojen muisti ja aloitetaan alusta. Näiden summa-arvojen perusteella mikropiirin ulostulo joko laitetaan päälle tai katkaistaan. Komento "eberstart" käynnistää ulostulopinnan ja käynnistää puolen tunnin ajastimen, joka sammuttaa lähdön automaattisesti. Komento "eberstop" puolestaan sammuttaa lähdön ja katkaisee ajastimen. Ajastin otetaan mikropiirin sisäisestä laskurista, josta näkyy kuinka pitkään AVR on ollut käynnissä. Tämän olisi voinut myös toteuttaa ulkoisella kellopiirillä, mutta sisäinen kello on tähän tarkoitukseen riittävä. Ajastin ja ulostulo voidaan myös tarvittaessa sammuttaa ulkoisella napilla, joka liitetään sisääntulopinniin. Koko koodi on nähtävissä liitteessä 2.

9 Laitteen rakentaminen

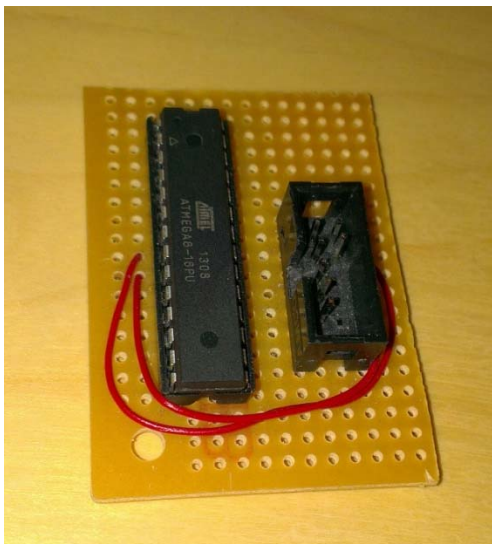
9.1 Arduinon ja piirin erot

Käyttämäni Arduino Mega on rakennettu ATmega2560-piirin ympärille, joka eroaa jonkin verran käyttämästäni ATmega8:sta. Eroavaisuudet tulevat liitännöiden määrästä, muistin koosta ja kellotaajuudesta. Muisti ja liitännät eivät aiheuta ongelmia, koska molempia on enemmän kuin tarpeeksi projektin tarpeisiin. Kellotaajuuden täytyy kuitenkin olla sama, että Megassa käytetty ohjelma toimisi myös itsenäisessä laitteessa. Tämä hoituu kuitenkin 16 MHz:n kiteellä, joka saa normaalisti 8 MHz:n kellotaajuudella toimivan ATmega8:n toimimaan kaksinkertaisella taajuudella.

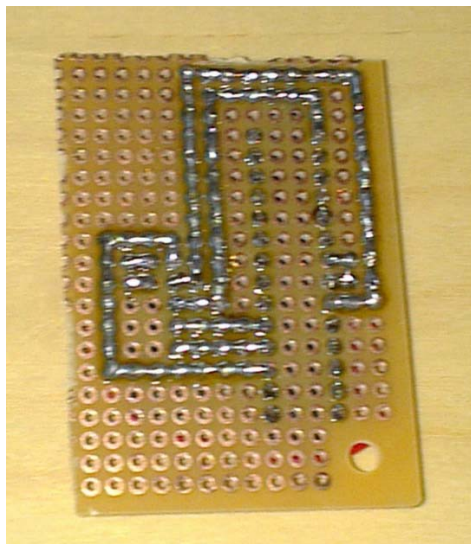
Ohjelmaa täytyi myös muuttaa pinninumeroitien osalta. Arduino Megassa käytin pinniä 13 releelle ja pinniä 12 napille. Nämä eivät kuitenkaan toimineet ATmega8:ssa, joten vaihdoin ne pinneiksi 8 ja 9.

9.2 Ohjelmointikanta

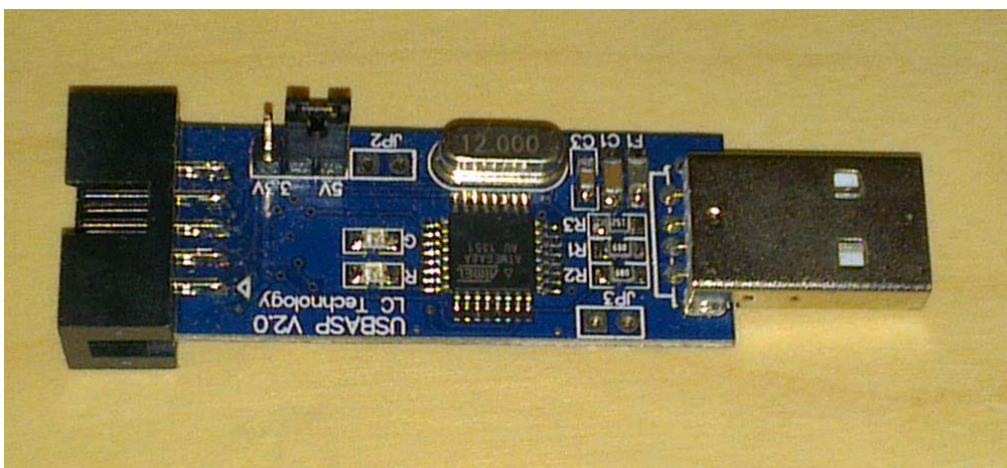
Tein ATmega-piirien nopeaa ohjelmointia varten liitinkannan (kuvat 10 ja 11) USBASP-ohjelmoijan (kuva 12) helppoa liittämistä varten. Adapteri on tehty reikälevylle, johon on juotettu paikoilleen DIP-28-kokoinen mikropiirikanta ja johtoliitin. Liitokset on tehty tinasilloilla ja johdoilla.



Kuva 10. Ohjelmointikanta.



Kuva 11. Kanta alapuolelta.



Kuva 12. USBASP.

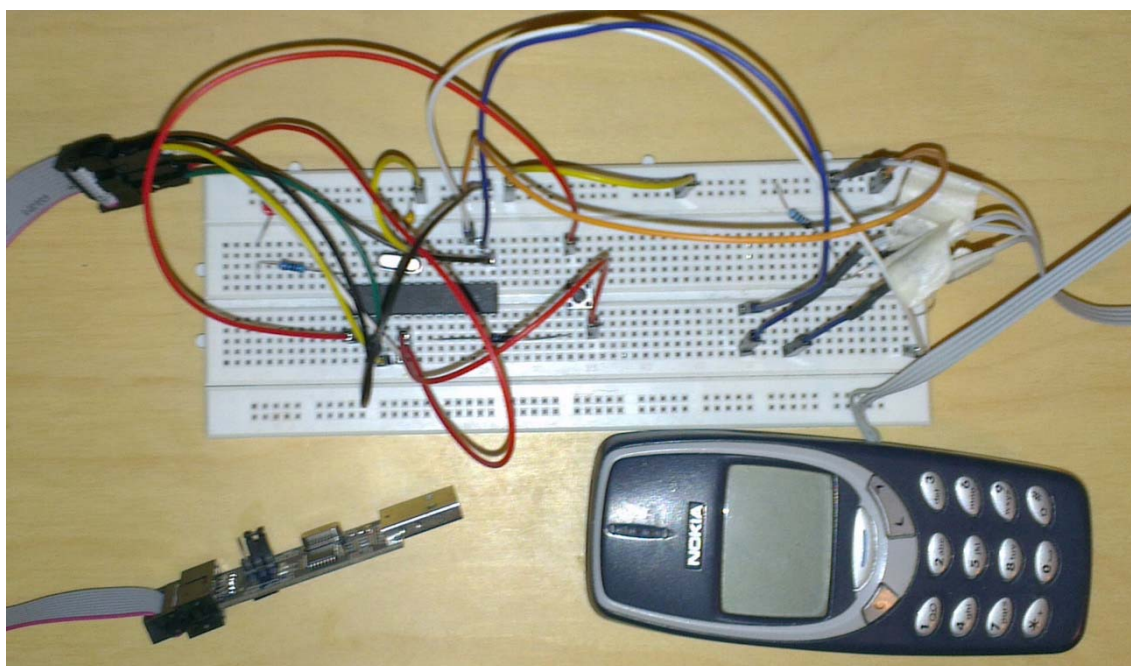
Kantaan olisi vielä voinut lisätä kiteen. USBASP:lla sulaketietoja vaihdettaessa piiri lakkaa toimimasta, koska se yrittää saada kellotaajuutta ulkoiselta lähteeltä. Tämä ei kuitenkaan haittaa, jos asetukset saa laitettua kerralla oikein.

9.3 Testaus ja kasaus

Ensimmäiseksi laitoin sulakeasetukset kuntoon AVRDUDElla. Tämä hoitui kommentoriviltä käsin. Sulakkeiden arvot voi nähdä ATmega8:n tietolehdeltä, sekä myös fusecalc-nimisellä ohjelmalla. Tässä tapauksessa sulakkeiden arvoksi tuli 0xFF ja 0xC9. Nämä arvot määräävät käytettävän kellotaajuuden lähteen, sekä

mm. piirin käynnistysajat. Kyseisillä arvoilla piiri käynnistyy mahdollisimman pitkään, jolloin se varmasti lähtee oikein päälle. Fusecalc antaa suoraan komentoriviltä AVRDUDElla ajettavan komennon, joten asetusten tekeminen oli helppoa.

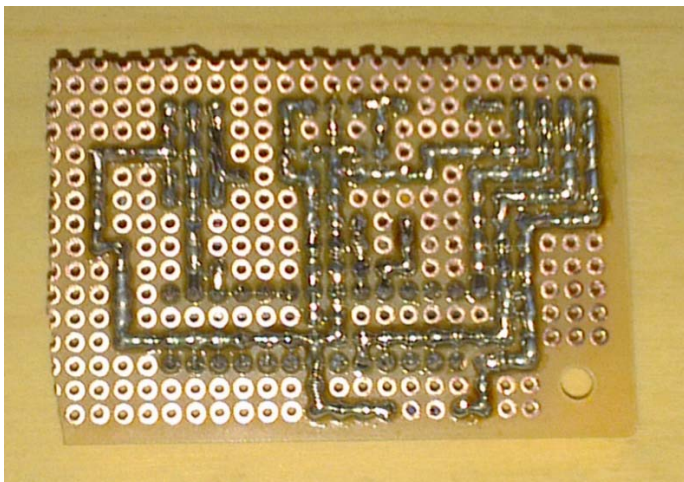
Sulakkeiden asettamisen jälkeen aiemmin Arduinolle tehdyn ohjelman saa lähettettyä ATmegaan USBASPia käyttämällä. Arduino IDE:stä valitaan ohjelmointikaapeliksi USBASP ja alustaksi ”Arduino NG or older w/ ATmega8”. Ennen laitteen kokoamista testasin sen toimivuutta vielä osat irrallaan (kuva 13).



Kuva 13. ATmega8 testikokoonpano.

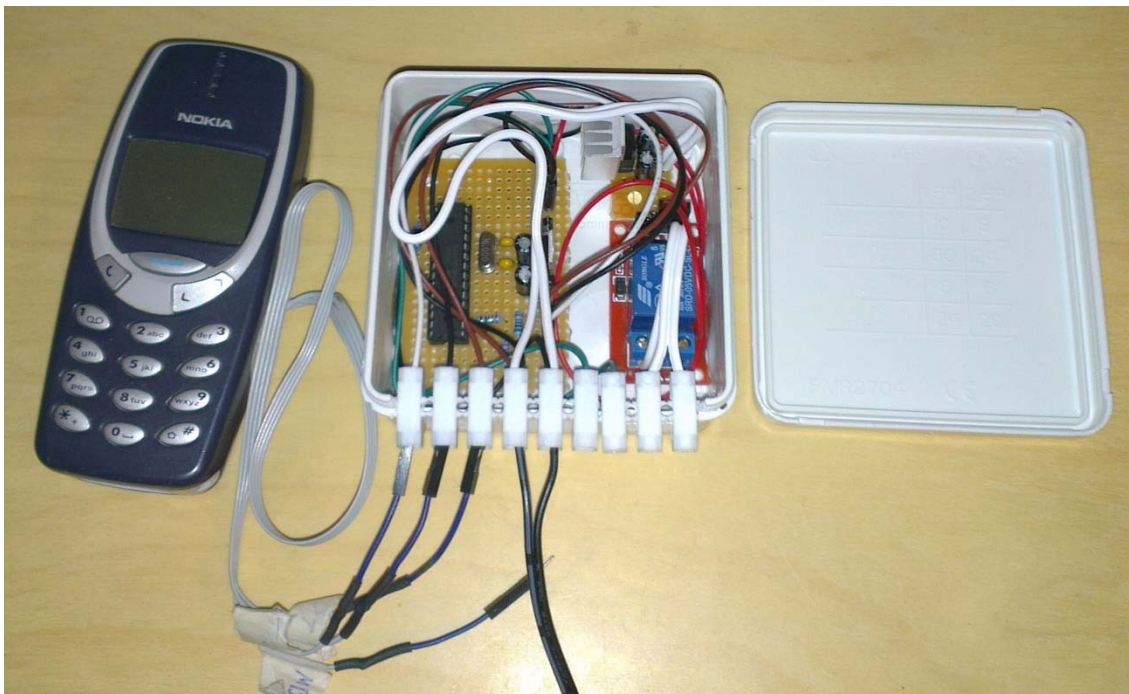
9.4 Valmis laite

Juotettuani laitteen komponentit paikoilleen, leikkasin reikälevyn sopivan kokoiseksi. Tinasillat onnistuivat hyvin ja laitteeseen joutui käyttämään vain yhden irtonaisen johdon. Kuvassa 14 laite on kuvattuna alhaalta, jolloin tinasillat näkyvät hyvin. Tein myös käytettävälle releelle oman 5 voltin virtalähteen, joka on samanlainen kuin piiriä syöttävä regulaattori. Tämä estää jännitteenlaskun releen kytkeytyessä päälle.



Kuva 14. Valmis laite alapuolelta kuvattuna.

Valmiin piirin liimasin sähköasennusrasiaan (kuva 15), sisään laitoin myös releen ja tämän virtalähteen. Koteloon tein kolon, johon laitoin ruuviliittimet johtoja varten. Kotelo on suhteellisen pieni ja kestävä, joten sen sijoittaminen käyttökohteeseen on helppoa. Laitteen liitännät kuvassa vasemmalta alkaen: FBUS GND, FBUS TX, FBUS RX, maa, jännite, nappi, nappi, rele, rele.



Kuva 15. Laite kotelossaan.

10 Asennus autoon

10.1 Asennuspaikka

Opinnäytetyön tekemisen välissä autoni vaihtui, eikä nykyisessä ole tällä hetkellä eberiä, mutta laitetta ei olekaan suunniteltu vain tiettyä autoa varten. Ohjaimen asennuspaikaksi käy esimerkiksi ohjauspyörän alta löytyvä tyhjä tila, kunhan sinne vain saa vedettyä akulta jatkuvan jännitteen. Tarvittaessa voi laittaa johdon myöskin manuaaliselle sammutusnapille.

Puhelimen olisi hyvä olla paikalla, josta siihen pääsee helposti käsiksi. Puhelimen laturille täytyy myös vetää jatkuva jännite.

10.2 Liittäminen lämmityslaitteeseen

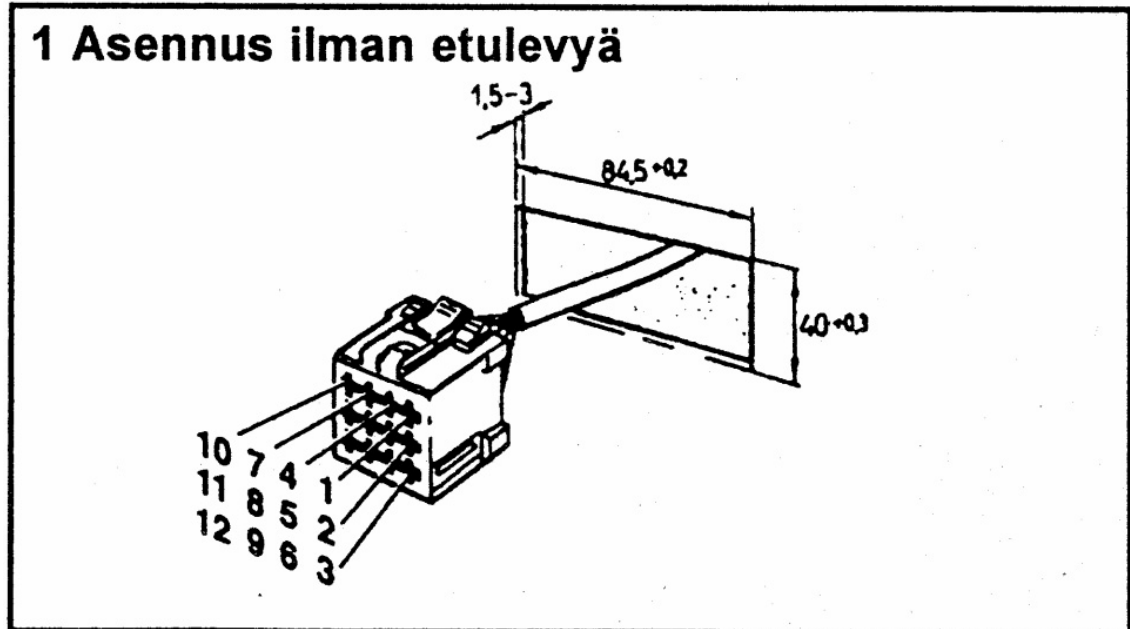
Liittäminen onnistuu suureen osaan Eberspächer-lämmittimiä, kellokytkimestä tulee vain löytyä mahdollisuus ulkoisen kytkimen liittämiseen. Laitte on mahdollista asentaa myös kelloon, jossa ei ole ulkoisen kytkimen paikkaa, tämän käyn läpi myöhemmin.

Kellon asennusnavat löytyvät yleensä sen takaa. Näihin napoihin on helppoa vetää johdotus GSM-ohjaimen releeltä. Napojen tarkempi paikka löytyy kellon asennusohjeesta, käyn kuitenkin läpi muutamia esimerkkejä.

10.2.1 Moduulikellokytkin

Moduulikellokytkin on Eberin perusohjain, joka tarjoaa viikoittaisen ajastuksen ja lisälaiteliitännöitä. Saatavilla on myös minikello, joka tarjoaa vain perustoiminnot, eli kerta-ajastuksen ja välittömän käytön.

GSM-ohjaimen releliittimet voidaan yhdistää esimerkiksi kellon takana olevan liittinpalan (kuva 16) 7. napaan ja auton runkoon. Tämä liitin on tarkoitettu ulkoiselle lämmityskytkimelle ja GSM-ohjain toimii tässä tapauksessa sellaisena.

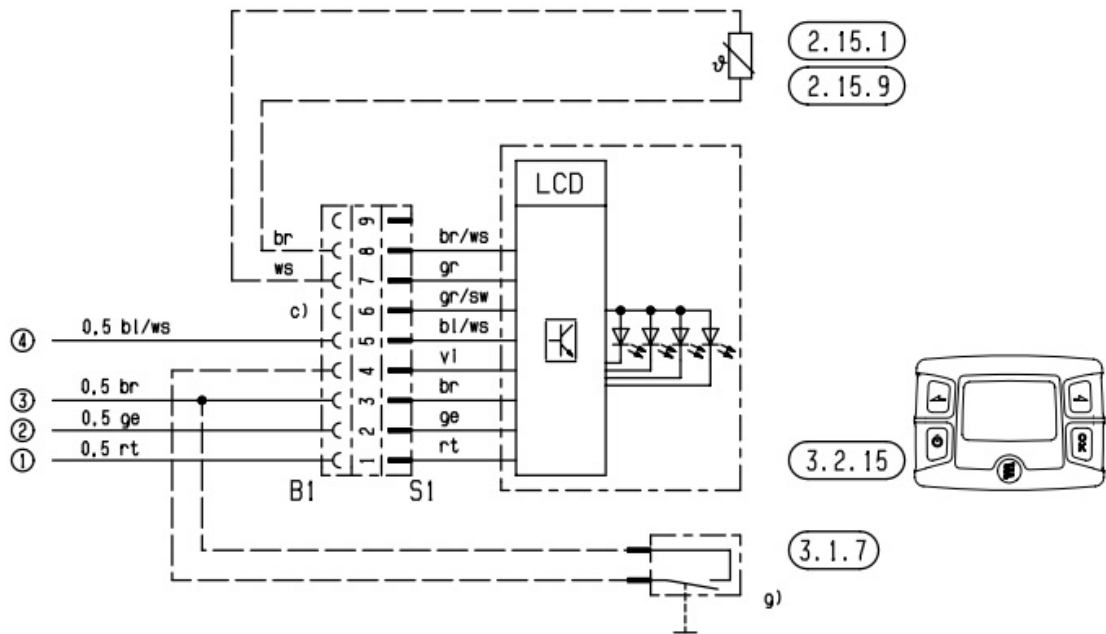


Kuva 16. Moduulikellokytkimen liitin. (Autola 2015).

10.2.2 Easystart T

Moduulikellokytkin korvataan nykyisin useimmin tuoreemmalla Easystart T -kellokytkimellä. Laitteiden ominaisuudet ovat hyvin lähellä toisiaan, suurin ero on ulkomuoto.

GSM-ohjaimen rele voidaan asentaa tässä tapauksessa kaaviossa (kuva 17) näkyvän 3.1.7 kytkimen mukaan, joka on vastaava käyttökytkimen asennuspaikka kuin moduulikellossa. Myös Easystart T:n liittinpala löytyy kellon takaa.



Kuva 17. Easystart T:n liitin. (Autotarvike 2015).

10.2.3 Minikellokytkin

Asennus minikellokytkimeen ei ole suoraan mahdollista, koska siitä puuttuu ulkoisen käyttökytkimen liittimet. Laitetta olisi mahdollista tosin muokata tähänkin sopivaksi muuttamalla ohjelmoinnista käynnistys- ja sammutuskomennot esim. puolen sekunnin pituisiksi pulsseiksi, jotka käyttäisivät relettä vain hetken päällä. Relieliittimet asennettaisiin kellon käynnistypainikkeen rinnalle, tällöin kellon joutuisi tosin avaamaan. Tämä asennustapa ei ole suositeltava.

Asennus voisi olla myös mahdollista suoraan kellon rinnalle, mutta tästä ei ole tietoa valmistajan ohjeissa. En kuitenkaan lähtisi yrittämään tällaista lähestymistapaa. Kello saa vikatietoja suoraan lämmittimeltä ja tällainen asennus voisi olla vahingollinen näille toiminnoille.

11 Pohdinta

Vaikka tämä projekti ei välttämättä vaikutakaan monimutkaiselta, on se silti henkilökohtaisesti auttanut ymmärtämään mikropiirien toimintaa ja GSM-teknologiaa. Myös binäärikoodin ja heksadesimaalien tulkitsemisesta on tullut tutumpaa. Rakennettua laitetta voi mahdollisesti käyttää myös muihin sovelluksiin ja sitä voi myös tarvittaessa laajentaa useammilla releillä. Jatkokehityksenä ohjaimen voisi myös tehdä laajemman tekstiviestien hallinnan, esim. viestin puhelimesta poiston ja tekstiviestien lähetyksen.

Etäohjaus tulee varmasti vielä kehittymään ja yleistymään lähitulevaisuudessa, kun tekniikan ja elektroniikan määrä vain jatkaa kasvuaan. Samalla esimerkiksi tehdastyössä ihminen korvautuu koneella yhä enemmän ja tämä varmasti vaikuttaa myös etähallinnan tarpeeseen. GSM-ohjaus puolestaan on varmastikin saavuttanut teknisen kehityksensä huipun, koska teknologian rajat tulevat vastaan. Toisenlaiset puhelin- ja viestintäteknikat tosin varmasti tuovat uusia mahdollisuuksia myös etäkäytön saralla. Vaikka GSM-pohjainen ohjaus ei enää kehittyisikään, on sillä silti sijansa jokapäiväisessä valvonnassa ja ohjauksessa jo ihan yleisyytensä puolesta.

Lähteet

- Altera. 2015. Enhanced Temperature Device Support. <https://www.altera.com/products/common/temperature/ind-temp.html>. 11.4.2015.
- Arduino. 2015a. Arduino Frequently Asked Questions. <http://arduino.cc/en/Main/FAQ>. 30.3.2015.
- Arduino. 2015b. Arduino GSM Shield. <http://arduino.cc/en/Main/ArduinoGSMShield>. 18.3.2015.
- Arduino. 2015c. Arduino pin current limitations. <http://playground.arduino.cc/Main/ArduinoPinCurrentLimitations>. 11.4.2015.
- Arduino. 2015d. Products. <http://arduino.cc/en/Main/Products>. 16.3.2015.
- Atmel. 2015a. Atmega8. http://www.atmel.com/Images/Atmel-2486-8-bit-AVR-microcontroller-ATmega8_L_datasheet.pdf. 18.3.2015.
- Atmel. 2015b. ATtiny. <http://www.atmel.com/products/microcontrollers/avr/tinyavr.aspx>. 11.4.2015.
- Autoextra. 2015. Easystart Call. <http://autoextra.fi/products/php/EasyStart%20Call%20gsm-ohjaus.php>. 11.4.2015.
- Autola. 2015. Eberspächer käyttöohjeet. http://www.autola.fi/tuoteryhmat/ajoneuvolammittimet/fi_FI/eberikayttoohjeet/. 9.2.2015.
- Auto-Standheizung. 2015. Easystart R+. <http://auto-standheizung.com/Bedienelemente/Eberspaecher/Eberspaecher-EasyStart-R.html>. 11.4.2015.
- Autotarvike Oy. 2015. Eberspächer asennusohjeita. <http://www.autotarvike.fi/eberspacher-asennusohjeita.html>. 18.3.2015.
- Eeperi. 2015. Eberspächer lisävarusteet. http://www.eeperi.com/eberspacher_lisavarusteet.html. 11.4.2015.
- Farhi, Paul. 17.2.2007. The Inventor Who Deserves a Sitting Ovation. <http://www.washingtonpost.com/wp-dyn/content/article/2007/02/16/AR2007021602102.html>. 11.4.2015.
- Google Play. 2015. Easystart Call. <https://play.google.com/store/apps/details?id=synergetic.easystartcall&hl=fi>. 11.4.2015.
- InsideGadgets. 12.1.2013. How to use Nokia F-bus to send an SMS message. <http://www.insidegadgets.com/2013/01/12/how-to-use-nokia-f-bus-to-send-an-sms-message/>. 18.3.2015.
- Karau, Mark D. 2003. Wielding the Dagger: The Marinekorps Flandern and the German War. USA: Greenwood Publishing Group. 91.
- Kfz-braun_info. 2015. Easystart Call <http://www.ebay.com/itm/Eberspacher-EasyStart-Call-rufen-Sie-Ihre-Standheizung-an-/110975615991>. 11.4.2015.
- Lichigan, Alexander. 2004. Chto takoe teletank? <http://www.odintsovo.info/news/?id=1683>. 11.4.2015.
- Peacock, Wayne. 9.7.2010. FBUS. <http://web.archive.org/web/20120712020156/http://www.embedtronics.com/nokia/fbus.html>. 18.3.2015.
- Raspberry Pi. 2015. Documentation. <http://www.raspberrypi.org/documentation/>. 18.3.2015.
- Robot Technology. 2015. History of Microcontroller ATMEL AVR. <http://robottechno.us/history-microcontroller-atmel-avr.html>. 11.4.2015.
- Sarkar, Tapan K. 2006. History of wireless. USA: Wiley-IEEE Press. 276-278.
- Savannah. 2015. AVRDUDE. <http://www.nongnu.org/avrdude/>. 11.4.2015.

- Stinson, Ben. 2015. Nokia's 3310: the greatest phone of all time. <http://www.techradar.com/news/phone-and-communications/mobile-phones/nokia-s-3310-the-greatest-phone-of-all-time-1287636>. 11.4.2015.
- The Telegraph. 2012. Eugene Polley. <http://www.telegraph.co.uk/news/obituaries/9285576/Eugene-Polley.html>. 18.3.2015.
- Tuominen, Panu. 1999. FBUS & MBUS adapters. <http://www.panuworld.net/nuukiaworld/hardware/cables/basics.htm>. 18.3.2015.
- Webasto. 2012. Tuoteluettelo. http://www.kaha.fi/file_root/Lisavarusteet/Webasto/Luettelot/WEBASTO-Luettelo-2012.pdf. 28.12.2014.
- VIA. 2015. VIA Spearhead. <http://www.via.com.tw/en/initiatives/spearhead/>. 18.3.2015.

FBUS-testikoodi

```
byte msg[] = {
0x1E, 0x00, 0x0C, 0xD1, 0x00, 0x07, 0x00, 0x01, 0x00, 0x03, 0x00, 0x01, 0x60,
0x00, 0x72, 0xD5 };

void setup() {
  Serial.begin(115200);
  delay(1000);
}

void loop() {
  // Initialise the F-bus
  for (int z = 0; z < 128; z++) {
    Serial.write(0x55);
  }
  Serial.println("");
  delay(100);

  // Send our command
  for (int x = 0; x < (sizeof(msg) / sizeof(byte)); x++) {
    Serial.write(msg[x]);
  }
  Serial.println("");

  // Wait for a reply
  while (1) {
    while (Serial.available() > 0) {
      int incomingByte = Serial.read();
      Serial.print(incomingByte, HEX);
      Serial.print(" ");
    }
  }
}
```


ATmega8 lähdekoodi

```
// Hardware Info viesti yksittäisinä tavuina
byte msg[] = {
0x1E, 0x00, 0x0C, 0xD1, 0x00, 0x07, 0x00, 0x01, 0x00, 0x03, 0x00, 0x01,
0x60, 0x00, 0x72, 0xD5 };

int checksum = 0;
long timeStop = 0;
long timeCurrent = 0;
int start = 0;
const int outputPin = 8;
const int buttonPin = 9;
int buttonState = 0;
int checkCount = 0;

// Alustetaan pinnit ja aloitetaan sarjayhteys
void setup() {
  pinMode(outputPin, OUTPUT);
  digitalWrite(outputPin, LOW);
  pinMode(buttonPin, INPUT);
  Serial.begin(115200);
  delay(1000);
}

// Looppi joka alustaa yhteyden, 128x U
void loop() {
  for (int z = 0; z < 128; z++) {
    Serial.write(0x55);
  }
  delay(100);

  // Lähetetään Hardware Info viesti
  for (int x = 0; x < (sizeof(msg) / sizeof(byte)); x++) {
    Serial.write(msg[x]);
  }

  // Looppi, odotetaan viestejä puhelimelta
  while (1) {
    while (Serial.available() > 0) {
      // Luetaan yksittäinen tavu
      int incomingByte = Serial.read();
      // Lasketaan vastaanotetusta tavusta checksum arvoa ja
      // tarkistetaan onko se hyväksyttävä
      checksum = checksum + incomingByte;
      if (checksum == 0x65 || checksum == 0xD6 || checksum == 0x12F || check-
sum == 0x16D || checksum == 0x214 || checksum == 0x29B || checksum ==
0x380 || checksum == 0x3F4 || checksum == 0x2D3 || checksum == 0x3B4) {
        checkCount = checkCount + 1;
      }
      else {
```

```
checksum = 0;
checkCount = 0;
}
if (checksum == 0x3F4 && checkCount == 8) {
  // Jos eberstart cheksum ja
  // arvoja on laskettu tarpeeksi, ulostulo päälle
  // 30min
  checksum = 0;
  checkCount = 0;
  start = 1;
  timeStop = timeCurrent + 1800000;
  digitalWrite(outputPin, HIGH);
  delay(1000);
}
if (checksum == 0x3B4 && checkCount == 7) {
  // Jos eberstop checksum ja tarpeeksi arvoja
  // sammutus
  checksum = 0;
  checkCount = 0;
  start = 0;
  digitalWrite(outputPin, LOW);
  delay(1000);
}
}
timeCurrent = millis();
buttonState = digitalRead(buttonPin);
if (start == 1 && timeStop < timeCurrent) {
  // Lopetus jos aikaa kulunut tarpeeksi
  start = 0;
  digitalWrite(outputPin, LOW);
  delay(1000);
}
// Manuaalinen sammutus
if (buttonState == HIGH) {
  start = 0;
  digitalWrite(outputPin, LOW);
  delay(1000);
}
}
}
```