SZAKDOLGOZAT

Katona Gergő

Debrecen

2009

Debreceni Egyetem

Informatikai Kar

Moduláris hangrendszer fejlesztése

Témavezető:
Dr. Tornai Róbert
egyetemi adjunktus

Készítette:
Katona Gergő
mérnök informatikus

Debrecen
2009

Tartalomjegyzék

1. Bevezetés

A dolgozat célja egy olyan program fejlesztése és bemutatása, amellyel egy adott hangszer hangját tudjuk kezelni, változtatni, ahogy a dolgozat címe is mutatja modulálni. Munkám célja a programozási technikák, megoldások ismertetése és vizuálissá tenni azt, amit füllel érzékelünk.

A mindennapi életben nem fordítunk figyelmet annak elemzésére, hogy milyen hangszerek, technikai eszközök segítségével jöhetett létre az a zene, amit éppen a rádióban vagy zenelejátszónkon hallgatunk, azt természetesnek vesszük, hogy tiszta, jó és szép hangzást ad vissza a stúdióban vagy koncerten felvett hanganyag. Ennek ellenére egy laikus el sem tudja képzelni mennyi effektet és egyéb eszközt használnak a zenészek, hangmérnökök, mire a végtermék megszületik.

Az 1960-as években a lemezek analóg hangrögzítéssel, nyolc-sávos mágnesszalagra készültek. Az igazi áttörést a digitális rögzítési eljárás jelentette, jelenleg a legtöbb hangstúdióban ezt alkalmazzák. A számítógép megjelenése forradalmasította a hangstúdiók működését és a zenészek lehetőségeinek a horizontját is kiszélesítette. Mivel napjainkban jóformán minden háztartásban megtalálhatóak a számítógépek így a magánszférában is elterjedtek a könnyen kezelhető stúdió programok, amelyekkel mindenki szabadon kezelheti, effektezheti a saját hangszereit vagy bármilyen soundtrack-jét.

Az én programom elkészítésének az ötlete egyik legkedvesebb időtöltésemből fakad, a gitározásból. Mint sok más zenész én is előszeretettel gyűjtöttem a különböző eszközöket, amelyekkel jobbá, élvezhetőbbé alakítottam a gitárom hangját, azonban ezek az eszközök roppant költségesek, és összeállításuk elég sok időt vesz igénybe, és egy hobby-zenész ezt az időt inkább zenéléssel tölti, minthogy a legutóbbi beállításait keresgélje. Ennek a problémának a kiküszöbölésére fejlesztettem egy olyan programot mely több, a mindennapi életben használt effektet tartalmaz valamint egy gitárhangolót, amelyek segítségével gyorsan és egyszerűen megoldható a mindennapi életben szükséges kiegészítők összeállítása néhány kattintással a számítógépünkön.

A dolgozatom fő célja ennek a programnak elkészítése és bemutatása, a különböző effektek leírása, programozási technikák bemutatása, valamint a felhasználói környezet bemutatása. A megértéshez szükséges ábrák, képek nagy része a saját programomból származik, a más forrásból származó illusztrációkat a forrás megnevezésével jelöltem.

A legnépszerűbb technológia bemutatásával kezdem a dolgozatomat, amelyet a Steinberg nevű, zenei szoftverekkel foglalkozó cég fejlesztett ki, és virtuális hangszerek, effektek létrehozására használható, tulajdonképpen ez ihlette a dolgozatot.

Aztán az általam választott programozási nyelv bemutatásával folytatom, kifejtem miért ezt használtam és mik az előnyei mindamellett, hogy az általam választott Mérés és folyamatirányítás szakirányon előszeretettel használjuk és elég szerteágazó ismerettel rendelkezünk róla. Ezt követően a program moduljait fogom bemutatni, sorba véve az összes effektet és eszközt, amelyek a felhasználó rendelkezésére állnak és a programozási technikákat is kifejtem. Majd a felhasználói felületet egyfajta felhasználói leírás formájában mutatom be.

2. VST (Virtual Studio Technology)

A VST, avagy Virtual Studio Technology egy olyan, már-már szabvánnyá vált technológia, melyet a Steinberg nevű, zenei szoftverekkel foglalkozó cég fejlesztett ki, és virtuális hangszerek, effektek létrehozására használható, mindez kiegészítve egy hangrögzítő rendszerrel. Pluginok ezrei elérhetőek, melyek között található freeware és kereskedelmi célból létrehozott komponens egyaránt. A VST pluginok általában egy digitális audió munkafelülettel együtt használhatóak, ami kezeli a hostot és további funkciókkal bír. Nagy mennyiségű VST plugin lehet titkosított, akár VSTi-ről vagy effektről van szó. Általában a pluginok gondoskodnak a grafikus felhasználói felületről, a lejátszás vezérléséről, fizikai kapcsolókról.

2.1. VSTi (Virtual Studio Technology Instruments)

A VST instrument-ek audio jelet genrálnak. VSTi-k a VST technológiát felhasználva vitt egy nagyon nagy pluszt a számítógép felhasználók életébe. Ezek az eszközök tartalmaznak egy virtuális szintetizátor szimulációt, amely ugyanazokkal a tulajdonságokkal van felruházva, mint az eredeti és ugyanúgy is hangzik. A VSTi-k MIDI-n keresztül az audio outputra küldik a hangokat, amíg az effekt plugin-ok dolgoznak az adaton a kívánt hangzás elérése érdekében.

A MIDI jelek szintén használhatóak mind a hangszer, mind az effekt modulok paramétereinek irányítására. A legtöbb alkalmazás lehetővé teszi, hogy az egyik VST audió output-ja, egy másik VST inputjakét legyen felhasználható. Például egy VST szintetizátor outputja egy másik VST visszhang moduljának bemeneteként értelmezhető. Arra alkalmas hardver és driver-ek segítségével, valós idejű módban is használhatóak ezek a modulok[1].

3. LabVIEW programozási nyelv

A LabVIEW, azaz Laboratory Virtual Instrument Engineering Workbench egy olyan fejlesztői környezet és egyben vizuális programozási nyelv, amelyet a National Instrument fejlesztett ki. Legfőbb alkalmazási területe a virtuális méréstechnika és automatizálás. 1986-ban Apple Macintoshra fejlesztették ki, de futtatható egyéb platformokon is. (Pl.: Windows, Unix különböző verziói, Linux) A LabVIEW kompatibilis más programokkal. (Pl.: Matlab, MathWorks, NI MMATRIXx.) Az alkalmazások Webhez történő kapcsolódására beépített eszközöket tartalmaz.

Beépített kontrol és indikátor gombjaival műszerek előlapját lehet megtervezni; a be- és kimenetek között a tényleges adatáramlás vezetékeken át, a program által definiált módon történik. A vezetékek csomópontokban találkoznak. Egy-egy csomópont lehet függvény, struktúra, valamint valamely alprogram is. A LabVIEW lehetővé teszi a megírt program működésének, az adatáramlásnak lépésenkénti vizsgálatát, így a virtuálisan megépített hardver elemek könnyen tesztelhetők.

A LabVIEW-ban való programozás teljesen más módon történik, mint azt korábban megszoktuk, ugyanis itt nem szöveg alapú forráskódot kell bepötyögnünk, hanem az úgynevezett blokk diagramot kell létrehoznunk, ez a programunk forráskódja és szorosan kapcsolódik a front panelhez, ahol tulajdonképpen a leendő felhasználó szemével láthatjuk a programunkat.

A programunkat grafikusan szerkeszthetjük meg, de emellett a LabVIEW különlegessége az adatfolyam programozásban rejlik. Az ilyen programozási nyelvek virtuálisan leképezik a valós adatfolyamokat, amelyeket specifikálni szeretnénk programunkban. Általában különálló események kezelésére vagy adatfolyamok feldolgozására használjuk. A nyelv szerkezetét és elemeit ennek az adatfolyamnak a minél egyszerűbb leírhatósága határozza meg. A hagyományos procedurális, illetve objektum orientált programozási elvek és konstrukciók nem, vagy csak áttételesen alkalmazhatók.

3.1. VI (Virtual Instrument)

A LabVIEW programozási nyelv programozási egységét virtuális műszereknek vagy VI-nak nevezzük. A VI-nak három fő része van:

- előlap panel (front panel)
- diagram panel (block diagram)
- ikon/konnektor

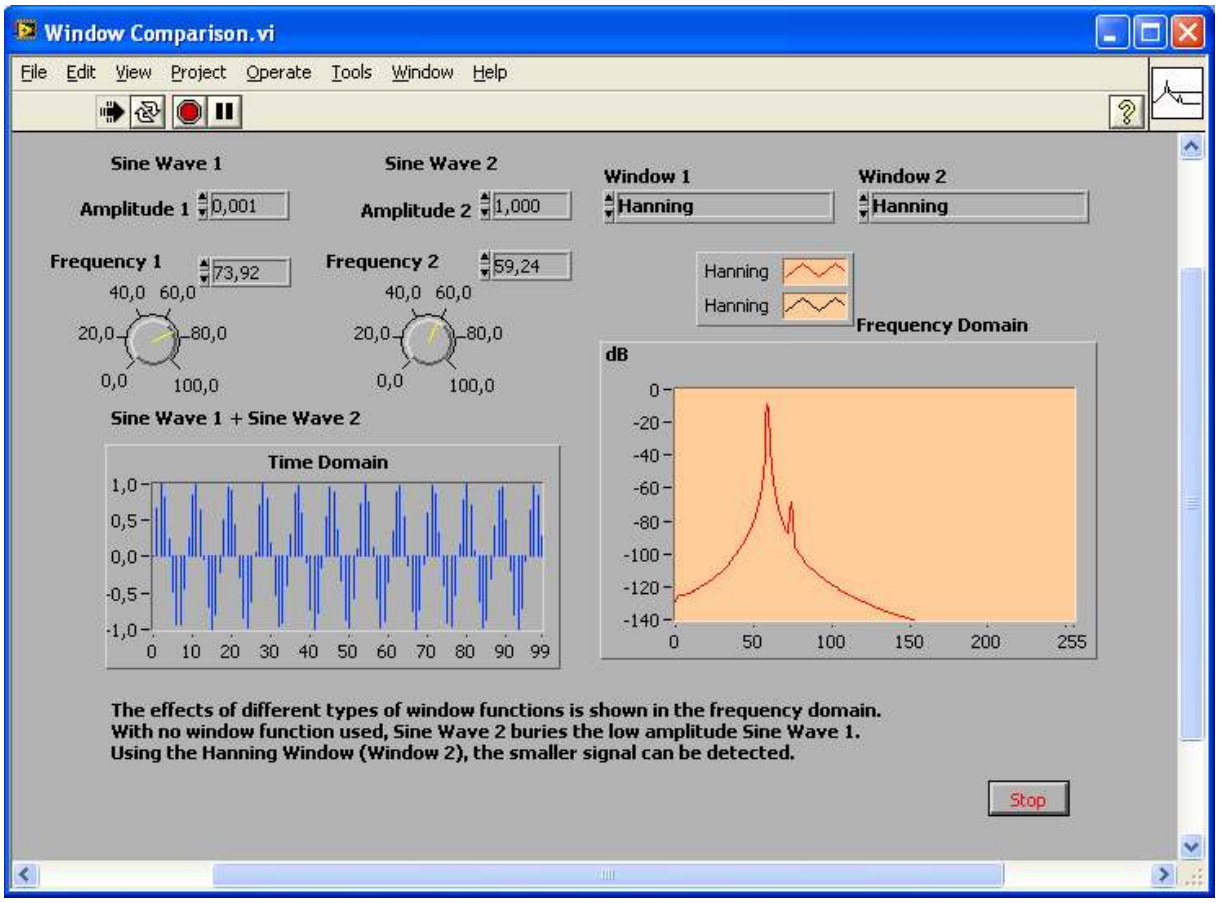
Az előlap panelen helyezhetjük el a beállító és megjelenítő elemeket, amelyek a felhasználói kapcsolattartás elemei lesznek, minden programozó célja a könnyű kezelhetőség elérése. Minden előlap panelnek van egy saját diagram panelja, amelyben a VI a program blokk diagramját hozhatjuk létre. A LabVIEW grafikus program nyelve segítségével építhetjük fel a blokk diagramot, amely egyben az elkészített program forrásnyelvi leírása.

Az ikon arra szolgál, hogy a VI-ból subVI-t hozzunk létre. Így lehetővé válik, hogy szubrutinként vagy függvényként alkalmazzuk más VI-okban. Az ikon grafikusan ábrázolja a VI-t más VI-ok blokk diagramjában. A konnektor csatlakozók megmutatják, hogy hova kell bekötnünk a VI bemeneteit és kimeneteit, amikor subVI-ként alkalmazzuk.

A LabVIEW-beli programozás legnagyobb előnye a VI-ok hierarchikus felépítésében rejlik. Egy VI létrehozása után, azonnal alkalmazhatjuk egy másik blokk diagramban, mint subVI-t. A hierarchiában nincs korlát a szubrutinok kölcsönös hívásának mélységére vonatkozóan.

3.1.1. Előlap

Az előlap panel segítségével a bemeneti értékeket állíthatjuk be, valamint a blokk diagrammal kezelt értékek kiírása, kirajzolása valósul meg ezen a felületen. Mivel az előlap panel megfelel a valódi műszerek előlapjának, ahol több kezelőszerv és megjelenítő is megtalálható, a bemeneti elemeket beállítóknak (controls), míg a kimeneti elemeket megjelenítőknek (indicators) nevezzük. Ezeket, az elemeket az 1. ábra szemlélteti. A LabVIEW fel van szerelve beállító és megjelenítő elemek teljes arzenáljával, ennek köszönhetően megtaláljuk a forgatógombokat, kapcsolókat, nyomógombokat, diagramokat és grafikonokat annak érdekében, hogy a lehető legpraktikusabbá és legérthetőbbé varázsoljuk a front panelünket.

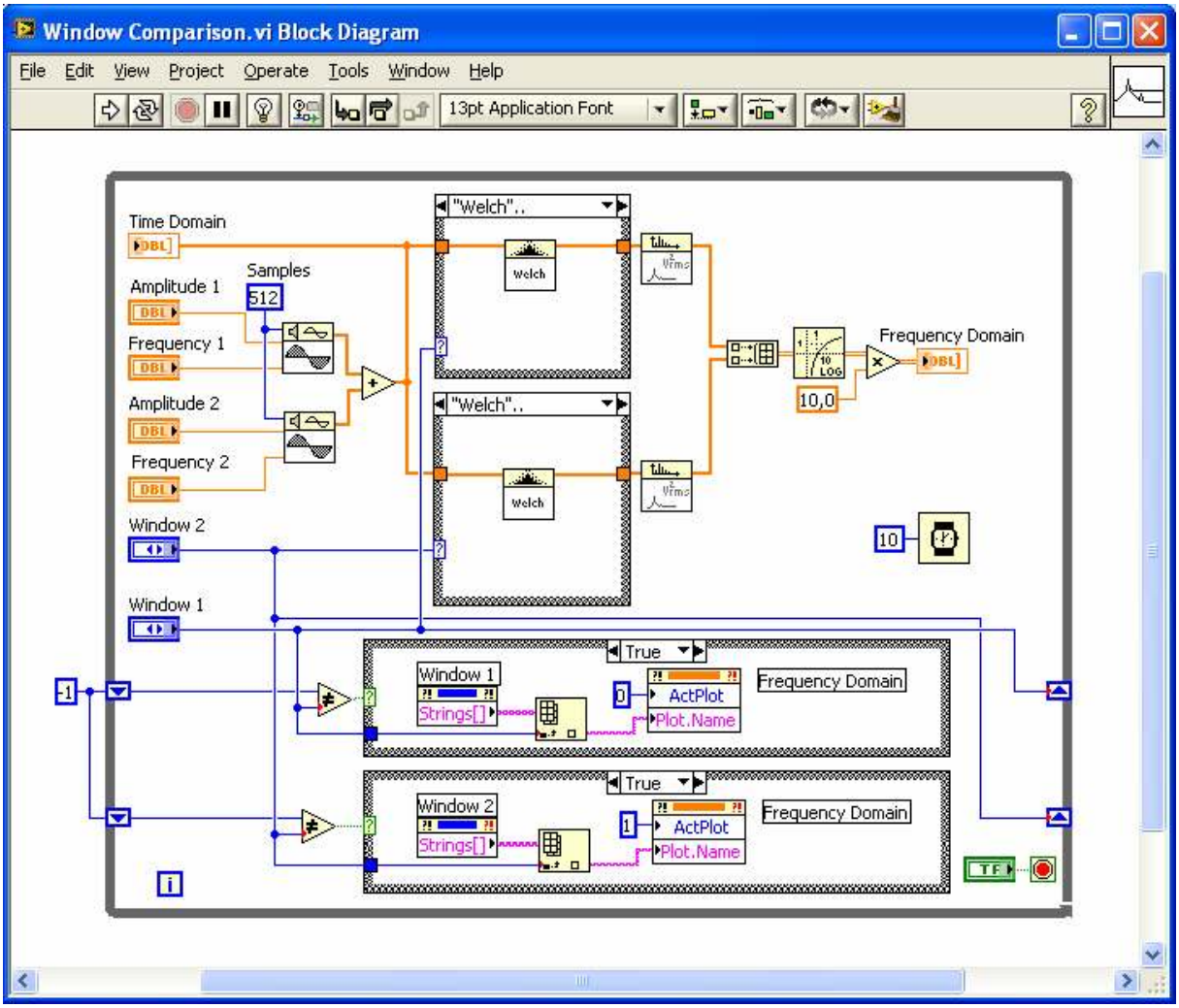


1. ábra. *Előlap*

(Forrás: NI LabVIEW/Help)

3.1.2. Diagram panel

Az előlap panellel szoros kapcsolatban lévő diagram panel tartalmazza a program kódot, magát a program szövegét. A blokk diagram elemeit a programozás csomópontjaiként értelmezzük, ilyenek a ciklusutasítások, a feltételes elágazások és az aritmetikai függvények. A különböző elemek közötti kapcsolatot adatvezetékek segítségével teremthetjük meg, amelyek egyben az adatfogalmat is definiálják a blokk diagramban. Az előbb felsoroltakra a második ábrán láthatunk példát.

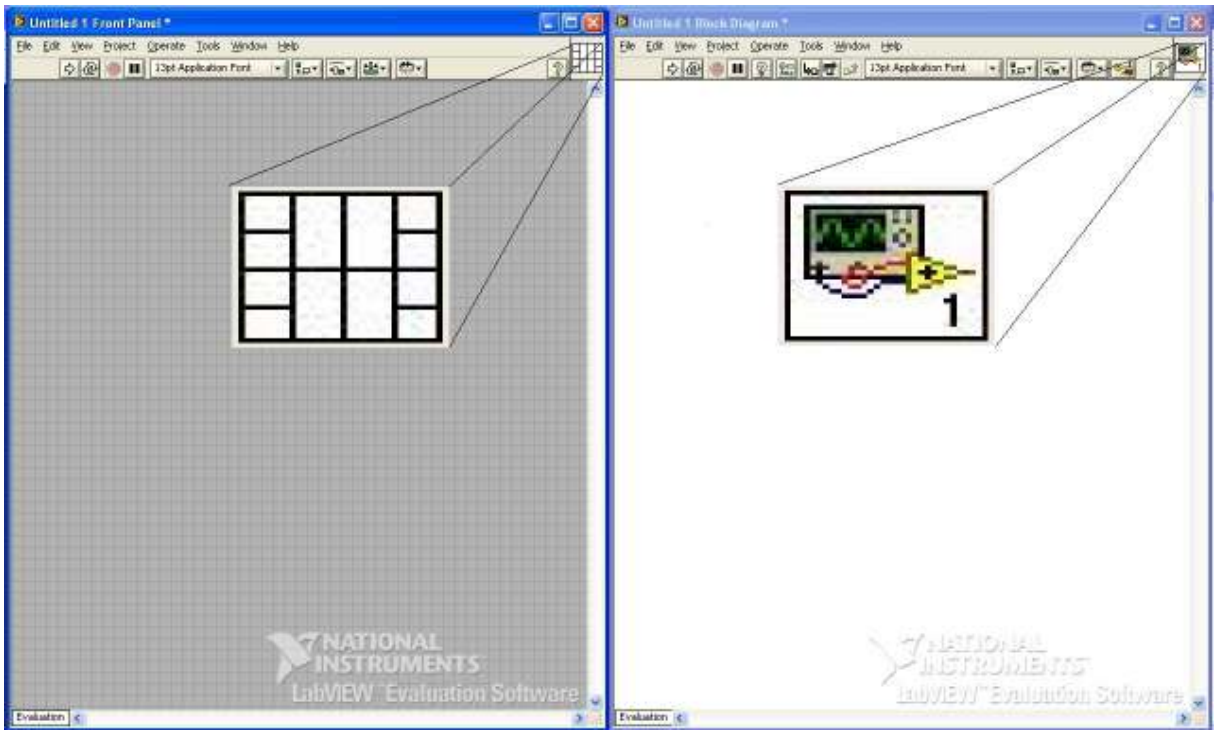


2. ábra. *Diagram panel*

(Forrás: NI LabVIEW/Help)

3.1.3. Ikon és konnektor

Az ikon és a konnektor segítségével, hozhatunk létre subVI-t a VI-ból, ezzel lehetővé téve más VI-okban való alkalmazásukat függvényként. Az ikonnal jelölhetjük a VI-t más VI-ok blokk diagramjában. A konnektor csatlakozói jelzik az ikonon lévő bemeneti és kimeneti kapcsolódási lehetőségeket. A konnektor csatlakozók tulajdonképpen a szubrutin paraméterek, melyek megfelelnek a VI előlap panelján lévő beállító és megjelenítő elemeknek. A csatlakozók rejtett állapotban helyezkednek el az ikon mögött addig, amíg láthatóvá nem tesszük őket, amelyet könnyen megtehetünk egy egyszerű nézet váltás segítségével, a 3. ábrán ezt a megváltoztatott nézetet láthatjuk.

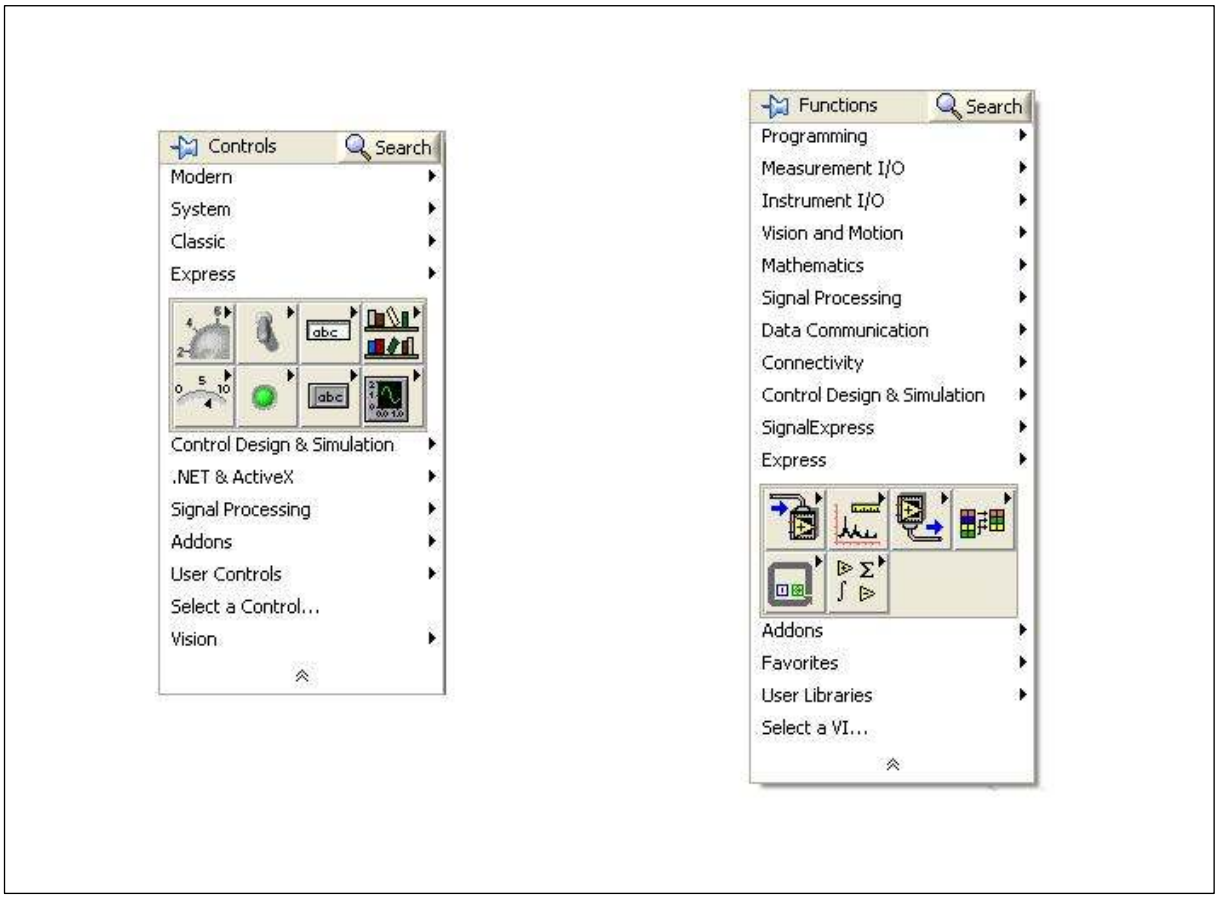


3.ábra. *Konnektor és ikon*

3.1.4. Paletták

A VI-ok írásához, futtatásához, hibakereséséhez, javításához szükséges eszközöket úgynevezett palettákba szervezik. Az előlaphoz és diagram panelhez külön-külön tartozik egy saját paletta.

A funkciók paletta csak a blokk diagramról érhető el. Ez a paletta a blokk diagram elkészítéséhez használható függvényeket és VI-okat tartalmazza. A kontrol paletta kizárólag a front panelről érhető el. Ez az előlap kialakításához szükséges kontrolokat és indikátorokat tartalmazza, ahogy a 4. ábrán látható[2].



4. ábra. *Kontrol és Funkció paletta*

4. Program és hangszer kapcsolata

A hangszer csatlakoztatása a számítógéphez a hangkártyán keresztül történik, amely felépítésének köszönhetően válik alkalmassá ennek a feladatnak az ellátásra. A legtöbb hangkártyán találhat egy LINE IN csatlakozó, ami a hang bemeneti csatornája, ide bármilyen hangkeltő eszköz csatlakoztatható, a beérkező analóg hangot a kártya képes digitalizálni. A másik bemenet, ami a program szempontjából hasznos lehet a mikrofon csatlakozó, amely mikrofonok csatlakoztatására alkalmas, ez jóval kisebb teljesítményű eszközökre van kitalálva, mint a LINE IN. Ezt a legtöbbször kihangosításra és Vocie-over-IP alkalmazásokhoz (pl. Skype) használják.

4.1. Csatlakoztatási lehetőségek

Tehát hangszerünket több féle képpen is a számítógéphez csatlakoztathatjuk:

- közvetlenül a LINE IN bemenetre kötjük, egy 3,5 mm nagyságú jack csatlakozóval
- egy mikrofon segítségével, a mikrofont a mikrofon bementre kötjük és így akusztikus gitárunk hangját is kezelhetjük vagy egy már erősítőről szóló és akár effektezett elektromos gitár gitár hangját is tovább színesíthetjük a program adta lehetőségeknek megfelelően

A gitár kimenete egy 0,25 inch-es aljzaton át vezethető ki a gitárból, ezzel a probléma az, hogy az ide bekötött kábel másik vége is általában egy ugyanekkora csatlakozóval végződik, tehát a hangkártyán található 3,5 mm-es aljzatba nem tudjuk bekötni. Ennek megoldására egy átalakítót használhatunk, ami egyszerűen és könnyedén megoldja a problémánkat, ennek ellenére van egy hátulütője is, mégpedig az, hogy zajos. A zaj mindig megtalálható, és minnél több átalakítót, kábelt használunk hangszerünk használata közben, annál több zavaró hatás keletkezik, ami rontja a hangszer hangjának minőségét.

4.2. Hang és zaj

A hang szó három jelentés tartalmat hordoz:

- fizikai jelenség
- füllel érzékelhető külső inger
- a hang értelmi és esztétikai hatása

A hang, mint fizikai jelenség, valamely rugalmas közeg mechanikai zavarási állapota, amely a közeg rugalmassága miatt az energia támadási helyén rezgés alakjában jelentkezik és a hatás a rendelkezésre álló térben továbbterjed.

A hang, mint hallható inger, sokkal kisebb terjedelmű – akár frekvenciában, akár intenzitásban –, mint a fizikai hangfogalom. Az ember által hallható hang frekvenciája 20 Hz-től és 20 kHz-ig terjed.

A harmadik jelentését értelmi és esztétikai hatása adja. A fizkai és hallható hangtól való megküönböztetésül hangélménynek is szokás nevezni. Az ember szempontjából ez a legfontosabb jelentéstartalma a hangnak. Erre mutat a zaj kifejezés is. Fizikai meghatározás szempontjából ugyanis a hang és a zaj teljesen azonos fogalmak. Ami mégis megkülönbözteti a két fogalmat az az emberi értékelés. A zaj elsődleges emberi megítélése a zavarás, a kellemetlenség , sőt a keletkezett időszakos ártalom megítélésén alapszik. A zaj fogalommal szemben áll a jelzést, értelmi tartalmat vagy esztétikai örömöt okozó informatív hang. Az akusztikai jelenségek ilyen formájú szétválasztása emberi találmány, erről sem a fizika, sem a természet nem vesz tudomást.

Egy akusztikus hangszert kihangosítás nélkül használó zenész számára aligha ismert a zaj fogalma, de egy elektromos, vagy esetleg kihangosított akusztikus hanszer esetén, már jól ismert tényező a zaj, amely nagy „ellensége” a hangmérnököknek, zenészeknek, ezért a programom egyik legfontosabb eleme egy szűrő (Filter) eszköz, mely ennek a gondnak a megoldását teszi lehetővé[3].

5. A program moduljainak bemutatása, megvalósítása

A szakdolgozat téma megszületésekor az elsődleges terv az volt, hogy néhány effektet, modult készítek egy már meglévő VST programhoz C++ programozási nyelvben. De idő közben megismertem a LabVIEW adta lehetőségeket és ráébredtem, hogy ennek a nyelvnek köszönhetően akár egy teljes programot is készíthetek, amelyet sokkal inkább a magaménak érzek és nem csupán néhány effektet, hanem más, a gitárosok, zenészek számára hasznos modult is belecsempészhetek a programba. Több ötlet alapos végiggondolása után három konkrét modulra redukáltam a végcélt:

- effektek
- hangoló (tuner)
- hangvilla (tuner-fork)

5.1. Effektek

Ez a programom legfontosabb része, ugyanis ez adta a dolgozat alapötletét. Mint a bevezetőben említettem több éve gitározom és ebből kifolyólag volt szerencsém kipróbálni megannyi effektet és rájöttem mire van legnagyobb szüksége egy gitárosnak.

Igaz ez nagyon stílus függő, de nincs az a gitáros, aki ha publikus szerzeményeiben nem is, de saját maga szórakoztatására ne használna torzított gitárhangot vagy más néven distortion-t. Legtöbb esetben a rock műfajban tevékenykedő zenészek használják, de szinte minden zenei stílusban megtalálható valamilyen formában, több-kevesebb hangsúlyt fektetve erre a hangzásra.

A másik nagyon fontos kelléke egy zenésznek a szűrő, ami segít a játék közben keletkezett zaj eltüntetésében, amit a kábelek, elektronika vagy egyéb alkatrész, külső hatás okozott. Az erre a célra kifejlesztett effektet a hétköznapi életben „zajzár”-nak nevezik a gitárosok és az egyik legdrágább része lehet az egymás után kötött effektek által alkotott „kisvasútnak”.

5.1.1. Distortion

A korai gitárerősítők világában a torzítás nem egy opcionális effekt volt, hanem a hangosítás nem kívánatos velejárója. A legtöbb erősítő úgy volt kialakítva, hogy egyszerre több, különböző hangszert is képes legyen kihangosítani, amik zavarták az akkoriban elterjedt felcsiptethető hangszedőket. A hangszedő által leadott jelet az akkor forgalomban lévő üreges testű gitárok rezgései is torzították, mint egy akaratlan visszacsatolás

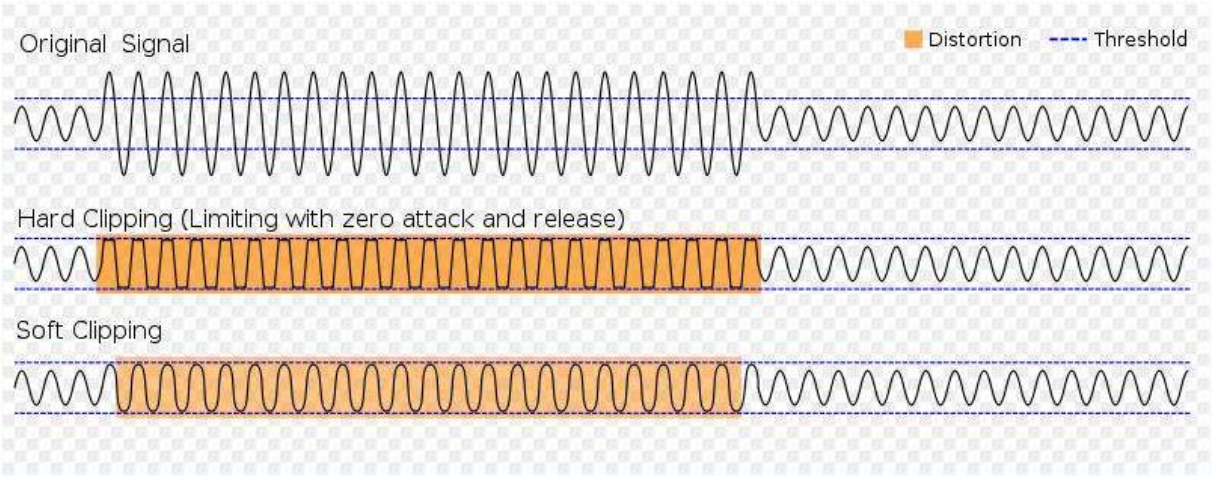
Később az 1950-es években elterjedtek a tömör testő elektromos-gitárok. Ezeknek a rezgései nem hatottak zavarólag és a kimeneti jelszintje is sokkal nagyobb és könyebben kezelhetővé váltak. A hangszerkészítők ezzel új korszakot nyitottak meg a zenei stílusok és a gitáreffektezés világában.

A szándékosan használt torzítás alapötlete, nem a korai erősítőkészítők érdeme. Az első példákat az erősítő valamilyen módú meghibásodása során keletkezett jeltorzítás szolgáltatta és a zenész vagy zenei producer úgy döntött, nekik ez tetszik és ilyen módon rögzítik az adott zeneszámot.

Az új hangzás felkeltette Leo Fender, a ma is világhírű Fender cég atyjának érdeklődését is és kifejlesztett olyan erősítőket melyek tartalmaztak torzításra alkalmas csatornákat a tiszta hangzás mellett. Fender ötletét lemásolva rengeteg cég kezdte el ezeknek az eszközöknek a gyártását és tették elterjedtté használatukat.

5.1.1.1. A torzítás gyakorlati megvalósítása

A distortion, azaz torzítás szó jelentése magába foglalja az output szignál bármilyen rendellenes eltérését az input jeltől. De a zenei hangszerek esetén, ahogy azt az 5. ábra mutatja, a levágás különböző formáját jelenti, aminek során megcsonkítjuk a bementi jel azon részét, ami meghaladja az adott feszültség szintet. Ennek következtében mind a csövek és a tranzisztorok lineárisan viselkednek az egyes feszültség szinteken belül, a torzító áramkörök úgy vannak behangolva, hogy a jel átlagos csúcsai, csak éppen hogy beleérjenek a levágandó tartományba, ezzel kis mértékű levágást és torzítást eredményezve. Így ha erősebben pengetjük a gitár húrjait, a torzítás mértéke és az eredményezett hangerő egyaránt nő, valamint óvatosabb pengetés tisztább hangot eredményez[4].



5. ábra. *Az eredeti jel csúcsainak levágása*

(Forrás: http://en.wikipedia.org/wiki/File:Clipping_compared_to_limiting.svg)

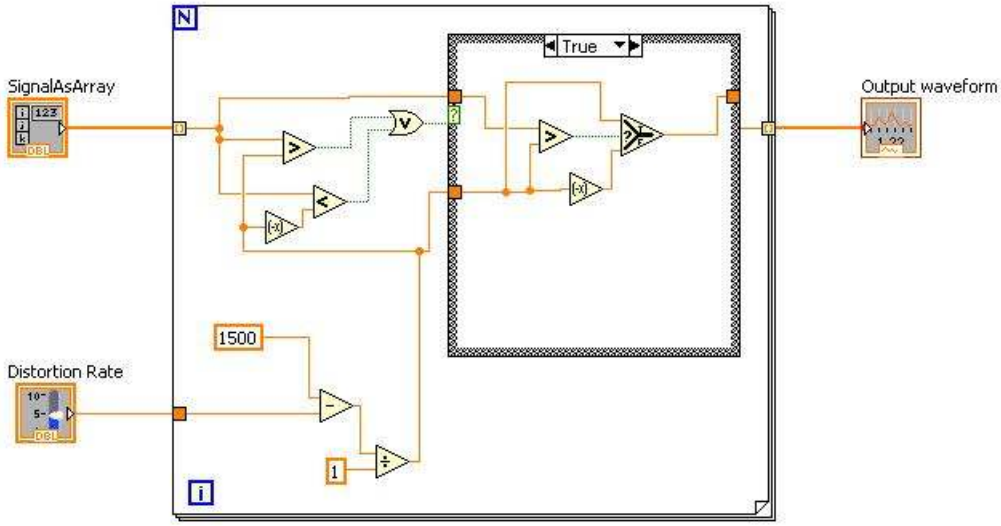
5.1.1.2. Distortion subVI

Ennek a subVI-nak két bemenete és egy kimenete van, ahogy a 6. ábra is mutatja. Bemenetként megkapja a hangkártyáról beolvasott jelet egy egydimenziós tömbként, valamint egy egész számot amivel a torzítás mértékét szabályozhatjuk. Kimenetén pedig a már kezelt, torzított jelet kapjuk meg hullámforma típusban.

A tömbként beérkező jel a hullámforma *y* értékeit, azaz az amplitudót tartalmazza. Ezeket az értékeket fogjuk vizsgálni a bemenetként kapott egész számhoz viszonyítva.

A „Distortion Rate” elnevezésű „Slide” segítségével a felhasználó beállíthatja a torzítás nagyságát. Ennek az értéknek vesszük a reciprokát, mivel azt adja meg, hogy mekkora legyen a tömb legnagyobb *y* értéke, ebből kifolyólag így érhető el a „csúszka” logikus használata. Tehát minnél nagyobb számot adunk meg, annál kisebb maximumot állítunk be, így a torzítás mértéke nő.

A kapott reciprokot és annak ellentettjét is összehasonlítom a tömbben érkező y értékekkel, mivel előre nem tudhatjuk, hogy éppen negatív vagy pozitív számot fogunk kapni. Ezt LabVIEW-ban a beépített Case struktúrával kezelem, mert ha az *y* érték kisebb, mint a felhasználó által megadott érték reciproka, akkor az eredeti értéket adjuk vissza, ellenben, ha nagyobb, akkor az összehasonlításnál használt érték lép a bemeneten megkapott *y* érték helyére.



6. ábra. *Distortion subVI*

5.1.2. Filter

Az elektromos szűrő egy olyan eszköz, amelyet a jelek egy csoportjának egy kevert jeltől való elfojtására, áteresztésére vagy szétválasztására használunk. A szűrők általában három nagy osztályba sorolhatóak:

- folyamatos idejű,
- mintavételezett,
- diszkrét idejű,

ez függ a jel típusától, amelyet a szűrő által kezelünk. Ennek következtében a jel fogalma alapozza meg a szűrő kivitelezését.

A jel függvénye egy vagy több független változónak úgy, mint az idő, tér, hőmérséklet stb., amelyek az információt szállítják. A független változók lehetnek folyamatosak vagy diszkrétek egyaránt. Feltételezve, hogy a jel az idő függvénye, folyamatos idejű vagy diszkrét idejű jelről beszélünk. A folyamatos idejű jelet minden időpillanatban, adott időközönként definiáljuk, míg a diszkrét idejű jelet konkrét idejű esetekben.

A valós világ szignáljait analógnak nevezzük, amely esetén az idő és mindkét amplitudó folyamatos. Ezeket a jeleket nem tudjuk kezelni digitális eszközökkel, hacsak nem konvertáljuk diszkrét idejű jelekké őket. Ezzel ellentétben a digitális jel leírható diszkrét értékekkel, amelyeket egy adott időpillanatban definiálunk. A digitális jel leírható véges számű értékekkel, amelyek általában bináris számok. A kapcsolat a folyamatos idejű jelek és a megfelelő diszkrét idejű jel között, a következő összefügéssel adhatjuk meg:

$$X(kT)=x(t)_{t=kT},\ k=0,1,2,\ldots,$$

ahol T a mintavételezési ciklus.

Mivel az én esetemben is analóg jelet kellett egy szűrővel kezelnem, így a Butterworth szűrőt használtam[5].

5.1.2.1. Butterworth filter

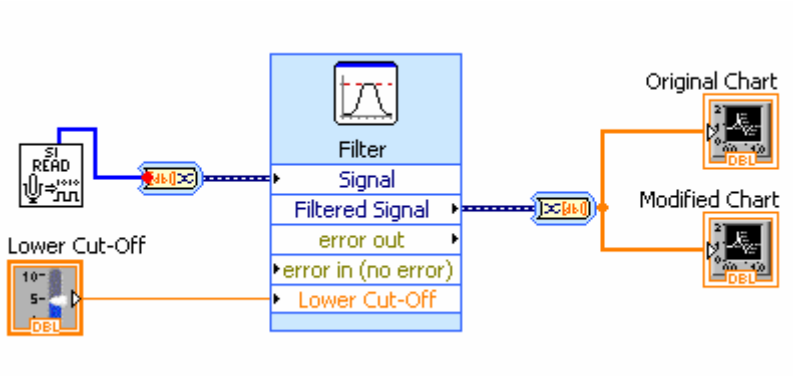
A hagyományos analóg szűrőformulák, mint a Butterworth, egyértelműen transzformálhatók IIR (Infinite Impulse Response) digitális szűrők specifikációjává. A Butterworth egy aluláteresztő szűrő. Ezek olyan áramkörök, melyek kisfrekvenciájú jeleket változatlanul átengednek, a nagyfrekvenciájú jeleket pedig a frekvencia növekedésével arányosan csillapítják[4].

5.1.2.1. Filter a LabVIEW-ban

A LabVIEW programozási nyelv egy beépített Express VI segítségével teszi lehetővé a szűrő létrehozását, ennek a VI-nak az ikonja a 7. ábrán látható. A VI bemenetként az inputról beolvasott jelet kapja meg dinamikus adattípussá való konvertálás után és ugyanúgy dinamikus adattípusként adja vissza, amit mielőtt a kimenetre küldenénk egy dimenziós tömbbé alakítunk.

Az Express VI tulajdonságait megjelenítve több féle opció is elénk tárul és ezeket saját igényeinknek megfelelően tudjuk beállítani. Először a szűrő típusát tudjuk állítani, az én esetemben egy aluláteresztő szűrőre esett a választás. Ezután a vágási frekvenciát tudjuk állítani, amit én a felhasználói felületen megjelenő csúszka segítségével a felhasználó számára elérhetővé tettem és ezt saját igényünk szerint tudjuk állítani futtatás közben.

Ezen beállításokat követően IIR és FIR szűrő közül tudunk választani, majd ezen belül a topológiát tudjuk beállítani, amit az én esetemben a fent említett Butterworth filterre állítottam és a használat során ez áll a felhasználó rendelkezésére.



7. ábra. *Filter Express VI*

5.1.3. Delay

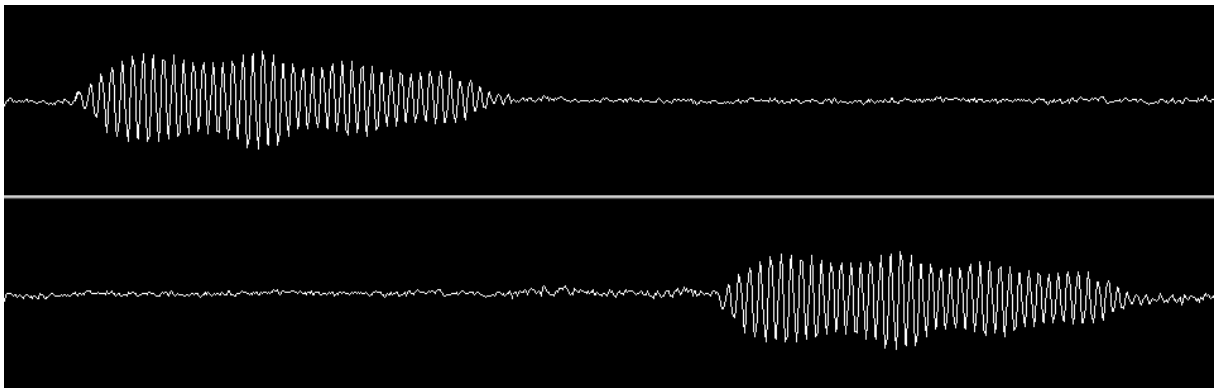
A delay effekt egy olyan hangeffekt, amely tárolja a bejövő jelet és egy bizonyos periódus idő után visszajátsza azt. Egyfajta időbeli shiftelést jelent, mivel a kimeneten pontosan a bemeneti jel jelenik meg, csak egy adott időintervallummal később, ahogy azt a 8. ábrán is láthatjuk .

Ez az effekt elég régi múltra tekint vissza, ugyanis léteznek korai analóg delay effektek is. Számos delay egység, analóg mágnesszalagos rögzítésen alapul, több mérnök is ezt a technológiát használta, úgy mint Ray Butt 1952-ben vagy Mike Battle 1959-ben vagy a Roland cég 1972-ben. Ezeknél az eszközöknél elektromos motor hajtotta a szalagot és rögzítette, majd játszotta vissza a hangot.

A Binson Echorec egy másik elterjedt egység, forgó mágneses dobokat használt, és ezzel tárolta a bejövő jelet. Ez biztosította az előnyét a szalaggal szemben, mivel tartósabb, strapabíróbb volt. Valamint hallhattunk elektroncső alapú megoldásokról is. Néhány modern zenész szerint ezzel más hangszínek is elérhetőek.

Az 1970-es évek végén, amikor elérhetővé váltak és olcsók lettek a digitális jelfeldolgozáshoz szükséges eszközök, a digitális delay effektek fejlesztése vált uralkodóvá. Kezdetben nagy méretű dobozokként és nagyon drágán voltak elérhetűk, de később elkezdtek ezek az eszközök zsugorodni és lábbal kapcsolható pedálok formájában jelentek meg a piacon. Az első ilyen delay-t a Boss cég gyártotta 1984-ben.

A digitális delay rendszerek úgy működnek, hogy mintát vesznek egy analóg-digitális átalakítón keresztül és ezt egy tároló bufferbe rögzítik, majd visszajátsza a felhasználó által beállított időegység után[6].

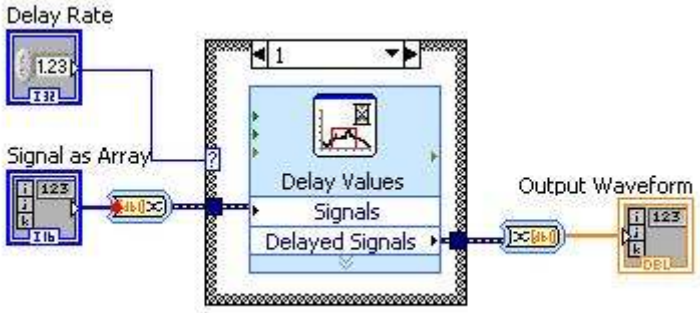


8.ábra *Delay hullámforma*

5.1.3.1. Delay subVI

A Delay subVI a 9. ábrán szemléltetett módon két bemenettel és egy kimenettel rendelkezik, a bemeneten a bemenő jelet kapja meg egy egy dimenziós tömbként, valamint a késleltetés mértékét egy egész szám formájában adhatjuk meg. A kimeneten pedig a bemenettel teljesen megegyező jelet adja vissza a VI, viszont a felhasználó által meghatározott időmennyiséggel később.

Egy beépített Express VI segítségével valósítottam meg a subVI magját, emiatt a tömbként beérkező bemeneti jelet dinamikus adattípussá konvertáltam, mert az Express VI csak ilyen formában tudja ezt kezelni. A „Delay Values” elnevezésű VI működése elég röviden leírható, egy a ciklusba bejövő adatot tárol el és azt a ciklusnak egy előre meghatározott lefutás után újra a rendelkezésünkre bocsájtja. Ennek a lefutásnak a számát tudja a felhasználó beállítani és ezzel a késleltetés mértékét megadni.

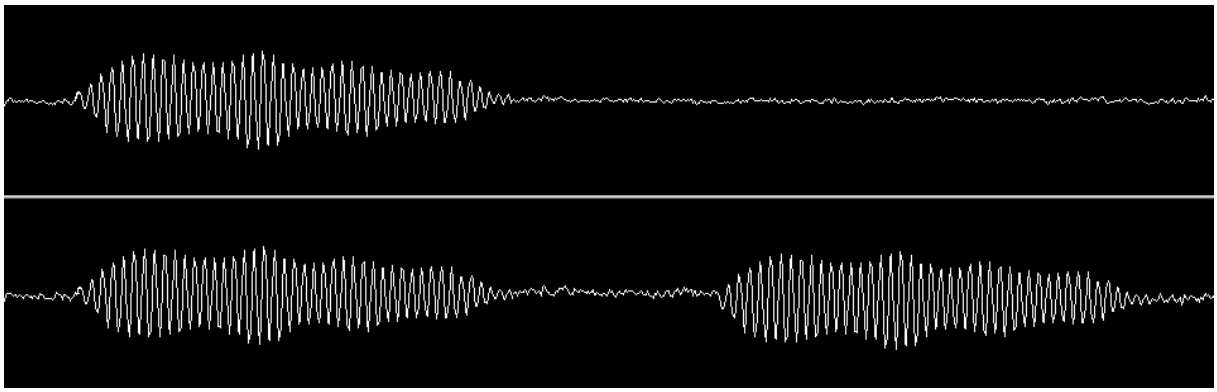


9. ábra. *Delay subVI*

5.1.4. Echo

Az echo, angol szó jelentése visszhang, amely szónak a jelentését elsősorban nem a zene és a hangszerek világából ismerjük, hanem egy a természetben, környezetünkben előforduló fizikai jelenség fogalmából.

Fogalmát úgy határozhatnánk meg, hogy a keletkezési helyre visszavert és itt újra észrevett hang, a hullámformáját a 10. ábrán láthatjuk. Ha valamely faltól, sziklafaltól vagy erdőszéltől bizonyos távolságra hangosan kiáltunk, annyi idő múlva, amely szükséges, hogy a hang a falhoz majd onnan viszatérjen, a hang újra hallatszik. Magyarországon is ismerünk több ilyen helyet, ahol ez a jelenség megfigyelhető, egyik legismertebb Tihanyban észlelhető. Ez valójában a XVIII. század közepe óta létezik, azaz a ma is álló apátság felépítése óta, ugyanis az ún. Visszhang-dombról elkiáltott szavak a több mint 300 méterre levő apátság faláról viszaverődve keltik a visszhangot. Csokonai Vitéz Mihály az 1878-ban írt versében is „echó”-nak nevezte a visszhangot.



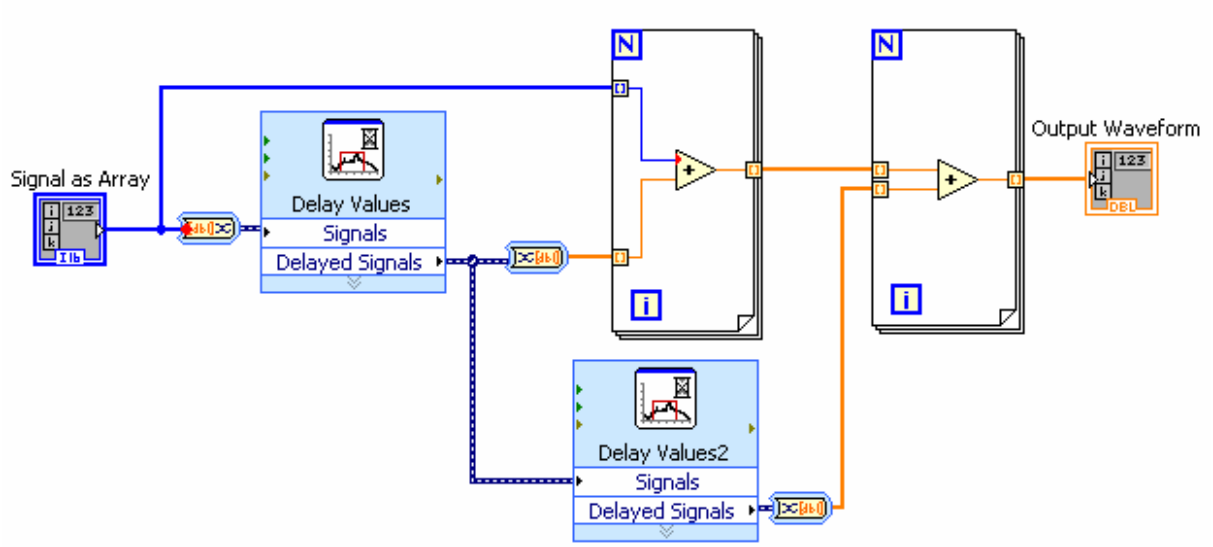
10.ábra. *Echo hullámforma*

5.1.4.1 Echo subVI

Az Echo subVI-nak csupán két bemenete és egy kimenete van, ahogy ez a 11. ábrán is látható. Mivel a felhasználónak nincs lehetősége finombeállításra az adott effekt esetében, így a bejövő jelen kívül nincs más érték a VI bemenetén.

A „Delay Values” elnevezésű Express VI-t használtam ehhez a subVI-hoz is, akár a Delay subVI esetében. A Delay Values által visszaadott értéket az éppen beérkező jelhez kapcsolva létrejön a visszhang. A végső verzióban kétszeresen késleltetem a hangot és így a visszhangot kétszer is halljuk, így szemléletesebb lett az eredmény.

Az ábrán látható „Signal as Array” nevű tömb jelzi a bemenetet és az „Output Waveform” pedig a kimeneti konnektora az adott VI-nak[6].



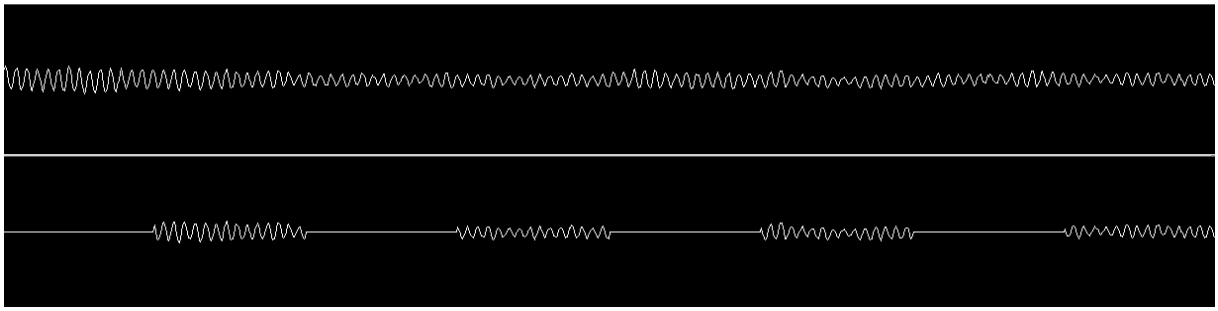
11.ábra. *Echo subVI*

5.1.5. Slicer

Az előzőleg bemutatott effektek nagyon gyakran használt és elterjedt effektek, ellentétben a most bemutatásra kerülő slicer, azaz magyarul „szeletelő”. A gitárosok egy igen kis csoportja ismeri és még kisebb részük használja ezt az effektek, de én nagyon érdekesnek találom, egyszerűsége ellenére nagyon fel tudja dobni az általunk játszott zenét.

Ahogy a neve is mutatja ez az effekt „felszeleteli” a gitárunk hangját, ezáltal a hang folytonosságát megszakítja és gitárunk által leadott jel adott hosszúságú részeit periódikusan kinullázza, ezt a jelenséget a 12. ábra nagyon jól szemlélteti.

Ezt az effektet gyakran kombinálják más effektekkel, legtöbbször torzítással, mivel nagyon jól kiemeli a játék dinamikáját és egyszerűségével színesíti a gitáros játékát.



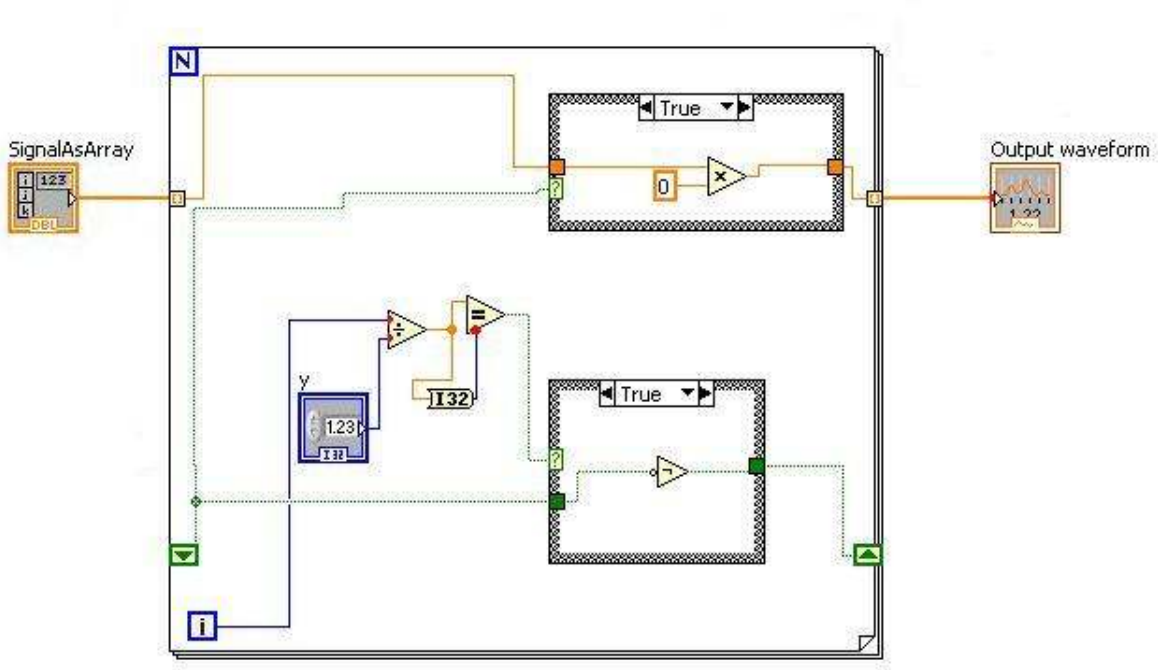
12. ábra. *Slicer hullámforma*

5.1.5.1. Slicer subVI

Ennek a subVI-nak két bemenete és egy kimenete van. Bemenetként egy egy dimenziós tömbként kapja meg a gitáron játszott dallamot, valamint a felhasználó által megadott egész számot, ami a vágások gyakoriságát határozza meg és kimenetként a módosított jelet adja vissza.

A VI bemenetén megkapott egész szám azt kontrollálja, hogy a mintavételezés során hány értéket nullázzon ki a program, majd ezt követően hány érték maradjon változatlanul és ez így folytatódik periódikusan. Ezt úgy érdemes megválasztani, hogy az általunk játszott zene ritmusához igazodjon a vágások gyakorisága.

A ciklusban egy általam készített, tulajdonképpen maradékos osztásnak megfelelő feltétel dönti el, hogy mikor engedi át az eredeti és mikor nullázza ki a program az értékeket, a részletek megértését a 13. ábra segíti.



13. ábra. *Slicer subVI*

5.2 Hangoló

A hangoló egy olyan eszköz, amelyet zenészek használnak a hangszerük által leadott hangok magasságának a mérésére. Ezek az eszközök többnyire egy mikrofonnal és egy jack-aljzattal is el vannak látva, hogy mind elektromos és mind akusztikus hangszerekhez is használni tudjuk. Az egyik legismertebb modell a 14. ábrán látható.

A legtöbb hangoló egy LCD kijelzőt használ arra, hogy jelezze a hangszer által keltett hangnak mekkora a frekvenciája. Valamint felismeri azt a fél hangot, amihez a legközelebb van az adott húr frekvenciája és később ehhez a félhanghoz fogja viszonyítani a hangmagasságot és ennek a kettőnek a viszonyát LED-ek segítségével jelzi a használójának. Általában kettő darab piros és egy zöld LED található egy ilyen eszközön, a bal oldali piros LED jelzi ha a gitárunk húrja mélyebb, a jobb oldali ha magasabb az adott félhangnál és a zöld tájékoztatja a zenészt, ha a hang megfelel az adott félhangnak.

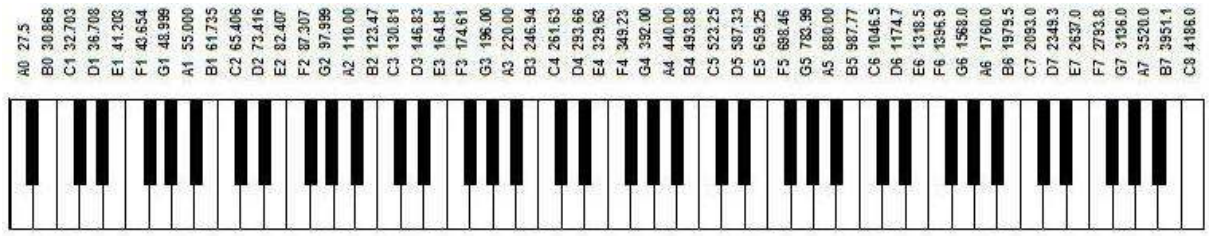
Az általam készített hangoló is hasonló virtuális külsővel rendelkezik és minden olyan szolgáltatást magában foglal, amit egy ilyen kézi készülék[7].

14. ábra. *Korg cég által gyártott hangoló*

(Forrás: http://i.ehow.com/)

5.2.1. Hangoló subVI

A hangoló által felismert hangterjedelmet a zongorán lejátszható hangskálához igazítottam, azaz a legmélyebb hang amit felismer a program az a 27.5 Hz frekvenciájú A és a legmagasabb a 4186,01 Hz-es C hang, ahogy ez a 15. ábrám is látható. Az ezen intervallumon kívül eső hangok esetében a program jelzi, hogy az adott hang magasabb vagy mélyebb az általa kezelt hangskála szélső értékeitől.



15.ábra *Zongora billentyűk frenkvenciái*

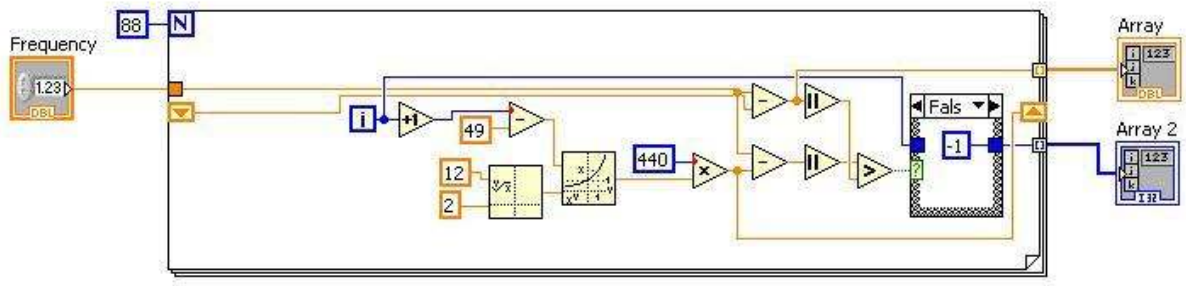
(Forrás: http://www.bioee.ee.columbia.edu/courses/ee3082/piano/piano_keys.jpg)

A következő függvény szükséges ennek a modulnak a működéséhez, amely megadja a a zongora n-edik billentyűjének a frekvenciáját:

$$F(n) = 440\ (\sqrt[12]{2}\,)^{n-49}$$

Ebből a függvényből ki tudjuk fejezni a frekvenciát és ezek után a beérkező hang frekvenciájának ismerete esetén könnyen meghatározhatjuk, hogy a zongora hanyadik fokához esik legközelebb az adott zenei hang[8].

A for ciklus ciklusváltozója a 16. ábrán látható módon 1-től 88-ig fut le és kiszámolja az adott bilentyűhöz tartozó frekvenciát, majd leellenőrzi, hogy az aktuális vagy a következő frekvenciájához van közelebb, és ennek megfelelően egy negatív értéket ad vissza, ha az aktuálishoz van legközelebb, ezeket az értékeket egy tömbben tárolja. Ezután a tömb minimumát megkeresve, annak indexe egyértelműen azonosítja, a bemeneten észlelt hang frekvenciájához legközelebb eső zenei hangot és egymáshoz való viszonyukat.



16. ábra. *Hangoló subVI for ciklusa*

A for ciklus után és miután eldöntöttük melyik hangot keressük egy logikai vizsgálat következik a programban. Az adott zenei hang frekvenciájához viszonyítva a program eldönti, hogy az általunk játszott hang magasabb vagy mélyebb és ennek megfelelően gyullad ki a bal oldali vagy a jobb oldali piros LED. Ha mélyebb hang érkezik a bemeneten akkor a bal, ha magasabb a jobb oldali LED gyullad ki és ha a megadött tűréshatáron belül esik a félhanghoz viszonyított eltérés, akkor a középső zöld LED gyullad ki, ez a tűréshatár 0,6 Hz.

5.3. Hangvilla

A hagyományos hétköznapi hangvilla egy akusztikus rezonátor, amelynek két ága van és a zenészek a normál „A” hang meghatározására használják. A 17. ábrán egy szokásos, mindennapi hangvilla látható. Ez az eszköz mindig 440 Hz-en rezeg és csak ezt az egy hangot tudja kiadni magából. Pontosan ezt a frekvenciát 1812-ben Párizsi Konzervatóriumban használták és fogadták el először. Az ezzel az eszközzel történő hangolás úgy történik, hogy az adott hangszeren az „A” húrt behangoljuk a hangvillához viszonyítva, majd ehhez a húrhoz viszonyítva egyenként behangoljuk a többit is.

A mai kor zenekaraiban sokszor találunk olyan hangolású hangszereket, amelyek eltérnek a hagyományostól és új irányba viszik el a zenei hangmagasságokat. Sok esetben használnak a szokásosnál egy-két vagy akár csak egy fél hanggal mélyebbre hangolt gitárokat, amelyek esetén körülményes a hangvilla használata, mivel nincs olyan tisztán megpengetett húr, amely „A” hangot adna ki. Ezt a problémát küszöböli ki az általam írt modul, mivel itt a zongora skáláján megtalálható összes hangot tudjuk reprodukálni és így megkönnyítjük, azoknak a felhasználóknak a dolgát, akik adott esetben nem tudják csatlakoztatni a hangszerünket a számítógéphez[9].

17. ábra. *Hagyományos hangvilla*
(Forrás: http://www.uxsight.com/product/images/)

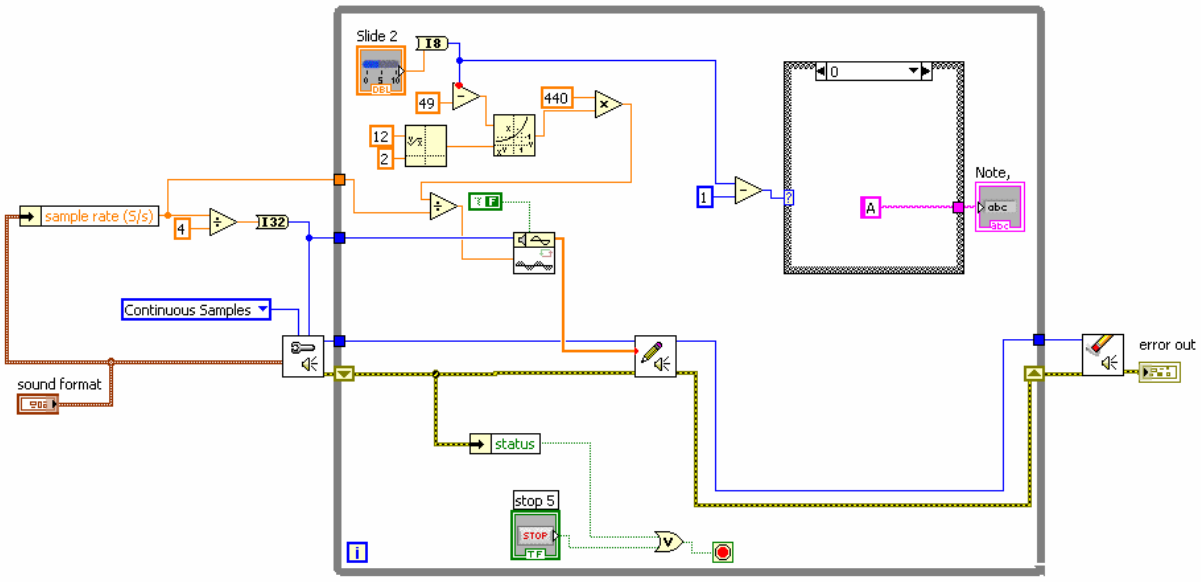
5.3.1. Hangvilla subVI

A hangvilla subVI csupán egy bemenettel rendelkezik, amelyen megadhatjuk, hogy a zongora hányadik fokán lévő hangot szeretnénk hallani egy csúszka segítségével. Ez program modul is 27,5 Hz-től 4186,01 Hz-ig terjedő frekvencia sávval dolgozik, ebbe az intervallumba és hangokat tud létrehozni. Kimenetként pedig az aktuális zenei hangot jelző karakter adja vissza a VI, ahogy ezt a 18. ábra szemlélteti.

Miután megkaptuk azt az egész számot amely a zongora egy billentyűjét azonosítja, a hangoló subVI-nál is használt képlettel a program kiszámítja a hozzá tartozó frekvenciát és ezt átadja a „Sine Wave” nevű beépített subVI-nak, amely megfelelően felparaméterezve, azaz a frekvencia és a mintavételek számának helyes megadásával szinuszjelként egy egy dimenziós tömb formájában adja vissza a kért zenei hangot.

Ezt a tömböt a „Sound Output Write” nevű beépített subVI-nak kell átadnia amely a kimenetre küldi és hallhatóvá teszi az általunk generált szinuszjelet. Ennek a VI-nak velejárója a „Sound Output Configure”, amellyel a kimeneti eszközünket tudjuk konfigurálni és a „Sound Output Clear” mellyel a folyamat befejezése után tudjuk lezárni a kimenetet.

A kimenetként megjelenő zenei hangot egy „Switch-case” struktúrával kapjuk meg, amelynek a zongora billentyűjének a sorszámát megadva, a megfelelő zenei hangot adja vissza[10].



18. ába. *Hangvilla subVI*

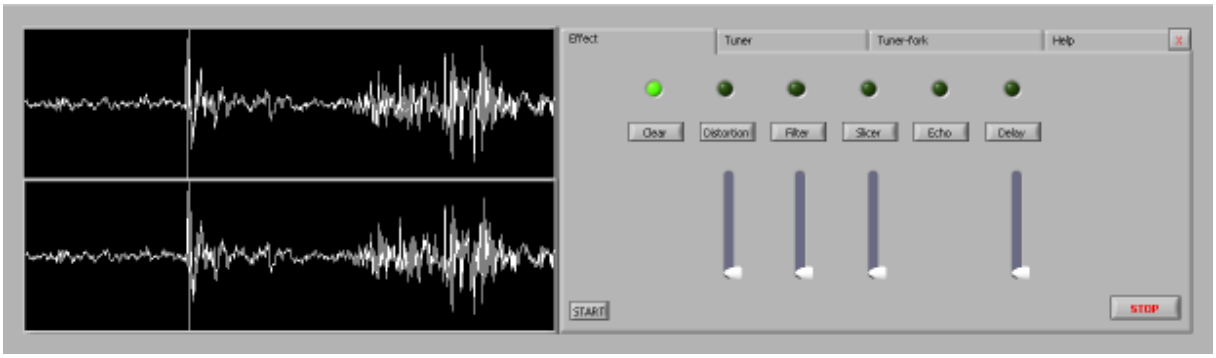
6. GUI (Graphical User Interface)

A GUI azaz Graphical User Interface, a grafikus felhasználói felületet jelenti, tehát a program használata közben a felhasználó elé táruló látványt, eszközöket jelenti. Az én esetemben, ahogy a 19. ábrán is látható két fő részből áll:

- grafikus kijelző felület
- kezelőfelület.

A kezelő felületet próbáltam úgy kialakítani, hogy praktikus, egyszerű és átlátható kezelést biztosítson a felhasználónak.

A változások folyamatos ábrázolása és a kapcsolók állapotának kijelzése nagyon fontos, ezért a bemenet és kimenet grafikus kijelzője minden esetben látszódik a felületen, a kapcsolók állapotát pedig LED-ek segítségével mutatja a program.

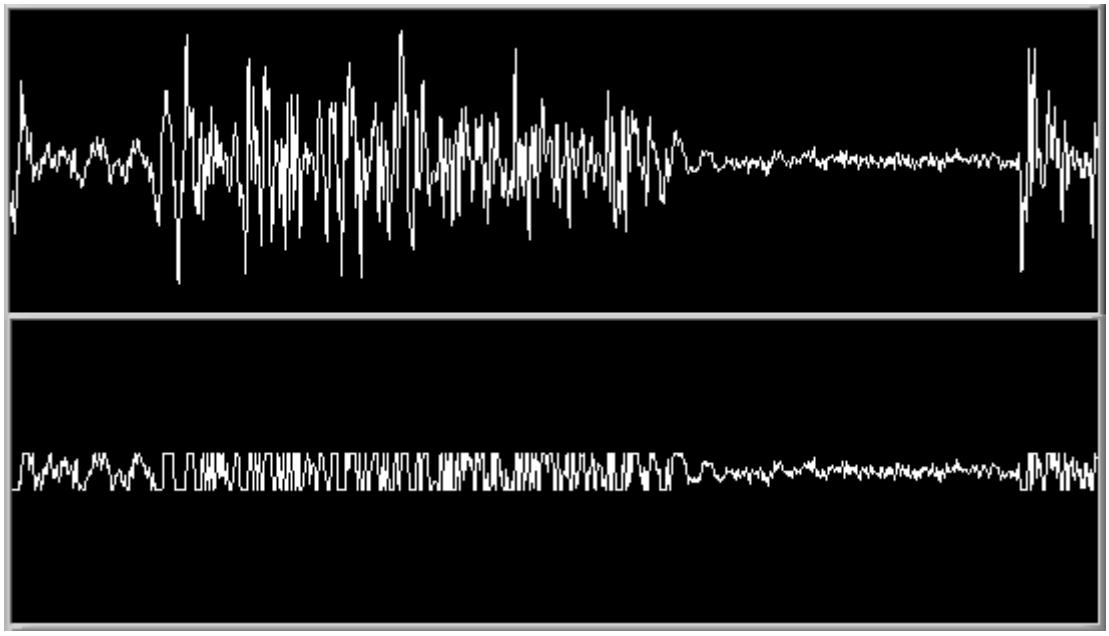


19. ábra. *Grafikus kijelző és kezelő felület*

6.1 Grafikus kijelző felület

A program által bemenetként kapott és kimenetként visszaadott modulált hang a felhasználó elé táruló felület bal oldalán található. A felső az eredeti az alatta lévő a program által módosított hullámformát jeleníti meg. A két jel közötti különbség a 20. ábrán nagyon jól észrevehető.

A *y* tengelyen a bejövő feszültségnek a nagysága, amplitudója jelenik meg, az *x* tengelyen pedig az idő változása jelenik meg. Az eltelt idő szinkronban van mind a két grafikonon, azaz mind a kettő azonos időpillanatot mutat. Viszont az amplitudó természetesen az effekteknek megfelelő változtatások szerint eltér az eredetihez képest.



20. ábra. *Grafikus kijelző*

6.2. Kezelőfelület

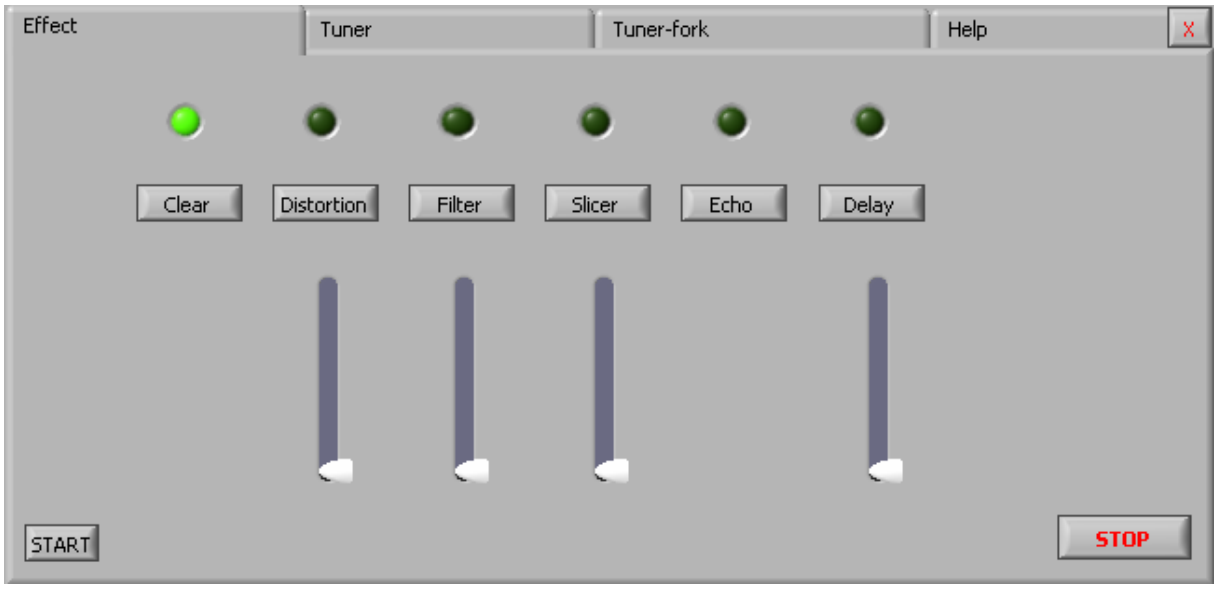
A kezelő felületen a három modul és egy „Help” menü között tudunk választani a következő sorrendben:

- Effect
- Tuner
- Tuner-fork
- Help

Lapfülek segítségével válthatunk a különböző opciók közül, így tudjuk a modulokat használni, elindítani és leállítani.

6.2.1. Effect

A program legfontosabb moduljának a felhasználói felülete alapvetően három dologból épül fel. Gombokból melyekkel az effekteket tudjuk váltani, LED-ekből melyek az aktuális csatornát jelölik, valamint majdnem valamennyi effekthez tartozik egy csúszka úgynevezett „slide” mellyel az egyes effektek paramétereit tudjuk változtatni.



21. ábra. *Effect modul kezelőfelülete*

A „Clear” a tiszta csatornát jelenti, azaz ebben az esetben a felhasználó pontosan az általa játszott formában kapja vissza hangszere hangját.

Ha a „Distortion” azaz torzítás csatornára váltunk, akkor a gomb alatt található csúszkával tudjuk változtatni a levágás nagyságát, tehát a torzítás mértékét.

A „Filter” vagyis a szűrő esetében pedig a levágási frekvenciát tudjuk állítani, tehát a felhasználó a számára legkedvezőbb szűrési beállítást tudja eszközölni az adott paraméter beállításával.

A következő effekt, amit a kezelő felületen találunk a „Slicer” nevű, eléggé szokatlan csatorna. Itt a játék közbeni kivágások idejének a hossza az amit be tudunk állítani és így a felhasználótól függ az effekt természete.

Az „Echo” azaz visszhang nem rendelkezik semmilyen beállítható paraméterrel, mivel én úgy véltem ez a leghasználhatóbb és legpraktikusabb beállítás, amit alapértelmezettnek állítottam be.

A „Delay” másnéven késleltető effekt csúszkájával a késleltetés mértékét tudjuk beállítani, és így a tetszőleges hatást érhetjük el, amely igazon az általunk játszott zene tempójához.

A 21. ábrán látható esetben a „Clear” csatorna aktív, ezt a felette világító zöld LED jelzi. Minden csatornaváltás esetén az éppen aktív csatornához tartozó LED gyullad ki.

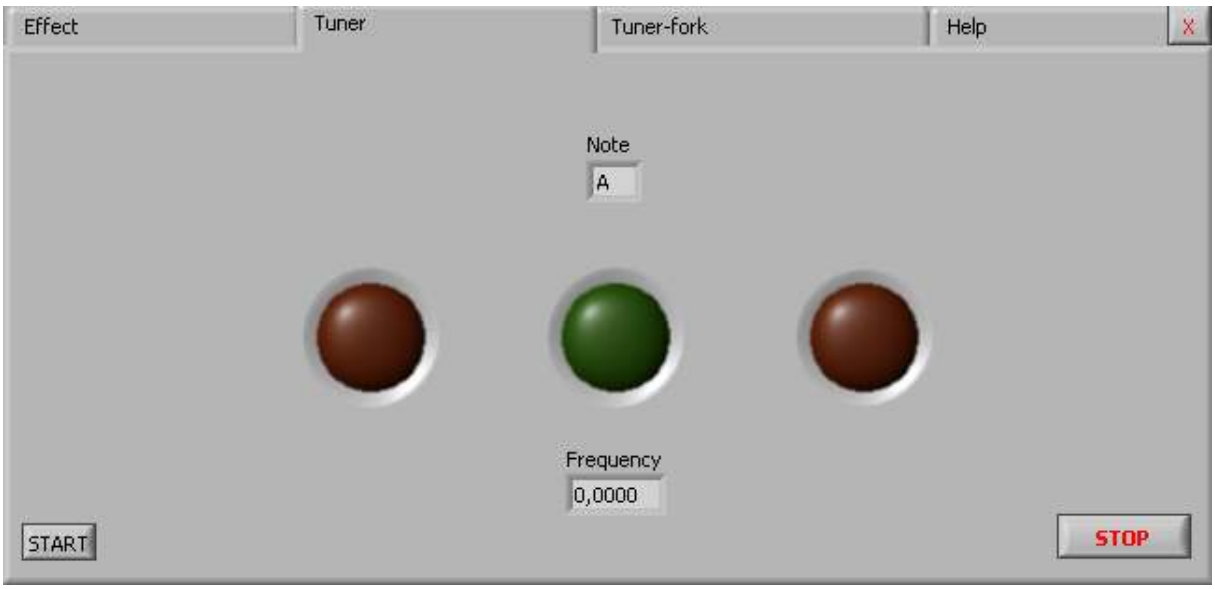
6.2.2. Tuner

A Tuner, azaz hangoló felhasználói felülete nagyon hasonlít egy analóg eszköz külsejére, ahogy azt a 22. ábra is jól mutatja.

A felületen található két szöveges felület, az egyik az éppen aktuális zenei hangot jelzi a felhasználó felé, a másik pedig a bejövő jel frekvenciáját írja ki a felhasználói felületre. A második információ nem biztos, hogy minden felhasználó számára értékes információt hordoz, de aki valamennyire jártas az egész és fél hangok frekvencia értékei világában, akkor az könnyebben igazodik el a hangok között a hangolás folyamat közben.

Az adott hang és a bemeneten kapott jel frekvenciájának viszonyát három LED segítségével közli a felhasználóval a program. A bal oldali piros LED akkor világít, ha a lejátszott hang mélyebb, a jobb oldali akkor, ha magasabb a bemenet érkező hang, mint az adott zenei hang. A zöld LED akkor világít, ha megközelítőleg megegyezik a két frekvencia értéke.

Ha erre a modulra váltunk a lapfülek segítségével, akkor ahhoz, hogy elindítsuk a „START” gomb megnyomása szükséges, valamint a hangolás befejezése után a „STOP” gombbal tudjuk leállítani azt.



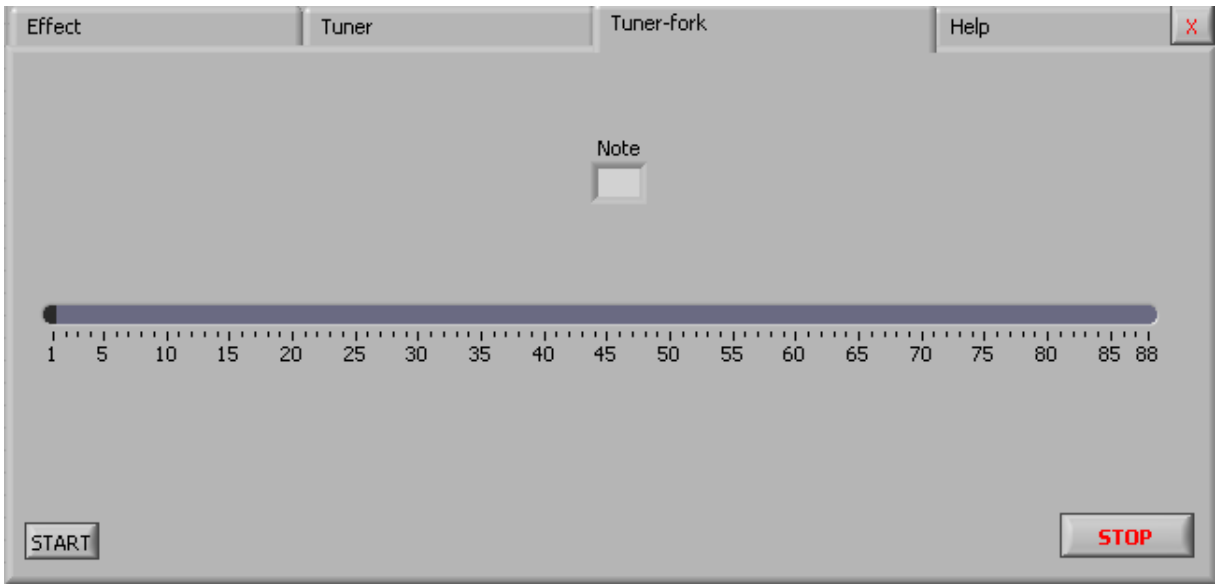
22. ábra. *Tuner modul kezelő felülete*

6.2.3. Tuner-fork

A „Tuner-fork” modul felhasználói felülete egy virtuális hangvillát tár a felhasználó szeme elé, amelynek igen egyszerű a használata, ezért könnyű és gyors segítséget nyújt, azoknak a felhasználóknak, akik nem rendelkeznek olyan eszközzel, melynek segítségével a számítógépéhez csatlakoztassa a hangszerét.

Ezt is a „START” gombbal tudjuk elindítani és a „STOP” gombbal tudjuk leállítani, a hangolás befejezése után, ezeket a jól látható gombokat a 23. ábra szemlélteti. Ez a legnagyobb előnyét képezi a hagyományos hangvillával szemben, mert amíg az csak meghatározott ideig, néhány másodpercig képes a normál zenei „A” hang reprodukálására, a modul által keltett hangot bármeddig hallgathatjuk.

A csúszka segítségével választhatunk, hogy melyik hangot szeretnénk hallani a zongora skálájáról. Az adott zenei hangot pedig a „slide” felett található szövegdobozban láthatjuk és győződhetünk meg róla, hogy a számunkra megfelelő hangot halljuk.



23. ábra. *Tuner-fork modul kezelő felülete*

7. Összefoglalás

A dolgozatban először az általam választhatott programozási nyelv, fejlesztői környezet a Labview bemutatásával kezdődött, amelynek szükségét éreztem a későbbi programozási eszközök, módok kifejtésének megértése miatt.

Ezután a program megvalósítását és a programozási technikákat írtam le, a programozás közben használt eszközöket fejtettem ki, ezáltál részletes képet nyújtva az olvasónak a fejlesztésbe befektetett energia mennyiségéről és folyamatáról. Minden modul bevezetéseként leírtam az adott eszköz működését, mi a célja és feladata, valamint a jövőbeni alkalmazásáról is próbáltam világos képet alkotni a laikusként olvasók számára is. Próbáltam vizuálissá tenni azokat a hallható elemeket, amik a mindennapjaink része, de nem feltétlenül veszünk róluk tudomást, miközben egy zenei számot hallgatunk a rádióban vagy bármilyen multimédiás eszközünkön.

Végül a grafikus felhasználó környezetet mutattam be, hogy az olvasó a program kipróbálása alkalmával könnyedén el tudjon igazodni a kapcsolók és csúszkák használatában, funkcióikat könnyedén értelmezze.

A szószedet segítségével próbáltam világossá tenni azokat a nem hétköznapi fogalmakat, amelyekkel az olvasó ritkán találkozhat és ködösítheti a megértést.

Remélem sikerült az olvasó számára érthetővé tennem az általam fejlesztett program használatát, felépítését, valamint azt, hogy hogyan kell egy ilyen program alapjait lefektetni és milyen felhasználói elvárásoknak kell megfelelnie.

Úgy érzem egy teljes és használható programot sikerült fejlesztenem, de mindig van lehetőség a továbblépésre. A későbbiekben szeretnék egy zeneszerkesztő modult létrehozni, valamint még több effektet implementálni a programban, amelyet vagy saját örömömre, vagy egy későbbi MsC dolgozat keretein belül sikerül véghezvinnem.

Irodalomjegyzék

1. VST: http://en.wikipedia.org/wiki/Virtual_Studio_Technology
2. LABVIEW 4.0 Basics Interactive: http://www.noise.physx.u-szeged.hu/DigitalMeasurements/LabVIEWTutor/
3. Tarnóczy Tamás: *Hangnyomás, hangosság, zajosság*, Akadémia Kiadó, Budapest, 1984.
4. Saeed V. Vaseghi: *Advanced Digital Signal Processing and Noise Reduction*, John Wiley & Sons, Ltd, 2006.
5. John G. Wbster: *Electrical Measurment, Signal Processing and Displays*, CRC Press LLC, 2004.
6. Kihong Shin, Joseph K. Hammond: *Fundamental of Signal Processing for Sound and Vibration Engineers*, John Wiley & Sons, Ltd, 2008.
7. Electronic Tuner: http://en.wikipedia.org/wiki/Electronic_tuner#Strobe_tuners/
8. Ctirad Smetana: *Zaj- és rezgésmérés*, Műszaki Könyvkiadó, Budapest, 1975.
9. Dr.-Ing. Ivar Veit: *Műszaki akusztika*, Műszaki Könyvkiadó, Budapest, 1977.
10. Musical Signal Processing with Labview: http://cnx.org/content/m15510/latest/

Szószedet

— **VST(Virtual Studio Technology)**

A VST, avagy Virtual Studio Technology egy olyan, már-már szabvánnyá vált technológia, melyet a Steinberg nevű, zenei szoftverekkel foglalkozó cég fejlesztett ki, és virtuális hangszerek, effektek létrehozására használható.

— **VSTI**

A VSTI egy úgynevezett VST instrument (eszköz), mely akár egy virtuális hangszer (pl.: dobgép,vagy szintetizátor) emulálását teszi lehetővé.

— **Plugin**

Egy adott szoftverbe vagy hardverbe opcionálisan beépíthető, annak képességeit bővíteni vagy módosító kiegészítő modul.

— **Freeware**

A freeware olyan, a szerzői jog által védett szoftver, ami ingyen használható, tetszőlegesen hosszú ideig

— **Vocie-over-IP**

Az Internet Protokoll feletti hangátvitel – elterjedt nevén VoIP, Voice over IP vagy IP-telefónia – a távközlés egy olyan formája, ahol a beszélgetés nem a hagyományos telefonhálózaton, hanem az Interneten vagy más, szintén IP-alapú adathálózaton folyik.

— **VI**

A LabVIEW programokat nevezzük VI-nak vagy virtuális műszernek.

— **LED**

A fénykibocsátó dióda vagy LED neve az angol Light Emitting Diode rövidítéséből származik. A dióda által kibocsátott fény színe a félvezető anyag összetételétől, ötvözőitől függ. A LED inkoherens keskeny spektrumú fényt bocsát ki. A fény spektruma az infravöröstől az ultraibolyáig terjedhet.