TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikka, elektroniikka
Marko Reponen

Tutkintotyö

Marko Reponen

Kaukosäädin säädettävään moottorinohjausjärjestelmään

Työn valvoja : Matti Ilmonen

Työn ohjaaja : Tatu Tikanmäki

Tampere 2006

Tekijä: Marko Reponen
Työn nimi: Kaukosäädin säädettävään moottorinohjausjärjestelmään

Päivämäärä: 03.05.2006
Sivumäärä: 17 sivua ja 26 liitesivua
Hakusanat: Kaukosäädin, mikrokontrolleri, moottorinohjaus, Tatech
Koulutusohjelma: Tietotekniikka
Suuntautumisvaihtoehto: Elektroniikka

Työn valvoja: Matti Ilmonen

Työn ohjaaja: Johtaja Tatu Tikanmäki
Softatech tmi, Pirkkala

Työssä tehtiin moottorinohjausjärjestelmään liitettävä ulkoinen säätölaite, jolla voidaan seurata reaaliaikaisesti moottorin tapahtumia sekä ohjata moottorin toimintaa. Laite on tarkoitus tehdä mahdollisimman kaupalliseksi ja helppokäyttöiseksi. Laitteisto toimii eräänlaisena kehitysalustana tuotannon aloittamiseen. Tuotteen ulkonäkö, ominaisuudet sekä käytettävyys muuttuu ennen kuin tuote kunnolla tullaan muuttamaan kokonaan kaupalliseksi.

Työ sisältää kilpailijoiden vastaavien tuotteiden tutkimista ja arvioimista, komponenttien kilpailuttamista ja valitsemista, piirikaavion ja piirikuvan piirron, piirilevyjen hankinnan, laitteistoläheisen ohjelman kirjoituksen sekä testauksen.

Komponenteille tehtiin testauksia, mitataan kytkinvärähtelyä, mitoitetaan ja mitataan virtoja sekä tuotetta käytetään todellisessa käyttöympäristössä. Tuotteelle asetetaan tiettyjä raja-arvoja jotka sen pitää täyttää, samoja rajoja asetetaan myös laitteen sisäisille komponenteille.

Laitteen tarkoitus säätö ja seuranta mahdollisuuksien lisäksi on korvata kokonaan kannettava tietokone haettaessa moottorin säätöarvoja kohdalleen, laite on toimintavalmis hyvin nopeasti ja sitä voidaan säilyttää autossa kokoajan. Loppukäyttäjälle tehdään myös kytkentäohje jonka perusteella laitteiston osaa liittää olemassa olevaan järjestelmään helposti.

Työkalut millä laitetta suunnitellaan ja testataan ovat joko ilmaisia testausversioita, esimerkiksi Codevisionin AVR-kääntäjä, tai sitten työn tekijän, esimerkiksi Fluke 123 digitaalinen oskilloskooppi.

Author: Marko Reponen
Title: Remote control to adjustable engine management system

Date: 03.05.2006
Number of pages: 17 pages and 26 appendix pages
Keywords: Engine management, microcontroller, remote control, Tatech
Program: Computer Systems Engineering
Specialisation: Electronics

Supervisor: Matti Ilmonen

Instructor: Manager Tatu Tikanmäki
Softatech tmi, Pirkkala

This work is considers of a remote control unit to adjustable engine management system, wich allows users to monitor events in engine realtime and tune engines functions. Device is planned to be commercial and as easy to use as possible. Equipment works as a sort of developement board for starting manufacture. Devices appearance, features and usability will change before it is ready to production.

Work includes competitors similar units study and evaluation, picking and component achieving, design of the schematic and lay-out of device, aquiring the printed circuit boards, writing and testing of systems firmware.

Component's will be tested, measure switch noise, calculate and measure currents and product will be used in real using enviroment. Product will have some limitations that it is required to fullfil, same requirements will be set to the inner components.

Device is meant to replace laptop in engines tune-up sessions, the device is up and running very fast and it can be stored in vehicle. End user will get connecting diagram so that the device can be connected easily to allready existing system.

Tools that are used to design and test this device are evaluation versions, for example Codevision AVR-programmer, or own equipments, for example Fluke 123 digital oscilloscope.

Sisällysluettelo

Käytetyt merkinnät ja termit

LCD	Liquid Crystal Display, nestekidenäyttö.
EEPROM	Electrically-Erasable Programmable Read-Only Memory, sähköisesti tyhjennettävissä oleva, vain luettavissa oleva muisti.
FLASH	Samanlainen muistityyppi kuin EEPROM, erona on muistin käsittelyn tyyli sekä yleensä suurempi muistin tiheys.
USB	Universal Serial Bus, nopea sarjaväyläyhteys differentiaalisilla tiedonsiirtojohtimilla.
UART	Universal Asynchronous Receiver-Transmitter, laitteiston osa joka hoitaa sarjaliikennöinnin rautatasolla.
I/O	Input / Output, sisääntulo tai lähtö mikrokontrollerissa.
PWM	Pulse-width Modulation, pulssin leveyttä säätelemällä tehtävä tehon ohjaus.
RS232	Standardoitu sarjaliikenne protokolla.
MILS	Mittayksikkö piirilevyn piirrossa.
EMC	Electromagnetic compatibility

Alkusanat

Tämä tutkintotyö on tehty Tampereen ammattikorkeakoulun elektroniikkalinjan insinöörityönä.

Kiitos työnantajalle hyvästä työn aiheesta sekä hyvistä neuvoista ohjelman kehityksen suhteen ja Matti Ilmoselle neuvoista työn kirjoittamiseen.

Tampereella 3.toukokuuta 2006

_____

Marko Reponen

Tehtävän anto

Laitteiston kuvaus

Työssä piti suunnitella ja rakentaa käyttäjän vapaasti ohjelmoitavaan moottorinohjausjärjestelmään kaukosäädin, jolla pääsee säätämään ja seuraamaan moottorin toimintaa. Lisäksi työn tekijät oppivat tärkeitä tiedonhankintataitoja, luovat yhteyksiä komponenttijakelijoihin sekä oppivat suunnittelemaan ja arvostelemaan kriittisesti oman työn jäljen ja suunnittelun suhteellista vaikutusta lopullisen tuotteen hintaan.

Tiedot moottorinohjauksesta siirtyvät sarjaväylää pitkin kaukosäätimelle, jossa ne pitää muuntaa sellaiseen muotoon, että niillä voidaan käyttää porteissa sijaitsevia ledejä tai tulostaa LCD-näytölle esimerkiksi imuilman numeerisina lämpötilatietona.

Työnantajana toimii Softatech tmi jonka tuotteeseen TATECH kaukosäädin liittyy. Kaukosäätimen tarkoitus on korvata kannettava tietokone moottorin säätöarvoja kohdalleen säätäessä esimerkiksi jarrudynamometrissä. Laitetta on tarkoitus pitää kokoajan ajoneuvossa mukana, käyttötilanne on normaalisti sellainen että näytetään vain moottoriin liittyviä parametrejä.

Laitteiston sisäiset vaatimukset

Laitteessa tuli olla sisäistä muistia tiettyjen näyttömoodien ja muiden mahdollisten lisätoimintojen asetusten tallentamiseen. Muistin tyyppi ja totetustapa oli vapaavalintainen, rajaten mahdolliset vaihtoehdot kuitenkin käytännössä joko EEPROM tai FLASH tyylisesti toteutettuun muistiin. Muistiin pitäisi myös pystyä lataamaan ohjelmamuisti sekä mahdollinen laitteistoläheinen ohjelmanlatausosio. Käyttäjä voisi siis itse päivittää laitteiston ohjelman omalla tietokoneella niin halutessaan.

Kaukosäätimeen liityntöinä mietittiin sarjaporttia, infrapunaa, bluetoothia sekä USB:tä. Näistä heti alussa oli järkevinä vaihtoehtoina vain sarjaportti sekä USB, muut ovat liian työläitä, eksoottisia sekä kalliita ratkaisuita. Useimmissa mikrokontrollereissa on sisäänrakennettuna UART tai USB-liityntä rautapohjaisesti, nämä liitynnät olivat ehdottomat kontrolleria harkittaessa.

Laitteiston ulkoiset vaatimukset

Laitteen ulkoisen olemuksen piti täyttää seuraavat kriteerit : olla siro, tyylikäs sekä toimiva. Kaukosäätimen käyttäjälle näkyväksi rajapinnaksi määriteltiin led-rypäs sekä taustavalaistu LCD-näyttö. Led-valoilla indikoidaan jatkuvasti moottorin lambda-arvoa, kun taas LCD-näytössä on käyttäjän vapaasti valitsemat tiedot.

Ledipylvään lukema-alueen piti kuvata välin 0.8 – 1.1 lambda-arvoa, kattaen

moottorin palamistiedon rikkaasta laihaan päähän. LCD-näytössä tuli olla luettavissa moottorin pyörimisnopeus, imusarjan ali- tai ylipaine sekä jäähdytysveden ja imuilman lämpötilat. Käyttäjän pitäisi myös päästä valitsemaan näytettävä näyttötila, päästä säätämään sytytysennakkoa, määräämään polttoaineen suihkutuksen kestoaika, aktivoimaan lähtömoodi sekä määrämään kierrostenrajoittimen kohta.

Näytön piti olla helposti luettavissa kirkkaassa päivänvalossa, laitteessa piti olla myös painonappeja ja pyöritettävä säätölaite toimintojen valitsemiseen sekä kartastojen säätämiseen. Painonappien piti olla hyvälaatuiset mekaanisesti sekä sähköisesti ja säätörullan piti olla joko potentiometri- tai kiertokytkin -tyylinen. Säätörullalla pitäisi päästä vaikuttamaan yksittäiseen polttoaine- tai sytytysennakkokarttaan tai päästä siirtämään kokonaisen kartaston tasoa. Säätörullalla ajateltiin olevan helpoin säätää polttoainemääriä sekä sytytysennakkoja nopeassa ajo- tai säätötilanteessa, painonapeissa saattaa käyttäjä mennä helposti sekaisin.

Suunnitteluohjelmien määrittely

Piirilevyn suunnittelun osalta oli päätettävä aluksi käytettävä ohjelma. Lopputyön piirtämisessä on käytetty Eaglen piirilevy-suunnitteluohjelmaa, opiskelijaversiota jolla voi arvioida tuotteen sopivuuden itselleen mahdollista tulevaisuudessa hankittavaa isompaa yrityskohtaista lisenssiä varten. Piirilevystä ensimmäinen versio oli tarkoitus syövyttää ja porata itse, näin voi karsia suurimmat ajatusvirheet pois suunnitelmasta ennen levyn suurempaa kaupallista tilausta, mahdolliset korjaukset eivät kuormita kovinkaan paljoan lopputuotteen kasausvaiheessa. Kaukosäätimen sisäisen ohjelmiston suunnitteluun ja toteutukseen päätettiin käyttää Codevision avr-ohjelman kokeiluversiota.

Loppukäyttäjälle tulostettavat kytkentäohjeet piirretään AutoCad:llä, sekä muut mahdolliset asennus tai kytkentäohjeet. Mittaustulokset mitä tulostetaan oskilloskoopilta tehdään taas Flukeview Scopemeter-nimisellä ohjelmistolla, tietokoneen sarjaporttiin liitetään muunnoskaapeli ja Fluke 123 oskilloskooppi liitetään siihen kiinni.

Ratkaisuvaihtoehdot

Vastaavat järjestelmät

Työssä piti kiinnittää huomiota tuotteen välillisiin sekä välittömiin kustannuksiin, tarkoittaen alinhankintana teetetyn kasauksen hintaa ja suoria komponenttien hintoja. Myös kilpailijoiden valmiita vastaavia tuotteita etsittiin ja tutkittiin.

Hestec tarjoaa kaukosäätimen omaan tuotteeseensa samalla periaatteella kuin tässäkin toteutettava kaukosäädin. Ainoastaan eroina on se että Hestecin yksikkö toimii vain näyttönä, sillä ei voi säätää polttoaine- eikä sytytyskartastoja ja Hestecin näyttö on 4x20 kooltaan ja väri on vihreä jossa mustat kirjaimet[1]. Haltec, Multec ja Bosch eivät tarjoa minkäänlaista ulkoista näyttömoduulia tuotteisiinsa, joten tuotetta voi tarjota tuotteistamisen jälkeen heidänkin tuotteeseensa muokattuna.

Komponenttivaihtoehdot

Kotelo

Kotelon vaihtoehtoina oli suunnitella itse kotelo sekä teetättää tarvittavat aukotukset ja kanttaukset alihankkijalla. Tästä luovuttiin kaukosäätimen ensimmäisen version ollessa kyseessä, sillä on olemassa se riski että tuote muokkaantuu vielä reilusti ennenkuin se saavuttaa lopullisen muotonsa ja toiminnallisuutensa. Vaihtoehtoisena kotelotyyppinä oli Hammondin läpinäkyvä sinisen sävyinen muovikotelo.

Hankintakanavat ja lämpötilakestot

Komponenttien osalta oli myös päätettävä lämmönkestoluokista, kotelointityypeistä ja jälleenmyyjistä joilta osat hankitaan. Verrokkina hintoihin on pidetty Farnellin nettisivuja, joiden hinnat edustavat suurimmalta osalta komponentteja kalleinta päätä, eli tuotteen hinta tulee tippumaan kun komponentit ostetaan suoraan isommilta jakelijoilta. Komponenttien lämpöluokitus tuli olla vähintään rajoihin -20°C - +60°C menevä, LCD-näytön osalta riitti 0°C asti toimiva näyttö.

Valinnoissa on mietitty komponenttien hintaa sekä tulevaisuuden saatavuutta. On vältettävä tilannetta että tuotteeseen joutuu suunnittelemaan uuden piirilevyn, teetättämään uudet maskit piirilevytehtaalle sekä miettimään koteloinnin uusiksi. Lisäksi useat rauta- ja ohjelma-kierrot lisäävät turhan paljon kehityskustannuksia, tämän vuoksi kannattaa pitäytyä yhdessä suunnitelmassa, sekä ottaa huomioon mahdollisimman paljon muuttuvia tekijöitä komponenttivalinnoissa.

Mikro-ohjain ja piirilevyn rakenne

Pääohjainprosessorina oli vaihtoehtoina Atmelin Atmega128 -mikrokontrolleri tai vastaava ARM7 tason mikro-ohjain. ARM7 vaihtoehtoa puolustaisi uuden kontrolleriperheen ohjelmoinnin ja käyttämisen oppiminen, toisaalta Atmegaa

puolusti taas sen helppokäyttöisyys jo muihin saman kontrolleri-perheen tuotteiden käyttöön tutustuneena. Molemmissa ohjaimissa on sisäistä EEPROM:ia, FLASH:ia, paljon I/O-nastoja sekä laskureita, mikä puoltaa näiden piirien valintaa.

Piirilevyn osalta mietittiin kahden levyn ratkaisua, jossa olisi erikseen CPU-kerros missä on mikrokontrolleri ja tietoliikennepiiri, toisella levyllä olisi LCD-näytön liityntä, indikointiledit, painonapit sekä käännettävä kiertokytkin tai potentiometri. Tämän lisäksi oli vaihtoehtona yhden levyn ratkaisu, jossa olisi samalla levyllä kaikki kahden levyn komponentit.

Indikointi- ja säätölaitteet

Työssä seuraavaksi mietittiin käyttäjälle näkyvän indikoinnin määrää, tarkoituksena oli kuitenkin käyttää vain 2-rivistä LCD-näyttöä sen hinnan takia, sekä ilmaista lambda-arvo jotenkin muuten kuin näytöllä pelkästään numeroina. Vaihtoehtoina oli käytännössä erillinen led-pylväs ja LCD-näytöllä tapahtuva graafisen palkin vieritys.

Valintanäppäimiksi mietittiin halvahkoja peruspainonäppäimiä, joissa ei ole mitään indikointia itsessään, tai parempilaatuisia valaistuja painonappeja. Säätörullan osalta mietittiin taas potentiometriä sekä kiertokytkintä, tarkoitushan oli päästä säätämään kartastoa luontevasti, mikä taas onnistuu helpoiten kierrettävällä kytkimellä. Painonappi korjauskartan käyttöön oli mielestämme hieman kömpelö.

Valittu ratkaisu

Komponenttivalinnat

Kotelon valinta

Kotelon vaihtoehtoa puntaroidessa kävi mielessämme tehdä itse tai teetättää alumiinista kotelo tietyn kokoiselle näytölle sekä led-valoille. Ajatuksesta kuitenkin luovuttiin, kun huomattiin teetetyn kotelon tulevan maksamaan liian paljon. Itse tehtyyn koteloon taas menisi liian kauan aikaa eivätkä kotelot olisi varmasti identtisiä. Tästä johtuen pitäydyttiin Farnellin kuvastosta löytyneessä sinisessä läpinäkyvässä kotelossa[2]. Kuvassa 1 on esiteltynä saman tuoteperheen erikokoisia vaihtoehtoja, tuotteeseen on valittu kuvassa alinna oleva kotelo. Kotelon ulkomitat kannen kanssa ovat : korkeus 40mm, leveys 120mm ja syvyys 65mm.

Kuva 1: Hammond 1591-C kotelo

Kontrolleri ja apupiirit

Työssä tullaan käyttämään Atmel Mega128-mikrokontrolleria pääpiirinä. Apupiireinä on MAX233-yhteensopiva SP233ECP piiri, joka hoitaa sarjaliikennetasoon TTL-tasosta sekä toisinpäin[3]. SP233ECP on pintaliitoskomponentti, kotelona on SOIC-20, samoin on myös Mega128, jonka kotelotyyppinä on TQFP-64. Mega128:ssa on 53 ohjelmoitavaa I/O-nastaa, joiden kaikkien toiminnallista suuntaa on mahdollisuus vaihtaa, sisääntuloina toimiessa on myös mahdollisuus käyttää sisäistä ylösvetovastusta, joka estää turhan pinnin heilumisen ohjelmallisessa mielessä. Lisäksi piirissä on LCD-näytön taustavalon säätöä ajatellen tärkeitä rautatason PWM-lähtöjä, joita kannattaa käyttää kyseiseen työhön. Samoin on useampi 8/16-bittinen laskuri, joilla voidaan toteuttaa laskentaa, vaikka saada kellonaika näyttöön. Samalla saadaan mahdollisuus led-pylväiden PWM-ohjauskäytön hoitamiseen ohjelmallisesti, jos tarvetta ilmenee. Kiteen arvoksi valittiin 14,765MHz ja tätä vastaaviksi kondensaattoreiksi kiteelle tuli datalehden mukaan 22pF[4]. Valintaa puolsi nopea kellotaajuus sekä 0% virheen tuova kerroin sarjaliikennettä ajatellen.

Tietoliikenneyhteytenä päälaitteen kanssa päätettiin pitää sarjaporttia. Tämä ohjainlaitteen takia, jossa on samalla tavalla toteutettu liityntä tietokoneeseen päin, muu lähestymistapa olisi ollut turhan työlästä. USB-liityntää kokeiltiin FTDI:n valmistamalla USB-RS232-muunninpiirillä ja tämä havaittiin toimivaksi, mutta Windowsin ajuriliitynnän kirjoittaminen ilmeni turhan vaikeaksi, joten ideasta luovuttiin muuten toimivana[5]. USB-muunninpiirin testaus tapahtui hyvin aikaisessa vaiheessa suunnitelmaa, testauspiirilevynä oli aikaisemmin suunniteltu muunninpiirilevy, jossa komponentit olivat jo valmiina.

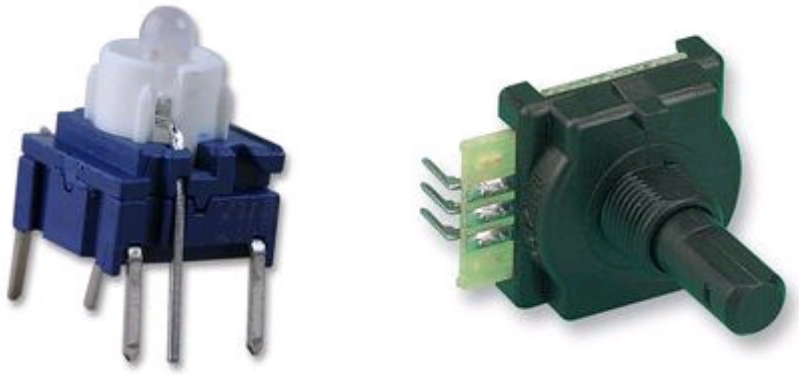
Mega128:n valintaa puoltaa se, että ei tarvita ulkoisia komponentteja porttiohjauksiin, sillä mikrokontrolleri sisältää tarpeeksi monta i/o-pinniä kaikkien hoitamiseen. MAX233-yhteensopivan piirin perustelu on taas sen ulkoisten jännitepumppukondensaattorien tarpeettomuus sekä markkinoilla olevien vastaavien piirien määrä, aina voi vaihtaa komponenttitoimittajan, jos joku toimittaja ei kykene toimittamaan komponentteja tarpeeksi.

Indikointi ja säätölaitteet

Työ jatkui miettimällä tarvittavien käyttöpainikkeiden määrä, toiminnallisuus sekä mahdolliset niissä olevat indikoinnit. Erilaisia vaihtoehtoja tutkittiin useiden eri komponenttitoimittajien kirjoista sekä verkkosivuilta.

Painonapeiksi päädyttiin valitsemaan MEC:n valmistamat painonapit, joissa on sisäänrakennettu 2-värinen valodiodi tilan indikointia varten[6]. Ledin väriä muutetaan vaihtamalla jännitteen napaisuus ledin syöttöjaloissa. Painonappien ominaisuutena on hyvin pieni, jopa olematon, kytkinvärähtely. Tämä säästää ulkopuolisen rc-piirin verran levytilaa ja pieni värähtelysuodin on tarpeen vaatiessa helppo tehdä vapaasti juokseviin laskureihin pohjautuen.

Kiertokytkimeksi valittiin pitkällisen etsiskelyn jälkeen Vishay 110E- mallinen 2-bittinen gray-koodattu lähetettä antava kytkin[7]. Kytkimen valinnassa vaikutti hinta sekä helppo liitettävyys. Potentiometri hylättiin sen vaatiman ADC-sisääntulon takia sekä sen rajoittuneen liikeradan takia, potentiometrin liikerata on rajoitettu maksimissaan kymmeneen kierrokseeen päästä päähän. Kuvassa 2 on esiteltynä sekä indikointiledein varustettu Mec:n painonappi että Vishayn kiertokytkin.



Kuva 2: Käyttökytkimet

Lambda-anturin antaman signaalin näytön indikointiin valittiin 500mCd kirkkaita ledejä, 3-perusväriä. Paikkaansa kirkkaat ledit puolustavat sillä, että näitä käyttämällä voidaan rajoittaa ledin ottama virta kohtuullisen pieneksi joten levylle ei tarvita erillisiä transistori-ajureita ajamaan ledejä. Tämä säästää tilaa, komponenttien määrä ja sekä rahaa. Laskennalliseksi kokonaisvirraksi ajateltiin kaikille ledeille kokoluokkaa 80 – 120mA, tämän virran mikrokontrolleri datalehden mukaan kestää ajaa itsensä läpi maahan. Ledien toiminnallinen ohjaus on siis maadoittava, esimerkiksi kun lähtörekisterissä on kaikki tilaa osoittavat bitit nollia, palaa aina kokonainen kahdeksan ledin rypäs.

LCD-näytöksi valittiin Probyte Oy:n maahantuoman 2x16-kokoinen yksikkö, tässä tyypissä taustavalo on sininen ja näkyvät kirjaimet ovat valkoisia. Tähän näyttöön päädyttiin sen erinomaisen kontrastin takia, eikä siististä ulkonäöstäkään suinkaan mitään haittaa ollut[8]. LCD:ssä on taustavalona yksittäinen kirkas ledi, eikä normaalia valaisumattoa, joten virransäästö on kertaluokkaa parempi valitun komponentin puolella. Ainoa haittapuoli löytyy siitä, ettei taustavalaistusta suurentamalla voi laajentaa näytön pakkaspuolen käyttöastetta paljoakaan. Näytöt tulevat suoraan ulkomailta tällähetkellä, hintaeroa on yli 75% suoratoimituksen puolesta.

Työn suoritus

Komponenttikirjastojen teko ja komponenttien esittelyä

Komponenttien valinnan jälkeen oli vuorossa tehdä EAGLE:n oma kirjasto komponenteille, jotta itse piirilevyn teko onnistuisi. Kirjaston teko onnistui helposti, tarkkana piti olla kuitenkin mitoituksen kanssa, oletusarvona EAGLE:ssa on mils-mitoitus kun taas datalehdissä on yleensä mm-mitoitus.

Komponentin piirtäminen aloitetaan tekemällä piirrosmerkkiä vastaava kytkentäkuvaus piirikaaviopuolelle. Tämän jälkeen piirretään vastaava piirikaaviopuolen komponenttikuvaus jossa on eriteltynä juotosjalat ja mahdolliset läpiviennit. Lisäksi piirikaaviopuolelle kannattaa piirtää valmistajan suositteleman kuvan mukaisesti ulkomittoja hahmottava kuvaelma sekä määritellä alueet, jonne ei saa piirilevyä tehtäessä laittaa juotteenestopinnoitetta.

Komponentteina on suurimmaksi osaksi pintaliitos-tyylisiä. Läpiladottavia komponentteja on ainoastaan piikkirimat, RS232-liitin, ledit sekä LCD:n kontrastinsäätö- ja taustavalon etuvastukset. Vastuksien kotelon koko on 1206. Koko on miellyttävä ensimmäiseen projektiin, komponentit on helppo juottaa itse kiinni ja levyn koko vain pienenee seuraavissa versioissa, joissa voi korvata vastukset 0805-koteloisilla.

Taustavalon kirkkaudensäädön hoitaa SOT23-koteloinen n-kanava mosfet, valmistaja Philips, tyyppi PHT6N06T, Rds(on) = 150mOhm, Ids(max) = 2,5A[9]. Komponentti kestää taustavalon ottaman mahdollisen isonkin syöksyvirran helposti, on helppo latoa käsin ja hintakin on halpa. Regulaattorina on SOT-223 koteloinen LM1117 800mA kuormaan ajava LDO-lineaariregulaattori[10]. Regulaattorissa on

myös sisäinen ylilämpösuojaus siltä varalta, että syötettävä laite menee oikosulkuun. Ledeinä on 500mCd kirkkauteen täydellä nimellisvirralla yltäviä superkirkkaita 3mm halkaisijaltaan kirkasrunkoisia vihreitä, keltaisia ja punaisia ledejä. Painonappeina on Mecin taustavalaistut palautuvat painonapit, joissa kärkitoiminto on sulkeutuva. Painonapit maadoittavat mikro-ohjaimen sisääntulonastan ja aiheuttavat ohjelmakiertoon keskeytyksen. Jännitevakavointiin on varattu levylle tilaa suodatuskondensaattorille ennen ja jälkeen lineaariregulaattorin, kondensaattorin tyyppinä on tantaali, jännitekestona 25V ja kapasitanssina 4,7µF.

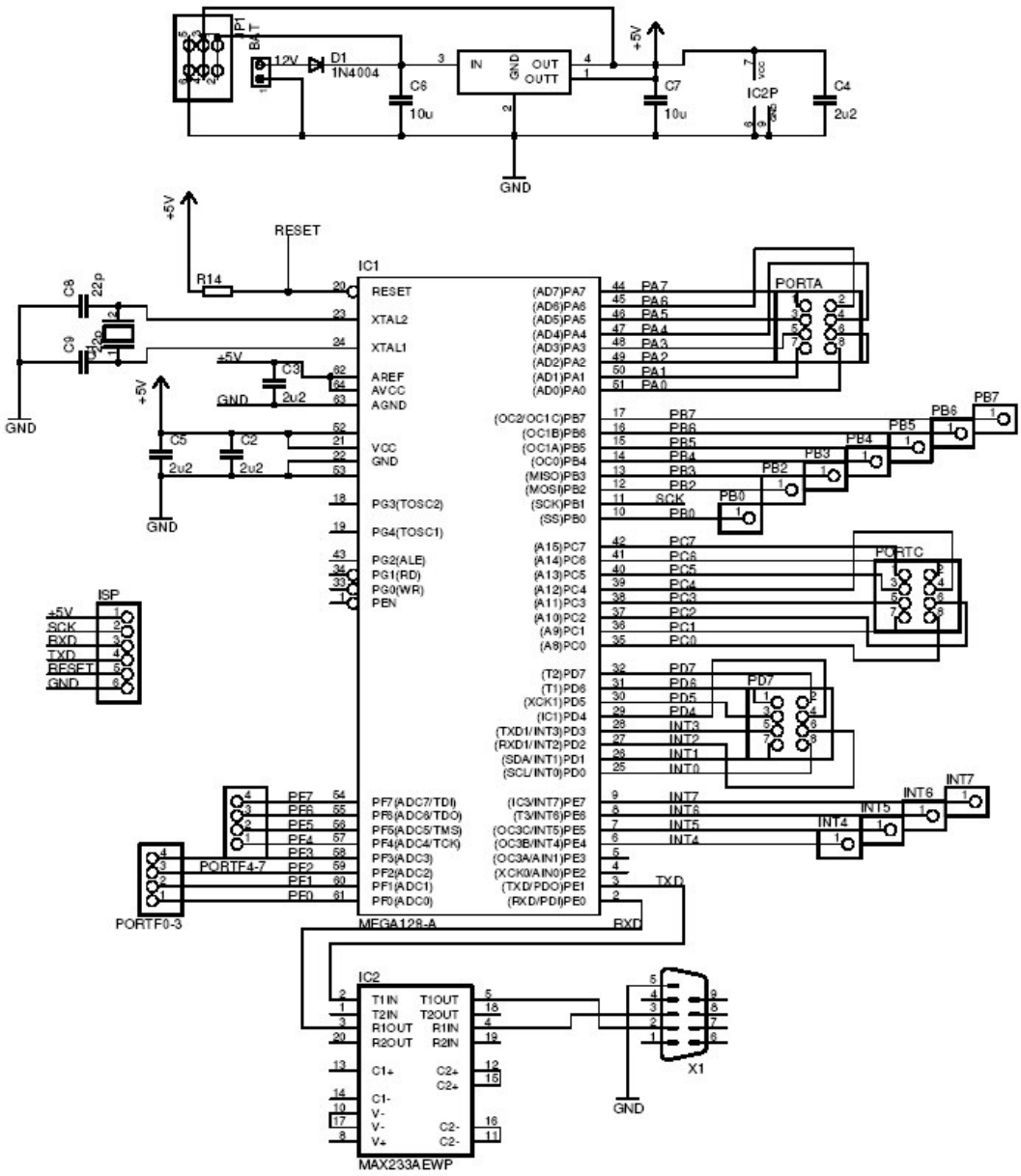
Piirilevyn piirto

Piirilevyn suunnittelu lähti liikkeelle mittamalla valitun kotelon sisämitat. Seuraavaksi edettiin mittaamalla sisään mahtuvan piirilevyn maksimimitat ja siirtämällä ne EAGLEen. Piirilevysuunnittelu-puolella ensiksi määriteltiin levyn koko siirtämällä levyn reunat rajaavaa kerrosta, tämän jälkeen oli aika siirtyä piirtämään puuttuvia komponentteja omaan lisäkirjastoon. Oman kirjaston tekeminen oli helppoa, sillä katsomalla tarkasti valmistajien datalehtiä saa siirrettyä komponenttien piirilevykuvan sekä toiminnallisen kuvauksen kirjastoon helposti[8][9][10].

Ensimmäinen vedos piirilevystä

Aluksi lähdettiin suunnittelemaan piirilevyä sillä ajatuksella että tuotteessa on erillinen CPU/RS232-tasomuunninlevy, jossa on vain piikkirimat erilliselle ulkoiselle I/O-levylle. Tämä sallisi erityyppisten näyttö ja kytkinkonfiguraatioiden käyttämisen samalla peruslevyllä. Levyssä itsessään olisi vain jänniteregulaattori, mikrokontrolleri sekä tasomuunnin sarjaliikenteelle.

Käyttäjää lähempänä olevan levyn, I/O-levyn, piirtämisen aloittaminen alkoi kopiomalla alemman levyn koko sekä piikkirimojen paikka toiselle levylle, näin menettelemällä levyjen sekä piikkirimojen kohdistus pysyy molemmissa samoina vaikka mahdollisesti välillä piirto-ohjelmassa menisi muuttamaan levyjen piirrossa käytettävää ruudukkokokoa. Kuvassa 3 on esiteltynä ensimmäisen piirilevyvedoksen CPU-kerrosta kuvaava kytkentä, melkein kaikki lähtö- ja tuloportit on kytketty kytkentärimoille.



Kuva 3: Ensimmäisen vedoksen piirikaaviokuva

Ensimmäisen levyn piirrettyämme ja syövytettyämme havaitsimme olleemme hieman hakoteillä tämän ratkaisun kalleuden sekä vikaherkkyyden takia : piikkirimat maksavat yllättävän paljon ja ovat tärinässä sekä ulkokäytössä kohtuuttoman vikaherkkiä. Piikkirimojen kohdistus kyllä onnistui hyvin, mutta edellä mainitut seikat tekivät selväksi sen, että levyn suunnittelu pitää tehdä uusiksi ja saada yhdelle levylle kaikki toimintaan liittyvät komponentit.

Toinen vedos piirilevystä

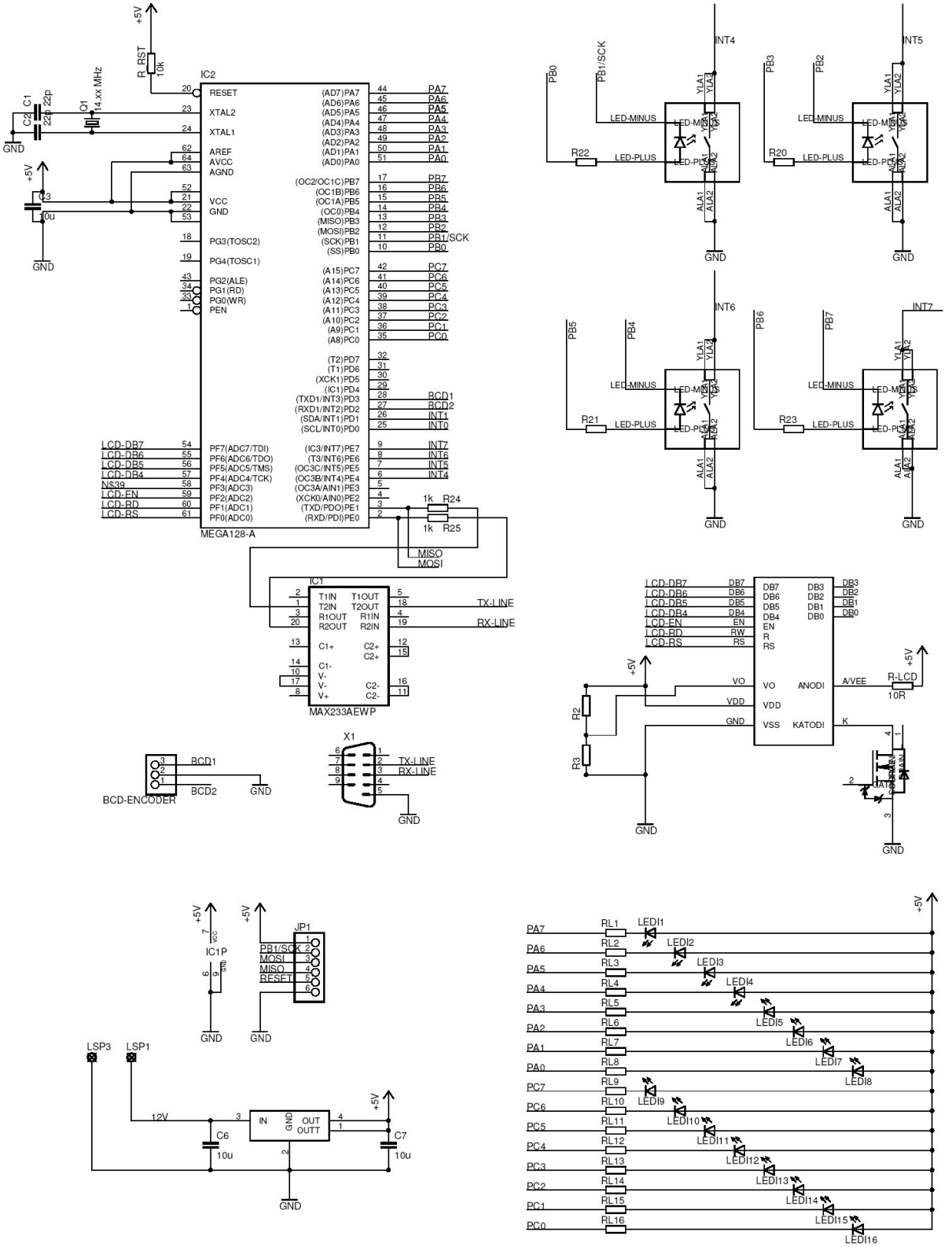
Toisen levyn piirtäminen onnistui paljon helpommin, suunnittelun tukityökaluna on ollut mahdollisuus käyttää kahta monitoria piirilevyn ja piirikaavion piirtämiseen, ja käytännössä havaitsimme tämän ratkaisun nopeuttavan radikaalisti itse kuparivetojen vetoa sekä helpottavan komponenttien sijoittelua paremmin. Suunnittelemalla näin saa aktiiviseksi esimerkiksi piirilevypuolella valitun komponentin

piirikaaviopuolelle ja toisinpäin.

Nämä toisen version piirilevyt tilasimme suoraan ulkomailta kokeilematta itse syövyttää edes protoversioita. Samaa piirikaavion pohjaa käyttäen pystyimme korjaamaan pari ajatusvirhettä suoraan ennen levykuvien toimittamista piirilevy-yritykseen. Ajatusvirheenä oli ensimmäisissä kaksikerroslevyissä se, että ledien sarjavastukset sijaitsivat liian lähellä piirilevyjen liittimiä, tehden laitteen kasauksen erittäin vaikeaksi.

Piirilevytehtaaksi valittiin Lontoossa Euroopan myyntiään pitävä Futurlec Ltd. Piirilevyjen suunnittelun alussa piti käydä valmistajan kotisivuilla tarkistamassa, mikä on heidän tuotantoprosessin johtimien välisten vetojen, reikien kokojen ja johtimien leveyksien minimit. Normaaliprosessilla he pääsevät 10mils vedon tarkkuuteen ja vedon minimileveyteen, reiän minimikoko on 0,6mm[11].

Muutoksina ensimmäiseen levyyn nähden oli osan mikrokontrollerin johtimien siirto piirilevyn vetoa ajatellen järkevämpään paikkaan. Näin ei tarvitsisi tehdä turhia alituksia eikä ylityksiä johtimilla, vaan johdinvedot olisivat selkeitä sekä siistejä. Rajaehtoina oli vain rautapohjaisen PWM-lähdön kytkeminen LCD:n taustavalon ohjaukseen sekä toimivan LCD-ohjainportin käyttäminen. Kuvassa 4 on esiteltynä koko piirikaaviokuvaus, erona näkyy porteilta suoraan lähtevät kytkennät painonapeille ja ledeille, sekä kuvasta voi erottaa itse piirretyn LCD-näyttöä kuvaavan komponentin.



Kuva 4: Toisen version piirikaaviokuva

Toiseen versioon jäi kuitenkin vielä pieni ajatusvirhe, sillä alunperin ajattelimme laittaa piirilevylle RS232-liittimen ja vetää kotelon kylkeen erillisen virtapistokkeen. Ajatuksesta kuitenkin luovuttiin juteltuamme työnantajan kanssa, joka ehdotti, että tuodaan laitteelle suoraan yksittäinen johto, jonka perässä on erillinen sarjaporttiliitin ja siitä samasta liittimestä josta haarautuu erillinen virtaliitin. Näin ei tarvitse

koteloon aukottaa koteloon kuin yksi läpivientireikä. Kuten kuvassa 3 esiteltiin, piirilevylle piirrettiin valmiiksi RS232-liittimelle paikka, joten piirilevypuolella on turhaa tilaa viemässä liittimen ulkomittoja vastaavan kokoinen alue, johon liitytään suoralla johtimella.

Piirilevyn kasaus ja testaus

Ongelmia havaitsimme myös aloittaessamme levyn kasausta : piirilevyn suunnittelussa tehty oma komponenttikirjasto ei ollut täysin kunnollinen, MEC:n kytkimien reunajohtimille oli kirjastossa ikävä kyllä liian pienet jalat sekä mikrokontrollerin syövytyskuva johtimille oli liian kapea sekä liian sisällä[6]. Ongelmat ratkaisimme siten että porasimme uusista levyistä reunajohtimet kytkinten kohdalta 0,2mm isommaksi sekä juotimme mikrokontrollerin käyttäen fluksia ja isoa tinapalloa, matkien siis infrapunauunia ja tinapastaa. Ledien saaminen oikeaan korkeuteen on kanssa aavistuksen verran vaikeaa, käytännössä puuhassa pitää olla jokin korkeuden määrittävä asennustuki joka pitää ledin suorassa kulmassa levyyn nähden sekä oikeassa korkeudessa.

Ohjelmiston kehitys

Ensimmäinen versio laitteen ohjelmakoodista sisälsi I/O-pinnien toiminnallisen määrittelyn, laskureiden määrittelyn sekä LCD-portin ohjauksen. Ohjelma yksinkertaisesti tulostaa LCD:lle tekstin, vilkuttaa indikointi-ledejä sekä vaihtaa painonappien ledien väriä, kun nappia painetaan. Levyn ensimmäisessä versiossa, kaksikerrosmallissa, itsesyövytetyn levyn kanssa oli vaikeuksia saada laitteiston ohjelmaa ladattua sisään. Vika korjaantui, kun juotimme kaikki piikkirimat molemmilta puolilta levyä, itse syövyttämällä kun ei saa tehtyä sähkön johtumiselle tärkeitä läpikuparointeja.

Ohjelmiston seuraava kehitys oli tehdä pelkkä näyttöyksikön ohjaus ohjainlaitteen kanssa, näyttönä oli väliaikaisesti 4x20 kokoinen LCD-näyttö, jossa oli osa tulevaan tuotteeseen tulevista auton moottorin parametreista. Tämä toiminnallinen testaus tehtiin tuotteen 2-piirilevyllisellä versiolla. Samalla tuotetta päästiin testaamaan todellisessa käyttöympäristössä.

Ohjelman tekemisessä tuli vastaan ongelmia LCD-portin ohjauksen kanssa. Codevision avr-ohjelmointiympäristön mukana tulleiden LCD.h kirjaston komennot tekevät muisti/rekisterialueelle osoituksen epäsuorasti. Lähtöportti F, missä LCD sijaitsee, sijaitsee osittain rekisterialueella ja osittain sram-alueella, mikä aiheutti toimamattomuuden alkuperäisen LCD-kirjaston kanssa. Ongelma korjaantui tutkimalla kirjastoa, opettelemalla AVR:n assembler-käskyjä ja muuttamalla epäsuorat osoitukset suoriksi osoituksiksi[4].

Laitteistoläheisessä koodissa on ennen main-funktiota mikrokontrolleri alustettu porttien toimintasuuntien , keskeytystenkohteiden , rautapohjaisen sarjaportin nopeus- sekä laskurien määrittelykomennoilla. Ohjelmallisesti toimintojen määrittely onnistuu kirjoittamalla suoraan rekisteriin haluttu bittikuvio. Laskureiden ja keskeytysten määrittely onnistuu samalla lailla, erona on ulkoisen kesketysfunktion

kirjoittaminen erikseen pääohjelmasta.
Ohjelmiston perusrakenteessa pitäisi painonapeilla pystyä ohjaamaan seuraavia arvoja tai muutoksia :

- Vaihtamaan näytössä lukevia arvoja
- Säätämään LCD:n taustavalon kirkkaus (muuttuja sijoitettuna EEPROM:n)
- Määräämään hälytysrajat tietyille anturitulojen arvoille
- Määräämään laitteistoläheisen ohjelmakoodin päivitysmoodiin

Ohjelman päälle rupesimme pikkuhiljaa rakentamaan toiminnallisia lisäyksiä, mm CRC-tarkistus tuleville biteille vastaavanlaisella tulokartastolla kuin lähtöpäässäkin on, tulevien sanojen muuttaminen vastaavaan näytölle kelpaavaan muotoon, jännitetulojen skaalaus sekä MAP-anturin lukema-palkin päivitys.

Ohjelmistoon pitäisi vielä lisätä Bootloader-osio, tämä sijaitsee ohjelmamuistin aloitusvektorissa, jossa voidaan tehdä laitteistoläheisen ohjelman päivitys ilman ISP-ohjelmointia suoraan RS232-liitynnän ylitse. Ohjelmoinnin aktivointimoodiin voidaan mennä esimerkiksi siten että pidetään tiettyä painonappia tai yhdistelmää pohjassa samalla kun laite käynnistyy.

Kuvassa 5 on laite testiohjelmalla käynnissä, näyttöön tulostetaan ensimmäiselle riville : Lopputyö ja toiselle riville : Repsa Remote. Samassa testiohjelmassa on ledien poltto sekä näppäinledien värien vaihto keskeytyspohjaisesti.

Kuva 5: Testimoodissa oleva kaukosäädin

Indikointiledien virranmitoitus

Työssä olevat ledit ovat superkirkkaita, tarkoituksena oli ajaa ledien läpi alle nimellisen kantavirran verran virtaa, mikrokontrollerin sanelemien ehtojen takia. Ledille sopivaan kantavirtaan päästiin mittaamalla etuvastuksen kanssa eri värisiä ledejä rinnakkain siten että kaikilla on sama valon intensiteetti. Allaolevassa laskuissa on ledin ylitse oleva mitattu jännite sekä ledin todellinen virta.
Ledien virran- ja jännitteenmittaukset on tehty Fluke 123 digitaalisella muistioskilloskoopilla. Alla on ote Mathcad-ohjelmalla lasketuista etuvastuksien kokojen esimäärittelystä sekä toteutuneiden arvojen listaus.

$Vcc := 5V$ $Ukelt := 2.068V$ $Ikelt := 8.55mA$

$Uvihr := 2.069V$ $Ivihr := 11.22mA$

$Upun := 1.749V$ $Ipun := 11.05mA$

$Upain\_pun := 1.878V$ $Ipain\_pun := 11.12mA$

$Upain\_vihr := 2.033V$ $Ipain\_vihr := 11mA$

$$Etuvastus\_pun\_led := \frac{Vcc - Upun}{Ipun} \qquad Etuvastus\_pun\_led = 294.208\,\Omega$$

$$Etuvastus\_kelt\_led := \frac{Vcc - Ukelt}{Ikelt} \qquad Etuvastus\_kelt\_led = 342.924\,\Omega$$

$$Etuvastus\_vihr\_led := \frac{Vcc - Uvihr}{Ivihr} \qquad Etuvastus\_vihr\_led = 261.23\,\Omega$$

$$Etuvastus\_vihr\_painike := \frac{Vcc - Upain\_vihr}{Ipain\_vihr} \qquad Etuvastus\_pun\_painike := \frac{Vcc - Upain\_pun}{Ipain\_pun}$$

$Etuvastus\_vihr\_painike = 269.727\,\Omega$ $Etuvastus\_pun\_painike = 280.755\,\Omega$

Valitut arvot :
punaiselle ja keltaiselle ledille 330Ω etuvastus,
vihreälle 220Ω etuvastus, valon intensiteetti on näin
valituilla komponenteilla samanlainen.

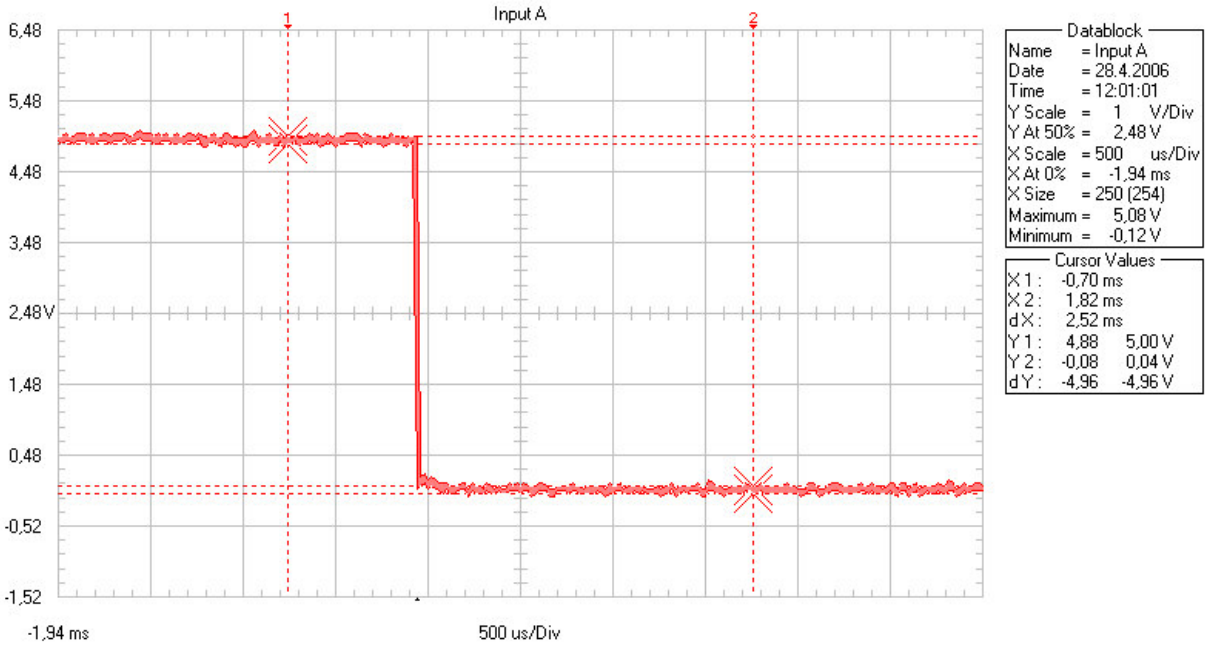
Painonapin ledin etuvastus saa olla kanssa 220Ω, indikoinnin
pitää kuitenkin näkyä päivänvalossakin.

Mitatut ledien virrat valituilla komponenteilla :
painonappi:
- punainen : 13,7mA
- vihreä : 13,0mA

3mm ledi :
- punainen : 9,6mA
- vihreä : 12,8mA
- keltainen : 8,6mA

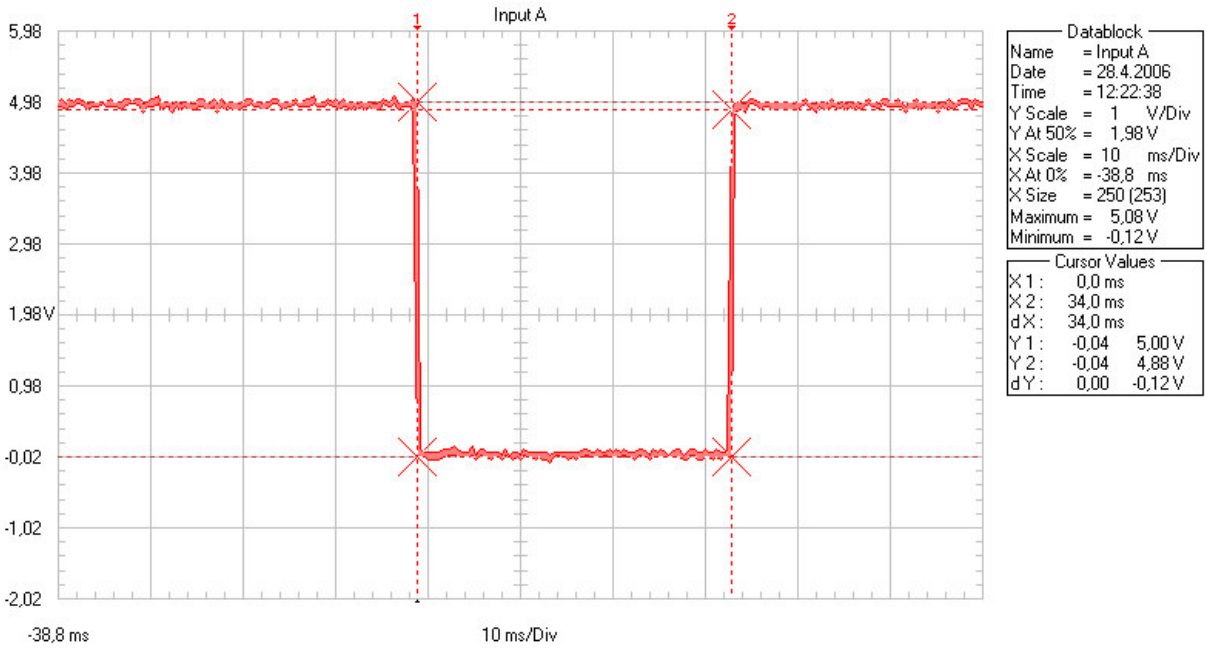
Painikkeiden mittaus

Työssä mitattiin myös painonappien kytkinvärähtely, joskus painonappia painaessa indikointiledi vaihtaa värinsä mutta palaa samaan lähtöväriin, esimerkiksi punaisesta käydään vihreässä mutta palataan saman tien punaiseen, vaikka nappia olisi painettu vain kerran. Kuvassa 6 on esitelty painonapin painamisen jälkeen oskilloskoopilla mitattu kuva, oskilloskooppi on liipaistu laskevaan reunaan.



Kuva 6: Painonapin laskeva reuna

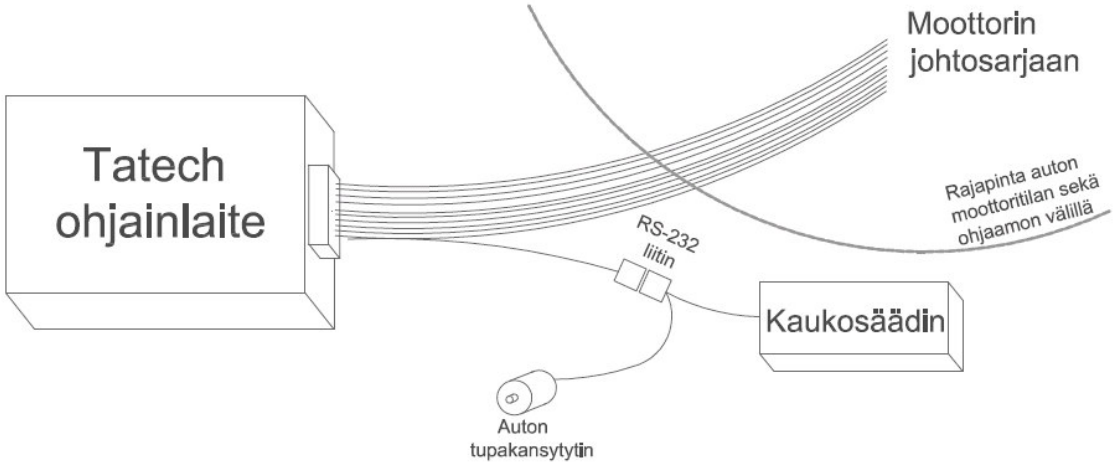
Keskimääräinen painonapin värähtelyaika on 8ms luokkaa, kuitenkin mitattiin myös kokonaiskytkentäaika, jolloin ei havaittu useista toistoista huolimatta minkäänlaista kytkinvärähtelyä. Kuvassa 7 on esitelty 34ms kestoinen napin painaminen.



Kuva 7: Painonapin aktiivinen aika

Käyttäjän saamat kytkentäohjeet

Kuvassa 8 on esitelty laitteen loppukäyttäjän saamat kytkentäohjeet laitteiston käyttöönottoon. Tuotteen liittäminen olemassaolevaan ohjainlaitteeseen on tehty mahdollisimman helpoksi, toimintaan saattaminen ei vaadi mitään muuta kuin kahden liittimen paikalleen laittamisen.



Kuva 8: Tatech-ohjaimeen liittyminen

Laitteisto liittyy olemassaolevaan ohjainjärjestelmään RS-232 liittimen kautta, samasta liittimestä haarautuu auton tupakansytyttimeen sopiva liitin, josta kaukosäädin saa käyttösähkönsä. Laitteisto aktivoituu välittömästi kun pistokkeeseen tulee jännite ja laitteisto jää odottamaan perustilaan jos yhteyttä moottorinohjeusjärjestelmään ei saada muodostettua.

Latausohjelmaosio jäi toteutumatta tässä vaiheessa suunnitelmaa, tarkoitus on tehdä toimiva latausohjelma näytön seuraavaan versioon. Latausohjelmaa varten joutuu teetättämään liittimen millä saadaan sarjaporttiliitin liitettyä ohjelmoivaan tietokoneeseen, naaras-naaras-tyylinen.

Yhteenveto ja parannusehdotukset

Laitteen suunnittelu eteni normaalin elektroniikkaprojektin tapaan, ensimmäisenä päätettiin reunaehdot tuotteelle, niin sisäiset kuin ulkoisetkin. Seuraavaksi tehtiin komponenttikierros, tässä valittiin tuotteeseen tulevat komponentit, kriteereinä hinta, saatavuus sekä laatu. Ohjelmiston kehitys alkoi I/O-nastojen sekä laitteiden testausosiolla, ohjelmiston kehityksessä, kuten raudan kehityksessä, tuli vastaan ongelmia suunnitelmissa. Ongelmat eivät olleet ylitsepääsemättömiä, mutta hidastivat projektin läpivientiä suhteettoman paljon.

Laitetta on käytetty ajoneuvoympäristössä ja puutteita on havaittu kotelon ESD-suojauksen suhteen, kotelo johtaa staattiset sähköpurkaukset sarjaportin kaapelia pitkin suoraan piirilevylle. Tästä voi tulla pidemmässä ajanjaksossa laitteiston vaurioitumisen aiheuttava haitta. Laitteiston käyttö mittarina on helppoa, kartastojen säätö valintakytkimellä tosin ei ole kovin mielekästä, ihmisen reagointi nopeisiin muutoksiin moottoria säätäessä on kohtuutoman hidasta. Riski että säätää vahingossa väärää kartastoa on erittäin suuri.

Seuraava versio laitteistosta ei sisällä kiertokytkintä, pelkästään painonappeja näyttömoodin vaihtamiseen. Näytön kokokin on rajoittunut, ei mahdu tarpeeksi informaatiota moottorin tilasta 2-riviselle LCD-näytölle, näyttö vaihtunee 4x20 kokoiseen tai graafiseen LCD-näyttöön. Graafisessa LCD-näytössä voi simuloida esimerkiksi auton analogisia mittareita lämpötilan tai imusarjan paineen näyttöön. Näytön pakkaskestävyyttä pitää parantaa, huonona puolena pakkaskestävyydessä on negatiivisen kontrastinsäätöjännitteen vaatimus näytölle.

Latausohjelman tunnistamiseen voisi tehdä sarjaporttitunnistimen, kytketään sarjaportin tasomuuntimesta normaalisti sarjaportissa olevia tilatietolinjoja suoraan mikrokontrollerille. Latauksen suorittava ohjelmisto nostaisi sarjaporttiin lähdön päälle ja tästä näyttölaite tunnistaisi että vuorossa on laitteistoläheisen ohjelmistokoodin päivitys.

Kotelon tyyppi tulee varmasti muuttumaan metalliseen, staattinen sähkö johtuu alumiinisen tai rautaisen kotelon pinnasta paremmin kuin muovisesta kotelosta. Samalla voi lisätä maadoituksen EMC-mielessä kotelon päälle sekä kotelosta saa juuri omien tarpeiden mukaisen.

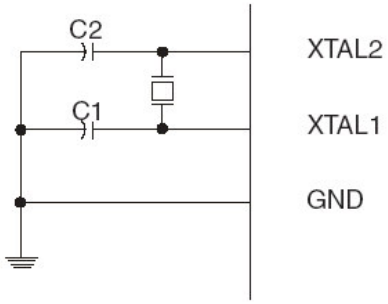
Joskus painonappeja käyttäessä värin vaihtuminen tapahtuu lähtöväristä lähtöväriin, kytkinvärähtely aiheuttaa ylimääräisen keskeytyksen ohjelmaan. Ohjelmaan täytynee lisätä ohjelmallinen suodatus joka estää ylimääräisen reunan näkymisen, aika joka voidaan suodattaa pois on 10-20ms luokkaa, kuvassa 7 on mitattuna kohtuullisen nopeasti painettuna aktiivinen aika, joka on kuitenkin jo 34ms.

Crystal Oscillator

XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an On-chip Oscillator, as shown in Figure 19. Either a quartz crystal or a ceramic resonator may be used. The CKOPT fuse selects between two different Oscillator Amplifier modes. When CKOPT is programmed, the Oscillator output will oscillate will a full rail-to-rail swing on the output. This mode is suitable when operating in a very noisy environment or when the output from XTAL2 drives a second clock buffer. This mode has a wide frequency range. When CKOPT is unprogrammed, the Oscillator has a smaller output swing. This reduces power consumption considerably. This mode has a limited frequency range and it can not be used to drive other clock buffers.

For resonators, the maximum frequency is 8 MHz with CKOPT unprogrammed and 16 MHz with CKOPT programmed. C1 and C2 should always be equal for both crystals and resonators. The optimal value of the capacitors depends on the crystal or resonator in use, the amount of stray capacitance, and the electromagnetic noise of the environment. Some initial guidelines for choosing capacitors for use with crystals are given in Table 8. For ceramic resonators, the capacitor values given by the manufacturer should be used.

Figure 19. Crystal Oscillator Connections



The Oscillator can operate in three different modes, each optimized for a specific frequency range. The operating mode is selected by the fuses CKSEL3..1 as shown in Table 8.

Table 8. Crystal Oscillator Operating Modes

CKOPT	CKSEL3..1	Frequency Range (MHz)	Recommended Range for Capacitors C1 and C2 for Use with Crystals
1	101[1]	0.4 - 0.9	–
1	110	0.9 - 3.0	12 pF - 22 pF
1	111	3.0 - 8.0	12 pF - 22 pF
0	101, 110, 111	1.0 -	12 pF - 22 pF

Note: 1. This option should not be used with crystals, only with ceramic resonators.

The CKSEL0 fuse together with the SUT1..0 fuses select the start-up times as shown in Table 9.

Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
($FF)	Reserved	–	–	–	–	–	–	–	–	
..	Reserved	–	–	–	–	–	–	–	–	
($9E)	Reserved	–	–	–	–	–	–	–	–	
($9D)	UCSR1C	–	UMSEL1	UPM11	UPM10	USBS1	UCSZ11	UCSZ10	UCPOL1	191
($9C)	UDR1	USART1 I/O Data Register								189
($9B)	UCSR1A	RXC1	TXC1	UDRE1	FE1	DOR1	UPE1	U2X1	MPCM1	189
($9A)	UCSR1B	RXCIE1	TXCIE1	UDRIE1	RXEN1	TXEN1	UCSZ12	RXB81	TXB81	190
($99)	UBRR1L	USART1 Baud Rate Register Low								193
($98)	UBRR1H	–	–	–	–	USART1 Baud Rate Register High				193
($97)	Reserved	–	–	–	–	–	–	–	–	
($96)	Reserved	–	–	–	–	–	–	–	–	
($95)	UCSR0C	–	UMSEL0	UPM01	UPM00	USBS0	UCSZ01	UCSZ00	UCPOL0	191
($94)	Reserved	–	–	–	–	–	–	–	–	
($93)	Reserved	–	–	–	–	–	–	–	–	
($92)	Reserved	–	–	–	–	–	–	–	–	
($91)	Reserved	–	–	–	–	–	–	–	–	
($90)	UBRR0H	–	–	–	–	USART0 Baud Rate Register High				193
($8F)	Reserved	–	–	–	–	–	–	–	–	
($8E)	Reserved	–	–	–	–	–	–	–	–	
($8D)	Reserved	–	–	–	–	–	–	–	–	
($8C)	TCCR3C	FOC3A	FOC3B	FOC3C	–	–	–	–	–	135
($8B)	TCCR3A	COM3A1	COM3A0	COM3B1	COM3B0	COM3C1	COM3C0	WGM31	WGM30	131
($8A)	TCCR3B	ICNC3	ICES3	–	WGM33	WGM32	CS32	CS31	CS30	134
($89)	TCNT3H	Timer/Counter3 – Counter Register High Byte								136
($88)	TCNT3L	Timer/Counter3 – Counter Register Low Byte								136
($87)	OCR3AH	Timer/Counter3 – Output Compare Register A High Byte								136
($86)	OCR3AL	Timer/Counter3 – Output Compare Register A Low Byte								136
($85)	OCR3BH	Timer/Counter3 – Output Compare Register B High Byte								137
($84)	OCR3BL	Timer/Counter3 – Output Compare Register B Low Byte								137
($83)	OCR3CH	Timer/Counter3 – Output Compare Register C High Byte								137
($82)	OCR3CL	Timer/Counter3 – Output Compare Register C Low Byte								137
($81)	ICR3H	Timer/Counter3 – Input Capture Register High Byte								137
($80)	ICR3L	Timer/Counter3 – Input Capture Register Low Byte								137
($7F)	Reserved	–	–	–	–	–	–	–	–	
($7E)	Reserved	–	–	–	–	–	–	–	–	
($7D)	ETIMSK	–	–	TICIE3	OCIE3A	OCIE3B	TOIE3	OCIE3C	OCIE1C	138
($7C)	ETIFR	–	–	ICF3	OCF3A	OCF3B	TOV3	OCF3C	OCF1C	139
($7B)	Reserved	–	–	–	–	–	–	–	–	
($7A)	TCCR1C	FOC1A	FOC1B	FOC1C	–	–	–	–	–	135
($79)	OCR1CH	Timer/Counter1 – Output Compare Register C High Byte								136
($78)	OCR1CL	Timer/Counter1 – Output Compare Register C Low Byte								136
($77)	Reserved	–	–	–	–	–	–	–	–	
($76)	Reserved	–	–	–	–	–	–	–	–	
($75)	Reserved	–	–	–	–	–	–	–	–	
($74)	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE	206
($73)	TWDR	Two-wire Serial Interface Data Register								208
($72)	TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE	208
($71)	TWSR	TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0	207
($70)	TWBR	Two-wire Serial Interface Bit Rate Register								206
($6F)	OSCCAL	Oscillator Calibration Register								39
($6E)	Reserved	–	–	–	–	–	–	–	–	
($6D)	XMCRA	–	SRL2	SRL1	SRL0	SRW01	SRW00	SRW11		29
($6C)	XMCRB	XMBK	–	–	–	–	XMM2	XMM1	XMM0	31
($6B)	Reserved	–	–	–	–	–	–	–	–	
($6A)	EICRA	ISC31	ISC30	ISC21	ISC20	ISC11	ISC10	ISC01	ISC00	87
($69)	Reserved	–	–	–	–	–	–	–	–	
($68)	SPMCSR	SPMIE	RWWSB	–	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN	279
($67)	Reserved	–	–	–	–	–	–	–	–	
($66)	Reserved	–	–	–	–	–	–	–	–	
($65)	PORTG	–	–	–	PORTG4	PORTG3	PORTG2	PORTG1	PORTG0	86
($64)	DDRG	–	–	–	DDG4	DDG3	DDG2	DDG1	DDG0	86
($63)	PING	–	–	–	PING4	PING3	PING2	PING1	PING0	86
($62)	PORTF	PORTF7	PORTF6	PORTF5	PORTF4	PORTF3	PORTF2	PORTF1	PORTF0	85

Register Summary (Continued)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
($61)	DDRF	DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0	86
($60)	Reserved	–	–	–	–	–	–	–	–	
$3F ($5F)	SREG	I	T	H	S	V	N	Z	C	9
$3E ($5E)	SPH	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	12
$3D ($5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	12
$3C ($5C)	XDIV	XDIVEN	XDIV6	XDIV5	XDIV4	XDIV3	XDIV2	XDIV1	XDIV0	41
$3B ($5B)	RAMPZ	–	–	–	–	–	–	–	RAMPZ0	12
$3A ($5A)	EICRB	ISC71	ISC70	ISC61	ISC60	ISC51	ISC50	ISC41	ISC40	88
$39 ($59)	EIMSK	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0	89
$38 ($58)	EIFR	INTF7	INTF6	INTF5	INTF4	INTF3	INTF	INTF1	INTF0	89
$37 ($57)	TIMSK	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	106, 138, 158
$36 ($56)	TIFR	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	106, 139, 158
$35 ($55)	MCUCR	SRE	SRW10	SE	SM1	SM0	SM2	IVSEL	IVCE	29, 42, 61
$34 ($54)	MCUCSR	JTD	–	–	JTRF	WDRF	BORF	EXTRF	PORF	51, 256
$33 ($53)	TCCR0	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	101
$32 ($52)	TCNT0	Timer/Counter0 (8 Bit)								103
$31 ($51)	OCR0	Timer/Counter0 Output Compare Register								103
$30 ($50)	ASSR	–	–	–	–	AS0	TCN0UB	OCR0UB	TCR0UB	104
$2F ($4F)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	COM1C1	COM1C0	WGM11	WGM10	131
$2E ($4E)	TCCR1B	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	134
$2D ($4D)	TCNT1H	Timer/Counter1 – Counter Register High Byte								136
$2C ($4C)	TCNT1L	Timer/Counter1 – Counter Register Low Byte								136
$2B ($4B)	OCR1AH	Timer/Counter1 – Output Compare Register A High Byte								136
$2A ($4A)	OCR1AL	Timer/Counter1 – Output Compare Register A Low Byte								136
$29 ($49)	OCR1BH	Timer/Counter1 – Output Compare Register B High Byte								136
$28 ($48)	OCR1BL	Timer/Counter1 – Output Compare Register B Low Byte								136
$27 ($47)	ICR1H	Timer/Counter1 – Input Capture Register High Byte								137
$26 ($46)	ICR1L	Timer/Counter1 – Input Capture Register Low Byte								137
$25 ($45)	TCCR2	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	156
$24 ($44)	TCNT2	Timer/Counter2 (8 Bit)								158
$23 ($43)	OCR2	Timer/Counter2 Output Compare Register								158
$22 ($42)	OCDR	IDRD/OCDR7	OCDR6	OCDR5	OCDR4	OCDR3	OCDR2	OCDR1	OCDR0	253
$21 ($41)	WDTCR	–	–	–	WDCE	WDE	WDP2	WDP1	WDP0	53
$20 ($40)	SFIOR	TSM	–	–	–	ACME	PUD	PSR0	PSR321	70, 107, 143, 228
$1F ($3F)	EEARH	–	–	–	–	EEPROM Address Register High				19
$1E ($3E)	EEARL	EEPROM Address Register Low Byte								19
$1D ($3D)	EEDR	EEPROM Data Register								20
$1C ($3C)	EECR	–	–	–	–	EERIE	EEMWE	EEWE	EERE	20
$1B ($3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	84
$1A ($3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	84
$19 ($39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	84
$18 ($38)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	84
$17 ($37)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	84
$16 ($36)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	84
$15 ($35)	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	84
$14 ($34)	DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	84
$13 ($33)	PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	85
$12 ($32)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	85
$11 ($31)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	85
$10 ($30)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	85
$0F ($2F)	SPDR	SPI Data Register								168
$0E ($2E)	SPSR	SPIF	WCOL	–	–	–	–	–	SPI2X	168
$0D ($2D)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	166
$0C ($2C)	UDR0	USART0 I/O Data Register								189
$0B ($2B)	UCSR0A	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0	189
$0A ($2A)	UCSR0B	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	UCSZ02	RXB80	TXB80	190
$09 ($29)	UBRR0L	USART0 Baud Rate Register Low								193
$08 ($28)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	228
$07 ($27)	ADMUX	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	244
$06 ($26)	ADCSRA	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0	245
$05 ($25)	ADCH	ADC Data Register High Byte								246
$04 ($24)	ADCL	ADC Data Register Low byte								246
$03 ($23)	PORTE	PORTE7	PORTE6	PORTE5	PORTE4	PORTE3	PORTE2	PORTE1	PORTE0	85
$02 ($22)	DDRE	DDE7	DDE6	DDE5	DDE4	DDE3	DDE2	DDE1	DDE0	85

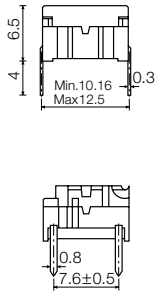
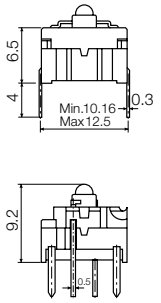
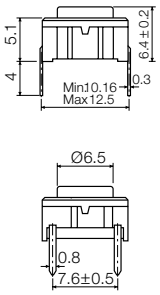
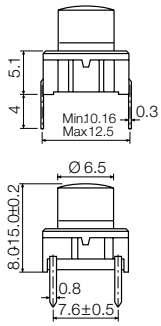
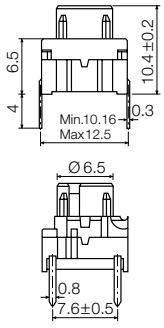
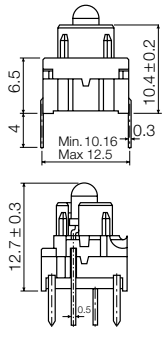
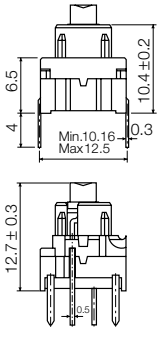
Register Summary (Continued)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
$01 ($21)	PINE	PINE7	PINE6	PINE5	PINE4	PINE3	PINE2	PINE1	PINE0	85
$00 ($20)	PINF	PINF7	PINF6	PINF5	PINF4	PINF3	PINF2	PINF1	PINF0	86

Notes:
1. For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.
2. Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers $00 to $1F only.

multimec® basic switch modules

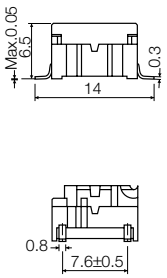
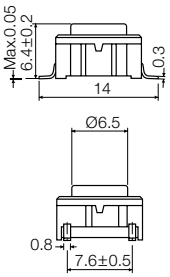
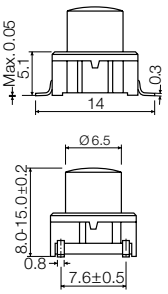
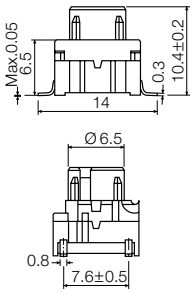
Through-hole Version

3A		3C	3E	3F		
For 1A/1B/1M	w/LED for 1C/1H			For 1D/1K/1N/1P/1S/1T/ 1U/1V/1X/1GA/1GC	w/LED for 1E/1F/1N/1Q/1R/1S/1X	w/LED for 1K1116/1T/1U/1V
						
Standard Versions						
3ATL6 3ATH9	3ATL600/20/40/80	3CTL6/3CTL9 3CTH9	3ETL9-H* 3ETH9-H*	3FTL6 3FTH9	3FTL600/20/40/80/ 2040/8020/8040	3FTL623/44/88

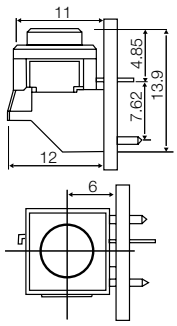
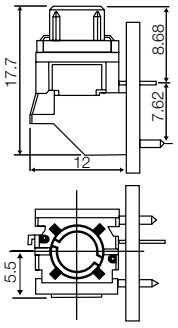
* 3E available in 6 standard heights: 08.0, 09.5, 10.4, 11.0, 12.0, 15.0 mm
Other heights between 08.0 and 15.0 mm are available upon request

Specials: Gold contacts, quiet version and actuation force other than 3.0N

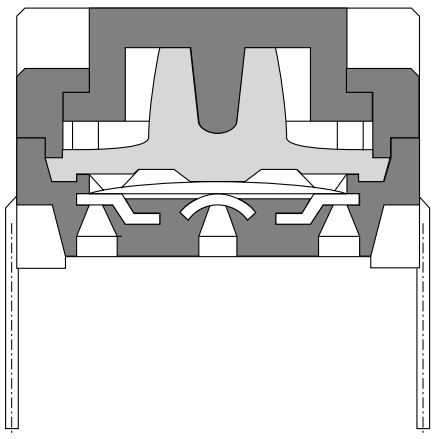
Surface Mount Versions

3A	3C	3E	3F
			
3ASH9/3ASH9R	3CSH9/3CSH9R	3ESH9/3ESH9R	3FSH9 / 3FSH9R
For 1A/1B/1M			For 1D/1K/1N/1P/1S/1T/ 1U/1V/1X/1GA/1GC

Right Angle Versions

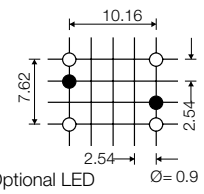
3C	3F
	
3CTL6RAS	3FTL6RAS
	For 1D/1K/1P/1S

multimec® Cross Section



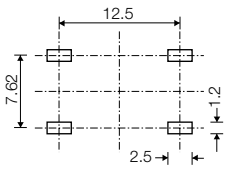
Recommended PCB layout

MULTIMEC
top view

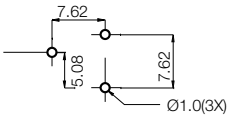


● Optional LED

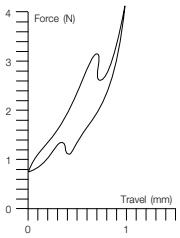
MULTIMEC
Surface-mount solder pad



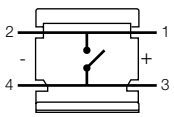
MULTIMEC
Right Angle Version
top view



Operating Force
(Typical example)



Circuit diagram



DIMENSIONS (mm) Unless otherwise specified, all tolerances ±0.2

mec Industriparken 23 · P.o. Box 26 · DK-2750 Ballerup · Denmark
Tel.: +45 44 97 33 66 · Fax: +45 44 68 15 14 · E-mail: danmec@mec.dk · www.mec.dk.

mec a/s may 2003

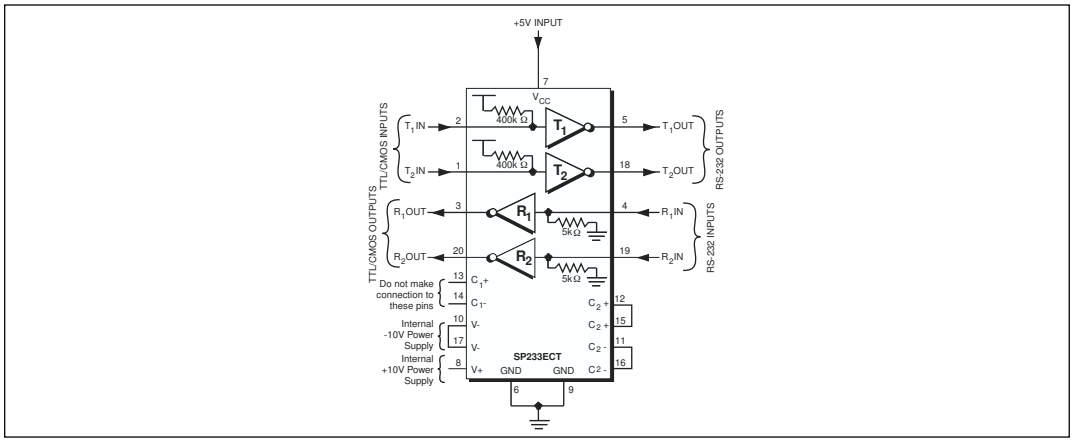


Figure 2. Typical Circuits using the SP233ECP and SP233ECT

The instantaneous slew rate of the transmitter output is internally limited to a maximum of 30V/µs in order to meet the standards [EIA RS-232-D 2.1.7, Paragraph (5)]. However, the transition region slew rate of these enhanced products is typically 10V/µs. The smooth transition of the loaded output from V_{OL} to V_{OH} clearly meets the monotonicity requirements of the standard [EIA RS-232-D 2.1.7, Paragraphs (1) & (2)].

Receivers

The receivers convert RS-232 input signals to inverted TTL signals. Since the input is usually from a transmission line, where long cable lengths and system interference can degrade the signal, the inputs have a typical hysteresis margin of 500mV. This ensures that the receiver is virtually immune to noisy transmission lines.

The input thresholds are 0.8V minimum and 2.4V maximum, again well within the ±3V RS-232 requirements. The receiver inputs are also protected against voltages up to ±15V. Should an input be left unconnected, a 5KOhm pulldown resistor to ground will commit the output of the receiver to a high state.

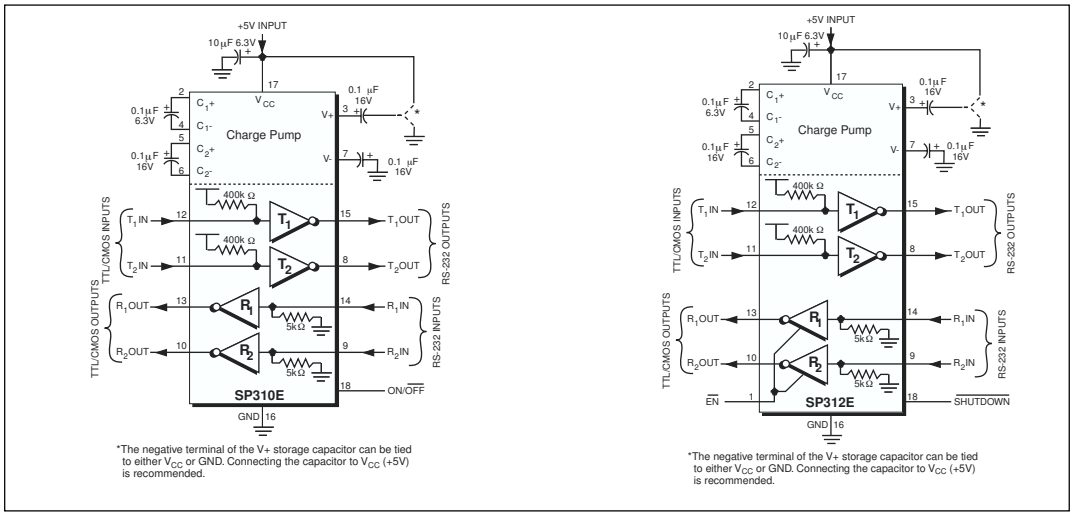


Figure 3. Typical Circuits using the SP310E and SP312E


```
/*****************************************************

Projekti: Lopputyö
Versio  : 1
Päivä   : 25.3.2006
Tekijä  : Marko Reponen
Koodi on CRC-tarkistusta lukuunottamatta täysin toimivaa, 4x20 testinäytön versio, DEFINE-komennolla voidaan määritellä
painonapeille toimintoja.

*****************************************************/

#include <mega128.h>
#include <io.h>
#include <delay.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define testi

#define LCD_BL OCR3AL
#define START_BLOCK_LENGTH              1
#define LEN_BLOCK_LENGTH                1
#define ID_BLOCK_LENGTH                 2
#define CRC_BLOCK_LENGTH                2
#define MSG_START_BYTE                  0xff;
#define BUFFER_EMPTY    0
#define BUFFER_STARTED  1
#define BUFFER_FULL     2

typedef unsigned char byte;

/* table for the user defined character lambda */
     flash byte char0[8]={
     0b1111111,
     0b1111111,
     0b1111111,
     0b1111111,
     0b1111111,
     0b1111111,
     0b1111111,
     0b1111111};

//-----------------------------------------------------------------------------------------------------------------------


#asm
   ; .equ __lcd_direction=0x61
     .equ __lcd_pin=0
     .equ __lcd_rs=0
     .equ __lcd_rd=1
     .equ __lcd_enable=2
     .equ __lcd_busy_flag=7
#endasm

#pragma used+
static unsigned char _base_y[4]={0x80,0xc0};
unsigned char _lcd_x,_lcd_y,_lcd_maxx;
#pragma used-

static void _lcd_delay(void)
{
#asm
     ldi    r31,15
```

```
65    __lcd_delay0:
66        dec   r31
67        brne  __lcd_delay0
68    #endasm
69    }
70
71    void _lcd_ready(void)
72    {
73    DDRF=0x0F;            //muutetaan portin lukusuuntaa
74    PORTF=PORTF|0x02;         //ja maskataan rd bitti ylös
75    PORTF=PORTF&~0x01;         //ja maskataan rs bitti alas
76    #asm
77    __lcd_busy:
78    #endasm
79    _lcd_delay();
80    PORTF=PORTF|0x04;         //ja maskataan EN bitti ylös
81    _lcd_delay();
82    #asm
83        in    r26,__lcd_pin
84    #endasm
85    PORTF=PORTF&~0x04;         //ja maskataan EN bitti alas
86
87    _lcd_delay();
88    PORTF=PORTF|0x04;         //ja maskataan EN bitti ylös
89    _lcd_delay();
90    PORTF=PORTF&~0x04;         //ja maskataan EN bitti alas
91    #asm
92        sbrc  r26,__lcd_busy_flag
93        rjmp  __lcd_busy
94    #endasm
95    }
96    //
97    static void _lcd_write_nibble(void)
98    {
99    #asm
100       andi  r26,0xf0
101       or    r26,r27
102       sts   0x62,r26           ;write
103      ; sbi   __lcd_port,__lcd_enable ;EN=1
104   #endasm
105   PORTF=PORTF|0x04;         //ja maskataan EN bitti ylös
106   _lcd_delay();
107   PORTF=PORTF&~0x04;         //ja maskataan EN bitti alas
108   _lcd_delay();
109   }
110   //
111   void _lcd_write_data(unsigned char data)
112   {
113   PORTF=PORTF&~0x02;         //ja maskataan RD bitti alas
114   #asm
115       lds   r26,97
116       ori   r26,0xf0 | (1<<__lcd_rs) | (1<<__lcd_rd) | (1<<__lcd_enable) ;set as output

117       sts   0x61,r26
118       lds    r27,0x62
119       andi  r27,0xf
120       ld    r26,y
121   #endasm
122       _lcd_write_nibble();            //RD=0, write MSN
123   #asm
124       ld    r26,y
125       swap  r26
126   #endasm
127       _lcd_write_nibble();            //write LSN
```

```
128         PORTF=PORTF|0x02;          //ja maskataan RD bitti ylös
129     }
130
131     void lcd_write_byte(unsigned char addr, unsigned char data)
132     {
133     _lcd_ready();
134     _lcd_write_data(addr);
135     _lcd_ready();
136     PORTF=PORTF|0x01;          //ja maskataan RS bitti ylös
137     _lcd_write_data(data);
138     }
139
140     static void _lcd_read_nibble(void)
141     {
142     PORTF=PORTF|0x04;          //ja maskataan EN bitti ylös
143
144     _lcd_delay();
145     #asm
146         in    r30,__lcd_pin            ;read
147     #endasm
148     PORTF=PORTF&~0x04;          //ja maskataan EN bitti alas
149     _lcd_delay();
150     #asm
151         andi  r30,0xf0
152     #endasm
153     }
154
155     static unsigned char lcd_read_byte0(void)
156     {
157     _lcd_delay();
158     _lcd_read_nibble();             // read MSN
159     #asm
160         mov   r26,r30
161     #endasm
162     _lcd_read_nibble();             // read LSN
163     PORTF=PORTF&~0x02;          //ja maskataan RD bitti alas
164     #asm
165         swap  r30
166         or    r30,r26
167     #endasm
168     }
169
170     // read a byte from the LCD character generator or display RAM
171     unsigned char lcd_read_byte(unsigned char addr)
172     {
173     _lcd_ready();
174     _lcd_write_data(addr);
175     _lcd_ready();
176     #asm
177         in    r26,__lcd_direction
178         andi  r26,0xf                  ;set as input
179         out   __lcd_direction,r26
180         sbi   __lcd_port,__lcd_rs      ;RS=1
181     #endasm
182     return lcd_read_byte0();
183     }
184
185
186     void lcd_gotoxy(unsigned char x, unsigned char y)
187     {
188     _lcd_ready(); // RS=0
189     _lcd_write_data(_base_y[y]+x);
190     _lcd_x=x;
191     _lcd_y=y;
```

```
192    }
193
194
195    void lcd_clear(void)
196    {
197    _lcd_ready();          // RS=0
198    _lcd_write_data(2);    // cursor home
199    _lcd_ready();
200    _lcd_write_data(0xc); // cursor off
201    _lcd_ready();
202    _lcd_write_data(1);    // clear
203    _lcd_x=_lcd_y=0;
204    }
205
206    #pragma keep+
207    void lcd_putchar(char c)
208    {
209    #asm
210        push r30
211        push r31
212        ld   r26,y
213        set
214        cpi  r26,10
215        breq __lcd_putchar1
216        clt
217    #endasm
218    ++_lcd_x;
219    if (_lcd_x>_lcd_maxx)
220       {
221       #asm("__lcd_putchar1:")
222       ++_lcd_y;
223       lcd_gotoxy(0,_lcd_y);
224       #asm("brts __lcd_putchar0")
225       };
226    #asm
227        rcall __lcd_ready
228    #endasm
229    PORTF=PORTF|0x01;         //ja maskataan RS bitti ylös
230    #asm
231        ld   r26,y
232        st   -y,r26
233        rcall __lcd_write_data
234    __lcd_putchar0:
235        pop  r31
236        pop  r30
237    #endasm
238    }
239    #pragma keep-
240    //
241    // // write the string str located in SRAM to the LCD
242    void lcd_puts(char *str)
243    {
244    #ifdef _MODEL_TINY_
245    #asm
246        clr  r31
247    #endasm
248    #endif
249
250    #ifdef _MODEL_SMALL_
251    #asm
252        ldd  r31,y+1
253    #endasm
254    #endif
255
```

```
256    #asm
257        ld   r30,y
258    __lcd_puts0:
259        ld   r26,z+
260        tst  r26
261        breq __lcd_puts1
262        st   -y,r26
263        rcall _lcd_putchar
264        rjmp __lcd_puts0
265    __lcd_puts1:
266    #endasm
267    }
268
269    // write the string str located in FLASH to the LCD
270    void lcd_putsf(char flash *str)
271    {
272    #asm
273        ld   r30,y
274        ldd  r31,y+1
275    __lcd_putsf0:
276    #endasm
277    #if defined _CHIP_ATMEGA128_ || defined _CHIP_ATMEGA128L_
278    #asm("elpm")
279    #else
280    #asm("lpm")
281    #endif
282    #asm
283        tst  r0
284        breq __lcd_putsf1
285        adiw r30,1
286        st   -y,r0
287        rcall _lcd_putchar
288        rjmp __lcd_putsf0
289    __lcd_putsf1:
290    #endasm
291    }
292
293    static void _long_delay(void)
294    {
295    #asm
296        clr   r26
297        clr   r27
298    __long_delay0:
299        sbiw  r26,1         ;2 cycles
300        brne  __long_delay0 ;2 cycles
301    #endasm
302    }
303
304    static void _lcd_init_write(unsigned char data)
305    {
306    PORTF=PORTF&~0x02;        //ja maskataan RD bitti alas
307    #asm
308        ;cbi  __lcd_port,__lcd_rd     ;RD=0
309        lds    r26,97
310        ori   r26,0xf7                ;set as output
311        sts   0x61,r26
312        lds    r27,0x62
313        andi  r27,0xf
314        ld    r26,y
315    #endasm
316        _lcd_write_nibble();             //RD=0, write MSN
317        PORTF=PORTF|0x02;        //ja maskataan RD bitti ylös
318
319    }
```

```
320
321     // // initialize the LCD controller
322     unsigned char lcd_init(unsigned char lcd_columns)
323     {
324     PORTF=PORTF&~0x04;          //ja maskataan EN bitti alas
325     PORTF=PORTF&~0x01;          //ja maskataan RS bitti alas
326
327     _lcd_maxx=lcd_columns;
328     _base_y[2]=lcd_columns+0x80;
329     _base_y[3]=lcd_columns+0xc0;
330     _long_delay();
331     _lcd_init_write(0x30);
332     _long_delay();
333     _lcd_init_write(0x30);
334     _long_delay();
335     _lcd_init_write(0x30);
336     _long_delay();
337     _lcd_init_write(0x20);
338     _long_delay();
339     _lcd_write_data(0x28);
340     _long_delay();
341     _lcd_write_data(4);
342     _long_delay();
343     _lcd_write_data(0x85);
344     _long_delay();
345     #asm
346         lds    r26,97
347         andi  r26,0xf                  ;set as input
348         sts   0x61,r26
349     #endasm
350        PORTF=PORTF|0x02;          //ja maskataan RD bitti ylös
351     if (lcd_read_byte0()!=5) return 0;
352     _lcd_ready();
353     _lcd_write_data(6);
354     lcd_clear();
355     return 1;
356     }
357
358     //-----------------------------------------------------------------------------------------------------------------------------yllä
        olevat olivat lcd-kirjastoa muutettuna
359
360
361     /* function used to define user characters */
362
363     void define_char(byte flash *pc,byte char_code)
364     {
365         byte i,a;
366         a=(char_code<<3) | 0x40;
367         for (i=0; i<8; i++) lcd_write_byte(a++,*pc++);
368     }
369
370             unsigned int Menu_level=0, Warning=0;
371         unsigned char buff[21];
372         unsigned char RXReciveTable[50]={0};
373         unsigned char RXBufferState=0;
374         unsigned int LastByteReceaved=0;
375         unsigned char MAP,WaterTemp,AirTemp,CurrentTraction,Gear,BatteryVoltage;
376         unsigned int RPM,Lambda,ExtHauseTemperature;
377
378         flash int crctab[] =     /* CRC lookup table */
379         {
380          //tämä on salaista,  sisältää käyttäjäkohtaisen CRC-taulukon.
381         };
382
```

```
flash   unsigned short LambdaDisplayFilter_0_5_AEM[30]=
{0
,156
,312
,468
,624
,780
,936
,1092
,1160
,1248
,1404
,1560
,1716//0.69
,1872
,2028
,2100
,2184//0.73
,2262
,2340 //19 arvo
,2496 //0.76
,2574
,2652
,2730//0.79
,2808
,2886//25 arvo
,2964
,3042
,3120
,3198//0.85
,3276//0.86 ja 30 arvo
};
flash unsigned short LambdaDisplayFilter_0_5_AEM_H[30]=
{
3276 //0.86
,3354 //0.87
,3432 //0.88
,3484
,3536
,3588 //0.91
,3666
,3744
,3796
,3848
,3900 //0.96
,3939
,3978
,4017 //14 arvo
,4056 //1.0
,4095
,4134
,4173
,4212 //1.04
,4251
,4290
,4329
,4368//1.08
,4399
,4430
,4461
,4493
,4524//1.13
,4555
,4586 //1.15//30 ARVO
```

```
447
448 };
449
450 unsigned int calculateCRC(unsigned char *message, unsigned char length)
451 {
452 //tämäkin on salaista.
453 }
454
455 interrupt [USART0_RXC] void uart0_rx_isr(void)
456 {
457
458     unsigned char Recived;
459     static unsigned char RXReciveIndex=0;
460
461     Recived=UDR0;
462     LastByteReceaved=0;
463
464
465     if ((RXBufferState==BUFFER_EMPTY))
466     {
467             RXBufferState=BUFFER_STARTED;
468             RXReciveIndex=0;
469
470             RXReciveTable[0]=MSG_START_BYTE;
471             return;
472     }
473     else if(RXBufferState!=BUFFER_EMPTY)
474     {
475         RXReciveIndex++;
476             if(RXReciveIndex==1)
477         {
478                 //DataLen=Recived;
479                 RXReciveTable[1]=Recived;//DataLen;
480         }
481         else
482         {
483             if(RXBufferState!=BUFFER_FULL)
484             {
485                 if(RXReciveIndex <= 38)
486
487                     RXReciveTable[RXReciveIndex]=Recived;
488                 else
489                     RXBufferState=BUFFER_FULL;
490             }
491
492             if(RXReciveTable[1]==(RXReciveIndex-1))
493             {
494                     RXBufferState=BUFFER_FULL;
495             }
496
497
498
499
500         }
501     }
502
503
504 }
505
506
507 void ClearValuesFromBuffer(void)
508 {
509
510     int n=0;
```

```
511
512         for (n=0;n<=50;n++)
513         {
514             RXReciveTable[n]=0;
515         }
516         RXBufferState=BUFFER_EMPTY;
517
518     }
519
520     void PrintLCD(void);
521
522     void CheckValuesFromBuffer(void)
523     {
524
525         unsigned char i=4;
526         unsigned char x=0;
527         unsigned char datalength = 0;
528            //     unsigned char comBuffer[53];
529
530
531         //unsigned int messageId = 0;
532         unsigned int calculatedCRC = 0;
533         unsigned int messageCRC = 0;
534
535
536            //memcpy(comBuffer, RXReciveTable, 30);
537
538
539
540         /* ToDo: Receive data from serial port */
541
542         memcpy(&datalength, RXReciveTable + START_BLOCK_LENGTH, LEN_BLOCK_LENGTH);/* Length
    */
543         calculatedCRC = calculateCRC(RXReciveTable, datalength);
544
545         /*Calculate actual data length */
546         datalength = datalength - 4;
547
548
549
550             memcpy(&messageCRC, RXReciveTable + START_BLOCK_LENGTH + LEN_BLOCK_LENGTH +
    ID_BLOCK_LENGTH + datalength, CRC_BLOCK_LENGTH);
551
552
553
554         if(calculatedCRC != messageCRC)
555         {/*Corrupted message */
556
557                             //lcd_putsf("moe");
558             RXBufferState=BUFFER_EMPTY;
559             ClearValuesFromBuffer();
560
561         }
562         else
563         {
564         MAP=RXReciveTable[i+1];
565
566
567         RPM=(unsigned int) RXReciveTable[i+2];
568
569             RPM=RPM*100;
570         RPM=RPM+(unsigned int) RXReciveTable[i+3];
571
572
```

```
573         Lambda=RXReciveTable[i+4];
574         Lambda=Lambda*100;
575             Lambda=Lambda+RXReciveTable[i+5];
576 
577         if(Lambda==0)
578             Lambda=5000;
579 
580 
581 
582                 if(Lambda<3237)
583                 {
584                     unsigned int CurrentTemp=0;
585                     CurrentTemp = LambdaDisplayFilter_0_5_AEM[1]/2;
586                     x=0;
587                     while(Lambda>CurrentTemp)
588                     {
589                         x++;
590                         CurrentTemp = LambdaDisplayFilter_0_5_AEM[x]+
    LambdaDisplayFilter_0_5_AEM[x+1];
591                         CurrentTemp = CurrentTemp /2;
592                     }
593                     Lambda=57 + x;
594                 }
595                 else
596                 {
597                     unsigned int CurrentTemp=0;
598                     CurrentTemp = LambdaDisplayFilter_0_5_AEM_H[0]+
    LambdaDisplayFilter_0_5_AEM_H[1];
599                     CurrentTemp = CurrentTemp /2;    //3393 eka
600                     x=0;
601                     while(Lambda>CurrentTemp) //3354 //0.
602                     {
603                         x++;
604                         CurrentTemp = LambdaDisplayFilter_0_5_AEM_H[x]+
    LambdaDisplayFilter_0_5_AEM_H[x+1];
605                         CurrentTemp = CurrentTemp /2;
606                     }
607                     Lambda=86 + x;
608                 }
609                 Lambda=Lambda*10;
610 
611                 if(Lambda<750)
612                     Lambda=750;
613 
614 
615 
616 
617             WaterTemp=RXReciveTable[i+8];
618             if(WaterTemp>40)
619                 WaterTemp=WaterTemp-40;
620             AirTemp=RXReciveTable[i+9];
621             if(AirTemp>40)
622                 AirTemp=AirTemp-40;
623 
624             ExtHauseTemperature=RXReciveTable[i+14];
625             ExtHauseTemperature=ExtHauseTemperature*20;
626 
627             BatteryVoltage=RXReciveTable[i+16];
628             Gear=RXReciveTable[i+17];
629             CurrentTraction=RXReciveTable[i+18];
630 
631             RXBufferState=BUFFER_EMPTY;
632             PrintLCD();
633             }
```

```
634
635     }
636
637     void lcd_alustus()
638     {
639     //01234567890123456789
640     //Tässä lukeepi sitten vapaavalintainen tatech display tms..
641     // RPM    MAP  LAMBDA
642     //xxxx  -0.15   0.92
643     //tähän lampada-palkki
644
645     lcd_gotoxy(1,1);
646     //lcd_putsf("RPM    MAP  LAMBDA");
647     lcd_putsf("RPM     Map    Lambda");
648
649     }
650
651     void PrintLCD(void)
652     {
653        char apu[6];
654        char xxx[20];
655     static   int paikka=0,wpaikka=10;
656        float fTemppi=0;
657        int MAPDisplayTable[31]={-100,-80,-70,-65,-60,-55,-50,-45,-40,-35,-30,-20,-10,0,10
       ,20,30,40,50,60,70,80,90,100,110,120,130,140,150,160,170}; //
658
659             Warning=0;
660
661         static int BatteryVoltageHigh=0, BatteryVoltageLow=0;
662
663         if ( Warning == 0)
664
665     //01234567890123456789
666     // RPM    MAP  LAMBDA
667     //xxxx  -0.15   0.92
668     //tähän lampada-palkki
669
670         {
671
672             //lcd_clear();
673             if (RPM<10000)
674             {
675                 lcd_gotoxy(4,2);
676                     lcd_putsf(" ");
677             }
678                 if ( RPM < 1000 )
679                 {
680                     lcd_gotoxy(0,2);
681                     lcd_putsf("    ");
682                     lcd_gotoxy(1,2);
683                     sprintf(buff,"%3d", RPM);
684                     lcd_puts(buff);
685                 }
686                 else
687                 {
688                     lcd_gotoxy(0,2);
689                     sprintf(buff,"%4d", RPM);
690                     lcd_puts(buff);
691                 }
692
693                 /*
694             lcd_gotoxy(7,0);
695             lcd_putsf("W");
696             sprintf(buff,"%3d", WaterTemp);
```

```
            lcd_puts(buff);
            lcd_putchar(0);

            lcd_gotoxy(12,0);
            lcd_putsf("A");
            sprintf(buff,"%3d", AirTemp);
            lcd_puts(buff);
           */

                     lcd_gotoxy(6,2);
                    // itoa(Lambda,apu);
                    // lcd_puts(apu);

fTemppi=MAPDisplayTable[MAP];
fTemppi=fTemppi/100;
if(fTemppi<0)
         ftoa(fTemppi, 2, buff);
else
              {
//       char xxx[20];
              ftoa(fTemppi, 2, xxx);
              sprintf(buff, " %s", xxx);
              }
 lcd_puts(buff);



          lcd_gotoxy(16,2);
          fTemppi=Lambda ;
          fTemppi=fTemppi/1000;

              ftoa(fTemppi, 2, xxx);
              sprintf(buff, "%s", xxx);

           lcd_puts(buff);




//01234567890123456789
// RPM    MAP  LAMBDA
//xxxx  -0.15   0.92
//tähän lampada-palkki


if (Lambda < 750 || Lambda > 1150)
{
    if (Lambda < 750)
    {
        paikka=0;
    }
    else
    {
        paikka=19;
    }
}
else
{
         paikka=(Lambda-750)/20;

}

```

```
761     //if(paikka=!wpaikka)
762     {
763          lcd_gotoxy(wpaikka,3);       //nollataan vanha paikka
764          lcd_putsf(" ");
765          lcd_gotoxy(paikka,3);
766          lcd_putchar(0);
767          //lcd_putsf("l");
768
769          wpaikka=paikka;
770     }
771
772                    /*
773               lcd_gotoxy(6,1);
774               sprintf(buff, "E%3d", ExtHauseTemperature);
775               lcd_puts(buff);
776               lcd_putchar(0);
777
778               BatteryVoltageHigh=BatteryVoltage/10;
779               BatteryVoltageLow=BatteryVoltage-BatteryVoltageHigh*10;
780
781               lcd_gotoxy(11,1);
782               sprintf(buff, "V%2d.", BatteryVoltageHigh);
783               lcd_puts(buff);
784               sprintf(buff, "%1d", BatteryVoltageLow);
785               lcd_puts(buff);
786               */
787          }
788     }
789
790
791     void testi()
792     {
793     char apu[6];
794
795     lcd_gotoxy(0,1);
796     lcd_putsf("RPM ");
797     itoa(TCNT1*10,apu);
798     lcd_puts(apu);
799     lcd_gotoxy(10,1);
800     lcd_putsf(" EGT 825 C");
801
802     lcd_gotoxy(0,2);
803     lcd_putsf("Oil 97 C Water 103 C");
804
805     lcd_gotoxy(0,3);
806     lcd_putsf("Lambda 0.9 Knock 102");
807
808     delay_ms(200);
809
810
811     }
812
813
814     // External Interrupt 0 service routine
815     interrupt [EXT_INT0] void ext_int0_isr(void)
816     {
817     // Place your code here
818
819     }
820
821     // External Interrupt 1 service routine
822     interrupt [EXT_INT1] void ext_int1_isr(void)
823     {
824     // Place your code here
```

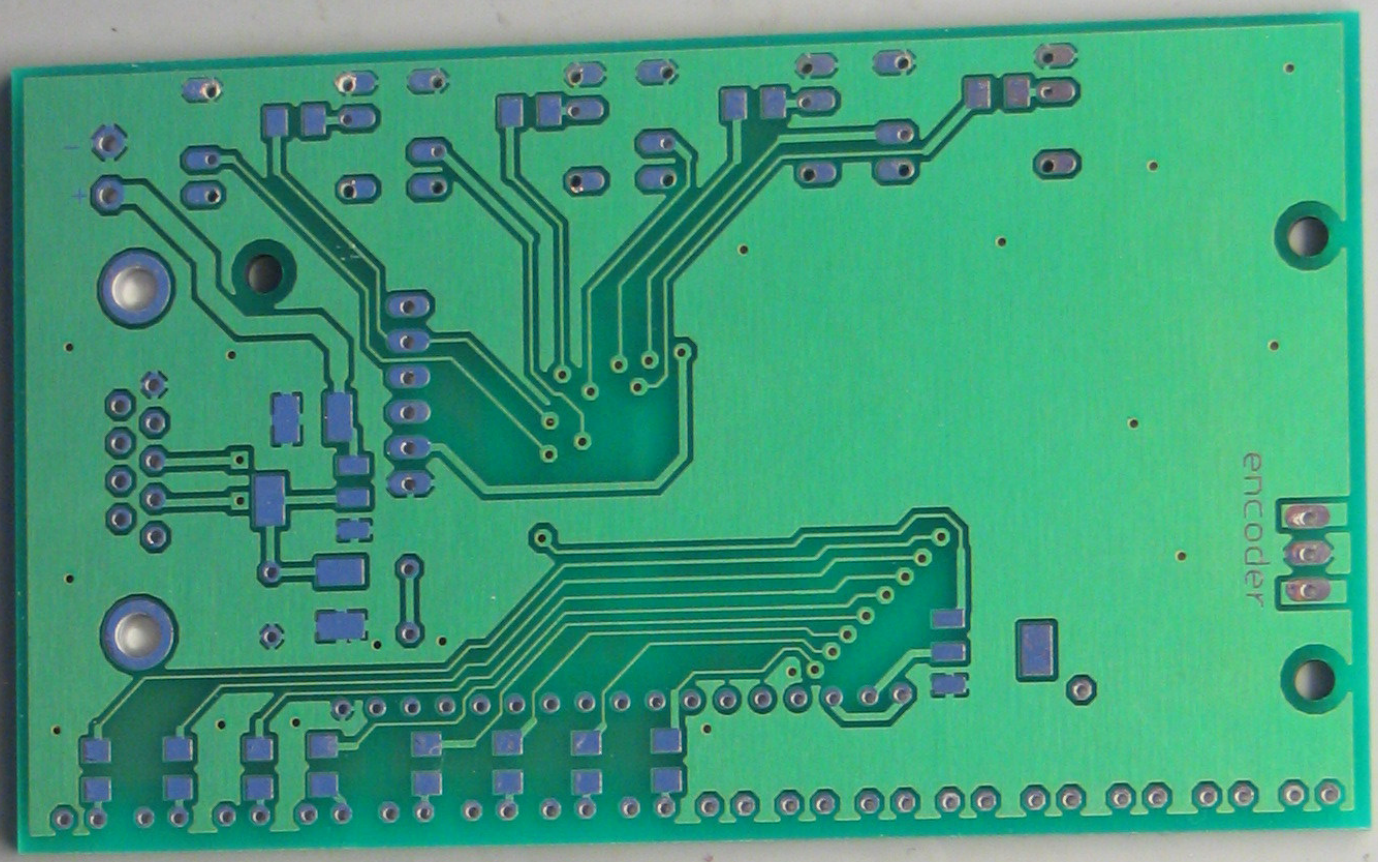
```
825
826     }
827
828     // External Interrupt 2 service routine
829     interrupt [EXT_INT2] void ext_int2_isr(void)
830     {
831     // Place your code here
832
833     }
834
835     // External Interrupt 3 service routine
836     interrupt [EXT_INT3] void ext_int3_isr(void)
837     {
838     // Place your code here
839
840     }
841
842     // External Interrupt 4 service routine      painonappi, vasemmanpuoleisin
843     interrupt [EXT_INT4] void ext_int4_isr(void)
844     {
845     #ifdef testi
846     // Place your code here
847     PORTB.0=~PORTB.0;
848     PORTB.1=~PORTB.1;  //painonappien värien vaihto
849     #endif
850     }
851
852     // External Interrupt 5 service routine      painonappi, toinen vasemmalta
853     interrupt [EXT_INT5] void ext_int5_isr(void)
854     {
855     // Place your code here
856     #ifdef testi
857     PORTB.2=~PORTB.2;
858     PORTB.3=~PORTB.3;  //painonappien värien vaihto
859
860             LCD_BL=+10;  //taustavalon kirkkauden lisäystä
861     #endif
862     }
863
864     // External Interrupt 6 service routine      painonappi, toinen oikealta
865     interrupt [EXT_INT6] void ext_int6_isr(void)
866     {
867     // Place your code here
868     #ifdef testi
869     PORTB.4=~PORTB.4;
870     PORTB.5=~PORTB.5;  //painonappien värien vaihto
871
872     LCD_BL=-10;           //taustavalon kirkkauden vähennystä
873     #endif
874     }
875
876     // External Interrupt 7 service routine      painonappi, oikeanpuoleisin
877     interrupt [EXT_INT7] void ext_int7_isr(void)
878     {
879     // Place your code here
880     #ifdef testi
881     PORTB.6=~PORTB.6;
882     PORTB.7=~PORTB.7;  //painonappien värien vaihto
883
884           if (PORTA==0 && PORTC==0)  //ledipalkiston siirtoa
885             {
886             PORTC=0xFF;
887             PORTA=0xFE;
888             }
```

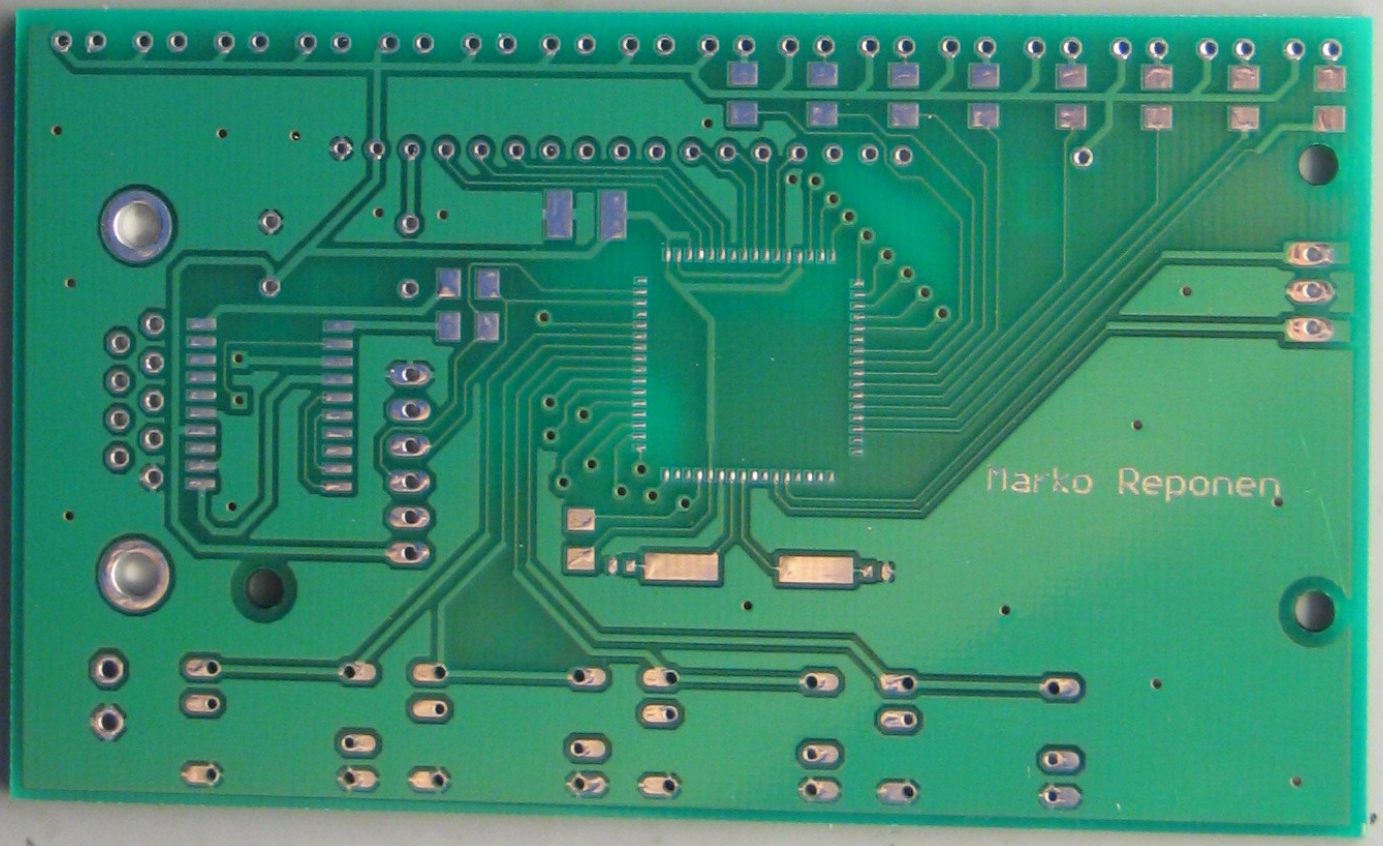
```
889
890
891           if (PORTA==0)
892             {
893             PORTC=PORTC/2;
894             }
895
896           PORTA=PORTA/2;
897     #endif
898     }
899
900
901     // Declare your global variables here
902
903     void main(void)
904     {
905     // Declare your local variables here
906     unsigned int i=0;
907     // Input/Output Ports initialization
908     // Port A initialization
909     // Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
910     // State7=1 State6=1 State5=1 State4=1 State3=1 State2=1 State1=1 State0=1
911     PORTA=0xFF;
912     DDRA=0xFF;
913
914     // Port B initialization
915     // Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
916     // State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
917     PORTB=0x00;
918     DDRB=0xFF;
919
920     // Port C initialization
921     // Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
922     // State7=1 State6=1 State5=1 State4=1 State3=1 State2=1 State1=1 State0=1
923     PORTC=0xFF;
924     DDRC=0xFF;
925
926     // Port D initialization
927     // Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
928     // State7=P State6=P State5=P State4=P State3=P State2=P State1=P State0=P
929     PORTD=0xFF;
930     DDRD=0x00;
931
932     // Port E initialization
933     // Func7=In Func6=In Func5=In Func4=In Func3=Out Func2=In Func1=Out Func0=In
934     // State7=P State6=P State5=P State4=P State3=0 State2=P State1=1 State0=P
935     PORTE=0xF7;
936     DDRE=0x0A;
937
938     // Port F initialization
939     // Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
940     // State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
941     PORTF=0x00;
942     DDRF=0x00;    //oli 0xFF eli lähtö
943
944     // Port G initialization
945     // Func4=In Func3=In Func2=In Func1=In Func0=In
946     // State4=P State3=P State2=P State1=P State0=P
947     PORTG=0x1F;
948     DDRG=0x00;
949
950     // Timer/Counter 0 initialization
951     // Clock source: System Clock
952     // Clock value: 125,000 kHz
```

```
953     // Mode: Normal top=FFh
954     // OC0 output: Disconnected
955     ASSR=0x00;
956     TCCR0=0x05;
957     TCNT0=0x00;
958     OCR0=0x00;
959
960     // Timer/Counter 1 initialization
961     // Clock source: System Clock
962     // Clock value: Timer 1 Stopped
963     // Mode: Normal top=FFFFh
964     // OC1A output: Discon.
965     // OC1B output: Discon.
966     // OC1C output: Discon.
967     // Noise Canceler: Off
968     // Input Capture on Falling Edge
969     // Timer 1 Overflow Interrupt: Off
970     // Input Capture Interrupt: Off
971     // Compare A Match Interrupt: Off
972     // Compare B Match Interrupt: Off
973     // Compare C Match Interrupt: Off
974     TCCR1A=0x00;
975     TCCR1B=0x00;
976     TCNT1H=0x00;
977     TCNT1L=0x00;
978     ICR1H=0x00;
979     ICR1L=0x00;
980     OCR1AH=0x00;
981     OCR1AL=0x00;
982     OCR1BH=0x00;
983     OCR1BL=0x00;
984     OCR1CH=0x00;
985     OCR1CL=0x00;
986
987     // Timer/Counter 2 initialization
988     // Clock source: System Clock
989     // Clock value: 15,625 kHz
990     // Mode: Normal top=FFh
991     // OC2 output: Disconnected
992     TCCR2=0x05;
993     TCNT2=0x00;
994     OCR2=0x00;
995
996
997     // Timer/Counter 3 initialization
998     // Clock source: System Clock
999     // Clock value: 230,391 kHz
1000    // Mode: Ph. correct PWM top=00FFh
1001    // Noise Canceler: Off
1002    // Input Capture on Falling Edge
1003    // OC3A output: Non-Inv.
1004    // OC3B output: Discon.
1005    // OC3C output: Discon.
1006    // Timer 3 Overflow Interrupt: Off
1007    // Input Capture Interrupt: Off
1008    // Compare A Match Interrupt: Off
1009    // Compare B Match Interrupt: Off
1010    // Compare C Match Interrupt: Off
1011    TCCR3A=0x81;
1012    TCCR3B=0x03;
1013    TCNT3H=0x00;
1014    TCNT3L=0x00;
1015    ICR3H=0x00;
1016    ICR3L=0x00;
```

```
1017    OCR3AH=0x00;
1018    OCR3AL=0x77;
1019    OCR3BH=0x00;
1020    OCR3BL=0x00;
1021    OCR3CH=0x00;
1022    OCR3CL=0x00;
1023
1024
1025    // External Interrupt(s) initialization
1026    // INT0: On
1027    // INT0 Mode: Falling Edge
1028    // INT1: On
1029    // INT1 Mode: Falling Edge
1030    // INT2: On
1031    // INT2 Mode: Falling Edge
1032    // INT3: On
1033    // INT3 Mode: Falling Edge
1034    // INT4: On
1035    // INT4 Mode: Falling Edge
1036    // INT5: On
1037    // INT5 Mode: Falling Edge
1038    // INT6: On
1039    // INT6 Mode: Falling Edge
1040    // INT7: On
1041    // INT7 Mode: Falling Edge
1042    EICRA=0xAA;
1043    EICRB=0xAA;
1044    EIMSK=0xFF;
1045    EIFR=0xFF;
1046
1047    // Timer(s)/Counter(s) Interrupt(s) initialization
1048    TIMSK=0x00;
1049    ETIMSK=0x00;
1050
1051    // Analog Comparator initialization
1052    // Analog Comparator: Off
1053    // Analog Comparator Input Capture by Timer/Counter 1: Off
1054    ACSR=0x80;
1055    SFIOR=0x00;
1056
1057    // USART0 initialization
1058    // Communication Parameters: 8 Data, 1 Stop, No Parity
1059    // USART0 Receiver: On
1060    // USART0 Receiver Interrupt : On
1061    // USART0 Transmitter: Off
1062    // USART0 Mode: Asynchronous
1063    // USART0 Baud rate: 19200
1064    UCSR0A=0x00;
1065    UCSR0B=0x90;
1066    UCSR0C=0x06;
1067    UBRR0H=0x00;
1068    UBRR0L=0x2F;
1069
1070    lcd_init(16);              //lcd:n koon määrittely
1071    PORTA=0xFF;                //8 vasemmanpuoleista lediä, 0-aktiivinen
1072    PORTC=0xFF;                //8 oikeanpuoleista lediä, 0-aktiivinen
1073    PORTB=0b10010101;          //painonappien ledit
1074    lcd_clear();               //lcd-näytön putsaus
1075    #ifdef testi
1076    lcd_putsf("    Lopputyo");
1077    lcd_gotoxy(0,1);
1078    lcd_putsf("  Marko Reponen");
1079    delay_ms(650);
1080    #endif
```

```
1081    #asm("sei")                 //sallitaan keskeykset.
1082    while (1)
1083          {
1084          #ifdef testi
1085          delay_ms(200);
1086
1087          if (PORTA==0 && PORTC==0)
1088            {
1089            PORTC=0xFF;
1090            PORTA=0xFE;
1091            }
1092
1093
1094          if (PORTA==0)
1095            {
1096            PORTC=PORTC/2;
1097            }
1098          PORTA=PORTA/2;
1099           #endif
1100
1101            delay_ms(10);
1102            if (RXBufferState==BUFFER_FULL)
1103            {
1104            CheckValuesFromBuffer();
1105            }
1106
1107          };
1108    }
1109
```


encoder


Marko Reponen