Izrada MIDI kontrolera pomoću Arduino platforme

Zaplotnik, Jasna

SVEUČILIŠTE U RIJECI
ODJEL ZA INFORMATIKU

Preddiplomski jednopredmetni studij informatike

Jasna Zaplotnik

Izrada MIDI kontrolera pomoću Ardiuno platforme

Završni rad

Mentor:

Doc. dr. sc. Miran Pobar

Rijeka, rujan 2019

Sadržaj

1. Sažetak

U ovom završnom radu biti će opisan proces proveden od kreiranja ideje za temu rada pa sve do finalne izvedbe, testiranja i implementacije. Način izvedbe ovog jednostavnog MIDI kontrolera inspiriran je tereminom, ali nije svojim specifikacijama sličan tom fascinantnom električnom glazbenom instrumentu, već je jedna hibridna kreativna zamisao kojom sam koristeći tehnologiju niske razine htjela dočarati zanimljivost instrumenta kojim možemo proizvoditi zvukove bez da ga dodirujemo. Ideja tog starog glazbenog instrumenta kojom sam inspirirana prenesena je u moderno doba gdje je elektronički kreirana glazba na vrhuncu, te je uz pomoć MIDI standarda za komunikaciju elektroničkih uređaja, i malo ultrazvučne magije kreiran hibridni MIDI kontroler koji funkcionira moglo bi se najjednostavnije opisati - kao beskontaktna klavijatura. Spoj od 5 ultrazvučnih senzora i Arduino platforme služi kao sučelje za ulazne podatke, koji se zatim putem serijske komunikacije i MIDI standarda šalju u programske aplikacije koje sintetiziraju MIDI poruke i omogućavaju nam da na zvučnicima računala čujemo proizvedenu glazbu.

2. Uvod

Cilj ovog završnog rada bio je prikazati proces izrade MIDI kontrolera koristeći se Arduino platformom. Stvoren je jedan jednostavni mali računalni sustav za interakciju računala i okoline koji ima specifični zadatak pretvaranja podražaja iz vanjskog svijeta putem ultrazvučnih senzora - u glazbu. Motivacija za ovu vrstu projekta krenula je iz moje fascinacije ugradbenim računalnim sustavima, njihovom jednostavnošću i učestalošću u svakodnevnom životu, a opet i raznolikošću koju nude u obliku kreativnih mogućnosti koje se uz malo mašte i nekoliko elektroničkih komponenti pretvaraju u zanimljive i zabavne projekte.

U poglavljima koja slijede objasniti ću postepeno osnovne pojmove koji su srž projekta, te zatim i specifikacije konkretnih hardverskih i softverskih komponenti koje su korištene za izradu ovog modela MIDI kontrolera. Započinjem od širih pojmova kako bi pokušala ukratko prikazati gdje točno moj projekt leži na spektru računarstva i digitalne elektronike, a zatim se sve više fokusiram na odabrane komponente i njihovu međusobnu interakciju u sklopu cjeline.

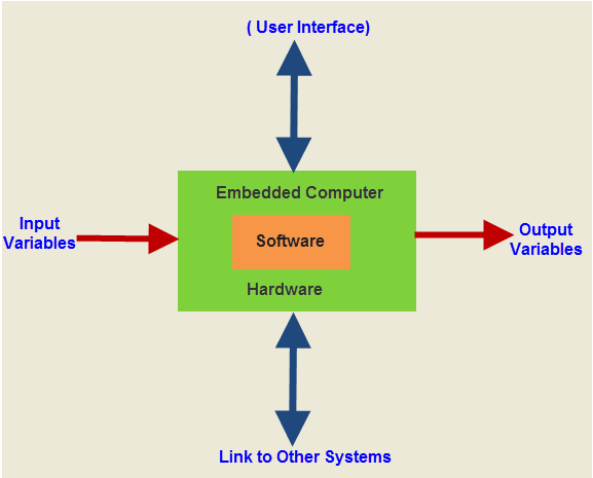
U sljedećem poglavlju predstavljen je koncept ugradbenih računalnih sustava i mikrokontrolera, te je opisana ideja serijskog komunikacijskog protokola između uređaja koji će se koristiti. Zatim se u idućim poglavljima osvrćem na Arduino platformu – njezine integralne djelove – softversku aplikaciju za pisanje programskog koda i razvojnu pločicu sa mikrokontrolerom, te u nastavku na ultrazvučne senzore koji zajedno sa Arduinom predstavljaju elektroničku srž ovog projekta. Nakon toga predstavljen je koncept MIDI protokola i MIDI uređaja koji se koriste u projektu, te je opisana njihova uloga u svijetu elektroničke glazbe. Naposlijetku opisane su i softverske aplikacije koje su korištene za virtualnu emulaciju ovog kontrolera i njihove konfiguracije, te je uz zaključak priložen i dodatak programskog koda koji se izvršava na Arduinu i pokreće cijeli proces stvaranje glazbe sa MIDI kontrolerom kojeg sam izradila.

3. Ugradbeni računalni sustavi

3.1. Općenito

Riječ sustav opisuje bilo koju organizaciju ili uređaj koji ima ulaz, izlaz i funkciju koja pretvara ulaz u izlaz. Ugradbeni računalni sustavi su kombinacija hardverskih komponenti i računalnog softvera, programabilni ili sa fiksnim programom, dizajnirani za specifičnu funkciju ili funkcije unutar nekog većeg sustava (Slika 1). Glavna karakteristika ugradbenog računalnog sustava je ta da obavlja specifičan zadatak pomoću ugrađenog sotverskog programa – *Firmware*-a. Osim toga njegove odlike su i reaktivnost na okolinu u stvarnom vremenu (s obzirom da su neki od njih ugrađeni i npr. u protupožarne alarme), mala veličina, ugrađena memorija i povezanost sa vanjskim svijetom kroz ulazne i izlazne varijable.

Mogu biti bazirani na mikroprocesorima ili mikrokontrolerima. U oba slučaja integrirani krug je središte priče. Razlika između mikroprocesora i mikrokontrolera je u tome što je mikrokontroler osposobljen za samostalan rad, dok mikroprocesor u sebi sadrži samo centralnu procesnu jedinicu i zahtjeva dodatne komponente kao što su vanjska memorija kako bi funkcionirao.



Slika 1 – Shema ugradbenog sustava u sklopu većeg sustava [1]

Ugradbeni sustavi mogu imati korisničko sučelje, ili u jednostavnijim izvedbama biti potpuno bez njega i samostalno obavljati svoj zadatak (npr. automatska rasvjeta). Neki najjednostavniji sustavi koriste tipkala, LED, LCD ekrane i sl., a neki sofisticiraniji mogu imati i ekrane na dodir ili biti povezani sa eksternim sučeljem putem serijske komunikacije ili mrežne konekcije. Moj projekt koristi Arduino platformu sa ATMega mikrokontrolerom kao bazom, ultrazvučne senzore kao sredstvo interakcije sa korisnikom, te serijsku komunikaciju sa računalom.

3.2. ATMega mikrokontroleri

Mikrokontroleri su malena računala koja na jednom čipu objedinjuju i integriraju memoriju, CPU, periferije i I/O sučelje. Ovdje se želim ukratko fokusirati na arhitekturu mikrokontrolera, i to specifično AVR arhitekturu jer se upravo ta koristi u Atmelovim ATMega mikrokontrolerima koji su „mozak“ Arduina – baze ovog projekta. AVR mikrokontroleri su 8-bitni RISC mikrokontroleri modificirane harvardske arhitekture. Oni na sebi sadržavaju: ALU (Aritmetičko logičku jedinicu), 32 registra opće namjene, registre specijalne namjene, SRAM, EEPROM i flash memoriju, timer/counter, komparator, watchdog timer, UART, SPI i I2C protokole, rukovatelj prekidima, te digitalne i analogne I/O pinove. AVR mikrokontroleri imaju ugrađen generator takta (eng. *Clock)* u obliku kristalnog oscilatora, koji određuje tempo izvođenja instrukcija. Na Arduino Uno razvojnoj pločici koju koristim nalazi se ATMega328 mikrokontroler čije ću specifikacije ukratko predstaviti.

3.2.1. Memorija

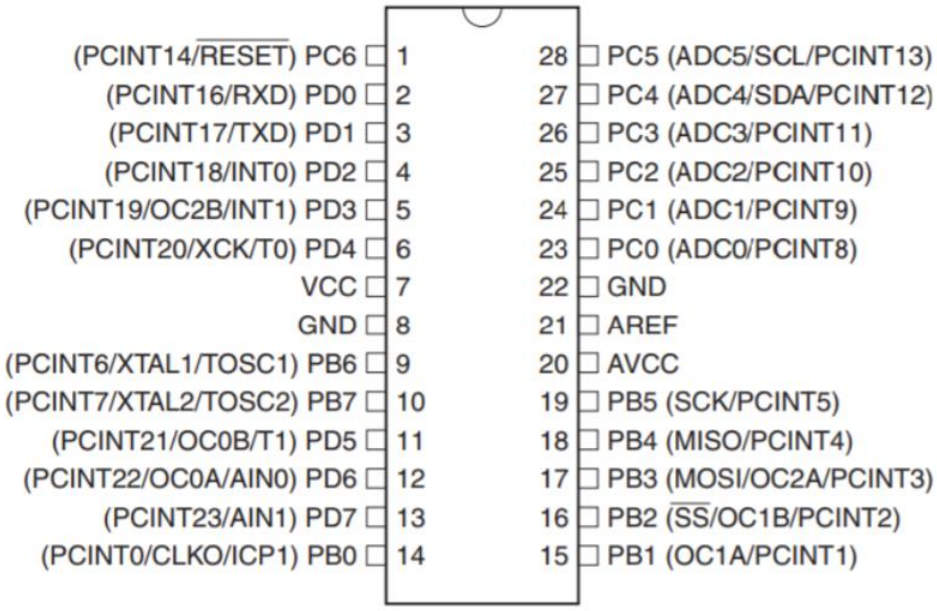
Flash, EEPROM (*Electrically Erasable Programmable Read Only Memory*) i SRAM (*Static Random Access Memory*) memorije integrirane su na čip. Programske instrukcije pohranjuju se u postojanu (eng. non-volatile) flash memoriju, čija se veličina obično indicira u samom nazivu uređaja. Tako npr. ATMega64x mikrokontroleri imaju 64KB flash memorije na raspolaganju. ATMega328 ima 32KB flash memorije od čega se 0.5KB koristi za bootloader, i 2KB SRAM, te 1KB EEPROM memorije.

3.2.2. GPIO portovi

GPIO označava *General Purpose Input/Output ports* (prev. ulazno/izlazni portovi opće namjene) na mikrokontoleru. Svaki od njih odnosi se na 8 pinova i kontroliraju ga tri različita 8-bitna registra. DDRx – *Data Direction Register* za port x kojim se određuje da li će pinovi biti ulazni ili izlazni. PORTx izlazni registar kojim se postavljaju vrijednosti na pinove porta x koji su konfigurirani kao izlazni, i PINx ulazni registar koji se koristi za čitanje ulaznih signala na portu x sa pinova koji su konfigurirani kao ulazni pinovi („x“ označava jedan od dostupnih portova i zamjenjuje njihovu abecednu oznaku u primjeru). To je logika iza rada sa pinovima, ali koristeći ATMegu sa Arduino platformom taj dio je pojednostavljen u programskog kodu i ne zahtjeva specifiziranje na ovako detaljnoj razini, kako će biti kasnije opisano. Konfiguracija pinova ATMega mikrokontrolera prikazana je na shemi u nastavku (Slika 2).

3.2.3. Komunikacija s periferijom

ATMega pruža nekoliko mogućnosti za komunikaciju sa periferijom, odnosno računalom, senzorima, modulima, drugim Arduino pločicama ili drugim mikrokontrolerima. Putem digitalnih pinova Tx i Rx omogućena je UART TTL serijska komunikacija. ATMega328 podržava i I2C, te SPI komunikaciju uz pomoć za to predviđenih biblioteka.



Slika 2 – konfiguracija pinova ATMega328 mikrokontrolera [2]

3.3. Serijska komunikacija

Ugradbeni računalni sustavi integriraju zajedno različite komponente koje vrše međusobnu interakciju. Za njihovu međusobnu komunikaciju, kako bi se svi razumjeli, potrebno je definirati zajednički komunikacijski protokol. Stotine komunikacijskih protokola definirano je kako bi se postigla efikasna razmjena podataka. Svi protokoli mogu se razvrstati u dvije skupine – serijske i paralelne.

U ovom projektu koristiti ćemo serijsku komunikaciju pa ću je ukratko opisati u ovom potpoglavlju. Dok paralelna komunikacija koristi slanje više bitova informacija odjednom, kroz odvojene fizičke vodove, serijskom komunikacijom podaci se prenose jedan po jedan bit, preko jednog fizičkog voda podataka. Iako je paralelna komunikacija u suštini brža od serijske, pogotovo kod programiranja mikrokontrolera češšće se koristi serijska, zbog ograničene količine pinova, tako da se žrtvuje potencijalno brže izvođenje za fizički prostor.

Postoje dvije vrste serijske komunikacije – sinkrona i asinkrona. Sinkrona komunikacija kontrolirana je generatorom takta (eng. *clock)*. Ovaj način općenito pruža jednostavniji i brži prijenos podataka ali zahtjeva barem još jednu fizičku žicu. Primjeri sinkronih komunikacijskih protokola su SPI i I2C. Asinkrona komunikacija je upravo suprotna, i nije oređena generatorom takta, a njezin primjer je UART.

Svaki blok podataka koji se šalje serijskom komunikacijom zapisuje se u formatu koji se naziva okvir (eng. frame) (Slika 3). Unutar okvira nalazi se nekoliko informacija zapisanih u bitovima. Prvi bit je uobičajeno startni bit, a zadnji stop bit, koji dogovoreno označavaju početak i kraj okvira. Osim toga glavna cjelina okvira su podatkovni bitovi stadardne veličine jednog bajta (moguće su varijacije u broju bitova). Osim veličine podatkovnog bloka, komunikacijski protokol

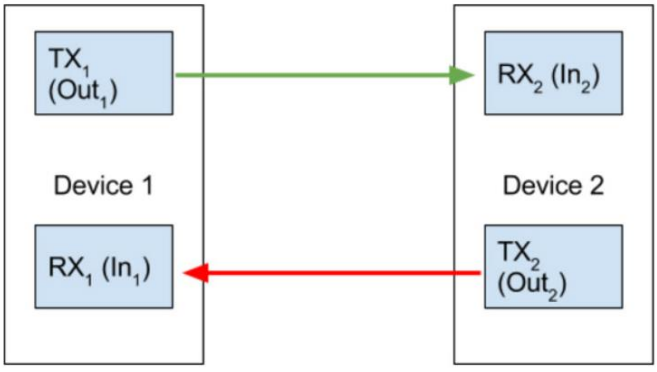
odgovoran je i za dogovor o poretku bitova, tj. odlučuje se na kojem mjestu će se nalaziti MSB (Most Significant Bit) a na kojem LSB (Least Significant Bit). Ako nije drukčije naglašeno, LSB se uobičano nalazi na prvom mjestu podatkovnog bloka.

Nakon podatkovnog bloka, a prije zaustavnog bita, u okviru se nalazi i opcionalni paritetni bit. On može biti 0 ili 1 i označava paritet podata koji se šalju. Koristi se kao jednostavna provjera pogreške. Taj bit se zapisuje na temelju zbroja količine jedinica u podatkovnom bloku, neparan broj jedinica postavlja paritetni bit na 1, a paran na 0.

Slika 3 – primjer serijskog okvira za slanje podataka [3]

Kod serijskih protokola bitno je naglasiti da obje strane komunikacije moraju biti podešene na jednaki protokol kako bi se podaci koji se prenose razumjeli, jer će inače biti samo nerazumljivi bajtovi nula i jedinica. Bitna stvar kod serijske komunikacije je *baud rate*, i označava brzinu prijenosa. Uobičajeno se izražava u bitovima po sekundi (bps). Baud rate obje strane komunikacije mora biti jednako podešen, a najčešće korištena brzina je 9600 bps. MIDI protokol koji ću u projektu koristiti za komunikaciju ima baud rate od 31250 bps.

Komunikacijski protokol koji se koristi kod Arduina putem pinova Tx i Rx je UART. UART je kratica za engleski naziv *Universal Asynchronous Reciever/Transmitter*, a označava serijski asinkroni protokol za odašiljanje i primanje podataka. Povezivanje u UART protokolu je jednostavno, Rx (prijamnik, eng. *reciever*) jedne strane spaja se sa Tx (odašiljačem, eng. *transmitter*) druge strane komunikacijske veze (Slika 4). Protokol je namjenjen za komunikaciju između samo dva uređaja istovremeno zbog nemogućnosti razlikovanja dolaznih podataka sa različitih izvora.



Slika 4 - UART primjer fizičkog spajanja [4]

Kako je UART asinkroni protokol, podaci koji se šalju formatiraju se u točno specifične veličine blokova podataka kako bi se osigurao nesmetan prijenos. Podaci koji se šalju putem ovakvog protokola malo su spojiri od sinkronih protokola zato što se uz podatke, šalju i dodatne informacije o samom prijenosu. Kod korištenja ovog protokola na platformama poput Arduina, korisnik se ne mora opterećivati detaljima oko prijenosa, već može koristiti predefnirane biblioteke *Serial* i *SoftwareSerial* kako bi implementirao UART komunikaciju u svoj projekt.

Osim UART-a, Arduino za komunikaciju s periferijom može koristiti i sinkrone I2C i SPI protokole.

4. Arduino

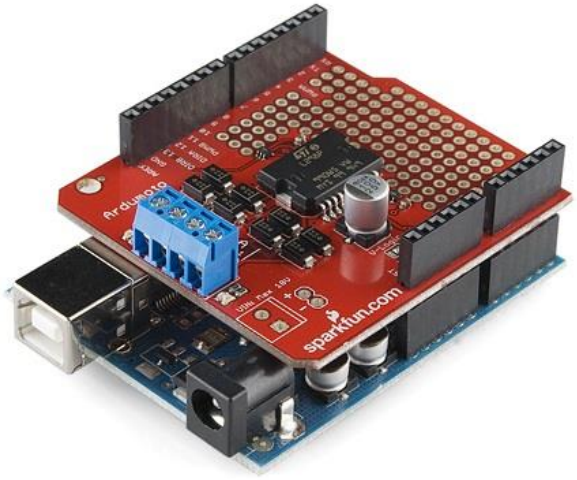
Arduino je platforma otvorenog koda posvećena elektronici i izradi elektroničkih projekata, bazirana na hardveru i softveru koji je lak i jednostavan za korištenje. U svojoj suštini sastoji se od fizičkih komponenti – programabilne mikrokontrolerske razvojne pločice, i softverske aplikacije Arduino IDE (en. Integrated Development Environment) koja se koristi za pisanje koda i prebacivanje istog na razvojnu pločicu pomoću USB kabela. Tokom godina, Arduino je postao i zajednica studenata, hobi elektroničara, umjetnika i drugih profesionalaca čija je kontribucija omogućila nastajanje vrlo velike baze podataka akumuliranog znanja, podržavajući koncept otvorenog koda.

Ideja Arduina začeta je 2005. godine na Ivrea Interaction Design instututu u Italiji, kao jednostavan alat koji bi bio dostupan kao alternativa postojećim alatima, studentima bez pozadine u elektronici i programiranju, za laku izradu raznih projekata. Danas, studenti i profesori koriste Arduino platformu za izradu prototipa znanstvenih instrumenata u raznim područjima, kao i za projekte vezane uz robotiku, automatizaciju i sl. Niska cijena komponenti, šarolik izbor različitih razvojnih pločica, starter kitova i dodatnih modula, te jednostavnost programskog jezika, otvoren kod i kompatibilnost sa različitim drugim platformama, softverskim aplikacijama i hardverskim komponenatama, glavni su razlozi zašto je Arduino danas toliko raširen i popularan.

4.1. Hardverske komponente

Osim softvera otvorenog koda, i Arduino hardverske komponente klasificiraju se kao open-source hardware, što znači da su informacije o njihovoj izradi javno dostupne svima i slobodne za rekreiranje. Tvrtka je licencom zaštitila samo ime Arduino, tako da sve druge kopije ne smiju koristiti njihovo ime u svojem nazivu. Osim što je hardver dostupan za rekreiranje, i što se zajednica Arduino korisnika svakodnevno nadopunjuje sa projektima koje drugi mogu koristiti, postoji čak i mogućnost da se razne komponente izvedene od strane drugih proizvođača/entuzijasta ukomponiraju i u službeni proizvod ukoliko ispunjavaju određene zahtjeve i odrednice, tako da zapravo svatko može doprinijeti na neki način.

Arduino komponente sastoje se od klasičnih razvojnih pločica sa jednostavnijim ili naprednijim značajkama, edukacijskih starter kitova, IoT komponenti i mekih pločica koje se mogu koristiti u projektima koji zahtjevaju hardversku fleksibilnost. Većina klasičnih Arduino razvojnih pločica bazira se na Atmelovim AVR ATMega mikrokontrolerima sa varijabilnim količinama flash memorije, pinova i drugih značajki. Pločice koriste jedan do dva reda pinova sa ženskim ili muškim nastavcima koji služe za spajanje sa drugim proizvoljnim krugovima, senzorima ili nekima od gotovih Arduino modula koji se nazivaju *shields* (prev. štitovi), i s kojima se mogu omogućiti razne funkcionalnost kao npr. povezivanje na internet sa Wi-Fi modulom i sl. (Slika 5). Većina pločica koristi ulazni napon od 5V i 16MHz kristalni ili keramički oscilator.



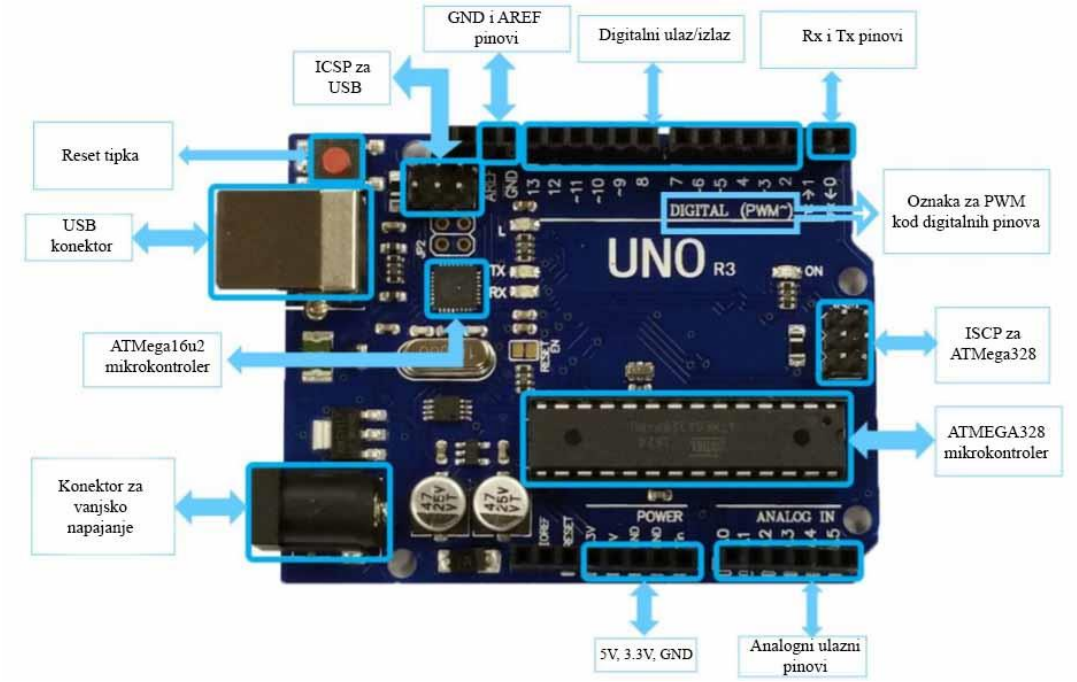
Slika 5 - primjer Arduino modula - Arduino Motor Driver Shiled [5]

Arduino razvojne pločice imaju unaprijed ugrađen firmware (eng. *bootloader*) koji pojednostavljuje učitavanje koda u flash memoriju mikrokontrolera tako da nije potrebna međukonekcija računala i Arduina programatorom, iako je ona po želji i moguća zaobilaženjem bootloadera. Programski kod se na Arduino učitava pomoću serijske konekcije USB kabelom na računalo uz USB-to-serial čip ili firmware koji je adaptiran na pločicu. Iako postoji mnogo različitih vrsta Arduino razvojnih pločica i modula, Arduino Uno je jedna od najčešće korištenih. Nju sam odabrala kao osnovu svojeg projekta te ću je detaljnije opisati u nastavku.

4.2. Fizički izgled i funkcionalnosti razvojne pločice Arduino Uno

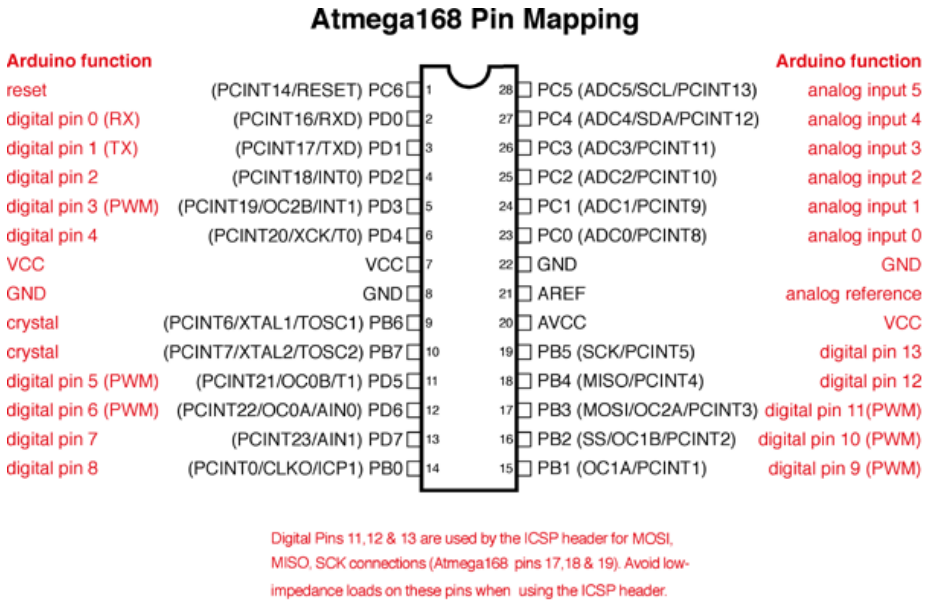
Mozak razvojne pločice Arduino Uno je ATMEGA328P-PU, 8-bitni mikrokontroler koji je opisan u prethodnom poglavlju. Pločica sadrži konektore za USB kabel i regularno vanjsko napajanje, 14 digitalnih ulazno/izlaznih pinova (od kojih se 6 može koristiti za PWM), 6 ulaznih analognih pinova, 16MHz kristalni oscilator koji se koristi za generator takta, dvije ICSP konekcije (jedna za USB, a druga za ATMegu328), pinove za napajanje vanjskih komponenti, Rx i Tx pinova za serijsku komunikaciju i ATMega16u2 mikrokontroler koji služi kao USB driver.

Uno radi na naponu od 5V koji dobiva putem USB kabela ili preko vanjskog napajanja sa adapterom, ali isto tako može ga se napajati i putem vanjske baterije koja se može spojiti na Vin i GND pinove. Preporučeni ulazni napon je 7-12V, dok su mu maksimalna ograničenja 6-20V. Na svojim digitalnim I/O pinovima ima ograničenje od maksimalno 40mA istosmjerne struje. Na pločici se nalaze i dva ICSP (In-Circuit Serial Programming) seta pinova, jedan za USB konekciju (ukoliko usb konektor nije dostupan) i jedan kojeg je moguće spojiti na drugi ATMega328 mikrokontroler putem kojeg se može upravljati Arduino. Detaljniji prikaz Arduino Uno pločice (Slika 6) i spajanja pinova mikrokontrolera ATMega328 na pločicu (Slika 7) dani su u nastavku.



Slika 6 - razvojna pločica Arduino Uno [5]

- *USB konektor* – služi za spajanje Arduina sa računalom putem USB veze i kao vanjsko napajanje
- *dodatno vanjsko napajanje* – Arduino se može koristiti i uz obično napajanje koje se ukopčava direktno u utičnicu
- *GND* (en. Ground – uzemljenje), pinovi koji služe za spajanje mase
- *5V i 3.3V*, pinovi koji omogućuju napajanje od 5 ili 3.3 volta na vanjske strujne krugove spojene na Arduino
- *analogni pinovi* pinovi od A0 do A5 označvaju pinove koji mogu očitavati mjerenja sa analognih senzora poput temperature i pretvarati ih u digitalne čitljive vrijednosti
- *digitalni pinovi* od A0 do A13 koji mogu biti korišteni ili za digitalni ulaz (npr. da li je tipkalo pritisnuto) ili za digitalni izlaz (npr. osvjetljavanje LED diode putem programskog koda)
- *PWM*, digitalni pinovi sa notacijom ~ pored brojčane oznake. To su pinovi koji po potrebi imaju i dodatnu funkciju generiranja PWM (*pulse-widht modulated)* signala, odnosno korištenja tehnike širinsko-impulse modulacije koja je opisana u prethodnom poglavlju
- *AREF* -koristi se za konfiguraciju referentnog napona za analogni ulaz
- *reset tipka* koja pritiskom ponovno pokreće programski kod koji je učitan na pločicu
- *ATMega16u2 mikrokontroler* – služi kao USB driver koji omogućuje USB konekciju na pločicu
- *ATMega328 mikrokontroler*, mozak Arduina



Slika 7 - ATMega328P shema pinova referentno sa pinovima Arduino pločice [6]

4.3. Arduino IDE

IDE je kratica za engleski naziv Integrated Development Envirinment i označava integrirano razvojno okruženje. Sastoji se od tekstualnog editora za pisanje koda, sekcije sa porukama kompajlera, tekstualne konzole, alatne trake sa gumbima za često korištene funkcionalosti te meni trake. Arduino razvojno okruženje koristi se za spajanje sa mikrokontrolerom putem USB veze, pisanje koda i debuggiranje, prevođenje koda, te prenošenje i spremanje programa u flash memoriju pločice kako bi se zatim mogao izvoditi. Slobodno je za korištenje i preuzeto je sa službenih stranica (https://www.arduino.cc/en/main/software). Primjer Arduino IDE grafičkog sučelja dan je u nastavku (Slika 8).

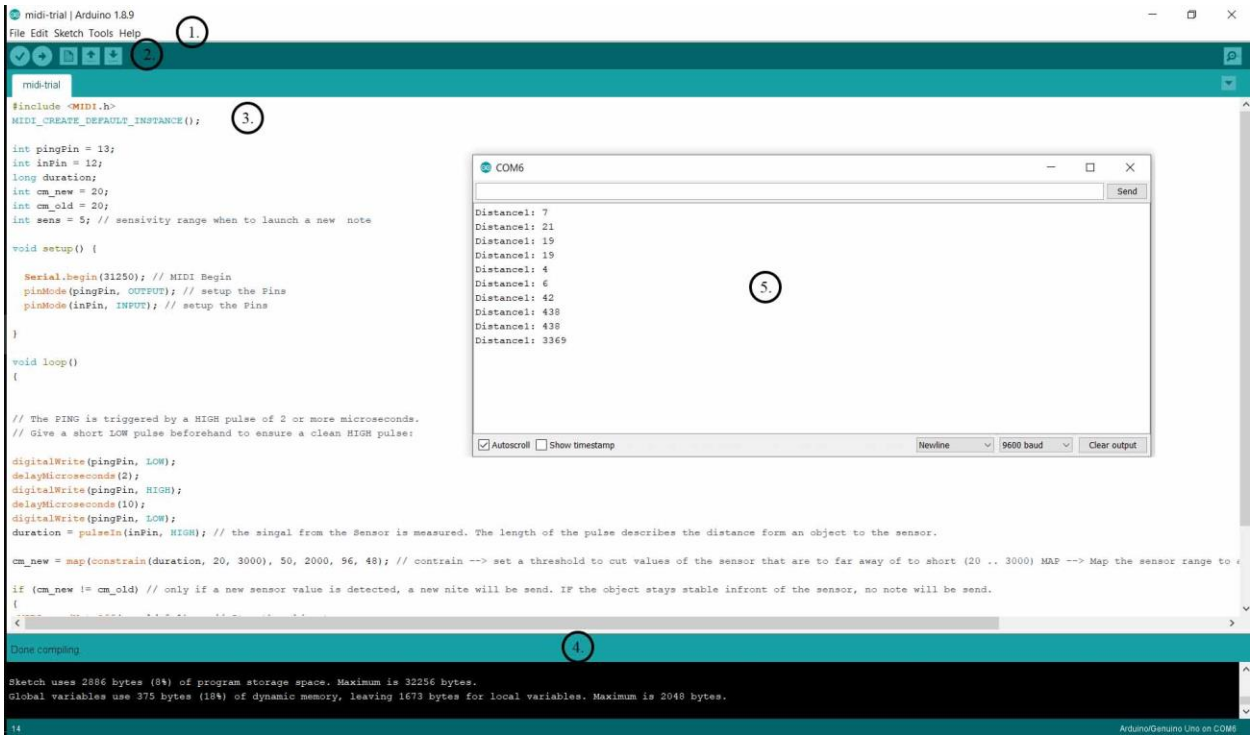
Programski jezik kojim se koristimo za programiranje Arduina je pojednostavljeni C++. Programi za Arduino koji se pišu u Arduino IDE nazivaju se još i *sketches* (en. skice) i spremaju se u računalu kao datoteke sa nastavkom *'.ino'*. Minimalna .ino skica mora sadržavati dvije funkcije: **setup()** i **loop().** *Setup()* je funkcija koja se poziva jednom i to na samom početku, i koja se koristi za incijalizaciju varijabli, ulaznih i izlaznih pinova i drugih biblioteka koje su potrebne u programu. Analogna je klasičnoj funkciji *main()* u C ili C++ programskom jeziku. *Loop()* se poziva nakon što se izvrši programski kod funkcije *setup(),* i njezin kod se izvršava opetovano dokle god je Arduino priključen na napajanje. Analogna je funkciji *while(1).*

Zahvaljujući standardnim Arduino bibliotekama koje omogućuju komunikaciju sa periferijom, glavne funkcije koje se koriste za manipulaciju i konfiguraciju pinova su ***pinMode(), digitalWrite(),*** te ***digitalRead(),*** koji pojednostavljuju programsko postavljanje port registara (PORTx,PINx i DDRx) naspram klasičnog programiranja AVR mikrokontrolera bez Arduino razvojne okoline. Analogni pinovi automatski su postavljeni kao ulazni pa ih nije potrebno konfigurirati, sa njih se čitaju vrijednosti pomoću funkcije *analogRead(),* dok se analogni izlazi na periferiju ispisuju pomoću funkcije *analogWrite()* i to samo na digitalne pinove koji su konfigurirani za širinsko impulsnu modulaciju (PWM).

Signali očitani sa analognog ulaza poprimaju u kodu vrijednosti između 0 i 1023, i preslikavaju se u napon između 0 i 5V. PWM pinovi mogu se koristiti kao analogni izlaz gdje se preslikavanjem vrijednosti u napon može dobiti privid analognog signala.

Biblioteke pružaju dodatne funkcionalnosti pri programiranju, kada je potrebno npr. manipulirati podacima ili komunicirati sa određenim fizičkim komponenatama. Biblioteke se u skicu učitavaju klasičnom *#include* naredbom na vrhu koda, ili se kroz meni traku mogu jednostavno učitati uz nekoliko klikova kada će IDE za nas upisati potrebne naredbe. Biblioteke se zajedno sa programskim kodom spremaju na Arduino tako da zauzimaju određen memorijski prostor, pa je najbolje pripaziti da se ne učitavaju ne korištene biblioteke s obzirom da je količina fizičke memorije relativno ograničena.

Teksutalna konzola prikazuje serijske podatke koji pristižu sa Arduina kroz USB kabel na računalo. Putem konzole podaci se mogu i slati na Arduino jednostavnim upisivanjem teksta. Pri korištenju konzole ponekad je potrebno i podesiti *baud rate* odnosno brzinu slanja podataka serijskom vezom.



Slika 8 - Arduino IDE

Na slici brojčanim oznakama su označeni redom:

- **(1)** – *Meni traka*
- **(2)** – *Alatna traka s gumbima*
- **(3)** – *Tekstualni editor za pisanje koda*
- **(4)** – *Sekcija sa porukama*
- **(5)** – *Konzola*

5. Senzori

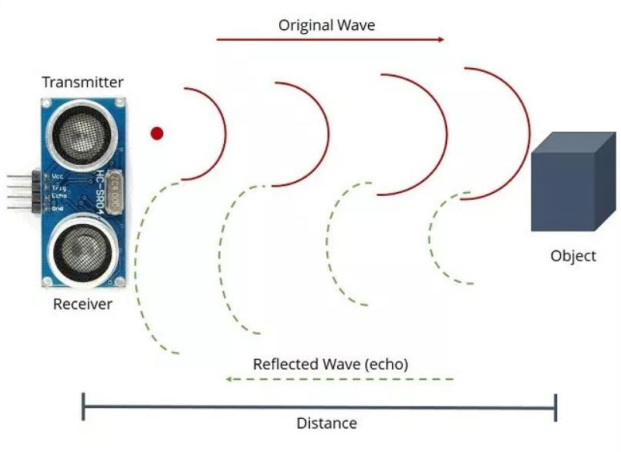
Senzori su uređaji koji očitavaju i reagiraju na podražaje iz okoline. Ti podražaji mogu biti temperatura, blizina, svijetlost, pritisak i slično. Očitanje je većinom analogni signal koji se može na licu mjesta u sklopu integriranog senzorskog modula prebaciti u neki format pogodan za čitanje ljudima i ispisati (npr. digitalni termometar), ili poslati na neki drugi uređaj (npr. mikrokontroler) gdje se podaci prebacuju u digitalni format i dalje obrađuju po potrebi.

5.1. Ultrazvučni senzor HC-SR04

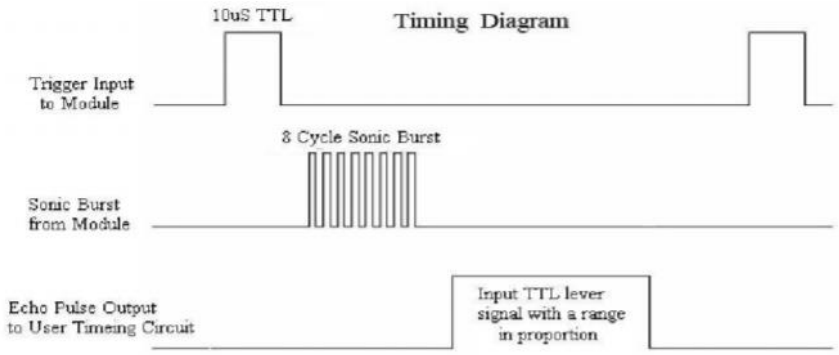
U svom projektu koristila sam ultrazvučne senzore HC-SR04 (Slika 9) koji mjere udaljenost koristeći ultrazvučne valove. Ultrazvučni valovi su zvučni valovi koji se nalaze iznad frekvencije koju ljudi mogu čuti, većinom iznad 20k Hz. Ultrazvučni sezor emitira i prima ultrazvučne valove koji se reflektiraju natrag od prepreke. Udaljenost se mjeri kao vrijeme koje je proteklo između emitiranja vala iz odašiljača i njegovog očitavanja na prijamniku nakon odbijanja (Slika 10). Odašiljač je spojen sa ulaznim pinom pod nazivom *Trig*, a prijamnik na pin *Echo*. Pinove možemo lako kontrolirati i očitavati spajanjem na Arduino digitalni izlaz i ulaz.

Slika 9 - ultrazvučni senzor HC-SR04

Raspon mjerenja HC-SR04 senzora je između 2 i 400 cm. Radni napon mu je 5V, frekvencija odašiljanja 40 kHz, a struja 15mA. Modul na sebi uključuje odašiljač, prijamnik i kontrolni integrirani krug sa četiri pina. Pinovi koji se nalaze na modulu su kao što je već spomenuto *echo* i *trig*, i uz njih i pinovi *5V* te *GND*. Osnovni princip rada modula spojenog na mikrokontroler podrazumjeva da se izlazni signal (digitalno „1“) šalje na *trig* pin barem 10 mikrosekundi, nakon čega modul automatski odašilje osam 40 kHz ultrazvučnih valova i detektira da li postoji povratni signal (Slika 11) te šalje rezultat preko *echo* pina na spojeni mikrokontroler. Udaljenost se zatim računa pomoću formule: D = (T*C)/2; gdje je D udaljenost, T vrijeme između odašiljanja i primanja signala, a C brzina zvuka (~340 m/s).

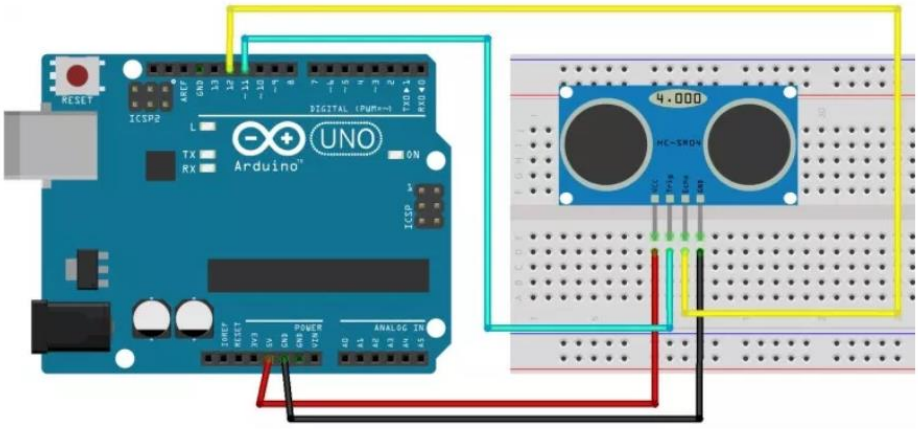


Slika 10 - ilustrirani princip rada ultrazvučnog senzora [7]



Slika 11 - vremenski dijagram funkcioniranja ultrazvučnog modula [8]

5.2 Interakcija HC-SR04 modula i Arduina



Slika 12 - shema spajanja jednog ultrazvučnog modula HC-SR04 na Arduino Uno [9]

Jednostavan primjer spajanja jednog modula na Arduino Uno može se postići sa sljedećom konfiguracijom (Slika 12). Pinovi 5V i GND spajaju se na Arduino pločicu na iste te pinove, dok se echo i trig pinovi sa modula spajaju na digitalne pinove Arduina. U programskom kodu potrebno je inicijalizirati i podesiti izlazni(trig) i ulazni(echo) pin, nakon čega se serijskom vezom šalje izlazni signal logičke jedinice 10 mikrosekundi pomoću funkcije *digitalWrite()* na izlazni pin, a trajanje odašiljanja očitava se funkcijom *pulseIn()* sa ulaznog pina, te se potom po prethodno navedenoj formuli računa udaljenost.

Očitanje sa senzora može se u stvarnom vremenu ispisivati na serijsku konzolu, ili prikazati drugim proizvoljnim metodama. Na ovaj način može se spojiti „proizvoljan“ broj senzora, odnosno onoliko senzora koliko je dostupno pinova na Arduinu.

6. MIDI

6.1. Standardizacija

Još u ranim 80-im godinama prošlog stoljeća nije postojao standardizirani način za sinkronizaciju glazbenih instrumenata koji su bili proizvdeni od strane različitih proizvođača. Svaki proizvođač je imao svoj vlastiti standard za komunikaciju između uređaja. Osnivač tvrtke Roland Corporation iz Japana Ikutaro Kakehashi smatrao je kako manjak standardizacije u komunikaciji uređaja koči napredak glazbene industrije te je predložio razvoj jednostavnog standarda koji bi ujedinio dosadašnji raskol mnoštva različitih protokola. Pokrenuvši lavinu promjene u industriji, zajedničkim snagama Kakehashi i drugi reprezentativci industrije uspostavili su MIDI protokol. Kao baza za uspostavljanje MIDI protokola koristio se jedan od tadašnjih vlasničkih standarda tvrtke Roland – DCB (eng. Digital Control Bus). Službena MIDI specifikacija objavljena je 1983. godine.

6.2. Što je sve MIDI?

Ako ste imali starije mobilne uređaje sa polifonim melodijama, igrali retro video igrice ili slušali popularnu EDM glazbu, vjerojatno ste čuli MIDI u akciji. MIDI je kratica za Music Instrument Digital Interface – što označava tehnički standard, protokol za komunikaciju i električne konektore koji spajaju širok spektar glazbenih uređaja, računala i druih povezanih uređaja za reprodukciju, snimanje i uređivanje glazbe. MIDI je protokol koji je standardiziran u 80im godinama prošlog stoljeća, i služi za komunikaciju i kontrolu između elektroničkih glazbenih instrumenata i digitalnih glazbenih alata. Protokol sam po sebi ne proizvodi nikakve zvukove već služi za prijenos informacija o onome što je potrebno pretvoriti u audio signal i reproducirati.

Osim samog protokola za prijenos informacija definiran je još i format datoteka za pohranu MIDI podataka na računalu - .SMF (Standard MIDI File). To je format datoteka koji pruža standardizirani način za pohranu, transport i djeljenje glazbenih sekvenci. Prednosti MIDI datoteka naspram regularnih audio datoteka je prvenstveno u veličini i lakoći modifikacije. MIDI datoteke su veoma malih formata, i zapisane su na takav način da je vrlo lako zapis modificirati po želji nakon što je on snimljen, bez da ga se mora ponovno snimati. U retro video igricama MIDI datoteke su bile korištene za pozadinsku glazbu igrice kako bi se uštedjelo na memorijskom prostoru koji je tada bio značajan. Tokom igre zvučna kartica računala služila bi kao prijamnik i interpretirala bi podatke iz MIDI datoteke te proizvodila audio signal po tim uputama, koji bi onda bilo moguće čuti na zvučnike.

MIDI protokol prenosi podatke porukama koje se šalju serijskom komunikacijom, a koje uključuju informacije o glazbi koju je potrebno reproducirati. Kroz jedan Kada glazbenik svira na MIDI instrumentu, svaki pritisak tipke, pomak klizača ili prekidača pretvaraju se u MIDI podatke. Zvuk glazbenog performansa na klavijaturi zvučati će specifično za taj instrument, no kako MIDI standard zapisuje samo relevantne informacije o odsviranim notama a ne sam zvuk, ta

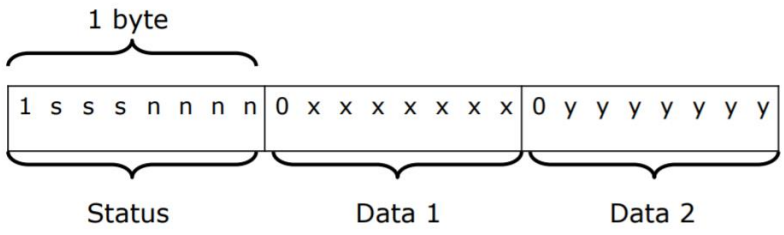
ista glazba se može vrlo lako reproducirati u stilu bilo kojeg drugog instrumenta – npr. gitare, harmonike ili flaute.

S obzirom da ne definira boju ili prirodu sintetiziranog zvuka, već samo opisuje akciju proizvođenja tog zvuka - na neki način MIDI je instrumentu ono što je HTML internetskom pregledniku, jezik za označavanje. Kao komunikacijski i kontrolni jezik za komunikaciju između glazbenih instrumenata i uređaja, već u samoj ideji nastanka ovog protokola, bitna smjernica bila je da bude brz kako bi omogućio kvalitetnu interakciju između uređaja i instrumenata. Iako je koncept ovog protokola relativno lak za razumjeti, odlikuje ga mnogo aspekata sa profesionalne glazbene strane, što ga i čini toliko moćnim i rasprostranjenim.

6.3. MIDI protokol

MIDI je serijski komunikacijski protokol koji funkcionira brzinom slanja podataka (baud rate) od 31,250. MIDI podatkovni okvir je jednosmjerni asinkroni okvir sa 10 bitova po bajtu (start bit, osam podatkovnih bitova, stop bit). MIDI naredba se sastoji od barem dva bajta: statusni i podatkovni bajt. MIDI vrijednosti su izražene u heksadecimalnoj notaciji kako bi se olakšalo referenciranje. Svi statusni bajtovi imaju minimalnu vrijednost 128 ili veću, tj. u binarnom zapisu to bi značilo da je prvi bit svakog statusnog bajta (MSB – Most Significant Bit) uvijek 1. Ta činjenica omogućuje uređaju koji dešifrira MIDI poruku da zna da se radi o statusnom bajtu. Kontrastno tome, svi podatkovni bajtovi imaju maksimalnu vrijednost 127 ili manju, pa je njihov prvi bit uvijek 0. Svaki MIDI bajt ima opseg od 128 različitih diskretnih vrijednosti koje može poprimiti, vrijednosti 0-127 za podatkovne, a 128-255 za statusne bajtove. Zbog tog označavanja vrste bajta, zapravo preostaje samo 7 bitova u svakom od tih bajtova za reprezentaciju MIDI poruke.

MIDI poruke počinju sa statusnim bajtom, u kojem se specificira tip poruke na temelju čega slijedi određen broj podatkovnih bajtova specifičnih za tu vrstu poruke (Slika 13). Prva tri bita statusnog bajta nakon jedinice – oznaka ***sss***, predstavljaju tip poruke (kojih postoji 8), a posljednja četiri – oznaka ***nnnn***, predstavljaju broj kanala na koji se poruka treba dostaviti (kojih je maksimalno 16, a označavaju se brojevima 0-15).



Slika 13 - konstrukcija okvira MIDI poruke u binarnom zapisu [10]

U nekim slučajevima statusni bajt može biti i zanemaren, te se tada očitavaju samo podatkovni bajtovi (osim u prvoj poruci ovog slijeda). Uređaj koji intepretira poruku uzima da je od svake poruke u tom slijedu statusni bajt jednak onom prvom. Ova opcija je korisna kod optimiziranje prijenosa podataka u slučaju da je potrebno poslati dugačku seriju jednakih poruka.

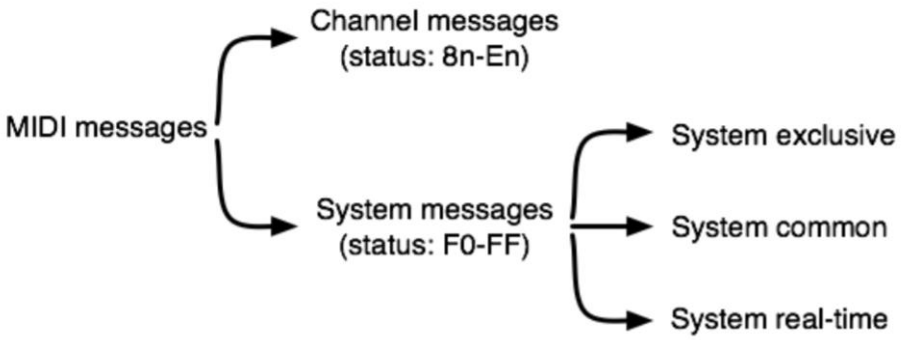
Određene su dvije vrste MIDI poruka (Slika 14):

- Poruke kanala (Channel messages)
- Sistemske poruke (System messages)

Sistemske poruke granaju se na sljedeće tri vrste:

- Poruke eksluzivne za sistem (System exclusive)
- Uobičajene sistemske poruke (System common)
- Poruke za sistem u stvarnom vremenu (System real-time)

Kao što im naziv implicira, poruke kanala odnose se na specifične individuale kanale, a sistemske na kompletan sustav. Postoji i iznimka u kanalnim porukama kada one nisu specificirane točno za jedan kanal. Kada je uključena opcija *„omni“*, prijamnici će primati poruke namjenjene za bilo koji kanal a ne samo za njihov.



Slika 14 - tipovi MIDI poruka [10]

6.3.1. Poruke kanala

Kanali u MIDI komunikaciji označavaju različite spojene uređaje u sekvenci uređaja. Kanal 1 može biti klavijatura, kanal 2 bass, kanal 3 sintisajzer itd. Ovo omogućuje jednoj MIDI konekciji prijenos podataka na više različitih uređaja odjednom. Postoji nekoliko vrsta statusnih poruka koje se odnose na komunikaciju kanalima, od kojih svaki tip poruke ima određen broj podatkovnih bitova. Tipovi poruka specificirani su na slici ispod (Slika 15).

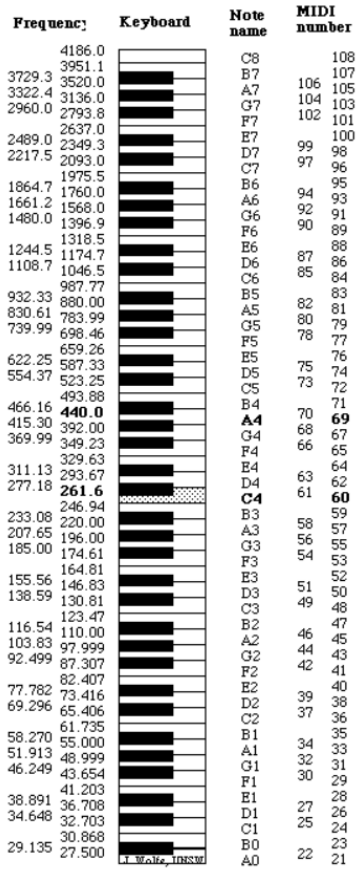
MIDI Status Messages					
Message Type	**MS Nybble**	**LS Nybble**	**Number of Data Bytes**	**Data Byte 1**	**Data Byte 2**
Note Off	0x8	Channel	2	Note Number	Velocity
Note On	0x9	Channel	2	Note Number	Velocity
Polyphonic Pressure	0xA	Channel	2	Note Number	Pressure
Control Change	0xB	Channel	2	Controller Number	Value
Program Change	0xC	Channel	1	Program Number	-none-
Channel Pressure	0xD	Channel	1	Pressure	-none-
Pitch Bend	0xE	Channel	2	Bend LSB (7-bits)	Bend MSB (7-bits)
System	0xF	further specification	variable	variable	variable

Slika 15 - statusne poruke kanala [5]

Jedne od najbitnijih poruka koje se šalju su *Note OFF* i *Note ON* poruke. Ove poruke čine najveći dio informacija koji se šalje MIDI kanalom. Sadrži dva podatkovna bajta od kojih prvi označava broj note a drugi brzinu pritiska tipke. Logično, *Note ON* označava početak pritiska tipke, dok *Note OFF* označava otpuštanje tipke (npr. na klavijaturi). Kada sintisajzer (u sklopu klavijature ili odvojen) primi *Note ON* počinje proizvoditi zvuk, a kada primi podatak *Note OFF* dobio je instrukciju da prestane (Slika 16).

Slika 16 - pritisak i otpuštanje tipke koje izazivaju Note ON i Note OFF poruke

Svaka *Note ON* i *Note OFF* poruka sastoji se dakle od tri bajta u obliku: 0xnc 0xkk 0xvv; gdje je n – naredba on ili off, c – kanal, kk – broj note (Slika 17), vv – brzina pritiska. Brzina pritiska 0 označava i alias za naredbu off. Sve note koje se mogu slati preko MIDI kanala mapirane su brojevnim vrijednostima na standardizirani način.



Slika 17 - mapiranje nota u MIDI brojeve

Polyphonic pressure ili aftertouch, te *Channel pressure/aftertouch* označavaju parametre koji se prikupljaju pri jačem pritisku tipke nakon što je ona inicijalno pritisnuta i sadrže 2, odnosno 1 podatkovni bajt. Polyphonic pressure označava post pritisak svake tipke zasebno, dok Channel pressure mjeri ukupni pritisak na svim tipkama. *Pitch bend* u dva podatkovna bajta prenosi podatke sa upravljačkog kotačića i specificira da li postoji promjena u visini tona.

Control change naredbe opisuju još neke parametre koji mogu biti poslani sa kontrolera osim ovih koji imaju svoje specifične posvećene poruke. *Program change* se koristi za promjenu pohranjenih *patcheva* u memoriji kontrolera.

6.3.2. Sistemske poruke

Sistemske poruke djele se na tri podkategorije: *System real time, System common* i *System exlusive.* Prva podkategorija odnosi se na statusne bajtove kojima je MSB višeg nibble-a 1, i to su specifične sistemske poruke namjenjene uređajima. One kontroliraju izvođenje tempiranih sekvenci u MIDI sustavu, a mogu biti npr. clock – koji nije uobičajeni sistemski sat već označava muzički tempo pri izvođenju glazbe, ili poruka za resetiranje uređaja. *System common* poruke namjenjene su kao i prethodne sistemske poruke, za cijeli sustav odnosno spoj svih uređaja. Njima se npr. može kontrolirati izvođenje unaprijed snimljenih MIDI sekvenci koje su pohranjene na uređajima ili naštimavanje uređaja. Posljednja skupina sistemskih poruka *System exclusive* koriste se za prijenos podataka koji je specifičan za određeni uređaj. Proizvođači uređaja u tim porukama specificirali su detalje za preciznije naštimavanje i kontrolu vlastitih uređaja, i jedinstveni identifikator svakog proizvođača je uključen u takve poruke.

6.4. MIDI uređaji

Glazbeni protokoli su standardizirani kako bi digitalnim i elektroničkim uređajima omogućili laku i funkcionalnu međusobnu komunikaciju i kontrolu. MIDI uređaji mogu se podijeliti u dvije skupine:

- kontroleri - uređaji koji šalju MIDI poruke
- prijamnici - uređaji koji primaju MIDI poruke i nešto s njima čine

MIDI uređaji mogu biti klavijature, sintisajzeri, uređaji za uzorkovanje, sekvenceri, elektronički bubnjevi, pedale za efekte i sl. Neki uređaji, poput sintisajzera mogu biti i kontroleri – tipke klavijature, i prijamnici – dio koji sintetizira primljene podatke (koji mogu doći i od drugih izvora) u audio signal za daljnju reprodukciju. Kontroleri sami po sebi ne proizvode zvuk nego primaju i obrađuju informacije (pretvaraju ih u MIDI format) koje su proizvedene iz njegovih vanjskih djelova kao što tipke, prekidači, gumbi, potenciometri itd., te ih prenose putem MIDI kabela na druge MIDI uređaje (prijamnike) koji ih mogu reproducirati. Setovi tih ulaznih parametara sa kontrolerskog uređaja mogu se i pohraniti u internu memoriju uređaja kao *patchevi* koji se mogu opetovano koristiti. Npr gitarist može podesiti svoj uređaj za efekte koji pritišće nogom na željene postavke i samo jednim klikom određenog prekidača, uz pomoć spremljenog patcha, promijeniti cijelu konfiguraciju glazbe koju ima unaprijed spremljenu (Slika 18).

Slika 18 - Meloaudio MIDI kontroler za efekte gitare koji može pohraniti 8 različitih setova parametara [11]

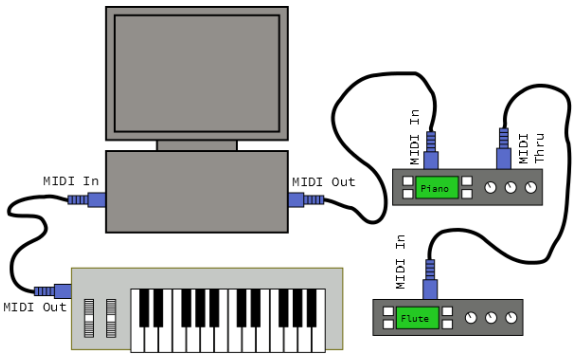
Najrasprostranjenije korišten MIDI kontroler je elektronička klavijatura (Slika 19). Kada glazbenik koristi električnu klavijaturu, svaka tipka koju stisne MIDI kontroleru daje signal da pošalje MIDI podatke o visini tona te jačini i trajanju pritiska tipke. Osim klavijatura, često se koriste i elektronička puhačka glazbala i elektronički bubnjevi. MIDI 'sposobnosti' mogu se ugraditi i u tradicionalne glazbene instrumente poput klavira kako bi oni postali kontroleri. Iako je najveća upotreba MIDI kontrolera u glazbenom svijetu, oni se također mogu koristiti i za kontrolu drugih MIDI-kompatibilnih uređaja kao što su svijetla pozornice i drugi uređaji za

proizvodnju zabavnih efekata, putem MSC (MIDI Show Control) ekstenzije, gdje se MIDI poruke interpretiraju kao specifične akcije koje uređaj treba izvesti umjesto kao note.

Slika 19 - Akai Professional MPK249 MIDI kontroler [12]

MIDI instrumenti (prijamnici) su uređaji koji primaju MIDI poruke i reproduciraju zvuk sukladno uputama koje su primili. Sadrže IN i OUT MIDI portove (kako bi mogli biti umreženi u sustav ukoliko ima više povezanih uređaja), procesorsku jedinicu za obradu podataka, sučelje koje omogućava korisničko programiranje i elektroničke komponente koje su potrebne za generiranje zvuka.

Prema MIDI standardu, pomoću jednog lanca MIDI kabela moguće je spojiti do 16 različitih uređaja zajedno u jednu komunikacijsko-kontrolnu cjelinu (16 kanala). MIDI signal sadrži nekoliko vrsta informacija među kojima se nalazi i kanal prijamnika. Korištenjem tih kanala dobiven je jednostavan način za prepoznavanje različitih uređaja koji su spojeni zajedno (Slika 20). Svaki uređaj se podešava za slušanje specifičnog kanala (ili nekoliko njih odjednom) i reagiraju samo na poruke kodirane za taj kanal. Na taj način moguće je kontolirati različite uređaje samo uz specifikaciju njihovog kanala koji treba primiti poruku koja je poslana iz kontrolera. Većina elektroničkih klavijatura nudi i opciju podjele tipki u sekcije proizvoljnih veličina koje mogu funkcionirati na različitim MIDI kanalima tako da se na jednoj klavijaturi istovremeno mogu proizvoditi zvukovi različitih priroda. Detaljnije o spajanju uređaja u nastavku.

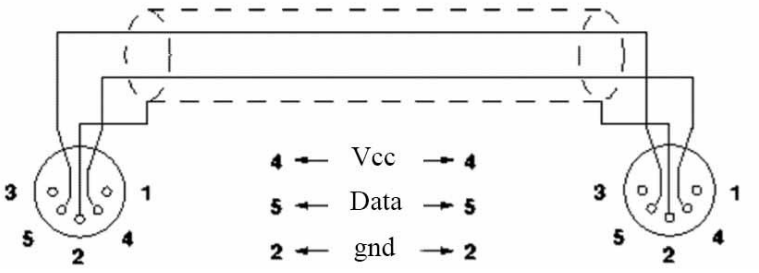


Slika 20 - primjer ulančavanja MIDI uređaja u "daisy chain" konfiguraciju [5]

6.5. MIDI konektori i međusklopovi

S obzirom da je MIDI jednosmjerni serijski protokol, kroz jedan kabel istovremeno podatci mogu 'teći' samo u jednom smjeru. Svaki uređaj većinom posjeduje jedan MIDI ulazni (MIDI IN) i jedan MIDI izlazni (MIDI OUT) konektor, i ovisno o vrsti uređaja – i dodatni MIDI konektora koji se naziva MIDI THRU. MIDI OUT odašilje MIDI podatke koji su kreirani unutar tog uređaja (kontrolera), MIDI IN prima MIDI podatke iz kontrolera, a MIDI THRU port odašilje kopiju podataka koje ulaze u uređaj kako bi se oni mogli poslati na sljedeći uređaj u ulančanom nizu više uređaja. MIDI THRU ne prenosi nikakve podatke generirane unutar tog uređaja na kojem se nalazi već služi samo kao most za spajanje.

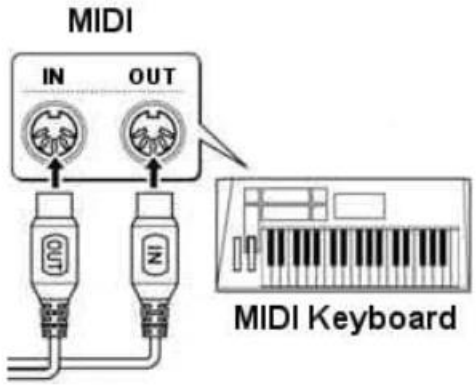
MIDI konektori na uređajima su pet polni ženski DIN (5-PIN DIN) konektori, dok na MIDI kabelu imamo s obje strane pet polni muški DIN konektor (Slika 21). MIDI međusklopovi uobičajeno dolaze u konfiguracijama sa 2-, 4- ili 8- portova od kojih svaki može odašiljati 16 kanala MIDI podataka. Dakle MIDI sučelje od 8 portova može slati na sveukupno 128 MIDI kanala.



Slika 21 - MIDI 5 PIN DIN pinout shema

Dva pina svakog MIDI konektora su nekorištena (oznake na slici – 1 i 3), dok se pinovi s oznakama 2, 4 i 5 koriste za masu, napon i slanje podataka. Zbog jednosmjerne komunikacije, svaki MIDI konektor može ili primati ili odašiljati podatke. Svaki se uređaj spaja na sljedeći na način da se MIDI OUT port kontrolera spoji kabelom na MIDI IN port instrumenta. Npr. ako

spajamo računalo sa sintisajzerskom klavijaturom i želimo dvosmjernu komunikaciju kako bi mogli kreirati MIDI putem stisknja tipki klavijature, ali i generirati zvukove na sintisajzeru koje smo napravili na računalu, MIDI OUT iz računala spojit ćemo na MIDI IN sintisajzera, a MIDI OUT sintisajzera na MIDI IN računala. Na taj način dobili smo željenu konfiguraciju (Slika 22).



Slika 22 - shema spajanja MIDI konektora na jedan uređaj [13]

Kada spajamo više uređaja u ulančani niz, na kontroleru se koristi MIDI OUT port kako bi se u lanac uređaja poslali MIDI podatci, na prvom sljedećem uređaju kabel se spaja u MIDI IN port, a iz njega putem MIDI THRU porta u sljedeći MIDI IN, i tako dalje do kraja lanca. Ovaj način spajanja nije idealan zbog latencije koja se javlja pri fizičkom slanju podataka kroz kablove zbog čega se ne preporuča spajanje više od 3-4 uređaja ovim putem, ali i zbog činjenice da ukoliko sa kontrolera šaljemo podatke na posljednji uređaj u lancu, svi uređaji prije njega moraju biti uključeni.

Koristeći MIDI međusklopove, osim za spajanje više od 16 uređaja ulančavanjem (što nije preporučeno) dobivamo lakši i jednostavniji način za prijenos podataka bez latencije i između manje količine uređaja. Međusklop prima podatke iz kontrolera i odašilje ih na spojene uređaje (Slika 23).

Slika 23 - primjer MIDI međusklopovolja [14]

6.6. MIDI + računalo

MIDI signali, osim na specijaliziranim glazbenim uređajima, danas mogu biti sekvencirani i pomoću računalnih softverskih aplikacija koji su zapravo reprezentacija tih fizičkih uređaja i njihovih osobina u grafičkom obliku. Takve softverske aplikacije zovu se DAW što označava skraćenicu za engleski izraz Digital Audio Workstation. DAW omogućuju snimanje, uređivanje, modificiranje i produkciju audio datoteka. Rad sa MIDI datotekama je također integralni dio DAW-a (Slika 24). Zato što MIDI datoteka nije audio datoteka već set komandi za izvedbu glazbe, ona se može uređivati i modificirati puno lakše i jednostavnije od regularnih audio datoteka.

Slika 24 - primjer MIDI datoteke u DAW softveru Reaper [15]

Koristeći DAW ili klasične sekvencere uz MIDI, moguće je promijeniti notu, boju tona, tempo i razne druge parametre MIDI zapisa, ili čak i potpuno presložiti slijed individualnih sekcija, te čuti kako novi aranžman zvuči u stvarnom vremenu, što omogućuju fleksibilnost i lakše eksperimentiranje. Osim modifikacije u DAW programu moguće je i kreirati MIDI zapise koji se mogu kasnije reproducirati na glazbenim uređajima. DAW prestavlja jednostavan način za manipulaciju sa elektronskom glazbom i uz korištenje MIDI formata omogućuje relativno lako korištenje čak i onima sa skromnijim znanjem o glazbenoj teoriji.

Usprkos svojim nedostatcima, moderna tehnologija sve više ide u korak sa analognim uređajima pa se i kvaliteta zvuka značajno poboljšava sa povećanjem procesorske moći današnjih računala. Osim toga, softverski sekvenceri su mnogo jeftiniji i portabilniji nego fizička glazbena oprema, i kompatibilniji sa softverskim aplikacijama za uzorkovanje pa su stoga i veoma korišteni.

7. Korištene komponente

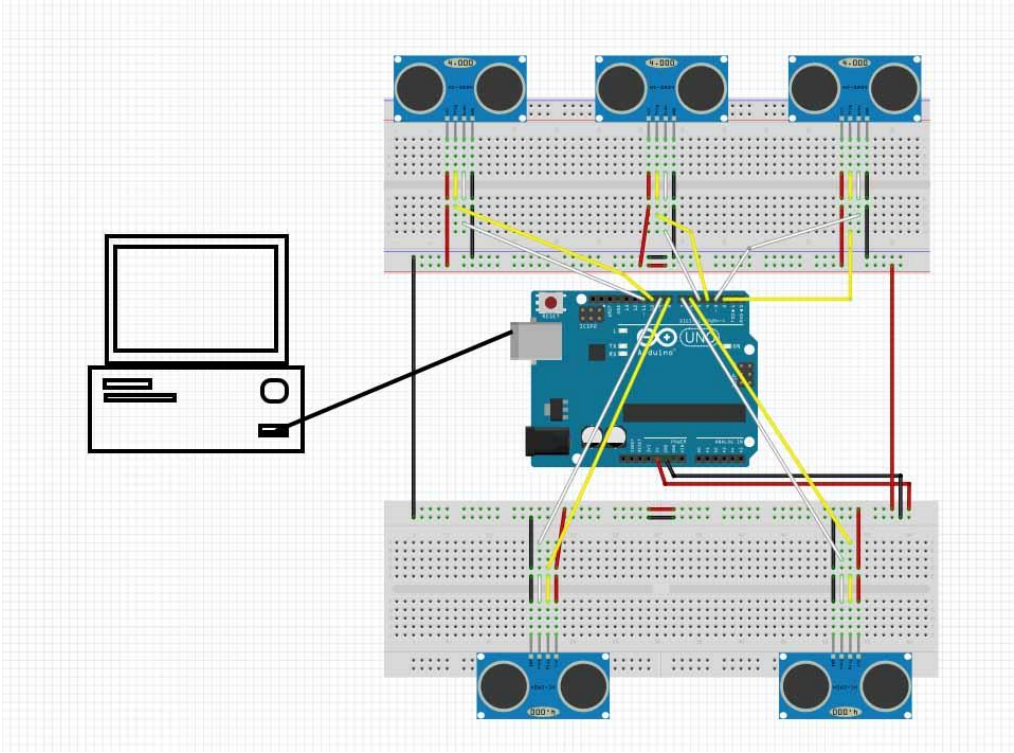
U projektu su korištene sljedeće hardverske i softverske komponente:

- Arduino Uno sa ATMega328P mikrokontrolerom
- HC-SR04 ultrazvučni senzori
- Hairless MIDI-serial bridge
- LoopMIDI driver
- DAW Cakewalk by BandLab

Arduino Uno razvojna pločica i HC-SR04 ultrazvučni senzori opisani su u prethodnim poglavljima tako da ću ovdje prikazati samo shemu spajanja djelova i shemu cjelokupnog projekta, te ću ukratko predstaviti korištene softverske komponente.

7.1.Shema spajanja hardverskih komponenti

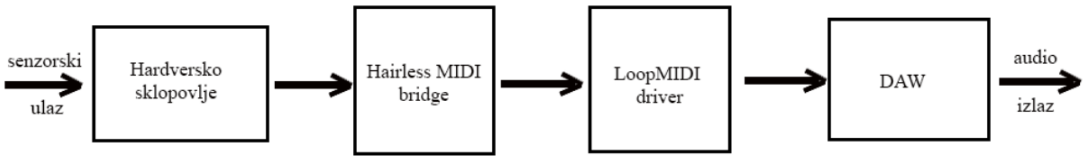
U projektu je korišteno 5 HC-SR04 ultrazvučnih senzora koji su u sljedećoj konfiguraciji spojeni na Arduino (Slika 25) i naposlijetku na računalo putem USB konektora.



Slika 25 - shema spajanja hardverskih komponenti korištenih u projektu

7.2. Shema projekta

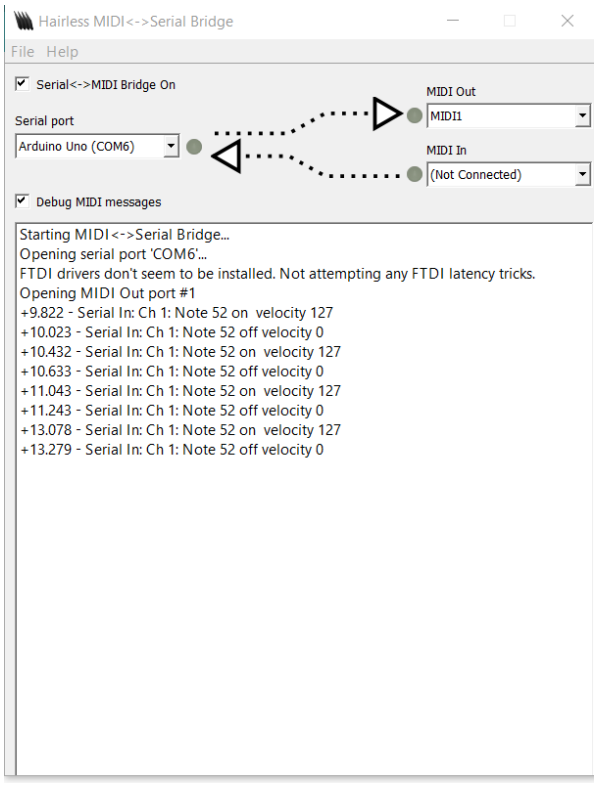
Na slici (Slika 26) je prikazan blok dijagram interakcije hardverskih i softverskih djelova projekta. Ulazni podaci se očitavaju na ultrazvučnim senzorima i obrađuju na hardverskom sklopovolju sastavljenom od Arduina i senzora. Putem serijskog protokola šalju se na USB port računala i pomoću HairlessMIDI aplikacije pripremaju za MIDI konekciju prema završnom odredištu. LoopMIDI aplikacija nalazi se u ulozi MIDI kabela između dva porta, polaznog i odredišnog, te se podaci putem nje dostvljaju u DAW radionicu gdje se naposlijetku sintetiziraju i prenose na zvučnike računala.



Slika 26 - shema interakcije hardverskih i softverskih komponenti

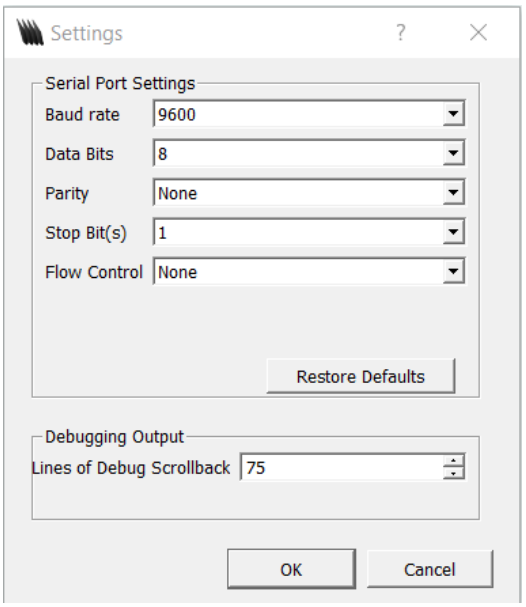
7.3. HairlessMIDI

Hairless je MIDI-serial bridge, jednostavna softverska aplikacija koja služi kao virtualna poveznica između Arduina i virtualnog MIDI porta (Slika 27). Aplikacija je besplatna za preuzimanje i skinuta je sa Githuba (https://projectgus.github.io/hairless-midiserial/). Odabrala sam ju jer je kompatibilna sa Arduino MIDI bibliotekom koju koristim za interpretaciju podataka koje očitavam sa senzora, i za ovakve projekte najviše preporučena za korištenje uz LoopMIDI aplikaciju.



Slika 27 - HairlessMIDI <-> serial bridge aplikacija

Postavila sam vrijednost *Serial port* (ulaz) na Arduino USB port računala (COM6), te *MIDI Out* na MIDI1 MIDI port koji sam virtualno dodala pomoću aplikacije LoopMIDI koja služi kao pomoćni driver za međukonekciju Hairless aplikacije i odabranog DAW-a. Osim odabira ulaznog i izlaznog porta ostale postavke su automatski podešene, a s obzirom da Arduino koristimo kao MIDI kontroler, port *MIDI In* ne ostavljamo isključenim. U dodatnim postavkama (File->Preferences) postavljen je baud rate da bude u skladu s onim koji smo postavili u programskom kodu, a to je 9600 bpsa (Slika 28). Ostale dodatne postavke ostavljene su na predefiniranim vrijednostima i nije bilo potrebe mjenjati ih.



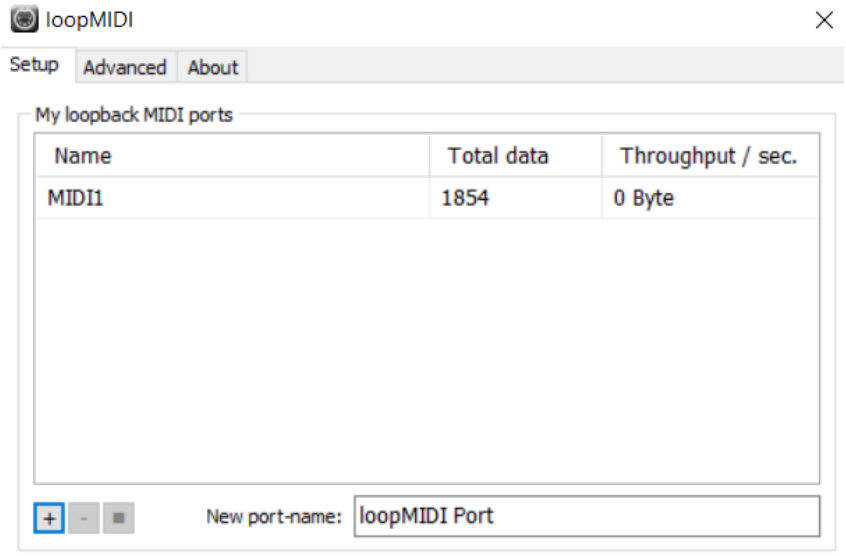
Slika 28 - postavke aplikacije

Označavanjem kvačice na Serial<->MIDI Bridge On opciju pokreće se veza između Arduino serijskog porta i aplikacije, što se indicira sistemskim porukama u tekstualnom ispisu u prozoru aplikacije. Pri učitavanju programskog koda na Arduino potrebno je isključiti bridge jer inače dolazi do konflikta. Nakon učitavanja koda možemo bridge ponovno spojiti. U ispisnim porukama navedena je uspješnost povezivanja, te zatim i ispis svih poruka koje se šalju preko kanala ili eventualnih poruka pogreške (pomoću Debug MIDI messages opcije) ukoliko nešto ne funkcionira kako treba.

Osim virutalnog MIDI izlaznog porta, defaultno se nudi i Microsoft GS Wavetable Synth – MIDI sintisajzer koji dolazi automatski sa Windows operativnim sustavom, ali on nije korišten u ovom djelu iz razloga što nije automatski kompatibilan sa Arduinom bez dodatnih FTDI drivera. Wavetable Synth korišten je kao izlazni MIDI u odabranom DAW-u za sintetiziranje MIDI zvuka na zvučnike računala.

7.4. LoopMIDI

LoopMIDI je softverska aplikacija za dinamičko kreiranje i uništavanje virtualnih MIDI portova (Slika 29). Aplikacija je slobodna za preuzimanje i korištenje i preuzeta je sa službene stranice vlasnika (https://www.tobias-erichsen.de/software/loopmidi.html). Aplikacija se u ovom projektu koristi kao međupoveznica između HairlessMIDI aplikacije i DAW-a kao virtualni port kojim se šalju podaci očitani sa ultrazvučnih senzora spojenih na Arduino. Odabrala sam LoopMIDI jer je jednostavan za korištenje i brz za instalaciju bez previše okolišanja, te je generalno preporučen za korištenje kao virtualni driver u ovakvim projektima.



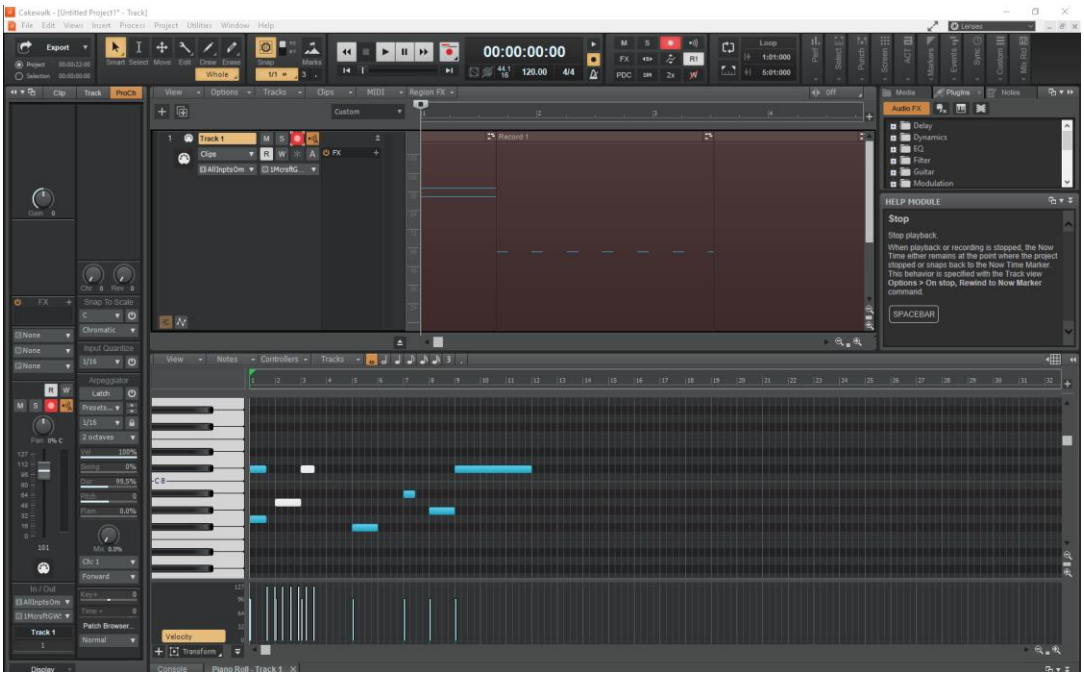
Slika 29 - LoopMIDI aplikacija

Na slici je prikazano sučelje aplikacije LoopMIDI sa otvorenim virutalnim portom imena MIDI1. Dodavanje i brisanje portova je vrlo jednostavno pomoću gumba sa oznakama „+“ i „-„ u donjem lijevom kutu. Navedeno ime porta koji je tu otvoren kofiguriran je u HairlessMIDI aplikaciji kao izlazni MIDI port, te u odabranom DAW-u kao ulazni MIDI port.

7.5. Cakewalk by BandLab DAW

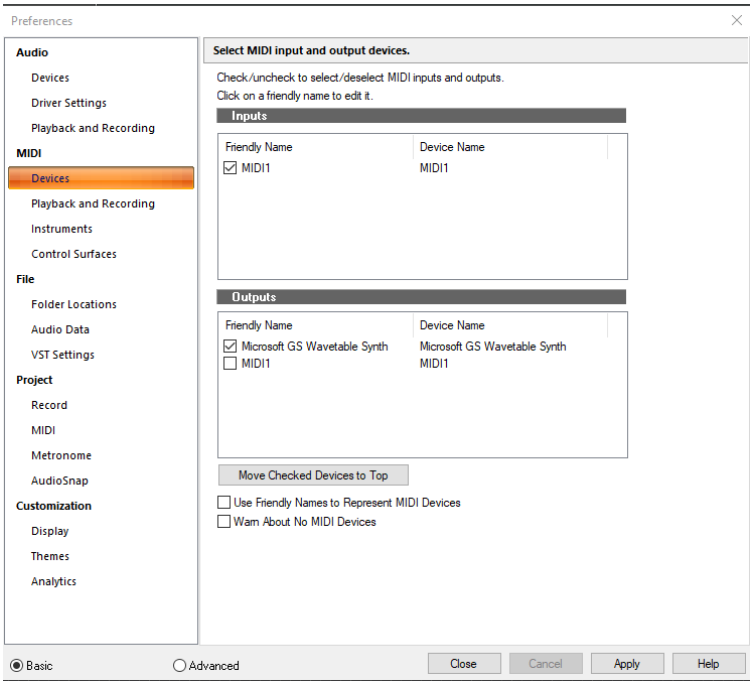
Cakewalk by BandLab digitalna audio radionica proizvod je tvrtke BandLab Technologies i slobodna je za preuzimanje i korištenje (Slika 30). Preuzeta je sa službene stranice proizvoda (https://www.bandlab.com/products/cakewalk). Cakewalk by BandLab je prije postojao kao vlasnički softver pod nazivom SONAR u vlasništvu tvrtke Cakewalk, i njegovo korištenje se plaćalo, no nakon prestanka proizvodnje preuzela ga je tvrtka BandLab Technologies i prenamjenila kao potpuno slobodan softver. Ovaj DAW odabran je zbog toga što je slobodan za korištenje, i preporučen kao kvalitetan i jednostavan virtualni sekvencer.

Ovaj DAW sadrži mnoge funkcionalnosti za stvaranje, snimanje i reprodukciju glazbe, a ja ću ovdje ukratko opisati one koje sam koristila u svom projektu.



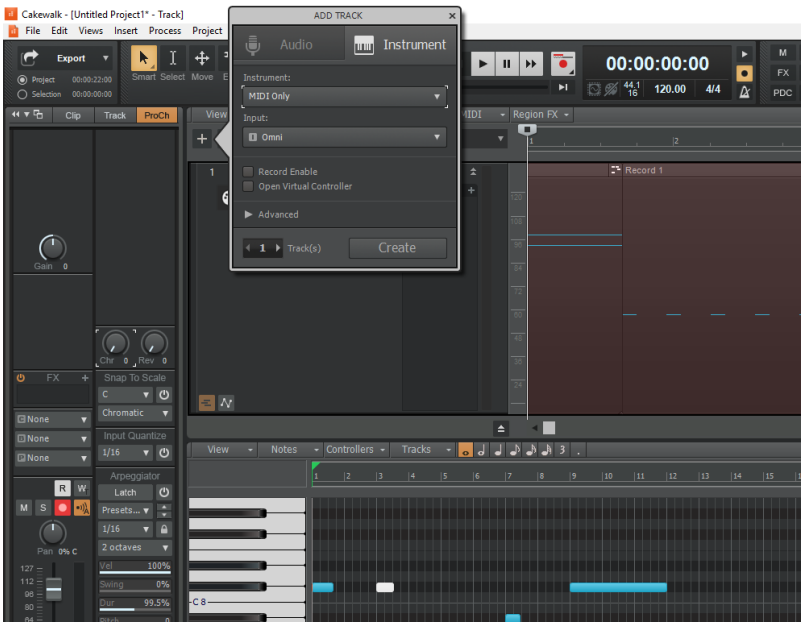
Slika 30 - Cakewalk by BandLab grafičko korisničko sučelje

Prvo nakon otvaranja novog praznog projekta su u postavkama (Edit->Preferences) postavljeni MIDI ulazni i izlazni uređaji. MIDI ulaz postavljen je na MIDI1 port koji smo virtualno otvorili pomoću aplikacije LoopMIDI i s njim prima ulazne podatke koji potječu sa ultrazvučnih senzora spojenih na Arduino. Kao MIDI izlaz postavljen je Microsoftov GS Wavetable Synth ugrađeni softver koji sintetizira ulazne MIDI podatke u zvuk na zvučnike računala. Druge postavke u ovom dijelu nisu mjenjane. Prikaz sučelja postavki prikazan je na slici u nastavku (Slika 31).



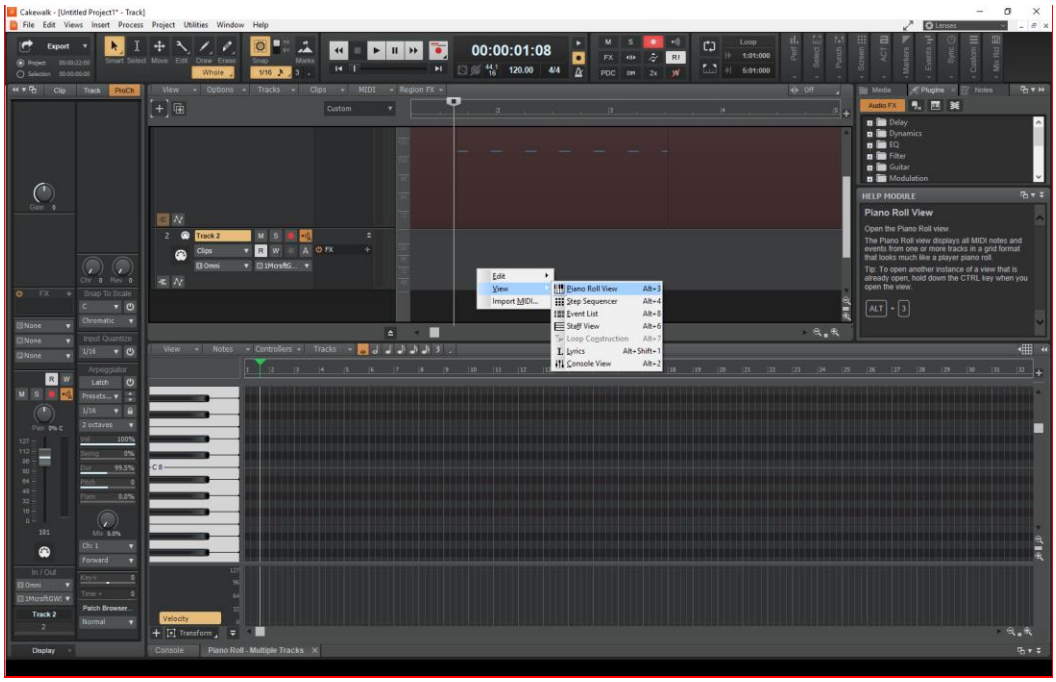
Slika 31 - postavke MIDI uređaja za komunikaciju sa softverom

Klikom na gumb sa oznakom „+“ otvara se mali izbornik sa mogućnošću dodavanja novog instrumenta u kompoziciju (Slika 32). Odabrani instrument za dodavanje je MIDI only, a ulazni kanal je stavljen na *„Omni“* kako bi DAW mogao primati MIDI ulaz sa bilo kojeg kanala. S obzirom da imamo samo jedan prikopčani uređaj preporučeno je postaviti ovu opciju ovako, no ukoliko bi ih imali spojeno više od jednoga bilo bi potrebno i specificirati točno na kojem kanalu će se nalaziti. Druge postavke u ovom prozoru nisu mjenjane i pritiskom na tipku *„Create“* dodaje se odabrani instrument. U kompoziciju se može dodavati još uređaja putem gumba sa oznakom „+“ na opisani način.



Slika 32 - dodavanje novog instrumenta - našeg MIDI ulaza

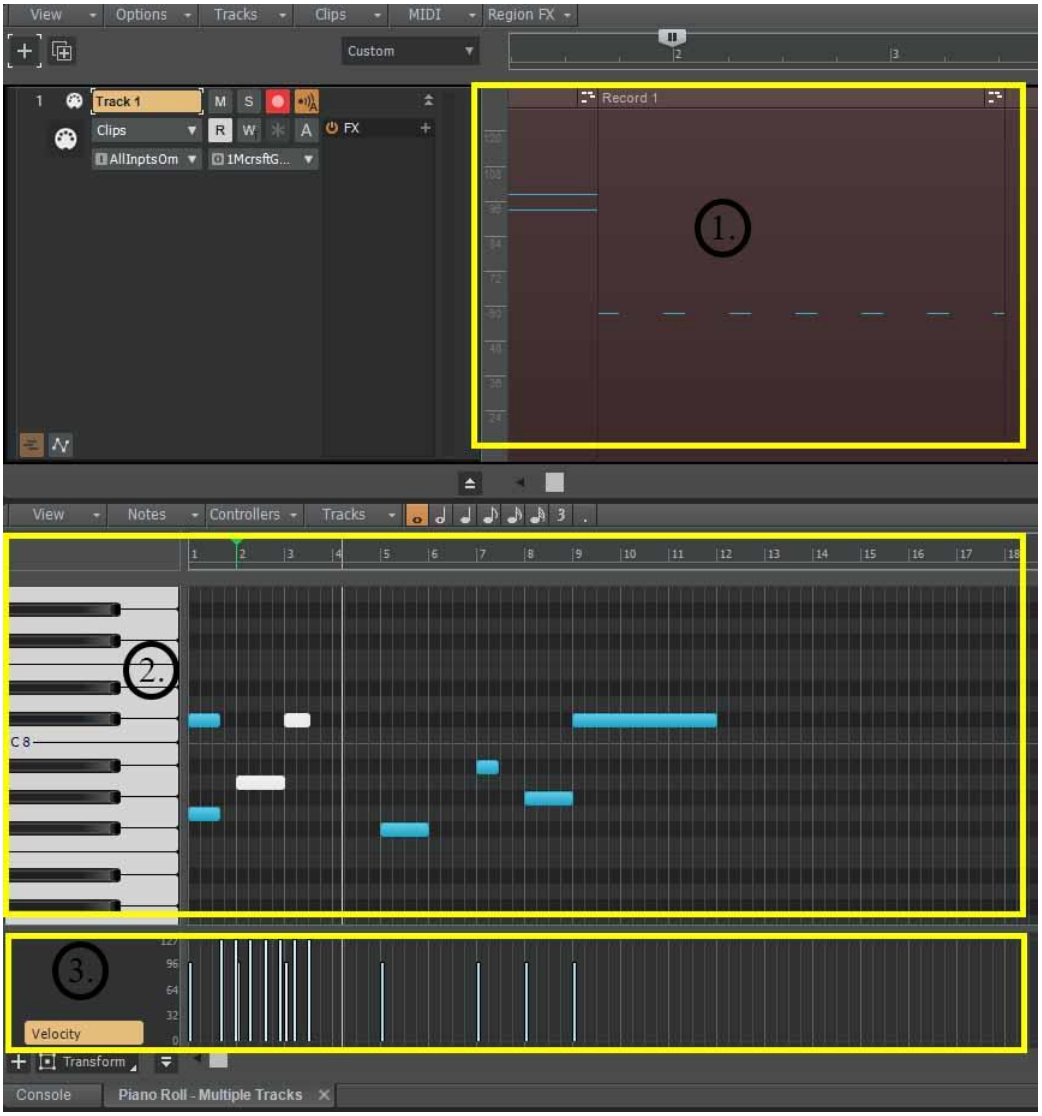
Desnim klikom na prazno područje odabrane trake instrumenta sa opcijom View->Piano Roll View na dnu sučelja otvara se novi dio prozora sa klavijaturom kako je prikazano na slici u nastavku (Slika 33) a kojeg inače inicijalni pri otvaranju projekta nema vidljivog. U Piano Roll pogledu note koje su odsvirane mogu se vizualno predočiti pomoću klavijature što pojednostavljuje uređivanje i reprodukciju. Ako označimo naš odabrani uređaj i stisnemo tipku za snimanje, sav naš MIDI ulaz koji dolazi sa Arduina snima se i predočen je u sučelju.



Slika 33 - omogućavanje Piano Roll pogleda

Na gornjem djelu (Slika 34, oznaka 1) predočeno je ono što je putem mikrokonrtolera i ultrazvučnog senzora snimljeno za specifičnu traku odnosno od specifičnog odabranog uređaja. Trenutno imamo samo jedan MIDI kontroler pa je zapis na jednoj traci, ali kada bi ih bilo više svaka traka sadržavala bi snimljeni zapis sa svojeg pridruženog kontrolera. Ispod toga u Piano Roll pogledu (Slika 34, oznaka 2) vidljiv je snimljen zapis (bijele boje) zajedno sa nekim dodatnim generiranim notama (plave boje) za usporedbu. Ispod klavijature s oznakom 3 (Slika 34) vidljivo je i predočenje brzine (eng. *velocity)*, kod zapisa koji je snimljen putem našeg mikrokontrolera ta vrijednost je ili 0 ili 127, jer je tako predodređeno u programskom kodu i MIDI protokolu, gdje 0 označava ranije opisanu *NoteOff* poruku, a 127 *NoteOn* poruku.

U Piano Roll pogledu snimljene note mogu se lakom manipulacijom sa DAW alatima mjenjati, povisivati, kombinirati sa drugima, reogranizirati ili brisati što je jako korisno kod stvaranja glazbe jer to znači da ako smo nešto pogriješili u samom izvođenju glazbe uvijek to možemo popraviti nakadno u našem uređivaču. Pomoću poznatih gumba na vrhu glazba se može i lako reproducirati.



Slika 34 - vizualni prikaz snimljenih podataka

8. Moguća poboljšanja projekta

Koristeći više od jednog ultrazvučnog senzora, u ovom slučaju njih 5, nisam naišla na komplikacije što se tiče spajanja hardvera. No ukoliko bi ideja bila razviti MIDI kontroler koji bi koristio više od 6 senzora, ili pak i neke druge komponente/senzore/module, zbog fizičkog nedostatka pinova na Arduinu bilo bi potrebno implementirati i neku međukomunikaciju. Jedna od ideja ukoliko bi se koristili npr. samo ultrazvučni senzori bila bi izraditi *breakout board* gdje bi se u suštini nalazio još jedan unaprijed programirani mikrokontroler koji bi primao signale paralelno od više različitih senzora te ih obrađivao i slao serijskom komunikacijom na Arduino.

Inspirirana ovim projektom, možda u budućnosti ova ideja bude proširena na malo kompliciraniju izvedbu uz ne samo ultrazvučne senzore koji šalju samo note, već i neke druge komponente poput potenciometara, LED-ica, tipkala i slično, kako bi se moglo ukomponirati više funkcionalnosti koje MIDI nudi u razvoju i produkciji glazbe.

9. Zaključak

Ovaj projekt bio je odlična vježba elektroničkih i softverskih vještina, te prilika za za primjeniti kreativnost i slobodu stvaranja kroz primjenu odabranog medija. U ovom slučaju medij je bio Arduino platforma i elektronička oprema koja primarno uključuje mikrokontroler i ultrazvučne senzore. Predstavljena je izrada fizičkog hardverskog sklopa jednostavnog MIDI kontrolera i njegove kombinacije sa softverskim alatima, te je predočen primjer programskog koda koji koristi MIDI biblioteku za slanje različitih nota ovisno o očitanjima ultrazvučnih senzora koji su spojeni na Arduino.

10. Literatura i izvori

Online izvori:

1. https://en.wikipedia.org/wiki/Wiki
2. http://www.homerecordingconnection.com
3. https://www.sweetwater.com
4. https://electronicmusic.fandom.com
5. https://www.fecegypt.com/uploads/dataSheet/1522237550_arduino%20uno%20r3.pdf
6. https://www.farnell.com/datasheets/1682209.pdf
7. https://en.scratch-wiki.info/wiki/MIDI_Notes
8. https://www.instructables.com/id/What-is-MIDI/
9. https://www.instructables.com/id/Send-and-Receive-MIDI-with-Arduino/
10. https://www.youtube.com/watch?v=c8Pm98v9NC8
11. http://fortyseveneffects.github.io/arduino_midi_library/index.html

Literatura:

1. Michael Margolis, Arduino Cookbook, 2nd edition, O'Reily Media, 2011
2. John Crisp, Introduction to microprocessors and microcontrollers, 2nd edition, Elsevier, 2003

Izvori slika:

1. https://www.edgefx.in/embedded-systems-basics-with-applications/
2. https://www.sparkfun.com/datasheets/Components/SMD/ATMega328.pdf
3. https://learn.sparkfun.com/tutorials/serial-communication
4. https://www.deviceplus.com/how-tos/arduino-guide/arduino-communication-protocols-tutorial/
5. https://learn.sparkfun.com/
6. https://www.arduino.cc/
7. https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/
8. https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf
9. https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/
10. https://www.nyu.edu/classes/bello/FMT_files/9_MIDI_code.pdf
11. https://meloaudio.com/collections/tone-shifter/products/meloaudio-midi-commander-guitar-floor-multi-effects-portable-usb-midi-foot-controller-foot-switches
12. https://www.akaipro.com/mpk249
13. https://www.iconnectivity.com
14. https://www.thomann.de/gb/iconnectivity_mio10.htm?sid=1be48269df89e1e29139224fadf619f3
15. https://www.norwegiancreations.com/

11. Dodatak

11.1. Programski kod s komentarima

```
//uključivanje MIDI biblioteke
#include <MIDI.h>

//inicijalizacija svih echo i trig pinova
int trigPin1 = 2;
int echoPin1 = 3;
int trigPin2 = 4;
int echoPin2 = 5;
int trigPin3 = 6;
int echoPin3 = 7;
int trigPin4 = 8;
int echoPin4 = 9;
int trigPin5 = 10;
int echoPin5 = 11;

//varijable koje se koriste za slanje različitih nota s obzirom na udaljenost od određenog senzora
int key_control_sensor1;
int key_control_sensor2;
int key_control_sensor3;
int key_control_sensor4;
int key_control_sensor5;


//varijable u koje će se spremati očitanje trajanja signala
//i izračunata udaljenost
long duration;
int distance;

//kreiranje instance MIDI
MIDI_CREATE_DEFAULT_INSTANCE();

void setup() {
  //postavljanje echo i trigger pinova kao ulaznih i izlaznih
  //trig - izlazni; slanje signala
  //echo - ulazni; očitavanje signala
  pinMode(trigPin1, OUTPUT);
  pinMode(echoPin1, INPUT);
```

```
  pinMode(trigPin2, OUTPUT);
  pinMode(echoPin2, INPUT);
  pinMode(trigPin3, OUTPUT);
  pinMode(echoPin3, INPUT);
  pinMode(trigPin4, OUTPUT);
  pinMode(echoPin4, INPUT);
  pinMode(trigPin5, OUTPUT);
  pinMode(echoPin5, INPUT);

  //baud rate moze biti proizvoljan jer koristimo softverski synth,
  Serial.begin(9600);
}

void loop() {

  //petlja koja se izvršava konstantno i očitava vrijednosti svih senzora jedan za drugim
  for(int i=1; i<6; i++){

    int pinNum = i;
    int trigPin = i+i;
    int echoPin = trigPin + 1;

    //poziv funkcije sensorRead za svaki senzor redom kako bi ocitali udaljenosti
    sensorRead(echoPin, trigPin, pinNum);

  }

  //50 mikrosekundi nakon ocitavanja provjeri udaljenosti i salji note
  delayMicroseconds(50);

//ocitanja prvog senzora
  if(key_control_sensor1 < 10) {
    MIDI.sendNoteOn(52, 127, 1);    // E nota - 165Hz (pitch 52, velocity 127, channel 1)
    delay(100);
    MIDI.sendNoteOff(52, 0, 1);     // Szaustavi slanje note - velocity = 0
  } else if(key_control_sensor1 >10 && key_control_sensor1 <15){
    MIDI.sendNoteOn(53, 127, 1);   //F nota - 175Hz
```

```
    delay(100);
    MIDI.sendNoteOff(53, 0, 1);
  } else if(key_control_sensor1 > 15 && key_control_sensor1 <25){
    MIDI.sendNoteOn(54, 127, 1);  //F# nota  - 185Hz
    delay(100);
    MIDI.sendNoteOff(54, 0, 1);
  }


 //ocitanja drugog senzora
  if(key_control_sensor2 < 10) {
    MIDI.sendNoteOn(48, 127, 1);    // C nota - 131Hz (pitch 48, velocity 127, channel 1)
    delay(50);
    MIDI.sendNoteOff(48, 0, 1);      //zaustavljanje slanja note
  } else if(key_control_sensor2 >10 && key_control_sensor2 <15){
    MIDI.sendNoteOn(49, 127, 1);   //C# nota - 139Hz
    delay(50);
    MIDI.sendNoteOff(49, 0, 1);
  } else if(key_control_sensor2 > 15 && key_control_sensor2 <25){
    MIDI.sendNoteOn(50, 127, 1);    //D nota - 147Hz
    delay(50);
    MIDI.sendNoteOff(50, 0, 1);
  }

//ocitanja treceg senzora
  if(key_control_sensor3 < 10) {
    MIDI.sendNoteOn(55, 127, 1);    // G nota - 196Hz (pitch 55, velocity 127, channel 1)
    delay(100);
    MIDI.sendNoteOff(55, 0, 1);     // zaustavljanje slanja note
  } else if(key_control_sensor3 >10 && key_control_sensor3 <15){
    MIDI.sendNoteOn(56, 127, 1);  ///G# nota-  208Hz
    delay(100);
    MIDI.sendNoteOff(56, 0, 1);
  } else if(key_control_sensor3 > 15 && key_control_sensor3 <25){
    MIDI.sendNoteOn(57, 127, 1);    //A nota - 220Hz
    delay(100);
```

```
    MIDI.sendNoteOff(57, 0, 1);
  }


//ocitanja cetvrtog senzora
  if(key_control_sensor4 < 10) {
    MIDI.sendNoteOn(62, 127, 1);    // D nota - 294Hz (pitch 62, velocity 127, channel 1)
    delay(100);
    MIDI.sendNoteOff(62, 0, 1);     // zaustavljanje slanja note
  } else if(key_control_sensor4 >10 && key_control_sensor4 <15){
    MIDI.sendNoteOn(63, 127, 1);   //D# nota - 311Hz
    delay(100);
    MIDI.sendNoteOff(63, 0, 1);
  } else if(key_control_sensor4 > 15 && key_control_sensor4 <25){
    MIDI.sendNoteOn(64, 127, 1);    //E nota - 330Hz
    delay(100);
    MIDI.sendNoteOff(64, 0, 1);
  }

//ocitanja petog senzora
  if(key_control_sensor5 < 10) {
    MIDI.sendNoteOn(65, 127, 1);    // F nota (pitch 65, velocity 127, channel 1)
    delay(100);
    MIDI.sendNoteOff(65, 0, 1);     // zaustavljanje slanja note
  } else if(key_control_sensor5 >10 && key_control_sensor5 <15){
    MIDI.sendNoteOn(66, 127, 1); //F# nota - 370Hz
    delay(100);
    MIDI.sendNoteOff(66, 0, 1);
  } else if(key_control_sensor5 > 15 && key_control_sensor5 <25){
    MIDI.sendNoteOn(67, 127, 1);    //G nota - 392Hz
    delay(100);
    MIDI.sendNoteOff(67, 0, 1);
  }

}
```

```
//funkcija za ocitavanje sa 5 senzora jedan za drugim
void sensorRead(int echoPin, int trigPin, int pinNum){

  //reset trig pina
   digitalWrite(trigPin, LOW);
   delayMicroseconds(2);

  //10 mikrosekundi odasilje se ultrazvucni signal
   digitalWrite(trigPin, HIGH);
   delayMicroseconds(10);

   //nakon isteka 10 mikrosekundi ponovno se trig pin postavlja na 0 kako bi prestao odasiljati
   digitalWrite(trigPin, LOW);

   //funkcija pulseIn ocitava jeku signala na echo pinu
   duration = pulseIn(echoPin, HIGH);

   //izracun udaljenosti po forumuli
   distance = duration*0.034/2;


  //ovisno o promjeni udaljenost od odredenog senzora ocitanja se spremaju
  //u 5 zasebnih varijabli, po jedna za svaki senzor
  if(pinNum == 1) key_control_sensor1 = distance;
  else if(pinNum == 2) key_control_sensor2 = distance;
  else if(pinNum == 3) key_control_sensor3 = distance;
  else if(pinNum == 4) key_control_sensor4 = distance;
  else if(pinNum == 5) key_control_sensor5 = distance;

}
```