VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

APLIKACE PRO PROGRAMOVÁNÍ MIDI KONTROLERŮ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

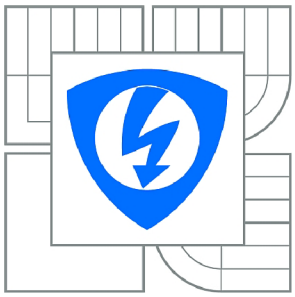
AUTOR PRÁCE
AUTHOR

MILOŠ DOLEJŠÍ

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

APLIKACE PRO PROGRAMOVÁNÍ MIDI KONTROLERŮ

APPLICATION FOR PROGRAMMING MIDI CONTROLLERS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

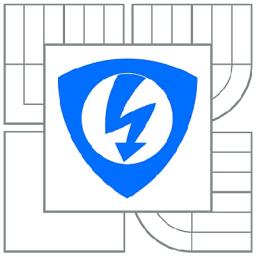
AUTOR PRÁCE — MILOŠ DOLEJŠÍ

AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ SCHIMMEL, Ph.D.

SUPERVISOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student: Miloš Dolejší ***ID:*** 155151
Ročník: 3 ***Akademický rok:*** 2014/2015

NÁZEV TÉMATU:

Aplikace pro programování MIDI kontrolerů

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte formát kanálových a systémových zpráv protokolu MIDI. Seznamte se s programovatelným MIDI kontrolerem Kenton ControlFreak a vytvořte aplikaci pro MS Windows s intuitivním grafickým rozhraním umožňující pomocí systémových zpráv protokolu MIDI naprogramovat funkce ovládacích prvků kontroleru ControlFreak a to ve dvou režimech: v jednoduchém režimu, kdy se pomocí menu budou volit vysílané MIDI zprávy, a v expertním režimu, kdy bude nutné zadávat MIDI zprávy v hexadecimálním kódu. Aplikaci poté modifikujte tak, aby bylo možné pomocí ní programovat i další typ MIDI kontroleru.

DOPORUČENÁ LITERATURA:

[1] The Complete MIDI 1.0 Detailed Specification. Document vision 96.1. The MIDI Manufacturers Association, Los Angeles, CA, 1997.
[2] KRUGLINSKI, D., J.; SHEPHERD, G.; WINGO, S. Programujeme v Microsoft Visual C++. Computer Press, Praha, 2003. 1014 s. ISBN 80-7226-362-5.
[3] VÁŇA, V. Atmel AVR, popis procesorů a instrukční soubor. Nakladatelství BEN – technická literatura, Praha, 2003. 336 s. ISBN 80-7300-083-0
[4] MATOUŠEK, D. Práce s mikrokontroléry Atmel AVR, 2. vydání. Nakladatelství BEN – technická literatura, Praha, 2006. 376 s. ISBN 80-7300-209-4.

Termín zadání: 9.2.2015 ***Termín odevzdání:*** 2.6.2015

Vedoucí práce: Ing. Jiří Schimmel, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být 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.

ABSTRAKT

Tato bakalářská práce se zabývá návrhem aplikace, která umožňuje naprogramovat pomocí MIDI System Exclusive zpráv funkce ovldádacích prvků kontrolerů ControlFREAK a Bitstream 3x. Teoretická část se zabývá rozborem komunikačního protokolu MIDI, ve kterém jsou popsána kanálová a systémová MIDI data. V praktické části jsou popsány jednotlivé části naprogramované aplikace a poté vysvětleny části naprogramovaného kódu. Aplikace je naprogramovaná v prostředí Visal Studio 2013, využívající knihovnu Windows Multimedia System, ze které jsou čerpány nezbytné funkce a struktury pro funkčnost aplikace. Komunikace mezi počítačem a kontrolerem je realizována pomocí převodníku MIDI-USB.

KLÍČOVÁ SLOVA

Aplikace, MIDI, System Exclusive, SysEx, kontroler, ControlFREAK, Bitstream 3x.

ABSTRACT

This bachelor thesis deals with designing of an application which allows programming of controls on ControlFREAK and Bitstream 3x controllers via MIDI System Exclusive messages. The theoretical part contains an analysis of the MIDI communication protocol, including description of channel and system MIDI data. The practical part describes individual parts of the application and then presents and explains parts of the source code. The application has been created in the Visual Studio 2013 SDK using Windows Multimedia System library, which contains functions and structures necessary for accomplishing the application's tasks. The communication between the PC and the controller is realized via MIDI-USB converter.

KEYWORDS

Application, MIDI, System Exclusive, SysEx, controller, ControlFREAK, Bitstream 3x.

DOLEJŠÍ, Miloš *Aplikace pro programování MIDI kontrolerů*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2014. 48 s. Vedoucí práce byl Ing. Jiří Schimmel, PhD.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Aplikace pro programování MIDI kontrolerů“ jsem vypracoval(a) samostatně pod vedením vedoucího bakalářské 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(ka) uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil(a) autorská práva třetích osob, zejména jsem nezasáhl(a) nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom(a) následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Jiřímu Schimmelovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této bakalářské práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .

podpis autora(-ky)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Tato bakalářská práce se zabývá návrhem aplikace pro MS Windows, která umožňuje naprogramovat, pomocí Sytem Exclusive zpráv protokolu MIDI, funkci zvoleného ovládacího prvku pro kontrolery ControlFREAK a Bitstream 3x. MIDI zprávy odesílané zvoleným ovládacím prvkem je možné zadávat ve dvou režimech a to v expertním a standardním režimu. MIDI zprávy v expertním režimu se zadávají v hexadecimálním kódu. Ve standardním režimu se MIDI zprávy vybírají z formulářového rolovacího menu, který je ovšem přístupný až po zadání správného hesla.

V teoretickém úvodu je popsáno komunikační rozhraní MIDI, ve kterém je stručně uvedena historie již zmíněného rozhraní, jeho hardwarové řešení a formát MIDI kanálových a systémových zpráv.

Praktická část je věnována samotnému návrhu aplikace a jeho řešení problémů, týkající se vlastní komunikace a implementace MIDI protokolu pro kontrolery. V této části je popsán význam jednotlivých záložek celé aplikace a jejich použití. Dále jsou zde vysvětleny části naprogramovaného kódu, jako je například princip posílání System exclusive zpráv, princip ukládání hesla do .ini souboru, záznam hesla v registrech nebo princip posílání kanálových MIDI zpráv.

Důležité funkce pro komunikaci s MIDI rozhraním jsou využity z knihovny Windows Multimedia System. Komunikace mezi počítačem a kontrolerem je realizována pomocí převodníku MIDI-USB.

1 KOMUNIKAČNÍ ROZHRANÍ MIDI

Rozhraní MIDI (Musical Instrument Digital Interface) bylo navrženo se specifickým protokolem, který by umožňoval vyměňovat si informace (noty, změny programů) mezi různými hudebními nástroji nebo jinými zařízeními, například sekvencery nebo počítačem, který by následně usnadnil záznam, editaci a zadávání dat. Rozhraní bylo původně koncipováno jen pro živá vystoupení, avšak následný vývoj přinesl další využití v audio-video produkcích a nahrávacích studiích.

Komunikační rozhraní MIDI bylo poprvé představeno ve finální verzi 1.0 v roce 1983 americkou společností Atari, která tímto rozhraním začala standardně vybavovat své počítače, tím si získala oblibu mezi hudebníky. Postupně bylo implementováno i mezi jiné společnosti.

Přenášená data neobsahují hudební složku, ale pouze řídící data, příkladem může být stlačení klávesy na klaviatuře, tím vyslat řídící data přijímacím zařízením, z nichž každé zařízení na ní může reagovat odlišným způsobem(hrát jiným nástrojem).

S využitím tohoto rozhraní je možné naprogramovat různé funkce tlačítek a tahových potenciometrů, kde se touto problematikou budeme v mé práci více zabývat.

1.1 Hardware rozhraní

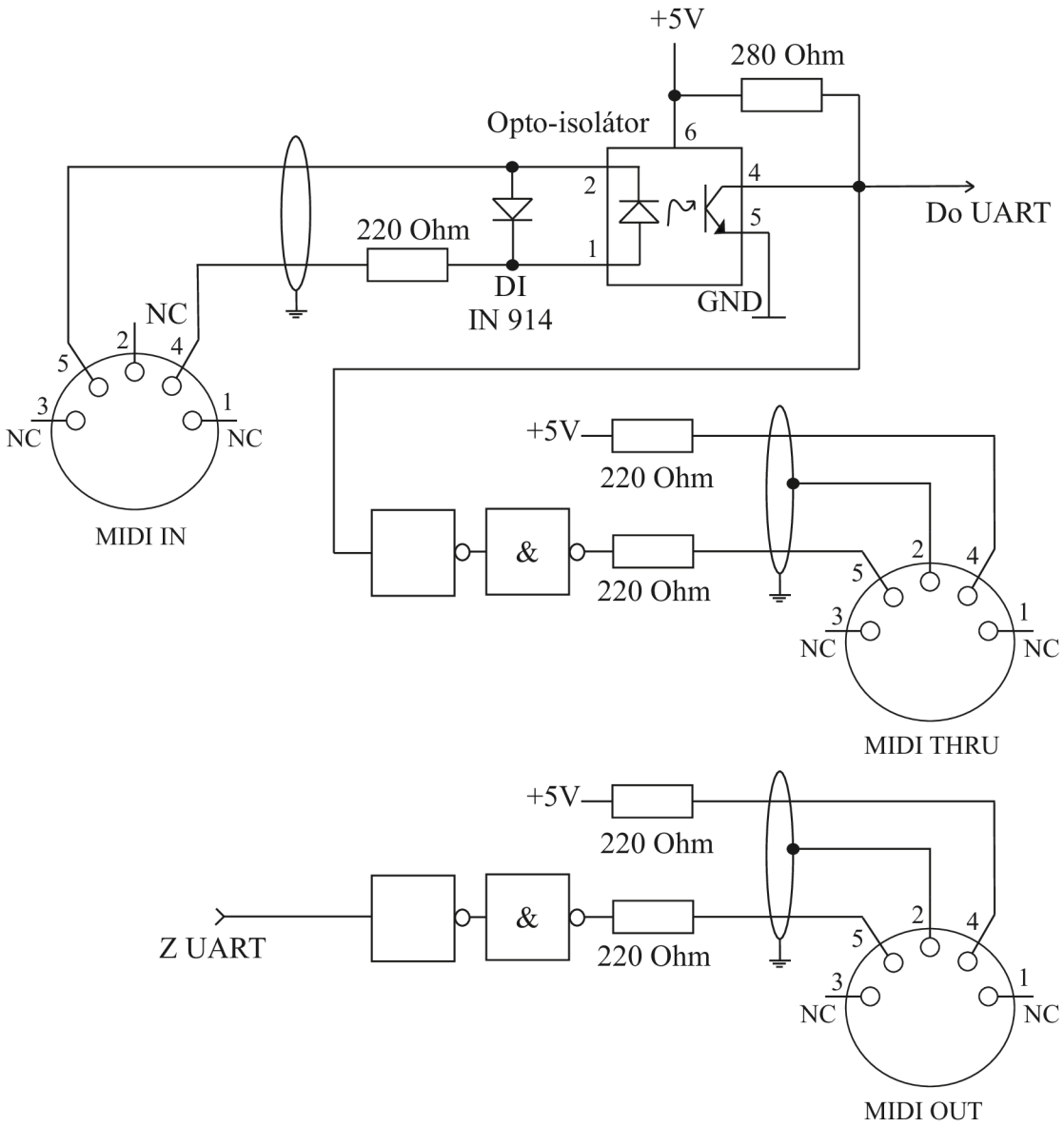
Rozhraní MIDI je tvořeno trojicí 5 pinových DIN konektorů(In, Out a Thru) z nichž první a třetí pin nemá žádné uplatnění. Druhý pin je využit jako stínění, které chrání před rušením nežádoucích signálů vyskytujících se v prostředí. Zbylé dva piny jsou použity pro samotnou komunikaci, protékající proud těmito piny znázorňuje směr toku dat. Jejich vnitřní zapojení je znázorněno na obr. 1.1. Rozhraní je 5 mA proudová smyčka, kde vstup je galvanicky oddělen. K přenosu dat se využívá sériového asynchronního datového přenosu s rychlostí 31,24 kBaudů [12, 11].

Funkce DIN konektorů:

- **In** - Vstup zařízení pro příjem MIDI zpráv.
- **Out** - Konektor pro výstup dat. Odtud vystupují MIDI zprávy generované vlastním zařízením.
- **Thru** - Na tomto konektoru se objevují kopírované zprávy ze vstupu zařízení. Přes tento konektor lze propojovat více zařízení najednou a docílit tím jejich zřetězení, ale každé přidané zařízení navíc, zároveň zvyšuje vzájemné časové zpoždění, proto se s oblibou využívají víceportová MIDI rozhraní nebo MIDI Merge box.

1.1.1 Topologie MIDI sběrnice

Konektor Thru slouží k propojení více MIDI přijímačů kaskádně, ale kvůli vzájemnému zpoždění se doporučuje zapojovat maximálně čtyři MIDI zařízení. Tento problém lze eliminovat s pomocí víceportového rozhraní (s Thru boxem) zapojením do hvězdicové topologie nebo jejich kombinacemi, výsledné zpoždění hvězdicové topologie je pak konstantní [11].



Obr. 1.1: Vnitřní zapojení rozhraní [16].

Sloučení MIDI dat

Pokud chceme sloučit data dvou nebo více MIDI vysílačů do jediného MIDI přijímače, použijeme k tomu tzv. Merge box, který sloučí data ze vstupů do jednoho výstupu. Pro možnost přenášení MIDI zpráv je Merge box vybaven mikroprocesorem se vstupy rozhraní UART a vyrovnávací pamětí. Dnes se již setkáme s Merge Boxem, umístěným přímo v zařízení, které je vybaveno funkcí *Soft Thru*, při jejíž aktivaci jsou vstupní MIDI data sloučena s MIDI daty generovanými zařízením.

1.2 Struktura datové zprávy

Základem komunikace na sběrnici MIDI je tzv. MIDI událost (event), což je 8 bitové datové slovo. Každý MIDI příkaz je sekvence bajtů, začínající stavovým a následovaná libovolným počtem datových bajtů, většinou však dvěma. K jejich identifikaci typu nám slouží nejvýznamnější bit. Stavový bajt má jeho hodnotu nastavenou na 1, tedy všechny hodnoty větší než $7F. V tab. 1.1 je zobrazena struktura datového a stavového bajtu.

Tab. 1.1: Struktura datového a stavového bajtu.

Stavový bajt	-	Datový bajt
$80-$FF		$$00-7F
1TTTnnnn		0vvvvvvv
Pozn: TTT-identifikátor typu zprávy; nnnn-identifikátor MIDI kanálu		

1.2.1 Stavový bajt

Stavový bajt se skládá z tříbitového *identifikátoru typu zprávy* určující kanálová nebo systémová data a dále z čtyřbitového *identifikátoru MIDI kanálu*, tedy šestnácti virtuálními kanály. Systémová data, na rozdíl od stavových, nenesou informaci o MIDI kanálu, ale jsou společná pro všechny kanály.

1.2.2 Datový bajt

Zde se využívá 7 bitů odpovídající 128 kombinacím, které jsou vyhrazeny například pro číslování not, volbě programu, úrovně hlasitosti atd.

1.2.3 Průběžný stav a priorita dat:

Abychom zvýšili propustnost sběrnice se neposílají bajty stavové, ale pouze datové, až do příchodu jiného stavového bajtu. Tomuto stavu říkáme *průběžný stav* (Running status).

Zařízení musí respektovat různou prioritu dat, např. pokud při přenosu kanálových dat dojde k příjmu dat reálného času, přenos kanálových dat se pozastaví [12].

Největší prioritu má resetování systému. Následují data reálného času, potom zvláštní a společná systémová data a jako poslední kanálová data.

1.3 Kanálová MIDI data

Jak již z názvu vyplývá, kanálová MIDI data se vztahují pouze k určitému virtuálnímu kanálu. Každá MIDI zpráva plní svojí specifickou funkci (viz obr. 1.2), která je tvořena jedním stavovým bajtem a dvěma nebo jedním datovým.

Tab. 1.2: Tabulka kanálových MIDI dat.

MIDI zpráva	Funkce	ID	Počet datových bajtů
Note off	Nota vypnuta	0	2 (číslo noty; dynamika)
Note on	Nota zapnuta	1	2 (číslo noty; dynamika)
Polyphonic key pressure	Individuální tlaková citlivost	2	2 (číslo noty; tlaková citlivost)
Control change	Změna kontroleru	3	2 (číslo kontr. hodnota k.)
Program change	Volba programu	4	1 (číslo programu)
Channel pressure	Společná tlaková citlivost	5	1 (tlaková citlivost)
Pitch bend change	Změna výsky tónu	6	2 (hodnota MSB; LSB)

1.3.1 Zpráva nota zapnuta/vypnuta

MIDI zpráva note on stejně jako note off nese dvě informace, číslo zahrané noty a dynamiku, kterou klávesa byla stlačena.

Uplatnění těchto zpráv najdete například při stisku klávesy, kdy vám zazní odpovídající tón, po uvolnění klávesy zařízení pošle zprávu *nota vypnuta*, nebo místo ní opět pošle *nota zapnuta*, kde hodnotu *velocity* (dynamiku) nastaví na nulu. Tento způsob umožňuje plné využití průběžného stavu MIDI sběrnice. Seznam MIDI čísel not je zobrazen v tabulce 1.3.

Příklad: $90 $30 $7F

Uvedený příklad popisuje příkaz note on pro 1. kanál, tón C2 a s maximální dynamikou.

Tab. 1.3: MIDI čísla not [12].

Oktáva	hud. označení	C	C#	D	D#	E	F	F#	G	G#	A	A#	H
-2	Sub-subkontra	0	1	2	3	4	5	6	7	8	9	10	11
-1	Subkontra	12	13	14	15	16	17	18	19	20	21	22	23
0	Kontra	24	25	26	27	28	29	300	31	32	33	34	35
1	Velká	36	37	38	39	40	41	42	43	44	45	46	47
2	Malá	48	49	50	51	52	53	54	55	56	57	58	59
3	Jednočárkovaná	60	61	62	63	64	65	66	67	68	69	70	71
4	Dvoučárkovaná	72	73	74	75	76	77	78	79	80	81	82	83
5	Tříčárkovaná	84	85	86	87	88	89	90	91	92	93	94	95
6	Čtyřčárkovaná	96	97	98	99	100	101	102	103	104	105	106	107
7	Pětičárkovaná	108	109	110	111	112	113	114	115	116	117	118	119
8	Šestičárkovaná	120	121	122	123	124	125	126	127				

1.3.2 Individuální tlaková citlivost

Individuální tlaková citlivost přenáší informaci o síle tlaku, kterým je působeno na stisknutou klávesu. Jsou přenášeny dva datové bajty, první přenáší informaci o čísle noty a druhý o působící síle [12].

1.3.3 Změna kontroleru

Změna kontroleru ovlivňuje celkové chování zařízení. První datový bajt, přenáší informace o čísle kontroleru, druhý datový bajt přenáší informace o změně obládacího parametru MIDI nástroje. Čísla kontrolerů nabývají hodnot od 0 do 127, kde jejich základní rozdělení je popsáno v tab. 1.4 [12].

Tab. 1.4: Základní rozdělení čísel kontroleru [12].

Číslo kontroleru	Funkce
0 - 31	MSB dat spojitých kontrolerů
32 - 63	LSB kontrolerů 0 - 31
64 - 96	Jednobitové kontrolery
97 - 101	Inkrementace/dekrementace a čísla parametrů
102 - 119	Nedefinované jednobitové kontrolery
120 - 127	Povely

Pro některé kontrolery je 128 hodnot nedostačující, proto jsou pro některé kontrolery vyhrazeny dvě MIDI čísla (MSB a LSB), mohou tedy nabývat až 16384 hodnot. Jednotlivé MIDI kontrolery se rozdělují na:

- Spojité kontrolery - nabývají v rozsahu 0 - 127, avšak některé kontrolery mají tzv. střední hodnotu.. Jejich využití najdeme například ve vyvážení hlasitosti, kde 0 značí maximální hlasitost v levém kanálu a hodnota 127 maximální hlasitost v pravém kanálu.
- Spínače - nabývají dvěma stavy, *vypnuto* pro hodnoty 0 - 63 a pro hodnoty 64 - 128 platí stav *vypnuto*.
- Inkrementační a dekrementační kontrolery - přijmutím čísla kontroleru dochází k přičtení nebo odečtení 1 od hodnoty parametru.
- Povely - ovlivňují, jakým způsobem budou zacházet s přijatými a odeslanými MIDI zprávami [12].

V tab. 1.5 je několik vybraných kontrolerů, přehled všech kontrolerů najdete v normě [16].

Tab. 1.5: Přehled vybraných kontrolerů definovaný normou [12].

Název	Číslo kontroleru	Význam
Bank select	0	Volba banky
Volume	7	Kanálová hlasitost
Balance	8	Poměr dvou zvukových zdrojů
Pan	10	Panorama
Sustain	64	Doznění
Sostemuto	66	Zadržení
Legato	68	Zapíná „vázanou hru“
Relase time	72	Doba doznění tónu po uvolnění klávesy
Attack time	73	Doba náběhu tónu
External effect deph	91	Hloubka extertního efektu
Tremolo deph	92	Hloubka efektu tremolo
All sound off	120	Všechny zvuky vypnuty
All sound off	121	Restet všech kontrolerů
All sound off	123	Všechny noty vypnuty
Omni off	124	Režim omni vypnut
Omni on	125	Režim omni zapnut

1.3.4 Změna programu

Pomocí této zprávy je možné měnit různé zvuky. Obsahuje 128 hodnot, které jsou pro většinu nástrojů nedostačující, proto jsou zvuky rozděleny do bank po 128 zvucích a tyto banky se přepínají pomocí kontrolerů *Bank Select* (Změna banky) [12].

1.3.5 Společná tlaková citlivost

Tato zpráva je společná pro všechny noty daného MIDI kanálu. Je zde přenášen jeden datový bajt udávající informaci o síle tlaku.

1.3.6 Změna výšky tónu

Účelem změny výšky tónu je efekt, kdy jedna nota plynule přechází do druhé. Zpráva obsahuje dva datové bajty přenášející hodnotu v rozsahu od -8192 do +8102, kde 0 je nulová změna výšky tónu [11].

1.4 Systémová MIDI data

Systémová MIDI data začínají stavovým bajtem v rozmezí od $F0 do $FF, z toho bajty $F4, $F5, $F9 a $FD nejsou definovány. Systémová MIDI data jsou společná pro celý systém, proto nemá smysl přenášet informaci o virtuálním kanálu [12].

1.4.1 Zvláštní systémová data

Jedná se o zprávy System Exclusive, zkráceně SysEx, umožňující posílání dat a informací, které jsou určené pro konkrétní zařízení, například data sekvenceru a otisk jeho paměti a podobně. Rozsah využití je velice široký. Zpráva může obsahovat libovolný počet bajtů, skládající se z 8 částí, jehož struktura je přehledně znázorněna v tabulce 1.6.

Tab. 1.6: Struktura SysEx zprávy.

$F0 -> ID výrobce -> ID zařízení -> Sub-ID#1 -> Sub-ID#2 ->
-> Datový blok -> Kontrolní součet -> $F7

Zpráva začíná stavovým bajtem $FH, následuje *ID výrobce*, které představuje identifikační číslo výrobce zařízení. Skládá se z jednobajtového nebo tříbajtového ID, členěného do skupin, podle Americké, Evropské, Japonské, speciální, či ostatní distribuce.

ID zařízení označuje číslo, dlouhé dva bajty, určené pro konkrétní zařízení. Z důvodu, aby se rozlišilo od ostatních zařízeních, stejného výrobce.

Dále zde jsou části *Sub-ID#1* a *Sub-ID#2*. Významů *Sub-ID#1* může nabývat několik. Může obsahovat informaci, zda zařízení SysEx zprávu posílá, nebo je o ní žádán, nebo jestli má posílat i kontrolní součet, atd. *Sub-ID#2* nese doplňkovou informaci *Sub-ID#1*. Například pro jaký prvek nebo skupinu je směrována.

Vlastní přenášená data tvoří *datový blok* ukončený kontrolním součtem, který je potřeba k detekci chyb. Ten však nemusí vždy být použit. SysEx zpráva je ukončena stavovým bajtem $F7 [12, 11].

Protože se v celé struktuře zvláštních systémových dat nesmí vyskytovat bajt s nejvyšší bitem nastaveným na 1, aby nedošlo k záměně se stavovým bajtem, je rozsah hodnot přenášených jedním bytem zvláštních systémových dat omezen na 0 - 127. Pokud je potřeba přenést větší hodnoty, je nutné je rozdělit na významnější a méně významnou část a přenést je pomocí dvou datových bajtů.

Universální systémová data

Jedná se o speciální případ SysEx dat, avšak nejsou určena pro konkrétního výrobce, ale pro všechna zařízení. Opět mají mnoho využití, požívají se například pro rozpoznání připojených zařízení, kdy tázající se zařízení vyšle zprávu *Identity request* a následně obdrží od ostatních zprávu *Identity Reply* obsahující jejich údaje.

Avšak důležitější jsou systémová data reálného času, která se používají například k synchronizaci zařízení nebo resetování systému.

2 NÁVRH APLIKACE SYSEX ZPRÁV

Aplikace byla naprogramovaná pro platformu Windows jazykem C++ v prostředí Visual Studio 2013 formou aplikace využívající grafickou knihovnu Windows Forms. Legální licenci Visual studia jsem stáhnul na stránkách programu MSDN Academic Alliance, která je pro studenty Vysokého učení technické v Brně zdarma ke stažení.

Komunikace mezi MIDI kontrolerem ControlFREAK a počítačem je realizována pomocí převodníku MIDI-USB. Převodník má jeden MIDI vstup, jeden MIDI výstup a USB připojení, které je zároveň použito jako napájení.

2.1 Specifikace MIDI kontroleru ControlFREAK

Tento MIDI kontrolér je jedním z výrobků od firmy Kenton Electronics. Konkrétně ControlFREAK Studio Edition, která je vybavena o dvojnásobný počet tahových potenciometrů a tlačítek víc, než ControlFREAK Original edition, jedná se tedy o kontroler, který má 16 plně programovatelných tahových potenciometrů a tlačítek uložitelných do 64 programů. Dále obsahuje 8 programovatelných globálně funkčních tlačítek určených pro všechny programy. Každý program je možné naprogramovat vyslanou MIDI SysEx zprávou nebo načítat a ukládat do počítače.

Kontroler můžeme ovládat pomocí 8 editačních tlačítek a otočného kolečka(Data entry) pro výběr dat a jednotlivé hodnoty parametrů nastavovat změnou polohy posuvného potenciometru nebo přečtením z tzv. real-time hodnot. Aktuální volená data se zobrazují na dvouřádkovém 16 znakovém LCD displeji.

Jak již bylo zmíněno, MIDI kontroler umožňuje editaci jednotlivých programů, které se aktivují pomocí tlačítka EDIT. Jeho menu nabízí funkce pro kopírování programů, výpis paměti, editaci tlačítek a tahových potenciometrů. Data tahového potenciometru a data tlačítka zabírají stejně velkou paměť, i přesto, že data tlačítka obsahují navíc položku o režimu tlačítka. Oběma prvkům se nastavují minimální a maximální hodnoty, z nichž pro potenciometr tyto hodnoty představují rozsah hodnot, pro tlačítko vyjadřují rozhodovací úrovně pro stavy vypnuto nebo zapnuto.

Dále můžeme editovat název celého programu, názvy tahových potenciometrů a tlačítek. Jednotlivý potenciometr nebo tlačítko můžeme pojmenovat až 16 znaky a následně mu přidělit jeden nebo více MIDI zpráv s maximální délkou 45 bajtů, který se má při akci provádět.

Programovatelná tlačítka pracují ve třech režimech. První je, že se při stisku tlačítka vyšle pouze jeden povel. V druhém režimu se při stisku tlačítka vyšle jeden povel a po puštění druhý povel a v posledním režimu se při prvním stisku tlačítka vyšle jeden povel a při druhém stisku jiný povel.

Celý ControlFREAK je umístěn v kovovém pouzdře, na zadní straně se nachází MIDI rozhraní s trojicí konektorů (In, Out, Thru) a dvěma vstupy s konektory 1/4"TS pro připojení pedálů (potenciometrů nebo spínačů).

Obr. 2.1: Ilustrační obrázek ControlFREAKu.

2.2 Specifikace MIDI kontroleru Bitstream 3x

MIDI kontroler Bitstream 3x (obr. 2.2) vyrobený francouzskou firmou Wave Idea, vzniklý v roce 1998, se chlubí mnoha druhy ovládacích prvků, jako je 24 otočných či 8 posuvných potenciometrů, 8 tlačítek, crossfader, X-Y joystick, dotykový ovladač, dále pak tlačítka pro ovládání sekvenceru, 3 otočné potenciometry pro práci s automatizací a několik prvků pro pohyb v menu. Celkově tedy disponuje 62 ovládacími prvky. Na zadní straně se nachází dva výstupní konektory MIDI, vstupní konektor, konektor MIDI thru, konektor Expansion pro připojení dalších analogových ovládacích prvků a je zde možnost připojení samostatného napájení. Pro komunikaci s počítačem je možné využít rozhraní MIDI, ale také rozhraní USB, jehož konektor vyžaduje propojovací kabel USB typu B [15].

Kontroler pracuje ve dvou základních režimech. První z nich je standardní, který je předprogramovaný a nabízí omezené množství editace, a druhý režim je uživatelský, který umožňuje přístup ke všem parametrům. Největší odlišnost najdeme v množství přenastavitelných parametrů. Zatímco standardní podporuje výběr pouze mezi dvěma MIDI zprávami (nota zapnuta a změna kontroleru), uživatelský režim podporuje libovolnou MIDI zprávu.

Uživatel zde ještě může nastavit rozšiřující parametry, a to buď pomocí menu nebo softwaru, například nastavení krajních hodnot pro každý prvek, zpoždění nebo způsob odesílání MIDI zpráv.

Co se týká možnosti volby rozdílných programů jako u kontroleru ControlFreak, u tohoto kontroleru jednotlivé prvky mohou být přiřazeny do 21 nezávislých skupin (nazývaných „Group“) umožňující najednou poslat až 21 odlišných zpráv.

Tento kontroler nabízí ještě další množství parametrů, jako je 32 virtuálních MIDI kanálů (16 kanálů na každý MIDI výstup), programovatelný nízkofrekvenční oscilátor LFO (4 základní průběhy), 100 scén pro záznam pohybu prvků, apod.

Obr. 2.2: Ilustrační obrázek Bitstream 3x.

2.3 Formát SysEx zprávy pro CotrolFREAK

Formát SysEx zprávy si ukážeme na příkladu, jehož struktura je zobrazená v tabulce 2.1. Výsledkem bude naprogramované tlačítko, které při stlačení bude vysílat MIDI zprávu *Note on*, na kanálu č. 2, s tónem odpovídající aktuálně nastavené hodnotě zvoleného potenciometru a maximální dynamikou.

Struktura dané zrpávy:

- *Začátek SysEx* ($F0) - začátek System Exclusive zprávy.
- *ID výrobce* ($00 $30 $90) - v našem případě tříbajtové identifikační číslo výrobce (Kenton Electronics).
- *ID zařízení* ($10) - identifikační číslo výrobku (ControlFREAK).

Tab. 2.1: Struktura SysEx zprávy pro ControlFREAK [1].

Začátek SysEx -> ID výrobce -> ID zařízení -> Sub-ID -> Program -> ->ID klávesy -> DATA -> Konec SysEx

- *SUB-ID* ($40) - identifikace, že se jedná o zprávu typu Single Data Dump, tj. odeslání bloku dat jednoho ovladače zvoleného programu (64B datový blok).
- Číslo zvoleného programu ($26) - výběrem jednoho z 64 programů.
- Číslo zvoleného tlačítka nebo tahového potenciometru ($11) - 2. tlačítko (viz tab. 2.2).
- *DATA* - jedná se o blok dat velikosti 64B. Ve skutečnosti je posíláno 128B, kdy se v prvním bajtu posílají méně významné 4 bity (LSB) informace a v druhém významnější 4 bity (MSB) informace, proto nyní místo čísla $37 musíme poslat $07 $03. SysEx zpráva nesmí v bloku obsahovat žádná stavová data, proto je každý bajt rozdělený na 2 bajty, ControlFREAK si informaci zpětně sestaví do 1 batju.

  Část *DATA* obsahuje následující hodnoty:
  - Název tlačítka - 16 znaků alfanumerického kódu rozdělené po 2 bajtech (TLACITKO NOTE ON - $04 $07 $0C $06 $01 $06 $03 $06 $09 $06 $04 $07 $0B $06 $0F $06 $0F $0F...).
  - Rozhodovací úroveň při stavu vypnuto ($00 $00) - pokud se jedná o tahový potenciometr, je zde minimální hodnota.
  - Úroveň při stavu zapnuto ($0F $07) - pokud se jedná o tahový potenciometr, je zde maximální hodnota.
  - Režim tlačítka ($00 $00) - při stisku vyšle zprávu.
  - Set midi channel 4 ($01 $0F) - nastavení MIDI kanálu na kanál 4.
  - Adresa note on ($02 $0B) - nejedná se přímo o MIDI příkaz *note on*, ale pouze její adresa uložená v kontroleru ControlFREAK. Všechny jiné adresy příkazů jsou zobrazeny v tab. A.1.
  - Set data form slider 4 ($0A $0A) - Nota odpovídající hodnotě 4. potenciometru (tab. A.1).
  - Dynamika ($07 $0F) - maximální dynamika (fortisimo forte).
- *Konec SysEx* ($F7) - konec System Exclusive zprávy.

Tab. 2.2: Tabulka adres tlačítek a kláves.

Adresa	Prvek
0-15	Tahový potenciometr
16-31	Tlačíko zapnuto
32-47	Tlačíko vypnuto
48-55	Funkční klávesa zapnuta
64-71	Funkční klávesa vypnuta
96-103	Programová data
112-119	Globalní data

2.4 Formát SysEx zprávy pro Bitstream 3x

SysEx zprávy, respektive zprávy pro Bitstream 3x, využijeme v situaci, kdy potřebujeme přeprogramovat nebo nahrát jakékoliv parametry z nebo do zařízení. Tyto zprávy mohou modifikovat jak globální parametry, tak parametry prvku.

Globální parametry můžeme chápat jako parametry společné pro všechny prvky. Příkladem může být výběr skupiny, scény nebo přejmenování názvu skupiny, apod.

Modifikace parametrů prvků vyžaduje příjem dvou SysEx zpráv. V první zprávě jsou obsaženy všechny parametry pro daný prvek. Mezi ně patří i MIDI zprávy, které se při akci odesílají a v druhé zprávě je poté posílán název prvku, který se bude na LCD displeji zobrazovat.

V tabulce 2.3 je popsán obecný formát těchto SysEx zpráv. *ID0* a *DATA* plní mnoho funkcí, a proto zde budou vysvětleny jejich hlavní části.

Tab. 2.3: Obecný formát SysEx zpráv pro Bitstream 3x[17].

Hodnota	Popis	Poznámka
$F0	Záčátek SysEx	-
$00 $20 $4F	ID výrobce	Wave Idea
$00 $01	ID zařízení	Bitstream 3x
ID0	SysEx ID 0	tabulka A.2
ID1	Délka části SysEx zprávy DATA	Rozsah 0-127
ID2	Číslo prvku	Rozsah 8-67
ID3	Číslo skupiny	Rozsah 0-20
DATA	SysEx Data	tabulka A.2
CS	Kontrolní součet	Součet *DATA* modulo 128
$F7	Konec SysEx	-

Nejdůležitější vlastnost bajtu *ID0* spočívá ve „Významu SysEx zprávy“ (modifikaci parametrů prvku nebo názvu prvku, zobrazovaný na displeji,). Rozlišují se v kombinaci dvou nejnižších bitů. Další bity označují směr odesílání dat, požadavek na potvrzení přijetí nebo žádost o vypsání parametrů prvku (viz tab. A.2).

Další část tvoří *DATA*. Jednotlivé bajty se rozdělují do dvoubajtů v opačném pořadí, než to bylo u Control Freaku (nejprve MSB a pak LSB).

Pro lepší pochopení této části byl v tab. 2.4 vytvořen příklad odpovídající naprogramovanému prvku č. 40 (přiřazený do skupiny nula), který vysílá MIDI zprávu $94 $45 $7F s názvem prvku NOTE ON.

Z tohoto příkladu je patrné, že z první SysEx zprávy lze MIDI zprávu lehce vyčíst (tučně vyznačená), než to bylo u Control Freaku, kde MIDI zpráva odpovídala zcela odlišné hodnotě a navíc číslo kanálu bylo odesíláno zvlášť. Další parametry

nastavují, po jaké křivce má případný potenciometr měnit své hodnoty, minimální a maximální rozsahy potenciometru nebo na jaký port se mají MIDI zprávy posílat, atd. Podrobný popis všech parametrů je uveden v tabulce A.3.

Druhá SysEx zpráva se věnuje pouze názvu prvku, který se bude při stlačení tlačítka nebo pohybu potenciometru zobrazovat, proto tedy část *DATA* tvoří alfanumerický řetězec.

Posledním parametrem před ukončením celé SysEx zprávy je kontrolní součet, který je možný vypnout (pouze pro globální data). Výpočet kontrolního součtu vznikne součtem všech bajtů části *DATA*, modulárně děleného 128.

Tab. 2.4: Příklad SysEx zprávy pro Bitstream 3x [17].

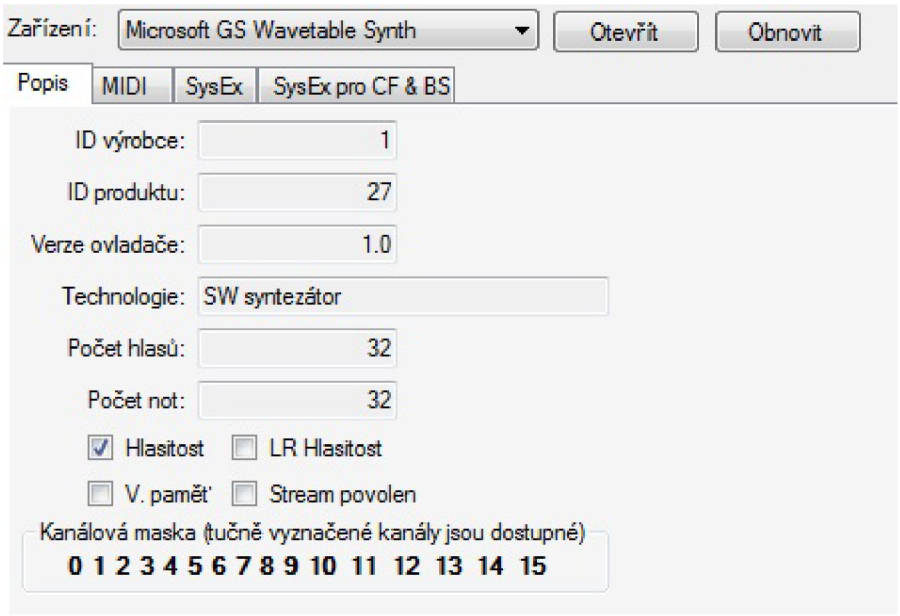
První SysEx zpráva	**Druhá SysEx zpráva**	Popis
$F0	$F0	-
$00 $20 $4F $00 $01	$00 $20 $4F $00 $01	Wave Idea Bitstream 3x
$1B $1C $28 $00	$19 $20 $28 $00	tabulka A.2 Délka části *DATA* Prvek Skupina
$02 $08 $0B $0C $00 $00 $00 $03 $00 $00 $00 $03 $00 $00 $00 $00 $07 $0F $00 $00 $00 $00 **$09 $04** **$04 $05 $07 $0F**	**$04 $0E $04 $0F $05 $04** **$04 $01 $02 $00 $04 $03** $02 $00 $02 $00 $02 $00 $02 $00 $02 $00 $02 $00 $02 $00 $02 $00 $02 $00 $02 $00	SysEx data
$69	$50	DATA modulo 128
$F7	$F7	-

2.5 Popis aplikace

Aplikace umožňuje poslat jak jednoduchou MIDI zprávu, tak i SysEx zprávu konkrétnímu zařízení, a to buď pomocí zápisu do řetězců nebo pomocí grafického zobrazení. Celý řetězec SysEx zprávy lze uložit na disk ve formátu XML a zpětně ji i s nastaveným parametrem zarovnání nahrát do aplikace. Tento program je rozdělen do čtyř záložek.

2.5.1 Záložka Popis

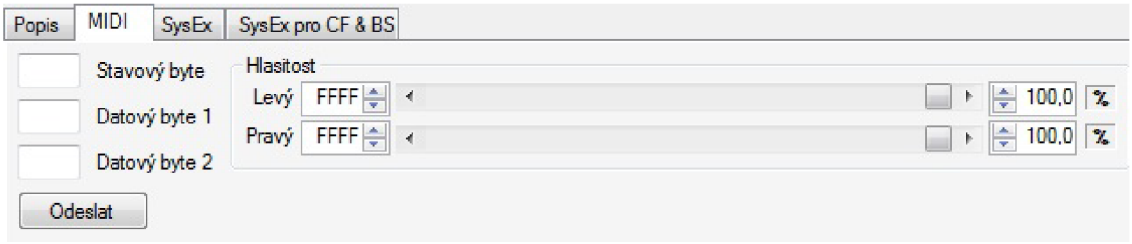
Tato část (obr. 2.3) zobrazuje informace o dostupných MIDI rozhraních na počítači, například název brány, verze ovladače, dostupné kanály, atd. V rolovacím menu můžete přepínat mezi jednotlivými rozhraními a tlačítkem *Otevřít* jej otevřít a tím ho připravit na vysílání dat, popřípadě zaktualizovat dostupnost stávajícího zařízení nebo nově připojeného zařízení.



Obr. 2.3: Záložka Popis.

2.5.2 Záložka MIDI

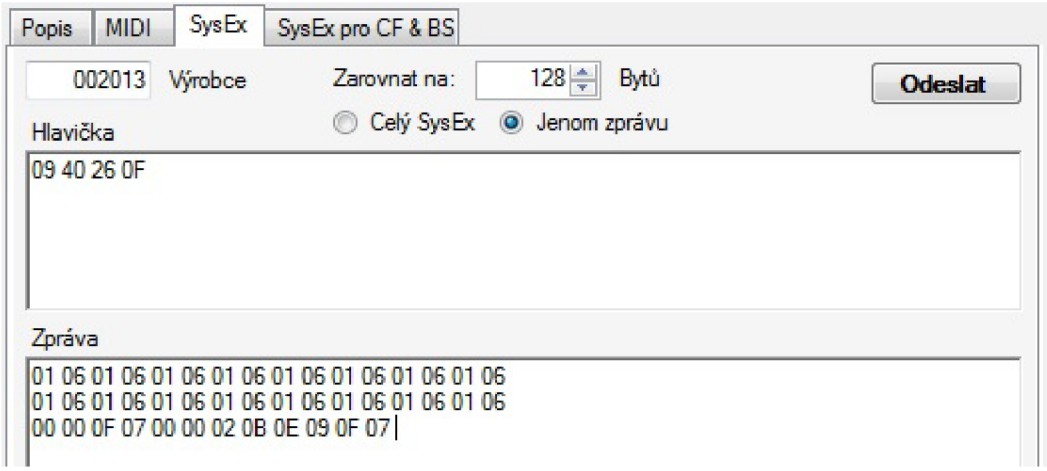
Záložka MIDI (obr. 2.4) umožňuje posílat kanálové i systémové MIDI zprávy, kromě SysEx zpráv, kde je potřeba do stavového bajtu zadat požadovaný příkaz s jeho parametry, a odeslat zprávu do zařízení.



Obr. 2.4: Záložka MIDI.

2.5.3 Záložka SysEx

Další záložka je určená pro posílání System Exclusive zpráv, která obsahuje tři textová pole (*Výrobce, Hlavička a Zpráva*). Do prvního textového pole *výrobce* se zadává identifikační číslo výrobce, do druhého textové pole *hlavička* se zapisují položky ID zařízení nebo SUB ID. Poslední textové pole je *zpráva* určené pro parametry konkrétního prvku. Celou SysEx zprávu můžeme zarovnat na libovolný počet bajtů, protože některá zařízení vyžadují zarovnání na konkrétní blok dat. Příkladem je kontroler ControlFREAK, který bez vhodného zarovnání nereaguje na žádné SysEx zprávy. Zarovnáním je pak docíleno toho, že se datový blok dorovná na příslušnou velikost bajty $0F.

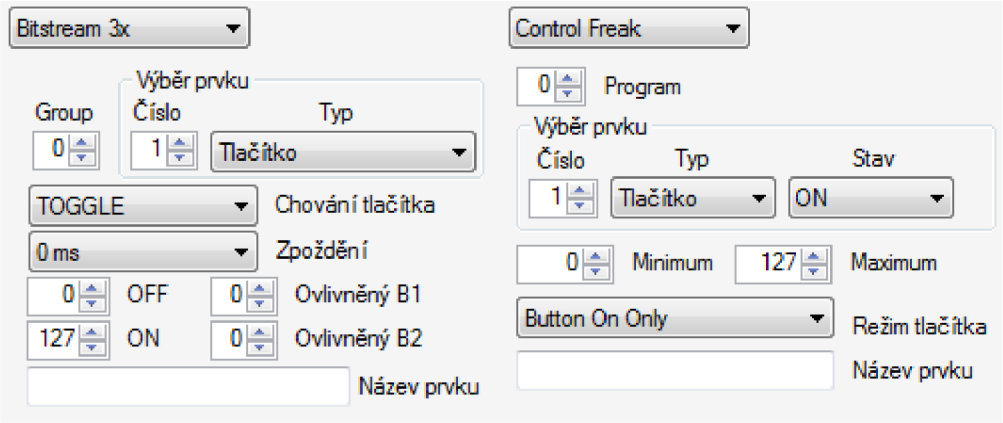


Obr. 2.5: Záložka SysEx.

2.5.4 Záložka SysEx pro CF & BS

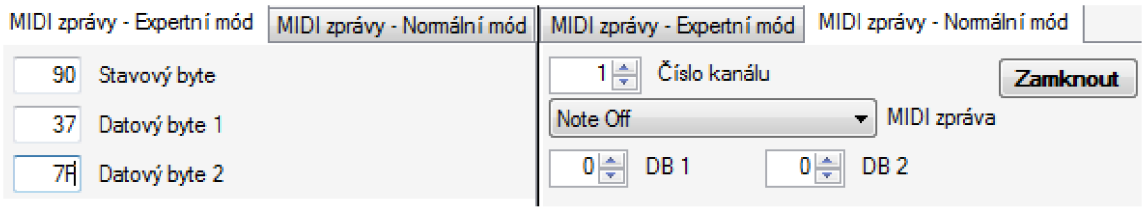
Poslední záložka je navržená tak, aby uživateli usnadnila naprogramovat SysEx zprávu intuitivně, bez znalosti vnitřního uspořádání zprávy za použití prvků formuláře. Přepínání mezi jednotlivými kontrolery zajišťuje rolovací menu, a protože každý kontroler je odlišný, má proto i rozdílné menu (obr.2.6).

MIDI zprávy, které se mají pomocí naprogramovaného prvku odesílat, se dají přidávat do databáze a následně tlačítkem *Odeslat SysEx* jsou poslány do zařízení jako SysEx zpráva s vhodně přiřazenými parametry pro úspěšné naprogramování. Tyto zprávy se zadávají ve dvou módech (Expertní a Normální, obr. 2.7). Expertní mód je navržen pro účely výuky a vyžaduje znalost všech částí MIDI zprávy, oproti Normálnímu módu, ve kterém je možné vybírat MIDI zprávy z rolovacího menu a také číslo kanálu. Normální mód je však přístupný pouze po zadání správného hesla,



Obr. 2.6: Výběr kontroleru a jejich parametry.

které je možné změnit (*Soubor->Změnit heslo*). Jelikož je heslo uloženo v registrech, je po restartovaní aplikace vyžadováno již z registrů načtené heslo.



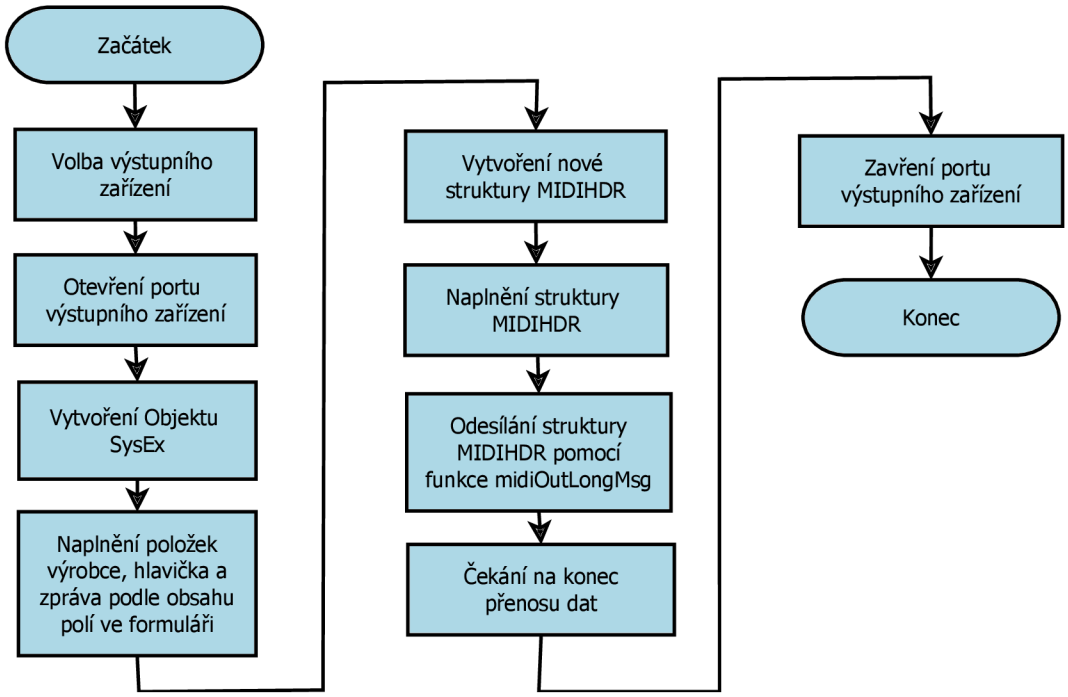
Obr. 2.7: MIDI zprávy - Expertní a Normalní mód.

2.6 Princip posílání SysEx zpráv

Základní kostru celého programu pro práci s MIDI rozhraním tvoří funkce a struktury, z knihovny Windows Multimedia System. Tato knihovna se objevila již ve Windows 3.11, její poslední aktualizace byla provedena ve Windows 2000. Program obsahuje cca pět tisíc řádků, z nichž asi polovina je naprogramovaný kód. Všechny třídy, struktury a metody jsou logicky po částech rozděleny do „.h“ a „.cpp“ souborů, aby se daly jednoduše využít i pro jiné aplikace, a aby byl kód lépe čitelný.

Na obr. 2.8 je zobrazen vývojový diagram popisující základní princip posílání SysEx zpráv. Každá entita vývojového diagramu plní svojí důležitou funkci, bez které by program nemohl fungovat. V této kapitole budou jednotlivé entity popsány.[1]

[1] Ze zdrojového kódu jsou vyjmuty pouze principiální části.



Obr. 2.8: Vývojový diagram posílání SysEx zpráv.

2.6.1 Volba výstupního zařízení

Pro výběr výstupního zařízení je nejprve nutné zjistit počet všech připojených zařízení. O to se stará funkce `midiOutGetNumDevs` [4] (viz výpis 2.1). Nemá žádné vstupní parametry a vrací pouze aktuální počet všech dostupných zařízení. Pomocí něho je z formulářové nabídky možné přistupovat ke všem jeho vlastnostem. Uložení vlastností zajišťuje statická metoda `get_dev_desc`. Vytvoří strukturu `MIDIOUTCAPS` [14] a zavolá funkci `midiOutGetDevCaps` [3], která naplní celou tuto strukturu.

Mezi jeho vstupními parametry patří číslo aktuálně zvoleného zařízení (proměnná `port_num`), ukazatel na strukturu a velikost struktury `MIDIOUTCAPS`.

Výpis kódu 2.1: `MIDI_accessor.cpp` - volba výstupního zařízení.

```
void MIDI_accessor::get_dev_desc(UINT port_num, MIDIOUTCAPS* out)
{ MIDIOUTCAPS vals; //vytvoření struktury výstupního zařízení
  midiOutGetDevCaps(port_num, &vals, sizeof(MIDIOUTCAPS));}
  //naplnění struktury vlastnostma výstupního zařízení
```

2.6.2 Otevření portu výstupního zařízení

Nyní jsme se dostali do fáze, kdy musíme připravit zařízení na vysílání dat. K tomu využijeme funkci `midioutopen` [7] (viz výpis 2.2). Vstupním parametrem je struk-

tura `LPHMIDIOUT lphmo`, která slouží jako ukazatel na `HMIDIOUT` handle.[2] Struktura `HMIDIOUT` se používá jako identifikátor otevřeného zařízení a je volána ostatními funkcemi spojené s tímto zařízením. Dalším parametrem je `uDeviceID`, totožné s číslem pro získání vlastností zařízení. Poslední tři proměnné zde nejsou použity.

Výpis kódu 2.2: `Windows.h` - funkce `midiOutOpen` [7].

```
MMRESULT midiOutOpen(
  LPHMIDIOUT lphmo, //ukazatel na handle
  UINT uDeviceID, //id zařízení (MIDI-USB)
  DWORD_PTR dwCallback, //callback
  DWORD_PTR dwCallbackInstance,
  DWORD dwFlags
);
```

Pro otevření portu je nutné vytvořit nový objekt (viz výpis 2.3), třídy `MIDI_accessor`, se vstupním parametrem čísla zvoleného zařízení.

Výpis kódu 2.3: `Form.h` - vytvoření objektu `MIDI_accessor`.

```
MIDI_accessor *mac; //dynamické vytvoření objektu
mac = new MIDI_accessor((UINT)cb_device_select->SelectedValue);
```

Privátní proměnnou `MIDI_accesor` tvoří struktura `HMIDIOUT` (viz výpis 2.4). Vytvořený objekt zavolá konstruktor, který otevře port výstupního zařízení. Podmínka pak testuje, zda nedošlo při otevírání k chybě. Pokud ano, vyvolá výjimku.

Výpis kódu 2.4: `Form.h` - otevření výstupního portu.

```
HMIDIOUT device; //handle zařízení
void MIDI_accessor::MIDI_accessor(UINT port_num){
  int flag = midiOutOpen(&device, port_num, 0, 0, CALLBACK_NULL;
  //otevři zařizení
  if (flag != MMSYSERR_NOERROR){//pokud došlo k chybě, spusť výjimku
    throw MIDI_exception(get_err_desc(flag));}}
```

2.6.3 Vytvoření objektu SysEx

Objekt `SysEx` má za úkol reprezentovat řetězec celé SysEx zprávy (viz výpis 2.5). Skládá se z proměnných `man, header a msg` (výrobce, hlavička a zpráva) typu řetězec. Jejich metody se starají a naplnění těchto řetězců příslušnými hodnotami.

[2] Do českého překladu rukojeť nebo držadlo.

Výpis kódu 2.5: `SysEx.h a SysEx.cpp` - objekt SysEx a jeho metody.

```
class SysEx{
private:
  string man; //ID výrobce
  string header; //hlavička
  string msg; //zpráva
public:
  SysEx();//konstruktor
  void set_man(string&); //naplněni řetězcem proměnnou man
  void set_header(string&); //naplněni řetězcem proměnnou header
  void set_msg(string&); //naplněni řetězcem proměnnou msg
};

SysEx::SysEx(){//konstruktor
  man = header = msg = "";//výchozí naplnění řetězců prázdným řetězcem
}
/*metody*/
void SysEx::set_man(string& str)//naplnění položky ID výrobce
  if ((str.length() == 1 && str[0] != 0) || (str.length() == 3 &&
      str[0] == 0)){//podmínka, zda není špatně napsán kód výrobce
    test_msb(str);
    man = str;}//přiřazení do proměnné man požadovaný řetězec
  else{
    throw MIDI_exception("Neplatne ID vyrobce.");}}//chyba

void SysEx::set_man_parse(string str){//formátování zprávy
  set_man(hexa_from_ASCII_to_str(str, misc_functions::message));}
```

2.6.4 Naplnění položek výrobce, hlavička a zpráva

Naplnění položek výrobce, hlavička a zpráva probíhá následovně:

1. Po vytvoření objektu `SysEx` se zavolá metoda `set_man_parse` (viz výpis 2.6), která má za úkol převést řetězec znaků takovým způsobem, kde dva znaky odpovídají jednomu odeslanému bajtu. Příkladem je řetězec „00204F“ (ID výrobce Bitstreamu 3x). Po zavolání této metody se řetězec převede na znaky, které odpovídají alfanumerické hodnotě $00, $20 a $4F, tím je docíleno, že každé dva znaky tvoří jeden bajt (pro nás má význam pouze alfanumerická hodnota znaku).
2. Následně se zavolá metoda `set_man`. V ní se ověří, zda je správně vyplněn „výrobce“. A řetězec uloží do proměnné `man`.
3. Zavolá se metoda `set_header_parse`, která vykoná totéž, co v předchozích dvou bodech s rozdílem, že řetězec „hlavička“ uloží do proměnné `header`.

Výpis kódu 2.6: `Form.h` - naplnění položek výrobce a hlavička.

```
SysEx sysex; //Vytvořeni objektu SysEx
sysex.set_man_parse("00204F"); //uloženi vyrobce do promenne man
sysex.set_header_parse("0001");//uloženi hlavičky do proměnné header
```

4. Naplnění řetězce `part1, part2 a calc_checksum` příslušnými hodnotami (viz výpis 2.7).[3] Do řetězce `part1` se uloží číslo prvku a skupina. Řetězec `part2` tvoří část DATA kontroleru Bitstream 3x, nebo ControlFREAK. Funkce `calc_checksum` vypočítá kontrolní součet z řetězce `part2`, která je nutná pro kontroler Bitrsream 3x.
5. Každý znak řetězce `part2` se rozdělí na dva znaky ($9A -> $9, $A). Rozklad provádí metoda `str_to_two`.
6. Všechny řetězce `part1, part2 a calc_checksum` se pomocí metody `sysex.set_msg` uloží do proměnné `msg`.
7. Metoda `send_sysex_msg` odešle SysEx zprávu.

Výpis kódu 2.7: `Form.h` - naplnění položky `msg` a odeslání zprávy.

```
...
part2 = misc_functions::str_to_two(part2);
//rozdělení na dva bajty
sysex.set_msg(part1 + part2 + calc_checksum(part2));
// uložení řetězce part1, part2, CS do promenné msg
mac->send_sysex_msg(sysex);
//odeslání SysExu
```

2.6.5 Vytvoření struktury MIDIHDR

Struktura `MIDIHDR` [13] definuje hlavičku, která se používá k identifikaci SysEx zpráv (viz výpis 2.8). Pro nás jsou důležité proměnné `lpData a deBufferLenght`. Do proměnné `LpData` se uloží řetězec celé SysEx zprávy a jeho délka se uloží do `dwBufferLength`.

Výpis kódu 2.8: `Windows.h` - struktura `MIDIHDR` [13].

```
typedef struct midihdr_tag {
  LPSTR               lpData;//SysEx zpráva
  DWORD               dwBufferLength;//délka zprávy
  DWORD               dwBytesRecorded;//aktuální množství v paměti
  DWORD_PTR           dwUser;//uživatelská data
  DWORD               dwFlags;//příznak
  struct midihdr_tag  *lpNext;//rezervoávno
  DWORD_PTR           reserved;//rezervováno
```

[3] Z důvodu přehlednosti naplnění řetězců `part1, part2 a calc_checksum` nejsou zobrazeny v tomto kódu.

```
  DWORD                dwOffset;//offset paměti
  DWORD_PTR            dwReserved[4];//rezervováno
} MIDIHDR, *LPMIDIHDR;
```

2.6.6 Naplnění struktury MIDIHDR

Před naplněním struktury `MIDIHDR` se nejprve zavolá metoda `send_sysex_msg(SysEx& sx)` (viz výpis 2.9), ta zavolá další metodu (`get_sysex()`), která sečte řetězce `man, header a msg` do jednoho, a přidá na začátek celého řetězce $F0 a na konec $F7. Poté se zavolá metoda `send_sysex_msg(string& str)`. Ta vytvoří z C++ řetězce C řetězec a zjistí délku celého řetězce. Naposled se zavolá metoda `send_sysex_msg(const CHAR msg, UINT msglen)`, která vytvoří dynamicky alokované pole znaků, poté do něj nakopíruje celou SysEx zrávu.

Nyní konečně můžeme vytvořit a naplnit strukturu `MIDIHDR`.

Výpis kódu 2.9: `MIDI_accessor.cpp` - naplnění struktury `MIDIHDR`.

```
void MIDI_accessor::send_sysex_msg(SysEx& sx){
  send_sysex_msg(sx.get_sysex());}//získání celého SysEx řetězce

void MIDI_accessor::send_sysex_msg(string& str){
  send_sysex_msg(str.c_str(), str.length());}
//vytvoření z C++ řetězce, C řetězec a zjištění délky řetězce

void MIDI_accessor::send_sysex_msg(const CHAR* msg, UINT msglen){
  CHAR *hlp = new CHAR[msglen + 1]; //pole znaků
  memcpy(hlp, msg, msglen + 1);//zkopírování zprávy do pole

  MIDIHDR header; //vytvoření struktury MIDIHDR
  header.lpData = hlp; //uložení SysEx zprávy
  header.dwBufferLength = header.dwBytesRecorded = msglen;
  //délka zprávy
  header.dwFlags = 0;
```

2.6.7 Odesílání struktury MIDIHDR pomocí funkce midiOutLongMsg

Funkce `midiOutPrepareHeader` [5] připraví SysEx zprávu na odeslání a funkce `midiOutLongMsg` [6] tuto zprávu odešle (viz výpis 2.10). Obě funkce jsou testovány v podmínce. To znamená, že pokud dojde při přípravě na odesílání (či při odesílání) k chybě, vypíše se výjimka.

Výpis kódu 2.10: `MIDI_accessor.cpp` - odesílání struktury `MIDIHDR`.

```
  UINT flag;//příznak
  if ((flag = midiOutPrepareHeader(device, &header, sizeof(MIDIHDR)
     )) || (flag = midiOutLongMsg(device, &header, sizeof(MIDIHDR))
     )){//pokud dojde k chybě při odesílání, spusť výjimku
    delete[] hlp;//odalokace SysEx zprávy
    throw MIDI_exception(get_err_desc(flag));//výjimka
    }
```

Jak funkce `midiOutLongMsg`, tak i `midiOutPrepareHeader` mají stejné vstupní parametry (viz výpis 2.11). Obsahuje strukturu `HMIDIOUT` (handle zařízení), referenci na strukturu `MIDIHDR` a její velikost.

Výpis kódu 2.11: `Windows.h` - funkce `midiOutLongMsg`.

```
MMRESULT midiOutLongMsg(
    HMIDIOUT  hmo,//handle zařízení
    LPMIDIHDR lpMidiOutHdr,//ukazatel na strukturu MIDIHDR
    UINT      cbMidiOutHdr//velikost struktury MIDIHDR
);
```

2.6.8 Čekání na konec přenosu dat

V podmíněném cyklu „`while`“ se testuje, zda došlo ke konci přenosu dat (viz výpis 2.12). Pokud nedošlo ke konci přenosu dat, funkce `Sleep` [10] počká deset milisekund a podmínku otestuje znovu. Pokud však došlo ke konci přenosu, celé pole SysEx zprávy se odalokuje. Zde metoda `send_sysex_msg(const CHAR msg, UINT msglen)` končí.

Výpis kódu 2.12: `MIDI_accessor.cpp` - čekání na konec přenosu dat.

```
  while (midiOutUnprepareHeader(device, &header, sizeof(MIDIHDR))
     == MIDIERR_STILLPLAYING){//podmínka, zda je konec přenosu dat
    Sleep(10);//čekej 10ms
  }
  delete[] hlp;}//odalokace SysEx zprávy
```

2.6.9 Zavření portu výstupního zařízení

Stlačením tlačítka „Zavřít“ (ve formulářovém okně) se zavolá destruktor, který obsahuje funkce `midiOutReset` [8] a `midiOutClose` [2] (viz výpis 2.13). Funkce `midiOutReset` přeruší probíhající MIDI zprávy „Note on“na všech kanálech a funkce `midiOutClose` uzavře výstupní port aktuálně otevřeného zařízení.

Výpis kódu 2.13: `MIDI_accessor.cpp` - zavření portu výstupního zařízení.

```
~MIDI_accessor(){
  midiOutReset(device);//přerušení posílání zprav
  midiOutClose(device);//uzavření výstupního portu
}
```

2.7 Uložení hesla do .ini souboru a vytvoření záznamu v registrech

Jak již bylo zmíněno, heslo nám umožňuje přístup do záložky „Normální mód“. Při prvním spuštění heslo není nastavené, a proto ho je nutné nastavit. Otevře se nám formulářové okno pro zadání výchozího hesla, které následně heslo uloží do .ini souboru a vytvoří záznam v registrech.

Před spuštěním formulářového okna pro zadání hesla se zavolá metoda `get_password` (viz výpis 2.14). Ta vytvoří objekt `pp`, třídy `PrivateProfile`. Jeho konstruktor nastaví výchozí cestu a sekci k souboru, kde .ini soubor bude, nebo se zde již nachází. Poté se vytvoří nový řetězec a pokud .ini soubor existuje, načte se z něj heslo a spustí se hlavní formulářové okno. Ale pokud heslo není nastaveno, zavolá se metoda `prepare_registry` (viz výpis 2.15).

Výpis kódu 2.14: `Form.h` - metoda pro práci s .ini souborem a registry.

```
  String^ get_password(){
      PrivateProfile pp(TEXT(".\\MIDI.ini"), TEXT("MIDI"));
      //nastavení výchozí cesty a sekce k .ini souboru
      TSTRING pass = pp.get_string(TEXT("Password"));
      //načtení hesla z .ini souboru
      if (pass == TEXT("")){//pokud je prázdné, zavolej okno pro vytvoření
                            //nového hesla
        prepare_registry();//vytvoření záznamu v registrech

        PassForm^ pf = gcnew PassForm(PassForm::PassFormType::NEW);
        //vytvoření nového formulářového okna
        if (pf->ShowDialog(this) == System::Windows::Forms::
            DialogResult::OK){
          set_password(pf->get_new_pass());}
        else{//ověření správně vyplněného hesla
          this->Close();}}//uzavření formulářového okna
      else{
        Password = gcnew String(pass.c_str());}
        //uložení hesla do .ini souboru
      return Password;
    }
```

Metoda `prepare_registry` má za úkol vytvořit záznam v registrech. Záznam se uloží do složky začínající stromovou strukturou `HKEY_CURRENT_USER`. Je to z důvodu, že kdybychom ho ukládali do `HKEY_LOCAL_MACHINE`, potřebovali bychom na uložení záznamu práva administrátora. Po vytvoření záznamu tato metoda končí.

Výpis kódu 2.15: `Form.h` - uložení záznamu do registrů.

```
void prepare_registry(){
    TCHAR szData[] = TEXT("USR:MIDI\\MIDI");
    //text, který se bude zobrazovat v záznamu

    Reg reg(HKEY_CURRENT_USER);//vstup do sekce HKEY_CURRENT_USER
    reg.create_subkey(TEXT("SOFTWARE\\Microsoft\\Windows NT\\
        CurrentVersion\\IniFileMapping\\MIDI.ini"), KEY_WRITE);
    //vytvoření podklíče
    reg.set_value(TEXT("MIDI"), VALUE_TYPE::SZ, (BYTE*)szData,
        sizeof(szData));//vytvoření záznamu
    }
```

Nyní se spustí formulářové okno pro zadání nového hesla. Po zadání hesla, které musí obsahovat minimálně šest znaků, se zavolá metoda `set_password` (viz výpis 2.16) a ta uloží metodou `write_string` nové heslo do .ini souboru. Poté se toto okno ukončí a otevře se hlavní formulářové okno.

Následně při dalším spuštění této aplikace není vyžadováno zadání výchozího hesla.

Výpis kódu 2.16: `Form.h` - uložení hesla do .ini souboru.

```
void set_password(String ^s){
      Password = s;
      PrivateProfile pp(TEXT(".\\MIDI.ini"), TEXT("MIDI"));
      //nastavení cesty a sekce k .ini souboru
      pp.write_string(TEXT("Password"), marshal_as<TSTRING>(s).
          c_str());//uložení hesla do .ini souboru
   }
```

2.8 Princip posílání MIDI zpráv - kanálová data

Pro začátek je nutné si zvolit výstupní zařízení a otevřít port. Stejně, jak to bylo u SysEx zpráv.

Kanálová MIDI data jsou omezena na jeden stavový a dva datové bajty, proto zde nepoužijeme funkci `midiOutLongMsg`, ale `midiOutShortMsg` [9]. Nevytváří se také struktura `MIDIHDR`, protože víme, jak dlouhá bude zpráva. K tomu nám budou stačit pouze tři proměnné reprezentující jednu MIDI zprávu. Tyto proměnné jsou

typu `CHAR`, o velikosti jednoho bajtu (viz výpis 2.17). Každá proměnná tedy bude reprezentovat jeden bajt.

Do proměnných `cmd, par1 a par2` se ve správném formátu přiřadí jeden stavový a dva datové bajty, které se budou odesílat. Převod na správný formát provádí metoda `hexa_from_ASCII_to_str`. Poté se ve dvou podmínkách otestuje, zda je správně zadán stavový (nejvýznamnější bit musí být roven jedné) a datové bajty (nejvýznamnější bit musí být roven nule). Následně se zavolá metoda `send_midi_msg`.

Výpis kódu 2.17: `Form.h - načtení stavového a dvou datových bajtů z form. okna.`

```
CHAR cmd = hexa_from_ASCII_to_str(marshal_as<string>(tb_midi_cmd->
    Text))[0];//stavový bajt
CHAR par1 = hexa_from_ASCII_to_str(marshal_as<string>(tb_midi_par1
    ->Text))[0];//1. datadový bajt
CHAR par2 = hexa_from_ASCII_to_str(marshal_as<string>(tb_midi_par2
    ->Text))[0];//2. datový bajt

if (!(cmd & 0x80))//ověření správně vyplněného stavového bajtu
  throw exception("Stavovy bajt musi zacinat bitem 1.");

if ((par1 & 0x80) || (par2 & 0x80))
//ověření správně vyplněného datového bajtu
  throw exception("Datove bajty museji zacinat bitem 0.");

mac->send_midi_msg(cmd, par1, par2);//příprava na odeslání MIDI zprávy
```

Objekt třídy `midi_accessor`, se vytváří při otevírání portu a metoda `send_midi_msg` se v této třídě nachází. Je zde vytvořen datový typ `union` (viz výpis 2.18), který nám umožní přistupovat k proměnným, do stejné oblasti dat. V ní je tedy vytvořená proměnná `word`, typu `DWORD` a pole `data`, typu `UCHAR`. Proměnná `word` má velikost čtyři bajty a má zároveň stejnou velikost jako celé pole `data`. Pole je naplněno proměnnými `cmd, par1 a par2` a poté je jako jedna proměnná (`data`) použita jako vstupní parametr k metodě se stejným názvem, ale pouze s jedním vstupním parametrem.

Výpis kódu 2.18: `MIDI_accessor.cpp` - změna formátu MIDI zprávy.

```
void MIDI_accessor::send_midi_msg(CHAR cmd, CHAR par1 = 0, CHAR
    par2 = 0){
  union//
  {
    DWORD word; //čtyřbajtová proměnná pro uložení celé zprávy
    UCHAR data[4];//jednobajtové pole o 4 prvcích reprezentující stejnou
  }                //oblast dat
  message;
```

```
  message.data[0] = cmd;//stavový bajt
  message.data[1] = par1;//1. datový bajt
  message.data[2] = par2;//druhý datový bajt
  message.data[3] = 0;//doplnění do čtyřech bajtů
  send_midi_msg(message.word);// odelsání MIDI zprávy
}
```

Nyní se konečně odešle celá MIDI zpráva, s využitím funkce `midiOutShortMsg` (viz výpis 2.19). V případě chybného odeslání se vypíše výjimka.

Výpis kódu 2.19: `MIDI_accessor.cpp` - odeslání MIDI zprávy.

```
void MIDI_accessor::send_midi_msg(DWORD msg){
  int flag;//příznak chyby
  if ((flag = midiOutShortMsg(device, msg)) != MMSYSERR_NOERROR){
  //odesílání MIDI zprávy
    throw MIDI_exception(get_err_desc(flag));
    //chyba při odesílání - výjimka
  }
```

Jako poslední nám zbývá zavřít port výstupního zařízení a tím ukončit komunikaci.

3 ZÁVĚR

V této bakalářské práci se úspěšně podařilo vytvořit aplikaci, umožňující naprogramovat zvolený ovládací prvek kontroleru ControlFREAK a Bistream 3x, pomocí MIDI System Exclusive zpráv. Tato aplikace byla ještě rozšířena o části umožňující poslat libovolné systémové a kanálové MIDI zprávy.

Zhodnocením obtížnosti programování pro oba kontrolery můžu říci, že naprogramovat vhodnou SysEx zprávu pro kontroler ControlFREAK bylo značně obtížnější, protože jsem nenašel dokumentaci vysvětlující řetězec části DATA, formátu SysEx zprávy pro tento kontroler. Navíc celou situaci ztěžovalo to, že odesílaná MIDI zpráva naprogramovaného prvku, odpovídala úplně jinému řetězci hodnot, obsaženém v SysEx zprávě, která se musela do kontroleru ControlFREAK odeslat. Na rozdíl od kontroleru Bitstream 3x, kde MIDI zpráva odesílaná ovládacím prvkem odpovídala totožnému řetězci obsaženém v SysEx zprávě. Celý tento formát byl kompletně vysvětlen v dokumentaci. Následně analýzou SysEx zpráv, odeslaných do počítače kontrolerem ControlFREAK, se mi podařilo přes program *Midi Analyzer* dohledat kompletní požadovaný formát.

Jednou z dalších odlišností bylo, že kontroler ControlFREAK má méně nastavitelných vlastností, které se dají naprogramovat. Celá SysEx zpráva je méně efektivně využívána, jelikož se vždy musí posílat 137 bajtů, přestože většinou není využita ani polovina SysEx zprávy. Avšak jednomu prvku lze přiřadit více MIDI zpráv, než u kontroleru Bitream 3x.

LITERATURA

[1] BARDEN, Anthony, Dave, MILLICHAMP, Peter HERMAN, John SMOUT, Paul GREGSON a GREGSON PAUL. KENTON ELECTRONICS. *USER MANUAL - MIDI CONTROL CENTRES: Control Freak STUDIO EDITION and Control Freak ORIGINAL*. UK., 1999. Dostupné z: www.kentonuk.com/kmanualspdf/cfrkman.pdf.

[2] Funkce midiOutClose. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798468%28v=vs.85%29.aspx.

[3] Funkce midiOutGetDevCaps. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798469%28v=vs.85%29.aspx.

[4] Funkce midiOutGetNumDevs. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798472%28v=vs.85%29.aspx.

[5] Funkce midiOutPrepareHeader. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798477(v=vs.85).aspx.

[6] Funkce midiOutLongMsg. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798474%28v=vs.85%29.aspx.

[7] Funkce midioutopen. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798476%28v=vs.85%29.aspx.

[8] Funkce midiOutReset. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798479%28v=vs.85%29.aspx.

[9] Funkce midiOutShortMsg. *Windows.h* [online]. 2015 [cit. 2015-05-18]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798481%28v=vs.85%29.aspx.

[10] Funkce Sleep. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/windows/desktop/ms686298%28v=vs.85%29.aspx.

[11] GUÉRIN, Robert. *Velká kniha MIDI: standardy, hardware, software*. Vyd. 1. Brno: Computer Press, 2004, 340 s. ISBN 80-722-6985-2.

[12] SCHIMMEL Jiří. *Studiová a hudební elektronika* Brno, 2012. ISBN 978-80-214-4452-2.

[13] Struktura MIDIHDR. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798449%28v=vs.85%29.aspx.

[14] Struktura MIDIOUTCAPS. *Windows.h* [online]. 2015 [cit. 2015-05-09]. Dostupné z: www.msdn.microsoft.com/en-us/library/dd798467%28v=vs.85%29.aspx.

[15] ŘÍHA, Ondřej. Wave Idea Bitstram 3x. *Programovatelný kontroler* [online]. 2008, Music Store, č. 08 [cit. 2015-05-18]. Dostupné z: www.music-store.cz/recenze/wave-idea-bitstream-3x.

[16] The Complete MIDI 1.0 Detailed Specification. *Document vision 4.2. The MIDI Manufacturers Association*, Los Angeles, CA, 1995.

[17] 17 WAVE IDEA. *Bitstream 3x: User Manual (V 1.2)*. Francie, 2004. Dostupné z: www.waveidea.com/en/downloads/bitstream_3x/bs3x_manual_e_v12.pdf.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

.NET	
$	Číslo vyjádřené v hexadecimální soustavě
BS	Bitstream
CC	Control Change
CF	ControlFREAK
DIN	Deutsches Institut für Normung
ID	Identifikační číslo
LCD	Liquid Crystal Display
LSB	Least Significant Bit
MIDI	Musical Instrument Digital Interface
MSB	Most Significant Bit
MSDN	Microsoft Developer Network
SysEx	System Exclusive
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial BUS

SEZNAM PŘÍLOH

A NÁVRH APLIKACE

A.1 Tabulky formátu SysEx zprávy pro CF

Tab. A.1: Adresy příkazů pro ControlFREAK.

Příkaz	Adresa		Příkaz	Adresa
Control change	$B0		Set midi channel 4	$F1
Begin SysEx	$B1		Set midi channel 5	$F2
Note On	$B2		Set midi channel 6	$F3
Note Off	$B3		Set midi channel 7	$F4
Program change	$B4		Set midi channel 8	$F5
NRPN MSB+only	$B5		Set midi channel 9	$F6
NRPN MSB+LSB	$B6		Set midi channel 10	$F7
RPN MSB only	$B7		Set midi channel 11	$F8
RPN MSB+LSB	$B8		Set midi channel 12	$F9
Pitch bend	$B9		Set midi channel 13	$FA
Mono aftertouch	$BA		Set midi channel 14	$FB
Poly Aftertouch	$BB		Set midi channel 15	$FC
Controller HI/lO	$BC		Set midi channel 16	$FD
MTC Quarter Frame	$BD		Invert data	$FE
Song position pointer	$BE		Set data from slider 16	$9E
Song select	$BF		Set data from slider 15	$9F
Set global MIDI chanel	$C9		Set data from slider 14	$A0
Set Clk Rate BPM	$CA		Set data from slider 13	$A1
Set Real Time Val1	$CB		Set data from slider 12	$A2
Set Real Time Val1	$CC		Set data from slider 11	$A3
Int clock start	$D0		Set data from slider 10	$A4
Int clock stop	$D1		Set data from slider 9	$A5
Int clock cont	$D2		Set data from slider 8	$A6
Timing clock	$D3		Set data from slider 7	$A7
Clock start	$D4		Set data from slider 6	$A8
Clock cont.	$D5		Set data from slider 5	$A9
Clock stop	$EC		Set data from slider 4	$AA
Set midi channel 1	$EE		Set data from slider 3	$AB
Set midi channel 2	$EF		Set data from slider 2	$AC
Set midi channel 3	$F0		Set data from slider 1	$AD

A.2 Tabulky formátu SysEx zprávy pro Bs3x

Tab. A.2: SysEx ID 0[17].

Bit č.	Popis	Poznámka
7	Vždy 0	-
6	Směr SysEx zprávy	0: do BS3X 1: od BS3X
5	Cíl SysEx zprávy	0: naprogramování 1: žádost o poskytnutí informací
4	Požadavek na potvrzení	0: žádný 1: ano
3	Požadavek na kontrolní součet	0: žádný kontrolní součet 1: s CS
2	Rezervováno	Vždy 0
1-0	Význam SysEx zprávy	0-0 globální parametry 0-1 změna názvu prvku 1-0 aktualizace firmwaru 1-1 parametry prvku

Tab. A.3: SysEx DATA pro bitsream 3x [17].

Bajt č.	Popis	Poznámka
0	Ctrl status 0	tab. A.4
1	Ctrl status 1	tab. A.5
2	Ctrl status 2	tab. A.6
3	Délka MIDI zprávy	Rozsah od 0 do $14
4	Pozice MIDI kanálu	Rozsah od 0 do $14
5	První ovlivněný bajt prvkem	Rozsah od 0 do $14
6	Druhý ovlivněný bajt prvkem	Rozsah od 0 do $14
7	Minimální hodnota	Rozsah od 0 do $7F
8	Maximální hodnota	Rozsah od 0 do $7F
9	Začátek kontrolního součtu	Rozsah od 0 do $14 0: kontrolní součet nevložen
10	Vazba s jiným prvkem	tab. A.7
11-31	Řetězec MIDI zpráv	maximálně 21 MIDI zpráv

Tab. A.4: Ctrl status 0 [17].

Bit č.	Popis	Poznámka
7	Výběr CC/Nota v stand. režimu	0: CC, 1: Nota zapnuta
6-0	CC nebo číslo noty v stand. režimu	Rozsah od 0 do 127

Tab. A.5: Ctrl status 1 [17].

Bit č.	Popis	Poznámka
7	Chování tlačítka	0: push, 1: toggle
6	Vyjmutí scény	0: žádné 1: Vyjmut
5	Automatické odeslání po startu	0: žádné, 1: povolen
4	USB výstup	0: zakázán, 1: aktivován
3	MIDI výstup 1	0: zakázán, 1: aktivován
2	MIDI výstup 2	0: zakázán, 1: aktivován
1-0	Zpoždení MIDI zpráv	00: žádné, 01: 30ms, 10: 60ms, 11: 90ms

Tab. A.6: Ctrl status 2 [17].

Bit č.	Popis	Poznámka
7-4	Rezervováno	Vždy 0
3	Rozdělit řídící hodnotu	0: odeslat jako 7 bitové číslo 1: rozdělit do dvou niblů
2-0	Řídící charakteristika potenciometru	000: lineární 001: invertně lineární 010: logaritmická 011: invertně logaritmická 100: náhodná 101: uživatelská křivka 1 110: uživatelská křivka 2 111: rezervováno

Tab. A.7: Vazba s jiným prvkem [17].

Bit č.	Popis	Poznámka
7	Virtuální cross-fader	0: žádný, 1: aktivován
6	Číslo prvku	1-67

B OBSAH PŘILOŽENÉHO CD

Přiložené CD obsahuje elektronickou verzi bakalářské práce, spustitelnou aplikaci a jeho zdrojový kód. Aplikace je pojmenovaná „MIDI_01.exe“ a nachází se v adresáři „\aplikace\Release\“. Zdrojové soubory se nachází v adresáři „\Zdrojový kód\MID_01\“.

Aplikace byla odzkoušena na operačním systému Microsoft Windows 7 v prostředí Microsoft .NET Framework v. 4.0.