VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2019 Bc. Matěj Korytář

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

MIDI KONTROLER PRO ŘÍZENÍ DAW SOFTWARE
MIDI CONTROLLER FOR DAW SOFTWARE CONTROL

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Matěj Korytář**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Jiří Schimmel, Ph.D.**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Audio inženýrství**
Ústav telekomunikací

Student: Bc. Matěj Korytář ***ID:*** 174456
Ročník: 2 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

MIDI kontroler pro řízení DAW software

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte využití protokolu MIDI při řízení zvukového produkčního software, tzv. digital audio workstation. Analyzujte a zdokumentujte způsob komunikace a komunikační protokol jednoho nebo dvou nejčastěji používaných softwarů v této oblasti. Následně navrhněte a realizujte hardwarový ovladač využívající protokolu MIDI k řízení těchto software. Ovladač bude obsahovat alespoň následující typy ovládacích prvků: tlačítka, digitální tahové potenciometry, rotační enkodery a display pro zobrazení hodnot parametrů a informací ze software.

DOPORUČENÁ LITERATURA:

[1] The Complete MIDI 1.0 Detailed Specification, document version 96.1, MIDI Manufacturers Association, Japan MIDI Standard Committee, 1997.

[2] Forró,D.:Počítačeahudba,Grada,Praha, 1994.

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 16.5.2019

Vedoucí práce: doc. Ing. Jiří Schimmel, Ph.D.
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

Abstrakt

Tato práce se zaměřuje na problematiku DAW kontrolerů pro řízení zvukového produkčního software. V rámci práce je analyzován jeden z používaných komunikačních protokolů, protokol Mackie Control. Jelikož je tento protokol založen na MIDI, je zde také popsán přehled typů MIDI zpráv. Práce obsahuje celkem dva návrhy vlastního DAW kontroleru. První je založen na protokolu Mackie Control, druhý na Generic Remote, který nabízí lepší přizpůsobitelnost.

Klíčová slova

DAW kontroler, MIDI, Mackie Control Protocol, Cubase Generic Remote, Atmel ATmega 2560

Abstract

This thesis focuses on problematics of DAW controllers for sound production software. In this thesis is analysed Mackie Control protocol, one of used communication protocols. There is also an overview of MIDI messages, because Mackie Control protocol is based on MIDI protocol. The thesis contains two own DAW controller designs. First one is based on Mackie Control protocol, second one is based on Generic Remote which offers better customization.

Keywords

DAW controller, MIDI, Mackie Control Protocol, Cubase Generic Remote, Atmel ATmega 2560

Bibliografická citace:

KORYTÁŘ, M. MIDI kontroler pro řízení DAW software. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2019. 70 s. *Vedoucí diplomové práce doc. Ing. Jiří Schimmel, Ph.D.*

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma MIDI kontroler pro řízení DAW software jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.
Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: **15. května 2019** ……

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce doc. Ing. Jiřímu Schimmelovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne: **15. května 2019** ……

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

1. ÚVOD

V dobách analogového záznamu zvuku používal zvukař při své práci klasický analogový mixážní pult v kombinaci s vícestopým páskovým magnetofonem. V mixážním pultu byla zvuku upravena vstupní úroveň pomocí předzesilovače, dále barva pomocí ekvalizéru a takto upravený zvuk byl poslán do stopy magnetofonu, kde byl nahrán na analogový pás. Záznam byl vícestopý a bylo tedy možné nahrávat například až do šestnácti nebo dvaceti čtyř stop najednou. Pokud chtěl zvukař při finální mixáži provést úpravu zvuku, kterou mixážní pult neumožňoval nebo přidat dozvuk případně jiný druh efektu, použil k tomu externí přístroje. Vše se ovládalo pomocí potenciometrů, tlačítek, přepínačů a tahových potenciometrů. Každý ovládací prvek má tedy svou jednu určitou roli. Tahový potenciometr, jinak také nazýván fader, ovládá hlasitost pouze svého kanálu, protože přímo přes něj prochází elektrický signál. Stejně tak ekvalizér na analogovém mixážním pultu obsahuje elektrický obvod, díky kterému je možné měnit barvu zvuku. Pokud má mixážní pult šestnáct kanálů, má většinou stejný počet ekvalizérů, individuálních pro každý kanál.

Po příchodu digitalizace a digitální technologie se záznam z magnetofonového pásu přesunul na zápis dat na harddisk a úprava zvuku z analogového mixážního pultu a externích přístrojů na zpracování signálů na osobním počítači. K digitalizaci zvuku a zpětnému přehrání je potřeba už jen zvuková karta obsahující A/D a D/A převodník. O vše ostatní už se postará počítač nebo notebook. Prostředí, které prakticky nahradilo výše zmiňované analogové přístroje se nazývá DAW – digital audio workstation. Mezi nejpoužívanější DAW software patří mimo jiné Steinberg Cubase, Avid Pro Tools, Presonus Studio One, Apple Logic Pro, Ableton, Reason a FL Studio.

Jednotlivé DAW mají odlišné grafické prostředí, způsob ovládání a také možnosti zpracování. Co se však neliší je princip. Vždy jde o nahrání zvuku nebo jeho vytvoření, za pomoci virtuálních hudebních nástrojů, a následné úpravě. K ovládání se používá standardních nástrojů, které má počítač běžně k dispozici, tedy klávesnice a myš. Možnosti střihu za použití digitálních technologií se razantně zdokonalily a jsou také oproti střihu na analogovém pásu nedestruktivní, úpravu lze tedy vždy vrátit zpět do původní podoby. Pro úpravu barvy a vyvážení jednotlivých zvukových stop máme k dispozici virtuální mixážní pult. Obsahuje rovněž ovládací prvky známé z analogových pultů, ty jsou však nyní jen virtuální a ovládají se pomocí myši a klávesnice. Mezi výhody virtuálního mixážního pultu patří, mimo jiné, volitelný počet kanálů. Můžeme si tedy vytvořit tolik zvukových stop, kolik právě potřebujeme. Z hlediska ovládání jsme však omezeni velikostí monitoru a musíme tedy přepínat mezi okny projektu, mixu a pluginů. Vždy tedy vidíme jen část celku a musíme se proklikat k jednotlivým nastavením. Pokud chceme nastavení automatizovat, tak aby se s časem například měnila hlasitost, je mnohem těžší udělat plynulý pohyb virtuálního faderu pomocí myši než pohyb fyzickým tahovým potenciometrem.

Pro rychlejší, efektivnější a také uživatelsky přívětivější práci s DAW software se tedy začaly využívat kontrolery. Stisk tlačítka je rychlejší než najetí myši na požadované místo a kliknutí, stejně jako pohyb fyzického enkodéru, nahrazující potenciometr. Fyzických tahových potenciometrů můžeme také ovládat několik najednou, kdežto myší pouze jeden. V dnešní době existují desítky kontrolerů, od malých (Presonus Fader Port), přes středně velké (Mackie Control Universal Pro) až po velké (Avid Icon D-Control). Jejich využití je tedy od malých nahrávacích studií zabývajícími se například nahráváním řeči nebo sólových nástrojů, až po velká studia pro zpracování rozsáhlých hudebních těles nebo zvuku pro filmový průmysl. Oproti analogovým přístrojům umožňují kontrolery přepínat funkci jednotlivých ovládacích prvků a také pracovat ve vrstvách. Jeden enkodér může tedy ovládat nastavení panoramy, práh kompresoru nebo frekvenci ekvalizéru. Uživatel si jednoduše zvolí, který parametr právě potřebuje mít k dispozici. Pozornost zvukaře se tak přesouvá od práce na monitoru opět směrem k pultu, který je nahrazen kontrolerem.

Navržený kontroler je určen pro mé potřeby a přizpůsoben mým požadavkům tak, abych DAW software ovládal především jím a zvýšila se tak rychlost a efektivita mé práce. Hlavní myšlenkou je soustředit pozornost na enkodéry s displejem a mít možnost tyto parametry volit a přepínat. Enkodéry totiž umožňují nastavení převážné většiny potřebných parametrů. Tento přístup zajišťuje univerzálnost a snadné přizpůsobení. Úroveň hlasitosti se nastaví na začátku a poté už hlasitost hlídá kompresor, který upravuje dynamiku signálu. Proto také pro mé potřeby dostačuje jeden tahový potenciometr, který nastavuje úroveň právě zvoleného kanálu.

Jelikož těchto požadavků nešlo dosáhnou za použití protokolu Mackie Control, především kvůli jeho pevně dané struktuře, využil jsem možnost ovládání pomocí Generic Remote, který umožnuje přiřadit MIDI zprávám potřebné parametry.

Práce se nejprve zaměřuje analýzou protokolu Mackie Control a na základě získaných zkušeností je v kapitole 3 navržen kontroler založený na tomto protokolu. V následující kapitole jsou rozebrány jednotlivé ovládací prvky kontroleru. Funkčnost těchto prvků byla ověřena pomocí Generic Remote. Po odzkoušení všech prvků jsem se rozhodl u Generic Remote díky jeho univerzálnosti zůstat a návrh kontroleru upravit, což je uvedeno v kapitole 5. V další kapitole je popsán návrh celkového programu pro mikročip kontroleru a v kapitole 7 je uvedena konstrukce kontroleru včetně návrhu desek plošných spojů.

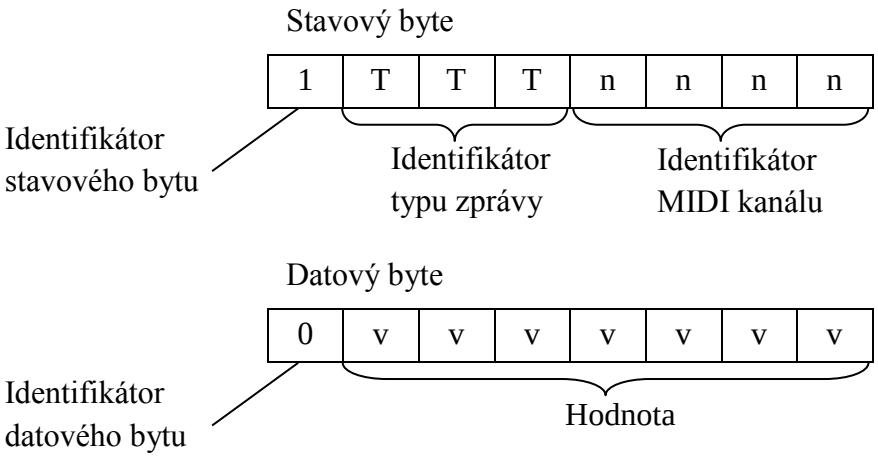
2. ANALÝZA KOMUNIKACE PROTOKOLU MACKIE CONTROL

Smysl kontroleru je na první pohled zřejmý, hardwarem ovládáme software. Tyto dvě části spolu tedy musí navzájem komunikovat. Aby toho bylo možné dosáhnout, bylo postupně zavedeno několik typů protokolů. Jako příklad můžeme uvést protokoly Mackie Control, HUI a Logic Control. Protokol Mackie Control podporuje většina DAW software na rozdíl od jiných protokolů, které byly navrženy pro určitého výrobce. Protokol HUI je primárně určen pro Avid Pro Tools, Logic Control pro produkty firmy Apple, jako je například Logic. [1]

V této práci se budeme zabývat protokolem Mackie Control. Protokol je, stejně jako další výše zmiňované, postaven na protokolu MIDI. Zkratka v překladu znamená digitální rozhraní hudebních nástrojů [2]. Jak název napovídá, byl protokol primárně navržen pro komunikaci mezi hudebními nástroji, nejčastěji klávesovými, uchytil se však i při ovládání DAW software. Protokol Mackie Control je tedy přiřazení nového standardizovaného významu stávajícím MIDI zprávám v protokolu MIDI pro potřeby ovládání DAW kontrolerem.

2.1 Protokol MIDI

Protokol MIDI je asynchronní sériový komunikační protokol, který nabízí několik předdefinovaných typů zpráv. Těmto zprávám říkáme MIDI zprávy. Každá zpráva obsahuje jeden stavový *byte* a několik datových *bytů*.



Obr. 2-1: Struktura stavového a datového bytu MIDI zprávy [2]

Stavový *byte* má nejvýznamnější *bit* nastaven na hodnotu 1. Následující 3 *bity* určují typ zprávy a poslední 4 *bity* nesou informaci o čísle MIDI kanálu [3]. Z počtu kombinací vychází, že máme k dispozici celkem 8 typů zpráv a 16 kanálů, na kterých můžeme tyto zprávy přenášet. Datový *byte* má nejvýznamnější *bit* nastaven na hodnotu 0, zbývajících 7 *bitů* je použito pro přenos dat. Data jsou čísla v rozsahu 0 až 127 a slouží jako doplňkové informace k předchozímu stavovému *bytu*.

Tabulka 2-1 Kanálové MIDI zprávy [2]

MIDI zpráva	Význam	ID	Počet databytů
Note Off	Nota vypnuta	0	2
Note On	Nota zapnuta	1	2
Polyphonic Key Pressure	Individuální tlaková citlivost	2	2
Control Change	Změna kontroleru	3	2
Program Change	Volba programu	4	1
Channel Pressure	Společná tlaková citlivost	5	1
Pitch Bend Change	Ohýbání tónu	6	2

Tabulka 2-1 uvádí přehled kanálových MIDI zpráv, vázaných k určitému MIDI kanálu. Dále však existují Systémové MIDI zprávy s identifikátorem 7, které jsou společné pro všechny MIDI kanály. Dolní 4 *bity* tedy v tomto případě identifikují typ systémových zpráv. Systémových zpráv existuje celkem 16 a patří mezi ně, mimo jiné, zvláštní systémová data (System Exclusive), informace o synchronizaci, ukazatele pozice skladby a další systémové příkazy. Výhodou zvláštních systémových zpráv je možnost přenosu většího počtu *bytů*, který se pohybuje v řádu desítek i stovek, a může se jednat o libovolná data případně i text v kódu ASCII. Celý blok dat ohraničují dvě MIDI zprávy. Zpráva System Exclusive značí začátek zvláštních systémových dat, zpráva End Of Exclusive jejich konec. Mezi těmito zprávami už se nachází data, která chceme přenést. Už v roce 1985 [2], kdy vyšla podrobná MIDI norma, mysleli inženýři dopředu a díky zvláštním systémovým datům dali ostatním výrobcům možnost přizpůsobit si část protokolu k vlastním potřebám.

Tabulka 2-2 Vybrané systémové MIDI zprávy [2]

MIDI zpráva	Význam	ID	Počet databytů
System Exclusive	Začátek zvláštních systémových dat	$F0	neurčen
End Of Exclusive	Konec zvláštních systémových dat	$F7	0

2.2 Kontroler Mackie MCU

Jelikož dokumentace k protokolu Mackie Control není volně dostupná a lze dohledat jen omezené informace, je nutné tento protokol analyzovat a komunikaci vysledovat. Pro analýzu komunikace protokolu Mackie Control jsem si zvolil kontroler Mackie MCU. Tento kontroler disponuje sekcí osmi kanálů, z nichž každý kanál má svůj tahový potenciometr, čtveřici tlačítek a rotační enkodér. Nad enkodéry se nachází dvouřádkový displej zobrazující převážně jména kanálů a parametr ovládaný enkodérem. Nemusí to být však pravidlem, protože zobrazení se váže k použitému DAW software. Devátý tahový potenciometr ovládá úroveň hlavní sběrnice, v anglicky psané literatuře jej můžeme najít pod pojmem Master fader. Nad ním se nachází tlačítka k přepínání vrstev a rolování kanálů. Pokud tedy náš projekt obsahuje například dvacet kanálů, můžeme se mezi nimi posouvat buď po osmi nebo po jednom. Toto je jedna z výhod kontrolerů, osmi fyzickými tahovými potenciometry můžeme ovládat desítky virtuálních. Které zrovna ovládáme záleží na zvolené vrstvě.

Nad tahovým potenciometrem hlavní sběrnice se dále nachází šest tlačítek na černém podkladu, kterými se volí funkce rotačních enkodérů. Tlačítky se dá přepnout, zda enkodér nastavuje panoramu kanálů, ekvalizér, parametry plugin efektů nebo úroveň posílanou do pomocných sběrnic například pro přidání umělého dozvuku. Pro bližší orientaci o funkci enkodérů slouží dvouznakový segmentový displej, který je užitečný především při posílání signálu do pomocných sběrnic značených jako Send. Pokud projekt v DAW Studio One obsahuje čtyři tyto sběrnice, jediným tlačítkem cyklicky přepínáme sběrnice, přičemž se na displeji postupně zobrazuje S1, S2, S3, S4. Písmeno "S" značí, že enkodéry slouží k nastavení úrovně do sběrnice Send, následující číslo udává pořadí sběrnice.

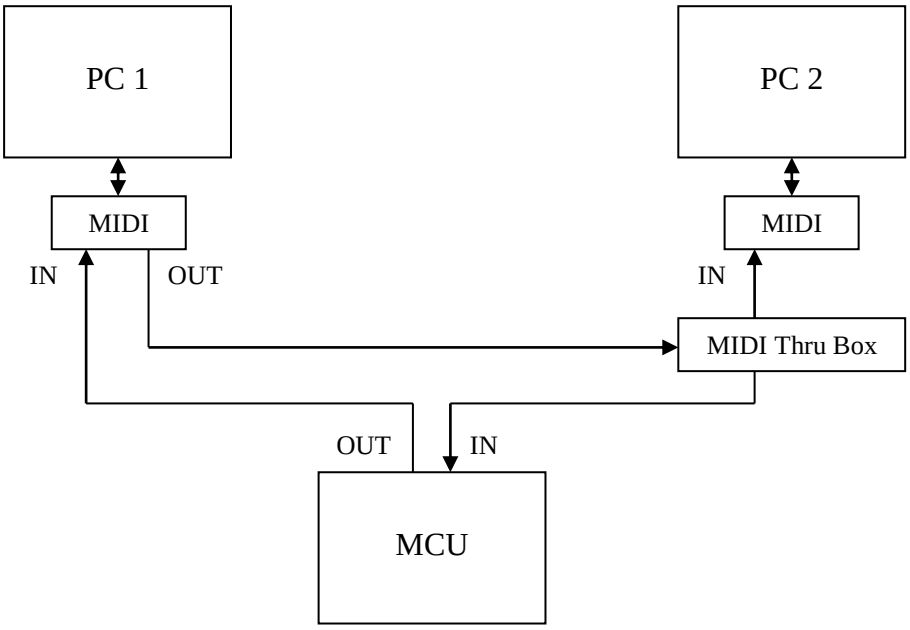
Obr. 2-2: Kontroler Mackie MCU Pro [4]

Na pravé straně kontroleru se nachází sekce tlačítek, jejichž význam je také vázán na použitý DAW software. Standardním obsahem balení kontroleru je sada několika šablon s popisy, která překryje stávající popis tlačítek. Každý výrobce DAW software si tedy může zvolit, jakou funkci mají tlačítka mít, respektive jak na ně bude program reagovat při zachování stejného hardware. Tlačítka, která mají svůj význam vždy stejný jsou tlačítka transportu. Jedná se o Start, Stop, Record a tlačítka pro přetáčení. Dále jsou to kurzorová tlačítka pro orientaci v projektu, která se dají přepnout na změnu zoomu, a otočný ovladač Jog wheel pro přetáčení.

Kontroler dále obsahuje segmentový display pro zobrazení údajů o pozici v projektu, a to buď v čase nebo v dobách. Volba zobrazení se přepíná tlačítkem pod displejem a je také indikována dvojicí LED diod po levé straně tohoto displeje.

2.3 Schéma analýzy komunikace

Hlavními objekty měření byl kontroler Mackie MCU a program Midi Analyser. Tento program zobrazuje všechny příchozí MIDI zprávy na zvoleném portu a vypisuje je do tabulky. Kromě samotných MIDI dat v hexadecimálním kódu také zobrazuje typ MIDI zprávy, hodnotu datových *bytů* v dekadické soustavě, MIDI kanál a čas ve kterém zpráva přišla, počítaný od spuštění programu.



Obr. 2-3: Blokové schéma zapojení pro analýzu komunikace

Schéma měření je znázorněno blokově na obrázku 2-3. Každý MIDI kabel přenáší data pouze jednosměrně, pro každý směr přenosu bylo tedy nutné použít individuální kabel. K analýze byly použity dva počítače, PC1 a PC2. Jelikož počítače běžně nedisponují MIDI porty, bylo pro připojení použito zvukové rozhraní Steinberg UR22, obsahující mimo zvukových vstupů a výstupů také MIDI porty IN a OUT. Na počítači PC1 byl spuštěn DAW software, který komunikoval s výše zmiňovaným kontrolerem Mackie MCU. Dále zde byl spuštěn program Midi Analyser pro zobrazení toku příchozích MIDI zpráv. Při běžném zapojení kontroleru by stačila dvojice MIDI kabelů, každý pro jeden směr. Jelikož však program Midi Analyser umožňuje sledovat pouze příchozí tok MIDI zpráv, nebylo by možné sledovat směr komunikace od DAW do kontroleru. Proto bylo zapojení rozšířeno o počítač PC2. MIDI výstup z PC1 byl připojen do MIDI Thru boxu, abychom k výstupu mohli připojit kontroler a zároveň tok MIDI zpráv odbočili do počítače PC2, na kterém rovněž běžel program Midi Analyser. Počítač PC2 tedy sloužil jen pro účely monitorování a žádné nové MIDI zprávy do zapojení nevnášel.

Díky tomuto zapojení bylo možné sledovat odděleně oba směry komunikace. Na PC1 se zobrazovaly MIDI zprávy vysílané kontrolerem a na PC2 zprávy vysílané z DAW software, které kontroler přijímal. Při každé události jako je stisk tlačítka nebo pohyb faderu vysílá kontroler zprávy i když není spuštěn DAW software. Na počítači PC1 tedy vidíme jednotlivé události, kdežto na PC2 není známa žádná aktivita. Teprve až po spuštění DAW software začne software na události z kontroleru reagovat. Nutno dodat, že je potřeba v DAW specifikovat typ kontroleru a uvést, na kterých MIDI portech je k počítači připojen. Existuje totiž několik typů kontrolerů včetně digitálních mixážních pultů, které se jako kontroler umí chovat. Z nich každý může mít jiný počet tahových potenciometrů, tlačítek a enkodérů. Proto musí DAW vědět který typ kontroleru je k němu připojen, aby mohl správně vyhodnocovat a reagovat na příchozí MIDI zprávy.

2.4 Motorizované tahové potenciometry

Hlavním požadavkem na tahové potenciometry je, pokud možno, co nejpřesnější údaj o pozici jezdce. Z datového *bytu* běžné MIDI zprávy máme k dispozici pouze 7 *bitů*, do kterých můžeme zakódovat číslo od 0 do 127, což je pro tahový fader o délce 10 cm nedostačující. Jelikož je virtuální dráha jezdce semilogaritmická, v nižších polohách by fader navíc trpěl na horší rozlišitelnost. Reálná dráha tahového potenciometru je však lineární, protože nás zajímá jen jeho pozice a neprochází přes něj zvukový signál. Přepočet na semilogaritmickou dráhu se provádí až v DAW software. Aby se průběh jevil jako spojitější, využívá se tedy pro přenos MIDI zpráva ohýbání tónu – Pitch Bend. Tato zpráva se podle MIDI normy skládá z jednoho stavového *bytu* a

dvou datových *bytů,* z nichž dvojice datových *bytů* přenáší jedno 14-ti bitové číslo. Přenášený údaj o pozici může být tedy v rozsahu od 0 do 16 383. Při analýze bylo však zjištěno, že změna polohy tahového potenciometru o jeden krok vyvolá změnu přenášeného čísla o hodnotu 16. Použitý A/D převodník je tedy jen 10-ti bitový s následnou úpravou rozsahu. Přesnost mezi jednotlivými kvantovacími kroky pro dráhu o délce 10 cm je 0,097 mm, což se dá považovat za velmi jemný krok.

Jelikož je MIDI zpráva Pitch Bend Change jedinečná pro každý kanál, musíme pro další tahové potenciometry využít následující MIDI kanály. Tahové potenciometry 1 až 8 jsou přenášeny po kanálech 1 až 8, tahový potenciometr hlavní sběrnice po kanále 9.

Kromě snímání polohy umožňuje tahový potenciometr i snímání doteku. Pokud bychom jím chtěli pohnout například tužkou, nebude to mít žádný vliv na změnu hlasitosti a nedojde ani k vyslání MIDI zprávy. Zabrání se tak nechtěné manipulaci. Po chvíli se jezdec vrátí zpět do původní polohy, návrat obstará kontroler. Informace o doteku se přenáší pomocí MIDI zprávy Nota zapnuta. Pokud kontroler vyhodnotí dotek, pošle zprávu Note On, kde první parametr neboli první datový *byte* slouží k identifikaci pořadí tahového potenciometru a druhý datový *byte* přenáší hodnotu 127. Když uživatel tahový potenciometr uvolní je poslána opět zpráva Note On se shodným prvním parametrem, jako druhý parametr je poslána hodnota 0. Všechny zprávy Note On využívají pro přenos MIDI kanál č. 1.

Tabulka 2-3 Parametry MIDI zpráv pro tahové potenciometry

Pořadí tahového potenciometru	Kanál pro přenos hodnoty Pitch Bend	Citlivost na dotek / 1. parametr Note On
Kanál 1	1	104
Kanál 2	2	105
Kanál 3	3	106
Kanál 4	4	107
Kanál 5	5	108
Kanál 6	6	109
Kanál 7	7	110
Kanál 8	8	111
Hlavní sběrnice	9	112

Citlivost na dotek má však mnohem důležitější význam. Pokud chceme parametry mixu automatizovat, například po dobu skladby upravovat hlasitost zpěvu, povolíme v DAW zápis hodnoty hlasitosti, pustíme skladbu a pohybujeme tahovým potenciometrem podle potřeby. Všechny provedené změny se uloží a jsou při následném přehrávání skladby vyvolány. Pokud je kontroler vybaven motory, můžeme

sledovat změnu také fyzicky, tahové potenciometry se začnou díky motorům pohybovat. Kdybychom se rozhodli část našeho zaznamenaného pohybu upravit, je možné jej přepsat. Opět bychom skladbu pustili, a od okamžiku doteku se začne zaznamenávat nová pozice. Podle nastavení v DAW se bude zápis po uvolnění prstu chovat následovně. Tahový potenciometr se buď ihned vrátí do předchozího zaznamenaného pohybu, nebo bude ještě po dobu přehrávání zaznamenávat aktuální hodnotu i po uvolnění. Dalším důvodem, proč je citlivost na dotek dobrá, je zamezení přetahování mezi člověkem a kontrolerem. Pokud dostane kontroler od DAW povel na změnu pozice jezdce a uživatel jej blokuje, dojde ke změně na požadované místo až po uvolnění. Tento stav je záležitostí naprogramování kontroleru a nesouvisí s DAW, naproti tomu automatizaci řeší a zpracovává jen DAW software.

Komunikace mezi kontrolerem a DAW software vypadá následovně a lze ji rozdělit na čtyři části. Nejdříve kontroler vyšle zprávu Note On s informací o doteku. V okamžiku kdy uživatel s jezdcem pohne, dojde k vyslání zprávy Pitch Bend Change vždy, když čip kontroleru zaznamená změnu hodnoty. Těchto zpráv může být v řádu desítek, když je změna jen o několik milimetrů, ale také v řádu stovek, když jezdcem neustále pohybujeme nahoru a dolů. Nikdy se však nepřenáší tatáž hodnota, aby nedošlo k zahlcení MIDI sběrnice. Pokaždé když dojde k odeslání nové polohy jezdce, vyšle DAW zpět do kontroleru jeho aktualizovanou hodnotu. V případě kdy na jezdci máme prst ale nepohybujeme s ním, k vysílání MIDI zpráv nedochází. Když jezdce uvolníme, kontroler vyšle zprávu o jeho uvolnění opět formou zprávy Note On.

Tabulka 2-4 Příklad komunikace při pohybu tahovým potenciometrem

Posláno	Dotek jezdce kanálu 1	$90 $68 $7F	Note On 104, velocity 127
Posláno	Změna polohy jezdce 1	$E0 $40 $61	Pitch Bend ch. 1, value 12 480
Přijato	Potvrzení nové pozice	$E0 $40 $61	Pitch Bend ch. 1, value 12 480
Posláno	Uvolnění jezdce kanálu 1	$90 $68 $00	Note On 104, velocity 0

Kontroler Mackie MCU podporuje nejen vysílání zpráv Pitch Bend Change, ale také jejich přijímání. V případě, kdy je požadavek na změnu pozice jezdce ze strany DAW, probíhá komunikace stejně jako ze směru od kontroleru. V tomto případě už ale jen v jednom kroku. Když přijde kontroleru zpráva Pitch Bend Change na kanálu 2 s parametrem 8 191, postará se o to, aby motor posunul jezdce kanálu 2 na střed dráhy tahového potenciometru. Zde si můžeme všimnout, že první datový *byte* nese *bity b0* až *b7*, druhý datový *byte* pak *bity b8* až *b14*.

Tabulka 2-5 Příklad komunikace při požadavku změny jezdce ze strany DAW

Posláno	Změna polohy jezdce 2	$E1 $7F $3F	Pitch Bend ch. 2, value 8 191

2.5 Význam tlačítek a jejich MIDI zprávy

Kontroler Mackie MCU obsahuje celkem 94 tlačítek, která si podle funkce můžeme rozdělit do několika skupin. První skupinu tvoří kanálová tlačítka. Jedná se o tlačítka Rec, Solo, Mute a Select, která jsou běžnou součástí DAW mixážních pultů vyjma tlačítka Select. V DAW kanál volíme buď kliknutím myši nebo kurzorovými tlačítky klávesnice. Na kontroleru k tomu máme vyhrazené tlačítko. Některé DAW jako například Cubase označí kanál i při doteku faderu, tato možnost se však dá v nastavení vypnout. Volba kanálu může urychlit práci na více stopách například při nastavení ekvalizéru nebo posílání signálu do efektových sběrnic. Dále zde máme tlačítko Rec, které aktivuje nahrávání daného kanálu, tlačítko Solo způsobující vypnutí ostatních kanálů a tlačítko Mute, které daný kanál umlčí. Tato sekce se tedy váže k práci s jednotlivými kanály.

Informace o stisku tlačítka se přenáší pomocí MIDI zpráv Note On na prvním MIDI kanále. Celkem jsou vyslány dvě MIDI zprávy. První při stisknutí a druhá při uvolnění tlačítka. DAW má tedy přesnou informaci a podle naprogramování zareaguje na jednu ze zpráv. Podobnost můžeme najít u analogových tlačítkových přepínačů, které při prvním stisku přepnou obvod hned a zůstanou zamáčknuty, naopak při druhém stisku dojde k přepnutí až po jejich uvolnění. Dalším hlediskem je děj, který probíhá po celou dobu držení tlačítka jako je například přetáčení vpřed nebo vzad. Význam má také při klávesových zkratkách, kdy jedno tlačítko držíme a mezitím stiskneme druhé.

Zprávu Note On tvoří podle MIDI normy dva datové *byty* přenášející číslo noty a sílu úderu. Zde číslo noty zastupuje číslo tlačítka a sílu úderu buď číslo 127 při stisknutí tlačítka nebo číslo 0 při jeho uvolnění. Zpráva Note On má kapacitu 127 not. Pro přenos všech tlačítek, která kontroler Mackie MCU obsahuje, stačí tedy jen jeden MIDI kanál.

Tabulka 2-6 Zjištěná čísla not používaná pro kanálová tlačítka

	Rec	**Solo**	**Mute**	**Select**
Kanál 1	0	8	16	24
Kanál 2	1	9	17	25
Kanál 3	2	10	18	26
Kanál 4	3	11	19	27
Kanál 5	4	12	20	28
Kanál 6	5	13	21	29
Kanál 7	6	14	22	30
Kanál 8	7	15	23	31

Kanálová a některá ostatní tlačítka jsou vybavena zpětnou indikací stavu. K indikaci slouží LED diody vedle tlačítka případně přímo v tlačítku. U novějšího modelu MCU Pro jsou kanálová tlačítka celá podsvícená. Když tedy například stiskneme tlačítko Mute, rozsvítí se u něj červená dioda, abychom byli informování o umlčení kanálu. Zapínání a vypínání LED diod je řízeno DAW softwarem. Používá se shodný typ zpráv se stejným číslem tlačítka jak při vysílání, tak při přijímání. V případně poslání této zprávy z kontroleru nese informaci o stisku tlačítka, v případě příjmu do kontroleru nese příkaz k rozsvícení LED diody.

Tabulka 2-7 Příklad komunikace při stisku tlačítka Mute

Posláno	Stisk Mute k. 1	$90 $10 $7F	Note On 16, velocity 127
Přijato	Rozsvícení LED Mute k. 1	$90 $10 $7F	Note On 16, velocity 127
Posláno	Uvolnění Mute k. 1	$90 $10 $00	Note On 16, velocity 0

V tabulce 2-7 je příklad komunikace při stisku tlačítka Mute na kanálu 1. Mohou však nastat složitější stavy. Tlačítko Solo indikuje funci solo na svém kanálu a zároveň ostatní kanály umlčí, což je indikováno stavem Mute. Při stisku tlačítka Solo na kanálu č. 1 vypadá situace následovně. Kontroler musí poslat informaci o stisknutí a poté vyhodnotit a vykonat všechny příchozí zprávy. Navíc je zde rozsvícení LED diody Rude Solo, která nás informuje o aktivaci funkce Solo kteréhokoli kanálu. Tato informace je užitečná například pokud se nacházíme v jiné vrstvě. Kontroler také obsahuje globální tlačítko Solo, při jehož stisknutí dojde k jednorázovému zrušení funkce solo a začnou opět hrát všechny kanály. Tlačítko rovněž obsahuje LED indikaci.

Tabulka 2-8 Příklad komunikace při stisku tlačítka Solo

Posláno	Stisk Solo k. 1	$90 $08 $7F	Note On 8, velocity 127
Přijato	Rozsvícení LED Solo k. 1	$90 $08 $7F	Note On 8, velocity 127
Přijato	Rozsvícení LED Mute k. 2	$90 $11 $7F	Note On 17, velocity 127
Přijato	Rozsvícení LED Mute k. 3	$90 $12 $7F	Note On 18, velocity 127
Přijato	Rozsvícení LED Mute k. 4	$90 $13 $7F	Note On 19, velocity 127
Přijato	Rozsvícení LED Mute k. 5	$90 $14 $7F	Note On 20, velocity 127
Přijato	Rozsvícení LED Mute k. 6	$90 $15 $7F	Note On 21, velocity 127
Přijato	Rozsvícení LED Mute k. 7	$90 $16 $7F	Note On 22, velocity 127
Přijato	Rozsvícení LED Mute k. 8	$90 $17 $7F	Note On 23, velocity 127
Přijato	LED globálního tlačítka Solo	$90 $5A $7F	Note On 90, velocity 127
Přijato	LED Rude Solo	$90 $73 $7F	Note On 115, velocity 127
Posláno	Uvolnění Solo k. 1	$90 $08 $00	Note On 8, velocity 0

Další sekcí tlačítek jsou tlačítka transportu. Patří mezi ně tlačítka play, stop, record a tlačítka pro přetáčení vpřed a vzad. Všechny jsou vybaveny indikací pomocí LED diody a platí pro ně stejná pravidla jako pro ostatní tlačítka.

Tabulka 2-9 Zjištěná čísla not pro tlačítka transportu

Funkce	Popis tlačítek	Číslo noty
Přetáčení vzad	Rewind	91
Přetáčení vpřed	Fast Fwd	92
Zastavení	Stop	93
Spuštění	Play	94
Nahrávání	Record	95

Další neméně důležitou sekcí jsou tlačítka pro posun mezi kanály. Posun je možný po jednom kanále nebo po takzvané bance, která odpovídá osmi kanálům. Do jednoho bloku byly sdruženy i další tlačítka týkající se motorizovaných tahových potenciometrů, které jsou společně uvedeny v tabulce 2-10. Pro nastavení režimu enkodérů slouží opět šest tlačítek, umístěných napravo od enkodérů. Zde se funkce liší na použitém DAW software, kterému také odpovídá jeho šablona s přepisem původního pojmenování tlačítek. V tabulce 2-10 je uvedena funkce pro DAW Studio One [5].

Tabulka 2-10 Tlačítka přepínání banky kanálů a režimů enkodérů

Ovládání	Popis tlačítek	Číslo noty	Funkce
Banky kanálů a režim tahových potenciometrů	Bank ←	46	Posun na předcházejících 8 kanálů
	Bank →	47	Posun na následujících 8 kanálů
	Channel ←	48	Rolování o 1 kanál vzad
	Channel →	49	Rolování o 1 kanál vpřed
	Flip	50	Prohození funkce tahového potenciometru a enkodéru
	Global view	51	Zobrazí společně všechny typy kanálů
Režim rotačních enkodérů	Track	40	Upravuje parametry zvoleného kanálu
	Send	41	Nastavení úrovně do pomocných sběrnic
	Pan / Surround	42	Nastavení panoramy jednotlivých kanálů
	Plug-in	43	Parametry plugin efektů
	EQ	44	Zapnutí a vypnutí insert efektů
	Instrument	45	není použito pro Studio One

2.6 Rotační enkodéry a způsob přenosu informace

Rotační enkodér je ovládací prvek, který slouží jako náhrada klasického potenciometru. Ovládá se rovněž otáčivým pohybem, ale na rozdíl od potenciometru není omezen zarážkami a může se tak pohybovat neomezeně doleva nebo doprava. Dráha enkodéru v celkovém rozsahu 360° je rozdělena na několik kroků. Jejich počet určuje citlivost, a tedy jemnost ovládání. Při změně na následující krok, tedy doprava, je vyslána informace, která nese význam hodnoty +1. Při pohybu doleva nese informace význam hodnoty -1. Někdy jsou proto také rotační enkodéry nazývaný jako inkrementační spínače. Informace není přenášena absolutně, ale pouze relativně k aktuálně nastavené hodnotě. Tato hodnota je uložena v paměti DAW. Pokud tedy například ovládáme panoramu jejíž aktuální hodnota je R20, tak při pohybu enkodéru doprava dojde k posunu o jeden krok na hodnotu R21. Jelikož princip snímání není mechanický, nemusí klást enkodér při otáčení téměř žádný odpor a lze s ním tedy otáčet volně. Některé enkodéry, například u kontrolerů Mackie, však mají dráhu krokovanou a při otáčení cítíme a slyšíme cvakání. Uživatel má tak představu o kolik kroků enkodér otočil.

Kontroler Mackie MCU obsahuje celkem 8 rotačních enkodérů s integrovaným tlačítkem. Funkce tlačítka se liší podle toho, co právě nastavujeme. Může to být nastavení na výchozí hodnotu, například u nastavení panoramy, nebo volba parametru.

Tabulka 2-11 Parametry MIDI zpráv pro rotační enkodéry

	Tlačítko / Číslo noty	**Pohyb / Číslo kontroleru**
Enkodér 1	32	16
Enkodér 2	33	17
Enkodér 3	34	18
Enkodér 4	35	19
Enkodér 5	36	20
Enkodér 6	37	21
Enkodér 7	38	22
Enkodér 8	39	23

K přenosu stavu tlačítek, integrovaných v enkodéru, se rovněž využívají zprávy Note On. Použité číslo noty pro jednotlivé enkodéry je uvedeno v tabulce 2-11. Druhý datový *byte* přenáší opět informaci o stisknutí nebo uvolnění tlačítka. Pro přenos informace o pohybu se používají zprávy Změna Kontroleru neboli Control Change.

U klávesových nástrojů je kontroler obecně ovládací parametr MIDI nástroje, jako je například hlasitost, modulace, banka zvuků a jiné [2].

MIDI zpráva Control Change se skládá ze stavového *bytu* a dvou datových *bytů*. První datový *byte* přenáší číslo kontroleru, v našem případě číslo enkodéru uvedené v tabulce 2-11, druhý datový *byte* přenáší nově nastavenou hodnotu. Protokol Mackie Control si však data ve druhém datovém *bytu* přizpůsobil svým potřebám. Experimentální analýzou byly zjištěny tyto posílané zprávy.

Tabulka 2-12 Analýza komunikace rotačních enkodérů

Událost	Pohyb	Stavový byte	1. datový byte	2. datový byte	2. datový byte binárně
Pohyb enkodéru 1 doleva	o jeden krok	$B0	$10	$41	0100 0001
	nejrychleji	$B0	$10	$4F	0100 1111
Pohyb enkodéru 1 doprava	o jeden krok	$B0	$10	$01	0000 0001
	nejrychleji	$B0	$10	$0F	0000 1111

Stavový *byte* $B0 nese informaci, že se jedná o zprávu typu Control Change na prvním MIDI kanále. První datový *byte* $10 nese číslo kontroleru, což je v decimální soustavě číslo 16, které odpovídá Enkodéru 1. Z druhého datového *bytu* toho v hexadecimální soustavě nelze příliš vyčíst, ale když si číslo převedeme do binární soustavy, stane se princip zakódování zřetelnější. MSB *bit* neboli *bit b7*, je v případě datového *bytu* nastaven vždy na 0, proto zůstává vždy stejný. U následujícího bitu *b6* si můžeme všimnout, že je při pohybu enkodéru doleva nastaven na 1 a při pohybu doprava na 0. Tímto bitem je tedy přenášen směr pohybu enkodéru. Bity *b5* a *b4* nejsou využity a jsou nastaveny na hodnotu 0. Při otáčení také hraje roli rychlost. Při pomalém pohybu se nastavovaný parametr v DAW mění pomalu a jsme tedy schopni dělat jemné změny. Když chceme udělat velkou změnu, například změnit panoramu z hodnoty L100 na R100, rychlým pohybem enkodéru dojde ke změně po větším kroku. V tabulce 2-12 můžeme vidět krajní případy pro nejnižší a nejvyšší rychlost otáčení enkodéru. Z binárního zápisu je zřejmé, že je rychlost přenášena *bity b3* až *b0*. Rozsah je tedy v decimální soustavě od 0 do 15. Čip kontroleru musí tedy nejprve vyhodnotit počet kroků za určitý časový okamžik a teprve poté odeslat zprávu Control Change.

Detailnější analýzou bylo zjištěno, že i když je rychlost otáčení kódována do 4 *bitů*, není využito všech 16 stavů, ale jen 5. Přesný výpis je uveden v tabulce 2-13. Z binárního zápisu si můžeme ověřit, že je rychlost kódována shodně pro oba směry otáčení, doleva i doprava.

Tabulka 2-13 Kroky rychlosti enkodéru kontroleru Mackie MCU

Směr pohybu doleva		Směr pohybu doprava	
2. datový byte dekadicky	2. datový byte binárně	2. datový byte dekadicky	2. datový byte binárně
1	0000 0001	65	0100 0001
4	0000 0100	68	0100 0100
8	0000 1000	72	0100 1000
12	0000 1100	76	0100 1100
15	0000 1111	79	0100 1111

2.7 Zobrazování na displeji

Displej je na kontroleru velmi důležitý, protože nám pomáhá s orientací v kanálech a dává nám zpětnou vazbu k nastavovaným parametrům. V závislosti na použitém DAW a režimu enkodérů se ne displeji mohou zobrazovat jména kanálů, úrovně zesílení, úrovně posílané do pomocných sběrnic, nastavení panoramy, parametry plugin efektů a jiné. Displej se může vázat k jednotlivým kanálům nebo může být využit jako celek například při vkládání nového plugin efektu za použití kontroleru. Displej kontroleru Mackie MCU umožňuje zobrazení 55 znaků na řádek, přičemž zobrazuje celkem 2 řádky. Pro oddělení popisu mezi kanály se nechává jeden znak prázdný. Celkem tedy na jeden kanál připadá 6 znaků a 1 prázdný znak. Prázdný znak je v kódu ASCII zastoupen zobrazením mezery, která má ASCII kód $20. Za osmým kanálem je prázdný znak vynechán, proto dojdeme k celkovému počtu 55 znaků na řádek.

Zobrazení na displeji má na starosti DAW software. Znaky, které mají být zobrazeny, se posílají pomocí zvláštních systémových zpráv – System Exclusive. Jelikož je displej pouze zobrazovací prvek, je typ komunikace jednosměrný a neočekávají se tedy žádné zprávy směrem do DAW. V tabulce 2-14 můžeme vidět vzor zprávy pro požadavek na přepsání jména kanálu, obsahující 7 znaků.

Tabulka 2-14 Vzor zprávy pro zobrazení na displej

$ (F0 00 00 66 14 12 00 50 69 61 6E 6F 20 20 F7)

Analýza byla provedena postupným přejmenováním kanálů a byla hledána podobnost jednotlivých zpráv a závislost na pozici zobrazení. Po nalezení podobnosti a jejím rozklíčování můžeme celou zprávu rozdělit na dílčí části, jak uvádí tabulka 2-15.

Tabulka 2-15 Dílčí části zprávy zobrazení na displej

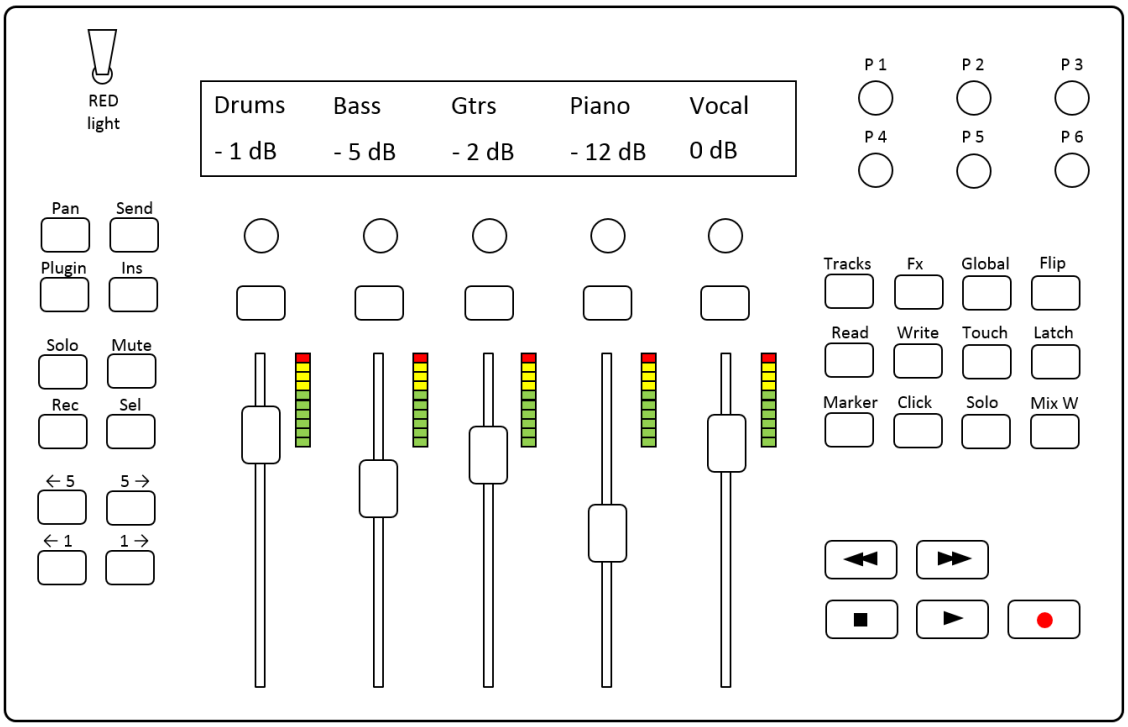
Zpráva System Exclusive v hexadecimálním tvaru	Význam jednotlivých *bytů*
F0	Začátek zvláštních systémových zpráv
00 00 66	3 *bytový* ID výrobce – Mackie
14	Device ID
12	Identifikátor zobrazení na LCD displej
00	Pozice kurzoru – první znak displeje
50 69 61 6E 6F 20 20	7 znaků ASCII – text „Piano “
F7	Konec zvláštních systémových zpráv

Jak již bylo popsáno, *byte* $F0 slouží k identifikaci začátku zvláštních systémových zpráv. Následující *byte* slouží jako identifikátor výrobce. Jelikož je hodnoty $00 víme, že identifikátor výrobce není 1 *bytový*, ale 3 *bytový* a včetně následujících dvou *bytů* ($00 $00 $66) identifikuje výrobce Mackie. Další *byte* v pořadí ($14) slouží k identifikaci přístroje a stejně jako předchozí *byty* je pro každou analyzovanou zprávu shodný. Nyní už následuje čistý blok dat. Pokud se má text zobrazit na LCD displej, je použit identifikátor $12. Dále je potřeba vědět, na jakou pozici se budou znaky zobrazovat. *Byte* $00 znamená, že se začne s výpisem od prvního možného místa, na displeji tedy hned vlevo nahoře. Další *byty* už nesou znaky v kódu ASCII. Může jich být libovolný počet, ale jen takový, abychom nepřesáhli konec displeje.

Při spuštění DAW dojde k inicializaci, a tedy přepsání všech 112-ti znaků. I když tedy displej fyzicky obsahuje jen 110 znaků, je zřejmě pro jednoduchost přiděleno 7 znaků na kanál i když osmý kanál má jen 6 znaků. I když je tedy tento znak poslán, nebude nikdy zobrazen. Konkrétně se jedná o 56. a 112. znak. Celkové přepsání kromě inicializace proběhne i při změně na jinou vrstvu, protože pracujeme s dalšími kanály a je potřeba přenést všechny jejich jména. Pokud je potřeba přepsat jen část displeje, například jméno kanálu, je zbytečné aktualizovat celý displej a dojde tedy k přepsání jen sedmi znaků, jak je uvedeno ve vzoru v tabulce 2-14 a 2-15. Indexace pozice kurzoru je číslována od nuly. Pozice na prvním řádku jsou tedy v rozsahu hodnot 0 až 55, na řádku druhém pak 56 až 111.

3. NÁVRH KONTROLERU ZALOŽENÝ NA PROTOKOLU MACKIE CONTROL

Jelikož je návrh kontroleru založen také na protokolu Mackie Control, bylo nejprve nutné provést analýzu a zjistit, co tento protokol nabízí a umožňuje. Také bylo třeba vyzkoušet, co vše se dá kontrolerem ovládat, co dokáže zobrazovat a jaký je styl práce a ovládání kontroleru. Na základě získaných zkušeností a potřeb byl navržen kontroler, uveden na obrázku 3-1, který bude obsahovat následující prvky a bude sloužit pro rychlý přístup k nejpoužívanějším parametrům.



Obr. 3-1: Rozmístění ovládacích prvků na kontroleru

3.1 Možnosti kontroleru, ovládací prvky a jejich rozložení

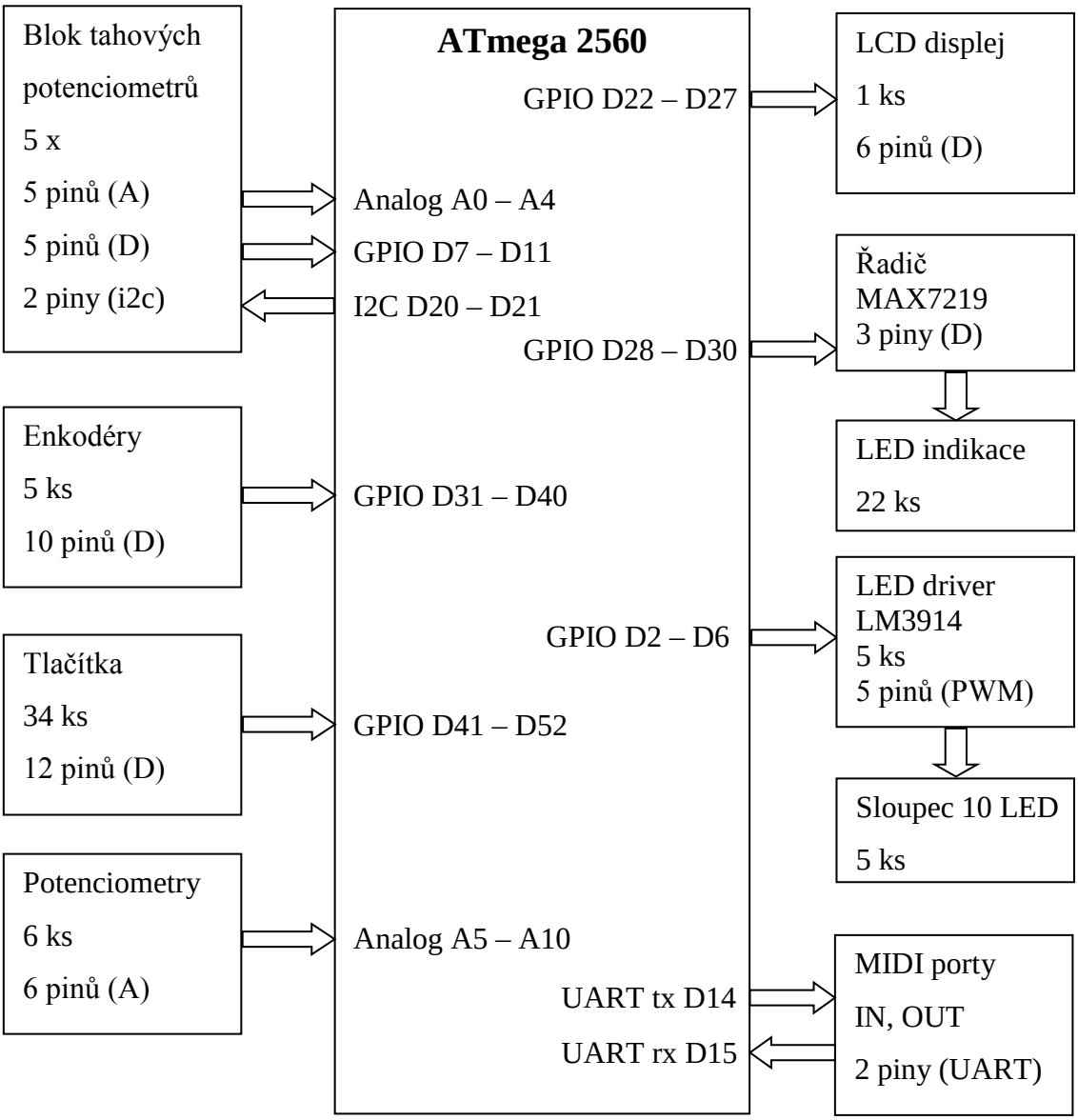
Kontrolerem je možné ovládat pět kanálů současně, s možností posuvu na následující nebo předchozí banku kanálů, tedy posun o pět kanálů doprava nebo doleva. Posun je také možný po jednotlivých kanálech, což je vhodné, když chceme mít v jeden okamžik pod kontrolou určité sousední kanály. Volba se bude provádět tlačítky po levé straně. Každý kanál obsahuje motorizovaný tahový potenciometr o délce 10 cm se snímáním informace o doteku. Nad tahovými potenciometry je umístěno tlačítko, jehož režim je volen rovněž tlačítky po levé straně, a bude nahrazovat tlačítka Record, Solo, Mute a Select kvůli ušetření místa. Každý kanál má také svůj enkodér s možností přepínaní režimů, mezi které bude patřit nastavení panoramy (Pan), úrovně posílané do efektových sběrnic (Fx), nastavení parametrů vložených plugin efektů (Plugin) a jejich vypnutí (Ins). Nad enkodéry je umístěn displej zobrazující názvy kanálů a parametry nastavované enkodéry. Podle původního návrhu měl kontroler obsahovat čtyři tahové potenciometry pro nastavování kanálů a jeden tahový potenciometr pro řízení úrovně hlavní sběrnice. Jelikož se však nevyrábí LCD displej s potřebným počtem znaků na řádek (28 znaků), bylo nutné zvolit delší displej (40 znaků). Návrh kontroleru byl tak upraven na pět kanálů bez řízení úrovně hlavní sběrnice. Vedle tahových potenciometrů jsou navíc umístěny indikátory vybuzení, obsahující sloupec deseti LED diod.

Na pravé straně kontroleru je umístěno šest potenciometrů, jejichž funkce bude volena uživatelem. Jeden z nich tedy může sloužit například pro nastavení úrovně hlavní sběrnice, ostatní mohou ovládat například parametry virtuálních hudebních nástrojů. Pod potenciometry je umístěna sekce dvanácti tlačítek. Na prvním řádku budou tlačítka, týkající se zobrazení určitých typů kanálů. Stiskem tedy bude možné zobrazit kanály projektu (Tracks), návratové kanály efektových sběrnic (Fx), případně všechny typy kanálů současně (Global). Posledním tlačítkem (Flip) prohodíme funkci tahových potenciometrů a enkodérů. To je vhodné v případech, kdy chceme například automatizovat změnu úrovně do efektové sběrnice. Tahový potenciometr nám tak umožní provést plynulejší pohyb. Na dalším řádku se nachází sekce tlačítek týkající se již zmiňované automatizace. Prvním tlačítkem (Read) zapneme čtení již zaznamenaného průběhu automatizace. Dalšími tlačítky (Write, Touch, Latch) volíme způsob zápisu automatizace. Na posledním řádku jsou umístěna tlačítka pro rychlý přístup. První tlačítko (Marker) slouží k posunu po kurzorech za použití tlačítek přetáčení. Následujícím tlačítkem (Click) zapneme metronom a dalším (Solo) jednorázově deaktivujeme funkci solo. Poslední tlačítko (Mix W) slouží k vyvolání okna mixážního pultu v DAW prostředí. Součástí ovládacích prvků jsou rovněž tlačítka transportu (Play, Stop, Rec a dvě tlačítka pro přetáčení). Posledním ovladačem je páčkový přepínač, který povolí rozsvěcování červeného světla při spuštění nahrávání. Nahrávané osoby tak budou informováni, kdy nahrávání probíhá. Funkce tlačítek odpovídá specifikaci pro použití s DAW Studio One [5].

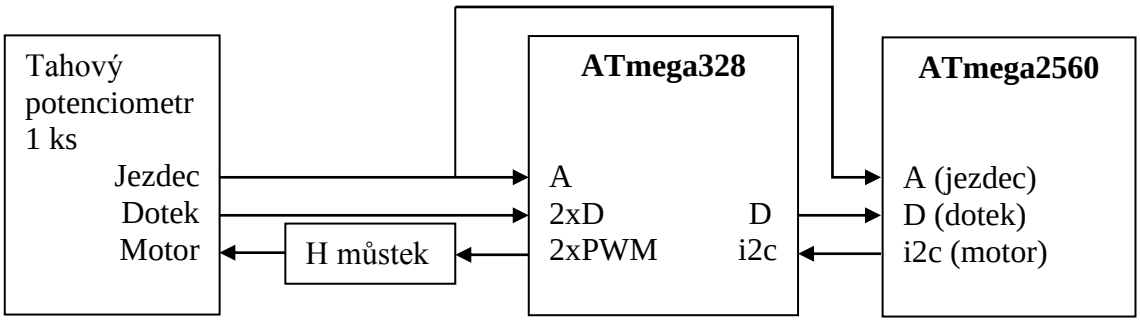
3.2 Blokové schéma zapojení

Zapojení kontroleru je postaveno na mikročipu ATmega2560. Tento mikročip nabízí potřebný počet vstupů, výstupů a také sériových sběrnic. Celkem obsahuje 16 analogových vstupních pinů a 54 vstupně výstupních digitálních pinů [6]. Blokové schéma celkového zapojení je znázorněno na obrázku 3-2. Led indikace jednotlivých tlačítek je zajištěna řadičem MAX7219, abychom hlavní program nezatěžovali nepřetržitým vykreslováním, kdyby byly LED diody zapojeny na principu mřížky. Sloupce LED diod pro zobrazení vybuzení řídí obvod LM3914, na jehož vstup je poslán PWM modulovaný signál. Spojení s DAW je zajištěno přes rozhraní UART, přes které jsou přenášeny příchozí a odchozí MIDI zprávy. Potenciometry jsou připojeny přímo do analogových vstupů, enkodéry do vstupů digitálních. Tlačítka jsou rovněž připojena do digitálních vstupů a zapojena na principu mřížky.

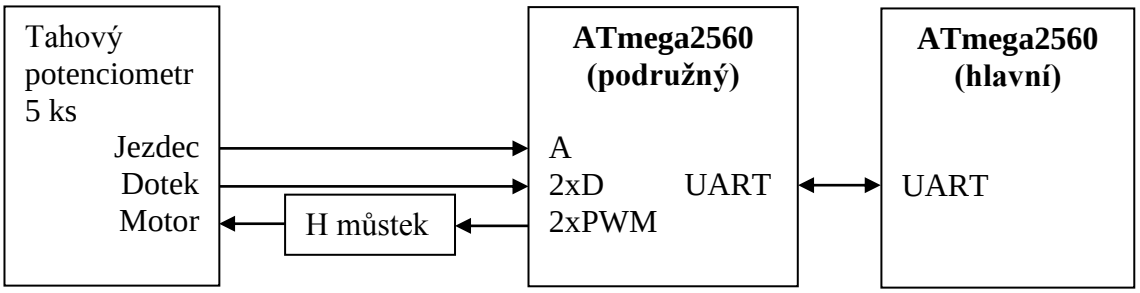
Nejrozsáhlejší blok tvoří blok motorizovaných tahových potenciometrů. Podle původního návrhu, na obrázku 3-3, měl mít každý tahový potenciometr svůj mikročip, který by se staral o řízení motoru a dopravil tak jezdce na stanovenou pozici. Všechny mikročipy by byly připojeny přes sběrnici i2c. Když by přišel z DAW pokyn na změnu pozice, hlavní mikročip by na adresu jednoho podružného mikročipu poslal hodnotu, do které má jezdce dopravit. Při pěti individuálních mikročipech by tak byl pohyb zcela nezávislý a oddělený. Návrh byl však přehodnocen a přišlo v úvahu řešit celou sekci tahových potenciometrů jedním mikročipem, a to opět mikročipem ATmega2560. Tento podružný mikročip zajišťuje jak čtení pozice jezdce na analogovém vstupu, tak vyhodnocení doteku na vstupech digitálních, a posílání těchto hodnot po sběrnici i2c nebo sběrnici UART. Mikročip také zajišťuje pohyb všech motorů tahových potenciometrů, připojených přes H můstky. Ze schématu na obrázku 3-2 je tak celý blok tahových potenciometrů připojen k hlavnímu mikročipu přes jedinou sběrnici, jak je znázorněno na obrázku 3-4.



Obr. 3-2: Blokové schéma zapojení kontroleru



Obr. 3-3: Zapojení bloku tahových potenciometrů s jedním mikroprocesorem pro každý potenciometr



Obr. 3-4: Zapojení bloku tahových potenciometrů s podružným mikroprocesorem

4. KONSTRUKCE A PROGRAMOVÁNÍ PRVKŮ KONTROLERU

Tato část práce se věnuje jednotlivým prvkům kontroleru, principu, na kterém pracují, způsobech zapojení do obvodu a vyhodnocením jejich aktivity. Práce s těmito prvky je základním kamenem při stavbě kontroleru. Když prvek funguje a program správně vyhodnocuje jeho aktivitu, můžeme relativně snadno kontroler rozšířit o libovolný počet těchto prvků a tím si jej přizpůsobit, s ohledem na počet volných vstupů a výstupů. Protože jednotlivé prvky předem odladíme, známe přesně jejich požadavky pro správnou funkci. Pro vyhodnocení aktivity prvků je použita vývojová deska Arduino Mega, s mikročipem Atmel ATmega2560.

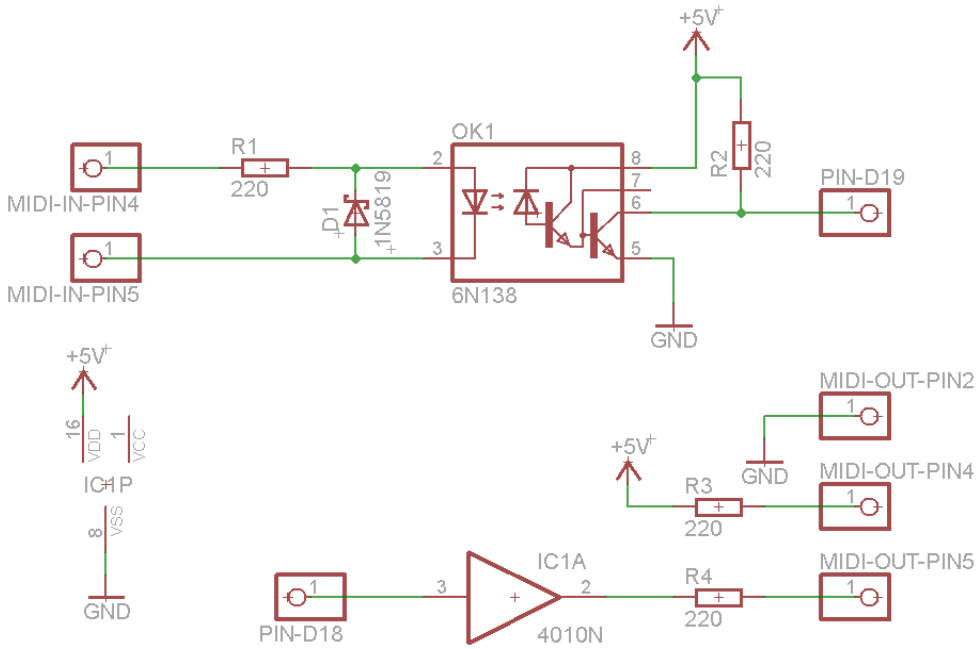
Obr. 4-1: Vývojová deska Arduino MEGA 2560 [6]

Tato deska disponuje USB portem, díky kterému se dá mikročip snadno naprogramovat a také umožňuje komunikaci s programovacím prostředím pomocí sériové sběrnice UART (Universal Asynchronous Receiver-Transmitter). Díky této komunikaci si můžeme na monitor počítače nechat vypisovat zprávy, abychom měli přehled, co mikročip vyhodnocuje, případně v jaké části programu se nachází. K programování se používá programovací jazyk Wiring, který se podobá jazyku C. Práce s tímto jazykem je popsána v [7].

4.1 MIDI rozhraní

Aby bylo možné testovat, jak DAW software na ovládací prvky reaguje, rozhodl jsem se jako první začít s připojením MIDI rozhraní. Čip jako takový MIDI rozhraním nedisponuje, nabízí nám ale čtveřici sériových portů UART, které je možné pro vysílání a přijímání MIDI zpráv využít. Je zde ale nutné rozšířit část hardware a nastavit pro sběrnici UART správné parametry přenosu.

MIDI sběrnice je proudová smyčka, kterou při logické nule protéká proud 5 mA. Při logické jedničce je proud smyčkou nulový. Datový přenos je asynchronní s rychlostí 31,25 kBaudů [2]. Při návrhu zapojení jsem se inspiroval schématem ze zdroje [2], které jsem mírně upravil. V části MIDI vstupu jsem optoizolátor PC900, který se již neprodává, nahradil optoizolátorem 6N138 a použil rychlou Schottkyho diodu 1N5819. Tato dioda je zapojena v opačném směru než dioda v optoizolátoru a v případě přepólování přes ni protéká proud. Dle katalogového listu [8] má optoizolátor maximální vstupní proud 20 mA a má tedy pro použití v MIDI sběrnici dostatečnou rezervu. V části MIDI výstupu jsem dva invertující logické budiče sběrnice nahradil jedním neinvertujícím, princip tedy zůstane zachován. K tomu jsem použil obvod HCF4010BE, který disponuje šesti budiči, z nichž jen jeden je využitý. Vstupy ostatních budičů jsem připojil na zem. Dále si můžeme všimnout, že stínění MIDI kabelu je připojeno jen na výstupním konektoru. Je to z důvodu zamezení zemních smyček mezi zařízeními, protože fyzicky nedojde ke spojení jejich zemí. Konečné schéma zapojení rozhraní můžeme vidět na obrázku 4-2.



Obr. 4-2: Schéma zapojení MIDI rozhraní

Pro testování rozhraní jsem použil převodník Roland UM-ONE, který převádí MIDI na USB. Abych si ověřil, zda rozhraní funguje správně, napsal jsem jednoduchý program na vysílání zpráv nota zapnuta, uveden v příloze č. 1 program 1. Co se týče nastavení sběrnice UART, je potřeba sběrnici aktivovat příkazem *begin* a nastavit ji přenosovou rychlost shodnou s rychlostí MIDI sběrnice, tedy 31250 Baudů. Program se skládá ze třech podprogramů. Podprogram s názvem *setup* se provede jen jednou při spuštění případně resetování mikročipu a slouží obvykle k nastavení vstupů a výstupů. Podprogram *loop* je smyčka, ve které se nachází hlavní program, který se vykonává stále dokola. V tomto testovacím programu tedy dojde k zahrání rozloženého akordu C dur a následném vypnutí not, kterého docílíme vysláním nota zapnuta s dynamikou nula. V DAW Cubase si vytvoříme MIDI stopu a budeme tyto zprávy zaznamenávat. DAW zprávy přijímá a akord zní, vysílání MIDI zpráv tedy funguje správně.

Nyní je třeba ověřit, zda funguje také přijímání MIDI zpráv přes optoizolátor. K tomu použijeme druhý testovací program, uveden v příloze č. 1 program 2. Tento program v části *setup* obsahuje také aktivaci sériové komunikace s počítačem, aby bylo možné příchozí MIDI zprávy zobrazovat. Funkce programu je jednoduchá. Pokud přijde MIDI zpráva, zobrazí *byty* této zprávy na monitor pomocí příkazu *println* s funkcí následného posunu kurzoru na další řádek. Nahranou MIDI stopu z prvního testovacího programu tedy necháme přehrávat a pošleme ji na MIDI výstup počítače. Mikročip zprávy přeposílá a máme tedy ověřeno, že komunikace po stránce hardware funguje oběma směry.

4.1.1 MIDI knihovna

Jelikož MIDI zprávy pro práci s kontrolerem obsahují také další typy zpráv jako změna kontroleru, ohýbání tónu a další, je vhodné s těmito zprávami pracovat na vyšší úrovni. K tomu nám poslouží knihovna MIDI, kterou je možné stáhnout z internetových stránek Arduina [9] a do programovacího prostředí nainstalovat. Návod na práci s touto knihovnou je uveden například v [10]. V programovacím prostředí je nejprve nutné knihovnu MIDI do programu zahrnout pomocí příkazu *include*. Dále vytvoříme instanci, kde definujeme sériový port, který budeme pro sběrnici MIDI používat a přiřadíme mu jméno, v našem případě MIDI. V části *setup* port inicializujeme pomocí příkazu *begin*, kde zároveň určíme, zda chceme přijímat pouze určitý MIDI kanál, nebo všechny v režimu *OMNI*. V tabulkách 4-1 a 4-2 jsou uvedeny vybrané příkazy pro vysílání a přijímání MIDI zpráv. Mimo těchto příkazům umožňuje také knihovna pracovat se zprávami System Exclusive [10].

Tabulka 4-1 Příkazy pro odchozí MIDI zprávy [10]

Příkaz	1. parametr	2. parametr	3. parametr
MIDI.sendNoteOn()	číslo noty	dynamika	kanál
MIDI.sendNoteOff()	číslo noty	dynamika	kanál
MIDI.sendProgramChange()	číslo programu	kanál	-
MIDI.sendControlChange()	číslo kontroleru	hodnota	kanál
MIDI.sendPitchBend()	hodnota	kanál	-
MIDI.sendPolyPressure()	číslo noty	tlak	kanál
MIDI.sendAfterTouch()	tlak	kanál	-

Tabulka 4-2 Příkazy pro příchozí MIDI zprávy [10]

Příkaz	Popis
MIDI.read()	Vrací log. 1 pokud je k dispozici příchozí zpráva, zároveň tuto zprávu přečte a vyjme ze zásobníku
MIDI.getType()	Vrací typ MIDI zprávy
MIDI.getData1()	Vrací hodnotu prvního datového *bytu*
MIDI.getData2()	Vrací hodnotu druhého datového *bytu*
MIDI.getChannel()	Vrací číslo kanálu, na kterém byla zpráva přijata

Díky přednastaveným typům zpráv není nutné počítat vysílaný stavový byte MIDI zprávy ručně. Zakódování obstará knihovna a program se díky tomu také zpřehlední. Co se týče přijímání MIDI zpráv, příkazem *read* příchozí zprávu přečteme a můžeme zjistit její typ, hodnoty datových *bytů* a použitý MIDI kanál. Každý sériový port mikročipu má svůj zásobník, který je schopen uchovat až 64 *bytů* [7]. Příkazem *read* dojde k přečtení a vyjmutí jednoho *bytu* ze zásobníku, čím současně uvolní místo dalšímu *bytu*. V případě použití MIDI knihovny příkaz *read* přečte celou MIDI zprávu, které v našem případě budou obvykle odpovídat 3 *byty*.

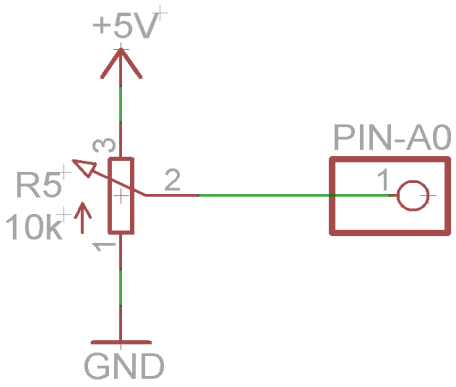
Z hlediska programování je vhodné v případě přijímání zpráv tyto zprávy třídit podle typu. K tomu použijeme přepínač *switch*. V příloze č. 1 program 3 je uveden testovací program pro práci s knihovnou. Program obsahuje potřebné nastavení pro správnou funkci knihovny, dále obsahuje v části *setup* vyslání zprávy Nota zapnuta, abychom ověřili vysílání zpráv. V části *loop* poté obsahuje přepínač *switch*, který přepíná do třech možností podle typu zprávy a podle něj vypíše na monitor text, o jakou zprávu se jedná.

V případě programování kontroleru bude hlavní program rozdělen na dvě části. První část bude vysílací, kdy bude vyhodnocena a odeslána aktivita jeho prvků. Druhá část hlavního programu bude přijímací, kdy na základě přijatých dat z DAW budou tato data na kontroleru zobrazena, nebo se případně uloží do proměnných.

Testovací program, uveden v příloze č. 3 program 3, který vychází z [10], bude tedy následně upravován a rozšiřován až do konečné podoby jako firmware kontroleru.

4.2 Lineární potenciometr

Lineární potenciometr je obvodový prvek s odporovou dráhou, jejíž závislost odporu na poloze jezdce je lineární funkce. Do obvodu jej zapojíme tak, že konce dráhy připojíme mezi napájecí svorky, tedy +5 V a GND, a jezdce připojíme na analogový vstup mikročipu. Hodnotu potenciometru volíme 10 kΩ, ze zdroje bude tedy podle Ohmova zákona odebírat 0,5 mA.



Obr. 4-3: Schéma zapojení potenciometru k mikročipu

Na výstupu jezdce se nám tedy může objevit napětí v celém rozsahu napětí napájecího, k čemu je také analogový vstup čipu dimenzován. Hodnota napětí bude převedena do digitální podoby na 10-ti bitové číslo. Jelikož budeme pro přenos informace předpokládat typ zprávy Control Change, budeme muset toto číslo převést na 7-mi bitové.

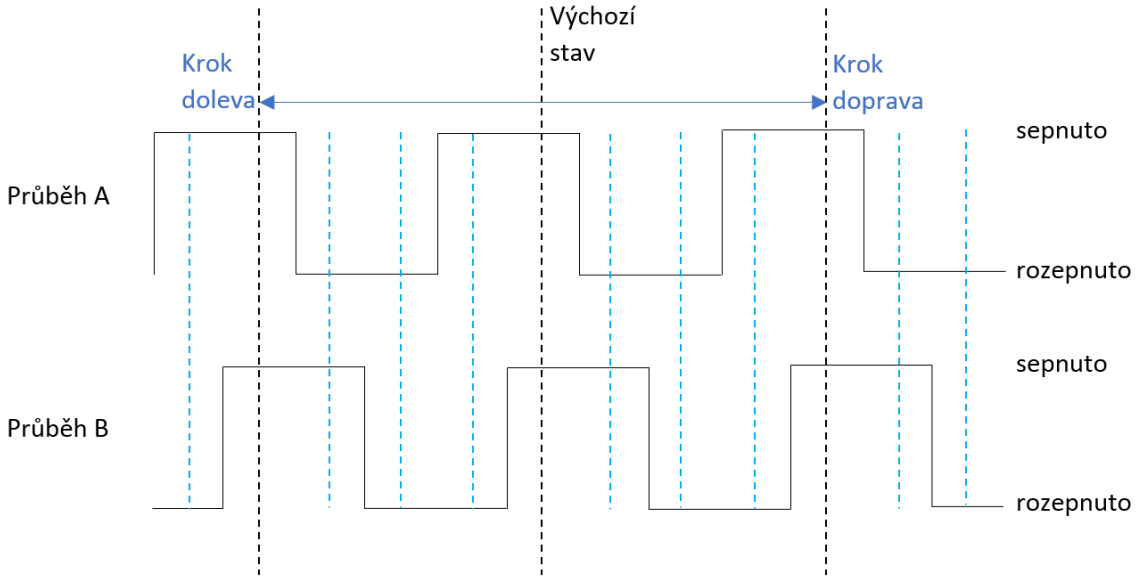
Z hlediska chodu programu není nutné, aby byl údaj o poloze přenášen neustále. Stačí jej přenést na počátku a poté až při jeho případné změně na novou hodnotu. V programu tedy vyhradíme globální proměnnou *potCClast*, kde si budeme ukládat jeho poslední hodnotu. Jelikož budeme zprávy vysílat na základě změny hodnot Control Change a ne přímo na změně hodnoty výstupu A/D převodníku, je vhodné do této proměnné ukládat přímo číslo v rozsahu 0 až 127. Ve smyčce *loop* tedy nejprve přečteme hodnotu analogového vstupu A0, následně ji převedeme do rozsahu 0 až 127 pomocí funkce *map,* a poté porovnáme s uloženou hodnotou v proměnné *potCClast*. Pokud budou hodnoty rozdílné, dojde k vyslání zprávy Control Change s číslem

kontroleru 71 a jeho novou hodnotou. Tuto novou hodnotu následně uložíme do proměnné *potCClast*. V příloze č. 1 program 4 je tato část programu uvedena. Abychom program přizpůsobili následným úpravám a změnám, budeme všechny vstupy mikročipu přiřazovat pomocí příkazu *define* v úvodní části programu. Díky tomu bude možné rychle změnit použitý vstupní pin za jiný, při zachování stávajícího programu.

Pro ověření funkčnosti budeme v DAW Cubase používat rozhraní obecného vzdáleného řízení pomocí Generic Remote. Zde můžeme naší zprávě Control Change s číslem kontroleru 71 přiřadit libovolnou funkci, například měnit úroveň hlasitosti hlavní sběrnice. Funkci řízení nastavíme příznakem Receive. Generic Remote také umožňuje, podobně jako protokol Mackie Control, odesílat zprávy do kontroleru při změně parametru v DAW (příznak Transmit). Jelikož ale nemáme možnost potenciometr do žádané polohy dopravit, nemá v tomto případě smysl tyto zprávy z DAW vysílat. Zpětné vysílání by v případě použití potenciometru mělo smysl tehdy, pokud bychom hodnotu zobrazovali na displeji a docházelo by k její změně, například vlivem automatizace. Abychom zabránili skokové změně parametru při kolizi polohy, nabízí Generic Remote funkci přichycení (příznak Pick-up), kdy potenciometr začne parametr ovládat až překryje jeho aktuální hodnotu. Použití potenciometru je tedy vhodné pro parametry, které nebudeme měnit přímo v DAW, například pro parametry virtuálních hudebních nástrojů případně pro nastavení hlasitosti poslechu. Výhodou potenciometru je jeho nižší cena a snadné navýšení počtu.

4.3 Rotační enkodér

Popis funkce rotačního enkodéru je uveden v kapitole 2.6, zde se zaměříme pouze na jeho princip a vyhodnocení směru otočení. Pro náš kontroler budeme používat enkodéry EC11-1S, které disponují integrovaným tlačítkem. Tento enkodér má 5 vývodů, z nichž 2 slouží jako kontakty integrovaného tlačítka a zbylé 3 jsou určeny pro vyhodnocení otáčení. Z hlediska principu funkce si můžeme enkodér představit jako dva spínače s vývody *A* a *B* [11]. Třetí vývod *C* je pro oba spínače společný a obvykle se připojuje na nulový potenciál. Při pohybu enkodéru dochází ke spínání a rozpínání těchto spínačů. Jelikož jsou signály *A* a *B* mezi sebou posunuty o čtvrtinu fáze, dostáváme celkem čtyři možné kombinace jejich stavů. Na základě znalosti předchozího stavu spínačů můžeme poté vyhodnotit směr otočení, viz obrázek 4-4.



Obr. 4-4: Průběh signálů A a B rotačního enkodéru

Použité enkodéry jsou navíc krokované a otočení hřídele o 360° odpovídá 20-ti krokům. Testováním bylo zjištěno, že za jeden krok projdou spínače všemi čtyřmi stavy a dostanou se tak opět do výchozího stavu. Kdybychom odesílali zprávy po každé změně kombinace spínačů, jednomu kroku by odpovídaly čtyři zprávy, čímž bychom se v DAW prostředí ochuzovali o přesnost. Z hlediska programování je tedy vhodné nastavit podmínku tak, aby byla vyhodnocena vždy při jednom kroku enkodéru.

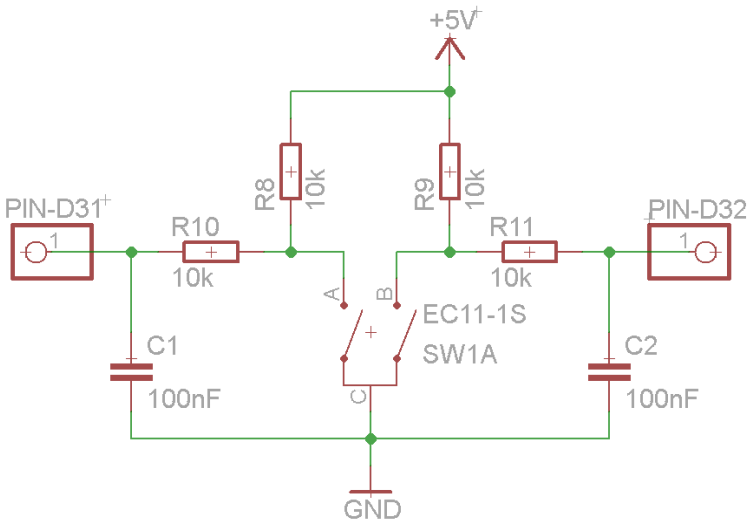
Z průběhů na obrázku 4-4 si můžeme všimnout, že ve výchozím klidovém stavu enkodéru jsou oba spínače sepnuty. Na digitálních vstupech tedy dostáváme logické nuly, protože je vstup čipu přes tyto spínače přiveden na nulový potenciál. Sepnutému stavu tedy odpovídá logická nula, rozepnutému stavu logická jednička.

V programu si do globální proměnné uložíme předchozí stav spínače *A*, tedy nulu, a testujeme, zda nedošlo k jeho změně. Pokud ano, testujeme stav spínače *B*. V případě že je stav *B* shodný se stavem *A*, došlo k pohybu doleva. Pokud je stav opačný, došlo k pohybu doprava. Nyní je potřeba aktualizovat proměnnou s posledním stavem *A*. Tímto jsme zredukovali počet vyslaných MIDI zpráv na dvě za jeden krok otočení, protože proměnná s posledním stavem *A* se za jeden krok změní dvakrát. Z nuly na jedničku a opět z jedničky na nulu. Podmínku tedy doplníme o další pravidlo tak, aby byla splněna, pokud je zároveň aktuální stav *A* roven jedné. Jelikož je nyní celková podmínka splněna jen jednou za dobu otočení o jeden krok, připadá jednomu kroku právě jedna MIDI zpráva. Vysílaná MIDI zpráva bude typu Control Change s číslem kontroleru 0 a hodnotou 1 v případě pohybu doleva, nebo hodnotou 127 v případě pohybu doprava. V rozhraní Generic Remote v DAW prostředí můžeme této zprávě opět přiřadit libovolnou funkci, například nastavení panoramy. Jelikož se jedná o relativní způsob přenosu informace, je nutné potvrdit příznak Relative. Tento příznak

daný parametr dekrementuje, pokud je datový *byte* v rozsahu 1 – 63, případně inkrementuje, pokud je v rozsahu 64 – 127.

Z hlediska hardware jsou spínače *A* i *B* připojeny jako běžná tlačítka. Abychom měli na vstupním pinu jednoznačný stav logické úrovně, je vstup připojen přes rezistor 10 kΩ na napájecí napětí + 5V a současně přes tlačítko případně spínač na zem, tedy 0 V. Pokud je kontakt tlačítka rozepnut, je na vstupu logická jednička, pokud je kontakt sepnut, je na vstupu logická nula a přes rezistor protéká proud. Tento proud je však díky velké hodnotě rezistoru malý, podle Ohmova zákona pouze 0,5 mA.

Zapojením tohoto obvodu s naprogramovaným mikročipem bylo však zjištěno, že vyhodnocení aktivity neprobíhá ideálním způsobem. Při pohybu enkodéru doprava byly místy vyslány nežádoucí zprávy o pohybu doleva. Také počtu kroků enkodéru neodpovídal počet odeslaných MIDI zpráv, těch bylo násobně více. Podle chování obvodu jsem usoudil, že bude chyba v části hardware. Při přechodu spínačů ze sepnutého do rozepnutého stavu, a naopak, docházelo k náhodným stavům, než se spínač ustálil. Jelikož čip vyhodnocuje data na vstupech velmi rychle, vyhodnotil tyto přechodové stavy jako několik otočení různými směry. Abychom tento jev eliminovali, bylo nutné do obvodu doplnit kondenzátory 100 nF, které tyto rychlé změny stavů svou kapacitou odfiltrují. Kondenzátor je ke každému spínači připojen paralelně přes rezistor 10 kΩ. Na vstup čipu je poté připojeno napětí právě z tohoto kondenzátoru, které nemá skokové změny napětí. Při rozepnutí spínače enkodéru je kondenzátor postupně přes dvojici rezistorů nabit na 5 V. Ke změně napětí z 0 V na 5 V dochází postupně, ale zároveň dostatečně rychle na to, aby byl kondenzátor plně nabit před dalším sepnutím spínače. Při sepnutí je kondenzátor zkratován přes rezistor, čím je zároveň chráněn proti velkému proudu a také k jeho vybíjení dochází postupně. Autor tohoto zapojení je Kevin Darrah [12]. Výsledkem této úpravy je plynulá změna stavů a správné vyhodnocení otočení. Část programu pro práci s enkodéry je uvedena v příloze č. 1 program 5 a konečné schéma zapojení na obrázku 4-5.

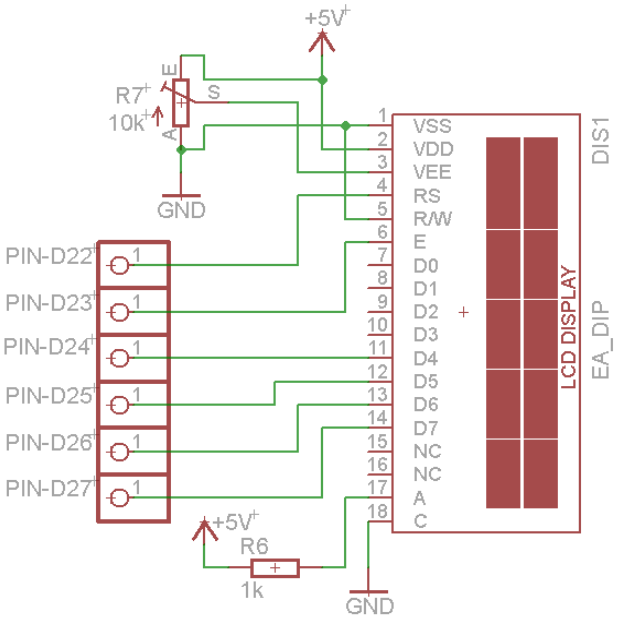


Obr. 4-5: Konečné schéma zapojení rotačního enkodéru [12]

4.4 LCD displej

K informování uživatele o hodnotách parametrů a zobrazování zpráv používáme obvykle LCD displej. Díky své univerzálnosti je zřejmě nejpoužívanější zobrazovací jednotkou. Displeje dělíme na několik typů, v našem případě budeme používat znakový monochromatický displej s podsvícením. Hlavním parametrem znakových displejů je počet znaků, které umožňují zobrazit. Počet se udává ve formátu *sloupce x řádky* a obvykle máme na výběr několik standardizovaných typů, například 16x2. Součástí znakových displejů bývá také řadič, který obsahuje znakovou sadu a stará se o vykreslení a zobrazení znaků.

K připojení displeje slouží 16 pinů. První dva piny jsou napájecí, na ně připojíme +5 V a 0 V. Napětím na třetím pinu řídíme kontrast. Zde pro získání napětí použijeme potenciometr, trimr nebo napěťový dělič. Z hlediska konstrukce je ideální volbou trimr. Nezabírá příliš místa a máme možnost kontrast kdykoli doladit. Další trojice pinů slouží k řízení, poté následuje osm datových pinů. Poslední dva piny jsou anoda a katoda podsvícení. Z katalogového listu displeje se dočteme, jaký maximální proud zde může protékat a podle něj vybereme hodnotu rezistoru, který připojíme do série. Změnou hodnoty rezistoru můžeme podsvícení regulovat na žádanou intenzitu, proud však nesmí překročit maximální proud. K mikročipu připojíme displej podle schématu na obrázku 4-6. Z řídících pinů slouží jeden pin k nastavení čtení nebo zápisu. Jelikož budeme do displeje jen zapisovat, nikoli z něj číst, připojíme pin napřímo k napětí 0 V. Z datových pinů nám dále budou k funkčnosti stačit pouze 4 piny, ostatní piny zůstanou nezapojeny [7].



Obr. 4-6: Schéma zapojení LCD displeje [7]

Co se týče programování, disponuje programovací prostředí již v základu knihovnou pro práci s LCD displeji. Princip zobrazení spočívá ve dvou krocích. Nastavením kurzoru na místo, od kterého chceme znaky vypisovat, a následně posláním řetězce znaků řadiči, který znaky na displej vypíše. Po vypsání řetězce se kurzor posune na následující místo za posledním vypsaným znakem. Další poslaný řetězec by se tedy začal vypisovat od tohoto místa, případně můžeme kurzor nastavit jinam. Pokud se na daném místě už znaky nachází, dojde k jejich přepsání. V případě, kdy chceme změnit obsah celého displeje, je vhodné jej pomocí příkazu *clear* nejprve celý smazat a poté začít s vypisováním. Pokud však zobrazovaný text zůstává stejný a potřebujeme pravidelně měnit jen část obsahu displeje, například číslo, nastavíme vždy kurzor na shodné místo, a tím stávající číslo přepíšeme na nové. V tabulce 4-3 je uveden výběr používaných příkazů z knihovny Liquid Crystal [7]. První dva příkazy slouží k informování knihovny, k jakým pinům mikročipu je řadič připojen, a kolika sloupci a řádky disponuje. S následujícími příkazy už pracujeme za chodu programu a slouží k výše zmiňovanému smazání displeje, nastavení kurzoru a vypsání řetězce. Na počátku programu je však nutné nejprve knihovnu do programu zahrnout pomocí příkazu *include*. V příloze č. 1 program 6 je uveden testovací program pro ověření správné funkčnosti displeje.

Tabulka 4-3 Příkazy pro práci s knihovnou Liquid Crystal [7]

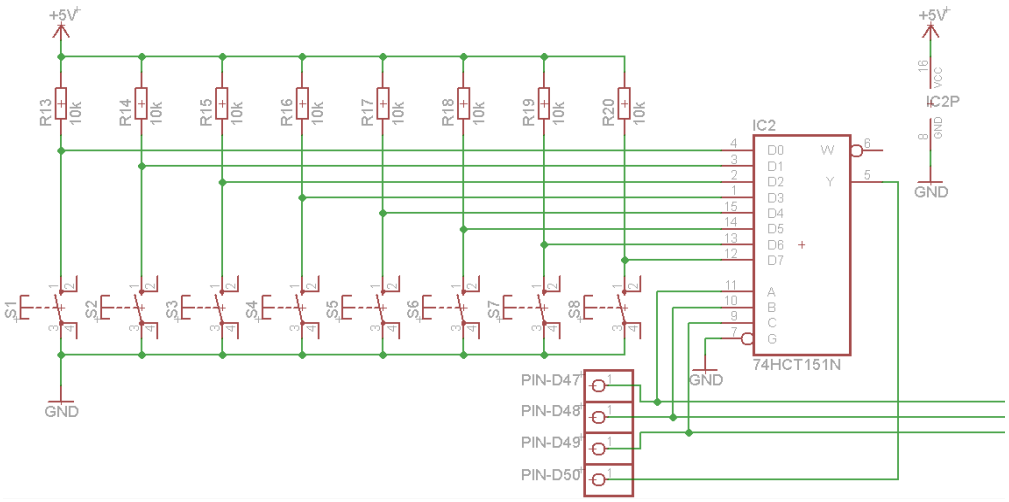
Příkaz	**Popis**
LiquidCrystal lcd (rs, e, d4, d5, d6, d7)	Vytvoří objekt lcd pro práci s displejem, parametrem jsou čísla pinů
lcd.begin(s,ř)	Zahájí práci s lcd, do parametru uvedeme počet řádků a sloupců, kterými displej disponuje
lcd.clear()	Smaže všechny znaky na displeji a nastaví kurzor do levého horního rohu
lcd.home()	Nastaví kurzor do levého horního rohu
lcd.setCursor(s,ř)	Nastaví kurzor na požadovanou pozici
lcd.print(data)	Vypíše na displej řetězec a posune kurzor o počet vypsaných znaků doprava

Pro použití displeje u kontroleru obvykle zobrazujeme popis parametru a pod ním hodnotu, kterou nastavujeme. Z hlediska počtu rotačních enkodérů pod displejem tak každému enkodéru náleží určitý počet znaků na displeji. Tyto znaky pojmenujeme jako sekci. Nejprve je nutné na první řádek displeje vypsat názvy parametrů do každé sekce. Jelikož na sebe sekce navazují, můžeme je sloučit a vyslat jako jeden řetězec. Na druhý řádek poté vypíšeme aktuální hodnoty parametru pro každou sekci. V případě pohybu enkodéru dojde ke změně hodnoty parametru v DAW, a tak je nutné také přepsat

zobrazovaný údaj. Zde budeme přepisovat jen sekce odpovídající hodnotám parametrů. Zároveň musíme před vysláním aktualizované hodnoty zabezpečit její délku, aby nepřesahovala do následující sekce a nedošlo tak k nechtěnému přepisu.

4.5 Tlačítkové spínače

Jak již bylo popsáno v kapitole 4.3., tlačítko připojujeme na vstup mikročipu zároveň s rezistorem, pro získání jednoznačné logické úrovně. Náš kontroler obsahuje celkem kolem třiceti tlačítek, a tak by bylo plýtváním, kdyby mělo každé tlačítko na mikročipu svůj vlastní vstup. Je tedy nutné počet vstupů nějakým způsobem zredukovat. Nejjednodušším řešením je zapojit tlačítka na principu mřížky. Pro třicet tlačítek se nabízí mřížka velikosti 5x6. Místo třiceti vstupů bychom tak obsadili jen jedenáct. Další možností je použití multiplexorů, pro které jsem se také rozhodl. Multiplexor je integrovaný obvod obsahující několik vstupů, v našem případě osm, a jeden výstup. Podle kombinace na řídících neboli adresových pinech pošle na výstup data z odpovídajícího vstupu. Zde pro připojení třiceti tlačítek postačí čtyři multiplexory. K připojení k mikročipu budeme potřebovat tedy jen sedm vodičů. Tři řídící, které budou společné pro všechny multiplexory, a čtyři datové, připojené k výstupům individuálních multiplexorů. Schéma zapojení jednoho multiplexoru můžeme vidět na obrázku 4-7. Pokračující adresové vodiče znázorňují připojení další trojice multiplexorů, z nichž každý multiplexor má svůj individuální výstup. Multiplexory budeme používat vysokorychlostní číslicové, typu CD74HCT151E, pracující přímo s logickými úrovněmi. Kromě řídících a datových vstupů obsahuje multiplexor také vstup *enable*, který musíme připojit na napětí 0 V, aby byl obvod aktivovaný.



Obr. 4-7: Schéma zapojení osmi tlačítek s multiplexorem

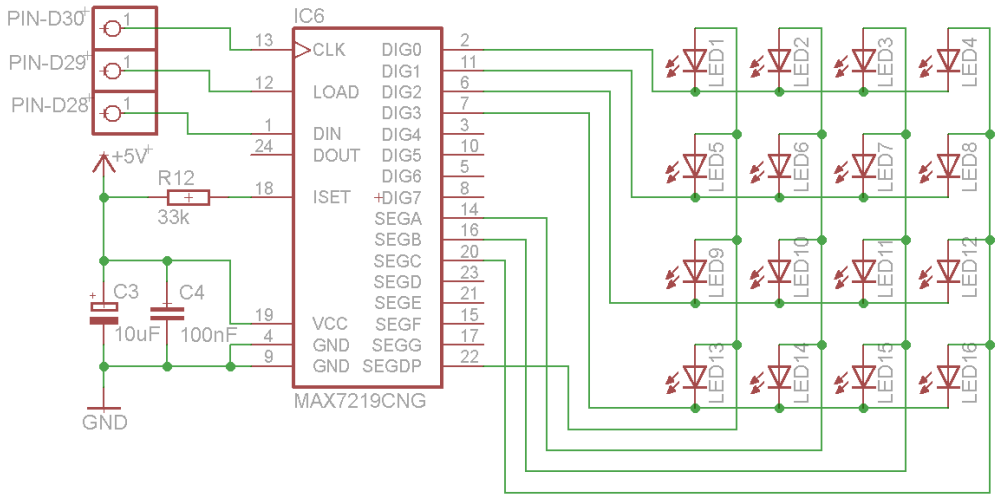
Co se týče výběru tlačítek, rozhodl jsem se použít tlačítka typu DT6. Jejich předností je nízká cena 12 Kč za kus, oproti tlačítkům s integrovanou LED diodou za 50 Kč. Ve větším počtu už je rozdíl ceny znatelný. Tlačítka se také dají pořídit v různých barvách, což je vhodné k odlišení jejich funkce.

Jelikož bude k odesílání MIDI zpráv docházet vždy při změně stavu tlačítka, je z hlediska programování nutné každému tlačítku přiřadit proměnnou, do které budeme ukládat předchozí stav. V podprogramu testování tlačítek bude nejprve potřeba nastavit číselnou kombinaci pro vybrání prvního vstupu multiplexorů a následně porovnat výstupy multiplexorů s proměnnými. V případě nerovnosti dojde k vyslání MIDI zprávy s hodnotou 127 v případě stisku, nebo 0 v případě uvolnění. Po porovnání čtveřice tlačítek dojde k přepnutí všech multiplexorů na druhý vstup a porovnávání pokračuje, než dojde k otestování stavu všech tlačítek a následnému opuštění podprogramu.

Pro budoucí použití jsem se rozhodl tlačítka doplnit o speciální tlačítko *Shift*, které při jeho držení dá ostatním tlačítkům novou funkci. V programu tak poté dojde k přidání další podmínky, zda je tlačítko *Shift* současně drženo. Jelikož ale toto tlačítko nebude připojeno na individuální vstup ale rovněž přes multiplexory, bude nutné jeho stav uložit do proměnné.

4.6 LED diody

LED diody u kontroleru používáme pro informování uživatele o stavu tlačítka, vedle kterého je dioda v těsné blízkosti, případně je přímo součástí podsvětleného tlačítka. Jelikož v návrhu počítáme s větším množstvím tlačítek, bude tomu odpovídat také větší množství LED diod. Stejně jako v případě tlačítek můžeme LED diody připojit buď jednotlivě na samostatný výstupní pin, nebo můžeme využít zapojení na principu mřížky. V mřížce je však zapotřebí neustále rozsvěcovat a zhasínat jednotlivé řádky, což nám ubírá čas chodu programu na řešení jiných úkolů. Z toho důvodu jsem se rozhodl použít řadič MAX7219 pro řízení mřížky velikosti 8x8. Tento řadič se k mikročipu připojuje pomocí trojice vodičů a zajistí kontrolu až 64 LED diod. Nejen že tedy ušetříme výstupní piny mikročipu, ale také programový čas. Rezistor R12, připojen na pin *I set*, nastavuje proud LED diodami. Podle maximálního proudu a nominálního napětí LED diod vybereme hodnotu rezistoru podle tabulky z [13]. Pro použité diody s doporučeným proudem 20 mA a napětím 1,8 V vychází hodnota rezistoru R12 na 28 kΩ, vybereme tedy nejbližší vyšší z řady E12, které odpovídá hodnota 33kΩ.



Obr. 4-8: Schéma zapojení řadiče MAX7219 pro 16 LED [13]

K ovládání řadiče existuje knihovna LedControl. V tabulce 4-4 je pro přehled uveden výpis používaných příkazů. Jelikož řadič disponuje datovým vstupem i výstupem, dají se mezi sebou řadiče navzájem propojovat a tvořit tak široký maticový displej zobrazující například textový řetězec. Z toho důvodu mají příkazy z knihovny LedControl vždy v parametrech adresu řadiče. Číslování adres i počet řadičů začíná od nuly. Jelikož nám pro potřeby kontroleru bude stačit jediný řadič, počet i adresa bude vždy nula.

Knihovna umožnuje práci s LED diodami po řádcích, po sloupcích, nebo s individuálními diodami. Pro potřeby kontroleru využijeme poslední způsob. Řízení diod můžeme rozdělit na dva typy. Prvním typem je řízení na základě MIDI zpráv, kdy rozsvěcování a zhasínání určuje DAW. Úkolem programu tedy bude vyhodnotit zprávu a odeslat příkaz řadiči. Druhým typem řízení je řízení mikročipem, na který DAW nemá vliv a diodám tak nepřiřazujeme MIDI zprávy. To použijeme v případě změny významu prvků kontroleru, jako například vrstev. Pokud jedním prvkem chceme ovládat několik parametrů v DAW, můžeme jeho funkci volit tlačítky. Jeden prvek tak může mít například čtyři funkce a jim odpovídající čtyři MIDI zprávy. Abychom uživatele informovali, jakou funkci prvek má, rozsvítíme LED diodu u tlačítka s popisem této funkce nebo celé vrstvy.

Tabulka 4-4 Příkazy pro práci s knihovnou Led Control [14]

Příkaz	Popis
LedControl led=LedControl (din, clk, cs, num)	Vytvoření objektu s názvem led, definování použitých pinů pro komunikaci a počtu řadičů
led.shutdown(addr, false)	Aktivování zobrazení na adrese daného řadiče
led.setIntensity(addr, 7)	Nastavení svítivosti v rozsahu 0 až 15
led.clearDisplay(addr)	Zhasnutí všech led diod
led.setLed(addr, ř, s, true)	Rozsvícení led diody v určeném řádku a sloupci
led.setLed(addr, ř, s, false)	Zhasnutí led diody v určeném řádku a sloupci
led.setRow(addr, ř, Bxxxxxxxx)	Rozsvícení nebo zhasnutí led diod v jednom řádku, za *x* dosadíme 1 nebo 0 podle pořadí diod
led.setColumn(addr, s, Bxxxxxxxx)	Rozsvícení nebo zhasnutí led diod v jednom sloupci, za *x* dosadíme 1 nebo 0 podle pořadí diod

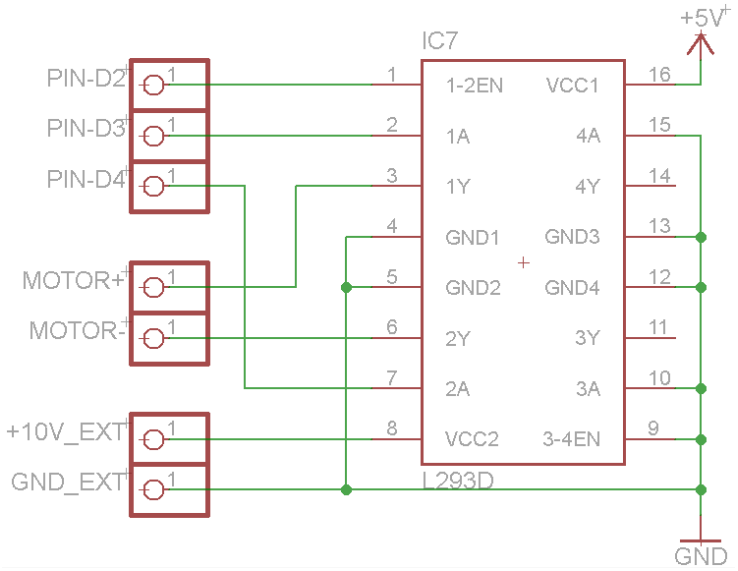
4.7 Motorizovaný tahový potenciometr

V případě motorizovaného tahového potenciometru je situace obdobná jako u lineárního potenciometru, navíc máme díky motoru možnost dostat jezdce na požadovanou pozici. Tento prvek si můžeme pomyslně rozdělit na dvě části. První částí je odporová dráha s jezdcem, pro které je schéma zapojení včetně části programu pro vysílání MIDI zpráv zcela shodné s lineárním potenciometrem, samozřejmě až na vlastní vstup a proměnnou s posledním stavem pozice tahového potenciometru. V případě změny pozice ze strany uživatele tak dojde k vyslání MIDI zprávy s novou pozicí. Druhou částí je stejnosměrný motor, připojený přes vodící pásek k jezdci, který umožňuje mikročipu s jezdcem posouvat. Díky tomu je zajištěna obousměrná vazba mezi kontrolerem a DAW. Když se změní pozice v DAW, informuje přes MIDI zprávu mikročip, aby změnil pozici fyzického tahového potenciometru.

Obr. 4-9: Motorizovaný tahový potenciometr [15]

Z hlediska programování je nutné k oběma směrům komunikace přistupovat jednotlivě a řešit je v oddělených podprogramech. Směr z kontroleru do DAW byl již popsán u lineárního potenciometru v kapitole 4.2. Směr z DAW do kontroleru poté řešíme v části programu pro příjem MIDI zpráv. Pokud přijde zpráva s novou pozicí, je nejprve nutné zjistit, zda se jezdec na pozici shodou okolností již nenachází. Tento stav může nastat při přepnutí na jiný kanál kdy DAW pošle všechny parametry nově zvoleného kanálu včetně pozice jezdce, který může být na stejné pozici jako u předchozího kanálu. Stejná situace může také nastat v případě potvrzovacích zpráv, kdy je provedena změna pozice přes kontroler a DAW vyšle zpět do kontroleru potvrzovací zprávu o nové pozici, která je rovněž shodná. V těchto případech tedy není nutné provádět žádnou akci a program pokračuje dále bez použití motoru. Toto chování v programu zajistíme podmínkou typu *if*. Pokud však používáme automatizaci nebo uděláme změnu pozice přímo v DAW, bude docházet k řízení pozice motorem. Zde je nejprve nutné zjistit potřebný směr pohybu. Ten zjistíme porovnáním, zda je rozdíl nové a staré hodnoty větší nebo menší než nula. Pokud je rozdíl kladný, dojde k pohybu nahoru, pokud je rozdíl záporný, dojde k pohybu jezdce směrem dolů. Zde rovněž použijeme podmínku typu *if*. Nyní, když známe směr, můžeme motor rozjet. V tomto místě je potřeba monitorovat pozici jezdce a zjišťovat rozdíl aktuální pozice vůči žádané. Tento rozdíl se při pohybu bude blížit nule až jezdec dojede na žádanou pozici a dojde k ukončení podmínky a následnému odpojení motoru. Jelikož však má jezdec určitou setrvačnost, musíme ji zohlednit také v podmínce. Podmínka tedy nebude ukončena, dokud není rozdíl nula, ale například dokud není rozdíl menší než jedna, případně dva. Jakmile je jezdec na žádané pozici, dojde k ukončení podprogramu pro přesun. Předtím je však nutné aktualizovat proměnnou s poslední pozicí tahového potenciometru na novou pozici, jinak by byla změna vyhodnocena a poslána MIDI zprávou zpět do DAW, což je v tomto případě nežádoucí.

K ovládání motoru po stránce hardware použijeme h-bridge neboli h-můstek typu L293D. Jedná se o integrovaný obvod určený pro řízení krokových motorů, servomotorů nebo klasických stejnosměrných motorů. Motorizovaný tahový potenciometr obsahuje poslední typ ze jmenovaných. H-můstek zde slouží jako výkonový stupeň ovládaný výstupy z mikročipu a umožňuje řídit směr i rychlost otáčení motoru. Z h-můstku využijeme dva logické vstupy, připojené k mikročipu, a jim odpovídající dvojici výstupů, na které připojíme vinutí motoru. Pokud na jeden vstup h-můstku přivedeme logickou jedničku a na druhý vstup logickou nulu, bude se motor otáčet. Pokud obě logické úrovně prohodíme, dojde k přepólování napětí na motoru což způsobí otáčení opačným směrem. Kromě dvojice vstupních pinů disponuje h-můstek také vstupem *enable*. Pokud na tento vstup přivedeme 0 V, chová se motor jako odpojený. Pokud zde přivedeme +5 V, je h-můstek aktivovaný a motor se točí plnou rychlostí. Vstupem *enable* můžeme také regulovat rychlost otáčení, když na něj přivedeme signál s pulzně šířkovou modulací (PWM) v rozsahu 0 až +5 V. Čím je napětí menší, tím se motor točí pomaleji.



Obr. 4-10: Schéma zapojení motoru přes h-můstek

Samotný h-můstek dále potřebuje ke své funkci napájení logických obvodů, které jsou napájeny běžným napětím +5 V. Jelikož se však stejnosměrné motory vyrábějí s různými nominálními napětími, disponuje h-můstek druhým napájecím pinem, který nám dovoluje připojit napětí až 36 V. Náš motorizovaný tahový potenciometr pracuje s nominální napětím 10 V, toto napětí je proto potřeba zajistit externím napájecím zdrojem a připojit k druhému napájecímu pinu h-můstku. Na základě vstupních hodnot tedy h-můstek připojuje toto napětí k motoru, případně otáčí polaritu nebo toto napětí spíná za účelem změny rychlosti.

Jelikož se běžně nevyrábí napájecí adaptér na 10 V, ale nejblíže na 9 nebo 12 V, doplnil jsem desku plošných spojů regulovatelným stabilizátorem napětí s LM317. Díky němu je možné k napájení použít 12 V napájecí adaptér. Protože je stabilizátor pouze doplňkem h-můstku, je schéma zapojení uvedeno v přiložených souborech Eagle.

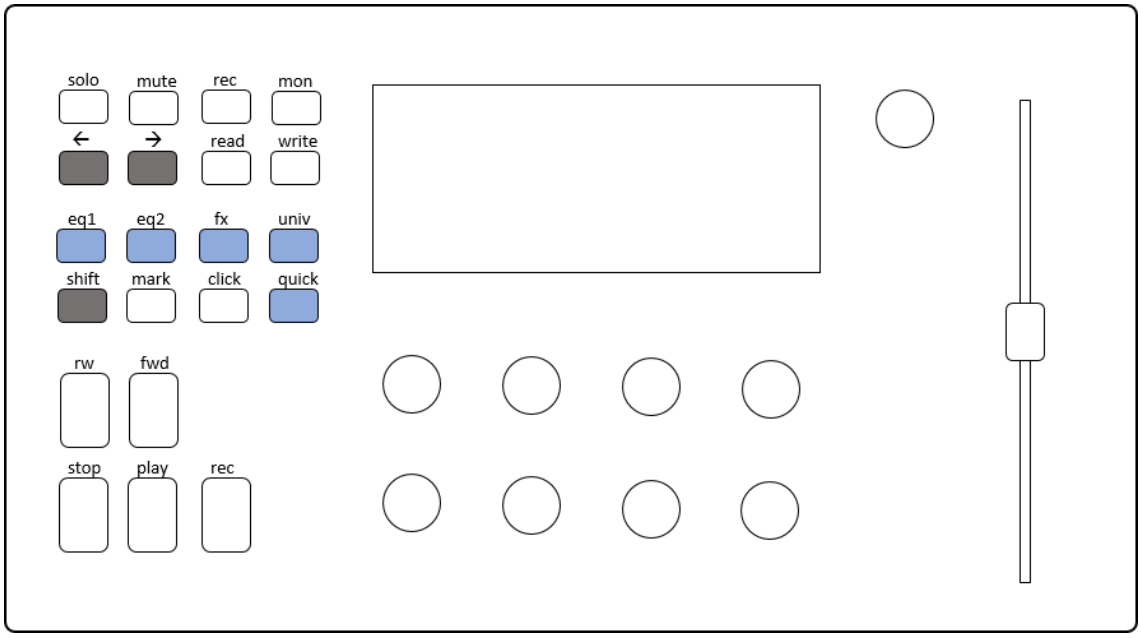
5. UPRAVENÝ NÁVRH KONTROLERU ZALOŽENÝ NA GENERIC REMOTE

Po zkušenostech z analýzy protokolu Mackie Control a zjištění, jak kontroler s tímto protokolem funguje, jsem došel názoru, že tento protokol je v jistých ohledech omezující a neumožňuje plné přizpůsobení ovládacích prvků jako při použití s Generic Remote, ve kterém jsem ovládací prvky odzkoušel. Generic Remote je rozhraní obecného dálkového řízení. Samotné okno nastavení Generic Remote v DAW Cubase má formát tabulky. Nastavení probíhá po řádcích, kde jednotlivým MIDI zprávám přiřadíme z nabídky potřebné funkce DAW. Tyto funkce jsou rozděleny do skupin podle významu a lze mezi nimi najít jak funkce pro nastavování kanálů, tak funkce pro celkové ovládání programu. Při původním návrhu vlastního kontroleru z kapitoly 3 jsem prakticky vycházel z již daných tlačítek a funkcí, které protokol nabízí a neměl jsem možnost zásadně měnit styl ovládání, neboť protokol je až na několik uživatelsky volených funkcí pevně daný. Dá se říci, že styl ovládání je naprogramován v DAW a kontroler v tomto případě pouze přeposílá aktivitu ovládacích prvků a nabízí zobrazovací prvky, které jsou řízeny přímo z DAW. Naproti tomu prostředí Generic Remote nabízí rozsáhlou sadu funkcí, umožňující ovládání kanálů, transportu a také globálního ovládání DAW. Tyto funkce je možné přiřadit libovolným MIDI zprávám a tím také umožňuje vytvoření vlastního komunikačního protokolu zcela podle vlastních potřeb. Daní za svobodu je však potřeba naprogramování samostatného zobrazování přímo uvnitř kontroleru, protože v tomto případě DAW nevysílá znaky, které se mají na displeji zobrazit, ale jen hodnoty v rozsahu 0 až 127, které se váží k nastavenému parametru. Tyto hodnoty je tedy potřeba upravit a zobrazit na displej včetně popisu parametru.

Po přehodnocení požadavků ovládání jsem upravil původní návrh kontroleru (obrázek 3-1) na podobu s několika sekcemi (obrázek 5-1). Sekci tlačítek, která umožnuje práci se zvoleným kanálem, dále zajišťuje přepínání funkce (modrá tlačítka) a také zajišťuje ovládání transportu (spodní sekce tlačítek). Dále sekci osmi enkodérů, které zde slouží jako hlavní ovládací prvky, a sekci displeje, který zobrazuje názvy a hodnoty parametrů, ovládané enkodéry. Po pravé straně se poté nachází otočný potenciometr, který bude sloužit pro nastavení celkové hlasitosti poslechu, a dále tahový potenciometr, sloužící pro nastavení úrovně zvoleného kanálu.

Pro tyto ovládací prvky a jejich počet jsem se rozhodl z několika hledisek. Nastavování zvuku v DAW spočívá především s prací s ekvalizérem, dynamickými procesory a posíláním signálu do efektů. Všechny tyto úkony se provádí přes enkodéry. V dnešní době dynamicky komprimované hudby není potřeba neustále dolaďovat úroveň pomocí tahových potenciometrů. Hlasitost nastavíme na počátku a poté už ji za nás hlídá kompresor. Pro svůj návrh jsem se tedy nakonec rozhodl využít jen jeden tahový potenciometr, který se nakonec jeví jako zcela dostačující daným potřebám.

Pokud bychom chtěli zaznamenat například automatizaci zpěvu, kanál se zpěvem si zvolíme a za pomoci tahového potenciometru můžeme automatizaci zaznamenat. Jelikož hlavními ovládacími prvky budou enkodéry s displejem, jsou enkodéry v návrhu umístěny dole uprostřed, aby byly lehce přístupné. Prvním hlediskem je tedy praktičnost a přehlednost. Druhým hlediskem je výrobní proces. Díky rozložení ovládacích prvků do sekcí bude snadnější výroba desek plošných spojů. Jednotlivé sekce budou mít své vlastní desky plošných spojů, které budou k mikročipu připojeny přes plochý více žílový kabel. Třetím hlediskem je univerzálnost. Díky volbě funkce enkodérů v několika vrstvách umožňuje kontroler ovládat až 40 virtuálních enkodérů s integrovanými tlačítky za pomoci osmi fyzických enkodérů. Vrstvy se přepínají pomocí modře značených tlačítek, přičemž při změně vrstvy dojde k přepsání popisu displeje podle aktuálních funkcí enkodérů.



Obr. 5-1: Konečný návrh vlastního DAW kontroleru

V tabulkách 5-1 až 5-7 jsou uvedeny funkce a MIDI zprávy ovládacích prvků kontroleru. Pro tlačítka jsou to zprávy typu Note On, proto je zde uvedeno číslo noty, pro potenciometry jsou to poté zprávy typu Control Change, zde je uvedeno číslo kontroleru. Enkodéry vysílají oba typy zpráv. Zprávy typu Control Change pro rotaci a zprávy Note On pro tlačítko. Kromě vysílání tyto prvky přijímají z DAW zpětně informaci o aktivitě ze strany DAW, a to na stejném čísle noty, případně kontroleru.

Tabulka 5-1 Funkce potenciometrů a čísla jejich kontrolerů

Ovládací prvek	Číslo kontroleru	Popis
Potenciometr	71	Ovládání úrovně hlavního výstupu
Tahový potenciometr	72	Ovládání úrovně zvoleného kanálu

Tabulka 5-2 Funkce tlačítek a jejich čísla not

Název	Číslo noty	Popis	Název	Číslo noty	Popis
Solo	0 *	Zapnutí funkce solo	Rw	8	Posun dozadu
Mute	1 *	Umlčení kanálu	Fwd	9	Posun dopředu
Rec	2 *	Nahrávání kanálu	Stop	10 *	Zastavení přehrávání
Mon	3 *	Poslech kanálu	Play	11 *	Spuštění přehrávání
←	4	Posun na kanál vzad	Rec	12 *	Spuštění nahrávání
→	5	Posun na kanál vpřed	Mark	13	Vložení markeru
Read	6 *	Čtení automatizace	Click	14 *	Zapnutí metronomu
Write	7 *	Záznam automatizace			

Poznámka: * značí přítomnost LED diody u tlačítka

Tabulka 5-3 Funkce enkodérů ve vrstvě EQ1

Enkodér	Číslo kontroleru	Číslo noty	Popis funkce
1	1	16	Zesílení EQ1
2	2	17	Frekvence EQ1
3	3	18	Činitel jakosti EQ1
4	4	19	Zapnutí / vypnutí EQ1
5	5	20	Zesílení EQ2
6	6	21	Frekvence EQ2
7	7	22	Činitel jakosti EQ2
8	8	23	Zapnutí / vypnutí EQ2

Tabulka 5-4 Funkce enkodérů ve vrstvě EQ2

Enkodér	Číslo kontroleru	Číslo noty	Popis funkce
1	9	24	Zesílení EQ3
2	10	25	Frekvence EQ3
3	11	26	Činitel jakosti EQ3
4	12	27	Zapnutí / vypnutí EQ3
5	13	28	Zesílení EQ4
6	14	29	Frekvence EQ4
7	15	30	Činitel jakosti EQ4
8	16	31	Zapnutí / vypnutí EQ4

Tabulka 5-5 Funkce enkodérů ve vrstvě FX

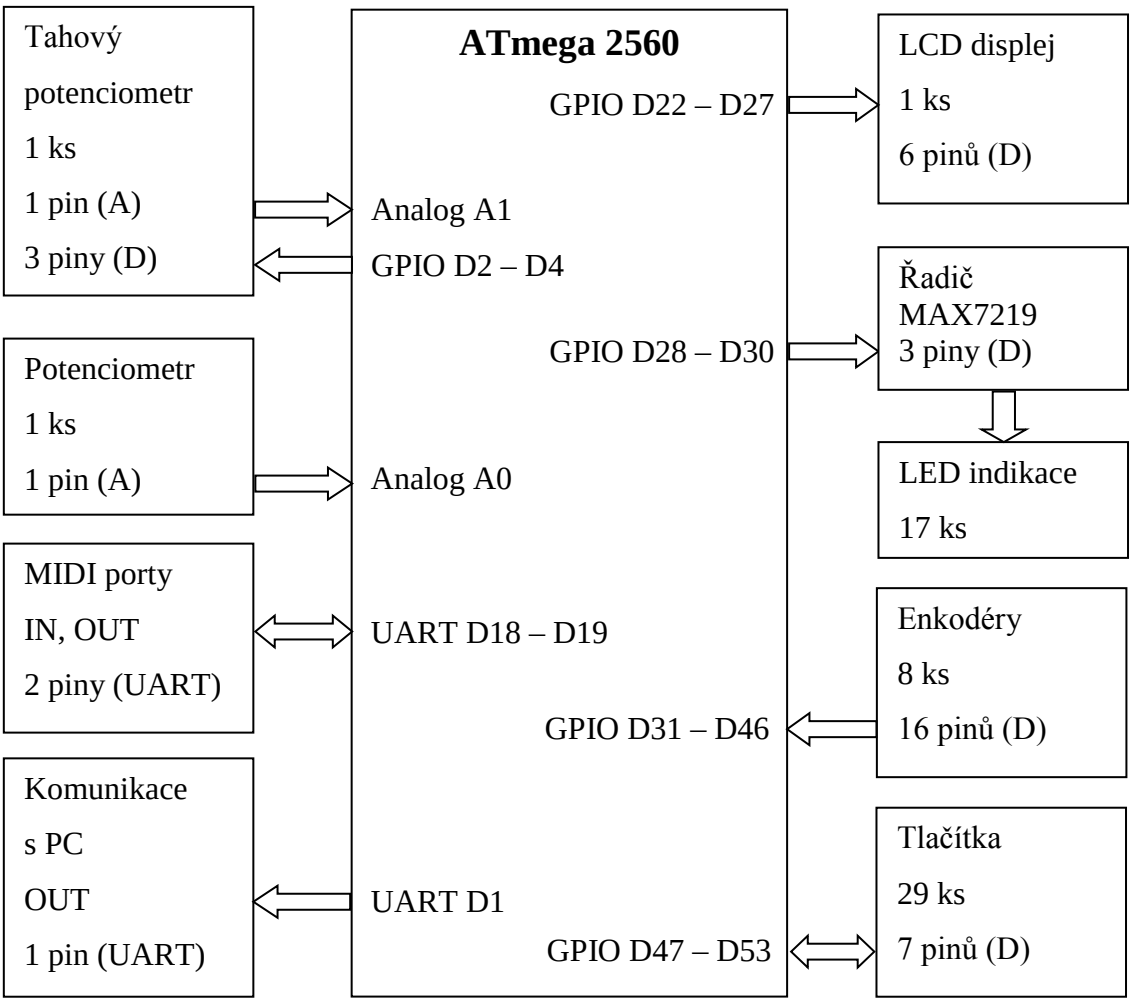
Enkodér	Číslo kontroleru	Číslo noty	Popis funkce
1	17	32	Úroveň do FX1
2	18	33	Úroveň do FX2
3	19	34	Úroveň do FX3
4	20	35	Úroveň do FX4
5	21	36	Úroveň do FX5
6	22	37	Úroveň do FX6
7	23	38	Úroveň do FX7
8	24	39	Úroveň do FX8

Tabulka 5-6 Funkce enkodérů ve vrstvě Univ

Enkodér	Číslo kontroleru	Číslo noty	Popis funkce
1	25	40	Panorama
2	26	41	Bypass EQ
3	27	42	Bypass Sends
4	28	43	Bypass Inserts
5	29	44	Zapnutí Insertu 1
6	30	45	Zapnutí Insertu 2
7	31	46	Zapnutí Insertu 3
8	32	47	Zapnutí Insertu 4

Tabulka 5-7 Funkce enkodérů ve vrstvě Quick

Enkodér	Číslo kontroleru	Číslo noty	Popis funkce
1	33	48	Quick control 1
2	34	49	Quick control 2
3	35	50	Quick control 3
4	36	51	Quick control 4
5	37	52	Quick control 5
6	38	53	Quick control 6
7	39	54	Quick control 7
8	40	55	Quick control 8



Obr. 5-2: Blokové schéma konečného zapojení kontroleru

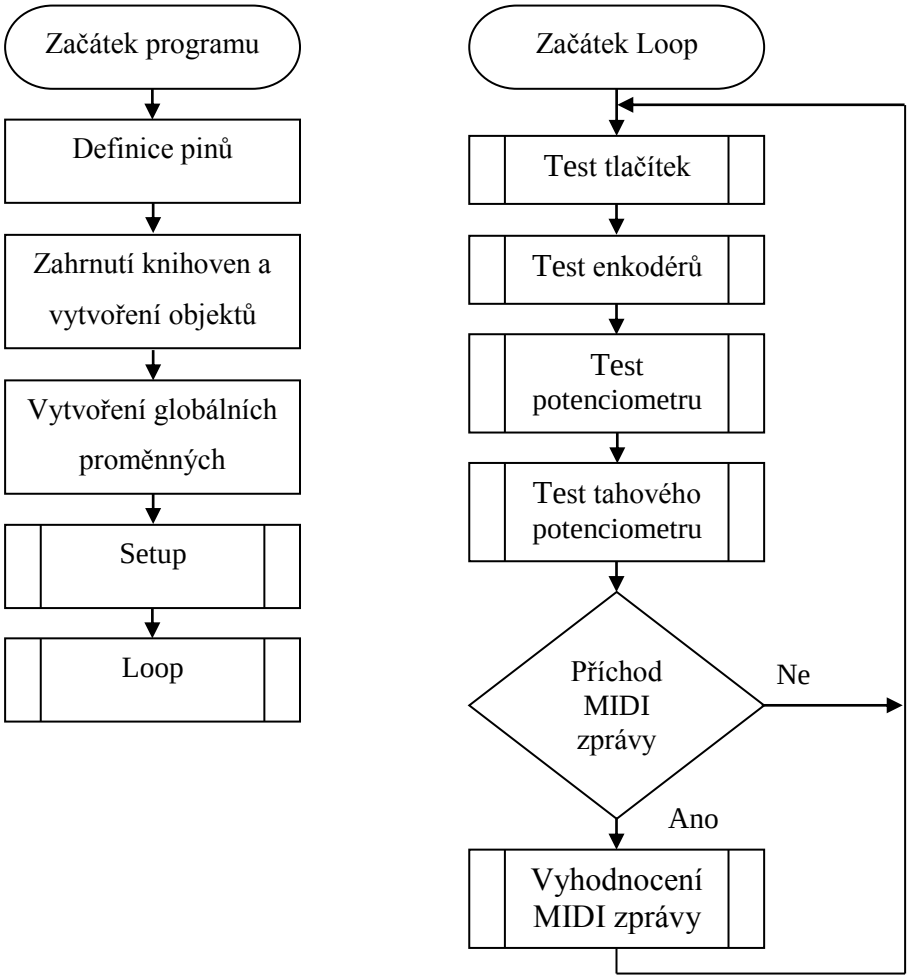
6. PROGRAMOVÁNÍ FIRMWARE

V předchozích kapitolách byl vysvětlen princip práce s elementárními ovládacími prvky včetně připojení k mikročipu a bloku programu, sloužící k vyhodnocení jejich aktivity. Dále byly na základě nabídky z rozhraní Generic Remote vybrány parametry, které budeme pomocí kontroleru ovládat. Těmto parametrům jsem přiřadil typy MIDI zpráv, které musí být specifikovány jak ve firmware kontroleru, tak v rozhraní Generic Remote v DAW. Nyní se zaměříme na programování firmware, který tyto požadavky bude splňovat a zabezpečí přenos informace do DAW a zpětně do kontroleru.

6.1 Hlavní program a globální proměnné

Základní myšlenkou je udělat hlavní program co nejjednodušší a rozdělit jej do několika bloků, jež jsou samostatně prováděny jako podprogramy. V případě kontroleru je za potřebí neustále testovat aktivitu všech ovládacích prvků a reagovat na příchozí MIDI zprávy. Hlavní program zde obstarává smyčka *Loop*, která je z principu zacyklená a provádí se tedy stále dokola. Z této smyčky tedy budeme volat podprogramy na testování aktivity tlačítek, enkodérů, potenciometru, tahového potenciometru a v případě, kdy dojde k příjmu MIDI zprávy, bude tato zpráva vyhodnocena rovněž v rámci svého podprogramu. Vývojový diagram smyčky *Loop* můžeme vidět na obrázku 6-1 vpravo. Rozdělení do podprogramů přináší také výhodu při odlaďování firmware. V případě potřeby tak můžeme vcelku jednoduše deaktivovat jednotlivé bloky ovládacích prvků.

Před samotným hlavním programem je však nejprve nutné provést celkové nastavení mikročipu. K tomu slouží podprogram *Setup*, který se provede jen jednou, a to před začátkem smyčky *Loop*. Jelikož budeme využívat knihovny a globální proměnné, je nutné je do programu zahrnout na úplném začátku programu ještě před podprogram *Setup*. Rozdíl mezi globální a lokální proměnnou je ten, že s lokální proměnnou můžeme pracovat jen v rámci podprogramu, ve kterém byla definována, a při opuštění tohoto podprogramu už k ní nemáme přístup. Globální proměnná se definuje mimo podprogram a můžeme k ní přistupovat ze všech podprogramů.



Obr. 6-1: Vývojový diagram celého programu (vlevo) a smyčky Loop (vpravo)

V programu nejprve definujeme vstupní a výstupní piny. Tím pinům přiřadíme název, díky kterému se v programu lépe orientuje. Nyní do programu zahrneme všechny potřebné knihovny a vytvoříme k nim instanci neboli objekt. Jedná se o knihovny MIDI, LiquidCrystal a LedControl. Dále je potřeba vytvořit všechny globální proměnné. Zde spadají proměnné pro poslední vyslané hodnoty otočného a tahového potenciometru ve formátu *byte*, tedy čísla o velikosti 8 *bitů*. Existuje několik datových typů pro uložení čísel a každý formát zabírá jinou velikost paměti. Proto je vhodné přizpůsobit typ proměnné rozsahu, abychom paměť co nejlépe využili. Další globální proměnné slouží k uložení posledního stavu tlačítek. Zde jsou proměnné rozděleny do čtveřice vektorů, z nichž každému multiplexoru odpovídá jeden vektor. Jelikož tlačítko nabývá pouze dvou stavů, je proměnná typu *boolean*. Následuje proměnná *layer*, ve které je uloženo číslo aktuální vrstvy enkodérů. Jelikož máme celkem 5 vrstev, bude proměnná nabývat hodnot 0 až 4. Pomocná proměnná *layerCH* slouží jako příznak

k povolení přepnutí do jiné vrstvy. Má sloužit k eliminaci stavu, kdy uživatel stiskne volbu dvou vrstev najednou. Aby se zabránilo neustálému přeskakování mezi vrstvami, povolí program změnu vrstvy až po uvolnění tlačítka aktuální vrstvy. Další proměnnou je příznak *shift* pro alternativní funkce. Následuje vektor posledního stavu enkodérů a za ním vektory pro adresování LED diod. Jelikož jsou diody uspořádány v mřížce, má každá dioda dvě souřadnice. Abychom však mohli pracovat jen s jedním pořadovým číslem, uložíme si odpovídající řádky a sloupce mřížky do dvojice vektorů. LED diody tak identifikujeme podle indexu. Posledním vektorem proměnných je vektor *hodnotyLCD*, ve kterém jsou uloženy sedmi bitové hodnoty nastavované enkodéry. Při přepnutí vrstvy je odpovídající část vektoru načtena a zobrazena na LCD displej.

6.2 Podprogram Setup

V podprogramu *Setup* provedeme úkony nutné pro následující činnost smyčky *Loop*. Dvojicí příkazů *begin* zahájíme komunikaci s PC a komunikaci po sběrnici MIDI. Dále zde specifikujeme počet řádků a sloupců displeje a nastavíme displej do výchozího stavu. Smažeme obsah a nastavíme kurzor na pole prvního znaku. Následně aktivujeme řadič pro řízení LED diod. Zde nastavíme výchozí intenzitu a provedeme zhasnutí všech diod. Dále zbývá nastavit funkci pinů, zda se bude jednat o vstup nebo výstup. V případě enkodérů nastavíme celkem šestnáct pinů jako vstup. Abychom nemuseli vypisovat šestnáct řádků kódu, využijeme smyčku typu *for*. Pro multiplexory nastavíme tři adresové piny jako výstupy a čtyři datové piny jako vstupy.

Po nastavení mikročipu zobrazíme uvítací displej, který je ve formě podprogramu. Na displej zobrazíme popis přístroje včetně aktuální verze firmware, tedy „DAW kontroler V1.0“ včetně roku výroby a autora. Uvítací displej po třech sekundách smažeme a necháme jednu sekundu prázdný. Následuje volání podprogramu k rozsvícení LED diody u aktivní vrstvy, která je nastavena v globální proměnné *layer* a jejíž výchozí hodnota je 0. Rozsvítí se tedy LED dioda u tlačítka EQ1. K dokončení prvotního nastavení už zbývá jen zobrazit na displeji popis parametrů pro zvolenou vrstvu, což obstarává podprogram *disp*. Tento podprogram nejprve smaže obsah celého displeje a zobrazí názvy parametrů pro danou vrstvu. Zobrazení tedy proběhne jen na první a třetí řádek. Zobrazení hodnot na druhý a čtvrtý řádek obstarává podprogram *dispHodnoty*. Jelikož však dosud nemáme potřebná data k zobrazení, nedochází k volání v rámci podprogramu *Setup,* ale až v bloku pro příjem MIDI zpráv.

6.3 Podprogram Tlačítka

Jelikož jsou tlačítka připojeny přes multiplexory, probíhá testovaní ve dvou krocích. Prvním krokem je nastavení adresy, druhým krokem poté čtení čtyř výstupů multiplexorů. Tyto dva kroky se opakují osmkrát po sobě. Z toho důvodu je druhý krok realizován jako pomocný podprogram, kterému je předána adresa a pořadí. Zde dojde k nastavení adresy a porovnání stavu tlačítek s předchozím stavem. Podle parametru pořadí se v případě stisknutí tlačítka počítá číslo noty pro MIDI zprávu. Fyzická tlačítka jsou číslována zleva doprava a shora dolů. Na prvním řádku tedy 1 až 4, na druhém 4 až 8. Pro první multiplexor, ke kterému jsou připojena kanálová tlačítka, číslo noty odpovídá přímo číslu pořadí. Pro druhý multiplexor je k pořadí přičteno číslo osm, abychom dostali čísla not v rozsahu 8 až 12 pro tlačítka transportu. Jelikož není tento multiplexor obsazen celý, dochází k porovnání jen tehdy, pokud je pořadí menší než 4. Čísla not tak odpovídají tabulce 5-2. Třetí multiplexor je připojen k integrovaným tlačítkům v enkodérech. Zde je k číslu pořadí přičteno číslo šestnáct, a navíc je přičten osminásobek čísla aktivní vrstvy. Díky této parametrizaci je počítáno číslo noty v rozsahu od 16 do 55, jak je uvedeno od tabulky 5-3 dále.

Čtvrtý multiplexor je připojen k tlačítkům vrstev, tlačítku *shift* a tlačítkům *marker* a *click*. U tlačítek vrstev v případě stisku dojde ke kontrole, zda je povolena změna vrstvy. Pokud ano, přepíše se proměnná *layer* na číslo pořadí a dojde ke změně rozsvícení LED diod u tlačítek vrstev a změny obsahu displeje. Změnu LED diod obstarává podprogram, který nejprve všechny diody u vrstev zhasne, a rozsvítí podle pořadí jen diodu u aktivní vrstvy. Tlačítko shift v případě zmáčknutí přepíše svou proměnnou na logickou jedničku, v případě uvolnění na logickou nulu. Využití je plánováno do budoucna. Jelikož je čtení aktivity tlačítek po částech, bude mnohem jednodušší v programu porovnávat proměnnou něž vstup mikročipu, který je závislý na nastavení adresy multiplexoru. Tlačítka *marker* a *click* mají číslo noty pevně přidělené, jelikož se jedná jen o dva vstupy z celkového multiplexoru.

6.4 Podprogram Enkodéry

Jelikož je princip testování aktivity enkodérů shodný pro všechny enkodéry, využíváme k němu pomocný podprogram s názvem *enctest*. Z podprogramu *enkodery* jej voláme osmkrát, zvlášť pro každý enkodér. Předávanými parametry jsou čísla dvojice pinů, ke kterým je enkodér připojen, a jeho pořadí. V podprogramu *enctest* poté dojde k vyhodnocení aktivity, jak bylo popsáno v kapitole 4.3.

Číslo programu pro MIDI zprávu typu Control Change je počítáno stejným způsobem, jako v případě integrovaných tlačítek enkodérů. Díky tomu, že máme poslední stav průběhu *A* uložen ve formátu vektoru, a v parametru předáváme pořadí enkodéru, použijeme pořadí také jako index v tomto vektoru.

6.5 Podprogramy pro otočný a tahový potenciometr

Když voláme podprogram *potenciometr,* dojde k porovnání aktuálního stavu s posledním stavem potenciometru, uloženým v proměnné, a v případě rozdílu těchto hodnot dojde k vyslání MIDI zprávy. V případě tahového potenciometru je princip totožný, akorát porovnáváme data z jiného vstupu s jinou proměnnou. Protože se v tomto případě jedná jen o samostatné ovládací prvky, jejichž princip včetně programů byl vysvětlen v předchozích kapitolách, není potřeba program přizpůsobovat pro větší množství a je tak začleněn do celkového programu kontroleru jako podprogram.

Jelikož rozhraní Generic Remote neumožnuje práci s typem MIDI zprávy Pitch Bend, musíme se omezit na sedmi-bitovou přesnost ve zprávě Control Change. I když MIDI norma umožňuje přenášet také šestnácti-bitové parametry kontrolerů za pomoci dvojice MSB-LSB kontrolerů, Generic Remote tyto dvojice neumožňuje vyhodnotit. V případě automatizace však nemusí být toto omezení zásadní, protože v průběhu křivky automatizace dochází k plynulému přechodu mezi sousedními body na křivce.

V této části by bylo dobré zmínit podprogram *motor*, který se váže k tahovému potenciometru. Tomuto podprogramu je předán jediný parametr *pozice*, ve kterém je přenesena hodnota z příchozí zprávy Control Change odpovídající tahovému potenciometru. Tento podprogram poté zajistí dopravení jezdce na určenou pozici, jak je uvedeno v kapitole 4.7.

6.6 Podprogram pro příjem MIDI zpráv

Tento podprogram se volá v případě, kdy je v hlavním programu zjištěn příjem MIDI zprávy. Nejprve si zde vytvoříme lokální proměnné pro typ zprávy, čísla noty, kontroleru, dynamiky a hodnoty kontroleru. Do proměnné typ zprávy uložíme pomocí příkazu *get.Type* typ příchozí MIDI zprávy, na základě kterého zprávy třídíme pomocí podmínky *case*. Pokud je typ *Note On* a číslo noty v rozsahu od nuly do čtrnácti, následuje příkaz s rozsvícením, případně zhasnutím LED diody se shodným indexem. Souřadnice LED diod odpovídají schématu na obrázku 4-8. Kontroler obsahuje celkem 16 LED diod, jejichž pořadí je číslováno zleva doprava a shora dolů podle umístění na kontroleru. Tlačítko *Solo* na obrázku 5-1 obsahuje první diodu v mřížce, tlačítko *Rec* poslední. Tlačítka, která LED diodami nedisponují, mají na daném indexu uvedeny volné souřadnice, na kterých se LED diody nenacházejí. MIDI zpráva s požadavkem na její rozsvícení by na základě nastavení Generic Remote neměla přijít. V případě že by se za určitých okolností tato situace vyskytla, je ošetřena volnou souřadnicí.

Pokud je typ zprávy *Control Change* s číslem kontroleru 16 až 55, jedná se o zprávy přenášející hodnotu nastavovanou enkodéry. Při příchodu této zprávy dojde nejprve k uložení hodnoty do paměti. Paměť je ve formě vektoru proměnných o délce čtyřiceti *bytů*. Abychom vektor proměnných využili od jeho začátku, uložíme hodnoty na pozici

o šestnáct menší, než je číslo kontroleru. Tak budou čísla kontroleru 16 až 55 odpovídat pozicím 0 až 39. Tento přístup obstará uložení hodnot pomocí jediného příkazu. Hodnoty ukládáme pro potřebu změny vrstvy. Pokud na kontroleru vrstvu změníme, musíme načíst hodnoty z paměti čipu, na rozdíl od protokolu Mackie Control, který při změně vrstvy načítá všechny informace z DAW. Nyní je potřeba ověřit, zda hodnota spadá pod aktuální vrstvu. Toho dosáhneme dvojicí podmínek *switch* a *if*. První podmínkou volíme rozsah čísel kontrolerů, které odpovídají aktuální vrstvě, druhou testujeme, zda se v tomto rozsahu nachází přijatá hodnota. Pokud ano, následuje volání podprogramu *dispHodnoty,* který zajistí přepsání hodnot na displeji. Pokud číslo kontroleru nespadá do aktuální vrstvy, displej zůstane beze změny a nová hodnota je tedy jen uložena do paměti a zobrazí se až při přechodu do jí odpovídající vrstvy.

Podprogram *dispHodnoty*, který obstarává výpis hodnot na displeji, je založen na příkazu *switch*. Podle aktuální vrstvy vybere indexy odpovídajících hodnot z paměti. Pro každou hodnotu nejprve nastaví souřadnici kurzoru a poté ji zobrazí. Nyní zbývá ošetřit případ, kdy kontroler hodnotu z DAW dosud nepřijal. Jelikož je vektor proměnných předem daný a naplněný výchozími hodnotami, došlo by k vypsání těchto výchozích hodnot. Abychom tomuto zobrazení zabránili, jsou všechny výchozí hodnoty rovny 250. Zobrazení proběhne jen tehdy, pokud je hodnota menší než 128, což odpovídá sedmi-*bitovému* rozsahu MIDI. Pokud už jednou MIDI zpráva s odpovídajícím číslem kontroleru přijde, bude se hodnota zobrazovat až do vypnutí zařízení. Zde je také dobré dodat, že by se měl kontroler zapnout a připojit k PC ještě před spuštěním DAW. Po spuštění DAW a načtení projektu dojde k odeslání všech hodnot parametrů projektu, které jsou specifikovány v Generic Remote.

Pokud je číslo kontroleru 72, je volán podprogram *motor*, který zajistí dopravení jezdce tahového potenciometru na žádanou pozici. Pokud přijde zpráva, jejíž typ není *NoteOn* ani *ControlChange*, dojde k vypsání informace na monitor počítače, že se jedná o jiný typ MIDI zprávy. To slouží pro případ kontroly programátora, jestli z DAW přišla neočekávaná MIDI zpráva. Při monitorování sběrnice byl také zjištěn příjem zpráv *Active Sensing* (vysvětleno v [2]). Jelikož docházelo k neustálému vypisování informace o příjmu jiného typu zprávy, jsou tyto zprávy prozatím ignorovány. Do budoucna mohou být zprávy využity k informování uživatele o odpojeném konektoru MIDI IN v případě, kdyby mikročip po určitém čase detekoval absenci příjmu těchto zpráv.

7. KONSTRUKCE HARDWARE

Problematika konstrukce obnáší navrhnout hardware tak, aby obsahoval všechny výše zmíněné součásti, vybrat typ součástek a k nim poté navrhnout, vyrobit a osadit desky plošných spojů. Zde se nabízí několik možných druhů řešení. Jelikož jsem měl součástky pro testovací účely ve vývodovém provedení, navrhl jsem desky plošných spojů jednovrstvé rovněž pro vývodové provedení. Návrh celkem obsahuje sedm desek plošných spojů, z toho čtyři obsahují přímo ovládací prvky a zbylé tři jsou pro další součástky. Jedna deska plošných spojů je tak oddělena jen pro H-můstek, který má oddělené napájení, další pro MIDI rozhraní včetně multiplexoru pro enkodéry, který by přímo u enkodérů zabíral příliš místa, a poslední slouží jako svorkovnice pro napojení kabeláže do svorkovnice Arduina. Enkodéry a tlačítka jsem od sebe oddělil kvůli různé potřebné hloubce zapuštění pod ovládací panel. Tlačítka jsem rozdělil na trojici desek pro lepší manipulovatelnost. Každá sekce tlačítek obsahuje svůj multiplexor. Multiplexory sdílí stejné adresové vstupy, proto je na deskách vždy dvojice adresových pinů, pro vstup a zároveň jako rozbočení pro následující multiplexor. Použité enkodéry jsou jiného výrobce, než které nabízí knihovna programu Eagle. Enkodéry však mají shodné rozměry a rozmístění děr pro vývody. Jediný rozdíl je v jiném pořadí vývodů. Zapojení v Eagle bylo proto nutné překreslit tak, aby pořadí odpovídalo katalogu podle výrobce. V souborech Eagle jsou proto z tohoto důvodu prohozeny piny B a C.

Deska plošných spojů pro MIDI rozhraní obsahuje také připojovací piny pro rozvod napájení pro ostatní desky, také trimr pro nastavení kontrastu displeje a rezistor pro nastavení úrovně jasu podsvícení. Napájení displeje se tak připojí k této desce a datové vodiče na svorkovnici k mikročipu. Každá deska plošných spojů obsahuje montážní díry pro uchycení. Desky s enkodéry a tlačítky obsahují vždy pět děr, čtyři v rozích a jednu uprostřed, zamezující prohýbání desky. Displej je uchycen čtyřmi šroubky. Ostatní desky mají montážní díry k uchycení ke spodní straně krabice výrobku. Fotografie osazených desek plošných spojů jsou uvedeny v příloze č. 2.

8. ZÁVĚR

Cílem této práce bylo analyzovat komunikaci protokolu Mackie Control pro řízení DAW software a následně navrhnout, naprogramovat a vyrobit vlastní DAW kontroler. První část této práce se zabývá protokolem Mackie Control a principem přenášení dat mezi kontrolerem a DAW s využitím MIDI. Jelikož struktura tohoto protokolu není volně dostupná, bylo zapotřebí veškerou komunikaci analyzovat a vyhodnotit, jaké MIDI zprávy protokol Mackie Control využívá a jakým způsobem jsou informace kódovány. Výsledky této analýzy včetně popisu jsou uvedeny v kapitole 2. Hlavním poznatkem analýzy je možnost pomyslně oddělit vstupní a výstupní část kontroleru na dva samostatně fungující a nezávislé celky. Vstupní část předává do DAW informace o aktivitě ovládacích prvků a přes výstupní část, obsahující LED diody, displej a motory tahových potenciometrů, DAW informuje uživatele o stavu parametrů. V samotném kontroleru tedy neprobíhá žádné řízení výstupu vstupem, vše obstarává DAW. Právě tato skutečnost byla motivem pro použití rozhraní Generic Remote, které umožňuje přiřazení parametru v DAW fyzickému ovladači na kontroleru, zcela volně podle potřeb programátora, případně uživatele.

Práce obsahuje dva návrhy kontrolerů. První je založen na protokolu Mackie Control, druhý na Generic Remote. U prvního návrhu je vidět podobnost s kontrolerem Mackie MCU právě kvůli pevně daným pravidlům protokolu a malé možnosti přizpůsobení. Druhý návrh je již přizpůsobený mým potřebám. Hlavní myšlenkou je ovládání založené na rotačních enkodérech a rozmístění ovládacích prvků do skupin podle významu.

V kapitole 4 je popsáno připojení ovládacích a zobrazovacích prvků k mikročipu ATmega2560 s popisem obslužného programu. V kapitole 6 je poté popsán celý program, který zajišťuje chod kontroleru. V obou kapitolách jsou uvedeny mechanismy ke snížení počtu potřebných vstupů a práce s pamětí mikročipu. Hlavní program je rozdělen do podprogramů vyhodnocujících vstupy, jako jsou tlačítka, enkodéry a potenciometry, a podprogramu pro příjem MIDI zpráv, které obvykle řídí výstupy. Na rozdíl od Protokolu Mackie Control zde tyto dva bloky pro vstupy a výstupy mezi sebou komunikují. Přepínání vrstev a zobrazování parametrů na displej má na starosti přímo mikročip.

I přesto že protokol Mackie Control nakonec nebyl pro vlastní návrh kontroleru použit, byla analýza cennou zkušeností pro získání povědomí o způsobu přenosu informace a vůbec celkového fungování a možností kontroleru.

Do budoucna plánuji kontroler rozšířit o další funkce, ovládané pomocí tlačítka shift. Napadá mě například možnost řídit svítivost LED diod, která je ovládána řadičem, a přizpůsobit tak jas diod okolnímu osvětlení, případně přidat nové funkce řízení DAW, které se po delším čase praktického používání projeví jako vhodné. Díky možnostem Generic Remote je možné funkci kontroleru přizpůsobit také dodatečně.

Literatura

[1] MACKIE. MCU Pro, Universal Control Surface, Owner's manual. *Mackie* [online]. [cit. 2018-10-15]. Dostupné z: https://mackie.com/products/mcu-pro-and-xt-pro

[2] SCHIMMEL, Jiří. Komunikační rozhraní MIDI. *Elektrorevue* [online]. Brno, 2002 [cit. 2018-10-15]. Dostupné z: http://www.elektrorevue.cz/clanky/02069/index.html

[3] GUÉRIN, Robert. *Velká kniha MIDI*. Brno: Computer Press, 2004. ISBN 80-7226-985-2.

[4] MACKIE. MCU Pro and XT Pro. *Mackie* [online]. [cit. 2018-10-17]. Dostupné z: https://mackie.com/products/mcu-pro-and-xt-pro

[5] PRESONUS. Studio One Mackie Control Support. *Presonus* [online]. [cit. 2018-11-24]. Dostupné z: https://www.presonus.com/products/Studio-One/downloads

[6] ARDUINO. Arduino MEGA 2560 Rev 3. *Arduino* [online]. [cit. 2018-12-11]. Dostupné z: https://store.arduino.cc/arduino-mega-2560-rev3

[7] VODA, Zbyšek. *Průvodce světem Arduina* [online]. Druhé. 2018 [cit. 2019-02-15]. Dostupné z: https://arduino.cz/2-vydani-pruvodce-svetem-arduina-jako-e-book-zdarma/

[8] TOSHIBA. 6N138. *Toshiba* [online]. 2014 [cit. 2019-02-15]. Dostupné z: https://toshiba.semicon-storage.com/us/product/opto/photocoupler/detail.6N138.html

[9] ARDUINO. Arduino MIDI Library. *Arduino* [online]. [cit. 2019-02-19]. Dostupné z: https://playground.arduino.cc/Main/MIDILibrary/

[10] MIDI Library For Communication With Musical Instruments. *PJRC* [online]. [cit. 2019-02-19]. Dostupné z: https://www.pjrc.com/teensy/td_libs_MIDI.html

[11] Inkrementální spínač EC11-1S. *GM Electronic* [online]. [cit. 2019-02-28]. Dostupné z: https://www.gme.cz/inkrementalni-spinac-ec11-1s

[12] DARRAH, Kevin. Rotary Encoder Tutorial with Arduino Code. In: *Youtube.com* [online]. [cit. 2019-03-02]. Dostupné z: https://www.youtube.com/watch?v=HQuLZHsGZdI

[13] ARDUINO. The MAX7219 and MAX7221 Led drivers. *The Arduino Playground* [online]. [cit. 2019-03-13]. Dostupné z: https://playground.arduino.cc/Main/MAX72XXHardware/

[14] M., Luboš. Arduino LED matice. *Arduino Návody* [online]. 14. 2. 2017 [cit. 2019-03-14]. Dostupné z: https://navody.arduino-shop.cz/navody-k-produktum/arduino-led-matice.html

[15] THOMANN. Yamaha Motorised Fader LS9/01V96. *Thomann* [online]. [cit. 2019-04-16]. Dostupné z: https://www.thomann.de/cz/yamaha_ls_9_01v_motorised_fader.htm

Seznam příloh

Příloha 1 - Testovací programy

Program č. 1: Testování MIDI výstupu

```
void setup() {
  Serial1.begin(31250);
}

void loop() {
  noteOn(60, 90);
  delay(1000);
  noteOn(64, 90);
  delay(1000);
  noteOn(67, 90);
  delay(1000);
  noteOn(60, 0);
  noteOn(64, 0);
  noteOn(67, 0);
  delay(1000);
}

void noteOn(byte vyska, byte dynamika) {
  Serial1.write(0x90);
  Serial1.write(vyska);
  Serial1.write(dynamika);
}
```

Program č. 2: Testování MIDI vstupu

```
byte prijato;

void setup() {
  Serial.begin(9600);
  Serial1.begin(31250);
}

void loop() {
  if (Serial1.available()>0) {
    prijato = Serial1.read();
    Serial.println(prijato);
  }
}
```

Program č. 3: Testování MIDI knihovny

```
#include <MIDI.h>
MIDI_CREATE_INSTANCE(HardwareSerial, Serial1, MIDI);

void setup() {
  MIDI.begin(1);
  Serial.begin(9600);

  MIDI.sendNoteOn(60,100,1);
  delay(1000);
  MIDI.sendNoteOn(60,0,1);
}

void loop() {
  int type, note, velocity, channel;
  if (MIDI.read()) {
    byte type = MIDI.getType();
    switch (type) {
      case midi::NoteOn:
        note = MIDI.getData1();
        velocity = MIDI.getData2();
        channel = MIDI.getChannel();
        if (velocity > 0) {
```

```
        Serial.println(String("Note On:  ch=") + channel + ", note=" + note + ",
velocity=" + velocity);
      } else {
        Serial.println(String("Note Off: ch=") + channel + ", note=" + note);
      }
      break;

    case midi::NoteOff:
      note = MIDI.getData1();
      velocity = MIDI.getData2();
      channel = MIDI.getChannel();
      Serial.println(String("Note Off: ch=") + channel + ", note=" + note + ", velocity="
+ velocity);
      break;

    case midi::ControlChange:
      note = MIDI.getData1();
      velocity = MIDI.getData2();
      channel = MIDI.getChannel();
      Serial.println(String("Control Change: ch=") + channel + ", controler=" + note + ",
value=" + velocity);
      break;

   }
  }
}
```

Program č. 4: Zpracování dat z potenciometru

```
#define pinPot A0     // potenciometr

#include <MIDI.h>
MIDI_CREATE_INSTANCE(HardwareSerial, Serial1, MIDI);

void setup() {
  MIDI.begin(1);
  Serial.begin(9600);
}

int potCClast;
```

```
void loop() {

  int potAR = analogRead(pinPot);
  byte potCC = map(potAR, 0, 1023, 0, 127);

  if (abs(potCC-potCClast)>0) {
    MIDI.sendControlChange(71,potCC,1);
    Serial.println(potCC);
    potCClast = potCC;
  }
}
```

Program 5: Zpracování dat z enkodéru

```
#define enc1A 52    // enkoder 1 pin A
#define enc1B 53    // enkoder 1 pin B

#include <MIDI.h>
MIDI_CREATE_INSTANCE(HardwareSerial, Serial1, MIDI);

int aLastState = 1;

void setup() {
  pinMode(enc1A, INPUT);
  pinMode(enc1B, INPUT);
  MIDI.begin(1);
  Serial.begin(9600);
}

void loop() {
  int aState = digitalRead(enc1A);
  if (aState != aLastState && aState == 1){
    if (digitalRead(enc1B) != aState) {
      MIDI.sendControlChange(0,127,1);
      Serial.print("doprava");
    }
    else {
      MIDI.sendControlChange(0,1,1);
```

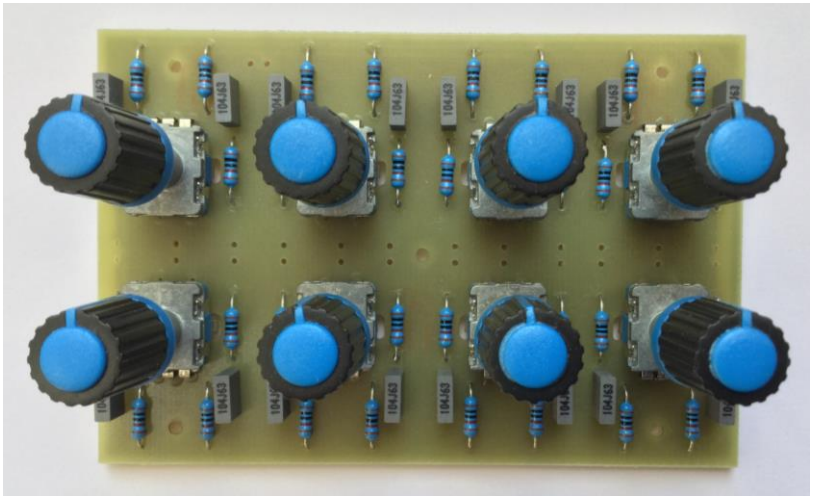
```
      Serial.print("doleva");
    }
  }
  aLastState = aState;
}
```

Program 6: Testování zobrazení na displeji

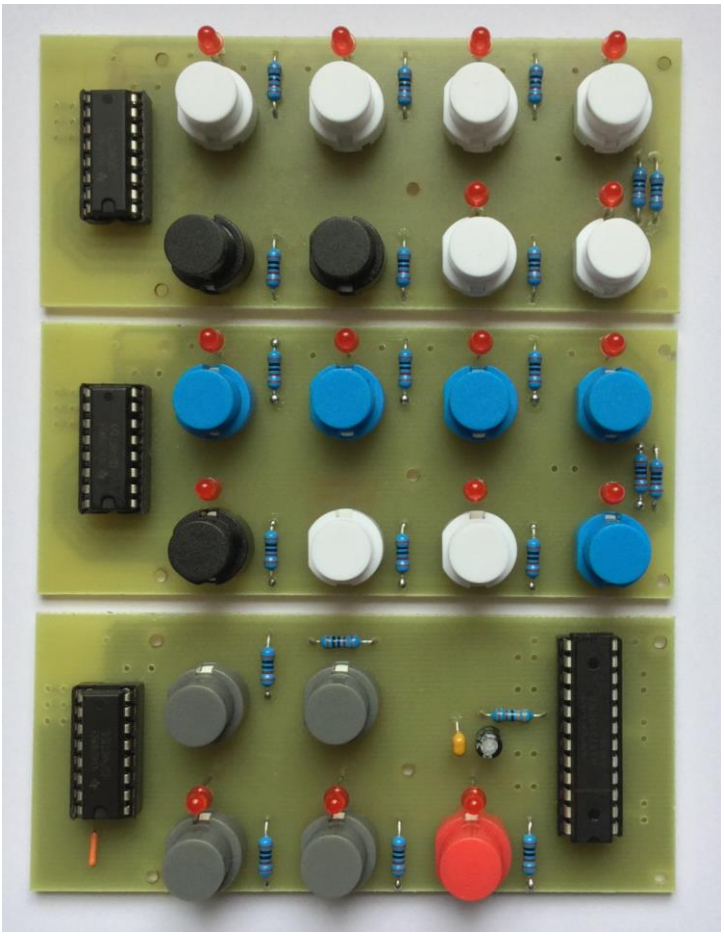
```
#include <LiquidCrystal.h>
LiquidCrystal lcd(22,23,24,25,26,27);

void setup() {
  lcd.begin(16,2);

  lcd.clear();
  lcd.setCursor(5,0);
  lcd.write("Hello");
  delay(1000);

  lcd.clear();
  lcd.setCursor(1,0);
  lcd.write("Cas od spusteni");
}

void loop() {
  if(millis()%1000 == 0) {
    lcd.setCursor(1,1);
    lcd.print(millis()/1000);
    lcd.print(" s");
  }
}
```

Příloha 2 - Fotografie desek plošných spojů

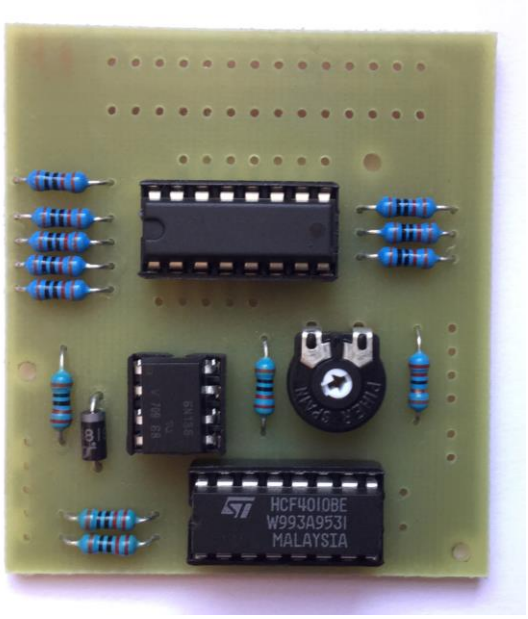
Osazená deska plošných spojů s enkodéry



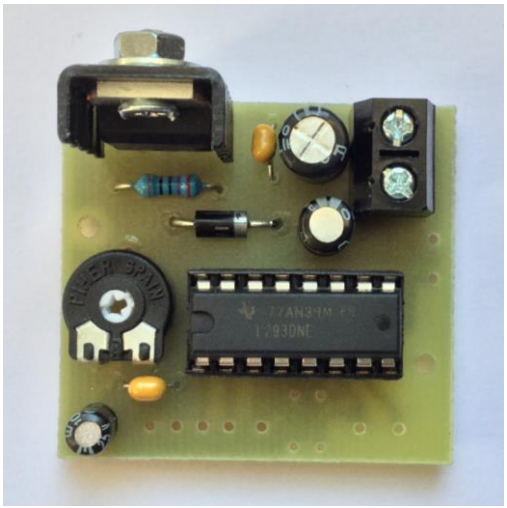
Osazené desky plošných spojů s tlačítky



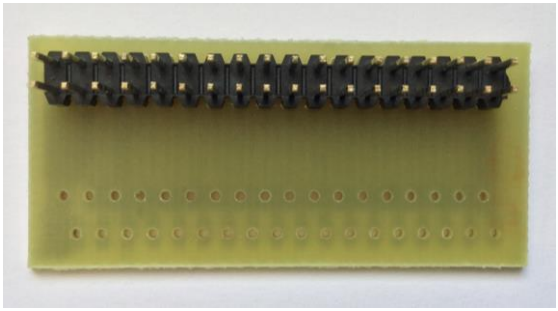
Osazená deska plošných spojů s obvody pro MIDI rozhraní



Osazená deska plošných spojů s h-můstkem



Osazená deska plošných spojů se svorkovnicí



Příloha 3 - Obsah přiloženého CD

Přiložené CD obsahuje:

- Elektronickou verzi diplomové práce
- Program kontroleru ve formátu Arduino file
- Program kontroleru ve formátu pdf
- Konfigurační soubor s parametry Generic Remote ve formátu xml
- Schémata a návrhy desek plošných spojů v programu Eagle