Fakulta elektrotechnická
Katedra radioelektroniky

Bakalářská práce

Bicí automat

Marek Vavřínek

Studijní program:
Elektronika a komunikace

Vedoucí práce:
doc. Ing. Stanislav Vítek, Ph.D.

Praha, 2022

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Vavřínek** Jméno: **Marek** Osobní číslo: **492091**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra radioelektroniky**

Studijní program: **Elektronika a komunikace**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Bicí automat

Název bakalářské práce anglicky:

Drum Machine

Pokyny pro vypracování:

Navrhněte a implementujte hudební nástroj - bicí automat. Při vypracování se řiďte následujícími pokyny:
- využijte jednodeskový mikropočítač Raspberry PI
- navrhněte vhodné uživatelské rozhraní
- prostudujte komunikační rozhraní hudebních nástrojů a vybrané rozhraní implementujte

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] FORRÓ, Daniel. Midi. Komunikace v hudbě. Grada, 1993.

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: **22.01.2022** Termín odevzdání bakalářské práce: _____

Platnost zadání bakalářské práce: **30.09.2023**

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

© ČVUT v Praze, Design: ČVUT v Praze, VIC

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.

Praha 1. května 2022 Marek Vavřínek

Poděkování:

Rád bych poděkoval vedoucímu práce doc. Ing. Stanislavu Vítkovi, Ph.D. za možnost zapůjčení potřebných zařízení a materiálů a hlavně za jeho nadšení do této práce.

Marek Vavřínek

Abstrakt

Tato práce popisuje realizaci bicího automatu. Bicí automat byl naprogramován ve vývojovém prostředí Processing. Do zařízení byl implementován 16 poziční sekvencer s 16 různými kanály obsahující 16 různých zvuků bicích nástrojů, dále byla implementována sériová komunikace přes rozhraní MIDI. Zařízení lze ovládat externí MIDI klávesnicí a lze ho využít i jako MIDI kontrolér k počítači, nebo k jinému zařízení podporující standard MIDI. Výsledné zařízení je zhotoveno pomocí mikropočítače Raspberry Pi 4 B s dotykovou obrazovkou zasazeného do těla vytvořeného pomocí 3D tisku.

Klíčová slova:

Bicí automat, sekvencer, Raspberry Pi, Processing, MIDI, 3D tisk

Abstract

This work describes the implementation of a drum machine. The drum machine was programmed in the Processing development environment. A 16 position sequencer with 16 different channels containing 16 different drum sounds was implemented into the device, further a serial communication has been implemented into it via MIDI interface. The device can be controlled by an external MIDI keyboard and can also be used as a MIDI controller for a computer, or for another device that supports the MIDI standard. The resulting device is made using a Raspberry Pi 4 B microcomputer with a touch screen embedded into an original body created using 3D printing.

Keywords:

Drum machine, sequencer, Raspberry Pi, Processing, MIDI, 3D print

Obsah

Seznam obrázků **xi**

Seznam ukázek kódů **xii**

Seznam použitých zkratek **xiii**

1 Úvod **1**

2 Teoretická část **3**
2.1 Bicí automaty 3
2.2 Komunikační rozhraní hudebních nástrojů 4
2.2.1 MIDI 4
2.2.2 Popis standardu MIDI 4
2.3 Prostředí Processing 5

3 Návrh bicího automatu **7**
3.1 Hardware 7
3.2 Tělo zařízení 8
3.2.1 Návrh 3D modelu 8
3.2.2 3D tisk 9
3.3 Software 9
3.3.1 Operační systém Raspberry Pi 10
3.3.2 Vlastní program 10
3.3.3 Automatické spuštění programu 10

4 Program **11**
4.1 Nastavení při spuštění 11
4.2 Vykreslení uživatelského rozhraní 13
4.3 Ovládání tlačítek 14
4.4 Vytvoření MIDI sběrnice 15
4.5 Ovládání MIDI klávesami 16
4.6 Přehrávání zvuků a odesílání MIDI zpráv 17
4.7 Použité vzorky zvuků 18

5 Bicí automat **19**
5.1 Popis funkcí a jejich ovládání 19
5.2 Přednastavené rytmy 21

6 Závěr **23**

Literatura a použité zdroje **25**

Seznam obrázků

1 CHAMBERLIN Rhythmate (model 25) z roku 1949 (upraveno) [3] . . 1
2 Bicí automat: ROLAND TR-707 [5] 3
3 Diagram MIDI zprávy typu Note-On (upraveno) [8] 4
4 Návrh modelu krabičky v programu Autodesk Fusion 360 8
5 Vytisknutá krabička na 3D tiskárně 9
6 Diagram programu bicího automatu 12
7 Uživatelské rozhraní bicího automatu 13
8 Bicí automat s připojenými klávesami 19
9 Uživatelské rozhraní s popiskem 19
10 Uživatelské rozhraní v režimu kláves 20
11 Uživatelské rozhraní v režimu uložení/výběru uloženého rytmu 21

Seznam ukázek kódů

1 Zadaný příkaz do terminálu pro vytvoření souboru .desktop 10
2 Obsah souboru machine.desktop pro automatické spuštění 10
3 Zkrácený příklad funkce setup() 11
4 Příklad vykreslení barevných tlačítek a ukazatele 14
5 Příklad vyhodnocení dvou tlačítek 15
6 Vytvoření a obnovení MIDI komunikace 16
7 Zpracování příchozích MIDI zpráv 17
8 Přehrávání zvuků a odesílání MIDI dat 18

Seznam použitých zkratek

BPM Beats Per Minute, tempo hudby

CV Control Voltage

DCB Digital Communication Bus, Digitální komunikační sběrnice

DIN Deutsches Institut für Normung, Německý ústav pro průmyslovou normalizaci

FPS Frames Per Second, snímková frekvence

HSB Hue Saturation Brightness, barevný model Odstín sytost jas

LAN Local Area Network, lokální síť

LED Light Emitting Diode, světlo emitující dioda

MIDI Musical Instruments Digital Interface, digitální rozhraní hudebních nástrojů

MSB Most Significant Bit, bit s největší váhou

PEI polyetherimid, amorfní termoplast

PLA polylaktid neboli kyselina polymléčná

RGB Red Green Blue, barevný model Červená zelená modrá

SD Secure Digital, záznamové médium

USB Universal Serial Bus, Univerzální sériová sběrnice

1 Úvod

Hudba byla součásti života už od pradávna. Lidé si zpívali, jiní hráli melodie na různé nástroje a někteří hráli rytmy na bicí nástroje. Dnes je skoro v každé kapele bubeník, který se stará o rytmus písničky. V případech, kdy bubeník z nějakého důvodu není přítomen, se musí improvizovat. Postupem času a s příchodem elektroniky se začaly objevovat bicí automaty, které byly schopny vydávat základní rytmus. Kapely pak byly například schopny zkoušet některé části písniček bez bubeníka. Bicí automaty však nejsou určeny jen pro kapely. Někteří jedinci jsou schopni na bicích automatech hrát sólo i hodiny v kuse.

Prvními bicími automaty byla elektronická zařízení, která vydávala charakteristický robotický zvuk. V roce 1949 Harry Chamberlin vytvořil bicí automat *Rhythmate*, který přehrával zvuky opravdových bicích nástrojů [1]. Obsahoval 14 magnetických pásků, které ve smyčkách přehrávaly nahrané rytmy. V osmdesátých letech mnoho firem začalo prodávat různé elektronické nástroje jako *Roland TR-707* nebo *Linn LM-1 Drum Computer*. Později s nárůstem výkonu osobních počítačů a s příchodem MIDI protokolu se začaly objevovat programy, které simulovaly bicí automaty na počítači a dokázaly vytvářet záznamy [2]. Dnes existují sofistikované počítačové programy, které obsahují mnoho různých hudebních sekvencerů, syntetizérů a zvukových filtrů inspirovaných fyzickými zařízeními. Dohromady tak tvoří plně virtuální studio, ve kterém je uživatel schopen složit celé písně na počítači.

Obrázek 1: CHAMBERLIN Rhythmate (model 25) z roku 1949 (upraveno) [3]

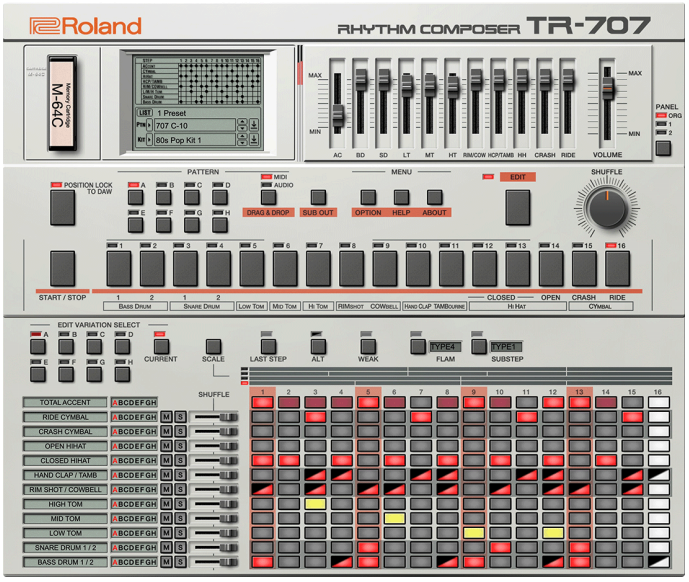
Cílem této práce je realizace hudebního sekvenceru, neboli bicího automatu, který bude v daném pořadí přehrávat zvukové vzorky bicích nástrojů. Výsledné zařízení bude sloužit při prezentaci školy, například při dni otevřených dveří, a k dalším potřebám nově vznikající skupiny na škole, zabývající se elektronickou hudbou. Bicí automat je koncipován tak, aby sám o sobě dokázal hrát nebo aby byl použit jako MIDI ovladač k jinému zařízení.

2 Teoretická část

2.1 Bicí automaty

Bicí automaty jsou elektronická zařízení využívající sekvencer, nebo systém opakovaní zvuků. Jedná se o zařízení, která jsou schopná přehrávat jednotlivé zvuky bicích nástrojů dle nastaveného vzoru uživatelem. Uživatel si nejprve nastaví rychlost přehrávání v úderech za minutu (BPM - beats per minute) a následně sestaví požadovaný rytmus označením v jakých dobách rytmu se má přehrát daný zvuk.

Prvními sekvencery byla analogová zařízení, později i digitální zařízení, která byla schopná podle nastaveného motivu vysílat řídící impulzy (*Trig*) do elektronických syntetizérů v opakující se smyčce [4]. Syntetizér při přijetí impulzu vygeneroval požadovaný zvuk. Složitější analogové sekvencery umožňovaly kromě vysílání řídících impulzů vysílat i sekvence hodnot napětí (*CV - Control Voltage*). Díky tomu bylo možné snadněji vytvářet opakující se smyčky s měnící se výškou tónu. Výšku tónu právě představoval signál *CV*.



Obrázek 2: Bicí automat: ROLAND TR-707 [5]

Analogové bicí automaty můžeme chápat jako spojení sekvencerů a syntetizérů v jedno kompaktní zařízení. Mnozí hudebníci našli zalíbení v těchto zařízeních a využívají je v elektronické hudbě [6]. Syntetizované zvuky nejsou věrohodné opravdovým bicím nástrojům. V některých případech je pravdou pravý opak. Elektronický někdy až praskavý robotický zvuk je pro ně charakteristický.

Digitální bicí automaty mohou generovat také syntetický zvuk. Příkladem mohou být 8 bitové syntetizéry, které generují zvuk se znatelným zkreslením, nebo vícebitové syntetizéry, které dokáží generovat věrohodný zvuk. Jejich výhoda oproti analogovým automatům však spočívá v schopnosti přehrávat nahrané vzorky zvuků z paměti. První digitální bicí automaty, které touto vlastností disponovaly, byly velmi drahé kvůli obsažené hardwarové paměti uchovávající nahrané vzorky [6]. Digitální bicí automaty jsou tímto způsobem schopné vydávat věrohodné zvuky bicích

nástrojů, a na rozdíl od bubeníka mohou hrát dlouho v kuse bez vyčerpání. Z těchto a z dalších důvodů si je oblíbili hudební producenti.

2.2 Komunikační rozhraní hudebních nástrojů

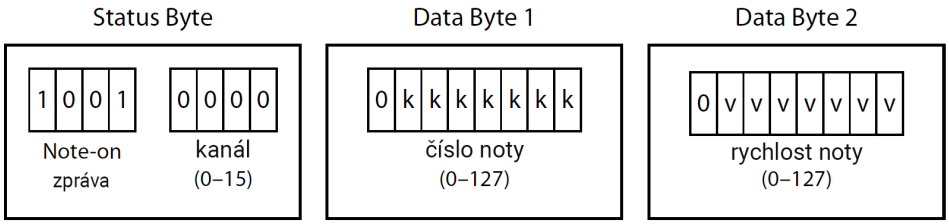
Od poloviny padesátých let dvacátého století začaly vznikat první pokusy o komunikaci mezi hudebními nástroji a mezi nástroji a počítači [7]. Od roku 1963 začaly vznikat první sekvencery, které pomocí *CV/Trig* signálů byly schopny ovládat analogové syntetizéry. Ovšem pomocí těchto systému se daly vytvářet pouze jednoduché opakující se řady. Koncem šedesátých let se začala objevovat různá paralelní rozhraní, žádné se však nestalo standardem.

V osmdesátých letech dvacátého století, když vývoj elektroniky a mikroprocesorů postoupil, objevil se systém *DCB* od firmy *ROLAND*. Tento systém byl schopný přenášet informace o stisku a uvolnění kláves a některé další informace. Používala ho však jen zmíněná firma.

V roce 1983 bylo představeno *MIDI* rozhraní a v následujících letech vznikla i norma zaručující kompatibilitu mezi výrobci. Toto rozhraní se okamžitě stalo standardem a je stále nejrozšířenější v upravené *USB* formě dodnes, i přes návrhy modernějších rozhraní využívající například *LAN*.

2.2.1 MIDI

MIDI je digitální rozhraní hudebních nástrojů, respektive sériový jednosměrný komunikační protokol [7]. Pomocí *MIDI* lze propojovat elektronické hudební nástroje se sekvencery, syntetizéry, záznamovými zařízeními, počítači atd. U starších zařízení se často objevují konektory typu *DIN*. Rozlišují se tři varianty zmíněného konektoru: *MIDI In*, *MIDI Out* a *MIDI Thru*. V dnešní době se tento konektor nahrazuje konektorem *USB B*, nebo *mini USB*, zejména u MIDI kontrolérů.



Obrázek 3: Diagram MIDI zprávy typu Note-On (upraveno) [8]

2.2.2 Popis standardu MIDI

Klasické *MIDI* rozhraní, využívající *DIN* konektor, využívá asynchronní vysílání dat rychlostí 31 250 Baudů v proudové smyčce[1] [7]. Samotná *MIDI* komunikace je interpretována zprávami o několika bytech. Zprávy jsou většinou tvořeny jedním stavovým bytem a jedním nebo dvěma datovými byty. U speciálních povelů může

[1]Logická nula je vyjádřena hodnotou tekoucího proudu a logická jednička proudem nulovým.

být struktura rozdílná. V této práci se zaměřím pouze na nejjednodušší typy zpráv *MIDI* komunikace, protože pouze ty se využijí při realizování bicího automatu. Jimi jsou zprávy *Note-On* a *Note-Off*, neboli zprávy o stisknutí a uvolnění klávesy.

Zprávy *MIDI* komunikace začínají stavovým bytem, který na místě bitu s největší váhou (MSB - most significant bit) má hodnotu 1, tím se liší od datových bytů, které mají na místě *MSB* hodnotu 0. Stavový byte tak dosahuje hodnot 128 - 255, zatímco datové byty obsahují hodnoty 0 - 127. Stavový byte nese informaci o čísle kanálu a typu funkce datových bytů. Standard *MIDI* obsahuje pouze 16 různých kanálů a 7 typů funkcí (nota zapnuta, nota vypnuta a další). Datové byty bývají zpravidla dva. První datový byte nese informaci o notě (například výšce tónu nebo číslo uloženého zvuku) a druhý datový byte nese informaci o rychlosti stlačení klávesy, viz obrázek 3. V jednom kanále tak lze přenášet 127 not, což je více než dostačující.

2.3 Prostředí Processing

Processing[2] je programovací jazyk a vývojové prostředí, které je open source, tedy zdarma k použití. Původně bylo vyvinuto pro výuku základů programování formou vizuálního kontextu [9]. Časem se *Processing* vyvinul do podoby vývojového nástroje pro profesionály. Ačkoliv je *Processing* koncipován hlavně pro programování vizuálních děl, díky tomu, že je open source, existuje velké množství knihoven, které rozšiřují jeho možnosti. Pomocí externích knihoven lze v *Processingu* naprogramovat nejen vykreslování vizuálních obrazů, ale i zpracování zvuku, ovládání robota, nebo webové aplikace.

V této práci je *Processing* zodpovědný, jak za vykreslování uživatelského rozhraní, tak za obstarávání zvukového média bicího automatu.

[2] https://processing.org/

3 Návrh bicího automatu

Návrh bicího automatu začal určením možných funkcí zařízení a velikostí sekvenceru. Rozhodl jsem se pro realizování 16 pozičního sekvenceru. Z důvodu jednoduchosti ovládání tlačítka musí ovládat více funkcí. Například při stisknutí pomocného tlačítka *SHIFT* se změní funkce hlavních ovládacích tlačítek. Tlačítka ovládající sekvencer pak mohou ovládat například i volbu nástroje. Realizováním 16 pozičního sekvenceru bude tedy možné pohodlně přepínat mezi 16 různými nástroji. Aby bicí automat opravdu dokázal simulovat rytmus bubeníka hrajícího na bicí, rozhodl jsem se, že zařízení bude přehrávat nahrané vzorky zvuků z paměti zařízení.

Protože nahrané vzorky zvuků mají relativně velký datový objem, byla při návrhu zavržena možnost použití mikrokontroléru typu *Arduino* z důvodu nedostatečné paměti. Jako ovládací jádro zařízení byl proto vybrán mikropočítač *Raspberry Pi 4 B*. Ten kromě dostatečně velké paměti má i dostatečně velký výkon na ovládání zařízení s barevnou dotykovou obrazovkou. Protože *Raspberry Pi* je plnohodnotný počítač, má i zvukový výstup, čímž se velmi zjednodušuje realizace celého projektu.

Posledním krokem při návrhu realizace bylo určení programovacího jazyka a vývojového prostředí. Opět z důvodu využití plnohodnotného počítače volba směřovala směrem k vyššímu programovacímu jazyku. S ohledem na programování grafického rozhraní a velký počet rozšiřujících knihoven bylo vybráno vývojové prostředí a programovací jazyk *Processing*, viz sekce 2.3.

3.1 Hardware

Vzhledem k tomu, že *Raspberry Pi* má dostatečně velký výkon a dokáže bezproblémově používat dotykovou obrazovku, rozhodl jsem se, že právě zmíněná obrazovka bude hlavním rozhraním bicího automatu. Na obrazovce bude zobrazena informace o nastaveném tempu (BPM), aktuální době v rytmu, vytvořených rytmech jednotlivých nástrojů (v jednotlivých kanálech) a ztišení daného kanálu. Jednoduchým upravením softwaru lze přidat další užitečné informace či ovládání. Odpadá tak nutnost signalizace pomocí světlo emitujících diod (LED) a s tím realizace ovládacích obvodů, čímž se zjednodušuje složitost hardwaru. Na obrazovce bude též vidět časová signalizace, právě aktivní funkce i stav stisknutých tlačítek.

Stejně jako desktopové počítače má *Raspbery Pi* USB rozhraní. Toto USB rozhraní lze při realizování bicího automatu využít pro sériovou komunikaci se standardem MIDI, viz sekce 2.2.1. Místo vytváření vstupních a výstupních obvodů pro MIDI rozhraní, lze použít USB na MIDI převodník, který se jednoduše zapojí do USB portu. Samotné programování MIDI komunikace tak bude také jednodušší.

Dotyková obrazovka sice umožňuje plné ovládání zařízení využitím dotykového rozhraní, to ovšem není zcela pohodlné pro všechny funkce a zvláště to není pohodlné při živém hraní. Z tohoto důvodu bude možné zařízení ovládat pomocí externí MIDI klávesnice, kterou bude možné připojit do zmíněného USB portu. Realizace softwaru nebude o tolik náročnější a zároveň nebude třeba vytvářet obvody pro fyzická tlačítka. Další výhodou je, že použité externí klávesy budou mnohem kvalitnější než jakákoliv běžná tlačítka. Výsledné zařízení bude tak i kompaktnější a levnější na výrobu.

Raspberry Pi má vlastní zvukový výstup ve formě 3,5 mm stereo jacku. Při navrhování a sestavování hardwaru tedy není třeba přidávat zvukový převodník a vytvářet sériovou komunikaci mezi *Raspberry Pi* a převodníkem.

Pro realizování celého výsledného bicího automatu bude použito pouze *Raspberry Pi 4 B* a dotyková obrazovka *Raspberry Pi 7"Touchscreen Display*[3].

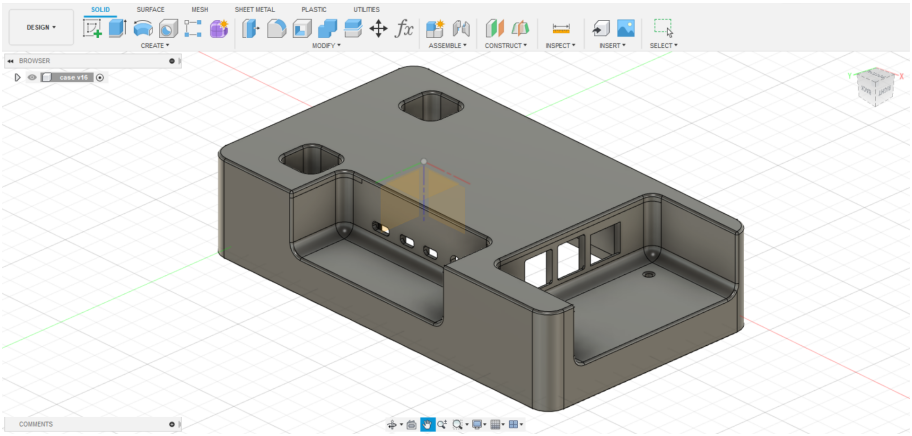
3.2 Tělo zařízení

Mikropočítač *Raspberry Pi*, který je základem bicího automatu v této práci, se prodává bez krabičky, či jiného těla, které by ho chránilo. Pokud by se nějaká krabička k *Raspberry Pi* dodávala, stejně by v této práci nemohla být použita, protože k mikropočítači se připojí dotyková obrazovka. Ta se uchycuje čtyřmi šrouby přímo k desce *Raspberry Pi*.

Aby výsledné zařízení bylo kompaktní a dalo se snadno přenášet chráněné, bude potřeba vyrobit vhodnou krabičku pro uložení elektroniky. K tomuto účelu je velmi efektivní navrhnout potřebné tělo zařízení jako digitální 3D model na počítači a následně ho vytisknout pomocí 3D tiskárny.

3.2.1 Návrh 3D modelu

K navržení digitálního 3D modelu těla zařízení byl použit program *Autodesk Fusion 360*. Tento program je vhodným nástrojem pro vytváření 3D modelů určených k 3D tisku. Pro studenty a pro jedince, kteří jej hodlají užít pro vlastní potřebu, je navíc zdarma[4].



Obrázek 4: Návrh modelu krabičky v programu Autodesk Fusion 360

Model těla zařízení byl navržen tak, aby se skládal pouze z jedné části, kterou lze vytisknout v kuse. Protože *Raspberry Pi* s dotykovou obrazovkou je všestranné zařízení, které lze použít i v jiných projektech, byla krabička navržena tak, aby byl zachován snadný přístup ke všem portům. Díky tomu krabička je vhodná nejen pro bicí automat, ale i pro další budoucí projekty využívající *Raspberry Pi* a 7"dotykovou obrazovku.

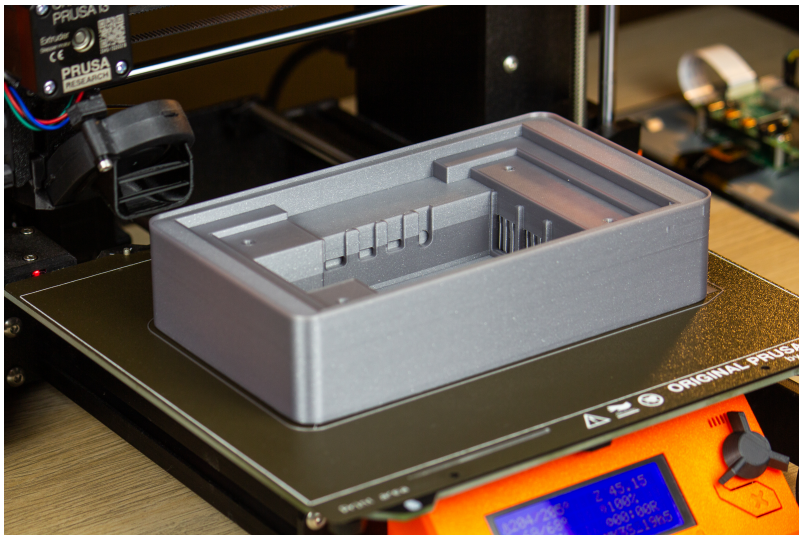
[3] https://www.raspberrypi.org/
[4] https://www.autodesk.com/products/fusion-360/free-trial

Na obrázku 4 lze vidět model krabičky, který je zobrazen obrazovkou dolů, kde se nachází větší otvor, kterým se *Raspberry Pi* již s připevněnou dotykovou obrazovkou zasune dovnitř. Uvnitř se nachází konstrukce, která podpírá jak obrazovku, tak samotný mikropočítač, aby byla při manipulaci se zařízením zajištěna maximální stabilita komponentů. Uchycení krabičky k obrazovce se provede čtyřmi šrouby M3 se zapuštěnou hlavičkou. Dva šrouby 20 mm dlouhé se umístí ve čtvercových otvorech a dva šrouby 10 mm dlouhé se umístí v otvorech před USB porty.

3.2.2 3D tisk

Krabička pro bicí automat byla vytisknutá na 3D tiskárně *Original Prusa i3 MK3S* na tiskovém plátu se zrnitým práškovým PEI povrchem. Jako tiskový materiál byl použit filament *Prusament PLA Galaxy Silver*. Model byl tisknut otvorem pro obrazovku směrem nahoru s použitím tiskových podpěr na tiskové ploše. Tisk trval přibližně 20 hodin. Použité parametry tiskárny byly následující: výška vrstvy 0,15 mm, teplota trysky 205 °C, teplota podložky 68 °C, hustota výplně 20 %, počet svislých perimetrů: 3, vzor výplně: kubický adaptivní, vzor spodní a horní stěny: monotónní.

Na obrázku 5 lze vidět dokončený tisk krabičky, který proběhl bez problémů. Před tiskem samotné krabičky předcházel testovací tisk dvou částí modelu pro vyzkoušení, zda všechny otvory pro konektory a šrouby sedí na správných místech. Po vyzkoušení testovacích modelů proběhla finální úprava daných rozměrů.



Obrázek 5: Vytisknutá krabička na 3D tiskárně

3.3 Software

Na rozdíl od *Arduina* či jiných mikrokontrolérů se do *Raspberry Pi* nenahrává program ve formátu `.hex` souboru. Software běžící na *Raspberry Pi* je podobný softwaru na běžném počítači. Na *Raspberry Pi* je třeba nainstalovat operační systém, potom daný program, a pokud je to vhodné, lze nastavit, aby se daný program spouštěl automaticky po zapnutí zařízení. Všechny tyto kroky jsou v této práci dále popsány a využity k realizování bicího automatu.

3.3.1 Operační systém Raspberry Pi

Protože *Raspberry Pi* je mikropočítač, stejně jako běžný počítač spouští operační systém. V případě *Raspberry Pi* se jedná o upravenou verzi linuxu - *Raspbian*[5]. Tento systém při zakoupení nového *Raspberry Pi* není nainstalovaný a není ani v obsahu balení. Stejně tak není součástí microSD karta (menší varianta Secure Digital karty), na kterou je potřeba operační systém nainstalovat.

Raspbian se musí stáhnout do jiného počítače než je instalované *Raspberry Pi*. Prostřednictvím programu *Raspberry Pi Imager*[5], nebo *Balena Etcher*[6] se nainstaluje na microSD kartu. Po dokončení instalace je možné kartu vložit do mikropočítače a poprvé ho zapnout.

3.3.2 Vlastní program

K realizování vlastní softwarové části bicího automatu bylo použito vývojové prostředí *Processing*, viz sekce 2.3. Při programování bicího automatu byly použity dvě externí knihovny. První a nejdůležitější byla knihovna *Minim* [10], která se stará o zpracování zvuku. V této knihovně jsou implementovány funkce, které správně přistupují ke zvukovému médiu a umí s ním zacházet. Lze tak velmi jednoduše napsat program, který bude v daných intervalech přehrávat zvuky z paměti, nebo je bude dokonce generovat. Druhou použitou knihovnou byla knihovna *TheMidiBus* [11], která poskytuje rychlý a jednoduchý způsob posílání a přijímání MIDI zpráv.

Vlastní program byl napsaný přímo ve vývojovém prostředí *Processing* na běžném počítači, ačkoliv by mohl být napsán přímo na *Raspberry Pi*, což by nemuselo být příliš pohodlné. Po dokončení programu byl program přesunut do *Raspberry Pi*, což je stejně jednoduchá operace jako přesunutí souboru pomocí flash disku z jedno počítače do druhého. Více o vlastním programu v sekci 4.

3.3.3 Automatické spuštění programu

Posledním krokem realizování softwarové části bicího automatu bylo nastavení automatického spouštění vlastního programu po zapnutí *Raspberry Pi*. Toho bylo docíleno následujícím postupem [12]. Do terminálu byl zadán příkaz (viz ukázka kódu 1), čímž se vytvořil soubor machine s příponou .desktop.

```
1  sudo nano /etc/xdg/autostart/machine.desktop
```

Ukázka kódu 1: Zadaný příkaz do terminálu pro vytvoření souboru .desktop

Do vytvořeného souboru se napsal kód dle ukázky kódu 2, který zajišťuje automatické spuštění programu bicího automatu po bootu mikropočítače. Uložení kódu se provedlo pomocí klávesové zkratky `ctrl + O`.

```
1  [Desktop Entry]
2  Name=DrumMachine
3  Exec=/home/pi/drummachine/application.linux-armv6hf/drummachine
```

Ukázka kódu 2: Obsah souboru machine.desktop pro automatické spuštění

[5] https://www.raspberrypi.com/software/
[6] https://www.balena.io/etcher/

4 Program

Psaní programu v *Processingu* je velmi podobné programovacímu jazyku C++ nebo Java s tím rozdílem, že zde není jedna hlavní funkce `main()`. Místo ní jsou tu dvě hlavní funkce, které je třeba při psaní programu nadeklarovat. První funkcí je funkce `setup()`, která je zavolána při spuštění programu a je zodpovědná za prvotní nastavení. Druhou funkcí je funkce `draw()`, která je po spuštění programu volána opakovaně obnovovací frekvencí obrazu a primárně je určena pro vykreslování grafiky. Lze však do ní napsat cokoliv, co se má provádět opakovaně.

Dále je možné definovat další funkce jako jsou `mousePressed()`, nebo `keyPressed()`, nebo vlastní funkce. Tyto funkce již nejsou povinné, program půjde spustit i bez nich.

4.1 Nastavení při spuštění

Při spuštění programu je automaticky zavolána funkce `setup()`. Akce, které jsou v ní napsané, se provedou jako první a pouze jednou. Toto je vhodné místo, kam napsat prvotní nastavení programu obsluhujícího bicí automat.

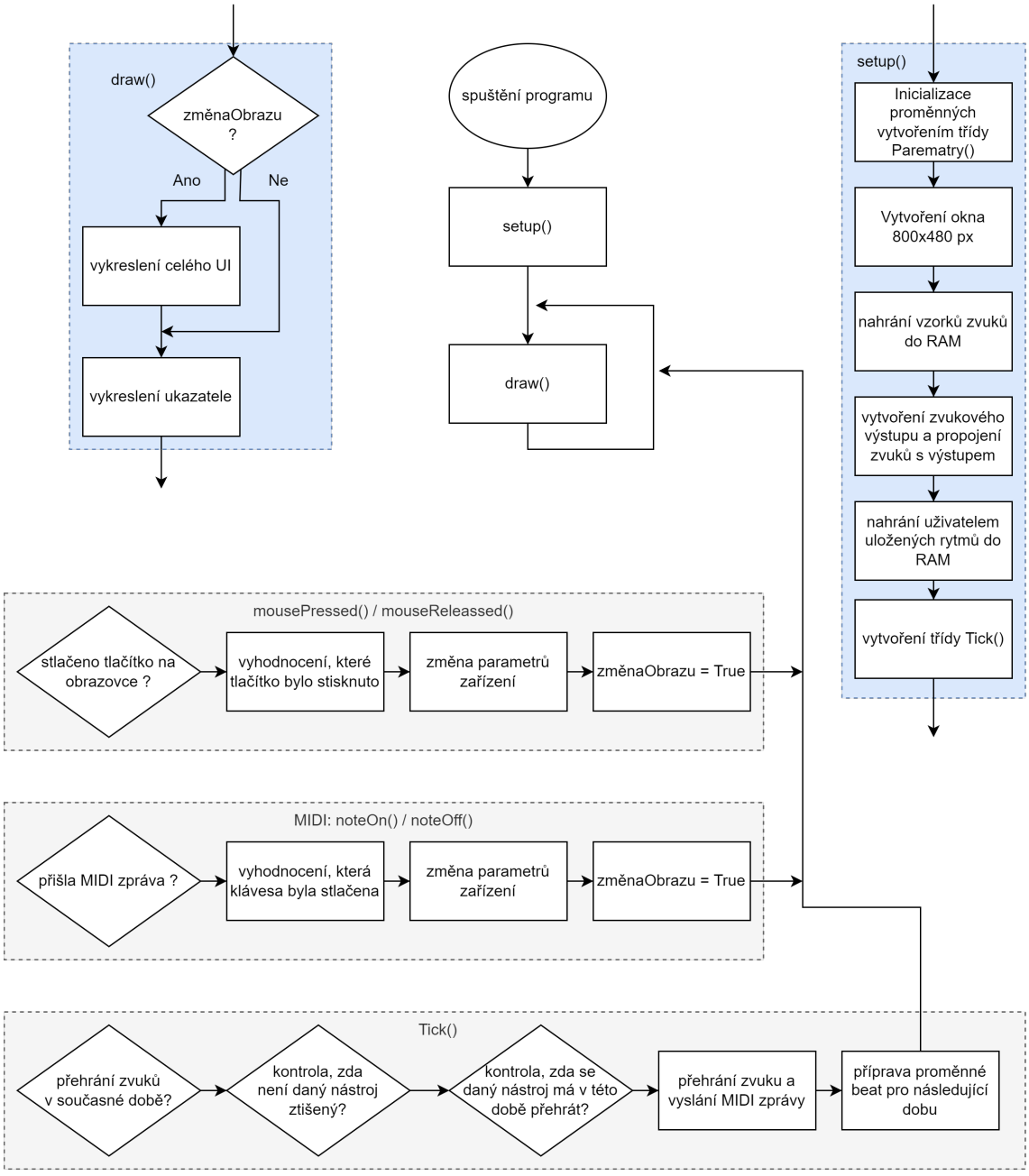
V ukázce kódu 3 lze vidět nejdůležitější část nastavení bicího automatu. Hned první použitou funkcí je funkce `size()`, která nastaví rozlišení vykreslovaného okna. Jejím parametrem je rozlišení 800×480 px, což je rozlišení použité obrazovky. Program tedy bude používat celou obrazovku pro vykreslování rozhraní.

Další funkce pocházejí z výše zmíněné knihovny *Minim* [10]. Nastavují důležité parametry bicího automatu jako zvukový výstup, zvukové soubory a jejich propojení s výstupem.

Následuje vytvoření třídy `Parametry()` a třídy `Tick()`. První třída obsahuje všechny důležité parametry bicího automatu jako je tempo (BPM), aktuální pozice v sekvenceru, vybraný nástroj, uložené vzory rytmů atd. Druhá třída je zodpovědná za chod sekvenceru. Více o třídě `Tick()` v sekci 4.6.

```
1  void setup(){
2      size(800,480);                                  // velikost okna
3      // nastaveni knihovny Minim
4      minim = new Minim(this);                        // knihovna zvuku
5      out = minim.getLineOut();                       // zvukovy vystup
6      zvuk[0]  = new Sampler( "BD.wav", 4, minim );// ulozeni zvukovych
7      zvuk[1] = new Sampler( "SD.wav", 4, minim ); // souboru do pameti
8      // a tak dale az ke zvuku 16, resp. 15
9      for (int i = 8; i<16 ; i++)                     // propojeni zvuku
10         zvuk[i].patch( out );                       // s vystupem
11
12     synth = new Parametry();                        // trida s parametry
13     synth.BPM = 145;                                // aktualni BPM
14     tik = new Tick();                               // trida sekvenceru
15 }
```

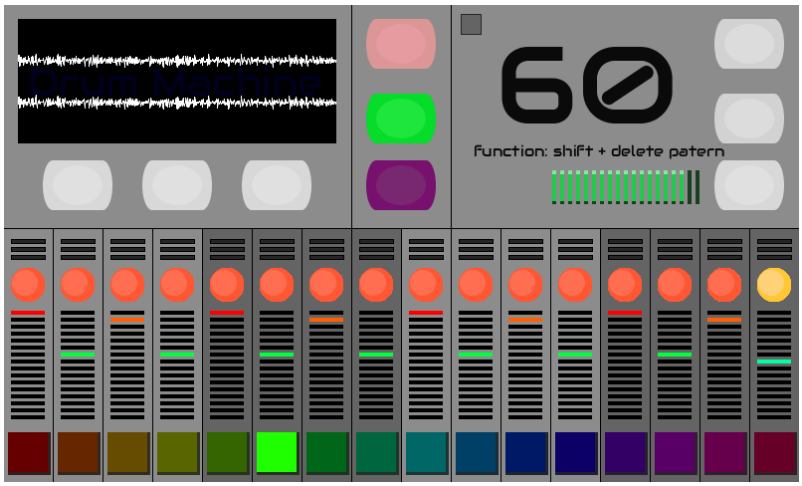
Ukázka kódu 3: Zkrácený příklad funkce setup()



Obrázek 6: Diagram programu bicího automatu

4.2 Vykreslení uživatelského rozhraní

Uživatelské rozhraní bicího automatu obsahuje veškeré potřebné ovládání zařízení. V pravé horní části je nastavení tempa (BPM) spolu s popiskem aktuálně vybrané funkce a nastavení hlasitosti. V prostřední horní části jsou tlačítka PLAY, STOP a SHIFT[7]. V levé horní části je vykreslení časového průběhu výstupního signálu spolu s třemi tlačítky pro volbu funkce, viz sekce 5.1. Ve spodní části je ukazatel právě přehrávané doby (beat) spolu s tlačítky pro nastavení rytmu a s tlačítky pro ztišení daného kanálu. Uprostřed dolní části je matice barevných čar zobrazující nastavené rytmy všech 16 kanálů.



Obrázek 7: Uživatelské rozhraní bicího automatu

Vykreslování grafiky v *Processingu* je velmi jednoduché. Funkce `draw()` je zavolána programem pro vykreslení každého snímku. Tedy pokud je obnovovací frekvence obrazu 30 FPS, pak je funkce `draw()` zavolána 30 krát za vteřinu. Při každém zavolání se provedou všechny funkce, které jsou v ní napsané.

Jednoduchými funkcemi jako je `rect()`, `ellipse()`, nebo `text()` lze postupně vykreslit hezké uživatelské rozhraní. Díky pomocným proměnným a funkci `if()` lze vykreslovat různou grafiku nebo měnit barvy dle nějaké podmínky. Funkce `fill()` nastavuje barvu vykreslování následujících objektů ve formátu RGB.

Vykreslování složitější grafiky celého uživatelského rozhraní každou třicetinu vteřiny může být ve výsledku docela náročná akce. Z tohoto důvodu je vhodné napsat program tak, aby každým snímkem obrazu nemuselo být celé rozhraní znovu vykresleno. Například pomocí vhodné proměnné typu boolean `zmenaObrazu` zjišťovat, zda došlo ke změně, kterou je potřeba vykreslit. Pokud je odpověď ne, může být ve funkci `draw()` přeskočena značná část kódu.

Kód může být i výrazně zkrácen vhodným zvolením způsobu vykreslování barev. Příkladem je ukázka kódu 4, ve které se vykresluje ukazatel právě přehrávaného vzorku a 16 barevných tlačítek. Barva ukazatele je dána RGB notací. Pokud bychom chtěli nastavovat barvy tlačítek tak jako v obrázku 7 pomocí RGB notace, bylo by to velmi nepřehledné a kód by byl velmi dlouhý. Pro nastavení každé barvy jednotlivého tlačítka by se musela zavolat funkce `fill()` s jinými parametry. Daleko výhodnější

[7]Tlačítko pro změnu nastavovaného zvukového kanálu

postup je přepnout barevný režim do HSB režimu s pouze 16 odstíny barev. Toho je docíleno pomocí funkce `colorMode()`. Při vykreslování tlačítek cyklem `for()` pak lze jednoduše nastavovat barvu funkce `fill()` s parametrem `i`. Barvy tlačítek se pak vykreslí ve stejném pořadí jako duha.

```
1  void draw(){
2      // vykresleni ukazatele prehravaneho vzorku kazdy snimek
3      colorMode(RGB, 255, 255, 255);        // zmena barevneho prostoru na RGB
4      for(int i = 0 ; i<16 ; i++){
5          if(beat == i) fill( 255, 200, 51);      // barva pro aktualni vzorek
6          else fill( 255, 87, 51);                // barva pro ostatni vzorky
7          ellipse(25+i*DispWidth/16, DispHight*3/5-6, 34, 34);   // vykresleni
8      }
9      // vykresleni ovladacich tlacitek, pokud dojde ke zmene
10     if(zmenaObrazu){
11         colorMode(HSB, 16, 100, 100);     // zmena barevneho prostoru na HSB
12         for(int i = 0 ; i<16 ; i++){
13             if(nastroj[i]) fill( i, 100, 100);    // vybrany nastroj
14             else fill( i, 100, 40);                      // ostatni nastroje
15             rect(5+i*DispWidth/16, DispWidth*15/16, 40, 40);// vykresleni
16         }
17     }
18 }
```

Ukázka kódu 4: Příklad vykreslení barevných tlačítek a ukazatele

Obdobným způsobem lze vykreslit všechny prvky celého uživatelského rozhraní. Při psaní části programu zodpovědné za vykreslení grafiky jsem vycházel přímo z dokumentace *Processingu* [13], kde je vše detailně popsáno a ukázáno na příkladech.

4.3 Ovládání tlačítek

V předchozí kapitole bylo popsáno, jak postupně vykreslit uživatelské rozhraní včetně požadovaných tlačítek. Ovšem bez části programu, která je schopná vyhodnotit stisknutí myši, jsou tlačítka pouze barevnými čtverečky na obrazovce.

V *Processingu* vyhodnocení stlačení tlačítka myši, respektive stlačení dotykové obrazovky, se provádí pomocí funkce `mousePressed()`. Tu je třeba nadeklarovat stejně jako funkci `draw()`, nebo funkci `setup()`. Po spuštění programu je pak tato funkce zavolána pokaždé, když dojde ke stlačení levého tlačítka myši, respektive při stlačení dotykové obrazovky. Při zavolání funkce si program automaticky uloží souřadnice, ve kterých byla myš stisknuta, do proměnných `mouseX` a `mouseY`. V samotné funkci `mousePressed()` pak pomocí funkce `if()` je třeba vyhodnotit, zda souřadnice spadají do oblasti vykresleného tlačítka.

V ukázce kódu 5 lze vidět příklad vyhodnocení stisknutí dvou tlačítek, která zvyšují, či snižují tempo (BPM). Například při stlačení tlačítka snižující BPM se zavolá funkce `mousePressed()`. Program nejprve vyhodnotí, že souřadnice stlačení myši nebyly v oblasti tlačítka, které zvyšuje BPM. Přeskočí tedy na vyhodnocení souřadnic v oblasti tlačítka, které snižuje BPM. Když leží souřadnice `mouseX` a `mouseY` v požadovaném intervalu, je tato podmínka splněna, a proto program postoupí dále, kde kontroluje, zda bicí automat je v režimu STOP a zda aktuální BPM je větší než 1. Pokud je i tato podmínka splněna, sníží se BPM.

```
1  void mousePressed(){
2      // tlacitko BPM +
3      if(mouseX > 15/16*DispWidth-35 && mouseX < *15/16*DispWidth+35 &&
4         mouseY > 15 && mouseY < 65){                //kontrola souradnic
5          if(!synth.synthPlay && synth.set_BPM < 399)//kontrola stavu sekvenceru
6              synth.set_BPM++;            //inkrementace BPM
7              synth.zmena_obrazu = true;  //aby se zmena vykreslila na obrazovce
8      }
9      // tlacitko BPM -
10     if(mouseX > 15/16*DispWidth-35 && mouseX < 15/16*DispWidth+35 &&
11        mouseY > 90 && mouseY < 140){              //kontrola souradnic
12         if(!synth.synthPlay && synth.set_BPM > 1) //kontrola stavu sekvenceru
13             synth.set_BPM--;            //dekrementace BPM
14             synth.zmena_obrazu = true;  //aby se zmena vykreslila na obrazovce
15     }
16 }
```

Ukázka kódu 5: Příklad vyhodnocení dvou tlačítek

Obdobným způsobem s využitím pomocných proměnných typu boolean uložených ve třídě `Parametry()` jsou vyhodnocována všechna tlačítka, která mají reagovat při stisknutí myši, respektive při stlačení dotykové obrazovky. Tlačítka, která mají reagovat i na uvolnění myši, respektive dotykové obrazovky, jsou vyhodnocována dvěma funkcemi - `mousePressed()` a `mouseReleased()`. Druhá zmíněná funkce funguje zcela stejným způsobem jako funkce `mousePressed()` s tím rozdílem, že je automaticky zavolána programem při puštění tlačítka myši, respektive dotykové obrazovky.

4.4 Vytvoření MIDI sběrnice

Bicí automat byl navržen tak, že může být ovládán MIDI klávesami a že může ovládat jiné zařízení přes MIDI. Aby bylo možné využívat každou funkci samostatně, nebo současně podle potřeby uživatele, bylo potřeba vytvořit dvě MIDI sběrnice, přičemž každá zajišťuje jednu ze zmíněných funkcí. Každá požadovaná funkce je jednosměrná, a proto každá sběrnice byla vytvořena také jednosměrná. Sběrnice pro ovládání bicího automatu klávesami byla vytvořená jako vstupní a sběrnice pro ovládání jiného zařízení bicím automatem jako výstupní. Při jiné aplikaci využívající MIDI standard by mohlo být vhodnější využít obousměrnou sběrnici. V této práci však byla tato možnost zavržena. O veškerou MIDI komunikaci se stará již zmíněná knihovna *TheMidiBus* [11].

V ukázce kódu 6 lze vidět program, který zajišťuje vytvoření a případné obnovení vstupní MIDI sběrnice pro přijímání zpráv od externích kláves *KORG nanoKEY2*. Uživatel vždy po připojení kláves musí stisknout tlačítko hned vedle tlačítka STOP (viz obrázek 7), které spustí tento program. Při prvním průchodu tímto programem jistě žádná MIDI sběrnice ještě není vytvořená, proto bude podmínka na řádku 2 přeskočena a bude platit podmínka na řádku 4. V této části programu se nejprve aktualizuje seznam připojených MIDI zařízení k *Raspberry Pi*, poté začne cyklus, ve kterém se postupně porovnají všechna dostupná vstupní MIDI rozhraní s hledanými klávesami *nanoKEY2*. Toto porovnání je zde třikrát se třemi různými názvy

zařízení. Důvodem je fakt, že v systému Windows zůstane MIDI zařízení pojmenované svým původním jménem, zatímco v systému Linux je za název zařízení připojen řetězec s dalšími informacemi, jako je například pořadí připojení MIDI zařízení. Tento program je tedy možné spustit jak na *Raspberry Pi* tak na běžném PC. Pokud se při porovnávání názvy shodují, znamená to, že připojené zařízení jsou klávesy *KORG nanoKEY2* a vytvoří se vstupní MIDI sběrnice pomocí třídy `MidiBus()`, kde první řetězec značí název vstupního zařízení a parametr -1 znamená žádné výstupní zařízení.

```
1  // uzavre predchozi MIDI sbernici, pri 1. pruchodu je vynechano
2  if(!MIDIinUnavaible) {myBusIn.close(); MIDIinUnavaible = true;}
3  // vytvoreni nove MIDI sbernice, hledani zarizeni ze 3 moznych oznaceni
4  if(MIDIinUnavaible){
5      MidiBus.findMidiDevices();  //aktualizuje seznam pripojenych MIDI zarizeni
6      for (int i = 0; i < available_inputs.length; i++){
7          // pro Raspi - nova MIDI sbernice pro vstup biciho automatu
8          if(available_inputs[i].equals("nanoKEY2 [hw:1,0,0]") == true){
9            myBusIn = new MidiBus(this, "nanoKEY2 [hw:1,0,0]", -1);
10           MIDIinUnavaible = false;
11         }
12         // pro Raspi - nova MIDI sbernice pro vstup biciho automatu
13         if(available_inputs[i].equals("nanoKEY2 [hw:2,0,0]") == true){
14           myBusIn = new MidiBus(this, "nanoKEY2 [hw:2,0,0]", -1);
15           MIDIinUnavaible = false;
16         }
17         // pro win - nova MIDI sbernice pro vstup biciho automatu
18         if(available_inputs[i].equals("nanoKEY2") == true){
19           myBusIn = new MidiBus(this, "nanoKEY2", -1);
20           MIDIinUnavaible = false;
21         }
22     }
23 }
```

Ukázka kódu 6: Vytvoření a obnovení MIDI komunikace

Pro obnovení sběrnice uživatel znovu stiskne stejné tlačítko. Udělá tak například při neočekávaném vypadnutí komunikace, nebo pokud odpojí a znovu zapojí MIDI klávesy. V tomto případě již MIDI sběrnice existuje a je nutné ji před vytvořením nové nejprve uzavřít. Z tohoto důvodu je v programu podmínka `if()` na řádku 2.

Podobný kód obsluhuje vytvoření a obnovení výstupní MIDI sběrnice. Rozdílem je pouze nahrazení výrazů `in` a `inputs` výrazy `out` a `outputs`. Nutné je také napsat vhodný název jména zařízení při vytváření sběrnice. V případě bicího automatu je jako výstupní zařízení používán externí převodník *USB MIDI Interface*. Navázání komunikace by bylo možné v případě řádku 19 nahrazením následujícím řádkem: `myBusOut = new MidiBus(this, -1, "USB MIDI Interface ");`

4.5 Ovládání MIDI klávesami

Pokud je úspěšně vytvořená vstupní MIDI sběrnice, je možné přijímat MIDI zprávy od kláves a ovládat tak bicí automat. Vytvoření MIDI sběrnice je popsáno v sekci 4.4. O příjem MIDI zpráv se starají funkce `noteOn()` a `noteOff()`, které je třeba

nadeklarovat stejně jako například funkce `draw()` nebo `mousePressed()`. Pokud systém zachytí MIDI zprávu, program automaticky zavolá příslušnou funkci `noteOn()`, nebo `noteOff()`. Pokud funkce není deklarovaná, nic se nestane. V těle příslušné funkce je možné napsat libovolný kód, který se má vykonat. V této práci jsou použity externí MIDI klávesy na ovládání bicího automatu místo nutnosti ovládání skrze dotykové rozhraní. Ovládání je proto realizováno tak, že každá klávesa má specifickou funkci a informace o čísle noty, respektive výšce tónu, je použita pouze pro identifikaci dané klávesy. Další informace v MIDI zprávě, jako jsou číslo kanálu a rychlost noty, nejsou využité.

V ukázce kódu 7 lze vidět příklad funkce `noteOn()`, která při stlačení klávesy s číslem noty 49 provede stejnou akci jako stlačení tlačítka SHIFT na dotykové obrazovce. Pokud by v této ukázce byla stlačena jiná klávesa než klávesa s číslem noty 49, nic by se nestalo.

Při realizování ovládání bicího automatu externími klávesami se postupovalo obdobným způsobem jako v ukázce kódu 7 a pro každou funkční klávesu se udělala podobná podmínka, ale s vyhodnocením jiného čísla noty. Použita byla i funkce `noteOff()`, která funguje podobně jako funkce `noteOn()` s tím rozdílem, že je automaticky zavolána po přijetí MIDI zprávy o uvolnění klávesy.

```
1  // prijem zprav typu NoteOn
2  void noteOn(int channel, int pitch, int velocity) {
3      if(pitch == 49){                // aktivace shift
4        synth.shifted = !synth.shifted; // stisknutim se zmeni shift
5        synth.klavesy = false;          // ostatni funkce musi byt vypnuty
6        synth.mutovani = false;
7        synth.samply = false;
8        synth.paterns = false;
9        synth.zmena_obrazu = true;      // aby se zmena vykreslila na obrazovce
10     }
11 }
```

Ukázka kódu 7: Zpracování příchozích MIDI zpráv

4.6 Přehrávání zvuků a odesílání MIDI zpráv

Přehrávání zvuků u bicího automatu vychází z příkladu v dokumentaci knihovny *Minim* [14]. O přehrání zvuku ve správný čas dle nastaveného tempa se stará třída `Tick`, která implementuje vlastnosti třídy `Instrument` deklarované knihovnou, viz ukázka kódu 8. Nastavením tempa a prvním zavoláním noty 0[8] při stusknutí tlačítka PLAY se spustí automatické zavolání funkcí `noteOn()` a `noteOff()` ve třídě `Tick`.

Nejprve je automaticky zavolána funkce `noteOn()`, ve které se kontroluje, zda daný zvuk není ztišený[9]. Pokud není, následuje kontrola matice, která obsahuje nastavení daného rytmu, zda v současné době[10] rytmu má být zvuk přehraný. Pokud ano, je zavolána funkce `trigger()`, čímž se daný zvuk přehraje, a zároveň se odešle funkcí `sendNoteOn()` MIDI informace *Note-On* v daném kanálu nástroje. Tento

[8] Nota, která při přehrávání nevydá žádný zvuk.
[9] Tím je realizována funkce MUTE bicího automatu
[10] Současnou dobu v rytmu představuje proměnná `beat`.

postup se cyklicky opakuje pro každý nástroj bicího automatu.

Po ukončení akce funkce `noteOn()` se automaticky zavolá funkce `noteOff()`. Jejím úkolem je připravení proměnné `beat` pro následující dobu rytmu, nastavení tempa, ve kterém se mají přehrát následující zvuky, a opětovné vyvolání celého procesu znovu zavoláním noty 0. Třída automatický hlídá, aby bylo dodrženo nastavené tempo tím, že počká správný čas, než opětovně zavolá funkci `noteOn()`.

```
1  class Tick implements Instrument{
2    void noteOn( float dur ){                // zavola se na zacatku prehravani
3       for(int i = 0; i < 16; i++)
4          if ( !synth.muted[i] )             // kontrola zda neni kanal ztiseny
5             if ( synth.matice[i][synth.beat]){//kontrola zda se ted ma prehrat
6                zvuk[i].trigger();           // prehrani zvuku
7                // odeslani Midi noteOn - kanal, cislo noty, rychlost
8                if(!MIDIoutUnable) myBusOut.sendNoteOn(i, MIDInote[11], 127);
9             }
10   }
11
12   void noteOff(){                          // zavola se na konci prehravani
13      beat = (synth.beat+1)%16;             // nastaveni nasledujiciho beatu
14      out.setTempo( synth.BPM );            // nastaveni tempa/BPM
15      out.playNote( 0, 0.25f, this );       // znovu zavolani sekvence
16   }
17 }
```

Ukázka kódu 8: Přehrávání zvuků a odesílání MIDI dat

Funkce pro příjem a odesílání MIDI dat implementuje externí knihovna *TheMidiBus*. V sekci 4.4 je popsáno vytvoření sběrnice. Samotné odesílání dat je velmi jednoduché. Přes vytvořenou MIDI sběrnici odešleme data ve formě not pomocí funkce `sendNoteOn()`, kde parametry jsou číslo kanálu, výška tónu a rychlost noty. Pro zjednodušení zápisu v ukázce kódu 8 je výška tónu brána z připraveného pole.

Protože bicí automat hraje zvuky pouze bicích nástrojů, důležitá je pouze informace, že byl nástroj zahrán, výška tónu tedy nemusí být správná. Není třeba ani odesílat informaci o ukončení noty pomocí funkce `sendNoteOff()`. V případě programování elektronických kláves by bylo potřeba tuto informaci odesílat.

4.7 Použité vzorky zvuků

V předchozích kapitolách byly popsány veškeré nutné kroky pro realizaci, jak hardwarové, tak softwarové části bicího automatu. Bicí automat v této práci je navržen tak, že přehrává nahrané vzorky zvuků bicích nástrojů. Žádný ze zvuků bicích nástrojů bicí automat sám negeneruje.

Pořadí zvuků bicích nástrojů je inspirováno bicím automatem *Korg Volca Beats Analog Rhythm Machine* [15]. Ten však neobsahuje 16 nástrojů, proto byly dodány další zvuky při zachování stejného logického pořadí. Použité vzorky zvuků pocházejí z internetových zdrojů a jsou volně ke stažení [16, 17].

5 Bicí automat

Bicí automat byl sestaven pomocí mikropočítače *Raspberry Pi 4 B* a dotykové obrazovky. Na obrázku 8 lze vidět dokončený bicí automat s připojenými klávesami a s ochranným víčkem. Dále v této kapitole budou popsány veškeré funkce a jejich ovládání.

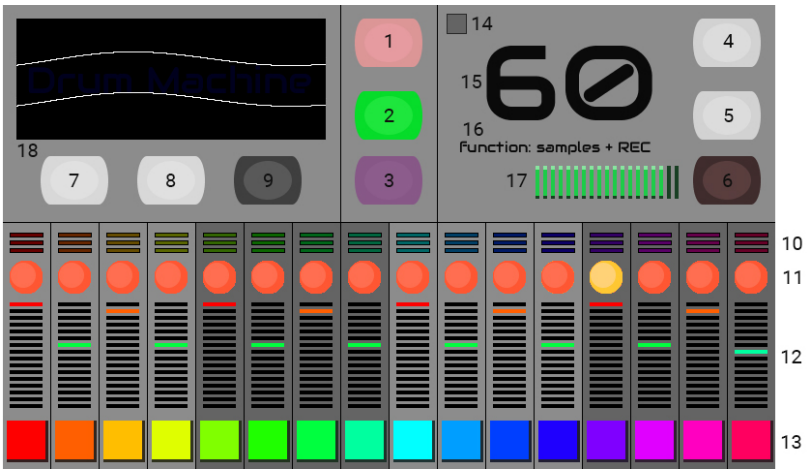
Obrázek 8: Bicí automat s připojenými klávesami

5.1 Popis funkcí a jejich ovládání

Následující popis funkcí a ovládání bicího automatu se vztahuje k obrázku 9.

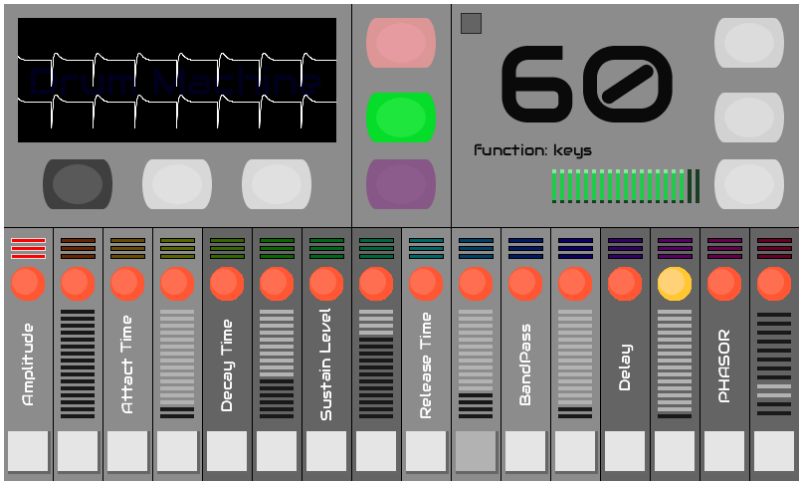
Zapínání: Bicí automat se zapne automaticky po připojení adaptéru do elektrické sítě.

Tempo: Nastavení tempa lze změnit, pouze pokud je bicí automat zastaven. Nastavení se provádí pomocí dvou tlačítek (4) a (5). Číslo (15) označuje zobrazení tempa.



Obrázek 9: Uživatelské rozhraní s popiskem

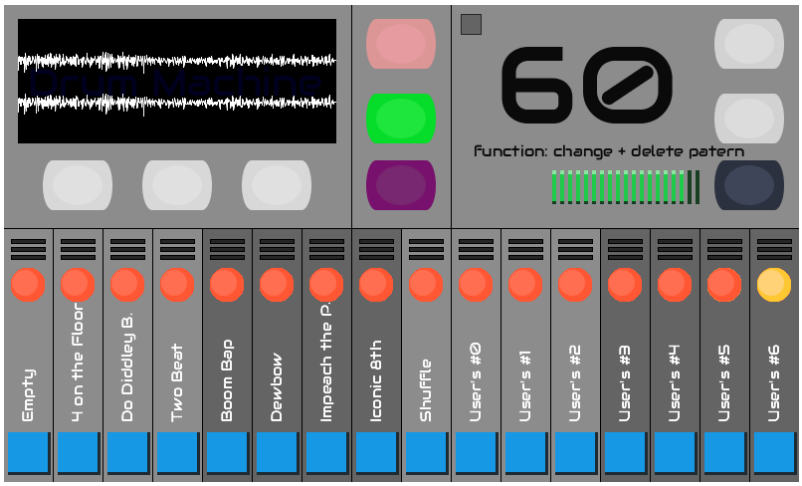
PLAY/STOP: Zelené tlačítko (2) a červené tlačítko (1).
Hlasitost: Nastavení výstupní hlasitosti se provádí zeleným barem (17).
Ukazatel doby: Ukazatel aktuálně přehrávané doby se pohybuje při přehrávání zleva doprava (11).
Zobrazení rytmu: V matici barevných symbolů představuje každý řádek jeden nástroj (12).
SHIFT: Tlačítko (3), po zmáčknutí hlavní tlačítka (13) slouží pro výběr nástroje. Opětovným zmáčknutím hlavní tlačítka slouží k nastavení rytmu vybraného nástroje. Zároveň po stisknutí tlačítka SHIFT (3) slouží tlačítko (6) k zobrazení uložených rytmů.
Přehrání stisknutím: Po stlačení tlačítka (9) je možné přehrávat zvuky nástrojů stisknutím hlavních tlačítek (13) a tlačítko (6) aktivuje funkci vytvoření rytmu přehráváním zvuků tlačítky (13).
Vytvoření rytmu: Vytvoření rytmu daného nástroje se provádí tlačítky (13), nebo po stisknutí tlačítka (9) a pak (6) lze rytmus vytvořit živým hraním.
MUTE: Ztlumení daného nástroje je možné dvěma způsoby. Tlačítko (8) aktivuje funkci MUTE, ztišení nástroje je pak možné hlavními tlačítky (13). Druhou možností jsou tlačítka (10), za podmínky, že jsou barevná. Pokud jsou černá, slouží ke smazání rytmu daného nástroje.
Klávesy: Po stisknutí tlačítka (7) se zobrazí rozhraní kláves a hlavní tlačítka (13) slouží jako klávesy, viz obrázek 10. Tlačítko (6) pak slouží k posunu tónů kláves o 1 oktávu.



Obrázek 10: Uživatelské rozhraní v režimu kláves

Smazání rytmu: Smazaní celého rytmu daného nástroje je možné po stisknutí tlačítka SHIFT (3) černými tlačítky (10). Smazání celého rytmu bicího automatu je možné vybráním uloženého rytmu "Empty".
Výběr uloženého rytmu: Po stisknutí tlačítka SHIFT (3) slouží tlačítko (6) k zobrazení uložených rytmů. Výběr uloženého rytmu se provádí tlačítky (13), viz obrázek 11.
Uložení rytmu: Uložení rytmu je možné po stisknutí tlačítka SHIFT (3) a pak tlačítka (6) do jedné z pozic *User's #* stisknutím symbolu ukazatele (11), viz obrázek 11.

MIDI: Po připojení MIDI zařízení do USB rozhraní je třeba zmáčknout tlačítko (14) pro navázání komunikace.
Popisek zvolené funkce: Označen číslem (16).
Osciloskop: Zobrazuje časový průběh výstupního signálu, označen číslem (18).



Obrázek 11: Uživatelské rozhraní v režimu uložení/výběru uloženého rytmu

5.2 Přednastavené rytmy

Jak vyplývá z textu výše, bicí automat obsahuje 8 přednastavených rytmů. Tyto rytmy uživatel může načíst a dále je upravovat dle libosti. Není však možné tyto rytmy přepsat na uložených pozicích. Při implementaci přednastavených rytmů jsem se inspiroval populárními rytmy z videa na platformě *Youtube* [18].

6 Závěr

Cílem této práce bylo navrhnout a sestavit funkční prototyp bicího automatu. Zadání práce bylo splněno ve všech bodech, jelikož hudební nástroj byl realizován po softwarové i hardwarové stránce.

Vlastní program byl napsán ve vývojovém prostředí *Processing*, ve kterém bylo implementováno grafické uživatelské rozhraní využívající celou plochu použité obrazovky. K realizaci byly použity dvě externí knihovny - *Minim* a *TheMidiBus*. Výhodou je, že se program bicího automatu přímo nepojí s hardwarem a lze ho spustit i na běžných počítačích. Ovládání je pak možné pouze myší a klávesnicí nebo pomocí MIDI kláves.

Hardware byl sestaven za použití mikropočítače *Raspberry Pi 4 B* s dotykovou obrazovkou. Ovládání zařízení je implementováno pomocí zmíněné dotykové obrazovky. Jelikož bylo podle pokynů zadání práce implementováno komunikační rozhraní hudebních nástrojů - MIDI, je možné k zařízení připojit MIDI klávesy přes USB rozhraní a jimi ho následně ovládat. Bicí automat lze i připojit k počítači přes MIDI rozhraní a využít ho jako MIDI kontrolér k libovolnému programu podporující MIDI standard.

Výsledný prototyp je zasazen v těle vytvořeném pomocí 3D tisku. Celé zařízení tak působí kompaktně a elegantně. Dále bylo vyrobeno i ochranné víčko na zmíněnou obrazovku. Při přenášení zařízení pak nehrozí poškrábání.

Bicí automat je tvořen 16 pozičním sekvencerem a obsahuje 16 různých zvuků bicích nástrojů. Díky tomu je schopný napodobit většinu základních rytmů. Některé nejznámější rytmy jsou uložené v paměti a lze je načíst a upravovat. Uživatel si může jednoduše uložit i 7 vlastních rytmů.

Plně funkční prototyp zůstane uložený ve škole a bude sloužit při reprezentaci školy, například při dni otevřených dveří a podobných akcích.

Literatura a použité zdroje

[1] S. Crab, "Chamberlin 'Rhythmate', Harry Chamberlin, USA, 1947." Dostupné z: https://120years.net/chamberlin-rhythmate-harry-chamberlin-usa1949/. [cit. 9. 4. 2022].

[2] N. Prior, "Software Sequencers and Cyborg Singers: Popular Music in the Digital Hypermodern," *New Formations*, vol. vol. 66, pp. 81–99, 2009.

[3] MATRIX, "Extremely Rare Chamberlin Rhythmate." Dostupné z: https://www.matrixsynth.com/2016/12/extremely-rare-chamberlin-rhythmate.html, 2016. [cit. 10. 4. 2022].

[4] M. Puckette, *The Theory and Techniques of Electronic Music*. World Scientific Publishing Company, 2007. ISBN 13 978-981-270-077-3.

[5] ROLAND, "TR-707." Dostupné z: https://www.roland.com/cz/products/rc_tr-707/. [cit. 18. 3. 2022].

[6] T. Brett, "Prince's Rhythm Programming: 1980s Music Production and the Esthetics of the LM-1 Drum Machine," *Popular music and society*, vol. 43(3), pp. 244–61, 2020.

[7] D. Forró, *Midi - Komunikace v hudbě*. Grada, 1993. ISBN 80-85623-56-0.

[8] B. Edstrom, *Arduino for Musicians: A Complete Guide for Arduino and Teensy Microcontrollers*. Oxford University Press, 2016. ISBN 978-0-19-930931-3.

[9] B. Fry and C. Reas, "A short introduction to the Processing software and projects from the community." Dostupné z: https://processing.org/overview, 2021. [cit. 27. 11. 2021].

[10] D. Quartz, "Audio Library Minim." Dostupné z: https://code.compartmental.net/minim/index.html. [cit. 18. 3. 2022].

[11] S. Smith, "The MidiBus Library." Dostupné z: http://www.smallbutdigital.com/projects/themidibus/. [cit. 18. 3. 2022].

[12] Y. Wate, "3 Ways to Run a Raspberry Pi Program or Script at Startup." Dostupné z: https://www.makeuseof.com/how-to-run-a-raspberry-pi-program-script-at-startup/, 2021. [cit. 21. 3. 2022].

[13] Processing, "Reference." Dostupné z: https://processing.org/reference. [cit. 25. 3. 2022].

[14] D. Quartz, "Minim: Example of Sampler." Dostupné z: http://code.compartmental.net/minim/sampler_class_sampler.html, 2009. [cit. 28. 11. 2021].

[15] Korg, "Volca Beats Analog Rhythm Machine." Dostupné z: https://www.korg.com/us/products/dj/volca_beats/. [cit. 1. 4. 2022].

[16] MusicRadar, "SampleRadar: 1,000 free drum samples." Dostupné z: https://www.musicradar.com/news/drums/1000-free-drum-samples. [cit. 1. 4. 2022].

[17] SampleFocus, "Free Tambourines Samples." Dostupné z: https://samplefocus.com/categories/tambourines. [cit. 1. 4. 2022].

[18] M. Burns, "10 Popular Drum Patterns Every Producer Should Know." Dostupné z: https://www.youtube.com/watch?v=c7ffMObdxro&t=1s. [cit. 2. 4. 2022].