Izrada 8-kanalnog uređaja za paralelno snimanje zvuka

Sabolski, Ivan

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

IZRADA 8-KANALNOG UREĐAJA ZA PARALELNO SNIMANJE ZVUKA

Diplomski rad

Ivan Sabolski

Osijek, 2017.

SADRŽAJ

1. UVOD

Cilj ovog diplomskog rada je napraviti maketu za paralelno snimanje zvuka pomoću 8 kanala. Maketa će biti korištena u edukacijske svrhe na kolegiju Sonarsko računarstvo. Pomoću 8 kanala moguće je snimiti zvuk iz jedne prostorije i odrediti točan položaj izvora zvuka. Također je moguće od više izvora zvuka izolirati točno određeni izvor i prigušiti ostale. Za ostvarenje makete bit će potrebno napraviti prototip dva kanala na *protoboard-u*(razvojna pločica), te testirati kanale. Zatim spojiti svih 8 kanala i ponovno testirati. Ako je testiranje uspješno slijedi dizajniranje tiskane pločice u programskom paketu Eagle. Točna dimenzija pločice treba biti 10x10 cm. Pločica može biti višeslojna, te treba imati mogućnost spajanja na Arduino platformu. Za kolegij Sonarsko računarstvo koristit će se pločica izrađena u SMT (engl. *Surface-mount technology*) tehnologiji, te će se na nju spajati Arduino Nano platforma. Ideja je spojiti maketu sa Arduinom, zatim spojiti SD (engl. *Secure Digital*) karticu i snimati zvuk. SD kartica bi služila za pohranu snimljenih zvukova. Najteži dio izrade makete je sama izrada, točnije lemljenje SMT komponenata na gotovu pločicu jer su komponente jako male. Sam rad će se sastojati od koraka izrade makete, dakle od samog prototipa pa sve do gotove pločice. Bit će prikazane slike i kodovi s kojima će se testirati maketa.

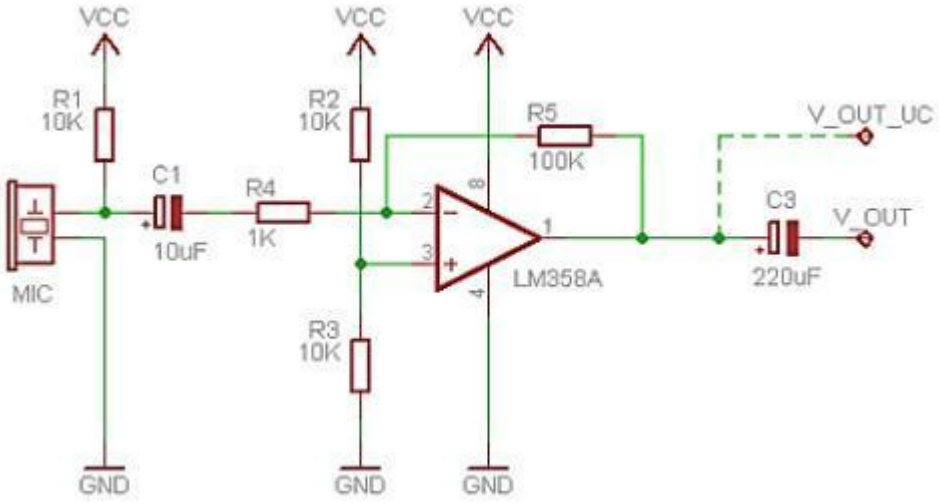
1.1. Zadatak završnog rada

U ovom diplomskom radu potrebno je izraditi maketu za paralelno snimanje zvuka pomoću 8-kanala. Dimenzije pločice trebaju biti veličine 10 x 10 cm. Namjena ove makete je u edukacijske svrhe za studente diplomskog studija na kolegiju Sonarsko računalstvo.

2. POTREBNE KOMPONENTE I ALATI

U ovom radu potrebno je izraditi sustav za prikupljanje zvuka pomoću Arduino Nano mikroupravljača i kapacitivnog electret mikrofona. Koristit će se dvostruko operacijsko pojačalo LM358, koje u sebi ima dva operacijska pojačala. Od toga će se jedno koristiti za mikrofonsko predpojačalo, a drugo za analogni anti-aliasing filter. Anti-aliasing filter je nisko propusni filter do frekvencije fs/2. Nakon pojačala i filtera signal je potrebno dovesti na 8-kanalni 12-bitni MCP3208 A/D pretvornik sa SPI (engl. *Serial Peripheral Interface*) serijskim sučeljem.

Na slici 2.1. je prikazano mikrofonsko predpojačalo pomoću operacijskog pojačala.



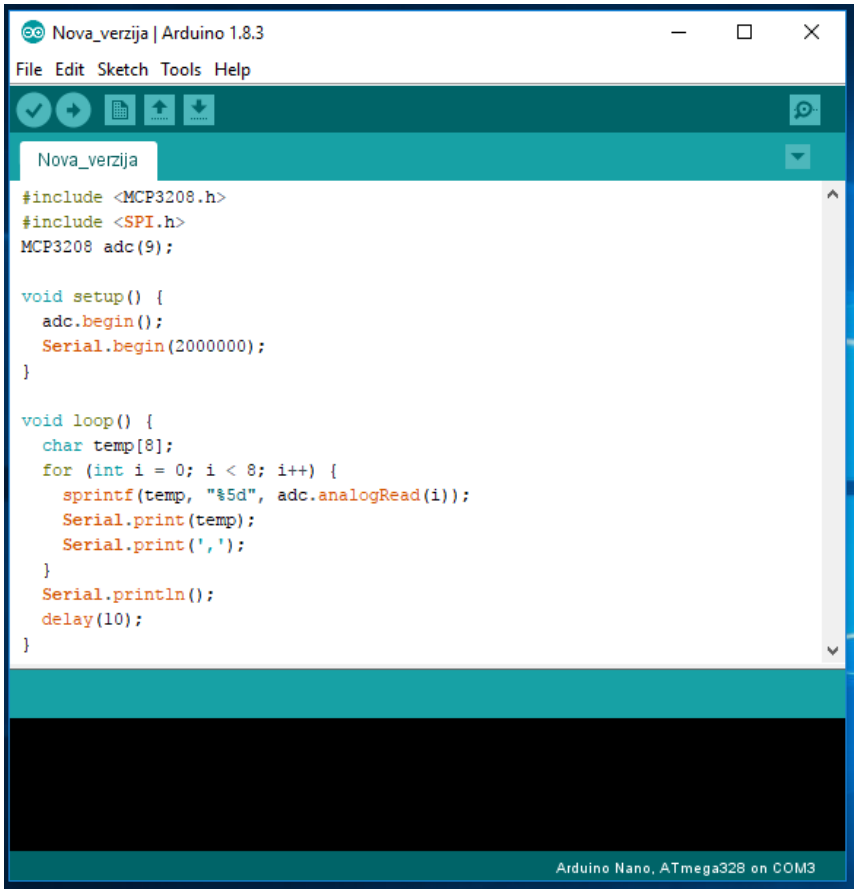
Sl. 2.1. *Mikrofonsko predpojačalo*

Prema slici 2.1. će se u radu spajati pojedini kanali, s korekcijom da će se umjesto otpornika R5 koristiti potenciometar 200-470 kΩ. Ostali otpornici će biti istih vrijednosti kao na slici. Isto tako koristit će se i kondenzatori s tim da će se izlaz iz LM358 spojiti izravno na A/D pretvornik bez kondenzatora C3.

U daljnjem tekstu bit će opisane sve komponente posebno s priloženim shemama.

2.1. Arduino razvojno okruženje

Arduino je ime za otvorenu računalnu i programsku platformu koja omogućava konstruktorima i dizajnerima stvaranje uređaja i naprava koje omogućavaju spajanje računala s fizičkim svijetom, tj. stvaranje interneta stvari (engl. *Internet of Things*). Arduino je stvorila talijanska tvrtka SmartProjects 2005 godine. Koriste 8-bitne mikrokontrolere Amtel AVR da bi stvorili jednostavnu malu i jeftinu platformu s kojom bi mogli lakše povezivati računala s fizičkim svijetom. [1] Arduino se sastoji od dva dijela, prvi dio je mikroupravljač koji predstavlja sklopovlje i sastoji se od dijelova kao što su procesor, memorija i ulazno izlazne jedinice, a drugi dio je programski dio. Programski dio ili Arduino IDE (engl. *Integrated Development Environment*) je višeplatformska aplikacija napisana u programskom jeziku Java. Arduino IDE se sastoji od 2 glavne funkcije, a to su *setup()* i *loop()*. *Setup()* je funcija koja se samo jednom izvršava prilikom pokretanja programa, a u njoj se obično definiraju varijable koje se kasnije koriste. *Loop()* funcija se poziva nakon *setup()* fukncije i ona je glavna fukcija svakog Arduino programa. U njoj se nalaze naredbe kojima se definira što Arduino treba raditi.



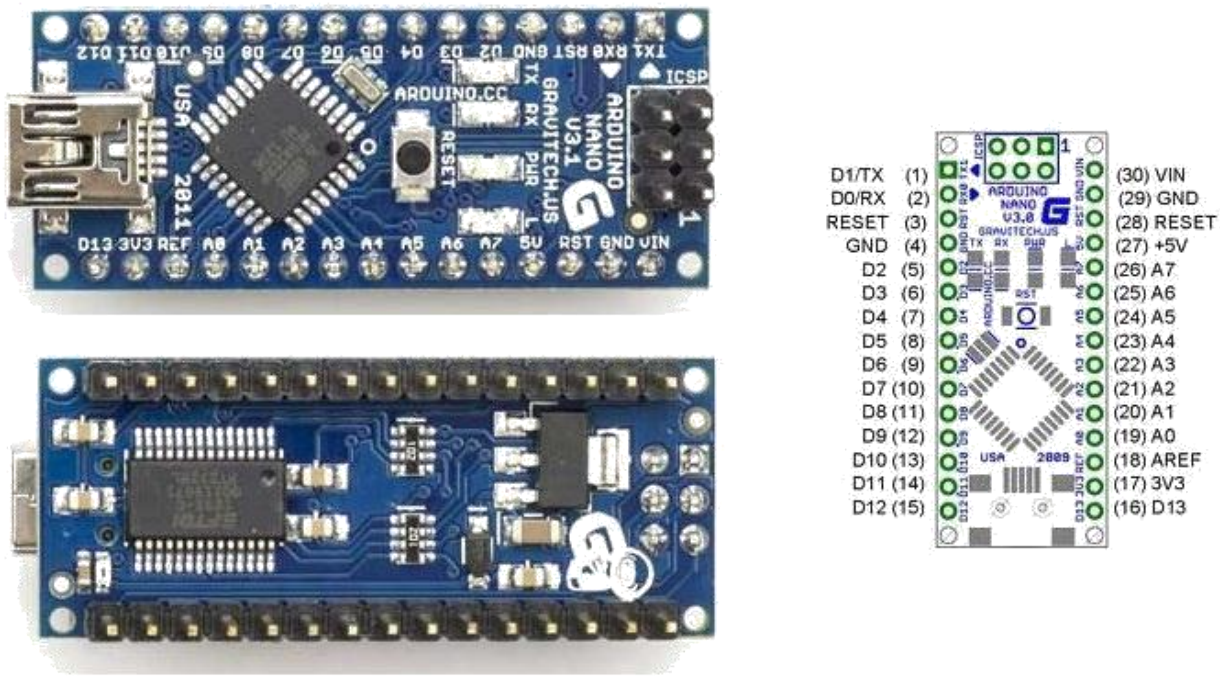
Sl. 2.2. *Arduino IDE*

2.1.1. Arduino Nano

Arduino Nano je mala, potpuna maketa bazirana na temelju Atmega328 (Arduino Nano 3.x) ili Atmega168 (Arduino Nano 2.x). Radni napon za logičke sklopove je 5 V, a preporučeni ulazni napon je između 7 i 12 V. Mali Arduino Nano može daleko više isporučiti, te mu je limit ulaznog napona do 20 V. DC struja po ulazu i izlazu je 40 mA. Ima 14 digitalnih ulaza/izlaza, od kojih 6 pružaju PWM (engl. *Pulse Width Modulation*) izlaz. Osim toga, analognih ulaznih pinova ima 8.

Što se tiče brze memorije, verzija Atmega128 ima 16 KB, a Atmega328 ima 32 KB, od kojih 2 KB koristi *bootloader*. Još jedna bitna stavka je da mikrokontroler ima brzinu takta obrade podataka od 16 MHz. [2]

Arduino Nano ima niz objekata za komunikaciju s računalom, drugim Arduinom ili drugim mikroupravljačem. Nano pruža UART TTL (5 V) serijsku komunikaciju, koja je omogućena na digitalnom pinu 0 (RX) i 1 (TX).



Sl. 2.3. *Arduino Nano*

Na slici 2.3. prikazan je Arduino Nano s obje strane, te njegov raspored pinova.

2.2. Electret Mikrofon

Electret mikrofon je vrsta elektrostatičkog mikrofona koji je baziran na kondenzatoru (engl. *capacitor-based*), što znači da uklanja potrebu za polarizirajućim napajanjem koristeći trajno punjivi materijal. [3] Electret je stabilni dielektrični materijal s trajno ugrađenim statičkim nabojem. Ime electret dolazi od pojma elektrostatičan i pojma magnet (engl. *Electrostatic and magnet*). U ovom slučaju koristit će se mikrofon 9x7 mm i prikazan je na slici 2.4.

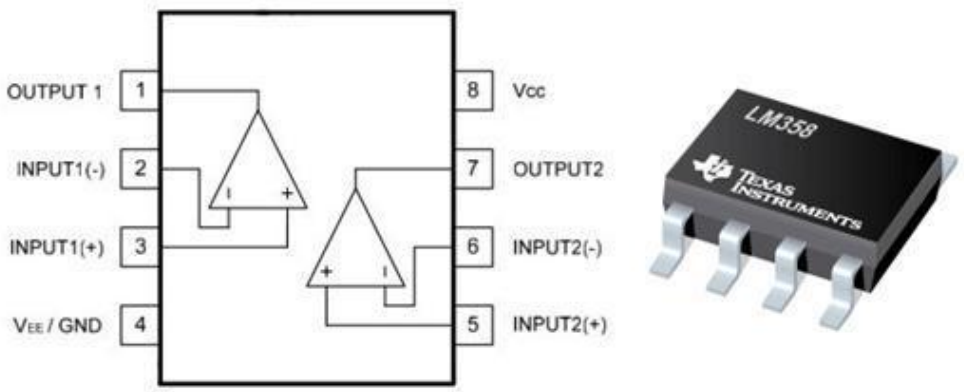
Sl. 2.4. *Electret mikrofon*

Na trećoj slici se vide kontakti mikrofona, prvi kontakt je izlaz ili plus kontakt, a drugi je uzemljenje. Na slici je prikazan mikrofon u SMT tehnologiji pošto nema izvedene nožice. Tako zauzima manje prostora.

2.3. Dvostruko operacijsko pojačalo

Operacijsko pojačalo je elektronički sklop ili integrirana elektronička komponenta koja ima mogućnost pojačanja izmjeničnog ili istosmjernog napona diferencijalnim ulazom, jednim izlazom i vrlo velikim naponskim pojačanjem kontroliranim negativnom povratnom vezom koja u cijelosti određuje njegova prijenosna svojstva. Kod suvremenih operacijskih pojačala, električni otpor na ulazu može se smatrati beskonačnim, dok je izlazni električni otpor zanemarivo mal. [4]

Dvostruko operacijsko pojačalo ili LM358 ima u sebi 2 operacijska pojačala kako i sam naziv kaže. Prvo operacijsko pojačalo će se koristiti za mikrofonsko predpojačalo, dok će se drugo koristiti na analogni anti-aliasing filter.



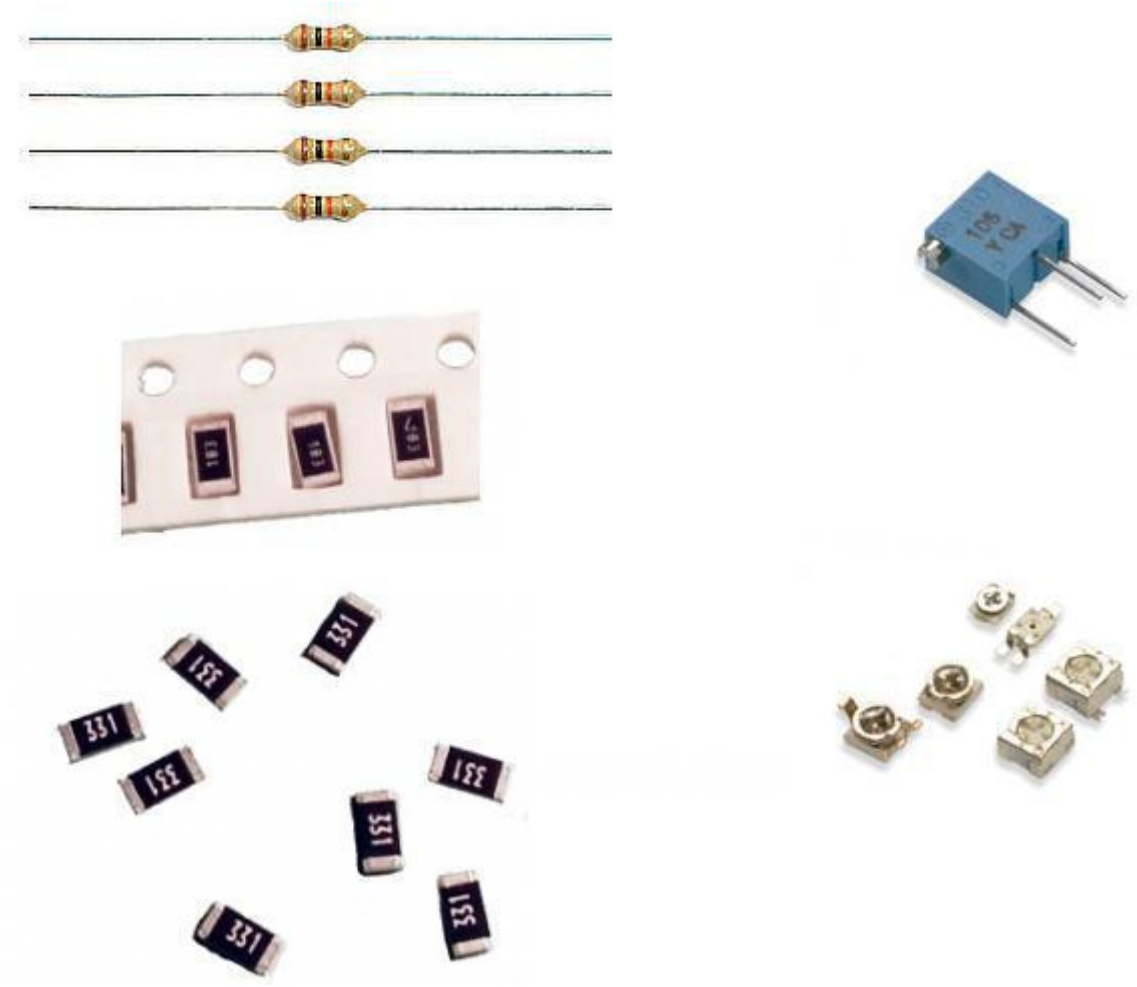
Sl. 2.5. *LM358*

Na slici 2.5. se vidi raspored pinova dvostrukog operacijskog pojačala. Pin 2 i 3 su ulazi u prvo pojačalo, a 5 i 6 su ulazi u drugo pojačalo. Izlazi su pin 1 i pin 7. I naravno pin 8 je +5 V, a 4 je uzemljenje. Također se vidi SMT izvedba LM385 komponente.

2.4. Otpornici

Otpornik je elektronska komponenta koja pruža otpor struji, stvarajući pritom pad napona između priključaka u skladu s Ohmovim zakonom. [5]

Prema slici 2.1. se vidi kako će trebati 4 otpornika po jednom kanalu i jedan promjenjivi otpornik (engl. *Trimmer potentiometer*). 3 komada će biti od 10 KΩ, jedan od 1 KΩ i trimer će biti 200-470 kΩ. U izradi prototipa koristit će se otpornik normalne veličine, a pri izradi tiskane pločice sve komponente će biti u SMT tehnologiji, tako i otpornici.



Sl. 2.6. *Fiksni i promjenjivi otpornici (trimmer)*

Na slici 2.6. na lijevoj strani se nalaze otpornici, s tim da su donji u SMT izvedbi, što znači da su puno sitniji nego li ovi gore. Desno se nalazi promjenjivi otpornik, također je donja slika u SMT tehnologiji.

2.5. Kondenzatori

Kondenzator je spremnik statičkog elektriciteta i energije električnog polja. Električno polje nastaje u prostoru između dva vodljiva tijela zbog razdvajanja električnog naboja. Električni kapacitet je karakteristična veličina kondenzatora i označava se sa C, a izražava u faradima (F). Kako je kapacitet od 1 farada jako velik, kondenzatori koje susrećemo u praksi imaju mnogo manje kapacitete, reda veličine 1 pF-10 mF.

U elektronici postoji potreba za velikim rasponom kapaciteta i drugih radnih svojstava, pa se proizvode tehnološki različite vrste kondezatora, npr. s folijama od različitih polimera, keramički, elektrolitski i slično. Električni kondenzator tvore dva metalna tijela nabijena raznoimenim nabojima istog iznosa. [6]

U ovom diplomskom radu, za izradu tiskane pločice za snimanje zvuka, koristit će se 1 kondenzator od 10 μF po kanalu. Sve ukupno 8 kondenzatora. Također za prototip neće biti potrebe za kondenzator u SMT tehnologiji, ali za tiskanu pločicu koristit će se u SMT tehnologiji.

Sl. 2.7. *Kondenzatori*

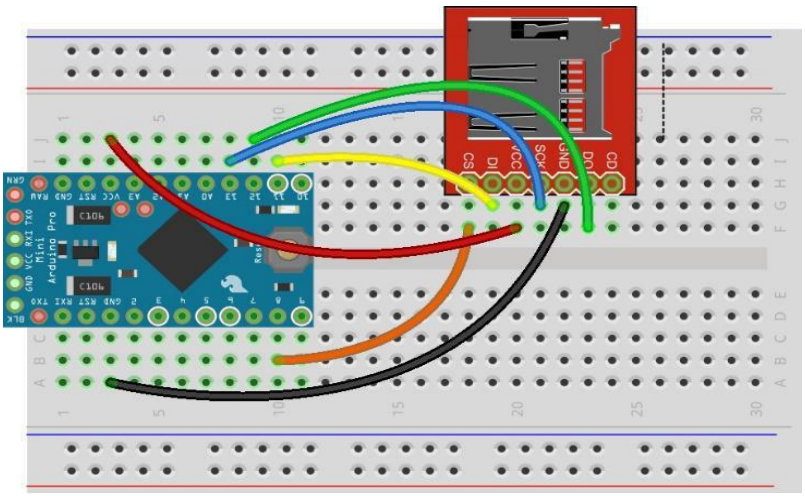
Na slici 2.7. se vide kondenzatori, s tim da su donji prikazani kondenzatori u SMT tehnologiji.

2.6. SD kartica i čitač kartica

Kao što je već rečeno, sav snimljeni zvuk će trebati pohraniti na SD karticu. Razlog pohrane na karticu je taj što je platforma Arduino Nano spojena preko USB konektora s računalom, pa bi direktan prijenos zvuka s tiskane pločice na računalo bio serijski. Da bi se ostvarilo paralelno snimanje zvuka potrebno je zvuk paralelno prikupljati sa svih 8 kanala i spremiti na karticu, te prebaciti na računalo. Upravo zbog svega toga potreba je kartica i čitač kartica kao na slici ispod.

Sl. 2.8. *Arduino čitač SD kartica i SD kartica*

Sa slike se vidi kako izgleda Arduino čitač kartica, a način spajanje je prikazan na slici 2.9.



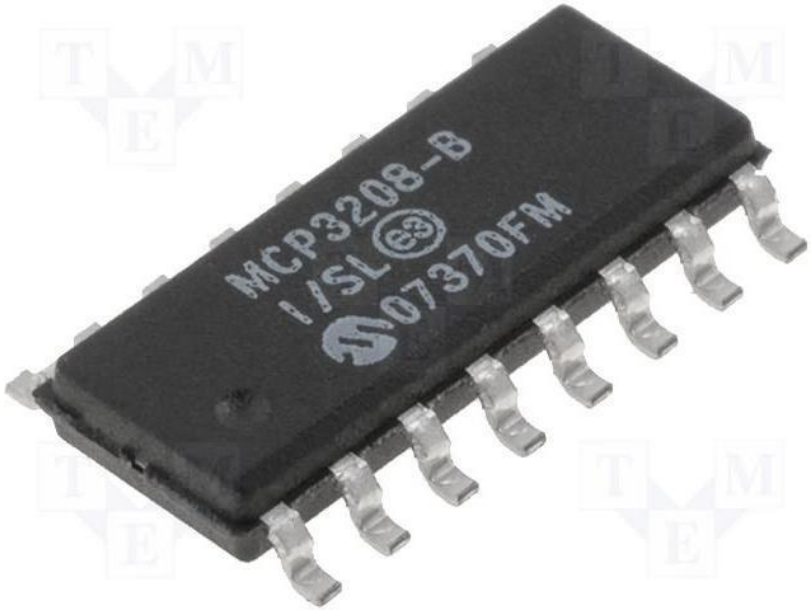
Sl. 2.9. *Način spajanja čitača SD kartica [7]*

2.7. A/D pretvornik s 8-kanala

U elektronici, A/D pretvornik je sustav koji pretvara analogni signal, kao što je zvuk prikupljen od mikrofona, u digitalni signal. A/D pretvornik također može osigurati izolirano mjerenje kao što je elektronički uređaj koji pretvara analogni ulaz ili napon na digitalni broj koji je proporcionalan veličini napona ili struje. Obično je digitalni izlaz zapravo binarni broj koji je proporcionalan ulazu, ali postoje i druge opcije.

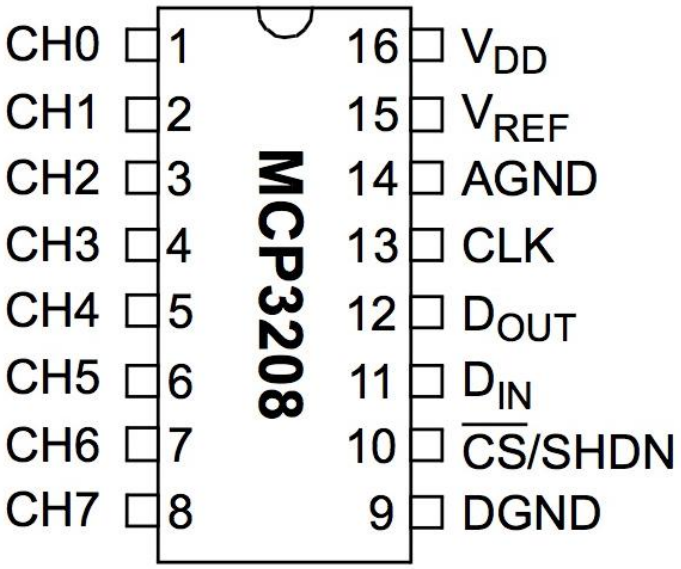
Postoji nekoliko ADC arhitektura. Zbog složenosti i potrebe za precizno prilagođenim komponentama, svi osim najspecifičnijih A/D pretvornika su implementirani kao integrirani krugovi. [8]

U ovom diplomskom je potrebno nakon pojačala i filtera signala, signal dovesti na 8 kanalni A/D pretvornik. Koristit će se 8-kanalni 12-bitni A/D pretvornik MCP3208.



Sl. 2.10. *MPC3208 u SMT tehnologiji*

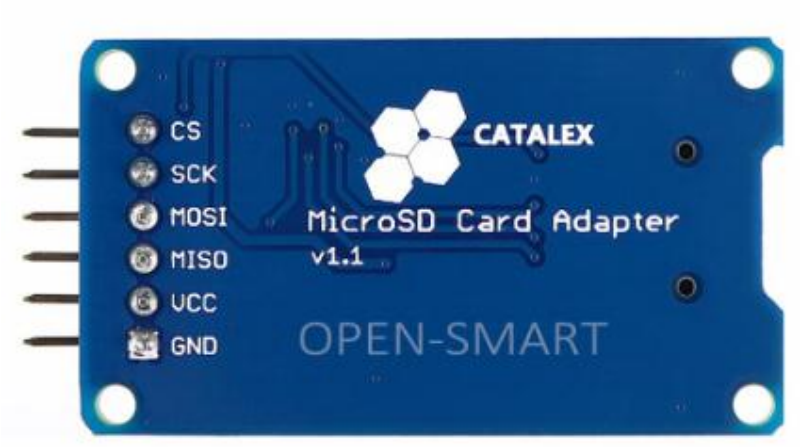
MCP3208 može biti programiran da pruža 4 para pseudo-diferencijalna ulaza ili 8 zasebnih ulaza. Komunikacija s uređajima se postiže korištenjem jednostavnog serijskog sučelja kompatibilnog sa SPI protokolom. Uređaj je sposoban za stopu pretvorbe i do 100 ksps, što znači 100 000 uzoraka po sekundi. Uređaj može raditi na naponu od 2.7 V pa do 5.5 V. Njegov „*low current*“ dizajn mu omogućava rad s tipičnom pripravnošću i aktivnom strujom od samo 500 nA i 230 μA. [9]



Sl. 2.11. *Raspored pinova MCP3208*

Na slikama 2.10. i 2.11. se vidi čip MCP3208 i njegov raspored pinova. Pin broj 9 je uzemljenje, a napajanje je na pinu 16. Kao što se vidi da ima 8 kanala kojima nazivi idu od CH0 do CH7. Na te pinove će se dovoditi signal s izlaza iz operacijskog pojačala.

Vref ili pin 15 se spaja također na napajanje, 5 V. AGND (pin 14) spaja se isto kao i pin 9, na uzemljenje. Pin 10 se spaja direktno na Arduino Nano na pin D9. Pin 13 (CLK) spaja se na čitač SD kartice i to na pin SCK. Čitač SD kartice ima pin *Chip Select* (CS) koji se spaja na Arduino Nano platformu na pin D10. Preostali pinovi na čitaču su VCC, GND, MISO i MOSI. MISO se spaja na Dout (Pin12), a MOSI na Din (Pin11).



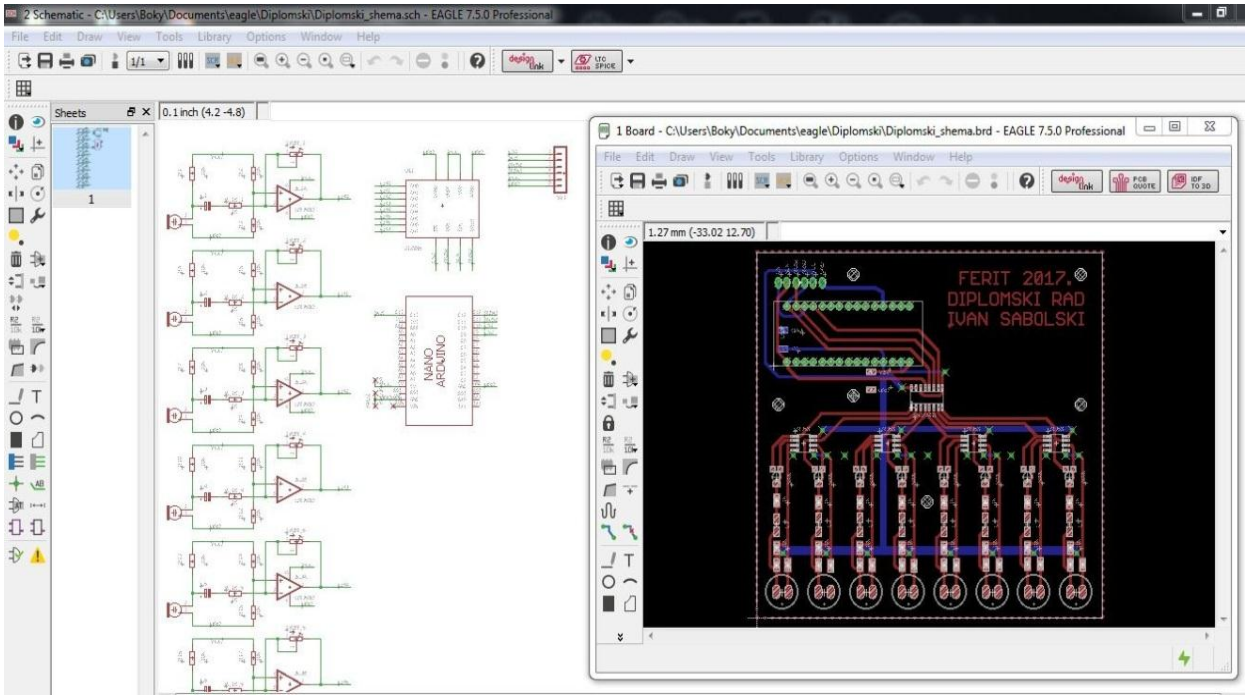
Sl. 2.12. *Čitač SD kartice – pinovi*

2.8. Programski paketi

U ovom radu je bilo potrebno koristiti par programskih paketa. Bio je potreban programski paket Eagle za dizajniranje PCB (engl. *Printed Circuit Board*) pločice. Nadalje se koristio programski paket AutoCAD za dizajniranje 3D modela zaštitnog kućišta za PCB pločicu, koji je kasnije izrađen s 3D printerom. Na posljetku je bilo potrebno napraviti aplikaciju koja će biti u interakciji s gotovom maketom, a to je napravljeno u programskom paketu Microsoft Visual Studio.

2.8.1. Programski paket Eagle

Eagle (engl. *the Easy Applicable Graphical Layout Editor*) je softver za izradu PCB pločica. Program nije kompliciran za korištenje te se sastoji od velikog broja alata. Ima tri modula, uređivač shema, uređivač slojeva i *autorouter* (promjene između automatskog i ručnog usmjeravanja). Također podržava ERC (engl. *Error check*) i moguće je crtati u čak 16 slojeva.



Sl. 2.13. *Eagle programski paket*

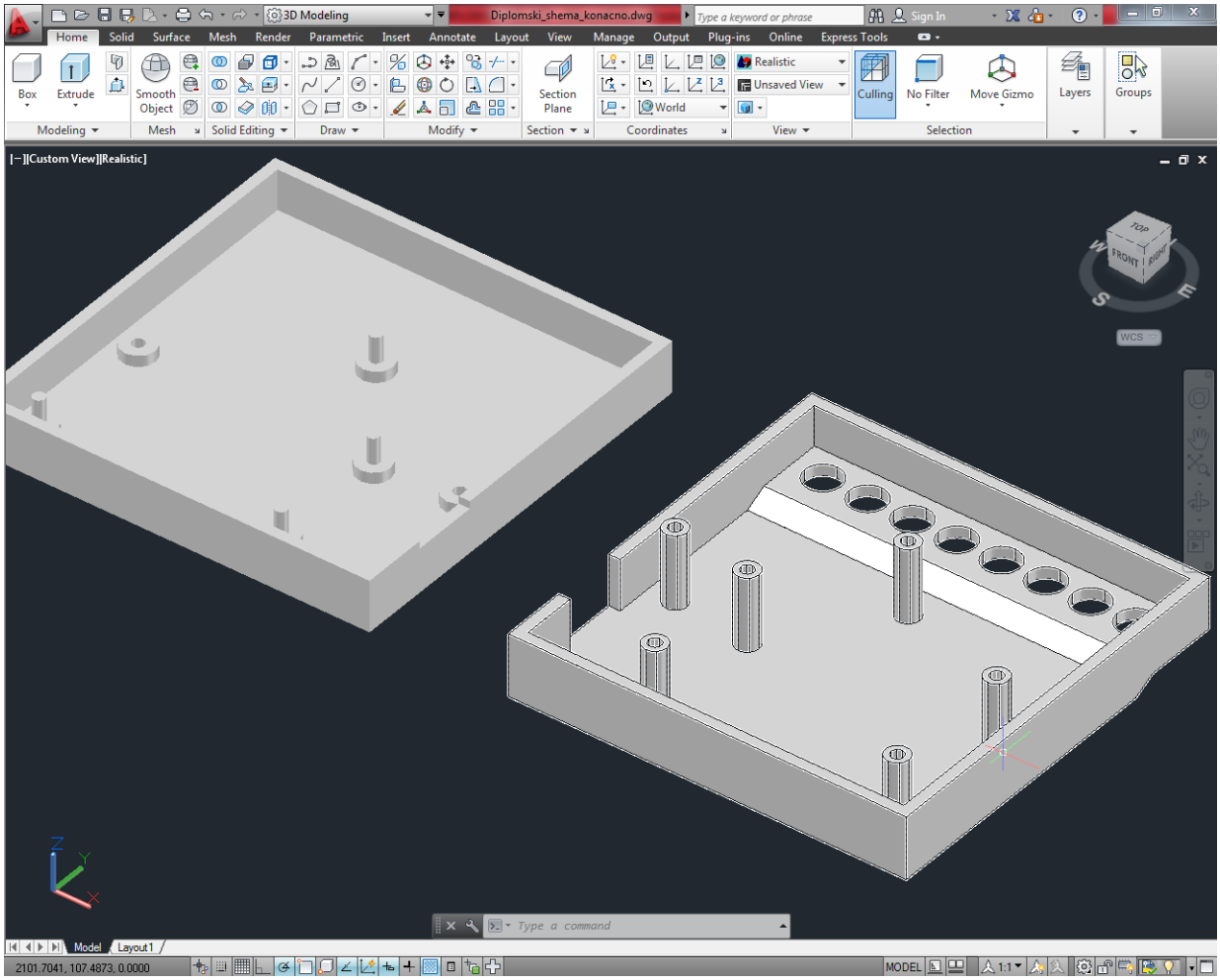
Sa slike 2.13. se vidi kako izgleda programski paket Eagle. Lijevi dio slike je uređivač shema na kojemu se dodaju komponente i crtaju sheme. Desno na slici je otvoren prozor s prikazanom pločicom. U ovom načinu je moguće komponente smjestiti na određeno mjesto na pločicu, u različitim slojevima i na kraju ih spojiti.

2.8.2. Programski paket AutoCad

AutoCad je jedan od najpoznatijih CAD (engl. *Computer Aided Design*) proizvoda za projektiranje pomoću računala kojeg je razvila tvrtka Autodesk. Autodesk nudi više od 75 specijaliziranih softverskih alata i pomagala za različita područja kao što su npr. elektronika, građevina, arhitektura, elektrotehnika i slično.

Osnovni proizvod AutoCAD je sofisticirani projektantski alat široke-univerzalne namjene i on podržava dvodimenzionalno i trodimenzionalno projektiranje. 2D projektiranjem se praktički zamjenjuje klasično projektiranje na papiru, jer je ovdje sve toliko pojednostavljeno. Uz široku paletu alata i veliku preciznost, više nema potrebe crtati na klasičan način na papiru. 3D modeliranje omogućava izradu kompleksnih objekata koji se u modelnom prostoru mogu okretati, zumirati, naginjati, prikazivati u projekcijama, presjecima i pogledima iz svih smjerova, s perspektivnim efektom ili bez njega. Moguće je i proizvoljno osvjetljavati i randerirati model tako da 3D-prikaz imitira fotografiju virtualnog objekta koji postoji samo u memoriji računala.

Za razliku od alternativnih programa za 2D i 3D modeliranje, AutoCad karakterizira sofisticirani sustav mjerila i visoka preciznost koja može ići i ispod milimikrona i automatski kalkuliran sustav dimenzioniranja (kotiranja) izmjera koji zadovoljava i najstrožije tehničke standarde. [10]



Sl. 2.14. *Radni prostor AutoCAD-a*

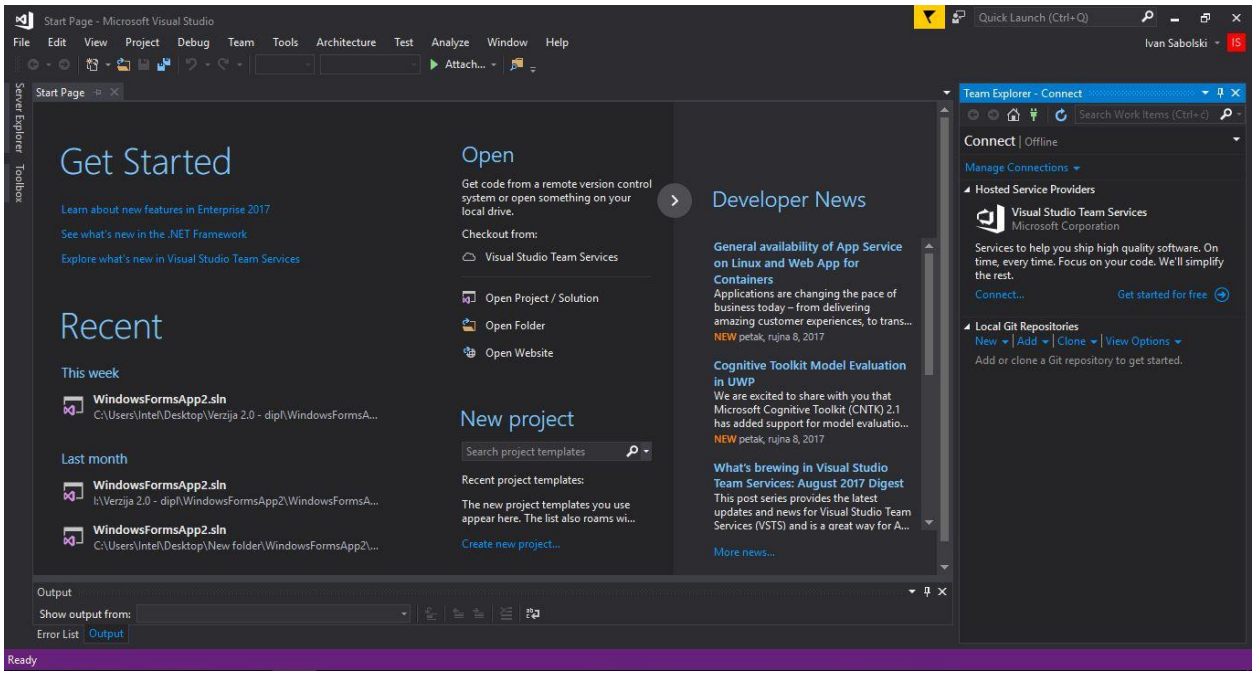
U ovom radu konkretno, AutoCad će se koristiti za dizajniranje zaštitnog kućišta u koje će se smjestiti maketa. Koristit će se prostor za 3D modeliranje s pripadajućim alatima, a radni prostor izgleda kao sa slike 2.14.

2.8.3. Programski paket Microsoft Visual Studio

Microsoft Visual Studio je integrirano razvojno okruženje dizajnirano od strane Microsoft-a. Njegova namjena je razvoj aplikacija, web stranica, računalnih programa za Windows i pružanje usluga. Koristi Microsoft-ove platforme za razvoj poput aplikativnih programskih sučelja (engl. *API*) za *Windows, Windows Forms, Windows Presentation Foudation, Windows Store* i *Microsoft Silerlight*. Može proizvesti nativni kod i upravljani kod.

Visual Studio sadrži urednik izvornog koda koji podržava *IntelliSense* i refaktoriranje koda. *IntelliSense* je komponenta koja predlaže ostatak koda. *Debugger* koji je integriran u Visual Studio radi na nivou izvornog i strojnog koda. Program sadrži razne alate poput dizajnera oblika koji se koristi za pravljenje aplikacija s grafičkim korisničkim sučeljem, dizajnera klasa, dizajnera shema baza podataka i web dizajnera. Također prihvaća proširenja za poboljšavanje funkcionalnosti na skoro svakom nivou dodavajući podršku sustava za upravljanje izvornim kodom i dodavajući nove alate poput urednika teksta i vizualnih dizajnera za jezik specifičnih domena ili za neke druge dijelove procesa razvoja programa.

Visual Studio podržava različite programske jezike i isto tako dozvoljava *debugger-u* i uredniku koda da podržavaju skoro svaki programski jezik, ali pod uvjetom da za taj jezik postoji servis. Ugrađeni jezici su VB.NET (preko Visual Basic .NET), C++/CLI (preko Visual C++), C++, C, C# (preko Visual C#), i F# (počevši od programa Visual Studio 2010). Podrška za programske jezike poput Ruby-ja, M, Python kao i ostalih je dostupan, ali instalacijom jezičnih servisa. Također podržava JavaScript, CSS, XML/XSLT, HTML/XHTML. [11]



Sl. 2.15. *Microsoft Visual Studio 2017 – početna stranica*

U ovom radu, za izradu aplikacije, bit će korišten programski jezik C#. C# je objektno orijentirani programski jezik, što znači da svi elementi unutar njega predstavljaju objekt. Nadalje objekt predstavlja strukturu koja sadrži podatkovne elemente, kao i metode i njihove međusobne interakcije. To je jezik visokih performansi za .NET platformu kojeg je razvila tvrtka Microsoft s ciljem da bude siguran, jednostavan, moderan i jezik opće namjene, a nastao je na temelju objektno orijentiranih programskih jezika Visual Basic i C++.

Semantika i sintaksa su preuzeti iz Java programskog jezika, a grafički objekti se temelje na Basic-u. Ima brojne mogućnosti vezane za klase, metode i njegovo definiranje, kao i korištenje enkapsulacije, polimorfizma i nasljeđivanja. Podržava XML stil unutar dokumenata, svojstva, sučelja, događaje te rad s pokazivačima. Također podržava prikupljanje smeća (engl. *garbage collection*) i time omogućava oslobođenje memorije od objekata koji se više ne koriste, a to je korisno jer programer ne mora voditi računa o tome.

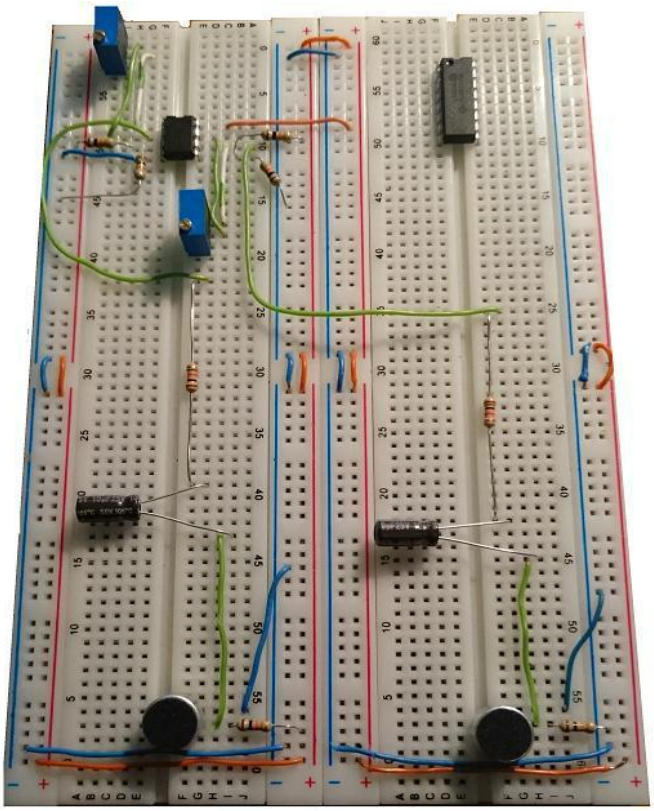
Slično kao i u C i C++, osnovni tipovi podataka su *char, int, float, long, short, double, decima, string*. Postoje i izvedeni tipovi, kao što su *uint, ulong i ushort*.

3. REALIZACIJA MAKETE

Realizacija makete počinje izradom prototipa na razvojnoj pločici. Nakon toga potrebno je testirati prototip i prijeći na izradu dizajna. Dizajn će biti rađen u programskom paketu Eagle. Nakon što pločica bude gotova slijedi postupak lemljena. Kada je maketa gotova opet je potrebno vršiti testiranja te ako je sve u redu prijeći na izradu aplikacije.

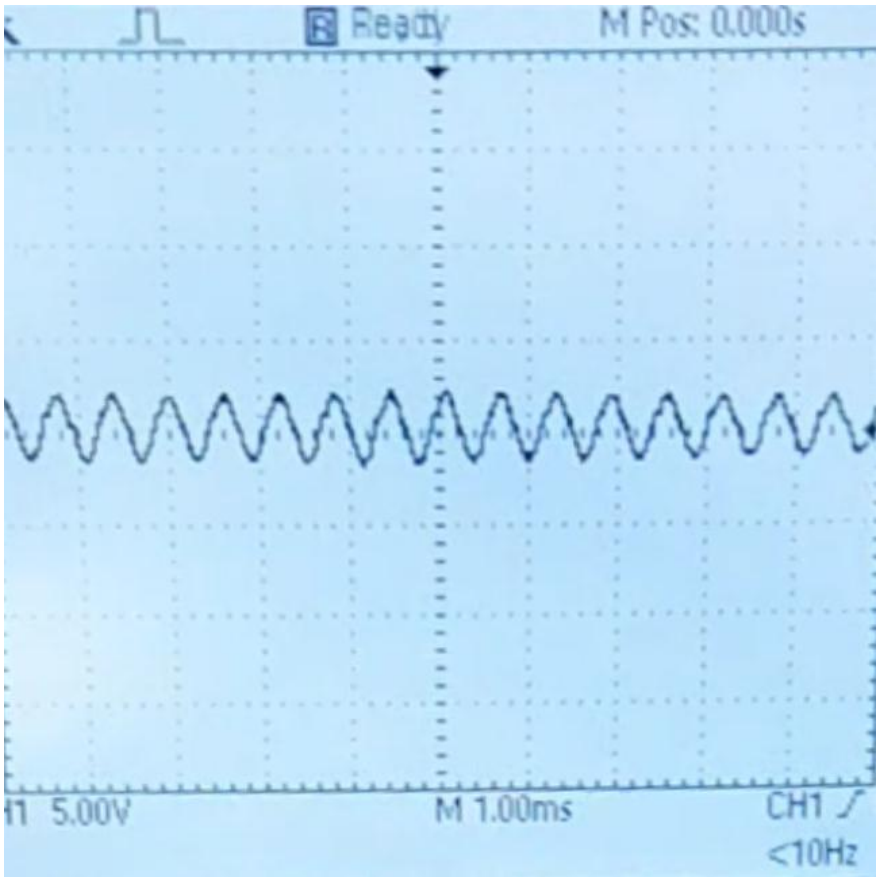
3.1. Prototip

Izrada prototipa počinje izradom 2 kanala za snimanje zvuka na razvojnoj pločici. Potrebni dijelovi za jedan kanal su 3 otpornika od 10 KΩ, jedan od 1 KΩ, promjenjivi otpornik od 200-470 kΩ, zatim je potreban jedan kondenzator od 10 µF, electret mikrofon i operacijsko pojačalo. Nakon što se sve spoji prema shemi na slici 2.1., izlaz iz operacijskog pojačala treba ići na A/D pretvornik s 8 kanala, točnije na pin CH0 pošto je to prvi kanal. Nakon toga se spaja drugi kanal i sve do kraja. Prototip je rađen s jednim kanalom, pa tek nakon što je testiran dodan je još jedan kanal. Nakon toga je vršeno testiranje i ta 2 kanala. Na slici 3.1. se može vidjeti razvojna pločica na kojoj su spojena 2 kanala.



Sl. 3.1. *Prototip s 2 spojena kanala*

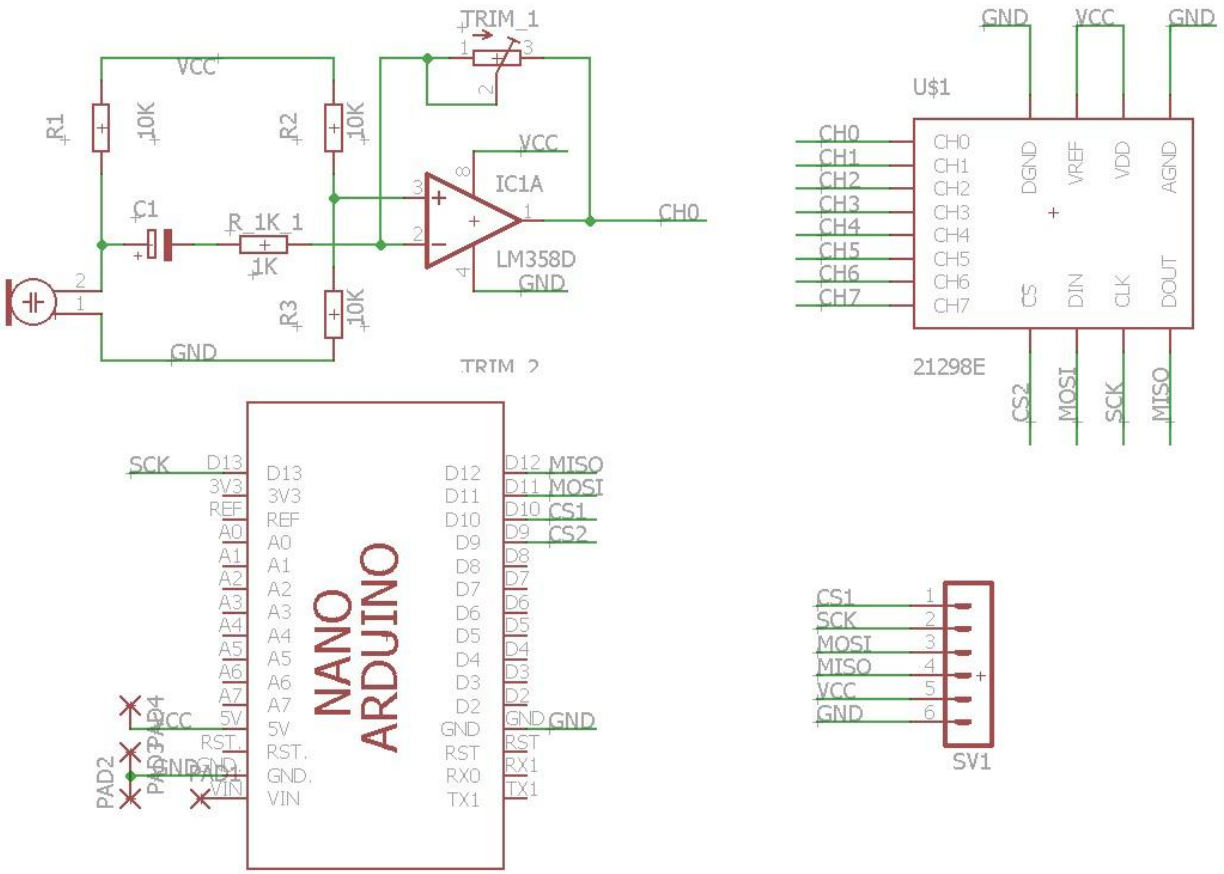
Testiranje je vršeno na osciloskopu, tako da se sonda spaja na izlazne pinove od A/D pretvornika. Na osciloskopu se moglo vidjeti kako zvuk dolazi od mikrofona, tako da se iza mikrofona stavi sonda i prati odziv. Taj odziv je puno slabiji kada se prati prije operacijskog pojačala. Ako sondu stavimo iza njega, vidi se kako je pojačan. Na slici 3.2. je prikazan zvuk koji je proizveden konstantim pištanjem.



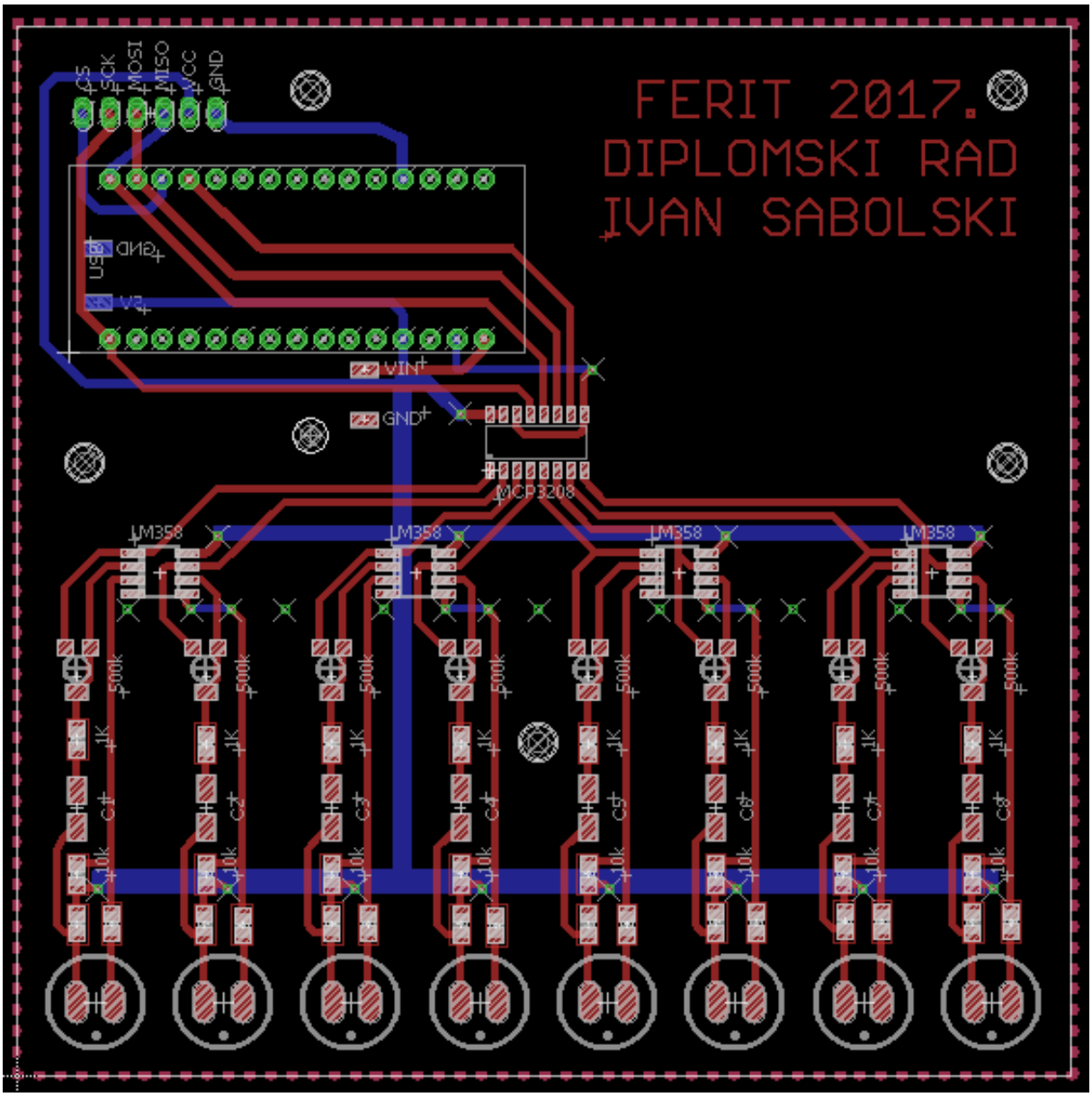
Sl. 3.2. *Snimljeni zvuk prikazan na osciloskopu*

3.2. Dizajn u Eagle-u

U programskom paketu Eagle dizajnirana je pločica. Na slici 3.3. se vidi shema, ali prikazana je shema spajanja jednog kanala. Takvih naravno ima 8. Vidi se shema spajanja Arduino platforme, A/D pretvornika i čitača SD kartice.



Sl. 3.3. *Shema u Eagle programskom paketu*

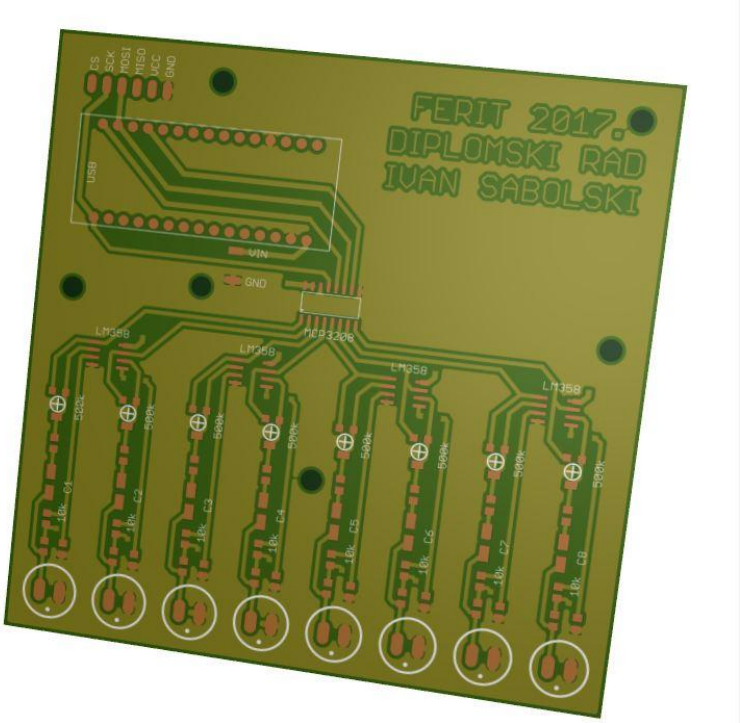


Sl. 3.4. *Dizajn pločice*

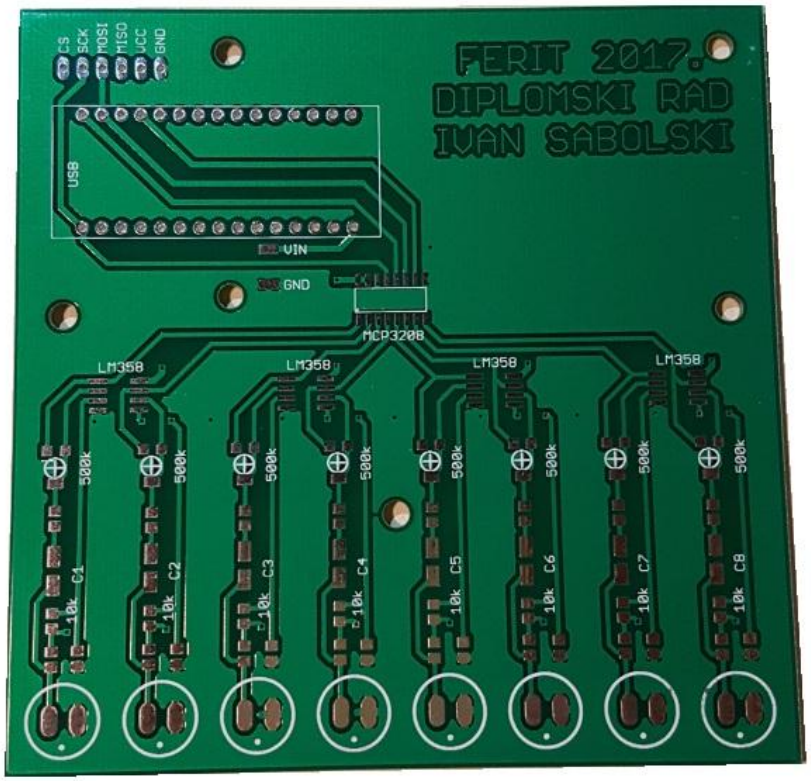
Na slici 3.4. vidi se gotovi dizajn pločice. Pri dizajniranju trebalo je paziti da mikrofoni budu na jednakoj udaljenosti jedan od drugoga, kako bi rezultati bili što precizniji pri snimanju. Isto tako trebalo je paziti da ostale komponente ne budu pre blizu jedna drugoj.

Radi SPI komunikacije, Arduino Nano je postavljen što je moguće bliže A/D pretvorniku i modulu za SD karticu. Pločica je dizajnirana u 2 sloja. Gornji sloj je routan crvene boje, a donji plave. Na pločici su predviđene rupe za učvrstiti pločicu za zaštitno kućište ili radnu površinu.

Nakon završenog dizajna, pločica je poslana na izradu. Moguće je unaprijed dobiti predodžbu kako će ta pločica izgledati. To se može vidjeti na slici 3.5., dok se na slici 3.6. može vidjeti kako stvarno izgleda pločica koja je spremna za daljnje lemljenje komponenata.



Sl. 3.5. *Predođba pločice prije printanja*



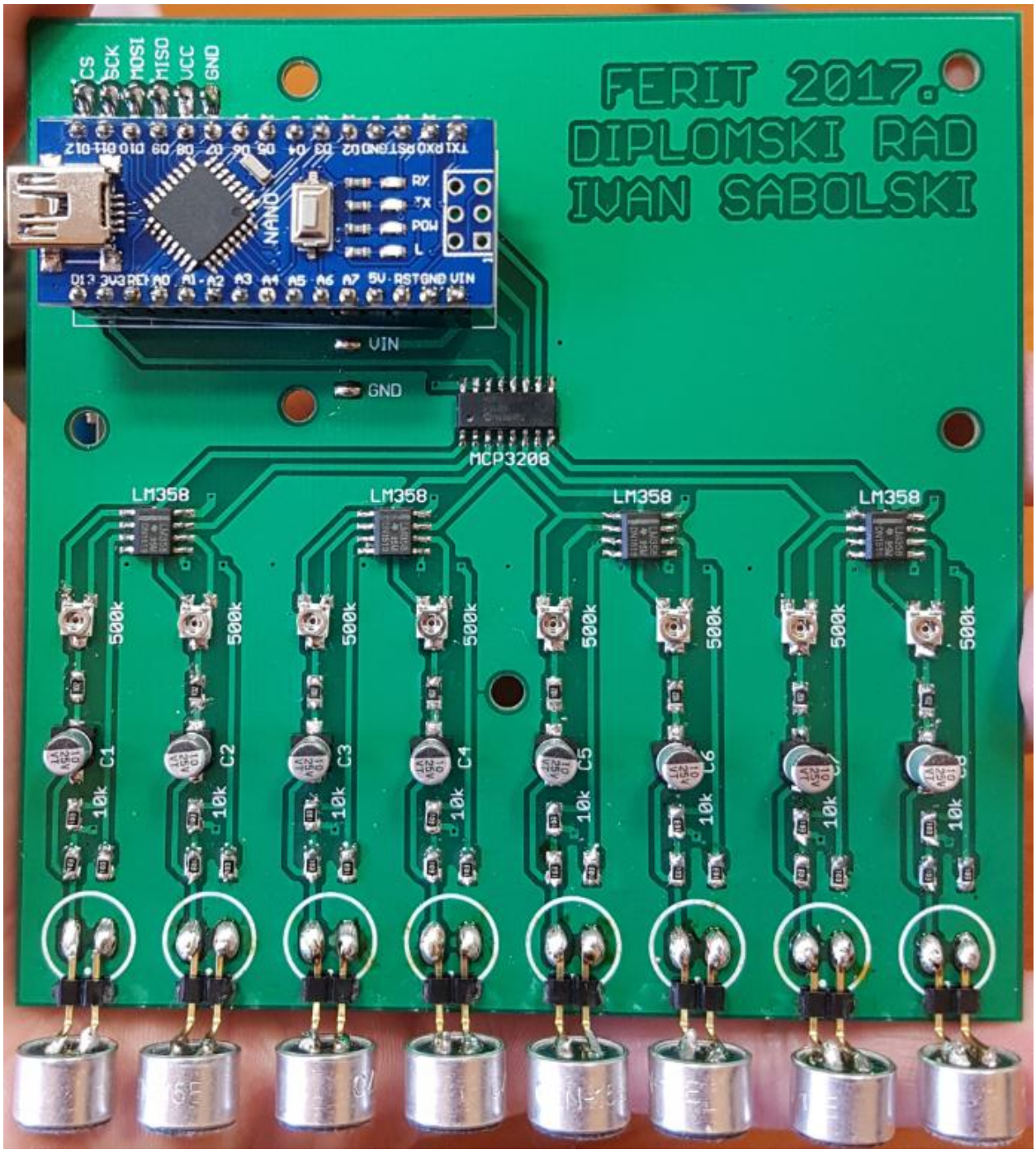
Sl. 3.6. *Pločica spremna za lemljenje komponenata*

3.3. Postupak lemljenja SMT komponenti na pločicu

Nakon što je izrađena pločica bilo je potrebno zalemiti sve komponente na nju. Budući da je SMT tehnologija jako sitna, korištena je kamera radi lakšeg lemljenja. (Sl. 3.7.). Također za precizno lemljenje je zaslužna i prijenosna digitalna lemilica, koja dozvoljava precizno podešavanje i održavanje temperature lemljenja.

Sl. 3.7. *Postupak lemljenja*

Na mjesto gdje dolazi Arduino Nano platforma zalemljen je *socket* radi mogućnosti skidanja same platforme. Gotova pločica spremna za testiranje prikazana je slikom 3.8.

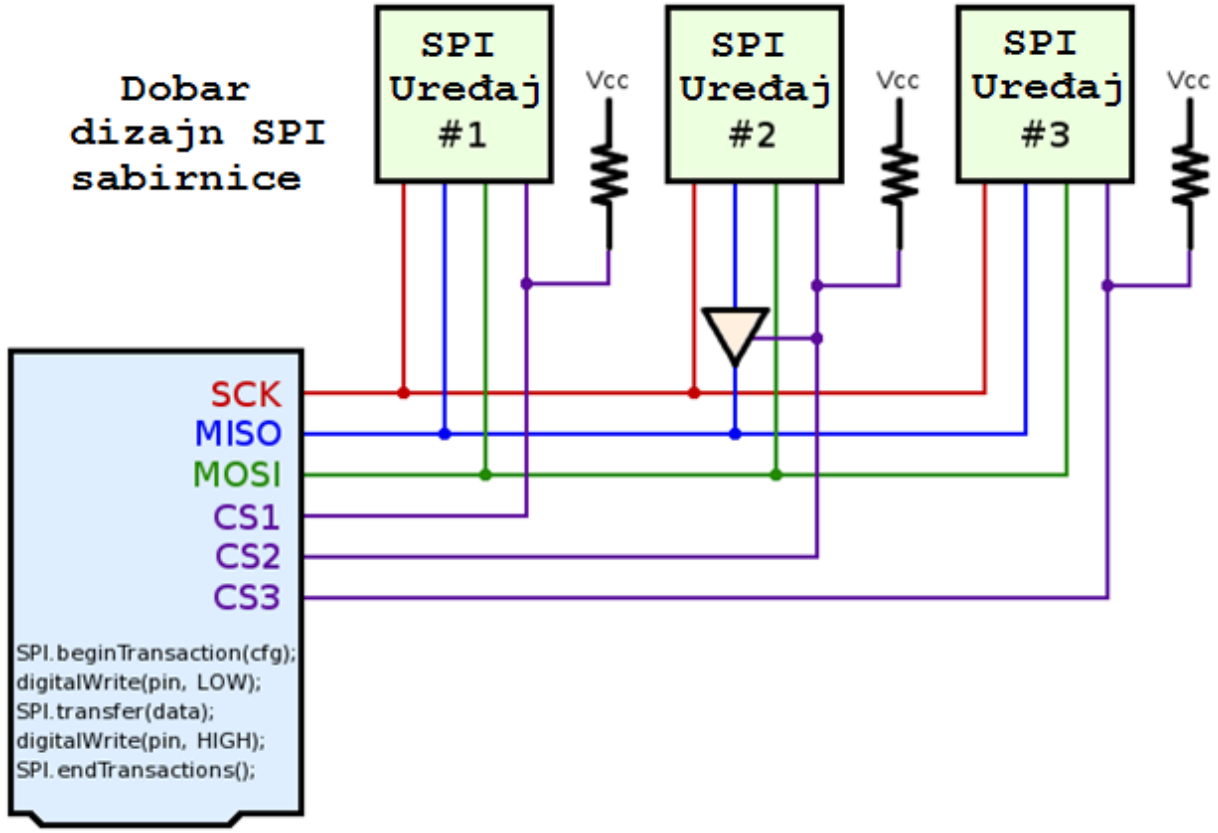


Sl. 3.8. *Gotova pločica spremna za testiranje*

3.4. Testiranje makete

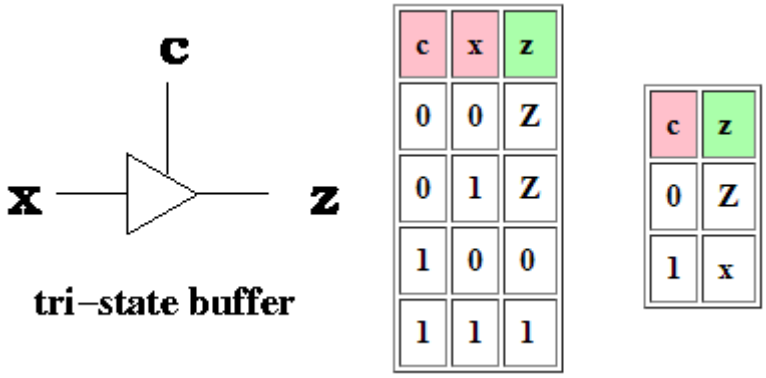
Testiranje je započeto ispitivanjem funkcionalnosti čitača SD kartice i same kartice. Naime pomoću jednostavnog koda za ispitivanje čitanja i pisanja je utvrđeno da čitač i kartica rade, Arduino šalje podatke i sprema na karticu. Naravno kartica je uklonjena iz čitača i spojena na drugo računalo radi provjere.

Sljedeći korak bio je provjeriti dali sklop prikuplja podatke (zvuk) te ako prikuplja, bilo je potrebno spremiti podatke na karticu. Prije nego je čitač SD kartice zalemljen na pločicu bilo je testirano prikupljanje podataka i sve je radilo kako i treba. Naime sada kada je čitač kartica zalemljen, uz puno pokušaja i korigiranje koda, sklop nije prikupljao podatke kao što je to činio prije. Radi se o tome da Arduino putem SPI (engl. *Serial Peripheral Interface*) sučelja komunicira i s čitačem kartica i s AD pretvornikom. Problem je što su na sabirnicu priključeni i AD pretvornik i čitač kartica, te zbog toga Arduino ne zna kako upravljati s onime što dolazi na sabirnicu. Da bi se riješio taj problem probalo se dodati pullup otpornike na sve CS (engl. *Chip Select*) signale kao na slici 3.9.



Sl. 3.9. *Poboljšavanje SPI dizajna*

Ni nakon što su dodani *pullup* otpornici prikupljanje podataka nije radilo. U daljnjem istraživanju pronađeno je rješenje u dodavanju takozvanog „*tri-state buffer-a*“. *Tri-state buffer* je nešto poput ventila. On je dobar za upravljanje podacima koji dolaze na sabirnicu. Postoje 2 tipa *buffera*, aktivan na '1' i aktivan na '0'. Za primjer neka se uzme *tri-state buffer* koji je aktivan na '1', prikazan je na slici 3.10.

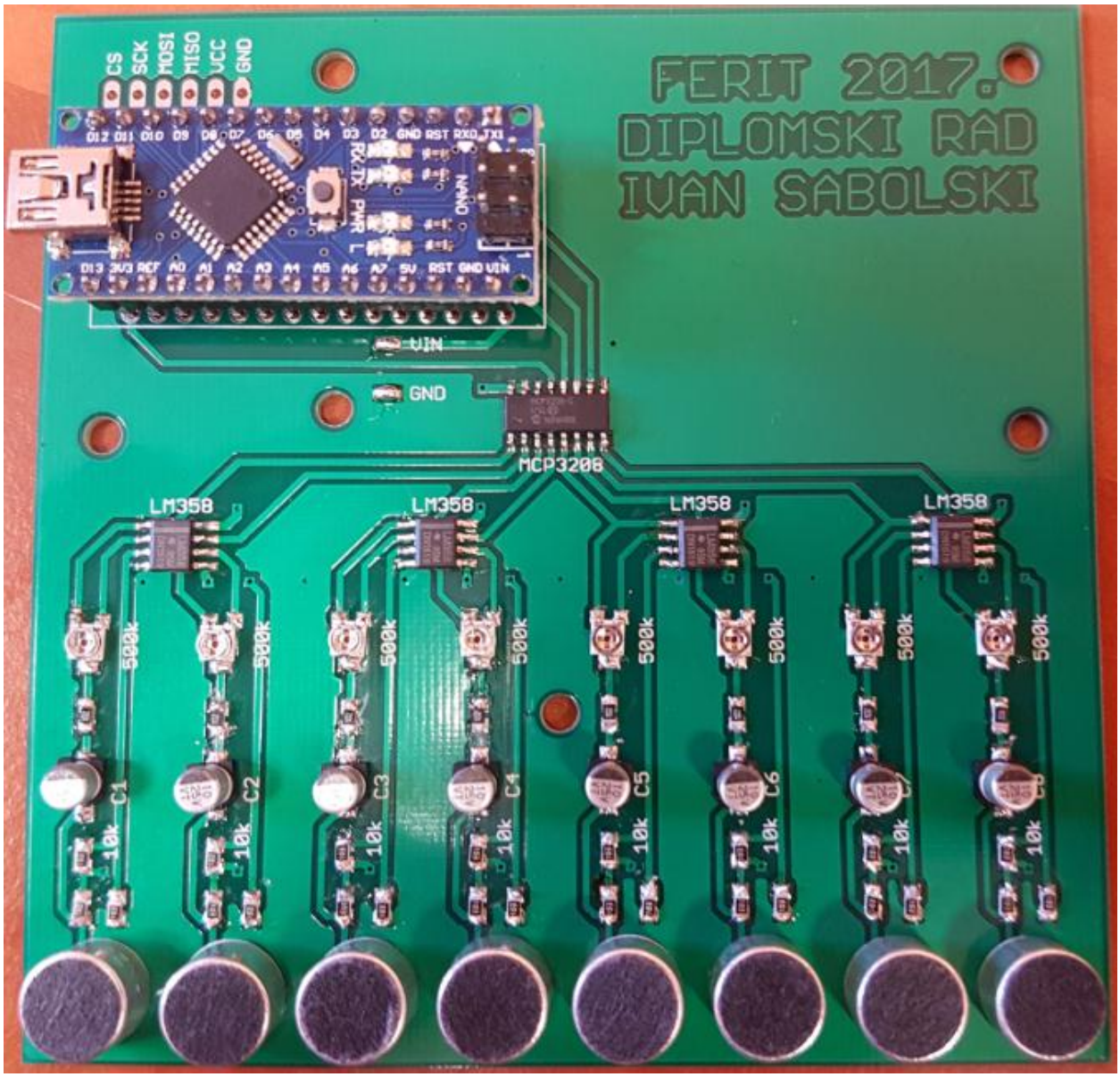


Sl.3.10. *Tri-state buffer s pripadajućim tablicama istinitosti*

Tri-state buffer ima 2 ulaza, kontrolni ulaz c (engl. *control input*) i podatkovni ulaz x. I ima 1 izlaz z. Kada je kontrolni ulaz aktivan, tada je izlaz jednak podatkovnom ulazu x. Kontrolni ulaz se ponaša kao ventil. Kada je kontrolni ulaz neaktivan onda je izlaz jednak z. Nije bitno dali je x jednak nuli ili jedinici, struja ne protječe kroz njega i izlaz je jednak z. Vrijednost podatka ne prolazi kroz tzv.ventil. Sve ovo se vidi i u tablici istinitosti na slici 3.10.

Iz svega ovoga se vidi kako je *tri-state buffer* dobar za kontrolu onoga što dolazi na sabirnicu i samim time bi to riješilo problem sa SPI komunikacijom. Ali budući da je printana pločica izrađena i sve je već zalemljeno odlučeno je snimati podatke u stvarnom vremenu, tj. serijski.

S pločice je odlemljen čitač kartice pa pločica sada izgleda prema slici 3.11.

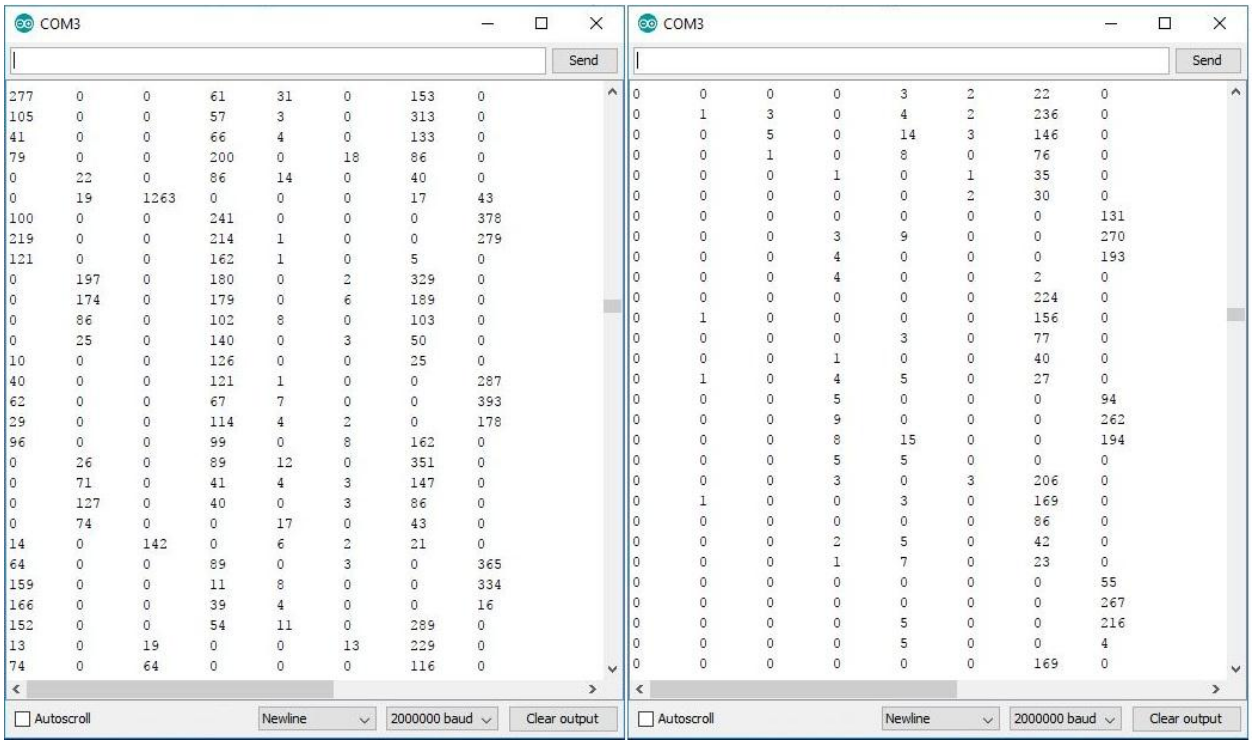


Sl. 3.11. *Konačan izgled makete*

Nakon što je čitač SD kartice odlemljen, sada sklop opet radi i prikuplja podatke. Odlučeno je napraviti aplikaciju koja će se putem serijskog porta spajati s pločicom. Aplikacija sadrži parametre o snimanju, kao što su odgoda prije snimanja i duljina snimanja. Njena svrha je dati jednostavniji pregled i korištenje krajnjem korisniku i naravno služi za prikupljanje rezultata koje Arduino šalje. Te rezultate aplikacija sprema u datoteku i dodjeljuje joj ime trenutnog datuma i vremena.

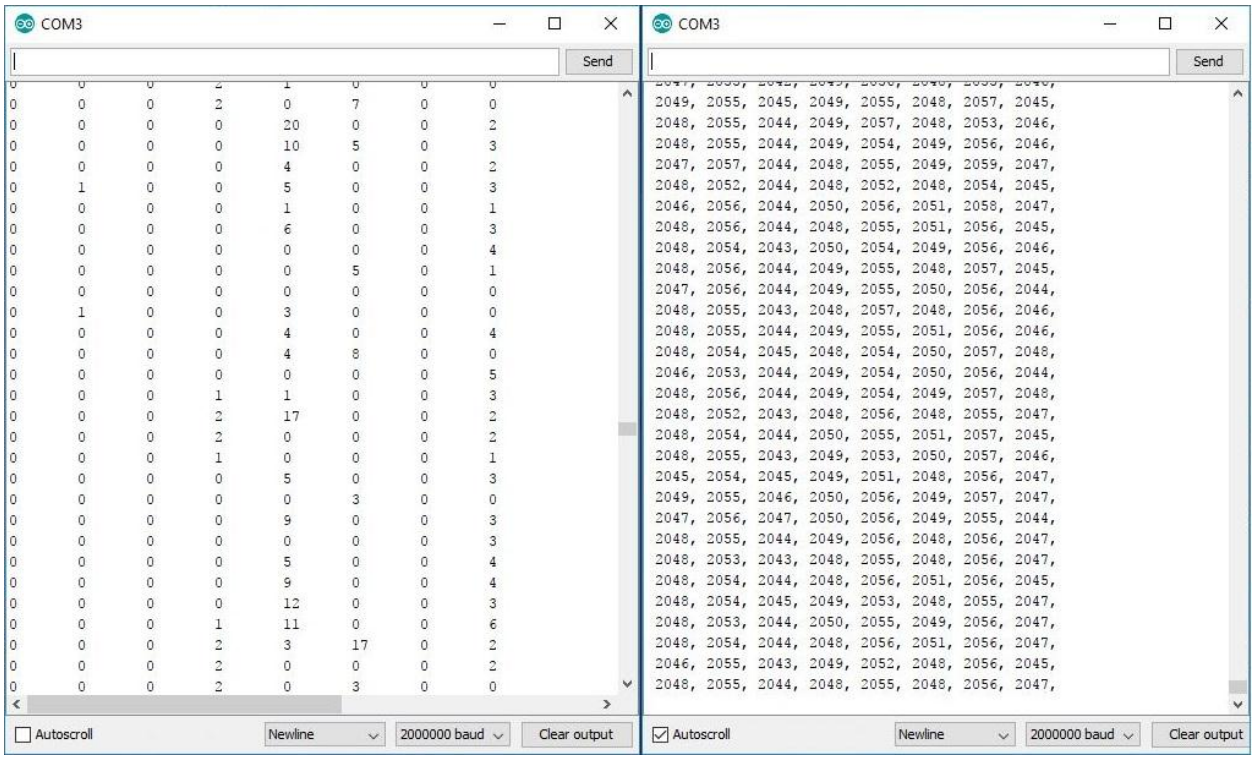
Bilo je zamišljeno da sklop ne prikuplja podatke, tj. ne snima, sve dok aplikacija ne pošalje svoje parametre Arduino kontroleru. To je bilo izvedeno s *timer-ima,* ali je sklop presporo radio. Pa je odlučeno da sklop vrši snimanje cijelo vrijeme, što znači od trenutka kada mu je dovedeno napajanje, sklop snima. Aplikacija će u tom slučaju prikupiti podatke u stvarnom vremenu i to od trenutka pritiska start gumba, pa sve dok ne istekne vrijeme snimanja koje je postavljeno (ili pritiska stop gumba). Nakon prestanka prikupljanja podataka, sklop će i dalje raditi u pozadini.

Sljedeće što je bilo potrebno napraviti je pokrenuti sklop i promatrati kanale na serijskom monitoru da bi se odradilo baždarenje kanala. Zbog promjenjivog otpornika svaki kanal prikuplja podatke s drugačijim intenzitetom. To je bilo potrebno uskladiti.



Sl. 3.12. *Baždarenje kanala u tišini*

Na lijevoj strani slike 3.12. se vidi kako je to izgledalo prije početka baždarenja. Naime neki kanali prikupljaju i najmanji zvuk, dok ostali nisu toliko „osjetljivi“, pa je potrebno postići da u tišini budu prikazane sve 0. Već na desnoj strani se vidi kako su prvih 6 kanala podešeni. Preostala su još zadnja 2 kanala za podesiti.



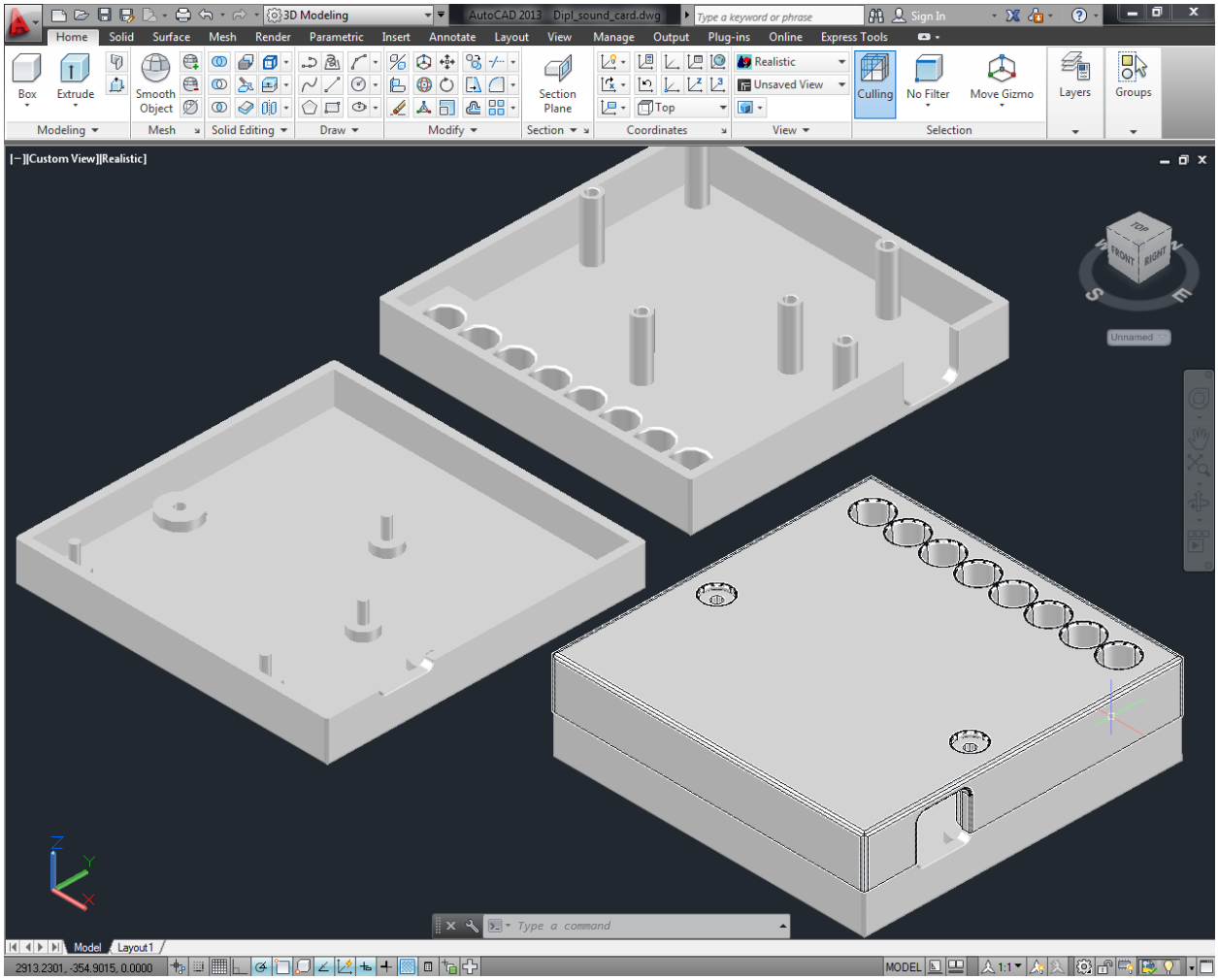
Sl. 3.13. *Podešeni kanali i dodan offset*

Lijeva strana slike 3.13. prikazuje podešene sve kanale. Ali budući da je zvuk zapravo val, on ima svoju valnu duljinu tj. brijeg i dol. U tom slučaju bi vrijednosti išle u minus ili bi se prikazivale kao duplo veće da se izbjegne minus. Zato je najbolje postaviti *offset*, pa tada 0 postaje 2048. Taj problem je riješen dodavanjem drugog *library-ja* za A/D pretvornik. Vrijednost 2048 je polovina od gornje granice 12 bitnog A/D pretvornika. Sada prilikom snimanja vrijednosti osciliraju oko 2048, tj. rastu prema 4096 ili padaju prema 0. Prema desnoj strani slike 3.13. vidi se podešenje kanala u tišini.

3.5. Izrada zaštitnog kućišta za maketu

Prvotna ideja bila je vrućim zrakom nalemit mikrofone bez nožica za pločicu. Time bi dobili točan razmak između pojedinog kanala i uz to bi ti mikrofoni bili dosta čvrsti. Zbog teškoće izvedbe samog lemljenja ipak su korišteni mikrofoni s nožicama. Samim time su podignuti u zrak i dosta su krhki. Ako bi se slučajno zapelo za njih, mogli bi se odlemiti ili izazvati kratki spoj. To je u konačnici otežavalo mogućnost prijenosa makete te je iz tog razloga bilo potrebno napraviti zaštitno kućište.

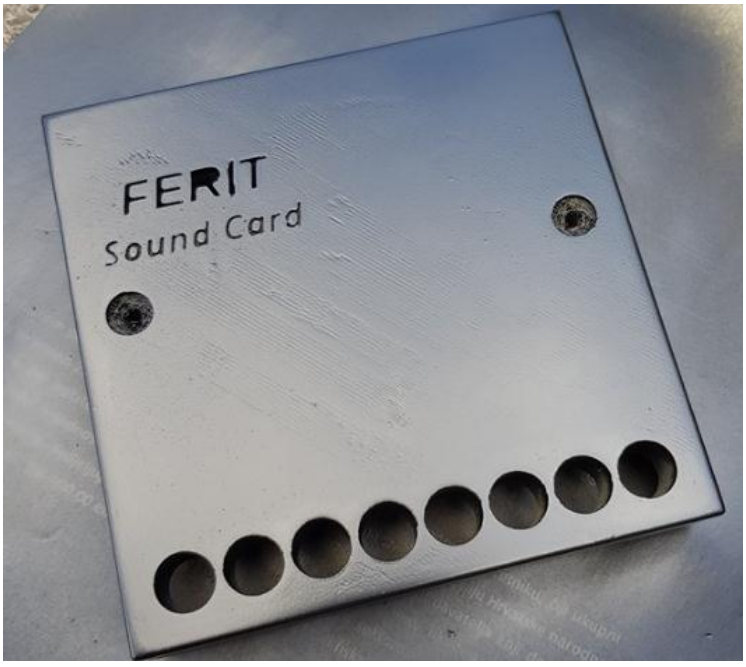
Kućište je dizajnirano u programskom paketu AutoCad kao što je već spomenuto. Bilo je potrebno uzeti točne mjere iz programskog paketa Eagle i ucrtati ih u AutoCAD. Na osnovu tih mjerenja stvoren je 3D model kao na slici 3.14.



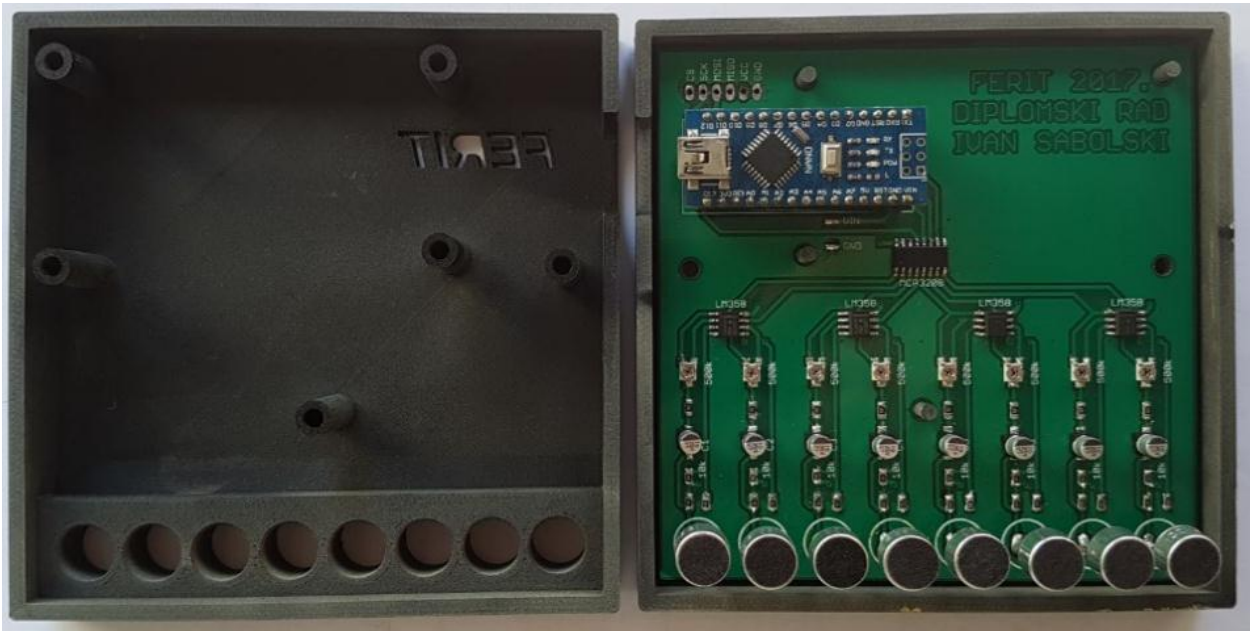
Sl. 3.14. *3D model kućišta*

Nakon printanja prvog kućišta i uz par prepravaka izrađen je konačan 3D model zaštitnog kućišta za printanu pločicu i prikazan je na slici 3.15.

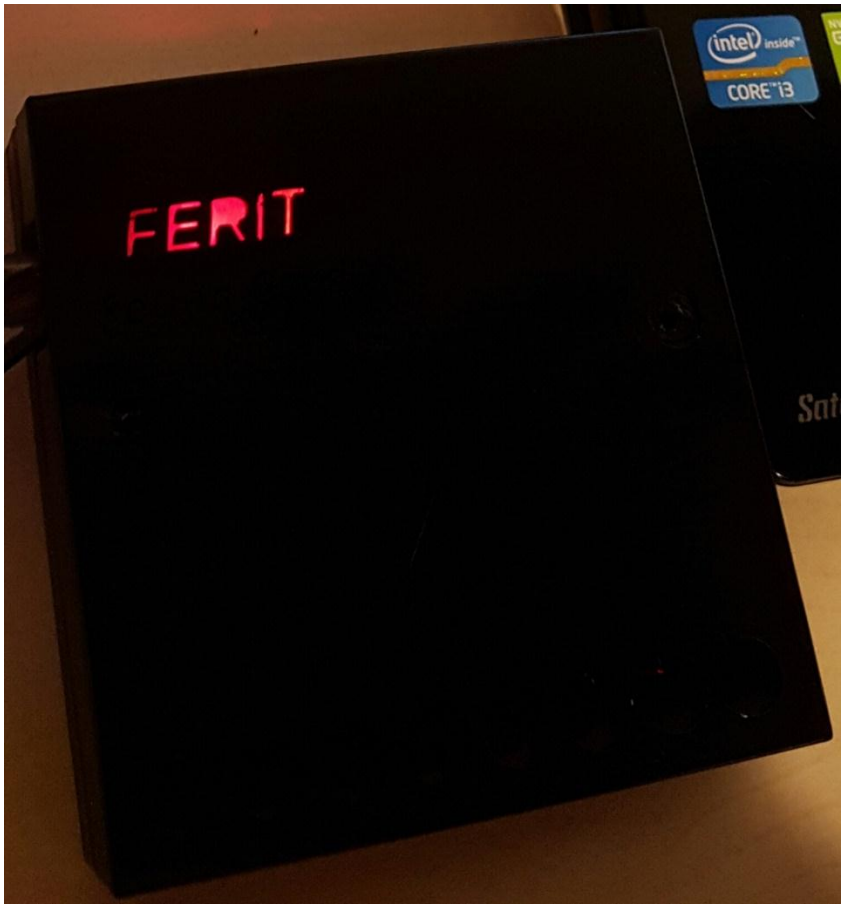
Sl. 3.15. *Gotov model zaštitnog kućišta*



Sl. 3.16. *Zaštitno kućište za maketu*



Sl. 3.17. *Maketa u kućištu*

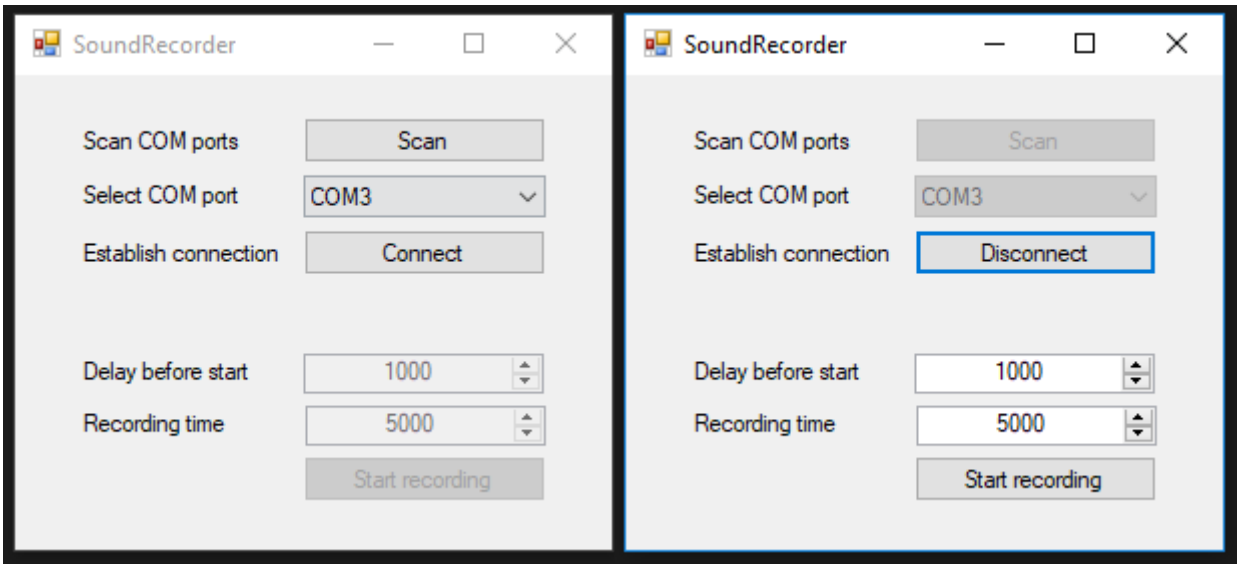


Sl. 3.18. *Gotova maketa u radu*

3.6. Izrada aplikacije za spajanje na maketu

Za spajanje na maketu bilo je potrebno izraditi C# aplikaciju koja bi omogućavala pretraživanje dostupnih COM portova, odabir željenog COM porta i ostvarivanje komunikacije. Nakon što je to ostvareno, dodani su još parametri za duljinu snimanja i odgodu snimanja. Odgoda snimanja je potrebna da se ne bi čuo klik miša koji pokreće snimanje, tj.da u tom slučaju ne bude šumova. Obično je odgoda postavljena na 1 sekundu, a moguće je staviti od 0-3 sekunde. Snimanje traje od 1-10 sekundi.

Nakon što se pritisne gumb za pokretanje snimanja, te nakon što istekne vrijeme odgode, maketa počinje snimati zvuk. U pozadini maketa prikuplja podatke i šalje ih aplikaciji koja ih upisuje u datoteku. Po završetku snimanja, aplikacija sprema datoteku i dodjeljuje joj datum i vrijeme u sam naziv datoteke.

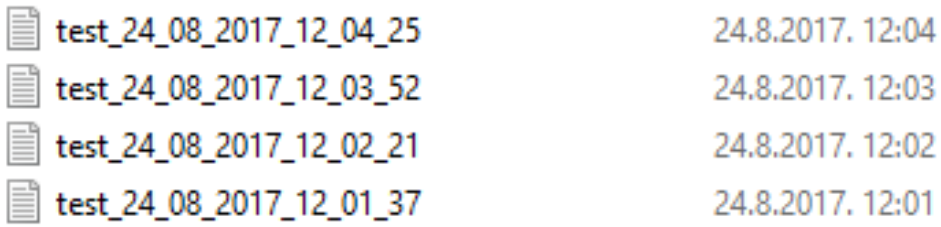


Sl. 3.19. *Dizajn aplikacije*

Na slici 3.19. se vidi dizajn aplikacije. Lijeva slika prikazuje kako izgleda aplikacija tek kad se pokrene, potrebno je pretražiti dostupne portove, odabrati port i spojiti se na njega. Na slici je odabran COM port 3. Nakon toga je moguće odabirati vrijeme odgode i vrijeme snimanja. Zatim je potrebno pritisnuti „*start recording*“.

4. REZULTATI

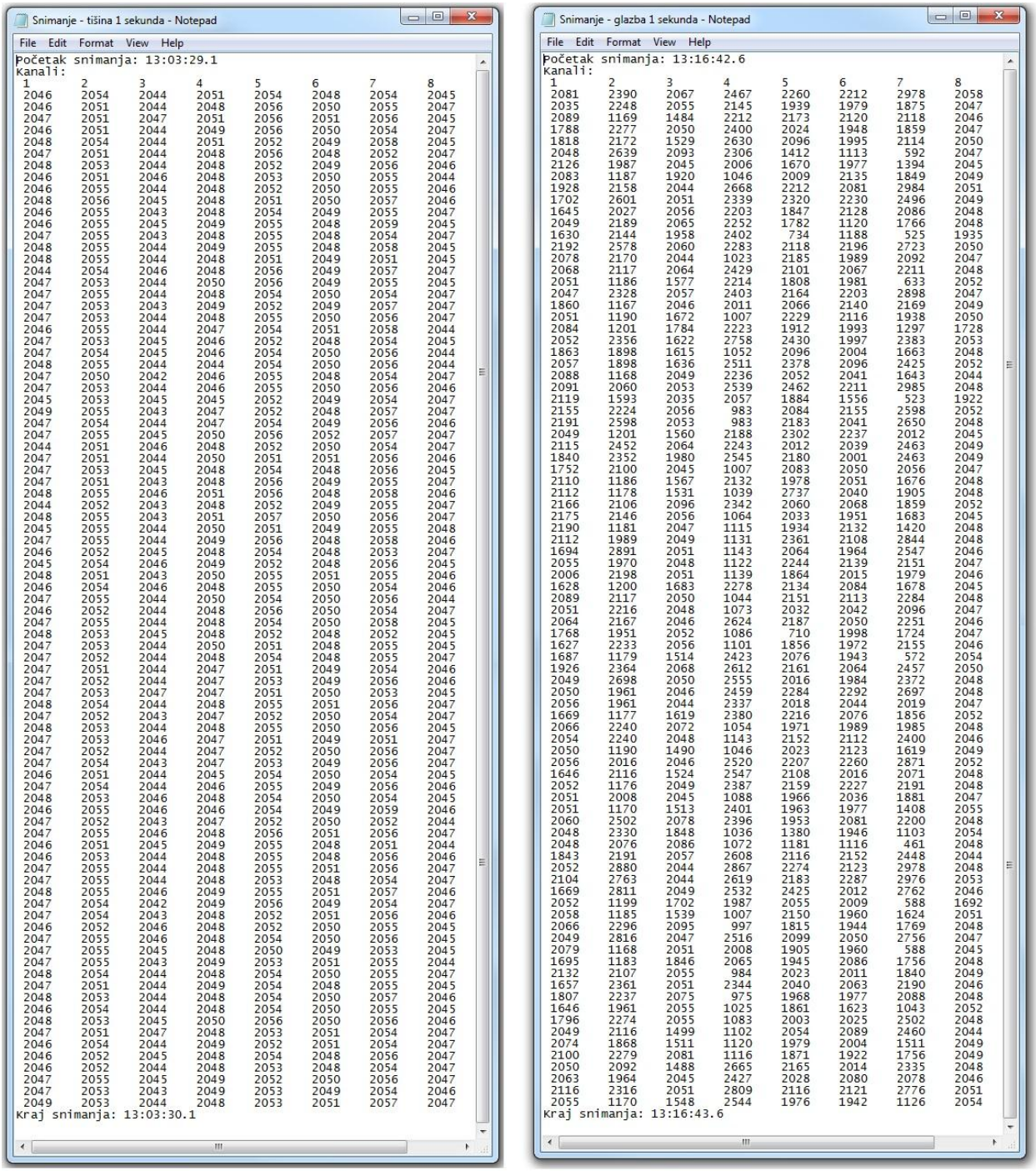
Rezultate dobivene od makete, aplikacija sprema u datoteku. Datoteka se nalazi pod imenom trenutnog datuma i vremena. Na slici 4.1. je prikazano kako to izgleda.



Sl. 4.1. *Spremljeni rezultati*

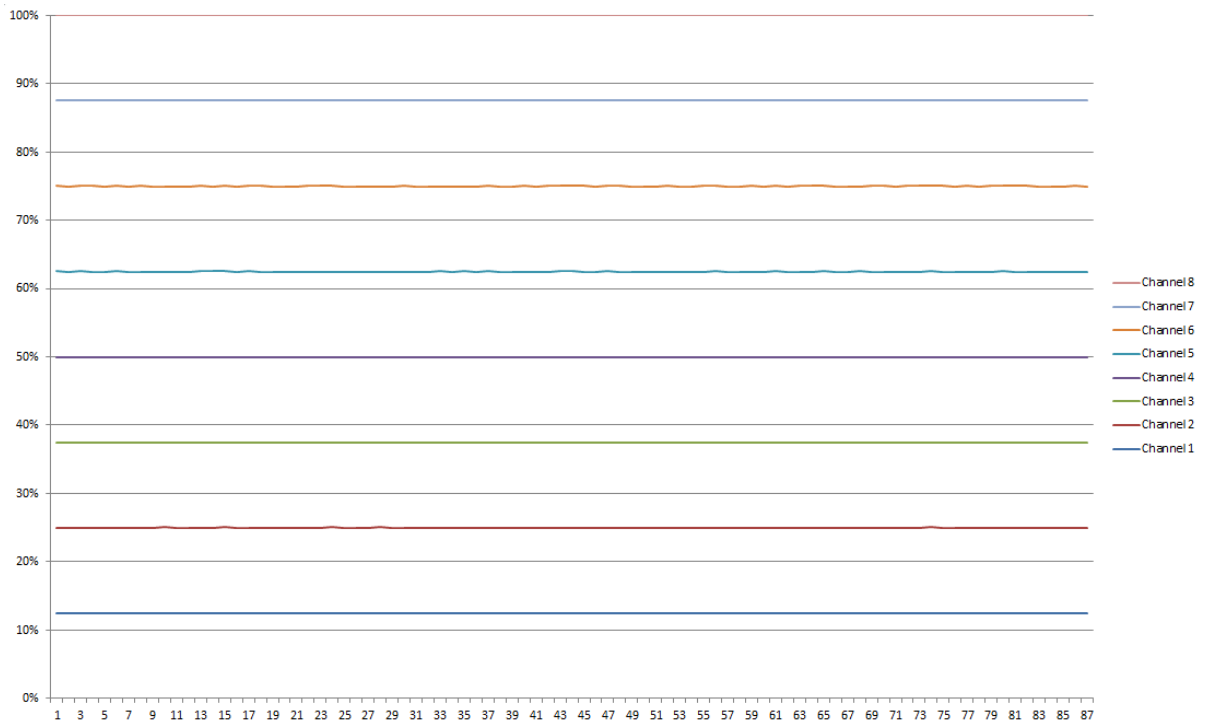
Ako je vrijeme snimanja postavljeno na 10 sekundi, a želi se prekinuti snimanje iz nekog razloga, moguće je pritisnuti stop. Nakon pritiska stop, aplikacija također sprema rezultate. Ali ako se stop pritisne dok traje vrijeme odgode snimanja, tada aplikacije ne kreira datoteku.

U svakoj datoteci piše točno vrijeme početka snimanja i na kraju u zadnjoj liniji se nalazi vrijeme završetka snimanja. Primjer je prikazan snimkom od 1 sekunde (Sl. 4.2.).



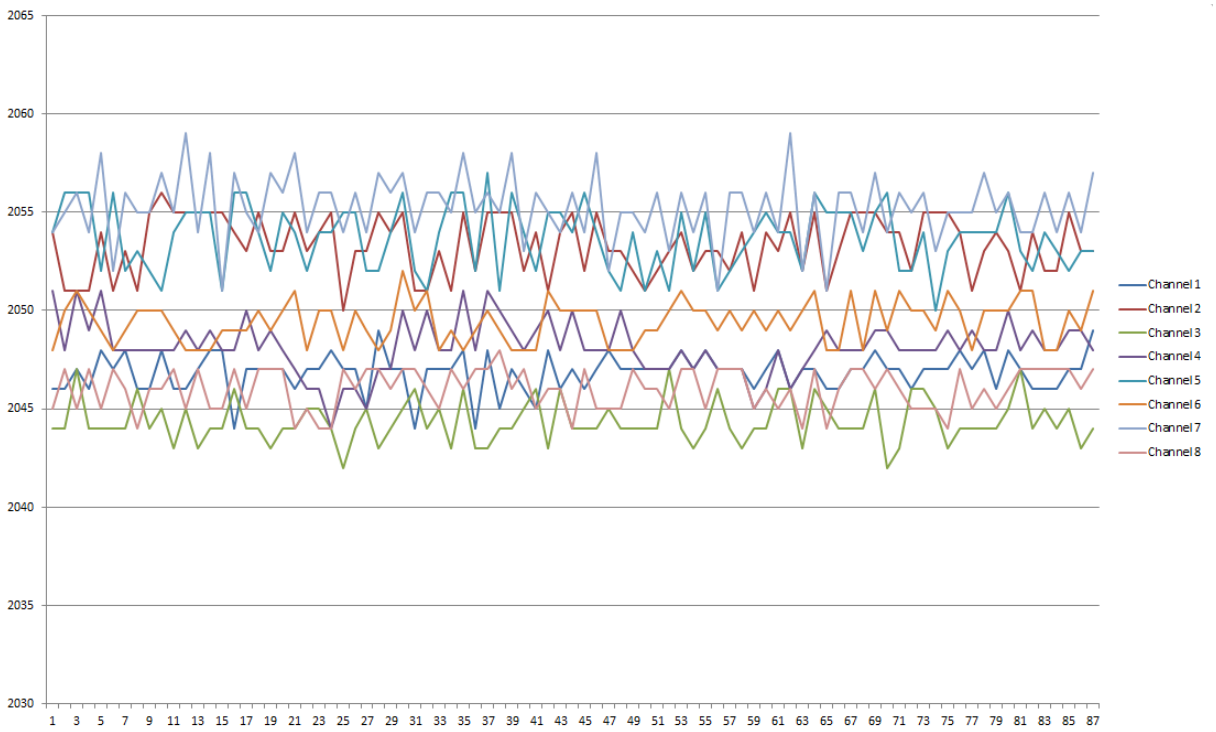
Sl. 4.2. *Rezultati snimanja u trajanju od 1 sekunde*

Na snimci se vidi razlika između tišine i glazbe. Na lijevoj strani je potpuna tišina tijekom snimanja, dok je na desnoj strani prikazano snimanje glazbe. Kada nema zvukova, tada na svakom kanalu dolazi vrijednost od oko 2048. Ako se proizvede nekakav zvuk, ta vrijednost će oscilirati. Da bi se to malo lakše vidjelo, ovaj primjer bit će prikazan pomoću grafova. Lijeva strana slike, tj. tišina u trajanju od jedne sekunde prikazana je slikom 4.3.

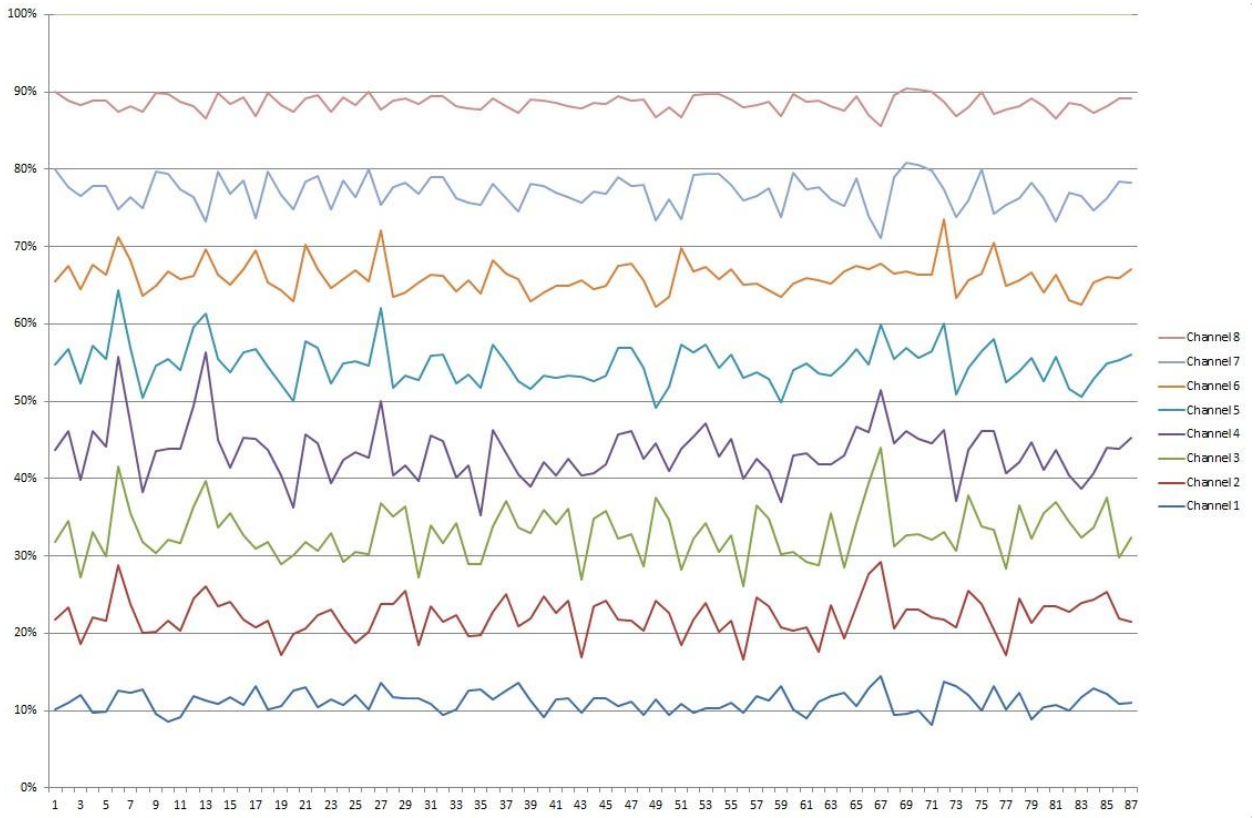


Sl. 4.3. *Grafički prikaz tišine po kanalima – linearan oblik*

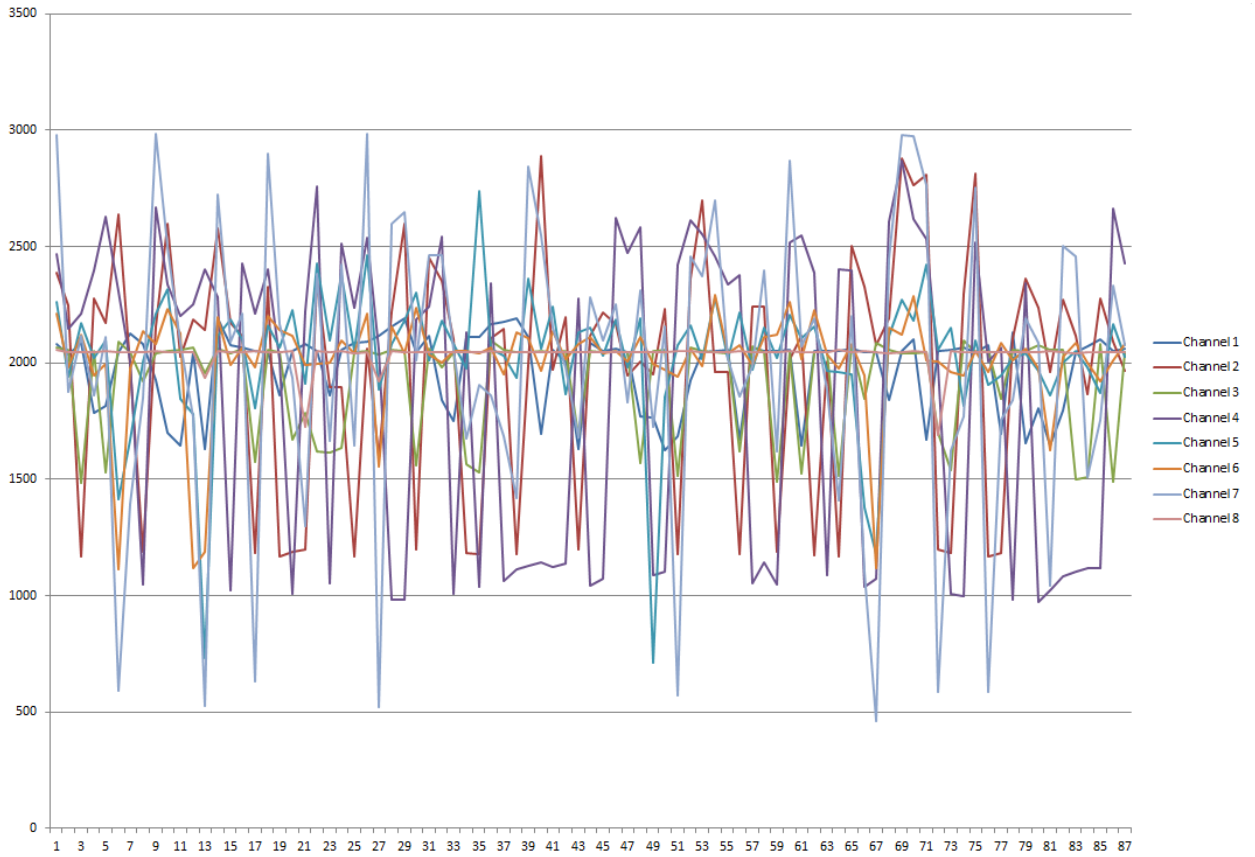
Na slici 4.3. moguće je vidjeti svih 8 kanala zasebno, svaki kanal je prikazan drugom bojom i grafički prikaz je linearan. Gotovo savršena tišina prikazana je kao pravac. Moguće je prikazati i ta mala odstupanja kada se svi grafovi stave kao prema slici 4.4. Odstupanja su od prilike od 2040 pa do 2060 što je tišina. Također je vidljivo da 1 sekunda sadrži 87 uzoraka.



Sl. 4.4. *Grafički prikaza gotovo savršene tišine*

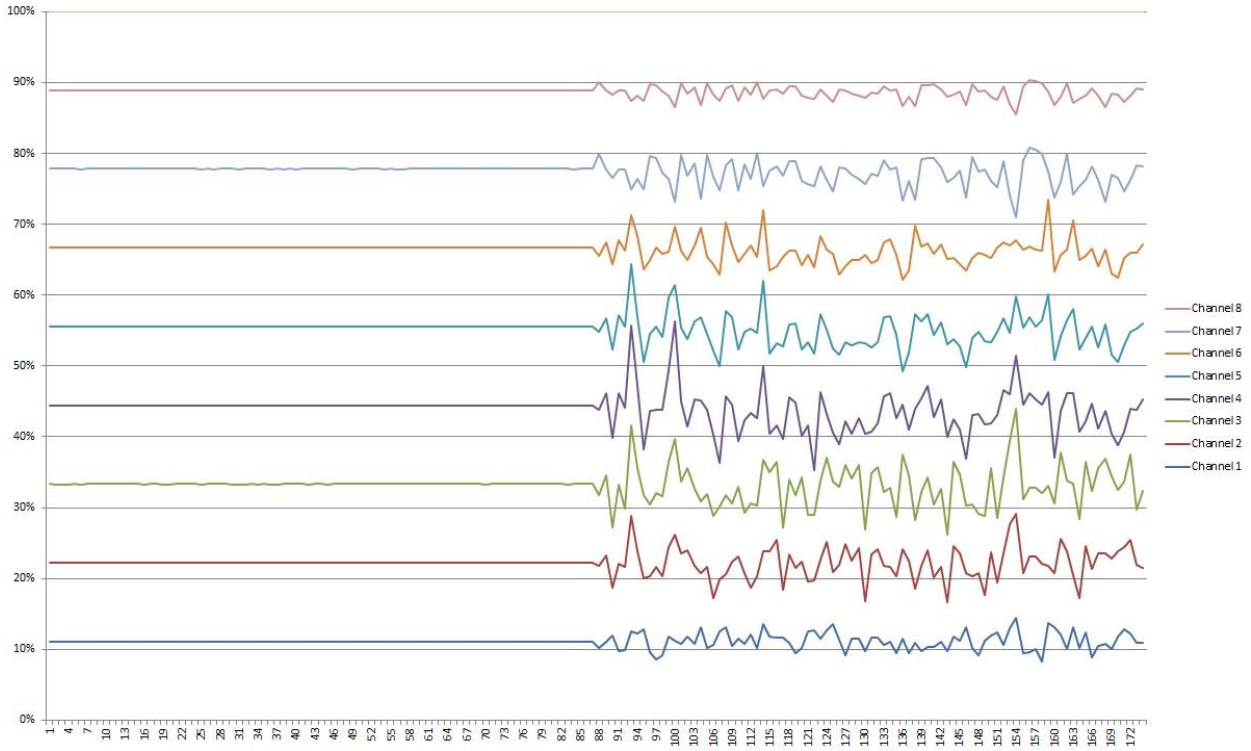


Sl. 4.5. *Grafički prikaz glazbe po kanalima*

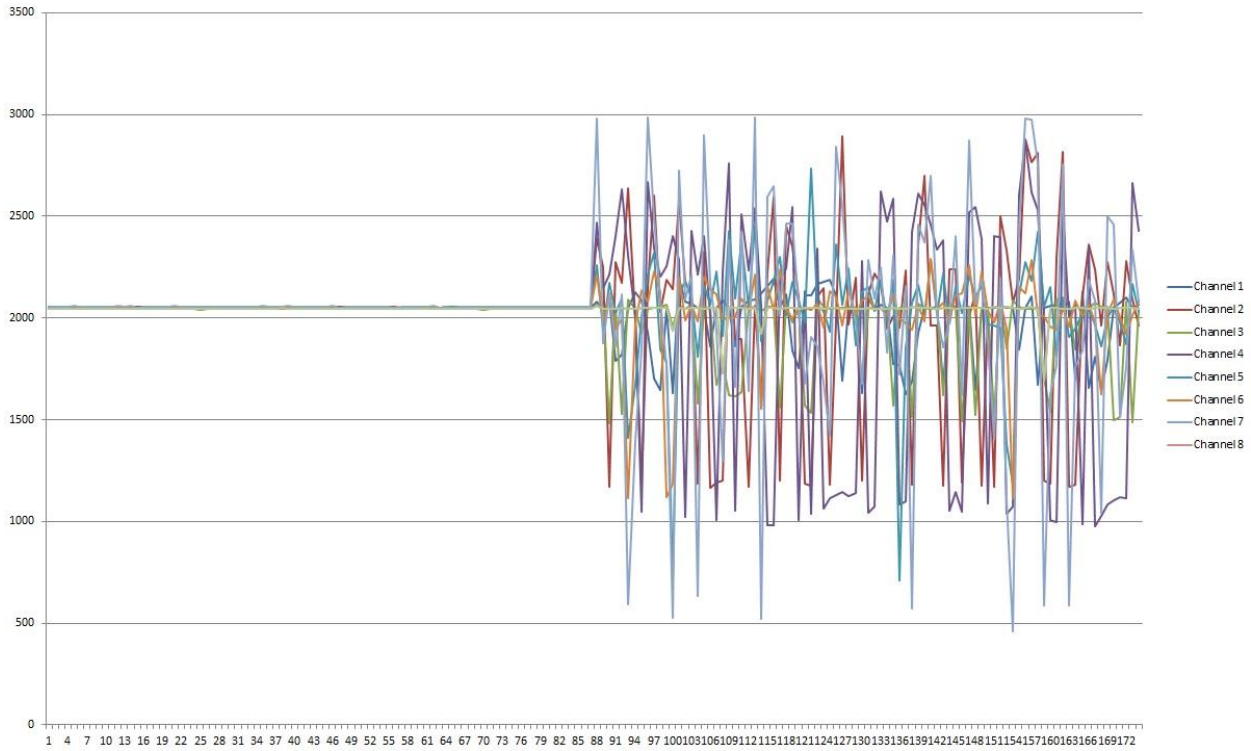


Sl. 4.6. *Grafički prikaz glazbe – svi kanali skupa*

Graf sa slike 4.5. prikazuje snimak glazbe (brojevi s desne strane slike 4.2.) za svaki kanal posebno i vidi se kako nije linearan kao na slici 4.3. Isto tako ako se grafovi svi stave na jedno mjesto kao što je to na slici 4.6. vidi se da su ovdje oscilacije od 500 pa čak do 3000 što se nikako ne može nazvati tišinom.

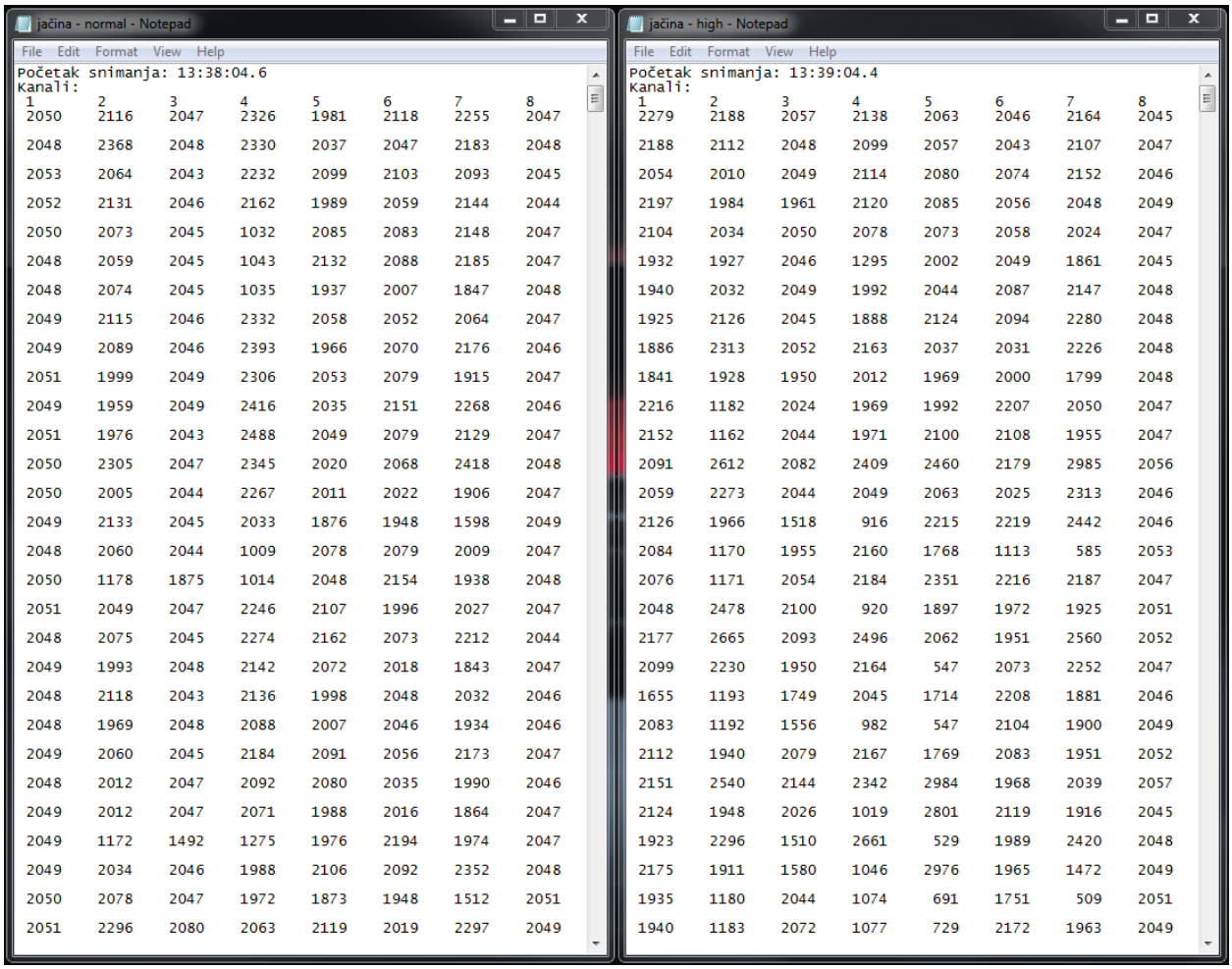


Sl. 4.7. *Grafički prikaz razlike između tišine i glazbe po kanalima*



Sl. 4.8. *Grafički prikaz tišine i glazbe svih kanala*

Rezultati snimanja sa slike 4.2. su uzeti i zajedno korišteni u izradi grafova sa slika 4.7. i 4.8. u želji da se pokaže koja je razlika između tišine i glazbe, a koju je malo teže vidjeti iz prvotnih brojeva. Naravno ako se bolje pogleda, može se i prema brojčanim rezultatima utvrditi jačina zvuka. Sljedeća slika pokazuje razliku između iste glazbe, ali različito pojačane.



jačina - normal - Notepad

File Edit Format View Help

Početak snimanja: 13:38:04.6
Kanali:

1	2	3	4	5	6	7	8
2050	2116	2047	2326	1981	2118	2255	2047
2048	2368	2048	2330	2037	2047	2183	2048
2053	2064	2043	2232	2099	2103	2093	2045
2052	2131	2046	2162	1989	2059	2144	2044
2050	2073	2045	1032	2085	2083	2148	2047
2048	2059	2045	1043	2132	2088	2185	2047
2048	2074	2045	1035	1937	2007	1847	2048
2049	2115	2046	2332	2058	2052	2064	2047
2049	2089	2046	2393	1966	2070	2176	2046
2051	1999	2049	2306	2053	2079	1915	2047
2049	1959	2049	2416	2035	2151	2268	2046
2051	1976	2043	2488	2049	2079	2129	2047
2050	2305	2047	2345	2020	2068	2418	2048
2050	2005	2044	2267	2011	2022	1906	2047
2049	2133	2045	2033	1876	1948	1598	2049
2048	2060	2044	1009	2078	2079	2009	2047
2050	1178	1875	1014	2048	2154	1938	2048
2051	2049	2047	2246	2107	1996	2027	2047
2048	2075	2045	2274	2162	2073	2212	2044
2049	1993	2048	2142	2072	2018	1843	2047
2048	2118	2043	2136	1998	2048	2032	2046
2048	1969	2048	2088	2007	2046	1934	2046
2049	2060	2045	2184	2091	2056	2173	2047
2048	2012	2047	2092	2080	2035	1990	2046
2049	2012	2047	2071	1988	2016	1864	2047
2049	1172	1492	1275	1976	2194	1974	2047
2049	2034	2046	1988	2106	2092	2352	2048
2050	2078	2047	1972	1873	1948	1512	2051
2051	2296	2080	2063	2119	2019	2297	2049

jačina - high - Notepad

File Edit Format View Help

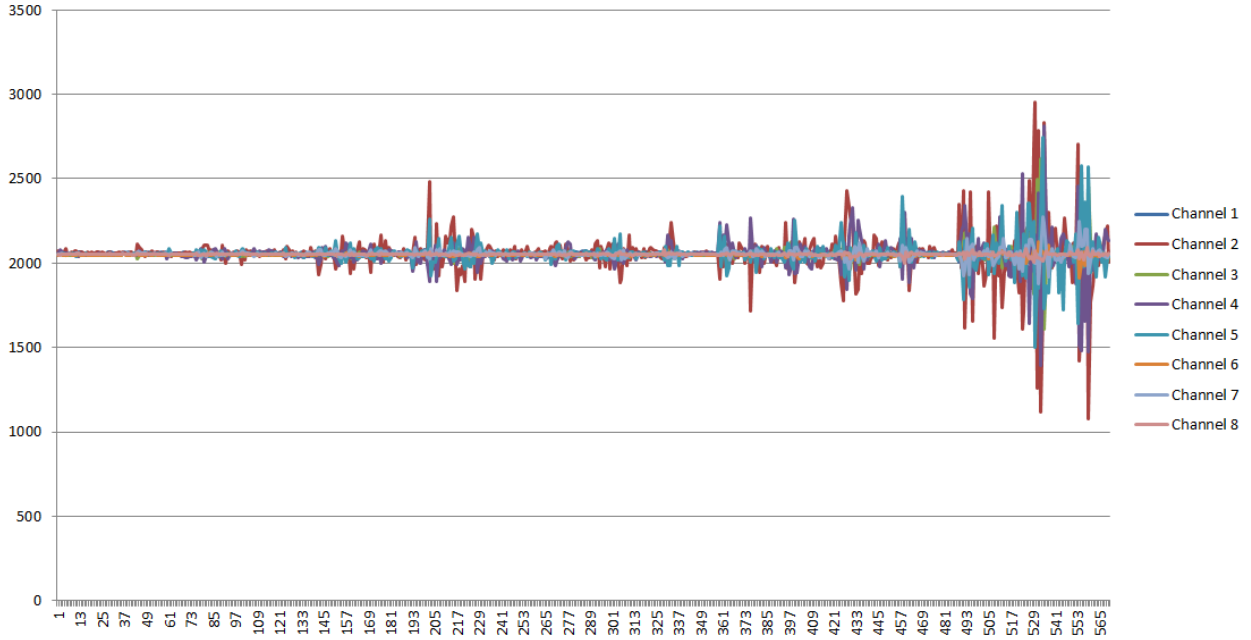
Početak snimanja: 13:39:04.4
Kanali:

1	2	3	4	5	6	7	8
2279	2188	2057	2138	2063	2046	2164	2045
2188	2112	2048	2099	2057	2043	2107	2047
2054	2010	2049	2114	2080	2074	2152	2046
2197	1984	1961	2120	2085	2056	2048	2049
2104	2034	2050	2078	2073	2058	2024	2047
1932	1927	2046	1295	2002	2049	1861	2045
1940	2032	2049	1992	2044	2087	2147	2048
1925	2126	2045	1888	2124	2094	2280	2048
1886	2313	2052	2163	2037	2031	2226	2048
1841	1928	1950	2012	1969	2000	1799	2048
2216	1182	2024	1969	1992	2207	2050	2047
2152	1162	2044	1971	2100	2108	1955	2047
2091	2612	2082	2409	2460	2179	2985	2056
2059	2273	2044	2049	2063	2025	2313	2046
2126	1966	1518	916	2215	2219	2442	2046
2084	1170	1955	2160	1768	1113	585	2053
2076	1171	2054	2184	2351	2216	2187	2047
2048	2478	2100	920	1897	1972	1925	2051
2177	2665	2093	2496	2062	1951	2560	2052
2099	2230	1950	2164	547	2073	2252	2047
1655	1193	1749	2045	1714	2208	1881	2046
2083	1192	1556	982	547	2104	1900	2049
2112	1940	2079	2167	1769	2083	1951	2052
2151	2540	2144	2342	2984	1968	2039	2057
2124	1948	2026	1019	2801	2119	1916	2045
1923	2296	1510	2661	529	1989	2420	2048
2175	1911	1580	1046	2976	1965	1472	2049
1935	1180	2044	1074	691	1751	509	2051
1940	1183	2072	1077	729	2172	1963	2049

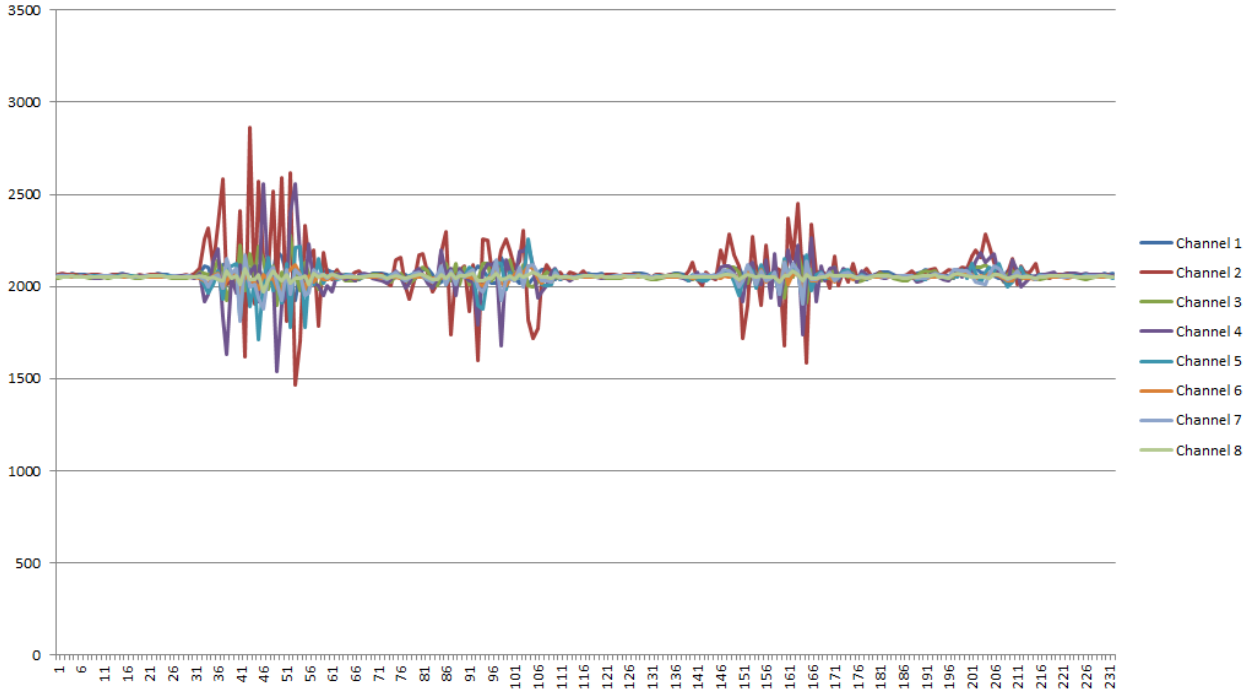
Sl. 4.9. *Snimak različite glasnoće*

Iz testiranja sa slike 4.9. može se vidjeti razlika u glasnoći. Na lijevoj strani je pjesma pojačana na 50% jačine, što se i vidi iz rezultata. Brojke osciliraju od 1000 pa do 2300, dok na desnoj strani, gdje je pojačano na 100% vidi se oscilacija od čak 500 pa do 2900.

Također je snimljen zvuk koji je pušten jako tiho te se postepeno pojačava s vremenom. Rezultati su vidljivi na slici 4.10.

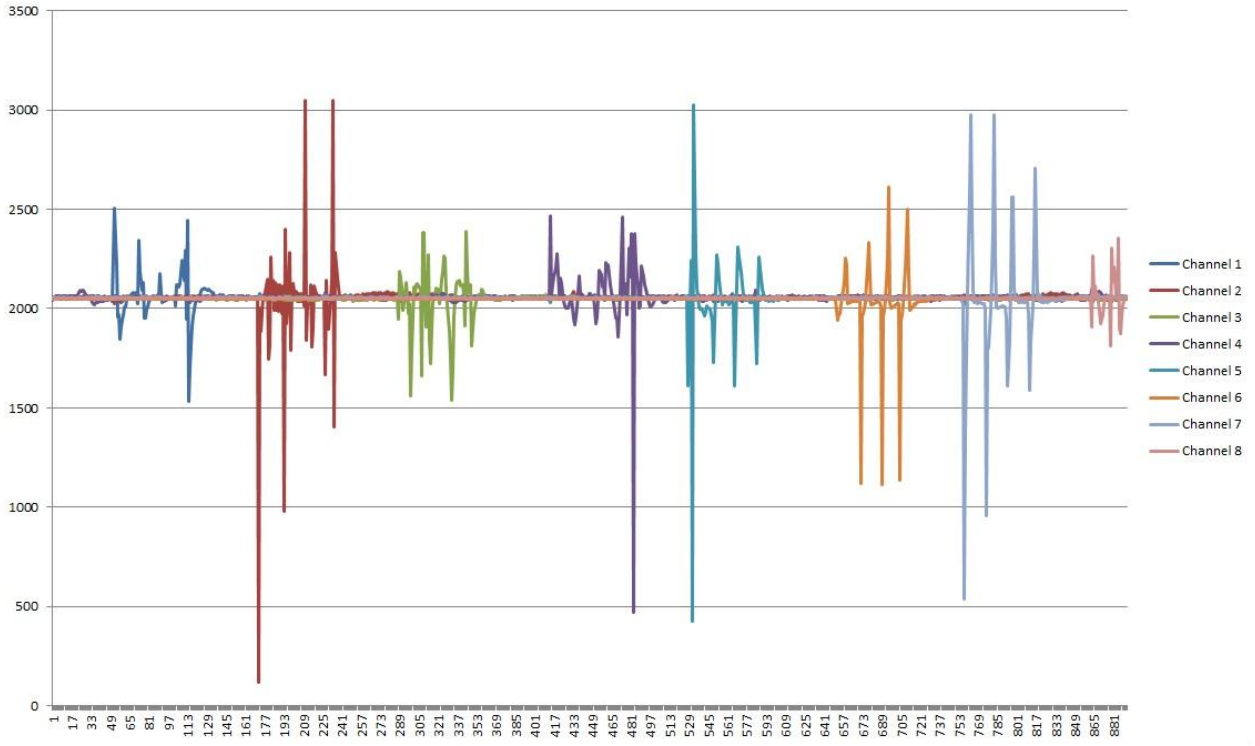


Sl. 4.10. *Grafički prikaz zvuka koji se postepeno pojačava*



Sl. 4.11. *Grafički prikaz govora*

Na slici 4.11. se vidi grafički prikaz govora. Prilikom snimanja izgovorena su prva 4 slova abecede, A, B, C i D. Podaci su zabilježeni i prema njima napravljen graf koji savršeno odgovara izgovorenim slovima i vremenskom razmaku prilikom izgovaranja.



Sl. 4.12. *Grafički prikaz koji otkriva koji je kanal primio zvuk*

Pošto su mikrofoni jedan uz drugoga, teško je pomoću puštanja glazbe utvrditi koji mikrofon je u tom trenutku primio zvuk, pa je to testirano na drugi način. Naime kada dirnemo mikrofon također proizvedemo određeni šum, samo što ga u tom slučaju samo taj mikrofon „čuje“. Test je protekao tako da se redom od prvog kanala pa do zadnjeg dodirnuo mikrofon. Rezultati su prikazani na slici 4.12. gdje se vidi da je prvo kanal 1 primio šum, pa kanal 2 i tako redom.

Kada bi se na maketu dodao *tri-state buffer* i kada bi paralelno snimanje bilo moguće, namjena makete bi mogla biti velika. Bilo bi moguće fokusirati mikrofone na jedan dio u prostoru i samo iz tog dijela primati zvuk. Ostale zvukove bi bilo moguće smanjiti. To je moguće jer ima više mikrofona, pa zvuk kada dolazi do jednog mikrofona, dolazi i do drugog, ali već kod drugog postoji kašnjenje u fazi. To kašnjenje se može izračunati s obzirom na to da je poznata brzina zvuka i onda bi bilo moguće pojačati val iz jednog smjera, a smanjiti iz drugog. Primjena bi bila slična konferencijskim mikrofonima, samo što bi se ovdje zvuk snimio paralelno i onda bi se taj snimak na računalu filtrirao. Prvi sirovi snimak bi sadržavao i zvukove sa strane, dok bi filtrirani sadržavao samo zvuk iz određenog dijela prostora. Takav proces zahtijeva veliku računalnu obradu i potrebno bi bilo imati veliki broj uzoraka pa bi bilo bolje koristiti DSP (engl. *Digital Signal Processing*) ili FPGA (engl. *Field-programmable gate array*). Bilo bi dobro i kada bi se koristili bolji mikrofoni, kao npr. mems mikrofoni, oni koji se ugrađuju u mobilne uređaje.

5. ZAKLJUČAK

U ovom radu je bio cilj napraviti maketu za paralelno snimanje zvuka s 8 kanala. Prvo je izrađen prototip makete s 2 kanala. Nakon testiranja je dizajnirana pločica u programskom paketu Eagle. Nakon što je izrađena pločica, zalemljene su sve SMT komponente. Lemljenje je vršeno uz pomoć kamere, da bi se jasnije vidjeli kontakti komponenata. Prilikom testiranja je ustanovljeno da bez *tri-state buffera* maketa ne može paralelno snimati zvuk i spremati ga na SD karticu. Odlučeno je da se snimanje i slanje podataka vrši putem serijske komunikacije. U tom slučaju nije potreban čitač SD kartice, pa je odlemljen s makete. Pošto je maketa stalno u prijenosu, odlučeno je izraditi zaštitno kućište. Kućište je dizajnirano u programskom paketu AutoCAD te izrađeno s 3D printerom. Na kraju je napravljena C# aplikacija u Visual Studio-u koja krajnjem korisniku olakšava interakciju s maketom. Napravljeno je sučelje za povezivanje (odabiranje COM porta i povezivanje) s maketom. Također je napravljen izbornik gdje se odabire koliko dugo će se snimati. Nakon završetka snimanja aplikacija spremi podatke u datoteku i moguće ih je kasnije čitati i od tih podataka napraviti grafički prikaz. Maketa prilično dobro radi, mikrofoni daju dobar odziv, rezultati su dobri, samo je mana to što ne prikuplja podatke paralelno kako je to bilo zamišljeno.

Maketu možemo poboljšati dodavanjem SRAM čipa ili jednostavnije korištenjem npr. Arduino Due, koji ima 96 KB SRAM-a te onda snimati paralelno i rezultate umjesto na SD karticu spremati u SRAM. Još bolja solucija bi bila koristiti DSP ili FPGA i umjesto electret mikrofona koristiti meme mikrofone (kao u mobilnim telefonima).

Namjena makete bi bila detektiranje izvora zvuka u polju mikrofona ili fokusiranje mikrofona na određeni dio prostora te izoliranje određenih zvukova tj.smanjenje šumova. Princip rada je sličan radu konferencijskih mikrofona.

LITERATURA

[1] Wikipedia, Arduino, URL: https://hr.wikipedia.org/wiki/Arduino, svibanj 2017.

[2] Arduino Nano, URL: https://www.arduino.cc/en/Main/ArduinoBoardNano, svibanj 2017.

[3] Electret microphone, URL: https://en.wikipedia.org/wiki/Electret_microphone, svibanj 2017.

[4] Operacijsko pojačalo, URL: https://hr.wikipedia.org/wiki/Operacijsko_poja%C4%8Dalo, svibanj 2017.

[5] Otpornik, URL: https://hr.wikipedia.org/wiki/Otpornik, svibanj 2017.

[6] Kondenzator, URL: https://hr.wikipedia.org/wiki/Elektri%C4%8Dni_kondenzator, svibanj 2017.

[7] Način spajanja SD kartice, URL: https://learn.sparkfun.com/tutorials/microsd-shield-and-sd-breakout-hookup-guide, srpanj 2017.

[8] Wikipedia, A/D pretvornik, URL: https://en.wikipedia.org/wiki/Analog-to-digital_converter, srpanj 2017.

[9] MCP3208, URL: http://ww1.microchip.com/downloads/en/DeviceDoc/21298c.pdf, srpanj 2017.

[10] Wikipedia, AutoCAD, URL: https://hr.wikipedia.org/wiki/AutoCAD, rujan 2017.

[11] Wikipedia, MS Visual Studio ,URL:https://bs.wikipedia.org/wiki/Microsoft_Visual_Studio, rujan 2017.

SAŽETAK

Naslov : Izrada 8-kanalnog uređaja za paralelno snimanje zvuka

U ovom diplomskom radu izrađen je hardver za paralelno snimanje zvuka pomoću 8 kanala. S tiskanom pločicom komunicira se Arduino Nano platformom. Dizajn je rađen u programskom paketu Eagle. Prije svega je rađen prototip makete na razvojnoj pločici. U konačnici, pločica snima zvuk s 8 kanala i upravljana je C# aplikacijom. Rezultati se spremaju u datoteku i prikazuju grafički.

Ključne riječi: Tiskana pločica, Snimanje zvuka, 8-kanalno snimanje, Arduino Nano, C#.

ABSTRACT

Title: Production of 8-channel device for parallel sound recording

In this graduation work is created hardware for parallel recording sound with 8 channels. It communicates with a printed circuit board with the Arduino Nano platform. The design was made in the Eagle program package. First of all, a prototype of a model on a development board was made. Ultimately, the board recorde sound with 8 channels and is controlled by the C# application. The results are stored in the file and displayed graphically.

Keywords: Printed circuit board, sound recording, 8 channel recording, Arduino Nano, C#.

ŽIVOTOPIS

Ivan Sabolski rođen je 26.10.1992. godine u Đakovu. Osnovnu školu pohađao je u razdoblju od 1999. do 2007. godine u Budrovcima. Od 2007 godine do 2011. je pohađao Srednju strukovnu školu Braće Radića u Đakovu, smjer Računalni tehničar za strojarstvo. Godine 2011. upisuje preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku. Preddiplomski studij računarstva završava 2014. godine završnim radom na temu „Performance Evaluation of Virtualization Tools in Multi-Threaded Applications“ koji je objavljen u časopisu *International Journal of Electrical and Computer Engineering Systems*. 5 (2015), 2; 57-62 (članak). Iste godine upisuje sveučilišni diplomski studij računarstva, smjer procesno računarstvo na fakultetu Elektrotehnike računarstva i informacijskih tehnologija u Osijeku. Položio je sve ispite te se trenutno nalazi pred obranom svog diplomskog rada.

PRILOG A – Arduino kod

```
#include <MCP3208.h>
#include <SPI.h>

MCP3208 adc(9);

void setup()
{
    adc.begin();
    Serial.begin(2000000);
}

void loop()
{
    char temp[8];
    for (int i = 0; i < 8; i++)
        {
            sprintf(temp, "%5d", adc.analogRead(i));
            Serial.print(temp);
            Serial.print(',');
        }
    Serial.println();
    delay(10);
}
```

PRILOG B – C# aplikacija

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.IO.Ports;
using System.IO;
using System.Threading.Tasks;

namespace SoundRecorder
{

    public partial class Main : Form
    {
        private SerialPort myport = new SerialPort();
        private string[] ports;
        private string comPort = "";
        private string recordings = "";
        private bool bRecording = false;
        private bool started = false;


        public Main()
        {
            InitializeComponent();
        }

        private void ScanComPorts()
        {
            ports = SerialPort.GetPortNames();
            selectedPort.Items.Clear();
            foreach (string port in ports)
            {
                selectedPort.Items.Add(port);
            }
            selectedPort.SelectedIndex = selectedPort.Items.Count - 1;
        }

        private void Main_Load(object sender, EventArgs e)
        {
            ScanComPorts();

            // Serial port
            myport = new SerialPort();
            myport.DtrEnable = true;
            myport.RtsEnable = true;
            myport.BaudRate = 2000000;
            myport.Parity = Parity.None;
            myport.DataBits = 8;
            myport.StopBits = StopBits.One;
            myport.PortName = comPort;
            myport.DataReceived += myport_DataReceived;
```

```
    startRecording.Enabled = false;
    D.Enabled = false;
    R.Enabled = false;
}

private void dropdown(object sender, EventArgs e)
{
    comPort = selectedPort.SelectedItem.ToString();
}

private void connect_Click(object sender, EventArgs e)
{
    if (myport.IsOpen)
    {
        myport.Close();
    }
    else
    {
        myport.PortName = comPort;
        myport.Open();
    }

    bool isPortOpen = myport.IsOpen;
    selectedPort.Enabled = !isPortOpen;
    startRecording.Enabled = isPortOpen;
    btnScan.Enabled = !isPortOpen;
    D.Enabled = isPortOpen;
    R.Enabled = isPortOpen;
    connect.Text = isPortOpen ? "Disconnect" : "Connect";
    bRecording = false;
    started = false;
}

private void startRecording_Click(object sender, EventArgs e)
{
    if (myport.IsOpen && !started)
    {
        int delay = int.Parse(D.Value.ToString());


        Task.Run(async () => {

            await Task.Delay(delay);

            recordings = "";
            updateRecordingDate("Početak snimanja: ");

            recordings += "Kanali: " + Environment.NewLine;

            for (int i = 0; i < 8; i++)
            {
                recordings += " " + (i + 1) + "\t";
            }
            recordings += Environment.NewLine;

            bRecording = true;
        });

        started = true;
```

```
                Task.Run(async () => {
                    await Task.Delay(delay + int.Parse(R.Value.ToString()));

                    writeRecordings();
                    D.Invoke(new Action(() => D.Enabled = true));
                    R.Invoke(new Action(() => R.Enabled = true));
                    startRecording.Invoke(new Action(() =>
startRecording.Text = "Start recording" ));
                });
            }
            else
            {
                writeRecordings();
            }

            D.Enabled = !started;
            R.Enabled = !started;
            startRecording.Text = started ? "Stop recording" : "Start
recording";
        }


        private void writeRecordings()
        {
            if (bRecording == true)
            {
                String dateRecorded =
DateTime.Now.ToString("dd_MM_yyyy_HH_mm_ss");

                updateRecordingDate("Kraj snimanja: ");

                File.WriteAllText("test_" + dateRecorded + ".txt",
recordings);
                bRecording = false;
            }
            started = false;
        }


        private void updateRecordingDate(String tag)
        {
            recordings += tag + DateTime.Now.ToString("HH:mm:ss.f");
            recordings += Environment.NewLine;
        }


         private void btnScan_Click(object sender, EventArgs e)
        {
            ScanComPorts();
        }


        private void myport_DataReceived(object sender,
System.IO.Ports.SerialDataReceivedEventArgs e)
        {
            string line = myport.ReadLine();
            BeginInvoke(new LineReceivedEvent(LineReceived), line);
        }

        private delegate void LineReceivedEvent(string line);
```

```
        private void LineReceived(string line)
        {
            if (bRecording) {
                // Ispisi mjerenja u konzolu
                //Console.WriteLine(line);

                String[] split = line.Split(',');
                foreach (string segment in split)
                {
                    recordings += segment + "\t";
                }

                recordings += Environment.NewLine;
            }
        }
    }
}
```

OSTALI PRILOZI

Ostali prilozi zajedno sa radom nalaze se na CD-u :

- Diplomski rad u .docx i .pdf formatu
- Arduino programska datoteka
- C# aplikacija
- Eagle shema i pločica
- AutoCAD model kućišta