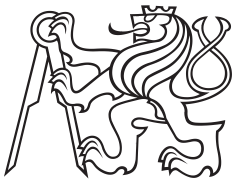
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra radioelektroniky

Digitální syntezátor

Ram Emelyanov

Školitel: doc. Ing. Stanislav Vítek, Ph.D.
Obor: Elektronika a komunikace
Květen 2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Emelyanov** Jméno: **Ram** Osobní číslo: **495624**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra radioelektroniky**

Studijní program: **Elektronika a komunikace**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Digitální syntezátor

Název bakalářské práce anglicky:

Digital Synthesizer

Pokyny pro vypracování:

Navrhněte a implementujte digitální syntezátor založený na platformě Arduino (nebo ESP32) s využitím knihovny Mozzi. Syntezátor bude možné ovládat pomocí místních ovládacích prvků (potenciometry a tlačítka) nebo vzdáleně pomocí vhodného protokolu (MIDI, OSC). Pro vzdálené ovládání syntezátoru navrhněte a pomocí frameworku Qt implementuje programové vybavení s grafickým rozhraním.

Seznam doporučené literatury:

[1] EDSTROM, Brent. Arduino for Musicians: A Complete Guide to Arduino and Teensy Microcontrollers. Oxford University Press, 2016.
[2] PUCKETTE, Miller. The Theory and Techniques of Electronic Music. World Scientific Publishing Company, 2007.
[3] LAZAR, Guillaume; PENEA, Robin. Mastering Qt 5: Create stunning cross-platform applications using C++ with Qt Widgets and QML with Qt Quick. Packt Publishing Ltd, 2018.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Stanislav Vítek, Ph.D. katedra radioelektroniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **30.01.2023** Termín odevzdání bakalářské práce: **26.05.2023**

Platnost zadání bakalářské práce: **22.09.2024**

doc. Ing. Stanislav Vítek, Ph.D. podpis vedoucí(ho) práce	doc. Ing. Stanislav Vítek, Ph.D. podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. | Datum převzetí zadání

Podpis studenta

Poděkování

Rád bych poděkoval svému vedoucímu doc. Ing. Stanislavu Vítkovi, Ph.D. za odborné konzultace a cenné rady.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Abstrakt

Tato práce se zabývá návrhem a implementací digitálního syntezátoru. Využívá mikroprocesorovou desku Arduino spolu s rozšířenou zvukovou knihovnou Mozzi. Digitální syntezátor je ovládán potenciometry, tlačítky a piezoelektrickými prvky. Zvukový výstup je přenášen na reproduktor a interaktivní menu je realizováno pomocí LCD modulu.

Klíčová slova: digitální syntezátor, Arduino, Mozzi, modulace zvuku, syntéza zvuku

Školitel: doc. Ing. Stanislav Vítek, Ph.D.

Abstract

This work deals with suggestion and implementation of a digital synthesizer. It uses the microprocessor board Arduino along with an expanded sound library Mozzi. The digital synthesizer is controlled by potentiometers, buttons and piezoelectric elements. The sound output is transferred onto a speaker and an interactive menu is implemented using an LCD module.

Keywords: digital synthesizer, Arduino, Mozzi, sound modulation, sound synthesis

Obsah

Obrázky

Tabulky

Kapitola 1

Úvod

Elektronika a produkce hudby mají velmi dlouhou historii. Zavedení zvuku a hudby do digitální podoby výrazně změnilo hudební průmysl. V současné době je digitální replikace hudby velmi rozšířená a lze ji vidět v mnoha aspektech každodenního života.

Implementace zvukového syntezátoru pomocí desky Arduino otevírá mnoho možností pro různé zvukové syntézy. Arduino je dostatečně výkonný nástroj ve spojení se vstupními a výstupními periferiemi k výrobě digitálního syntezátoru. Výhodou této implementace je široká škála možností přizpůsobení a úpravy různých zvukových parametrů podle potřeb uživatele.

Deska Arduino je relativně levný způsob, jak navrhnout syntezátor a je také kompatibilní s mnoha rozhraními, což rozšiřuje jeho použitelnost.

Kapitola 2
Teoretická část

2.1 Syntéza zvuku

Zvuk lze syntetizovat různými metodami, jako je například syntéza frekvenční modulací, subtraktivní syntéza a aditivní syntéza [1]. Aby bylo možné generovat různé audio výstupy, lze také použít syntézu vzorků a *wavetable* syntézu.

Každá kategorie syntéz se zabývá různými parametry zvukových vln a jejich frekvencí. Tato implementace digitální syntézy využívá základní generování vln a použití zvukových vzorků, které jsou následně syntetizovány k vytvoření zvuku. Zvuk bude zpracován pomocí tří modulačních technik, kterými jsou frekvenční modulace, amplitudová modulace a fázová modulace.

2.1.1 Amplitudová modulace

Modulace je proces změny jednoho nebo několika parametrů vysílané vlny známé jako nosná vlna. Změny nosné vlny jsou v souladu s variacemi druhého signálu známého jako modulační vlna. Existuje mnoho modulačních metod, které kombinují nosné a modulační vlny. Výsledkem takových modulací je modulovaný signál [2].

Amplitudová modulace je základní metoda modulace vln, která se používá při rádiovém přenosu a v souvislosti s digitální syntézou při modulaci zvukových vln.

Základem amplitudové modulace je to, že parametr amplitudy nosné vlny se mění úměrně s parametrem amplitudy modulační vlny.

K dosažení této formy modulace může být modulační vlna definována následovně [2]

$$m\left(t\right) = A_{\mathrm{m}} \sin\left(2\pi f_{\mathrm{m}} t\right), \tag{2.1}$$

kde A_{m} je amplituda a f_{m} je frekvence modulační vlny.

Nosnou vlnu lze definovat jako

$$n\left(t\right) = A_{\mathrm{n}} \sin\left(2\pi f_{\mathrm{n}} t\right), \tag{2.2}$$

kde A_{n} je amplituda a f_{n} je frekvence nosné vlny.

K dosažení požadovaného výsledku modulace se používá následující kombinace dvou vln

$$s\left(t\right) = \left[A_{\mathrm{n}} + A_{\mathrm{m}} \sin\left(2\pi f_{\mathrm{m}} t\right)\right] \sin\left(2\pi f_{\mathrm{n}} t\right). \tag{2.3}$$

Rovnici lze přeskupit a určit důležitý modulační parametr. Přeuspořádání rovnice dává následující výsledek

$$s\left(t\right) = A_{\mathrm{n}} \left[1 + \left(\frac{A_{\mathrm{m}}}{A_{\mathrm{n}}}\right) \sin\left(2\pi f_{\mathrm{m}} t\right)\right] \sin\left(2\pi f_{\mathrm{n}} t\right). \tag{2.4}$$

Poměr amplitud modulační a nosné vlny je znám jako modulační index. Tento index je definován následovně

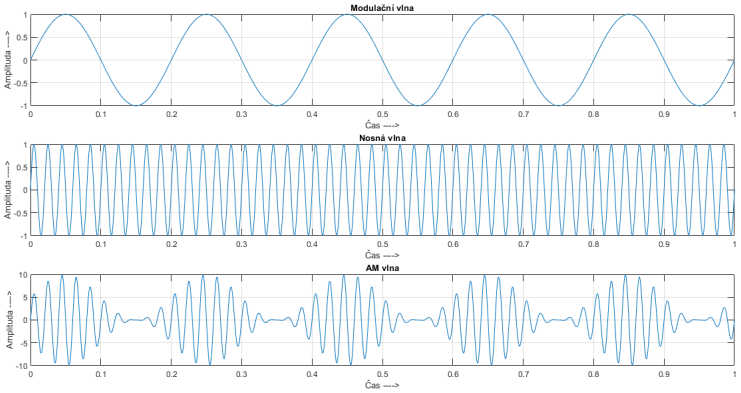
$$m_{\mathrm{a}} = \left(\frac{A_{\mathrm{m}}}{A_{\mathrm{n}}}\right). \tag{2.5}$$

Výsledná podoba rovnice amplitudové modulace po dosazení modulačního indexu vypadá takto [2]

$$s\left(t\right) = A_{\mathrm{n}} \left[1 + m_{\mathrm{a}} \sin\left(2\pi f_{\mathrm{m}} t\right)\right] \sin\left(2\pi f_{\mathrm{n}} t\right). \tag{2.6}$$

Grafické znázornění amplitudové modulace je na obrázku 2.1 Modulační vlna a nosná vlna mají obě amplitudu 1. Frekvence nosné vlny je 50 Hz a frekvence modulační vlny je 5 Hz.

Účinky amplitudové modulace jsou jasně pozorovány. Frekvence modulovaného signálu zůstává stejná jako u nosného signálu, ale amplituda osciluje podle hodnoty frekvence modulační vlny. Když modulační signál dosáhne své maximální amplitudy, obálka modulované vlny dosáhne svého nejvyššího bodu. Když je modulační vlna na svém nejnižším bodě, obálka modulovaného signálu se zúží a je také na svém nejnižším místě.



Obrázek 2.1: Amplitudová modulace.

2.1.2 Frekvenční modulace

Dalším typem modulace v této implementaci je úhlová modulace. Tato třída modulace zahrnuje jak frekvenční modulaci, tak fázovou modulaci.

K určení úhlové modulace lze použít následující základní rovnici [2]

$$s\left(t\right) = A\ \cos \int_0^t w\left(t\right)\ dt, \tag{2.7}$$

kde $w(t)$ je okamžitá hodnota úhlové rychlosti vlnového vektoru. Tuto hodnotu lze vyjádřit jako součet dvou složek. První složka, kterou lze označit jako w_n, je konstantní a odpovídá úhlové rychlosti nemodulované nosné vlny. Druhá složka označená jako $w_1(t)$ je závislá na čase a odpovídá amplitudě modulačního signálu.

$$w\left(t\right) = w_\mathrm{n} + w_1\left(t\right). \tag{2.8}$$

V případě frekvenční modulace lze složku $w_1(t)$ zapsat jako

$$f_1\left(t\right) = bA_\mathrm{m} \cos\left(2\pi f_\mathrm{m} t\right), \tag{2.9}$$

kde A_m je amplituda modulační vlny, f_m je frekvence modulační vlny a b je konstanta rovna frekvenčnímu posunu.

Časově závislou složku v rovnici (2.8) lze nahradit vzorcem z rovnice (2.9) který se pak dosadí do integrálu v rovnici (2.7) a získá se následující výsledek

$$s\left(t\right) = A_\mathrm{n} \cos\left(2\pi f_\mathrm{n} t + \int_0^t 2\pi b A_\mathrm{m} \cos\left(2\pi f_\mathrm{m} t\right)\right)\ dt. \tag{2.10}$$

Po provedení integrace může být výraz bA_m nahrazen proměnou f_s, což je parametr známý jako frekvenční výkyv.

Další substitucí $m = \frac{f_\mathrm{s}}{f_\mathrm{d}}$ se určí modulační faktor, který je poměrem frekvenčního výkyvu a frekvenční odchylky. Při zohlednění obou těchto substitucí lze získat následující výsledek

$$s\left(t\right) = A_\mathrm{n} \cos\left(2\pi f_\mathrm{n} t + \left(\frac{m f_\mathrm{d}}{f_\mathrm{m}}\right) \sin\left(2\pi f_\mathrm{m} t\right)\right). \tag{2.11}$$

Modulační index označený jako m_f lze definovat následovně

$$m_\mathrm{p} = \frac{m f_\mathrm{d}}{f_\mathrm{m}} \tag{2.12}$$

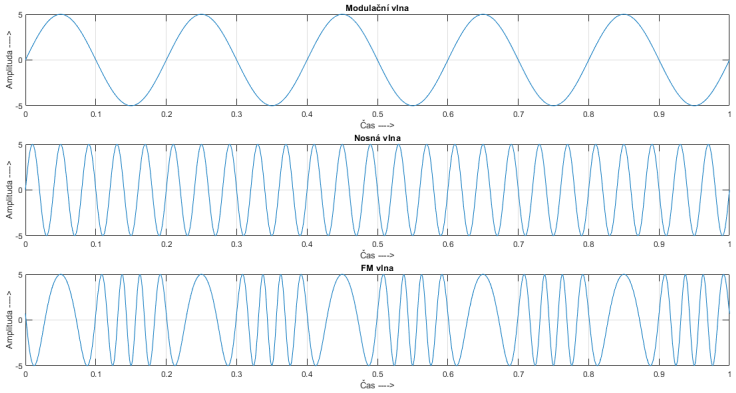
Dosazením modulačního indexu do rovnice (2.11), konečný výraz pro frekvenční modulaci se získá následovně [2]

$$s\left(t\right) = A_\mathrm{n} \cos\left(2\pi f_\mathrm{n} t + m_\mathrm{p} \sin\left(2\pi f_\mathrm{m} t\right)\right). \tag{2.13}$$

Obrázek 2.2 ukazuje frekvenční modulaci graficky. Parametry vlny a modulace jsou následující: index modulace je roven 3, frekvence modulační vlny je 5 Hz a frekvence nosné vlny je rovna 25 Hz.

Amplituda modulovaného signálu má hodnotu 5 protože amplituda vlny se u tohoto typu modulace nemění. Změny ve frekvenci lze pozorovat jako

„komprese“ a „zředění“. Vizuálně modulovaný signál připomíná tvar pružiny. Stupeň takových kompresí a zředění je úměrný amplitudě modulačního signálu. Výskyt těchto kompresí a zředění odpovídá frekvenci nosné vlny.



Obrázek 2.2: Frekvenční modulace.

2.1.3 Fázová modulace

Další typ úhlové modulace je známý jako fázová modulace. Fázová modulace je proces, který kóduje modulační signál jako změny v okamžité fázi nosné vlny.

Pro odvození vztahu pro fázovou modulaci je definována následující rovnice [2]

$$\phi\left(t\right) = b_1 A_{\mathrm{a}} \cos 2\pi f_{\mathrm{m}} t, \tag{2.14}$$

kde A_{m} je amplituda modulační vlny, f_{m} je frekvence modulační vlny a b_1 je konstanta rovna fázovému posunu.

Tento výraz definuje okamžitou hodnotu fázového úhlu a může být také vyjádřen následujícím způsobem

$$\phi\left(t\right) = \int_0^t w_1\left(t\right)\, dt. \tag{2.15}$$

Když se rovnice (2.14) zkombinuje s rovnicemi (2.7) a (2.8), lze získat následující signál

$$s\left(t\right) = A_{\mathrm{n}} \sin\left(2\pi f_{\mathrm{n}} t + b_1 A_{\mathrm{m}} \sin\left(2\pi f_{\mathrm{m}} t\right)\right). \tag{2.16}$$

Analogicky k substituci provedené ve frekvenční modulaci lze $b_1 A_{\mathrm{m}}$ nahradit $m\phi_{\mathrm{d}}$ a dostat

$$s\left(t\right) = A_{\mathrm{n}} \sin\left(2\pi f_{\mathrm{n}} t + m\phi_{\mathrm{d}} \sin\left(2\pi f_{\mathrm{m}} t\right)\right), \tag{2.17}$$

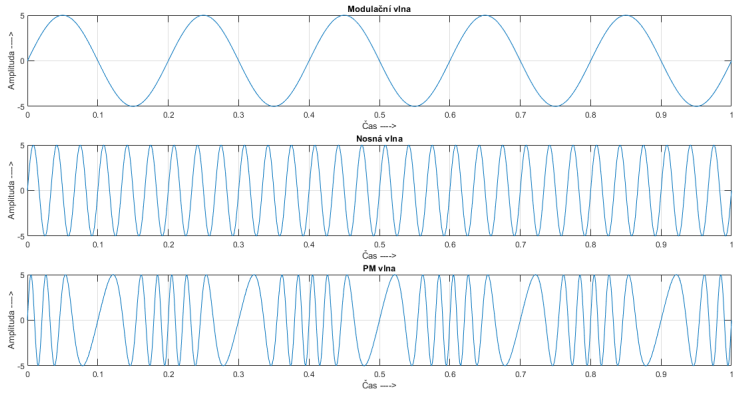
kde m je modulační faktor a ϕ_{d} je fázový posun produkovaný maximální amplitudou modulační vlny.

V rovnici (2.15) je znázorněn vztah mezi okamžitou hodnotou fázového posunu a okamžitou hodnotou úhlové rychlosti. Úhlová rychlost je úměrná frekvenci vlny. To znamená, že jakýkoli fázový posun vlny bude doprovázen posunem frekvence a naopak.

Frekvenčně modulovanou vlnu lze uvažovat z hlediska fázového posunu nosného vektoru. Totéž platí, že fázově modulovanou vlnu lze uvažovat z hlediska frekvenčního posunu.

Grafické znázornění fázové modulace je vidět na obrázku 2.3. Amplituda modulované vlny zůstává stejná jako amplituda nosné vlny. Modulační index je roven 4, frekvence modulační vlny je 5 Hz a frekvence nosné vlny je rovna 30 Hz.

Účinky frekvenční modulace a fázové modulace pro sinusové průběhy lze pozorovat jako velmi podobné na obrázku 2.3. Vzhledem k dříve stanovenému vztahu mezi fázovým a frekvenčním posunem a z důvodu zvukové implementace jsou obě tyto modulace zaměnitelné. To není stejný případ pro obdélníkové nebo trojúhelníkové průběhy.



Obrázek 2.3: Fázová modulace.

2.1.4 Vzorkování

Aby bylo možné replikovat zvuky nahrané z analogového zdroje, lze tento analogový signál digitalizovat. Když je zvukový vzorek digitalizován, může být uložen a replikován. Různé analogové zvuky, jako jsou zvuky hudebních nástrojů, lze zaznamenat kvůli replikaci. To je velmi běžná praxe v mnoha formách elektronické hudby [1].

Pro záznam vzorků se používá koncept známý jako vzorkování. Vzorkování je proces převodu signálu ve spojitém čase na signál s diskrétním časem [2]. Aby toho bylo dosaženo, používá se kvantizace, která mapuje spojité nekonečné hodnoty analogového signálu do menší sady diskrétních konečných hodnot. Proces kvantování lze pozorovat na obrázku 2.4.

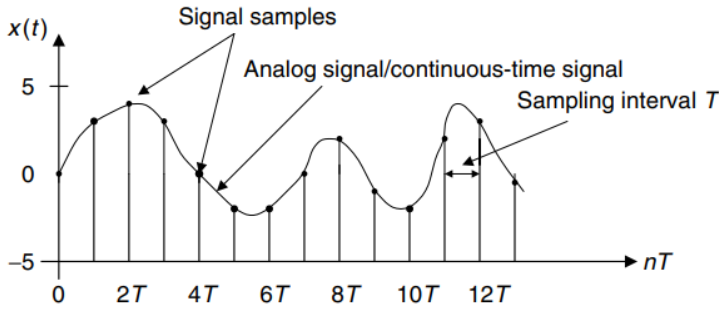
Velmi důležitým parametrem, který je třeba vzít v úvahu při vzorkování signálu, je vzorkovací frekvence, která je vyjádřena jako [3]

$$f_{\mathrm{s}} = \frac{1}{T}, \tag{2.18}$$

kde T je interval definován jako čas mezi dvěma body vzorkování.

Čím vyšší je vzorkovací frekvence, tím více informací o signálu lze získat. Velmi zásadní podmínkou je, že musí platit Nyquistův–Shannonův vzorkovací teorém. Tato věta říká, že vzorkovací frekvence musí být alespoň dvakrát větší než maximální frekvence vzorkovaného signálu [4]. To se provádí, aby se zabránilo aliasingu, ke kterému dochází, když vzorkovací frekvence není dostatečně vysoká, aby přesně reprezentovala signál. Věta může být reprezentována následovně [3]

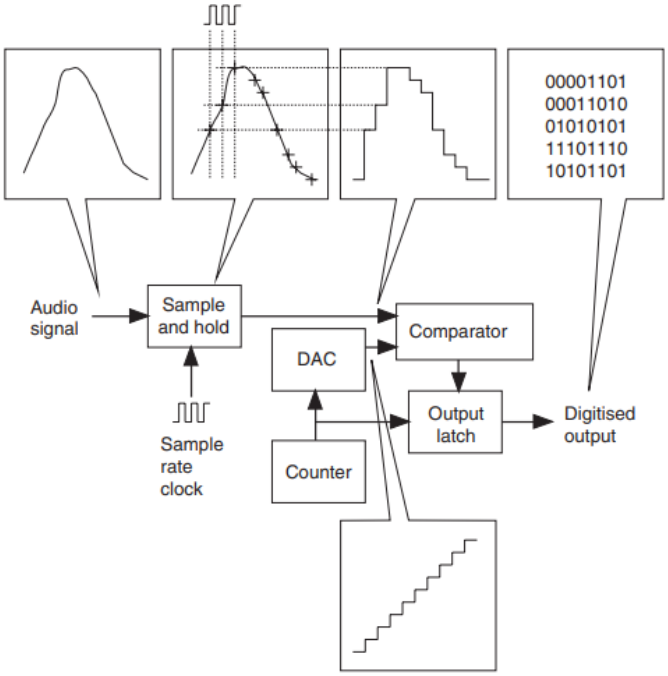
$$f_{\mathrm{s}} \geq 2f_{\max}. \tag{2.19}$$



Obrázek 2.4: Princip kvantování [3].

K fyzické implementaci vzorkování a převedení analogového signálu do digitalizované podoby lze použít metodu *sample and hold*. Blokové schéma této metody je vidět na obrázku 2.5.

Analogově digitální převodník získává zvukový průběh, který je v pravidelných intervalech zkoumán. Analogová paměť známá jako *sample and hold* zachycuje okamžitou hodnotu napětí a uchovává ji. Tato držená hodnota je poté porovnána s hodnotou vytvořenou čítačem připojeným k digitálně analogovému převodníku. Pokud komparátor ukazuje, že se obě hodnoty shodují, pak je hodnota vzorku ukotvena na výstupu analogově-digitálního převodníku. Čítač se vynuluje a další vzorek se odebere se vzorkovací frekvencí a proces se opakuje [1].



Obrázek 2.5: Blokové schéma pro metodu *sample and hold* [1].

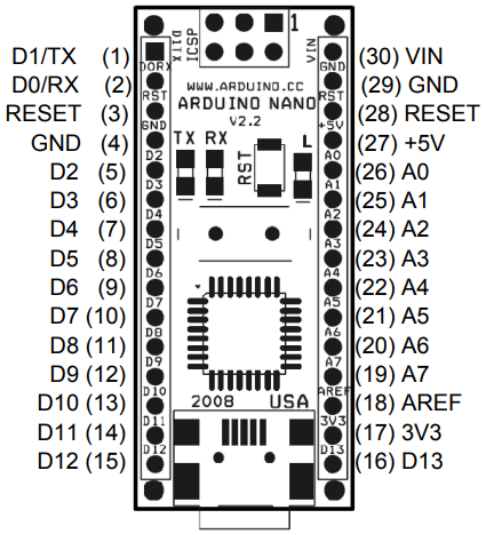
2.2 Arduino

Arduino je open source elektronická platforma založená na softwaru i hardwaru. Desky Arduino jsou schopny přijímat a číst vstupy a produkovat různé výstupy [5]. Funkce desky je dána instrukcemi, které jsou naprogramovány v mikroprocesoru umístěném na samotné desky.

K Arduinu lze připojit různé vstupní periferie, jako jsou potenciometry, piezo prvky a tlačítka. Výstupní periferie, jako jsou LED diody, reproduktory a LCD obrazovka, lze také připojit k mikroprocesoru Arduino. Pro případ vytvoření digitálního syntezátoru je Arduino dostatečně výkonný nástroj pro přizpůsobení požadovaných vstupů a výstupů pro vytváření efektů zvukové syntézy.

2.2.1 Hardware

Arduino nabízí řadu různých desek, ale pro tento projekt byl použit model Arduino Nano. Schéma desky je vidět na obrázku 2.6.



Obrázek 2.6: Rozložení pinů Arduina[6].

Arduino Nano má 30 portů s digitálními a analogovými vstupy a digitálními výstupy pro připojení externích periferií za účelem čtení nebo odesílání dat a příkazů. Deska má také port pro elektrické uzemnění a elektrické napájení. Funkce a popisy jednotlivých portů jsou uvedeny v tabulce 2.1.

Tabulka 2.1: Arduino Nano porty s popisem a funkcí.

Číslo pinu	Název	Typ	Popis
1	D1/TX	I/O	Digitální vysílač
2	D0/RX	I/O	Digitální přijímač
3, 28	RESET	Vstup	Resetování
4, 29	GND	PWR	Uzemnění
5-16	D2-D13	I/O	Digitální porty pro vstup a výstup
17	3V3	Výstup	+3.3V výstup
18	AREF	Vstup	Reference ADC
19-26	A7-A0	Vstup	Analogové vstupní porty
27	+5V	Vstup nebo výstup	+5V výstup (na desce), +5V vstup (externí)
30	VIN	PWR	Napájecí napětí

Deska má 2 KB RAM a 32 KB flash paměti [5]. To jsou velmi důležité parametry, což je třeba vzít v úvahu zejména při práci na projektech, které zabírají hodně paměti a výpočetního výkonu, jako je tento projekt.

Desku lze připojit k počítači pomocí MINI-B USB kabelu. Jakmile je kód nahrán z počítače, deska ke svému chodu vyžaduje pouze napájení.

2.2.2 Software

Integrované vývojové prostředí Arduino používá k programování mikroprocesoru na desce programovací jazyk Arduino. Pracuje prostřednictvím API a je založen na jazycích C a C++.

Nejzákladnější struktura kódu Arduino obsahuje funkci nastavení, kde jsou nejprve definovány všechny parametry. Má také funkci zvanou smyčka, kde se všechny příkazy provádějí ve smyčce a kde se odehrává hlavní část kódu.

Pro tuto implementaci je zahrnuta další knihovna nazvaná Mozzi. Zvukový výstup Arduino je zcela základní, takže pro vytváření mnohem složitějších zvuků a typů syntézy a modulace se používá knihovna Mozzi.

Velmi základním parametrem knihovny Mozzi je vzorkovací frekvence, která je 16384 Hz [7]. Struktura kódu Mozzi také obsahuje funkci s názvem aktualizace zvuku. Tato funkce vrací hodnoty zvuku, který bude generován.

Kapitola 3
Praktická implementace

3.1 Zapojení obvodu

Tato kapitola se zabývá fyzickým zapojením obvodu a také příjmem a zpracováním vstupních signálů. Zahrnuje připojení desky Arduino spolu s ovládacími prvky a výstupními prvky. V této implementaci existuje několik komponent, které posílají informace na vstupy desky Arduino.

První skupinou ovládacích prvků je piezoelektrický prvek. Účelem této součásti je působit jako aktivátor syntetizovaného zvuku. Funkce je analogická ke klávesám klavíru nebo syntezátoru.Vzhledem k omezenému výpočetnímu výkonu a počtu analogových vstupních portů, kterými Arduino disponuje, jsou k desce připojeny pouze tři z těchto piezo senzorů.

Když je detekován vstup z piezoelektrického prvku, signalizuje, že by měl být generován zvuk. K obvodu jsou také připojeny potenciometry, které regulují různé modulační parametry. Tyto potenciometry dokážou upravovat parametry zvuku jak při přehrávání zvuku (stisknutí a přidržení piezo senzoru), tak mezi generováním zvuku (piezo senzor není stisknutý). Úpravy potenciometrů jsou plynulé a nediskrétní, což dává plynulý přechod v parametrech zvuku.

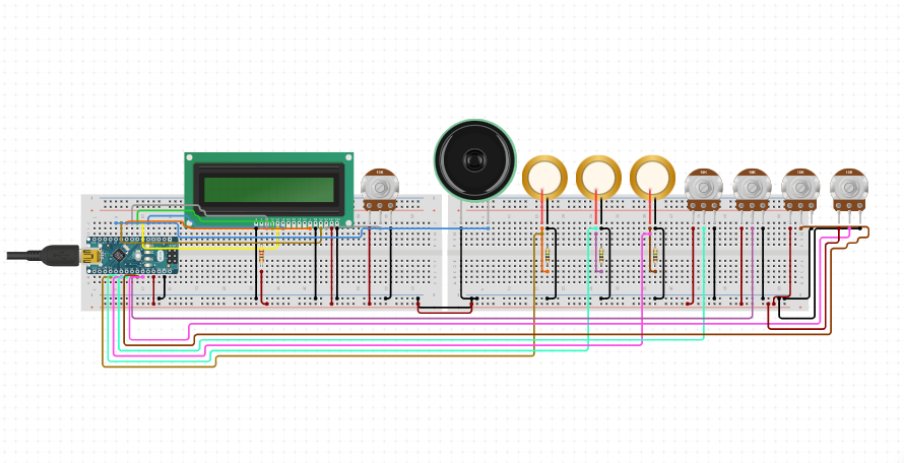
Poslední ovládací prvek je umístěn na LCD obrazovce. Modul použitý pro zobrazení se jmenuje LCD Shield Robot. Tento modul umožňuje jak odebírat data z desky Arduino, tak současně odesílat informace z tlačítek umístěných na modulu.

Poslední komponentou připojenou k desce Arduino je reproduktor. Reproduktor je připojen k digitálnímu výstupnímu kolíku a je uzemněn. Účelem je přijímat výstup z desky Arduino a reprodukovat zvuk. Zapojení analogových a digitálních portů je uvedeno v tabulce 3.1.

Tabulka 3.1: Připojení analogových a digitálních portů a jejich funkce.

Název pinu	Součást	Funkce
D2	D7 (LCD)	Přenos dat
D3	D6 (LCD)	Přenos dat
D4	D5 (LCD)	Přenos dat
D5	D4 (LCD)	Přenos dat
D6	V0 (LCD)	Nastavení jasu
D9	Reproduktor	Reprodukce zvuku
D11	E (LCD)	Hodiny (Enable)
D12	RS (LCD)	Register Select
A0	A0 (LCD)	Úroveň tlačítek
A1	Piezo č. 1	Analogový vstup
A2	Piezo č. 2	Analogový vstup
A3	Piezo č. 3	Analogový vstup
A4	Potenciometr 1	Analogový vstup
A5	Potenciometr 2	Analogový vstup
A6	Potenciometr 3	Analogový vstup
A7	Potenciometr 4	Analogový vstup

Kromě analogových a digitálních vstupních a výstupních pinů je port 5V na Arduinu připojen k jedné z nohou všech potenciometrů a také k modulu LCD. Další noha každého potenciometru je připojena k uzemněnému portu Arduina. Všechny piezoelektrické prvky jsou také uzemněny přes odpor. Uzemněný port Arduina slouží také k uzemnění GND pinu LCD modulu. Pin VSS na LCD, který působí jako uzemnění systému, je také připojen k uzemněnému portu desky Arduino. Posledním pinoutem LCD, který musí být uzemněn, je RW (*Read-Write*) port. Nakonec port "VIN" LCD, který slouží jako napájení, je připojen k portu Arduino se stejným názvem. Schéma zapojení Arduina ke všem prvkům je vidět na obrázku 3.1.



Obrázek 3.1: Zapojení desky Arduino s komponenty.

3.2 Odebírání a zpracování vstupu

Následující část se zabývá získáváním vstupů ze tří hlavních ovládacích prvků, kterými jsou LCD tlačítka, potenciometry a piezo senzory. Příjem vstupů musí být optimalizován jak z hlediska hardwaru, tak i softwaru. Tato část také pokrývá zpracování vstupu způsobem, který je optimální pro implementaci digitálního syntezátoru.

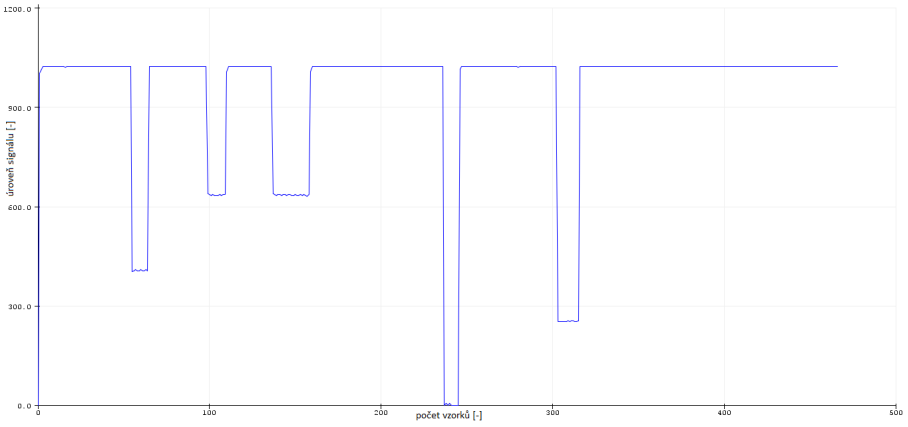
Arduino používá vícekanálový desetibitový analogově digitální převodník. Model, který se používá pro tuto implementaci, je Arduino Nano, který má provozní napětí 5 voltů. To znamená, že 5 voltů bude mapováno na celočíselnou hodnotu 0 až 1023. Rozlišení bude 5 voltů děleno 1024 jednotkami. To dává každé jednotce hodnotu 0.0049 V nebo 4.9 mV.

Softwarová funkce Arduino IDE, která čte hodnoty z analogových vstupů, se nazývá funkce *analogRead*. Výstupem této funkce je celé číslo v rozsahu 0 až 1023. Pro tuto implementaci digitálního syntezátoru je použita knihovna Mozzi. Výše zmíněná funkce není kompatibilní se strukturou kódu Mozzi kvůli přerušení, proto je použita velmi podobná funkce s názvem *mozziAnalogRead*. Tím je zajištěno, že ostatní operace v procesu nebudou přerušeny. Výstup funkce Mozzi také dává celé číslo od 0 do 1023, ale v praktické aplikaci se analogová hodnota ze vstupů může mírně lišit od standardní funkce.

3.2.1 Tlačítka

Účelem tlačítek na modulu LCD v této implementaci je změna různých režimů a zobrazení jejich popisu. Každý režim používá jinou zvukovou modulaci a techniku syntézy k produkci různých zvuků.

Jak je uvedeno v tabulce 3.1, port A0 na desce Arduino je zodpovědný za přijímání vstupu stavů tlačítka. Obvykle jsou tlačítka připojena k jednotlivým analogovým portům, avšak v případě modulu LCD je jeden port zodpovědný za čtení úrovně všech tlačítek. Způsob jeho implementace je vidět na obrázku 3.2. Různé úrovně vstupního signálu indikují stisknutí různých tlačítek.



Obrázek 3.2: Úroveň signálu pro tlačítka.

Princip, který tato metoda využívá, je založen na různých hodnotách odporu připojených k tlačítkům. Jak je znázorněno na obrázku 3.2, existuje klidový stav. Tento stav ukazuje nejvyšší hodnotu a signalizuje, že není stisknuto žádné tlačítko. Na ilustraci jsou také různé impulsy, které indikují stisknutí určitého tlačítka v závislosti na úrovni signálu.

Tabulka 3.2 ukazuje rozsahy, ve kterých se stav tlačítka změní na další. Při stisku tlačítka *RIGHT* se úroveň signálu pohybuje kolem nulové hodnoty. Další tlačítko *LEFT* dává hodnotu přibližně 105. Jak lze pozorovat při pohledu na úrovně signálu, protože je signál analogový, není dokonale stabilní. Aby bylo možné získat přesný údaj, je třeba zvolit vhodný práh, který je umístěn mezi dvěma úrovněmi tlačítek. Když je hodnota signálu nižší než prahová hodnota, program oznámí, že bylo stisknuto určité tlačítko. To platí pro všechna tlačítka.

Tabulka 3.2: Tlačítkové úrovně s prahy.

Název tlačítka	Rozsah úrovně signálu	Prahová hodnota úrovně
RIGHT	0-104	60
UP	105-254	200
DOWN	255-399	350
LEFT	400-624	600
SELECT	625-800	800
Klidový stav	800-1023	-

Další velmi důležitou věcí, kterou je třeba vzít v úvahu kromě úrovně signálu, je délka pulzů signálu. Jak je znázorněno na obrázku 3.2, délky impulsů se mohou lišit v závislosti na době, po kterou jsou stisknuty. Funkce *mozziAnalogRead* dokáže přečíst stovky vzorků za sekundu. To znamená, že bude indikovat, že prahová hodnota byla dosažena mnohokrát, a několikrát změní režim. Aby nedocházelo k vícenásobným změnám režimů při jednom stisku tlačítka, algoritmus, který se pro tuto implementaci používá, zohledňuje také to, kdy pulz skončí a signál se vrátí do klidového stavu. Teprve když jsou splněny dvě podmínky změny a návratu do klidového stavu, program oznámí, že bylo stisknuto tlačítko.

Tato metoda umožňuje, aby tlačítka byla stisknuta vícekrát za sebou, na rozdíl od indikace, že tlačítko bylo stisknuto, aniž by bylo známo, kolikrát bylo stisknuto.

3.2.2 Potenciometry

Další skupinou ovládacích prvků, které jsou v tomto syntezátoru implementovány, jsou potenciometry.

Potenciometr pracuje na principu řízení průtoku proudu změnou odporu. Použité potenciometry mají hodnotu 10 kΩ. Knoflík potenciometru umožňuje otáčení od 0° do 300°.

Opět desetibitový analogově digitální převodník slouží ke čtení hodnoty napětí, jenž je následně převedena na hodnotu v rozsahu 0 až 1023. Toto jsou minimální a maximální hodnoty, které lze číst ze vstupu potenciometru v závislosti na otáčení.

Pro usnadnění práce se čtením potenciometrů se používá následující optimalizační technika. Metoda spočívá v automatickém mapování a škálování hodnot přijímaných z potenciometru. To má zajistit pohodlné nastavení a úpravu parametrů.

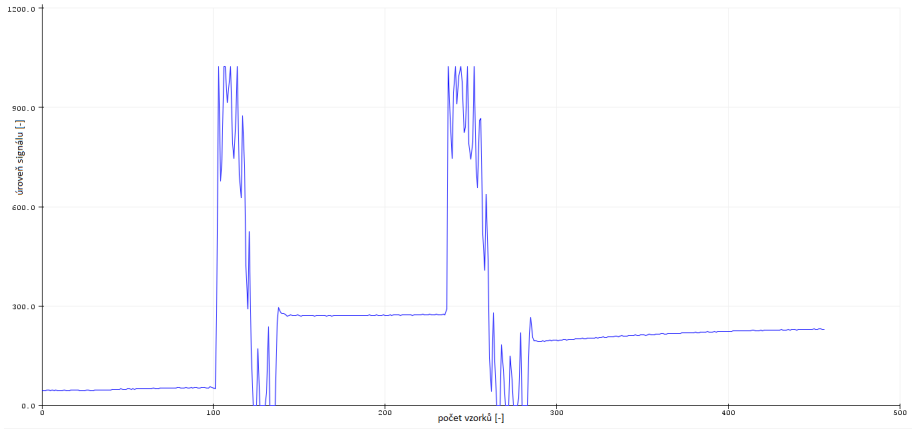
Mapování lze dosáhnout vytvořením třídy nazvané *AutoMap*. V této třídě může být deklarována funkce, která má čtyři argumenty. První dva argumenty jsou minimální a maximální čtené hodnoty, které jsou v případě potenciometru 0 až 1023. Třetí argument je mapovaná minimální hodnota a čtvrtý je mapovaná maximální hodnota. To také umožňuje pohodlné přeškálování pro obrácení dynamiky zvuku pro určité parametry.

3.2.3 Piezoelektrický prvek

Posledním ovládacím prvkem v je piezo snímač. Tato část pojednává o tom, jak je přijímán a optimalizován vstup z piezo snímače z hardwarového i softwarového hlediska.

Výhody použití piezoelektrického prvku ve srovnání s komponentou, jako je tlačítko pro tuto implementaci, jsou četné. Piezo senzor je velmi citlivý na změny, takže lze detekovat signál s různými úrovněmi tlaku a dobou, po kterou je tlak aplikován. Piezo při stisku nevydává zbytečný zvuk jako tlačítko, které by rušilo snahu o co nejlepší zvuk syntezátoru. Výstup tlačítka je plochý signál, zatímco piezodetektor ukazuje signál v závislosti na úrovni tlaku, což je parametr, který lze využít při změně syntézy zvuku.

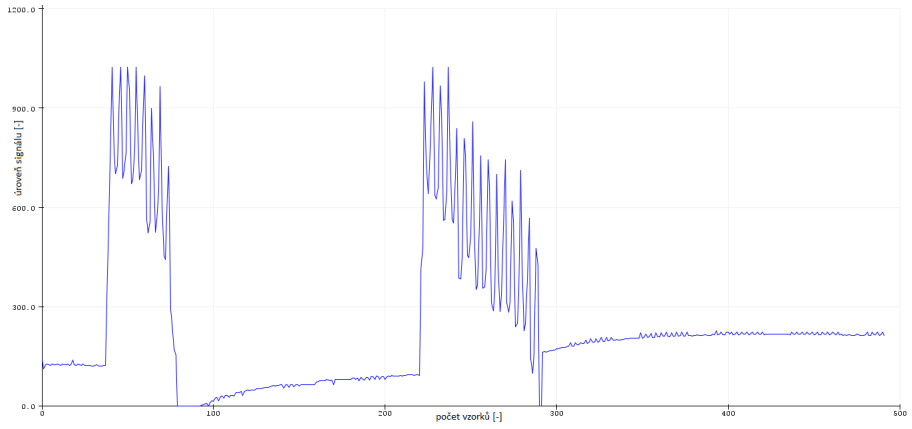
Když je jeden vodič od piezo snímače připojen k zemnící svorce a druhý vodič je připojen k analogovému vstupnímu portu, lze získat následující výsledky, které jsou znázorněny na obrázku 3.3.



Obrázek 3.3: Vstup z piezoelektrického prvku.

Obrázek 3.3 ukazuje dva impulsy, ke kterým dochází při klepnutí na piezoelektrický prvek. Po každém klepnutí na piezo snímač je získán impuls. Toto čtení se získá pomocí funkce *analogRead*. Lze pozorovat, že po impulsu se signál zcela nevrátí do nulové hodnoty. Dosažení klidového nulového stavu trvá delší dobu. Protože úroveň impulsu je výrazně vyšší než doba vybíjení, lze navrhnout řešení, ve kterém je zavedena prahová hodnota. Jakmile je dosaženo tohoto prahu, zazní zvuk.

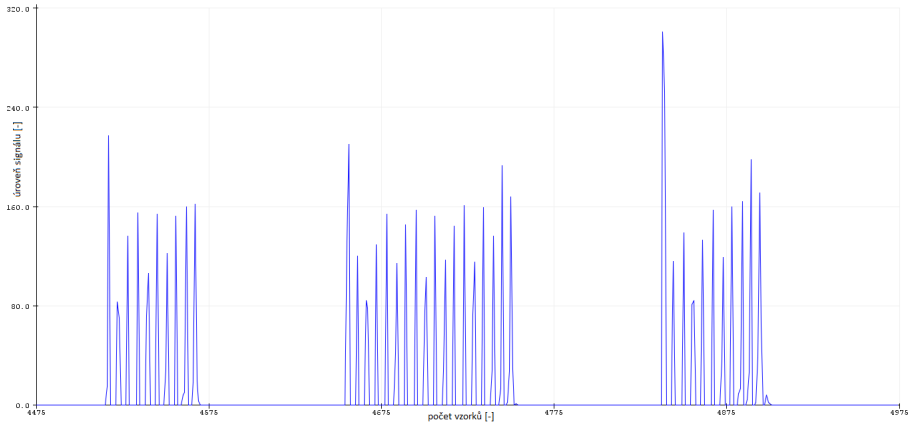
Tato implementace využívá knihovnu Mozzi, která používá funkci *mozziAnalogRead*. Jak lze pozorovat na obrázku 3.4, čtení z této funkce poskytuje mírně odlišný výsledek od standardní funkce. Po uvolnění piezo snímače signál klesá, ale pak se pomalu zvyšuje. To znamená, že kromě implementace prahové hodnoty musí existovat i jiné řešení.



Obrázek 3.4: Vstup z piezoelektrického prvku pomocí Mozzi.

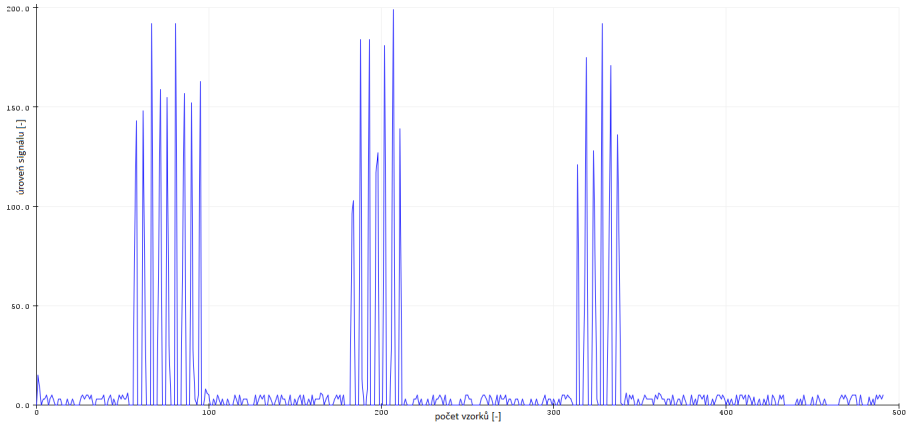
Řešením, které by mohlo tento problém vyřešit, je uzemnění analogového vstupu z piezo snímače. Hodnota odporu ovlivní citlivost piezoelektrického prvku. Čím vyšší je hodnota rezistoru, tím citlivější bude piezo na aplikovaný tlak. Tato implementace používá rezistor 100 kΩ, protože je dostatečně citlivý na dosažení požadovaných výsledků.

Když je k piezoelektrickému prvku připojen rezistor, výsledek lze pozorovat na obrázku 3.5. Problém návratu signálu do klidového stavu je odstraněn. Přidání rezistoru přidalo další problém, který je třeba vyřešit. Při stisknutí lze pozorovat, že pulsy oscilují nahoru a dolů. To znamená, že pokud má být implementován práh, signál nebude spojitý a zvuk bude přerušovaný a bude nepříjemný.



Obrázek 3.5: Vstup z piezoelektrického prvku s odporem 100 kΩ.

Problém se ještě umocní, když se místo standardní funkce použije funkce *mozziAnalogRead*. Kvůli omezením této funkce je pozorován šumový signál namísto klidového stavu. To je vidět na obrázku 3.6.



Obrázek 3.6: Vstup z piezoelektrického prvku s odporem 100 kΩ pomocí Mozzi.

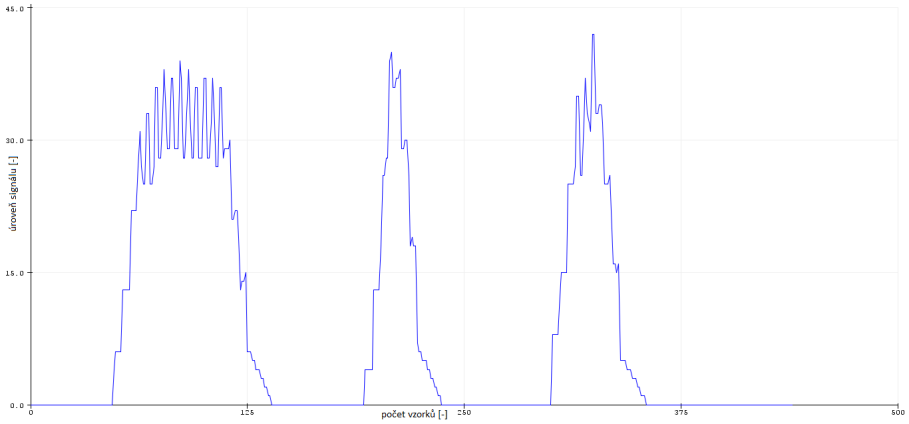
K vyřešení tohoto problému lze použít třídu *RollingAverage*. Vstupní parametr funkce vytvořené touto třídou určuje počet hodnot, které se mají zprůměrovat. Aby bylo možné tuto funkci použít, musí být počet hodnot k průměrování mocninou dvou.

Klouzavý průměr je typ matematické konvoluce, která vytváří časovou řadu, která je konstruována na základě průměrů několika po sobě jdoucích hodnot jiné časové řady. Pokud existuje původní řada $y_1, ..., y_n$, klouzavý průměr může být reprezentován pomocí následující rovnice [8]

$$z_t = \frac{1}{2k+1} \sum_{j=-k}^{k} y_t + j, \tag{3.1}$$

kde $t = k+1, k+2, ..., n-k$.

Pomocí konceptu dané rovnicí (3.1) jsou na obrázku 3.7 získány následující výsledky. Tyto výsledky jsou získány pomocí funkce analogového čtení z knihovny Mozzi.



Obrázek 3.7: Vstup z piezoelektrického prvku s odporem 100 kΩ a metodou klouzavého průměru

V klidovém stavu již není žádný šumový signál. Signál nekolísá celou cestu dolů k nule, což znamená, že lze implementovat práh, kde bude signál trvale nad ním. Nevýhodou této metody je, že vyžaduje značné množství výpočetního výkonu.

Kapitola 4
Softwarové zpracování

4.1 Výběr režimu a obrazovky

Tato kapitola se zabývá zpracováním vstupů za účelem dosažení modulace a syntézy zvuku a také odesláním výsledných signálů do reproduktoru, aby se vytvořil zvuk. Zahrnuje také výběr různých režimů, které určují, jakou techniku zvukové syntézy použít.

Jak bylo popsáno v části 3.2.1, tlačítka na modulu LCD určují, který režim má syntezátor zapnutý. Levé a pravé tlačítko umožňuje přepínání z pěti režimů. Tlačítka nahoru a dolů umožňují zobrazit popis funkce každého potenciometru pro daný režim. Text s názvem režimu a popisem je odeslán k zobrazení na LCD. Aby to bylo možné implementovat, používá se dvourozměrné pole. Kdykoli výběr režimu dosáhne posledního a znovu se stiskne tlačítko, vrátí se do prvního režimu. Stejný algoritmus platí pro popisy. To umožňuje neomezené přepínání mezi režimy v libovolném směru.

4.2 Režim 0: Přehrávání not

Režim nula v této implementaci umožňuje hrát různé noty. Rozsah not je uveden v tabulce 4.1. Noty pokrývají jednu oktávu a jsou v sekvenčním pořadí.

Noty jsou umístěny v poli s počáteční pozicí nula. Funkcí prvního potenciometru je procházet polem not. Každému piezo snímači je přiřazena počáteční pozice v poli, která se liší podobně jako u hudebního syntezátoru. První piezo senzor začíná na pozici 0 a může jít až do pozice 7 v závislosti na hodnotě potenciometru. Druhý piezoelektrický prvek začíná na pozici 2 a může jít až do 9 a třetí piezo senzor má počáteční pozici 4 a může jít až do 11. Potenciometrem lze otáčet při stisknutí piezo senzoru, čímž se okamžitě změní hraná nota.

Tabulka 4.1: Seznam not s frekvencemi [9].

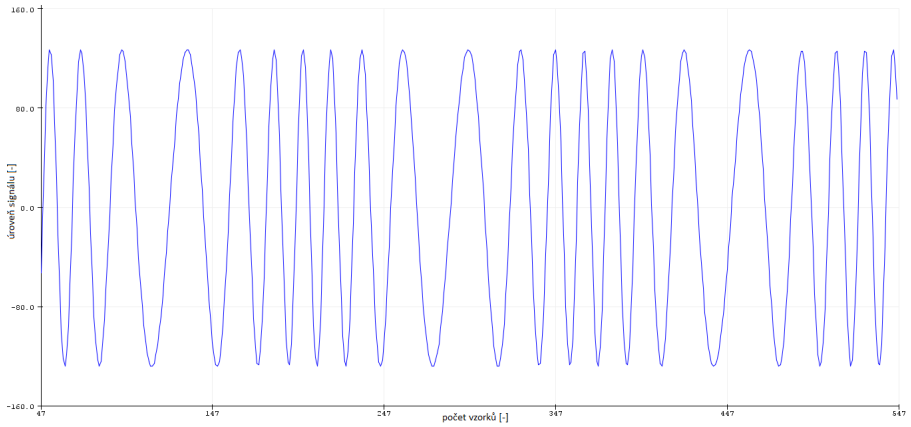
Pozici v poli	Tón	Frekvence [Hz]	Vlnová délka [cm]
0	C5	523	65.9
1	C#5	554	62.2
2	D5	587	58.7
3	D#5	622	55.4
4	E5	659	52.3
5	F5	698	49.4
6	F#5	740	46.6
7	G5	784	44.0
8	G#5	831	41.5
9	A5	880	39.2
10	A#5	932	37.0
11	B5	988	34.9

Jakmile je nota vybrána, frekvence pro sinusovou vlnu se nastaví podle frekvence not uvedených v poli. Druhý potenciometr je zodpovědný za regulaci zesílení nebo hlasitosti vlny. Sinusová vlna bude násobena přeškálovanou hodnotou druhého potenciometru. To umožňuje hrát notu při různých úrovních hlasitosti. Jakmile je frekvence vlny nastavena jako frekvence noty, může být odeslána do reproduktoru.

4.3 Režim 1: Vibrato

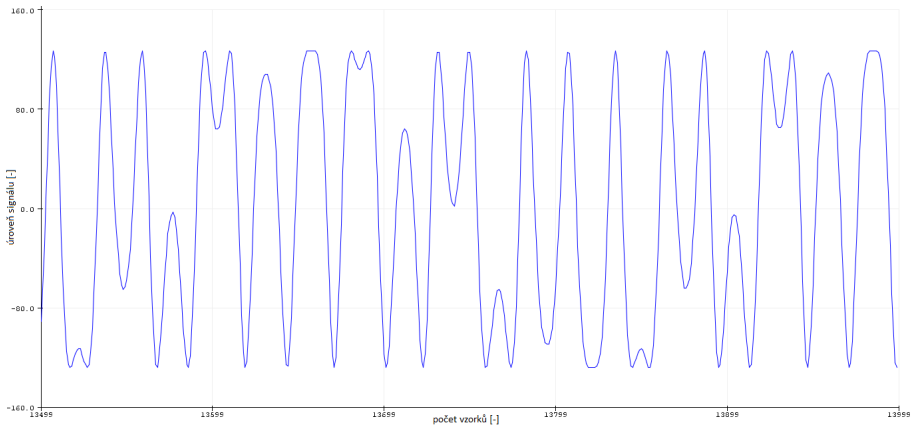
Tento režim využívá fázovou modulaci k vytvoření vibrato efektu, jak se tomu v hudbě říká [1]. Knihovna Mozzi obsahuje funkci nazvanou *phMod*, která funguje na principu znázorněném v rovnici (2.17). Funkce může provádět fázovou modulaci jedné vlny třídy *Oscil* (což je třída knihovny Mozzi pro vlny) jinou vlnou stejné třídy.

Obrázek 4.1 ukazuje modulovanou vlnu po aplikaci funkce fázové modulace s ohledem na to, že frekvence nosné vlny je 100 Hz a frekvence modulační vlny je 700 Hz.



Obrázek 4.1: Vibrato s frekvencemi 100 Hz pro nosnou vlnu a 700 Hz pro modulační vlnu.

Modulovaná vlna může mít různé tvary v závislosti na frekvenčních hodnotách vln, které jsou použity při její tvorbě. Například obrázek 4.2 představuje modulovanou vlnu, která je způsobena fázovou modulací nosné vlny s frekvencí 50 Hz a modulační vlny s frekvencí 300 Hz.



Obrázek 4.2: Vibrato s frekvencemi 50 Hz pro nosnou vlnu a 300 Hz pro modulační vlnu

Funkcí jednoho z potenciometrů je regulovat frekvenci modulační vlny. Druhý potenciometr je zodpovědný za regulaci frekvence nosné vlny. To umožňuje mnoho kombinací a možností produkce různých vibrato zvuků. Každý piezo snímač má jinou počáteční hodnotu pro modulovanou vlnu, která umožňuje každému piezoelektrickému snímači produkovat jedinečné zvuky.

Podobně jako v předchozím režimu umožňuje třetí potenciometr regulovat zesílení modulované vlny. Tento režim umožňuje, aby se frekvence modulační vlny pohybovala od 0 do 1023 Hz a frekvence nosné vlny od 15 až do 2818 Hz.

4.4 Režim 2: Fázová a frekvenční modulace

Tento režim zahrnuje složitější modulaci než dříve. Všechny potenciometry mají funkci pro tento režim. Každý potenciometr reguluje určitý parametr výsledné modulované vlny.

První potenciometr reguluje frekvenci nosné vlny. Pro tento režim je rozsah, který pokrývá nosná vlna, 100 až 500 Hz. Další potenciometr reguluje poměr mezi frekvencemi nosné vlny a modulační vlny. Poměr se pohybuje od 1 do 20. Funkci obou potenciometrů lze matematicky ilustrovat v následující rovnici

$$f_{\mathrm{m}} = f_{\mathrm{n}} \cdot m_{\mathrm{p}} \cdot m_{\mathrm{var}}, \tag{4.1}$$

kde f_{m} je frekvence modulační vlny a f_{n} je frekvence nosné vlny, m_{p} je konstantní modulační poměr s hodnotou 5 a m_{var} je variabilní modulační poměr.

Třetí potenciometr ovládá rychlost modulace a čtvrtý potenciometr intenzitu modulační vlny. Funkce může být reprezentována matematicky následovně

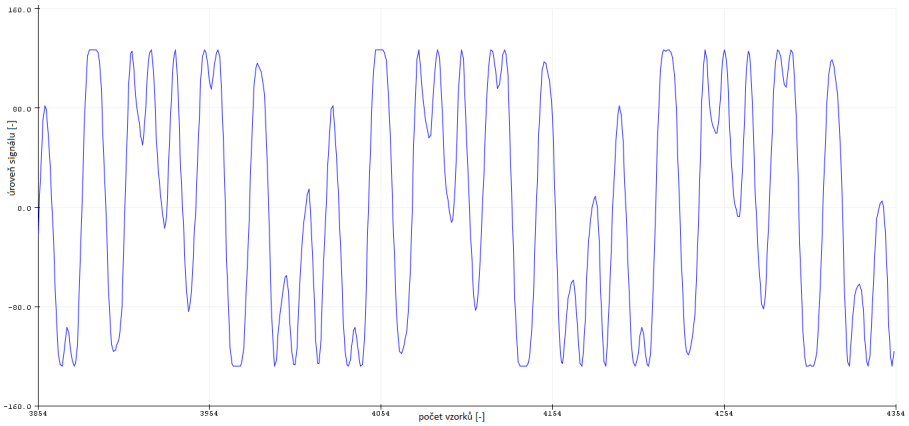
$$In_{\mathrm{cel}}(t) = In_{\mathrm{var}} \cdot \cos\left(2\pi f_{\mathrm{MR}} t\right) + K, \tag{4.2}$$

kde $In_{\mathrm{cel}}(t)$ je vlna pro celkovou intenzitu, In_{var} je proměnná intenzita, která se pohybuje od 10 do 750. f_{MR} je hodnota frekvence zodpovědná za rychlost modulace (v rozsahu od 0.001 do 10) a K je konstanta s hodnotou 128.

Konečný výraz pro modulační vlnu lze zapsat jako

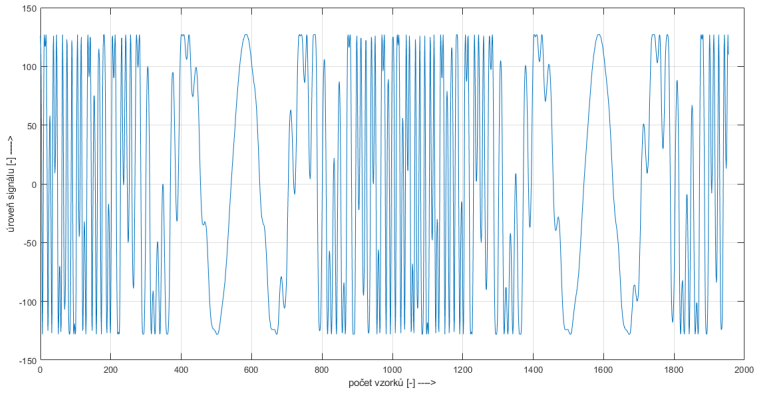
$$m_1(t) = In_{\mathrm{cel}}(t) \cdot \cos\left(2\pi f_{\mathrm{m}} t\right). \tag{4.3}$$

Mezi nosnou vlnou a modulační vlnou je pak aplikována funkce *phMod*, která poskytuje výsledek, který je vidět na obrázku 4.3. Frekvence nosné vlny prvního piezo snímače odpovídá aktuální mapované hodnotě prvního potenciometru. Další piezoelektrický prvek je o 100 Hz vyšší než první a třetí piezoelektrický prvek má frekvenci o 200 Hz vyšší než první piezoelektrický prvek. Tím je zajištěno, že každý piezoelektrický prvek bude znít jinak než ten předchozí.



Obrázek 4.3: Funkce *phMod* s nosnou vlnou 100 Hz.

Na obrázku 4.4 lze pozorovat složitější průběhy. To je důsledkem zvýšení intenzity modulace. Rozdíl je po zvukové stránce velmi patrný.

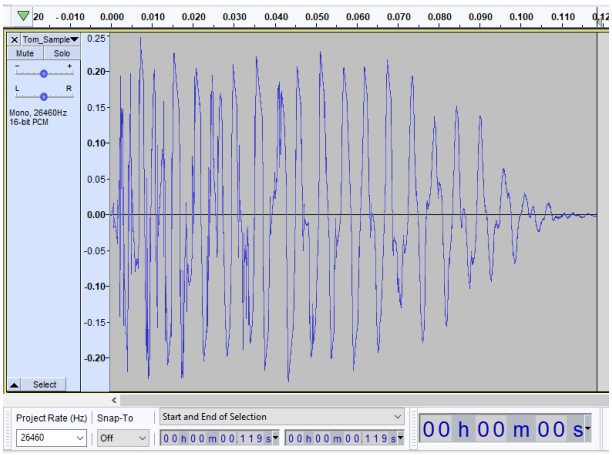


Obrázek 4.4: Funkce *phMod* s nastavením vysoké intenzity.

4.5 Režim 3: Vzorky

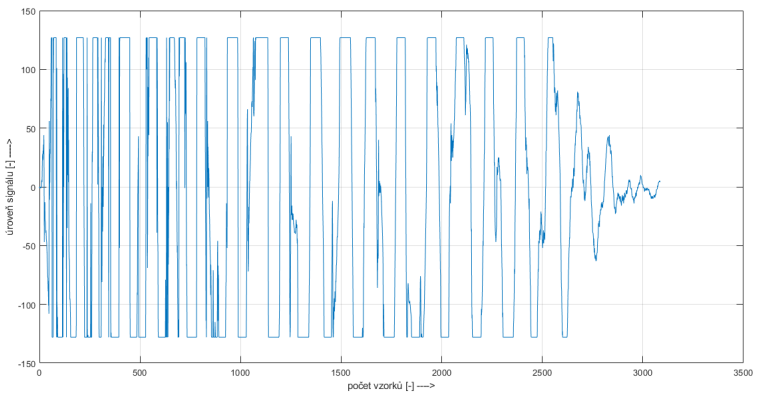
V tomto režimu se vzorky používají pro generování jedinečných zvuků. Každý piezo senzor aktivuje analogový vzorek bicího nástroje. To lze provést s jakýmkoliv vzorkem velikosti dostatečné pro uložení na desce Arduino.

Na obrázku 4.5 je znázorněn průběh hraní na tom-tom bubnu. Tento analogový vzorek je digitalizován a lze jej replikovat prostřednictvím digitální platformy.



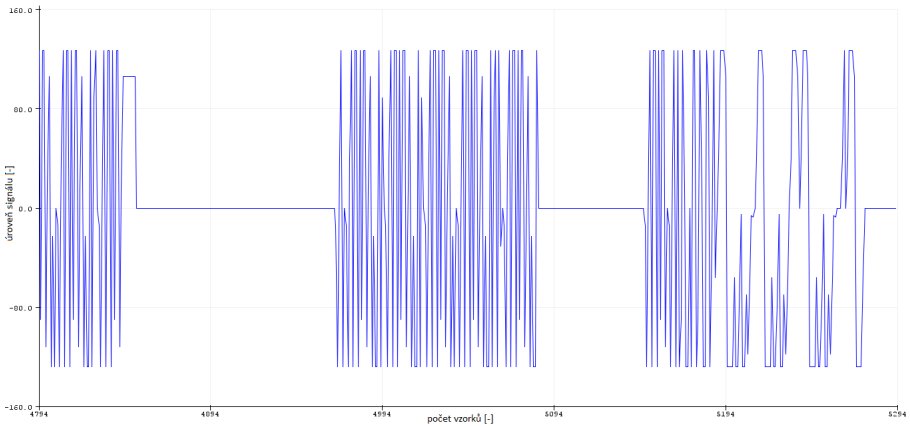
Obrázek 4.5: Průběh vzorku bubnu.

Vzorek zvuku je konvertován pomocí předem napsaného programu dostupného v knihovně Mozzi s názvem *char2mozzi*. Tento program převede soubor RAW do *wavetable* sekvence čísel. To umožňuje, aby byl vzorek kompatibilní s deskou Arduino. Když je vykreslena posloupnost čísel, výsledný průběh lze pozorovat na obrázku 4.6. Některá data se při konverze ztratí, ale výsledky jsou dostatečně blízko, aby produkovaly podobný zvuk.



Obrázek 4.6: Průběh vzorku bubnu po převodu na *wavetable* data.

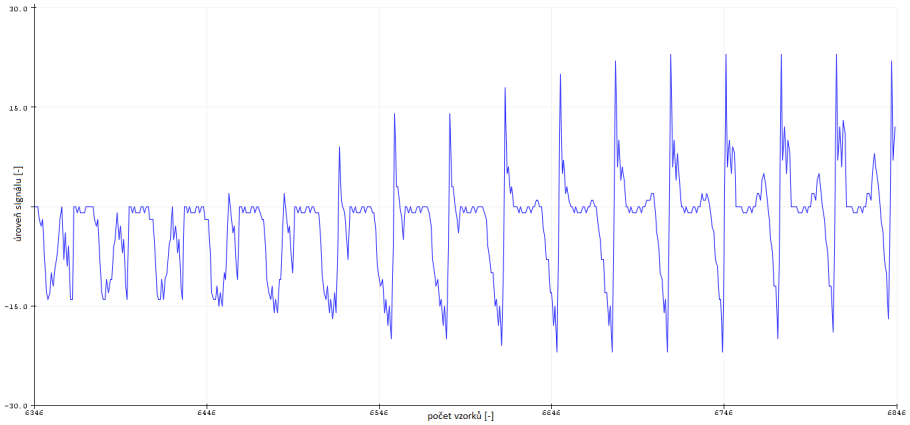
Jeden z potenciometrů v tomto režimu ovládá hlasitost, při které je vzorek přehráván. Druhý potenciometr reguluje výšku, při které je vzorek přehráván. Na obrázku 4.7 lze pozorovat vzorek přehrávaný v různých výškách.



Obrázek 4.7: Průběhy vzorků bubnu s různými výškami.

4.6 Režim 4: Syntéza vzorku

Poslední režim kombinuje metodologii režimu 2 s režimem 3. Umožňuje to, že vzorky mohou být syntetizovány pomocí různých modulačních parametrů. Protože samotný vzorek není periodický, nejlepší způsob, jak pozorovat účinky teto modulace, je držet piezo. Obrázek 4.8 ukazuje přehrávaný vzorek spolu s modulačními parametry. Když je vzorek přehráván vícekrát a vytváří kvaziperiodický signál, tento typ syntézy se nazývá *wavetable* syntéza [1]. To umožňuje produkovat velmi zajímavý zvuk a umožňuje modulaci jakéhokoli vzorku pomocí výše zmíněných technik.



Obrázek 4.8: Průběhy vzorků bubnu s různými parametry modulace zvuku.

Kapitola 5

Závěr

Tato práce popisuje teoretický základ syntézy zvuku. Pojednává také o návrhu a implementaci digitálního syntezátoru s využitím desky Arduino a knihovny Mozzi. Je popsáno fyzické zapojení obvodu a funkce každého prvku. Vstupní signály ze vstupních periferií se zpracují a vytvoří se syntetizovaný zvukový výstup.

Bylo vyřešeno mnoho překážek, aby bylo možné implementovat účinný syntezátor, který produkuje různé zvuky využívající techniky modulace zvuku. Optimalizace musela být provedena pro softwarové i hardwarové komponenty. Konečným výsledkem je funkční digitální zvukový syntezátor. Existuje mnoho dalších metod zvukové syntézy, které lze implementovat pomocí knihovny Mozzi. Paměť mikroprocesoru byla téměř vyčerpána, což znamená, že pro budoucí rozšíření takového syntezátoru je třeba použít výkonnější mikroprocesor.

literatura

1. RUSS, M. *Sound Synthesis and Sampling*. Taylor & Francis, 2012. Music technology series. ISBN 9781136122149. Dostupné také z: https://books.google.cz/books?id=X9h5AgAAQBAJ.
2. TIBBS, C.E.; JOHNSTONE, G.G. *Frequency Modulation Engineering*. Chapman a Hall, 1947. ISBN 9780608185798. Dostupné také z: https://books.google.cz/books?id=skYhAAAAMAAJ.
3. TAN, L.; JIANG, J. *Digital Signal Processing: Fundamentals and Applications*. Elsevier Science, 2013. ISBN 9780124159822. Dostupné také z: https://books.google.cz/books?id=M9-OhaJSwAEC.
4. SMITH, S.W. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Pub., 1999. ISBN 9780966017649. Dostupné také z: https://books.google.cz/books?id=ZtfaNwAACAAJ.
5. ARDUINO. *Arduino Nano User Manual V3.0* [online]. [B.r.]. [cit. 2023-06-23]. Dostupné z: https://robotics.ee.uwa.edu.au/nano/doc/nano-user-manual.pdf.
6. ARDUINO. *Arduino Nano User Manual* [online]. 2008. [cit. 2023-06-23]. Dostupné z: https://www.arduino.cc/en/uploads/Main/ArduinoNanoManual23.pdf.
7. BARRASS, T. Mozzi: Interactive Sound Synthesis on the Open Source Arduino Microprocessor. *Independent Artist*. 2013. Dostupné také z: https://rb.gy/xtwje.
8. STUART, A.; ORD, K. *Kendall's Advanced Theory of Statistics: Volume 1: Distribution Theory*. Wiley, 2009. Kendall's Advanced Theory of Statistics, č. v. 1; v. 1994. ISBN 9780340614303. Dostupné také z: https://books.google.cz/books?id=tW18thQWJQIC.
9. SUITS, B.H. *Physics Behind Music: An Introduction*. Cambridge University Press, 2023. ISBN 9781108956796. Dostupné také z: https://books.google.cz/books?id=EqC3EAAAQBAJ.

Příloha A

Seznam datových příloh

A.1 Kód pro program Arduino

Zdrojový kód hlavního programu:

Digital_synth_arduino.ino

Zdrojové soubory pro vzorky a noty:

pitches.h
Cymbal_Sample.h
Rim_Sample.h
Tom_Sample.h