VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

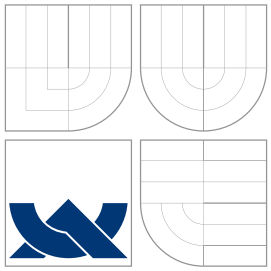
ÚPRAVA VOKÁLNÍCH STOP V SW POSTPRODUKCI HUDBY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

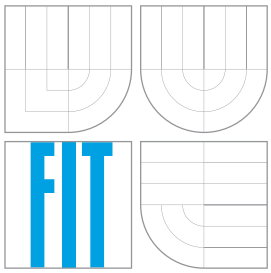
AUTOR PRÁCE
AUTHOR

TOMÁŠ TRKAL

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ÚPRAVA VOKÁLNÍCH STOP V SW POSTPRODUKCI HUDBY

MODIFICATION OF VOCAL TRACKS IN SW POST-PRODUCTION OF MUSIC

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ TRKAL

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. Dr. Ing. JAN ČERNOCKÝ

BRNO 2014

Abstrakt

Tato práce se zabývá návrhem a implementací aplikace pro zpracování vokálních stop. Aplikace je složena z několika dílčích modulů reprezentujících zvukové procesory a efekty a je realizována v podobě pluginu vytvořeného pomocí technologie VST. Plugin disponuje jednoduchým systémem ovládání a integrovanou databází presetů, jenž pomáhají převážně méně zkušeným uživatelům výrazně urychlit proces zvukového zpracování vokálních stop. Z uživatelského hodnocení vyplývá, že aplikaci lze úspěšně využít jak při tvorbě demonahrávek, tak i pro profesionální mix hudebních skladeb.

Abstract

This thesis describes a design and an implementation of a vocal tracks processing application. The application consists of several sub-modules representing audio processors and effects. It is implemented as a plugin created with the VST technology. The plugin provides a simple control system and an integrated database of presets which help mainly less experienced users to significantly speed up the process of processing the vocal tracks. The user evaluation shows that application is successful both in creating demos and professional mixing of music tracks.

Klíčová slova

VST, plugin, nahrávání, mix, mastering, postprodukce, akustika, zvuk, signál, hudba, vokální stopy, rap, DSP, efekt, multiefekt, procesor, kompresor, filtr, ekvalizér, limiter, noise gate, zkreslení, zpožďovací linka

Keywords

VST, plugin, recording, mix, mastering, post-production, acoustics, signal, music, vocal tracks, rap, DSP, effect, multi-effect, processor, compressor, filter, equalizer, limiter, noise gate, distortion, delay line

Citace

Tomáš Trkal: Úprava vokálních stop v SW postprodukci hudby, bakalářská práce, Brno, FIT VUT v Brně, 2014

Úprava vokálních stop v SW postprodukci hudby

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doc. Dr. Ing. Jana Černockého. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Tomáš Trkal
18. května 2014

Poděkování

Chtěl bych poděkovat zejména doc. Dr. Ing. Janu Černockému za vstřícnost při konzultacích, cenné rady a věcné připomínky, které vedly ke zkvalitnění této bakalářské práce. Rád bych také poděkoval Lucii Sližové za profesionální pomoc při vytváření grafického uživatelského rozhraní, Ondrovi Žatkuliakovi a Zdenkovi Kamenickému za jejich drahocenný čas a sdílení dlouholetých zkušeností s postprodukcí hudby a všem, kteří se zúčastnili testování aplikace Executor.

Obsah

Seznam použitých zkratek

A/D	Analogově digitální
ASIO	Audio Streaming Input/Output
BPM	Beats Per Minute
D/A	Digitálně analogový
DAW	Digital Audio Workstation
DDL	Dynamic Link Library
DSP	Digital Signal Processing
EQ	Equalizer
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FS	Full Scale
GUI	Graphical User Interface
HW	HardWare
IIR	Infinite Impulse Response
ISRC	International Standard Recording Code
MATLAB	MATrix LABoratory
MIDI	Musical Instrument Digital Interface
PCI	Peripheral Component Interconnect
RMS	Root Mean Square
RTAS	Real-Time AudioSuite
SDK	Software Development Kit
SR	Sampling Rate
SW	SoftWare
USB	Universal Serial Bus
VST	Virtual Studio Technology

Kapitola 1

Úvod

V poslední době ceny hardwarových zvukových zařízení znatelně klesají a současně výrazně roste výkonnost osobních počítačů. V důsledku toho si takřka každý hudebník může zrealizovat vlastní domácí nahrávací studio. Úspěch hudební skladby však nezávisí pouze na talentu a dovednostech interpreta, ale také na zvukovém zpracování nahrávky. Klíčovým procesem pro zvýšení kvality hudební skladby je takzvaná postprodukce. Hudební postprodukcí se myslí editace, míchání (mix) a mastering nahrávky. Posprodukční praktiky ovšem bývají často netriviální a časově náročné a jejich úspěšná aplikace zpravidla vyžaduje značnou dávku trpělivosti a zkušeností.

Cílem této bakalářské práce je návrh a implementace aplikace pro zpracování vokálních stop. V rámci seznámení s digitálním zpracováním signálu a s principy zvukových procesorů a efektů byl vytvořen prototyp pomocí interaktivního programového prostředí MATLAB. Výsledná aplikace Executor má podobu zásuvného pluginu a byla implementována v programovacím jazyce C++ s využitím technologie VST, která umožňuje zpracování zvuku v reálném čase. Plugin se skládá z několika dílčích modulů a disponuje integrovanou databází přednastavených presetů a jednoduchým systémem ovládání. Tyto mechanismy pomáhají zejména méně zkušeným uživatelům k dosažení co nejlepších zvukových výsledků v co nejkratším čase.

V první kapitole této práce jsou vysvětleny základy akustiky, které seznamují čtenáře s elementární terminologií a fyzikálními vlastnostmi zvuku. Druhá kapitola se zabývá nahráváním vokálních stop a základními i pokročilejšími technikami, které se využívají při editaci, mixu a masteringu hudebních skladeb. Ve třetí kapitole je uvedeno rozdělení zvukových efektů a procesorů s detailním popisem jejich parametrů, funkce a využití. Zbývající kapitoly se věnují návrhu a implementaci zmíněné aplikace a jejímu testování.

Kapitola 2

Základy akustiky

V této kapitole jsou vysvětleny fyzikální základy velmi rozsáhlého a komplexního vědního oboru *akustiky*. Jednou ze zásadních oblastí akustického výzkumu je fyzikální akustika, zabývající se vznikem a následným šířením, odrazem a pohlcováním zvuku v různých prostředích a materiálech. Další významnou oblastí akustiky je akustika hudební. Toto odvětví studuje fyzikální vlastnosti hudebních nástrojů a hudby obecně. Stavební akustika zkoumá ideální podmínky pro poslouchání hudby, respektive řeči, v uzavřených prostorách a její poznatky jsou hojně využívány především při stavbě nahrávacích a postprodukčních studií. Vznikem zvuku v lidských hlasových orgánech a naopak vnímáním zvuku pomocí uší se zabývá akustika fyziologická. Poslední z disciplín tohoto vědního okruhu, kterou zmíním, je akustika elektroakustická, zkoumající možnosti záznamu, přenosu a opětovné reprodukce zvuku s využitím elektrického proudu.

Seznámení se základními principy akustiky a vlastnostmi zvuku je důležitým předpokladem pro pochopení problematiky týkající se využívání zvukových efektů během procesu postprodukce hudby. Tato kapitola ovšem, jak už bylo naznačeno výše, seznamuje čtenáře pouze se základy, nutnými pro orientaci v následujících kapitolách. Pro podrobnější informace odkazuji na zdroje [6], [21] a [15], ze kterých jsem vycházel.

2.1 Fyzikální vlastnosti zvuku

Zvuk lze nejlépe definovat jako kmitavý pohyb částic v látkovém prostředí, při němž dochází k přenosu energie mezi částicemi, díky které se zvuk šíří v prostoru v podobě mechanického vlnění od zdroje zvuku k posluchači, kde vyvolává zvukový vjem. Zdrojem zvuku je kmitající těleso, tedy například kytarová struna, membrána reproduktoru či úder paličky do bubnu. Frekvenční rozsah slyšitelného zvuku je velmi individuální a obvykle se s rostoucím věkem snižuje. V odborné literatuře jsou běžně uváděny hodnoty v rozsahu od 16 Hz do 20 kHz. Extrémně nízké zvukové frekvence pod hranicí 16 Hz jsou označovány jako *infrazvuk*, naopak vysoké frekvence nad 20 kHz se nazývají *ultrazvuk*. Lidské tělo je schopné vnímat infrazvuk pomocí hmatu. Ultrazvuk je využíván k sonografii. Periodický pohyb lze popsat rovnicí:

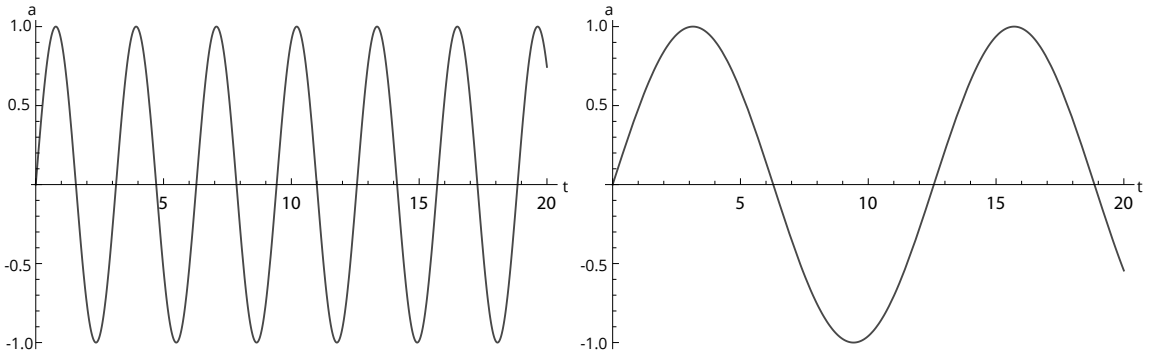
$$a = A \sin(2\pi f t + \varphi), \tag{2.1}$$

kde a je okamžitá výchylka v čase t, A maximální amplituda, φ fázový posun a f frekvence, která označuje počet kmitů za jednotku času. V literatuře můžeme narazit na pojem perioda T, udávající dobu trvání jednoho kmitu. Frekvence f je převrácenou hodnotou periody T, tedy platí vztah $f = 1/T$.

Zvuk se ve volném prostoru šíří všemi směry ve formě takzvaných vlnoploch rychlostí c, která je závislá na látkovém prostředí, jeho teplotě a vlnové délce λ. Standardizovaná rychlost zvuku ve vzduchu, odpovídající teplotě $13,6\ °C$, je 340 m/s. V reálném akustickém poli zvukové vlny dopadají na překážky, přičemž dochází k odrazům, pohlcení či ohybu zvuku okolo překážky. Množství odražené akustické energie závisí na velikosti, tvaru a materiálu překážek. Úhel odrazu zvuku je vždy stejný jako úhel dopadu. V místnosti dochází k četným odrazům od stěn, načež k posluchači přichází jednotlivé vlny s různým zpožděním. Vzniká tak jev zvaný dozvuk, který se může negativně projevit například nesrozumitelností vokálních stop při poslechu, nahrávání či postprodukci. K eliminaci dozvuku v nahrávacích studiích se v praxi používají akustické absorbéry a difuzéry [18].

2.1.1 Výška a barva zvuku

Na rozdíl od takzvaného *hluku*, který se vyznačuje neperiodickým charakterem, nazýváme pravidelné periodické kmitání *tónem*. Kromě praskání a brumu lze mezi hluk zařadit i většinu bicích nástrojů. Mezi základní vlastnosti každého tónu patří výška, barva a hlasitost. Vnímání výšky tónu závisí na frekvenci zvuku. Čím vyšší je jeho frekvence, tím vyšší je i tón a naopak. Jako základní tón se uvádí tzv. komorní A, jehož frekvence odpovídá přibližně 440 Hz. Poměr vyšší a nižší frekvence dvou tónů se nazývá hudební interval. Nejdůležitějším hudebním intervalem, který je základem většiny hudebních stupnic, je oktáva s poměrem frekvencí 2:1. Tón o oktávu vyšší získáme zdvojnásobením frekvence nižšího tónu.



Obrázek 2.1: Vysoký a nízký tón.

V praxi se převážně setkáváme s tóny *složenými*, které na rozdíl od tónů *jednoduchých* obsahují kromě průběhu základní frekvence také další vyšší harmonické průběhy. Počet vyšších harmonických tónů a velikost jejich amplitud zásadně ovlivňují barvu tónu, což vede k tomu, že stejné tóny zahrané na elektrickou kytaru a klavír mají výrazně odlišný charakter. Rozdílná barva hudebních nástrojů je dána zejména materiálem, tvarem a velikostí nástroje. Jednoduché tóny lze uměle vytvořit použitím generátoru. Složený tón lze popsat funkcí:

$$f(t) = \sum_{n=1}^{x} A_n \sin\left(n2\pi ft + \varphi_n\right). \tag{2.2}$$

Speciální odvětví zvuků tvoří *šumy*, jejichž barevné označení je odvozeno od analogie se spektrem barevného světla. Za zmínku stojí především *bílý* a *růžový* šum. Jako bílý šum lze označit signál s náhodným průběhem a rovnoměrně rozloženou výkonovou spektrální

hustotou. Jinými slovy stejně široká pásma mají shodný výkon. Bílý šum se používá pro testování filtrů a zesilovačů. U růžového šumů je výkonová frekvenční hustota přímo úměrná převrácené hodnotě frekvence. Růžový šum se využívá k nastavení ekvalizéru při ozvučování.

2.1.2 Hlasitost a intenzita zvuku

Při šíření zvuku dochází k periodickým změnám atmosférického tlaku v prostoru, které uchem vnímáme jako subjektivní hlasitost, závislou na citlivosti sluchu. Oproti tomu *intenzita zvuku* I je objektivní a je definována jako podíl výkonu P a plochy S, kterou zvuk prochází. Jako tzv. *práh slyšení* označujeme hranici akustického tlaku 20 μPa, od které ucho začíná zvuk vnímat. Naopak hranice, od které zvuk vyvolává pocity bolesti, se nazývá *práh bolesti* a odpovídá hodnotě 130 Pa. Hladina intenzity L je pak definována vztahem:

$$L = 10 \log \frac{I}{I_0} = 20 \log \frac{p}{p_0}, \tag{2.3}$$

kde I_0 označuje intenzitu prahu slyšení. Analogicky lze tento vztah vyjádřit pomocí akustického tlaku, kde p_0 je akustický tlak odpovídající prahu slyšení. *Hladina intenzity zvuku* se uvádí v logaritmických jednotkách decibelech $[dB]$. *Decibel* je bezrozměrná jednotka, která se využívá k nepřímému vyjádření intenzity zvuku, vztažené k referenční hodnotě intenzity zvuku prahu slyšení.

V tabulce 2.1 jsou uvedeny příklady běžných zvuků a odpovídající hodnoty hladin intenzity zvuku. Pokud dojde k současnému působení dvou zvuků, jejichž hladiny intenzity zvuku se liší o více než 10 dB, může dojít k situaci, kdy silnější zvuk zamaskuje zvuk slabší tak, že jej ucho nedokáže rozpoznat. Kromě intenzity zvuku je zvukový vjem silně závislý na frekvenci. Maximální citlivost sluchu se pohybuje mezi 500 a 4000 Hz, pro nižší a vyšší frekvence výrazně klesá.

L [dB]	**Zvuk**	**L [dB]**	**Zvuk**
0	práh slyšení	80	orchestr
10	šelest listí	100	vrtačka
20	tikot hodinek	110	koncert
40	tichý rozhovor	120	letadlo
60	ruch v davu	130	práh bolesti

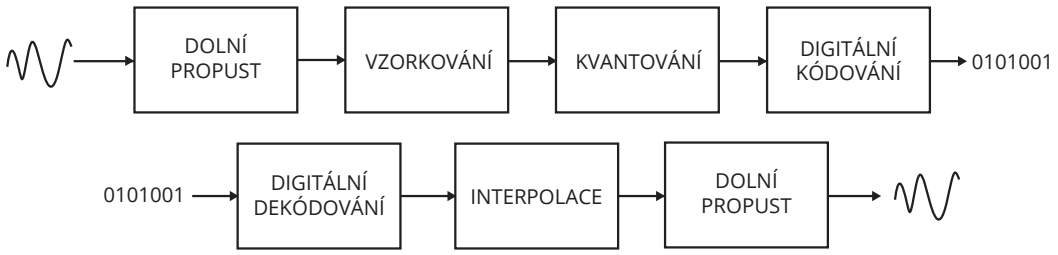
Tabulka 2.1: Příklady hladin intenzity zvuků [15].

2.2 Digitalizace zvuku

V dnešní době se můžeme setkat se dvěma odlišnými přístupy ke zpracování zvukového signálu. *Analogový* zvukový signál je na rozdíl od toho *digitálního* spojitý, tzn. jeho hodnota je definována pro všechny časové okamžiky. Fyzicky může být reprezentovaný například rýhou gramofonové desky nebo vrstvou kovu na magnetické pásce. Oproti tomu digitální signál je diskrétní, tedy definovaný pouze v určitých časových okamžicích. Převod analogového signálu na digitální se skládá ze dvou kroků.

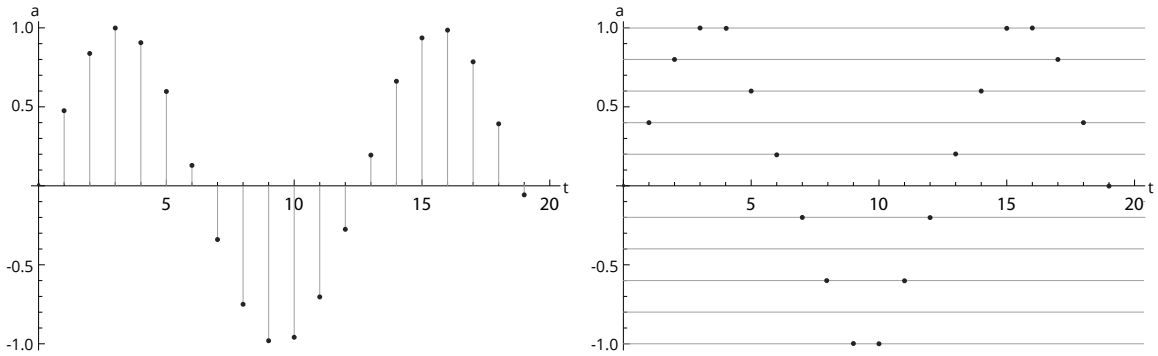
Prvním krokem je proces *vzorkování*, při kterém je vstupní signál transformován na posloupnost periodicky odebraných diskrétních vzorků. Aby mohl být signál zpětně reprodukovaný bez zkreslení, musí být vzorkovací kmitočet, dle *Nyquistova teorému*, minimálně

dvakrát vyšší, než je nejvyšší frekvence obsažena ve vzorkovaném signálu. Jelikož lidské ucho neslyší zvuky s frekvencí nad 20 kHz, stala se vzorkovací frekvence 44,1 kHz nejpoužívanější pro většinu komerčních zařízení, včetně kompaktních disků. Pokud není dodržena Nyquistova podmínka, může dojít k nevratnému zkreslení zvanému *aliasing*, proto je obvykle vstupní signál před vzorkováním frekvenčně omezen dolní propustí.



Obrázek 2.2: Konverze mezi analogovými a digitálními signály [15].

Druhým krokem převodu je proces *kvantování*, při kterém jsou navzorkované hodnoty signálu zaokrohleny k nejbližším kvantovacím hladinám a dále zakódovány do binární podoby. Počet bitů použitých pro uložení jednoho vzorku se označuje jako *bitová hloubka* a udává konečnou množinu hodnot, kterých zakódovaný signál může nabývat. S ohledem na výpočetní a paměťovou náročnost jsou nejčastěji voleny hodnoty 16 či 24 bitů. Výhodou při práci s digitálními záznamy jsou téměř neomezené možnosti střihu a editace zvuku a jeho kopírování bez ztráty kvality. Nevýhodou pak ztráta informace mezi jednotlivými vzorky analogového signálu, kterou je nutné při zpětné reprodukci interpolovat.



Obrázek 2.3: Vzorkovaný a kvantovaný signál.

S nástupem digitální techniky do zvukové praxe byla stanovena maximální hodnota amplitudy signálu na 0 dBFS. Zkratka *FS* značí *Full Scale* neboli *plný rozsah* a oznamuje, že měřená hodnota není vztažena k referenční hodnotě prahu slyšení, nýbrž právě k maximální hodnotě amplitudy. Převody mezi velikostí amplitudy signálu a a jeho hlasitostí v decibelech l_{dBFS} lze vyjádřit pomocí vztáhů:

$$l_{dBFS} = 20 \log_{10} a, \tag{2.4}$$

$$a = 10^{l_{dBFS}/20}. \tag{2.5}$$

Kapitola 3

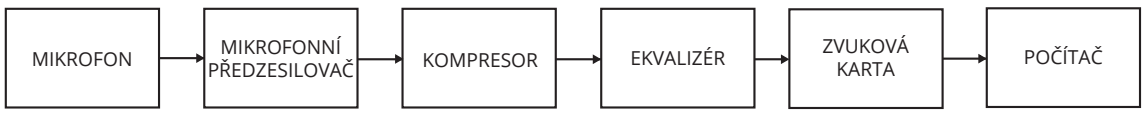
Etapy vzniku nahrávky

Tato kapitola seznamuje čtenáře s jednotlivými etapami vzniku skladby, kterými jsou nahrávání, editace, mix a mastering. Postupně jsou zde nastíněny základní i pokročilejší metody a nástroje využívané při nahrávání a během postprodukce hudby. Při psaní této kapitoly jsem vycházel zejména z vlastních zkušeností a z knih [18], [8] a [13].

3.1 Nahrávání vokálních stop

První fází procesu tvorby hudební skladby, která předchází editaci, mixu a masteringu, je *nahrávání*. Důležitost nahrávání bývá nezřídka podceňována, proto je zapotřebí si uvědomit, že následné kroky postprodukce se mohou značně zkomplikovat v případě, kdy editujeme nekvalitně nahraný materiál s vysokým obsahem šumu či jiných nečistot. V první řadě je nezbytně nutné mít vhodně akusticky uzpůsobenou nahrávací místnost, aby nedocházelo k nežádoucím odrazům zvuku nebo kumulaci frekvencí pouze určitého, například basového, pásma.

Dále je nutné eliminovat, nebo alespoň výrazně zredukovat, výskyt dvou typů hlásek. Prvním z nich jsou takzvané *retnice*, neboli hlásky explozivního charakteru vytvářející prudké zvukové rázy, které při dopadu na membránu mikrofonu způsobují rušivé hluky. Mezi retnice lze zařadit *b, p, d* či *t* a lze je výrazně potlačit použitím nylonového, případně kovového, pop filteru a vhodnou vzdáleností interpreta od mikrofonu. Druhým typem jsou *sykavky*, mezi které patří hlásky *c, s* a *z*. Sykavky se v praxi odstraňují použitím jiného typu mikrofonu nebo při mixu použitím ekvalizéru nebo de-esseru [18].



Obrázek 3.1: Nahrávací řetězec.

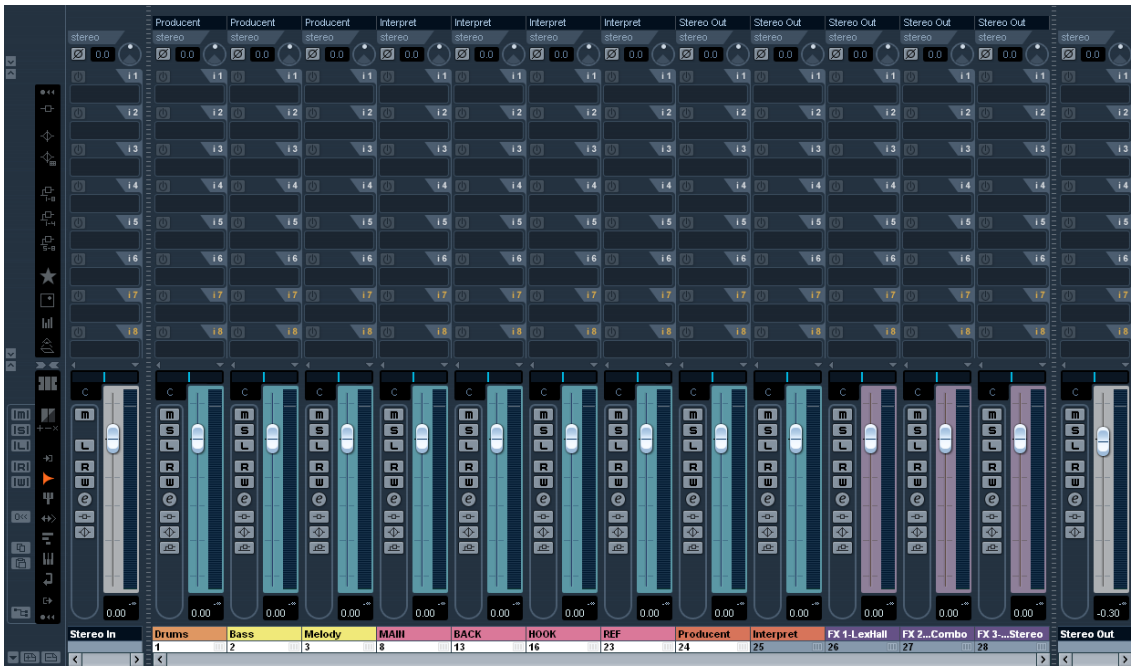
Na obrázku 3.1 je uveden příklad hardwarového nahrávacího řetězce. Prvním velmi důležitým prvkem je *mikrofon*. Při nahrávání vokálních stop se nejčastěji setkáme s kondenzátorovými velkomembránovými mikrofony, které mají oproti svým dynamickým protějškům mnohem vyšší citlivost. *Mikrofonní předzesilovač* je zařízení, které zesiluje a upravuje výstupní signál z mikrofonu pro jeho další využití. Kromě toho zpravidla disponuje fantomovým napájením právě pro kondenzátorové mikrofony.

V řetězci mohou být zapojeny i efekty typu kompresor a ekvalizér pro úpravu dynamiky a frekvenčního spektra signálu. Pro A/D a D/A převod signálu se využívá audio rozhraní v podobě interní či externí *zvukové karty*. Je důležité si uvědomit, že kvalita zvukového signálu závisí na nejslabším článku řetězce. I zvuk nahraný pomocí nejkvalitnějších mikrofonů bude degradován při použití laciného zvukového rozhraní pro převod zvuku do digitální podoby.

3.2 Editace a mix skladby

Nahráním jednotlivých stop nástrojů a zpěvu proces vzniku nahrávky zdaleka nekončí. Druhou a asi nejvýznamnější fází tohoto procesu je *editace* a *mix* skladby. Úkolem studiového inženýra je poskládat všechny zvukové stopy do jednoho celku tak, aby každý nástroj nalezl v mixu svůj prostor a zároveň z něj nepřirozeně nevyčníval.

Tento úkon obnáší využití audio procesorů pro frekvenční úpravy zvuku, během nichž jsou dominantní frekvence signálu zvýrazněny a naopak ty neužitečné potlačeny. Kromě editace frekvenčního spektra signálu je nezřídka nutné upravit dynamický rozsah a hlasitostní poměry jednotlivých stop a jejich umístění ve zvukovém panoramatu skladby. Důkladnějšímu popisu parametrů, funkce a využití zvukových procesorů a efektů se podrobně věnuji v kapitole 4.



Obrázek 3.2: Mixážní pult aplikace Steinberg Cubase.

V minulosti představoval střih a mix nahrávky časově náročný a velmi pracný proces, jelikož veškeré stopy byly zaznamenávány na magnetických páskách a jen přesunutí na začátek nahrávky znamenalo přetočení celé pásky. Základem dnešních profesionálních nahrávacích a produkčních studií bývají zařízení zvaná *DAW* (Digital Audio Workstation), která jsou výhradně určena pro nahrávání a editaci zvuku. Jak bylo naznačeno v předchozí

kapitole, tyto stanice lze do určité míry nahradit i univerzálním počítačem se zvukovou kartou a vhodným softwarem. Mezi nejpoužívanější programy pro práci se zvukem patří produkty Cubase a Nuendo od firmy Steinberg[1] a Protools od firmy Avid[2].

Samotnému procesu mixu skladby většinou předchází fáze *editace*, při které jsou stopy postupně procházeny a čištěny od nežádoucích zvuků. Vyjma toho dochází k selekci užitečných stop a k zahození těch, které nebudou při mixu využity. Pokud se například refrénové vokální stopy nahrávají vícekrát pod sebou, může dojit k situaci, kdy nemusí paralelní fráze přesně sedět na sebe a je nutné je rytmicky dorovnat. Velmi užitečnou a rozšířenou praktikou bývá použití funkcí *fade in* a *fade out* pro vytvoření lineárního, logaritmického či exponenciálního náběhu nebo doběhu hlasitosti v určitém úseku stopy. K plynulému prolnutí dvou stop se používá fukce *crossfade* [8].

Na obrázku 3.2 je zobrazen virtuální mixážní pult aplikace Steinberg Cubase, který umožňuje podobně jako jeho fyzický ekvivalent nastavit poměr úrovní jednotlivých stop skladby. Kromě toho poskytuje prostředky pro vkládání zvukových efektů do signálové cesty a integrovaný čtyřpásmový ekvalizér. Většina moderních aplikací umožňuje vytvářet vlastní sběrnice, do kterých lze zapojit výstupy jednotlivych stop a následně je editovat jako celek. Úpravy zvuku lze aplikovat jak destruktivně tak nedestruktivně. Při destruktivní editaci dochází ke změnám souboru, ve kterém je zvuk digitálně uložen, zatímco nedestruktivní úpravy probíhají pouze virtuálně a v reálném čase. Nevýhodou pak ovšem může být zvýšené vytížení procesoru.

Existuje celá řada metod, jak postupovat při mixu nahrávky, a záleží pouze na zvukaři, kterou z nich si vybere a osvojí. Osobně preferuji nejdříve začít s rytmickou sekcí, tedy mixem jednotlivých stop bicích, pokračovat přimícháním basové linky a následně ostatních hudebních nástrojů a vokálních stop. Zpěvu je zapotřebí věnovat zvýšenou pozornost, jelikož hlasový projev, narozdíl od hudebního nástroje, nese kromě melodické informace i informaci textovou. Mezi pokročilejší praktiky mixu patří například využití paralelní komprese či automatizace [9].

3.3 Mastering skladby

Mastering je poslední etapou vzniku skladby, která by nahrávce měla dokreslit charakter. Pojem mastering bývá laiky nezřídka zaměňován za pojem mix a naopak. Na první pohled se oba procesy mohou jevit podobné, nicméně je třeba si uvědomit, že při mixu dochází k úpravám jednotlivých stop, zatímco mastering pracuje s nahrávkou jako s celkem. Úkolem masteringu rozhodně není opravovat chyby, které vznikly při mixu skladby. Pokud například nahrávka nemá vhodně nazvučenou kytaru, je mnohem vhodnějším řešením vrátit náhravku zpět k přemíchání, než se snažit zvuk kytary opravit.

Cílem masteringu je doladit skladbu zpravidla v rámci většího celku. Jinými slovy skladby z jednoho alba by měly působit uceleně a posluchač by při jeho poslechu neměl být ničím rušen. Součásti masteringu je i příprava dat pro výrobu nosičů včetně vyplnění metadat se jmény interpretů, názvy jednotlivých skladeb a jejich ISRC kódu. ISRC kód nahrávky je jedinečný dvanáctimístný alfanumerický kód identifikující audio nahrávky a videoklipy. ISRC kódy spravuje a přiděluje INTEGRAM[3].

[1] Webové stránky firmy Steinberg: http://www.steinberg.net

[2] Webové stránky firmy Avid: http://www.avid.com

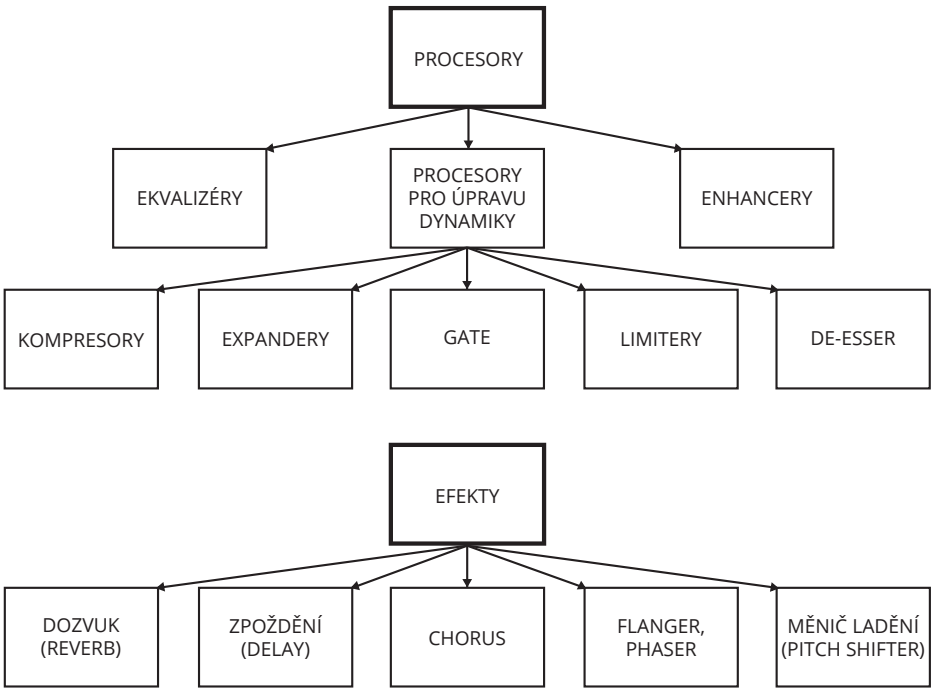
[3] INTEGRAM, nezávislá společnost výkonných umělců a výrobců zvukových a zvukově obrazových záznamů o.s.: http://www.intergram.cz

Kapitola 4

Zvukové procesory a efekty

Systémy využívané ke zpracování zvuku můžeme rozdělit do dvou kategorií, viz obrázek 4.1. První kategorii tvoří takzvané zvukové *procesory*, z jejichž výstupu odebíráme signál, který byl dynamicky, frekvenčně či jinak upraven. Tyto procesory zpravidla nedisponují ovládáním poměru přímého (dry) a upraveného (wet) signálu. Druhá kategorie je tvořena zvukovými *efekty*, které výsledný signál získávají tak, že ke vstupnímu signálu přičítají jeho zpožděné kopie.

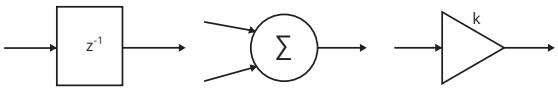
Zvukové procesory a efekty mohou být realizovány buď v podobě hardwarových zařízení či nástrojových pedálů nebo jako softwarové moduly pro audio programy. Nespornou výhodou fyzických zařízení je jejich originální zvukový charakter a přítomnost ovládacích prvků pro pohodlnou manipulaci. Na druhou stranu, pokud si pořídíme softwarový plugin, lze ho v projektu aplikovat opakovaně. Při psaní této kapitoly jsem čerpal ze zdrojů [18], [20] a [19], které se problematice zvukových efektů a procesorů věnují podrobně.



Obrázek 4.1: Základní rozdělení zvukových procesorů a efektů dle [18] a [9].

4.1 Procesory pro úpravu frekvenčního spektra signálu

Filtr je lineární systém využívaný pro úpravu frekvenčního spektra signálu. Pomocí filtru tedy lze některé frekvenční složky signálu potlačit nebo naopak zvýraznit. V analogové sféře se setkáme s filtry *aktivními*, které na rozdíl od filtrů *pasivních*, používají vedle odporů, kondenzátorů a cívek i zapojení aktivního prvku. Aktivním prvkem může být operační zesilovač či tranzistor. Analogové filtry získávají s postupem času jedinečný charakter způsobený opotřebením součástek. Použití číslicových filtrů namísto analogových přináší výhody ve formě snadného návrhu a především vysoké přesnosti filtru.



Obrázek 4.2: Základní stavební bloky pro realizaci číslicových filtrů.

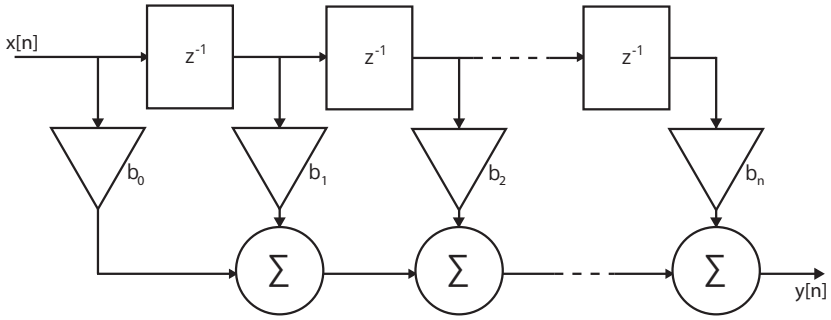
Číslicový filtr lze sestavit s použitím třech základních stavebních elementů, viz obrázek 4.2. Kromě zpožďovacího bloku se jedná o blok sčítací a blok pro násobení. Filtr lze popsat diferenční rovnicí, impulzní odezvou či kmitočtovou charakteristikou. Pro získání přenosové funkce diskrétního systému je nutné přejít do komplexní roviny použitím takzvané z-transformace. Ekvivalentem z-transformace pro spojité systémy je Laplaceova transformace. Z hlediska odezvy lze diskrétní lineární filtry rozdělit na filtry s konečnou a nekonečnou impulzní odezvou [19].

4.1.1 Filtry s konečnou impulzní odezvou

Jak je vidět na obrázku 4.3, filtry s konečnou impulzní odezvou (FIR, Finite Impulse Response) využívají k výpočtu výstupního signálu pouze aktuální a zpožděné vzorky signálu vstupního. Na základě absence zpětnovazebních bloků lze tyto filtry označit jako nerekurzivní. Diferenční rovnice těchto filtrů má tvar:

$$y[n] = \sum_{k=0}^{n} b_k x[n-k]. \tag{4.1}$$

Mezi hlavní výhody filtrů s konečnou impulzní odezvou patří zejména intuitivní návrh a jednoduchá realizace. Tyto filtry jsou nerekurzivní a tedy vždy stabilní a tudíž je vyloučeno kmitání. Oproti filtrům s nekonečnou impulzní odezvou je ovšem třeba pro dosažení strmých přechodů mezi propustným a nepropustným pásmem použít filtry mnohem vyššího řádu.



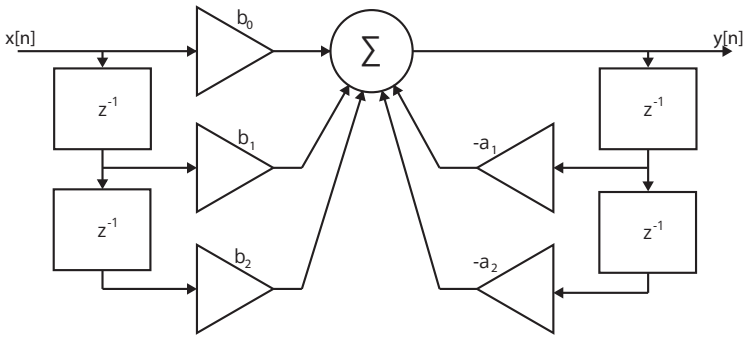
Obrázek 4.3: Obecné blokové schéma filtru s konečnou impulzní odezvou.

4.1.2 Filtry s nekonečnou impulzní odezvou

Pro zpracování zvukových signálu se mnohem častěji využívají filtry s nekonečnou impulzní odezvou (IIR, Infinite Impulse Response) a proto se jim v této části práce budu věnovat podrobněji. Na rozdíl od filtrů s konečnou impulzní odezvou využívají zpětnovazebních smyček, pročež bývají označovány jako rekurzivní a jejich obecná diferenční rovnice obsahuje i autoregresní člen ovlivňující rychlost odezvy:

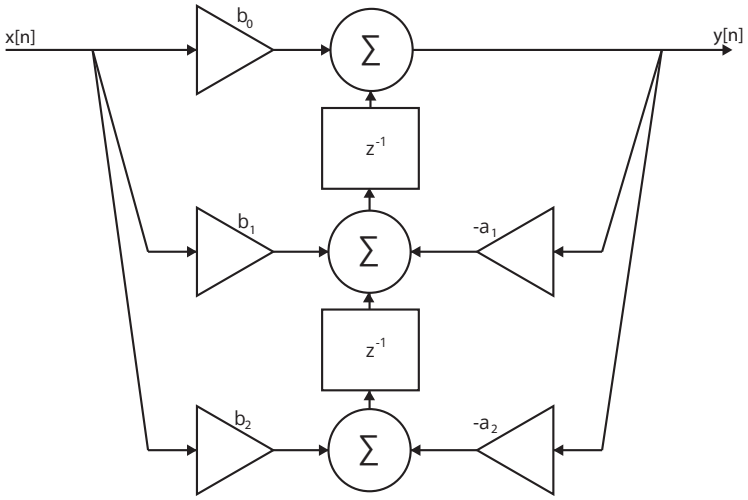
$$y[n] = \sum_{i=0}^{N} b_i x[n-i] - \sum_{i=1}^{M} a_i y[n-i]. \tag{4.2}$$

Nevýhodou IIR filtrů je vzhledem k jejich rekurzivnímu provedení relativně složitý a méně intuitivní návrh. Nespornou výhodou ovšem zůstává, že i při malém řádu filtru lze dosáhnout velmi strmých přechodů mezi propustným a nepropustným pásmem. K IIR filtrům narozdíl od filtrů FIR obvykle existuje analogový ekvivalent [14].



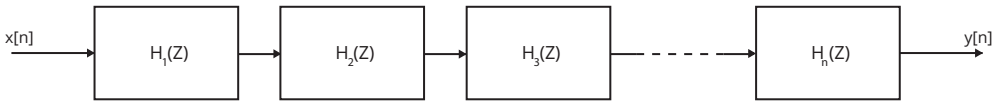
Obrázek 4.4: Blokové schéma filtru IIR druhého řádu v I. přímé formě.

IIR filtry lze realizovat v několika různých formách dle požadavků na jejich paměťovou a výpočetní složitost. Na obrázku 4.4 je uveden příklad IIR filtru druhého řádu v I. přímé formě vhodný zejména pro procesory pracující s pevnou řádovou čárkou, neboť operuje pouze s jedním sčítacím prvkem.



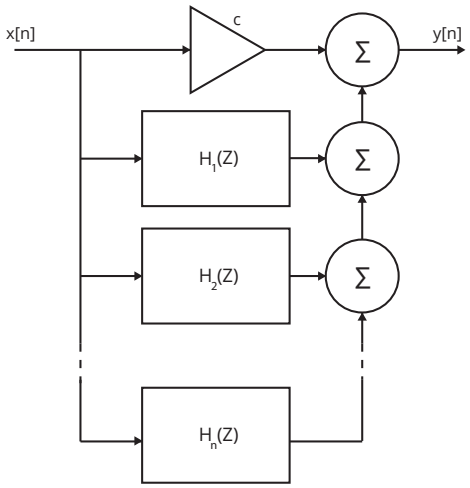
Obrázek 4.5: Blokové schéma filtru IIR druhého řádu v duální formě.

Transpozicí lze vytvořit takzvanou duální formu filtru, která využívá minimum paměťových prvků a naopak více prvků sčítacích. Z tohoto důvodu je pro aritmetiku v plovoucí řádové čárce vhodnější než přímá forma. Příklad blokového schématu v duální formě je zobrazen na obrázku 4.5. Proces transpozice spočívá ve třech snadných krocích. Nejprve je třeba obrátit směr toku signálu ve všech větvích, následně zaměnit vstup filtru za jeho výstup a nakonec zaměnit všechny uzly za sumarizační bloky a naopak [16].



Obrázek 4.6: Sériové neboli kaskádní zapojení filtrů.

Filtry vyšších řádů je z hlediska výpočetní a paměťové náročnosti vhodné realizovat pomocí systému složeného z více filtrů nižších řádu. Rozložením přenosové funkce na součin dílčích funkcí prvního a druhého řádu získáme sériovou neboli kaskádní formu zapojení (obr. 4.6). Paralelní formu zapojení (obr. 4.7) získáme rozložením přenosové funkce na parciální zlomky. Filtry druhého řádu používané pro konstrukci větších systémů se nazývají *bikvady* a jsou charakterizovány koeficienty b_0, b_1, b_2, a_1 a a_2. Výpočet koeficientů jednotlivých typů filtrů je uveden v příloze B.

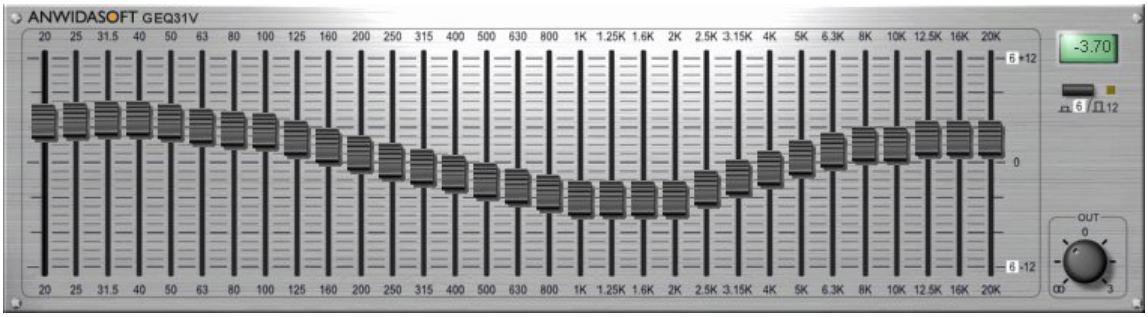


Obrázek 4.7: Paralelní zapojení filtrů.

4.1.3 Ekvalizéry

Ekvalizér je jedním z nejdůležitějších a nejpoužívanějších nástrojů pro práci se zvukem. Jeho uplatnění nalezneme od nahrávání ve studiu, přes mix a mastering až po ozvučování živých koncertů. Ekvalizér si lze představit jako sestavu filtrů, pomocí kterých můžeme detailně upravit frekvenční spektrum zvukové stopy. Jinými slovy s ním můžeme potlačit některé nežádoucí frekvence a zároveň ty nevýrazné zesílit. Na trhu je nespočet jak analogových tak digitálních ekvalizérů, které lze rozdělit na dvě základní skupiny [13].

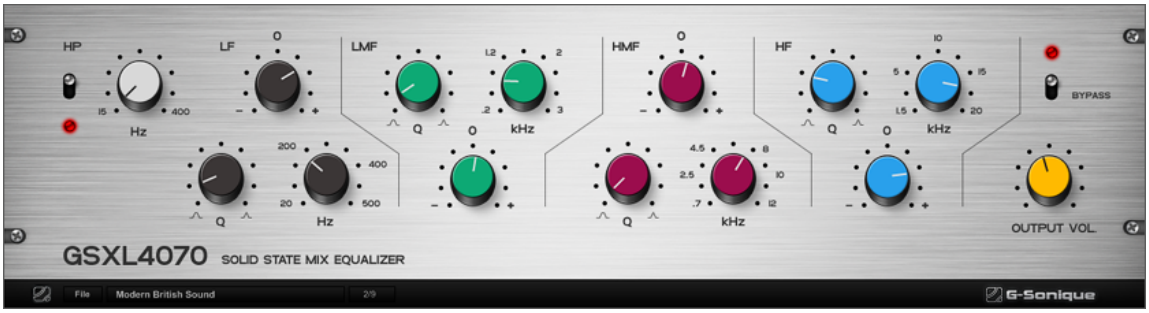
První skupinou jsou vícepásmové *grafické* ekvalizéry (obr. 4.8), které svůj název získaly podle osazení tahovými potenciometry, jejichž pozice symbolizují pomyslný graf frekvenční charakteristiky. Frekvenční pásmo je rovnoměrně rozděleno na několik dílčích pásem



Obrázek 4.8: VST grafický ekvalizér Anwida Soft GEQ31V.

a každému z nich odpovídá jeden tahový potenciometr (fader, jezdec). Mixážní pulty obvykle disponují integrovanými třípásmovými ekvalizéry pro úpravu basového, středového a výškového pásma. V profesionální praxi se však můžeme setkat i s ekvalizéry s deseti až jednatřiceti pásmy.

Druhou skupinu tvoří ekvalizéry *parametrické*, kterými lze díky nastavení parametrů jednotlivých pásem dosáhnout například úzkopásmových korekcí. Prvním z parametrů je nastavení centrální frekvence pásma F_c, čímž se mimo jiné parametrické ekvalizéry výrazně odlišují od ekvalizérů grafických, jejichž pásma jsou fixně přednastavena. Druhým parametrem je šířka pásma, někdy označována také jako kvalita, případně jakost. Čím vyšší je činitel jakosti Q, tím užší je pásmo, se kterým pracujeme a naopak. Třetím a posledním parametrem je takzvaná kvantita ovlivňující zesílení respektive zeslabení pásma.



Obrázek 4.9: VST parametrický ekvalizér G-Sonique GSXL4070.

Kromě zmíněných dvou kategorií se můžeme setkat ještě s třetí hybridní architekturou ekvalizérů, jenž pro jednotlivá pásma disponují různým počtem ovládacích prvků. Tedy zatímco například pro úpravu vysokých a nízkých frekcencí jsou k dispozici pouze ovladače pro zesílení, pro středové kmitočty nalezneme ovládací prvky i pro nastavení středové frekvence a šířky pásma. Důvodem zavedení hybridních ekvalizéru jsou většinou požadavky na ušetření místa v rámci ovládání větších zařízení. Parametry, pro které nejsou k dispozici ovládací prvky, jsou továrně nastaveny.

Ekvalizéry by měly být využívány s citem. Obecně platí, že pro korekce frekvenčního spektra signálu bývá aplikován výraznější útlum v úzkém pásmu, naopak dobarvení zvuku lze dosáhnout mírným zdvihem širšího pásma. Pro zachování věrohodnosti a přirozenosti by zesílení nemělo přesáhnout 6 dB.

4.2 Procesory pro úpravu dynamiky signálu

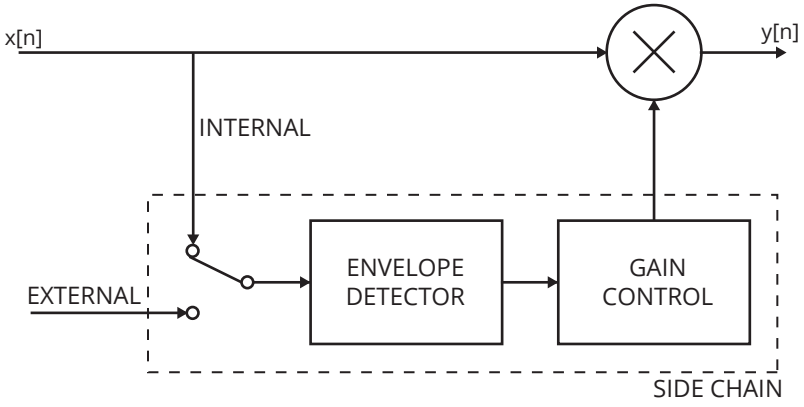
Dynamiku lze definovat jako rozdíl mezi nejtišším a nejhlasitějším zvukem v průběhu skladby, případně i několika skladeb, ať už se jedná o zvukovou stopu nástroje, hlasu nebo kompletní nahrávky. Jak již bylo zmíněno v kapitole 2.1.2, lidské ucho slyší zvuky od prahu slyšitelnosti až po práh bolesti, tedy je schopno rozlišit zvuky v dynamickém rozsahu až 130 dB. Při zpracování zvuku je dynamika vymezena rozsahem systému. Spodní hranice je určena odstupem užitečného signálu od šumu, naopak horní hranice je limitována maximální úrovní signálu, kterou dokáže systém zpracovat [18].

Použití procesorů k úpravě dynamiky signálu závisí především na hudebním žánru nahrávky. Zatímco u vážné hudby by mělo být záměrem co nejvěrněji zachovat dynamiku jednotlivých nástrojů tak, jak je slyšíme v koncertní síni, v populární hudbě, založené převážně na rytmickém základu, jsme často nuceni výrazně upravit dynamický rozsah zpěvu či akustických nástrojů. V současné populární hudbě se dynamiky nahrávky dosahuje vesměs střídáním jednotlivých nástrojů než změnami jejich poměru v mixu. Tiché pasáže vokálních stop by bez použití dynamických procesorů mohly v nahrávce zanikat za zvukově agresivnějšími nástroji, jakými jsou například bicí, naopak hlasité pasáže by z mixu mohly nepřirozeně vyčnívat.

Snížení dynamiky nahrávky najde své opodstatnění, pokud je skladba poslouchána v hlučném prostředí, například v pracovní dílně či za jízdy v automobilu, kde tišší pasáže skladby mohou být maskovány okolním hlukem. Tento fakt je jedním z důvodů, proč rozhlasové stanice snižují dynamiku vysílaných skladeb. Dalším důvodem využití komprese dynamiky je ochrana nahrávací, vysílací či ozvučovací aparatury proti přebuzení.

4.2.1 Kompresor

Kompresor je nástroj pro snížení dynamiky zvukového signálu, bez kterého se práce ve studiu dnes již prakticky neobejde. Kompresor je nelineární systém, tudíž může ovlivnit frekvenční spektrum zvuku. Základem všech kompresorů je obvod *side chain*, který kontroluje úroveň vstupního signálu a dle nastavení parametrů omezuje úroveň na výstupu. Jak je vidět na obrázku 4.10, může tento obvod pracovat jak s interním signálem, tak se signálem externím, čehož se využívá například v mixu k útlumu basové linky při úderu velkého bubnu či v rozhlasových stanicích, kdy je úroveň hudby automaticky snížena, pokud moderátor promluví.



Obrázek 4.10: Blokové schéma kompresoru podle [17].

Řídicí parametry kompresoru lze rozdělit do dvou kategorií. První kategorii tvoří parametry *úrovňové* [7], mezi které řadíme *threshold*, *ratio* a *make up*. Parametr *make up* slouží pro zesílení výstupního signálu, jenž prošel kompresí. Po překročení citlivostního prahu *threshold* T je vstupní signál přesahující práh zeslaben v poměru *ratio* R, viz rovnice:

$$y_G = \begin{cases} x_G & x_G \leq T \\ T + (x_G - T)/R & x_G > T \end{cases}. \tag{4.3}$$

Některé typy kompresorů umožňují nastavit parametr *knee* [7]. Kromě klasického ostrého přechodu *hard knee* lze zpravidla nastavit i přechod zaoblený neboli *soft knee*, viz obrázek 4.11. Použitím zaobleného přechodu můžeme dosáhnout měkkého a příjemného zvuku, cehož lze využít například při zvučení vokálních stop. Naopak ostrý přechod se všeobecně používá při kompresi bicích. Při použití přechodu *soft knee* o šírce přechodu W lze výstupní hodnotu signálu odvodit z rovnice:

$$y_G = \begin{cases} x_G & 2(x_G - T) < -W \\ x_G + (1/R - 1)(x_G - T + W/2)^2/(2W) & 2|(x_G - T)| \leq W \\ {}_T + (x_G - T)/R & 2(x_G - T) > W \end{cases}. \tag{4.4}$$

Druhou kategorií jsou parametry *časové* [17], mezi něž lze zařadit *attack* a *release*. *Attack* je doba náběhu neboli zpoždění, po kterém kompresor začne reagovat na překročení prahu. Analogicky k tomu doba doběhu *release* je čas, který kompresor potřebuje pro návrat do normálního stavu poté, co úroveň signálu klesne opět pod práh. Doba náběhu *attack* a doba doběhu *release* bývají obvykle implementovány pomocí IIR filtrů prvního řádu, které simulují analogový obvod detektoru složeného z rezistoru a kondenzátoru. Chování tohoto systému je popsáno vztahem:

$$y_L[n] = \begin{cases} \alpha_A y_L[n-1] + (1 - \alpha_A) x_L[n] & x_L[n] > y_L[n-1] \\ \alpha_R y_L[n-1] + (1 - \alpha_R) x_L[n] & y_L[n] \leq y_L[n-1] \end{cases}. \tag{4.5}$$

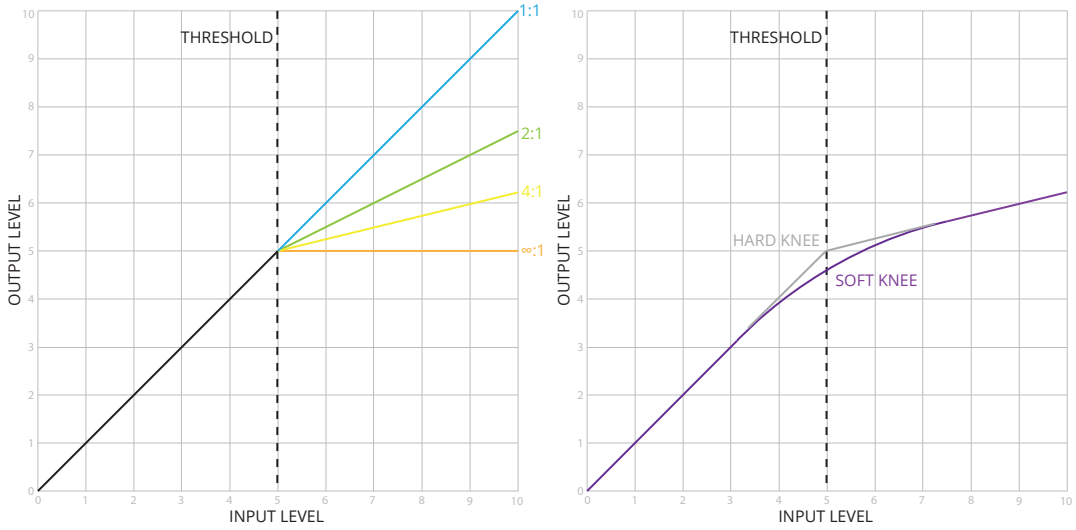
Koeficienty filtrů [7] lze vypočítat dle rovnice (4.8), kde τ je doba náběhu či doba doběhu. Některé modely disponují i takzvaným časem držení *hold time*, který zabraňuje přejít kompresoru do fáze dojezdu před uplynutím nastavené doby. Z hlediska detekce vstupní hlasitosti mohou kompresory reagovat buďto na špičkovou úroveň signálu (peak) nebo na hodnotu *RMS* (Root Mean Square). Detekce RMS přináší hladší a jemnější průběh komprese a lze vypočítat dle vztahu:

$$y_L^2[n] = \frac{1}{M} \sum_{m=-M/2}^{M/2-1} x_L^2[m - m]. \tag{4.6}$$

Tento výpočet ovšem není zcela ideální pro real-time implementaci z důvodu vynucené latence $M/2$ vzorků, pročež se využívá aproximace pomocí IIR filtru prvního řádu charakterizovaného diferenční rovnicí:

$$y_L^2[n] = \alpha y_L^2[n-1] + (1 - \alpha) x_L^2[n], \tag{4.7}$$

$$\alpha = e^{-1/(\tau f_s)}. \tag{4.8}$$



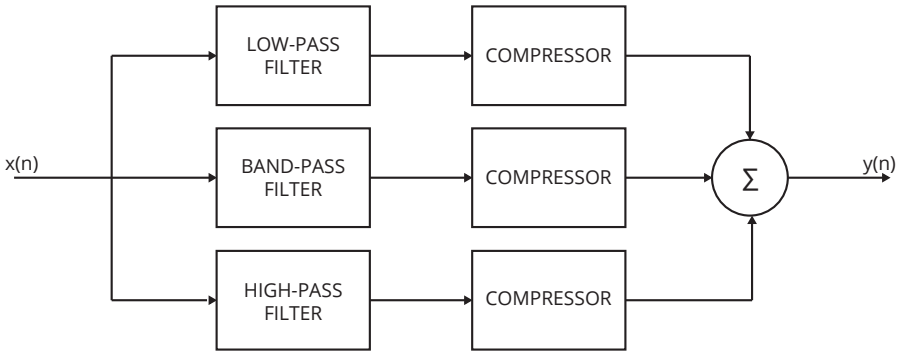
Obrázek 4.11: Křivky komprese s různým nastavením parametrů ratio a knee [11].

4.2.2 Limiter

Specifickým typem kompresorů jsou *limitery*, které se od klasických kompresorů liší extrémním nastavením parametru *ratio*. Maximální hodnota tohoto nastavení je teoreticky ∞:1, v praxi však jako limitery označujeme kompresory s kompresním poměrem alespoň 10:1. Úkolem limiteru je omezit maximální úroveň signálu určenou nastavením parametru *threshold*. Limitery bývají často součástí zvukových zařízení, kde omezují vstupní signál a zabraňují tak přebuzení.

4.2.3 Vícepásmový kompresor

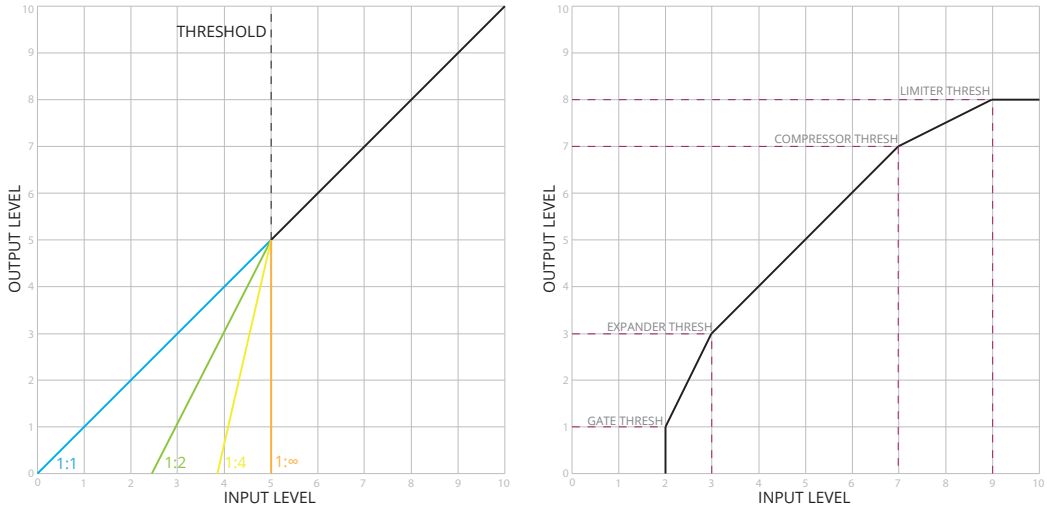
Občas se můžeme setkat se situací, kdy použití klasického kompresoru vede k otupení zvuku. Tomuto stavu se lze vyhnout pomocí mechanismů *vícepásmového kompresoru*, který rozděluje signál do několika frekvenčních pásem a každému z nich poskytuje plně nastavitelný samostatný kompresor, viz obrázek 4.12. Vícepásmové kompresory se využívají zejména při masteringu, kdy lze jejich aplikací dosáhnout zvýšení celkové hlasitosti o 3 až 6 dB bez výskytu nežádoucích zvukových artefaktů. Podmnožinou vícepásmových kompresorů jsou takzvané *de-essery*, které potlačují úzké vysokofrekvenční pásmo pro eliminaci sykavek.



Obrázek 4.12: Blokové schéma vícepásmového kompresoru dle [3].

4.2.4 Expander a noise gate

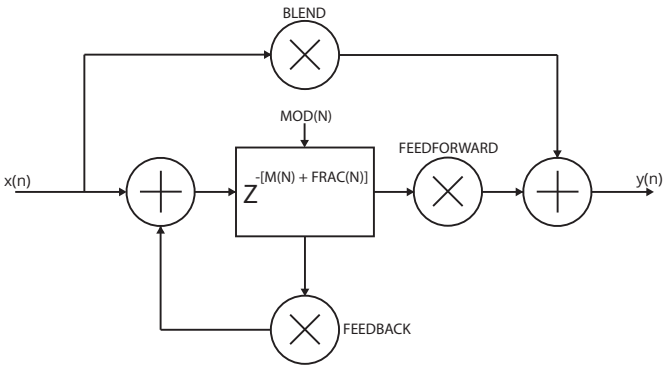
Expander je dynamický procesor, který funguje na opačném principu než klasický kompresor. Jeho úkolem je v daném poměru zeslabit signál, který nedosáhl citlivostní úrovně prahu. Podobně jako kompresor disponuje i expander parametry *attack* a *release* pro řízení doby náběhu a doběhu. Expandery se hojně využívají při mixu živě nahraných bicích. Bicí souprava bývá nahrávána více mikrofony zároveň, přičemž jednotlivé stopy obsahují přeslechy z ostatních mikrofonů, které lze potlačit právě použitím expanderu. Expanzní poměr je udáván opačně než poměr kompresní. Nastavením expanzního poměru na maximum, teoreticky až 1:∞, získáme dynamický procesor zvaný *noise gate*.



Obrázek 4.13: Různá nastavení poměru expanderu a křivka dynamického multiefektu.

4.3 Efekty využívající zpožďovací linku

Oproti zvukovým procesorům, které většinou vychází z analogového základu, prakticky všechny zvukové efekty dnes pracují na principu digitálního zpracování signálu. Nejjednodušší efekty lze realizovat pomocí zpožďovací linky, jejíž obecné blokové schéma je zobrazeno na obrázku 4.14.



Obrázek 4.14: Obecné blokové schéma zpožďovací linky dle [20].

Jedná se o systém s proměnlivou délkou zpoždění, zpětnou vazbou a přemostěním. Čím vyšší je hodnota parametru *feedback*, tím dochází k více opakování a tedy i delšímu času doznívání. Parametr *blend* ovlivňuje poměr mezi vstupním a zpožděným signálem. Úroveň zpožděného signálu lze ovlivnit parametrem *feedforward*. Dle nastavení jednotlivých parametrů zpožďovací linky a volby typu a hloubky modulace zpoždění můžeme rozlišit efekty *delay*, *flanger*, *vibrato* a *chorus* [10].

Efekt	*Blend*	*Feedforward*	*Feedback*	*Zpoždění [ms]*	*Hloubka modulace [ms]*	*Typ modulace*
Delay	0	1	volitelné	> 25	0	žádná
Flanger	0.7	0.7	0.7	0-15	0-2	0.1-1 Hz sinus
Vibrato	0	1	0	0	0-3	0.1-5 Hz sinus
Chorus	0.7	1	-0.7	1-30	1-30	náhodná

Tabulka 4.1: Průmyslové standardy jednotlivých efektů [10].

4.3.1 Delay

Delay je základní efekt založený na principu zpožďovací linky, který na výstupu kombinuje vstupní signál se signálem zpožděným. Zpoždění signálu je statické a mělo by být minimálně 25 ms. V opačném případě nelze od sebe vstupní a zpožděný signál rozlišit. Některé delay také umožňují synchronizovat délku zpoždění s tempem skladby a následné přepínání v hudebních poměrech reprezentujících notové délky. Přepočet lze uskutečnit dle rovnice:

$$t = \frac{60 \cdot T}{BPM}, \tag{4.9}$$

kde T je počet celých dob na takt, t je doba taktu v sekundách a BPM tempo neboli počet úderů za minutu. Delay se převážně využívá pro vokální stopy a elektrické kytary. Moderní delay disponují filtry pro frekvenční úpravu zpožděného signálu. *Stereo delay* umožňuje oproti *mono delay* nastavení parametrů obou kanálů samostatně.

4.3.2 Flanger

Efekt *flanger* získáme použitím krátkého zpoždění, jehož délka je modulována sinusovým signálem. Dle průmyslových standardů je délka zpoždění od 1 do 15 ms, a modulační frekvence od 0.1 do 1 Hz. Flanger využívá zavedení poměrně intenzivní zpětné vazby, čímž přispívá k působivým změnám barvy zvuku. Systém se chová jako hřebenový filtr, tedy některé frekvence potlačuje a jiné zvýrazňuje. Flanger se uplatňuje především pro zvučení elektrických kytar a klávesových nástrojů.

4.3.3 Vibrato

Vibrato je velmi rozšířený zvukový efekt, který používá zpožděný signál bez kombinace se signálem přímým. Využívá rychlejší modulaci délky zpoždění signálu než flanger, což vede k jemným změnám ladění nástroje či hlasu. Pojem vibrato bývá někdy nesprávně zaměňován za pojem *tremolo*. Tremolo je efekt, který cyklicky mění amplitudu hraného tónu, nikoliv však jeho výšku. S aplikací vibrata se nejčastěji setkáme u strunných nástrojů.

4.3.4 Chorus

Zvukový efekt *chorus* využívá zpoždění signálu do 30 ms s mírnou modulací délky zpoždění, čímž vzniká iluze více hlasů hrajících zároveň s mírnými rozdíly v ladění i časovém průběhu. Oproti předchozím efektům využívá vícenásobnou zpožďovací linku, tudíž k originálnímu signálu přimíchává více jeho upravených kopií. Chorus se často využívá pro zjemnění zvuku elektrických kytar či oživení barvy zvuku syntetických nástrojů.

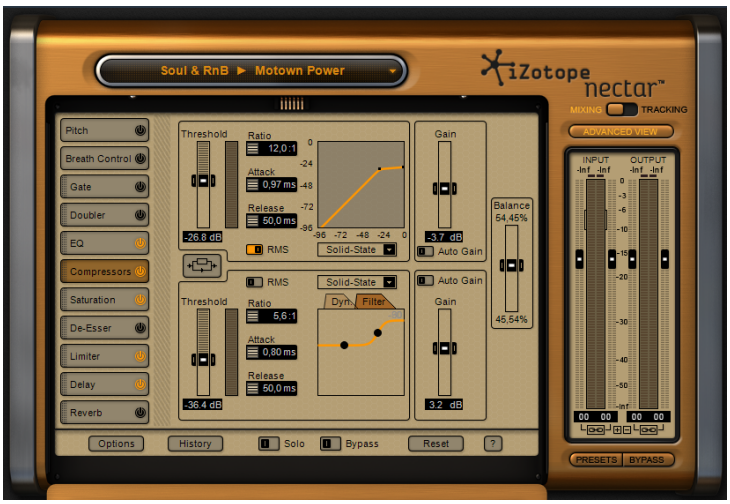
4.4 Dozvukové efekty

Tato kategorie efektů vytváří umělý dozvuk napodobující průběh dozvuku v reálném prostředí. Výhodou digitálních dozvukových efektů je jejich nízká cena a velmi vysoká věrohodnost, nevýhodou pak výrazně složitější návrh a realizace. Základním parametrem je čas dozvuku *decay time*, což je doba za kterou poklesne dozvuk o 60 dB. Parametr *pre-delay* je doba, během níž dorazí první odraz k místu poslechu.

Odlišné materiály pohlcují a odráží různé frekvence signálu. Z tohoto důvodu bývají tyto efekty vybaveny ovládáním času dozvuku pro více frekvenčních pásem. Většina modelů umožňuje nastavení velikosti místnosti a volbu konkrétního algoritmu pro jednotlivá prostředí. Mezi nejčastěji používaná prostředí, se kterými se můžeme setkat, patří *hall*, *room*, *chamber* či *church*.

4.5 Multiefekty

Zvukovým multiefektem se myslí systém složený z více modulů pro zpracování zvukových signálů. Mezi špičku multiefektů určených pro postprodukci vokálních stop neodmyslitelně patří komerční řešení *Nectar* od přední vývojářské firmy *iZotope*[1]. Plugin Nectar obsahuje ekvalizér, gate, kompresor, limiter, de-esser, pitch-correction, delay a reverb. V rukou profesionála se může stát mocným nástrojem, který sjednocuje vše potřebné na jednom místě. Pro nezkušeného uživatele však může působit značně nepřehledně. Další nevýhodou je poměrně vysoká cena produktu.



Obrázek 4.15: iZotope Nectar.

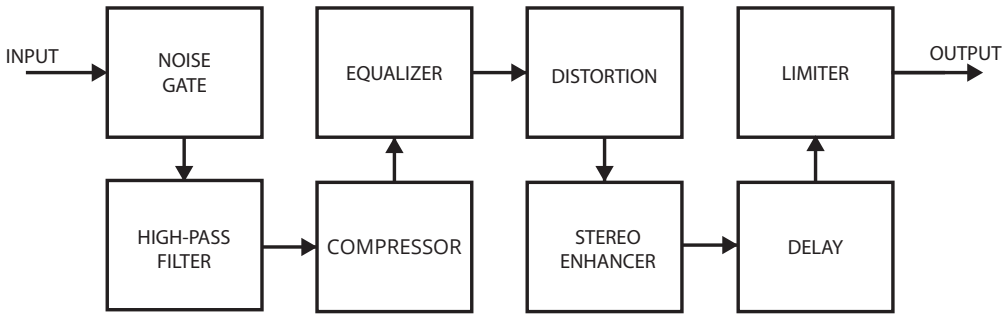
[1] Webové stránky firmy iZotope: http://www.izotope.com

Kapitola 5

Návrh aplikace

V předchozí kapitole byly představeny základní efekty a procesory pro úpravu zvukového signálu, ze kterých jsem vybral a sestavil řetězec vhodný pro zpracování vokálních stop, jenž je zobrazený na blokovém schématu 5.1. Prvním procesorem je *noise gate*, který nalezne uplatnění především pro potlačení přeslechů a jiných rušivých zvuků, přičemž je však nutné opatrně volit parametr *threshold* tak, aby nedošlo k utlumení počátečních či koncových hlásek. *Horní propust* slouží k eliminaci neužitečných nízkých kmitočtů signálu, jenž by se mohly negativně projevit při pozdějším zpracování.

Kompresor je důležitým nástrojem, který snížením dynamiky zvukové stopy vyrovnává úrovňové poměry jednotlivých hlasových úseků tak, aby nezanikaly či naopak příliš nevynikaly v mixu. *Ekvalizér* lze využít pro eliminaci rušivých frekvencí či dobarvení a projasnění vokálních stop. Zajímavých výsledků lze dosáhnout použitím jednotky *zkreslení*. Pro vytvoření či zdůraznění stereo obrazu zvukové stopy se využívá efekt *stereo enhancer*. Aplikací zvukových efektů založených na zpožďovací lince lze obohatit atmosféru zejména refrénových úseků nahrávky. Poslední procesor *limiter* omezuje úroveň výstupního signálu.



Obrázek 5.1: Řetězec zvukových procesorů a efektů.

Převážně méně zkušení uživatelé se mohou ztrácet ve vysokém počtu ovládacích prvků jednotlivých modulů. Cílem této práce je vytvoření zvukového multiefektu, který právě pomocí minimálního počtu ovládacích prvků a zabudované databáze přednastavených presetů výrazně urychlí proces postprodukce vokálních stop. Tyto presety by měly pokrývat celou škálu hlasových projevů. V rámci seznámení s digitálním zpracováním signálu jsem vytvořil off-line implementaci modulů pomocí programového prostředí MATLAB. Výsledná aplikace *Executor* je implementována v programovacím jazyce C++ s využitím technologie VST, která umožňuje zpracování zvuku v reálném čase.

Kapitola 6

Off-line implementace a testování

Jako off-line implementaci můžeme považovat zpracování zvuku, při kterém je celá zvuková stopa načtena a uložena do paměti. Dále je vybrán požadovaný algoritmus, nastaveny jeho parametry a spuštěno zpracování. Upravený zvuk lze přehrát až po dokončení výpočtů, čímž se off-line implementace zásadně liší od real-time implementace, při které je možné poslouchat zvuk a přenastavovat parametry i během zpracování signálu. V této kapitole se věnuji zpracování vokálních stop pomocí aplikace MATLAB. Při implementaci off-line algoritmů a testování jejich funkčnosti jsem využil publikací [12], [7] a [20].

6.1 MATLAB

MATLAB (MATrix LABoratory) je vysokoúrovňový multiplatformní jazyk a interaktivní programové prostředí pro numerické výpočty, vizualizaci, simulace a programování, vyvíjené firmou MathWorks[1]. Použitím MATLABu můžeme analyzovat data, vyvíjet algoritmy a vytvářet modely a aplikace vhodné pro úpravu digitálního signálu. MATLAB má slabou dynamickou typovou kontrolu, podporuje objektově orientované programování a disponuje celou řadou rozšíření v podobě knihoven.

6.2 Vstupní data

Pro nastavení a testování off-line i real-time implementace jsem sestavil soubor šedesáti pěti vokálních stop. Tyto stopy byly nahrány v několika různých nahrávacích studiích přes různé nahrávací řetězce. Najdou se mezi nimi nahrávky jak z domácích či poloprofesionálních studií, tak i ze studií, které patří mezi profesionální špičku v Čechách a na Slovensku. Jako příklad mikrofonů, které byly využity při nahrávání, můžu úvest kondenzátorové velkomembránové mikrofony Neumann TLM 102 a AKG C 214.

	Mužský	**Ženský**	**Celkem**
Zpěv	3	14	17
Rap	47	1	48
Celkem	50	15	65

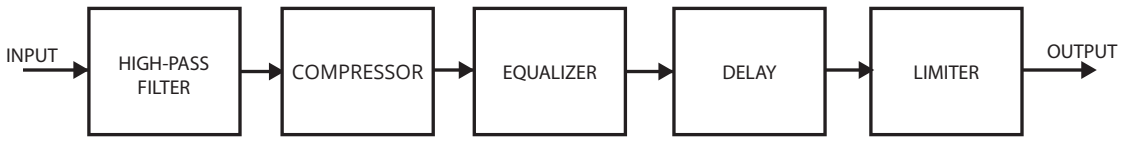
Tabulka 6.1: Zastoupení hlasových projevů v souboru vokálních stop.

[1] Webové stránky společnosti MathWorks jsou dostupné z: http://www.mathworks.com

Soubor vokálních stop jsem sestavil tak, aby v něm bylo zastoupeno co nejvíce různých typů hlasových projevů, viz tabulka 6.1. Průměrná délka zvukové stopy je 33 s, minimální 9 s a maximální 3 min a 22 s. Zvukové stopy jsou uloženy ve formátu *wav* s bitovou hloubkou 16 *bitů* a vzorkovací frekvencí 44100 Hz.

6.3 Implementace

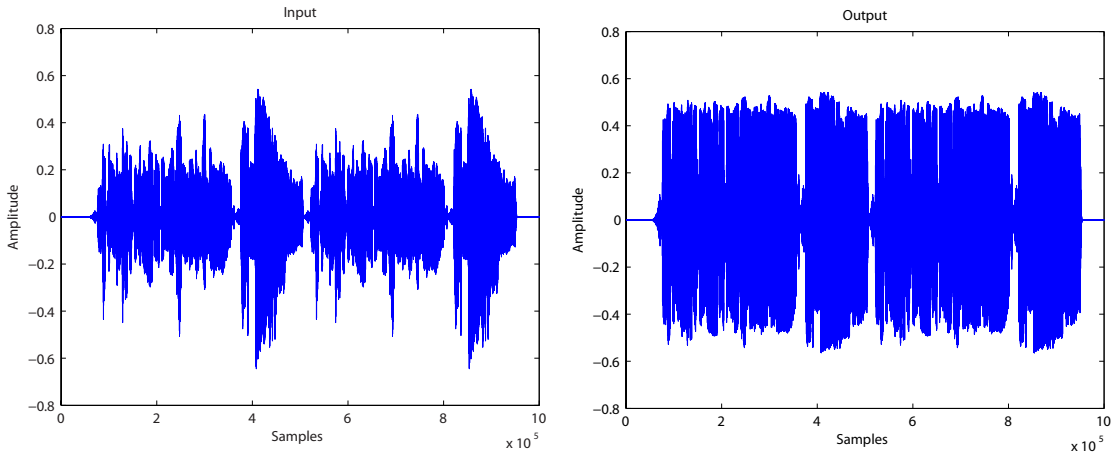
Soubory jsou načítány funkcí `wavread`, která vrací pole vstupního signálu, vzorkovací frekvenci `Fs` a bitovou hloubku `bits`. Pro přehrání souboru lze využít funkci `sound`, pro vizualizaci pak funkci `plot`. Výstupní signál lze uložit do souboru pomocí funkce `wavwrite`. V rámci off-line implementace byl vytvořen řetězec zvukových procesorů a efektů zobrazený na obrázku 6.1. Zde off-line implementace skončila a dále se pokračovalo s real-time verzí.



Obrázek 6.1: Blokové schéma off-line řetězce zvukových procesorů a efektů.

Dynamické procesory

Pro snížení dynamického rozsahu signálu byla implementována třída `Compressor`, jejímiž parametry jsou hlasitostní práh `thresh`, poměr komprese `ratio`, doba náběhu `att` a doba doběhu `rel`. Komprese signálu je řízena metodami `Gain` a `Detector`. Metoda `Detector` reprezentuje IIR filtr prvního řádu, jehož koeficienty jsou vypočítány dle rovnice (4.8). Jelikož se jedná o off-line implementaci, lze výstupní signál, jenž prošel kompresí, normalizovat na maximální úroveň vstupního signálu, viz obrázek 6.2. Obdobou třídy `Compressor` je třída `Limiter`, která omezuje úroveň signálu na hodnotu parametru `thresh`, přičemž ostatní parametry jsou fixně nastaveny – kompresní poměr 100:1, doba náběhu 0 ms a doba doběhu 10 ms.



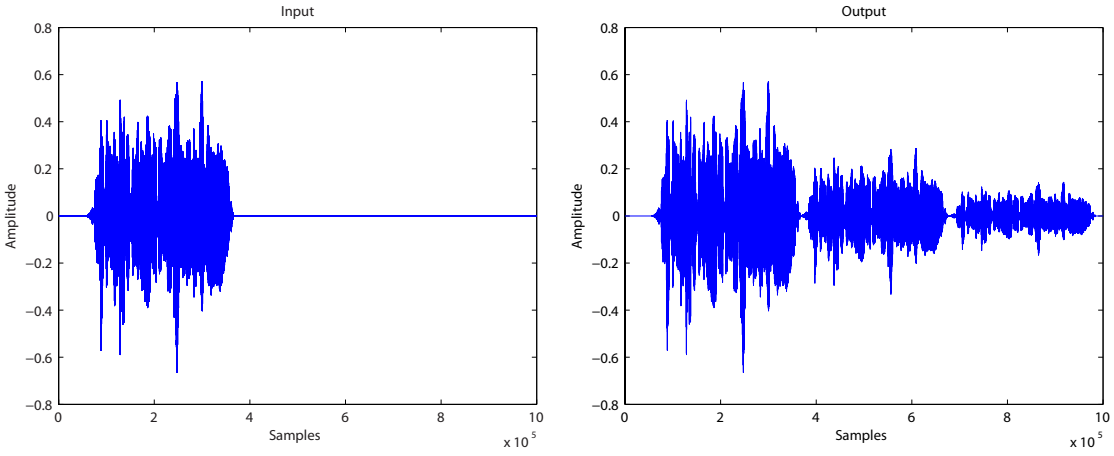
Obrázek 6.2: Vizualizace použití silné komprese signálu.

Ekvalizér

Ekvalizér je realizovaný pomocí IIR filtrů druhého řádu zapojených do kaskády. Jednotlivé filtry jsou reprezentovány třídami `Biquad` a jejich koeficienty lze vypočítat dle rovnic uvedených v dodatku B. Parametry filtrů jsou typ filtru `type`, centrální frekvence `Fc`, šířka pásma `Q` a zesílení pásma `peakGain`. Centrální kmitočet může být nastaven od 20 Hz do 20 kHz, šířka pásma od 0.1 do 10 a zesílení pásma od −30 do 30 dB. Typy filtrů lze nastavit na *low-pass*, *high-pass*, *band-pass*, *notch*, *peak*, *low-shelf* či *high-shelf*, viz příloha B. Třída `Biquad` je využita i pro realizaci bloku `high-pass filter`.

Zpožďovací efekt

Třída `Delay` umožňuje sečíst vstupní signál s jeho zpožděnými kopiemi, čehož lze využít například pro obohacení refrénových pasáží. Parametry třídy `Delay` jsou zpoždění signálu `time`, počet opakování `tap` a velikost zpětné vazby `feed`. Zpoždění signálu lze nastavevním příznaku `sync` synchronizovat s tempem skladby dle rovnice (7.1).



Obrázek 6.3: Ukázka použití zpožďovacího efektu.

6.4 Testování a vyhodnocení

Kromě zmíněného souboru vokálních stop byly při testování využity i jednoduché periodické a neperiodické signály, včetně bílého a růžového šumu. Jednotlivé moduly byly testovány samostatně s různým nastavením parametrů i v zapojení s ostatními moduly. Zvukový výstup jsem poslouchal prostřednictvím akitvních studiových monitorů *KRK Rokit 6 G2* a sluchátek *AKG K270 Studio* a *Beats Studio*.

Pomocí programového prostředí *MATLAB* lze oproti klasickým programovacím jazykům poměrně snadno a rychle implementovat kvalitní algoritmy pro off-line zpracování zvukového signálu. Velkou výhodou je možnost využití rozsáhlé databáze knihoven a nástrojů pro poslech a vizualizaci zvuku. Během poslechu však není možné operativně upravovat nastavení parametrů jednotlivých modulů, pročež se použití tohoto systému stává velmi zdlouhavým a nepohodlným.

Kapitola 7

Real-time implementace

Tato kapitola se zabývá návrhem a realizací real-time implementace VST modulu, určeného pro úpravu vokálních stop. Jako real-time aplikaci můžeme označit program, který provádí výpočtové algoritmy během poslechu, tudíž nepracuje s celou zvukovou stopou naráz, ale postupně s jejími navazujícími segmenty. Klíčovou vlastností takového systému je jeho zpoždění neboli doba, za kterou systém zpracuje jeden segment. Zpoždění závisí nejen na rychlosti procesoru či velikosti paměti, ale také na typu a nastavení ovladačů zvukové karty a operačním systému [5].

7.1 Virtual Studio Technology

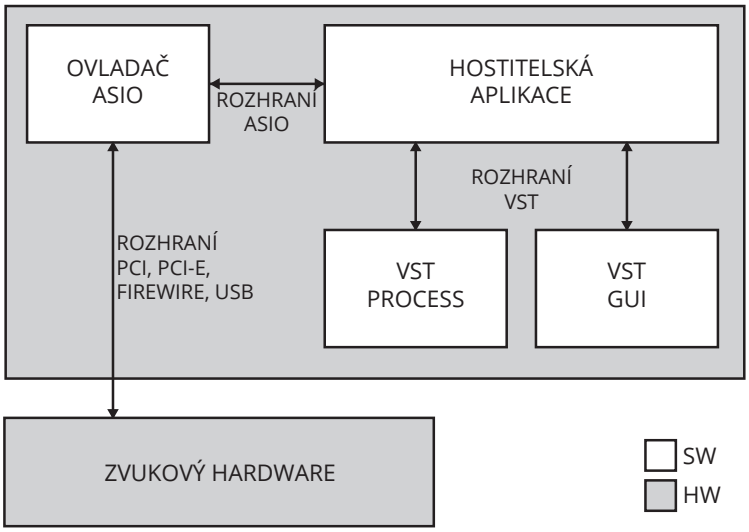
Virtual Studio Technology, neboli zkráceně *VST*, je standard vyvíjený německou firmou *Steinberg Media Technologies*[1] pro vytváření zásuvných audio modulů do softwarových nahrávacích a produkčních systémů. Na trh byl uveden v roce 1996 současně s vydáním hostileské aplikace *Steinberg Cubase 3.02*, která obsahovala první VST pluginy Espacial, Chorus, Stereo Echo a Auto-Panner. V následujících letech se technologie VST masivně rozšířila, o čemž svědčí i fakt, že největší konkurenční firma *Desidesign*[2] umožnila její použití ve svých systémech *Protools*, které do té doby používali rozhraní *RTAS*. V dnešní době najdeme podporu VST pluginů teměř ve všech produkčních systémech, včetně programů Logic Pro, Acid a FL Studio. Mezi firmy zabývající se jejich vývojem neodmyslitelně patří Waves, Native Instruments a iZotope, ale najdeme mezi nimi i hardwarové výrobce jako Solid State Logic a Korg.

Obrázek 7.1: Logo Steinberg VST.

VST modul zpracovává tok audio dat po blocích, tzv. bufferech, které jsou přidělovány hostitelskou aplikací. Velikost bufferů je závislá na nastavení ovladačů zvukového rozhraní *ASIO* (Audio Streaming Input/Output). ASIO je standard rovněž vyvíjený fir-

[1] Webové stránky firmy Steinberg: http://www.steinberg.net
[2] Webové stránky firmy Digidesign: http://www.digidesign.com

mou Steinberg, a jeho cílem je minimalizovat zpoždění signálu mezi aplikací a zvukovým rozhraním. Zpravidla jsou ovladače ASIO dodávány výrobcem spolu se zvukovou kartou a podobně jako technologie VST jsou dnes podporovány většinou produkčních programů. Obecně platí, že čím větší je velikost bufferu, tím delší je čas výpočtů a tedy i zpoždění programu a doba odezvy systému, neboli latence. Z tohoto důvodu je nutné věnovat pozornost vhodnému nastavení ovladačů, aby systém stíhal všechny výpočty a nedocházelo tak ke zvukovým artefaktům.



Obrázek 7.2: Schéma systému pracujícího s ASIO a VST [10].

Hostitelská aplikace vidí VST plugin jako černou skříňku s určitým počtem parametrů, na jejíž vstupy dodává audio data a poté, co jsou pluginem zpracována, je z výstupů odebírá. Existují celkem tři různé typy pluginů. První kategorii tvoří takzvané *VST nástroje*, které pracují jako virtuální syntezátory či samplery, jejichž úkolem je generovat zvuk. Druhou a velice rozšířenou kategorií jsou *VST efekty* sloužící ke zpracování a modifikaci vstupního signálu. Do této kategorie lze zařadit ekvalizéry, kompresory, ale i monitorovací efekty určené ke spektrální analýze. Třetí a poslední kategorií jsou *VST MIDI efekty*, které na vstup dostávají data ve formátu MIDI, modifikují je a směrují je na další VST nástroj, případně hardwarové zařízení [10].

MIDI (Musical Instrument Digital Interface) je mezinárodně standardizovaný komunikační protokol určený pro komunikaci zařízení převážně hudebního charakteru v reálném čase. Kromě samotných hudebních nástrojů využívají tuto technologii sekvencery, kontroléry, bicí moduly i pódiová osvětlovací zařízení. Samotná MIDI informace neobsahuje žádný zvuk, ale pouze řídicí data, dle kterých generátor zvuk posléze vytvoří. MIDI signál obsahuje informace pouze o výšce a délce tónu a síle úhozu, díky čemuž lze například pomocí kláves zahrát jednoduše tóny smyčců.

7.1.1 Software Development Kit

Společnost Steinberg v nedávné době zveřejnila na svém webu VST SDK verze 3.0, které se od stávající verze 2.4 značně odlišuje. Z důvodu vysoké podpory a rozšířenosti mezi hostitelskými aplikacemi jsem použil starší verzi 2.4, která je zpětně kompatibilní s verzí 1.0.

VST SDK je implementováno v programovacím jazyce C++. Existují však i varianty v jazyce Java či Delphi. Základem SDK je třída `AudioEffectX`, která je potomkem rodičovské třídy `AudioEffect`. Základní strukturu VST pluginu získáme vytvořením vlastní třídy odvozené ze třídy `AudioEffectX`. V následující tabulce jsou uvedeny metody, které slouží k identifikaci pluginu v rámci hostitelské aplikace [1].

`getEffectName`	Vrací řetězec s názvem efektu
`getProductString`	Vrací řetězec s názvem produktu
`getVendorString`	Vrací řetězec s názvem výrobce
`getVendorVersion`	Vrací verzi definovanou výrobcem
`getPlugCategory`	Specifikuje kategorii pluginu

Chování pluginu lze ovlivňovat pomocí předem definovaného počtu parametrů. Parametry jsou reprezentovány proměnnými datového typu *float*, které nabývají hodnot od 0.0 do 1.0. Pokud plugin nedisponuje vlastním grafickým uživatelským rozhraním, hostitelská aplikace automaticky vygeneruje okno a pro každý parametr ovládací prvek. Pro správu parametrů je k dispozici tato sada metod:

`setParameter`	Nastavuje hodnotu parametru
`getParameter`	Vrací hodnotu parametru
`getParameterDisplay`	Definuje způsob zobrazení parametru
`getParameterLabel`	Vrací jednotku parametru
`getParameterName`	Vrací název parametru

Za zmínku stojí především metoda `getParameterDisplay`, která určuje způsob, jakým se hodnota parametru zobrazí. Pro zjednodušení práce lze využít funkcí `float2string` a `dB2string`. Pro převod počtu vzorků na milisekundy či hertze jsou k dispozici funkce `ms2string` a `Hz2string`, které získávají aktuální vzorkovací frekvenci pomocí volání funkce `getSampleRate`. Soubor hodnot všech parametrů pluginu lze označit jako takzvaný *program*. Programy umožňují uživateli využívat, editovat či vytvářet vlastní *presety* [10].

`setProgram`	Slouží k nastavení parametrů pluginu dle programu
`setProgramName`	Umožňuje změnit jméno programu pomocí hostitele
`getProgramName`	Vrací jméno programu

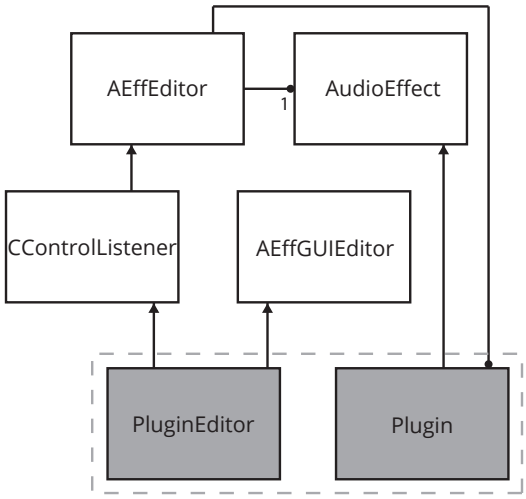
Hostitelská aplikace opakovaně volá metodu pluginu `processReplacing`, které předává ukazatel na pole se vzorky signálu. Tato metoda nahrazuje vstupní vzorky signálu za data získaná aplikací procesních algoritmů, čímž se zásadně liší od původní metody `process`, která výsledek výpočtu sčítala se vstupním signálem. Ve verzi SDK 2.4 je doporučeno metodu `process` nepoužívat a ve verzi 3.0 je již zcela zakázána. Vzorky signálu jsou uloženy jako proměnné datového typu *float*, nabývající hodnot od -1.0 do 1.0. Pro vyšší přesnost výpočtů lze využít metodu `processDoubleReplacing` pracující s datovým typem *double*.

`process`	Původní metoda pro úpravu signálu
`processReplacing`	Metoda pro úpravu signálu
`resume`	Opětovný start procesu
`suspend`	Pozastavení chodu procesu

Seznam všech metod VST SDK s podrobným popisem lze nalézt v [1].

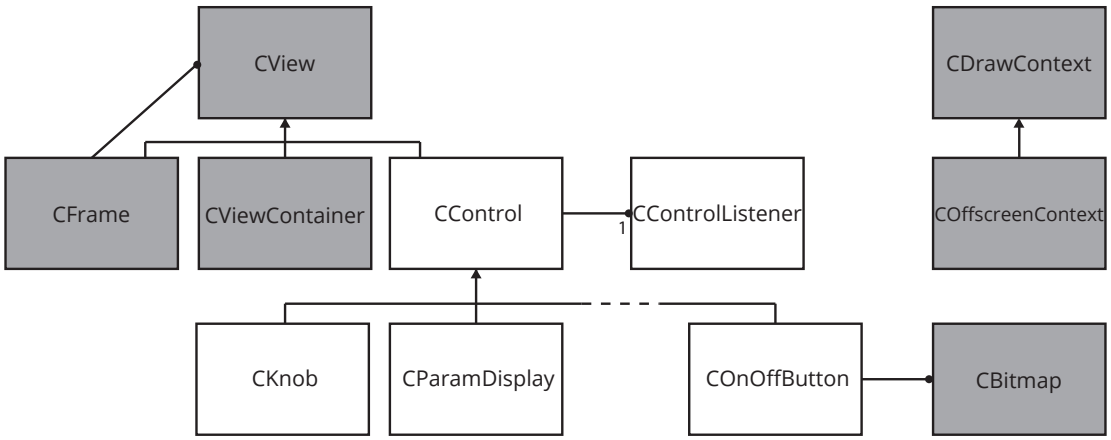
7.1.2 Graphic User Interface

Jak bylo naznačeno v předcházející kapitole, v případě, kdy nemá plugin implementováno grafické uživatelské rozhraní, vygeneruje hostitelská aplikace okno a pro každý parametr pluginu ovládací prvek i s popisky. Vzhled pluginu pak tedy čistě závisí na použitém hudebním softwaru. Například výchozí ovládací prvky aplikace Steinberg Cubase jsou realizovány pomocí posuvných faderů, oproti tomu aplikace FL Studio je zobrazuje v podobě otočných knobů.



Obrázek 7.3: Diagram zobrazuje vztahy mezi třídami pluginu a editoru [1].

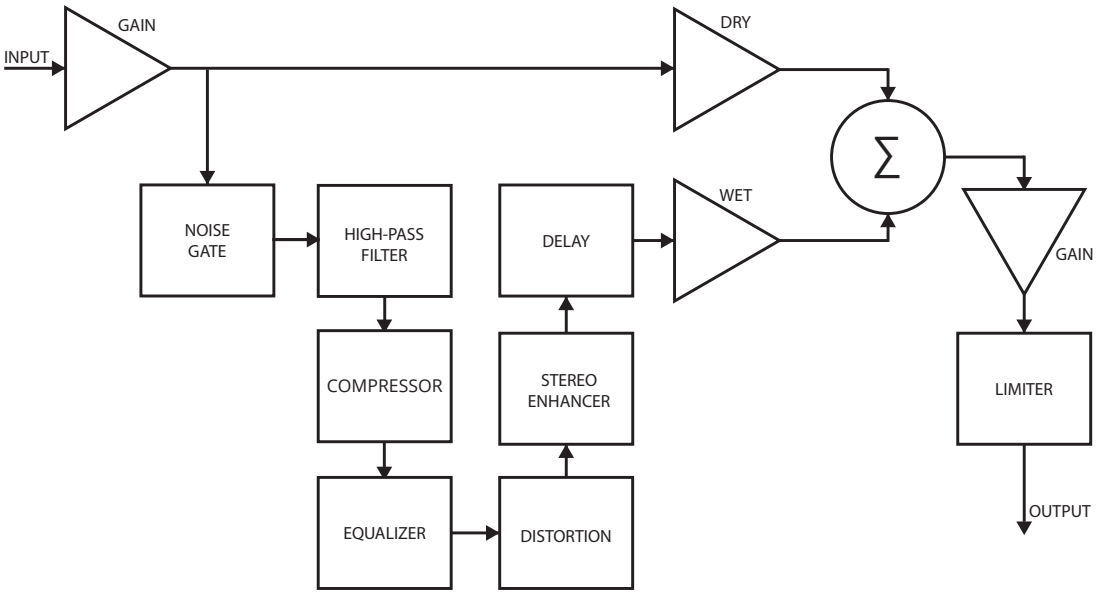
Rozšířením základní sady VST SDK je knihovna pro tvorbu grafických uživatelských rozhraní. Tato knihovna disponuje mechanismy pro vytvoření různých typů tlačítek, přepínačů, jezdců a prezentaci číselných hodnot či textu. Vzhled grafických elementů je uložen v podobě bitmapových souborů. GUI lze zavést vytvořením a definicí třídy `Editor`, jenž je odvozena z tříd `AEffGUIEditor` a `CControlListener`. Ovládací prvky rozhraní jsou potomky rodičovské třídy `CControl`, viz diagram 7.4.



Obrázek 7.4: Diagram tříd VST GUI [1].

7.2 Aplikace Executor

V rámci této bakalářské práce jsem navrhl a vyvinul aplikaci *Executor*. Tato aplikace je implementována v podobě VST pluginu, napsaného v programovacím jazyce C++ s využitím knihoven VST SDK verze 2.4 a VST GUI. K implementaci programu jsem zvolil vývojové prostředí Microsoft Visual Studio s kompilátorem MS Visual C++. Plugin se skládá z několika modulů, které jsou zobrazeny na blokovém schématu 7.5.



Obrázek 7.5: Blokové schéma pluginu Executor.

Struktura pluginu *Executor* je od základu koncipována s ohledem na znovupoužitelnost jednotlivých modulů i potenciální rozšíření stávající funkčnosti. Jednotlivé bloky lze libovolně kombinovat. Nejlepších zvukových výsledků však bylo dosaženo zapojením podle blokového schématu z obrázku 7.5. Poměr mezi přímým a upraveným signálem lze ovlivnit pomocí násobících bloků *dry* a *wet*. Signál můžeme zesílit na požadovanou hlasitostní úroveň pomocí bloků *gain* a to jak před aplikací výpočetních algoritmů, tak po ní. Podrobněji se jednotlivým blokům věnuji v následujících částích této kapitoly.

Studnicí znalostí pro mě během návrhu i realizace pluginu byly webové portály *Audiozone*[3], *KVR Audio*[4] a *Music-DSP*[5]. Tyto weby se zabývají jak postprodukčními praktikami a novinkami ze světa hudby, tak i samotnou implementací zvukových algoritmů. K vytvoření dynamických modulů jsem použil velmi kvalitně zpracovanou a zdokumentovanou knihovnu *Chunkware Simple Source*[6]. Při implementaci ekvalizéru jsem vycházel z knihovny *Biquad* a s ní spojené série článku webového serveru *EarLevel Engineering*[7]. Za velice užitečnou považuji knihu *The Audio Programming Book* [2], která seznamuje čtenáře se základními i pokročilejšími principy a konstrukcemi digitálního zpracování signálu v programovacím jazyce C++.

[3] Audiozone; Server o hudbě a zpracování zvuku: http://www.audiozone.cz
[4] KVR audio; Novinky a informace ohledně audio pluginu: http://www.kvraudio.com
[5] Music-DSP; Kolekce algoritmů shromážděných DSP komunitou: http://www.musicdsp.org
[6] Chunkware Simple Source lze volně stáhnout z: https://github.com/music-dsp-collection
[7] EarLevel Engineering; Practical digital signal processing: http://www.earlevel.com

Dynamické procesory

Základem dynamických procesorů je třída `AttRelEnvelope`, která zastupuje funkci obvodu sidechain, čili kontroluje a upravuje hlasitostní úroveň výstupního signálu. Parametry `attack` a `release` jsou realizovány pomocí IIR filtrů prvního řádu v podobě instancí třídy `EnvelopeDetector`, přičemž koeficienty filtrů lze vypočítat dle vztahu (4.8). Jednotlivé moduly dynamických procesorů, jejichž podrobná funkcionalita byla probrána v kapitole 4.2, jsou odvozeny právě od třídy `AttRelEnvelope`.

Pro kompresi zvuku jsou k dispozici třídy `SimpleComp` a `SimpleCompRMS`, které se od sebe liší způsobem detekce vstupní hlasitosti signálu. Pro RMS detekci je opět využito IIR filtru prvního řádu v podobě třídy `EnvelopeDetector`. Prahovou úroveň signálu a kompresní poměr lze nastavit pomocí metod `setThresh` a `setRatio`. Upravený signál je po kompresi zesílen podle nastavení parametru `cGain`. Třídy `SimpleGate` a `SimpleGateRMS` umožňují utlumit signál, který nedosáhl požadované hlasitostní úrovně prahu. Výstupní signál lze limitovat pomocí třídy `SimpleLimit`. Pro využití funkce `lool-ahead` musí limiter zavést počáteční zpoždění signálu, o němž informuje hostitelskou aplikaci voláním funkce `setInitialDelay`.

Ekvalizér

Třída `Equalizer` se skládá z šesti datových struktur `TChannel`. Struktura `TChannel` obsahuje dvě třídy `Biquad`, které reprezentují IIR filtry druhého řádu, jenž jsou popsány čtyřmi parametry – typ filtru `type`, centrální frekvence `Fc`, šířka pásma `Q` a zesílení pásma `peakGain`. Jednotlivé filtry jsou zapojeny do kaskády. Typ filtru lze nastavit na *low-pass, high-pass, band-pass, notch, peak, low-shelf* či *high-self*. Centrální frekvence může nabývat hodnot od 20 Hz do 20 kHz, přičemž zadaná hodnota musí být nejdříve normalizována v poměru k frekvenci vzorkovací.

Parametr `Q` může být nastaven od 0.1 do 10. Poslední parametr `peakGain` udává zesílení frekvenčního pásma od -30 do 30 dB. Nastavení parametrů ekvalizéru probíhá pomocí struktury `TEqualizer`, jenž se skládá z šesti struktur `TFilter`, které obsahují hodnoty parametrů jednotlivých filtrů. Při vytvoření filtru či změně jeho parametrů je pro výpočet koeficientů volána metoda `calcBiquad`. Koeficienty jsou vypočítány dle vztahů uvedených v příloze B. Pro výpočet výstupní hodnoty signálu je k dispozici metoda `process`.

Zpožďovací efekt

Zpožďovací efekt je implementovaný pomocí třídy `Delay`, která je založena na principu jednoduché zpožďovací linky se statickým zpožděním. Základem tohoto efektu je struktura `TBuf`, která obsahuje dvě dynamicky alokovaná pole pro uložení zpožděných vzorků obou kanálů vstupního signálu. Velikost těchto polí odpovídá čtyřem dobám a je vypočítána dle vztahu:

$$n = \frac{4 \cdot 60 \cdot SR}{BPM}, \tag{7.1}$$

kde n je velikost pole ve vzorcích, BPM tempo v počtu úderů za minutu a SR je vzorkovací frekvence v počtu vzorků za sekundu. Informaci o nastavení tempa a vzorkovací frekvence lze od hostitelské aplikace získat voláním funkce `getTimeInfo`, která požadované informace vrácí v datové struktuře `VstTimeInfo`.

Nastavením příznaků `sync1` a `sync2` lze zpoždění signálu synchronizovat s tempem nahrávky. Délku zpoždění je možné ovlivnit parametry `time1` a `time2` pro nesynchronizované zpoždění či parametry `frac1` a `frac2` pro zpoždění synchronizované. Velikost zpětné vazby obou kanálů je nastavena parametry `feedback1` a `feedback2`. Třída `Delay` disponuje blokem pro zkreslení signálu a sadou filtrů s nimiž lze vytvářet pozoruhodné zvukové výstupy. Tento efekt může dle nastavení proměnné `mode` pracovat jak v režimu `mono`, tak v režimu `stereo`. Zajímavou modifikací by mohlo být přídání režimu `ping-pong`, u kterého by docházelo k přepínání kanálu po každém opakování.

Stereo

Stereo signál je narozdíl od mono signálu přenášen dvěma kanály, jenž mohou nést odlišnou zvukovou informaci. Právě změnou poměru signálu levého a pravého kanálu lze uspořádat panorama nahrávky, kde například kytara hraje více nalevo a smyčce více napravo. Vokální stopy jsou zpravidla umístěny na středu. V případě, kdy nahrávka obsahuje mnoho nástrojů úmístěných v prostoru, může středová vokální stopa působit poněkud úzce. K navození širšího dojmu lze využít algoritmů třídy `Stereo`. Pokud zvuková stopa již nese různé stereo informace, lze je zvýraznit, respektive potlačit, použitím metody `stereoEnhancer`, jejíž funkce je popsána rovnicemi:

$$
\begin{aligned}
mono &= (in_l + in_r)/2, && (7.2)\\
stereo &= (in_l - mono) \cdot width, && (7.3)\\
out_l &= mono + stereo, && (7.4)\\
out_r &= mono - stereo, && (7.5)
\end{aligned}
$$

kde parametr `width` ovlivňuje výslednou šířku signálu. Pokud oba kanály obsahují totožný signál, lze stereo efektu dosáhnout opožděním jednoho z nich, přičemž zpoždění musí být menší než 30 ms. Tento algoritmus je implementovaný metodou `monoToStereo`. Podobně jako třída `Delay` ukládá i třída `Distortion` zpožděné vzorky signálu do dynamicky alokovaných polí, jejichž velikost je dána součinem vzorkovací frekvence a délky maximálního zpoždění. Nastavením parametrů `time1` a `time2` lze ovlivnit zpoždění jednotlivých kanálů.

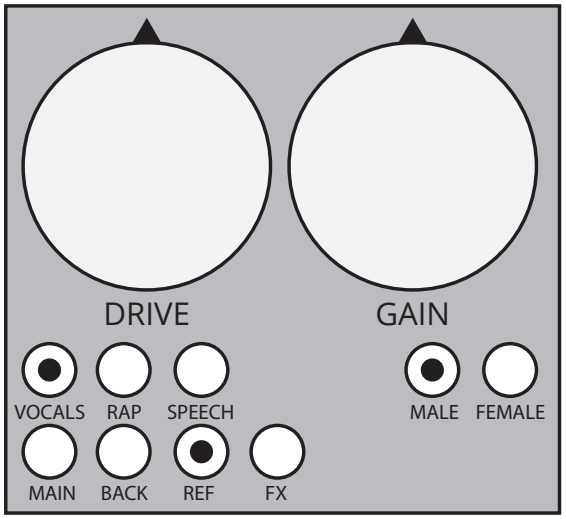
Distortion

Třída `Distortion` umožňuje jednoduché zkreslení signálu, jenž se využívá především pro ozvučení elektrických kytar. Aplikací na vokální stopy však můžeme dosáhnout efektu, který je podobný zvuku rozbité vysílačky. První parametr třídy `Distortion` je citlivostní práh `threshold`, jenž signál musí překročit, aby došlo ke zkreslení. Druhým parametrem je parametr `ratio`, který ovlivňuje poměr mezi vstupním a zkresleným signálem na výstupu. Zkreslení signálu je popsáno rovnicí:

$$
y = \begin{cases} t + (1-t) \cdot \tanh\left(\dfrac{x-t}{1-t}\right) & x > 0 \\ -(t + (1-t)) \cdot \tanh\left(\dfrac{x-t}{1-t}\right) & x \leq 0 \end{cases} . \qquad (7.6)
$$

7.3 Návrh a realizace uživatelského rozhraní

Primárním účelem pluginu Executor je zjednodušení práce při postprodukci vokálních stop, čemuž by měl odpovídat i vzhled aplikace. Většina méně zkušených uživatelů může mít problém s nastavením mnoha ovládacích prvků, proto jsem se rozhodl jejich počet minimalizovat. Původní návrh můžeme vidět na obrázku 7.6. K dispozici jsou dva ovládací knoby – *drive* pro řízení poměru mezi originálním a upraveným signálem a *gain* pro kontrolu hlasitosti výstupního signálu. Přednastavení jednotlivých modulů by pak bylo možné volit pomocí série přepínačů.



Obrázek 7.6: Prvotní návrh vzhledu pluginu.

Od svého prvního návrhu prošel vzhled pluginu celou řadou proměn. Mezi nejzásadnější považuji změnu konceptu správy presetů. Původní návrh byl v tomto ohledu značně omezený, neboť by nebylo možné přidávat či mazat přednastavené presety. Nový koncept využívá systém programů popsaný v kapitole 7.1.1. Lze tak poměrně lehce připravit banku presetů i pro hudební nástroje bez zásahu do struktury či vzhledu programu. Další změnou bylo přidání knobu pro ovládání vstupní hlasitosti signálu. Světelné diody pod ovládacími prvky signalizují, které moduly jsou momentálně aktivní. Grafický vzhled pluginu byl realizován tak, aby co nejvíce připomínal hardwarové zařízení.

Obrázek 7.7: Finální vzhled pluginu Executor.

7.4 Presety

Při vytváření banky presetů jsem vycházel zejména z vlastních zkušeností. Poslechem a experimentováním jsem sestavil množinu zhruba šedesáti presetů, ze kterých jsem následně vybral dvě desítky nastavení, jenž jsou obsaženy ve finální verzi pluginu. Při selekci jsem využil konzultací s lidmi, kteří se v problematice postprodukce pohybují a mají s ní dlouholeté zkušenosti. Výsledný soubor presetů pokrývá celou škálu použití včetně aplikace speciálních efektů. Snažil jsem se volit krátké, úderné a lehce zapamatovatelné názvy, které co nejvíce vystihují zvukový charakter nastavení. V tabulce 7.1 je uvedeno zapojení modulů pro jednotlivé presety, přičemž parametry modulů jsou individuálně přednastaveny v souboru `Preset.cpp` na přiloženém DVD.

	Preset	**ng**	**hp**	**cmp**	**eq**	**dst**	**str**	**del**	**lim**
0	Classic rap	✗	✓	✓	✓	✗	✗	✗	✗
1	Classic rap back	✓	✓	✓	✓	✗	✓	✗	✗
2	Double compression	✗	✓	✓	✗	✗	✗	✗	✓
3	Triple compression	✗	✓	✓	✗	✗	✗	✗	✓
4	Huge voice	✗	✓	✓	✓	✗	✗	✗	✗
5	Princess rap	✗	✓	✓	✓	✗	✗	✗	✗
6	Princess rap back	✓	✓	✓	✓	✗	✓	✗	✗
7	Aggressive rap	✗	✓	✓	✓	✗	✗	✗	✗
8	Nice vocal	✗	✓	✓	✓	✗	✓	✓	✗
9	Clean vocal	✓	✓	✓	✓	✗	✗	✗	✗
10	Grieving Queen	✗	✓	✗	✓	✗	✗	✓	✗
11	Stereo enhancer	✗	✗	✗	✗	✗	✓	✗	✓
12	Speech	✗	✓	✓	✓	✗	✗	✗	✗
13	Who are you	✗	✓	✓	✓	✗	✗	✓	✓
14	Rumburak	✗	✓	✓	✓	✗	✓	✓	✓
15	Tape crash	✗	✓	✓	✓	✗	✓	✓	✓
16	Hello Apollo	✗	✓	✓	✓	✗	✓	✓	✓
17	Broken transmitter	✓	✓	✗	✓	✓	✗	✗	✓
18	Radio man	✗	✗	✓	✓	✗	✗	✗	✓
19	Nothing to do	✗	✗	✗	✗	✗	✗	✗	✗

Tabulka 7.1: Zapojení modulů v rámci presetů.

Legenda

- **ng** – noise gate
- **hp** – high-pass filter
- **cmp** – compressor
- **eq** – equalizer
- **dst** – distortion
- **str** – stereo
- **del** – delay
- **lim** – limiter

Kapitola 8

Testování

8.1 Testy funkčnosti systému

Jednotlivé moduly pluginu byly během implementace postupně testovány v několika různých hostitelských aplikacích. Za zmínku stojí především program *VST Plugin Analyser*[1], který disponuje prostředky pro detailní testování VST pluginů. Kromě integrovaného generátoru bílého a růžového šumu obsahuje i zabudovaný spektrální analyzátor, na kterém lze sledovat frekvenční spektrum editovaného signálu. Druhým programem, který jsem často používal při testování během vývoje, byl *Minihost* vyvíjený firmou *Tobybear Productions*[2].

Obrázek 8.1: Spektrální analyzátor Voxengo SPAN.

Po úspěšné implementaci a otestování samostatných modulů s různým nastavením parametrů jsem z nich utvořil celek v podobě pluginu *Executor*. Tento plugin jsem testoval ve velmi rozšířených zvukových editorech *Cubase 5*[3] od firmy *Steinberg* a *FL Studio 10*[4]

[1] VST Plugin Analyser lze stáhnout ze stránek autora: http://www.savioursofsoul.de/Christian
[2] Webové stránky Tobybear Produtions jsou dostupné z: http://www.tobybear.de
[3] Trial verzi Cubase lze stáhnout z: http://www.steinberg.net/en/products/cubase/trial.html
[4] Demoverzi FL Studio lze stáhnout z: http://www.image-line.com/flstudio

od firmy *Image-Line*. Aplikace Cubase podporuje zpracování signálu jak v real-time, tak v off-line režimu. Nejdříve byla testována odezva systému na bílý šum a jednoduché periodické i neperiodické signály, následně jsem využil soubor vokálních stop, o kterém jsem se zmínil v podkapitole 6.2. K frekvenční analýze signálu jsem používal spektrální analyzátor Voxengo SPAN[5] pracující na principu rychlé Fourierovy transformace.

Minimální i doporučené požadavky pro běh hostitelských aplikací bývají zpravidla uvedeny na oficiálních stránkách výrobců. Pro testování jsem použil počítač s operační pamětí o kapacitě 6 GB a tříjádrovým procesorem řady *AMD Athlon II X3* o frekvenci 3000 MHz. Pro převod digitálního signálu na analogový jsem využíval zvukových rozhraní *Konnekt 8* a *Impact Twin* od firmy *TC Electronic*. Výsledný zvuk jsem poslouchal prostřednictvím aktivních monitorů *KRK VXT 6* a studiových sluchátek *Beats Studio*.

8.2 Uživatelské testování

Bohužel není jednoduché najít univerzální a objektivní měřítko pro hodnocení kvality zvukového výstupu pluginu. Z tohoto důvodu jsem se rozhodl pro uživatelské testování formou dotazníku. Sestavil jsem dotazník obsahující třináct otázek, jejichž plné znění je uvedeno v dodatku A.1. Otázky se týkají zkušeností uživatele s nahráváním a postprodukcí hudebních nahrávek a především pak mírou spokojenosti s provedením a praktickým využitím pluginu Executor. Subjektivní posudek může být ovlivněn celou řadou faktorů, mezi které patří zejména typ reproduktorové sestavy, akustika poslechové místnosti a citlivost sluchu posluchače.

Uživatelského testování se zúčastnilo celkem dvacet šest uživatelů, mezi nimiž byli jak lidé s bohatými a dlouholetými zkušenostmi v oblasti postprodukce hudby, tak i hudební producenti, skladatelé a interpreti, kteří se danou problematikou zabývají krátce či pouze okrajově. Všichni testující si z webových stránek[6] stáhli plugin Executor a byli seznámeni s jeho funkcí a použitím, přičemž těm, kteří nedisponují prostředky pro získání testovacích dat, byl poskytnut zmiňovaný soubor vokálních stop. Uživatelům byl následně předložen dotazník, jehož vyhodnocení je uvedeno v tabulce 8.1. Několik zkušenějších uživatelů bylo pak požádáno o sepsání krátkých posudků, které jsou uvedeny v dodatku A.2.

	Rozsah	**Průměrná známka**	**Směrodatná odchylka**
Využití pro demo	1-5	1.3	0.586
Využití pro mix	1-5	1.6	0.841
Celkový dojem	1-5	1.2	0.4

Tabulka 8.1: Vysledky uživatelského testování.

Z výsledků uživatelského testování můžeme vyvodit závěr, že aplikaci Executor lze úspěšně využívat jak při tvorbě demonahrávek, tak během postprodukce hudebních skladeb. Všechny testující uživatele zaujal vzhled a intuitivní ovládání aplikace. Rozsah a nastavení zabudovaných presetů bylo hodnoceno jako plně vyhovující. Někteří zkušení uživatelé, kteří se posprodukcí nahrávek zabývají profesionálně, postrádali detailnější nastavení parametrů jednotlivých modulů pluginu. Často kladenou otázkou bylo, zdali plánuji vytvoření podobného typu pluginu i pro hudební nástroje, případně jestli uvolním jednotlivé moduly v podobě samostatných pluginů.

[5] Spektrální analyzátor Voxengo SPAN lze stáhnout z: http://www.voxengo.com/product/span
[6] Aktuální verzi pluginu Executor lze stáhnout z: http://executor.tntrecords.cz

Kapitola 9

Závěr a budoucí práce

9.1 Shrnutí

V rámci této bakalářské práce jsem se seznámil s principy zvukových efektů a procesorů, osvojil si základní i pokročilejší postupy nahrávání a postprodukce hudebních skladeb a získané znalosti a zkušenosti zúročil při návrhu a realizaci systému pro zpracování vokálních stop. Výstupem projektu je funkční a otestovaná aplikace v podobě VST pluginu, která prostřednictvím integrované banky presetů pomáhá uživateli výrazně urychlit postprodukci hlasových nahrávek. Při implementaci aplikace byl kladen důraz na znovupoužitelnost zdrojových kódů. Plugin se skládá z několika modulů, které lze zaměnit či využít k tvorbě samostatných projektů.

9.2 Využití aplikace

Využití pluginu Executor vidím především při tvorbě demonahrávek jak v domácích a poloprofesionálních, tak i v profesionálních nahrávacích studiích. Výsledky uživatelského testování potvrzují vhodně zvolený rozsah i nastavení presetů. Aplikace vzhledem připomíná reálné hardwarové zařízení s minimem ovládacích prvků, čímž pozitivně oslovila drtivou většinu testujících uživatelů. Plugin lze využít i při mixu hudebních skladeb, přičemž je však zkušený uživatel limitovaný právě absencí pokročilejších voleb nastavení. Koncept pluginu byl zvolen tak, aby bylo možné snadno připravit banku presetů i pro hudební nástroje bez zásahů do struktury či vzhledu pluginu.

9.3 Budoucí práce

Během vývoje aplikace mě napadla celá řada inovací, které bych chtěl postupem času zrealizovat. V první řadě grafické uživatelské rozhraní postrádá vizualizaci úrovně hlasitosti signálu. Dalším rozšířením by mohla být implementace dozvukového efektu, jenž by našla uplatnění především pro zpívané vokální stopy. Modifikací zpožďovací linky modulu delay by bylo možné poměrně snadno získat zvukové efekty chorus, flanger a vibrato.

V reakci na názory pokročilých uživatelů zabývajících se postprodukcí hudby jsem začal připravovat komplexnější verzi programu Executor s výrazně detailnějším ovládáním jednotlivých bloků. Samostatné moduly jsem se rozhodl vydat v podobě edice pluginů, jejichž podobu můžete vidět na následující stránce. V dlouhodobém časovém horizontu bych se rád soustředil na podrobnější studii simulace reálných zvukových procesorů.

Obrázek 9.1: VST plugin Gate z edice TNT.

Obrázek 9.2: VST plugin Limiter z edice TNT.

Obrázek 9.3: VST plugin Stereo z edice TNT.

Obrázek 9.4: VST plugin Gain z edice TNT.

Literatura

[1] Steinberg Media Technologies GmbH – VST Plug-Ins SDK Documentation [online]. Dostupné z: http://www.gersic.com/vstsdk, 2003 [cit. 21. 05. 2014].

[2] Boulanger, R. C.; Lazzarini, V.: *The Audio Programming book*. Cambridge, Massachusetts, USA: MIT Press, 2011, ISBN 978-0-262-01446-5, 889 s.

[3] Brickhill, T.: Issues with Multiband Compressor. Technická zpráva, The University of Sydney, Faculty of Architecture, Desing and Planning, 2012.

[4] Bristow-Johnson, R.: Cookbook formulae for audio EQ biquad filter coefficients [online]. Dostupné z: http://www.musicdsp.org/files/Audio-EQ-Cookbook.txt, 2001 [cit. 21. 05. 2014].

[5] Ekeroot, J.: *Implementing a parametric EQ plug-in in C++ using the multi-platform VST specification*. Extended essay, Luleå University of Technology, 2003.

[6] Everest, F. A.: *Master Handbook of Acoustics*. New York: McGraw-Hill, Čtvrté vydání, 2002, ISBN 0-07-139974-7, 592 s.

[7] Giannoulis, D.; Massberg, M.; Reiss, J. D.: Digital Dynamic Range Compressor Design. Technická zpráva, Queen Mary University of London, 2012.

[8] Grace, R.: *Hudba a zvuk na počítači*. Praha: Grada Publishing, první vydání, 1999, ISBN 80-716-9519-X, 288 s.

[9] Hoffmann, P.: *Využití Cubase pro zpracování audio signálu*. Diplomová práce, UTB ve Zlíně, Fakulta technologická, 2005.

[10] Kravařík, J.: *Implementace zpožďovacího efektu ve VST*. Bakalářská práce, České vysoké učení technické v Praze, Fakulta elektrotechnická, 2008.

[11] McLean, P.: Modeling dynamic range compression in the digital domain. Technická zpráva, The University of Sydney, Faculty of Architecture, Desing and Planning, 2012.

[12] McLoughlin, I.: *Applied Speech and Audio Processing*. Cambridge, Spojené království: Cambridge University Press, první vydání, 2009, ISBN 978-0-521-51954-0, 206 s.

[13] Petelin, R.: *Vytváříme a mixujeme hudbu v programu FL Studio*. Brno: Computer Press, první vydání, 2006, ISBN 80-251-1125-3, 246 s.

[14] Redmon, N.: EarLevel Engineering [online]. Dostupné z: http://www.earlevel.com, 2014 [cit. 21. 05. 2014].

[15] Reichl, J.; Všetička, M.: Encyklopedie fyziky [online].
Dostupné z: http://fyzika.jreichl.com, 2006 [cit. 21. 05. 2014].

[16] Ručkay, L.: *Implementace IIR filtru - bikvadratické sekce - na FPGA*. Diplomová práce, ČVUT v praze, Fakulta Elektrotechnická, 2005.

[17] Stedman, C.: Technology review of dynamic range compression. Technická zpráva, The University of Sydney, Faculty of Architecture, Desing and Planning, 2012.

[18] Vlachý, V.: *Praxe zvukové techniky*. Praha: Muzikus, třetí vydání, 2008, ISBN 978-80-86253-46-5, 298 s.

[19] Vích, R.; Smékal, Z.: *Číslicové filtry*. Praha: Academia, první vydání, 2000, ISBN 80-200-0761-X, 218 s.

[20] Zölzer, U.; Amatriain, X.; Arfib, D.; aj.: *DAFX – Digital Audio Effects*. New York City: John Wiley & Sons, 2002, ISBN 0-471-49078-4, 554 s.

[21] Škvor, Z.: *Akustika a elektroakustika*. Vodičkova 40, Praha: Academia, 2001, ISBN 80-200-0461-0, 528 s.

Dodatek A

Uživatelské hodnocení aplikace

A.1 Dotazník

Pro získání zpětné vazby od uživatelů, kterým jsem poskytl aplikaci k testování, jsem sestavil tento dotazník. První tři otázky jsou zaměřeny na zkušenosti uživatelů s postprodukcí hudby a lze na ně odpovědět buďto *ano* nebo *ne*.

- Máte zkušenosti s nahráváním hudebních skladeb?
- Zabýváte se postprodukcí hudebních nahrávek?
- Setkal/a jste se dříve s podobným typem pluginu?

Dalších sedm otázek se týká spokojenosti s provedením aplikace. Cílem těchto otázek je získání potřebných informací pro budoucí inovace. Odpovědi na tuto sadu otázek jsou *ano*, *spíše ano*, *nevím*, *spíše ne* a *ne*.

- Jste spokojený/á se vzhledem aplikace?
- Přijde Vám ovládání aplikace intuitivní?
- Vyhovuje Vám minimální počet ovládacích prvků?
- Uvítal/a byste možnost detailnějšího ovládání jednotlivých modulů?
- Vyhovuje Vám množství přednastavených presetů?
- Považujete sadu presetů za vyhovující pro dané účely?
- Uvítal/a byste sadu presetů i pro různé hudební nástroje?

Poslední tři otázky hodnotí celkovou použitelnost aplikace při postprodukci hudby a lze je oznámkovat známkami 1-5 dle české školní stupnice, tedy jednička znamená nejlepší hodnocení, pětka naopak nejhorší.

- Jak byste ohodnotil/a aplikaci z hlediska využití pro tvorbu demo nahrávek?
- Má dle Vašeho názoru aplikace potenciál pro využití při mixu nahrávky?
- Jak byste ohodnotil Váš celkový dojem z pluginu Executor?

A.2 Hodnocení

Jindřich Pevný – hudební skladatel a producent

Plugin Executor na první pohled překvapí vzhledem uživatelského rozhraní, za které by se nemusely stydět ani velké vývojářské společnosti zabývající se tvorbou VST pluginů. Vzhledem k celkovému konceptu programu, který má být jednoduchou možností jak markantně zlepšit kvalitu nahrávek především méně zkušených uživatelů, to považuji za velké plus. Jednotlivé presety jsou rovnoměrně rozloženy do celého spektra užití – od jednoduchých (např. Clean Vocal) přes efekty pracující s atmosférou nahrávky (skvěle znějící Grieving Queen) až po destruktivní algoritmy (Broken Transmitter).

Pro maximální využití síly tohoto pluginu bych osobně uvítal možnost pokročilého nastavení, kdy by uživatel mohl přímo ovlivňovat jednotlivé bloky pluginu, ovšem vzhledem k tomu, že program je cílen především na méně zkušené uživatele, je zcela pochopitelné, že tato možnost tu chybí. Plugin jsem testoval na několika syrových studiových nahrávkách a posun v kvalitě byl více než znatelný. Užití Executora, i když bylo tak zamýšleno, se neomezuje jen na vokální stopy – poměrně zajímavých výsledků jsem dosáhl i při aplikaci na zejména středobasově orientované syntezátorové linky.

Štěpán Bárta – audio inženýr, hudební producent

Při prvním otevření mě mile překvapil vzhled pluginu, který vypadá mnohem lépe než pluginy, co stojí mnohdy i několik tisíc korun. Ovládání je, myslím, i pro laika jednoduché a intuitivní. Použití stylu one knob byl dobrý nápad, jelikož právě s velkým počtem ovládacích prvků mají laici velký problém – ví, jak by jim to mělo znít, ale neumí to na klasickém ekvalizéru ani kompresoru správně nakroutit.

Presety jsou udělané skvěle – líbí se mi, že autor myslel na všechny základní možnosti a v jednotlivých presetech jsou přesně ty věci, které člověk vždy pracně přidává několika pluginy – ekvalizér, kompresor, delay apod. Jejich aktivitu navíc člověk může sledovat ve spodní části pluginu. U presetu back vocal mě mile překvapil gate. Dobrým nápadem bylo i přidání oblíbených zvláštních efektů jako je například telefonní efekt. Myslím, že laikovi stačí na každé stopě nasadit tento plugin, zvolit vhodný preset a má skvělou demo nahrávku skoro bez práce.

Tomáš Rapant – hudební skladatel, majitel nahrávacího studia

Již devět let se velmi aktivně věnuji komponováním hudby a vlastním nahrávací studio. Bohužel mé časové vytížení mi nedovoluje hlouběji se zabývat mixem a masteringem skladeb, proto zpravidla volím cestu zaslání svých nahrávek do některého z postprodukčních studií. Často je ale potřeba velmi rychle připravit kvalitní náslech například pro natáčení videoklipu. S hudebními nástroji si s trochou úsilí poradím tak, abych byl spokojený, nicméně mým největším kamenem úrazu je právě zvučení vokálů.

Ze začátku jsem byl k použití pluginu Executor poněkud skeptický, o to více mě pak jeho výsledky pozitivně překvapily. Plugin obsahuje řadu presetů, které umožňují velmi efektivně a jednoduše ošetřit vokál tak, aby zněl přinejmenším obstojně. Z pluginu mám velmi pozitivní dojmy, pokuď bych však měl jmenovat jeho nedostatky, byl by to zejména fakt, že svojí velmi striktní jednoduchostí může zkušeného uživatele při používání značně limitovat.

Zdenko Kamenický – hudební skladatel, majitel nahrávacího studia

Po otevření pluginu na první pohled zaujme velice čistý a moderní vzhled, který s lehkou nadsázkou vzbuzuje dojem HW nástroje. Vhodně je zvolené také one knob ovládání, které patřičně ocení zejména poloprofesionální studia při tvorbě náslechu pro interpreta – snadno se dá obejit zdlouhavý proces premixu nahrávky a interpret si může v pohodlí svého domova, případně už v autě na cestě ze studia naposlouchat nahrávku. Díky velkému množství presetů se dá poměrně snadno nastavit požadovaný zvuk a tím získává interpret jasnou představu, zdali se má tímto směrem ubírat celá nahrávka či nikoliv.

Zde ovšem narážíme na fakt, že největší přednost pluginu Executor je zároveň i jeho největší slabinou. I přes širokou škálu presetů chybí možnost jejich editace či modifikace – při testování jsem například vybral preset, který zapadal do nahrávky, ale potřeboval jsem mírně doladit některé frekvence a zjemnit delay. Určitě velkým plusem by bylo zapínání a vypínání jednotlivých efektů s možností nastavení jejich síly. Tím by Executor získal další rozměr a věřím, že by si našel široké uplatnění i v profesionálních studiích. Nicméně aktuální verze je pro tvorbu náslechů více než dostačující.

Po pár dnech používání jsem ušetřil spoustu času ve studiu – interpreti jsou spokojení, jelikož platí kratší čas ve studiu, zatímco já mohu pracovat s dalšími interprety. Tato deviza se pluginu Executor nedá upřít. Executor by mohl být vyhledávaným pluginem pro amatérská a poloamatérská studia, protože se řídí heslem – za málo pěnez hodně muziky. Dle mého názoru by se dalo po provedení určitých úprav (např. přídání detailnějšího uživatelského rozhraní) uvažovat o jeho komerčním uvedení na trh.

Tomáš Lašan – interpret, majitel nahrávacího studia

Co nelze přehlédnout je krásný vzhled pluginu. Reálné zpracování designu navodí příjemný pocit, jako byste měli skutečný přístroj zasunutý do Vašeho monitoru. S rostoucím technologickým vývojem programů pro mix a mastering je kladen důraz také na kvalitu nahrávek a jejich postprodukci. V tomto ohledu je Executor šikovný pomocník, který zajišťuje zvýšení kvality nahraného hlasu. V jednoduchosti je síla. V tomto duchu je navržen i Executor, který se ovládá pouze třemi ovládacími prvky.

Databáze nabízí dostatečné množství chytře pojmenovaných presetů. Pokud máte přesnou představu, jak by měla vokální stopa znít, jednoduše si vyberete z banky přednastavení. Jednoduché ovládání bez dlouhého studování a okamžitá možnost použití jsou pro laiky velice vítané. Profesionál by však hledal i detailnější nastavaní modulů, ovšem to neznamená, že by tento VST plugin nebyl prospěšný i pro ty, kteří mají více zkušeností.

Ondřej Žatkuliak – audio inženýr, hudební skladatel a producent

Executor je dobře vypadající univerzální presetový efekt. Intenzitu jednotlivých přednastavení lze ovládat dominantním potenciometrem drive. Dále lze upravovat ještě vstupní a výstupní gain úrovně. Vše ostatní dělá pod kapotou software za Vás. Funguje dle očekávání a jeho hlavní využití předpokládám v oblasti zpracování vokálů v moderním mixu. Dobrá práce!

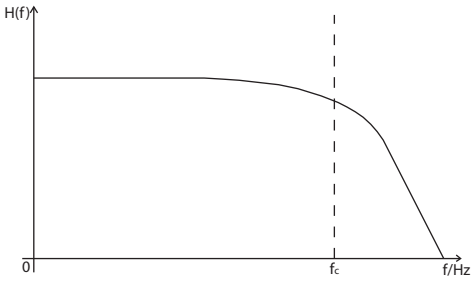
Dodatek B

Výpočet koeficientů filtrů

V této příloze jsou uvedeny typy filtrů i s postupem výpočtu jejich koeficientů. Jedná se o filtry IIR druhého řádu, takzvané *bikvady*, které byly využity k realizaci ekvalizéru. Přenosové funkce filtrů byly odvozeny z analogových prototypů a digitalizovány s použitím bilineární transformace. Parametry výpočtu koeficientů jsou centrální frekvence F_c, vzorkovací frekvence F_s, šířka pásma Q a parametr $peakGain$, který ovlivňuje zesílení (boost) či potlačení (cut) frekvenčního pásma. Podrobný popis transformace a odvození následujících rovnic pro výpočet koeficentů lze nalézt v [4] a [14].

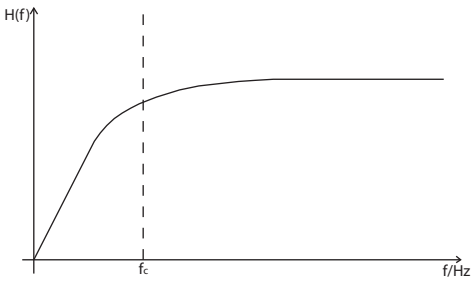
$$V = 10^{|peakGain|/20} \tag{B.1}$$

$$K = \tan\left(\pi F_c / F_s\right) \tag{B.2}$$



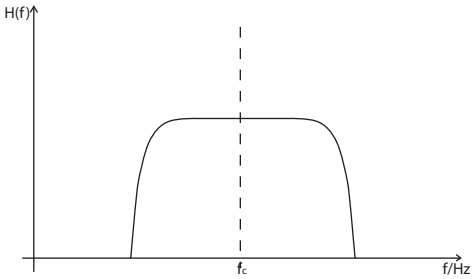
$$\begin{aligned} norm &= \frac{1}{1 + \frac{K}{Q} + K^2} \\ b_0 &= K^2 \cdot norm \\ b_1 &= 2b_0 \\ b_2 &= b_0 \\ a_1 &= 2\left(K^2 - 1\right) \cdot norm \\ a_2 &= \left(1 - \frac{K}{Q} + K^2\right) \cdot norm \end{aligned}$$

Tabulka B.1: Filtr typu low-pass.



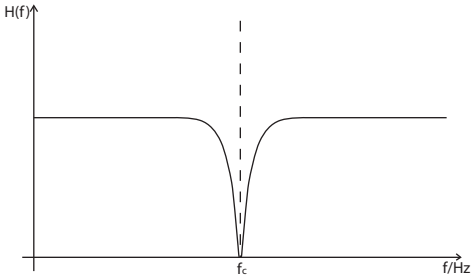
$$\begin{aligned} norm &= \frac{1}{1 + \frac{K}{Q} + K^2} \\ b_0 &= 1 \cdot norm \\ b_1 &= -2b_0 \\ b_2 &= b_0 \\ a_1 &= 2\left(K^2 - 1\right) \cdot norm \\ a_2 &= \left(1 - \frac{K}{Q} + K^2\right) \cdot norm \end{aligned}$$

Tabulka B.2: Filtr typu high-pass.



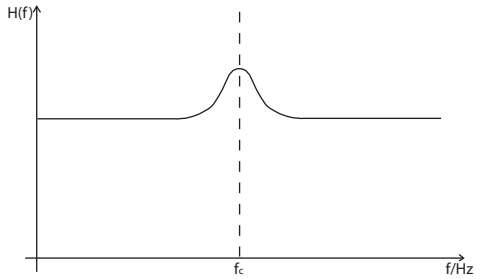
$$
\begin{aligned}
norm &= \frac{1}{1+\frac{K}{Q}+K^2} \\
b_0 &= \frac{K}{Q}\cdot norm \\
b_1 &= 0 \\
b_2 &= -b_0 \\
a_1 &= 2\left(K^2-1\right)\cdot norm \\
a_2 &= \left(1-\frac{K}{Q}+K^2\right)\cdot norm
\end{aligned}
$$

Tabulka B.3: Filtr typu band-pass.



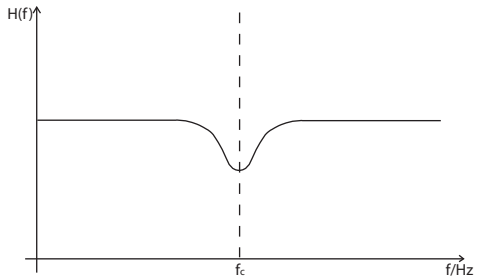
$$
\begin{aligned}
norm &= \frac{1}{1+\frac{K}{Q}+K^2} \\
b_0 &= \left(1+K^2\right)\cdot norm \\
b_1 &= 2\left(K^2-1\right)\cdot norm \\
b_2 &= b_0 \\
a_1 &= b_1 \\
a_2 &= \left(1-\frac{K}{Q}+K^2\right)\cdot norm
\end{aligned}
$$

Tabulka B.4: Filtr typu notch.



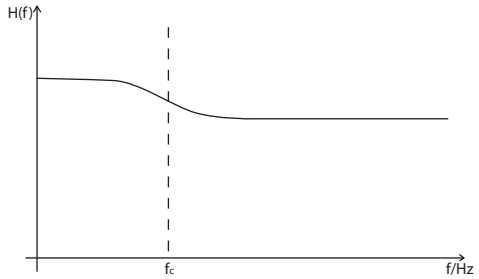
$$
\begin{aligned}
norm &= \frac{1}{1+\frac{K}{Q}+K^2} \\
b_0 &= \left(1+\frac{VK}{Q}+K^2\right)\cdot norm \\
b_1 &= 2\left(K^2-1\right)\cdot norm \\
b_2 &= \left(1-\frac{VK}{Q}+K^2\right)\cdot norm \\
a_1 &= b_1 \\
a_2 &= \left(1-\frac{K}{Q}+K^2\right)\cdot norm
\end{aligned}
$$

Tabulka B.5: Filtr typu peak – boost.



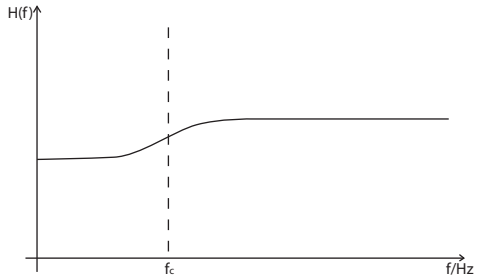
$$
\begin{aligned}
norm &= \frac{1}{1+\frac{VK}{Q}+K^2} \\
b_0 &= \left(1+\frac{K}{Q}+K^2\right)\cdot norm \\
b_1 &= 2\left(K^2-1\right)\cdot norm \\
b_2 &= \left(1-\frac{K}{Q}+K^2\right)\cdot norm \\
a_1 &= b_1 \\
a_2 &= \left(1-\frac{VK}{Q}+K^2\right)\cdot norm
\end{aligned}
$$

Tabulka B.6: Filtr typu peak – cut.



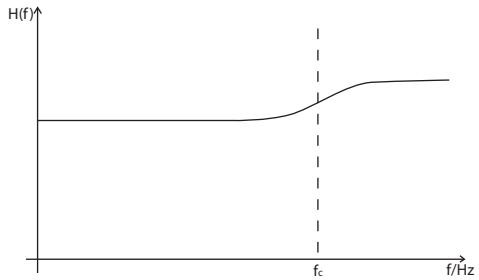
$$
\begin{aligned}
norm &= \frac{1}{1+\sqrt{2}K+K^2} \\
b_0 &= \left(1+\sqrt{2V}K+VK^2\right)\cdot norm \\
b_1 &= 2\left(VK^2-1\right)\cdot norm \\
b_2 &= \left(1-\sqrt{2V}K+VK^2\right)\cdot norm \\
a_1 &= 2\left(K^2-1\right)\cdot norm \\
a_2 &= \left(1-\sqrt{2}K+K^2\right)\cdot norm
\end{aligned}
$$

Tabulka B.7: Filtr typu low-shelf – boost.



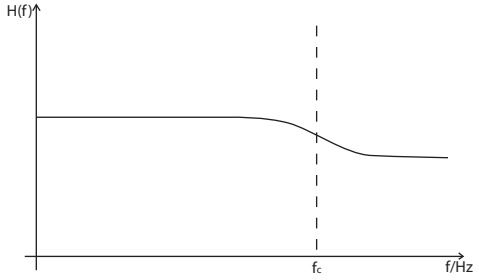
$$
\begin{aligned}
norm &= \frac{1}{1+\sqrt{2V}K+VK^2} \\
b_0 &= \left(1+\sqrt{2}K+K^2\right)\cdot norm \\
b_1 &= 2\left(K^2-1\right)\cdot norm \\
b_2 &= \left(1-\sqrt{2}K+K^2\right)\cdot norm \\
a_1 &= 2\left(VK^2-1\right)\cdot norm \\
a_2 &= \left(1-\sqrt{2V}K+VK^2\right)\cdot norm
\end{aligned}
$$

Tabulka B.8: Filtr typu low-shelf – cut.



$$
\begin{aligned}
norm &= \frac{1}{1+\sqrt{2}K+K^2} \\
b_0 &= \left(V+\sqrt{2V}K+K^2\right)\cdot norm \\
b_1 &= 2\left(K^2-V\right)\cdot norm \\
b_2 &= \left(V-\sqrt{2V}K+VK^2\right)\cdot norm \\
a_1 &= 2\left(K^2-1\right)\cdot norm \\
a_2 &= \left(1-\sqrt{2}K+K^2\right)\cdot norm
\end{aligned}
$$

Tabulka B.9: Filtry typu high-shelf – boost.



$$
\begin{aligned}
norm &= \frac{1}{V+\sqrt{2V}K+K^2} \\
b_0 &= \left(1+\sqrt{2}K+K^2\right)\cdot norm \\
b_1 &= 2\left(K^2-1\right)\cdot norm \\
b_2 &= \left(1-\sqrt{2}K+K^2\right)\cdot norm \\
a_1 &= 2\left(K^2-V\right)\cdot norm \\
a_2 &= \left(V-\sqrt{2V}K+K^2\right)\cdot norm
\end{aligned}
$$

Tabulka B.10: Filtr typu high-shelf – cut.

Dodatek C

Návod k použití aplikace

Plugin *Executor* je určený především pro operační systémy *Microsoft Windows XP* a vyšší. Aplikace byla implementována pomocí vývojového prostředí *Microsoft Visual Studio 2013*. Soubor s projektem, pomocí něhož lze aplikaci jednoduše přeložit, je umístěn v adresáři se zdrojovými kódy na přiloženém DVD. Kromě zdrojových kódu obsahuje DVD i přeloženou aplikaci v podobě dynamické knihovny *dll*.

Pro použití pluginu je nutné mít nainstalovaný software, který bude sloužit jako hostitelská aplikace. Plugin byl úspěšně otestován v aplikacích *Steinberg Cubase 5* a *Image-Line FL Studio 11*. Měl by však bez problému fungovat i ve všech ostatních programech podporujících VST pluginy. Kromě výše zmíněných programů to jsou například *Canewalk Sonar*, *Ableton Live* či *Steinberg Nuendo*.

Před spuštěním hostitelské aplikace je zapotřebí překopírovat plugin ve formátu *dll* do adresáře, ve kterém hostitelská aplikace VST pluginy očekává. Umístění tohoto adresáře lze zpravidla zjistit v nastavením aplikace. Po spuštění programu a vytvoření nového respektive otevření existujícího projektu je již možné vkládat plugin *Executor* na jednotlivé zvukové stopy či sběrnice virtuálního mixážního pultu.

Dodatek D

Obsah přiloženého DVD

Přiložené DVD obsahuje následující adresáře:

- `pdf` – zde je uložen text technické zprávy ve formátu *pdf*
- `latex` – adresář obsahuje zdrojové soubory technické zprávy
- `matlab` – v této složce jsou uloženy soubory s algoritmy pro MATLAB
- `samples` – složka s vokálními stopami použitými při testování
- `using` – zde jsou uloženy zvukové ukázky získané využitím pluginu
  - `mix` – adresář obsahuje kompletní mixy skladeb
  - `examples` – složka s příklady prezentujícími použití presetů
  - `using.html` – webové rozhraní pro poslech zvukových ukázek
- `bin` – obsahuje výslednou aplikaci v podobě dynamické knihovny *dll*
- `src` – v tomto adresáři se nachází zdrojové kódy aplikace
  - `win` – adresář obsahuje projekt pro MS Visual Studio ve formátu *sln*
  - `vst` – složka se zdrojovými kódy VST SDK a VST GUI