VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2016 Robin Zoň

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

VST PLUG-IN MODUL PRO ŘÍZENÍ DYNAMIKY ZVUKOVÉHO SIGNÁLU
VST PLUG-IN FOR DYNAMIC PROCESSING OF AUDIO SIGNAL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Robin Zoň**
AUTHOR

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

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Audio inženýrství**
Ústav telekomunikací

Student: Robin Zoň ***ID:*** 165032
Ročník: 3 ***Akademický rok:*** 2015/16

NÁZEV TÉMATU:

VST plug-in modul pro řízení dynamiky zvukového signálu

POKYNY PRO VYPRACOVÁNÍ:

V prostředí Matlab realizujte funkci pro zvukový efekt pro zpracování dynamiky zvukových signálů, který bude realizovat efekt kompresor, limiter, expander a šumová brána a bude umožňovat připojení řídicí větve k jinému zdroji signálu, než je signál přímé větvě. Vytvořte signály pro otestování dynamického procesoru a v práci uveďte signály v jednotlivých bodech řídicí větve. Tento efekt následně realizujte jako zásuvný modul technologie VST, který bude umožňovat zobrazení a uložení vstupních a výstupních signálů jednotlivých bloků řídicí větve efektu.

DOPORUČENÁ LITERATURA:

[1] ZÖLZER, U. DAFX – Digital Audio Effects, 1st ed. New York: John Wiley & Sons, Ltd, 2002, 533 p. ISBN 0-471-49078-4.

[2] VLACHÝ, V. Praxe zvukové techniky. Nakladatelství Muzikus, Praha, 1995. ISBN 80-901537-6-3

[3] BOURLANGER, R., LAZZARINI, V., The Audio Programming Book. MIT Press, 2011. ISBN 978-0-262-01-46-5

[4] PIRKLE, W. Designing Audio Effect Plug-Ins in C++. Focal Press, 2013. ISBN 978-0-240-82515-1

Termín zadání: 1.2.2016 ***Termín odevzdání:*** 1.6.2016

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

doc. Ing. Jiří Mišurec, CSc., *předseda oborové rady*

ABSTRAKT

Cílem této bakalářské práce je realizace efektů pro úpravu dynamiky signálu. Práce zahrnuje čtyři základní dynamické efekty – kompresor, expander, limiter a šumovou bránu. První část práce zahrnuje vypracování efektů v softwarovém prostředí Matlab. Toto řešení nabízí napojení řídící větve na jiný zdroj signálu, zobrazení průběhů jednotlivých bloků řídící větve, nahrávání výstupního signálu a umožňuje ukládání výstupních charakteristik. Ve druhé části práce byl efekt realizován pomocí technologie VST, což nabízí možnosti úpravy parametrů v reálném čase a zobrazení a nahrání signálu v jednotlivých bodech řídící větve signálu. Vypracování zahrnuje i krátké pojednání o využití takových dynamických efektů v praxi a sbírku vzorových příkladů pro demonstraci použití takovýchto efektů.

KLÍČOVÁ SLOVA

Úprava dynamiky signálu, číslicové filtry, dynamické efekty, kompresor, expander, limiter, šumová brána, Matlab, VST, C++, JUCE

ABSTRACT

The aim of this thesis is the implementation of effects for adjusting the signal dynamics. This project includes four basic dynamic effects - compressor, expander, limiter and noise gate. The first part is developed in Matlab software. This solution offers signal sidechain input, display of signal waveforms of each block of sidechain branch, recording output signal and allows storage of output characteristics. In the second part, the effect is being realized using VST technology, which offers the possibility of changing the parameters in real time and displaying and recording the signal at various points within the control branch. Project also includes a brief discussion of the use of such dynamic effects in practice and collection of model examples to demonstrate the use of such effects.

KEYWORDS

Adjusting signal dynamics, digital filters, dynamic effects, compressor, expander, limiter, noise gate, Matlab, VST, C++, JUCE

ZOŇ, R. *VST plug-in modul pro řízení dynamiky zvukového signálu*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2016. 64 s. Vedoucí bakalřské práce Ing. Jiří Schimmel, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „VST plug-in modul pro řízení dynamiky zvukového signálu“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

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

Brno ....

podpis autora

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

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

PODĚKOVÁNÍ

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

Brno . . . . . .

. . . . . .

podpis autora

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OBSAH

Úvod **12**

1 Dynamika signálů **13**
1.1 Akustické signály . . . 13
1.2 Číslicové signály . . . 13
1.3 Současný trend . . . 14
1.3.1 Maximální přirozenost . . . 14
1.3.2 Maximální hlasitost . . . 14

2 Efekty pro úpravu dynamiky **15**
2.1 Kompresor . . . 15
2.2 Expander . . . 16
2.3 Limiter . . . 17
2.4 Šumová brána . . . 18
2.5 Efekty s kmitočtovými filtry v řídící větvi . . . 19
2.5.1 Deesser . . . 19
2.5.2 Vícepásmový kompresor . . . 19

3 Úprava dynamiky signálu **20**
3.1 Sledovač obálky . . . 20
3.1.1 Sledovač obálky špičkové hodnoty . . . 20
3.1.2 Sledovač obálky efektivní hodnoty . . . 21
3.2 Výpočet zesilovacího činitele . . . 22
3.2.1 Výpočet zesilovacího činitele kompresoru . . . 22
3.2.2 Výpočet zesilovacího činitele expanderu . . . 22
3.3 Vyhlazení zesilovacího činitele . . . 23
3.3.1 Dynamický filtr signálu . . . 23
3.3.2 Dynamický filtr s parametrem hold . . . 24

4 Realizace v prostředí Matlab **26**
4.1 Struktura programu . . . 26
4.1.1 Struktura řídící větve . . . 26
4.1.2 Funkce konkrétních efektů . . . 27
4.1.3 Funkce pro zjednodušení práce s efekty . . . 27
4.2 Spuštění vzorových příkladů . . . 28
4.3 Výsledky práce . . . 29
4.3.1 Kompresor . . . 29
4.3.2 Expander . . . 31

4.3.3 Limiter . . . 33
4.3.4 Šumová brána . . . 35
4.3.5 Ducking . . . 37

5 Realizace v jazyce C++ 38
5.1 Struktura programu . . . 38
5.1.1 Třídy pro reprezentaci signálu . . . 38
5.1.2 Třídy pro reprezentaci výpočetních bloků . . . 39
5.1.3 Třídy výpočetních bloků . . . 39
5.1.4 Hlavní výpočetní třída . . . 40
5.2 Vzhled a ovládání . . . 40
5.3 Nahrávání výstupu . . . 41
5.4 Předvolby . . . 42
5.4.1 Vzorový příklad č. 1 . . . 43
5.4.2 Vzorový příklad č. 2 . . . 44
5.4.3 Vzorový příklad č. 3 . . . 45
5.4.4 Vzorový příklad č. 4 . . . 46
5.4.5 Vzorový příklad č. 5 . . . 47
5.4.6 Vzorový příklad č. 6 . . . 48
5.4.7 Vzorový příklad č. 7 . . . 49
5.4.8 Vzorový příklad č. 8 . . . 50

6 Závěr 51

Literatura 52

Seznam symbolů, veličin a zkratek 53

Seznam příloh 55

A Vypracování v prostředí Matlab 56
A.1 Graf signálových toků . . . 58

B Vypracování v jazyce C++ 59
B.1 VST plugin modul . . . 59
B.2 Projekt v MS Visual Studio 2015 . . . 59
B.3 Zdrojový kód . . . 59
B.3.1 Výpočetní část . . . 59
B.3.2 Grafická část . . . 60
B.3.3 Procesní část . . . 62

C Audioukázky **63**
C.1 Implementace Matlab 63
C.2 Implementace VST 63

D Parametry funkce runexample **64**

SEZNAM OBRÁZKŮ

2.1 Převodní charakteristika kompresoru s prahovou úrovní -30 dB a poměrem 3 : 1. 15
2.2 Převodní charakteristika expanderu s prahovou úrovní -20 dB, poměrem 1 : 3 a maximálním útlumem -20 dB. 16
2.3 Převodní charakteristika limiteru s prahovou úrovní -30 dB. 17
2.4 Převodní charakteristika šumové brány s prahovou úrovní -30 dB. . . 18
3.1 Blokové schéma zapojení efektu dynamiky. 20
3.2 Blokové schéma sledovače špičkové hodnoty [2]. 21
3.3 Blokové schéma sledovače špičkové hodnoty [2]. 22
3.4 Blokové schéma integrátoru pro vyhlazení zesilovacího činitele [2]. . . 23
4.1 Obecná struktura bloku řídící větve. 26
4.2 Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 1. . . 29
4.3 Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 4. . . 30
4.4 Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 6. . . 31
4.5 Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 8. . . 32
4.6 Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 10. . 33
4.7 Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 11. . 34
4.8 Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 14. . 35
4.9 Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 15. . 36
4.10 Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 17. . 37
5.1 Grafické rozhraní VST plugin modulu. 40
5.2 Průběhy signálů VST předvolby č. 1. 43
5.3 Průběhy signálů VST předvolby č. 2. 44
5.4 Průběhy signálů VST předvolby č. 3. 45
5.5 Průběhy signálů VST předvolby č. 4. 46
5.6 Průběhy signálů VST předvolby č. 5. 47
5.7 Průběhy signálů VST předvolby č. 6. 48
5.8 Průběhy signálů VST předvolby č. 7. 49
5.9 Průběhy signálů VST předvolby č. 8. 50
A.1 Graf signálových toků ve vypracování Matlab. 58

SEZNAM TABULEK

D.1 Vstupní parametry n funkce runexample(). 64
D.2 Vstupní parametry do_mode funkce runexample(). 64

ÚVOD

Řízení dynamiky zvukového signálu je jedna ze základních operací při zpracovávání zvukových signálů. Posluchači může poskytnout větší zřetelnost a konzistenci hudebních signálů, dokáže potlačit šumové složky a výrazně přispívá k nejen k ochraně výkonových zesilovačů a reproduktorů. Mezi čtyři základní dynamické efekty patří kompresor, expander, limiter a šumová brána. Úprava dynamiky signálu spočívá především v jeho útlumu v určitém rozmezí, což může snížit nebo zvýšit jeho dynamický rozsah.

Vhodnou kompresí a expanzí signálu je možné zvýšit odstup signálu od šumu při přenosu vysokofrekvenčního signálu. Kompresorem je možné redukovat dynamický rozsah hudební nahrávky, expander zase napomáhá odstranit nežádoucí ruchy. Šumová brána má velký potenciál při zpracování zvuku bicích nástrojů, a limiter najde své využití především na konci celého řetězce, kde zajišťuje ochranu před přebuzením.

Cílem této práce je návrh a realizace řešení této čtveřice efektů ve vývojovém prostředí Matlab a pomocí technologie VST (Virtual Studio Technology). Práce nabízí možnosti zobrazení průběhu signálu v jednotlivých blocích řídící větve efektu a umožňuje ukládání těchto signálů a grafů převodních charakteristik. Realizace ve vývojovém prostředí Matlab nabízí navíc připojení řídící větve k jinému zdroji signálu.

1 DYNAMIKA SIGNÁLŮ

1.1 Akustické signály

Dynamika zvukových signálů je určena jejím rozsahem, tj. maximální a minimální úrovní nebo hladinou udávanou v dB. Maximální hladina je zpravidla největší možná velikost zvukového signálu, kdežto minimální hladina je určena maximální hladinou šumu okolí. V dnešní době se lze nejčastěji setkat se zvukovými zdroji s rozsahy až 95 dB. Vezmeme-li v potaz, že se běžná hladinu šumu v tiché místnosti může ve městských prostředích pohybovat kolem 35 $\mathrm{dB_{SPL}}$ (Sound Pressure Level) [6], může signál s dynamikou 95 dB dosáhnout úrovně až 130 $\mathrm{dB_{SPL}}$, což je daleko za prahem bolesti lidského ucha.

Kdybychom se pokusili reprodukovat hudbu v tak velikém dynamickém rozsahu (nehledě na zvýšenou hladinu šumu vlivem vlastního šumu zesilovačů) došlo by při delším poslechu nejspíše k trvalému poškození sluchových orgánů posluchačů [9]. Proto je nutné snížit dynamiku takového zvukového signálu tak, aby byly i nejtišší pasáže hudby dostatečně slyšitelné a hlasité pasáže nepřekročily hladinu 100 $\mathrm{dB_{SPL}}$, kdy už při delším poslechu může nastat k fyzickému pocitu tlačení nebo bolesti v uších [9].

1.2 Číslicové signály

Dynamický rozsah číslicového signálu je dán jeho tzv. bitovou hloubkou. Pro 16bitové signály s pevnou řadovou čárkou bez znaménka je minimální hladina, často označována jako digitální nula, −96 dBFS, a maximální 0 dBFS. Pro 24bitový signál je pak minimální hladina −144 dBFS, 32bitový signál až −192 dBFS [9]. V dnešní době se většina zvukových nahrávek ukládá do stop s bitovou hloubkou 16 bitů. Dynamický rozsah číslicového řetězce je tedy často mnohem větší, než rozsah řetězce akustického. Např. velmi oblíbené studiové monitory Adam Audio A7X nabízí maximální hladinu 114 $\mathrm{dB_{SPL}}$ [1], což po odečtení šumu pozadí nechává prostor asi pro 85 dB zvukové informace.

1.3 Současný trend

Existují dva pohledy při zpracovávání hudebních signálů v dnešní době: snaha o maximální přirozenost nebo maximální hlasitost zvuku [4].

1.3.1 Maximální přirozenost

V ideálním případě nastává rovnost mezi dynamikou zdrojového signálu během nahrávání a dynamikou při reprodukci (poslechu). Nastává tak dojem věrné reprodukce zvukového tělesa, což s sebou však přináší řadu problémů. U signálu je znatelný velký poměr mezi jeho špičkovou a efektivní hodnotou. Při reprodukci takové hudby je pak kladen nárok zejména na kvalitu akustického řetězce, především reproduktory. Docílení takového stavu může být finančně velmi nákladné, nemluvě o tom, že současný trend je naprosto opačný. Poslech takové hudby v prostředí s vysokou úrovní šumu pak může znamenat i podstatnou ztrátu hudební informace.

1.3.2 Maximální hlasitost

V tomto modelu je kladen důraz na minimální dynamiku a maximální hlasitost záznamu. To umožní použití méně kvalitního akustického řetězce a zvýší odstup hudebního signálu od šumu pozadí, což se pozitivně projeví například při poslechu hudby v dopravních prostředcích. Efektivní hodnota signálu se přiblíží hodnotě špičkové, zvukový signál působí lepším a ucelenějším dojmem, ačkoli se nejedná o věrnou reprodukci.

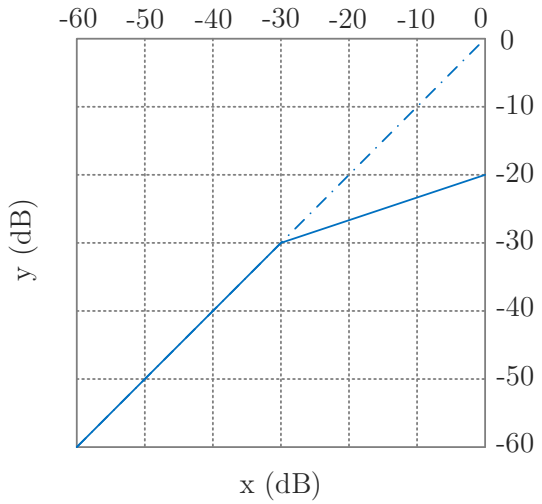
Snaha dosáhnout maximální hlasitost může narůst do extrémních případů, kdy je nahrávka na pokraji slyšitelného zkreslení. V posledních třech desetiletích se spousta umělců a zvukových techniků v hlasitosti nahrávek předhání, což vedlo ke vzniku ustáleného pojmu *Loudness War* (hlasitostní válka). Největší podíl na tomto boji má kapela Metallica. Jejich album Death Magnetic je totiž mnohými hudebními kritiky označováno za vítěze této války o bity [10].

2 EFEKTY PRO ÚPRAVU DYNAMIKY

Rozlišujeme dva základní efekty pro zpracování dynamiky signálu - kompresor a expander. S kompresorem dále souvisí efekt typu limiter, který je jeho hraničním případem. S efektem typu expander poté efekt typu šumová brána.

2.1 Kompresor

Efekt typu kompresor je takový efekt, který zeslabuje signály nad jeho prahovou úrovní podle určitého poměru. Pod touto úrovní zůstávají signály nezměněny. Zeslabující poměr kompresoru lze běžně nastavit mezi hodnotami $\langle 1, \infty \rangle$, přičemž pokud je nastaven na maximální hodnotu, lze o tomto efektu mluvit jako o méně přesném limiteru. To je z důvodu toho, že kompresor, narozdíl od limiteru, používá sledovač obálky efektivní hodnoty signálu, kdežto limiter používá sledovač obálky hodnoty špičkové. Nastaví-li se tedy zeslabovací poměr, neboli *ratio*, na hodnotu $3:1$, bude veškerý signál nad rozhodovací úrovní zeslaben na $1/3$ své původní velikosti. Dynamický rozsah signálu bude tedy zmenšen.



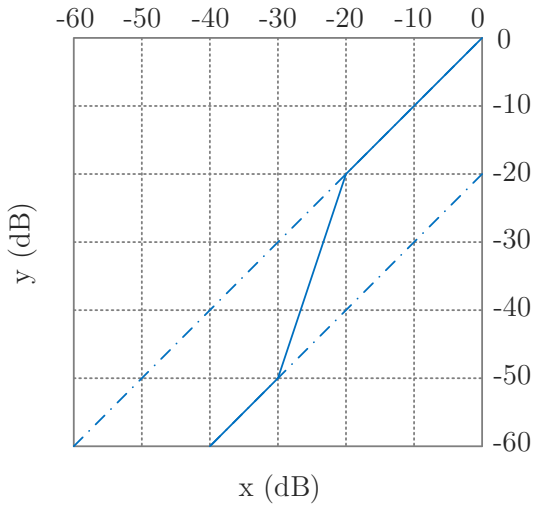
Obr. 2.1: Převodní charakteristika kompresoru s prahovou úrovní -30 dB a poměrem $3:1$.

Kompresor je nejpoužívanějším ze čtveřice zpracovaných efektů. Může sloužit jak ke kompresi rychlých a razantních změn (např. úderu na buben), tak k zušlechtění již hotové hudební nahrávky, kde je jeho odezva pomalá, a poměr nepřesáhne hodnoty větší než $1{,}5:1$.

Kompresor se taktéž používá např. při vysokofrekvenčním přenosu signálů, kdy je nutné během transportu signálu oddělit signál od šumu. Signál je tedy před vysláním zkomprimován, a tím vznikne jeho větší odstup od šumu. Na přijímači je pak signál expandován v obráceném poměru.

2.2 Expander

Expander je typ dynamického efektu, který zeslabuje signály pod svou prahovou úrovní, a ty, jež jsou nad ní, nechává nezměněné. Pracuje prakticky opačným způsobem než kompresor. Pokud je jeho poměr nastaven na hodnotu 1 : 3, je veškerý vstupní signál pod jeho prahovou úrovní zeslaben na třetinu. Dynamický rozsah signálu bude tedy zvětšen. Expander bývá často doplněn o parametr maximálního útlumu, často označovaného slovem *range* nebo *depth*.



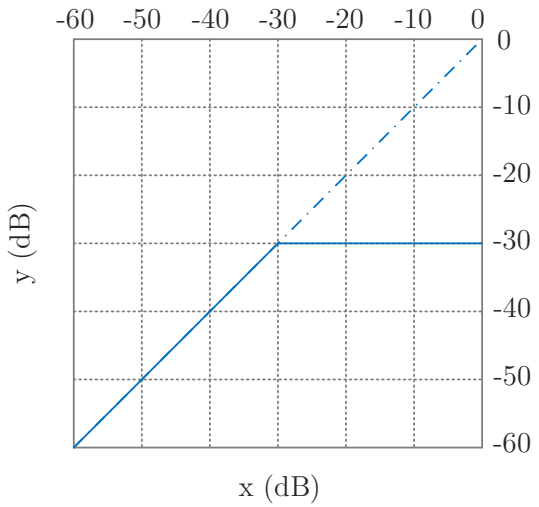
Obr. 2.2: Převodní charakteristika expanderu s prahovou úrovní -20 dB, poměrem 1 : 3 a maximálním útlumem -20 dB.

Expander se používá především k očištění signálů od nežádoucích ruchů, a to ať už u mluveného slova, kdy je nastavena velká časová odezva tak, aby skoky v úrovních šumu pozadí nebyly tak velké, tak např. při zpracování zvuku bicích nástrojů, kdy je téměř nutné potlačit přeslechy kanálů jednotlivých bubnů, aby mohly být dále zpracovány.

Expander může ve většině případů sloužit jako náhrada nebo přirozenější alternativa šumové brány. Ve světě analogových efektů ho lze nejčastěji najít právě jako rozšíření efektu šumové brány, kde je možné vhodným nastavením dosáhnout obou efektů.

2.3 Limiter

Limiter je efekt jehož převodní charakteristika je nastavena tak, že veškeré signály nad jeho prahovou úrovní jsou zeslabeny v poměru $\infty : 1$, a tedy v ideálním případě veškeré takové signály nemohou přesáhnout jeho rozhodovací úroveň. Všechny signály pod jeho rozhodovací úrovní zůstávají nezměněny. Limiter tedy, obdobně jako kompresor, snižuje dynamický rozsah signálu. Na rozdíl od kompresoru používá sledovač obálky špičkové hodnoty, což umožňuje přesnější a rychlejší odezvu na vstupní signál.



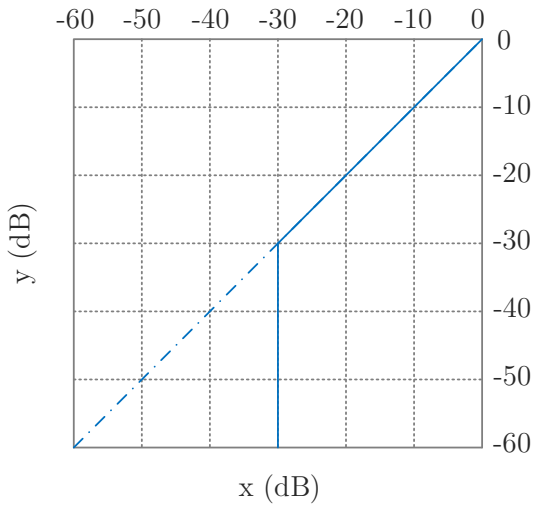
Obr. 2.3: Převodní charakteristika limiteru s prahovou úrovní -30 dB.

Zvukový efekt limiter se používá převážně pro ochranu před přebuzením zesilovačů nebo jiných elektronických zařízení. Při zpracování hudby pak limiter najde své nepostradatelné uplatnění úplně na konci celého mixovacího řetězce, kdy je potřeba záznam zesílit, ale špičkové hodnoty signálu to již nedovolí (docházelo by ke zkreslení signálu). Limiter se tedy nastaví tak, aby signál nemohl přebudit výstupní A/D převodník (respektive nepřesáhl digitální úroveň 0 dBFS).

Limitery často uživateli nabízí pouze volitelný parametr prahové úrovně, nebo zesílení celého vstupního signálu s pevně nastaveným prahem na 0 dB. Najdou se i takové limitery, převážně ochranné, které nenabízí vůbec žádná nastavení.

2.4 Šumová brána

Šumová brána je speciálním případem efektu expander, primárně sloužící pro úplné potlačení signálů pod prahovou úrovní, které jsou zeslabeny v poměru $1 : \infty$. Tento efekt lze použít k absolutnímu potlačení šumu, kdy společně s vhodnou volbou parametrů *attack*, *release* a *hold* může např. pomoci eliminovat nežádoucí šumy v tichých pasážích řečového signálu.



Obr. 2.4: Převodní charakteristika šumové brány s prahovou úrovní -30 dB.

Šumová brána je krajním případem expanderu, a uplatnění nalézá většinou v analogových řešeních, kde není příliš prostoru pro nastavení více parametrů. Často se lze setkat s šumovou bránou, která je ovládána pouze dvěma parametry, a to je nastavení prahu a rychlosti reakce.

Neodborné nastavení šumové brány však může vést k velmi nežádoucím výsledkům. Typické je, že během tichého zpěvu dochází ke krátkým výpadkům, které pak spíše nabuzují dojem vadné techniky (kabeláže).

2.5 Efekty s kmitočtovými filtry v řídící větvi

Dynamické filtry bývají často předřazeny filtry horní a dolní propusti. To nabízí možnost komprimovat signál pouze podle změn v určitém pásmu, což se může velmi pozitivně projevit při komprimaci signálů s velikým frekvenčním rozsahem.

2.5.1 Deesser

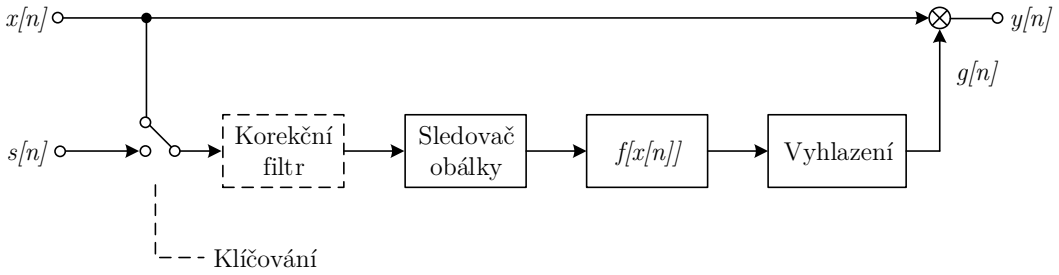
Deesser je kmitočtově řízený dynamický filtr. Jedná se o speciální případ kompresoru s velmi krátkou časovou odezvou, který je klíčován úzkým kmitočtovým filtrem v rozmezí 3 kHz až 8 kHz. Jeho hlavním účelem je potlačení a eliminace sykavek, které nejčastěji doprovázejí hlásku *s*.

2.5.2 Vícepásmový kompresor

Vícepásmový kompresor je efekt, který signál přímé větve rozdělí na několik kmitočtových pásem, které dále samostatně zpracuje. Je zapojen jako paralelní soustava kompresorů, jejichž kmitočtové filtry se nachází v přímé větvi před odbočením do jednotlivých řídících větví. Tento typ kompresoru umožňuje upravení dynamiky jednotlivých kmitočtových pásem (kterých může být libovolně mnoho), a tak poskytuje větší kontrolu nad zpracovaným signálem a umožňuje preciznější úpravu v celém spektru vstupního.

3 ÚPRAVA DYNAMIKY SIGNÁLU

Úprava dynamiky signálu je v číslicových systémech realizována blokem sestávajícím z přímé větve a řídící větve, která obsahuje veškeré výpočetní bloky. V řídící větvi je zařazen sledovač obálky, dále blok provádějící výpočet zesilovacího činitele, který je následně vyhlazen integrátorem. Tato výsledná funkce pak ovládá VCA (Voltage Controlled Amplifier - Napěťově řízený zesilovač), který modifikuje zesílení přímé větve s hudebním signálem. Řídící větev může být napojena na externí zdroj signálu, případně může být předřazena kmitočtovým filtrem.



Obr. 3.1: Blokové schéma zapojení efektu dynamiky.

3.1 Sledovač obálky

Sledovač obálky převádí skutečný signál na odhad jeho obálky. V efektech pro úpravu dynamiky se používají dva typy těchto sledovačů - sledovač obálky špičkové hodnoty a sledovač obálky efektivní hodnoty. Tuto kapitolu podrobněji popisuje [2] a [11].

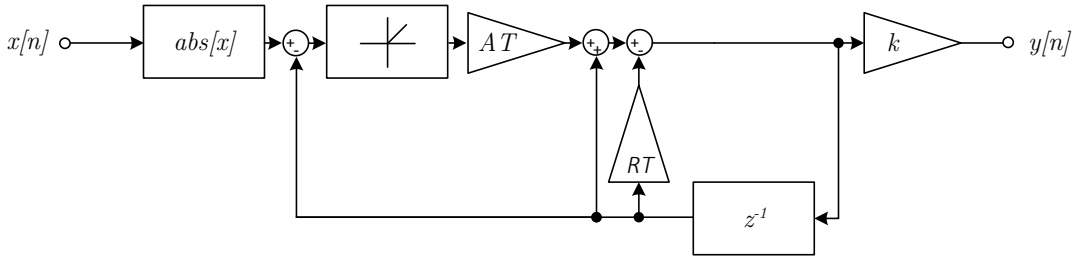
3.1.1 Sledovač obálky špičkové hodnoty

Tento sledovač převádí signál na jeho absolutní hodnotu a poté v závislosti na předchozím vzorku upravuje hodnotu vzorku aktuálního: pokud signál narůstá, použije se aktuální (vyšší) hodnota, pokud signál klesá, systém poníží předchozí špičkovou hodnotu signálu. Konstanty AT a RT určují, jak rychle má špičková hodnota růst a klesat. Rovnice je citována ze zdroje [2].

$$y[n] = \begin{cases} AT \cdot |x[n]| + (1 - AT - RT) \cdot y[n-1], & |x[n] - y[n-1]| > 0 \\ (1 - RT) \cdot y[n-1], & |x[n] - y[n-1]| < 0 \end{cases} \tag{3.1}$$

Konstanty AT a RT lze určit pomocí následujících vztahů [2], kde t_{at} je časová konstanta náběhu, t_{rt} je časová konstanta poklesu a f_{s} je vzorkovací frekvence.

$$\begin{aligned} AT &= 1 - \mathrm{e}^{\frac{-2.2}{t_{at} \cdot f_s}} \\ RT &= 1 - \mathrm{e}^{\frac{-2.2}{t_{rt} \cdot f_s}} \end{aligned} \tag{3.2}$$



Obr. 3.2: Blokové schéma sledovače špičkové hodnoty [2].

Přesnost sledování závisí obzvláště na správně zvolených parametrech AT a RT. Ty musí být zvoleny tak, aby byl sledovač schopen reagovat v celém akustickém pásmu, respektive v celém spektru, které obsahuje hudební signál. Čím větší jsou konstanty nastaveny, tím je přesnost odhadu menší. Vzhledem k tomu, že sledovač obsahuje jeden zpožďovací blok, je nutné nastavit počáteční podmínku. Tu je vhodné nastavit na 0, případně, pokud si to žádají okolnosti, na hodnotu v intervalu $\langle 0{,}1 \rangle$.

3.1.2 Sledovač obálky efektivní hodnoty

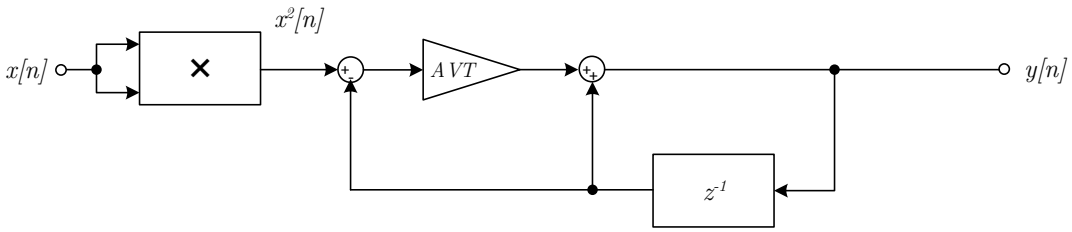
Sledovač efektivní hodnoty signálu převádí signál na jeho druhou mocninu, která je poté integrována. Konstanta AVT nastavuje dobu průměrování, a tedy přesnost filtru. Tento sledovač nedetekuje odhadovanou hodnotu signálu, ale jeho druhou mocninu. Systém je popsán rovnicí [2]:

$$y[n] = AVT \cdot x^2[n] + (1 - AVT) \cdot y[n-1]. \tag{3.3}$$

Výpočet časové konstanty průměrování AVT je pak prakticky totožný s předchozími časovými konstantami:

$$AVT = 1 - \mathrm{e}^{\frac{-2.2}{t_{avt} \cdot f_s}}. \tag{3.4}$$

Protože odmocnění je výpočetně náročná operace, je téměř nezbytně nutné, aby systémy následující po tomto bloku byly upraveny pro výpočet s druhou mocninou.



Obr. 3.3: Blokové schéma sledovače špičkové hodnoty [2].

3.2 Výpočet zesilovacího činitele

Poté, co je získána obálka signálu, je nutné provést výpočet zesilovacího činitele, který slouží jako řídící signál pro napěťově řízený zesilovač.

3.2.1 Výpočet zesilovacího činitele kompresoru

Pro efekty kompresorového typu lze použít následující vztah [5]:

$$f[n] = \begin{cases} slope \cdot (thr - env[n]), & env[n] > thr \\ 0, & env[n] <= thr, \end{cases} \tag{3.5}$$

kde *slope* je logaritmovaná míra strmosti filtru, *thr* logaritmovaná rozhodovací úroveň filtru a *env* logaritmovaná obálka vstupního signálu. Vstupem této funkce je tedy logaritmus obálky signálu přímé větve. V případě, že se pro výpočet obálky použije sledovač efektivní hodnoty signálu, je pro správný výsledek důležité obálku po logaritmování vydělit dvěma, neboť se v ní lineární veličina vyskytuje v druhé mocnině.

Parametr *slope* lze vypočítat z parametru *ratio* pomocí:

$$slope = 1 - \frac{1}{ratio}. \tag{3.6}$$

3.2.2 Výpočet zesilovacího činitele expanderu

Pro efekty expanderového typu s parametrem *range* platí následující vztah:

$$f[n] = \begin{cases} slope \cdot (thr - env[n]), & env[n] <= thr \wedge env[n] > thr_b \\ range, & env[n] <= thr \wedge env[n] <= thr_b \\ 0, & env[n] > thr, \end{cases} \tag{3.7}$$

kde *slope* je logaritmovaná míra strmosti filtru, *thr* logaritmovaná rozhodovací úroveň filtru, *range* je logaritmovaná míra maximálního útlumu, *env* logaritmovaná obálka vstupního signálu a thr_{b} je logaritmovaná dolní rozhodovací úroveň filtru, pod kterou je signál zeslabován pouze o hodnotu *range*.

K výpočtu dolní rozhodovací úrovně thr_{b} pak slouží vztah:

$$thr_b = \frac{thr - \frac{range+thr}{ratio}}{slope} - range. \tag{3.8}$$

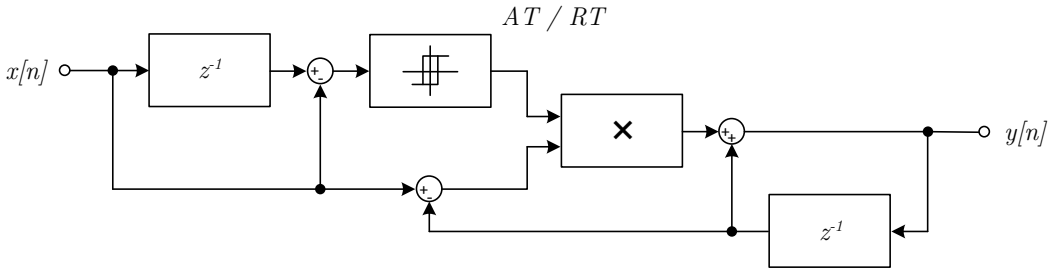
3.3 Vyhlazení zesilovacího činitele

Funkce zesilovacího činitele může obsahovat skokové změny (především při výpočtu zesilovacího činitele šumové brány), a tak je nutné tuto funkci vyhladit tak, aby na vstupním signálu nedocházelo ke generování nových harmonických složek, resp. aby těchto složek bylo co nejméně.

Vhodným nastavením časových konstant filtrů lze pak docílit různé odezvy na změny signálu - je tedy např. možné nastavit filtr tak, aby reagoval jen na pomalé změny signálu, což se hodí u komprese hudebních signálů, zvláště při masteringu [7].

3.3.1 Dynamický filtr signálu

K vyhlazení skokových změn signálu slouží integrátor s dvěma časovými konstantami. Jedna slouží pro náběžnou a druhá pro sestupnou hranu signálu.



Obr. 3.4: Blokové schéma integrátoru pro vyhlazení zesilovacího činitele [2].

V případě, že je obálka vstupního signálu vyšší než je rozhodovací úroveň, je konstantou tohoto integrátoru hodnota AT, jedná se tedy o náběžnou hranu. V opačném případě se jedná o hranu sestupnou, a konstantou je hodnota RT. Výpočet je převzán z [2].

$$f[n] = \begin{cases} (1 - AT) \cdot f[n-1] + AT \cdot g[n], & g[n] >= f[n-1] \\ (1 - RT) \cdot f[n-1] + RT \cdot g[n], & g[n] < f[n-1] \end{cases} \tag{3.9}$$

Funkce *g[n]* je zesilovací činitel vypočtený v předchozím bloku po odlogaritmování.

Je velice důležité uvědomit si, že při filtraci zesilovacího činitele kompresorového typu jsou časové konstanty AT a RT navzájem zaměněny. Nástupná hrana kompresoru (komprimace) má v logarytmickém měřítku klesající charakter. Pro výpočet dynamického filtru kompresoru lze tedy užít vztah:

$$f[n] = \begin{cases} (1 - RT) \cdot f[n-1] + RT \cdot g[n], & g[n] >= f[n-1] \\ (1 - AT) \cdot f[n-1] + AT \cdot g[n], & g[n] < f[n-1]. \end{cases} \tag{3.10}$$

3.3.2 Dynamický filtr s parametrem hold

Předchozí integrátor je možné doplnit o časovou konstantu *hold* (HT) - dobu, po kterou filtr setrvává na své hodnotě nehledě na pokles vstupního signálu [5]. Při filtraci zesilovacího činitele kompresorového typu hodnota setrvává vždy, při filtraci zesilovacího činitele expanderového typu je konstanta HT přítomna pouze v případě, že je expander plně otevřen.

Otevření expanderu je v ideálním případě stav, kdy je hodnota zesilovacího činitele rovna 1. Vyhlazování však může produkovat hodnoty velmi blízké hodnotě 1, a tak je nutné nastavit práh, kdy je expander považován za otevřený. Tento práh by určitě neměl přesáhnout rozlišovací hranici ucha (1 dB). V této práci je prahová hodnota nastavena na $0,982$, což odpovídá přibližně $-0{,}15$ dB.

Pro výpočet dynamického filtru s parametrem hold pro efekt expanderového typu platí rovnice:

$$f[n] = \begin{cases} (1 - AT) \cdot f[n-1] + AT \cdot g[n], \\ \qquad\qquad g[n] >= f[n-1] \\ f[n-1], \\ \qquad\qquad g[n] < f[n-1] \wedge (f[n-1] > 0.982 \wedge u_{cnt} < h_{samp}) \\ (1 - RT) \cdot f[n-1] + RT \cdot g[n], \\ \qquad\qquad g[n] < f[n-1] \wedge (f[n-1] \leq 0.982 \vee u_{cnt} >= h_{samp}). \end{cases} \tag{3.11}$$

Pro výpočet dynamického filtru s parametrem hold pro efekt kompresorového

typu platí rovnice:

$$f[n] = \begin{cases} (1 - AT) \cdot f[n-1] + AT \cdot g[n], & g[n] <= f[n-1] \\ f[n-1], & g[n] > f[n-1] \wedge u_{cnt} < h_{samp} \\ (1 - RT) \cdot f[n-1] + RT \cdot g[n], & g[n] > f[n-1] \wedge u_{cnt} >= h_{samp}. \end{cases} \tag{3.12}$$

Hodnota u_{cnt} uvádí aktuální počet vzorků, kdy je brána otevřena ačkoli signál je pod prahovou úrovní, funkce *g[n]* je zesilovací činitel vypočtený v předchozím bloku. Konstanta h_{samp} uvádí maximální počet vzorků signálu, po kterou filtr setrvává na své hodnotě nehledě na pokles vstupního signálu. Je udán rovnicí:

$$h_{samp} = HT \cdot f_s, \tag{3.13}$$

kde f_{s} je vzorkovací kmitočet.

4 REALIZACE V PROSTŘEDÍ MATLAB

Výpočetní software Matlab byl vybrán za účelem zjednodušení práce s vektory vzorků jednotlivých signálů. Umožňuje relativně jednoduše zobrazovat průběhy hodnot a poskytuje dostatečně velké množství funkcí, které jsou potřeba pro práci se signály. Vypracované zadání je pak algoritmem pracujícím dávkově, který zpracuje celý signál a poté zobrazí jeho výstupy. Není tedy možné, na rozdíl od VST zásuvných modulů, upravovat jeho jednotlivé parametry v reálném čase.

Zdrojový kód byl zkomponován v prostředí Matlab verze R2013b. Kvůli požadavkům na kompatibilitu byl upraven a testován na verzi R2010a a následně i na verzi R2015a. Výsledný kód je tedy kompatibilní s nejčastěji používanými verzemi software Matlab.

4.1 Struktura programu

Všechny realizované efekty mají navrženou strukturu tak, aby co nejpřesněji kopírovaly schéma signálových toků jejich analogových variant. Jednotlivé body zpracování je tak možné přemostit, či vyměnit jiným algoritmem. To si s sebou však nese nárůst výpočetní doby, protože je nutné některé kroky opakovat.

Kód je napsán i okomentován v anglickém jazyce. V komentářích pak lze najít vysvětlivky k jednotlivým krokům, stejně jako popis vstupně-výstupních parametrů funkcí. Seznam souborů s krátkým popisem je uveden v příloze A na straně 56.

4.1.1 Struktura řídící větve

Řídící větev byla sestavena tak, aby kopírovala strukturu analogového obvodu. Sestává ze tří hlavních prvků. První blok provádí výpočet obálky signálu, druhý pak výpočet zesilovacího činitele a poslední blok se stará o vyhlazení zesilovacího činitele. Mezi bloky může probíhat konverze signálu mezi lineární a logaritmickou formou.

Obr. 4.1: Obecná struktura bloku řídící větve.

Jednotlivé funkce těchto bloků pak vykonávají následující:

`envelopepeak()`	Výpočet obálky špičkové hodnoty signálu
`enveloperms()`	Výpočet obálky efektivní hodnoty signálu
`gainstaticcompressor()`	Výpočet zesilovacího činitele kompresorového typu
`gainstaticexpander()`	Výpočet zesilovacího činitele expanderového typu
`gaindynamicar()`	Vyhlazení zesilovacího činitele
`gaindynamicarh()`	Vyhlazení zesilovacího činitele s parametrem hold

4.1.2 Funkce konkrétních efektů

Systém obsahuje čtyři hlavní funkce, které představují konkrétní efekty pro úpravu dynamiky signálu:

`compressor()`	Efekt typu kompresor
`expander()`	Efekt typu expander
`limiter()`	Efekt typu limiter
`noisegate()`	Efekt typu šumová brána

Všechny funkce obsahují stejné výstupní rozhraní. Jejich návratové hodnoty jsou vždy: výstup efektu (hlavní větve), obálka signálu řídící větve, zesilovací činitel, vyhlazený zesilovací činitel. Vstupní rozhraní se u jednotlivých funkcí liší, zpravidla platí že první čtyři argumenty jsou: vstupní signál, signál řídící větve, vzorkovací kmitočet, prahová úroveň filtru.

4.1.3 Funkce pro zjednodušení práce s efekty

Pro zjednodušení práce a vykreslení signálu jednotlivých efektů slouží čtveřice funkcí s prefixem `do`, které zjišťují načtení vstupních souborů, zavolání příslušného efektu, jeho vykreslení a případné uložení nebo přehrání dat. První dva parametry těchto funkcí jsou parametr `do_mode` a název souboru se vstupními daty. Poslední parametr je volitelným parametrem názvu souboru s daty, které reprezentují vstupní signál řídící větve. Parametr `do_mode` určuje akci po vypočítání a vykreslení signálu, a jeho hodnoty jsou jsou popsány v tabulce D.2.

`docompressor()`	Načte, zavolá a vykreslí data kompresoru
`doexpander()`	Načte, zavolá a vykreslí data expanderu
`dolimiter()`	Načte, zavolá a vykreslí data limiteru
`donoisegate()`	Načte, zavolá a vykreslí data šumové brány

4.2 Spuštění vzorových příkladů

Vstupní uživatelské rozhraní nabízí spuštění několika vzorových příkladů voláním funkce `runexample(n, do_mode)`, kde `n` je číslo vzorového příkladu a `do_mode` je volitelný atribut akce. Parametry `n` mohou nabývat hodnot 1 až 17, přičemž jsou seřazeny do skupin po čtyřech. Jeden efekt tedy nabízí zobrazení čtyř vzorových hudebních signálů. Příklad s číslem 17 je speciálním, protože obsahuje jiný signál v přímé a řídící větvi. Jedná se o kompresor v tvz. zapojení *ducking*. Hodnoty parametru `do_mode` nabízí možnosti přehrání a uložení výsledné stopy. Vložení nulového (žádného) parametru znamená pouze zobrazení grafů signálu v jednotlivých bodech efektu.

Seznam zvukových stop používaných ve vzorových příkladech lze najít v příloze C.1 na straně 63.

Při volbě parametru `do_mode` většího než 2 se společně s vykreslením grafů také zapíší výstupy do souboru. Výstupní zvuková stopa se ukládá ve formátu WAV (Waveform audio file format) do souboru `output.wav`. Současně se zvukovou stopou se ukládají zobrazené grafy do souborů `output_characteristics.png` a `output_sidechain.png`. V případě, že je nutné, aby výstupní obrazové soubory byly vektorové, lze formát jednoduše přepnout ve funkci `evaldomode()`.

Popis vstupních parametrů této funkce je uveden v tabulkách D.1 a D.2 na straně 64. Graf signálových toků je přiložen v příloze A.1 na straně 58.

4.3 Výsledky práce

V této kapitole je uvedeno několik vzorových příkladů, které lze spustit voláním funkce `runexample()`. Zobrazené průběhy jsou exportem této funkce.

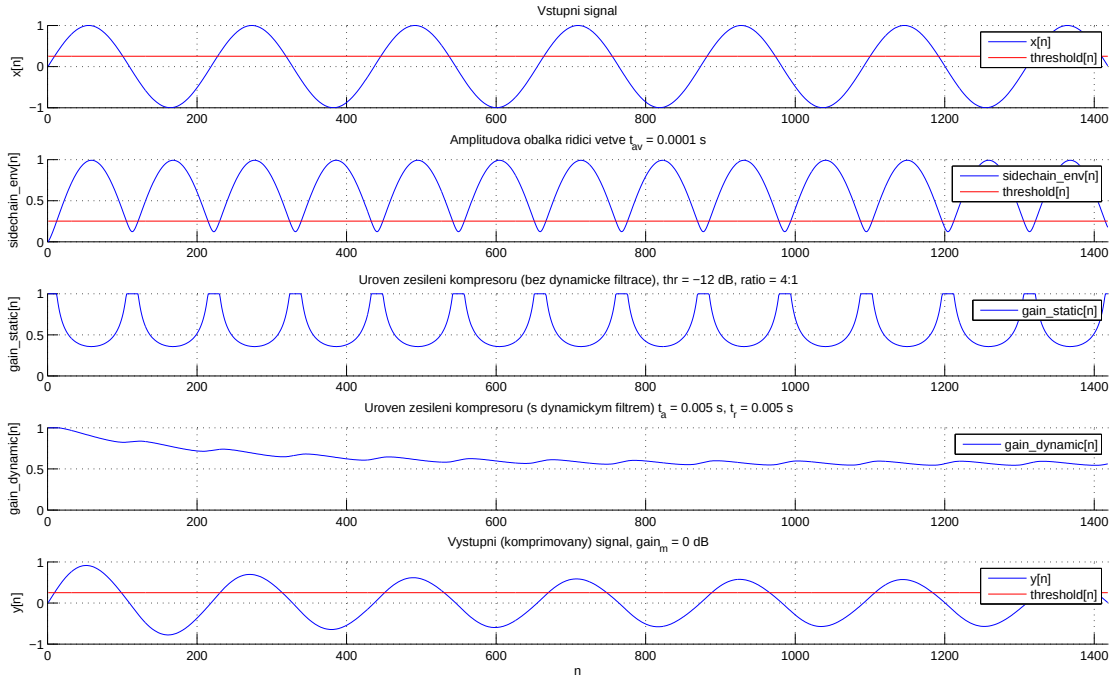
4.3.1 Kompresor

Vzorový příklad číslo 1

Tento příklad obsahuje zvukovou stopu konstantní frekvence a amplitudy. Na tomto příkladu je demonstrován postupný náběh kompresoru a reakce na změnu úrovně vstupního signálu. Poté, co je časový průběh sledovače a dynamického filtru ustálen, efekt zeslabuje veškerý vstupní signál nad jeho rozhodovací úrovní na 1 / 4 původní úrovně.

Parametry tohoto příkladu jsou:

Prahová úroveň:	$-12\,\mathrm{dB}$
Poměr:	$4:1$
Průměrování:	$0{,}1\,\mathrm{ms}$
Attack:	$5\,\mathrm{ms}$
Release:	$5\,\mathrm{ms}$



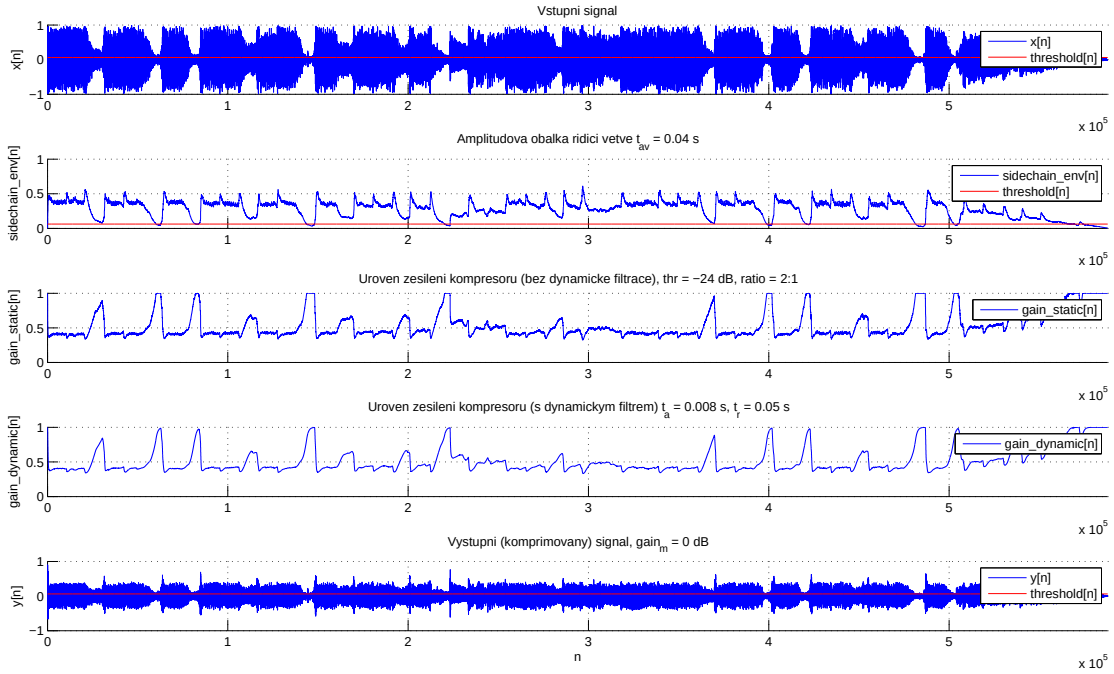
Obr. 4.2: Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 1.

Vzorový příklad číslo 4

Příklad demonstrující použití efektu typu kompresor na reálných hudebních signálech je nastaven tak, aby vyhlazoval rozdíl mezi tichými a hlasitými pasážemi skladby. Na první pohled si lze všimnout několika zbylých špiček. Ty jsou způsobeny nedostatečně rychlou odezvou sledovače obálky. Kdyby se vstupní signál o několik desítek až stovek mikrosekund opozdil, dokázala by řídící větev předvídat změny v signálu, a tak by se mohla komprimace celého signálu stát více přirozenou a méně slyšitelnou. Vzniklé špičky by pak nemusely vůbec vzniknout.

Parametry tohoto příkladu jsou:

Prahová úroveň:	$-24\,\mathrm{dB}$
Poměr:	$2:1$
Průměrování:	$40\,\mathrm{ms}$
Attack:	$8\,\mathrm{ms}$
Release:	$50\,\mathrm{ms}$



Obr. 4.3: Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 4.

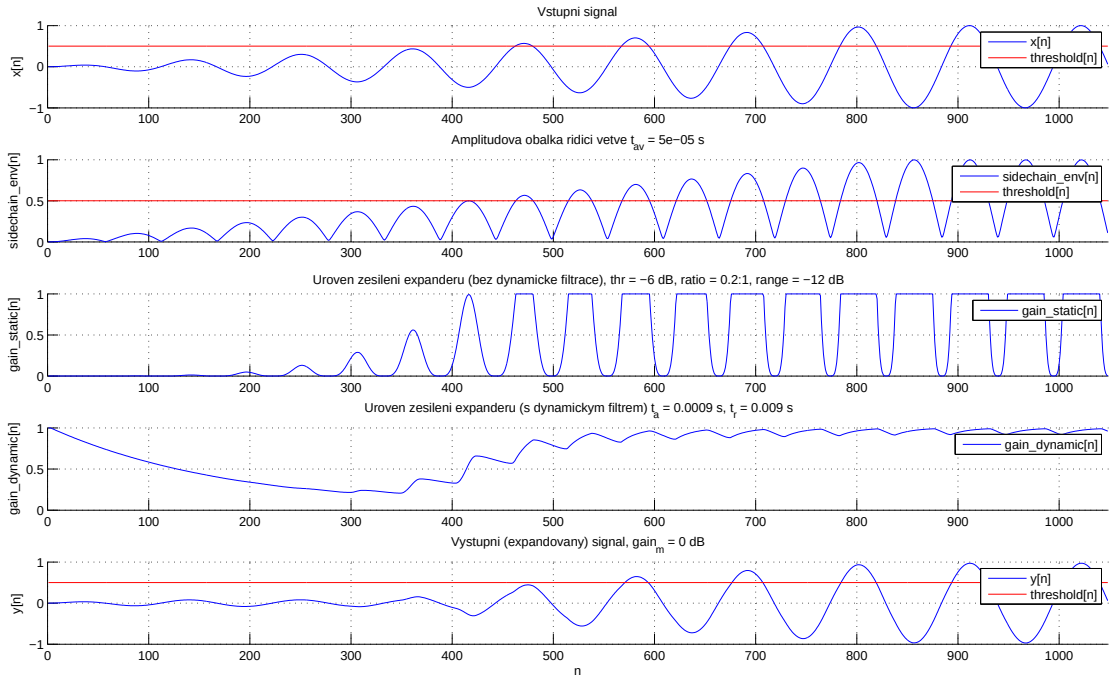
4.3.2 Expander

Vzorový příklad číslo 6

Příklad číslo 6 demonstruje použití expanderu na sinusové vlně postupného náběhu. Je zcela zřetelné, že signál, který je pod prahovou úrovní efektu je zeslaben, konkrétně tedy v poměru $1:5$. Časové konstanty jsou vhodně nastaveny tak, aby nedocházelo k deformaci vlny v oblastech nad prahovou úrovní.

Parametry tohoto příkladu jsou:

Prahová úroveň:	$-6\,\mathrm{dB}$
Poměr:	$1:5$
Range:	$-12\,\mathrm{dB}$
Průměrování:	$0{,}05\,\mathrm{ms}$
Attack:	$0,\mathrm{ms}$
Release:	$9\,\mathrm{ms}$
Hold:	$0{,}05\,\mathrm{ms}$



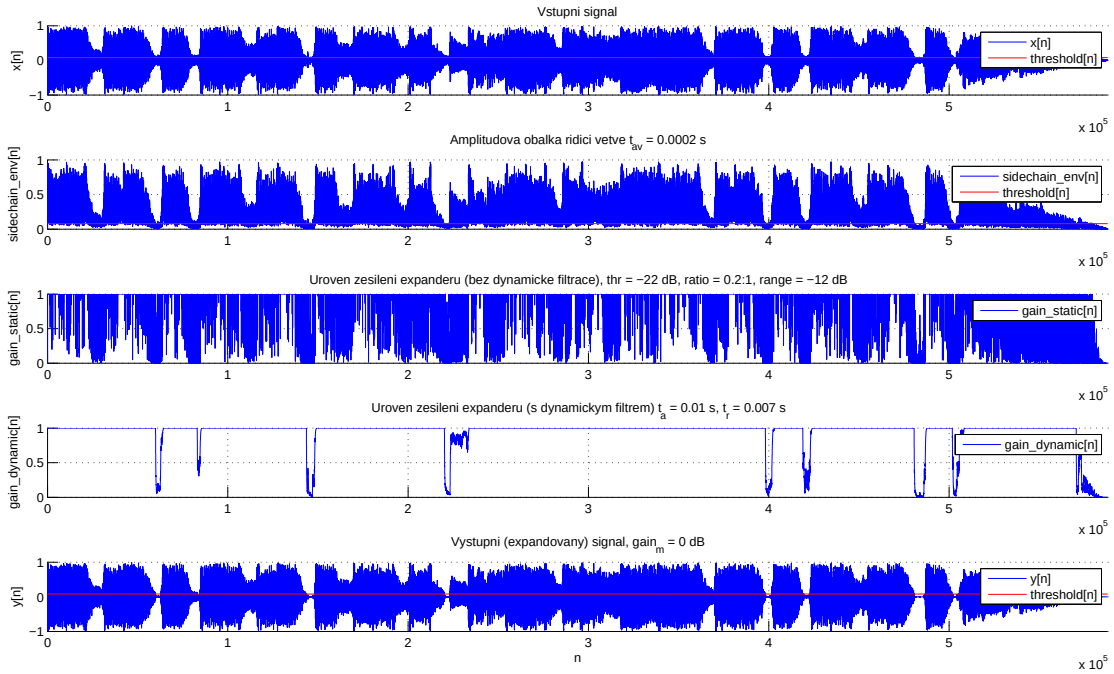
Obr. 4.4: Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 6.

Vzorový příklad číslo 8

Zvukový efekt expander je u příkladu s číslem 8 využit pro zeslabení tichých pasáží hudební ukázky. Zeslabení signálu má maximální hranici $-12\,\mathrm{dB}$, což se jeví jako přirozenější, než zeslabení do digitální nuly. Signál, jež je nad touto hranicí, zůstává nezměněn.

Parametry tohoto příkladu jsou:

Prahová úroveň:	$-22\,\mathrm{dB}$
Poměr:	$1:5$
Range:	$-12\,\mathrm{dB}$
Průměrování:	$0{,}05\,\mathrm{ms}$
Attack:	$0{,}9\,\mathrm{ms}$
Release:	$9\,\mathrm{ms}$
Hold:	$0{,}05\,\mathrm{ms}$



Obr. 4.5: Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 8.

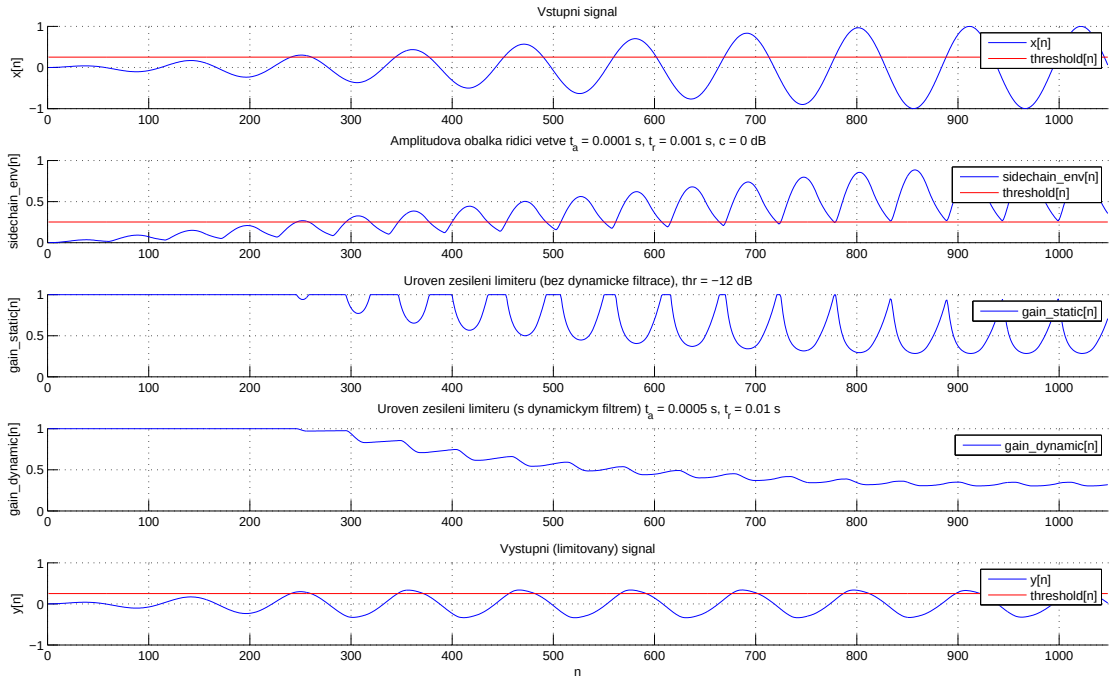
4.3.3 Limiter

Vzorový příklad číslo 10

Vzorový příklad s číslem 10 je nastaven na zpracování signálu zvukové vlny konstantní frekvence s postupným náběhem. Na zobrazených grafech lze zřetelně vidět, že limiter začal signál zeslabovat až od 300. vzorku. Po ustálení efekt pracuje přesně dle předpokladů – zatlumí vstupní signál tak, aby úroveň signálu nepřesáhla prahovou úroveň efektu.

Parametry tohoto příkladu jsou:

Prahová úroveň:	$-12\,\mathrm{dB}$
Attack sledovače:	$0{,}1\,\mathrm{ms}$
Attack filtru:	$0{,}5\,\mathrm{ms}$
Release sledovače:	$1\,\mathrm{ms}$
Release filtru:	$10\,\mathrm{ms}$



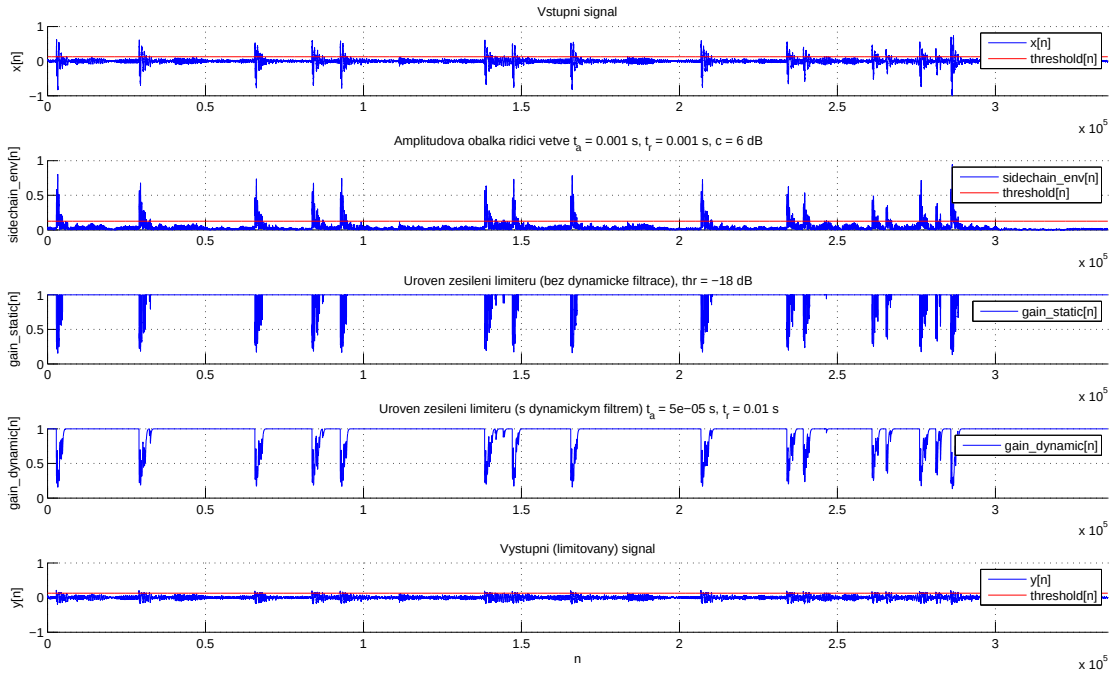
Obr. 4.6: Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 10.

Vzorový příklad číslo 11

Vzorový příklad s číslem 11 obsahuje zvukovou stopu velkého bubnu se spoustou přeslechů způsobených převážně basovou kytarou a malým bubínkem. Efekt je nastaven tak, aby snížil úroveň úderů velkého bubnu. Vyobrazené průběhy ukazují, že limiter reaguje na údery s velkou přesností.

Parametry tohoto příkladu jsou:

Prahová úroveň:	−18 dB
Attack sledovače:	1 ms
Attack filtru:	0,05 ms
Release sledovače:	1 ms
Release filtru:	10 ms
Korekce obálky:	6 dB



Obr. 4.7: Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 11.

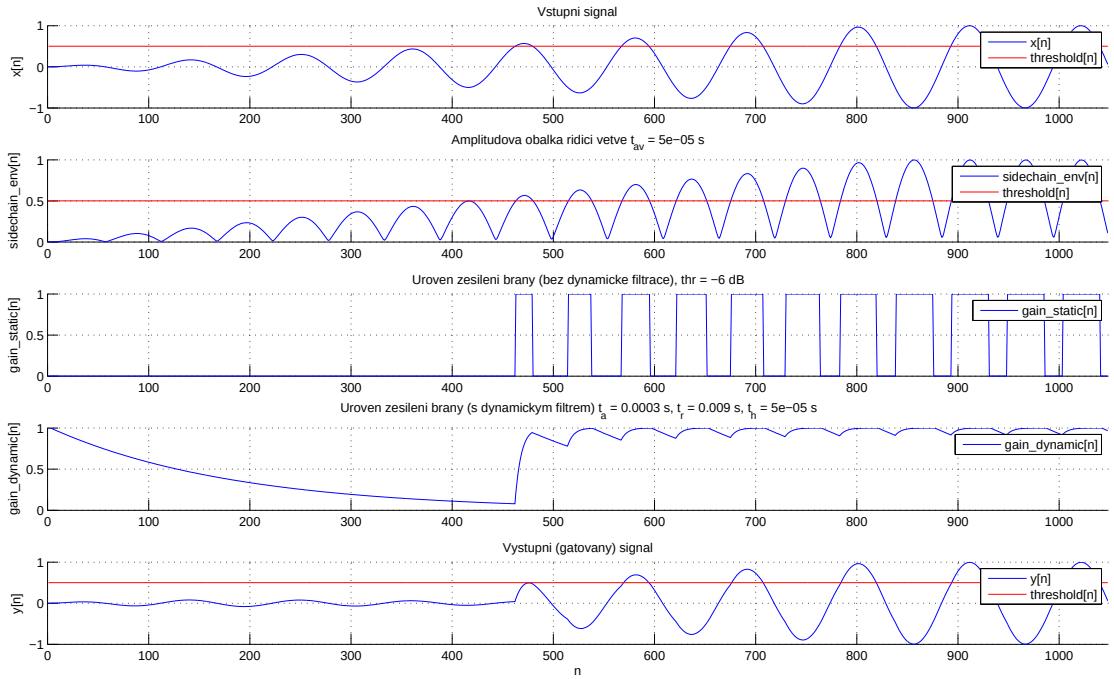
4.3.4 Šumová brána

Vzorový příklad číslo 14

Příklad s číslem 14 zobrazuje signál s postupným náběhem, kdy je brána nastavena tak, aby opozdila a zkrátila náběh signálu. Vzhledem k vhodně nastaveným časovým konstantám je pak výsledný signál viditelně hladký a není na něm zjevná žádná skoková změna.

Parametry tohoto příkladu jsou:

Prahová úroveň:	−6 dB
Průměrování:	0,05 ms
Attack:	0,3 ms
Release:	9 ms
Hold:	0,05 ms



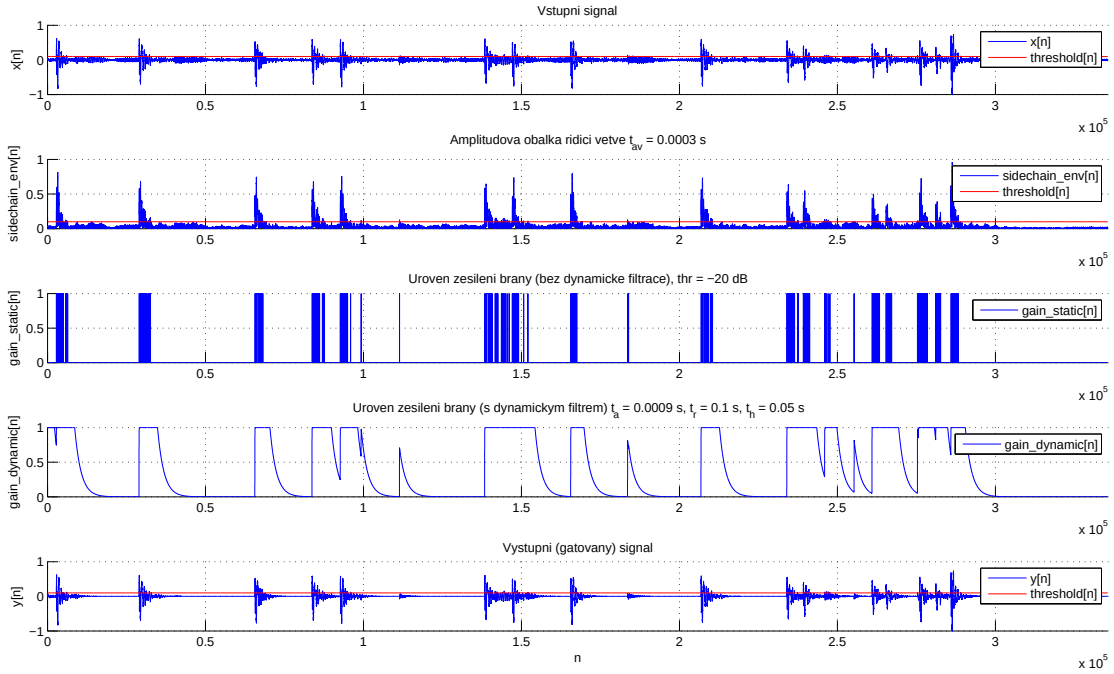
Obr. 4.8: Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 14.

Vzorový příklad číslo 15

V posledním z čtveřicí příkladů je zobrazena část zvukové stopy velkého bubnu s výraznými přeslechy, na které je názorně zobrazena funkce šumové brány. Její rozhodovací úroveň je nastavena tak, aby potlačila veškeré nežádoucí přeslechy aniž by jakkoli zasáhla do zvuku samotného bubnu.

Parametry tohoto příkladu jsou:

Prahová úroveň:	$-20\,\mathrm{dB}$
Průměrování:	$0{,}3\,\mathrm{ms}$
Attack:	$0{,}9\,\mathrm{ms}$
Release:	$100\,\mathrm{ms}$
Hold:	$50\,\mathrm{ms}$



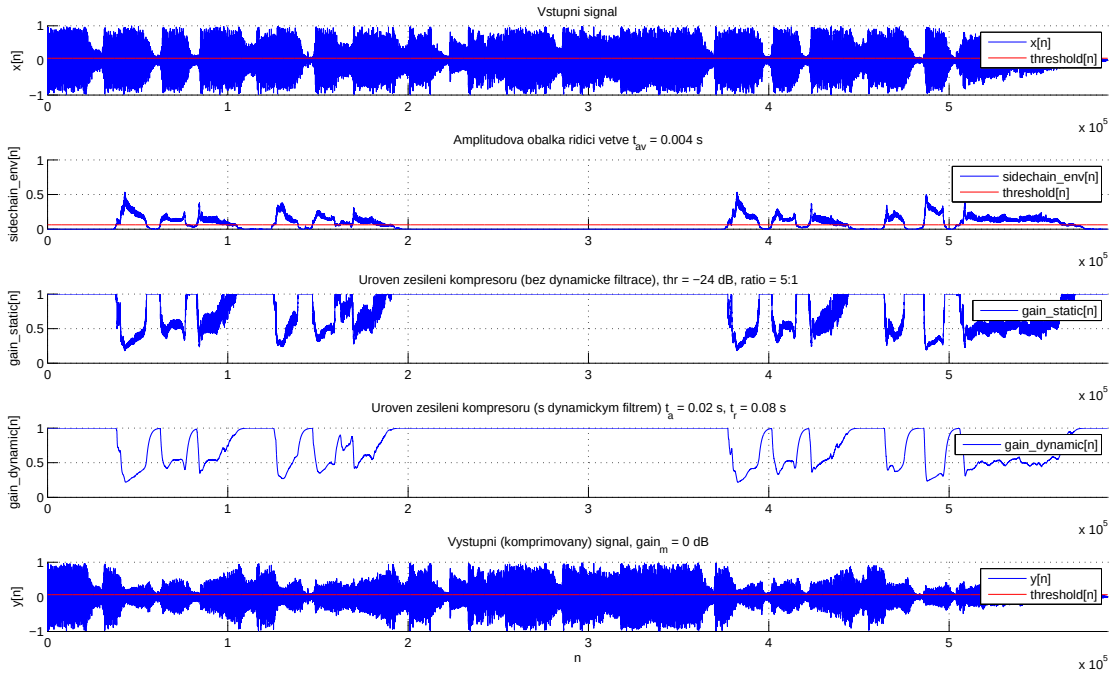
Obr. 4.9: Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 15.

4.3.5 Ducking

Příklad s číslem 17 obsahuje oproti všem předešlým jiný signál ve větvi přímé a větvi řídící. V přímé větvi je část skladby, zatímco větev řídící obsahuje pouze část vokálové linky této skladby. V době, kdy je v řídící větvi signál se přímá větev zeslabí. Pokud by se tento algoritmus rozšířil, a za kompresorem by bylo možné smixovat oba signály v určitém poměru, vznikl by zvukový dojem, kdy v dobu zpěvu utichá hudba, zatímco zpěv zůstává nezměněn.

Parametry tohoto příkladu jsou:

Prahová úroveň:	$-24\,\mathrm{dB}$
Poměr:	$5:1$
Průměrování:	$4\,\mathrm{ms}$
Attack:	$20\,\mathrm{ms}$
Release:	$80\,\mathrm{ms}$



Obr. 4.10: Zobrazení signálu v jednotlivých bodech řídící větve příkladu č. 17.

5 REALIZACE V JAZYCE C++

Aby bylo možné tyto efekty pro úpravu dynamiky signálu používat v reálném čase, je nutné zvolit některou z technologií, která má přístup k hardwarovým komponentám, nebo jako zásuvný modul do aplikací, které zpracování v reálném čase podporují. Pro tuto práci byla vybrána technologie VST, což je bezesporu jedna z nejrozšířenějších možností, kterou podporují prakticky veškeré moderní systémy pro zpracování zvuku (DAW - Digital Audio Workstation). VST moduly jsou psány v jazyce C++. Autorem jejich myšlenky je firma Steinberg, která zdarma poskytuje vývojářské nástroje pro jejich tvorbu.

Vypracování tohoto modulu obsahující modul, zdrojové kódy a projekt v prostředí MS Visual Studio 2015 je v příloze B.1.

Jádro práce je napsáno v C++ frameworku JUCE, který poskytuje bohaté nástroje pro práci jak s grafickým rozhraním, tak se signálem samotným. Framework nabízí export do několika různých formátů audio pluginů, a to jak do formátu VST, tak i např. AU (Audio Units) nebo AAX (Avid Audio eXtension). Ke kompilaci do jednotlivých formátů poté slouží externě napojené knihovny, v případě VST je to tedy VST knihovna firmy Steinberg. Framework JUCE je volně stáhnutelný z webové stránky www.juce.com, kde lze mimojiné najít i dokumentaci a ukázky hotových kódů. Knihovny pro tvorbu VST od firmy Steinberg lze stáhnout na oficiálních stránkách firmy Steinberg www.steinberg.net.

5.1 Struktura programu

Základním kamenem řešení této práce je balík tříd představující jednotlivé bloky řídící větve efektů a třídy pro zjednodušení práce se signálem. Tyto třídy nejsou závislé na technologii VST ani na frameworku JUCE, a je možné je použít i pro jiné účely. Seznam přiložených souborů s krátkým popisem je uveden v příloze B.3 na straně 59.

5.1.1 Třídy pro reprezentaci signálu

Pro zjednodušení práce se signálem jsou v systému zavedeny dvě šablonové třídy: `RZ::SignalBlock` a `RZ::SignalBlockWithBuffer`. Tyto třídy slouží jako kontejner dat, nepřejímají vlastnictví, a při smazání tedy nenastává dealokace dat signálu. Díky přetíženému operátoru pole je možné jednoduše pracovat s jejich daty.

Třída `RZ::SignalBlockWithBuffer` obsahuje kruhový buffer `RZ::Buffer`, jehož vzorky jsou přístupné zápornými indexy pole. Poslední vzorek vložený do bufferu je tedy možné získat indexem -1.

5.1.2 Třídy pro reprezentaci výpočetních bloků

Všechny výpočetní bloky dědí šablonovou třídu `RZ::ProcessorElement`, která reprezentuje elementární výpočetní blok s jedním vstupem a jedním výstupem. Třída obsahuje veřejné metody `countOutput()` a `countAndReturnOutput()`, které by měly být zavolány pro získání výstupního signálu na základě vloženého signálu vstupního. Zatímco metoda `countOutput()` ukládá výstup do svého druhého argumentu, metoda `coundAndReturnOutput()` přijímá pouze jeden argument a vrací novou instanci třídy `RZ::SingalBlock` obsahující výstupní data.

Třída `RZ::ProcessorElement` si může udržovat svůj vnitřní buffer, nastavení jeho délky zajišťují virtuální metody `getInputBufferLength()` pro vstupní buffer a `getOutputBufferLength()` pro buffer výstupní. Ukládání bufferu je automatické. Pro nastavení výchozích hodnot bufferu, které nastane při prvním zavolání metody `countOutput()`, slouží metoda `getInitialBufferValue()`, která vrací výchozí hodnotu vzorků bufferu.

V případě potřeby rozšíření systému o nový výpočetní blok stačí vytvořit potomka třídy `RZ::ProcessorElement` a přetížit metodu `_processReplacing()` která zajišťuje výpočet (převod vstupních vzorků na vzorky výstupní). Pokud je potřebné načítat vzorky z historie, stačí pouze přetížit odpovídající metody pro návrat délky bufferu, a třída si zbytek režie vyřeší sama.

5.1.3 Třídy výpočetních bloků

V systému je vytvořeno několik potomků třídy `RZ::ProcessorElement`, které zajišťují výpočet signálu efektů pro úpravu dynamiky signálu:

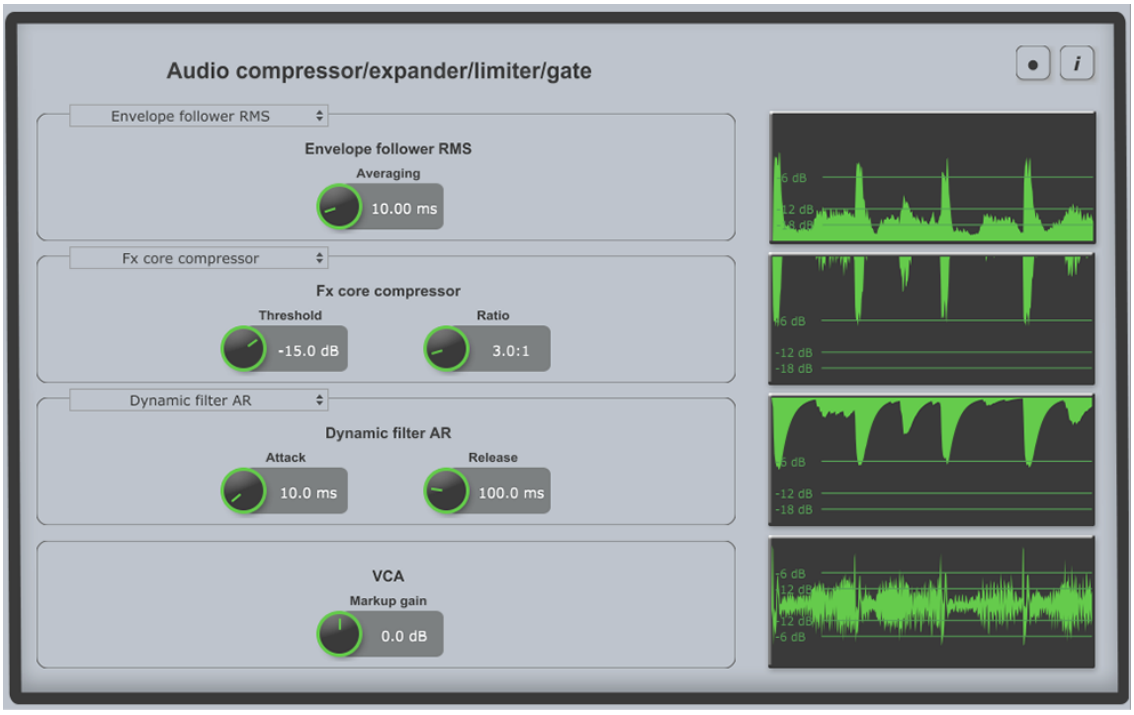
`RZ::ConvertorLinToLog`	Převod signálu lin/log
`RZ::ConvertorLinSquareToLog`	Převod signálu lin^2/log
`RZ::ConvertorLogToLin`	Převod signálu log/lin
`RZ::DynamicFilter`	Reprezentuje obecný vyhlazovač signálu
`RZ::DynamicFilterAR`	Vyhlazení signálu (`attack` a `release`)
`RZ::DynamicFilterARH`	Vyhlazení signálu (`attack`, `release` `hold`)
`RZ::EnvelopeFollower`	Obecný sledovač obálky
`RZ::EnvelopeFollowerPeak`	Sledovač obálky špičkové hodnoty
`RZ::EnvelopeFollowerRMS`	Sledovač obálky efektivní hodnoty
`RZ::FxCore`	Obecné výpočtové jádro zesilovacího činitele
`RZ::FxCoreCompressor`	Výpočetní jádro efektu kompresorového typu
`RZ::FxCoreExpander`	Výpočetní jádro efektu expanderového typu

5.1.4 Hlavní výpočetní třída

Mateřskou třídou projektu je třída `RzAudioProcessor`. Je potomkem JUCE třídy `AudioProcessor`, a obsahuje prakticky veškeré funkce pro správné fungování VST plugin modulu. Její metody `processBlock()` a `processBlockBypassed()` poté zajišťují odpovídající převedení vstupních vzorků na výstupní. V těchto blocích se vyskytují volání všech výpočetních tříd popsaných výše. Třída dále obsahuje například metody pro načtení předvoleb (presetů), uložení aktuálních nastavení do paměti DAW, metody obsluhující nahrávání nebo například změnu parametru efektu.

5.2 Vzhled a ovládání

Práce obsahuje celkem čtrnáct tříd pro reprezentaci jednotlivých vizuálních prvků. Obecně se jedná o tlačítka, otočné potenciometry, visualiséry, nebo třeba grafické reprezentace samotných bloků řídící větve signálu. Grafické rozhraní bylo uzpůsobeno tak, aby vizuálně odpovídalo grafickým prvkům DAW Studio One 2.



Obr. 5.1: Grafické rozhraní VST plugin modulu.

Grafické rozhraní je rozděleno horizontálně do čtyř částí:

- Sledovač obálky
  - Sledovač obálky špičkové hodnoty
  - Sledovač obálky efektivní hodnoty
- Výpočet zesilovacího činitele
  - Výpočet zesilovacího činitele kompresoru
  - Výpočet zesilovacího činitele expanderu
  - Výpočet zesilovacího činitele limiteru
  - Výpočet zesilovacího činitele šumové brány
- Vyhlazení zesilovacího činitele
  - Vyhlazení filtrem s parametry `attack` a `release`
  - Vyhlazení filtrem s parametry `attack`, `release` a `hold`
- Provedení VCA (s možností zesílení výstupu)

Výstup každé této části zobrazuje příslušný visualisér, poslední visualisér zobrazuje výstupní signál celého efektu.

Jednotlivé bloky obsahují přepínače typu, a tak je možné nakombinovat jednotlivé výpočetní bloky dle libosti, např. sledovač obálky špičkové hodnoty, výpočetní jádro expanderu, a dynamický filter s parametrem *hold* (ARH). Jednotlivé parametry, a to včetně přepínačů typu, je možné nastavovat automatizací. Jemný posuv parametru lze aktivovat tlačítkem CTRL. V pravém horním rohu jsou umístěny tlačítka nahrávání a info okénka.

5.3 Nahrávání výstupu

Vlevo od tlačítka info okénka je umístěno tlačítko pro nahrávání výstupu. Při stisku tohoto tlačítka je uživatel vyzván pro vybrání souboru, do kterého mají být vzorky zaznamenány. Data jsou uložena ve formátu CSV (Comma-separated values), jednotlivé sloupce odpovídají signálům zobrazených ve visualisérech. Nahrávání je taktéž možné aktivovat automatizací parametru `Recording`. V tomto případě je soubor uložen s automaticky vygenerovaným názvem na plochu uživatele.

5.4 Předvolby

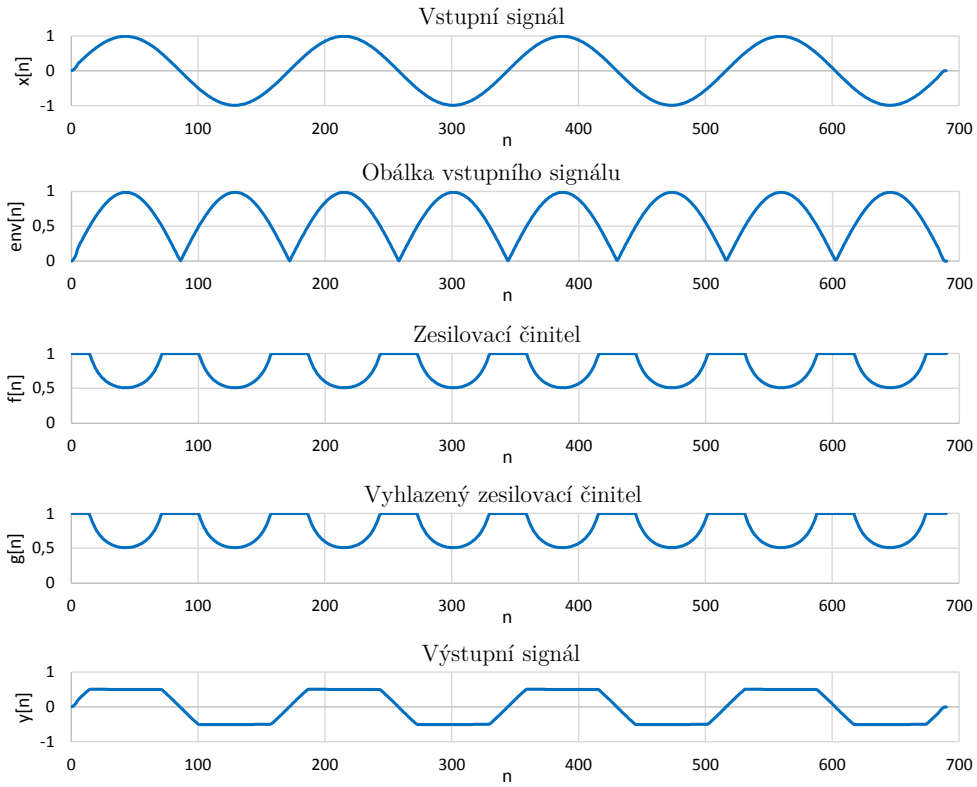
Modul obsahuje několik předvoleb, ze kterých lze vycházet při nastavování konkrétních typů efektů. Součástí tohoto balíku je pak 8 předvoleb, které slouží k demonstraci modulu na vzorových signálech. Vzorové signály jsou označeny názvy $S1$ až $S4$, a tento jejich názvech je uveden i v názvu předvolby. Tyto signály obsahují různé generované průběhy o frekvenci 1 Hz, a jsou určeny pro přehrávání ve smyčce. Seznam souborů se signály lze najít v příloze C.2 na straně 63.

5.4.1 Vzorový příklad č. 1

Příklad demonstruje použití limiteru pro absolutní ořezání křivky nad úrovní −6 dB. Za povšimnutí stojí korekční zesílení obálky špičkové hodnoty o 6 dB, které je nutné, aby úroveň obálky odpovídala vstupnímu signálu.

Parametry této předvolby jsou:

Sledovač obálky	sledovač obálky špičkové hodnoty
Attack:	1 ms
Release:	1 ms
Korekční zesílení:	6 dB
Výpočetní jádro	výpočet zesilovacího činitele limiteru
Prahová úroveň:	−6 dB
Vyhlazení	filtr s parametry `attack` a `release`
Attack:	1 ms
Release:	1 ms



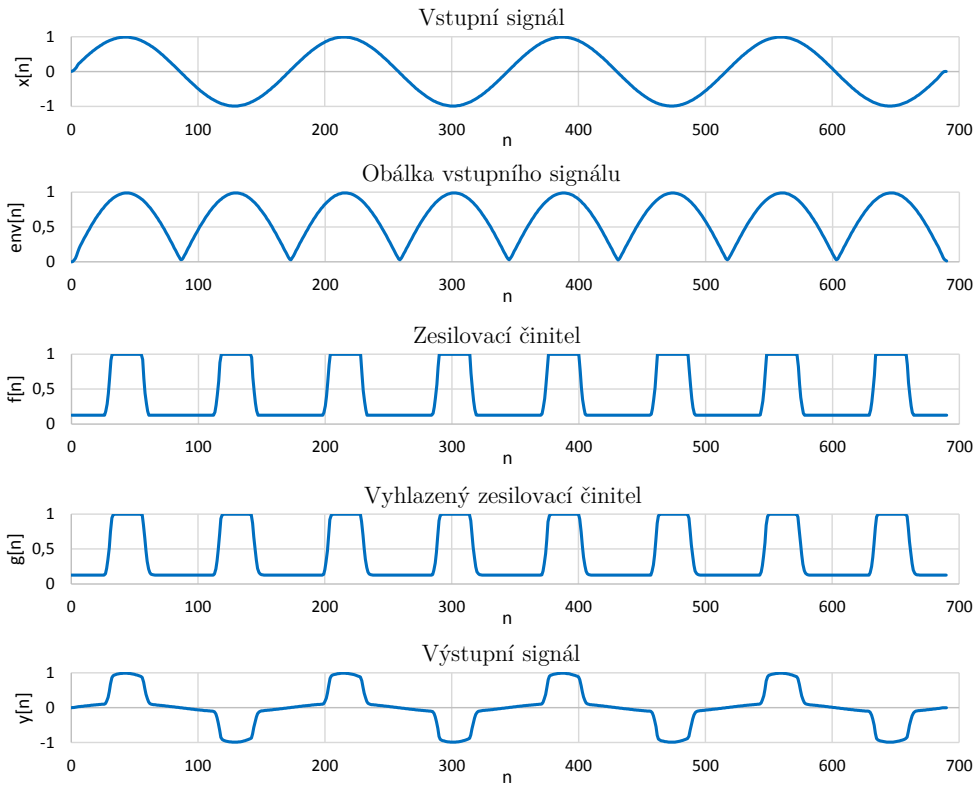
Obr. 5.2: Průběhy signálů VST předvolby č. 1.

5.4.2 Vzorový příklad č. 2

V tomto příkladu je použit expander pro výrazné potlačení signálu pod úrovní −1 dB s maximálním útlumem −18 dB.

Parametry této předvolby jsou:

Sledovač obálky	sledovač obálky efektivní hodnoty
Průměrování:	10 ms
Výpočetní jádro	výpočet zesilovacího činitele expanderu
Prahová úroveň:	−1 dB
Poměr:	1 : 20
Maximální útlum:	−18 dB
Vyhlazení	filtr s parametry `attack` a `release`
Attack:	8 ms
Release:	14 ms



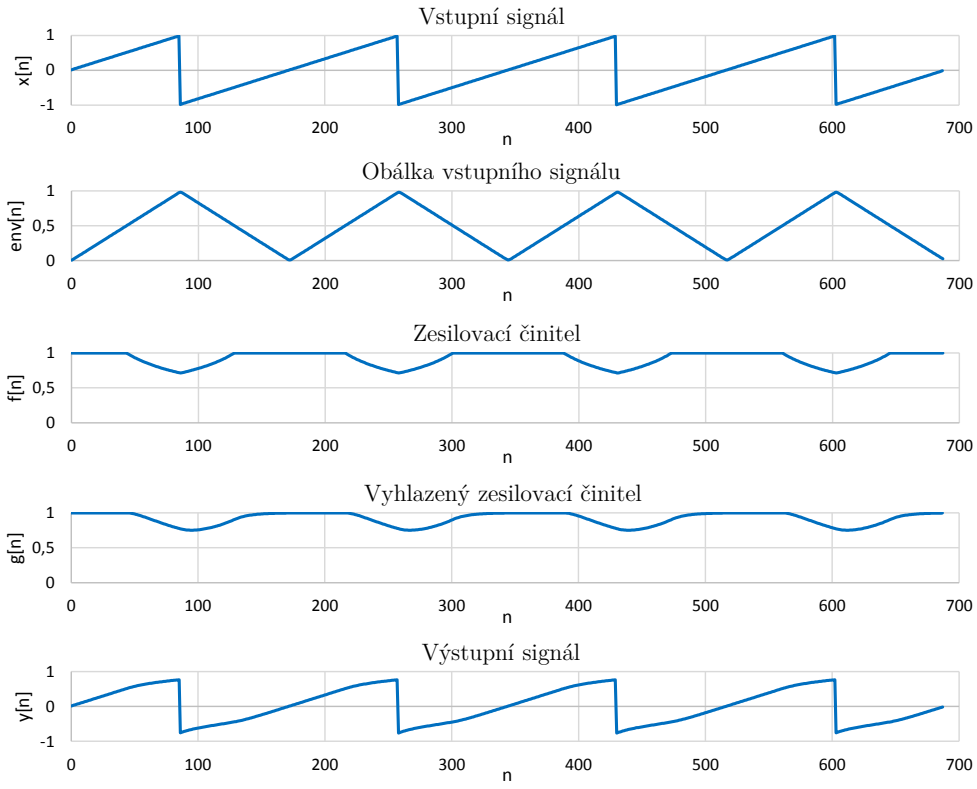
Obr. 5.3: Průběhy signálů VST předvolby č. 2.

5.4.3 Vzorový příklad č. 3

Příklad č. 3 zobrazuje použití mírného kompresoru v poměru 2 : 1 s dlouhými časovými konstantami, díky kterým má výsledná křivka relativně hladký průběh.

Parametry této předvolby jsou:

Sledovač obálky	sledovač obálky efektivní hodnoty
Průměrování:	10 ms
Výpočetní jádro	výpočet zesilovacího činitele kompresoru
Prahová úroveň:	−6 dB
Poměr:	2 : 1
Vyhlazení	filtr s parametry `attack` a `release`
Attack:	150 ms
Release:	150 ms



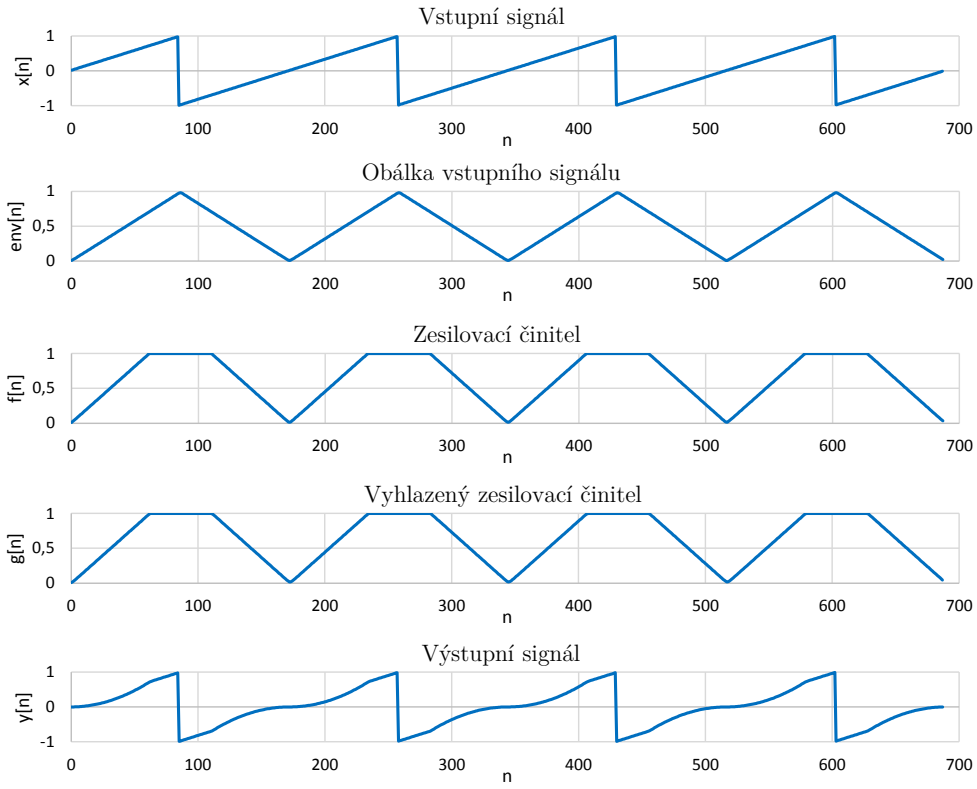
Obr. 5.4: Průběhy signálů VST předvolby č. 3.

5.4.4 Vzorový příklad č. 4

Tento příklad demonstruje použití mírného expanderu v poměru 1 : 2 o maximálním útlumu −48 dB. Výsledná křivka neobsahuje žádné skokové změny, ačkoli jsou časové konstanty nastaveny na velmi nízké hodnoty.

Parametry této předvolby jsou:

Sledovač obálky	sledovač obálky efektivní hodnoty
Průměrování:	10 ms
Výpočetní jádro	výpočet zesilovacího činitele expanderu
Prahová úroveň:	−3 dB
Poměr:	1 : 2
Maximální útlum:	−48 dB
Vyhlazení	filtr s parametry `attack` a `release`
Attack:	10 ms
Release:	10 ms



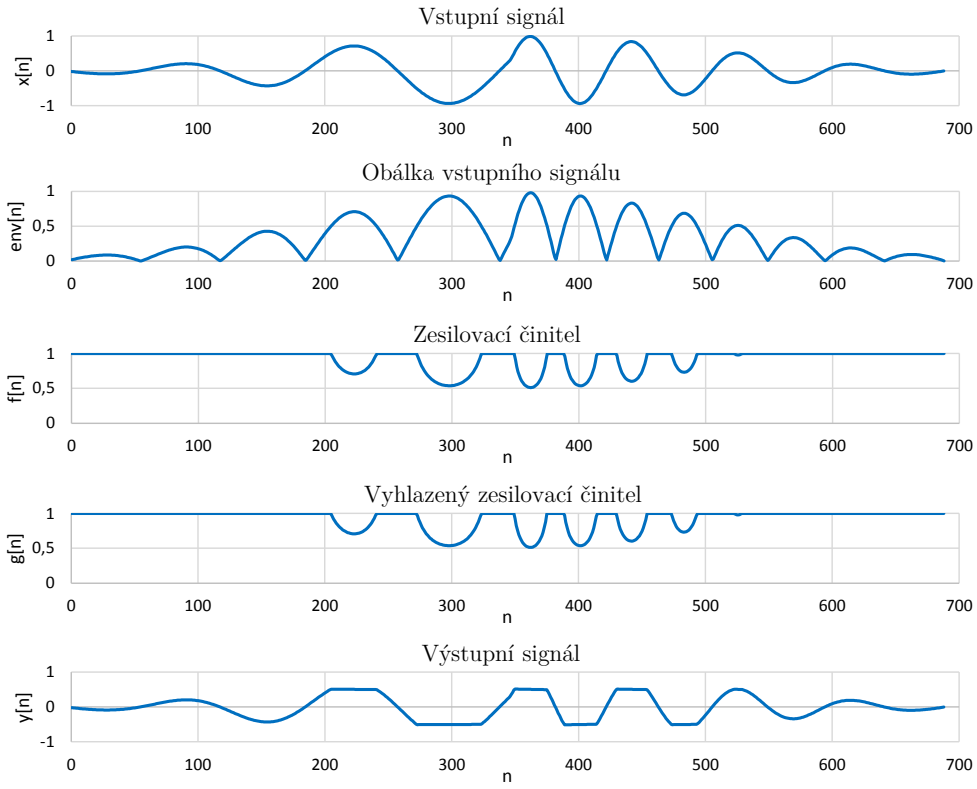
Obr. 5.5: Průběhy signálů VST předvolby č. 4.

5.4.5 Vzorový příklad č. 5

Tento příklad, obdobně jako příklad první, ukazuje použití limiteru pro absolutní ořezání křivky nad úrovní −6 dB. Vstupní signál je amplitudově modulován, a lze tak dobře vidět, že limiter pracuje pouze nad svou rozhodovací úrovní.

Parametry této předvolby jsou:

Sledovač obálky	sledovač obálky špičkové hodnoty
Attack:	0,1 ms
Release:	10 ms
Korekční zesílení:	0 dB
Výpočetní jádro	výpočet zesilovacího činitele limiteru
Prahová úroveň:	−6 dB
Vyhlazení	filtr s parametry `attack` a `release`
Attack:	2 ms
Release:	2 ms



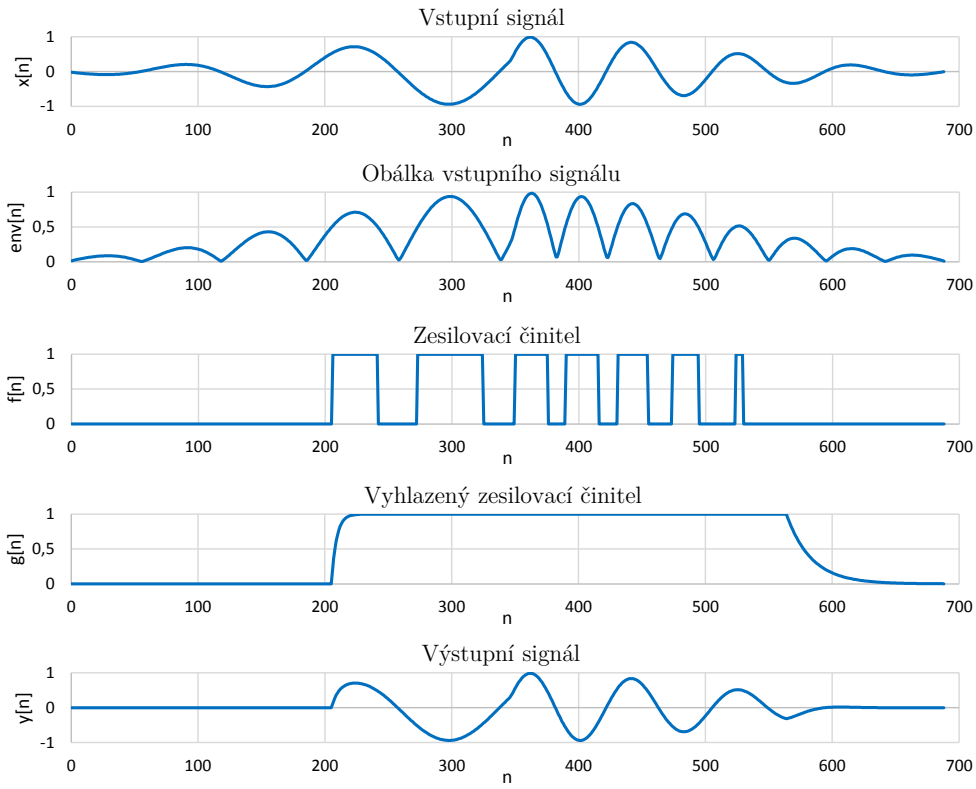
Obr. 5.6: Průběhy signálů VST předvolby č. 5.

5.4.6 Vzorový příklad č. 6

Příklad demonstrující použití šumové brány a dlouhých časových konstant. Výsledkem je velmi přirozené ořezání, které nijak nedeformuje křivku v oblastech, kdy jsou hodnoty její dlouhodobé amplitudové obálky nad rozhodovací úrovní brány.

Parametry této předvolby jsou:

Sledovač obálky	sledovač obálky efektivní hodnoty
Průměrování:	10 ms
Výpočetní jádro	výpočet zesilovacího činitele šumové brány
Prahová úroveň:	−6 dB
Vyhlazení	filtr s parametry `attack`, `release` a `hold`
Attack:	50 ms
Release:	250 ms
Hold:	200 ms



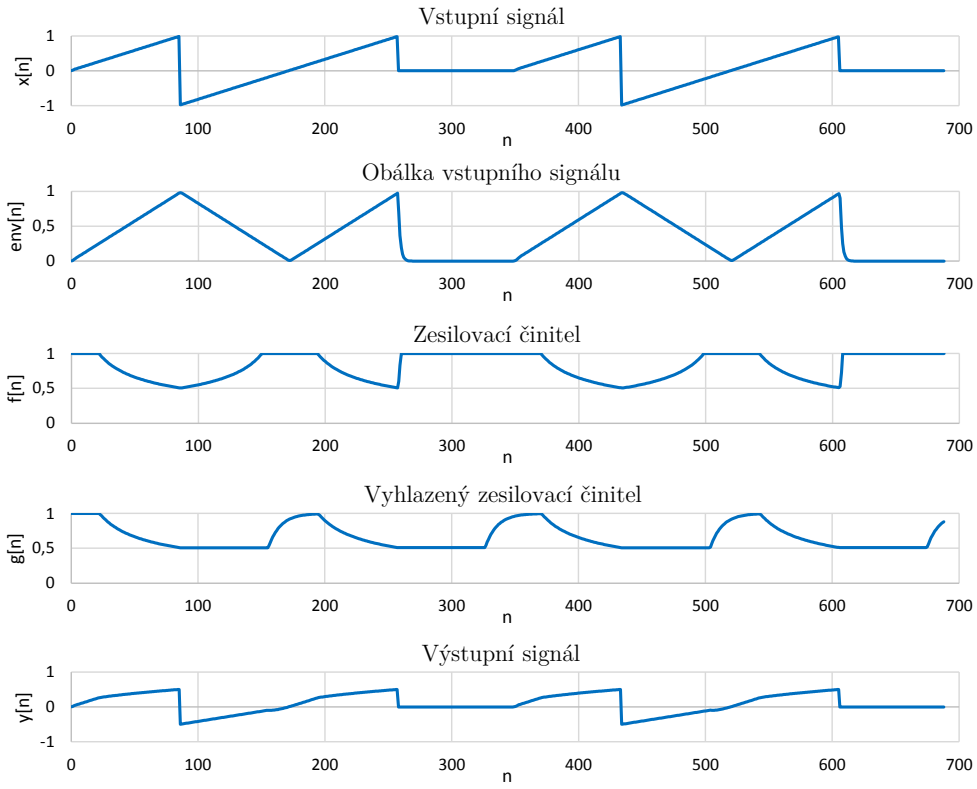
Obr. 5.7: Průběhy signálů VST předvolby č. 6.

5.4.7 Vzorový příklad č. 7

Příklad s číslem 7 ukazuje poněkud atypickou kombinaci kompresoru a dynamického filtru s parametrem hold. Útlum vstupního signálu tedy nejprve setrvá 400 ms na své maximální hodnotě, a až poté se přesune do sestupné fáze.

Parametry této předvolby jsou:

Sledovač obálky	sledovač obálky efektivní hodnoty
Průměrování:	10 ms
Výpočetní jádro	výpočet zesilovacího činitele kompresoru
Prahová úroveň:	−12 dB
Poměr:	2 : 1
Vyhlazení	filtr s parametry `attack`, `release` a `hold`
Attack:	10 ms
Release:	120 ms
Hold:	400 ms



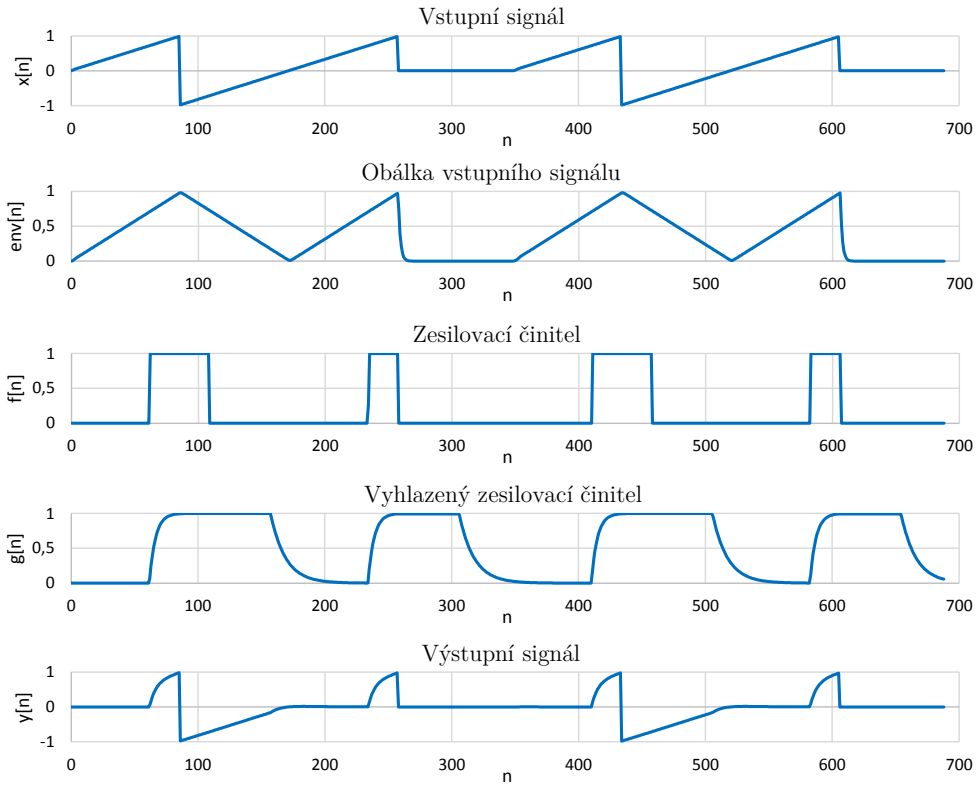
Obr. 5.8: Průběhy signálů VST předvolby č. 7.

5.4.8 Vzorový příklad č. 8

Poslední ze vzorových příkladů je opakem příkladu sedmého – použitím efektu šumová brána nastává útlum signálu pod prahovou úrovní, parametr *hold* zde nastavuje dobu, po kterou je dynamický filtr otevřen, ačkoli vstupní signál poklesl pod prahovou úroveň efektu.

Parametry této předvolby jsou:

Sledovač obálky	sledovač obálky efektivní hodnoty
Průměrování:	10 ms
Výpočetní jádro	výpočet zesilovacího činitele šumové brány
Prahová úroveň:	−6 dB
Vyhlazení	filtr s parametry `attack`, `release` a `hold`
Attack:	60 ms
Release:	150 ms
Hold:	280 ms



Obr. 5.9: Průběhy signálů VST předvolby č. 8.

6 ZÁVĚR

V práci jsem se zabýval návrhem a implementací algoritmů pro úpravu dynamiky zvukového signálu v prostředí Matlab a VST. Toto zadání jsem vypracoval dle programátorských konvencí tak, aby byl znovupoužitelný, částečně separovatelný, velmi dobře čitelný a laditelný.

Vztahy, jež jsou použity pro výpočet v jednotlivých krocích efektů byly převzaty z ověřených zdrojů a mírně doplněny o vlastní poznatky. Byly zkomponovány tak, aby struktura programů co nejvíce odpovídala struktuře takovýchto efektů v analogovém světě, což umožňuje jednodušší pochopení celého procesu.

Zdrojový kód v prostředí Matlab je napsán v anglickém jazyce a to včetně bohatých komentářů, které popisují nejen jednotlivá volání funkcí, ale i vstupně-výstupní parametry těchto funkcí. Byl upraven tak, aby byl lehce srozumitelný a použitelný i pro výuku. Popředná kompatibilita je pak nespornou výhodou používání algoritmů v budoucnosti bez nutnosti aktualizace kódu.

Výsledné zpracování nabízí práci s dynamikou hudebních signálů a umožňuje i klíčování řídící větve. Jedná se o sestavu čtyř nezávislých funkcí pro výpočet těchto efektů, které jsou nadstaveny taktéž čtyřmi funkcemi, které umožňují vykreslení a uložení jejich jednotlivých částí do grafu. Velmi jednoduché je taktéž uložení nebo přehrání výstupního signálu efektu. Tyto funkce pak zastřešuje jedna společná funkce nabízející spouštění vzorových příkladů, a je tedy velice lehké uvést celý algoritmus do chodu.

K implementaci algoritmů do podoby VST plugin modulu bylo použito C++ frameworku JUCE, který zjednodušuje práci s grafickými i výpočetními procesy. Práce byla sestavena tak, aby nebylo příliš náročné rozšiřovat modul o další funkčnosti. Modul nabízí zobrazení a uložení signálů řídící větve do formátu CSV. Všechny parametry lze ovládat automatizací, výpočetní bloky je možné přepínat za chodu, a modul je tak velmi dobře použitelný i pro výukové účely. Grafické rozhraní modulu bylo sestaveno tak, aby svou úrovní odpovídalo požadavkům dnešní doby.

LITERATURA

[1] ADAM Professional Audio, *A7X - Description*, [cit. 30. 11. 2015]. Dostupné z URL: <http://www.adam-audio.com/en/pro-audio/products/a7x/technical-data/>.

[2] BALÍK, M., *Číslicové zpracování akustických signálů*, Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2013, 118 s.

[3] BOURLANGER, R., LAZZARINI, V., *The Audio Programming Book*, MIT Press, 2011. ISBN 978-0-262-01-46-5

[4] DERUTY, E., *'Dynamic Range' & The Loudness War*, Sound On Sound, září 2011. [cit. 30. 11. 2015]. Dostupné z URL: <https://www.soundonsound.com/sos/sep11/articles/loudness.htm/>.

[5] DRIESSEN, P., *ELEC 484 – Audio Signal Processing*, University of Victoria, 2009, 40 s.

[6] GJESTLAND, T., *Background noise levels in Europe*, SINTEF ICT, červen 2008. Dostupné z URL: <https://easa.europa.eu/system/files/dfu/Background_noise_report.pdf/>.

[7] HAAS, W., *The SOS Guide To Mix Compression*, Sound On Sound, květen 2008. [cit. 30. 11. 2015]. Dostupné z URL: <https://www.soundonsound.com/sos/may08/articles/mixcompression.htm/>.

[8] SENIOR, M., *Mixing Secrets For The Small Studio*, [cit. 30. 11. 2015]. Dostupné z URL: <http://www.cambridge-mt.com/ms-mtk.htm/>.

[9] SCHIMMEL, J., *Studiová a hudební elektronika: skriptum*, Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2012, 58-60 s. ISBN 978-80-214-4452-2

[10] SHEPHERD, I., *So Taylor Swift is louder than Motorhead, AC/DC and The Sex Pistols... – wait, WHAT?*, Mastering Media (Production Advice) Ltd. 2015, [cit. 29. 11. 2015]. Dostupné z URL: <http://productionadvice.co.uk/taylor-swift-loudness/>.

[11] ZÖLZER, U., *DAFX - Digital Audio Effects, 1st ed*, New York: John Wiley & Sons, Ltd, 2002, 533 s. ISBN 0-471-49078-4

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

AAX	Avid Audio eXtension
AT	Časová konstanta náběhu
AU	Audio Units
AVT	Časová konstanta průměrování
CSV	Comma-separated values
DAW	Digital Audio Workstation
depth	Depth - maximální útlum filtru
env	Envelope - obálka vstupního signálu
f_s	Vzorkovací frekvence
HT	Doba, po kterou je šumová brána otevřena i když je signál pod rozhodovací úrovní
h_samp	Maximální počet vzorků, kdy je šumová brána otevřena i když je signál pod rozhodovací úrovní
range	Range - maximální útlum filtru
ratio	Zesilovací poměr dynamického filtru
RT	Časová konstanta poklesu
slope	Strmost dynamického filtru
SPL	Sound Pressure Level
t_at	Časová konstanta náběhu
t_avt	Časová konstanta průměrování
thr	Threshold - rozhodovací úroveň filtru
thr_b	Threshold bottom - dolní rozhodovací úroveň filtru typu expander
t_rt	Časová konstanta poklesu
u_cnt	Počet vzorků, kdy je brána otevřena ačkoli signál je pod prahovou úrovní

VCA	Voltage Controlled Amplifier - Napěťově řízený zesilovač
VST	Virtual Studio Technology
WAV	Waveform audio file format

SEZNAM PŘÍLOH

A Vypracování v prostředí Matlab **56**
A.1 Graf signálových toků 58

B Vypracování v jazyce C++ **59**
B.1 VST plugin modul 59
B.2 Projekt v MS Visual Studio 2015 59
B.3 Zdrojový kód 59
B.3.1 Výpočetní část 59
B.3.2 Grafická část 60
B.3.3 Procesní část 62

C Audioukázky **63**
C.1 Implementace Matlab 63
C.2 Implementace VST 63

D Parametry funkce runexample **64**

A VYPRACOVÁNÍ V PROSTŘEDÍ MATLAB

Zdrojový kód byl komponován a testován v prostředí Matlab R2013a a upraven pro kompatibilitu zpětnou (R2010a) i popřednou (R2015a). Pro detailní popis vstupně-výstupních parametrů a funkčnosti je doporučeno nahlédnout do bohatě okomentovaného zdrojového kódu. Kód je uložen v souboru s názvem *Implementace Matlab*.

Jeho struktura je rozdělena na několik souborů:

audioreadmultiple.m
: Funkci `audioreadmultiple()` pro načítání vstupních souborů do dynamických efektů. V případě rozdílných délek vstupních souborů při použití klíčování je pak kratší soubor nadstaven prázdnými vzorky.

compressor.m
: Funkce `compressor()` provádějící kompresi signálu.

dbtolin.m
: Jednoduchá funkce `dbtolin()` pro konverzi hodnot z dB do lineární.

docompressor.m
: Nádstavba funkce `compressor()` která navíc obsahuje načtení vstupních dat a vykreslení grafů.

doexpander.m
: Nádstavba funkce `expander()` která navíc obsahuje načtení vstupních dat a vykreslení grafů.

dolimter.m
: Nádstavba funkce `limiter()` která navíc obsahuje načtení vstupních dat a vykreslení grafů.

donoisegate.m
: Nádstavba funkce `noisegate()` která navíc obsahuje načtení vstupních dat a vykreslení grafů.

envelopepeak.m
: Funkce `envelopepeak()` provádějící převod signálu na jeho obálku špičkové hodnoty.

enveloperms.m
: Funkce `enveloperms()` provádějící převod signálu na jeho obálku efektivní hodnoty.

evaldomode.m
: Funkce `evaldomode()` sloužící pro ukládání výstupního signálu a grafů.

expander.m
: Funkce `expander()` provádějící kompresi expanzi signálu.

gaindynamicar.m

Funkce `gaindynamicar()` pro vyhlazení průběhu zesilovacího činitele.

gaindynamicarh.m

Funkce `gaindynamicarh()` pro vyhlazení průběhu zesilovacího činitele s parametrem `hold`.

gainstaticcompressor.m

Funkce `gainstaticcompressor()` pro výpočet zesilovacího činitele pro efekty typu kompresor.

gainstaticexpander.m

Funkce `gainstaticexpander()` pro výpočet zesilovacího činitele pro efekty typu expander.

limiter.m

Funkce `limiter()` provádějící limitaci signálu.

logtolin.m

Funkce `logtolin` pro převod logaritmických hodnot na hodnoty lineární.

noisegate.m

Funkce `noisegate()` provádějící expanzi signálu.

ratiotoslope.m

Funkce `ratiotoslope()` pro převod kompresního poměru na logaritmickou strmost filtru.

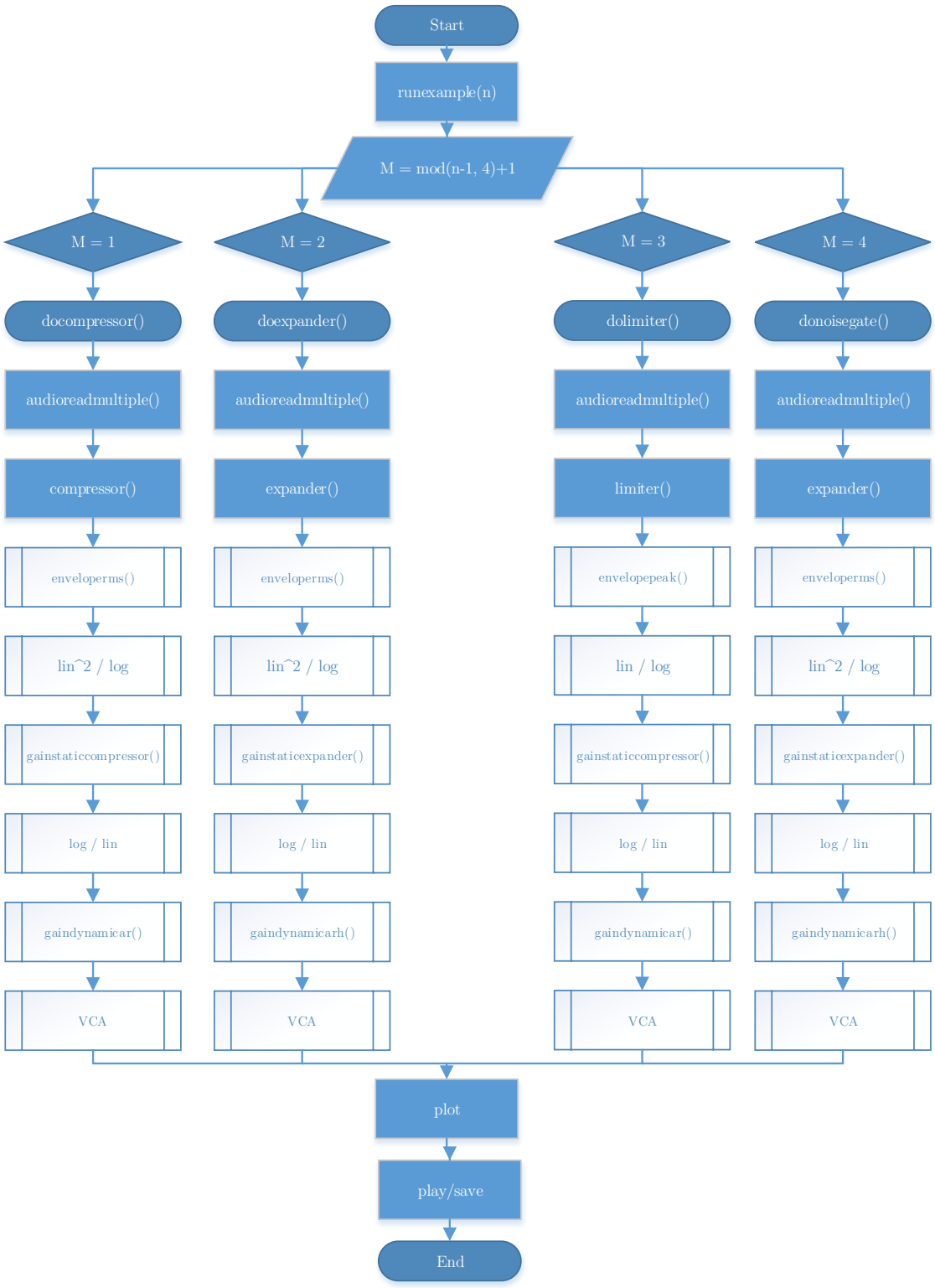
runexample.m

Obsahuje funkci `runexample()` pro spouštění vzorových příkladů.

timebyperiod.m

Funkce `timebyperiod()` pro převod časové konstanty v sekundách na časové konstanty vztažené vůči vzorkovací frekvenci.

A.1 Graf signálových toků



Obr. A.1: Graf signálových toků ve vypracování Matlab.

B VYPRACOVÁNÍ V JAZYCE C++

Zdrojový kód C++ byl komponován ve vývojovém prostředí MS Visual Studio 2015 a obsahuje zpětně nekompatibilní volání funkcí. Je závislý na knihovně JUCE a VST knihovně firmy Steinberg. Veškerý obsah uveden v této příloze je uložen v adresáři *Implementace VST*.

B.1 VST plugin modul

Vypracovaný VST plugin modul verze 2.4 je přiložen pod názvem *RzDynamics.dll*, a v hostitelských programech jej lze najít pod názvem *RzDynamics*.

B.2 Projekt v MS Visual Studio 2015

Součástí práce je projekt ve vývojovém prostředí MS Visual Studio 2015, který je možné upravit a kompilovat na všech počítačích s tímto vývojovým prostředím. V případě, že je požadována přenositelnost výstupního `dll` souboru, je nutné kompilovat projekt v módu *release*. Projekt lze otevřít spuštěním souboru *Implementace VST/Projekt/RzDynamics/Builds/VisualStudio2015/RzDynamics.sln*.

B.3 Zdrojový kód

Zdrojový kód je strukturálně rozdělen na tři sekce - výpočetní, grafická a procesní.

B.3.1 Výpočetní část

Výpočetní část obsahuje veškerou práci se vzorky signálu. Tato část je nezávislá na frameworku JUCE. Zdrojové kódy této části jsou uloženy v adresáři *Fx*.

RzBuffer.h
: Obsahuje šablonovou třídu `RZ::Buffer` reprezentující kruhový buffer.

RzConvertor.h
: Obsahuje třídy pro konverzi signálu mezi logaritmickými a lineárními hodnotami.

RzDynamicFilter.h
: Obsahuje třídy reprezentující dynamické filtry.

RzEnvelopeFollower.h
: Obsahuje třídy sledovačů obálky signálu.

RzFunctions.h

Obsahuje nezařaditelné funkce.

RzFxCore.h

Obsahuje funkce reprezentující výpočty zesilovacích činitelů efektů.

RzProcessorElement.h

Obsahuje mateřskou šablonovou třídu `RZ::ProcessorElement`, kterou dědí ostatní výpočetní bloky.

RzSignalBlock.h

Obsahuje třídu `RZ::SignalBlock` zjednodušující práci se signálem.

RzSignalBlockWithBuffer.h

Obsahuje třídu `RZ::SignalBlockWithBuffer` zjednodušující práci se signálem a bufferem.

B.3.2 Grafická část

Grafická část obsahuje třídy reprezentující konkrétní grafické elementy modulu, a to od potenciometru až po kompletní grafické komponenty. Zdrojové kódy této části jsou uloženy v adresáři *Gui*.

RzAudioVisualiserComponent.cpp

RzAudioVisualiserComponent.h

Třída `RzAudioVisualiserComponent` reprezentující visualisér signálu.

RzBackground.cpp

Soubor s obrázkem pozadí modulu.

RzButton.cpp

RzButton.h

Třída `RzButton` reprezentující tlačítko.

RzComboBox.cpp

RzComboBox.h

Třída `RzComboBox` reprezentující klikací seznam prvků.

RzGuiDynamicFilterAR.cpp

RzGuiDynamicFilterAR.h

Třída `RzGuiDynamicFilterAR` je komponenta dynamického filtru.

RzGuiDynamicFilterARH.cpp

RzGuiDynamicFilterARH.h

Třída `RzGuiDynamicFilterARH` je komponenta dynamického filtru s parametrem `hold`.

RzGuiEnvelopeFollowerPeak.cpp
RzGuiEnvelopeFollowerPeak.h

Třída `RzGuiEnvelopeFollowerPeak` je komponenta sledovače obálky špičkové hodnoty.

RzGuiEnvelopeFollowerRMS.cpp
RzGuiEnvelopeFollowerRMS.h

Třída `RzGuiEnvelopeFollowerRMS` je komponenta sledovače obálky efektivní hodnoty.

RzGuiFxCoreComp.cpp
RzGuiFxCoreComp.h

Třída `RzGuiFxCoreComp` je komponenta výpočetního jádra kompresoru.

RzGuiFxCoreExp.cpp
RzGuiFxCoreExp.h

Třída `RzGuiFxCoreExp` je komponenta výpočetního jádra expanderu.

RzGuiFxCoreGate.cpp
RzGuiFxCoreGate.h

Třída `RzGuiFxCoreGate` je komponenta výpočetního jádra šumové brány.

RzGuiFxCoreLim.cpp
RzGuiFxCoreLim.h

Třída `RzGuiFxCoreLim` je komponenta výpočetního jádra limiteru.

RzGuiVCA.cpp
RzGuiVCA.h

Třída `RzGuiFxCoreLim` je komponenta posledního bloku (VCA).

RzInfoWindow.cpp
RzInfoWindow.h

Třída `RzInfoWindow` obsahující info okénko.

RzKnob.cpp
RzKnob.h

Třída `RzKnob` reprezentující otočný potenciometr.

B.3.3 Procesní část

Procesní část obsahuje třídy samotného jádra modulu. Zdrojové kódy této části jsou uloženy v adresáři *Core*.

RzAudioParameterBool.cpp
RzAudioParameterBool.m
: Třída `RzAudioParameterBool` reprezentuje parametr modulu s booleanskou hodnotou.

RzAudioParameterFloat.cpp
RzAudioParameterFloat.m
: Třída `RzAudioParameterFloat` reprezentuje parametr modulu s hodnotou s plovoucí desetinnou čárkou.

RzAudioParameterInt.cpp
RzAudioParameterInt.m
: Třída `RzAudioParameterInt` reprezentuje parametr modulu s celočíselnou hodnotou.

RzAudioProcessor.cpp
RzAudioProcessor.m
: Třída `RzAudioProcessor` je mateřskou třídou modulu a obsahuje veškeré procesní a výpočetní metody specifické pro audio plugin modul.

RzAudioProcessorEditor.cpp
RzAudioProcessorEditor.m
: Třída `RzAudioProcessor` je mateřskou třídou modulu a obsahuje veškeré metody grafického charakteru specifické pro audio plugin modul.

C AUDIOUKÁZKY

C.1 Implementace Matlab

Práce obsahuje 5 zvukových souborů typu WAV, které jsou použity ve vzorových příkladech.

sample_440.wav
: Velmi krátký zvukový soubor obsahující asi 1500 vzorků konstantní sinusové vlny o frekvenci 440 Hz.

sample_440_fade.wav
: Velmi krátký zvukový soubor o dvojnásobné délce obsahující sinusovou vlnu s frekvencí 440 Hz s postupným náběhem.

sample_kick.wav
: 8sekundová zvuková nahrávka velkého bubnu s velmi výraznými přeslechy ostatních nástrojů rockové kapely.

sample_music.wav
: Část hudební skladby s délkou asi 13 sekund stažena z volně dostupného zdroje [8].

sample_voice.wav
: Nahrávka zpěvu stejné hudební skladby.

C.2 Implementace VST

K demonstraci funkcí VST plugin modulu jsou dodány čtyři zvukové ukázky.

S1.wav
: Sinusový signál o frekvenci 1 Hz o délce čtyř period.

S2.wav
: Pilový signál o frekvenci 1 Hz o délce čtyř period.

S3.wav
: Amplitudově modulovaný sinusový signál o frekvenci 2,2 Hz o délce šesti period.

S4.wav
: Přerušovaný pilový signál o frekvenci 1 Hz o délce čtyř period.

D PARAMETRY FUNKCE RUNEXAMPLE

Tab. D.1: Vstupní parametry n funkce runexample().

n	Typ efektu	Typ ukázky
1	Limiter	Krátký vzorek 440Hz signálu
2	Limiter	Krátký vzorek 440Hz signálu s vytišením (fade out)
3	Limiter	Zvuková ukázka velkého bubnu s přeslechy
4	Limiter	Hudební ukázka
5	Kompresor	Krátký vzorek 440Hz signálu
6	Kompresor	Krátký vzorek 440Hz signálu s vytišením (fade out)
7	Kompresor	Zvuková ukázka velkého bubnu s přeslechy
8	Kompresor	Hudební ukázka
9	Expander	Krátký vzorek 440Hz signálu
10	Expander	Krátký vzorek 440Hz signálu s vytišením (fade out)
11	Expander	Zvuková ukázka velkého bubnu s přeslechy
12	Expander	Hudební ukázka
13	Šumová brána	Krátký vzorek 440Hz signálu
14	Šumová brána	Krátký vzorek 440Hz signálu s vytišením (fade out)
15	Šumová brána	Zvuková ukázka velkého bubnu s přeslechy
16	Šumová brána	Hudební ukázka
17	Kompresor	Hudební vzorek v nastavení typu ducking

Tab. D.2: Vstupní parametry do_mode funkce runexample().

do_mode	Akce
0	Žádná akce
1	Přehrání výstupního signálu
2	Přehrání vstupního i výstupního signálu
3	Přehrání vstupního i výstupního signálu a uložení výstupu
4	Uložení výstupu