Technická univerzita v Liberci
FAKULTA PEDAGOGICKÁ

Katedra: Inženýrské Informatiky
Studijní program: 2. stupeň Zš
Kombinace: Angličtina- Informatika

Elektronická ladička akustické kytary

Acoustic guitar tuner for PC

Diplomová práce: DP/KAI/5/7790/001

Autor:
Andrej Gulyáš

podpis:
Andrej Gulyáš

Adresa:
Dělnická 806
280 02, Kolín 2

Vedoucí práce: Mgr. David Kmoch
Konzultant: Ing. Jan Horák

Počet

stran	slov	obrázků	tabulek	pramenů	příloh
59	10449	20	-	7	1

V Liberci dne: 05. 1. 2005

UNIVERZITNÍ KNIHOVNA
TECHNICKÉ UNIVERZITY V LIBERCI
3146086086

TU v Liberci, FAKULTA PEDAGOGICKÁ
461 17 LIBEREC 1, Hálkova 6 Tel.: 048/535 2515 Fax: 048/535 2332

Katedra: ...inženýrské informatiky...

ZADÁNÍ DIPLOMOVÉ PRÁCE

(pro magisterský studijní program)

pro (diplomant) ...Andrej Gulyáš.....

adresa: ...Dělnická 806, Kolín 2, 28002.....

obor (kombinace): ...IF-AJ.....

Název DP: ...Elektronická ladička akustické kytary na PC .....

Název DP v angličtině: ...Accoustic guitar tuner for PC.....

Vedoucí práce: ...Mgr. David Kmoch.....

Konzultant: ...Ing. Jan Horák.....

Termín odevzdání: ...30. dubna 2004.....

Pozn. Podmínky pro zadání práce jsou k nahlédnutí na katedrách. Katedry rovněž formulují podrobnosti zadání. Zásady pro zpracování DP jsou k dispozici ve dvou verzích (stručné, resp. metodické pokyny) na katedrách a na Děkanátě Fakulty pedagogické TU v Liberci.

V Liberci dne 5.12.2003

.....
děkan

.....
vedoucí katedry

TECHNICKÁ UNIVERZITA V LIBERCI
Univerzitní knihovna
Voronežská 1329, Liberec 1
PSČ 461 17
V 33/06 M

+CD

KAI/IF
58 s., [1] s. příl
tab., grafy

Převzal (diplomant): Andrej Gulyáš

Datum: 18. 12. 2003

Podpis: Andrej Gulyáš

Cíl:

Vytvořit program v Borland Delphi, který bude snímat hrané tóny z elektrické kytary. Program bude umožňovat záznam frekvencí tónů. Dále bude schopen tyto frekvence graficky znázornit a porovnat s frekvencemi tónů uložených ve vnitřní databázi, která bude rovněž součástí programu. Uživatel by měl být schopen pomocí tohoto programu v jednoduchém uživatelském rozhraní naladit běžnou akustickou kytaru. Součástí práce bude teoreticky nastínit notový zápis hrané sekvence tónů.

Literatura:

SMETANA, Ctirad. Ozvučování. Státní nakladatelství Praha, 1987

DOWLING, A. P.- FFOWCS, WILLIAMS, J. E.. Sound and Sources of Sound. New York: John Willey, 1983

NĚMEČEK, Pavel. Hluk v technické praxi Díl 1. nakladatelství TU v Liberci, 1998

HASSAL, J. R.- ZAVERI, K.. Accoustic Noise Measurements. Naerum: Brukel and Kjaer, 1988

SEDLÁČEK, Jiří- SLABA, Jiří. Delphi v Kostce. Nakl: BEN, 1997

Elektronická ladička akustické kytary

Andrej Gulyáš DP-2005 Vedoucí: Mgr. David Kmoch

Anotace

Cílem diplomové práce bylo vytvořit aplikaci v systému MS Windows, ladičku pro akustické kytary. Vychází se z předpokladu, že digitální ladička bude dosahovat přesnějších a stabilnějších výsledků, nežli klasické analogové ladící přístroje, a že se v příštích letech budou ve větší míře používat hudební nástroje zapojené do počítače. V první části diplomové práce jsou prezentovány teorie a technologie, bez jejichž znalosti by nebylo možné program vytvořit. Ve druhé, praktičtější části jsou představeny konkrétní postupy a řešení problémů při tvorbě aplikace. Navrženo je většinou více řešení, ze kterých je na základě testování a následné úvahy vybráno to nejefektivnější. Na závěr diplomové práce je teoreticky stručně navržené řešení aplikace, nazvané „notový zápisník", která má zaznamenávat hrané melodie jako text (obdoba notového zápisu).

Acoustic guitar tuner for PC

Summary

The main objective of the diploma thesis was to create an aplication in the operating system MS Windows- a tuner for acoustic guitars. The key assumption was that the digital tuner would accomplish more exact and stable results than analogue tuners; and that the musicians would plug their instruments into computers more often in the future. In the first part of the diploma thesis, there are theories and technologies presented, which are necesary for developing the aplication. In the second part, there are specific problems' solutions of the aplication development presented. The most effective solution of the problem is chosen as a result of proper testing and considering. There is also a theoretic and short sheme of another aplication presented at the end- a notepad for writing down music in text format.

1 Obsah

2 Úvodní tón

Myšlenka vytvoření vlastního programu, pomocí kterého budu schopen naladit kytaru zapojenou do osobního počítače, je stará několik let, a vedlo mě k ní několik důvodů. Zaprvé to byla značná nedokonalost některých ladících přístrojů (pravda, přístrojů spíše nižší cenové kategorie), se kterými jsem měl možnost pracovat. Později, když jsem začal experimentovat s kompozicí hudby na osobním počítači, to bylo neustálé, nepraktické „přepichování" kabelu ze zvukové karty do ladičky a nazpět. S postupem času, když už bylo připojení na internet dostupnější, jsem konečně přišel do styku i s ladícími programy na PC, ovšem i ty většinou nebyly plně vyhovující. Vím tedy, že se nepouštím do něčeho zcela nového, přesto chci do své práce zahrnout to, co si myslím, že u předchozích pokusů chybělo, nebo co by dokonce zvýšilo výkon onoho zařízení.

Většina výrobců ladících přístrojů se soustředí na tzv. „hands free" ladění, to jest takové, při kterém má hudebník položen ladící přístroj kdesi poblíž a může se věnovat pouze vlastní manipulaci s hudebním nástrojem. Toto má za následek, že ladící přístroj se stává více interaktivním a reaguje sám na většinu podnětů- vlastně ho je potřeba jen zapnout a vypnout. Sen každého hudebníka? Nikoli. Přinejmenším začátečníci hry na kytaru budou chováním přístroje zmateni a nebude pro ně lehké ho používat. Cílem diplomové práce je tedy udělat program velice přehledný i pro začátečníka, který neví o hře na kytaru vůbec nic. Dopomoci by tomu nejvíce měly obrázky strun, aby hudebník opravdu věděl kterou strunu právě ladí a kde ji najde jak na obrazovce, tak na nástroji. Program tedy bude do jisté míry i rozšiřovat znalosti začátečníků.

Dalším vylepšením, které sice zhorší „hands free“ vlastnosti programu, zato zvýší možnost vlastního nastavení a přehlednost, bude možnost přednastavit si každou laděnou strunu na požadovaný tón. Tato funkce mi chyběla takřka u všech ladících přístrojů, ale občas se v nějaké podobné formě objeví, zvláště u dražších přístrojů. Možnost tohoto nastavení budou využívat hudebníci, kteří chtějí reprodukovat jiného umělce, jenž nepoužívá standardní ladění kytary, nebo pokud sami chtějí hrát v jiném ladění. Rovněž tato funkce bude, pro svou názornost, působit edukativně na uživatele.

Mým cílem je při vývoji programu vycházet ze svých vlastních zkušeností nejen s hrou, ale i s výukou hry na kytaru, a využít tyto zkušenosti co nejlépe k vytvoření co možná nejpřijatelnějšího uživatelského rozhraní.

Původně měl mít program další, mezi ladícími přístroji ojedinělou funkci, a to možnost ladit sedmi-strunné a osmi-strunné kytary. Uživatelů těchto nástrojů je ale stále příliš málo a ovládání programu by se zase zbytečně komplikovalo vzrůstajícím počtem tlačítek.

Nejdůležitější ze všech funkcí patrně zůstává primární funkce ladícího zařízení, a tou je výkonné snímání signálu z nástroje a jeho přesná analýza. Již mnoho nadšenců a odborníků přede mnou se snažilo výkon ladičky zvýšit co možná nejvíce, používaje při tom nejrůznějších metod a znalostí, např. digitální filtry, nebo rozklad snímané frekvence na základní, a vyšší harmonické frekvence. Ambice této diplomové práce nejsou tak vysoké, aby mohl program aspirovat na nejvýkonnější ladící přístroj, který kdy snímal zvuk světa, přesto se chci pokusit přijít na efektivní systém záznamu zvuku a jeho následné analýzy. Dosáhnout toho hodlám nejprve pokusy, měřeními a vysledováním nejefektivnějšího záznamu signálu. Potom chci použít algoritmus pro rozklad frekvence, čímž se odstraní ostatní zvukové vlny, které vcházejí do vstupu zvukové karty spolu se signálem. Dále bude zajímavé pozorovat všechny

vnější faktory, které značně ovlivňují výkon ladícího přístroje, jako jsou třeba: kvalita kytarových snímačů, nebo kvalita mikrofonu; druh strun (nylonové, nebo kovové), způsob rozvibrování struny (trsátkem, nebo bříškem prstu); hlasitost signálu a okolní šum; i konečně vlastní zabarvení zvuku nástroje, jenž je pro nástroj tak charakteristický, jako otisky prstů pro člověka.

Součástí diplomové práce je i teoreticky vysvětlit možné řešení notového zápisu odehrané sekvence tónů. Jedná se o jakýsi „zápisníček" pro hudebníka, jež by chtěl zaznamenat nějakou melodii. Tuto funkci se pravděpodobně nikomu nepodaří vypracovat tak, aby byla plně použitelná, neboť jeví se mi nyní mnohá úskalí, na kterých může ztroskotat. Nicméně, tato teorie je již velice blízká k používání hudebního nástroje jako vstupního zařízení, pomocí jehož by se dal ovládat nějaký systém, či program. Doufám tedy, že má práce bude prospěšná někomu, kdo se touto problematikou bude chtít zabývat.

Jako dozvuk „Úvodního tónu" se nabízí úvaha o smysluplnosti celé práce. Oponenti namítnou, že program má velice malou využitelnost v hodinách hudební výchovy i počítačů na základních školách. Pravda, jen málo žáků vůbec hraje na nějaký nástroj, takže program by se dal použít spíš pro zajímavost jako demonstrace využitelnosti PC. Dalším argumentem je fakt, že zatím jen málo kytarových hráčů používá svůj nástroj ve spojení s osobním počítačem.

Na tomto místě chci zdůraznit neustále se zvyšující počet funkcí počítače v domácnosti. Dnes už počítač, jeden přístroj, zastane funkce několika jiných přístrojů jako je videohra, televize, video, hi-fi věž, a do budoucna budou funkce stále přibývat. Domnívám se, že zanedlouho bude většina kytaristů považovat PC za součást hudebního nástroje k němu připojeného. Již delší dobu totiž existuje software (např. CoolEditPro, Logic, CatWalk), který

plnohodnotně nahradí skutečné nahrávací studio. Kompozice i reprodukce hudby v těchto „virtuálních studiích" má obrovské množství výhod a věřím, že takové studio bude povinnou výbavou i amatérských muzikantů. Kdyby jen existovalo takovéto studio v dobách pana Antonína Dvořáka, který při tvorbě neměl k dispozici celý orchestr, a nemohl si tedy poslechnout, jak bude konečný produkt znít... Nezbytnou součástí virtuálních nahrávacích studií bude samozřejmě i ladící přístroj pro strunné nástroje.

3 Teoretická část

V teoretické části diplomové práce jsou prezentovány obecné znalosti, teorie, metody a technologie, které je třeba míti na paměti při vývoji programu. Konkrétní praktická řešení jednotlivých problémů a výsledky testování ladičky jsou obsaženy ve druhé části diplomové práce.

3.1 Kytara, malý orchestr

Bližší seznámení s nejpopulárnějším a nejrozšířenějším strunným hudebním nástrojem je naprostou nezbytností. Když zjistíme jak tento nástroj funguje, bude hned jasnější jak dále postupovat při vývoji ladícího zařízení, které bude s nástrojem komunikovat.

obr.1 Akustická kytara

Kytara se skládá ze tří hlavních částí (viz. obrázek 1). *Tělo*, duté a většinou oválného tvaru, nejvíce ovlivňuje zvuk nástroje. *Krk* je podlouhlá část, vyrobená z tvrdého dřeva, spojující tělo s hlavou. *Hlava* slouží jako úchyt strun, pomocí kterého je možné měnit napětí strun a ladit tak nástroj. Běžná kytara má šest strun, ale může jich mít i více, třeba sedm, nebo dokonce

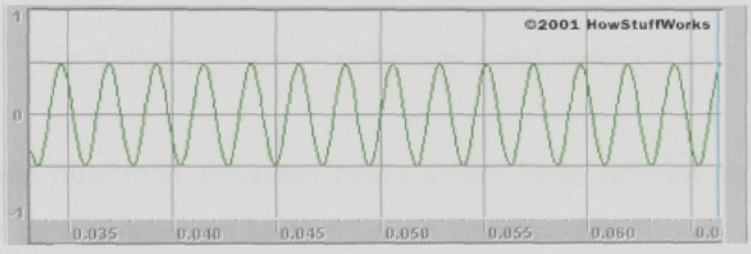
dvanáct. Zvuk na kytaře vzniká vibrací struny. Ta rozechvívá kobylku, jež je součástí mechanizmu pro úchyt strun na těle. Kobylka přenáší vibrace na celé, duté tělo, které funguje jako přírodní zesilovač. Vibrace struny pak slyšíme dostatečně hlasitě. U elektrické kytary se vibrace strun snímají pomocí snímačů, umístěných pod strunami. Snímače převedou zvuk na elektrický signál, který je potom pomocí elektrického zesilovače a vhodného hracího zařízení reprodukován zpět na zvuk.

Tón, který struna vydává, je ovlivněn její délkou, tloušťkou, napětím a pružností materiálu, z něhož je vyrobena. Délku struny může hudebník měnit tak, že ji stlačí na některém z pražců, umístěných na krku. Takto zkrácená struna potom vydává vyšší tón. Na obrázku 1 si můžeme povšimnout, že vzdálenosti mezi jednotlivými pražci nejsou vždy stejné. Vzdálenosti mezi pražci jsou přesně vypočítané, takže pokud si někdo chce vyrobit svůj vlastní nástroj, musí tyto rozměry znát a skutečně poctivě je při výrobě dodržovat- pokud má nástroj vydávat čisté tóny o předepsaných frekvencích. Tloušťku struny hudebník pochopitelně měnit nemůže, a právě proto má k dispozici strun šest, přičemž každá je jinak tlustá. Organizace strun je od nejhrubší k nejtenčí, směrem odshora dolů. Nejníže umístěná struna je tedy nejtenčí a vydává při standardním naladění tón E, o frekvenci 329,6 Hertz (jednotka Hertz udává počet oscilací za vteřinu). Nad E je struna H (v Americe je značena písmenem B), která má hrát zvuk o frekvenci 246,9 Hz. Následují struny G (196 Hz), D (146,8 Hz), A (110 Hz), a nejsvrchnější je opět E, ovšem o frekvenci 82,4 Hz. Toto uspořádání strun má velice praktický důvod- snadné, pro člověka fyzicky proveditelné vytváření akordů při stlačení více strun najednou.

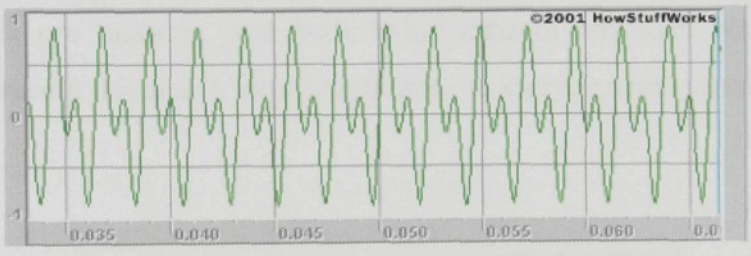
3.1.1 Zvuk kytary

Základní popis kytary je tedy za námi a nyní je čas na bližší průzkum zvuku tohoto nástroje. Proč různé hudební nástroje, vydávající tóny o stejné frekvenci, znějí každý jinak? Je to z toho důvodu, že např. kytara nevydává při

záchvěvu struny pouze *čistou frekvenci* tónu, nýbrž i *celočíselné násobky* této frekvence. Například při tónu A (440 Hz) přidává vibrující struna i zvuk o frekvenci 880 Hz, 1320 Hz, 1760 Hz atd. Těmto násobkům se říká *vyšší harmonické frekvence,* nebo *Fourierova řada.* Tělo kytary má tu vlastnost, že může některé frekvence zesilovat více a jiné méně- záleží na materiálu, z něhož je tělo vyrobeno; dále jsou to rozměry těla a jeho tvar. Proto je zvuk kytary, potažmo jiných hudebních nástrojů tak charakteristický a nezaměnitelný. Čistý tón A (440 Hz) na obrázku č.2 můžeme porovnat s tónem složeným z frekvencí 440 Hz a 880 Hz na obrázku č.3.



obr.2 Čistý tón A (440 Hz)

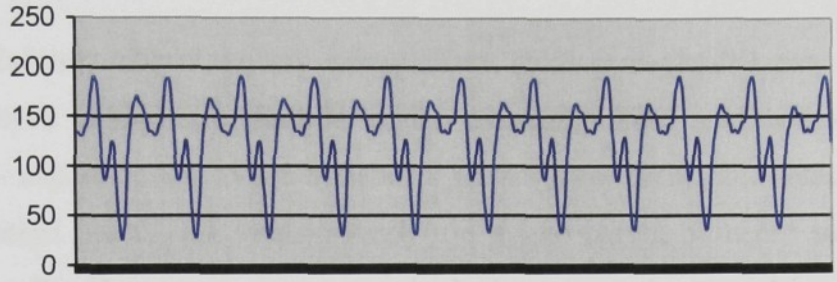


obr.3 Tón A složený z frekvencí 440 Hz a 880 Hz

Čisté tóny, jako je na obrázku 2, se v reálném světě moc nevyskytují, a vlastně ani nejsou nijak zajímavé. Křivka vlny na obrázku 3 má složitější průběh a je už o poznání blíže skutečnému zvuku kytary.

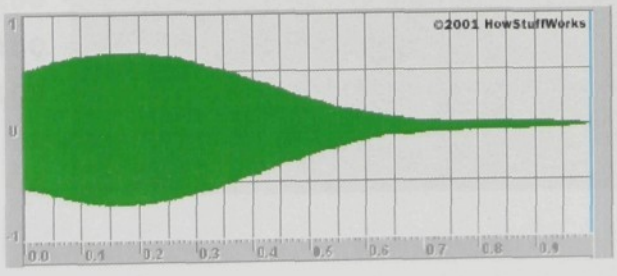
Zvuk vibrující struny je dále obohacen o záchvěvy ostatních strun. Jak již bylo řečeno, vibraci struny přejímá kobylka, která ale nerozechvívá pouze tělo, nýbrž logicky i ostatní struny, přestože je hudebník vůbec nepoužívá.

Výsledná frekvence tónu je tedy velice složitá, a díky tomu je zvuk kytary zajímavý oproti počítačem generovaným zvukům. Na obrázku 4 je zobrazen tón A (110 Hz), tak jak ho zahraje elektrická kytara.



obr.4 Složitější tón A zachycený z elektrické kytary

Dalším důležitým aspektem zvuku kytary bude viditelný na obrázku 5. Jedná se o celý průběh zvukové vlny od rozvibrování struny, až po úplné zklidnění. *Hlasitost* zahraného tónu se mění. Krátce po rozechvění struny je možné sledovat nárůst hlasitosti. Tento střední úsek vlny bude pravděpodobně nejkvalitnější pro analýzu, jenž bude ladící zařízení provádět. Naopak úplný začátek vibrace a doznívání, jsou krajní fáze, ve kterých nemá signál potřebnou kvalitu.

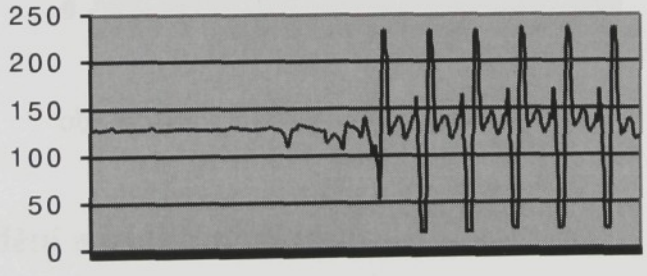


obr.5 Hlasitost rozechvěné struny

Doznívání, též zvané jako *uvolnění*, je nevhodné k přesné analýze signálu, protože v této fázi už není signál dostatečně hlasitý a navíc je již dosti

zkreslený (vhodnější slovo je asi „zakulacený"). Graf vlny v této fázi již nemá tak výrazně ostré body jako jsou třeba minimum a maximum, a vlna se pomalu zplošťuje. Tělo kytary navíc již dávno zvýraznilo ostatní a vyšší harmonické frekvence.

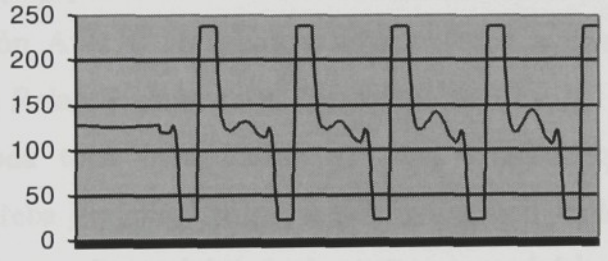
Začátek vibrace, zvaný *nástup tónu*, také není vhodný pro analýzu, jelikož bývá nejsložitější složkou celého tónu. Z pokusů s aditivní syntézou (stručně- napodobování zvuků hudebních nástrojů) se zjistilo, že nástup tónu je nejdůležitější pro to, jak vnímáme celý tón a jeho barvu. Skutečné tóny znějí „opravdově" díky tomu, že sinusové složky nejsou harmonické. A právě v nástupní fázi tónu jsou disharmonicity složek značné. Kromě toho záleží i na tom, jakým způsobem je struna rozvibrována (jakým předmětem, na jakém místě apod). Nástup tónu je tedy velice zajímavý a důležitý, nejen pro zvuk kytary, nicméně pro analýzu signálu není nejvhodnější. Na obrázku 6 je vidět složitý nástup tónu, při rozvibrování struny nějakým tuhým ostrým předmětem, jako je například trsátko, nebo nehet. Na obrázku 7 je vyobrazen nástup tónu, vzniklého rozechvěním struny bříškem prstu. Oba dva tóny jsou A (110 Hz), ale obrázky jsou v různých měřítkách.



obr.6 Struna rozvibrovaná trsátkem

Při podrobnějším průzkumu obou obrázků lze odhalit drobné rozdíly i v průběhu vlny, nejen v jejím začátku. Nejvíce viditelná je odlišnost v okamžiku těsně před dosažením globálního minima jednoho cyklu. U struny rozechvěné trsátkem můžeme sledovat ostřejší vrcholy, a celkově neklidnější průběh křivky grafu, s většími amplitudami. Ladící přístroj bude muset být uzpůsoben tak, aby tyto rozdíly nepovažoval za důležité.

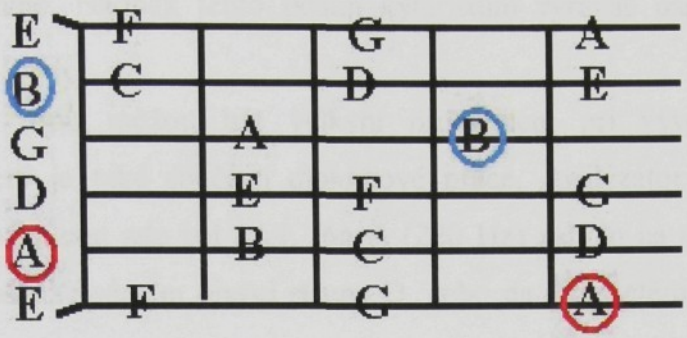
Z obou obrázků vyplývá, že způsob jakým hudebník rozechvívá struny, ovlivní zvuk jeho nástroje, a tak i celkový dojem z vystoupení. Je známo, že nejproslavenější kytaroví hráči kladli velký důraz na techniku rozechvívání strun. Nejlepším příkladem je Eric Clapton, světoznámý kytarista, který se proslavil neobvyklou technikou hraní bříšky prstů. Hráči basových kytar také raději používají bříška prstů, aby více zvýraznili basový zvuk nástroje. Sóloví kytaristé jsou zase velmi citliví při výběru vhodného trsátka.



obr.7 Struna rozvibrovaná bříškem prstu

3.1.2 Ladění a uspořádání tónů

Poslední částí kapitoly o kytaře bude teorie ladění a uspořádání tónů. Rozbor této teorie se neobejde bez obrázku 8. Na něm jsou vlevo označené struny na pomyslném nultém pražci. Nejhornější je vysoké E (329,6 Hz), jinak na kytaře umístěné úplně dole. Obrázek může být zpočátku matoucí, protože struny jsou na něm umístěny zrcadlově v porovnání se skutečným



obr.8 Uspořádání tónů

umístěním strun na kytaře. V této perspektivě se zobrazuje i tabulátorový notový zápis a akordy. Je to z toho důvodu, že když se hudebník podívá na nástroj při hraní, vidí vysoké E (329,6 Hz) nejdále od svých očí, jakoby nahoře. Na obrázku jsou na některých pražcích zobrazeny tóny, které struna hraje, pokud se stlačí právě na tomto pražci. Například pokud strunu nízké E (82,4 Hz; na obrázku dole) stlačíme na prvním pražci, změní se struna na tón F (87,3 Hz). Důležitý je pátý pražec. Pokud na něm stlačíme strunu nízké E (82,4 Hz), vydává kytara tón A (110 Hz), což je identický tón s prázdnou strunou A (rovněž 110 Hz). Relace je znázorněna červenými kroužky.

Tato shoda tónů bývá často využívána k ladění kytary. Pokud si kytarista naladí třeba jen jednu strunu, je potom schopen podle sluchu naladit i ostatní struny. Pro muzikanty bývá shoda tónů na sousedních strunách základní znalostí. Když si hráč uvědomí, že např. šestý pražec na struně nízké E (82,4 Hz) vydává stejný tón jako první pražec na sousední struně A (110 Hz), má najednou několik možností jak tóny hrát. Pozornému oku neunikne čtvrtý pražec na struně G (196 Hz), označený modře, který se vymyká teorii „pátých pražců“. Podle něj je totiž naladěná struna H (na obrázku označená jako B). Důvodem tohoto posunu ladění o jeden půl tón je snadnější skládání akordů. Začátečníkům dělá posun mezi strunami G a H potíže, zejména při hraní

notových stupnic. Naopak tento posun kytaristům výrazně usnadňuje hraní akordů.

Shody tónů můžou být velkým problémem při vývoji notového zápisníku, který je také součástí diplomové práce. Analyzátor frekvencí by totiž musel rozpoznat zda byl např. tón A (220 Hz) zahrán na druhém pražci struny G, nebo na sedmém pražci struny D, nebo na dvanáctém pražci struny A.

3.2 Hudba a zvuk ve Windows

O tom jak vzniká zvuk na hudebních nástrojích již bylo dosti řečeno v předchozí kapitole, takže nyní je čas se podívat na druhý konec mikrofonového kabelu. Tedy, lépe řečeno, je čas zjistit jakým způsobem je zvuk do počítače zaznamenán.

3.2.1 Základní hardware

Základním předpokladem k nahrání zvuku do počítače je hardwarové vybavení PC. Prvním prostředníkem mezi hudebním nástrojem a počítačem je již zmiňovaný *mikrofon*. Člověk vnímá zvuk jako změny tlaku vzduchu na ušní bubínky. Mikrofon může tyto změny zachytit a konvertovat je na elektrický signál. Při převodu na elektrický signál může být zvuk znázorněn jako vlna, znázorňující průběh vibrací v čase (Tato vlna má většinou tvar sinusové vlny, která má dva parametry- *amplitudu* a *frekvenci*. Amplitudu vnímáme jako hlasitost, frekvenci jako výšku tónu). Elektrický signál je potom z mikrofonu pomocí *mikrofonového kabelu* veden do vlnového audio-zařízení, běžně označovaného jako *zvuková karta*.

Zvuková karta je dnes již běžným doplňkem kupovaného počítače. Bývá integrovaná do základních desek, jako standardní vybavení. Kupující si samozřejmě může vybrat kvalitnější zvukovou kartu, kterou potom nainstaluje do vyhrazeného slotu na základní desce. Na trhu se běžně vyskytují zvukové karty, schopné nahrávat zvuk v 16-bitech na vzorkovací frekvenci až 44100 Hz (kvalita CD), a jsou velmi levné.

Vlnové zařízení funguje tak, že převádí vstup z mikrofonu nebo jiného analogového zařízení na sérii digitalizovaných vzorků, které se na disku

ukládají jako soubory s příponou WAV. Tento záznam je možné zpět konvertovat na analogový signál a přehrát je pomocí výstupního zařízení.

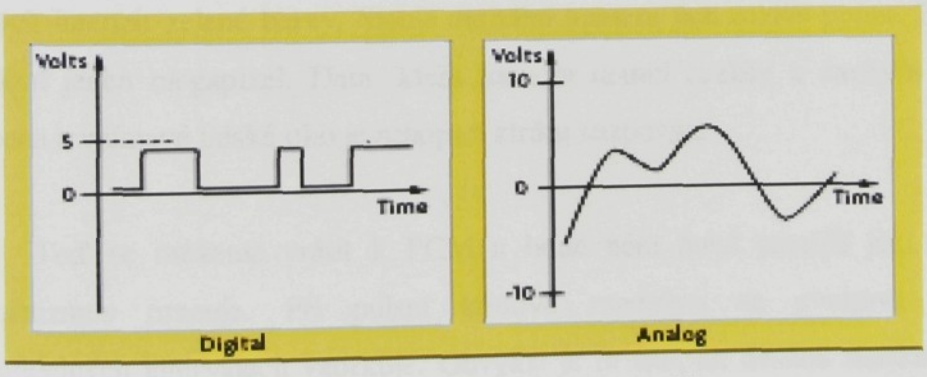
Dále zvuková karta často obsahuje MIDI zařízení. MIDI (Musical Instrument Digital Interface) je schopno hrát tóny na základě krátkých binárních zpráv. Kromě toho se ke kartě MIDI dá připojit vstupní zařízení, jako třeba klávesy.

3.2.2 Pulsní kódová modulace

Co se děje se signálem ve zvukové kartě? Protože počítače pracují s čísly, bylo nutné vyvinout mechanizmus pro převod zvuku na čísla a z čísel zpět na zvuk. *Pulsní kódová modulace* (PCM) je nejjednodušší metoda bez datové komprimace, kterou se převod dá provést. Používá se na kompaktních discích, digitálních audio-páskách a ve Windows.

3.2.2.1 Analogová a digitální hodnota

Než se blíže podíváme na to jak funguje PCM, je vhodné udělat nejprve malou odbočku a podrobněji definovat rozdíl mezi již několikrát zmiňovanými pojmy *analogický* a *digitální* signál. Digitální systém je takový, který používá diskrétní, nespojité hodnoty, měnící se "skokem".



obr.9 Digitální a analogový signál

Slovo *digitální* pochází z latinského *digit-* prst. Počítání na prstech je považováno za diskrétní, jelikož můžeme vyjádřit pouze celočíselné hodnoty (např. hodnoty 2 a 3, ale nic mezi nimi). Naopak analogový signál je takový, který průběžně, harmonicky a spojitě mění svou hodnotu. Na obrázku 9 jsou znázorněny digitální i analogový signál v čase. Zatímco u digitálního signálu dochází pouze ke dvěma hodnotám na ose *y* (Volts), nula a pět voltů, u analogového signálu funkce nabývá plynule všech hodnot od nula do pět.

Velká nevýhoda analogového systému, a tím pádem i nevýhoda klasické analogové ladičky hudebních nástrojů, je hluk. Při přenosu informací pomocí analogových metod, do signálu vždy vstupuje určité množství hluku. Příčin hluku je velké množství, např. data vysílaná radiem mohou být špatně přijímaná, nebo mohou být rušená jiným signálem, elektrické impulsy mohou být zeslabeny velkým odporem vodiče apod. Zatímco u digitálního přenosu též dochází k jeho degradaci kvality, tyto odchylky mohou být klidně ignorovány. U analogového přenosu mohou být data těmito odchylkami dosti deformovaná; naopak u digitálního přenosu se jakákoliv hodnota blížící se požadované hodnotě považuje za onu hodnotu.

Když je nutné převést analogový signál na digitální, aby mohl být například zpracován jinými digitálními systémy, může dojít ke ztrátě některých dat. Převodník z analogového na digitální signál je omezen rozlišením, ve kterém převod provádí. Například lidské oko je schopno vnímat desítky tisíc různých intenzit zelené barvy. Slabší digitální kamera jich rozliší pouze 256 v rozlišení jeden megapixel. Data, která kamera neumí rozlišit a zachytit jsou ztracena a vnímavé lidské oko je schopno ztrátu rozpoznat.

Teď se můžeme vrátit k PCM a bude nám hned jasnější jak tento mechanizmus pracuje. Při pulsní kódové modulaci se zvuková vlna v pravidelných intervalech vzorkuje. Obvykle je to několik desítek tisíckrát za vteřinu. V každém vzorku se změří *amplituda vlny*. Konverzi velikosti

amplitudy na číslo provádí hardwarové zařízení, nazývané jako *analogově-digitální převodník* (ADC). Podobně je možné zpět konvertovat čísla na elektrické signály pomocí *digitálně-analogového převodníku* (DAC). Při těchto převodech dochází ke zkreslení, takže signál vycházející z DAC není přesně takový jaký vcházel do ADC. Výsledná vlna má ostré hrany tvořené vysokofrekvenčními složkami. Proto se za DAC převodník obvykle zařazuje nízkofrekvenční propust, která odstraní vysoké frekvence a vyhladí tak hrany. Na vstupní straně bývá tato propust umístěna před převodníkem ADC.

Pulsní kódová modulace pracuje se dvěma parametry- se *vzorkovacím kmitočtem* a se *šířkou vzorku*. Vzorkovací kmitočet udává kolikrát za vteřinu se bude amplituda vlny měřit a šířka vzorku je počet bitů, v nichž se hodnota amplitudy bude ukládat. Čím je vyšší vzorkovací kmitočet a šířka vzorku, tím kvalitnější záznam zvuku dostaneme. Existuje ovšem úroveň, za kterou nemá smysl vzorkovací kmitočet zvyšovat a rozšiřovat vzorek, protože kvalita je potom za hranicemi schopnosti vnímání lidského ucha. Pokud ale bude vzorkovací kmitočet příliš nízký a šířka vzorku příliš malá, může při reprodukci hudby a zvuku docházet ke zkreslení.

Ladící zařízení nebude zvuky reprodukovat, takže zvuk nemusí být zaznamenán v té nejvyšší kvalitě. Správné nastavení vzorkovacího kmitočtu a šířky vzorku bude ovšem velice důležité, takže se nyní podrobněji s oběma aspekty nahrávání zvuku seznámíme.

3.2.3 Kmitočet vzorkování

Z toho co již bylo o vzorkovacím kmitočtu prozrazeno vyplývá, že vzorkovací frekvence omezuje nejvyšší frekvenci zvuku, kterou je možno zaznamenat a reprodukovat. Pokud je sinusová vlna vzorkována s příliš nízkým vzorkovacím kmitočtem, výsledná vlna bude mít nižší frekvenci než ta původní. Tento jev se označuje jako *alias*. Obecně platí, že vzorkovací frekvence musí být dvojnásobkem nejvyšší zaznamenávané frekvence.

Problémům s aliasy se předchází tím, že se na vstupu používá filtr, který ořezává všechny frekvence vyšší než polovina vzorkovacího kmitočtu. Stejný filtr se zařazuje za výstup digitálně-analogového převodníku, protože na výstupní straně jsou ostré hrany signálu způsobeny přítomností harmonických frekvencí s vyšším kmitočtem, než je dvojnásobek vzorkovacího.

Při digitálním záznamu na CD se používá vzorkovací frekvence 44100 vzorků za sekundu, tedy 44,1 kHz. Toto číslo vzniklo na základě několika podmínek. Člověk vnímá frekvence do 20 kHz, takže pro zachycení celého slyšitelného spektra je potřebný dvojnásobný vzorkovací kmitočet, 40 kHz. Toto číslo by dále mělo být asi o 10 % vyšší, protože frekvenční filtr má vždycky nějaké „náběhové" pásmo. Tím dostaneme 44 kHz. Často je ale digitální zvuk zaznamenáván společně s obrazem, takže vzorkovací kmitočet by měl být celočíselným násobkem snímkové frekvence televizního obrazu podle americké i evropské normy, což jsou kmitočty 30 Hz a 25 Hz. Společným násobkem se dostaneme na konečných 44,1 kHz.

Při vzorkovací frekvenci 44,1 kHz se dosáhne již velice kvalitního záznamu zvuku, s velkým množstvím dat. Ne vždycky je ale nutné záznam provádět v takovéto kvalitě, například pokud nahráváme jenom hlas, nebo jeden tón kytary. Obvyklé standardy vzorkování jsou proto 44,1 kHz, 22,05 kHz, 11,025 kHz, a 8 kHz.

Který ze standardních kmitočtů bude pro efektivitu ladícího zařízení nejvhodnější? Má-li být vzorkovací frekvence dvojnásobek nejvyššího možného tónu, postačila by frekvence 8 kHz, protože nejvyšší tón kytary je kolem 1000 Hz (záleží na počtu pražců, který se výrazně liší u různých druhů kytar). Nejvyšší tón, který bude zařízení zaznamenávat je dokonce ještě nižší, jelikož nejtenčí struna je tón vysoké E (329,6 Hz) a do vyšších tónů se kytara většinou neladí.

3.2.4 Šířka vzorku

Po vzorkovací frekvenci je šířka vzorku v bitech druhým parametrem pulsně-kódové modulace. Označuje se rovněž jako *dynamický rozsah* a jedná se o rozdíl mezi nejtišším a nejhlasitějším zaznamenávaným zvukem. Vnímání hlasitosti lidským uchem je, stejně jako vnímání frekvence, logaritmické. Intenzita zvuku se dá z vlny vypočítat druhou mocninou amplitudy. Jedná se vlastně o součet amplitud všech harmonických frekvencí. Rozdíl v intenzitě mezi dvěma zvuky se vyjadřuje pomocí jednotek 1 B (bell), což je desetinásobné zvýšení intenzity zvuku, a 1 dB (decibel) je desetina bellu po stejných násobných přírůstcích. Jeden dB znamená zvýšení intenzity 1,26 krát (10. odmocnina z 10), takže zvýšení amplitudy je 1,12 krát (20. odmocnina z 10). Dynamický rozsah mezi dvěma zvuky v decibelech se vypočítá podle vzorce:

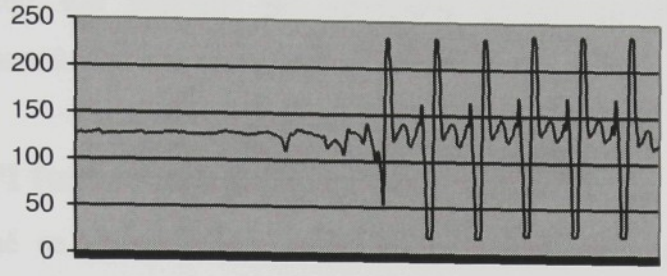
$$dB = 20 * log (A_1 / A_2)$$

Kde A_1 je amplituda prvního zvuku a A_2 je amplituda zvuku následujícího. Poměr mezi nejnižší a nejvyšší amplitudou je, při šířce vzorku 8 bitů, známých 256. Takže dynamický rozsah je 48 decibelů.

$$dB = 20 * log (256)$$

Charles Petzold ve své knize „Programování ve Windows“ uvádí, že rozdíl 48 dB odpovídá přibližně rozdílu mezi tichou místností a zvukem silnější sekačky na trávu. Rozsah mezi prahem vnímání a prahem bolesti je u člověka asi 100 dB. Vzorky ukládané v 8 bitové šířce tedy dosahují hodnot od 0 do 256, přičemž hodnota 128 zde zastupuje hodnotu nula. Vzorky s hodnotou menší než 128 zastupují záporné hodnoty, vzorky s hodnotou vyšší jsou hodnoty kladné. Na obrázku 10 je vidět jak se vlna nejprve pohybuje okolo

hodnoty 128, to znamená, že záznam byl zpočátku pouze ticho, nebo slabý šum. Později dochází k amplitudám s hodnotami v rozmezí od 20 do 230.



obr10 Nástup tónu

Pokud rozsah vzorku zdvojnásobíme na 16 bitů, dostaneme:

$$dB = 20 * log\ (65536)$$

což je už 96 dB, to znamená přibližně rozsah mezi prahem bolesti a prahem vnímání. Pro záznam a reprodukci hudby se tato šířka vzorku považuje za optimální- používá se na CD. Při 16 bitovém záznamu se ticho ukládá jako řetězec nul, takže ostatní hodnoty již jsou chápány jako čísla se znaménkem.

Pokud je zvuk zaznamenáván se šířkou vzorku 8 bitů, získáme velikost prostoru potřebného k uložení nekomprimovaného zvukového záznamu tak, že délku záznamu v sekundách vynásobíme vzorkovacím kmitočtem. Každý vzorek má velikost 1 byte. Při 16 bitovém záznamu je nutné rezervovat dvojnásobně velký prostor. Když se ještě navíc jedná o stereofonní záznam, výsledek se zdvojnásobuje ještě jednou. Vzorky se potom ukládají střídavě, levý kanál s pravým. Nyní je již jasné proč má CD velikost nad 635 MB. Hodinový záznam v kvalitě CD- tedy vzorkovací kmitočet 44,1 kHz, stereo, 16 bitů vzorek- je právě 635 MB.

Šířka vzorku 8 bitů (48 dB) bude stačit pro záznam vlny, kterou bude ladící zařízení analyzovat. Nepředpokládá se, že rozdíl v hlasitosti jednoho zahraného tónu nebude možné bezpečně vyjádřit rozsahem od 0 do 256. Na obr 10 je zobrazen tón zahraný při maximální možné nastavené hlasitosti kytarových snímačů.

3.2.5 API funkce pro podporu multimédií

Konečně se dostáváme k tomu, jak si poradí se záznamem zvuku operační systém, konkrétně MS Windows.

Zkratka API (application program interface) je název označující soubor protokolů, procedur, funkcí a nástrojů pro tvorbu softwarových aplikací. Dobré API usnadňuje tvorbu programu, protože poskytuje pomyslné bloky, ze kterých je program složen. Programátor pouze uspořádává bloky. Většina operačních systémů, jako MS Windows, poskytuje své API, aby mohli programátoři psát aplikace konzistentní s operačním systémem. API dále zajišťuje, že všechny aplikace budou mít podobné rozhraní, čímž se usnadňuje používání aplikace pro samotné uživatele.

Podpora multimédií v MS Windows byla důležitým vývojovým krokem a objevila se poprvé v roce 1991 jako takzvaná Multimedia Extensions. O rok později byly uvedeny Windows 3.1, které podporu multimédií řešily jako další kategorii volání funkcí API. Práce s multimédii je v podstatě otázkou toho, jak dokáží API funkce, nezávislé na zařízeních, pracovat s různými hardwarovými zařízeními.

API funkce jsou rozděleny do dvou kategorií. Označují se jako rozhraní *vyšší* a *nižší* úrovně. V obou úrovních se nacházejí funkce pro záznam a přehrávání zvuku, je však mezi nimi rozdíl, takže programátor si musí dobře rozmyslet jakým způsobem chce záznam provádět.

Nízko-úrovňové funkce pro záznam a reprodukci vlnového zvukového signálu začínají prefixy *waveIn* a *waveOut*. Tyto funkce se nazývají nízko-úrovňové proto, že program, ve kterém jsou použity, obsluhuje většinu záznamového a reprodukčního procesu. To znamená, že sice program je zaneprázdněn komunikací s hardwarovým zařízením, ale jako výhodu získá větší kontrolu nad záznamem a možnost zpracovávat data již v průběhu nahrávání. Například pokud chceme přehrát soubor WAV, musíme nejprve nahrát část dat z tohoto souboru do bloku paměti (*buffer*), který předáme nízko-úrovňové funkci pro výstup. Ta potom pošle balík dat do zvukové karty. Pro plynulé přehrávání je nutné vytvořit buffery dva, jeden bude naplňován daty, zatímco z druhého již budou čerpána data pro přehrávání. Jelikož si sám program dávkuje buffery, může v průběhu přehrávání data filtrovat, mixovat, nebo ovlivňovat tempo přehrávání. Při záznamu může s daty zaznamenanými do bufferu provádět jakékoliv operace ještě v průběhu nahrávání.

Vysoko-úrovňové rozhraní, nazývané MCI (Media Control Interface) odlehčí programu, neboť sám operační systém Windows se postará o přehrání či záznam zvuku. Aplikace tak ztrácí větší kontrolu nad daty, protože je mu dovoleno pouze proces odstartovat, zastavit, použít pauzu, nebo přetáčet data. Ovládání je velice podobné tomu, jakým si například přehráváme stopy z CD na hi-fi věži. Pokud chceme měnit přehrávaná data, musíme nejdřív soubor načíst, potom změnit a uložit. Teprve potom může být soubor znovu otevřen pro přehrávání.

Práce s MCI je jednodušší, na druhou stranu není MCI tak pružné jako nízko-úrovňové funkce *waveIn* a *waveOut*. Programátor si tedy musí vybrat mezi univerzálností a jednoduchostí. Pro ladící zařízení bude vhodnější používat komplikovanější nízko-úrovňové funkce. Aby na odezvu ladícího zařízení nečekal uživatel příliš dlouho, bude program zpracovávat každý buffer o velikosti zhruba 4000 bytů. Tak se zajistí, že zařízení bude téměř okamžitě

reagovat na vibraci struny. Kdyby byl buffer větší, může se zvětšovat prodleva mezi ukončením nahrávaní uživatelem, a naplněním celého bufferu.

3.3 Borland Delphi

Vývojový prostředek Delphi je zde pomyslnou cílovou stanicí zvuku, linoucího se přes všechna výše zmíněná zařízení. Aplikace "kytarová ladička" bude postavena právě v tomto vývojovém prostředí, takže se sluší o něm říci několik pochvalných slov.

Delphi je vývojovým nástrojem, jehož první verze byla vypuštěna firmou Borland v roce 1994. Dnes patří Delphi k nejpoužívanějším programovacím prostředkům. Přestože je založeno na jazyce Pascal, jenž je sám již poněkud za zenitem, jeho popularita snad stále roste.

Delphi v sobě spojuje silnou sadu *vizuálních* nástrojů pro tvorbu všech částí aplikace s výkonným *kompilátorem*. Architektura Delphi je založená na *objektově* orientovaném modelu s *komponentami*. Každá aplikace je vytvářena z těchto komponent, které můžeme považovat za základní stavební kameny. Komponenty jsou reprezentovány *objekty*, jenž vytvářejí ucelený hierarchicky koncipovaný aplikační rámec- Visual Component Library (VCL). Jako příklad komponenty budiž uvedena jedna z nejpoužívanějších, *Button* (tlačítko) nebo třeba *Scroll-bar* (posuvnítko). VCL je vlastně objektově orientovaná knihovna, na které je postavené i samotné Delphi. Dá se tedy říci, že Delphi je vytvořeno *samo sebou*. Architektura VCL není uzavřená, takže se do ní mohou komponenty dále přidávat, nebo si uživatel může naprogramovat své vlastní.

K manipulaci a propojení komponent Delphi nabízí *Object Inspector*. V něm uživatel definuje akce a nastavuje vlastnosti komponent. Díky *Object Inspectoru* se objekty ve smyslu OOP (objektově orientovaného programování) stávají *reálnými* objekty na pracovní ploše a je s nimi jako s takovými nakládáno (tzn. může je přemisťovat myší apod.). Kromě toho, že uživatel manipuluje s fyzickou reprezentací abstraktních objektů může libovolně přecházet mezi vizuálním návrhem a příslušným generovaným kódem. Toto je

princip *dvoucestných nástrojů* (two-ways tools), který transparentně zajišťuje okamžitou synchronizaci mezi provedenými změnami ve vizuálním vývojovém prostředí a kódem. Například, pokud přidáme novou komponentu do formuláře, objeví se v kódu příslušná deklarace jejího použití. Dvoucestné nástroje mají i *didaktický význam*, jelikož uživatel může snadno sledovat celý proces vytváření aplikace. Delphi je tedy z tohoto pohledu plnohodnotným nástupcem Turbo Pascalu, který si rovněž zakládal na didaktičnosti. Z toho vyplývá, že Delphi je vhodné pro programátory neprofesionály, neboť tvorba aplikací je vizuálně názorná a jednoduchá. Zřejmá výhoda Delphi je hlavně v tom, že se operační systém sám stará o velkou část a nechává na starost programátorovi pouze algoritmizaci. Například obyčejné menu bychom v Turbo Pascalu programovali celý večer; v Delphi je několik typů menu na výběr ze seznamu komponent, stačí dvakrát kliknout myší.

Typickou vlastností programování v Delphi je *událostmi řízené programování*. *Událost* (event) je situace, kterou komponenta rozeznává a na kterou program může, nebo musí reagovat. Když uživatel provede nějakou činnost s komponentou, například na ni klikne, generuje tato komponenta odpovídající *událost*. Někdy jsou události také generovány operačním systémem, bez zjevného uživatelova přispění. Události se tedy rozdělují na *uživatelské* a *systémové*. Z technického hlediska jsou *události* Delphi vyvolávány poté, co dojde k obdržení *zprávy* (message) operačního systému, který detekuje určitou situaci. Události a jejich zpracování tvoří podstatnou část logiky aplikace v událostmi řízeném programování. Běžný program je vlastně sekvencí zpracování, která se případně větví podle voleb, které provedl uživatel. V událostmi řízeném programování rozhoduje o zpracování mnohem více uživatel.

3.4 Analýza signálu

Proces zpracování signálu (signal processing) má širokou oblast použití. Původně se používal pro kompresi signálu před jeho přenosem, dále při přenosu televizního, radiového nebo telefonního signálu, ale dnes se používá i při získávání *příznaků* pro rozpoznávání. Často se jedná o posloupnost úkonů, které ze vstupního signálu, jež má obecně analogový charakter (spojitý), vytváří *diskrétní příznaky*, které se mohou rychleji (komprese) nebo bezpečněji přenést- např. jako televizní signál. Po přenosu se z těchto *příznaků* opět může zrekonstruovat původní signál.

U vývoje programu ladícího zařízení začíná být problém analýzy signálu aktuální v okamžiku, kdy dokážeme příchozí signál z hudebního nástroje zaznamenat a digitalizovat. Tehdy máme k dispozici data a nabízí se několik způsobů jak s nimi naložit. Účelem analýzy je samozřejmě zjištění přesné frekvence signálu.

3.4.1 Měření délky vlny

Jedno z nejjednodušších, nikoliv nejúčinnějších, řešení analýzy je založeno na pouhé znalosti záznamu zvuku do PC, tedy- dá se vypracovat po přečtení předchozích kapitol o zvuku. Jelikož frekvence je vlastně počet oscilací v jednotce času, určující je délka vlny jednoho cyklu. Délka vlny se dá změřit, pokud známe vzorkovací frekvenci záznamu. Když vydělíme vzorkovací frekvenci frekvencí zvuku, získáme délku vlny dané frekvence, respektive získáme počet vzorků jednoho cyklu. Například, frekvence 110 Hz má mít při vzorkovací frekvenci 22 kHz délku vlny 200 vzorků (22000 / 110 = 200). Pokud tedy budeme zaznamenávat signál, jehož délka jedné vlny je přesně 200 vzorků, víme, že zaznamenávaný zvuk má frekvenci 110 Hz. Pro

analýzu signálu a získání frekvence by tedy mohlo postačit počítání délky cyklu vlny. Tento způsob analýzy je sice zajímavý a překvapivě jednoduchý, ale může být poněkud nespolehlivý. Nicméně některé ladičky jsou na tomto systému analýzy založeny. Jelikož tvar vlny se v průběhu oscilace struny velice mění, algoritmus založený na rozpoznání začátku a konce cyklu vlny může snadno selhat. Také při slabší intenzitě signálu je vlna více zploštěná a je náročnější na ní hledat např. přesný okamžik minima nebo maxima.

Druhé řešení analýzy je založené na přesnější a spolehlivější metodě- *digitální zpracování signálu*. Toto řešení, konkrétně *Fast Fourier Transformation* (FFT- rychlá Fourierova transformace), je vědecky ověřitelná metoda a mělo by se pomocí ní dosahovat mnohem přesnějšího a stabilnějšího výkonu ladícího zařízení.

3.4.2 Fourierova Transformace (Fourier Transformation)

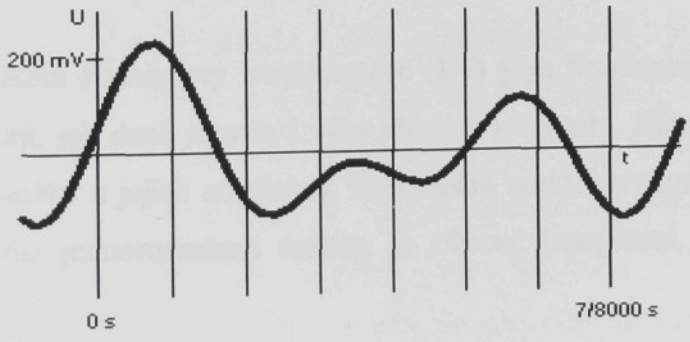
Obecně řečeno, *transformace* je vytváření *příznaků* vstupního signálu. Vhodně zvolenou transformací můžeme získat vhodné příznaky, které pak zjednoduší následnou klasifikaci obrazu transformace. Transformace se mohou i různě kombinovat. Vítaným efektem je redukce počtu příznaků. To byl také původní smysl transformačních kódování. Signál se před přenosem transformoval do výhodnější podoby z hlediska irelevantní a redundantní informace nebo citlivosti k přenosovým chybám. A po průchodu digitální sítí byl signál zpětnou transformací převeden na původní tvar. Podstata všech možných transformací spočívá v tom, že v *neznámém signálu hledáme známé průběhy*, (u Fourierovy analýzy jsou to sinusové křivky s různou frekvencí), čímž se výrazně redukují irelevantní a redundantní informace. Většinou se tak děje převodem z časové oblasti do oblasti *frekvenční*. Výsledkem transformace jsou pak tzv. *spektrální koeficienty transformace*- v podstatě se jedná o amplitudy jednotlivých známých průběhů, které v signále hledáme. Cílem je,

aby úplná informace o signálu byla obsažena v co nejmenším počtu těchto koeficientů.

Fourierova transformace, nám pomůže rozložit signál na základní frekvenci a další frekvence obsažené v signálu- získáme tedy Fourierovu řadu. Vychází se z předpokladu, že každý signál lze vyjádřit jako superpozici nekonečně mnoha sinusových signálů.

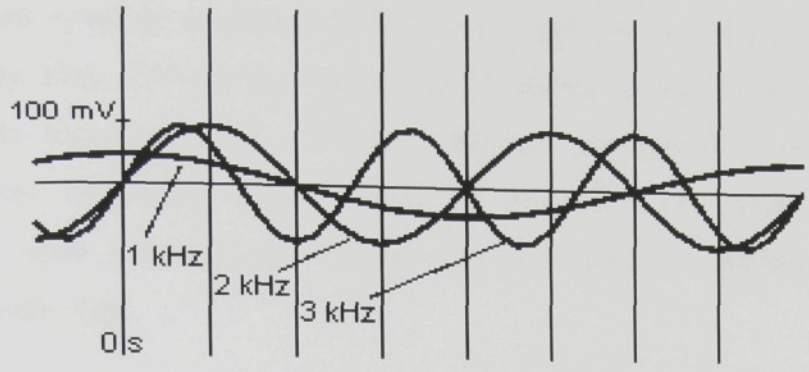
$$f(x) = \sum_{n=-\infty}^{\infty} F_n e^{inx},$$

Periodická funkce $f(x)$ je vyjádřena jako součet sinusových vln. F_n je (komplexní) amplituda signálu, e je základ přirozeného logaritmu a i je imaginární jednotka ($i^2 = -1$). Například, dostaneme signál, zobrazený na obrázku 11, tvořený osmi vzorky o hodnotách 50 mV, 206 mV, -100 mV, -65 mV, -50 mV, -6 mV, 100 mV, -135 mV. Vzorkovací frekvence je 8 kHz.



obr.11 Neznámý signál

Naším cílem je rozložit jej na sinusové vlny (obrázek 12), jejichž součtem se stane původní signál.



obr.12 Signál rozložený na sinusové vlny

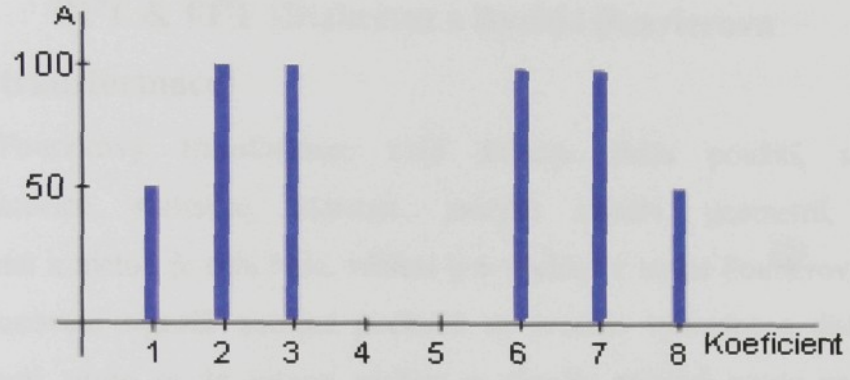
Dostáváme tři sinusové vlny. První z nich má frekvenci 1 kHz, amplitudu 50 mV a fázový posun o Pi/2 (nebo-li dvě osmitisíciny sekundy), protože v čase 0 je ve své největší amplitudě. Druhá vlna má frekvenci 2 kHz, amplitudu 100 mV a fázový posun 0. Poslední má frekvenci 3 kHz, amplitudu 100 mV a fázový posun rovněž 0. Původní signál S(t) je tedy složen takto:

$$\begin{aligned} S(t) = \; & 50mV \,.\, sin\,(2\,pi\,1000\,(t + 2/8000)) + \\ & 100mV \,.\, sin\,(2\,pi\,2000\,(t + 0\quad)) + \\ & 100mV \,.\, sin\,(2\,pi\,3000\,(t + 0\quad)) \end{aligned}$$

Výsledkem Fourierovy transformace (FT) jsou Fourierovy koeficienty, které nám říkají, jak daná sinusová vlna přispívá k signálu. Hlavním cílem FT je tedy najít složky a jejich amplitudy. Frekvenční spektrum signálu, zvané též Fourierův obraz jednorozměrné funkce, je obecně komplexní, takže ho lze rozepsat takto:

$$X(w) = R(w) + I(w)$$

kde *X(w)* je vektor komplexních spektrálních koeficientů, *R(w)* je jeho reálná část, která vyjadřuje amplitudy kosinů, *I(w)* je pak část imaginární a obsahuje amplitudy sinů. Provedením transformace získáme stejný počet koeficientů jako bylo vstupních vzorků, v tomto případě osm (obr 13). Obecně nám koeficienty nevyjadřují přímo amplitudy sinusových vln, proto se jejich hodnota musí *normalizovat*. To se provádí výpočtem absolutní hodnoty komplexního čísla.



obr.13 Normalizované frekvenční spektrum FT

Na obrázku 13 je zobrazen graf výsledku Fourierovy transformace, tedy frekvenční spektrum, normalizované. První index odpovídá frekvenci 1 kHz a amplitudě 50 mV, druhý frekvenci 2 kHz a amplitudě 100 mV, atd. Z grafu vyplývá, že získané koeficienty jsou symetrické vůči středu. Je to způsobeno tím, že konkrétními diskrétními hodnotami lze proložit vždy dvě sinusové vlny, jejichž frekvence jsou symetrické. Je možné uchovávat a zobrazovat pouze polovinu frekvenčního spektra, protože druhá polovina je z první lehce odvoditelná. Algoritmus FT nalezl v signálu dle očekávání pouze tři frekvence. Příslušné koeficienty jsou nenulové, naproti tomu ostatní koeficienty jsou nulové. Pro případnou zpětnou rekonstrukci signálu je třeba si uchovat koeficienty čtyři. Tři z nich vyjadřují jakou amplitudu mají příslušné sinusové vlny a čtvrtý (koeficient s indexem nula) vyjadřuje střední hodnotu signálu.

Jedná se v podstatě o sinusovku s nulovou frekvencí, což je vodorovná přímka. Její nejlepší proložení je vedeno středem signálu, a vyjadřuje tak posun ve svislé ose. Na tomto příkladu není žádný posun, ale pokud by byl celý signál posunut o hodnotu 50 po ose *y* (amplituda), měl by nultý koeficient hodnotu 50.

3.4.3 DFT & FFT (Diskrétní a Rychlá Fourierova transformace)

Fourierovy transformace mají širokou škálu použití, např. v kombinatorice, statistice, akustice, analýze signálu, geometrii, optice. Algoritmů a metod je celá řada, většina jich vychází z teorie Fourierových řad. Při zpracování signálů pomocí počítačů se pracuje výhradně s diskrétními hodnotami, proto se do vstupu analýzy ze signálu používá pouze periodická část reprezentovaná počtem *k* diskrétních hodnot. Jednorozměrná *DFT* (diskrétní fourierova transformace) je definovaná tak, že pokud je *x(k)* posloupnost konečných reálných signálových hodnot pro $k \in \{0, 1, \ldots, N-1\}$, potom platí následující vztah:

$$f_j = \sum_{k=0}^{n-1} x_k e^{-\frac{2\pi i}{n} jk} \qquad j = 0, \ldots, n-1$$

kde *j* je index spektrálního koeficientu. Koeficient *k* je index vzorku signálu a reprezentuje v podstatě čas. Hodnota e je základ přirozeného logaritmu, i je imaginární jednotka ($i^2 = -1$) a π je Pi. Součtem periodických komplexních exponenciálů $e_k = e^{-(2\pi i/n)jk}$ lze vyjádřit periodickou sekvenci signálových

hodnot $x(k)$, jelikož těmto hodnotám odpovídající Fourierova řada koresponduje právě se součtem harmonicky příbuzných komplexních exponenciálů.

Nevýhoda DFT spočívá v tom, že pro výpočet N hodnot $X(k)$ Fourierovy transformace je třeba N^2 komplexních násobení a $N(N-1)$ sčítání, čili doba potřebná pro výpočet je přibližně úměrná N^2.

Pro praktické použití je diskrétní Fourierova transformace relativně výpočetně náročná. Rychlejší algoritmy sice byly známy již na počátku století, ale nenašlo se pro ně praktického využití. Dnes existuje těchto algoritmů celá řada- využívají periodičnosti a různých symetrií v signálu. Jedním z nich je i algoritmus *rychlé Fourierovy transformace* (FFT) podle Cooleyho-Tuckeyho (1965). Počet operací v tomto algoritmu byl snížen z (n^2) na pouhých ($n \log n$).

Cooleyho-Tuckeyho FFT algoritmus je založen na rekurzivním rozdělování DFT do dvou menších DFT o velikosti $n/2$ v každém kroku- proto je počet vstupních vzorků určen mocninou čísla dvě. Další omezení počtu vstupních vzorků vychází z faktu, že výsledné frekvenční spektrum FFT je symetrické vůči středu (obrázek 13). Z toho vyplývá, že pokud je vstupních vzorků N, může FFT zjistit frekvence pouze do $N/2$. Počet vzorků musí být dvojnásobně vyšší než maximální možná frekvence v signálu a musí být mocninou čísla dvě. Přesnost a citlivost FFT k drobným odchylkám ve frekvencích se zvyšuje s počtem vstupních vzorků. Pro přesné měření frekvence je tedy lepší použít velké množství dat, ovšem s ohledem na počet operací, jenž budou s daným množstvím dat provedeny.

4 Praktická Část

Hlavní náplní praktické části diplomové práce je navrhovat a zvažovat konkrétní řešení programu ladícího zařízení. Nejprve je nutné se zamyslet nad celou koncepcí aplikace, teprve potom je možné zvažovat konkrétní řešení jednotlivých částí programu. V obou případech je nutné mít na vědomí teoretické poznatky deklarované v předcházející části diplomové práce. V rozborech řešení jednotlivých problémů bude občas nutné doplnit konkrétní teoretické znalosti, pro které nebylo místo v teoretické části.

4.1 Základní funkce

Činnost kytarové ladičky lze rozdělit do tří fází. Jejich správné propojení a součinnost výrazně ovlivní ovládání a výkon ladícího zařízení.

1. *záznam zvuku*
2. *analýza signálu*
3. *výstup- zpětná vazba uživateli*

V první fázi jde o to zaznamenávat signál co nejvhodnějším způsobem v přijatelné podobě. Ve druhé fázi bude ladící zařízení pracovat s uloženými daty a vznese z nich nějaký závěr. Nakonec je potřeba tento výsledek zobrazit zpět uživateli na obrazovku.

4.2 Synchronizace

Synchronizace součinnosti všech tří funkcí ladičky (záznam, analýza, výstup dat) je problém, jehož řešení je vhodné přizpůsobit nejvíce metodě záznamu.

Jedno z možných řešení synchronizace je takové, že by uživatel dostal časově vymezený prostor pro záznam dat, a analýza s následným výstupem by proběhla až po ukončení záznamu signálu, tzn. až po získání kompletní sady dat. Tento způsob synchronizace by sice přinesl nějaké výhody, například že by ladička uživateli zobrazila až konečnou analýzu (a ne pouze mezihodnoty), jenže nevýhodou se stává nepružnost záznamu a mála interaktivnost ladícího zařízení. Uživatel by příliš dlouho musel čekat na zobrazení výsledku.

Vzhledem k tomu, že právě interaktivnost ladičky a pružnost záznamu jsou vysoce žádané vlastnosti, všechny fáze musí probíhat takovým způsobem, aby doba mezi záznamem signálu a jeho výstupem na obrazovku byla co nejkratší. Toho se docílí tak, že analýza signálu a výstup na obrazovku bude probíhat v podstatě zároveň se záznamem. Tedy tak se to bude jevit uživateli. Ve skutečnosti se hned po zaplnění jednoho *bufferu* předají data k vyhodnocení. Vzápětí se začne zaplňovat další blok paměti a analýza dat z předchozího bufferu se bude zobrazovat na monitor. Velikost bufferu nesmí být příliš malá, aby nebyl program zahlcen manipulací s nadměrně velkým množstvím datových bloků, a také aby jeden buffer obsahoval dostatečně velké množství dat pro analýzu. Pokud by byl buffer naopak příliš velký, bude se snižovat pružnost záznamu a interaktivnost ladícího zařízení.

4.3 Záznam signálu- Jak na to?

Jedním z hlavních problémů je vlastní provádění záznamu. V MS Windows máme prakticky dvě možnosti jak záznam provádět (v rámci nízkoúrovňových API). Buď vymezíme konkrétní kapacitu, kterou chceme celou zaplnit daty (např. 10 sekundový záznam), nebo použijeme tzv. metodu *bufferování*. V prvním případě se jedná o jednodušší způsob záznamu, ale jeho velká nevýhoda je, že není "pružný". Zaznamenat totiž musíme vždy oněch předvolených deset sekund i když zrovna potřebujeme pouze pět. Druhý případ

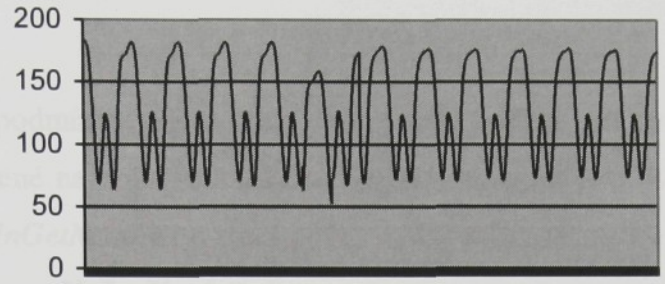
(bufferování) je celkově náročnější, jako výhody ale získáme možnost kdykoliv zastavit záznam, snazší přístup k datům a snazší synchronizaci všech tří funkcí ladičky.

4.3.1 "Bufferování"

Buffer je blok paměti vymezený na určité množství dat. Obvyklá optimální velikost tohoto bloku je 3-4 kb, avšak po testování metody analýzy signálu (FFT) vyšlo najevo, že buffer musí být minimálně dvakrát tak velký, aby byla analýza přesná. Ladící zařízení tedy používá bloky paměti o velikosti 8192 bytů.

Záznam pomocí paměťových bloků probíhá tak, že nízko-úrovňová API funkce vyšle jeden buffer na vstup zvukové karty. Když v tomto okamžiku začne záznam, začne se buffer zaplňovat daty. V případě že dojde k jeho zaplnění, musí se na zvukovou kartu vyslat další prázdný blok paměti a ten zaplněný se uvolní k manipulaci s daty. Takto záznam může probíhat teoreticky nekonečně dlouhou dobu, pokud se budou důsledně uvolňovat nepotřebné paměťové bloky.

Při zvukovém záznamu, který budeme chtít později co nejlépe reprodukovat se používá metoda tzv. *double-buffering*, tedy použití dvou (a více) bufferů najednou. Při použití pouze jednoho bufferu je totiž možná ztráta dat v okamžiku kdy se zaplněný buffer uvolňuje, odesílá zpět ke zpracování a na zvukovou kartu se posílá nově vytvořený blok paměti. Tato ztráta dat se projeví jako "trhlina" ve vlně signálu, která tím ztratí plynulý i periodický průběh. Na obrázku 10 je přibližně uprostřed grafu vidět trhlinu způsobenou únikem dat. Dvojité bufferování tento problém řeší tak, že na zvukovou kartu přidává záložní buffer, který začne být okamžitě naplňován daty v případě, že dojde k zaplnění prvního bufferu. Záložních bufferů může být i více.



obr.14 Ztráta plynulého periodického průběhu vlny

Záznam pomocí bloků paměti je složitější než záznam do vymezeného prostoru (v prostředí Borland Delphi), protože je nutné vytvořit tzv. *call-back-function* (funkci zpětného volání), která bude manipulovat se *zprávami* operačního systému. Když dojde k zaplnění celého bufferu, vyšle systém informační zprávu. V tomto okamžiku je možné na základě obdržené zprávy nastartovat odpovídající proceduru. Většinou je nutné zpracovat uložená data a přidat další buffer na zvukovou kartu, případně přidat záložní buffer do řady.

4.3.2 Záznam zvuku (API funkce)

V této kapitole je znázorněno, jakým způsobem je z nízko-úrovňových API funkcí poskládaná první fáze činnosti ladícího zařízení, tedy záznam zvuku. Pro záznam se používají API funkce s prefixem *WaveIn*. V následujícím seznamu jsou uvedeny API funkce použité v programu; seřazeny jsou podle prvního použití v programu.

WaveInGetNumDevs
WaveInOpen
WaveInPrepareHeader
WaveInAddBuffer
WaveInStart
WaveInUnPrepareHeader
WaveInReset

WaveInClose

První podmínka, která musí být nutně splněna při tvorbě jakékoliv aplikace založené na práci se zvukem, je přítomnost zvukové karty v počítači. Funkce *WaveInGetNumDevs* vrací počet audio-zařízení, na kterých je možné provádět záznam. Nejlepší místo kam tuto kontrolu vložit, je do fáze, kdy se program teprve spouští. Pokud má počítač více audio-zařízení, je vhodné uživatele upozornit, aby měl hudební nástroj zapojen do aktivního zařízení. Pokud naopak nemá přístroj zvukovou kartu ani jednu, program nemůže pokračovat.

Následuje nastavení *formátu dat*, ve kterém se budou zaznamenaná data ukládat. Rovněž je nutné zjistit, zda zvuková karta vůbec zvolený formát podporuje. Struktura WAVEFORMATEX definuje formát dat vlnového zvuku. Pro nastavení dat do požadovaného formátu je nutné v deklaraci vytvořit proměnnou typu WAVEFORMATEX a potom ji v tělu programu naplnit požadovanými údaji o formátu, jako je např. vzorkovací frekvence a šířka vzorku.

Nyní je nutné zjistit zda zvuková karta vůbec podporuje zvolený formát dat. Pokud ano, může se audio-zařízení nechat otevřít. Funkce *waveInOpen* otevírá audio-zařízení k záznamu a dá se použít i k dotazu na podporovaný formát. Zařízení otevřeme příkazem:

waveInOpen (HWaveIn, 0, @pFormat, form1.handle, 0,
CALLBACK_WINDOW);

kde v posledním parametru uvádíme, že chceme používat typ volání vzdálených funkcí nebo procedur (tzv. *call-back-window*), na základě zpráv posílaných z jádra systému. Pomocí těchto zpráv můžeme potom kontrolovat

záznam. Nejužitečnější zpráva je MM_WIM_DATA, která nastane pokud jsou data přítomna v bufferu a ten je odesílán zpět aplikaci. Na zprávu bude aplikace reagovat spuštěním další procedury, která zpracuje data z odesílaného bufferu a připraví nový blok paměti. Po zaplnění nového bufferu se opět vygeneruje zpráva a záznam tak může cyklicky pokračovat teoreticky nekonečně dlouho.

Pokud se úspěšně povedlo otevřít audio-zařízení, je nyní nutné poslat do něho blok paměti, na který se bude provádět záznam. Buffer se deklaruje jako *typ pole bytů* (array of byte) o velikosti 8192 bytů. Dále je nutné, aby program alokoval i strukturu WAVEHDR. Jedná se o hlavičku, která se používá k předávání bufferů API funkcím.

Dalším krokem je příprava, nebo-li aktivace hlavičky pomocí API funkce *waveInPrepareHeader*. Následně je na zvukové zařízení poslán první blok paměti za použití funkce *waveInAddBuffer*. Obě funkce se použijí vždy když je nutné na zvukové zařízení poslat nový prázdný blok paměti. Pomocí funkce *waveInAddBuffer* přidáme do zásoby další dva záložní buffery, aby se dosáhlo plynulého záznamu dat.

Záznam se spustí funkcí *waveInStart*. Data se začnou ukládat do bufferu na audio-zařízení. Po zaplnění bufferu přijde zpráva (message) MM_WIM_DATA. Na základě obdržení této zprávy se spustí procedura *MMInDone*.

Procedura *MMInDone* zaopatřuje příjem dat ze zaplněného bufferu, analýzu dat a přípravu nového bufferu. Pro získání dat je před vyprázdněním bufferu nejprve nutné použít funkci *waveInUnPrepareHeader*, která je jakýmsi komplementem funkce *waveInPrepareHeader*, a zruší se tak příprava hlavičky.

Potom již můžeme načítat data z *pole [i]* (array) a provádět s nimi různé operace. Nový buffer se vytvoří stejným způsobem jako ten první, tedy přípravou hlavičky a odesláním bloku paměti na audio-zařízení. Celý proces obdržování zpráv a spouštění procedury *MMInDone* pokračuje do té doby, než se změní hodnota kontrolní proměnné typu Boolean, takže již nedojde k zaslání nového bufferu na audio-zařízení. Změnu hodnoty provede uživatel ukončením záznamu stiskem příslušného tlačítka.

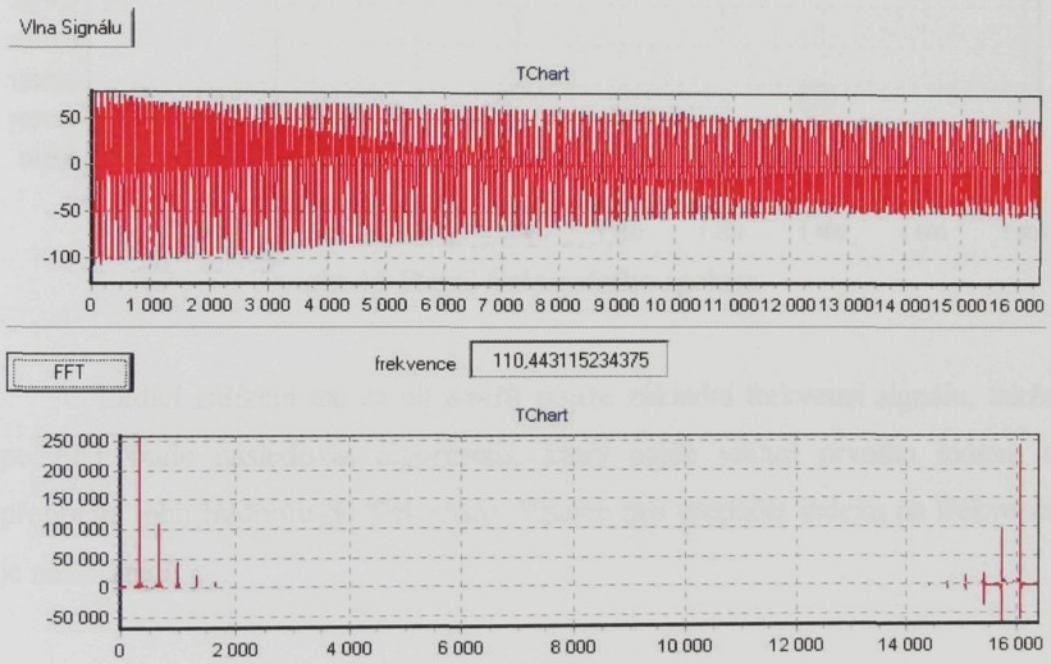
Zastavení záznamu proběhne pomocí funkce *WaveInReset.* Tato funkce dále způsobí, že se vynuluje obsah používaného bufferu a spolu s dalšími záložními buffery je vrácen aplikaci.

Poslední krok je uzavření audio-zařízení pro záznam- funkcí *WaveInClose*

4.4 Analýza signálu

4.4.1 Použití FFT

Data záznamu nyní mohou být použita v algoritmu FFT, který přetransformuje vzorky signálu na frekvenční spektrum daného úseku dat. Z frekvenčního spektra získáme přehled o tom, které frekvence jsou v signálu obsaženy a přispívají největší měrou k signálu. Nejpodstatnější informací ovšem bude index základní frekvence signálu. Následující obrázek 15 ukazuje výsledek algoritmu FFT, který zpracoval zvuk struny A (110 Hz).

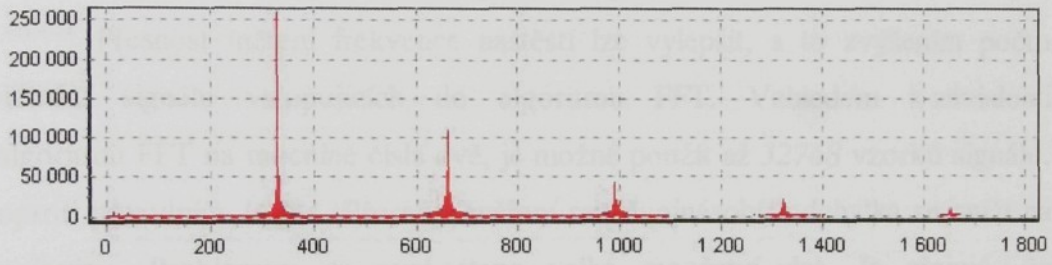


obr.15 Výsledek algoritmu FFT

Vrchní graf ukazuje vlnu signálu. Svislá osa zastupuje amplitudu frekvence, vodorovná reprezentuje počet vzorků (*16384*)- tedy v podstatě čas. Dolní graf zobrazuje na svislé ose intenzitu frekvence a na vodorovné ose indexy frekvenčního spektra. Indexy s nenulovou hodnotou reprezentují frekvence obsažené v signálu. Na grafu je patrná souměrnost podle svislé osy

vedené středem grafu, tedy indexem *i= 8192*. Levá strana je již normalizovaná, takže na ní vidíme jen kladné hodnoty. Pravá strana nevstupuje vůbec do výpočtu frekvence, takže je zbytečné s ní provádět operace a zpomalovat tak aplikaci.

Při detailním pohledu na použitelnou část frekvenčního spektra, obrázek 16, můžeme zjistit hodnotu prvního indexu, který dosáhne vysokých hodnot (*i=329*). Tento reprezentuje základní frekvenci signálu. Další indexy (*i= 660, 990, 1321, 1653*), jsou přibližně celočíselnými násobky prvního indexu a odpovídají vyšším harmonickým frekvencím.



obr.16 Detail frekvenčního spektra

Ladící zařízení má za cíl změřit pouze základní frekvenci signálu, takže po FFT bude následovat algoritmus, který najde vrchol prvního indexu a přepočítá jeho hodnotu do frekvence. Vzorec pro přepočet indexu na frekvenci je následující:

$$F = (i * SampleRate) / Psize / 2$$

kde *i* je index vzorku základní frekvence, *SampleRate* je vzorkovací frekvence záznamu (v tomto případě 11 kHz) a *Psize* je počet vzorků vstupujících do Fourierovy transformace (*16384*).

4.4.2 Přesnost FFT a výpočtu frekvence

Při pohledu na index (*i= 329*), reprezentující základní frekvenci 110 Hz, se nabízí úvaha o přesnosti měření a výpočtu frekvence. Logicky vyvodíme, že jeden indexový vzorek odpovídá přesně frekvenci 0,3 Hz. Například, pokud se frekvence signálu zvýší o 1 Hz, zvedne se index základní frekvence o tři. Vezmeme-li v potaz občasné odchylky při opakovaném měření stejné frekvence až o tři indexové vzorky, není takováto přesnost dostačující. Ladící zařízení musí mít schopnost zaznamenat i slabé odchylky od požadované frekvence struny. Struny rozladěné již o pouhé 2 Hz vydávají falešné tóny snadno rozpoznatelné vytrénovaným uchem.

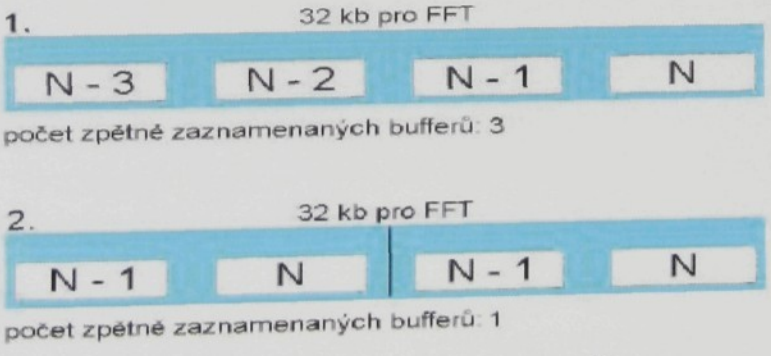
Přesnost měření frekvence naštěstí lze vylepšit, a to zvýšením počtu vzorků signálu vstupujících do algoritmu FFT. Vzhledem k závislosti algoritmu FFT na mocnině čísla dvě, je možné použít až *32768* vzorků signálu, oproti původním *16384*. Přesnost měření se zdvojnásobí, odchylka se sníží na polovinu. Problémem se nyní stane velké množství dat. Je zřejmé, že záznamový buffer nemůže být velký 32 kb. Při vzorkovací frekvenci 11 kHz by se jednalo o tři vteřiny dlouhý záznam, což je skutečně příliš. V tomto okamžiku vývoje programu je nutné vyzkoušet všechna možná nastavení záznamu a přesnosti analýzy- a vybrat z nich nejpřijatelnější řešení jako kompromis mezi dvěma vlastnostmi ladičky- což je přesnost měření frekvence a rychlost zobrazování aktuální frekvence.

4.4.3 Sladění záznamu s analýzou

Původně plánovaná velikost záznamového bufferu (*3-4 kb*) se ukazuje být nepoužitelná. Pro naplnění 32 kilobytů velkého balíku dat vstupujícího do FFT by muselo být použito osm takových bufferů. Analýza se tak zbytečně komplikuje. V určitých momentech dokonce dochází ke zhoršení přesnosti měření- a to tehdy když data v bufferech na sebe přesně nenavazují. Sedm trhlin ve vlně signálu dokáže výrazně ovlivnit výsledek FFT. V optimálním

řešení synchronizace záznamu a analýzy je nakonec použit buffer o velikosti 8192 bytů. Při vzorkovací frekvenci 11 kHz je potom reakční prodleva ladičky necelá jedna vteřina od okamžiku rozvibrování struny.

Pro zaplnění 32 kilobytů vstupujících do FFT potřebujeme data ze čtyř bufferů velikosti 8 kb. Zatím máme jeden. Jak zaplnit zbývajících 24 kilobytů? Nabízí se myšlenka použít uložená data z předcházejících bufferů. Pojmenujeme-li aktuálně zaplňovaný buffer písmenem *N*, předcházející buffery jsou potom *N-1*, *N-2* a *N-3*. Při dvojitém bufferování záznamu získáme plynulý záznam bez trhlin a dosáhneme velice přesného změření frekvence, která bude zobrazována v přijatelném zpoždění necelé jedné sekundy. Naneštěstí ani toto řešení, označené na obrázku 17 pořadovým číslem 1, není nejefektivnější. Nevýhodou je, že při začátku záznamu jsou buffery *N-1*, *N-2* a *N-3* ještě prázdné, takže FFT je přesná až zhruba po třech vteřinách. Dalším nedostatkem je, že FFT bere v potaz ještě data z bufferu *N-3*, i když dojde ke změně frekvence. Pokud dojde ke změně signálu, např. utažením nebo povolením struny na nástroji, do FFT bude vstupovat nová frekvence ještě se starými daty z bufferu *N-3*. Přesnou aktuální frekvenci se tedy uživatel doví opět až poté, co se novými údaji naplní všechny buffery, čili za tři vteřiny. Nejvíce si tohoto nedostatku lze povšimnout pokud rozvibrovanou strunu náhle utlumíme. Ukazatel frekvence začne postupně klesat, a klesá ještě dlouho po tom co již nástroj žádný zvuk nevydává.



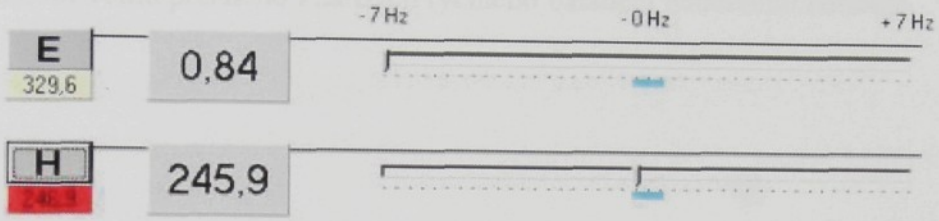
obr.17 Uspořádání bufferů pro analýzu

Optimální bude druhé řešení, ve kterém se zaznamenává pouze jeden předchozí buffer *N-1*. Ten se společně s *N* použije dvakrát tak, aby se využilo co nejvíce návaznosti začátku dat v *N* na končící data v *N-1* (obrázek 17). I toto řešení má nevýhodu, a to nenávaznost začátku dat v bufferu *N-1* na konec dat v *N*. Nenávaznost může mít za důsledek trhlinu v datech, která je vyznačena černou čarou přesně uprostřed 32 kilobytového bloku dat pro FFT. Podle výsledků měření a testů se došlo k závěru, že tato nevýhoda má ze všech předchozích metod nejmenší důsledek na přesnost výpočtu frekvence a jeví se tedy jako nejefektivnější. Pouhé jedno porušení periodičnosti a plynulosti vlny může mít odchylku maximálně jednoho až dvou indexů ve frekvenčním spektru, což je přijatelná odchylka maximálně do 0,2 Hz.

Plné přesnosti FFT je dosaženo po zaplnění pouhých dvou bufferů, oproti předchozím čtyřem. Pokud uživatel nechá strunu rozvibrovanou po dobu 1,5 vteřiny (jeden buffer je přibližně 0,75 vteřiny), teprve až potom je výsledek opravdu přesný bez zkreslení.

4.5 Zobrazení frekvence

Poslední funkcí ladičky je (po záznamu a analýze signálu) zobrazení výsledku měření. Smysl ladícího zařízení spočívá v porovnání aktuální frekvence struny s tónem, který má při správném naladění struna vydávat. Jedno z nejkomplexnějších a přitom nejjednodušších řešení pro ladičky je prezentováno na obrázku 18.



obr.18 Zobrazení frekvence

Toto je výřez obrazovky, kterou vidí v programu uživatel. Zde máme k dispozici pouze dvě spodní struny *E* a *H*- uživatel má možnost vidět všechny struny. Vlevo jsou tlačítka s označením tónů strun, jimiž uživatel volí, kterou strunu bude ladit. Pod každým tlačítkem je informační panel, zobrazující správnou frekvenci tónu v jednotkách Hertz. Struna H má informační panel zbarven červeně, jelikož právě na této struně probíhá měření frekvence. Na velkém šedém panelu vpravo od tlačítek je číselně zobrazována aktuální frekvence struny. Vpravo od něho je posuvnítko (*track-bar*), které graficky zobrazuje odchylku struny od správné frekvence tónu.

Posuvnítko má nastaveno počet hodnot na 42 a zobrazuje okolí správné frekvence tónu (hodnota 21), v rozmezí přibližně 15 Hz. Přesněji řečeno, okraje posuvnítka, hodnoty 0 a 42 lze považovat za následující nebo předcházející tón stupnice. V případě tónu *H* (na obrázku) jsou předcházející tón *Hm* a následující je *C*. Pro uživatele je posuvnítko orientačním ukazatelem, podle kterého provádí hrubé přiblížení k požadovanému tónu. Jakmile se

frekvence struny dostane do hodnot posuvnítka 20, 21 a 22, vyznačených modrým pruhem, může dále uživatel sledovat jen panel s číselným zobrazením frekvence, podle kterého provede konečnou, přesnou korekci utažení struny.

Většina běžných ladících přístrojů zobrazuje naladění strun pouze posuvnítkem, nebo podobným ukazatelem analogového typu, který ale neukáže přesnou hodnotu frekvence. Hudebník se pomocí takovéhoto přístroje pouze přiblíží k požadovanému tónu, ale nedosáhne nikdy úplně přesného naladění. Kombinací posuvnítka a digitálního ukazatele frekvence v číslech lze dosáhnout velmi přesného i zároveň rychlého naladění hudebního nástroje.

5 Notový zápisník

V současnosti má hudebník dvě možnosti jak zaznamenat hudbu. První je klasický zvukový záznam na příslušné médium. Druhá možnost je notový zápis prováděný ručně- ať už na papír, nebo do počítačového editoru. Návrh aplikace „notový zápisník" vychází z předpokladu, že když je možné počítačem zaznamenat hudbu jako vlnový zvuk, proč by nemohlo být možné ji zaznamenat jako noty, čísla, text? Účelem této kapitoly je shrnout poznatky z předchozích částí diplomové práce a pokusit se je aplikovat na problematiku zápisu kytarové hudby textově. Jelikož se jedná o poměrně složitou aplikaci, která by vydala na další diplomovou práci, bude úvaha o ní velice stručná a teoretická.

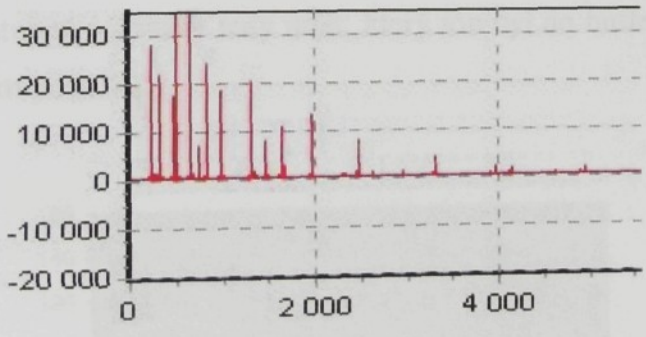
Nejprve je nutná malá odbočka k možným způsobům zápisu hudby. Klasický notový zápis je univerzální, takže z něho může hudbu reprodukovat jakýkoliv nástroj. Nevýhodou notového zápisu je, že opravdu jen hudebník odchovaný hudební školou je schopen ho prakticky používat. Proto byly vyvinuty přehlednější a jednodušší metody zápisu tónů pro jednotlivé nástroje. Speciálně pro kytaru byl objeven *tabulátorový zápis*, který schematicky naznačuje kde se daný tón na hmatníku nachází- jak je vidět na následujícím příkladu:

```
E |----------------2---------------------|
H |----------2-----------------------------|
G |------8-----8-------8---8-----8-----|
D |----6---6-----6-------6-----6---6--|
A |--4----------------------2----------|
E |-----------------------------------|
```

Počet řádků je šest, což odpovídá počtu strun. Horní řádek reprezentuje spodní strunu na kytaře. Čísla určují, na kterém pražci se tón hraje. Tóny se čtou zleva doprava. Prvním zahraným tónem je *Dm* na struně *A*, nacházející se na čtvrtém pražci (je označen červeně). Druhým tónem je *Am* nacházející se na struně *D*- šestý pražec (označen modře) atd. Z tohoto zápisu hudby není pochopitelně možné hrát jiným nástrojem, než kytarou. Tabulátorový zápis je doposud nejrozšířenější, jelikož poskytuje řadu výhod, a proto bude „notový zápisník" pro kytaristy hudbu zapisovat právě tímto způsobem.

5.1 Rozklad akordů

V teoretické části diplomové práce jsou prezentovány dvě možnosti analýzy signálu- Fourierova transformace a měření délky zvukové vlny. Algoritmus FFT je vhodný zejména pro přesnou analýzu signálu, která zjistí všechny frekvence obsažené v signálu. Tímto algoritmem je možné analyzovat hrané akordy (akord je souznění tří a více tónů) a zapisovat jejich tónové složení do tabulátoru. Následující obrázek 19 ukazuje frekvenční spektrum akordu A-dur.



obr.19 Frekvenční spektrum akordu A-dur

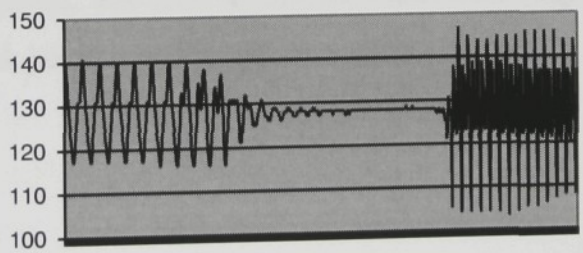
Na vodorovné ose sledujeme indexy frekvenčního spektra a na svislé ose sílu signálu. Každý nenulový index reprezentuje frekvenci obsaženou v akordu. Algoritmus pro zápis akordů do tabulátoru musí umět najít všechny

nenulové indexy a přepočítat je do frekvence. Dále je nutné odstranit vyšší harmonické frekvence, které jsou celočíselnými násobky jejich základní frekvence. Zůstane nám právě tolik základních frekvencí, kolik hudebník použil tónů (max. 6) při hraní akordu. Tóny se následně očíslují podle pražců a zapíší se do tabulátoru do jednoho sloupce, jelikož všechny tóny akordu se uskuteční ve stejném čase.

Nevýhoda tohoto systému převodu akordů na tabulátor vychází z podstaty algoritmu FFT, který dokáže analyzovat pouze velký úsek dat najednou. Může se stát, že v jedné sadě dat jsou zahrané akordy dva, nebo tři. FFT sice dokáže všechny akordy rozpoznat, ale nedokáže říci, který byl zahrán jako první a který jako poslední. Z toho vyplývá, že záznam do tabulátoru nemůže za použití FFT probíhat jako klasický záznam zvuku, tedy stlačením tlačítka *Record* a po odehrání celé skladby ukončením tlačítkem *Stop*.

5.2 Zápis melodie do tabulátoru

Druhá metoda analýzy signálu- měření délky zvukové vlny- je ideální pro zápis odehrané sekvence jednotlivých tónů do tabulátoru. Není sice vhodná k zápisu složených tónů a akordů jako FFT, ale její výhodou je, že prochází balíkem dat postupně a dokáže tedy určit, který tón byl do bufferu uložen jako první a který jako poslední.



obr.20 Dva různé tóny v bloku dat

Algoritmus zápisu jednoduché melodie začne prvním vzorkem bufferu a bude balík dat prohledávat dokud nenajde kompletní zvukovou vlnu. Pokud se délka této vlny bude opakovat periodicky, dejme tomu minimálně ještě pětkrát, máme jistotu že se jedná o tón a ne o šum. Algoritmus identifikuje tón, zapíše jeho umístění na kytaře do tabulátoru a pokračuje dál v procházení dat. Pokud se délka vlny změní jako na obrázku 20, a opět se periodicky opakuje, jedná se o jiný tón. Zapíše se do tabulátoru za tón předchozí.

Efektivní notový zápisník musí dále umět řešit problémy spojené s identickými tóny, umístěnými na různých strunách (viz. Kapitola Ladění a uspořádání tónů). Následující dva tabulátory mají zapsánu identickou melodii, ale jeden ze zápisů je zbytečně komplikovaný.

```
E |---------------|/---------------
H |---------------|/--------1------
G |--2---5--------|/--2------------
D |----3----5-----|/---------------
A |---------------|/-----8----10--
E |---------------|/---------------
```

Identické tóny mají v obou případech stejnou barvu. Zatímco první zápis je skutečně možné zahrát, druhý tabulátor je nesmyslně komplikovaný příliš velkou vzdáleností mezi pražci 1, 2, 8, 10. Notový zápisník musí být dostatečně inteligentní, aby jeho algoritmus umisťoval tóny do tabulátoru co nejblíže k sobě, tak jak je to vidět v prvním zápisu.

6 Závěr

Hlavním cílem diplomové práce bylo vytvořit aplikaci ve vývojovém prostředí Borland Delphi, v systému MS Windows, která bude umět zaznamenávat tóny z akustické kytary a porovnávat je s optimálními frekvencemi tónů. Za pomoci grafického zobrazení rozdílu obou frekvencí by potom mělo být možné nástroj přesně naladit. Program se skládá ze dvou částí. V první, nazvané Standard Tuning, se uskutečňuje vlastní ladění strun. Ve druhé části, Custom Tuning, může uživatel přednastavit odlišné ladění každé struny zvlášť.

Aby byl program „kytarová ladička" skutečně použitelný a efektivní v praxi, měl by splňovat tři podmínky. Interaktivnost, přesnost a názornost. Podařilo se dosáhnout všech tří předpokladů. Interaktivnost ladičky byla dosažena díky průběžnému testování a nastavení optimálního způsobu záznamu, analýzy, zobrazení výsledku a jejich synchronizace. Přesnosti bylo dosaženo použitím algoritmu FFT (Rychlá Fourierova Transformace) a jeho vhodnou adaptací na získaná data ze záznamu. Názornosti bylo dosaženo schematickým zobrazením strun, tak aby i uživatel-začátečník věděl, kterou strunu právě ladí. K dobré názornosti přispělo i řešení zobrazení výsledku měření frekvence- použitím posuvnítka (track-bar) i číslicového zobrazení frekvence.

Podařilo se vypracovat program, který dokáže změřit frekvenci s odchylkou maximálně 0,3 Hz, čímž překonává některé analogové ladičky. V průběhu vývoje programu se ukázalo, že lze výkon a funkčnost ladičky ještě vylepšit, což znamená velký příslib do budoucího rozvoje aplikace. Zejména v oblasti analýzy signálu- algoritmu FFT- lze dosáhnout mnohem stabilnějších a přesnějších výkonů.

7 Použité zdroje

[1] <http://en.wikipedia.org/> [online vědecká encyklopedie].

[2] Marshall, B.: *How Acoustic Guitars Work* [online].
<http://stuffo.howstuffworks.com/guitar.htm>

[3] Oppenheim, A. Schafer, R.: *Discrete-Time Signal Processing*. London: Prentice-Hall International Limited, 1989. 879 str.

[4] Petzold, Ch.: *Programování ve Windows*. Praha: Computer Press, 1999. 1216 str.

[5] Sedláček, J. Slaba, J.: *Delphi v Kostce*. Praha: Nakladatelství BEN, 1996. 424 str.

[6] Maršálek, T. Merta, M.: *Transformace signálů* [online].
< http://home.zcu.cz/~tmarsal/kro/>

[7] Brasseur, E.: *Discrete Fourier series Transform*. 1994 [online].
< http://www.4p8.com/eric.brasseur/fouren.html>

8 Příloha č.1

CD s aplikací „Tuner.exe“ a se zdrojovými kódy aplikace.