ŽILINSKÁ UNIVERZITA V ŽILINE

FAKULTA RIADENIA A INFORMATIKY

DIPLOMOVÁ PRÁCA

Štúdijný odbor: Informačné a riadiace systémy

Marián Boďa

Subtraktívny syntetizátor VST

Vedúci: RNDr. Miroslav Benedikovič

Reg. č.: 93/2007 Máj 2008

Žilina

Anotačný list

Resumé

Táto diplomová práca sa zaoberá problematikou syntézy v hudbe, vysvetľuje princípy rôznych typov syntézy a opisuje vývoj subtraktívneho syntetizátora VST od jeho návrhu až po záverečné testovanie.

Summary

This thesis deals with sound synthesis in music, describes principles of different kinds of synthesis and shows development of subtractive VST synthesizer from design to final testing.

Poďakovanie

Ďakujem pánovi RNDr. Miroslavovi Benedikovičovi, vedúcemu tejto diplomovej práce, za ochotu, cenné rady, trpezlivosť a čas, ktorý mi venoval pri jej vypracovaní. Tiež ďakujem svojmu bratovi, Ing. Petrovi Boďovi, za konzultácie ohľadom syntézy zvuku a precízne testovanie syntetizátora. Ďakujem aj členom komunity Corenforce za ochotnú pomoc a odborné rady. Osobitne ďakujem mojej snúbenici, Mgr. Kataríne Buzalkovej, za podporu a pomoc s jazykovou úpravou tejto práce.

Obsah

Úvod

Zvuk syntetizátorov je bežnou súčasťou mnohých hudobných štýlov už viac ako tridsať rokov. Počas tejto doby sa ich zvuk, ale aj forma ich existencie výrazne menili. V dnešnej dobe, keď sú počítače už takmer nevyhnutnou súčasťou každej domácnosti, je použitie syntetizátorov prístupnejšie oveľa širšiemu okruhu hudobných nadšencov. Rovnako dostupná je aj možnosť vytvorenia vlastného virtuálneho nástroja, a to už nielen pre pokročilých programátorov, ale aj pre bežných používateľov, ktorí majú základné vedomosti o zvukovej syntéze. Pre nich existuje niekoľko vizuálnych prostredí, ktoré im poskytujú dostatok nástrojov na realizáciu svojich predstáv. Pre tých náročnejších však stále zostáva len cesta kompletného programovania.

V tejto diplomovej práci sa snažím zachytiť základné princípy fungovania syntetizátorov a ich stručnú históriu. Usilujem sa vysvetliť základné princípy fungovania rôznych typov syntézy, zameriavam sa na softvérovú syntézu a predstavujem najpoužívanejšie rozhrania, pre ktoré sa syntetizátory vyvíjajú. Ďalej opisujem rozhranie VST a spôsoby tvorby syntetizátorov pre toto rozhranie. V ďalších kapitolách bližšie rozoberám aktuálny stav voľne šíriteľných subtraktívnych syntetizátorov pre rozhranie VST a nakoniec ukazujem na príklade návrh a implementáciu subtraktívneho VST syntetizátora v jazyku C++. Pri návrhu tohto nástroja kladiem dôraz hlavne na kvalitu implementovaných častí a na príjemnosť a praktickosť používateľského rozhrania. Podrobnejšie sa venujem problematike generovania antialiasovaných signálov, návrhu digitálneho filtra a optimalizácii rýchlosti kódu. Na konci práce vyhodnocujem výsledok implementácie spolu s odbornými názormi hudobníkov.

1. Zvuková syntéza v hudbe

1.1 História syntézy zvuku

Hudba je ľudstvu známa už tisícročia. Najstaršie hudobné nástroje, o ktorých dnes vieme, boli nájdené pri vykopávkach sumerského mesta Ur.[1] Podľa analýz pochádzajú z obdobia okolo roku 2500 pred naším letopočtom. Medzi týmito nástrojmi boli lýry, harfy, flauty a cymbaly. Tieto nálezy poukazujú na to, že už v tomto ranom období ľudia poznali mnohé druhy hudobných nástrojov, a to prinajmenšom strunové, dychové a bicie. To znamená, že už v tej dobe dokázali ľudia vytvárať zvuk mnohými spôsobmi, a tiež, že už vtedy poznali polyfóniu[2].

Počas nasledujúcich tisícročí sa vyvinulo množstvo rôznych hudobných nástrojov. Väčšina dnes používaných akustických nástrojov bola vynájdená pred 18. storočím. Dnes sú tieto nástroje vyrábané dokonalejšími technológiami a z kvalitnejších materiálov, ale princípy týchto nástrojov zostali nezmenené. Pred elektronickou érou bolo zrejme ľudstvo presvedčené, že už nemožno vynájsť nejaký úplne nový druh hudobného nástroja s úplne iným zvukovým charakterom, ako mali dovtedy známe nástroje. Prvým nástrojom, ktorý sa zapísal do histórie ako „elektronický“, bol tzv. *clavecin electrique*, ktorý v roku 1759 vynašiel jezuitský mních *Jean-Baptiste de La Borde*. Bol to klávesový nástroj podobný klavichordu, ktorý pri stlačení klávesu statickou elektrinou rozrezonoval zvonce. Aj keď je tento nástroj považovaný za elektronický, využíval elektrickú energiu len na ovládanie akustických zvoncov, a preto ho nemožno považovať za syntetizátor.

[1] M. de Schauensee: Two Lyres from Ur [1].

[2] Polyfónia je schopnosť hudobného nástroja vydať súčasne viac rôznych tónov.

1.1.1 Analógové syntetizátory

Prvý skutočný syntetizátor vynašiel americký vynálezca *Elisha Gray* v roku 1876. Gray náhodou zistil, že dokáže ovládať zvuk generovaný samooscilláciou elektromagnetického obvodu. Zostrojil teda prístroj s malou klaviatúrou, a syntetizátor bol na svete. Tento vynález je dnes známy ako „Hudobný telegraf“. V nasledujúcich desaťročiach prišlo mnoho ďalších objavov a nových syntetizátorov, ale obstaranie a prevádzka týchto nástrojov boli veľmi drahé. V roku 1964 *Robert Moog* predstavil prototyp syntetizátora, ktorý bol oveľa modernejší a menší ako dovtedy dostupné syntetizátory. Avšak jeho použitie stále vyžadovalo odborné vedomosti a jeho prestavenie na iný zvuk trvalo aj hodiny.

Až v sedemdesiatych rokoch začali byť syntetizátory cenovo dostupné, a zároveň aj jednoduchšie ovládateľné. V týchto rokoch sa syntetizátory rýchlo rozšírili medzi tvorcami populárnej hudby. Boli to hlavne výrobky firiem *Moog Music*, *ARP Instruments*. Postupne na trh syntetizátorov prišli medzi inými aj značky *Korg*, *Yamaha*, *Roland* a *Oberheim*. Tieto syntetizátory boli monofonické, len niektoré výnimky boli schopné zahrať dve rôzne noty naraz. Až od roku 1976 s nástupom modelov *Yamaha CS-80* a *Oberheim Four-Voice* sa začala rozmáhať polyfónia. Prvým syntetizátorom, ktorý umožňoval uloženie aktuálneho nastavenia parametrov do pamäte a obnovenie týchto nastavení jednoduchým stlačením tlačidla, bol *Prophet-5* od firmy *Sequential Circuits*. Prophet-5, ako v tejto dobe už viacero syntetizátorov, využíval mikroprocesor napríklad na ovládanie frekvencií oscilátorov[3] a spúšťanie generátorov obálok, avšak samotné generovanie zvuku zostalo analógové. Firmy *Korg* a *Kawai* v tej dobe už používali pre niektoré svoje analógové syntetizátory digitálne oscilátory. Pre tieto nástroje využívajúce digitálne prvky bol vytvorený štandard *MIDI*[4].

Koncom osemdesiatych rokov analógové syntetizátory boli nahradené digitálnymi, začiatkom deväťdesiatych rokov sa však analógové nástroje znovu stali predmetom veľkého záujmu, pretože tieto boli staršie, a teda cenovo dostupnejšie ako nové

[3]Vo vtedajších syntetizátoroch bolo veľmi problematické udržať oscilátory naladené na požadovanej frekvencii.

[4]MIDI - Musical Instrument Digital Interface - digitálne rozhranie pre hudobné nástroje.

digitálne. Zároveň digitálne emulácie niektorých starých kultových analógových syntetizátorov boli natoľko nedokonalé, že sa hudobníci často vracali k analógovým originálom. Asi najkultovejším syntetizátorom v dejinách sa stal *Roland TB-303*, ktorý bol v roku 1982 uvedený na trh s cenou 395 amerických dolárov[5]. Dnes, po 26 rokoch existencie, sa tento syntetizátor predáva použitý za ceny okolo 2 600 dolárov[6]. Pritom jeho možnosti sú veľmi obmedzené na určitý druh efektov a basových zvukov, ale práve jeho špecifický zvuk ho urobil slávnym.

Analógové syntetizátory sú vyrábané dodnes, aj keď už tvoria len malú časť celkovo vyrobených syntetizátorov. Príkladmi sú *Alesis Andromeda* a *Prophet 08*.

1.1.2 Digitálne syntetizátory

Prvým digitálnym syntetizátorom bol *Fairlight CMI* vyrábaný od roku 1978, a od začiatku osemdesiatych rokov bola väčšina vyrobených syntetizátorov digitálna. Digitálna éra syntetizátorov priniesla množstvo nových možností generovania zvuku a vznikli rôzne typy syntetizátorov podľa spôsobu, akým je zvuk vytváraný. Digitálne syntetizátory využívajú na syntézu zvuku mikroprocesory pre spracovanie signálu DSP. Dnešné digitálne syntetizátory sú veľmi výkonné, ich polyfónia dosahuje podľa zložitosti a zamerania nástroja 16 – 128 hlasov pri multitimbralite 4 – 32 nástrojov[7] a takmer všetky disponujú aj výkonnou efektovou jednotkou. Niektorí výrobcovia dokonca ponúkajú niekoľko verzií syntetizátora s rôznym počtom DSP. Výkonnejšie verzie potom ponúkajú väčšiu polyfóniu, a tomu je prispôsobená aj ich cena.

1.1.3 Softvérové syntetizátory

Rýchly vývoj osobných počítačov a stále klesajúce ceny na prelome osemdesiatych a deväťdesiatych rokov vytvorili ideálne podmienky pre využitie týchto strojov v hudbe. Už 8-bitové platformy sa využívali tak na ovládanie hardvérových syntetizátorov cez rozhranie MIDI, ako aj na samotné generovanie zvuku. Rapídnym zvyšovaním výkonu sa zvyšovala aj zložitosť softvérových syntetizátorov a ich nároky na výkon. Počí-

[5] Podľa Wikipedie [3].
[6] Podľa aktuálnych aukcií na americkom portáli eBay dňa 10. 5. 2008.
[7] Multitimbralita je schopnosť polyfonického syntetizátora hrať súčasne na viacerých nástrojoch.

tače boli už cenovo dostupné takmer všetkým, vytváranie nových syntetizátorov bolo neporovnateľne jednoduchšie oproti hardvérovým syntetizátorom, a navyše omnoho lacnejšie. Možnosť vytvoriť vlastný syntetizátor už nebola výsadou niekoľkých ľudí na špičkových univerzitách, ale bola prístupná širokej verejnosti. Toto viedlo k očakávaniam rýchleho rastu konkurencie pre výrobcov syntetizátorov. Paradoxne kvalita softvérových nástrojov nikdy nedosiahla kvalitu hardvérových implementácií, a namiesto toho bolo ľudstvo zasypané množstvom nekvalitných nástrojov. Veľkou mierou sa na tom podieľal aj operačný systém *Windows*, ktorý je momentálne najrozšírenejší zo všetkých operačných systémov. Ten síce dokáže krátkodobo poskytovať dostatočnú pôdu pre rôzne implementácie hudobných nástrojov, ale jeho nespoľahlivosť, nestabilita a nepredvídateľné správanie sa pri aplikáciách pracujúcich v reálnom čase ho robia profesionálne nepoužiteľným.

V súčasnosti je obľúbenou platformou pre syntézu zvuku platforma *Macintosh* a stále viac produktov sa objavuje aj pre operačný systém *Linux*. Významnými výrobcami softvérových syntetizátorov sú napríklad firmy *Native Instruments*, *VirSyn* a *Ueberschall*.

1.1.4 Akcelerované softvérové syntetizátory

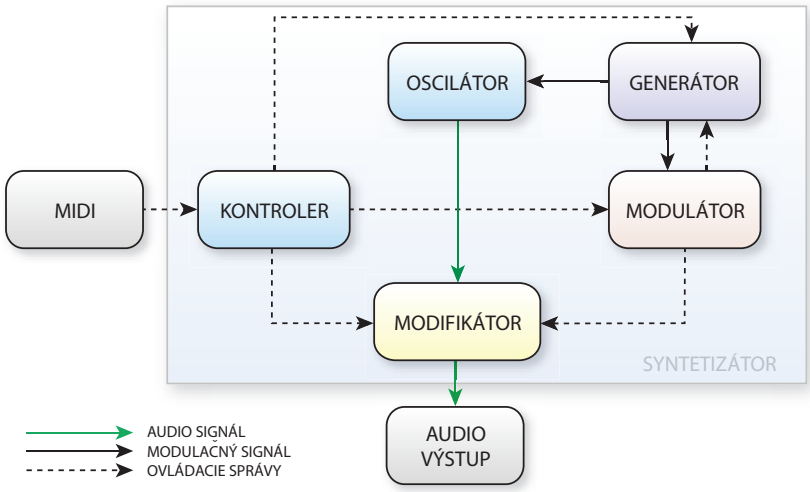
Všestranne použiteľné procesory (General Purpose Processor - GPP) majú mnoho nevýhod oproti dedikovaným procesorom na spracovanie signálu (Digital Signal Processor - DSP) v oblasti spracovania zvuku. Preto už v deväťdesiatych rokoch sa na trhu objavili hardvérovo-softvérové syntetizátory, ktoré fungovali len použité s počítačom. Najčastejšie mali podobu zvukovej karty pre rozhrania ISA[8] a PCI[9] a ovládací softvér. Moderné akcelerované systémy sú implementované hlavne pre rozhranie FireWire a USB, ale aj pre staršie rozhranie PCI. Typickými príkladmi sú produkty *E-MU Emulator X2* (PCI) a *tc electronic PowerCore* (FireWire). Tento typ syntetizátorov sa pre používateľa javí rovnako ako čisto softvérový, ale jeho výpočty prebiehajú na vlastnom hardvéri, a teda nezaťažujú procesor počítača.

[8] ISA - Industry Standard Architecture - zbernicový štandard z roku 1984.
[9] PCI - Peripheral Component Interconnect - zbernicový štandard používaný od 90. rokov.

1.2 Princípy Syntetizátorov

Syntetizátory môžu mať rôznu fyzickú podobu. Najznámejšia z nich je určite klávesový syntetizátor, disponujúci niekoľkými oktávami klávesov a ovládacími tlačidlami a potenciometrami. Okrem tohto typu existujú tzv. „desktopové“a „rackové“modely. Desktopové sú syntetizátory bez klaviatúry, ale s dostatočným množstvom ovládacích prvkov. Rackové verzie sú určené na montáž do rackov, ovládací panel je často menší ako pri desktopových verziách. Populárne syntetizátory sa často vyrábajú vo viacerých verziách.

1.2.1 Architektúra súčasných syntetizátorov



Obr. 1.1: Všeobecná schéma bežného syntetizátora

Funkcia syntetizátora spočíva v transformácii interaktívnych vstupov hudobníka a aktuálnych nastavení parametrov syntetizátora na jeden alebo viacero zvukových výstupov. Bežný syntetizátor sa skladá z niekoľkých komponentov, z ktorých každý má svoju vlastnú funkciu. Základná architektúra syntetizátora je znázornená na obr. 1.1. Komponent *kontroler* prijíma vstupy a na ich základe posiela riadiace inštrukcie do ďalších komponentov, ktoré sa spolu podieľajú na tvorbe zvukového signálu. *Generátor* je komponent, ktorý generuje určitý druh signálu. Môže to byť periodický signál použitý na generovanie samotného základu zvuku syntetizátora prostredníctvom komponentu *oscilátor*, nízkofrekvenčný periodický signál alebo rôzne nepe-

riodické signály použité ako vstupy do modulátora. *Modulator* je komponent, ktorý modifikuje parametre ostatných komponentov na základe vstupných hodnôt. Jednoduchým príkladom použitia modulátora je ovládanie hlasitosti alebo frekvencie oscilátora podľa signálu generovaného v ľubovoľnom generátore. *Modifikátor* je komponent, ktorý spracúva zvukový signál vygenerovaný oscilátorom, a určitým spôsobom ho modifikuje. Príkladom môže byť ľubovoľný reťazec efektov a filtrov.

1.2.2 Komponenty syntetizátorov

Najčastejšie implementované komponenty v súčasných syntetizátoroch sú:

Oscilátor – základný komponent, ktorý generuje samotný základ zvuku. Najjednoduchší oscilátor vytvára sínusový signál, ale bežné implementácie ponúkajú rôzne periodické, ale aj neperiodické priebehy. Jeho frekvencia je závislá od stlačenej noty.

Filter – komponent, ktorý odstraňuje určité frekvencie z prichádzajúceho signálu. Najčastejšie sa používajú rezonančné filtre typu low-pass (dolnopriepustný), high-pass (hornopriepustný), band-pass (prepúšťajúci určité pásmo) a band-stop[10] (filtrujúci určité pásmo). Aktívna frekvencia (Hraničná pre low-pass a high-pass, priepustná pre band-pass a filtrujúca pre band-stop) týchto filtrov býva ovládateľná, ako aj miera rezonancie, resp. šírka pásma.

Generátor obálky – generuje neperiodický signál, ktorý má za úlohu najmä ohraničiť amplitúdu signálu v priebehu hrania noty. Základný princíp obálky je určiť rýchlosť rastu amplitúdy signálu pri stlačení klávesu, priebeh počas hrajúcej noty a rýchlosť klesania po pustení noty. Funkcia obálky nie je obmedzená na moduláciu amplitúdy signálu, ale môže byť implementovaná pre moduláciu ktoréhokoľvek parametera elementov syntézy.

Nízkofrekvenčný oscilátor (LFO) – generuje nízkofrekvenčný [11] signál, ktorý býva použitý na moduláciu niektorého parametra syntetizátora. Tento generátor

[10] Pri niektorých syntetizátoroch uvádzaný ako BAND-REJECT alebo NOTCH filter.

[11] Nízkymi frekvenciami v kontexte syntézy zvuku v hudbe rozumieme frekvencie prevažne nižšie ako počuteľné spektrum. To neznamená, že nízkofrekvenčné oscilátory nemôžu generovať počuteľné frekvencie, ale že ich primárnym cieľom nie je generovať zvuk.

môže mať, podobne ako zvukový oscilátor, rôzne priebehy.

Modulačná matica (Modulation Matrix) – tabuľka zapojení modulačných zdrojov ku modulovaným parametrom.

Efektová jednotka – komponent, ktorý na vygenerovaný signál aplikuje rôzne efekty obohacujúce zvuk. Tento komponent býva zapojený až po spojení signálov jednotlivých hlasov.

1.2.3 Vstupy syntetizátorov

Existujú rôzne spôsoby, akými môže hudobník so syntetizátorom interagovať. Najčastejšia forma je ovládanie prostredníctvom klaviatúry podobnej klavíru, ale nie je výnimočné ani používanie strunových senzorov pre gitary, dychové ovládače, alebo dokonca senzory pohybu. Pre jednotné rozhranie ovládania syntetizátorov, ale aj iných hudobných zariadení, a ich vzájomnú komunikáciu a interakciu bol vyvinutý štandard MIDI. Okrem MIDI inštrukcií môže syntetizátor prijímať aj audio signál a použiť ho ako modulačný vstup.

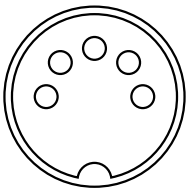
1.2.4 Výstupy syntetizátorov

Výstupom syntetizátorov je jeden alebo viacero zvukových signálov. V praxi sa bežne používa minimálne jeden stereovýstup, väčšina moderných syntetizátorov však disponuje dvoma a viacerými stereovýstupmi. Vďaka viacerým výstupom je možné zo syntetizátora súčasne nahrávať napríklad surový a efektovaný signál alebo v prípade multitimbrálnych syntetizátorov nahrávať rôzne nástroje ako samostatné stopy.

Kvôli komunikácii s inými hudobnými zariadeniami disponujú syntetizátory aj MIDI výstupom.

1.3 MIDI štandard

MIDI je štandardizované rozhranie pre komunikáciu, ovládanie a synchronizáciu elektronických hudobných nástrojov. MIDI nepracuje s audiosignálom, ale prenáša správy a inštrukcie (napríklad o tom, ktorú notu a s akou intenzitou má nástroj zahrať), ovládacie (napr. zmena parametrov) a synchronizačné signály.



Obr. 1.2: Štandardný konektor pre MIDI

Tento štandard bol od uvedenia v roku 1983 veľmi úspešný a všetci významní výrobcovia elektronických hudobných nástrojov ho veľmi rýchlo implementovali do svojich výrobkov. Za 25 rokov existencie sa tento štandard udržal v takmer nezmenenej podobe. Počas týchto rokov bolo viacero pokusov o modernizáciu tohto štandardu a tiež o vytvorenie nových, ako napríklad OSC[12], ale žiadny z nich nezískal dostatočnú pozornosť na to, aby nahradil štandard MIDI.

1.3.1 Architektúra MIDI

MIDI zariadenia sa podľa pôvodného návrhu zapájajú sériovo. Takmer každé MIDI zariadenie disponuje konektorom THRU, ktorý vlastne len posiela ďalej signál zo vstupu. MIDI rozhranie ponúka 16 kanálov, ktoré zdieľajú šírku pásma 30,5 kb/s. Teoreticky nie je obmedzené, koľko zariadení je možné takýmto spôsobom zapojiť, avšak každé zariadenie vysiela prijaté dáta ďalej s určitým oneskorením a aj šírka pásma, ktorá pri tomto starom rozhraní je na dnešné pomery veľmi úzka, sa pri dlhšom zreťazení rýchlo zahltí. Preto sa v dnešnej dobe, keď už existuju rôzne MIDI smerovače, odporúča zapájať len jedno zariadenie na každú vetvu. Nástroje, ktoré využívajú vysokú multitimbraritu (zvukové moduly, samplery), disponujú viacerými MIDI vstupmi/výstupmi.

1.3.2 MIDI správy

MIDI protokol je založený na elementárnych správach. Správa pozostáva z jedného alebo viacerých bajtov. Každá správa začína takzvaným stavovým bajtom, ktorý určuje typ správy. Tento bajt sa odlišuje od ostatných tým, že má nastavený prvý bit na

[12]OSC - Open Sound Control.

hodnotu 1. Takto je možné rozoznať začiatok novej správy. Prvá polovica bajtu definuje typ správy a druhá určuje, do ktorého zo šestnástich MIDI kanálov správa patrí.

Najdôležitejšie typy správ a im prislúchajúce hodnoty sú:

Note Off (8) – zastavenie noty,

Note On (9) – spustenie noty,

Polyphonic Aftertouch (A) – zmena prítlaku klávesu[13],

Control Change (B) – zmena hodnoty ovládača,

Program Change (C) – zmena programu[14],

Aftertouch (D) – zmena prítlaku platná pre všetky noty v rámci kanálu,

Pitch Wheel (E) – ovládač určený hlavne pre ohýbanie tónu.

Týmito správami presne určujeme syntetizátoru, čo má robiť. Prvý krok je stlačenie noty, ktoré vyšle správu stlačenia noty spolu s dvomi parametrami - ktorá nota bola stlačená a akou rýchlosťou bola stlačená. Táto rýchlosť sa označuje slovom VELOCITY. Kým je nota stlačená, zmenou prítlaku sa vysielajú hodnoty AFTERTOUCH, prípadne POLY-AFTERTOUCH. Hodnoty CONTROL CHANGE (CC) sú hodnoty ovládacích prvkov - väčšinou potenciometrov. Pri zmene konkrétneho ovládača sa vyšle správa CC s číslom ovládača a jeho hodnotou. Pri pustení klávesu sa vysiela správa NOTE OFF a rýchlosť pustenia klávesu[15].

Noty, rýchlosť stlačenia, ovládače, hodnoty ovládačov sú v rozmedzí 0 – 127. Jednou z mála výnimiek je ovládač PITCH WHEEL, ktorý nadobúda hodnoty –8192 až +8191.

1.4 Typy syntézy

Rokmi vývoja syntetizátorov uzrelo svetlo sveta veľa rôznych typov syntézy. Od veľmi jednoduchého kombinovania niekoľkých harmonických signálov až po veľmi zložité

[13] Polyfonický Aftertouch je v MIDI klávesoch veľmi zriedka implementovaný.

[14] Program je zoznam konkrétnych nastavení všetkých parametrov nástroja. Zmena programu teda znamená zmenu nastavení na iné nastavenia uložené v pamäti.

[15] Rýchlosť pustenia klávesu tiež nie je často implementovaná v MIDI klávesoch.

fyzikálno-akustické simulácie. Hlavne digitálne syntetizátory priniesli obrovské množstvo nových možností.

1.4.1 Aditívna syntéza

Asi najjednoduchším používaným typom syntézy je aditívna syntéza. Už názov tejto metódy napovedá, že ide o sčítavanie určitých elementárnych signálov. Základ zvuku všetkých hudobných nástrojov je zložený zo základnej frekvencie a jej vyšších harmonických frekvencií[16]. Amplitúdy jednotlivých harmonických zložiek sa pri týchto nástrojoch časom menia. Na tomto princípe je založená aditívna syntéza, ktorá generuje určitý počet harmonických signálov, pričom každá z nich má vlastnú amplitúdovú obálku. Tieto obálky umožňujú meniť pomery harmonických zložiek, a tým meniť farbu zvuku.

Táto technika ponúka napríklad veľmi zaujímavé možnosti simulácie strunových nástrojov a organov. Moderné aditívne syntetizátory kombinujú aditívnu syntézu s prvkami iných typov syntéz. Významným hardvérovým aditívnym syntetizátorom je *KAWAI K5000* a zo softvéru sú to najmä *VirSyn Cube* a *Camel Audio Cameleon*.

1.4.2 Subtraktívna syntéza

Subtraktívna syntéza je zrejme najstarší populárny typ syntézy. Väčšina analógových syntetizátorov fungovala na tomto princípe, a preto sa digitálnym a softvérovým subtraktívnym syntetizátorom často hovorí aj „virtuálne analógové“.

Ich princíp spočíva v generovaní signálu bohatého na harmonické zložky a následnom filtrovaní rezonančnými filtrami. Väčšina subtraktívnych syntetizátorov využíva 2 alebo 3 oscilátory s voliteľnými priebehmi, medzi ktorými najčastejšie býva píla (SAWTOOTH), trojuholník (TRIANGLE) a obdĺžnik (SQUARE). Hlavne píla a obdĺžnik sú priebehy s vysokou úrovňou harmonických frekvencií a pri týchto priebehoch sa filtrovaním dosahujú špecifické zvuky. Napríklad syntetizátor *Roland TB-303* sa preslávil hlavne svojím originálne znejúcim filtrom. Ďalším známym subtraktívnym nástrojom

[16]Harmonická frekvencia určitej frekvencie je jej celočíselný násobok, napríklad frekvencia 100 Hz má harmonické frekvencie 200 Hz, 300 Hz, 400 Hz,...

je *Clavia Nord Lead*.

1.4.3 Wavetable syntéza

Wavetable syntéza je v niečom podobná aditívnej a v niečom sample-based syntéze. Využíva digitálne vzorky akustických nástrojov, lenže na rozdiel od sample-based syntézy používa len niekoľko krátkych vzoriek, väčšinou dlhých len jednu periódu základnej frekvencie. Medzi týmito vzorkami pri prehrávaní plynule prechádza z jednej do druhej, a vytvára tak kváziperiodický signál a tým pomerne realistický zvuk. Podobnosť s aditívnou syntézou je v tom, že vzhľadom na použitie vzoriek o dĺžke rovnajúcej sa jednej perióde aj wavetable syntéza využíva len harmonické frekvencie, pretože neharmonické frekvencie nemôže takáto vzorka obsahovať. Podobný je aj spôsob plynulého prechádzania z jednej vzorky do druhej, čo vlastne produkuje podobný efekt, ako zmeny pomerov harmonických frekvencií pri aditívnej syntéze. Oproti aditívnej syntéze má ale táto metóda výhodu v tom, že vyžaduje oveľa menej výpočtov v reálnom čase. Populárnym wavetable syntetizátorom je *Microwave* od firmy *Waldorf*.

1.4.4 FM syntéza

V roku 1983 predstavila spoločnosť *Yamaha* modely *DX7* a *DX9*, ktoré využívali frekvenčnú moduláciu oscilátorov ako spôsob vytvárania zvuku. Model DX7 sa stal veľmi populárnym a jeho emulácie sú dodnes používané. Niektoré FM syntetizátory používajú sínusový priebeh pre modulačný aj modulovaný signál, iné dávajú na výber z viacerých priebehov. Táto metóda umožňuje vytvárať širokú škálu zvukov. Pre vytvorenie harmonických zvukov je dôležité, aby modulačný a modulovaný signál boli vzájomne harmonické.

1.4.5 Granulárna syntéza

Granulárna syntéza je založená na veľmi krátkych vzorkách (bežne 1 – 50 ms) nahratého alebo syntetizovaného zvukového materiálu. Tieto vzorky sa nazývaju granule a pri prehrávaní sú navzájom vrstvené a prehrávajú sa s premenlivým vzájomným časovým posunom, premenlivou frekvenciou, fázou a amplitúdou. Týmto sa dosahujú

rôzne umelé efekty a surrealistické zvuky. Táto syntéza je pomerne málo rozšírená. Využíva ju napríklad softvérový syntetizátor *Buero Stelkens crusherX-Studio*.

1.4.6 Formantová syntéza

Formantová syntéza je známa najmä zo systémov syntetizujúcich reč. Funguje na báze formantov – úzkych pásmovopriepustných filtrov. Určitý počet týchto formantov usporiadaných určitým spôsobom vo frekvenčnej doméne vytvára konkrétne hlásky podobné ľudským. A práve týmto kombinovaním a rozmiestňovaním formantov vo frekvenčnej doméne a ich posúvaním a zmenami v čase alebo v závislosti od iných faktorov je možné syntetizovať zvuky odlišné od všetkých ostatných metód. Vydarenou implementáciou tejto formy syntézy disponuje syntetizátor *Yamaha FS1R*.

1.4.7 Phase distortion syntéza

Syntézu deformácie fázy predstavila v roku 1984 spoločnosť *Casio* s radom syntetizátorov *CZ*. Tento typ syntézy väčšinou používa ako základný priebeh sínusovú vlnu. Bežné syntetizátory sa pri konštantnej výške tónu lineárne posúvaju vo fáze prehrávanej vlny. V tomto je metóda deformácie fázy iná. Aplikuje určitú matematickú funkciu medzi čas a fázový posun na vzorku generovaného signálu. Týmto procesom vznikne iná vlna s bohatým harmonickým obsahom. Parametrizácia tejto fázovej funkcie ponúka veľmi pestré možnosti tvorby atmosferických zvukov vyvíjajúcich sa v čase. Teoreticky metóda nie je obmedzená na sínus ako základný priebeh, ale použitie sínusu poskytuje oveľa viac variácií pri deformácii, ako iné priebehy známe zo subtraktívnych syntetizátorov.

1.4.8 Physical modelling syntéza

Syntéza pomocou fyzikálneho modelu využíva určité fyzikálno-akustické modely, väčšinou hudobných nástrojov, na výpočet zvukového signálu. Takýto model môže napríklad popisovať správanie sa bubna pri údere. V takom prípade sa ako vstupné parametre môžu zobrať rýchlosť a presné miesto úderu. Nasleduje simulácia šírenia sa vĺn po dvojrozmernej membráne, ich odrazy od okrajov a prenos na valcové telo bubna a jeho rezonácia. Táto metóda dokáže verne kopírovať skutočný zvuk pri

jednoduchých nástrojoch, ale jej výpočtová zložitosť rastie veľmi rýchlo so snahou presnejšie modelovať daný systém. Veľmi známy nástroj využívajúci prvky tejto syntézy je *Korg Prophecy*.

1.4.9 Sample-based syntéza

Syntéza, založená na prehrávaní navzorkovaných signálov, je populárna od začiatku existencie digitálnych technológií. Veľmi úspešným modelom bol v osemdesiatych rokoch *E-mu Emulator*. V priebehu vývoja nástrojov s týmto typom syntézy sa začali oddeľovať takzvané „samplery“ od syntetizátorov. Samplery boli väčšinou schopné nahrať zvuk, upraviť ho pre potreby prehrávania, poskladať z rôznych navzorkovaných signálov nástroj[17] a pri použití tieto vzorky prehrávať. Mnohé firmy ponúkali tzv. „samplovacie cédéčka“, čo boli vlastne zvukové banky pre samplery. Sample-based syntetizátory na rozdiel od samplerov využívali vzorky hlavne ako základné priebehy pre svoje oscilátory a ďalej fungovali na princípe subtraktívnej syntézy. Väčšinou nedisponovali funkciami na nahrávanie alebo editáciu vzoriek. Bolo v nich možné narábať len s pôvodnou bankou dodanou výrobcom. V dnešnej dobe nie je hranica medzi týmito dvomi typmi taká ostrá. Samplery disponujú funkciami syntetizátorov a sample-based syntetizátory ponúkajú množstvo funkcií samplerov. Tento typ syntézy je oproti ostatným jednoduchý na výpočty, a preto často disponuje vysokou polyfóniou a multitimbralitou. Napríklad model *E-mu Proteus 2000* disponuje polyfóniou 128 hlasov a multitimbralitou 32 nástrojov.

1.4.10 Kombinovaná syntéza

Drvivá väčšina moderných syntetizátorov implementuje viacero typov syntézy a kombinuje ich jednotlivé prvky. Kedysi boli takéto nástroje označované ako „hybridné“, ale dnes už je výnimočné, ak syntetizátor využíva len prvky jedného typu syntézy. Napríklad dnešné subtraktívne syntetizátory ponúkajú aspoň FM moduláciu, prípadne disponujú širokou škálou rôznych typov oscilátorov aplikujúcich rôzne typy syntézy,

[17] Typickým príkladom skladania nástroja z viacerých vzoriek je nahratie akustického nástroja na rôznych výškach tónu a priradenie vzoriek jednotlivým MIDI notám.

prípadne kombinované s deformáciou fázy.

2. Softvérové syntetizátory

Dnes určite najpopulárnejším typom syntetizátorov sú softvérové nástroje. Tie existujú v rôznych formách pre rôzne platformy. Na svete je oveľa viac výrobcov tohto druhu syntetizátorov, ako výrobcov ostatných typov. Okrem toho, softvérové výrobky sú dostupné na predaj komukoľvek na svete prostredníctvom internetu. Navyše, vďaka internetu si človek veľmi jednoducho vyberie výrobok, ktorý najlepšie vyhovuje jeho potrebám, prečíta si názory ľudí na produkt, o ktorý má záujem, a takmer každý výrobca softvéru poskytuje skúšobné verzie svojich produktov zadarmo. Demoverzie syntetizátorov bývaju väčšinou obmedzené tým, že v intervale niekoľkých desiatok sekúnd vygenerujú nejaký rušivý zvuk, aby sa zabránilo ich použitiu v serióznych projektoch.

2.1 Rozhrania pre softvérové syntetizátory

Prvé softvérové syntetizátory boli súčasťou určitých hudobných programov alebo išlo o samostatné programy, ktoré reagovali na MIDI správy. V deväťdesiatych rokoch sa začali vyvíjať syntetizátory vo forme zvanej „plugin“.

> *Plugin je počítačový program, ktorý interaguje s hostiteľskou aplikáciou pre uskutočnenie určitej špecifickej funkcie „na požiadanie“.* [5]

Plugin syntetizátory nie je možné spustiť samostatne, ale potrebujú hostiteľský program, takzvaný „host“, ktorý pre ne sprostredkúva vstupy a preberá výstupy. Táto forma syntetizátorov má veľa výhod:

- programátor sa nemusí starať o ovládače pre MIDI, pre zvukový výstup a viacero systémových vecí a má možnosť sa zamerať len na to dôležité – na algoritmy syntézy a na správanie sa syntetizátora podľa zmien parametrov,

- hudobné programy a sekvencery[1] dokážu jednoducho využiť ktorýkoľvek plugin podporovaného rozhrania,
- týmto spôsobom je značne zjednodušené portovanie[2] pluginov pre iné platformy,
- zákazník si môže vybrať sekvencer, ktorý mu vyhovuje, a syntetizátory, ktoré mu vyhovujú.

2.1.1 Samostatné syntetizátory

Samostatné syntetizátory nepotrebujú hostiteľskú aplikáciu, dodávajú sa vo forme spustiteľných súborov (.exe pre platformu Windows). Výhoda týchto samostatných syntetizátorov je zrejme len v prípade, že sú využívané a ovládané externým sekvencerom, a teda hostiteľský program nemusí zbytočne zaberať prostriedky počítača. Často aj v hostiteľských programoch bývajú určité funkcie nesprávne implementované, takže samostatný syntetizátor je zodpovedný sám za svoje správanie. Samostatné verzie bývajú často súčasťou balíkov spolu s rôznymi plugin verziami.

2.1.2 VST syntetizátory

Najpopulárnejším rozhraním pre syntetizátory je v súčasnej dobe rozhranie *VST*. VST je skratka pre *Virtual Studio Technology*, čiže technológia virtuálneho štúdia. Práve virtuálne štúdio bolo víziou spoločnosti *Steinberg Media Technologies GmBH*, keď v roku 1996 toto rozhranie prvýkrát predstavila vo svojom programe *Cubase VST 3.0*. Tento program sa počas svojho vývoja zameral na všestranné pokrytie profesionálnych potrieb hudobníka.

VST je otvorené rozhranie,. Ktokoľvek po zaregistrovaní sa na webstránke spoločnosti Steinberg má možnosť stiahnuť si VST SDK (VST Software Development Kit), čo je balík zdrojových kódov, príkladov, nástrojov a dokumentácie, nevyhnutný pre vývoj VST nástrojov. Rozhranie VST je platformovo nezávislé a balík VST SDK poskytuje základy pre vývoj aplikácií pre rôzne platformy.

[1] Sekvencer je program alebo zariadenie, ktoré umožňuje vytvoriť časové postupnosti (sekvencie) nôt a iných MIDI udalostí, uložiť ich a prehrávať, a tým ovládať nástroj s podporou MIDI.

[2] Portovanie je vytvorenie verzie pre inú platformu.

VST syntetizátory sa tiež označujú ako *VSTi* (VST Instrument) a VST je všeobecnejšie označenie zahŕňajúce aj efektové pluginy.

2.1.3 DX syntetizátory

DX je skratka pre rozhranie *DirectX* firmy *Microsoft*. Toto rozhranie je veľmi podobné s rozhraním VST, avšak DirectX funguje len pod platformou Windows. Podobné je aj v tom, že pre nástroje sa za skratku rozhrania píše malé „i“ – *DXi*. Pre tvorbu DX pluginov ponúka Microsoft na svojej webovej stránke DirectX SDK.

Vzhľadom na podobnosť rozhraní DX a VST bývajú mnohé syntetizátory ponúkané pre obidve tieto rozhrania. Je len málo DX nástrojov na trhu, ktoré by nemali verziu pre VST, a súčasne mnoho tých, ktoré existujú len pre rozhranie VST. Z týchto dôvodov a aj vďaka multiplatformovosti VST stráca postupne rozhranie DX význam.

2.1.4 AU syntetizátory

Rozhranie *Audio Unit* je natívne rozhranie platformy *Mac Os X* firmy *Apple Computer*. Toto rozhranie je výpočtovo veľmi efektívne, pretože je priamo podporované jadrom operačného systému Mac Os X. Pluginy AU sú u užívateľov počítačov Macintosh veľmi populárne.

2.1.5 Iné rozhrania

Existuje niekoľko ďalších rozhraní, ktoré sú ale menej rozšírené oproti doteraz spomenutým.

RTAS – Real-Time Audio Suite – štandardné rozhranie pre profesionálny softvér *Pro Tools*. Pluginy RTAS využívajú výkon procesora hostujúceho počítača.

TDM – Time Domain Multiplex – štandardné rozhranie pre Pro Tools. Pluginy TDM využívajú pre výpočty dedikované DSP karty.

MAS – MOTU Audio System – rozhranie pluginov pre audiosoftvér firmy *MOTU*.

Rozhrania RTAS a TDM sú uzavreté, firma Digidesign má veľmi prísne kritériá pri výbere výrobcov syntetizátorov, ktorým dá povolenie vyvíjať pluginy pre tento typ

rozhrania. Okrem toho, licencie a poplatky za používanie týchto rozhraní sú veľmi drahé.

2.2 Rozhranie VST

Obr. 2.1: Virtual Studio Technology

Rozhranie VST podporuje tri rôzne druhy pluginov:

VST Instrument – Syntetizátor,

VST Effect – Efekt,

VST MIDI effect – MIDI Efekt – jeho vstupom aj výstupom sú MIDI správy.

2.2.1 Architektúra rozhrania VST

Hostiteľskú aplikáciu nezaujíma, čo sa deje vo vnútri, len sprostredkuje MIDI a iné VST špecifikáciou definované udalosti pluginu, prípadne audiovstup (v prípade efektu), a prijíma MIDI alebo audiovýstupy. VST špecifikácia definuje aj niekoľko správ, ktorými sa plugin a host dorozumievajú.

Pre platformu Windows má VST plugin podobu dynamicky linkovanej knižnice (DLL). VST host po načítaní pluginu vytvorí preň systémové okno a potom sa už plugin stará o udalosti prebiehajúce v jeho okne sám.

Hostiteľská aplikácia má nasledujúce spôsoby interakcie s pluginom:

- vytvára a ruší plugin,
- môže na pokyn užívateľa uložiť aktuálne nastavenia pluginu do súboru a kedykoľvek na požiadanie opäť obnoviť,
- upozorňuje plugin na zmenu nastavení hostu (napr. vzorkovacia frekvencia, spôsob fungovania potenciometrov),

- posiela MIDI správy,
- môže nastaviť ktorýkoľvek parameter na požadovanú hodnotu (využíva sa pri automatizácii),
- môže si vyžiadať hodnotu parametra a textové informácie o parametri (pre potreby zobrazenia mimo okna pluginu, napríklad pri automatizácii),
- môže vyžiadať spracovanie bloku vzoriek.

Plugin má nasledujúce spôsoby interakcie s hostom:

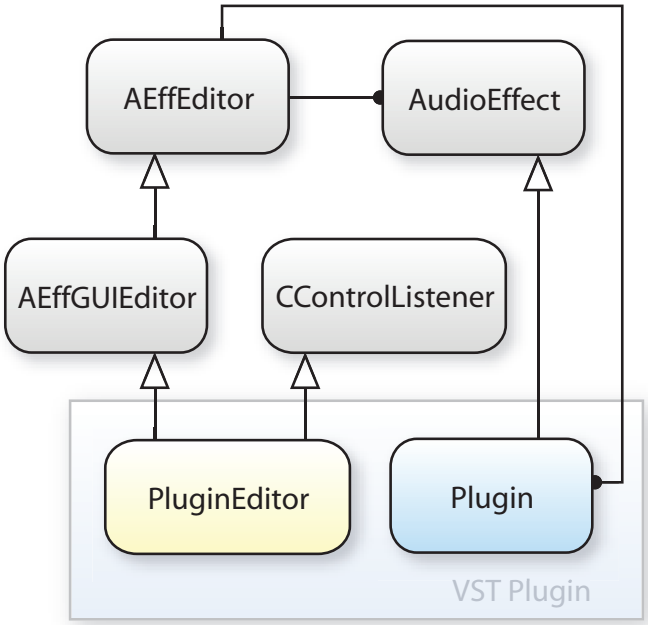
- môže zistiť aktuálne informácie hudobného projektu (napr. tempo sekvencera),
- môže zistiť systémové nastavenia hostu,
- môže infomovať host o interakcii užívateľa s oknom pluginu (napríklad zmena parametra).

Audiosignál vstupu a výstupu je typu *32bit float*, hodnoty vzoriek by sa mali pohybovať v základnom dynamickom pásme –1,0 – 1,0. Vyššie hodnoty sú povolené, len aby sa predišlo skresleniu signálu, ale hodnoty mimo tohto rozsahu by sa nemali bežne používať. Hodnoty parametrov na rozhraní s hostom sú tiež typu *32bit float*.

2.2.2 VST SDK

VST SDK je balík zdrojových kódov, príkladov, nástrojov a dokumentácie, pripravený spoločnosťou Steinberg pre vývoj VST nástrojov a efektov. VST SDK poskytuje framework vo forme zdrojových kódov v jazyku C++ aj s projektom pre vývojové prostredie *Microsoft Visual Studio*. Tieto zdrojové kódy obsahujú všetko nevyhnutné pre vyvíjanie VST pluginov. Základné metódy komunikácie s hostom sú implementované vo funkciách jazyka C, a tieto sú obalené triedami jazyka C++.

Na obrázku 2.2 sú znázornené vzťahy medzi základnými triedami VST SDK. Triedy Plugin a PluginEditor sú základné triedy pluginu zdedené z tried VST SDK. AudioEffect je trieda, ktorá má implementované všetky metódy komunikácie s hostom. AEffEditor a AEffGUIEditor sú triedy užívateľského rozhrania a CControlListener je ovládač udalostí.



Obr. 2.2: Vzťahy medzi triedami VST SDK

VST SDK framework je zvláštne navrhnutý, nelogicky štrukturovaný a veľmi slabo a nekompletne zdokumentovaný. Spolu s frameworkom sa v balíku nachádza aj niekoľko ukážkových zdrojových kódov, ktoré dávajú o niečo lepší obraz o fungovaní tejto architektúry. Dokument [6] kompletnejšie vysvetľuje architektúru VST a pojednáva aj o chybách a nedostatkoch vývojárskeho balíka VST SDK.

2.2.3 Knižnica VSTGUI

VSTGUI je knižnica určená na implementáciu grafického používateľského rozhrania pre pluginy VST a je súčasťou balíka VST SDK. VSTGUI poskytuje možnosti implementácie platformovo nezávislého používateľského rozhrania. Obsahuje základné ovládacie prvky, napríklad potenciometre, tlačidlá, posúvacie lišty a veľa ďalších. VSTGUI pracuje s grafickými súbormi BMP, vo verzii 3.0 však podporuje aj formát PNG s alfakanálom pre priehľadné a polopriehľadné grafické prvky. K používaniu súborov PNG je nevyhnutné použitie knižníc *libpng*[3] a *zlib*[4].

[3]libpng je knižnica pre podporu formátu PNG. Táto knižnica je závislá na knižnici *zlib*. [http://www.libpng.org]

[4]zlib je knižnica sprostredkujúca funkcie kompresie a dekompresie dát. [http://zlib.net]

Knižnica VSTGUI poskytuje okrem základných komponentov aj abstraktné triedy, ktoré je možné jednoducho použiť na implementáciu vlastných komponentov. VSTGUI ponúka aj základné funkcie pre vykresľovanie primitívnych grafických tvarov a textov. Toto vykresľovanie je ale pod platformou Windows veľmi zastaralé a nepodporuje pokročilejšie funkcie, ako napríklad antialiasing[5].

2.3 Spôsoby tvorby VST syntetizátorov

Od vzniku rozhrania VST bolo dlhé roky nutné ovládať programovacie jazyky, infraštruktúru operačného systému a mnoho iných vecí na vytvorenie nástroja pre toto rozhranie. V dnešnej dobe ale už existujú aj rôzne vizuálne prostredia, ktoré značnou mierou zjednodušujú tvorbu týchto nástrojov.

2.3.1 Vizuálne prostredia pre tvorbu VST

Najznámejšie vizuálne prostredia sú *SynthEdit*[6] a oveľa modernejší *SynthMaker*[7] od firmy *Outsim*. Tieto prostredia poskytujú nástroje pre rýchly vývoj syntetizátorov a efektov, veľké množstvo predprogramovaných komponentov a možnosti doprogramovania vlastných komponentov. Tvorba zložitých nástrojov v takýchto prostrediach ale trpí nižšou efektivitou kódu a vyšším zaťažením procesora. Obidve spomenuté prostredia sú komerčné a cena prostredia SynthMaker je rôzna pre osobné a pre komerčné účely.

2.3.2 Knižnice a frameworky pre tvorbu VST

Priamou kompiláciou efektívne navrhnutého kódu je možné dosiahnuť vyšší výkon VST nástrojov, ako pri tvorbe vo vizuálnom prostredí. Tu je ale nevyhnutná pokročilá znalosť programovania, programovacieho jazyka a architektúry operačného systému a hardvérových platforiem, pre ktoré sa nástroj vyvíja.

[5] Antialiasing v kontexte grafiky je odlišný problém, ako pri vzorkovaní zvuku. V grafike ide o spôsob vykresľovania grafických elementov s vyhladzovaním okrajov objektov metódou miešania farieb objektu a pozadia.

[6] SynthEdit – [http://www.synthedit.com]

[7] SynthMaker – [http://synthmaker.co.uk]

Pre programovanie a kompiláciu VST nástrojov existujú knižnice a frameworky pre jazyky:

C++ – VST SDK,

Delphi – Delphi VST SDK[8] je framework pre Delphi, ktorý obsahuje vylepšenia a zjednodušenia originálneho VST SDK pre C++,

Java – jVSTwRapper[9] je wrapper tried VST SDK pre programovací jazyk Java.

Vzhľadom na to, že aj programovacie prostredia sú väčšinou komerčné a ich cena často niekoľkonásobne prevyšuje vizuálne vývojové prostredia, určite je pre záujemcov, ktorí nemajú ambície vyvíjať zložité profesionálne nástroje, výhodnejšie zakúpiť jedno z uvedených vizuálnych prostredí.

[8] Delphi VST SDK – [http://www.tobybear.de/d_template]
[9] jVSTwRapper – [http://jvstwrapper.sourceforge.net]

3. Súčasný stav VST syntetizátorov

Pre potreby návrhu a implementácie akéhokoľvek druhu softvéru je nevyhnutné pochopiť princípy jeho fungovania, osvojiť si zvyky a štandardy, zaužívané v aplikáciách tohto typu, zistiť nevýhody a chyby v existujúcich implementáciách a tiež nechať sa inšpirovať zaujímavými a užitočnými prvkami týchto implementácií. Je veľmi užitočné urobiť prieskum medzi odborníkmi pracujúcimi v danej oblasti, ktorí sa každodenne s daným typom softvéru stretávajú, čo im na existujúcom softvéri vyhovuje, čo im prekáža a čo im chýba. Toto isté platí aj pre vytváranie softvérového syntetizátora.

3.1 Ponuka voľne šíriteľných VST syntetizátorov

V súčasnosti je k dispozícii veľké množstvo rôznych druhov VST syntetizátorov a aj niekoľko internetových stránok ponúkajúcich databázy existujúcich produktov. Najvýznamnejšou z týchto stránok je stránka *KVR AUDIO* (kvraudio.com), ktorá ponúka komplexný prehľad o existujúcich nástrojoch a efektoch, tak komerčných, ako aj voľne šíriteľných, a ponúka aj mnohé iné služby pre vývojárov aj pre samotných používateľov. Na stránke KVR AUDIO je možné vyhľadávanie v databáze podľa mnohých kritérií.

Témou tejto práce je navrhnúť a implementovať subraktívny syntetizátor. Na účely zhodnotenia stavu subtraktívnych nástrojov pre rozhranie VST som vybral skupinu voľne šíriteľných syntetizátorov, pretože komerčné nástroje sú často komplexné, kombinujúce rôzne druhy syntéz, a navyše nie všetky ponúkajú plne funkčnú demoverziu. Na webstránke KVR AUDIO som pri vyhľadávaní vybral kategóriu freeware subtraktívnych syntetizátorov pre rozhranie VST kompatibilné s operačným systémom Windows XP. Z 235 nájdených výsledkov som rýchlym testovaním vyradzoval tie, ktoré hneď v prvých minútach testovania pôsobili neseriózne, neprístupne a

nestabilne. Týmto testom prešlo úspešne len vyše 60 syntetizátorov, z ktorých som hlbším testovaním vybral 30 najlepších. Keďže nemám so syntetizátormi zďaleka toľko skúseností, ako tvorcovia hudby, požiadal som štyroch elektronicky zameraných hudobníkov z komunity *Corenforce* o pomoc pri testovaní.

3.2 Kritériá testovania

Najdôležitejšími kritériami pri testovaní boli:

Kvalita zvuku – harmonický obsah, ruchy, aliasy, kvalita filtrov a efektov,

Rozsiahlosť možností – či je syntetizátor schopný vytvoriť široké spektrum rôznych zvukov,

Ovládanie a intuitívnosť – či syntetizátor reaguje predvídateľne a správne na používateľské vstupy,

GUI a prehľadnosť – kvalita používateľského rozhrania, logická štruktúra a rozmiestnenie komponentov.

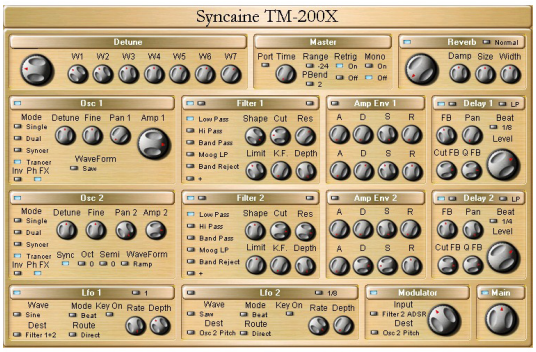
Od hudobníkov som požadoval krátke zhrnutie pozitívnych a negatívnych stránok každého syntetizátora, a celkové hodnotenie použitím stupnice 0 – 10 bodov.

3.3 Vyhodnotenie výsledkov testu

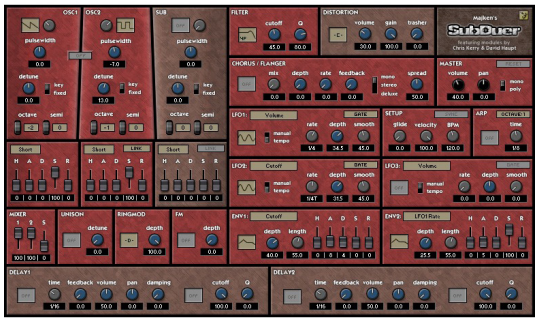
umiestnenie	názov (výrobca)	priemerné hodnotenie
1.	**Superwave P8** (Superwave)	7,8
2.	**Syncaine TM-200X** (Syncer)	7,5
	SubDuer (Majken)	7,5
4.	**Evol** (Synthtronic)	6,9
5.	**Lallapallooza** (Buzzroom)	6,8

Table 3.1: Výsledok hodnotenia syntetizátorov hudobníkmi

Superwave P8



Syncaine TM-200X



SubDuer

Obr. 3.1: Prvé tri najlepšie subtraktívne syntetizátory

Bodové hodnotenia syntetizátorov od jednotlivých hudobníkov boli celkom podobné, čo naznačuje určitú mieru objektivity výsledkov testovania. Priemerná odchýlka hodnotenia konkrétneho syntetizátora hudobníkom od priemeru jeho hodnotení je 1,2 bodu. Päť najlepšie ohodnotených syntetizátorov je uvedených v tabuľke 3.1. Bodové hodnotenia sami o sebe nenesú hodnotnú informáciu o konkrétnych výhodách alebo chybách a nedostatkoch jednotlivých syntetizátorov užitočných pre návrh nového nástroja. Z tohto dôvodu som vyžadoval od testujúcich hudobníkov aj krátky textový popis zdôvodňujúci hodnotenie.

3.3.1 Chyby a nedostatky testovaných syntetizátorov

Pri porovnávacích testoch vyplávalo na povrch mnoho chýb a nedostatkov. Tie významné nedostatky beriem do úvahy pri návrhu vlastného syntetizátora.

Chyby vo zvukovom výstupe

Falošné tóny – nepresné prepočty frekvencií zodpovedajúcich konkrétnym notám spôsobujú počuteľné rozladenie vysokých a nízkych oktáv.

Pískanie rezonancie filtra – rezonančné pásmo filtra je príliš úzke a namiesto príjemného zvýraznenia hraničného pásma vzniká nepríjemné pískanie. Touto chybou trpí veľká skupina testovaných syntetizátorov, zrejme z dôvodu použitia toho istého filtrovacieho algoritmu.

Iné nedostatky filtra – málo druhov filtra, napríklad len lowpass, nežiaduce efekty pri zmene parametrov.

Chyby v ovládaní

Neštandardný systém točenia potenciometrov – nástroj ignoruje nastavenie hostu o systéme točenia potenciometrov, prehnaná citlivosť parametrov, pri ktorej je obtiažne trafiť konkrétnu hodnotu.

Neštandardné fungovanie komponentov – ovládanie nástroja nezodpovedá zaužívaným spôsobom fungovania syntetizátorov a pri takomto nedostatku nie je možné s nástrojom intuitívne pracovať.

Absencia zobrazovania hodnôt parametra – bez zobrazovania hodnoty parametra nemožno nastaviť požadované hodnoty presne.

Malý rozsah parametra – nemožnosť nastaviť určité hodnoty, napríklad lowpass filter pri maximálnej hodnote hraničnej frekvencie neprepúšťa celé počuteľné spektrum, ale počuteľne filtruje vysoké frekvencie.

Pukanie pri točení parametrov – pri rýchlom točení parametrov sa hodnoty menia nespojito, čo pri viacerých parametroch vytvára nežiaduce vysokofrekvenčné impulzy.

Ostatné chyby

Neestetický dizajn – veľmi negatívne ovplyvňuje prácu s nástrojom.

Nelogické alebo neprehľadné rozmiestnenie ovládacích prvkov – nie je možné intuitívne meniť požadované parametre, slabý kontrast ovládacích prvkov oproti pozadiu.

Zbytočne veľká plocha syntetizátora – pracovná plocha monitora je veľmi drahá a syntetizátory využívajúce túto plochu neefektívne značne znižujú produktivitu práce.

Nečitateľnosť – použitie neštandardných skratiek s cieľom ušetriť plochu vedie k nejasnosti, čo ktorý parameter ovláda.

Nestabilita – pri určitých nastaveniach syntetizátor prestane vydávať zvuk a je nutné ho reštartovať.

Absencia podpory rôznych vzorkovacích frekvencií – neberie do úvahy vzorkovaciu frekvenciu a generuje zvuk vždy konštantne vzhľadom na vzorky alebo po zmene vzorkovacej frekvencie prestáva fungovať úplne.

3.3.2 Výhody testovaných syntetizátorov

Niektoré syntetizátory vynikali oproti iným určitými ojedinelými vlastnosťami, ktoré buď rozširovali možnosti použitia syntetizátora, alebo uľahčovali prácu s ním.

Neštandardné priebehy oscilátorov – ponúkajú iné spektrá zvukov ako bežne používané priebehy.

Rozsiahle modulačné možnosti – čím viac je možností modulácie, tým menej obmedzené je spektrum zvukov a zvukových efektov, ktoré je možné vygenerovať.

Viac ako jeden filter – viac filtrov s možnosťou rôzneho vzájomného zapojenia prináša nielen väčšiu kontrolu nad frekvenčným spektrom zvuku, ale aj väčšiu variabilitu strmosti filtrovania.

Rôzne strmosti filtrov – bežne používané 12, 18 a 24 dB/oct.

Unisono – pridanie jedného alebo viacerých mierne rozladených oscilátorov pre generovanie mohutnejšieho zvuku.

Integrované efekty – chorus, delay, reverb, phaser alebo iné efekty pre spestrenie zvukových možností.

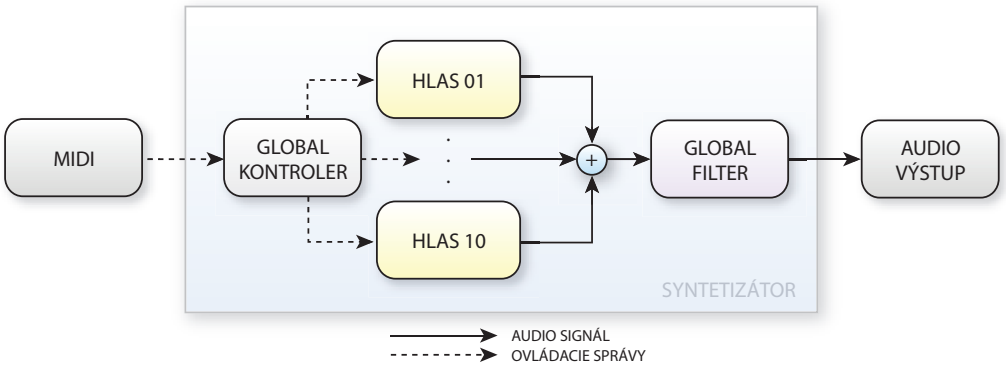
Nadštandardné obálky – čo najväčšia kontrola nad vývinom zvuku v čase.

Graficky znázornené parametre – pre čo najväčšiu prehľadnosť a intuitívnosť.

4. Návrh syntetizátora

4.1 Návrh architektúry syntetizátora

Prvým krokom návrhu syntetizátora by mal byť návrh jeho architektúry, popis komponentov a ich úloha v celom systéme. Na obrázku 4.1 je znázornený základný princíp fungovania s polyfonickými hlasmi.

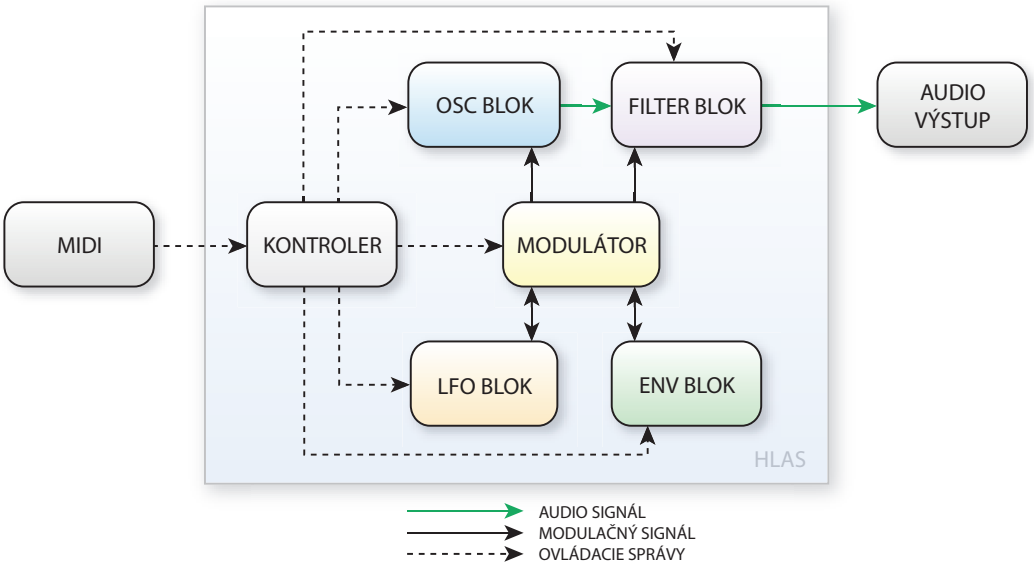


Obr. 4.1: Architektúra syntetizátora

Syntetizátor prijíma MIDI správy, GLOBAL KONTROLER ich analyzuje a relevantné správy posiela jednotlivým hlasom polyfónie. Syntetizátor disponuje polyfóniou 10 hlasov. Ich jednotlivé signály sa sčítavajú a vysledný signál sa filtruje v bloku GLOBAL FILTER od nepočuteľných nízkych frekvencií. Vyfiltrovaný signál sa posiela na výstup.

Obrázok 4.2 znázorňuje štruktúru jednotlivých polyfonických hlasov, bloky komponentov a toky signálov a správ medzi nimi.

Blok KONTROLER analyzuje prijaté MIDI správy a distribuuje ich ostatným blokom. Bloky OSC, LFO, ENV a FILTER prijímajú len MIDI noty, MODULÁTOR prijíma aj ostatné správy, ktoré syntetizátor podporuje. MODULÁTOR prijíma aj signály generované blokmi LFO a ENV a vysiela modulačné signály do všetkých štyroch funkčných



Obr. 4.2: Štruktúra hlasu polyfónie

blokov. Blok OSC je zodpovedný za generovanie zvukového signálu a blok filter filtruje tento signál. Zvukový výstup z filtra sa posiela na výstup celého hlasu.

4.2 Funkčné bloky architektúry

OSC blok

OSC blok sa skladá zo štyroch oscilátorov, mixéra a zosilňovača. Oscilátory generujú jednoduchý signál, v mixéri sa sčítavajú, pričom amplitúdy a smerovania[1] výstupov jednotlivých oscilátorov je možné ovládať parametrami a moduláciami. Signál z mixéra ešte prechádza zosilňovačom. Jeho úloha je dynamicky ovládať amplitúdu výstupu OSC bloku, najčastejšie podľa obálky.

FILTER blok

FILTER blok filtruje signál dvoma filtrami, ktoré môžu byť zapojené sériovo alebo paralelne, a filtrovaný signál posiela na výstup.

[1]Smerovanie (pan) v kontexte zvukového signálu je jeho pozícia medzi pravým a ľavým kanálom.

LFO blok

LFO blok obsahuje štyri nízkofrekvenčné oscilátory a nimi vygenerovaný signál posiela modulátoru.

ENV blok

ENV blok obsahuje štyri generátory obálok a vygenerovaný signál posiela modulátoru.

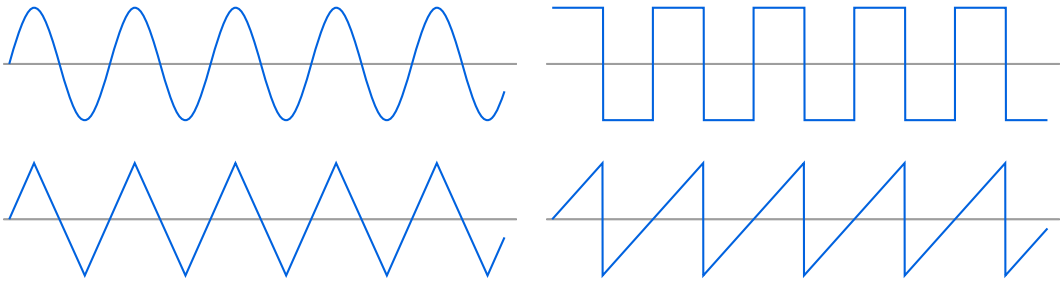
MODULÁTOR

MODULÁTOR je blok, ktorý obsahuje prostriedky na distribúciu modulačných signálov modulovaným parametrom podľa modulačnej matice. Modulačná matica je zoznam modulácií (trojíc - zdroj, cieľ a miera modulácie), podľa ktorej sa modulujú signály a parametre v syntetizátore.

4.3 Návrh komponentov

Základné funkcie jednotlivých komponentov už boli opísané. V tejto sekcii sa zameriam na vyšpecifikovanie požiadaviek na jednotlivé komponenty.

4.3.1 Oscilátor



Obr. 4.3: Základné analógové priebehy

Oscilátor by mal disponovať piatimi základnými priebehmi:

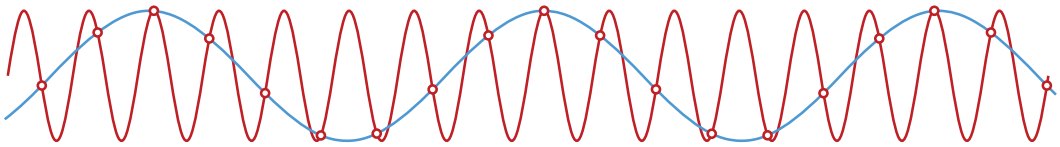
- sínus,
- píla,

- trojuholník,
- obdĺžnik,
- šum.

Počítanie týchto priebehov by bolo zbytočne náročné na výkon procesora, a preto by mal syntetizátor tieto priebehy predpočítať v dostatočnom rozlíšení a uložiť do pamäte.

Priebehy *píla*, *trojuholník* a *obdĺžnik* ale nie je také jednoduché vytvoriť, pretože jednoduchý geometricky presný priebeh nezodpovedá v navzorkovanom signále signálu analógovému. Pri jeho prehrávaní by vznikli digitálne artefakty vo forme nežiaducich frekvencií – aliasy. Tu vzniká potreba návrhu nejakého antialiasového opatrenia.

Antialiasing oscilátora



Obr. 4.4: Ukážka aliasovania signálu

Na obrázku 4.4 je názorná ukážka vzorkovania signálu s frekvenciou väčšou, ako je polovica vzorkovacej frekvencie. Tento signál sa do digitálnej podoby premietne ako signál s oveľa nižšou frekvenciou. Po takomto navzorkovaní sa už tento nežiaduci efekt vo všeobecnosti nedá odstrániť, takže jediný spôsob ako zabrániť aliasovaniu je odstrániť tieto vysoké frekvencie ešte pred vzorkovaním signálu.

Softvérový syntetizátor ale v žiadnej fáze generovania zvuku nepracuje s analógovým signálom, a preto je nutné nájsť spôsob, ako vytvárať tieto vlny antialiasované na všetkých počuteľných frekvenciách s celým počuteľným harmonickým spektrom.

Na vytvorenie vlny pre konkrétnu frekvenciu možno použiť Fouriérove rady.

Píla

$$x(t) = \frac{2}{\pi} \sum_{k=1}^{\infty} \frac{\sin(2\pi k f\, t)}{k}$$

Obdĺžnik

$$x(t) = \frac{4}{\pi} \sum_{k=1}^{\infty} \frac{\sin((2k-1)2\pi f\, t)}{(2k-1)}$$

Trojuholník

$$x(t) = \frac{8}{\pi^2} \sum_{k=1}^{\infty} \sin\left(\frac{k\pi}{2}\right) \frac{\sin(2\pi k f\, t)}{k^2}$$

Symbol ∞ pri generovaní je potrebné nahradiť konštantou $kmax$, pri ktorej by hodnota $kmax \times f$ bola menšia ako nyquistova frekvencia.

Tento spôsob síce vytvorí antialiasovanú vlnu pre konkrétnu frekvenciu, ale syntetizátor musí byť schopný vygenerovať akúkoľvek frekvenciu v rámci počuteľného spektra. Táto vlna by pri prehrávaní na vyššej frekvencii vytvárala aliasy a pri nízkej frekvencii by zas neobsahovala dostatok vysokých harmonických zložiek.

Za najpraktickejšie riešenie tohto problému považujem vytvorenie tabuľky digitálnych vĺn s rôznym harmonickým obsahom pre každý priebeh. To znamená, že pre každý priebeh (píla, trojuholník, obdĺžnik) vytvorím tabuľku s určitým počtom vĺn rozmiestnených rovnomerne na logaritmickej frekvenčnej osi, napríklad jednu vlnu pre každú oktávu.

To by znamenalo, že každú vlnu budeme prehrávať maximálne v rozsahu jednej oktávy, čiže minimálna frekvencia prehrávania by bola polovica maximálnej. A teda, aby sme sa vyhli aliasovaniu, pri maximálnej frekvencii prehrávania vlny pre vzorkovaciu frekvenciu 44 100 Hz môže vlna obsahovať frekvencie do 22 kHz, ale prehrávaná na minimálnej frekvencii by jej najvyššia harmonická zložka mala len 11 kHz, a to by už bolo počuteľné orezanie vysokých frekvencií.

Ja som zvolil pre svoj syntetizátor odstupy jednotlivých digitálnych vĺn jednu tretinu oktávy, čiže 4 poltóny. Najnižšiu frekvenciu v tabuľke som zvolil 27,5 Hz, čo zodpovedá note A0, a harmonické zložky sa generujú po hranicu 20 000 Hz. S odstupmi po 4 poltóny, až kým prvá harmonická zložka nie je nad hranicou 20 000 Hz, je to spolu 27 vĺn. Vlny sa generujú v rozlíšení 384 000 Hz, čo je dvojnásobok maximálnej komerčne používanej vzorkovacej frekvencie, aby sa zaručila čo najlepšia

presnosť oscilátorov.

Na generovanie akejkoľvek reálnej počuteľnej frekvencie sa budú vzorky prepočítavať z dvoch susedných vĺn lineárnou interpoláciou, pričom z každej vlny sa aktuálna vzorka počíta tiež lineárnou interpoláciou susedných hodnôt.

Výpočet frekvencie pre konkrétnu MIDI notu sa vykonáva vzorcom

$$f = 440 \times 2^{\left(\frac{n-69}{12}\right)}$$

kde n je číslo MIDI noty, 69 je číslo noty A4, 440 je frekvencia noty A4 v Hertzoch a $\frac{1}{12}$ je podiel noty v oktáve na logaritmickej osi.

Parametre oscilátora

Ladenie – Oscilátor musí mať možnosť pomerne presného ladenia vo veľkom rozsahu. V uživateľskom rozhraní to budú tri potenciometre: OCTAVE pre ladenie v rozsahu od –4 do +4 oktávy, COARSE po poltónoch vrozsahu od –12 do +12 poltónov, a FINE pre mikroladenie v rozsahu od –100 do +100 centov.

Vlnový priebeh – Výber medzi spomínanými priebehmi.

Synchronizácia – Možnosť synchronizovať s jedným iným oscilátorom metódou SoftSync[2].

4.3.2 Generátor obálky

Obálka je veľmi dôležitý prvok pri syntéze, pretože definuje vývoj niektorého parametra, a tým aj celého zvuku v čase. Na obrázku 4.5 sú načrtnuté parametre obálky a ich vplyv na jej priebeh.

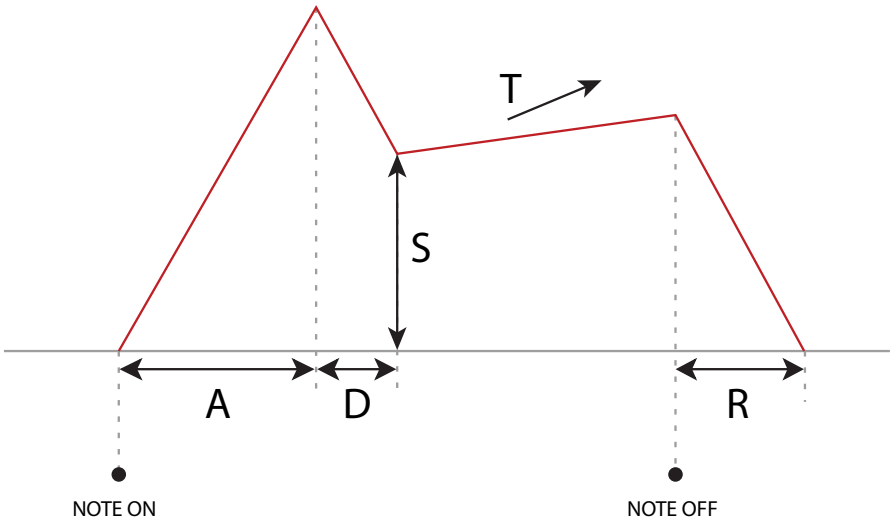
Parametre obálky

Attack – čas nábehu signálu na maximálnu hodnotu,

Decay – čas poklesu signálu z maximálnej úrovne na úroveň SUSTAIN,

Sustain – úroveň, na akú má signál klesnúť vo fáze DECAY s rozsahom 0 – 100 %,

[2]SoftSync – pri tejto synchronizácii je fáza synchronizovaného oscilátora vynulovaná vždy, keď synchronizačný oscilátor začína novú periódu.



Obr. 4.5: Obálka ADSTR

Time – tendencia signálu stúpať alebo klesať po dosiahnutí hodnoty SUSTAIN, nadobúda hodnoty v rozsahu od –100 % do +100 %, kde 0 % znamená žiadne stúpanie/klesanie, kladné hodnoty určujú stúpanie a záporné klesanie.

Release – čas poklesu úrovne signálu po uvoľnení klávesy na nulu.

Časové parametre majú rozsah 0 – 12 sekúnd, pričom je potrebná väčšia hustota hodnôt blízko nuly ako pri vysokých hodnotách.

Obálka sa spúšťa prijatím MIDI udalosti NOTE ON. Po prechode fázami ATTACK a DECAY zotrváva vo fáze SUSTAIN až do prijatia správy NOTE OFF a prechádza do fázy RELEASE. V prípade prijatia správy NOTE OFF skôr, ako vo fáze SUSTAIN, sa rovnako prechádza do fázy RELEASE okamžite.

4.3.3 LFO – nízkofrekvenčný oscilátor

Nízkofrekvenčný oscilátor využíva rovnaké priebehy ako oscilátor s dvoma rozdielmi:

- nízkofrekvenčný oscilátor nepotrebuje antialiasové opatrenia,
- namiesto šumu využíva postupnosti náhodných hodnôt.

Priebeh náhodných hodnôt

Syntetizátor využíva 2 typy priebehov náhodných hodnôt, pričom obidva generujú jednu náhodnú hodnotu pre každú periódu. Rozdiel medzi nimi je v tom, že prvý túto hodnotu drží počas celej periódy, pričom v druhom sa hodnoty susedných períód interpolujú.

Parametre nízkofrekvenčného oscilátora

frekvencia – frekvencia oscilátora v rozsahu 0,01 Hz – 100 Hz,

fázové posunutie – fázové posunutie oscilátora v rozsahu od –180° do +180°,

opozdenie – čas opozdenia oscilácie v rozsahu 0 – 12 sekúnd,

nábeh – čas nábehu, čiže postupného zvyšovania amplitúdy oscilácie v rozsahu 0 – 12 sekúnd,

globálny/lokálny – prepínač, určujúci či bude oscilátor lokálny pre každý hlas alebo globálny pre všetky hlasy spolu.

Nízkofrekvenčný oscilátor je spúšťaný MIDI správou NOTE ON, spustí sa fáza opozdenia. Po uplynutí času opozdenia sa začína generovať signál s fázovým posunutím podľa hodnoty príslušného parametra. V čase nábehu sa amplitúda plynule zvyšuje z nuly na maximálnu úroveň.

4.3.4 Filter

Úloha filtra je odstrániť alebo potlačiť určité pásma frekvencií v signále. Pri analógových filtroch to robia RC články. V digitálnej doméne existujú dva druhy filtrov:

FIR – Finite Impulse Response – filtre s konečnou dĺžkou impulzovej odozvy[3].

IIR – Infinite Impulse Response – filtre s nekonečnou dĺžkou impulzovej odozvy.

Digitálne filtre FIR fungujú na princípe spočítavania aktuálnych vzoriek s jednou alebo viacerými predchadzajúcimi vzorkami vynásobenými koeficientami filtra. Ko-

[3]Impulzová odozva je výstup zariadenia po poslaní impulzu na vstup. Impulz pre digitálne zariadenia je signál, kde prvá vzorka má maximálnu úroveň, a všetky nasledujúce majú hodnotu 0.

eficienty filtra presne definujú frekvenčnú odozvu filtra[4]. Rád filtra je maximum zo vzdialeností vzoriek použitých pri výpočtoch od aktuálnej vzorky. Ak filter využíva dve predošlé vzorky, je to filter druhého rádu.

Filtre IIR fungujú podobne ako filtre FIR, ale okrem predošlých vstupov používajú aj predošlé výstupy. Preto sa nazývajú aj filtrami so spätnou väzbou. Ich výhodou oproti filtrom FIR je, že dokážu pri nízkych rádoch poskytnúť strmé frekvenčné odozvy, na aké by bolo nutné použiť filter FIR oveľa vyššieho rádu. IIR filtre sú teda výpočtovo rýchlejšie, avšak nie je zaručená ich stabilita[5] a tiež fázová odozva[6] sa pri nich ťažko udržiava lineárna.

Rád filtra je dôležitý parameter, pretože určuje strmosť filtrovania, a má tiež veľký vplyv na výpočtovú zložitosť filtrovania. V hudobných aplikáciách sa využívajú filtre so strmosťou 12 – 24 dB na oktávu. Pre môj syntetizátor som zvolil IIR filter druhého rádu (označovaný aj ako „bikvadratický filter“[7] , alebo „biquad“), ktorého schéma je na obrázku 4.6. Sériovým zapojením dvoch filtrov tohto typu vytvoríme filter so strmosťou 24 dB/oktávu.

Prenosová funkcia bikvadratického filtra

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Element z^{-1} predstavuje opozdenie signálu o jednu vzorku, b_0, b_1, b_2, a_1 a a_2 sú koeficienty vypočítané z parametrov filtra.[8]

Parametre filtra

Typ filtra – prepínač štyroch typov filtra (lowpass, highpass, bandpass, band-stop).

Aktívna frekvencia – hraničná frekvencia pre typy lowpass a highpass, priepustná pre band-pass a filtrujúca pre band-stop. Rozsah týchto frekvencií je od 20 Hz do 20 000 Hz.

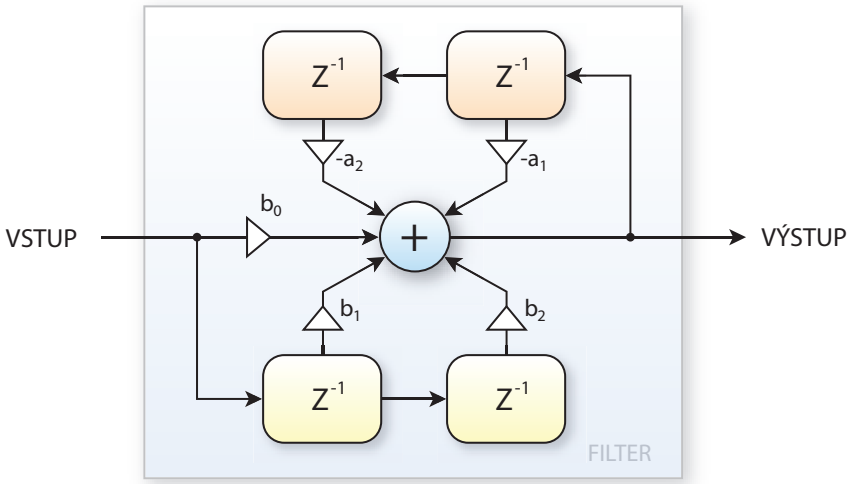
[4]Frekvenčná odozva filtra opisuje vzťah medzi zložkami vstupného a výstupného signálu v závislosti od ich frekvencie.

[5]Stabilita filtrov je tendencia impulzovej odozvy sa časom približovať k nule.

[6]Fázová odozva je vzťah fáz zložiek vstupného a výstupného signálu v závislosti od ich frekvencie.

[7]Bikvadratický filter – tento názov je odvodený od prenosovej funkcie v Z-doméne, ktorá je definovaná ako pomer dvoch kvadratických funkcií.

[8]Vzorce na výpočet koeficientov bikvadratického filtra sú opísané v dokumente [8].



Obr. 4.6: Schéma bikvadratického filtra

Q – rezonancia hraničného pásma pre typy lowpass a highpass, šírka pásma pre typy band-pass a band-stop. Rozsah parametra Q je 0 – 10.

Keyfollow – miera vplyvu výšky zahratej noty na aktívnu frekvenciu v rozsahu od –100 % do +100 % (0 % - žiadny vplyv, +100 % - aktívna frekvencia sa posúva presne podľa frekvencie zahratej noty, –100 % - aktívna frekvencia sa posúva v rovnakých intervaloch, ako zahratá nota, ale v opačnom smere).

Mix – percentuálny podiel signálu, ktorý je filtrovaný.

4.3.5 Mixér

Mixér je komponent, ktorý sčítava signály z oscilátorov.

Parametre mixéra

Úroveň oscilátorov – percentuálna úroveň signálu pre každý oscilátor.

Smerovanie oscilátorov – smerovanie signálu pre každý oscilátor. Hodnoty parametra sú v rozsahu 100L – 0 – 100R. Hodnoty predstavujú percentuálne umiestnenie signálu od stredu doľava (L) alebo doprava (R).

4.3.6 Zosilňovač

Zosilňovač je komponent, ktorý umožňuje regulovať úroveň signálu z mixéra. Jeho úloha v digitálnych syntetizátoroch nie je taká podstatná. V navrhovanom syntetizátore sprostredkúva moduláciám celkovú úroveň hlasu. Tento komponent nemá žiadne ovládacie parametre.

4.3.7 Modulačná matica

Modulačná matica obsahuje 32 slotov pre aplikáciu modulácií.

Parametre modulačných slotov

Zdroj modulácie - predstavuje modulačný signál.

Cieľ modulácie - predstavuje modulovaný parameter.

Miera modulácie - miera modulácie v percentách v rozsahu od –100 % do +100 %. Záporné hodnoty vynásobia modulačný signál hodnotou –1.

Zdrojom modulácie môže byť:

- MIDI správa,
- obálka,
- signál z nízkofrekvenčného oscilátora.

Cieľom modulácie môže byť každý spojitý parameter syntetizátora okrem miery modulácie.

Modulačný signál sa pri modulácii pripočítava k hodnote parametra vynásobený mierou modulácie. Výnimkou je modulácia úrovní signálov oscilátorov mixéra alebo celkovej úrovne zosilňovača obálkami alebo MIDI správami CC. Vo vymenovaných prípadoch dochádza k násobeniu namiesto sčítavania a pri zápornej miere modulácie obálkami sa signál obálok posúva pripočítaním maximálnej úrovne signálu obálok do nezáporných hodnôt.

4.3.8 Interpolácia hodnôt parametrov

Nespojitá zmena spojitých parametrov môže v mnohých prípadoch spôsobiť nežiaduce efekty vo zvukovom výstupe. Preto je potrebné takto citlivé parametre ošetriť pred nespojitou zmenou interpoláciou novej hodnoty s hodnotou pôvodnou. Na tento účel plne postačuje lineárna interpolácia s pevným časom prechodu z pôvodnej hodnoty do novej. Pre tento čas som určil hodnotu 50 milisekúnd.

4.4 Perzistentné ukladanie nastavení

Pre prácu so syntetizátorom je nevyhnutná možnosť uloženia aktuálnych nastavení parametrov a ich obnovenia kedykoľvek podľa potreby. Tiež je dôležité, aby sa tieto nastavenia ukladali automaticky pri ukladaní projektu hostiteľskej aplikácie a aby sa obnovili spolu s obnovením projektu.

Zoznam nastavení parametrov syntetizátora je v terminológii VST **program**. Často sa v rovnakom význame používajú aj termíny *preset* a *patch*. Množina viacerých programov sa označuje termínom **banka**.

VST SDK vo verzii 2.4 ponúka možnosť uložiť aktuálne nastavenia hostiteľským programom bez nutnosti implementácie takejto funkcionality v syntetizátore. Tento automatický spôsob ukladania nastavení hostom má niekoľko výhod:

- programátor nemusí implementovať metódu pre ukladanie nastavení,
- o ukladanie nastavení sa stará host, ktorý by mal zaručiť platformovo nezávislý spôsob uloženia,
- host má väčšie možnosti manipulácie s nastaveniami.

Z týchto dôvodov som sa rozhodol využiť tento spôsob perzistentného uloženia programov.

4.5 Návrh GUI

Pre grafické používateľské rozhranie som si stanovil nasledovné požiadavky:

Prehľadnosť – na prvý pohľad musí byť jasné, ktorá časť grafického rozhrania na čo slúži. Komponenty by mali byť veľmi rýchlo rozoznateľné, zreteľne označené názvom. Ovládacie prvky musia byť rozoznateľné od statických grafických prvkov. Vlnové priebehy, obálky a filtre môžu byť pre väčšiu prehľadnosť graficky interpretované.

Estetickosť – návrh musí rešpektovať pravidlá estetickosti, využívať harmonické farby a pôsobiť príjemne.

Jednoznačnosť – podoba komponentov, ich rozmiestnenie a označenie musia jednoznačne identifikovať ich funkciu.

Ovládateľnosť – parametre musia byť jednoducho ovládateľné, musí byť viditeľná ich aktuálna hodnota.

Pre návrh som zvolil ako základnú farbu bledomodrú a použil rôzne farby pre rôzne komponenty. Spojité parametre sú navrhnuté na spôsob potenciometrov. Každý je zreteľne označený a každý má zobrazenú aktuálnu hodnotu na príslušnom displeji pod potenciometrom. Nespojité parametre sú interpretované formou grafických alebo textových tlačidiel. Priebehy oscilátorov, nízkofrekvenčných oscilátorov, frekvenčné odozvy filtrov sú názorne zobrazené na tlačidlách zmeny týchto parametrov. Tvar obálky sa vykresľuje na displeji v reálnom čase zmenou parametrov. Modulačná tabuľka používa na výber zdrojov a cieľov vyrolovacie menu. Návrh je zobrazený na obrázku 4.7.

Obr. 4.7: Návrh GUI

5. Implementácia syntetizátora

Pre implementáciu syntetizátora som si vybral jazyk C++ a vývojové prostredie Microsoft Visual Studio 2005. Do projektu bolo potrebné pridať súbory frameworku VST SDK. Pri implementácii som rozložil funkcionality syntetizátora do rôznych tried, ale z dôvodu nezlučiteľnosti s požiadavkami na čo najmenšiu záťaž procesora som neaplikoval niektoré princípy objektovo orientovaného programovania.

Základ syntetizátora pre rozhranie VST je trieda zdedená z triedy AudioEffectX. V mojej implementácii sa táto trieda volá MirSynth. Prvá vec, ktorú bolo nutné implenmentovať, je globálna funkcia *createEffectInstance*, ktorá má za úlohu vytvoriť inštanciu syntetizátora a vrátiť smerník na ňu.

VST SDK vyžaduje implementáciu mnohých metód pre rôzne funkcie komunikácie s hostom. Sú to metódy pre komunikáciu ohľadom parametrov, systémových nastavení a programov.

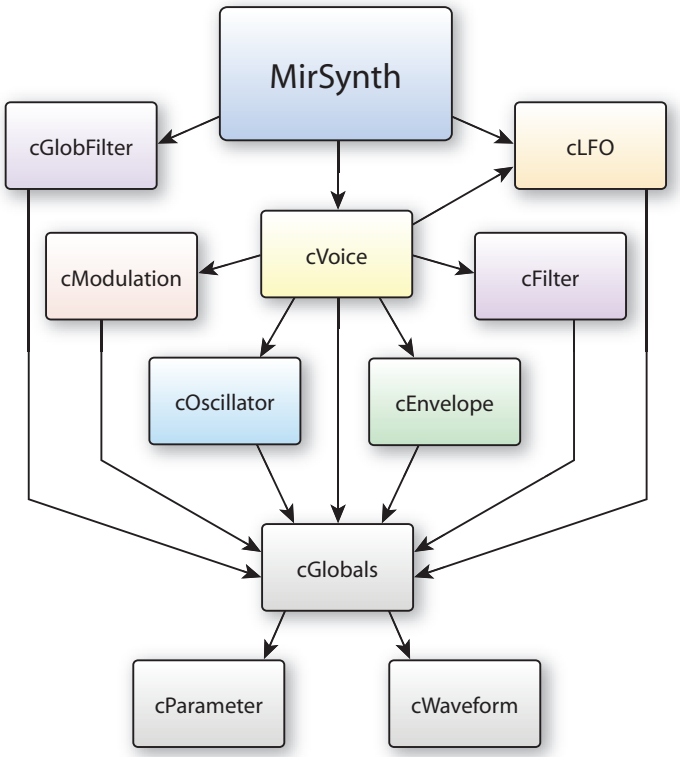
5.1 Popis tried

Jadro syntetizátora pozostáva z jedenástich tried.

MirSynth – je hlavná trieda zdedená z triedy AudioEffectX frameworku VST SDK. VST SDK vyžaduje implementáciu mnohých metód pre rôzne funkcie komunikácie s hostom. Sú to metódy pre komunikáciu ohľadom parametrov, systémových nastavení, programov a samotného spracovania udalostí a zvuku.

cVoice – predstavuje jeden hlas v polyfónii. Tento hlas obsahuje oscilátory, obálky, nízkofrekvenčné oscilátory, filtre a modulátor. Všetky operácie na úrovni hlasu sa vykonávajú v rámci tejto triedy.

cOscillator – predstavuje oscilátor.



Obr. 5.1: Triedy syntetizátoru

cFilter - predstavuje filter.

cGlobFilter – globálny filter – trieda MirSynth vlastní dve inštancie tohto filtra. Je určený na odfiltrovanie spodných nepočuteľných frekvencií. Má pevne nastavené parametre – hraničná frekvencia je 20 Hz, rezonancia je nulová. Tieto filtre sú dva v sérii kvôli väčšej strmosti filtrovania.

cEnvelope – predstavuje obálku.

cLFO – predstavuje nízkofrekvenčný oscilátor.

cModulation – predstavuje modulátor.

cGlobals – je trieda určená na sprostredkúvanie globálnych premenných ostatným triedam. Okrem iného vlastní aj inštanciu triedy cWaveform pre generovanie vĺn oscilátormi a nízkofrekvenčnými oscilátormi. Vlastní aj pole inštancií triedy cParameter pre uloženie aktuálnych parametrov.

cParameter – je trieda vytvorená na uloženie hodnoty a vlastností parametrov. Táto trieda poskytuje aj textové interpretácie vlastností parametrov.

cWaveform – trieda určená na prácu s vlnovými priebehmi. Má metódy na čítanie a interpolovanie vzoriek z tabuliek vĺn pre oscilátory aj pre nízkofrekvenčné oscilátory.

Pre generovanie tabuliek vĺn priebehov oscilátora som si vytvoril jednoduchú aplikáciu v jazyku C#, ktorá vygenerovala celú tabuľku pre konkrétny priebeh a uložila vlnu do binárneho súboru. Zmenou vzorcov som vypočítal aj tabuľky ostatných priebehov. Tie som následne pridal do projektu ako resources.

5.2 Popis významných operácií

5.2.1 Metóda *processReplacing*

Metóda *processReplacing* je virtuálna metóda určená na spracovanie zvuku. Keď host volá túto metódu, očakáva od syntetizátora zvukový výstup. Nasledujúci pseudokód opisuje fungovanie tejto metódy v mojej implementácii:

```
01 vynuluje sa buffer
02 interpolujú sa zmeny parametrov
03 vypočítajú sa hodnoty globálnych LFO
04 pre všetky hlasy polyfónie sa vykoná
        05 ak hlas hrá, vypočíta sa jeho vzorka a pripočíta sa k bufferu
06 buffer sa vyfiltruje globálnymi filtrami
07 buffer sa pošle na výstup
```

Pre prípad, že je veľkosť bloku väčšia ako 1 vzorka, tento kód sa vykonáva pre každú vzorku.

5.2.2 Výpočet vzorky hlasu

Výpočet vzorky hlasu prebieha v metóde *getSample* triedy *cVoice* nasledovne:

```
01 vynuluje sa buffer
02 vykonajú sa modulácie
03 ak hlas hrá, vykoná sa:
        04 vypočítajú sa hodnoty obálok
        05 vypočítajú sa hodnoty LFO
        06 posunú sa fázy oscilátorov
        07 vypočítajú sa vzorky oscilátorov a pripočítajú sa k bufferu
08 hodnota bufferu sa filtruje
09 buffer sa posiela na výstup
```

Posun fázy oscilátorov a výpočet ich hodnôt sú oddelené operácie, pretože kvôli synchronizácii je nutné najskôr vypočítať fázy všetkých štyroch oscilátorov, až potom počítať hodnoty vzoriek.

5.2.3 Spracovanie MIDI udalostí

Syntetizátor reaguje na MIDI udalosti nasledovne:

NOTE ON : Ak je v poli hlasov hlas, ktorý nehrá, spustí sa s parametrami udalosti (nota, velocity). Ak hrajú všetky hlasy, reštartuje sa s novými parametrami ten najstarší.

NOTE OFF : Ak je v poli hlasov hlas, ktorý hrá príslušnú notu, hlas sa zastaví.

CC : Zmena MIDI ovládača sa zaznamená do poľa hodnôt MIDI ovládačov.

AFTERTOUCH : Zmena hodnoty sa zaznamená do pamäte.

PITCH-WHEEL : Zmena hodnoty sa zaznamená do pamäte.

O spracovanie udalostí sa stará metóda *processEvents* triedy *MirSynth*.

5.2.4 Spustenie a zastavenie hlasu

Spustenie hlasu vykonáva metóda *playNote* triedy *cVoice*.

```
01 nastaví sa nota, velocity, poradové číslo noty
02 resetujú sa oscilátory, LFO, obálky a filtre
03 nastaví sa indikátor hrania noty na true
```

Zastavenie hlasu má na starosti metóda *stopNote* triedy *cVoice*. Tá uvedie obálky do fázy RELEASE. Po klesnutí všetkých hodnôt obálok na nulu sa nastaví indikátor hrania noty na hodnotu *false*.

5.3 Optimalizácia kódu

Výkon procesora je veľmi drahým zdrojom pri práci so zvukom. Aby bolo zaručené neprerušené generovanie a spracovanie zvukového signálu v reálnom čase, špička záťaže procesora by nikdy nemala vystúpiť na hodnotu 100%. Okrem toho, aj keby záťaž syntetizátora bola dosť nepatrná, aj tak je dôležité optimalizovať jeho kód, pretože pri práci s hudbou a zvukom sa zvykne používať veľké množstvo syntetizátorov a efektov súčasne.

V priebehu programovania syntetizátora som narazil na problémy s neprípustne vysokou záťažou procesora. Zmenou nastavení kompilátora, vypnutím všetkých

funkcií pre ladenie (debugging), zapnutím optimalizácií kompilátora a zapnutím podpory pre inštrukcie SSE2[1] som znížil zaťaženie procesora asi na desatinu pôvodnej hodnoty. Keďže staršie počítače nepodporujú inštrukčnú sadu SSE2, rozhodol som sa skompilovať dve nezávislé verzie, s podporou a bez podpory SSE2.

V syntetizátore prebieha mnoho operácií niekoľkostotisíckrát za sekundu. Napríklad pre vzorkovaciu frekvenciu 96 000 Hz sa počíta vzorka oscilátora pre každý stereo hlas 96 000 x 2 x 4 = 768 000-krát. Pri súčasnom hraní viacerých nôt, ktorých môže byť až desať, je to potom násobok tejto hodnoty. Z toho vyplýva, že aj ušetrenie niekoľko málo cyklov procesora v operácii môže výrazne ovplyvniť záťaž procesora.

Po dlhšom skúmaní náročností jednotlivých operácií [7] som v zdrojovom kóde identifikoval a optimalizoval nasledovné problémy:

Pretypovanie float na int – je vysoko neefektívna operácia, ak nie je použitá inštrukčná sada SSE2. Typický čas konverzie je 40 cyklov procesora. Pre túto konverziu som použil funkciu *lrintf*[2], ktorá je čisto assemblerová.

Neefektívne používanie typov – zbytočné miešanie typov float a double, použitie *double* pri násobení alebo delení, kde pre požadovanú presnosť postačuje typ *float*. Tento problém je potrebné riešiť logickým prehodnotením a preusporiadaním operácií.

Viacnásobné rovnaké výpočty – je veľmi neefektívne, ak sa pre každú vzorku počíta nejaká hodnota, ktorá sa mení len pri určitých udalostiach. Napríklad, ak nastane zmena určitého parametra a pre výpočty vzoriek používam jeho zložitý prepočet (napríklad mocnina), je potrebné si túto hodnotu hneď prepočítať a vo výpočtoch už pracovať s touto hodnotou.

Redundantné výpočty – sú výpočty, ktoré je možné matematickými úpravami zmeniť tak, že sa zmenší ich výpočtová zložitosť. Napríklad:

$$10^a \times 10^b \Rightarrow 10^{a+b}$$

[1] SSE2 je inštrukčná sada zavedená firmou *Intel* zameraná hlavne na efektívnejšiu floating-point aritmetiku. SSE2 podporujú procesory *Intel Pentium 4* a novšie a procesory *AMD Opteron* a novšie.

[2] Funkcia je opísaná v dokumente [7].

V uvedenom príklade sa namiesto dvoch mocnín a jedného násobenia vykonáva len jedna mocnina a jedno sčítanie[3]. Kompilátory sú často schopné zjednodušiť niektoré základné typy redundancie, ale pri mierne zložitejších výpočtoch je zjednodušovanie ponechané na programátora.

Delenie - delenie čísel je ďalšia neefektívna operácia. V prípadoch, keď je deliteľ konštantný pre viacero operácií, je efektívnejšie si predpočítať jeho obrátenú hodnotu, a tou v ďalších krokoch násobiť.

Príliš veľa rozhraní - prechody rozhraniami objektov a zapuzdrenie môžu znamenať výrazný pokles efektivity výpočtov. Ja som sa rozhodol pre prístup k atribútom objektov využívať priame smerníky.

Reťazenie závislosti za sebou nasledujúcich operácií – moderné procesory disponujú paralelným počítaním niektorých kombinácií inštrukcií. Táto funkcionalita sa nazýva *Out of order execution*. Aby ale tento princíp vykonávania kódu bol efektívny, je dôležité, aby po sebe nasledujúce výpočty neboli na sebe závislé.

Denormalizácia - denormalizované čísla sú floating-point čísla, ktoré sú tak blízko nuly, že na ich vyjadrenie nepostačuje bežný spôsob zápisu v pamäti. Tieto čísla používajú vlastný spôsob zápisu a operácie, kde je aspoň jeden operand denormalizovaný, sú extrémne pomalé, obzvlášť na procesoroch Intel. Všetky regeneratívne algoritmy sú náchylné k denormalizácii. V mojej implementácii IIR filtra spôsobuje spätná väzba náchylnosť k denormalizácii. Na ošetrenie náchylných hodnôt som použil metódu pripočítania a odpočítania veľmi malej konštanty, ktorá je ale dosť veľká na to, aby pri spočítaní s denormalizovanou hodnotou túto hodnotu pohltila. Po odpočítaní sa pôvodne denormalizovaná hodnota rovná nule. Pri normálnych hodnotách nenastane žiadna zmena. Pre túto konštantu som určil hodnotu 1×10^{-18}.

[3] Sčítanie je pre procesor oveľa jednoduchšia operácia ako násobenie.

5.4 Implementácia GUI

Grafické používateľské rozhranie je implementované pomocou multiplatformnej knižnice VSTGUI. Knižnica VSTGUI je súčasťou balíka VST SDK. Táto knižnica poskytuje základné triedy a komponenty na vytvorenie grafického rozhrania.

Použité triedy:

cMirEditor – samotné grafické rozhranie,

CAnimKnob – potenciometer parametra,

CParamDisplay – displej s hodnotou parametra,

cMultiStateButton – viacpolohové prepínacie tlačidlo,

cEnvDisplay – displej obálky,

cModAmtKnob – ovládač miery modulácie,

cModSelect – výberové menu pre zdroje a ciele modulácie,

cSyncButton – tlačidlo pre synchronizáciu oscilátora.

Na prípravu grafických materiálov som použil program Adobe Photoshop. Pre animáciu potenciometrov bolo nutné vytvoriť jeden obrázok, v ktorom boli vertikálne pospájané všetky možné polohy potenciometra. Pre bežné parametre to bolo 128 polôh a pre mieru modulácie 201 polôh. Ručné vytváranie týchto materiálov by bolo nesmierne náročné na čas. Ja som riešil tento problém automatizovaním Photoshopu použitím javaScriptu.

Druhý významný problém v priebehu implementácie bolo vytvorenie displeja pre obálky, a vykresľovanie ich aktuálneho tvaru. VSTGUI síce poskytuje základné funkcie pre kreslenie čiar, avšak nemá implementáciu antialiasovania čiar pre platformu Windows. Preto som musel implementovať funkcie pre antialiasované čiary sám. Na to som použil veľmi efektívny algoritmus *Wu*[4].

[4]Wu je algoritmus pre vykresľovanie antialiasovaných tvarov. Je nazvaný podľa svojho vynálezcu Xiaolina Wu. [10]

5.5 Testovanie

Testovanie syntetizátora som robil priebežne počas celej implementacie spolu s ďalším testerom. Na jeho procesore Intel Pentium 4 taktovanom na 3,7 GHz syntetizátor vykazoval nadmernú záťaž. Ja som testoval na procesore Intel Core2 Quad na 2,66 GHz a na AMD Athlon 2800 taktovanom na 1,6 GHz. Na týchto konfiguráciách bola záťaž procesora prijateľná, takže som nebol schopný overiť a identifikovať pôvod zvýšenej záťaže na procesore Pentium. Finálnu verziu syntetizátora som nechal testovať rovnako, ako boli testované syntetizátory v kapitole 3. Výsledky sú zhodnotené v nasledujúcej kapitole.

6. Zhodnotenie

V 3. kapitole testovali, komentovali a hodnotili syntetizátory štyria hudobníci, ktorí pracujú so syntetizátormi často. Z ich skúseností, tak pozitívnych, ako aj negatívnych, som sa snažil vychádzať pri návrhu a implementácii vlastného syntetizátora, vyvarovať sa chýb, ktoré syntetizátory mali, a inšpirovať sa tým, v čom boli zaujímavé a kvalitné.

Vzhľadom na to, že títo hudobníci sú z podobného prostredia a podobných štýlov a aj na to, že môj syntetizátor vychádza vo veľkej miere z ich skúseností a osobných preferencií, ich hodnotenia môjho syntetizátora nemožno pokladať za objektívne. Zároveň však cieľovou skupinou používateľov boli práve oni, takže ich spokojnosť budem považovať za meradlo úspešnosti tejto implementácie.

6.1 Hodnotenie

Hudobníci uviedli nasledovné silné stránky implementovaného sntetizátora.

- pekný, prehľadný, intuitívny dizajn,
- jednoduché ovládanie,
- kvalitné filtre,
- príjemná rezonancia, nepíska,
- možnosť zapojiť filtre sériovo/paralelne,
- veľká modulačná matica, prehľadná, veľa možností modulácií,
- dosť oscilátorov, obálok, LFO, filtrov,
- veľmi dobre spravené základné oscilátory, znejú dobre od najnižších po najvyššie oktávy,
- nepuká pri zmene parametrov,

- zvuk presný, žiadne známky nekvality.

Z týchto hodnotení vyplýva, že sa veľmi dobre vydarilo grafické používateľské prostredie. Hudobníci uviedli, že je estetické, prehľadné a intuitívne. Zo zvukových častí vyzdvihli kvalitu oscilátorov a aj kvalitu filtrov.

Medzi slabšími stránkami syntetizátora uvádzali hudobníci často absenciu určitých funkcionalít, ktoré nepatria medzi základné prvky subtraktívnej syntézy a ktorých implementácia je mimo rozsahu tejto práce. Boli to najmä PW, ring, frekvenčná modulácia, rôzne typy oscilátorov a filtrov a efekty. Beriem do úvahy tieto body ako inšpiráciu pre budúci vývoj, ale nepovažujem ich za nedostatky tejto práce.

Medzi ďalšimi slabými stránkami uvádzajú absenciu parametrov na celkové ladenie a celkovú hlasitosť, absenciu synchronizácie LFO a prepínača monofónie a polyfónie. Tieto chýbajúce prvky sú rozširujúce ovládacie prvky, ktoré priamo nezvyšujú kvalitu zvuku a ani neovplyvňujú množinu možností využitia syntetizátora, ale skôr robia ovládanie o niečo pohodlnejším. V prípadných budúcich verziách syntetizátora sa budem snažiť tieto prvky implementovať.

Posledné dve hudobníkmi uvedené slabé stránky boli veľká záťaž syntetizátora a obtiažnosť nastavovania parametrov presne na požadovanú hodnotu. Obidva tieto parametre sú závislé od konfigurácie používateľského počítača. Problematiku vysokej záťaže som stručne opísal pri testovaní v predchádzajúcej kapitole. Presnosť nastavovania parametra v grafickom prostredí je závislé od citlivosti vstupného zariadenia. Pri použití laserovej myši s rozlíšením 1600 dpi tento problém nebol pozorovaný. Používatelia s optickými myšami alebo touch-padmi museli mať s ovládaním väčšie problémy. Preto pre odstránenie takýchto nedostatkov je dôležité priebežné testovanie na viacerých rôznych konfiguráciách. Odstránenie týchto dvoch nedostatkov bude pri implementácii ďalšej verzie prioritné.

6.1.1 Bodové hodnotenie

Hudobníci obodovali syntetizátor bodmi 6, 7,5, 8,5 a 9. Priemerné hodnotenie teda je ∼7,8. Viacerí sa vyjadrili, že pridaním rozširujúcich funkcií by toto hodnotenie bolo vyššie.

Hodnotenie hudobníkov a teda aj celú implementáciu syntetizátora považujem za veľmi úspešné. Priemerné hodnotenie 7,8 zodpovedá presne hodnoteniu syntetizátora *SuperWave P8*, ktorý sa pri testovaní v tretej kapitole ukázal ako najlepší z dostupných freeware syntetizátorov. To ale neznamená, že mnou implementovaný syntetizátor by mal patriť medzi najlepšie syntetizátory vôbec, ale skôr že sa podľa hudobníkov kvalitou blíži ku komerčnej skupine syntetizátorov. Vo vyššie uvedených prípomienkach dostávam cenné rady ako syntetizátor ďalej skvalitňovať.

Záver

Úlohou tejto práce bolo navrhnúť a implementovať subtraktívny syntetizátor pre rozhranie VST v jazyku C++. Aby som mohol opísať jednotlivé postupy a problematiky riešenia tejto úlohy, v prvej kapitole som stručne opísal históriu syntetizátorov od prvých experimentov cez prvé komerčne úspešné analógové až po dnešné digitálne a softvérové syntetizátory. Priblížil som architektúru a princípy fungovania syntetizátorov používajúcich rôzne typy syntézy a aj ich vzájomnú komunikáciu a používateľskú interakciu prostredníctvom rozhrania MIDI.

V duhej kapitole sa venujem softvérovým syntetizátorom a rozhraniam pre ich použitie. Zameral som sa na rozhranie VST, priblížil som aj vývojový balík VST SDK určený pre vývoj syntetizátorov pre rozhranie VST. Stručne som spomenul aj iné spôsoby tvorby VST nástrojov.

V tretej kapitole som analyzoval súčasný stav voľne šíriteľných subtraktívnych VST syntetizátorov. Tu som vybral väčší počet serióznych produktov, a tie som dal testovať štyrom skúseným hudobníkom. Výsledky som vyhodnotil a vytvoril som zoznam dobrých vlastností, ktoré ma mali inšpirovať pri návrhu vlastného nástroja, a aj zoznam chýb a nedokonalostí, ktorým som sa mal v čo najväčšej miere vyhnúť.

Návrh nástroja je podrobne popísaný vo štvrtej kapitole. Tu som opísal návrh architektúry, návrh komponentov a aj riešenie problémov objavujúcich sa vo fáze návrhu. Pri návrhu nástroja som kládol dôraz najmä na kvalitu implementovaných častí a jednoduchosť používania.

Piata kapitola je venovaná implementácii syntetizátora. Okrem samotnej implementácie som podrobnejšie rozpísal problematiku optimalizácie kódu a navrhol som riešenia pre určité vzory neoptimálnych častí kódu.

V šiestej kapitole uvádzam hodnotenia implementovaného syntetizátora od hu-

dobníkov a vyhodnocujem úspešnosť implementácie vzhľadom na zadanie práce.

Vypracovanie tejto práce bolo pre mňa veľmi zaujímavé a prínosné. Veľmi ma teší pozitívne hodnotenie hudobníkov z komunity Corenforce, bez pomoci ktorých by výsledok implementácie určite nebol taký úspešný. Syntetizátor som nazval Miriyaki a po drobných úpravách sa ho chystám zverejniť ako voľne šíriteľný softvér. Dúfam, že tento syntetizátor nájde uplatnenie v hudbe viacerých amatérskych producentov.

Použitá literatúra

[1] de Schauensee, Maude: *Two Lyres from Ur*,
UPenn Museum of Archaeology, 1–16 (2002)

[2] www.obsolete.com: *Electronic Musical Instrument 1870 - 1990*,
http://www.obsolete.com/120_years/machines/telegraph (2005)

[3] Wikipedia, the free encyclopedia: *Roland TB-303*,
http://en.wikipedia.org/wiki/Roland_TB-303

[4] Bristow-Johnson, Robert: *Wavetable Synthesis 101, A Fundamental Perspective*,
http://www.musicdsp.org/files/Wavetable-101.pdf

[5] Wikipedia, the free encyclopedia: *Plugin*,
http://en.wikipedia.org/wiki/Plugin

[6] Tätilä, Veli-Pekka: *The Steinberg VST Plug-in Architecture*,
University of Oulu http://www.student.oulu.fi/∼vtatila/vst_architecture.html

[7] Fog, Agner: *Optimizing software in C++*,
Copenhagen University College of Engineering (2008)
http://www.agner.org/optimize/optimizing_cpp.pdf

[8] Bristow-Johnson, Robert: *Cookbook formulae for audio EQ biquad filter coefficients*,
http://www.musicdsp.org/files/Audio-EQ-Cookbook.txt

[9] Bizzetti, Fabio: *Sound Filtering For The Masses*,
http://membres.lycos.fr/amycoders/tutorials/maverick/sfftm.html (1999)

[10] Wikipedia, the free encyclopedia: *Xiaolin Wu's line algorithm*, http://en.wikipedia.org/wiki/Xiaolin_Wu's_line_algorithm

[11] Yvan Grabit: *VST 3rd party developer support*, http://ygrabit.steinberg.de/~ygrabit/public_html/index.html

[12] ndc Plugs: *Open source VST plugins for Windows and OSX*, http://www.niallmoody.com/ndcplugs/

[13] *The MIDI Specification*, http://www.borg.com/~jglatt/tech/midispec.htm

Príloha - CD

Obsah

- zadanie práce,
- text práce,
- binárne súbory syntetizátora,
- freeware hostiteľský program,
- zdrojové kódy syntetizátora,
- používateľská príručka.