VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

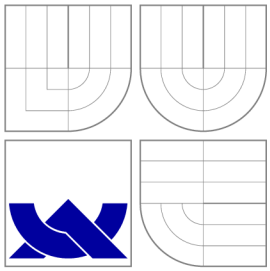
PŘEHRÁVAČ HUDBY (SYNTETIZÉR) POMOCÍ FITKITU

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

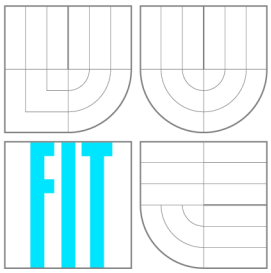
AUTOR PRÁCE
AUTHOR

VOJTĚCH MELICHAR

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

PŘEHRÁVAČ HUDBY (SYNTETIZÉR) POMOCÍ FITKITU

MUSIC PLAYER (SYNTHESIZER) BY FITKIT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — VOJTĚCH MELICHAR
AUTHOR

VEDOUCÍ PRÁCE — Ing. ZDENĚK VAŠÍČEK, Ph.D.
SUPERVISOR

BRNO 2013

Abstrakt

Tato práce se zabývá metodami syntézy zvuku a návrhem syntezátoru pro platformu FITkit. V první části jsou charakterizovány jednotlivé metody syntézy zvuku a je zde také uvedena stručná historie syntézy zvuku. Dále je charakterizován zvukový čip SID 6581. V druhé části práce je uveden návrh jednotlivých částí syntezátoru: oscilátoru, generátoru obálky, hlasu a filtru. Tyto komponenty jsou následně složeny do větších celků tak, aby vytvořily syntezátor. Korektnost implementace je ověřena pomocí simulace v programu ModelSim 6.6d. Jednotlivé komponenty jsou simulovány samostatně a poté jsou simulovány ve větších celcích.

Abstract

This bachelor's thesis deals with sound synthesis methods and with design of a synthesizer for FITkit platform. In the first part of the thesis, there are described methods of sound synthesis and the history of sound synthesis is also stated here. Next part contains a brief characteristic of sound chip SID 6581. In the next part of the thesis, there is a design of each part of the synthesizer: oscillator, envelope generator, voice and filter. Then, these components are put together to make up an synthesizer. Correctness of the implementation is verified by a simulation in ModelSim 6.6d. Individual components are simulated separately and then in mayor groups.

Klíčová slova

Syntéza zvuku, syntezátor, oscilátor, generátor obálky, filtr, VHDL, FPGA, FITkit

Keywords

Sound synthesis, synthesizer, oscillator, envelope generator, filter, VHDL, FPGA, FITkit

Citace

Vojtěch Melichar: Přehrávač hudby (syntetizér) pomocí
FITkitu, bakalářská práce, Brno, FIT VUT v Brně, 2013

Přehrávač hudby (syntetizér) pomocí FITkitu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Zdeňka Vašíčka, Ph.D. a uvedl jsem v ní všechny použité literární a jiné odborné zdroje.

.....
Vojtěch Melichar
14. května 2013

Poděkování

Děkuji panu Ing. Zdeňku Vašíčkovi, Ph.D. za odborné rady a cenné připomínky, kterými přispěl k vypracování této bakalářské práce.

Obsah

Seznam obrázků

Kapitola 1

Úvod

Syntéza zvuku je proces umělého generování zvuku. Tento proces umožňuje generovat zvuky, které by bylo těžké nebo dokonce nemožné generovat pomocí reálného hudebního nástroje. Jedna z oblastí syntézy zvuku se tedy zabývá hledáním nových zvuků. Při hledání nových zvuků vstupuje do tohoto procesu i náhoda a to hlavně díky tomu, že některé metody syntézy mají nepředvídatelný výsledek. Nový zvuk je možné nalézt postupným nastavováním parametrů syntézy a čekáním dokud výsledkem nebude nějaký nový zvuk.

Druhou oblastí syntézy zvuku je snaha co nejvěrněji modelovat zvuk generovaný určitým hudebním nástrojem. Tato snaha je ztížena vlastnostmi reálné předlohy, které není možné vždy přesně simulovat. Při snaze simulovat nějaký hudební nástroj je nutná cílenější snaha. Nestačí pouze měnit parametry a čekat, jaký bude výsledek. Je nutná podrobná studie modelovaného nástroje. Tyto dvě oblasti se ve svém základním pojetí prolínají.

Zvuková syntéza má za sebou více jak stoletou historii. V počátcích byly používány k syntéze zvuku částečně mechanické přístroje. Nevýhodou takovéhoto přístupu byla jejich vysoká cena a také většinou velká hmotnost. Z tohoto důvodu se nejdříve syntezátory používaly pro výzkumné účely a nebyly určeny pro hudebníky. Postupně docházelo ke zmenšování nástrojů. Za zmenšením stojí především fakt, že nástroje již nebyly napůl mechanické. Dalšího zmenšení se nástroje dočkaly s příchodem tranzistoru. S tím, jak se syntezátory vyvíjely, klesala i jejich cena. To umožnilo rozšíření syntezátorů mezi hudebníky a větší využití syntézy zvuku pro tvorbu hudby. Další změna v syntéze zvuku nastala s příchodem dostatečně výkonných počítačů, které umožňují syntézu zvuku v reálném čase s velmi vysokou kvalitou. V dnešní době jsou syntezátory rozšířeny a je možné je slyšet na mnoha hudebních nahrávkách.

Cílem této práce je implementovat syntezátor. Implementace syntezátoru proběhla v jazyce VHDL, který slouží k popisu hardwaru. Cílovou platformou je školní přípravek FITkit, který je osazen čipem FPGA firmy Xilinx. Konkrétně se jedná o čip Spartan 3 XC3S50-4PQ208C. Zde implementovaný syntezátor je ovládán mikrokontrolerem z rodiny MSP430 firmy Texas Instruments. Pro mikrokontroler byla napsána knihovna umožňující nastavení syntezátoru.

Za účelem implementace syntezátoru byly nastudovány jednotlivé metody syntézy zvuku. V první části práce je také uvedena stručná historie syntézy zvuku. Následovaná popisem jednotlivých metod syntézy zvuku. Nejprve jsou uvedeny metody, které přímo generují zvuk. Poté je probrána metoda amplitudové obálky, kterou je možné využít v kombinaci s metodami generujícími zvuk. Metoda amplitudové obálky upravuje časový průběh znění tónu. Metody generující zvuk v kombinaci s amplitudovou obálkou lépe modelují zvuk tvořený reálným hudebním nástrojem.

Další částí práce je popis zvukového čipu SID 6581, z kterého tato práce vychází. SID 6581 byl ve své době přelomovým zvukovým čipem, který používá několik různých technik k syntéze zvuku. Techniky využité v čipu SID jsou amplitudová obálka, kruhová modulace a filtrace výstupního signálu.

Po popisu zvukového čipu SID následuje návrh syntezátoru implementovaného v této práci. Nejprve je popsán návrh jednotlivých jednotek, jako je například generátor obálky. Z těchto jednotek se sestavují složitější jednotky. Výsledkem návrhu je pak struktura celého syntezátoru.

Poslední částí této práce je ověření korektní funkce celého syntezátoru. Ověření korektnosti probíhalo po jednotlivých částech tak, jak byly navrženy v kapitole o návrhu. Korektně fungující jednotky byly poté spojeny do složitějších. Tyto jednotky byly následně také ověřeny, čímž byla dokázána korektní funkce celého syntezátoru.

Kapitola 2

Syntéza zvuku

Syntéza zvuku je proces tvorby zvuku, který využívá již existující zvukové nahrávky, které dále zpracovává, nebo může využívat elektrických či mechanických principů [12]. Dále je také možné zvuk tvořit za pomoci různých matematických nebo fyzikálních postupů. Syntézu zvuku lze rozdělit na dvě základní oblasti [3].

První oblastí je tvorba zvuků, které mají za cíl se co nejvíce přiblížit zvukům vytvářeným reálným hudebním nástrojem. Oblast generování zvuků reálných hudebních nástrojů vyžaduje podrobné studium způsobů, jakými jsou tóny jednotlivými nástroji tvořeny. Důležité je také studium parametrů hudebních nástrojů, jež mají na výsledný tón významný vliv, např. velikost ozvučnice. Faktorem, který do značné míry ztěžuje tento přístup, je fakt, že většina posluchačů je dobře obeznámena se zvukem produkovaným reálným hudebním nástrojem. Posluchači tedy mohou celkem jednoduše porovnávat syntetizovaný zvuk s jeho reálnou předlohou.

Druhou neméně významnou oblastí je tvorba zvuků, které nemají za cíl přiblížit se jakémukoliv reálnému nástroji. Tvorba zvuků bez reálného protějšku je jednodušší tím, že posluchač nemá možnost porovnávat s klasickým nástrojem. Tvůrce má tak širší pole pro experimentování s různými nastaveními parametrů. Obě metody se ale od sebe v základním pojetí příliš neliší a navzájem se prolínají.

Zvuková syntéza využívá řadu rozmanitých metod. Mezi základní metody syntézy zvuku patří aditivní (součtová), subtraktivní (rozdílová), modulační, tvarová, granulační, samplovací a fyzikální modelování (virtuální akustická syntéza) [3]. Modulační syntézu je možné rozdělit na frekvenční modulaci a amplitudovou modulaci. Také tvarová modulace je nadmnožinou pro několik metod, kterými jsou nelineární tvarování.

2.1 Historie

Syntéza zvuku a syntezátory mají za sebou již více jak stoletou historii [4] a [1]. První experimenty s elektronickými nástroji probíhaly již koncem 19. století. Nejranější využití elektrické energie pro hudební účely se ale datuje již do šedesátých let 18. století. Hermann von Helmholtz v tomto období postavil několik elektro-mechanických oscilátorů, které využíval pro své studium lidského vnímání zvuku.

V roce 1906 byl veřejnosti představen nástroj nazvaný Telharmonium. Jednalo se o polyfonní přístroj. Funkci oscilátorů zde zastávalo 145 upravených dynam s induktory, které vytvářely zvuky o slyšitelných frekvencích. Telharmonium byl podobně jako předchozí nástroj také elektro-mechanický přístroj, vážil téměř 200 tun a zabíral plochu $18m^2$.

Dalším nástrojem, který ale dosáhl větších úspěchů a je dokonce dodnes používán, byl přístroj ruského vynálezce Lvova Těrmenova. Přístroj je pojmenován po svém vynálezci Theremin a byl představen veřejnosti v roce 1920. Princip Theremina je založen na dvou oscilátorech, z nichž jeden má fixní frekvenci 170 kHz a druhý frekvenci proměnnou v rozsahu 168 až 170 kHz. Oba oscilátory jsou tedy naladěny na frekvence, jež lidský sluch není schopen vnímat. Přístroj se ovládal pomocí dvou antén, ke kterým hráč přibližoval ruce a tím rozlaďoval obvod. Tímto způsobem rozlaďovaný obvod již generoval zvuky o slyšitelných frekvencích.

Velmi osobitým nástrojem, který byl představen v roce 1935, byly Hammondovy varhany. Ty si získaly reputaci díky svému typickému a nezaměnitelnému zvuku. Hammondovy varhany je možné vidět a především slyšet na mnoha koncertních pódiích a hudebních nahrávkách. Zvuk je v Hammondových varhanech tvořen pomocí tvarovaných disků pohybujících se v magnetickém poli.

V roce 1951 se zrodila idea elektronického hudebního studia. V témže roce Robert Beyer a Fritz Enkel ustanovili komisi, která měla za cíl toto studio vybudovat. Studio bylo dokončeno o dva roky později a stalo se vzorem pro další studia podobného typu. V začátcích bylo studio vybaveno pouze jedním generátorem sinusového průběhu, generátorem bílého šumu a filtry.

V 50. letech minulého století díky rozšíření tranzistorů prožívá revoluci i odvětví syntézy zvuku. Hlavní změnou byla napěťově řízená technologie, která umožňovala kontrolu výstupních charakteristik jak oscilátorů tak i zesilovačů. Prvním napěťově řízeným syntezátorem byl nástroj Haralda Bode se jménem Melochord představený v roce 1961. O tři roky později Robert Moog a Donald Buchla nezávisle na sobě zkonstruovali své syntezátory. Společnosti obou výše zmíněných konstruktérů se staly na několik let dominantními společnostmi v tomto oboru.

Tyto analogové syntezátory však mají jednu značnou nevýhodu a tou je, že se vlivem okolního prostředí, např. teplotou, mohou rozladit. Proto v 50. letech 20. století začal pracovník Bell Telephone Laboratories, Max Mathews, experimentovat s digitálním počítačem jako s nástrojem pro syntézu zvuku. Výsledkem těchto experimentů byl v roce 1957 program MUSIC I, následovaný o rok později programem MUSIC II. V obou případech se jednalo o kód v assembleru, který umožňoval generovat jednoduché zvuky. Ani jeden z programů však neumožňoval syntézu v reálném čase vzhledem k nedostatečnému výkonu počítače pro který byly určeny. V následujících letech se objevilo mnoho verzí těchto programů od různých autorů.

Do 60. let 20. století se datuje začátek použití frekvenční modulace pro účely syntézy zvuku. Jako první se využitím frekvenční modulace pro syntézu zvuku zabýval John Chowning. Frekvenční modulace byla nejdříve využívána pouze v rádiích pro přenos signálu, později se stala metodou pro syntézu zvuku.

Roku 1975 John Appleton, Sydney Alonso a Cameron Jones vytváří prototyp syntezárotu, jenž je později znám pod názvem Synclavier. Nástroj Synclavier byl na svou dobu velice moderní. O čtyři roky později byl představen Fairlight CMI, který byl schopen vytvářet zvuk za použití různých metod, jako jsou aditivní syntéza, subtraktivní syntéza nebo samplovací syntéza. Ve stejném roce představila společnost Digital Music Systems počítač, jenž byl optimalizován pro zpracování audio signálu. Takto upravený počítač se dá považovat za předchůdce DSP (Digital Signal Processor).

Vývoj metod syntézy zvuku a stejně tak syntezátorů pokračuje i v dnešní době. Ale i přes neustálý vývoj jsou některé starší syntetizátory dodnes používány pro svoje hudební kvality, např. Hammondovy varhany.

2.2 Metody syntézy zvuku

Než-li si vysvětlíme jednotlivé metody syntézy zvuku, bude uveden stručný popis tvorby zvuku reálným hudebním nástrojem. Mnoho metod totiž z těchto základních principů vychází.

Zvuk je možné definovat jako mechanické vlnění šířící se v čase s určitou rychlostí v pružném prostředí s určitou intenzitou a frekvencí [10]. Jestliže má zvuk pravidelný periodický průběh, jde v tomto případě o zvuk hudební, čili tón. Každý periodický signál lze pomocí Fourierovy transformace rozložit na sumu nekonečného počtu jednoduchých sinusových signálů [17] a [9]. Jednotlivé sinusové signály se liší svými vlastnostmi jako je amplituda nebo frekvence. Signál se tedy neskládá pouze ze své základní frekvence, která je například pro komorní A 440 Hz, ale i z celočíselných násobků této frekvence. Celočíselným násobkům základní frekvence se říká vyšší harmonické a jejich zastoupení v produkovaném tónu umožňuje od sebe rozlišit jednotlivé nástroje. Na obrázku 2.1 je uveden příklad složení složitějšího signálu pomocí několika jednoduchých signálů se sinusovým průběhem.

Reálný zvuk ovšem není akordem složeným pouze z harmonických, ale utvářejí ho i další parametry. Reálný hudební nástroj je možné rozdělit na tři základní části, které ovlivňují jeho zvuk [10] a [9].

- Budič zvuku
- Zdroj zvuku
- Rezonátor

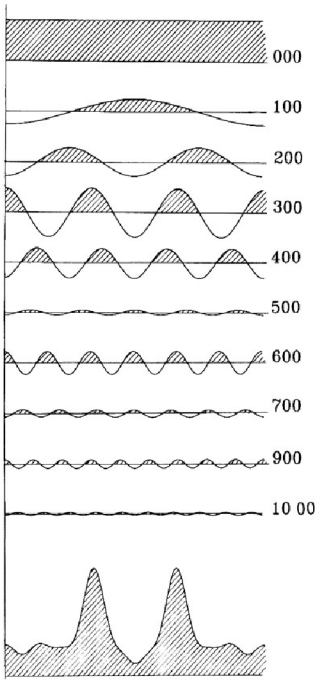
Budič zvuku je část nástroje, která jej rozezvučí, například smyčec u houslí. *Zdroj zvuku* je rozezvučen pomocí budiče a osciluje, čímž vytváří periodické vlnění. U výše zmíněných houslí je zdrojem zvuku struna, jež je rozkmitána smyčcem. Poslední částí hudebního nástroje je rezonátor. *Rezonátor* je většinou tělo nástroje, které bývá zpravidla duté. Frekvence vytvářené zdrojem zvuku jsou vlastní oscilací rezonátoru buď zesilovány nebo zeslabovány. Rezonátor tedy zastává funkci filtru. Z tohoto principu tvorby zvuku reálným hudebním nástrojem vychází řada metod pro syntézu zvuku.

2.2.1 Aditivní syntéza

Aditivní syntéza [3] patří k nejstarším metodám syntézy zvuku. Princip metody vychází přesně z principů, jež byly vysvětleny výše. Vychází hlavně z principu skládání složitých signálů pomocí součtu signálů jednoduchých, viz obrázek 2.2. Součtem jednoduchých signálů se do výsledného signálu přidávají potřebné vyšší harmonické. Celý princip je vyjádřen vztahem:

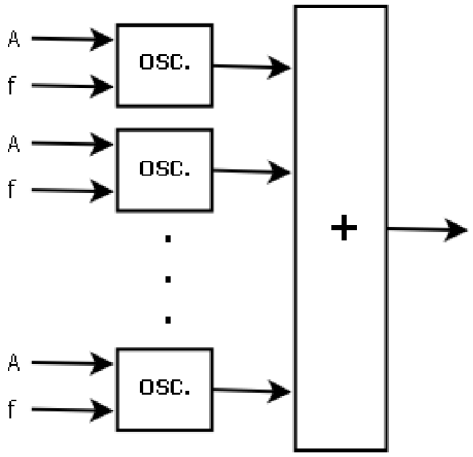
$$f(t) = \sum_{k=1}^{N} A_k sin(\omega_k t + \varphi_k) \tag{2.1}$$

kde A_k je amplituda, ω_k je frekvence a φ_k je fáze k-té složky tónu. Počet slyšitelných vyšších harmonických může být v praxi vysoký [12]. Za předpokladu, že nejnižší požadovaná frekvence pro nízké A je 55 Hz, jsou vyšší harmonické na frekvencích 110 Hz, 165 Hz, 220 Hz, 275 Hz, 330 Hz, atd. Frekvence 64. harmonické je 3520 Hz, což je stále ve slyšitelném spektru. Pro dostatečně kvalitní výstup dostačuje podle [3] prvních deset vyšších harmonických. Tato metoda je považována za jednu z nejdokonalejších metod. Navzdory



Obrázek 2.1: Složení signálu pomocí několika sinusových signálů [16].

své zvukové dokonalosti aditivní metoda nedosáhla velkého komerčního úspěchu. Hlavním důvodem je její relativně složité ovládání, při kterém musí uživatel zadávat velké množství parametrů. Aditivní syntéza se tedy objevuje spíše v kombinaci s jinou metodou syntézy zvuku.

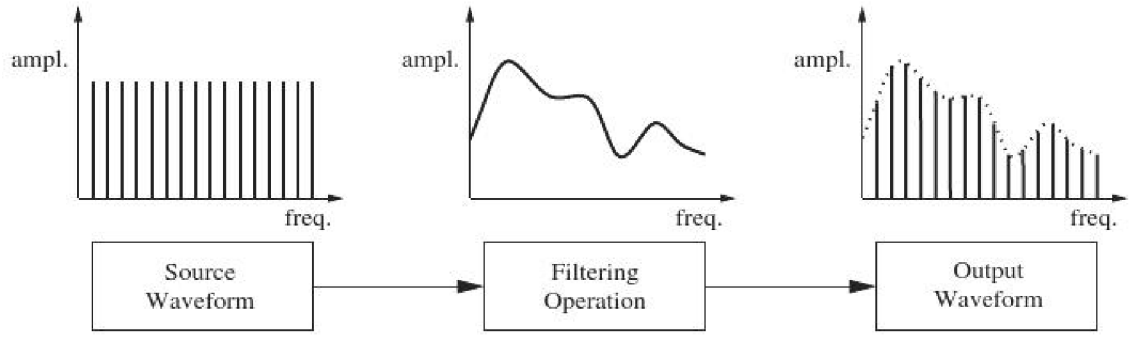


Obrázek 2.2: Aditivní syntéza.

2.2.2 Subtraktivní syntéza

Subtraktivní syntéza [8] je založena na principu, který místo skládání signálu z jednotlivých složek, začíná se signálem bohatým na vyšší harmonické. Vstupní signál je následně filtrován a tím jsou některé frekvence potlačeny a jiné zesíleny. Výstupním signálem je pak signál,

jenž obsahuje pouze požadované frekvence. Subtraktivní syntéza odpovídá způsobu tvorby zvuku v hudebních nástrojích, kde například struna je zdroj signálu a tělo nástroje funguje jako filtr. Princip subtraktivní syntézy je uveden na obrázku 2.3.

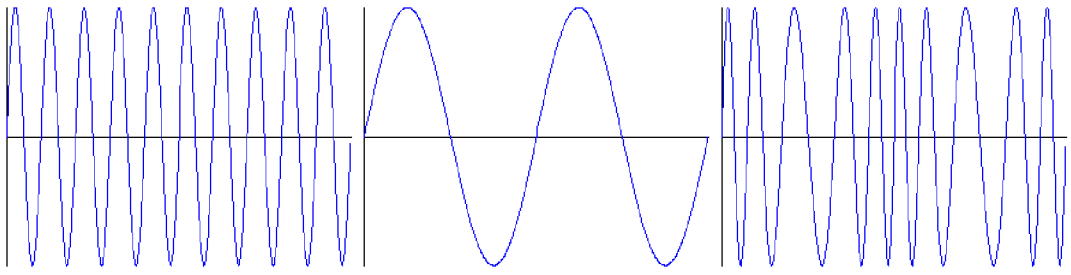


Obrázek 2.3: Subtraktivní syntéza [8].

Jistým omezením této metody je fakt, že nevznikají žádné nové frekvence. Tím pádem není možné generovat tak širokou paletu zvuků, jako je tomu například u aditivní syntézy. Tento nedostatek je možné odstranit kombinací s jinou metodou syntézy. Na druhou stranu má subtraktivní syntéza na rozdíl od aditivní syntézy jednodušší ovládání. Díky tomu je jednou z komerčně nejúspěšnějších metod.

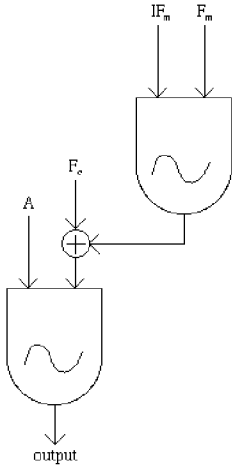
2.2.3 Frekvenční modulace

Metoda ve své nejjednodušší formě využívá dva oscilátory. První oscilátor generuje takzvanou nosnou vlnu a nazývá se nosič viz Obr. 2.4 levá část. Zatímco druhý oscilátor vytváří modulační vlnu a je tím pádem modulátor viz Obr. 2.4 prostřední část. Výstup modulačního oscilátoru je připojen na vstup oscilátoru generujícího nosnou vlnu a ovlivňuje frekvenci jím produkovaného signálu. Výstup modulovaného oscilátoru je na Obr. 2.4 pravá část. Tato základní konfigurace frekvenční syntézy není náročná na výpočetní zdroje, vzhledem k tomu, že se jedná pouze o dva navzájem propojené oscilátory. Základní zapojení je vyobrazeno na obrázku 2.5.



Obrázek 2.4: Princip frekvenční modulace [1].

Dnešní komerčně využívané syntezátory mohou mít pro každý svůj hlas dle [17] i 8 oscilátorů, jako například syntezátor FM8 od Native Instruments. Všechny oscilátory je možné libovolně navzájem propojit a to i tak, že oscilátor může modulovat sám sebe. Jednotlivým oscilátorům je pak možné nastavovat parametry, jako jsou frekvence nebo amplituda.

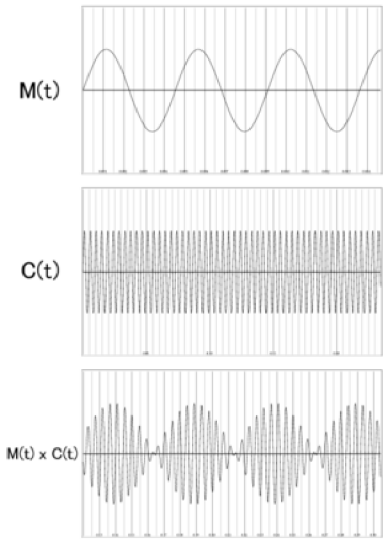


Obrázek 2.5: Základní zapojení frekvenční modulace [1].

Frekvenční modulace i přes svou výpočetní jednoduchost generuje zvuk bohatý na vyšší harmonické. Nevýhodou frekvenční syntézy je těžko předvídatelný výsledek, zejména u složitějších propojení více oscilátorů.

2.2.4 Amplitudová modulace

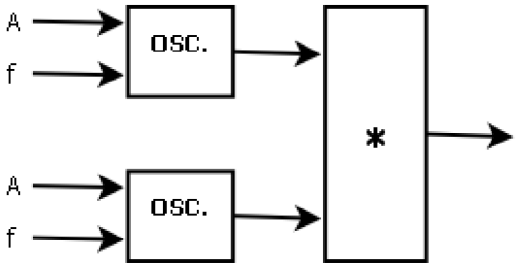
Amplitudová modulace je velmi podobná frekvenční modulaci. Hlavním rozdílem je, že místo frekvence je modulována amplituda nosného signálu. Princip metody demonstruje obrázek 2.6. Schéma zapojení je uvedeno na obrázku 2.7.



Obrázek 2.6: Amplitudová modulace [6].

Amplitudových modulací existuje celá řada. Jednotlivé amplitudové modulace se liší, tím jaké složky zachovávají ve výsledném signálu. Pro účely této práce je nejzajímavější kruhová modulace. Tento typ modulace zcela potlačuje nosnou vlnu a přidává obě postranní

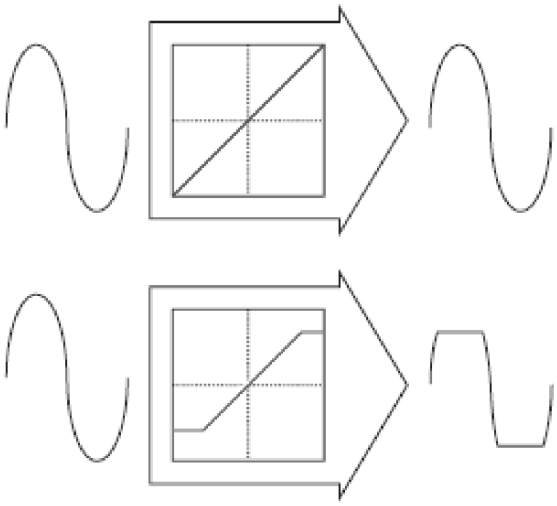
pásma.



Obrázek 2.7: Schéma zapojení amplitudové modulace.

2.2.5 Nelineární tvarování

Metoda nelineárního tvarování je obvykle založena na nelineárních zesilovačích [12]. Tyto zesilovače umožňují kontrolu nad způsobem, jakým zpracovávají vstupní signál. Způsob, jakým zesilovače převádějí vstupní signál na výstupní signál, je popsán pomocí přenosové funkce. Pokud přenosová funkce není lineární, je tvar výstupního signálu pozměněn, viz obrázek 2.8. Změna tvaru křivky přenosové funkce změní příslušným způsobem i tvar výstupního signálu. Zkreslením tvaru signálu se změní jeho spektrum. Ve většině případů jsou harmonické spíše přidávány, než-li odebírány.



Obrázek 2.8: Nelineární tvarování [12].

Pokud je přenosová funkce symetrická kolem horizontální osy nebo má-li rotační symetrii, jsou přidány liché harmonické do výsledného spektra. Jestliže je však přenosová funkce symetrická kolem vertikální osy nebo obsahuje-li zrcadlové symetrie, jsou do výsledného spektra přidány sudé harmonické. Za použití signálu se sinusovým průběhem a nelineárního zesilovače s různými přenosovými funkcemi je možné generovat velkou škálu různých spekter výsledného signálu.

2.2.6 Fázové zkreslení

Fázové zkreslení souvisí podle [12] s nelineárním tvarováním a to tak, že se jedná o jiný pohled na přenosovou funkci. Základem metody je tabulka obsahující navzorkovaný průběh sinusového signálu. Principem metody je čtení z této tabulky s různou rychlostí simulující účinky přenosové funkce na vstupní signál.

2.2.7 Granulační syntéza

V případě granulační syntézy se dle [12] nejedná o zcela obvyklou metodu. Nelze ji ani zařadit do typického modelu zdroje zvuku a jeho modifikátoru. Na místo toho využívá metoda krátkých zvukových částí nebo-li zrn. Jednotlivá zrna mají velmi krátké trvání, mezi 10-100 milisekundami. Podobně jako film využívá nedokonalosti lidského oka, využívá granulační metoda nedokonalosti lidského sluchu. Pokud jsou dva zvuky přehrány rychle za sebou, lidské ucho je považuje za jeden souvislý zvuk. Vzdálenost dvou zrn musí být 10-50 milisekund, což je limit lidského ucha pro rozlišení dvou nezávislých zvuků. Hlavními parametry granulační syntézy jsou počet zrn za jednotku času, jejich frekvenční obsah a jejich amplituda. Každé zrno může být jiné. Zrno může obsahovat kupříkladu různé průběhy s různými frekvencemi nebo hluk, jenž byl filtrován filtrem typu pásmová propust.

Tato metoda syntézy zvuku se příliš neprosadila v klasických hardwarových syntezátorech, ale s nástupem softwarových syntezátorů její popularita vzrostla.

2.2.8 Samplovací syntéza

V případě samplovací syntézy [17] a [3] se nejedná o syntézu v pravém smyslu slova. Samplovací syntéza totiž přímo žádný zvuk negeneruje, ale využívá navzorkovaných zvuků. Vzorky jednotlivých zvuků jsou uloženy v paměti nástroje a jsou následně přehrávány. Na zvuky uložené v paměti nejsou kladeny žádné limity, může se tedy jednat o zvuky reálných nástrojů, zvuky produkované jinými syntezátory nebo různé hluky.

Kvalita nástrojů, založených na samplovací metodě, je ovlivněna především vzorkovací frekvencí a počtem bitů použitých pro reprezentaci jednotlivých vzorků. Vzorkovací frekvence musí dodržet vzorkovací teorém. Dodržení vzorkovacího teorému spočívá ve vzorkování signálu nejméně dvojnásobnou frekvencí, než je nejvyšší frekvence vzorkovaného signálu. Při vzorkování dvojnásobnou frekvencí je následně zaručena korektní rekonstrukce původního signálu.

Největší nevýhodou samplovací metody je nemožnost výsledný zvuk během hraní výrazným způsobem měnit. Tento jev je zapříčiněn podstatou metody, kterou je práce s předem vytvořenými vzorky, jež není možné v průběhu hraní výrazněji upravovat. Další nevýhodou může být fakt, že pokud chceme pomocí této metody dokonale simulovat nějaký hudební nástroj, potřebujeme velké množství vzorků. Výsledný tón, například u bicích, totiž závisí i na typu paličky použité při hraní nebo na způsobu, jakým je udeřeno kupříkladu do činelu. Činel, pokud je rozezvučen úderem do svého středu, vydá jiný zvuk, než pokud je udeřeno na jeho okraj.

Na všechny tyto vlastnosti musí být brán zřetel a výsledný syntezátor musí tedy obsahovat příslušné množství vzorků, což je paměťově náročné. V dnešní době je však tato skutečnost smazána relativně nízkými cenami pamětí.

2.2.9 Fyzikální modelování

Fyzikální modelování [3] a [17] bylo představeno v roce 1993. Jedná se o metodu se zcela odlišným přístupem k syntéze zvuku. Princip této metody vychází z pozorování způsobu, jakým je utvářen zvuk klasickými hudebními nástroji a jaké parametry mají největší vliv na produkovaný zvuk.

U dechových nástrojů má na zvuk vliv například materiál, ze kterého je nástroj vyroben, vlastnosti plátku, hubičky, tvar vzduchového válce uvnitř nástroje. Všechny získané znalosti byly poté použity pro sestavení matematického modelu reprezentujícího příslušný nástroj.

V začátcích, které se datují do 80. let minulého století, byl problém s velkou výpočetní náročností těchto modelů. S narůstajícím výpočetním výkonem a důkladnou optimalizací modelů bylo následně možné tuto metodu využít pro konstrukci syntezátoru. Syntezátor na principu fyzikálního modelování funguje tak, že do matematického modelu jsou zadány příslušné parametry a výsledkem simulace je zvuk.

Tato metoda na rozdíl od ostatních metod nemodeluje tvorbu zvuku, jak je tomu například u aditivní metody, ale modeluje samotný hudební nástroj. Díky tomuto faktu je výstup metody fyzikálního modelování předvídatelný. Pokud tedy změníme třeba velikost rezonátoru, dá se předpokládat, že se příslušným způsobem změní i výsledný zvuk. Zvětšením rezonátoru se stane zvuk hlubším přesně tak, jak by tomu bylo u reálného hudebního nástroje. Syntezátory založené na této metodě jsou ovládány pomocí ovladačů, jež velice připomínají modelovaný hudební nástroj.

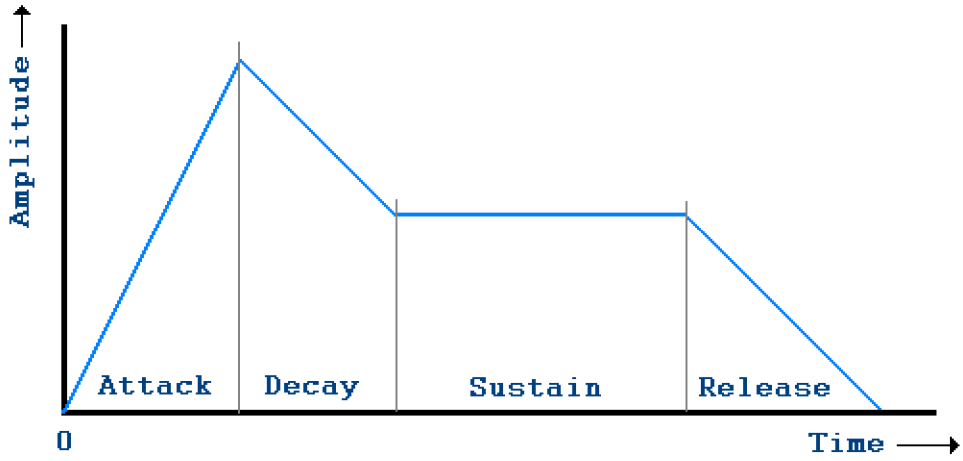
2.3 Amplitudová obálka

Technikou, která zvuk přímo negeneruje, ale používá se pro jeho zlepšení, je metoda amplitudové obálky. Tato technika je nezávislá na použité metodě generující zvuk. Tón produkovaný reálným hudebním nástrojem nemá po celou dobu svého znění konstantní hlasitost, ale jeho hlasitost se s časem mění. Některé nástroje mají rychlý nástup z nulové do maximální amplitudy, po dosažení maximální amplitudy následuje pokles na poloviční úroveň, na které tón setrvává, dokud je zmáčknuta klávesa. Po uvolnění klávesy pak dlouho odeznívá. Jednotlivé nástroje se průběhem hlasitosti při znění tónu liší, což do značné míry utváří jejich unikátní zvuk. Tyto vlastnosti jednotlivých nástrojů se snaží modelovat amplitudové obálky. Amplitudových obálek je několik druhů. Jednotlivé druhy se od sebe liší především počtem segmentů, na něž je obálka rozdělena.

Nejběžněji používané jsou ADSR obálky, skládající se ze čtyř segmentů. U jednotlivých segmentů je možné nastavovat dva parametry, rychlost a úroveň. V praxi se však pro každý segment využívá pouze jeden parametr. Obecný tvar ADSR obálky je na obrázku 2.9.

Obálka je popsána následujícími parametry:

- Attack (A) - rychlost, s jakou signál naroste z nulové amplitudy do své maximální amplitudy
- Decay (D) - rychlost, s jakou signál klesne ze svého maxima na úroveň nastavenou parametrem Sustain
- Sustain (S) - úroveň, na které setrvává amplituda po dobu stisku klávesy
- Release (R) - rychlost, s jakou klesne amplituda signálu z úrovně Sustain na nulu po uvolnění klávesy



Obrázek 2.9: ADSR obálka [19].

ADSR obálka je generována pomocí generátoru obálky, jehož výstup je přiveden na vstup amplitudového modulátoru ovlivňujícího příslušným způsobem amplitudu signálu připojeného na jeho druhý vstup.

Kapitola 3

Sound Interface Device (SID)

Čip 6581 SID je zvukový čip, který býval v různých verzích dle [7] součástí počítačů Commodore 64, Commodore 128 a Commodore MAX. SID dohromady s grafickým čipem VIC-II udělal z počítače Commodore 64 jeden z nejlépe prodávaných počítačů.

Zvukový čip byl vyvinut Robertem Yannesem, který vedl tým složený z dalších dvou techniků a CAD operátora. SID byl dokončen ve velmi krátké době, a to za pět měsíců v roce 1981. Rychlost, s jakou byl čip dokončen, měla však negativní vliv na jeho kvalitu, protože některé vlastnosti nebylo možné v tak krátkém období dokončit. Kupříkladu čip měl být původně 32 hlasý s tím, že všechny hlasy budou sdílet jediný oscilátor. Další vlastností, která ale byla vynechána kvůli nedostatečné ploše čipu, byla frekvenční vyhledávací tabulka. Frekvenční tabulka by zabírala značnou plochu čipu. Vlastnost, která byla implementována i přesto, že v počítači neměla reálné využití, byl vstup pro audio signál. Tento vstup ale umožnil rozšíření čipu jako jednoduchého efektového procesoru. Na výrobu čipu byla použita $7\mu m$ technologie.

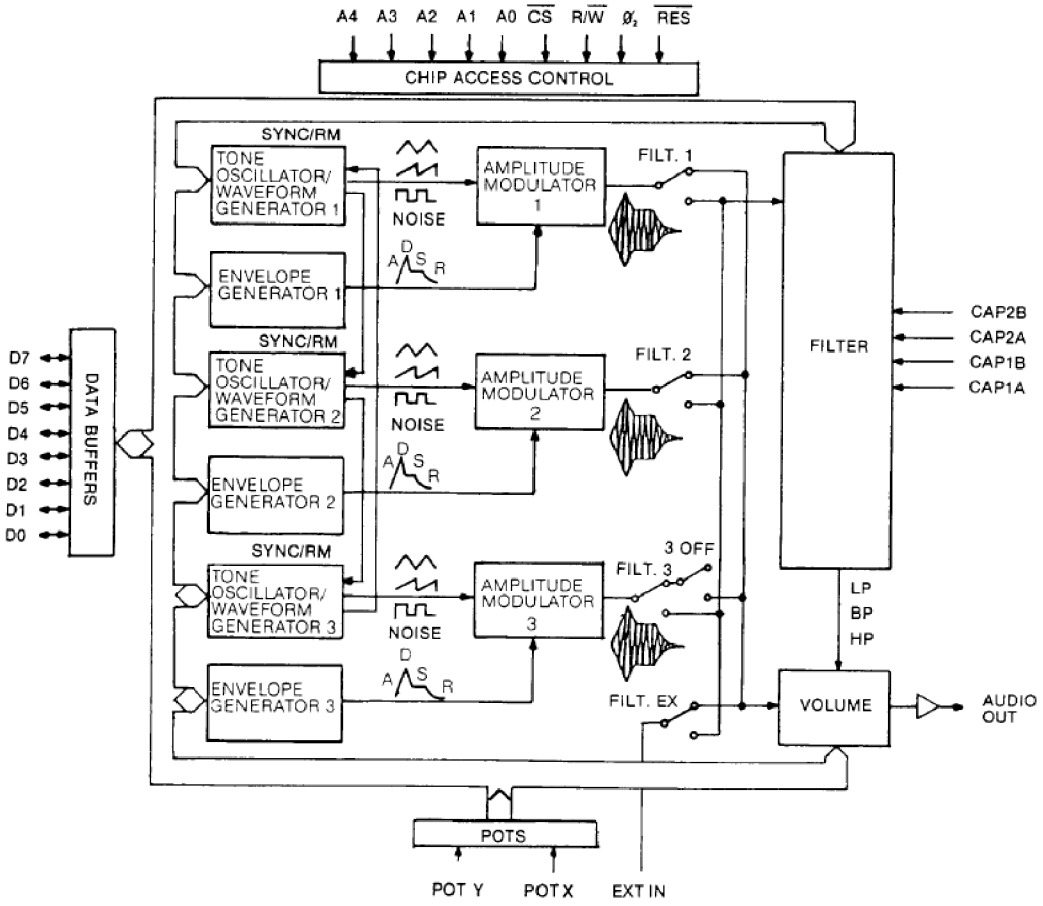
SID 6581 byl stejně jako počítač Commodore 64, jehož byl součástí, představen na veletrhu CES (Consumer Electronic Show) v roce 1982. I v dnešní době má SID řadu obdivovatelů a existují různé programy, jež se snaží s větším či menším úspěchem emulovat jeho zvuk.

3.1 Popis SID 6581

SID 6581 [5] je integrovaný obvod spojující v sobě jak digitální, tak i analogovou část. Všechny kontrolní porty jsou digitální, výstupní audio port a port pro vstup externího audia jsou analogové.

Na obrázku 3.1 je zobrazena vnitřní struktura čipu. SID se skládá ze tří hlasů, z nichž každý se skládá z oscilátoru, generátoru obálky a modulátoru amplitudy. Výstupem z oscilátoru může být signál ve tvaru pily, trojúhelníku, obdélníku nebo výstupem může být šum. Jako výstup oscilátoru může být použita i kombinace dvou a více průběhů, v tom případě se jedná o logický AND těchto průběhů. Generované průběhy jsou bohaté na vyšší harmonické. Každý oscilátor je propojen s předchozím oscilátorem, tím je umožněna kruhová modulace nebo synchronizace oscilátorů. Dynamika hlasitosti je řízena modulátorem amplitudy, který je řízený generátorem obálky. Generátor obálek vytváří amplitudovou obálku s programovatelným průběhem.

Další částí je programovatelný filtr umožňující generování komplexního tónu pomocí subtraktivní syntézy. Filtr je dle [13] napěťově řízený filtr neboli VCF (Voltage Controlled



Obrázek 3.1: Vnitřní struktura SID 6581 [5].

Filter) a jedná se o zcela analogový filtr, který ke své činnosti potřebuje dva z vnějšku připojené kondenzátory. Jestliže jsou k filtru připojeny kondenzátory s kapacitou 2200 pF, je možné dle [5] nastavit frekvenci v rozsahu 30 Hz až 12 kHz. Filtr je možné nastavit jako dolní propust, horní propust, pásmovou propust nebo pásmovou zádrž.

Oscilátor a generátor obálky třetího hlasu umožňují, aby byly jejich výstupy čteny mikroprocesorem. Mohou tedy fungovat jako generátory náhodných čísel nebo jako zdroje pro tvorbu efektů, jako je vibráto atd. Pokud je tento hlas použit jako generátor náhodných čísel, je možné jeho výstup odpojit od audio výstupu.

SID čip z programátorského hlediska obsahuje 29 osmibitových registrů, z nichž některé utvářejí dohromady registry s větší datovou šířkou. Registry jsou rozděleny do pěti skupin. U prvních tří skupin se jedná o tři sady stejných registrů, z nichž každá nastavuje parametry jednomu ze tří hlasů. Další skupinou jsou registry pro nastavení parametrů filtrů. Do těchto čtyř skupin je možné data pouze zapisovat. Poslední skupinou registrů, ze kterých je možné data pouze číst, jsou registry obsahující data z připojených potenciometrů, výstupy oscilátoru a generátoru obálky třetího hlasu.

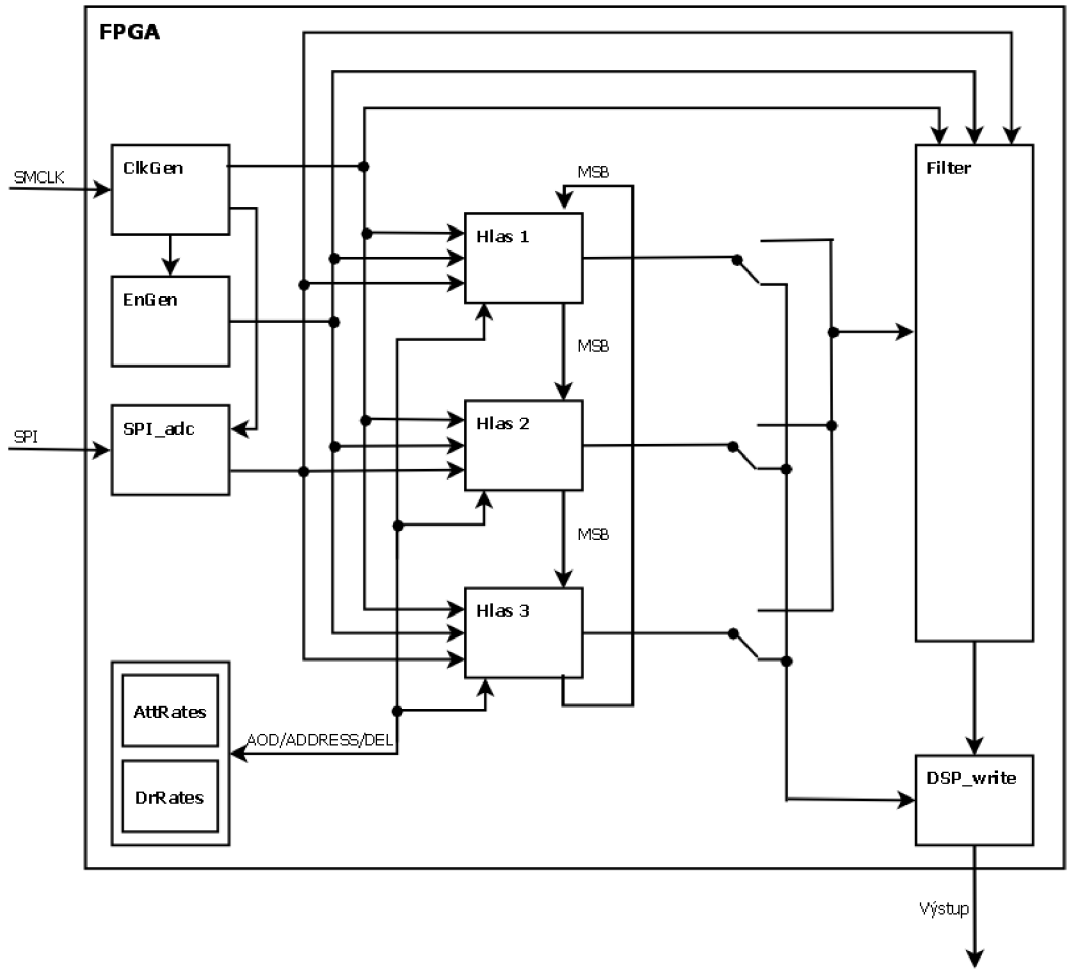
Kapitola 4

Návrh

Syntezátor, implementovaný v této práci, je zobrazen na obrázku 4.1. Hlavními částmi, jejichž návrh je probrán v následujícím textu, jsou oscilátor, generátor obálky, amplitudový modulátor a filtr. Do výsledného syntezátoru byl také zahrnut generátor hodinového signálu, na obrázku označen jako *ClkGen*. Tato komponenta je již implementována a dostupná pro FITkit. Úkolem generátoru hodinového signálu je vytvořit z hodinového signálu, který je připojen na FPGA, signál s vyšší frekvencí. V našem případě byl hodinový signál, tvořený touto komponentou, nastaven na frekvenci 39,8131 MHz. Tento hodinový signál je pak používán k řízení celého syntezátoru a v dalším textu je považován za hlavní hodinový signál. Z hodinového signálu je dále odvozen povolovací signál s frekvencí 995327,5 Hz, na obrázku označen jako *EnGen*. Povolovací signál je připojen ke všem zde navrženým komponentám a řídí jejich činnost. Další komponentou jsou vyhledávací tabulky *AttRates* a *DrRates*, které jsou na obrázku zobrazeny dohromady. Vyhledávací tabulky jsou připojeny k ostatním komponentám třemi signály. Na obrázku 4.1 jsou pro přehlednost tyto signály spojeny do jednoho. Více o smyslu těchto vyhledávacích tabulek je uvedeno v kapitole o generátoru obálky.

Komponenty oscilátor, generátor obálky a amplitudový modulátor tvoří dohromady jeden hlas, jehož schéma je na obrázku 4.2. Na obrázku je uvedeno propojení jednotlivých komponent. Dále jsou zde zobrazeny registry, které ovládají činnost těchto komponent. Nejzajímavějším registrem je kontrolní registr, který nastavuje chování celého hlasu. Pomocí kontrolního registru je možné nastavit tvar generovaného průběhu nebo nastavit synchronizaci a kruhovou modulaci. Všechna tato nastavení ovlivňují pouze chováni oscilátoru. Jediný bit kontrolního registru, který ovlivňuje generátor obálky je *gate bit*. *Gate bit* spouští generování obálky. Dokud není *gate bit* nastaven, negeneruje hlas žádný zvuk. Funkce ostatních registrů je uvedena v příslušných kapitolách zabývajících se návrhem oscilátoru a generátoru obálky.

Posledními komponentami, které nebyly rozebrány, jsou komponenty pro komunikaci syntezátoru s komponentami mimo FPGA. Syntezátor je pro svou korektní funkci nutné propojit s mikrokontrolerem a audio kodekem. Mikrokontroler řídí celý syntezátor nastavováním hodnot do registrů syntezátoru. Pro komunikaci s mikrokontrolerem bylo použito rozhraní SPI, na obrázku 4.1 označeno jako *SPI adc*. Výstup audio signálu zajišťuje rozhraní DSP, které umožňuje komunikaci s audio kodekem, na obrázku 4.1 označeno jako *DSP write*. Pro rozhraní SPI a DSP byly použity komponenty, jež jsou dostupné pro FITkit. Jejich návrh ani implementace nebude v následujícím textu rozebrána. Na straně mikrokontroleru byla napsána knihovna funkcí, které zjednodušují nastavení potřebných parametrů zvukového čipu. Jedná se především o funkce pro nastavení audio kodeku a nastavování

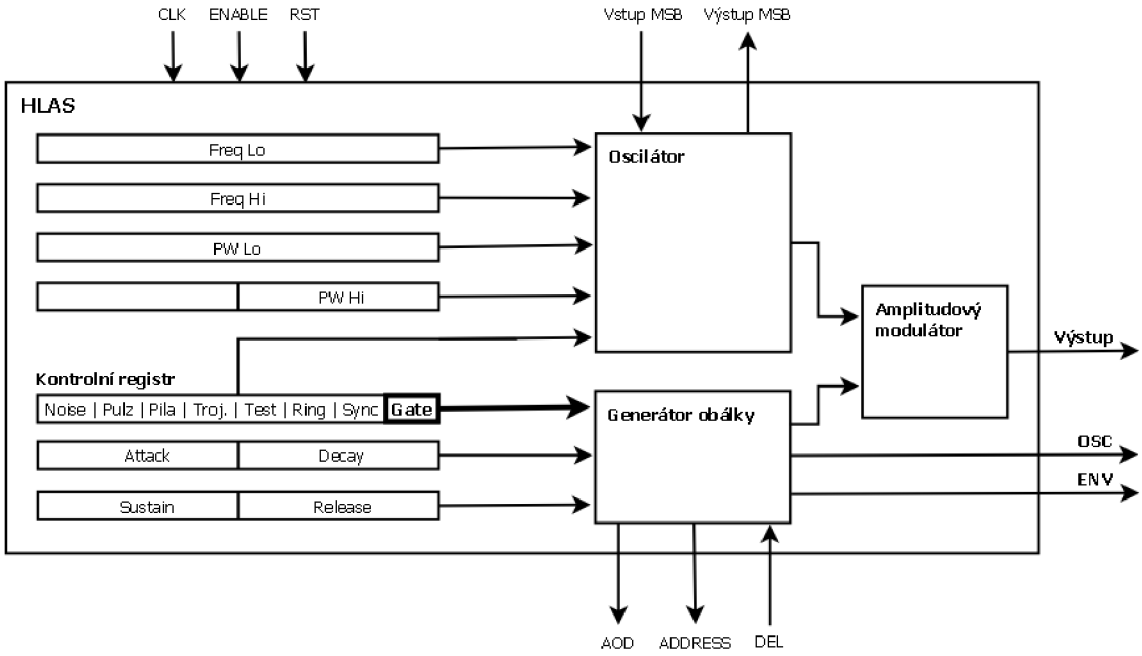


Obrázek 4.1: Vnitřní struktura navrhovaného syntezátoru.

parametrů zvukového čipu. Kompletní seznam implementovaných funkcí včetně krátkého popisu je uveden v příloze A.

4.1 Rozhraní

Komunikace s mikrokontrolerem je zajištěna pomocí rozhraní SPI. Všechny potřebné komponenty jsou již implementovány. Je potřeba je pouze správně nakonfigurovat a propojit s okolními komponentami. Rozhraní využívá dvou komponent, SPI řadiče a SPI adresového dekodéru. SPI řadič zajišťuje samotnou komunikaci, převádí paralelní data přijatá z mikrokontroleru na sériovou sběrnici. Na SPI řadič je pak připojen SPI adresový dekodér, který rozpoznává, zda jsou přenesená data určena pro komponentu, na kterou je připojen. Parametry SPI dekodéru jsou šířka přenášených dat, šířka adresy, bázová adresa a šířka výstupní adresy. Tyto parametry byly nastaveny na následující hodnoty: šířka přenášených dat - 8 bitů, šířka adresy - 8 bitů, bázová adresa - 0, šířka výstupní adresy - 5 bitů. Bázová adresa byla nastavena na nulu proto, aby adresy registrů odpovídaly přesně adresám registrů originálního čipu. Registrová sada je tedy převzata ze SID 6581 a je uvedena v příloze B. Data jsou po úspěšném přenosu následně zapisována do příslušných registrů. Z



Obrázek 4.2: Schéma zapojení hlasu.

některých registrů jsou data také přenášena zpět do mikrokontroleru. Směr přenosu je řízen mikrokontrolerem. Více o rozhraní SPI implementovaného na FITkitu je možné nalézt v [18].

Výstup audio signálu je zajištěn pomocí rozhraní DSP. Pro toto rozhraní jsou na platformě FITkit implementovány dvě komponenty. První je určena pro čtení dat z audio kodeku, zatímco druhá umožňuje zápis dat do audio kodeku. Obě komponenty předpokládají, že je audio kodek nastaven jako master. Z toho vyplývá, že všechny potřebné signály pro synchronizaci komunikace generuje sám audio kodek a z FPGA jsou pouze čtena nebo zapisována data jednotlivých audio vzorků. Nastavení audio kodeku je nutné provést z mikrokontroleru, jenž je připojen pomocí SPI na vstupy audio kodeku, přes něž jsou nastavovány jeho parametry. Vzhledem k tomu, že zde implementovaný čip data pouze zapisuje, byla použita druhá komponenta. Jediné možné nastavení v případě této komponenty je šířka vzorku pro jeden kanál, v tomto případě 16 bitů. Jednotlivé vzorky jsou pak posílány na základě synchronizačního signálu, jehož frekvence se odvíjí od nastavené vzorkovací frekvence. Více detailů o obou komponentách je v [15] a podrobnosti o způsobu nastavení audio kodeku z mikrokontroleru je možné nalézt v [14].

4.2 Oscilátor

Oscilátor je základní částí celého syntezátoru. Jeho úkolem je generování signálu s požadovaným tvarem a frekvencí. Oscilátor se skládá ze samotného oscilátoru, generujícího požadovanou frekvenci a generátoru průběhu. Generátor průběhu pak z dat z oscilátoru vytváří jeden ze čtyř možných průběhů. Parametry oscilátoru jsou nastavovány pomocí registrů *Freq Lo*, *Freq Hi*, *PW Lo*, *PW Hi* a kontrolního registru. *Freq Lo* a *Freq Hi* tvoří dohromady 16 bitový registr ovlivňující generovanou frekvenci. Registry *PW Lo* a *PW Hi* tvoří 12 bitový registr, který nastavuje střídu signálu v případě, že výstupním signálem je

pulz. Na ostatní průběhy nemají tyto registry žádný vliv. Pomocí kontrolního registru je možné vybrat výstupní průběh, případně resetovat oscilátor nastavením *test bitu*. Dále je možné nastavit kruhovou modulaci nebo synchronizaci dvou oscilátorů.

Jádrem oscilátoru je 24 bitový čítač, ke kterému je přičítána 16 bitová hodnota z registrů *Freq Lo* a *Freq Hi*. Jako výstup je použito horních 12 bitů. Hlavní výhodou takto použitého 24 bitového čítače je fakt, že při změně generované frekvence nedochází k žádným negativním jevům. Čítač je řízen povolovacím signálem, který je odvozen z hlavních 39,8131 MHz hodin a má frekvenci 995327,5 Hz. Generovaná frekvence je závislá na registrech *Freq Lo* a *Freq Hi* a je ji možné spočítat dle následujícího vztahu:

$$F_{out} = (F_n \cdot F_{clk}/16777216)Hz \tag{4.1}$$

Kde F_{out} je výsledná frekvence, F_n je 16 bitové číslo z registrů a F_{clk} jsou hodiny, kterými je čítač řízen. Tabulka hodnot potřebných pro generování 8 oktáv temperovaného ladění je uvedena v příloze C. Hodnoty uvedené v tabulce jsou vypočteny pomocí vztahu 4.1. Hodnota čítače je také ovlivňována nastavením *test bitu*, resetem nebo nastavením synchronizace. Při nastavení *test bitu* nebo při resetu je čítač vynulován. Jediným rozdílem je, že při nastavení *test bitu* je čítač vynulován a navíc je zastavena jeho činnost po dobu nastavení *test bitu*. Při resetu je čítač pouze vynulován a pokračuje ve své činnosti. Nastavením synchronizace je čítač rovněž nulován, ale na základě nejvýznamnějšího bitu z předchozího oscilátoru. V tomto případě je čítač nulován pouze tehdy, přejde-li nejvýznamnější bit předchozího oscilátoru z logické nuly do logické jedničky. Tím pádem se oba oscilátory synchronizují.

Další součástí oscilátoru je generátor průběhu, který je připojen na výstup oscilátoru a pracuje tedy s horními 12 bity čítače. Čip dovoluje generování čtyř průběhů: pila, trojúhelník, pulz a hluk. Signál ve tvaru pily je generován jednoduše posláním vstupu generátoru průběhu přímo na výstup bez jakékoliv změny. Trojúhelník je generován tak, že na dolních 11 bitů vstupu je použita funkce *XOR* s tím, že druhým operandem je nejvýznamnější bit vstupu, který je následně zahozen. Výsledek funkce *XOR* je pak posunut doleva, čímž je zachována frekvence i amplituda, jakou mají ostatní generované průběhy. Použití funkce *XOR* umožňuje také kruhovou modulaci. Kruhové modulace je dosaženo jednoduše tím, že jako druhý operand funkce *XOR* je namísto nejvýznamnějšího bitu vstupu použit nejvýznamnější bit předchozího oscilátoru. Kruhovou modulaci je tedy možné použít pouze v případě, že je jako výstupní průběh zvolena pila. Dalším možným průběhem je pulz s nastavitelnou střídou. Střída udává poměr části cyklu signálu, kdy je signál na své maximální úrovni vůči části, kdy je na nule. Střídu lze nastavit pomocí registrů *PW Lo* a *PW Hi* tvořících 12 bitové číslo. Po dosazení tohoto čísla do vzorce 4.2 je možné určit střídu výsledného signálu.

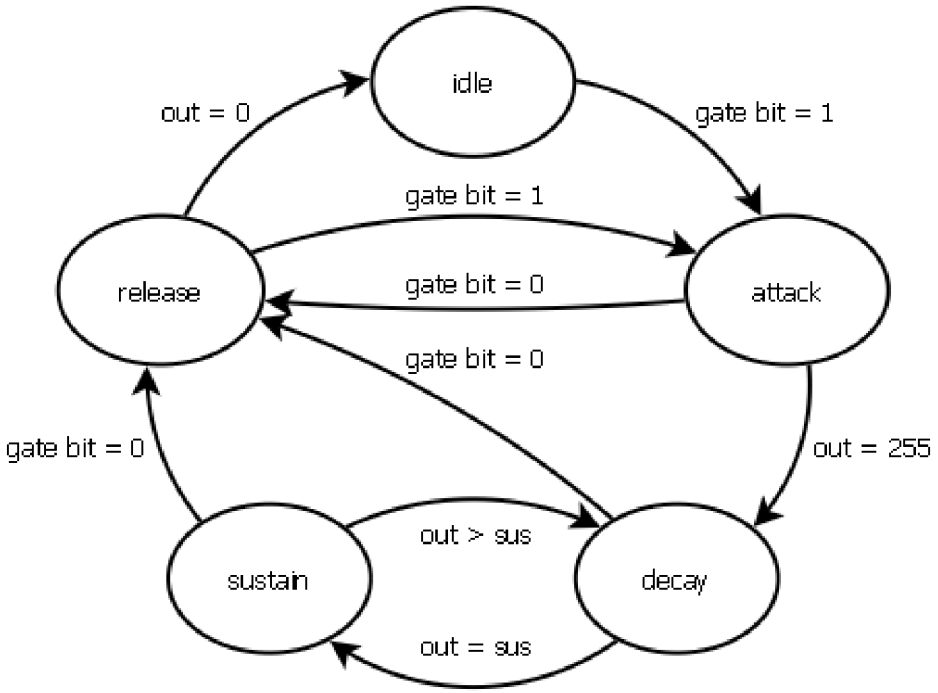
$$PW_{out} = (PW_n/40,95)\% \tag{4.2}$$

Kde PW_{out} je výsledná střída a PW_n je číslo uložené v registrech. Tento průběh je generován za použití komparátoru porovnávajícího vstup s číslem v registrech *PW Lo* a *PW Hi*. Pokud je vstup menší než toto číslo, je výstupem maximální hodnota, jestliže je však větší, je výstupem nulová hodnota. Posledním generovaným průběhem je šum, který je dobře napodobitelný pomocí generátoru pseudonáhodných čísel. V hardwaru je možné celkem dobře pseudonáhodná čísla generovat pomocí posuvného registru, který má na svůj vstup připojen výsledek logické operace s několika svými bity. Použitou logickou funkcí je *XNOR*. Bity vstupující do funkce *XNOR* ale nemohou být jakékoliv bity z posuvného

registru. Musí se jednat o přesně dané bity. Počet a pořadí bitů je závislé na délce posuvného registru. Potřebné bity je možné nalézt například v [2]. Pro účel zde implementovaného čipu byl zvolen 23 bitový posuvný registr, schopný generovat až $2^{23} - 1$ různých čísel. Posuvný registr je časován pomocí jednoho bitu ze vstupu, aby byla zhruba zachována požadovaná frekvence. Pokud dojde k resetu, je posuvný registr nastaven na svou počáteční hodnotu, a tím pádem začíná generovat od začátku celou posloupnost pseudonáhodných čísel. Jako výstup generátoru průběhu může být jeden ze zde uvedených průběhů. Je možná i kombinace více průběhů, ale jedná se pak o logický součin vybraných průběhů.

4.3 Generátor obálky

Výstupem generátoru obálky je hodnota, která koresponduje s aktuálním stavem generátoru. Tato hodnota slouží k modulaci amplitudy signálu produkovaného oscilátorem. Zde generovaná obálka je rozdělena na čtyři segmenty a jedná se o ADSR obálku. Obecný tvar ADSR obálky je na obrázku 2.9. Chování generátoru obálky je ovládáno pomocí registrů *attDec*, *susRel* a kontrolního registru. Registr *attDec* nastavuje rychlost vzestupu z nuly na maximální amplitudu a rychlost poklesu na hodnotu nastavenou registrem *susRel*. Registr *susRel* nastavuje mimo jiné také rychlost poklesu na nulovou hodnotu z hodnoty *sustain*. Z kontrolního registru má na generátor obálky vliv pouze *gate bit*, jehož nastavení spouští generování obálky. Generátor obálky je implementován jako čítač, jehož rozsah je 0 - 255. Čítač je řízen konečným automatem, který určuje rychlost a směr čítání. Diagram přechodů konečného automatu je na obrázku 4.3.



Obrázek 4.3: Konečný automat řídící generátor obálky.

Konečný automat začíná ve stavu *idle*, kde čeká, dokud není v kontrolním registru nastaven *gate bit*. Nastavení *gate bitu* simuluje zmáčknutí klávesy hudebního nástroje a je tedy příkazem k rozezvučení tónu. Do doby, než je nastaven *gate bit*, a za předpokladu,

že nedobíhá předchozí cyklus obálky, je výstup generátoru nulový. Tím pádem tento hlas negeneruje žádný zvuk. Po nastavení *gate bitu* se konečný automat přesune do stavu *attack*, ve kterém je nastavena požadovaná rychlost čítání, směr čítání je nastaven na inkrementaci a je povoleno čítání. Čítač se inkrementuje do té doby, než jeho hodnota dosáhne 255, poté okamžitě automat přechází do stavu *decay*. Ve stavu *decay* je rychlost čítání nastavena tak, aby odpovídala požadované hodnotě v registru *attDec* a směr čítání je změněn na dekrementaci. Při dekrementaci čítače jsou horní čtyři bity porovnávány s hodnotou v registru *susRel*, konkrétně s horními čtyřmi bity. Pokud se obě hodnoty shodují, je zastaveno čítání a automat přechází do stavu *sustain*. Automat může setrvávat ve stavu *sustain* neomezeně dlouhou dobu a to dokud je nastaven *gate bit*. Pokud je během stavu *sustain* změněna hodnota v registru *susRel* na nižší, přechází automat do stavu *decay* do té doby, dokud se hodnoty opět nerovnají. Jestliže je ale hodnota v registru nastavena na vyšší, tak obálka již tuto změnu nereflektuje. Po vymazání *gate bitu* v kontrolním registru přechází automat do stavu *release*, kde je opět povoleno čítání a je nastavena příslušná rychlost tohoto čítání. Jakmile dosáhne čítač hodnoty nula, přechází automat do stavu *idle*. Ve stavu *idle* je zakázáno čítání a automat čeká na nastavení *gate bitu*. Automat může ze kteréhokoliv svého stavu okamžitě po smazání *gate bitu* přejít do stavu *release* a tím zahájit odeznění tónu. Jedinou vyjímkou je samozřejmě stav *idle*, ve kterém *gate bit* není nastaven. Tato vlastnost umožňuje generování širšího množství obálék. Je tedy možné například uprostřed *attack fáze* spustit *release fázi* a tím vytvořit obálku, která bude mít tvar trojúhelníku. Také je možné z *fáze release* nastavením *gate bitu* opět přejít do *attack fáze*.

Čítač je dále řízen signálem, který určuje rychlost čítání. Tento signál je odvozen od povolovacího signálu. Ke snížení jeho frekvence je použita stejná technika, jaká je použita u oscilátoru. Jedná se tedy o 24 bitový čítač, ke kterému je přičítána určitá hodnota. Tyto hodnoty jsou dopočítány podle vzorce 4.1. Vypočtené hodnoty jsou uloženy ve dvou tabulkách. V jedné tabulce jsou hodnoty pro *attack fázi* a ve druhé jsou hodnoty pro *decay* a *release fázi*, které mají stejnou hodnotu pro obě fáze. Při změně rychlosti čítání je adresována patřičná tabulka a vrácená hodnota je použita pro dělení frekvence povolovacího signálu.

4.4 Hlas

Poslední částí zbývající k vytvoření jednoho hlasu je amplitudový modulátor. Úkolem modulátoru amplitudy je úprava amplitudy signálu generovaného oscilátorem podle průběhu ADSR obálky. Takto modulovaný signál je mnohem dynamičtější a lépe modeluje zvuky tvořené reálným hudebním nástrojem. Amplitudový modulátor je možné v hardware udělat jakožto součin vstupních signálů. Z výsledného součinu pak pouze část slouží jako výstup.

Výstupem jednoho hlasu je tedy signál s danou frekvencí a tvarem průběhu, jehož amplituda je modulována přesně definovanou obálkou. Každý hlas má ještě výstup pro nejvýznamnější bit svého oscilátoru a je připojen na vstup následujícího hlasu. Nejvýznamnější bit se, jak již bylo řečeno, používá ke kruhové modulaci a synchronizaci dvou oscilátorů. Dalšími výstupy jsou horních 8 bitů z výstupu oscilátoru a výstup generátoru obálky. Pouze u třetího hlasu jsou však tyto dva výstupy připojeny k registrům. Z těchto registrů je pak může číst mikrokontroler a použít je jako generátor náhodných čísel.

4.5 Filtr

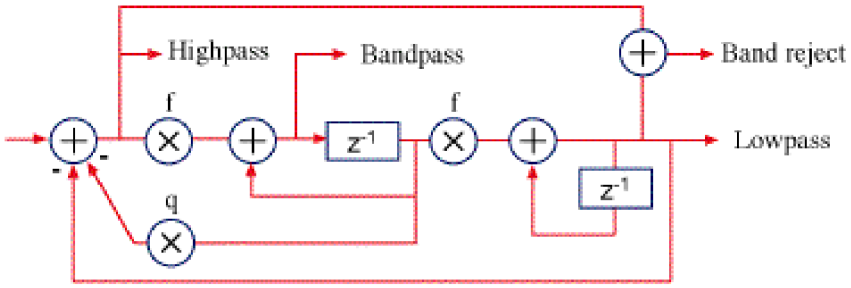
V případě zde implementovaného filtru se jedná o SVF neboli state variable filter [11]. Lépe řečeno o jeho digitální modifikaci. Digitální state variable filter vznikl z analogového state variable filtru náhradou analogových částí za odpovídající digitální části. Hlavní výhodou tohoto filtru je, že během jednoho výpočtu jsou dostupná data z dolní propusti, horní propusti, pásmové propusti a pásmové zádrže. Chování filtru je řízeno dvěma koeficienty. První koeficient řídí frekvenci a druhý určuje rezonanci. Oba koeficienty je možné dopočítat podle následujících vztahů.

$$f = 2 \cdot sin\left(\frac{\pi \cdot F_c}{F_s}\right) \tag{4.3}$$

Kde F_s je vzorkovací frekvence a F_c je ořezová frekvence filtru.

$$q = \frac{1}{Q} \tag{4.4}$$

Kde Q je číslo z rozsahu 0,5 až nekonečno, kde filtr osciluje.



Obrázek 4.4: State variable filter [11].

Na obrázku 4.4 je schéma filtru. Celý filtr je navržen tak, že pracuje s čísly v pevné desetinné čárce. Toto omezení snižuje výpočetní náročnost a zabranou plochu na čipu. Nevýhodou je ale nižší přesnost výpočtu. Filtr je řízen konečným automatem, který v každém taktu hlavních hodin provede jednu operaci. Tím je zajištěna správnost výpočtu a je využita pouze jedna násobička. Po šesti taktech hodin je celý výpočet dokončen a jsou dostupná data pro všechny čtyři druhy filtru.

Kapitola 5

Ověření korektní funkce

V této kapitole byla ověřena korektní činnost všech zde navržených součástí. Jedná se především o oscilátor a k němu připojený generátor průběhu a generátor obálky. Oscilátor a generátor průběhu byly testovány jako jedna jednotka. Dále bylo prověřeno korektní propojení výše uvedených částí, čímž byla ověřena korektní funkce jednoho hlasu. Po ověření jednoho hlasu byly zapojeny dohromady dva tyto hlasy, aby mohly být ověřeny funkce, jako je kruhová modulace a synchronizace oscilátorů. Poslední testovanou jednotkou byl filtr.

Pro testování jednotek napsaných v jazyce VHDL byly vytvořeny simulační modely. Simulace jednotlivých jednotek probíhala v programu ModelSim SE 6.6d. Jednotlivým testovaným částem je nutné během simulace posílat nějaké vstupy. Pro generování vstupů během simulace se používají speciální jednotky tzv. testbenche. Jedná se o standartní programy v jazyce VHDL. Testbench obsahuje instanci testované jednotky a poskytuje jí vstupy, jako je například hodinový signál atd. Vstupy je možné v průběhu simulace měnit a pozorovat reakce testované jednotky na tyto změny.

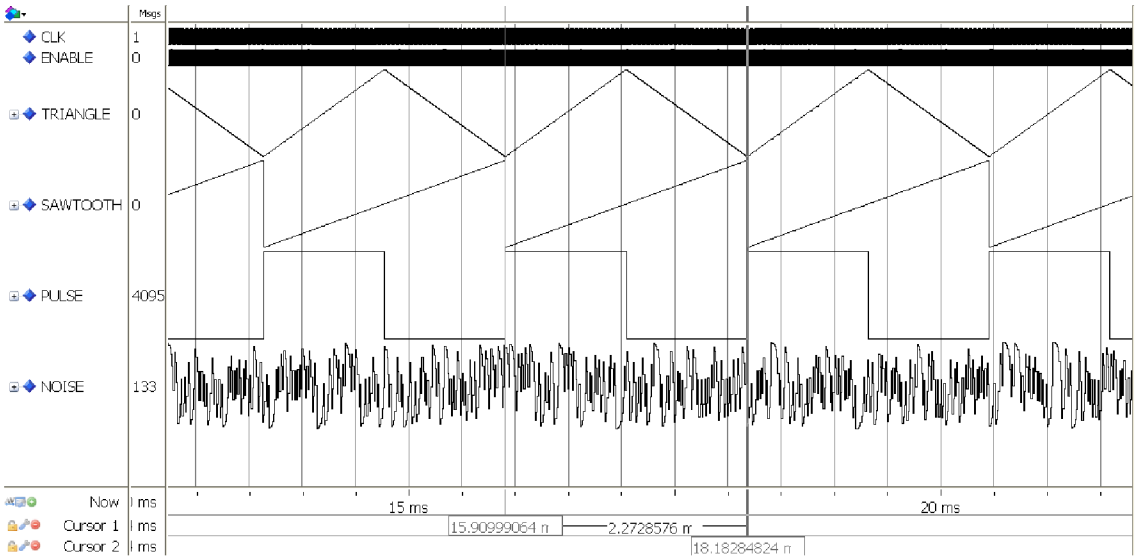
5.1 Oscilátor

U oscilátoru bylo ověřeno, jestli signál jím generovaný má požadovanou frekvenci. Také bylo nutné ověřit, jestli všechny čtyři generované průběhy mají stejnou amplitudu a požadovaný tvar. U signálu ve tvaru pulzu bylo nutné otestovat, zda je střída signálu měněna dle změn v příslušných registrech. V neposlední řadě bylo sledováno chování oscilátoru na změnu frekvence. Při změně frekvence může docházet k problémům, zejména při změně z nízké frekvence na vysokou frekvenci. Zde implementovaný oscilátor by však touto chybou trpět neměl a přechod by měl být plynulý.

Nejprve bylo ověřeno generování požadované frekvence. Frekvence byla nastavena na 440 Hz, což je frekvence komorního A temperovaného ladění. Střída u pulzu byla nastavena na 50 %. Délka jedné periody by tedy měla být 2,2727 ms. Výstup ze simulace je na obrázku 5.1.

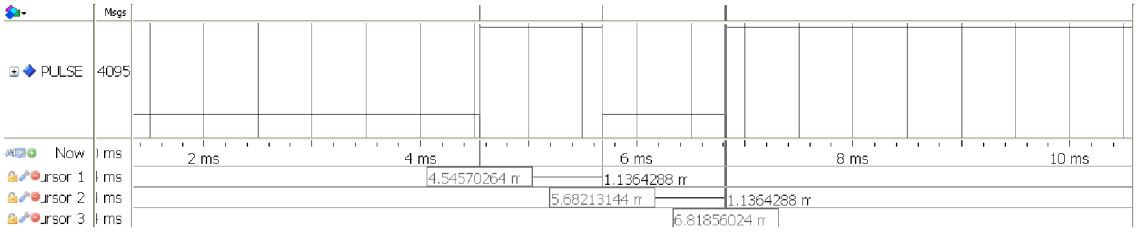
Dle simulace je délka jedné periody generovaného signálu 2,2728576 ms, což odpovídá frekvenci 439,97477 Hz. Jedná se tedy o chybu 0,02523 Hz (0,00573409091 %), způsobenou nepřesným generováním hodinového signálu, kterým je celý čip řízen. Na obrázku 5.1 je také vidět, že tvary signálů odpovídají požadovaným průběhům.

Dále byla ověřena korektní funkce změny střídy u signálu ve tvaru pulzu. Do registrů kontrolujících střídu byly postupně nastaveny tyto hodnoty: 0, 2048, 4095. Po dosazení těchto hodnot do vzorce 4.2 vychází, že je střída nejprve 0 %, 50 % a pak 100 %. Na



Obrázek 5.1: Simulace oscilátoru - frekvence 440 Hz.

obrázku 5.2 je výsledek simulace.



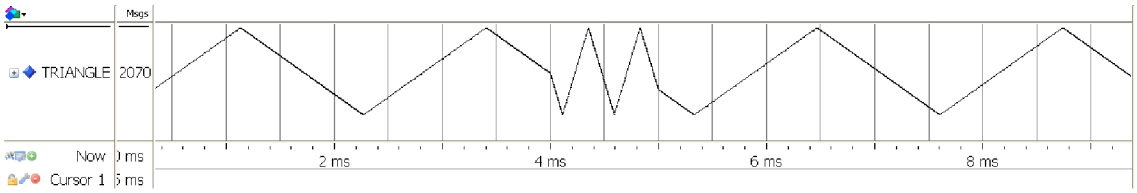
Obrázek 5.2: Změna střídy signálu.

Obrázek 5.2 ukazuje výsledný signál, který má ve své první fázi střídu 0 %. Poté byla střída změněna na 50 %. Tuto změnu je možné pozorovat v prostřední části signálu. Časový úsek, kdy je signál na maximální hodnotě, se shoduje s délkou časového úseku, ve kterém je signál na nulové hodnotě. Signál má tedy střídu 50 %. V poslední části má signál nastavenu střídu 100 %. Signál tedy v této části setrvává na své maximální hodnotě.

Poslední ověřovanou vlastností oscilátoru byla jeho plynulost při změně frekvence. Frekvence byla z komorního A změněna na C v 7. oktávě s frekvencí 2093 Hz. Ke změně by mělo dojít v čase 4 ms. O další 1 ms byla frekvence opět změněna zpět na frekvenci komorního A. Výstup ze simulace je vidět na obrázku 5.3. Z obrázku je zřejmé, že ke změně frekvence dojde okamžitě bez jakéhokoliv zpoždění v časech 4 ms a 5 ms. Změna je zcela plynulá. Oscilátor tedy funguje dle očekávání. Generované průběhy mají požadovanou frekvenci i tvar. Změny všech parametrů jsou plynulé a nedochází k žádným slyšitelným chybám.

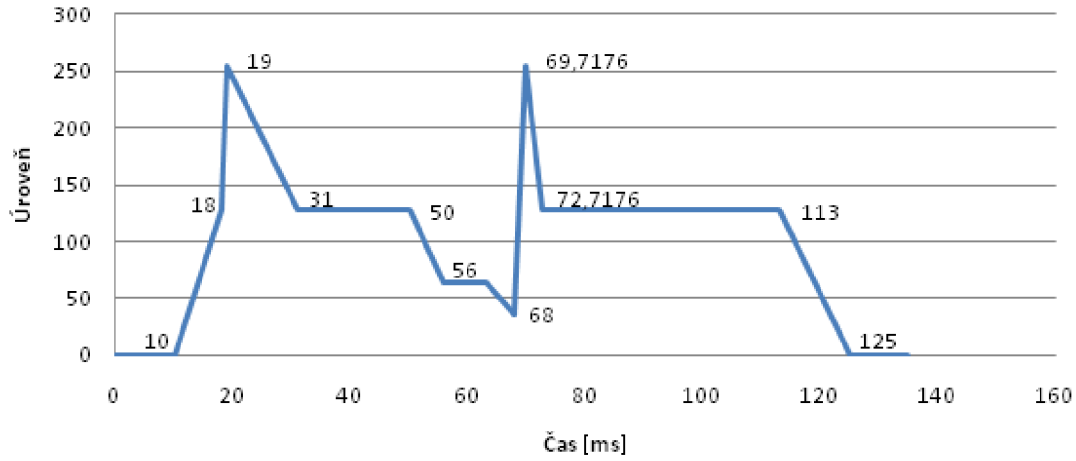
5.2 Generátor obálky

Hlavní vlastností generátoru obálky, která byla testována, je strmost jednotlivých fází obálky. Strmost se liší podle toho, jak dlouho má každá fáze trvat. Dále byl testován přechod z jedné fáze na jinou bez ukončení předchozí fáze, například započetí nové *attack fáze*



Obrázek 5.3: Změna frekvence z A4 na C7 a poté zpět na A4.

bez dokončení *release fáze*. Pro testování generátoru obálky byl napsán testbench, který nastavuje jednotlivé parametry generátoru. Parametry byly nastaveny tak, že nejprve bylo trvání *attack fáze* nastaveno na 8 ms, *decay* na 24 ms, *sustain* úroveň byla nastavena na půlku amplitudy a *release* na 48 ms. Po 10 ms byl nastaven *gate bit*, a tím bylo spuštěno generování obálky. V čase 18 ms bylo trvání *attack fáze* změněno na 2 ms, což způsobilo strmější nárůst obálky. Po dosažení amplitudy následovala *decay fáze* a po ní *sustain fáze*. Obálka setrvávala na poloviční úrovni amplitudy do času 50 ms. V čase 50 ms byla nastavena nižší *sustain* úroveň, kterou by měla obálka sledovat dolů. Změněné *sustain* úrovně dosáhla obálka v čase 56 ms. Za dalších 7 ms byl smazán *gate bit* a následovala *release fáze*, která byla v čase 68 ms přerušena opětovným nastavením *gate bitu*. V témže čase byla také změněna rychlost *decay fáze* na 6 ms. Po *decay fázi* následovala *sustain fáze*, která byla v čase 113 ms přerušena smazáním *gate bitu* a byla zahájena *release fáze*. Předpokládaný tvar obálky, včetně výše uvedených časů jednotlivých akcí, je na obrázku 5.4.

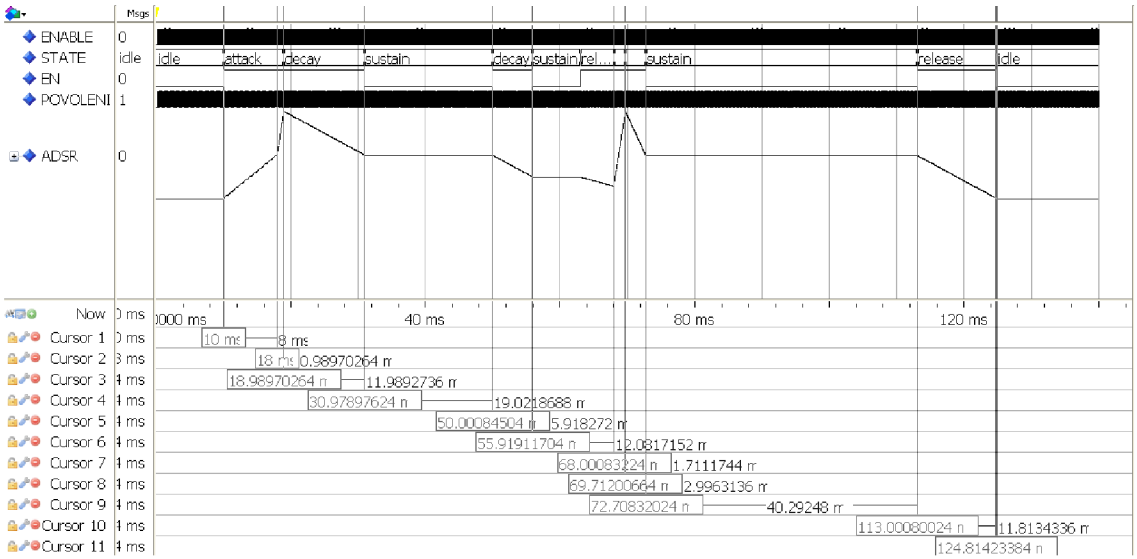


Obrázek 5.4: Předpokládaný průběh obálky.

Výsledek simulace je uveden na obrázku 5.5. Z obrázku je patrné, že se jednotlivé časy s malými odchylkami shodují s předpokládanými časy. Tím pádem se shoduje i tvar simulované obálky s tvarem předpokládaným podle hodnot nastavovaných během simulace.

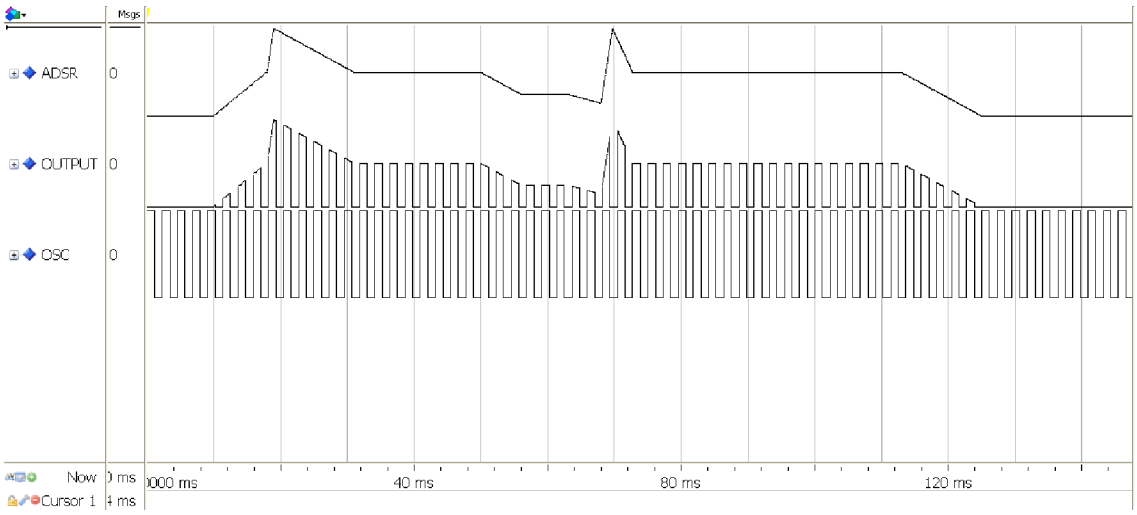
5.3 Hlas

V případě hlasu bylo možné testovat pouze správné propojení oscilátoru s generátorem obálky. Jedná se vlastně o testování amplitudového modulátoru. Očekávaným výsledkem simulace byl signál, jehož amplituda byla modulována podle amplitudové obálky. Jednotlivé



Obrázek 5.5: Simulace generátoru obálky.

části byly nastaveny tak, že oscilátor generoval obdélníkový signál o frekvenci 440 Hz. Generátor obálky generoval stejnou obálku jako v předchozí kapitole. Výsledný signál je vidět na obrázku 5.6.

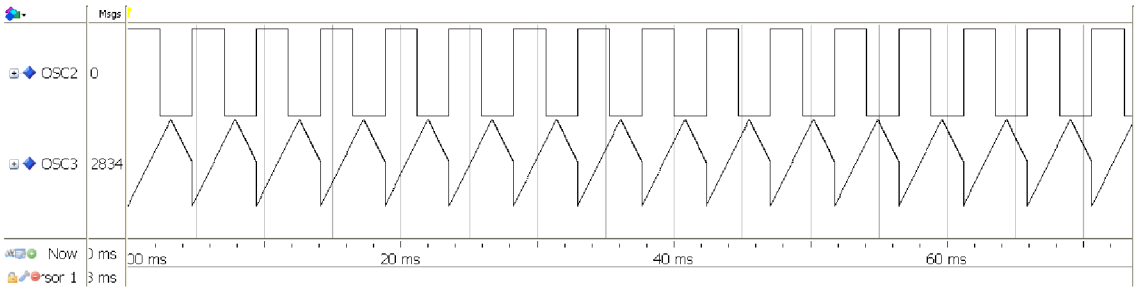


Obrázek 5.6: Výstup hlasu.

5.4 Synchronizace a kruhová modulace

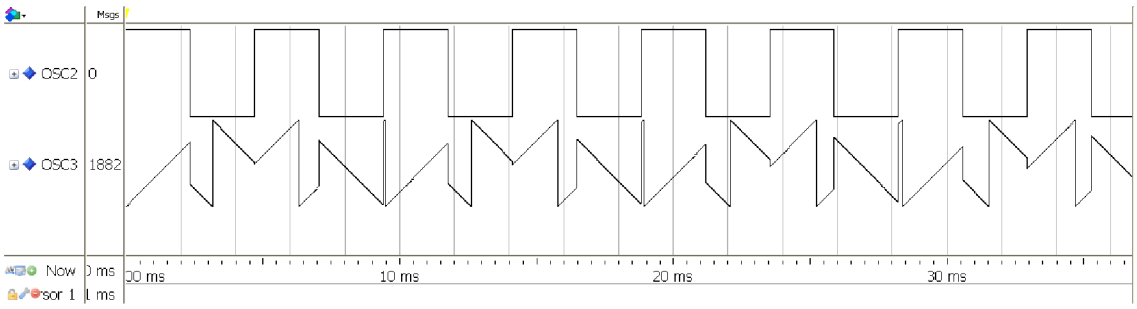
Poslední vlastností, kterou bylo nutné otestovat, byla synchronizace oscilátorů a kruhová modulace. Tyto vlastnosti lze otestovat pouze při spojení nejméně dvou oscilátorů. Nejvýznamnější bit oscilátoru je připojen na vstup následujícího oscilátoru a ovlivňuje jím produkovaný průběh. Nejprve byla testována synchronizace dvou oscilátorů. Synchronizace probíhá tak, že synchronizovaný oscilátor je vynulován na základě přechodu nejvýznamnějšího bitu předchozího oscilátoru z logické 0 na logickou 1. Tím dochází k synchronizaci

oscilátorů. Perioda synchronizovaného oscilátoru může být přerušena a zahájena nová perioda. Tento jev je uveden na obrázku 5.7.



Obrázek 5.7: Synchronizace dvou oscilátorů.

Kruhová modulace má slyšitelný výstup pouze tehdy, je-li jako výstup modulovaného oscilátoru použit signál ve tvaru trojúhelníku. Poté je použit nejvýznamnější bit předchozího oscilátoru jako vstup do funkce *XOR*, která normálně generuje trojúhelníkový průběh. Více o generování signálu ve tvaru trojúhelníku bylo uvedeno v kapitole 4.2. Výsledek modulace oscilátoru průběhem ve tvaru obdélníku je na obrázku 5.8.



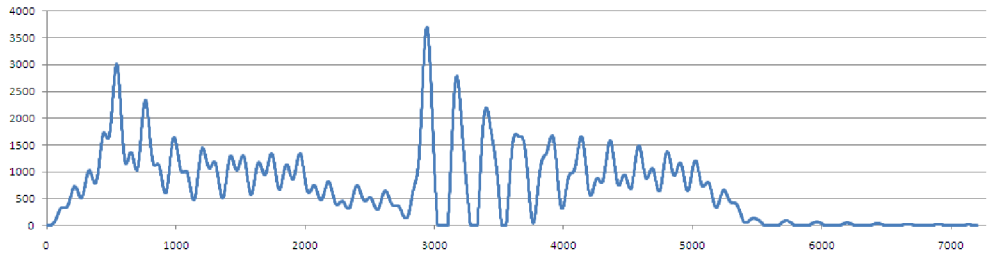
Obrázek 5.8: Kruhová modulace.

V simulaci obě techniky, jak synchronizace tak i kruhová modulace, generovaly očekávané výsledky. Implementace obou technik je tedy korektní a je možné je použít při syntéze zvuku.

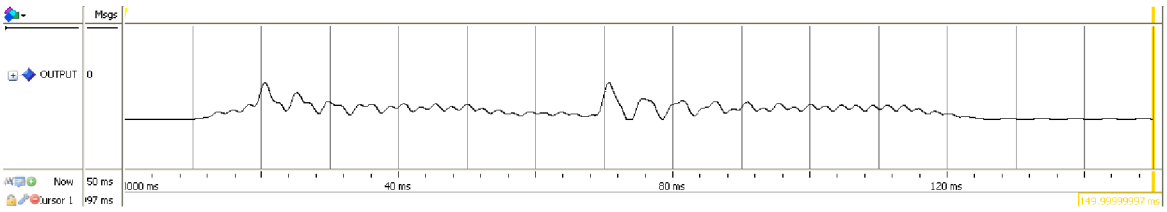
5.5 Filtr

Korektní implementace filtru byla testována porovnáním výstupu simulace filtru s výstupy stejného filtru implementovaného pomocí jazyka C. Program v jazyce C byl vytvořen tak, aby četl vstupní data ze souboru. V tomto souboru byla uložena data získaná ze simulace jednoho hlasu. Výstupní signál hlasu byl vzorkován s frekvencí 48 kHz. Tato frekvence je nižší než jakou frekvencí jsou vzorky generovány hlasem. Nižší frekvence byla zvolena proto, aby počet vzorků nebyl příliš vysoký. Snížení vzorkovací frekvence nemá žádný vliv na výsledný signál, protože frekvence 48 kHz je dostatečná pro kvalitní popis signálu. Generovaný signál měl frekvenci 440 Hz a tvar trojúhelníku.

Oba filtry byly nastaveny na stejné hodnoty. Frekvence filtrů byla nastavena na 204 Hz a rezonance na 0,125. Nejprve byla vygenerována data pomocí filtru implementovaného v jazyce C. Filtrovaný signál je zobrazen na obrázku 5.9. Následně byla spuštěna simulace filtru implementovaného v jazyce VHDL. Výsledný signál je zobrazen na obrázku 5.10.



Obrázek 5.9: Výstup filtru v jazyce C.



Obrázek 5.10: Simulace filtru.

Z obrázků 5.9 a 5.10 je patrná shoda obou signálů, nejedná se však o přesnou shodu. Nepřesnosti jsou způsobeny různou přesností výpočtu. Filtr v jazyce C počítá výstup v plovoucí řádové čárce, zatímco filtr ve VHDL výsledný signál počítá v pevné řádové čárce. Tyto nepřesnosti by bylo možno odstranit zvýšením počtu bitů reprezentujících číslo nebo použitím čísel v plovoucí řádové čárce. Zvýšením přesnosti by se také zvýšila výpočetní náročnost a využitá plocha čipu.

Kapitola 6

Závěr

Cílem práce bylo implementovat syntezátor. Návrh syntezátoru vychází z teoretických poznatků o metodách syntézy zvuku. Syntezátor byl při návrhu rozdělen na několik jednoduchých jednotek, které byly následně navrženy odděleně. Návrh syntezátoru zahrnoval i návrh způsobu komunikace syntezátoru s mikrokontrolerem a audio kodekem. Pro implementaci tohoto rozhraní byly použity součásti dostupné pro platformu FITkit. Jednotkami navrženými v této práci byly oscilátor, generátor obálky, hlas a filtr.

Oscilátor byl navržen jako 24 bitový čítač, jehož frekvence je řízena pomocí 16 bitů. Oscilátor umožňuje generovat průběh ve tvaru pily, trojúhelníku, pulzu s nastavitelnou střídou. Generovaný signál může mít šumový charakter, což je generováno pomocí generátoru pseudonáhodných čísel. Oscilátor je možné synchronizovat s jiným oscilátorem nebo je možné provést kruhovou modulaci. Generátor obálky generuje průběh ADSR obálky. Obálka je generována pomocí čítače, který může čítat nahoru i dolů. Generování obálky je řízeno konečným automatem, který nastavuje směr a rychlost čítání. Generátor obálky umožňuje generování širokého spektra různých obálek.

Z obou výše zmíněných jednotek byl složen jeden hlas, který vystupuje jako základní jednotka generující zvuk. Obě jednotky jsou spojeny tak, že obálka vytvářená generátorem obálky ovlivňuje amplitudu výstupního signálu oscilátoru. Ve zde implementovaném syntezátoru jsou 3 tyto hlasy.

Poslední navrženou součástí je filtr. Filtr slouží k úpravě spektra signálu. Spektrum signálu je upraveno ponecháním či odstraněním některých složek. Filtr byl navržen jako state variable filter. Hlavní výhodou tohoto filtru je, že při jednom výpočtu jsou dostupná data pro dolní, horní, pásmovou propust i pro pásmovou zádrž.

Následně byla ověřena korektní činnost všech jednotek. Ověření činnosti probíhalo v simulaci. K provedení simulace byl zvolen program ModelSim 6.6d. Všechny jednotky byly ověřovány samostatně a až poté z nich byly vytvořeny složitější jednotky. U složitějších jednotek byla také ověřena korektnost jejich činnosti.

První testovanou jednotkou byl oscilátor. U oscilátoru byla testována generovaná frekvence, tvar výstupního signálu. V případě generování pulzu bylo ještě ověřeno korektní generování požadované střídy signálu. Dále byl ověřen generátor obálky. Nejdůležitější ověřovanou vlastností byl tvar výsledné obálky. Byla ověřena strmost jednotlivých fází a plynulost přechodu mezi fázemi. V případě hlasu, který se skládá z již ověřených jednotek, bylo ověřeno korektní tvarování výstupního signálu podle ADSR obálky. Poslední kontrolovanou částí byl filtr. Výstup filtru byl porovnán s výstupem stejně nastaveného filtru implementovaného v jazyce C. Všechny zde navržené jednotky pracují korektně a tedy i výsledný syntezátor pracuje správně.

Syntezátor implementovaný v této práci by bylo možné vylepšit použitím vyhledávací tabulky pro tón. Tato tabulka by obsahovala vzorky tónu vybraného nástroje. V tomto případě by byl výstup oscilátoru použit jako adresa do vyhledávací tabulky. Výstupním signálem by pak byla hodnota na příslušné adrese vyhledávací tabulky. Tato vyhledávací tabulka má ale nevýhodu v tom, že by zabírala značnou plochu čipu FPGA.

Literatura

[1] Digital Synthesis of Musical Sounds. online, 1992 [cit. 2013-03-14].
URL
<http://alumni.media.mit.edu/~gan/Gan/Education/NUS/Physics/MScThesis/>

[2] Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators. online, 1996-07-07 [cit. 2013-04-25].
URL <http:
//www.xilinx.com/support/documentation/application_notes/xapp052.pdf>

[3] Metody zvukové syntézy: Jak vytvořit zvuk. online, 2009-06-26 [cit. 2013-03-14].
URL
<http://elektronicka-hudba.telotone.cz/clanky/metody-zvukove-syntezy/>

[4] Historie elektronických nástrojů. online, 2009-06-28 [cit. 2013-03-14].
URL <http://elektronicka-hudba.telotone.cz/clanky/
historie-elektronickych-nastroju>

[5] Commodore SID 6581 Datasheet. online, 2010-03-26 [cit. 2013-04-25].
URL <http:
//www.waitingforfriday.com/index.php/Commodore_SID_6581_Datasheet>

[6] Sound Synthesis Theory/Modulation Synthesis. online, 2011-04-07 [cit. 2013-04-25].
URL <http:
//en.wikibooks.org/wiki/Sound_Synthesis_Theory/Modulation_Synthesis>

[7] MOS Technology SID. online, 2013-04-18 [cit. 2013-04-25].
URL <http://en.wikipedia.org/wiki/MOS_Technology_SID>

[8] Bilbao, S.: *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*. Hoboken: Wiley, 2009, 457 s., ISBN 978-0-470-51046-9.

[9] Denius: Zvuková syntéza a tvorba, samply a samplery. online, 2007-06-11 [cit. 2013-03-14].
URL <http://www.techno.cz/clanek/21526/
pocitacova-hudba-3-zvukova-synteza-a-tvorba-samply-asamplery>

[10] Geist, B.: *Akustika: Jevy a souvislosti v hudební teorii a praxi*. Praha: Muzikus, 2005, 281 s., ISBN 80-86253-31-7.

[11] Redmon, N.: The digital state variable filter. online, 2003-03-02 [cit. 2013-04-25].
URL <http:
//www.earlevel.com/main/2003/03/02/the-digital-state-variable-filter/>

[12] Russ, M.: *Sound Synthesis and Sampling*. Burlington: eElsevier/Focal Press, 2009, 559 s., ISBN 978-0-240-52105-3.

[13] Skov-Nielsen, J. D.: Interview with Bob Yannes. online, 2010-03-26 [cit. 2013-04-25]. URL <http://sid.kubarth.com/>

[14] Slaný, K.: Ovladač audio kodeku na FITkitu 2.0. online, 2009-03-31 [cit. 2013-04-25]. URL <http://merlin.fit.vutbr.cz/FITkit/docs/firmware/mcu_codec.html>

[15] Slaný, K.: Audio kodek na FITkitu 2.0. online, 2009-10-20 [cit. 2013-04-25]. URL <http://merlin.fit.vutbr.cz/FITkit/docs/firmware/fpga_codec.html>

[16] Teocharisová, V.: Sound Design II: Způsoby syntézy zvuku, aditivní syntéza. online, 2008-02-05 [cit. 2013-04-25]. URL <http://www.muzikus.cz/pro-muzikanty-serialy/Sound-Design-II-Zpusoby-syntezy-zvuku-aditivni-synteza~05~unor~2008/>

[17] Teocharisová, V.: *Sound Design: Zvuková syntéza a tvůrčí programování zvuků v praxi*. Praha: Muzikus, 2009, 109 s., ISBN 80-86253-53-4.

[18] Vašíček, Z.: Komunikační systém. online, 2009-09-19 [cit. 2013-04-25]. URL <http://merlin.fit.vutbr.cz/FITkit/docs/firmware/fpga_interconect.html>

[19] Winfield, A.: Analogue Synthesis. online, 2011-04-07 [cit. 2013-04-25]. URL <http://www.ias.uwe.ac.uk/~a-winfie/teach2003/st_pems4.htm>

Dodatek A

Funkce v jazyce C

void audioInit()

Funkce *audioInit* nepřijmá žádné parametry. Účelem funkce je nastavit audio kodek do výchozího stavu.

void set(unsigned char sada, unsigned char reg, unsigned char hodnota)

Tato funkce je určena pro nastavení jakéhokoliv registru syntezátoru. Parametrem *sada* se určuje jestli je nastavován hlas 1, 2, 3 nebo filter. *reg* určuje registr a *hodnota* je 8 bitová hodnota zapisována do registru.

void setNote(unsigned char hlas, unsigned char oktáva, unsigned char nota)

Funkce *setNote* nastavuje příslušnému hlasu notu z temperovaného ladění v příslušné oktávě.

void setFilt(int frekvence, unsigned char rezonance)

Tato funkce nastavuje filtr. Parametr *frekvence* nastavuje ořezovou frekvenci filtru, *rezonance* nastavuje rezonanci filtru.

unsigned char get(unsigned char reg)

Funkce čte data z registrů syntezátoru. Data čte především z registrů Osc3 a Env 3.

Dodatek B

Registrová sada

Reg.	data								Název
00	F7	F6	F5	F4	F3	F2	F1	F0	Freq Lo 1
01	F15	F14	F13	F12	F11	F10	F9	F8	Freq Hi 1
02	PW7	PW6	PW5	PW4	PW3	PW2	PW1	PW0	PW Lo 1
03	-	-	-	-	PW11	PW10	PW9	PW8	PE Hi 1
04	HLUK	PULZ	PILA	TRI	TEST	RING	SYNC	GATE	Kont. reg. 1
05	ATK3	ATK2	ATK1	ATK0	DCY3	DCY2	DCY1	DCY0	Att/Dec 1
06	STN3	STN2	STN1	STN0	RLS3	RLS2	RLS1	RLS0	Sus/Rel 1

Reg.	data								Název
07	F7	F6	F5	F4	F3	F2	F1	F0	Freq Lo 2
08	F15	F14	F13	F12	F11	F10	F9	F8	Freq Hi 2
09	PW7	PW6	PW5	PW4	PW3	PW2	PW1	PW0	PW Lo 2
0A	-	-	-	-	PW11	PW10	PW9	PW8	PE Hi 2
0B	HLUK	PULZ	PILA	TRI	TEST	RING	SYNC	GATE	Kont. reg. 2
0C	ATK3	ATK2	ATK1	ATK0	DCY3	DCY2	DCY1	DCY0	Att/Dec 2
0D	STN3	STN2	STN1	STN0	RLS3	RLS2	RLS1	RLS0	Sus/Rel 2

Reg.	data								Název
0E	F7	F6	F5	F4	F3	F2	F1	F0	Freq Lo 3
0F	F15	F14	F13	F12	F11	F10	F9	F8	Freq Hi 3
10	PW7	PW6	PW5	PW4	PW3	PW2	PW1	PW0	PW Lo 3
11	-	-	-	-	PW11	PW10	PW9	PW8	PW Hi 3
12	HLUK	PULZ	PILA	TRI	TEST	RING	SYNC	GATE	Kont. reg. 3
13	ATK3	ATK2	ATK1	ATK0	DCY3	DCY2	DCY1	DCY0	Att/Dec 3
14	STN3	STN2	STN1	STN0	RLS3	RLS2	RLS1	RLS0	Sus/Rel 3

Reg.	data								Název
15	FC7	FC6	FC5	FC4	FC3	FC2	FC1	FC0	FC Lo
16	FC15	FC14	FC13	FC12	FC11	FC10	FC9	FC8	FC Hi
17	-	-	-	-	FILT EX	FILT 3	FILT 2	FILT 1	Filt
18	3 OFF	HP	BP	LP	-	-	-	-	Mode

Reg.	data								Název
19	RES7	RES6	RES5	RES4	RES3	RES2	RES1	RES0	Res Lo
1A	RES15	RES14	RES13	RES12	RES11	RES10	RES9	RES8	Res Hi
1B	O7	O6	O5	O4	O3	O2	O1	O0	Osc3
1C	E7	E6	E5	E4	E3	E2	E1	E0	Env3

Dodatek C

Tabulka hodnot pro generování osmi oktáv

Nota	Freq (Hz)	Osc Fn (dec)	Osc Fn (hex)
C0	16,35	276	114
C0$	17,32	292	124
D0	18,35	309	135
D0$	19,44	328	148
E0	20,6	347	15B
F0	21,83	368	170
F0$	23,12	390	186
G0	24,5	413	19D
G0$	25,96	438	1B6
A0	27,5	464	1D0
A0$	29,14	491	1EB
B0	30,87	520	208
C1	32,7	551	227
C1$	34,65	584	248
D1	36,71	619	26B
D1$	38,89	656	290
E1	41,2	694	2B6
F1	43,65	736	2E0
F1$	46,25	780	30C
G1	49	826	33A
G1$	51,91	875	36B
A1	55	927	39F
A1$	58,27	982	3D
B1	61,74	1041	411
C2	65,41	1103	44F
C2$	69,3	1168	490
D2	73,42	1238	4D6
D2$	77,78	1311	51F
E2	82,41	1389	56D
F2	87,31	1472	5C0

Nota	Freq (Hz)	Osc Fn (dec)	Osc Fn (hex)
F2$	92,5	1559	617
G2	98	1652	674
G2$	103,83	1750	6D6
A2	110	1854	73E
A2$	116,54	1964	7AC
B2	123,47	2081	821
C3	130,81	2205	89D
C3$	138,59	2336	920
D3	146,83	2475	9AB
D3$	155,56	2622	A3E
E3	164,81	2778	ADA
F3	174,61	2943	B7F
F3$	185	3118	C2E
G3	196	3304	CE8
G3$	207,65	3500	DAC
A3	220	3708	E7C
A3$	233,08	3929	F59
B3	246,94	4162	1042
C4	261,63	4410	113A
C4$	277,18	4672	1240
D4	293,66	4950	1356
D4$	311,13	5244	147C
E4	329,63	5556	15B4
F4	349,23	5887	16FF
F4$	370	6237	185D
G4	392	6608	19D0
G4$	415,3	7000	1B58
A4	440	7417	1CF9
A4$	466,16	7858	1EB2
B4	493,88	8325	2085
C5	523,25	8820	2274
C5$	554,37	9344	2480
D5	587,33	9900	26AC
D5$	622,25	10489	28F9
E5	659,25	11112	2B68
F5	698,46	11773	2DFD
F5$	740	12473	30B9
G5	783,99	13215	339F
G5$	830,61	14001	36B1
A5	880	14833	39F1
A5$	932,33	15715	3D63
B5	987,77	16650	410A
C6	1046,5	17640	44E8
C6$	1108,73	18689	4901
D6	1174,66	19800	4D58

Nota	**Freq (Hz)**	**Osc Fn (dec)**	**Osc Fn (hex)**
D6$	1244,51	20977	51F1
E6	1318,51	22225	56D1
F6	1396,91	23546	5BFA
F6$	1479,98	24947	6173
G6	1567,98	26430	673E
G6$	1661,22	28001	6D61
A6	1760	29667	73E3
A6$	1864,65	31430	7AC6
B6	1975,53	33299	8213
C7	2093	35280	89D0
C7$	2217,46	37377	9201
D7	2349,32	39600	9AB0
D7$	2489,01	41955	A3E3
E7	2637,02	44450	ADA2
F7	2793,83	47093	B7F5
F7$	2959,95	49893	C2E5
G7	3135,96	52860	CE7C
G7$	3322,44	56003	DAC3
A7	3520	59333	E7C5
A7$	3729,31	62861	F58D
B7	3951,06	66599	10427