Ville Eerola

Ultraäänikaiuttimien ohjainlaite

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Sähkötekniikka

Insinöörityö

6.4.2016

Metropolia

Tekijä Otsikko Sivumäärä Aika	Ville Eerola Ultraäänikaiuttimen ohjainlaite 22 sivua + 2 liitettä 6.4.2016
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Sähkötekniikka
Suuntautumisvaihtoehto	Elektroniikka
Ohjaajat	Lehtori Esko Tattari Lehtori Jari Savolainen

Tässä insinöörityössä suunniteltiin ja valmistettiin ultraäänikaiuttimen ohjainlaite. Suunniteltu laite perustui aikaisemmin tehtyyn ultraäänipesurin ohjainlaitteeseen. Ohjainlaitteesta pyrittiin tekemään soveltuva useampaan käyttötarkoitukseen. Suunnittelussa kiinnitettiin huomiota erityisesti mikrokontrollerin suojaamiseen mahdolliselta ylijännitteeltä. Työssä käsitellään myös ultraäänen luonti sekä sen mahdollisia käyttökohteita.

Ohjainlaiteen suunnittelun ja valmistamisen jälkeen ohjainlaitteelle suoritettiin tarvittavat tarkistusmittaukset. Mikrokontrollerille tehtiin ohjelmakoodi C-kielellä. Ohjelmoinnin jälkeen ohjainlaitteen toiminta testattiin vaiheittain. Esiin tulleet viat paikannettiin erilaisten vianhakumenetelmien avulla. Mittauksissa käytettiin apuna yleismittaria ja oskilloskooppia.

Mikrokontrollerin suojauksessa ylijännitteeltä onnistuttiin hyvin. Työssä havaittiin vielä tarvetta jatkaa ohjainlaitteen kehittelyä häiriöiden poistamiseksi.

Avainsanat	Ultraääni, ultraäänikaiutin, ohjainlaite, suojaus

Author	Ville Eerola
Title	Ultrasonic Speaker Control Device
Number of Pages	22 pages + 2 appendices
Date	6 April 2016
Degree	Bachelor of Engineering
Degree Programme	Electrical engineering
Specialisation option	Electronics
Instructors	Esko Tattari, Senior Lecturer Jari Savolainen, Senior Lecturer

The aim of this Bachelor's degree thesis was to design and manufacture a device to control an ultrasonic loudspeaker. The new device is based on modified principles of an ultrasonic cleaning machine's control unit. One of the goals in this project was to design a control unit that can be used in wide range of applications. During the design process special attention was paid to protect the device's microcontroller unit from surge and overvoltage. In this thesis also the different possibilities for usage of ultrasound and basic principles behind the technology are discussed.

Calibration and step by step testing of the finalized device were carried out by using a computer program specially developed for this purpose. The program code was created by using code C. Testing the device revealed some shortcomings in the design. These issues were located and solved by using different troubleshooting methods. Digital multi meters and oscilloscopes were also used to measure the design and its performance in this project.

The overvoltage protection construction designed for the device worked well. There is further need for ongoing research and development as the device suffers from outside electrical interference.

Keywords	ultrasound, ultrasonic speaker, control device, protection

Sisällys

1 Johdanto

Tämä insinöörityö on tehty Metropolia Ammattikorkeakoululle. Työssä esitellään suunniteltu ja rakennettu ultraäänikaiuttimen käyttölaite. Laitteen on tarkoitus olla yleismallinen, jolloin sitä voitaisiin käyttää useissa eri sovelluksissa ohjaamaan ultraäänikaiuttimia. Mahdollisia sovelluksia olisivat ultraäänipesuri, ultraänilevitaatio, jossa saadaan kappaleen ja kuljettimen välinen kitka mahdollisimman pieneksi. Lisäksi mahdollisia käyttökohteita ovat mudan ja lumenpoistosovellukset.

Metropolia Ammattikorkeakoululle on aiemmin tehty insinöörityö, joka käsitteli ultraäänipesuria. Kyseinen työ toimii pohjana nykyisessä työssä. Tavoitteena on parannella aiemmin tehtyä ohjainlaitetta ja tehdään tarvittavat tarkastusmittaukset. Kyseinen ohjainlaite on toteutettu mikroprosessorilla. Työ koostuu laitteen elektroniikka- ja piirilevysuunnittelusta, kokoonpanon toteutuksesta, mikrokontrollerin ohjelmoinnista sekä kokonaisuuden testimittauksista.

2 Ultraääni

Ultraääni on ilman tai muun väliaineen värähtelyä kuten normaalisti kuultava äänikin. Ultraääni on taajuudeltaan ihmisen kuuleman äänialueen yläpuolella. Taajuusalue on välillä 20 kHz - n. 10 THz. Alaraja määräytyy ihmisen kuuloalueen mukaan, ja yläraja johtuu atomien massasta ja niiden välillä olevista voimista. [1, s. 157.]

2.1 Ultraäänen synnyttäminen ja havainnointi

Ultraääntä tuotetaan ja kyetään havainnoimaan pietsosähköisten kiteiden avulla. Kide muuttaa kokoaan, kun sen vastakkaisille sivuille kytketään jännite. Tällä ilmiöllä on myös käänteisilmiö, jos kidettä puristettaessa sen sivuille syntyy jännite. Tätä kutsutaan sähköstriktioksi. Tämän ominaisuuden avulla kyetään havainnoimaan ultraääniä. Pietsosähköisistä kiteistä valmistettu kaiutin pystyy lähettämään sekä havaitsemaan ultraääntä. Myös värähtelevän sähkökentän avulla voidaan saada ferromagneettinen metallinkappale värähtelemään. Tätä ilmiötä kutsutaan magnetostriktioilmiöksi. Tälläkin tavoin voidaan synnyttää ultraääntä. [1, s. 157 - 158.]

2.2 Ultraäänen käyttökohteet

Ultraääntä käytetään kuvantamiseen, koska ultraääni heijastuu rajapinnasta, jos akustinen ominaisimpedanssi on erisuuri rajapinnan eri puolilla. Kuvantamisella voidaan tutkia niin ihmisen kudoksia tai sikiötä, sekä eri materiaalin sisäisiä rakenteita. Kaikuluotaus on yleinen ultraäänisovellus, jolla voidaan mitata veden syvyyttä ja paikallistaa kappaleita.

Ultraääni saa aikaan mekaanista värähtelyä. Tätä käytetään hyödyksi ultraäänipesureissa. Ultraääni johdetaan pestävään kappaleeseen pesunesteen välityksellä. Näin tekemällä saadaan aikaan pesunesteessä paineenvaihteluita. Tämä synnyttää pestävän kappaleen pinnalle kavitaatiokuplia, jotka aiheuttavat suuria painehuippuja ja aikaansaavat hyvän pesutuloksen. Ultraäänellä voidaan myös liittää kappaleita toisiinsa. Tätä kutsutaan ultraäänihitsaukseksi. Ultraääni aiheuttaa liitoskohdassa värähtelyä, joka kuumentaa liitoksen ja sulattaa materiaalit yhteen. [1, s. 159.]

Ultraäänilevitaatiossa saadaan kappale leijumaan pinnan päällä, esimerkiksi kuljettimen hihnan yläpuolella niin, että kuljettimen ja kappaleen välinen kitka on lähes nolla. Kevyitä kappaleita voidaan myös vangita ultraäänikenttään, jossa niitä voidaan ohjailla kolmiulotteisessa tilassa.

Ultraäänellä voidaan myös karkottaa eläimiä. Hiirien ja rottien karkotukseen sopiva taajuus on noin 23,5 kHz. Eläin kokee äänen ärsyttäväksi ja karttaa sitä. Kissoilla taajuus on noin 23 kHz eikä ääni saa olla jatkuva. Muuten kissa tottuu ääneen. Koirilla taajuuden täytyy olla 30 kHz tai yli. Ultraääntä käytetään myös koirien koulutukseen ja haukunestopannoissa. [1, s. 5 - 6.]

3 Ohjainlaite

Ohjainlaite perustuu Juha Vaaksion insinöörityössään tekemään ohjainlaitteeseen. Laitetta kehitetään ja siihen tehdään parannuksia. Peruskytkennät ja komponentit pidetään samoina. Kytkentään lisätään optoerottimia erottamaan matalan jännitteen puoli korkeamman jännitteen puolesta. Korkeampi jännite on tarkoitettu kaiuttimien syöttöä varten. Tämä toteutetaan erillisellä jo rakennetulla rengasydin muuntajalla ja tasasuuntauspiirillä. Tämä tuottaa 140 V:n tasajännitteen. Muut tarvittavat jännitteet toteutettiin Tracopowerin jännitelähteellä. Näin saimme viiden ja kahdentoista voltin tasajännitteet.

3.1 Mikrokontrollerin kytkennät

Mikrokontrollerina päätettiin pitää Atmega32:ta, joka on sama kuin Juha Vaaksion insinöörityössä. Kontrollerin pinnit vcc (10) avcc (30) ja reset (9) kytkettiin suoraan viiden voltin jännitteeseen. Gnd-pinnit 11 ja 31 kytkettiin maihin. Viiden voltin ja maan välille kontrollerin välittömään läheisyyteen kytkettiin 100 nF:n kondensaattori suodattamaan käyttöjännitettä. Kiteelle tarkoitettujen pinnien 12 ja 13 väliin kytkettiin 8 MHz:n kide. Pinnit 12 ja 13 kytkettiin lisäksi 22 pF:n kondensaattorin kautta maihin.

Kontrolleriin kytkettiin viisi painokytkintä maata vasten. Kytkimien pinnit ohjelmoitiin toimimaan ylösvedetyssä tilassa, jolloin kontrollerin sisällä tapahtuu linjan ylösveto. Näin toimimalla ei tarvitse kytkeä kytkinlinjoihin erillisiä ylösvetovastuksia. Kytkimet tulivat pinneihin 3, 14, 15, 16 ja 17. Pinnit 3, 16 ja 17 mahdollistavat kytkimillä tapahtuvan keskeytyksen.

Ajastimen pinneihin 28 ja 29 kytkettiin 32,768 kHz:n kide. Pinnit 19 ja 20 kytkettiin optoerottimille. Lisäksi piirin portti A:han (pinnit 33 - 40) kytkettiin LCD-näyttö. Näyttö tarvitsi kontrollerin kytkennän lisäksi käyttöjännitteet itselleen ja taustavalolle sekä kontrastin säädön. Kontrastin säätö tapahtui säätövastuksella, jonka kiinteät jalat kytkettiin viiden voltin ja maan välille ja liuku LCD-näytölle.

3.2 Optoerotimet kytkennässä

Suojauskyvyn, riittävän nopeuden ja saatavuuden perusteella optoeroittimeksi valittiin H11L3. Siinä on myös schmitt-liipaisin ulostulossa. Tämä takaa lähes digitaalisen ulostulosignaalin. Etuvastus kytkettiin optoeroittimen sisääntulopinniin 1. Etuvastuksen arvo saatiin kaavalla 1:

$$R_{led} = \frac{U_r}{I} \quad (1)$$

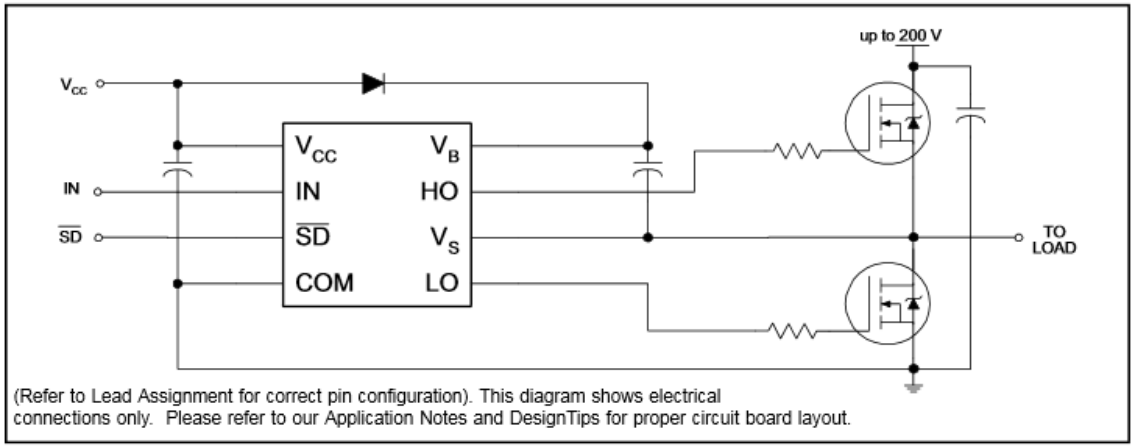
Kaavassa R_{led} on etuvastuksen resistanssiarvo. U_r etuvastuksen yli oleva jännite ja I ledin läpimenevä virta. Virta saadaan H11L3:n datalehdestä I = 10 mA [3]. Ledin kynnysjännite selvitettiin yleismittarilla mittaamalla. Kynnysjännitteeksi saatiin 1,2 V.

U_r saadaan vähentämällä ledin kynnysjännite piirin syöttöjännitteestä. Kontrolleri syöttää viiden voltin jännitettä, joten voimme laskea U_r:n:
U_r = 5 V - 1,2 V = 3,8 V.
Näin ollen voimme laskea ledin etuvastuksen R_{led}:in arvon.
R_{led} = 3,8 V / 10 mA = 380 Ω
Etuvastukseksi valittiin 390 ohmin vastus, koska tarkkuus riittää ja kyseinen arvo löytyy vastusten E12-sarjasta.

Optoerottimen ulostulon kytkentä tehtiin noudattamalla datalehden testikytkentää. Kytkennässä oleva 280 ohmin vastus korvattiin E12-sarjasta löytyvällä 270 ohmin vastuksella. Jännitteeksi piirille syötettiin 12 V pinniin 6. Jännitteen ja maan väliin kytkettiin 100 nF:n suodatuskondensaattori. Kytkennän toimivuus testattiin funktiogeneraattoria ja oskilloskooppia apuna käyttäen. Näin varmistettiin suunnitellun suojakytkennän toiminta.

3.3 Ajuripiirin kytkentä

Ajuripiirinä käytettiin IRS2004:ää ohjaamaan kahta fettiä. Fetteinä käytettiin IRF640N. Nämä kykenevät riittävään nopeuteen sekä kestävät riittävän suuren jännitteen. Ohjaus onnistuu näillä 5 V:n Gate-jännitteellä. Kytkentä toteutettiin kuvan 1 mukaisesti. [4, s. 16.]



Kuva 1: Ajuripiirin kytkentä [5]

Suodatuskondensaattoriksi kytkettiin käyttöjännitteiden välille 100 nF:n kondensaattori piirin välittömään läheisyyteen. Diodina käytettiin MUR120RLG:tä saatavuuden ja jännite kestojen takia. Fettien etuvastuksiksi valittiin 50 ohmin vastukset. Yleisesti Gate-vastukset ovat arvoltaan 10 - 500 ohmia.

V_b-pinnin ja V_s-pinnin välistä kondensaattoria kutsutaan bootstrap-kondensaattoriksi. Tämä ja diodi huolehtivat high-side-ohjauslogiikan käyttöjännitteestä. [4, s. 24.]

3.4 Bootstrap-kondensaattorin mitoitus

Bootstrap-kondensaattorin mitoituksessa huomioitava kondensaattorin tulee kyetä pitämään tarvittava jännite high-side-ohjaimen tarvitseman ajan. Lisäksi kondensaattorin sisäisen vastuksen tulee olla mahdollisimman pieni. Tätä ilmaistaan ESR (Equivalent Series Resistance) -arvolla.

Bootstrap-kondensaattorin arvo saadaan kaavalla 2:

$$C_{boot} = \frac{Q_{tot}}{\Delta V_{bs}} \tag{2}$$

Kaavassa 2 Q_{tot} on kytkennässä esiintyvien varausten summa ja ΔV_{bs} on suurin sallittu jännitteen putoama.
Q_{tot} saadaan laskettua kaavalla 3:

$$Q_{tot} = Q_G + Q_{LS} + (I_{GSS} + I_{QBS} + I_{LK} + I_{LK\_DIODE} + I_{LK\_CAP} + I_{DS-}) * T_{HON} \tag{3}$$

Kaavassa 3 Q_G on MOSFETin hilan varaus, Q_{LS} on ajuripiirin tasonvaihtajien tarvitsema lataus, I_{GSS} on MOSFETin vuotovirta hilalta lähteelle, I_{QBS} on ajuripiirin kelluvan osion kuluttama virta, I_{LK} on ajuripiirin kelluvan osion vuotovirta, I_{LK_DIODE} on kytkennän diodin vuotovirta, I_{LK_CAP} on bootstrap-kondensaattorin vuotovirta, I_{DS-} on ajuripiirin biasvirta sen päällä ollessa ja T_{HON} on aika jonka ajuripiiri high-side-puoli on päällä. [4, s. 25.]

ΔV_{bs} saadaan laskettua kaavalla 4:

$$\Delta V_{bs} \leq V_{CC} - V_F - V_{GEmin} - V_{CEon} \tag{4}$$

Kaavassa 4 V_{CC} on piirin käyttöjännite, V_F on kytkennän diodin myötäsuuntainen jännite, V_{GSmin} on alin haluttu jännite MOSFETin hilalta lähteelle ja V_{CE} on MOSFETin yli häviävä jännite. [4, s. 26.]

Laskennassa käytettyjen muuttujien arvot ja tiedon lähde selviävät taulukosta 1.

Taulukko 1. Muuttujien arvot ja lähteet

Muuttujat ja niiden arvot	Lähde/selite
Q_G = 67 nC	datasheet IRF640N
I_{QBS} = 55 µA	datasheet IRS2004
I_{LK_CAP} = 0 µA	keraamisella kondensaattorilla
V_{CEon} = 1,3 V	datasheet IRF640N
V_{GEmin} = 8 V	datasheet IRS2004
Q_{LS} = 5 nC	kun kysymyksessä on alle 600 voltin MGD (MOSFET gate driver)
I_{LK} = 50 µA	datasheet IRS2004
I_{LK_DIODE} = 100 µA	alle 100 ns:n toipumisaikaisella diodilla
I_{DS-} = 0 µA	keraamisella kondensaattorilla
T_{HON} = 12,5 µs ja 5 µs	Lasketaan sekä 40 kHz:n että 100 kHz:n taajuudella. Digitaalisella signaalilla päälläoloaika puolet jakson ajasta.
V_{CC} = 12 V	piirin käyttöjännite
V_F = 1,04 V	datasheet MUR120
I_{gss} = 100 nA	datasheet IRF640N

Näillä arvoilla saadaan laskettu kaavasta 4 ΔV_{bs}:ssän ja kaavasta 3 Q_{tot}:in molemmille taajuuksille. Näillä saadaan laskettu C_{boot} kaavasta 2 molemmille taajuuksille.

$\Delta V_{bs} \leq 12\ V - 1{,}04\ V - 8\ V - 1{,}3\ V = 0{,}66\ V$

$Q_{tot} = 67\ nC + 5\ nC + (100\ nA + 55\ \mu A + 50\ \mu A + 100\ \mu A + 0 + 0) * 12.5\ \mu s = 7{,}456 * 10^{-8}\ C$

$Q_{tot} = 67\ nC + 5\ nC + (100\ nA + 55\ \mu A + 50\ \mu A + 100\ \mu A + 0 + 0) * 5\ \mu s = 7{,}303 * 10^{-8}\ C$

$C_{boot1} = 7{,}456 * 10^{-8}\ C / 0{,}66\ V = 1{,}130 * 10^{-7}\ F = 113{,}0\ nF$

$C_{boot2} = 7{,}303 * 10^{-8}\ C / 0{,}66\ V = 1{,}107 * 10^{-7}\ F = 110{,}7\ nF$

Laskettu arvo on kondensaattorin minimiarvo. Tämä on hyvä kertoa 15:lla. Tällä vältetään ohjainpiirin hajoaminen. [6, s. 3.]

$C_{boot1} = 113\ nF * 15 = 1{,}695\ \mu F$

$C_{boot2} = 110{,}7\ nF * 15 = 1{,}6605\ \mu F$

Laskettujen arvojen ja komponenttien saatavuuden perusteella päädytään käyttämään 1 µF:n elektrolyyttikondensaattorin ja 100 nF:n keraamisen kondensaattorin rinnankytkentää. Tällöin bootstrap-kondensaattorin arvoksi tulee 1,1 µF. Tämä mahdollistaa korkeampien yli 100 kHz:n taajuuden käyttämisen sekä on riittävä 40 kHz:n taajuudella.

3.5 Ohjainlaitteen käyttöliittymä

Ohjaimen käyttö tapahtuu viiden painonapin avulla. Painonapit on aseteltu riviin levyn alaosaan. Niiden järjestys vasemmalta oikealle on S4, S1, S2, S3 ja S5. Virran kytkennän jälkeen on mahdollista säätää taajuuden syötön kestoaikaa ja taajuusaluetta. Painonappi S4 vaihtaa tilaa ajan asetuksen ja taajuuden asetuksen välillä. LCD-näytöllä ajan säätötilassa lukee ”Aseta aika: xx h xx m”. Aika asetetaan painonapeilla S2 ja S3. Aika lisääntyy painamalla S3 ja vähenee S2:ta painettaessa. Painonapeissa S2 ja S3 on 150 ms:n viive ajan ja taajuuden säädön helpottamiseksi.

Taajuusalue vaihdetaan painamalla kerran S4. Näyttöön tulee teksti ”Aseta taajuus: 40 kHz”. Taajuus on aluksi 40 kHz. Taajuus alue säätyy 20 kHz:n välein 40 - 100 kHz. S3 lisää taajuutta ja S2 vähentää. Oikeasti ulostuleva taajuus heiluu hieman valitun taajuuden ympärillä. Heilunnan suuruus vaihtelee taajuuksittain. Heilunta on toteutettu mahdollisia käyttökohteita ajatellen. Tasainen taajuus voi aiheuttaa seisovia aaltoja, jotka voivan vahingoittaa esineitä, esimerkiksi ultraäänipesurissa.

Painonapilla S1 käynnistetään taajuuden syöttö. Tällöin näyttöön tulee teksti ”Aikaa jaljella: xx h xx min”. Asetetusta ajasta aletaan vähentää TIMER2:n antamien keskeytysten mukaan. Keskeytyksistä huolehtii ajastimen pinneihin 28 ja 29 kytketty 32,768 kHz:n kide. Kiteeltä saatu signaali jaetaan 1024:lla. Tällöin saamme 32 pulssia sekunnissa. 60 sekunnin kulunuttua vähennetään minuutti ajasta.

Painonapilla S5 keskeytetään taajuuden syöttö sekä nollataan aika. S5 antaa keskeytyksen Int2, joka keskeyttää ohjelman välittömästi. Tämän jälkeen palataan ajan asetustilaan. Painonapilla voidaan nollata aika myös säätövaiheessa.

3.5 Taajuuden tuotto

Taajuus tuotetaan käyttämällä kontrollerin Timer/Counter1:stä. Laskurille annetaan taajuudeksi piirin kellotaajuus (8 MHz) ilman esijakajaa. Taajuuden tuottamisessa käytetään nopeaa pulssinleveysmodulaatiota FPWM (Fast Pulse Width Modulation). Ajastin asetetaan moodiin 15. Tällöin ajastin laskee nollasta annettuun raja-arvoon ja aloittaa alusta. [7, s. 99 - 101.]

Ajastimen toiminnan asettamiseksi moodiin 15 täytyy asettaa seuraavat bitit WGM10, WGM11, WGM12 ja WGM13 ykkösiksi. Bitit WGM10 ja WGM11 ovat kaksi alinta bittiä TCCR1A-rekisterissä. Bitit WGM12 ja WGM13 ovat neljäs ja viides bitti TCCR1B-rekisterissä. TCCR1B-rekisterin alin bitti CS10 asetetaan ykköseksi. Tällä asetetaan jakajan arvoksi yksi. CS10-bitti käytetään myös ohjaamaan taajuuden tuottoa päälle ja pois. TCCR1A-rekisterin seitsemäs bitti COM1A0 asetetaan ykköseksi. Tämä valitsee laskurin ulostulomoodin. Laskuri kytkeytyy nyt pinniin 19 ja vaihtaa tämän tilaa laskurin mukaan. Näin saadaan aikaan tasainen kanttiaalto, jonka pulssisuhde on 50 %. [7, s. 107 - 110.]

Laskurilta saatu taajuus voidaan laskea kaavalla 5:

$$F_{out} = \frac{kellotaajuus}{2*N*(1+OCRnA)} \quad (5)$$

Kaavassa 5 kellotaajuus on piirin kellotaajuus 8 MHz. N on jakaja, jonka arvoina voi olla 1, 8, 64, 256 tai 1024. Jakajaksi valittiin yksi. OCRnA on kontrollerin rekisteri. Tämän rekisterin arvoja muuttamalla saadaan aikaan taajuuden säätö ja heilunta.

4 Ohjainlaitteen toteuttaminen

4.1 Piirilevyn valmistus

Piirikaavio ja piirilevy suunniteltiin Mentor Graphicsin PADS-ohjelmalla. Piirilevy päädyttiin valmistamaan kaksipuoleisena. Piirilevystä tehtiin GERBER-tiedostot jyrsintää varten. Tiedostot sisältävät tiedot piirilevyn vedoista kummaltakin puolelta, levyn mitat sekä poraustiedot. Nämä käsitellään CircuitCam-ohjelmistolla. CircuitCamilla luodaan jyrsintä ohjaavan ohjelman tarvitsemat tiedostot. Jyrsintä ohjataan BoardMaster-ohjelmalla. Jyrsintä tehtiin LPKF:n piirilevyjyrsimellä.

Jyrsinnän jälkeen huomattiin ajuripiirin korkeamman jännitteen eristeväli liian pieneksi levyn kuparoinnissa. Eristevälin tulisi olla vähintään 1.25 mm:ä 151 - 300 V:n jännitteillä [8, s. 15]. Ongelma ratkaistiin jyrsimällä käsikäyttöisellä jyrsimellä eristevälit riittävän suuriksi. Ennen komponenttien paikoilleen juottamista juotettiin levyyn tarvittavat läpiviennit. Tämä toteutettiin juottamalla vastuksien jaloista otettuja pätkiä läpiviennin reikiin. Peruskomponentit löytyivät koulun varastosta. Joitakin komponentteja jouduttiin tilaamaan Farnellilta.

4.2 Kontrollerin ohjelmointi

Mikrokontrolleri ohjelmointiin AVRISP2 ISP (In Serial Programming) -ohjelmointilaitteella. Ohjainkorttiin ei tehty ohjelmointiliitintä, joten ohjelmointia varten piti rakentaa erillinen alusta. Alusta rakennettiin verolevylle. Kontrolleri ohjelmoitiin käyttämään ulkoista kidettä kellopulssin aikaansaamiseksi, joten myös ohjelmointialustalle tarvitsi 8 Mhz:n kiteen ja 22 pF:n kondensaattorit. Lisäksi 1 kohmin ylösvetovastus reset-pinnille. Muut linjat kytkettiin suoraan ohjelmointilaitteelta kontrollerille.

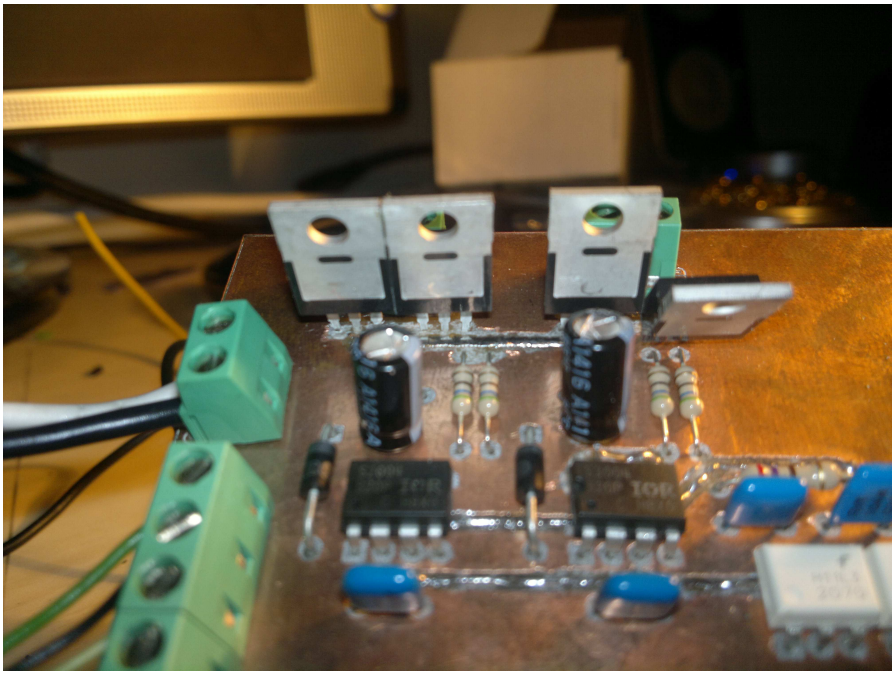
4.3 Kontrollerin ohjelmakoodi

Ohjelmakoodi on rakennettu tilakone rakenteella. Ohjelman alussa alustetaan aliohjelmissa sekuntien luku sekä ulostulotaajuus. Tämän jälkeen kirjoitetaan näytölle haluttu teksti. Näytön teksteistä huolehtii sitä varten tehty aliohjelma. Ohjelmassa on käytössä vahtikoira-ajastin. Tämä tarkkailee, ettei piirin ohjelma ole jumittunut. Vahtikoira-ajastimen arvo pitää aika ajoin nollata; muuten piiri käynnistetään uudelleen. Varsinaisessa ohjelmassa on neljä tilaa. Tilassa 0 asetetaan taajuuden ajolle haluttu aika painonappeja käyttämällä. Painonappien lukeminen on toteutettu silmukassa. Säädön helpottamiseksi napin lukemisessa käytetään lyhyttä viivettä. Mikäli mitään nappia ei paineta, ohjelma kiertää silmukkaa pää- ja aliohjelman välillä. Taajuuden asetus tapahtuu tilassa 2. Taajuuden asetus on toteutettu samalla tavalla kuin ajan asetus.

Tilassa 3 käynnistetään taajuuden syöttö kaiuttimille. Tämä tapahtuu alustamalla tarvittavat rekisterit. Samalla käynnistetään sekuntien laskeminen. Tämän jälkeen siirrytään tilaan 1. Tilassa 1 tarkastellaan ajan loppumista ja ajetaan aliohjelmaa, joka huolehtii taajuuden sopivasta heilunnasta. Tässä tilassa ollaan niin kauan, kunnes aika kuluu loppuun tai painetaan pysäytys painonappia, joka aiheuttaa keskeytyksen ja palauttaa ohjelman tilaan 0.

5 Ohjainlaitteen testaaminen ja muutokset

Kokoonpanon jälkeen tarkistettiin piirilevy silmämääräisesti juotosvikojen varalta. Silmämääräisessä tarkistuksessa huomatut tinasillat poistettiin. Tarkastelussa huomattiin vierekkäisten Fettien runkojen ottavan toisiinsa kiinni. Piirilevy oli jo niin pitkällä, että uutta levyä ei kannattanut valmistaa testejä varten. Ongelman ratkaisemiseksi päätettiin taivuttaa Fetit erilleen toisistaan kuvan 2 mukaisesti. Näin vältyttiin oikosuluilta ja testejä voitiin jatkaa.



Kuva 2. Fettit ovat vasemmalla yhdessä ja oikealla taivutettu erilleen.

Tämän jälkeen mitattiin yleismittarilla jänniteliittimet huomaamatta jääneiden oikosulkujen varalta. Suoria oikosulkuja ei löytynyt. Tämän jälkeen kytkettiin matalammat jännitteet yksi kerrallaan. Ensimmäiseksi kytkettiin 5 V:n jännite ja mitattiin, että se kytkeytyy oikeisiin paikkoihin. Tämän jälkeen kytkettiin 12 V:n jännite ja tarkistettiin mittaamalla, että jännite kytkeytyy vain oikeisiin paikkoihin. Lopuksi mitattiin vielä kaiuttimien syöttöjänniteliittimet, mutta jännitettä ei vielä kytketty.

5.1 Ohjelman testaus ja kehitys

Aluksi kontrollerille ohjelmoitiin testiohjelma näytön toiminnan ja painonappien testaamiseksi. Aluksi LCD-näyttölle ei tulostunut mitään järkevää. Vikaa etsittiin aluksi koodista, kun oltiin sitä mieltä, että koodi toimii oikein, vikaa alettiin etsiä piirilevystä. Viaksi paljastui huono liitos näytön jännitteen syötössä. Korjauksen jälkeen näytölle tulostui haluttu teksti. Testiohjelmassa painonappien painaminen vaihtoi näytön tekstiä. Painonappeja kokeiltaessa huomattiin yhden painonapin olevan kontrollerin mielestä kokoajan painettuna. Suurennuslasin avulla maapotentiaalissa olevan kuparoinnin ja painonapin johtimen välistä löytyi pieni tinasilta. Tinasillan poiston jälkeen painonapit toimivat halutusti.

Seuraavaksi piirille ohjelmoitiin jo lähes valmis ohjelma. Tässä vaiheessa myös sulakebitit ohjelmoitiin niin, että piiri käyttää ulkoista kidettä. Tämän ohjelma syötti signaalia ulos kontrollerista ja ajanlaskenta toimi. Signaalin taajuus tarkistettiin, jotta se vastaisi haluttua taajuutta. Samalla todettiin optoerottimien toiminta ja signaalin kulkeminen haluttuun ajuripiirin pinniin. Lisäksi tarkastettiin ajuripiirin sammutussignaalin toiminta. Mittaukset suoritettiin PicoScope pc-oskilloskoopilla. Ohjelman toiminnan toteamiseksi kytkettiin kaiuttimien syöttöjänniteliittimiin 20 V:n jännite ja kaiuttimen tilalle 1 kohmin vastus. Vastuksen yli olevaa jännitettä mitattiin oskilloskoopilla.

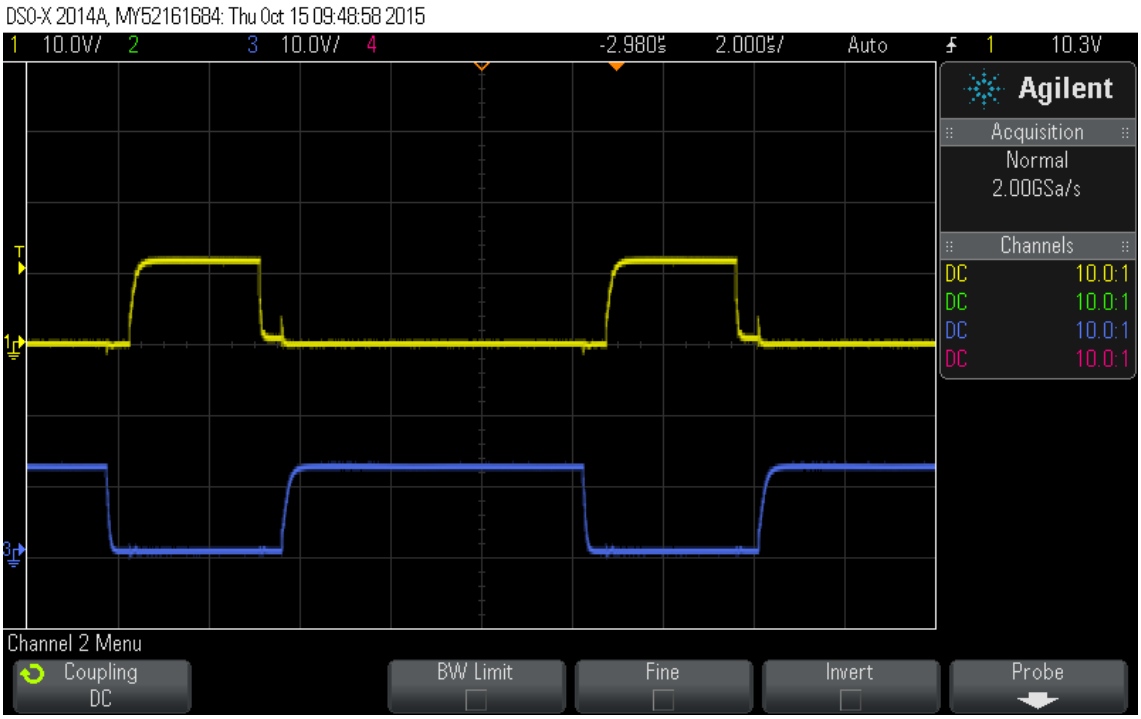
5.2 Piirilevyn mittaus ja korjaus

Kaikki vaikutti toimivalta, joten kytkettiin kaiuttimien syöttöjännitteeksi tarkoitettu 140 V:n jännite. Testissä käytettiin vielä pientä 1 kohmin kuormaa. Toiminnassa ilmeni epävakautta. Välillä ajo ei kytkeytynyt päälle ja välillä ohjelma keskeytyi kesken taajuuden syötön. Vian selvittämiseksi kytkettiin kaiuttimien syöttöjännitteeksi 20 V turvallisuuden vuoksi. Suurennuslasia apuna käyttäen vikaa ei löytynyt. Seuraavaksi mitattiin yleismittarilla jännitteiden kytkeytymiset. Kaikki jännitteet vaikuttivat kytkeytyvän oikein. Lopuksi tarkasteltiin oskilloskoopilla, ettei signaali kytkeydy väärään paikkaan. Mittauksissa huomattiin 8 MHz:n värähtelyä kuparikentässä. Tämän johdosta huomio kiinnittyi kiteeseen. Kiteen toisen jalan juotoksessa oli tinasilta kyseiseen kuparikenttään. Tinasillan poistamisen jälkeen oskilloskooppimittauksilla varmistettiin, ettei mikään signaali enää kytkeydy kuparikenttiin.

Korjauksien jälkeen piiri vaikutti toimivan hyvin pienellä kuormalla, joten siirryttiin testaamaan käyttäen Piezo -ultraäänikaiutinta kuormana. Tässä vaiheessa kuitenkin kaiuttimia ohjaavat Fetit räjähtivät vieden mukanaan osan kaiuttimen syöttöjännitteen johtimesta. Viaksi paljastui Fettien virheellinen asento. Fetit olivat taittuneet niin, että rungot olivat yhdessä kuten edellä olleessa kuvassa 2 vasemman puoleiset. Yleismittarilla mitattuna toinen Feteistä oli rikkoutunut. Korjauksessa päädyttiin vaihtamaan molemmat fetit ja ajuripiiri varmuuden vuoksi. Kaiuttimien syöttöjännitteen johdin korjattiin myös.

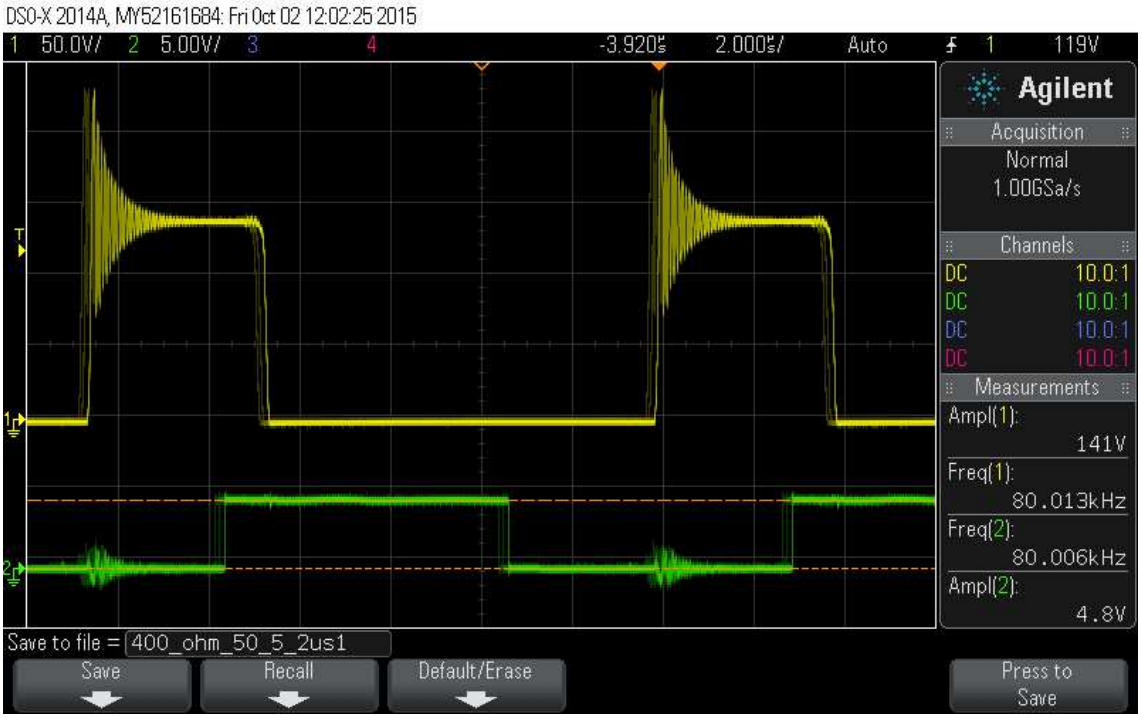
Signaalin kulkeutuminen kaiuttimen liittimelle testattiin vaiheittain oskilloskoopilla mittaamalla. Mittauksissa käytettiin aluksi 20 V:n syöttöjännitettä ja 1 kohmin vastusta. Järkeväksi todettiin testata toimintaa lisäämällä kuormaa vähän kerrallaan. Tämä toteutettiin säätövastuksella ja 140 V:n syöttöjännitteellä. Säätövastuksilla kuormitettuna ohjainlaite toimi oikein. Seuraavaksi kytkettiin Piezo -ultraäänikaiutin ja testattiin. Fetit paloivat jälleen. Tällä kerralla fettien lisäksi syöttöjännitteen johdin, sekä optoerottimet tuhoutuivat. Tässä vaiheessa pystyttiin toteamaan kontrollerin suojaus toimivaksi. Vioittuneet Fetit, ohjainlaite ja optoerottimet vaihdettiin uusiin ja syöttöjännitteen johdin korjattiin.

Opettaja Matti Fischer ehdotti vian johtuvan mahdollisesti Fettien olemisesta yhtä aikaa johtavassa tilassa. Tämän ajatuksen johdosta päädyttiin vaihtamaan Fettien etuvastukset 10-ohmisiksi. Mittaukset aloitettiin uudestaan vaiheittain 20 V:n syöttöjännitteellä ja säätövastuksella. Näiden mittausten jälkeen kytkettiin syöttöjännitteeksi 140 V:n ja kuormaksi säätövastus. Mittauksissa tarkistettiin Fettien gaten jännitteitä. Mittauksissa todettiin, että etuvastuksen vaihtamisen jälkeen vain toinen Fetti johtaa kerrallaan kuten alla olevasta kuvasta 3 huomataan.



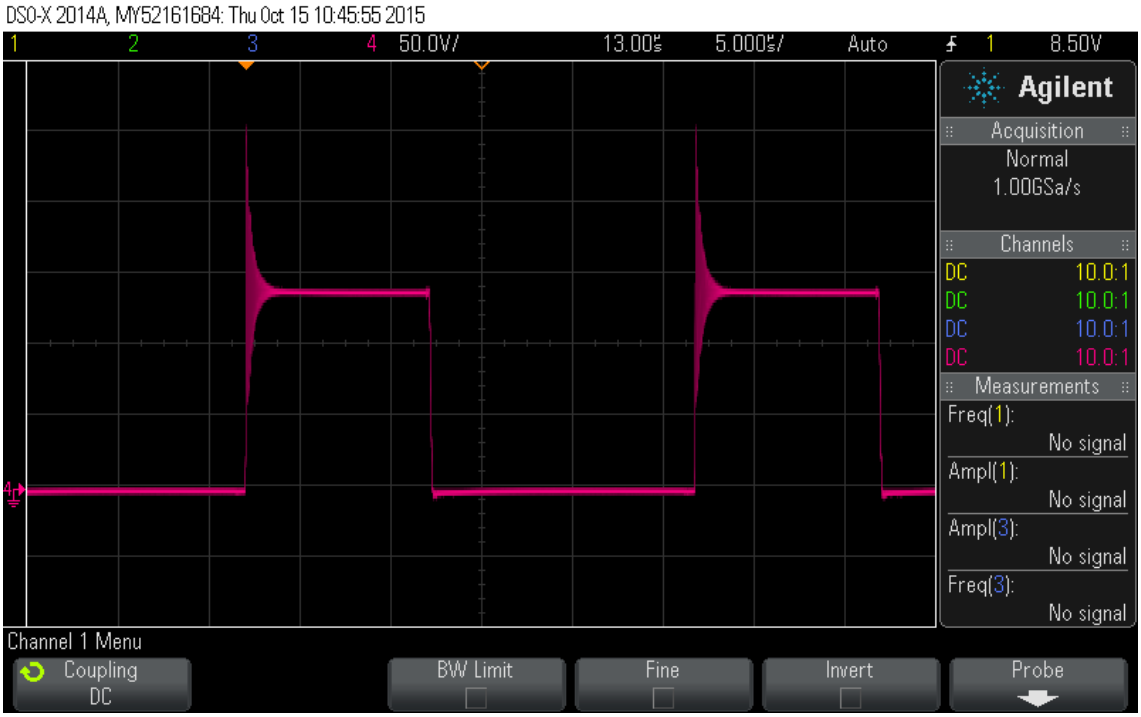
Kuva 3. Fettien gate-jännitteet.

Mittauksessa ilmeni, että pulssisuhde on vääristynyt. Tästä syystä tarkistettiin mittaamalla kontrollerin syöttämä taajuus ja säätövastuksen yli oleva jännite. Kontrollerin syöttämän signaalin pulssisuhde oli 50 %, kuten oli suunniteltu. säätövastuksen yli olevan jännitteen suhde oli muuttunut jonkin verran kuten alla olevasta kuvasta 4 käy ilmi. Signaalien taajuudet ovat kuitenkin säilyneet yhtenäisinä.



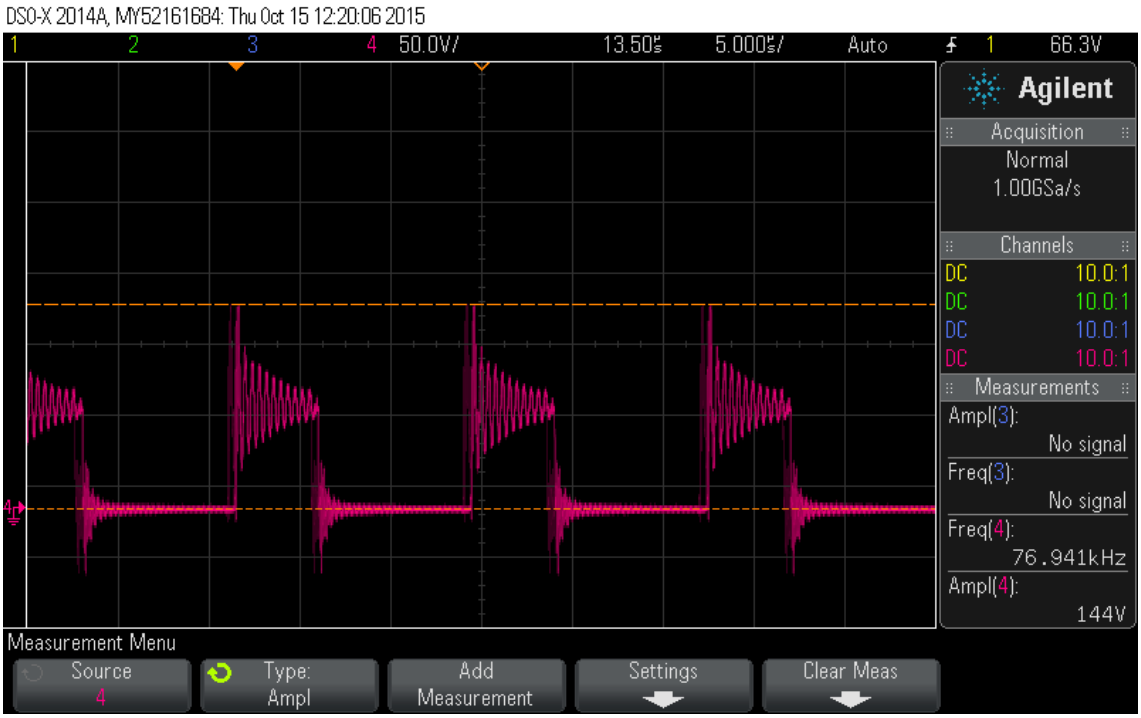
Kuva 4. Vihreällä kontrollerin syöttämä signaali. Keltaisella on säätövastuksen yli oleva jännite.

Kaikki näytti toimivan, joten kytkettiin kuormaksi Piezo -ultraäänikaiutin. Fetit kestivät kaiuttimen. Kaiuttimen yli olevaa signaalia haluttiin mitata eri taajuuksilla käynnistysvärähtelyn käyttäytymisen takia. 40 kHz:n taajuudella värähtely on lyhytkestoista, kuten kuvasta 5 nähdään.



Kuva 5. Kaiuttimen yli oleva jännite taajuudella 40 kHz.

80 kHz:n taajuudella värähtely ei ehdi loppua ennen päälläolon loppua, kuten seuraavasta kuvasta 6 nähdään. Tätä voisi saada jonkin verran pienenemään syöttöjännitettä suodattamalla.



Kuva 6. Kaiuttimen yli oleva jännite taajuudella 80 kHz.

Mittauksien jälkeen testattiin suurempaa Piezo -ultraäänikaiutinta. Tämä kuitenkin aiheutti liikaa häiriötä, joka johti kontrollerin resetoitumiseen.

6 Yhteenveto

Työn tavoitteena oli parannella aikaisemmin tehtyä ultraäänikaiuttimien ohjainlaitetta ja suojata ohjauselektroniikkaa vikatilanteessa. Työhön kuului elektroniikka- ja piirilevynsuunnittelu, piirilevynvalmistus, kokoonpano, ohjelmointi sekä testaus. Ohjainlaitetta paranneltiin enimmäkseen käyttöliittymän ja ohjainkontrollerin suojauksen osalta.

Työssä haasteina oli suojaukseen sopivien komponenttien valinta, ohjelmointi ja vianhaku. Suojauskomponenttien valinnassa onnistuttiin hyvin. Ohjelmoinnissa haasteena oli rekisterien muokkaaminen oikein ja sulakebittien oikeiden asetuksien löytäminen. Vianhaku sujui suurilta osin tutuilla menetelmillä.

Ohjainlaite toimii niin kuin oli tavoite. Ohjainlaite vaatii vielä jatkokehittelyä komponenttien sijoittelun (Fetit) sekä häiriöiden poistamiseksi. Myös ohjelmakoodia voisi kehittää vielä yleiskäyttöisemmäksi.

Lähteet

1 Peltonen Hannu, Perkkiö Juha, Vierinen Kari. Insinöörin (AMK) fysiikka. Osa II. Lahti: Lahden Teho-Opetus Oy, 2000.

2 INFORMATION ON ANIMAL REPELLERS. (WWW-dokumentti.) <*http://www.contattoitalia.it/TEORIA-EN.pdf*>.2009. Luettu 28.10.2015.

3 MICROPROCESSOR COMPATIBLE SCHMITT TRIGGER OPTICALLY COUPLED ISOLATOR. (WWW-dokumentti.) <*http://www.farnell.com/datasheets/90941.pdf*>. 2008. Luettu 19.01.20016.

4 Juha Vaaksio. Ultraäänipesurin suunnittelu. (WWW-dokumentti.) <*http://www.theseus.fi/bitstream/handle/10024/14433/lopputyo.pdf?sequence=1*>. 2010. Luettu 19.01.2016.

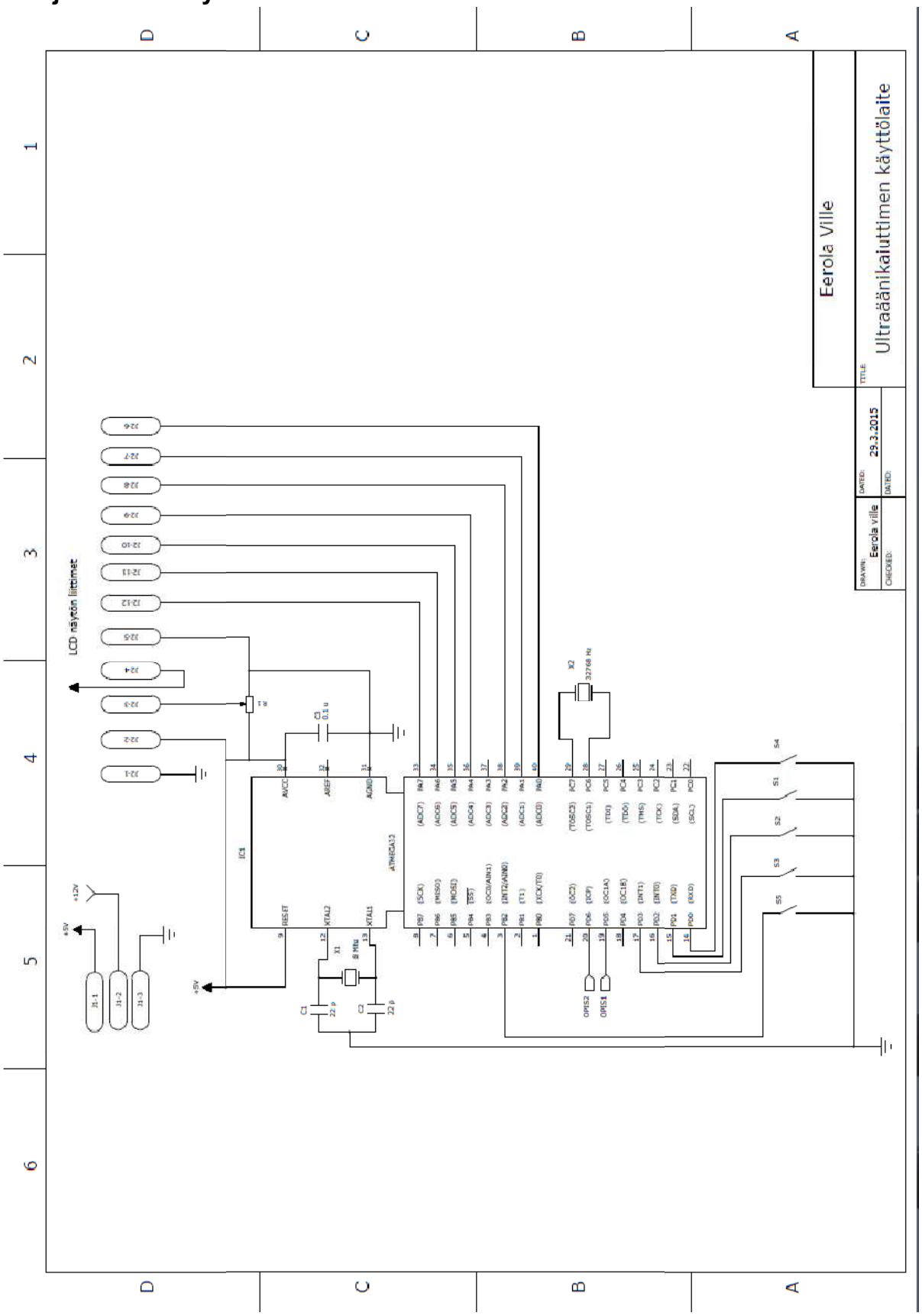
5 IRS2004(S)PDF. (WWW-dokumentti.) <*http://www.irf.com/product-info/datasheets/data/irs2004pbf.pdf*>. 2006 Luettu 19.01.2016.

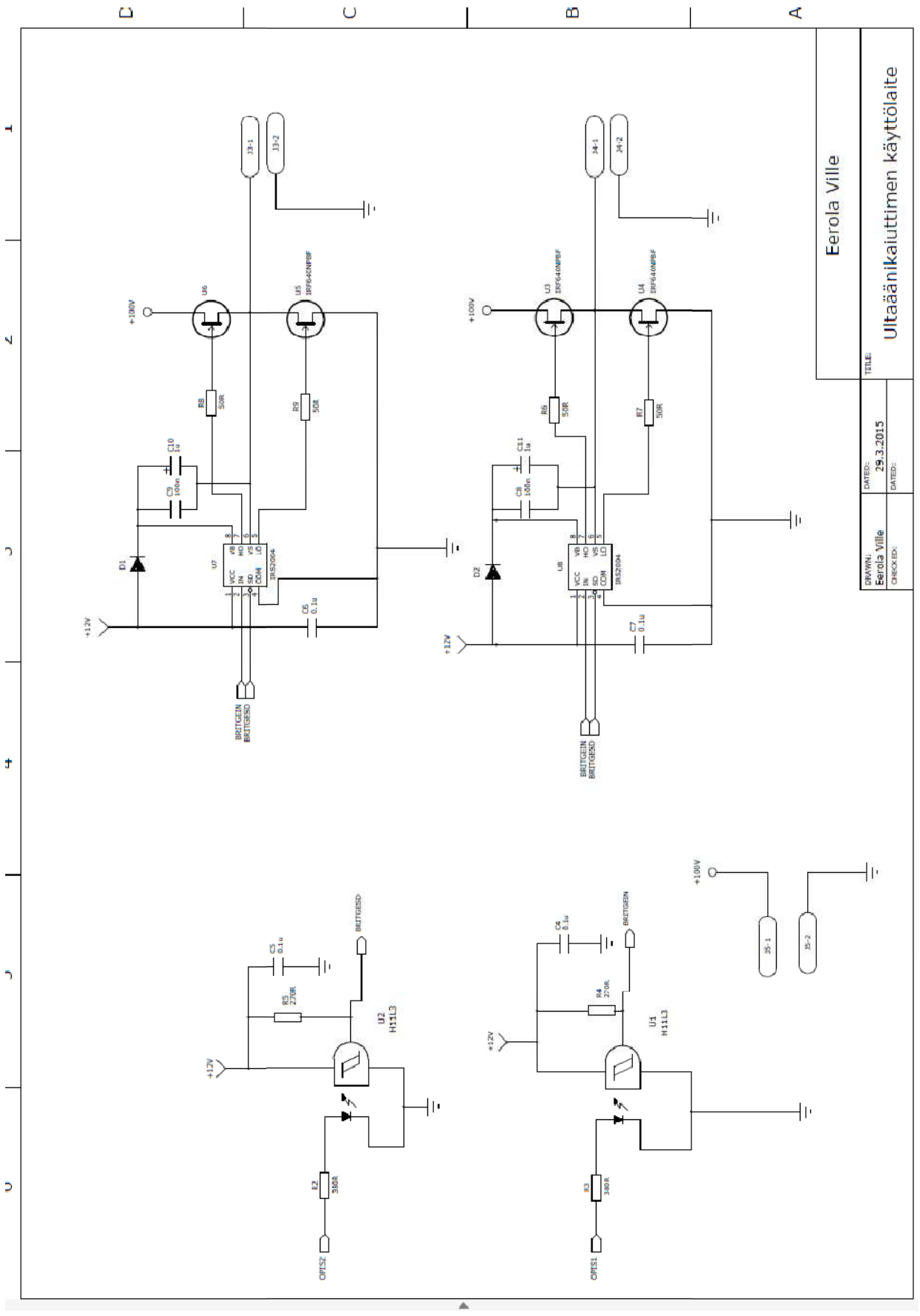
6 Jonathan Adams. Bootstrap Component Selection For Control IC's (WWW-dokumentti) <*www.irf.com/technical-info/designtp/dt98-2.pdf*>. 2001. Luettu 9.2.2016

7 ATmega32A – Datasheet. (www-dokumentti). <*http://www.atmel.com/Images/doc2503.pdf*>. 2011. Luettu 10.2.2016.

8 Leinonen Tomi. STRESSTEST 1000 – Testausjärjestelmä mittalaitteiden testaamiseen (www-dukumentti). <*https://theseus.fi/bitstream/handle/10024/40928/opinnaytetyo_D4433.pdf?sequeseque=1*>. 2012. Luettu 10.2.2016

Ohjainlaitteen kytkentäkaavio




Eerola Ville
Ultaäänikaiuttimen käyttölaite
DRAWN: Eerola Ville
DATED: 29.3.2015
CHECKED:
DATED:
TITLE:
+100V
+12V
BRITGEIN
BRITGESD
OPTS1
OPTS2
IRS2004
IRF640NPBF
H11L3

Ohjelmakoodi

```
/* Atmega32, 8MHz. */

#include <avr/io.h>
#include <avr\interrupt.h>
#include <avr\wdt.h>
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <util/delay.h>
#include "lcd_tat.h"

/* Määrittelyt */
#define F_CPU 8000000UL
/* Funktioiden prototyypit */

void init_aika();
void init_taajuus();
void init_sekunnit();
void aseta_aika();
void aseta_taajuus();
void aani();
void naytto();

/* Muuttujia */
volatile uint16_t aika = 0;
uint8_t sekunti = 0;
uint8_t minuutit = 0;
uint8_t tunnit = 0;
volatile unsigned char run = 0;
unsigned char taajuus = 40;
volatile uint8_t heilunta = 0;
char merkkijono[16];
char merkkijono_2[16];
unsigned char merkkijono3[16];
unsigned char merkkijono4[16];

/* Pääohjelma alkaa */
int main(void)
{
init_sekunnit(); // Timer2:sen initialisointi, lasketaan sekunteja

init_aika(); // Ajanluvun initialisointi
init_taajuus(); // Timer1:sen initialisointi, taajuus tällä
cli();
WDTCR = (1 << WDE) | (1 << WDP2) | (1 << WDP1) | (1 << WDP0);
//Watchdog päälle, suurin jakaja -> väli noin 2.1s
sei(); // Sallitaan keskeytykset globaalisti
 LCD_init(1, 0, 0);

while(1)
{
naytto();

if(run == 0)
{
```

```
_delay_ms(20);
aseta_aika();
wdt_reset(); // Resetoidaan watchdog
}

else if(run == 2)
{
_delay_ms(20);
aseta_taajuus();
wdt_reset(); // Resetoidaan watchdog
}

else if(run == 3)
{
if(aika == 0) // Jos aika = 0, ei käynnistetä taajuuden ajoa.
{
TCCR1B &= ~(0 << CS10);
TCCR2 &= 0xF8;
sekunti = 0;
run = 0;
}

else
{
TCCR2 |= (1 << CS22) | (1 << CS21) | (1 << CS20) ;
// Käynnistetään timer2
TCCR1B |= (1 << CS10);
// Käynnistetään timer1
PORTD &= 0xBF ;
// Enable ajuripiirille
run = 1;
sekunti = 0;
}
}

else
{
aani();

if(aika == 0) // Jos aika on 0, niin timer1 pois päältä
{
TCCR1B &= ~(1 << CS10); // = ei taajuutta kaiuttimille.

PORTD |= 0x40; // Ajuripiirin enable nollaksi.
run = 0;
heilunta = 0;
TCCR2 &= 0xF8;
sekunti = 0;
}
}
}
}

void init_sekunnit() // Timer2:sella reaaliaikakello
{
// Ulkoiselta 32.768 kiteeltä kello
ASSR = (1 << AS2); // Kellon lähteeksi TOSC1, mihin kide.
TCCR2 = (1<<WGM21) | (0 << CS22) | (0 << CS21) | (0 << CS20);
// Jakajaksi 1024 -> 32768/1024 = 32
// CTC - Mode (Clear Timer on Compare)
```

```
TIFR = (1<<OCF2);
// Asetetaan OCF2 ->nollataan keskeytykset
TIMSK = (1<<OCIE2);
// Timer/Counter2 Output Compare Match Interrupt Enable
OCR2 = 32;
// Vertoarvoksi 32, kun laskettu on kulunut 1 sekunti.
while(ASSR&(1<<OCR2UB));
// Odotetaan rekisterin päivittymistä
sei(); // Globaalit keskeytykset päälle
}

void init_taajuus()
{
            cli();
TCCR1A = (1<<COM1A0) | (1<<WGM11) | (1<<WGM10);
// COM1A0 asettaa duty cycleksi 50%, kun käytetään ajastimen moodia 15.
// WGM10-13 asettavat moodin.
TCCR1B = (0<<CS10) | (1<<WGM13) | (1<<WGM12);
// CS10 asettaa jakajan, tässä 1.
sei();
}

void init_aika()
{
PORTD = 0x4F;
DDRD = 0x60; // PD alustukset

PORTB =0x04;
DDRB = 0x00;

GICR |= 1<<INT2; // INT2 sallittu

}

void aseta_aika()
{
while(run == 0)
{
naytto();

if(bit_is_clear(PIND,2)) // Niinkauan kuin nappi on painettu, lisätään aikaa
{
aika--;
naytto();
_delay_ms(150); // Joka 150:s millisekunti
wdt_reset(); // Resetoidaan watchdog
}

else if(bit_is_clear(PIND,3))
{
aika++;
naytto();
_delay_ms(150); // Tähänkin 150ms viive
wdt_reset(); // Resetoidaan watchdog
}

else
_delay_ms(10);

if(bit_is_clear(PIND,0))
```

```
{
wdt_reset(); // Resetoidaan watchdog
_delay_ms(300);
run = 2;
}

if(bit_is_clear(PIND,1))
{
_delay_ms(300);
wdt_reset(); // Resetoidaan watchdog
run = 3;
}

}
}
void aseta_taajuus()
{
while(run == 2)
{
naytto();

if(bit_is_clear(PIND,2))
{
taajuus -= 20;
wdt_reset(); // Resetoidaan watchdog
naytto();
_delay_ms(150); // Joka 150:s millisekunti
if(taajuus == 20)
taajuus = 40;
}

else if(bit_is_clear(PIND,3)) // keskeytys int1:sta = aikaa vähemmäksi
{
taajuus += 20;
wdt_reset(); // Resetoidaan watchdog
naytto();
_delay_ms(150); // Joka 150:s millisekunti
if(taajuus == 120)
taajuus = 100;
}

else
_delay_ms(10);

if(bit_is_clear(PIND,0))
{
wdt_reset(); // Resetoidaan watchdog
_delay_ms(300);
run = 0;
}

if(bit_is_clear(PIND,1))
{
_delay_ms(300);
wdt_reset(); // Resetoidaan watchdog
run = 3;
}
}
}
```

```
void aani()
{
heilunta += 1;
if(taajuus == 40)
{

if(heilunta <= 40)
{
 OCR1A = 98;
_delay_ms(10);
}

else if(heilunta <= 80)
{
 OCR1A = 99;
_delay_ms(10);

}

else if(heilunta <= 120)
{
 OCR1A = 100;
_delay_ms(10);
}

else if(heilunta <= 160)
{
 OCR1A = 101;
_delay_ms(10);
}

else
{
 OCR1A = 102;
_delay_ms(10);

}
 if(heilunta == 200)
{
            heilunta = 0;
}
}

else if(taajuus == 60)
{
if(heilunta <= 40)
{
 OCR1A = 64;
_delay_ms(10);
}

else if(heilunta <= 80)
{
 OCR1A = 65;
_delay_ms(10);
}

else if(heilunta <= 120)
{
 OCR1A = 66;
```

```
_delay_ms(10);
}

else if(heilunta <= 160)
{
 OCR1A = 67;
_delay_ms(10);
}

else
{
 OCR1A = 68;
_delay_ms(10);
}
if(heilunta == 200)
{
            heilunta = 0;
}
}

else if(taajuus == 80)
{

if(heilunta <= 40)
{
 OCR1A = 49;
_delay_ms(10);

}

else if(heilunta <= 80)
{

 OCR1A = 50;
_delay_ms(10);

}

else
{
 OCR1A = 51;
_delay_ms(10);

}
if(heilunta == 160)
{
            heilunta = 0;
}
}


else if(taajuus == 100)
{
if(heilunta <= 40)
{
 OCR1A = 40;
_delay_ms(10);

}
```

```
else if(heilunta <= 80)
{
 OCR1A = 41;
_delay_ms(10);
}

else
{
 OCR1A = 42;
_delay_ms(10);
}

if(heilunta == 160)
{
            heilunta = 0;
}
}

}

void naytto()
{
wdt_reset(); // Resetoidaan watchdog
if(aika == 0)
{
tunnit = 0;
minuutit = 0;
}
else
{
minuutit = aika % 60;
tunnit = (aika - minuutit)/60;
}

if (run == 1)
{
sprintf(merkkijono, "Aika jaljella: ");
sprintf(merkkijono_2, "  %d h  %d min" , tunnit , minuutit);

unsigned char *merkkijono3 = (char*) merkkijono;
unsigned char *merkkijono4 = (char*) merkkijono_2;
LCD_clear();
LCD_writeString(merkkijono3);
LCD_setCursorXY(0, 1);
LCD_writeString(merkkijono4);

wdt_reset(); // Resetoidaan watchdog
}

else if (run == 0)
{
sprintf(merkkijono, "  Aseta aika: ");
sprintf(merkkijono_2, "  %d h  %d min" , tunnit , minuutit);

unsigned char *merkkijono3 = (char*) merkkijono;
unsigned char *merkkijono4 = (char*) merkkijono_2;

LCD_clear();
LCD_writeString(merkkijono3);
LCD_setCursorXY(0, 1);
```

```
LCD_writeString(merkkijono4);
wdt_reset(); // Resetoidaan watchdog
}

else
{
sprintf(merkkijono, " Aseta taajuus: ");
sprintf(merkkijono_2, "    %d KHz " , taajuus);

unsigned char *merkkijono3 = (char*) merkkijono;
unsigned char *merkkijono4 = (char*) merkkijono_2;

LCD_clear();
LCD_writeString(merkkijono3);
LCD_setCursorXY(0, 1);
LCD_writeString(merkkijono4);
wdt_reset(); // Resetoidaan watchdog
}
}


ISR(INT2_vect) //  (INT2)
{
TCCR1B &= ~(1 << CS10); // = ei taajuutta kaiuttimille.
PORTD |= 0x40; // Ajuripiirin enable nollaksi.
TCCR2 &= 0xF8;
run = 0;
heilunta = 0;
aika = 0;
sekunti = 0;
}

ISR(TIMER2_COMP_vect) // Sekuntien laskun keskeytysvektori
{
sekunti++; // lisätään joka sekunti yksi sekunti
if(sekunti >= 60) // Kun kulunut 60 sekuntia = 1 minuutti.
{
aika--; // Vähennetään aikaa yhdellä.
sekunti = 0; // Nollataan sekunnit.
}
}
```