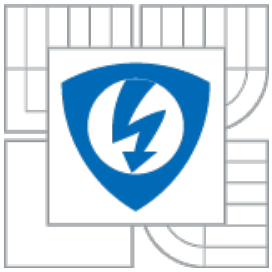
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

PŘEVOD ZVUKU DO MIDI FORMÁTU

SOUND TO MIDI CONVERSION

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE — PAVEL PLEVA
AUTHOR

VEDOUCÍ PRÁCE — ING. ZBYNĚK FEDRA, PH.D.
SUPERVISOR

BRNO 2010

VYSOKÉ UCENÍ
TECHNICKÉ V BRNE
Fakulta elektrotechniky
a komunikačních technologií
Ústav radioelektroniky

Bakalářská práce

bakalářský studijní obor

Elektronika a sdělovací technika

Student: **Pavel Pleva** ***ID:*** **72936**

Ročník: 3 ***Akademický rok:*** 2009/2010

NÁZEV TÉMATU:

Převod zvuku do MIDI formátu

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte vlastnosti MIDI formátu a možnosti převodu zvuku (jednoduchého hudebního signálu) do MIDI formátu. Otestujte (např. v prostředí Matlab) potřebné algoritmy a jejich náročnost. Na základe předchozích výsledku a po dohodě s vedoucím projektu se pokuste implementovat testované algoritmy na zvolenou výpočetní platformu. Zhodnoťte možnosti takové implementace, její náročnost a věrohodnost výsledného MIDI záznamu.

DOPORUCENÁ LITERATURA:

[1] GUÉRIN, R. Velká kniha MIDI-standardy, hardware, software. Computer Press, 2005.

[2] ZAPLATÍLEK, K., DONAR, B. MATLAB - začínáme se signály. Praha: BEN - technická literatura, 2006.

Termín zadání: 8.2.2010 ***Termín odevzdání:*** 28.5.2010

Vedoucí práce: Ing. Zbyněk Fedra, Ph.D.

Konzultanti bakalářské práce:

prof. Dr. Ing. Zbyněk Raida

Předseda oborové rady

ANOTACE

Práce se zabývá možnostmi převodu jednoduchého hudebního signálu do MIDI formátu. Obsahuje stručné uvedení do vybraných částí standardu MIDI, které jsou pro tuto problematiku důležité. Dále jsou probrány charakteristické vlastnosti zvuku, jež je potřeba z hudebního signálu pro zápis do SMF zjistit nebo které jejich zjišťování jistým způsobem ovlivňují. Na konci práce je popsán algoritmus pro převod implementovaný v prostředí Matlab.

KLÍČOVÁ SLOVA

MIDI, detekce základního tónu, hudební signál, tón

ABSTRACT

This thesis deal with possibilities of conversion simple musical signal to MIDI format. It contains brief introduction into chosen parts of MIDI standard which are important for this theme. Further there are examined characteristic attributes of sound which are needed to detect from musical signal to be written in SMF or which affects detection of useful attributes. There is given an account of algorithm for conversion implemented in Matlab.

KEY WORDS

MIDI, basic pitch extraction, musical signal, key

PLEVA, P. *Převod zvuku do MIDI formátu.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 52 s. Vedoucí semestrální práce Ing. Zbyněk Fedra, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Převod zvuku do MIDI formátu“ jsem vypracoval 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 uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské 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 § 152 trestního zákona č. 140/1961 Sb.

V Brně dne . . . .

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu bakalářské práce Ing. Zbyňku Fedrovi, Ph.D za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne …………… ………………………..

(podpis autora)

Obsah

Seznam ilustrací

Seznam tabulek

ÚVOD

MIDI (Musical Instrument Digital Interface) je komunikační rozhraní pro elektronické hudební nástroje. Již od začátků své existence se tento protokol těší velké oblibě jak u laické, tak u odborné hudební veřejnosti, a to nejen díky cenové dostupnosti zařízení, ale hlavně pro svou kompatibilitu se staršími hudebními nástroji a zařízeními. Stejně tak je hojně využíván v hudebním průmyslu. Přesto, že již mírně zastarává a pomalu přestává postačovat velkým nárokům moderní studiové techniky je stále jediným komunikačním protokolem svého druhu. MIDI skýtá velkou škálu využití a široké možnosti práce s hudebními daty a zvukem vůbec.

Po MIDI sběrnici mohou komunikovat různé elektronické nástroje a zařízení. Primárně jsou však využívány klávesové hudební nástroje, jelikož se pro tento účel principiálně hodí. Existují i tzv. dechové kontroléry, které umožňují tvorbu MIDI signálu podobně, jako při hře na dechový nástroj.

Cílem této práce je prozkoumat možnosti převodu jednoduchého hudebního signálu tvořeného hrou na klasický hudební nástroj snímaného mikrofonem. Budou zde probrány základní vlastnosti zvuku potažmo tónu důležité pro tuto problematiku. Algoritmus pro převod bude implementován v prostředí Matlab.

1 MIDI

MIDI je akronym pro slovní spojení Musical Instrument Digital interface (digitální rozhraní pro hudební nástroje). Jedná se o standard, který definuje způsob komunikace mezi hudebními nástroji a počítačem, popřípadě dalšími MIDI zařízeními. Toto označení ovšem nezahrnuje pouze komunikační protokol, nýbrž je současně i názvem normy, jazyka a soupisem specifikací, spadá sem i celá rodina zařízení. [1]

1.1 Historie

Za jeho počátek se dá považovat rok 1981, kdy se na výstavě NAMM (National Association of Music Merchants) v Anaheimu v USA setkali prezidenti společností Sequential Circuits, Oberheim a Roland. Byly zde projednány předběžné návrhy na univerzální komunikační rozhraní pro elektronické hudební nástroje. Posléze se k těmto návrhům připojily firmy Yamaha, Korg a Kawai. V říjnu roku 1981 představili Dave Smith a Chet Wood na konferenci Audio Engineering Society v New Yorku první ucelený návrh hudebního rozhraní USI (Universal Synthesizer Interface).

O rok později (1982) se na téže výstavě setkali představitelé hudebních firem a obohatili koncept hudebního rozhraní o další úpravy a vylepšení. K projektu se zde připojilo celkem 15 amerických a japonských firem. Finální název MIDI navrhla japonská firma Roland. Již v lednu roku 1983 byly představeny první MIDI nástroje, a to SCI Prophet 600 (následovník legendárního Prophet 5) a Roland Jupiter JP-6. Za pozornost také stojí zařízení Roland MPU-401, které jako první umožnilo propojení hudebních nástrojů s počítačem skrze MIDI a i přes svou relativní zastaralost je dodnes využíváno.

Dne 5. 8. 1983 byla uveřejněna konečná verze MIDI normy 1.0. Obsahovala však značné množství nejasností a nebyla dostatečně podrobná. Proto byly založeny normativní organizace MMA (MIDI Manufacturers Association) a JMSC (Japan MIDI Standard Committee), které měly dohlížet na dodržování normy a její vývoj.

V září roku 1985 vyšla podrobná MIDI norma, avšak již necelý rok nato do ní byly zaneseny dodatky a úpravy. Tento standard je stále aktualizován a vyvíjen. I dnes se těší velké oblibě napříč uživatelským spektrem. [1][2]

1.2 Princip

Po sběrnici hudebního komunikačního rozhraní neproudí klasický hudební signál, nýbrž jsou zde pouze sekvenčně přenášena řídící data, která odpovídají akci hudebníka, to je zahrání noty, sešlápnutí pedálu, změně rejstříku atd. Jsou-li tato data zaznamenána a posléze přehrána, syntezátor věrně reprodukuje nahranou skladbu. Záznam je možno také editovat (např. na počítači), výstup tedy může znít zcela odlišně od originálu. [1][2]

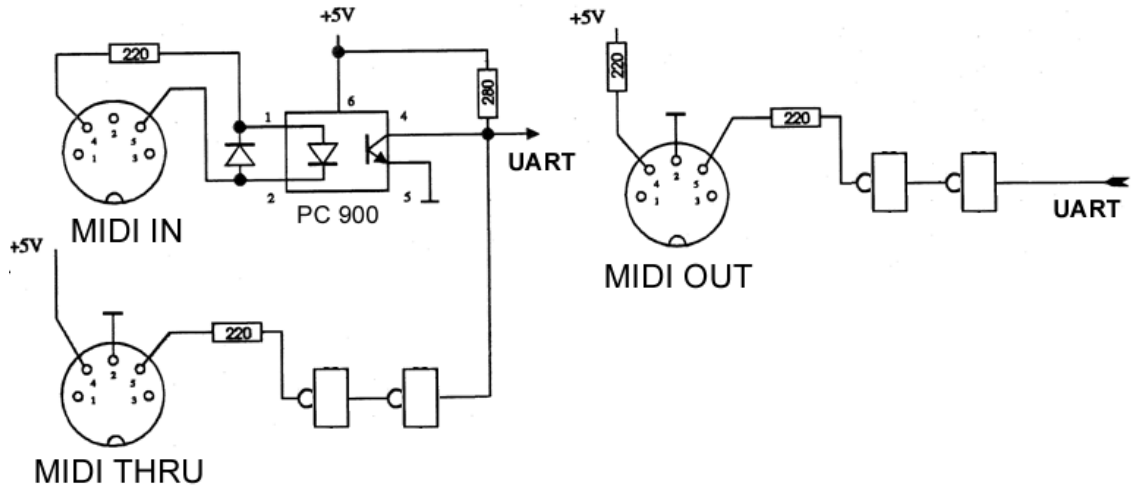
1.3 Technická specifikace

Rozhraní MIDI se skládá ze tří 5-pinových konektorů DIN značených In, Out a Thru. Konektor In je vstupní, konektor Out je výstupní a konektor Thru slouží k přeposílání informací ze vstupu a nemusí být vždy použit. Některá zařízení používají funkci Soft Thru, což znamená, že slučují vstupní data s výstupními a posílají je na konektor Out. [2]

obr. 1: Zadní panel elektronického syntezátoru s trojicí konektorů MIDI [3]

1.3.1 Hardware rozhraní

Sběrnice MIDI je realizována 5 mA proudovou smyčkou, logické nule odpovídá protékající proud. Je zde použit sériový asynchronní datový přenos s rychlostí 31,25 kBaudů. Formát rámce je: jeden startbit, osm datových bitů a dva stopbity. Nepoužívá se parita.



obr. 2: Schéma rozhraní MIDI [3]

Nedoporučuje se propojovat země přístrojů a konektorů (z důvodu předejití zemním smyčkám), MIDI vstup nástroje je vhodné od vstupu do přístroje galvanicky oddělit optoizolátorem s časem reakce menším než 2 ms (např. Sharp PC-900).

K propojení se zvukovou kartou počítače je potřeba MIDI kabel, který již

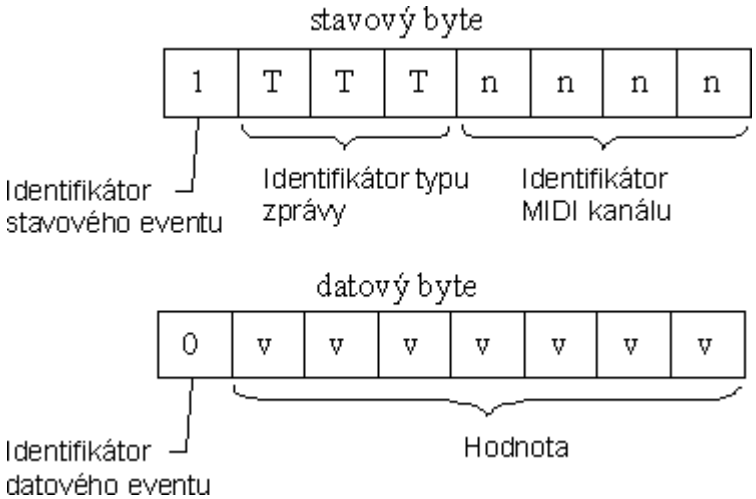
obsahuje zařízení z obr. 2, jelikož běžné počítače nemají z prostorových důvodů osazen MIDI port, nýbrž zde bývají vyvedeny pouze UART signály do konektorů gameportu. U levnějších MIDI kabelů hrozí absence izolačního optočlenu, což může způsobovat problémy.

Maximální počet zařízení zapojených v řetězci Thru je čtyři, protože obsažený optoizolátor zavádí s každým připojeným přístrojem zpoždění. Toto zpoždění by bylo při větším počtu zařízení neúnosné. [3]

1.3.2 Struktura MIDI protokolu

Základní datový blok přenášející určitou informaci se nazývá „MIDI zpráva“ (MIDI message). MIDI zpráva sestává vždy z jednoho stavového a několika datových bytů. Tyto byty jsou označovány „MIDI události“. Je to osmibitový datový typ, podle nejvýznamnějšího bitu se určuje, jde-li o stavový byte (MSB=1) nebo datový byte (MSB=0).

Existují dva druhy MIDI zpráv, tzv. kanálová data a systémová data. U kanálových dat je ve stavovém eventu přenášena informace o virtuálním datovém kanále. Protože ve stavovém eventu je pro rozlišení MIDI kanálu vyhrazen méně významný nibl, mohou být po jedné fyzické MIDI sběrnici přenášena kanálová data až v 16 virtuálních kanálech. Systémová data, na rozdíl od stavových, nenesou informaci o MIDI kanále - jsou společná pro všechny kanály. Méně významný nibl stavového eventu identifikuje typ systémových dat. Ta se dělí na zvláštní systémová data(SysEx = System Exclusive message) umožňující přenos větších datových bloků a data reálného času sloužící k vzájemné časové synchronizaci několika zařízení.



obr. 3: Struktura stavového a datového bytu protokolu MIDI [3]

Různé typy MIDI zpráv se liší počtem datových bytů. Např. zpráva „Nota zapnuta“ obsahuje číslo noty (tón) a sílu stlačení klávesy. Oba tyto datové byty mohou

nabývat 128 hodnot (0 – 127), neboť MSB je vyhrazen pro identifikátor. [3]

Running Status

Tento mód se používá kvůli zvýšení průchodnosti sběrnice. Pokud se u kanálových dat nemění typ zprávy, není nutné vysílat stále kompletní MIDI zprávy, ale stačí posílat pouze datové byty, kompletní zpráva potom sestává z odpovídajícího počtu datových bytů. Tím se ušetří cca třetina kapacity sběrnice. Tento stav se nazývá Running Status (průběžný stav) a je ukončen přijetím odlišného stavového bytu.

Interpretace a priorita dat

MIDI nástroje jsou nastaveny tak, že ignorují všechny nerozpoznané stavové byty a s nimi související datové byty. Ignorovány jsou také datové byty bez příslušného stavového bytu, pokud však není sběrnice v průběžném stavu.

Data v MIDI systému mají následující prioritu:

1. resetování systému
2. data reálného času
3. zvláštní systémová data
4. společná systémová data
5. kanálová data

Pokud MIDI přijímač detekuje uvnitř přijímaného datového bloku stavový byte dat reálného času (který pozná podle nastaveného MSB), pozastaví přijímání datových bytů a zpracuje MIDI zprávu reálného času. Poté se vrátí k přerušenému přijímání dat. Data reálného času neruší průběžný stav sběrnice. [3]

1.4 Kanálová data

Jsou vždy vztažena ke konkrétnímu MIDI kanálu, jehož číslo je méně významným niblem stavového bytu. Samotný identifikátor kanálových dat zabírá tři bity, což znamená, že může nabývat osmi hodnot. Osmý identifikátor je vyhrazen pro systémová data, proto je celkem sedm typů kanálových dat.

MIDI zpráva	Význam	ID	počet databytů
Note On	nota zapnuta	0	2
Note Off	nota vypnuta	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

tab. 1: Kanálová MIDI data [2]

Rozlišují se čtyři základní režimy, ve kterých mohou MIDI přijímače pracovat, určují způsob, jímž budou kanálová MIDI data zpracována. Řídí se nastavením OMNI ON/OFF a POLY/MONO.

režim 1: Omni On, Poly

režim 2: Omni On, Mono

režim 3: Omni Off, Poly

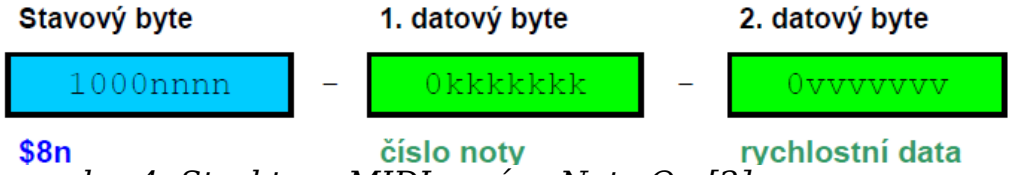
režim 4: Omni Off, Mono

Pomocí režimů MONO a POLY se nastavuje způsob přiřazení jednotlivých hlasů přijímače při současném přijetí více not. Je-li aktivní režim MONO, hraje každý hlas přijímače noty přijímané na odpovídajícím kanále monofonně. Naproti tomu v režimu POLY jsou hrány přijímané noty polyfonně.

V režimu OMNI zapnuto je nástroj nastaven na přijímání a provádění dat ze všech kanálů. V režimu OMNI vypnuto reaguje nástroj pouze na data na zvoleném kanále. Pokud zařízení neumožňuje volbu kanálu, mělo by být vždy nastaveno na kanál 1. V režimu 4 jsou přehrávány noty přijaté na MIDI kanálech N až N+M-1, kde N je základní kanál nastavený na přijímači a M je počet MIDI kanálů specifikovaný MIDI zprávou přepnutí režimu MONO. Hodnota M=0 je speciální případ, který říká přijímači, aby použil všechny dostupné hlasy pro hraní not přijatých na kanálech N až 16. [2]

1.4.1 MIDI zprávy Nota zapnuta a Nota vypnuta

MIDI zprávě Note On (nota zapnuta) odpovídá identifikátor 1. Nese informaci o stisku klávesy, tedy o zahraném tónu. První datový byte obsahuje informaci o MIDI čísle noty a druhý datový byte nese rychlostní data, tedy informaci o dynamice zahrané noty.



obr. 4: Struktura MIDI zprávy Note On [2]

Čísla MIDI not mohou nabývat standardně 128 hodnot od 0 do 127. Noty jsou v MIDI číslovány od C2 do G8 (viz tab. 2), přičemž rozsah klaviatury klavíru je od A0 do C7, tedy převedeno do MIDI čísel od 21 do 108.

Oktáva	Hud. označení	značení	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	30	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				

tab. 2: MIDI čísla not [2]

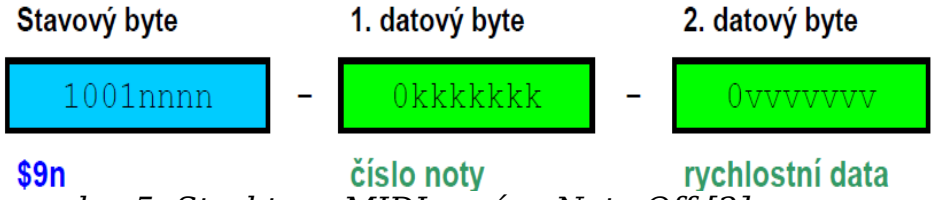
Ve druhém datovém bytu zprávy Note on je jistým způsobem obsažena dynamika zahrané noty. Jelikož není jednoduché vytvořit elektronický klávesový nástroj, který by snímal dynamiku úhozu, používá se v MIDI přímé úměry mezi rychlostí a silou úhozu. Díky tomu stačí, aby nástroj snímal dobu mezi sepnutím dvou spínačů postupně aktivovaných při stlačení klávesy. Tato informace je posléze zaznamenána jako tzv. rychlostní data (Velocity). Vztah mezi rychlostními daty a dynamikou je vyjádřen v tab. 3.

Velocity	Dynamika	Hudební označení
0	off	vypnuto
1	*ppp*	*pianissimo piano*
	pp	*pianissimo*
	p	*piano*
64	*mp*	*mezzo-piano*
	mf	*mezzo-forte*
	f	*forte*
	ff	*fortissimo*
127	*fff*	*fortissimo forte*

tab. 3: MIDI rychlostní data [2]

MIDI zařízení bez rychlostního snímače by měla vysílat Velocity s hodnotou 64. Zvláštním případem je Velocity s hodnotou 0, znamená „hraj notu s nulovou dynamikou“ tedy v podstatě „nota vypnuta“ (Note Off). Je tedy možné místo MIDI zprávy „nota vypnuta“ posílat zprávu „nota zapnuta“ s nulovou hodnotou Velocity. MIDI zařízení po přijetí takovéto zprávy přestane danou notu hrát. Tohoto se využívá pro vypínání a zapínání tónů v průběžném stavu.

MIDI zprávě Note Off (nota vypnuta) odpovídá identifikátor 0 a nese informace o uvolnění klávesy. V prvním datovém bytu je zaznamenáno MIDI číslo noty, ve druhém informace o tom, s jakou rychlostí byla klávesa uvolněna. Rychlostní informace o uvolnění klávesy umožňují napodobení některých technik hry např. na strunné nástroje, používá se však zřídkakdy. [3]



obr. 5: Struktura MIDI zprávy Note Off [2]

1.5 Standardní MIDI soubory

Tento formát souborů, přidaný k MIDI normě roku 1988, slouží k přenosu MIDI dat. Označuje se jako „standardní MIDI soubor“ (Standard MIDI File, zkráceně SMF). Používá strukturu „Resource Interchange File Format“ (RIFF). To znamená, že základním prvkem je informační blok zvaný „chunk“. Každý takovýto blok musí začínat čtyřznakovou identifikací, za níž následuje 32-bitová délka bloku. Jednotlivé bloky do sebe lze vnořovat, nepoužívají se však více než tři úrovně. MIDI soubory obsahují dva typy bloků, a to hlavičku (Header chunk) a stopy (Track chunk).

1.5.1 Hlavička standardního MIDI souboru

Je prvním blokem a nese literální označení „MThd“. Následuje 4-bytová informace o délce, která je vždy 6. Další údaje jsou po dvou bytech:

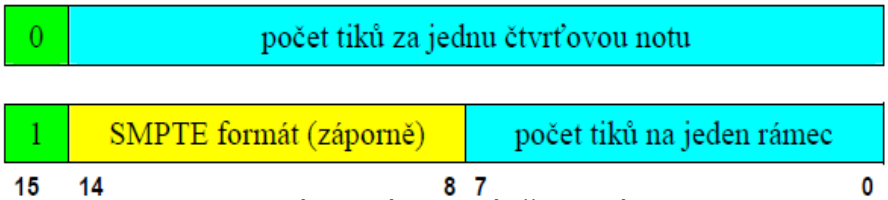
format - formát souboru (0, 1 nebo 2)

ntrks - počet stop (bloků stop) v souboru

division - základní dělení (relativní časová jednotka)

Formát 0 SMF může obsahovat pouze jednu vícekanálovou stopu, formát 1 může obsahovat jednu a více stop se společným časem a formát 2 jeden a více časově nezávislých vzorů s jednou stopou.

SMF používá jako základní časovou jednotku přírůstek času, tzv. „delta time“. Tato jednotka může být udávána ve dvou formátech rozlišených nejvyšším bitem. Časová jednotka je udávána buď v počtu tiků, tj. rytmických pulsů za čtvrťovou notu, nebo v SMPTE formátu času a počtu tiků na jeden SMPTE rámec. [1] [3]



obr. 6: Formáty základní časové jednotky SMF souboru [2]

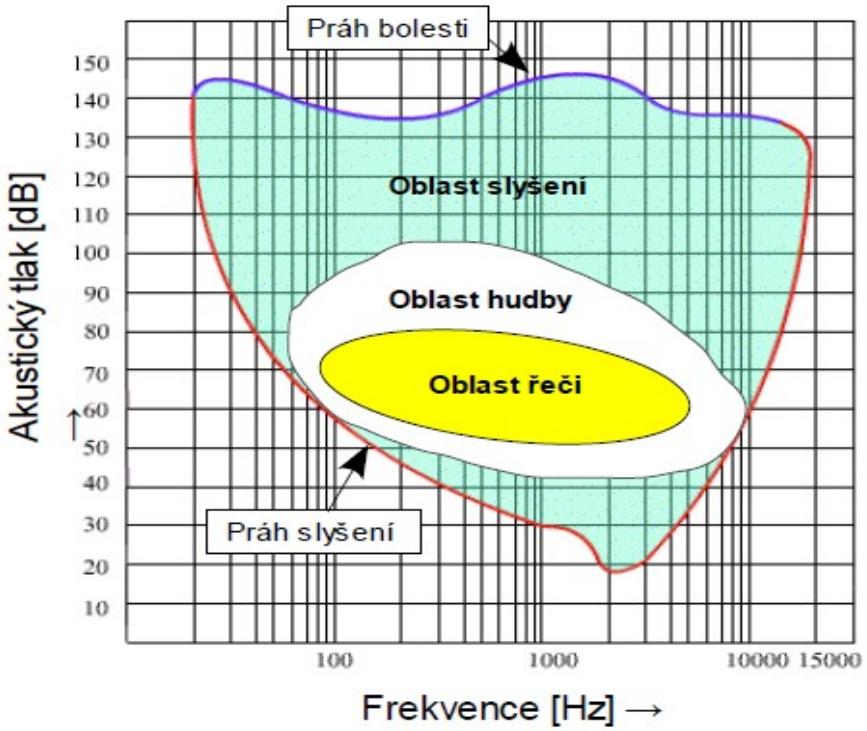
1.5.2 Stopy standardního MIDI souboru

Datový blok stopy nese literální označení „MTrk“, které je shodné pro všechny formáty SMF. Opět následuje 4-bytová informace o délce. Tento datový blok obsahuje sled po sobě jdoucích MIDI událostí (MTrk event), které se povinně skládají z relativního času zprávy (delta-time) a samotné MIDI zprávy (event). Relativní čas udává počet časových jednotek definovaných základním dělením uloženým v hlavičce souboru, které uplynuly od poslední události k dané MIDI zprávě.

Dále je možné přenášet tzv. Meta-zprávy definující ostatní hudební události, které nejsou definovatelné pomocí MIDI zpráv. Jsou to např. číslo sekvence, tempo, rytmus, předznamenání, různé názvy a texty atd. Tyto zprávy mají shodný formát s MIDI zprávami, pouze jejich stavový byte obsahuje hodnotu $FF. V MIDI protokolu by to znamenalo System Reset, ale v SMF nemá tento příkaz uplatnění. Rozlišuje se přibližně patnáct Meta zpráv. Pokud je některé zařízení nerozpozná, jsou ignorovány. [1] [3]

2 HUDEBNÍ SIGNÁL

Hudební a řečový signál se rozprostírá v oblasti slyšitelného zvukového spektra, jak je vyznačeno na obr. 7, přičemž řečový signál je z frekvenčního i dynamického hlediska podskupinou hudebního. Základní frekvence tónů v hudbě pokrývají takřka celý rozsah lidského sluchu. Klasická hudba se pohybuje od cca 16 Hz do 8,4 kHz, zatímco moderní hudba, obzvláště potom elektronická, zahrnuje i tóny s vyšší základní frekvencí. Také s prvkem dynamiky se v hudbě hojně pracuje, vyskytují se zde tóny takřka na prahu slyšitelnosti i blížící se 110 dB.



obr. 7: Charakteristické oblasti vnímání akustického signálu lidským sluchem [4]

2.1 Tón

Tímto termínem se označuje zvuk se stálou frekvencí. Je to v podstatě základní stavební prvek hudby. Zapisuje se formou not do notové osnovy. V hudební terminologii se takto také označuje interval mezi dvěma celými tóny, jako např. C a D.

2.1.1 Výška tónu

Udává se buďto absolutně kmitočtem daného tónu nebo, což je v praxi obvyklejší, v relativní míře, tedy pomocí intervalů. Ty udávají poměr kmitočtů. Základním intervalem je tzv. oktáva, která odpovídá poměru 2 : 1, jinak řečeno tón o oktávu vyšší než základní má dvojnásobnou frekvenci. Celé spektrum hudebních signálů je takto možné rozdělit do deseti oktáv, jejichž výčet je uveden v tab. 4.

Název oktávy	označení	poznámka
subkontra	*C2*	basový klíč
kontra	*C1*	
velká	*C*	
malá	*c*	
jednočárkovaná	c^1	houslový klíč
dvoučárkovaná	c^2	
tříčárkovaná	c^3	
čtyřčárkovaná	c^4	
pětičárkovaná	c^5	
šestičárkovaná	c^6	

tab. 4: Názvy a značení hudebních oktáv [2]

Oktávu je dále možno rozdělit do řady tónů, zvané stupnice. Jednotlivé tóny stupnice vzájemně svírají intervaly, z nichž základní jsou:

- prima (interval 1 : 1)
- velká sekunda (interval 9 : 8)
- malá tercie (interval 6 : 5)
- velká tercie (interval 5 : 4)
- kvarta (interval 4 : 3)
- kvinta (interval 3 : 2)
- velká sexta (interval 5 : 3)
- velká septima (interval 15 : 8)
- oktáva (interval 2 : 1)

Uspořádání vzájemných výškových poměrů jednotlivých tónů stupnice hudebního nástroje se nazývá ladění, přičemž vzájemná poloha tónů v řadě není určena především rozdílem kmitočtů, nýbrž jejich podílem. Aby se usnadnily výpočty při porovnávání intervalů odlišných ladění, používá se logaritmická stupnice. Vznikne rovnoměrným rozdělením intervalu temperovaného ladění (viz dále) na sto dílů, potom se jeden díl nazývá cent a je definován vztahem:

$$1\text{cent} = \sqrt[100]{q} = \sqrt[100]{\sqrt[12]{2}} = \sqrt[1200]{2} \quad , \tag{1}$$

kde q je velikost intervalu, pro rovnoměrně temperované ladění je $q = \sqrt[12]{2}$.[4]

2.1.2 Síla tónu

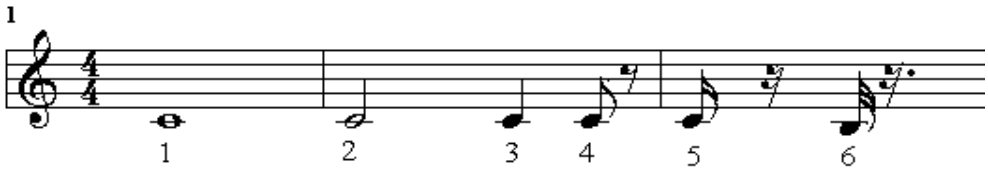
Závisí na intenzitě zvuku, což je akustický výkon procházející jednotkovou plochou v čase. Intenzita závisí na energii zdroje, která byla vydána při tvorbě tónu. Všechny tyto děje jsou matematicky popsatelné. V hudební terminologii se však používá subjektivní dynamická stupnice, která není přímo matematicky závislá na intenzitě zvuku. Nejslabší i nejsilnější tón je dán možnostmi daného hudebního nástroje. Jednotlivé názvy z této stupnice pocházejí z italštiny a jsou uvedeny v tab. 5.

Název	Označení	Význam
pianissimo piano	*ppp*	co nejslaběji
pianissimo	*pp*	velmi slabě
piano	*p*	slabě
mezzo-piano	*mp*	středně slabě
mezzo-forte	*mf*	středně silně
forte	*f*	silně
fortissimo	*ff*	velmi silně
fortissimo forte	*fff*	co nejsilněji

tab. 5: Názvy a značení hudební dynamiky [2]

2.1.3 Délka tónu

Je to doba, po kterou daný tón zní. V hudebním zápisu se délka tónu vyjadřuje různými typy not, které se od sebe liší tvarem (viz obr. 8)



obr. 8: Grafická reprezentace not [4]

1 – celá nota (4 doby)

2 – půlová nota (2 doby)

3 – čtvrťová nota (1 doba)

4 – osminová nota (½ doby)

5 – šestnáctinová nota (¼ doby)

6 – dvaatřicetinová nota ($^1/_8$ doby)

Délku doby určuje tempo skladby, které se udává v BPM – počet úhozů za minutu (Beats Per Minute). Udává se buďto číselnou hodnotou nebo přibližně slovním výrazem.

2.1.4 Barva tónu

Je to vlastnost, která lidskému sluchu umožňuje rozlišovat různé zdroje zvuku, i když vydávají tón o stejné výšce, tedy například flétnu od klavíru a podobně. Barva tónu je subjektivní pojem a není přímo navázána na konkrétní veličinu. Je dána průběhem spektrální funkce. Z výše řečeného vyplývá, že dva tóny o totožné výšce hrané na různé nástroje nemají stejné spektrum, shodná je pouze základní harmonická složka.

Většina hudebních nástrojů vydává zvuk s periodickým průběhem a proměnnou amplitudou, k čemuž přistupují případné přechodné děje na začátku a na konci tónu, tzv. nakmitávací a dokmitávací pochody. Tento periodický děj lze pomocí Fourierovy transformace rozložit na součet harmonických složek s různou amplitudou a fází. Jelikož lidské ucho není citlivé na fázový posun spektrálních složek, nemá tento na barvu zvuku vliv. Výška tónu odpovídá poloze první (základní) harmonické složky. U některých nástrojů může mít základní harmonická složka menší amplitudu než vyšší harmonické či může být dokonce úplně potlačena (např. u žesťů).

Barva tónu tedy závisí na počtu vyšších harmonických složek (alikvotních tónů), na jejich amplitudách či spíše na poměru jejich amplitud a pozici ve spektru. Významným lokálním extrémům ve spektru, které mají zásadní vliv na barvu tónu, se říká formanty, mohou být:

- harmonické - výskyt ve spektru je v celých násobcích frekvence základní harmonické složky
- neharmonické - výskyt ve spektru není v celých násobcích frekvence základní harmonické složky
- pevné - jejich poloha není závislá na výšce tónu
- pohyblivé - jejich poloha je závislá na výšce tónu

Obecně lze říci, že čím více jsou ve spektru zastoupeny vyšší harmonické složky, tím je tón ostřejší a naopak. [2]

2.2 Soustavy ladění

Ladění určuje přesné vzdálenosti mezi jednotlivými tóny hudební stupnice. Udává frekvence tónů a jejich poměry. Pomineme-li historická a exotická, dělí se ladění v zásadě na dvě skupiny, a to čistá a temperovaná.

2.2.1 Čistá ladění

Využívají pouze tóny, jejichž vzájemné poměry frekvencí se dají vyjádřit celými čísly. Např. Pythagorejské ladění používá pro odvození všech tónů pouze oktávu (2 : 1) a kvintu (3 : 2).

Výhodou těchto soustav je přirozená velikost intervalů, daná poměrem celých čísel. Jsou-li tato čísla dostatečně malá, znějí souzvuky „čistě“.

Pro současnou evropskou hudbu však převažují nevýhody, které takřka znemožňují využití těchto ladění. Např. při důsledném uplatnění čistých ladění by byl počet tónů v oktávě nekonečný a při využití pouze omezeného počtu tónů se zase objevují „nečisté“, nelibozvučné intervaly. Dalším problémem je nemožnost modulace do jiné tóniny, neboť by v ní některé intervaly zněly rozladěně. To je způsobeno rozličnými frekvencemi stejných tónů v různých soustavách (např. nota A v C dur má jinou frekvenci než nota A v D dur). Navíc zde není možná tzv. enharmonická záměna většiny tónů (tedy např. C# není stejná nota jako Db), což působí potíže při konstrukci a ladění nástrojů s pevnými výškami tónů. [5]

2.2.2 Temperovaná ladění

Kvůli výše zmíněným nevýhodám čistých ladění byla zavedena ladění temperovaná, která tyto nevýhody odstraňují či alespoň zmírňují. Některé intervaly jsou zde záměrně rozladěny z důvodu přesnějšího sladění jiných. Tyto soustavy umožňují modulace mezi stupnicemi bez výskytu disharmonických intervalů a také enharmonickou záměnu, což umožňuje snížit počet tónů v oktávě, např. v evropské hudbě na dvanáct.

Nerovnoměrně temperovaná ladění upravují frekvence jistých tónů, tak aby tóniny blízké základní zněly co možná nejlépe, což je však na úkor vzdálenějším, které už znějí pouze přijatelně. Jisté „preferované“ intervaly jsou temperované tak, aby vycházely „čistě“, naproti tomu některé zní poněkud disonantně.

V současné evropské hudbě je nejpoužívanější ladění rovnoměrně temperované. Všechny intervaly stejného druhu (kvinty, kvarty, tercie atd.) jsou zde stejně velké (stejně „rozladěné“), kromě oktáv není žádný interval úplně „čistý“. Oproti nerovnoměrně temperovanému ladění má výhodu, že je možná modulace do libovolně vzdálené tóniny, aniž by to mělo vliv na zvukovou kvalitu intervalů. [5]

3 PŘEVOD HUDEBNÍHO SIGNÁLU DO MIDI PROTOKOLU

Jelikož je SMF v podstatě souhrn informací o hudbě, potřebujeme tyto informace z hudebního signálu vyzískat. Na první pohled je stěžejní správná detekce kmitočtu jednotlivých tónů zkoumané melodie, neboť právě kmitočet je pro tón určujícím parametrem. Jelikož nebudeme zvukový soubor zpracovávat celý naráz, je třeba jej nejprve korektně nasegmentovat. Dalšími parametry potřebnými k převodu do MIDI protokolu jsou intenzita a délka trvání tónu. Nyní již zbývá jen zápis do SMF. Tato témata budou probrána podrobněji probrána v následujících kapitolách. Pro ilustraci průběhu programu a jednotlivých kroků jsou v příloze viz. *příloha B* bloková schémata.

3.1 Segmentace

Nejprve musíme stanovit délku segmentu, z něhož se bude získávat základní frekvence tónu. Uživatel zadává informace o dané skladbě, a to tempo a nejkratší notu, jejíž délce posléze odpovídají segmenty. Přesně se délka úseku stanoví podle následujícího vzorce:

$$n_{samp} = t_{note} * f_{samp} = \frac{\frac{1}{note} * 4 * 60}{tempo} * f_{samp} \qquad (2)$$

kde n_{samp} je počet vzorků na segment, t_{note} délka nejkratší noty ve vteřinách, f_{samp} vzorkovací kmitočet v Hz, *note* udává, která je nejkratší nota (8 pro osminovou, 16 pro šestnáctinovou a nejčastěji používané 32 pro dvaatřicetinovou notu) a *tempo* je tempo v BPM pro danou skladbu.

Jelikož se však zvolený segment a reálná nota nemusejí vždy přesně překrývat, ať už kvůli vzájemnému časovému posuvu či nedodržením rytmu ze strany hudebníka, jsou vybrané hodnoty vynásobeny Hannovým oknem, aby tak případný vliv sousedních segmentů byl co možná nejmenší.

Při tomto postupu je minimální rozlišitelný kmitočet daný tempem a zvolenou nejkratší notou. V případě standardních hodnot sto dvacet BPM, jako nejkratší nota dvaatřicetina vzorkovací kmitočet 44100 Hz vychází nejmenší detekovatelná frekvence ze vztahu:

$$\Delta f = \frac{f_{samp}}{n_{samp}} = \frac{44100}{2756} = 16\,Hz \quad , \qquad (3)$$

kde Δf je nejmenší detekovatelný kmitočet, f_{samp} vzorkovací kmitočet a n_{samp} počet vzorků na segment. Horní hranice je potom

dána Shannon-Kotělnikovým teorémem, tedy polovina vzorkovacího kmitočtu. Vzhledem k frekvenčnímu rozsahu hudebních nástrojů je tento rozsah dostačující.

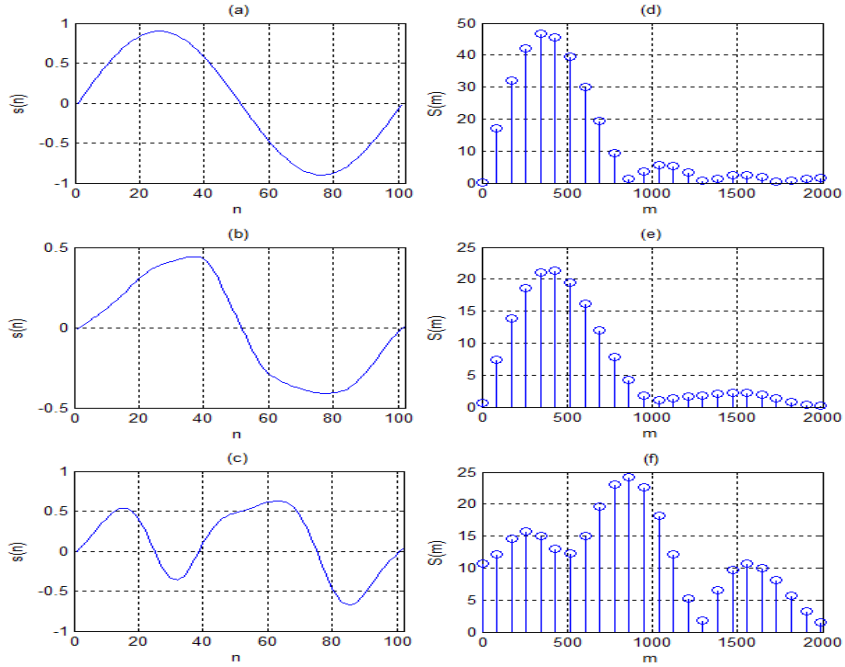
3.2 Detekce základního kmitočtu tónu

V této práci jsou testovány dvě metody pro odhad základního kmitočtu, a to detekce v časové oblasti a detekce ve spektrální oblasti.

Před započetím hledání základního kmitočtu je třeba nejprve věnovat pozornost průběhu a chování zkoumaného – zdrojového signálu.

3.2.1 Rozbor vlastností zkoumaného signálu

Zdrojem signálu pro odhad základního kmitočtu je dechový nástroj – konkrétně příčná flétna. V ideálním případě se průběh tónu blíží funkci sinus o daném kmitočtu viz *obr. 9 a),b*. Je patrné, že oba průběhy jsou si velmi podobné a to jak v časové, tak ve spektrální oblasti. Oproti tomu signál z *obr.9 c)* se již na první pohled velmi liší, přestože se jedná o tentýž tón hraný s o něco málo větší intenzitou.



Obr. 9: Porovnání jedné periody sinusoidy 440 Hz a reálného hudebního signálu 440 Hz

a) časový průběh sinusoidy d) a její spektrum

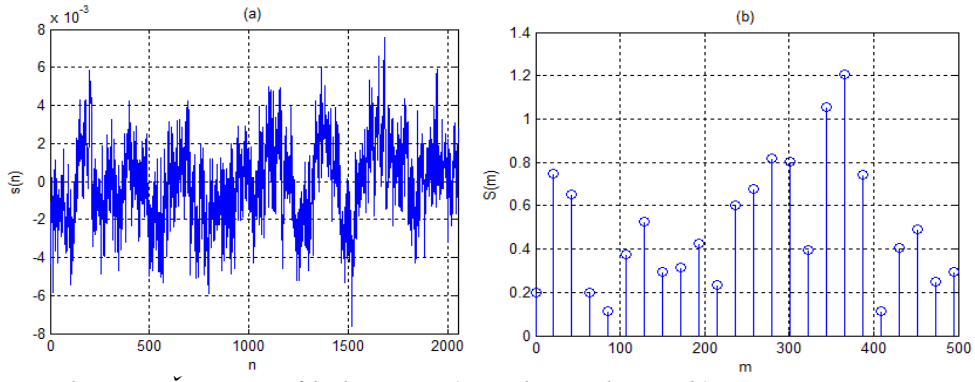
b) časový průběh hraného tónu e) a jeho spektrum

c) časový průběh téhož hraného tónu – jiná realizace

f) a jeho spektrum

Ve spektru na *obr.9 f)* je možno pozorovat přesun maxima na dvojnásobek základního kmitočtu, druhá harmonická složka má tedy v tomto signálu vyšší energii než první. Tento rozdíl je dán vlivem formantů – zbarvením tónu. Z toho plyne, že jak časový průběh, tak analogicky spektrální složení tónu je u dechového nástroje silně závislé na „způsobu“ hry, byť je zdrojem zvuku stále stejný nástroj. Při běžné hře na flétnu -když nejsou používány flažolety, přerývané tóny atp. - však převažuje spíše průběh podobný *obr.9 b)*, případně s mírnými zákmity v maximu a minimu.

Pro další zpracování signálu je také velmi důležité vědět, „co se děje“ v pomlkách, tedy když vstupní zařízení snímá „ticho“. Mezery mezi hranými tóny pochopitelně obsahují šum, který je dán jednak hlukem na pozadí při nahrávání a jednak samotným šumem mikrofonu.



Obr. 10: Časový průběh šumu a) a jeho spektrum b)

Na *obr.10 a)* je zachycen časový průběh šumu. Z jeho spektra (*obr.10 b)*) je patrné, že byť má šum velmi malou amplitudu, vyskytují se v něm náhodně různé frekvenční složky a je z jeho průběhu možno detekovat „základní tón“. V tomto případě by dokonce spadala detekovaná frekvence do rozsahu nástroje a byl by tak v pomlce chybně detekován tón. Z toho plyne nutnost zavedení hladiny „ticha“, pod jejíž úrovní je signál považován za šum. Toto je dále využíváno v obou popisovaných metodách.

3.2.2 Detekce v časové oblasti

Tato metoda je v podstatě založena na výpočtu autokorelační funkce AKF, která určuje míru podobnosti v rámci jednoho signálu, z jejího průběhu je tedy možné určit, zda se signál v rámci zkoumaného úseku opakuje, případně s jakou periodou.

Autokorelační funkce je definována[6]:

$$R(m) = \sum_{n=0}^{N-1-m} s(n)s(n+m) \quad , \tag{4}$$

kde *R(m)* je vektor autokorelačních koeficientů, *N* je délka signálu a *s(n)* původní posloupnost.

S využitím symetrie autokorelačních koeficientů dostaneme[6]:

$$R(m) = \sum_{n=m}^{N-1} s(n)s(n-m) \tag{5}$$

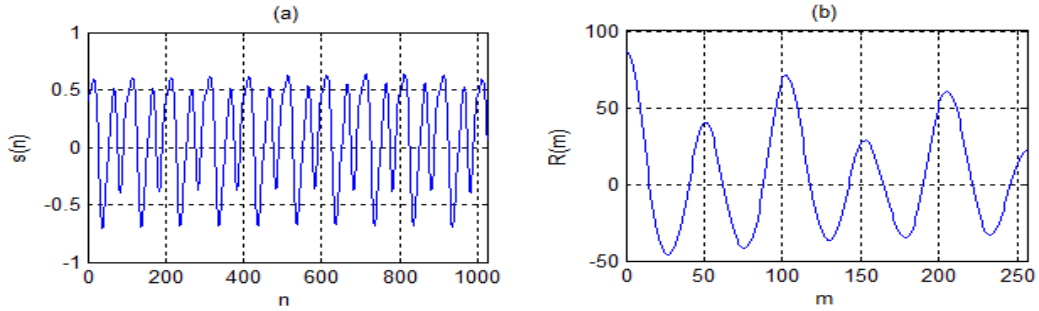
Protože je však tato metoda poměrně časově náročná nahrazuje se výpočet takzvanou rychlou korelací, která využívá rychlé Fourierovy transformace FFT pro převod segmentu do frekvenční oblasti, kde je následně modul hodnot umocněn nadruhou a pomocí inverzní rychlé Fourierovy transformace IFFT převeden zpět do časové oblasti[7].

Na *obr. 11* je zobrazen průběh komorního A *a)* a jeho jednostranná autokorelační

funkce *b)*. Z AKF jsou jasně patrné opakující se vrcholy s indexy odpovídajícími $\frac{f_{vz}}{F_0}, \frac{2f_{vz}}{F_0}, \frac{3f_{vz}}{F_0}, \ldots$, kde f_{vz} je vzorkovací kmitočet a F_0 právě hledaný základní kmitočet tónu. Pro jeho určení je tedy zapotřebí zjistit index druhého, respektive následujících píků. První špička odpovídá energii signálu. Hodnota základního tónu pro daný segment je tedy dána vzorcem[7]:

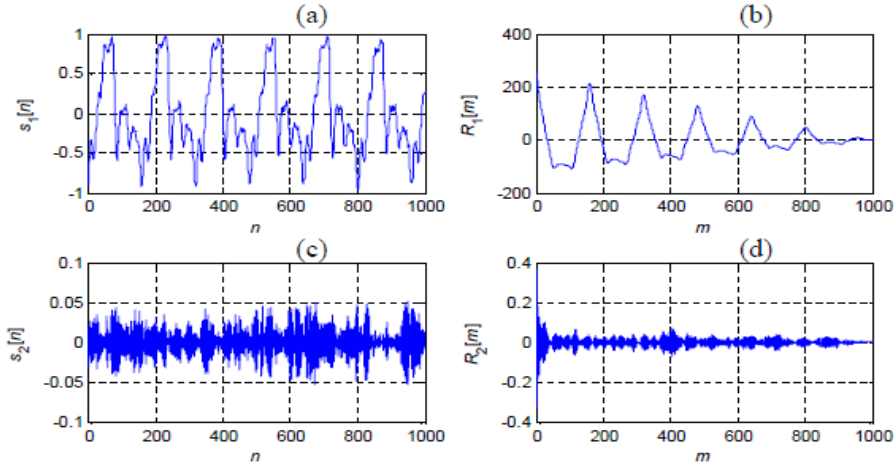
$$F_0 = \frac{f_{vz}}{k} \ [Hz] \ , \tag{6}$$

kde kromě výše zmíněných je *k* index špičky.



Obr. 11: Časový průběh komorního A a jeho jednostranná AKF

Před samotným výpočtem frekvence tónu je třeba určit, zda je rámec znělý či neznělý. Toto má význam hlavně pokud je zkoumaným signálem řeč, kde se vyskytují znělé hlásky, jako např. „a“, ale také neznělé, jako např. „s“ viz *obr.12*. Pokud by se rámce nerozlišovaly a byly bez výjimky považovány za relevantní pro odhad základního tónu, docházelo by ke značné chybovosti.



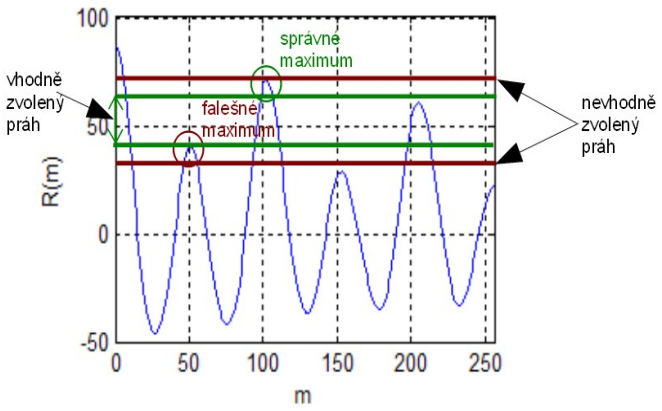
Obr. 12: Časový průběh úseku hlásky „a“ $s_1[n]$ (a) a jeho jednostranná AKF $R_1[m]$ (b). Časový průběh úseku hlásky „s“ $s_2[n]$ (c) a jeho jednostranná AKF $R_2[m]$ (d). [7]

Mohlo by se zdát, že při práci s hudebním signálem není tento krok zapotřebí, jelikož všechny úseky musí být nezbytně znělé, taková domněnka je nicméně mylná. I když by signál neobsahoval pomlky – obsahují pouze šum snímacího zařízení – mohlo by docházet k chybné detekci vlivem již výše zmiňované přítomnosti formantů. Jak ukazuje *obr.13*, znělost rámce se dá určit porovnáním velikosti prvního maxima s hodnotou nultého autokorelačního koeficientu udávajícího energii úseku, který je vždy největší[6]:

$$\begin{aligned} R_{max} &< \alpha\, R(0) \rightarrow neznělý \\ R_{max} &\geq \alpha\, R(0) \rightarrow znělý \end{aligned} \quad , \tag{7}$$

kde *Rmax* představuje hodnotu prvního detekovaného maxima, α je prahovací konstanta, která se musí zvolit experimentálně a *R(0)* je nultý – maximální – autokorelační koeficient.

Optimální volba prahovací konstanty má zásadní vliv na úspěšnost detekce základního tónu a musí se provádět pro signály různého charakteru zvlášť. Pokud je zvolená hodnota pro danou aplikaci příliš nízká, mohou být detekována falešná maxima a odhadnutá frekvence neodpovídá skutečnosti. Na druhou stranu je-li zvolená konstanta pro danou aplikaci příliš vysoká, mohou být znělé rámce považovány za neznělé.



Obr. 13: Lokalizace prvního maxima

Pro rámec z *obr.13* by bylo nutno zvolit práh nejméně $\alpha = 0.65$, což je ovšem poměrně vysoká hodnota, které by nemusely vyhovět ani některé znělé rámce. Proto se používají další metody, jimiž je signál předzpracován a teprve poté se provádí autokorelace. Jednou z těchto metod je centrální klipování.

3.2.3 Centrální klipování

Aplikuje se na segmentovaný signál a jeho cílem je zvýraznění základní frekvence. Používá se pro odstranění některých nedostatků autokorelační metody a to hlavně chyb způsobených výskytem postranních maxim v autokorelační funkci. Je dobré připomenout, že za tato postranní maxima jsou zodpovědné formanty (rezonanční

frekvence), neboť zapříčiňují zákmity v rámci jedné periody. Jelikož autokorelační metody nedostatečně potlačují tato postranní maxima, může díky nim dojít k chybné detekci [6], [8].

Algoritmus centrálního klipování je založen na Sondiho a Rabierově postupu a je vyjádřen funkcí[8]:

$$c(x(k)) = \begin{matrix} \therefore \\ \cdots \\ \ddots \end{matrix} \begin{matrix} +1 \; pro \; x(k) > h \\ 0 \; pro \; x(k) \leq |h| \\ -1 \; pro \; x(k) < -h \end{matrix} \quad , \tag{8}$$

kde c je vektor klipovaných hodnot vektoru x a h je experimentálně volený práh.

Variantou tohoto postupu je[6]:

$$c(x(k)) = \begin{matrix} \therefore \\ \cdots \\ \ddots \end{matrix} \begin{matrix} x(k) \; pro \; x(k) > h \\ 0 \; pro \; x(k) \leq |h| \\ x(k) \; pro \; x(k) < -h \end{matrix} \tag{9}$$

Protože amplituda signálu neustále kolísá, nemůže se nastavit klipovací úroveň přes celý signál pevně, nýbrž se musí volit pro každý rámec, ze kterého je odhadován základní tón. Nejsnazší je tento přístup[6]:

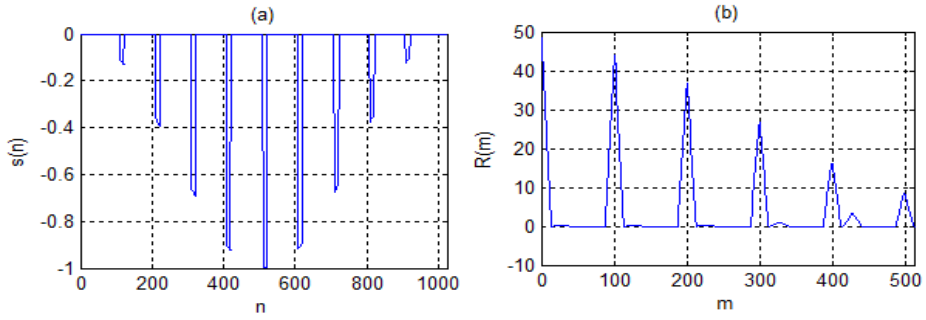
$$c_L = k \max_{n=0\ldots N-1} |x(n)| \quad , \tag{10}$$

Kde c_L je klipovací úroveň neboli práh a redukční konstanta k se volí od 0,6 do 0,8. Lepším i když o něco složitějším přístupem je rozdělení rámce na tzv. mikrorámce o třetinové délce původního. Z těchto se potom vybírá „nejslabší maximum, které louží pro výpočet prahu[6]:

$$c_L = k \min\{\max|x_1(n)|, \max|x_2(n)|, \max|x_3(n)|\} \quad , \tag{11}$$

Problém může nastat v pomlkách, kde je přítomen pouze šum, především pokud je klipování prováděno podle rovnice 8. Proto je dobré zvolit tzv. úroveň ticha, je-li maximu v rámci menší než zvolená úroveň, znělost ani tón se v rámci neurčují[6].

Obr. 14 a) zobrazuje klipovaný průběh signálu z *obr. 11 a)*. Pozorovateli se může zdát signál značně „zdeformovaný“, jelikož v rámci nezůstala po klipování žádná kladná hodnota. Z *obr. 14 b)* je nicméně vidět, že jeho jednostranná autokorelační funkce již neobsahuje nežádoucí „falešná“ maxima, což znamená, že byl odstraněn nebo alespoň silně potlačen vliv formantů.



Obr. 14: Klipovaný signál a), a jeho jednostranná AKF b).

3.2.4 Detekce ve spektrální oblasti

Tato metoda je založena na převodu jednotlivých segmentů do frekvenční, tj. spektrální oblasti za pomoci diskrétní Fourierovy transformace DFT, která je definována[9]:

$$S(k)=R_N(k)\sum_{n=0}^{N-1} s(n)\mathrm{e}^{-jk\frac{2\Pi}{N}n} \quad ,k=0,1,\ldots,N-1, \tag{12}$$

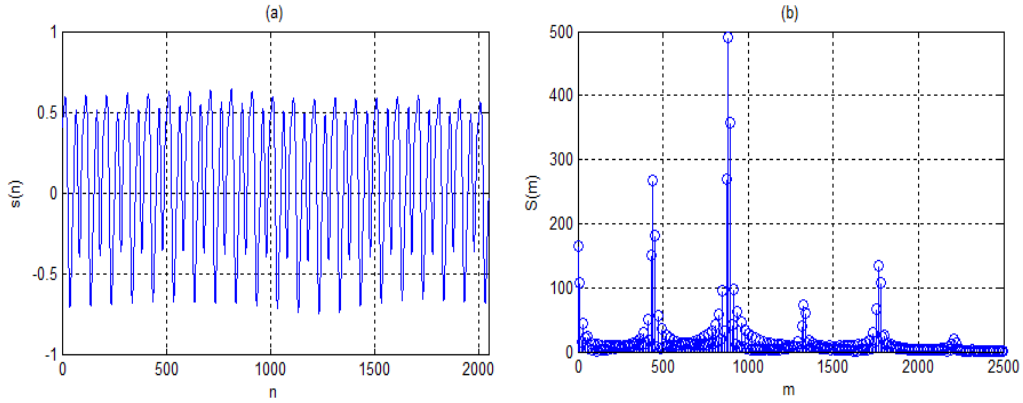
kde $S(k)$ je hodnota k. spektrální složky, $R_N(k)$ je tzv. okénková posloupnost (v literatuře se často neuvádí, neboť je ve vztahu z početního hlediska nadbytečná), $s(n)$ je hodnota vzorku vstupního signálu a N délka signálu ve vzorcích.

K samotnému převodu se opět využívá rychlá Fourierova transformace, jelikož je oproti DFT podstatně méně náročná na čas i výpočetní výkon.

Jak již bylo zmíněno v podkapitole o segmentaci, jednotlivé úseky odpovídají délkou nejkratší zvolené notě, což by při vyšším tempu mohlo znamenat příliš malé frekvenční rozlišení. Proto je každý segment doplněn nulami na délku 4096 vzorků. Tak je nejnižší detekovatelná frekvence a zároveň frekvenční rozlišení 10,76 Hz. To je vzhledem k rozsahu flétny (nejnižší tón H3 – 246,94 Hz) a tedy nejmenšímu rozdílu dvou tónů (H3 a C4 – 14,69 Hz) dostatečné.

Po převedení do spektrální oblasti je možno přistoupit k detekci základního tónu. Nejprve je v segmentu vyhledána maximální hodnota. Podle jejího indexu je na frekvenční ose vyhledán patřičný kmitočet, který je následně prohlášen za základní.

Problém může nastat, obsahuje-li signál kromě základního také alikvotní tóny, tzn. vyšší harmonické složky. Pokud je energie jedné z nich vyšší než energie základní harmonické složky, prosté vyhledání maxima chybně označí její kmitočet za hledaný. Tento případ ilustruje *obr.15*. Proto je potřeba zavést opatření, které vychází z pravidla, že se vyšší harmonické složky vyskytují právě na celých násobcích základního kmitočtu. Stačí tedy ověřovat, zda se na polovičním, případně třetinovém, kmitočtu nenachází významná špička. Hranice, kdy je špička významná se musí určit experimentálně jako procentuální hodnota absolutního maxima v segmentu.



Obr. 15: Časový průběh komorního A a) a jeho spektrum b)

3.3 Stanovení intenzity tónu

Při zjišťování intenzity snímaného tónu vycházíme z poněkud zjednodušené představy, že síla, s níž je nota zahrána je konstantní po celou dobu jejího trvání. Díky tomuto zjednodušení je možné stanovit intenzitu pro daný segment detekcí maximální amplitudy a jejím přiřazením pro celý úsek. Tato hodnota je zjišťována v časové oblasti. Nejmenší síla tónu, tedy ztlumeno je přiřazena nulové hodnotě signálu, respektive zvolené hodnotě „ticha“, která odpovídá hladině šumu. Maximální síla je přiřazena největší amplitudě, kterou je schopno snímací zařízení vyhodnotit.

3.4 Stanovení délky trvání tónu

Délka trvání tónu je určována na základě známé délky segmentů, na něž je vstupní signál rozdělen. Každé okno vzorků je identifikováno detekovaným základním kmitočtem. Ten je vyhledán v převodní tabulce a je mu přiřazeno odpovídající MIDI číslo noty. V této fázi se zaznamenává, kolik po sobě jdoucích segmentů náleží ke stejné notě. Pokud nota spadá do povoleného rozsahu – rozsah nástroje – a je-li změna frekvence dostatečně velká na to, aby byla vyhodnocena jako změna tónu, zapíše se délka jako počet shodných oken krát délka trvání jednoho okna. Nejkratší tón odpovídá délce jednoho segmentu – nejkratší zvolená nota (většinou dvaatřicetina), nejdelší tón není třeba omezovat.

3.5 Zápis do SMF

Nejprve je vytvořen tzv. „MThd“ neboli header chunk, který nese informace o MIDI souboru, jak bylo zmíněno v kapitole MIDI. K němu je připojena hlavička tzv. „MTrk“ neboli track chunku. Jako první event je zapsána informace o tempu zadaná uživatelem.

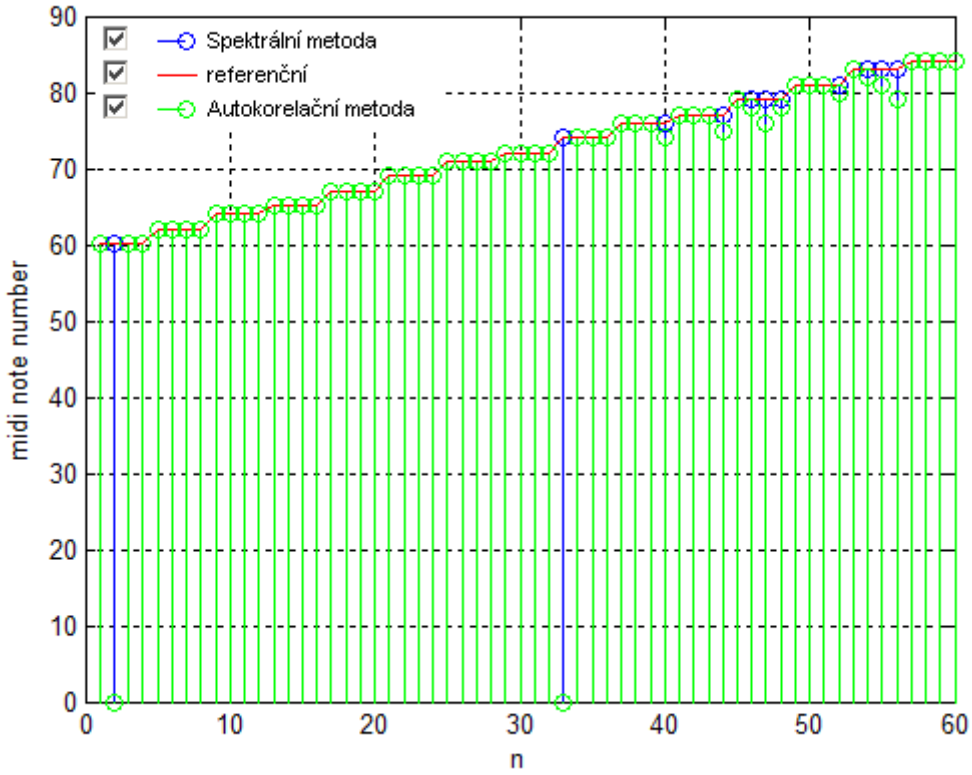
Následně je cyklicky načítáno okno vzorků z vstupního signálu. Detekuje se

základní frekvence tónu. Ta je vyhledána v převodní tabulce a přiřazena k záznamu s nejmenší kvadratickou odchylkou. Index záznamu je zároveň MIDI číslo noty. Ve stejném kroku je zjišťována intenzita signálu pro daný úsek.

Dokud po sobě jdoucí segmenty náleží stejnému MIDI číslu noty, je inkrementován čítač. Jakmile dojde ke změně, je zapsána zpráva nota vypnuta s číslem a délkou trvání – delta časem – předchozí noty, hned za ni se zapíše zpráva nota zapnuta s číslem, délkou trvání a intenzitou aktuální noty. Jelikož je použit mód running status, jsou zprávy nota vypnuta nahrazeny zprávami nota zapnuta s nulovou rychlostí. Na konec souboru je zapsána povinná událost konec stopy.

4 POROVNÁNÍ ÚSPĚŠNOSTI METOD

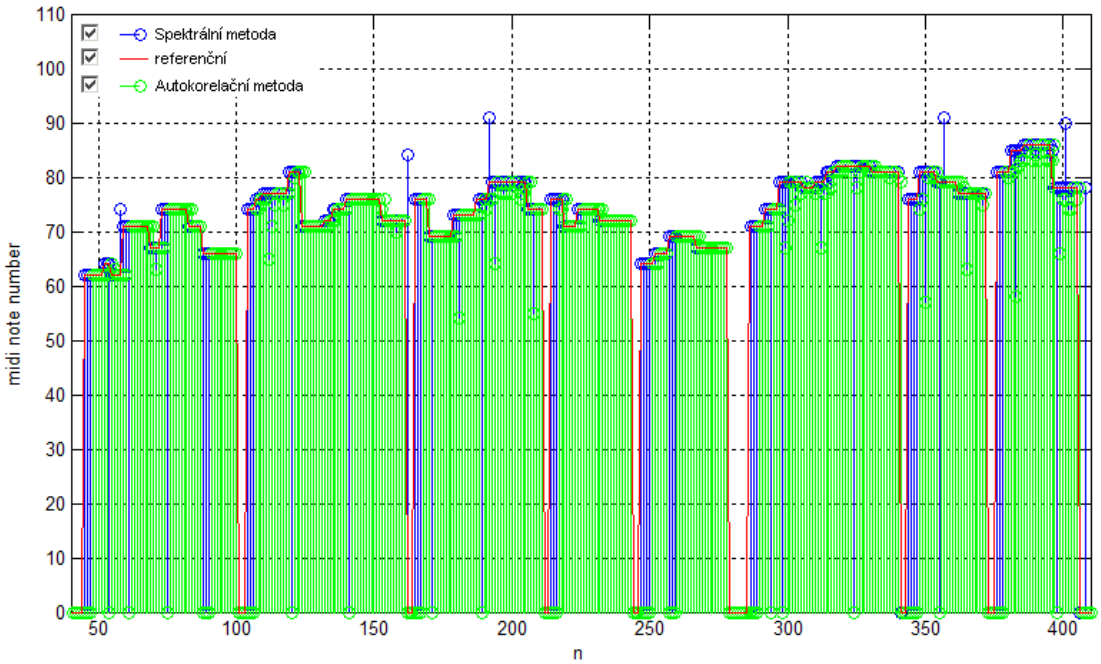
Úspěšnost jednotlivých metod je porovnávána na základě počtu korektně převedených segmentů. Celková úspěšnost je potom dána počtem správně určených úseků vůči všem. Na *obr.16* jsou vidět výsledky jednotlivých metod.



Obr. 16: Porovnání úspěšnosti metod detekce základního tónu (signál 1)

Pro testovací signál – stupnice C-dur 2 oktávy – mají obě metody vysokou úspěšnost, a to konkrétně spektrální metoda 100% a autokorelační 81,67%. Při převodu reálného hudebního signálu je však úspěšnost obou metod nižší, jelikož může docházet k rozladění nástroje, nedodržení tempa či rytmu. Úspěšnost autokorelační metody je také silně závislá na vhodné volbě vstupních parametrů.

Na *obr.17* je zobrazeno porovnání výsledků obou metod pro druhý testovací signál, a to úryvek z „Romanze“ Maxe Regera. Jak je vidět, spektrální metoda má opět lepší úspěšnost, v tomto případě 97,95%, nejlepší výsledek dosažený autokorelační metodou je 65,46%.



Obr. 17: Porovnání úspěšnosti metod detekce základního tónu (signál 2)

Tabulka *tab.6* ukazuje úspěšnost autokorelační metody pro různé kombinace vstupních parametrů (testováno na druhém signálu, viz. výše), přičemž hodnota 0 odpovídá jak u klipování, tak u prahování stavu vypnuto.

Klipovací úroveň	Prahovací konstanta	Úspěšnost
-	-	%
1,0	0,0	33,41
0,8	0,0	47,01
0,6	0,0	58,64
0,4	0,0	60,91
0,2	0,0	65,46
0,0	0,0	23,41
1,0	0,2	40,46
0,8	0,2	53,18
0,6	0,2	56,82
0,4	0,2	60,91
0,2	0,2	62,73
0,0	0,2	23,41
1,0	0,5	39,77
0,8	0,5	53,41
0,6	0,5	54,55
0,4	0,5	57,05
0,2	0,5	62,01
0,0	0,5	23,41
1,0	0,7	37,96
0,8	0,7	52,96
0,6	0,7	57,01
0,4	0,7	60,46
0,2	0,7	62,27
0,0	0,7	23,41

Tab. 6: Výsledky autokorelační metody pro různé vstupní parametry

Pokud není použita metoda centrálního klipování, dosahuje se poměrně dosti nízké úspěšnosti. Paradoxně nejlepšího výsledku je dosaženo bez použití prahování autokorelační funkce. To je zřejmě způsobeno nízkým podílem formantů na struktuře signálu.

Metoda	Úroveň „ticha“	Úspěšnost
	-	%
Autokorelačn	0,000	46,14
Autokorelačn	0,015	65,46
Autokorelačn	0,100	24,32
Spektrální	0,000	81,59
Spektrální	0,015	97,59
Spektrální	0,100	88,64

Tab. 7: Vliv úrovně "ticha" na úspěšnost metod detekce základního tónu

Tabulka *tab.7* obsahuje výsledky metod pro různé hodnoty parametru úroveň „ticha“, přičemž 0 odpovídá stavu bez použití této konstanty. Je patrné, že správné zvolení této hodnoty je důležité. Pokud je zvolena příliš nízká, ovlivňuje šum detekci, pokud naopak příliš vysoká, je „ořezáván“ i užitečný signál. Pro použitý mikrofon se zdá být optimální hodnota 0,015.

ZÁVĚR

V této práci jsou stručně popsány části standardu MIDI vztahující se k problematice převodu jednoduchého (jednohlasého) hudebního signálu do formátu MIDI. Dále jsou zde uvedeny charakteristické vlastnosti zvuku, respektive tónu, které je třeba pro zápis do SMF z hudebního signálu vyzískat nebo jež mají na detekci těchto informací vliv.

Je zřejmé, že stěžejním a zároveň, metodicky i výpočetně, nejnáročnějším úkolem je správné určení základního kmitočtu tónu. V této práci je jako vstupní signál použit zvuk příčné flétny, v jehož spektru má povětšinou první harmonická složka největší amplitudu, nicméně může docházet k ovlivnění formanty. Jsou zde zkoumány možnosti dvou detekčních metod – autokorelační, tj. v časové oblasti a spektrální, tj. ve spektrální oblasti.

Algoritmus implementovaný v prostředí Matlab snímá vstupní signál po jednotlivých segmentech. Je vždy načten jeden úsek, z něhož se zjistí potřebné informace, které jsou následně zapsány do SMF, teprve poté je načten další úsek signálu. Algoritmus je vytvořen tak, aby po určitých úpravách mohl fungovat v reálném čase, z čehož plyne případné použití MIDI zařízení pro dechový nástroj.

Spektrální metoda dosahuje dobrých výsledků, a to díky možnosti přizpůsobení konkrétnímu nástroji, v obecném případě má naopak horší úspěšnost, než ostatní metody. Při zkoušení na prvním testovacím signálu (*signál 1*) obstála na 100%. Reálně však dosahuje menší úspěšnosti, kvůli nedodržení tempa, rytmu či naladění nástroje, což se také projevilo (i když ne výrazně) při zkoušení na druhém testovacím signálu (*signál 2*).

Autokorelační metoda měla pro první testovací signál (*signál 1*) úspěšnost 81.67%. Její výsledky však silně závisí na volbě vstupních parametrů, a to především klipovací a prahovací úrovně, což je patrné z tabulky *tab.6*.

Obě metody jsou potom značně závislé na správném nastavení úrovně „ticha“, jak je vidět z tabulky *tab.7*.

Seznam použité literatury

[1] GUÉRIN, R. *Velká kniha MIDI : Standardy, Hardware, Software*. Brno : Computer Press, 2004. 340 s. ISBN 80-7226- 985-2.

[2] KÁŇA, L, SCHIMMEL, J. *Studiová a hudební elektronika.* Elektronická skripta, VUT v Brně [online]. 2004 [cit. 2009-12-20]. Dostupný z WWW: <http://rapidshare.com/files/21009574/studiova_a_hudebni_elektronika.zip>

[3] SCHIMMEL, J. *Komunikační rozhraní MIDI*. *Elektrorevue : Časopis pro elektrotechniku* [online]. 2002, [cit. 2009-12-20]., č. 69 Dostupný z WWW: <http://www.elektrorevue.cz/clanky/02069/index.htm#kap1>.

[4] KRUPIČKA, J. *Převod not jednohlasé melodie ze zvukového signálu do protokolu MIDI.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 52 s. Vedoucí diplomové práce Ing. Jaromír Mačák.

[5] Wikipedie*, otevřená encyklopedie: Ladění* [online]. [2009] , 16. 8. 2009 [cit. 2009-12-22]. Čeština. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Ladění>.

[6] ČERNOCKÝ, J. *Zpracování řečových signálů – studijní opora.* Elektronické skriptum. Brno: Speech@FIT, Ústav počítačové grafiky a multimédií, Fakulta informačních technologií, Vysoké učení technické v Brně, 6. prosince 2006 http://www.fit.vutbr.cz/~cernocky

[7] ATASSI, H. Metody detekce základního tónu řeči. Elektrorevue [online]. 21.01.2008, 2008, 4, [cit. 2010-05-15]. Dostupný z WWW: <http://www.elektrorevue.cz/cz/clanky/zpracovanisignalu/0/metodydetekce-zakladniho-tonu-reci/>. ISSN 1213-1539.

[8] SOVA, J. Hlasové pole. Praha, 2008. 83 s. Bakalářská práce. České vysoké učení technické v Praze.

[9] ŠEBESTA, V., SMÉKAL, Z. *Signály a soustavy*. Elektronické skriptum Brno: VUT v Brně, 2003. s. 1-165. ISBN: REL117.

SEZNAM ZKRATEK, VELIČIN A SYMBOLŮ

AKF	Autokorelační funkce
BPM	Beats per minute (úderů za minutu)
FFT	Fast Fourier Transform (rychlá Fourierova transformace)
JMSC	Japanese Midi Standard Committee
MIDI	Musical Instrument Digital Interface (digitální rozhraní pro hudební nástroje)
MMA	MIDI Manufacturers Association
MSB	Most Significant Bit (bit s nejvyšší hodnotou)
NAMM	National Association of Music Merchants
RIFF	Resource Interchange File Format
SMF	Standard Midi File (standardní MIDI soubor)
SMPTE	Society of Motion Picture and Television Engineers
USI	Universal Synthesize Interface - předchůdce MIDI
UART	Universal Asynchronous Receiver/Transmitter

Δf	frekvenční rozlišení
f_{samp}	vzorkovací frekvence
kBaud	jednotka modulační rychlosti
ms	milisekunda
t_{note}	délka noty
note	nota
n_{samp}	*počet vzorků*
$S(k)$	obraz diskrétní Fourierovy transformace

Seznam příloh

A OBSAH CD

Adresáře:

Text	-	obsahuje elektronickou verzi práce ve formátu .pdf a .odt
Matlab	-	obsahuje skripty pro Matlab, ve kterých jsou implementovány obě metody detekce a dále pomocné funkce.
Ukázky	-	obsahuje zvukové soubory ve formátu .wav a .mid

Soubory:

Matlab

cntrclip.m	-	funkce využívaná autokorelační metodou pro centrální klipování vstupního signálu
segment.m	-	funkce využívaná pro segmentaci vstupního signálu
s2mauto.m	-	obsahuje implementaci autokorelační metody
s2mspec.m	-	obsahuje implementaci spektrální metody

Text

bakalarska_prace.pdf	-	elektronická verze práce ve formátu PDF
bakalarska_prace.odt	-	elektronická verze práce ve formátu OpenOffice

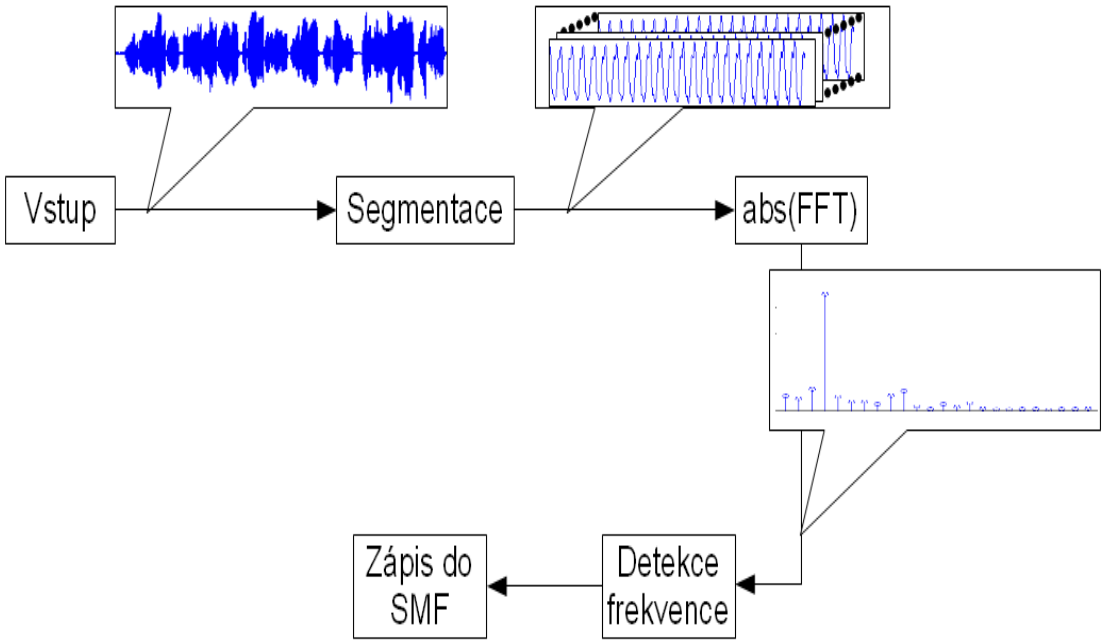
Zvuk

cdur100BPM.wav	-	zvuková nahrávka stupnice C dur s tempem 100 BPM ve formátu WAV
cdur100BPM_sp.mid	-	zvuková nahrávka stupnice C dur s tempem 100 BPM ve formátu MID převedená spektrální metodou
cdur100BPM_au.mid	-	zvuková nahrávka stupnice C dur s tempem 100 BPM ve formátu MID převedená autokorelační metodou
Romanze.wav	-	zvuková nahrávka úryvku skladby „Romanze“ Maxe Regera ve formátu WAV

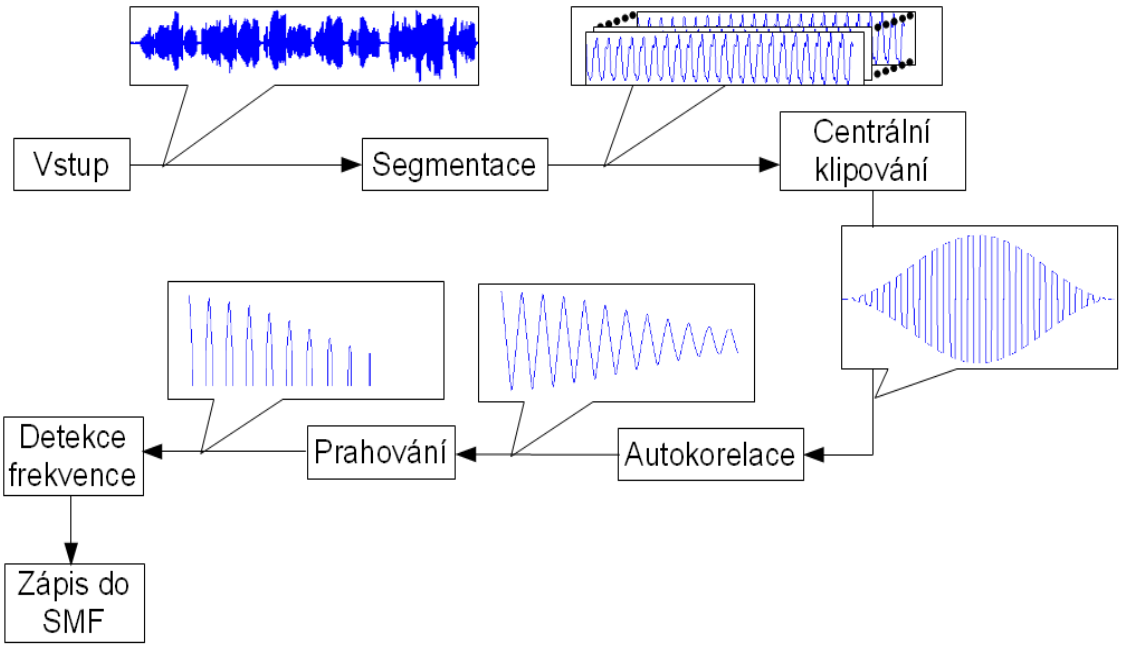
Romanze_sp.mid	-	zvuková nahrávka úryvku skladby „Romanze“ Maxe Regera ve formátu MID prevedena spektrální metodou
Romanze_au.mid	-	zvuková nahrávka úryvku skladby „Romanze“ Maxe Regera ve formátu MID prevedena autokorelační metodou

B BLOKOVÁ SCHÉMATA

Blokové schéma – spektrální metoda



Blokové schéma – autokorelační metoda



C ZDROJOVÉ KÓDY FUNKCÍ

Zdrojový kód funkce s2mauto:

```
function s2mdet(infname, k, silence, treshold, sh_note, tempo)
% waw -> midi
% sound2midi(infname, win_size, <tempo>, <tsig1>, <tsig2>, outfname)
%
% Creates a MIDI file from WAV file
%
% Input arguments:   INFNAME = input file name (*.wav)
%                    K = reduction konstatn for clipping
%                    SILENCE = level of silence
%                    TRESHOLD = treshold for ACF
%                    SH_NOTE = shortest note
%                    TEMPO = bpm, beats per minute (default 120)
%
% Example: sound2midi('samp.wav', 0.2, 0.015, 0.7, 32, 100)
%          creates samp.mid from samp.wav wiht the shortest note 1/32
%          tempo = 100

if isempty(nargin), error('No imput arguments'); end
if nargin < 2, k = 0.2; end
if nargin < 3, silence = 0.015; end
if nargin < 4, treshold = 0.7; end
if nargin < 5, sh_note = 32; end
if nargin < 6, tempo = 120; end

outfname = [infname(1:end-4) '_s2mdet.mid'];
% loads input wav file
[samp, Fs, bits] = wavread(infname);

% divides tempo to 3 bytes
tmp = dec2bin(tempo, 24);
tmp1 = bin2dec(tmp(1 : 8));
tmp2 = bin2dec(tmp(9 : 16));
tmp3 = bin2dec(tmp(17 : 24));

% creates MThd_chunk headr
MThd_chunk = [77 84 104 100 0 0 0 6 0 0 0 1 1 192]; %14 bytes

% creates track_chunk headr
MTrk_chunk = [77 84 114 107 0 0 0 0];% 255 51 3 tmp1 tmp2 tmp3];

headr = [MThd_chunk MTrk_chunk];

fid = fopen(outfname, 'w');
if fid == -1, error('Could not create otput file'); end

% wtires headr into midi file
count = fwrite(fid, headr);
if count ~= length(headr), error('Could not write headr to output
file'); end
```

```
st = fclose(fid);
if st == -1, error('Could not close output file'); end

% creates frequency -> midi note number conversion chart
for n = 0 : 127
    f = 440 * 2 ^ ((n - 69) / 12);
    fr_tab(n + 1) = round(f);
end

% opens file for attachment of midi event
fid = fopen(outfname,'a');
if fid == -1, error('Could not open output file'); end

% win = lenth of win from which the frequenci is being detected
win = round(Fs * ((4 * (1 / sh_note) * 60) / tempo));

% time of one work window in miliseconds
win_time = (win + 1) * 1/Fs * 1000;

%segmentation of input signal
segments = cntrclip(segment(samp, win), k, silence);

%deciding what is row and what is column :-)
len = length(segments(:,1));
[m n] = size(segments);
if m == len, rows = n;
else rows = m; end

% inicialization of counters etc.
m = 4;
cntr = 0;
n(1:4) = 0;

% the heart of whole function :-)
for i = 1 : rows
    % finds frequency of analyzed signal
    buff = fft(segments(:,i),4096);
    buff = buff .* conj(buff);
    buff = ifft(buff);
    buff = buff(1:end/2);
    firstmax = max(buff);
    buff(treshold*firstmax > buff) = 0;

    if find(buff>0)
        lag = 2; ind = 1; p = 1; q = 2;
        while lag(q-1) > 1
            if p == 1, lag(p)=0; end
            if ~isnan(find(buff((sum(ind)+sum(lag)):end) == 0,1,
'first'))
                ind(p) = find(buff((sum(ind)+sum(lag)):end) == 0,1,
'first');
                if(sum(ind)+sum(lag)) <= length(buff)
                    [M(p) lag(p)] = max(buff(sum(ind)+sum(lag):end));
                    p = p + 1; q = p;
                else
                    buff(1:end) = 1;
```

```
                end
            else
                lag(p) = 0;
                p = p + 1; q = p;
            end
        end

        if p>2
            freq(i) = Fs/((sum(ind)+sum(lag)-ind(p-2)-1)/(p-2));
        else
            freq(i) = 0;
        end
    else
        freq(i) = 0;
    end

    [min_diff, n(m)] = min(abs(fr_tab - freq(i)));
    n(m) = n(m) - 1;

    % writes midi event

    if (n(m) == n(m - 1)) || (n(m) < 30) || (n(m) > 95)% || (delta <
sh_note)
      cntr =cntr + 1;

    else

        if n(m - 1) ~= n(m - 2), cntr = 1; end
        delta = round(win_time * cntr);
        bytes = encode_var_length(delta);
        note_off =[bytes 144 n(m - 1) 0];
        count = fwrite(fid, note_off);
        note_on = [0 144 n(m) 64];
        count = fwrite(fid, note_on);
        cntr = 0;
    end
     m = m + 1;

end

% writes end of track
delta = round(win_time * cntr);
bytes = encode_var_length(delta);
end_of_track = [bytes 144 n(m - 1) 0 0 255 47 0];
count = fwrite(fid, end_of_track);

% closes writtn file
st = fclose(fid);
if st == -1, error('Could not close output file after all'); end

% writes a length of MTrk
fid = fopen(outfname, 'r+');
if fid == -1, error('Could not open output file for write length of
MTrk'); end
count = fread(fid );
len = length(count) - length(MThd_chunk) - 8;
```

```
tmp = dec2bin(len, 32);
tmp1 = bin2dec(tmp(1 : 8));
tmp2 = bin2dec(tmp(9 : 16));
tmp3 = bin2dec(tmp(17 : 24));
tmp4 = bin2dec(tmp(25 : 32));
tmp = [tmp1 tmp2 tmp3 tmp4];
st = fseek(fid, 18, -1);
if st == -1, error('Could not set the offset'); end
count = fwrite(fid, tmp );
if count ~= 4, error('Faild to write length of MTrk'); end
st = fclose(fid);
if st == -1, error('Closing output file faild'); end

% converts dec to be written as variable length quantity
function bytes = encode_var_length(val)

binStr = dec2base(round(val), 2);
Nbytes = ceil(length(binStr) / 7);

binStr = ['00000000' binStr];
bytes = [];
for i = 1 : Nbytes
  if (i == 1)
    lastbit = '0';
  else
    lastbit = '1';
  end
  B = bin2dec([lastbit binStr(end - i  * 7 + 1 : end - (i - 1) * 7)]);
  bytes = [B bytes];
end
```

Zdrojový kód funkce s2mspekt:

```
function s2mspec(infname, sh_note, tempo, level)
    % waw -> midi
    % sound2midi('infname', <sh_note>, <tempo>)
    %
    % Creates a MIDI file from WAV file
    %
    % Input arguments:   INFNAME  = input file name (*.wav)
    %                            SH_NOTE   = the shortest note (defaul
demisemiquaver
    %                               -> 1/32)
    %                    TEMPO    = bpm, beats per minute (default 120)
    %                    LEVEL    = level for half frequency detection
    %
    % Example: sound2midi('samp.wav', 32, 100, 0.5)
    %            creates samp.mid from samp.wav wiht the shortest note
1/32
```

```
%           tempo = 100

if isempty(nargin), error('No imput arguments'); end
if nargin < 2, sh_note = 32; end
if nargin < 3, tempo = 120; end
if nargin < 4, level = 0.5; end

%creates output name
outfname = [infname(1:end-3) 'mid'];

% loads input wav file
[samp, Fs, bits] = wavread(infname);

% divides tempo to 3 bytes
tmp = dec2bin(tempo, 24);
tmp1 = bin2dec(tmp(1 : 8));
tmp2 = bin2dec(tmp(9 : 16));
tmp3 = bin2dec(tmp(17 : 24));

% creates MThd_chunk headr
MThd_chunk = [77 84 104 100 0 0 0 6 0 0 0 1 1 192];

% creates track_chunk headr
MTrk_chunk = [77 84 114 107 0 0 0 0];
Tempo_event = [0 255 51 3 tmp1 tmp2 tmp3];
headr = [MThd_chunk MTrk_chunk Tempo_event];

fid = fopen(outfname, 'w');
if fid == -1, error('Could not create otput file'); end

% wtires headr of midi file
count = fwrite(fid, headr);
if count ~= length(headr), error('Could not write headr to output
file'); end
st = fclose(fid);
if st == -1, error('Could not close output file'); end
```

```
% creates frequency -> midi note number conversion chart
for n = 0 : 127
    f = 440 * 2 ^ ((n - 69) / 12);
    freq(n + 1) = round(f);
end

% creates intensity -> velocity conversion chart
vel = logspace(1, 3, 128);
vel = vel / max(vel);

% opens file for attachment of midi event
fid = fopen(outfname,'a');
if fid == -1, error('Could not open output file'); end

% win = lenth of win from which the frequenci is being detected
win = round(Fs * ((4 * (1 / sh_note) * 60) / tempo));

% time of one work window in miliseconds
win_time = (win + 1) * 1/Fs * 1000;

% marks frequency axis
f = Fs / 2 * linspace(0, 1, 4096 / 2);

%silence = level of silence
silence = 0.1;

%segmentation of input signal
segments = segment(samp, win);

%deciding what is row and what is column :-)
len = length(segments(:,1));
[m n] = size(segments);
if m == len, rows = n;
else rows = m; end

% inicialization of counters etc.
m = 3;
```

```
cntr = 0;
han = hann(win);
n(1:4) = 0;
ftest(1:2) = 0;
% reads a window of samples
for i = 2 : rows
    buff = segments(:,i) .* han;
    velocity(m) = max(buff);

    if velocity(m) > silence
        [vel_diff(m), velocity(m)] = min(abs(vel - velocity(m)));
        buff = fft(buff, 4096);
        buff = buff(1:end/2) .* conj(buff(1:end/2));
        % finds frequency of analyzed signal
        [M l] = max(buff);
        half = floor(l/2);
        if half>1
            if (buff(half) > level*M)
                ftest(i) = round(f(floor(l/2)));
            else
                ftest(i) = round(f(l));
            end
        else
            ftest(i) = round(f(l));
        end

    else
        ftest(i) = 0;
    end
    [min_diff, n(m)] = min(abs(freq - ftest(i)));
    n(m) = n(m) - 1;
    % conditions of relevant input signal
    if (n(m) == n(m - 1)) || (n(m) < 55) || (n(m) > 95)
  cntr =cntr + 1;

    else
        delta = round(win_time * cntr);
```

```
            bytes = encode_var_length(delta);
            note_off =[bytes 144 n(m - 1) 0];
            count = fwrite(fid, note_off);
            note_on = [0 144 n(m) 64];
            count = fwrite(fid, note_on);
            cntr = 0;
        end
         m = m + 1;

    end

    % writes end of track
    delta = round(win_time * cntr);
    bytes = encode_var_length(delta);
    end_of_track = [bytes 144 n(m - 1) 0 0 255 47 0];
    count = fwrite(fid, end_of_track);

    % closes writtn file
    st = fclose(fid);
    if st == -1, error('Could not close output file after all'); end

    % writes a length of MTrk
    fid = fopen(outfname, 'r+');
    if fid == -1, error('Could not open output file for write length of
MTrk'); end
    count = fread(fid );
    len = length(count) - length(MThd_chunk) - 8;
    tmp = dec2bin(len, 32);
    tmp1 = bin2dec(tmp(1 : 8));
    tmp2 = bin2dec(tmp(9 : 16));
    tmp3 = bin2dec(tmp(17 : 24));
    tmp4 = bin2dec(tmp(25 : 32));
    tmp = [tmp1 tmp2 tmp3 tmp4];
    st = fseek(fid, 18, -1);
    if st == -1, error('Could not set the offset'); end
    count = fwrite(fid, tmp );
    if count ~= 4, error('Faild to write length of MTrk'); end
```

```
    st = fclose(fid);
    if st == -1, error('Closing output file faild'); end

    % converts dec to be written as variable length quantity
    function bytes = encode_var_length(val)

    binStr = dec2base(round(val), 2);
    Nbytes = ceil(length(binStr) / 7);

    binStr = ['00000000' binStr];
    bytes = [];
    for i = 1 : Nbytes
      if (i == 1)
        lastbit = '0';
      else
        lastbit = '1';
      end
       B = bin2dec([lastbit binStr(end - i  * 7 + 1 : end - (i - 1) *
7)]);
      bytes = [B bytes];
    end
```

Zdrojový kód funkce segment:

```
function segments = segment(vector, win)


    % choses only one track from stereo if needed
    [row column] = size(vector);
    if column == 2, vector = vector(:,1) + vector(:,2); end
    if row == 2, vector = vector(1,:) + vector(2,:); end

    vector = vector - mean(vector);

    A = zeros(win - mod(length(vector), win), 1);

    vector = [vector;A];
```

```
    segments = reshape(vector, [win length(vector)/win]);
```

Zdrojový kód funkce cntrclip:

```
function clip_seg = cntrclip(segments, k, silence)
    %central clipping function
    %function clip_seg = cntrclip(segments, k, silence)
    %input  - matrix of samples
    %output - matrix of center clipped samples

    len = length(segments(:, 1));
    [m n] = size(segments);
    if m == len, rows = n;
    else rows = m; end
    han = hann(len);
    clip_seg = zeros(len, rows);
    if k ~= 0
    for i = 3:rows
    level = k * min([max(abs(segments(:, i-2))) max(abs(segments(:,
i)))]);
    buff = segments(:, i-1);
    buff(level < segments(:, i-1)) = 1;
    buff(-level > segments(:, i-1)) = -1;
    buff(level > abs(segments(:, i-1))) = 0;
    if (level < silence), buff = zeros(len, 1);end
    clip_seg(1:len, i-1) = han .* buff;
    end
    end
```