DESIGN AND IMPLEMENTATION OF VIRTUAL STUDIO TECHNOLOGY INSTRUMENT PLUG-IN FOR KIRAR

Exodus Getahun, Abykab Behailu and Menore Tekeba
zetsgtmu@gmail.com, abykabbehailu11@gmail.com, menore.tekeba@aait.edu.et
School of Electrical and Computer Engineering, Addis Ababa Institute of Technology, AAU

Key Words: DSP, Kirar, MIDI, VSTi,

ABSTRACT

With the rapid increase of computing resources within the last decade, several real-time applications such as sound editing and processing are becoming more popular. Virtual Studio Technology (VST) was introduced by Steinberg Media Technologies GmbH in 1996 which provided a platform for sound effect developers to implement their own effect plugins. In 1999 a standard framework for software sequencers, so called VST instruments (VSTi), was introduced in which music instruments are simulated in the virtual studio. To design an instrument as part of the virtual studio, a series of steps from audio sample collection and recording to integrating the samples using a plug-in to the VST is required. To the best of our knowledge, there is no VSTi plugins developed for Ethiopic music instruments so far. Therefore, we have effectively used the structure of the VSTi API and its Software Development Kit (SDK) tools to design and implement a plugin for Kirar (Ethiopian music instrument) and effectively integrated and tested. Our plugin was successfully recognized and operated in the Cubase5 and FL studio Digital Audio Workstations (DAWs) as a VSTi instrument which initiates its own graphical user interface (GUI) with full mouse and keyboard controls having 3 level Musical Instruments Digital Interface (MIDI) mapping.

INTRODUCTION

Virtual Studio Technology is an interface for integrating audio synthesis and effects plugins with DAW. In simple terms it is a way to get all those hardware instruments effects like racks, key boards and drum machines into music software such as Cubase, FL Studio or Logic Pro and others [1]. According to statement given in [2], VST allows the integration of virtual effect processors and instruments into your digital audio environment. It also includes software recreations of hardware effect units and instruments or new creative effect components into the VST systems [2].

With VST, Steinberg founded the world's leading and most widely supported standard for plug-ins and virtual instruments in 1996 for the first time in Germany [2]. A few years later, in 1999, Steinberg updated their VST specification which also allows VST plug-ins to receive MIDI data [3]. The current VST version is VST 3.x and marks an important milestone in audio technology with a code base providing many new features with the most stable and reliable VST platform ever. These VST modules have the sound quality of the best hardware units, yet are far more flexible [2]. Prior to 1996 DAW had been used to control keyboards and samplers via

MIDI and then routing all their external hardware through a traditional mixing desk to produce digital music [3].

Since the VST3 SDK is available as a free technology, open in use for any developer, we have used this SDK for our plug-in development and testing [2]. The VST technology is now a days serving as one focal point in machine human interaction to track finger movements in order to take inputs from human fingers movements an instruction to a VST system [4].
In the musical world, there are so many issues which need to be addressed in order to fulfill as possible as desires of music producers and composers. As a composer or musical producer, one needs to have full pack of the required software and hardware gear.
From the many software gears needed, one of the most powerful and modern tool of the time is having adequate VSTi instrument plugins at hand. This raises a question of do we really have enough plugins in the market?

The answer is no, and that's because hundreds of plugins are made and improved every year since the technology has been created in 1996.
One other big factor for creating more VSTi plugins in the industry is that, the more civilized countries become and try to build their musical industry, the more they become digitalized.

This leads to creating a virtual studio technology instrument plugin for their cultural instruments. This approach has been a trend in the musical industry as many of the major musical instrument developer companies have a world and ethnic category on their sound banks.

This brings us to the conclusion that there has to be a VST plugin which represents Ethiopian unique traditional instruments in the industry.
Therefore in this paper, we are going to show the development of VSTi plug-in for Ethiopian traditional music instrument known in Amharic language as Kirar. Important steps of audio sample recording, different digital signal processing methods, development of important modules of the plug-in and its GUI, integration with VST systems and testing are shown in the paper.

LITERATURE REVIEW

The research article given in [5] presents the review of the impact of VST systems and realization of different hardware instruments into the VST systems in the form of the plug-ins. It discusses the impact of the existence of all those massive instruments in music production industry. It also presents wide spread use of VST and the plug-ins. It concludes that the VST technology has totally revolutionized the music production world and the VST instrument plug-ins are very important components in the VST systems.

The research work given in [6] shows the automatic generation of VST audio plug-ins from MATLAB code using the Audio System Toolbox from Math Works. It provides MATLAB code for three complete examples of plug-ins, discusses problems that may be encountered in generating the plug-ins automatically, and describe a workflow to generate VST plug-ins as quickly and easily as possible.

The dissertation given in [7] discusses the impact of using digital production technologies especially for music production and has made three approaches of the investigation on the impact from multiple perspectives. It also concludes that the digital music technology has made a huge impact on the pedagogical and professional industry because it creates new directions for effectiveness and efficiency of technology implemented for music composition and pedagogy.

A similar work of ours has been done in Plectra series 4 for Turkish traditional music instrument known as Oud as it is given in [10] by Dimitris Plagiannis. As it is shown in Figure 1, they have developed a plug-in of Turkish Oud for commercial purpose and the web article lists the features of the plug-in developed.

Figure 1: GUI of the Turkish Oud VSTi [10]

The importance of VST technology and the plug-ins of hardware instruments to be used in VST systems is well-known and all the literatures supports this statement. Therefore, for Ethiopic style music to play important role in the global music market, the indigenous and traditional music instruments shall have their own plug-ins in the VST systems and this research project has designed, implemented a VSTi plug-in for Kirar and the developed plug-in has been successfully integrated and tested in CUBASE and FL Studio DAWs.

VST PLUG-IN

Even in its earliest incarnation, the VST system allowed third-party developers to produce real-time effect modules that could "plug in" to the host application (initially Cubase). But later, Steinberg introduced the second version of the VST plug-in standard, which enables to send MIDI data to and from such effects.

This opened an opportunity to developers to add more features such as MIDI control of effect parameters and locking of effect settings to tempo and integrate new instruments into the VST systems for more diverse effects and controls on the soft music productions.

Such advancements in protocol latter enabled MIDI information to be used to run synthesis engines, rather than just simple effects processors [3].

According to the web article on [8], a plug-in is a software unit, which does not usually function by itself, which comes to be grafted as part of a more complex program and enhances the functions of this program. Initially plug-ins appeared on graphic creation software like Photoshop in the form of filters, XPress with Xtensions, the concept of the plug-in was then transposed in to the digital audio domain.

With the audio effects plug-ins which can add audio effects like reverb to the MIDI/Audio sequencers and to the digital audio software and more recently in the form of instruments plug-ins which can add a synthesizer or a virtual sampler to the Midi/Audio program.

For the users these plug-ins have so many advantages such as creation of a modular software configuration, according to the needs, consequently reducing the cost of configuration
[8]. Figure 2 shows example of the instrument plug-ins for NI Guitar Rig 5 which works with VST as multi-effect processor

Figure 2: NI Guitar Rig 5 VST – Guitar multi-effects processor [9]

DESIGN AND IMPLEMENTATION OF VSTI PLUG-IN FOR KIRAR

In the process of making VSTi for Kirar, we choose to implement it through four phases. In the first phase, we have collected samples and made pre-processing on the samples of Kirar audio data. In the second phase, we have chosen the DSP methods and algorithms to add effects on the samples collected. Then in the third phase, the software of the plug-in was developed using C++ following MIDI API standards. In the last phase, the plug-in developed has been integrated with the major DAW Platforms (CUBASE and FL Studio in this case) and we have tested the plug-in with these two DAW systems. The phases of the development are described from Sections 4.1 to 4.4 as follows:

Sampling and Audio Collection of Kirar Music Instrument

Since Kirar has got the standard discrete sounds of music, we sampled 12 WAV files, each representing the 12 default root key/note scale values of a standard MIDI keyboard for 3rd octave. We used standard key frequencies for the third octave only as given in Table 1 below [11] and we have collected the samples for the octave as Kirar is functional for one octave only. We used a standard condenser microphone (Beringer-B1), USB sound card (AVID) and professional audio recorder DAW software (Cubase 5). Then we used audio tuner software (Audio Tuner v1.0) to tune the Kirar in the required frequency. We have made an emphasis on the tuning accuracy of the recorded samples by refining the pitch values so that they go with standard frequency values.

After sampling the KIRAR with standard sampling frequency of 44.1 KHz and root key value, our result was 12 independent 32 bit .WAV sample files, meaning one for each of the 12 notes. As shown in Figure 3, sampling and pitch correction GUI in the VST helps us to see the effect of our sampling.

In addition we have sampled more layers of samples to achieve the 3-level MIDI mapping. The challenges are in getting the samples and accurately deciding on the parameters of the envelope function used.

Table 1: Standard key frequency in hertz (Hz) for the three octaves [11]

Octave Note	1	2	3
C	32.703	65.406	130.81
C♯/D ♭	34.648	69.296	138.59
E ♭/D♯	36.708	73.416	146.83
E	41.203	82.407	164.81
F	43.654	87.307	174.61
F♯/G ♭	46.249	92.499	185.00
G	48.999	97.999	196.00
A ♭/G♯	51.913	103.83	207.65
A	55.000	110.00	220.00
B ♭/A♯	58.270	116.54	233.08
B	61.735	123.47	246.94

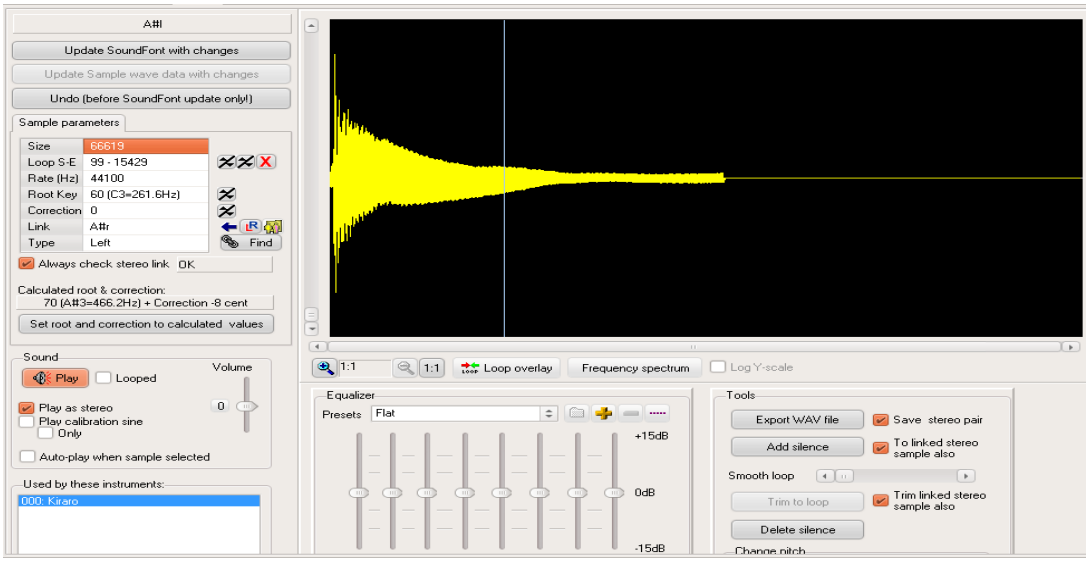

Figure 3: Sampling and pitch correction

DSP Methods used to Process Samples

After the samples are recorded and collected, different types of processing modules are implemented. Some of the methods used are:

Midi Filter: The MiDi Filter receives the input signal and filters it by root key/note and velocity then sends to next module. It allows controlling each MiDi input to give a MiDi output (no generators, no synthesis). We have used the VSTi SDK's built-in MiDiFilter module.

MidiToCv: After receiving a filtered input from, MidiFilter, MidiToCv controls the voltage and multiplexes the gate to the Oscillator and Attack, Decay, Sustain and Release (ADSR) module. In addition, it sends the pitch and velocity of the input to the Oscillator.

KirarOscillator: An oscillator is a repeating waveform with a fundamental frequency and peak amplitude which forms the basis of most popular music synthesis techniques these days. In addition to the frequency or pitch of the oscillator and its amplitude, one of the most important features is the shape of its waveform. Different types of oscillators, based on their shape, are sinusoidal oscillator, square wave oscillator, saw tooth oscillator, triangular oscillators and others [12]. We have used the built-in VST oscillator with low frequency for our plug-in development.

ADSR: An envelope generator (sometimes, called a transient generator) makes an audio signal that smoothly rises and falls as if to control the loudness of a musical note automatically. Amplitude control by multiplication is the most direct, ordinary way to use one, but there are many other possible ways. Even though Envelope generators have come in many forms over the years, ADSR envelope generator is the simplest and favorite one. ADRS is the short form of Attack, Decay, Sustain and Release which combines the four DSP operations on audio input [13]. The concept of ADRS is associated with the principle that the sound output of musical instruments does not immediately build up to its full intensity nor does the sound fall to zero intensity instantaneously. That means it takes a certain amount of duration for the sound to build up in intensity and a certain amount of period for the sound to die away as well. The interval during which a musical tone is building up to some amplitude (volume) is called the "attack time" and the period required for the tone's intensity to partially die away is called its "decay time". The time of attenuation at the final stage is known as "release time". Several instruments also allow holding of a tone for certain period of time known as "sustain time". The sustain time helps to achieve various note durations in music composition [14] as it is shown in Figure 4.

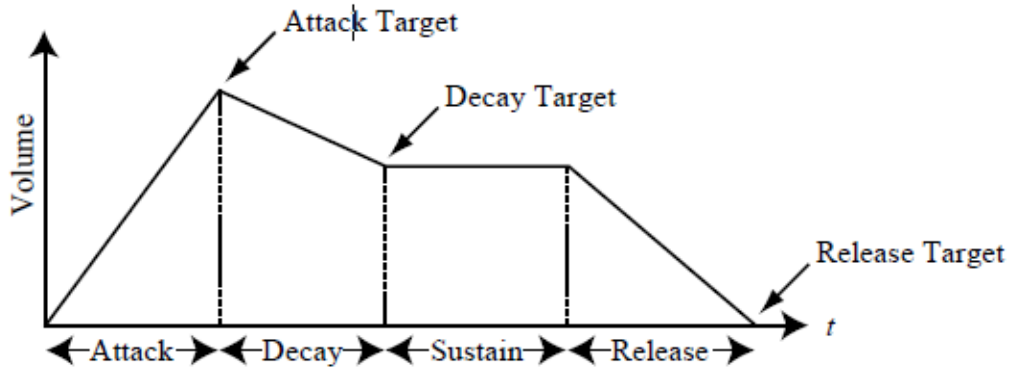

Figure 4: Classic ADSR envelope [14]

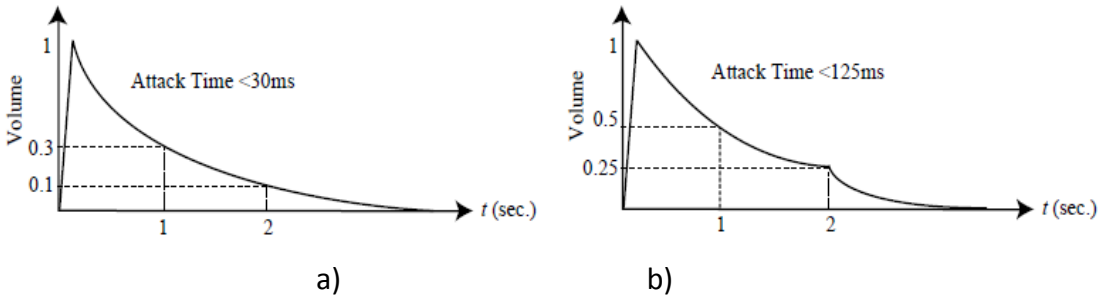

Figure 5: (a) ADSR envelopes for guitar, (b) ADSR envelopes for piano [14]

A synthesizer duplicates (increases intensity in case of Attack and decreases intensity in cases of Decay and Release) the intensity (volume) variation of the tone by multiplying (modulating) the amplitude of the sinusoid with a scale factor dictated by the ADSR envelope, a(t) as it is given in [14].

$$y(t) = a(t) * x(t) \quad (1)$$

In our plug-in development, we have used the ADSR envelope given for piano in Figure 5(b) above since piano is similar to kirar, as they both have a discrete time value and their sound pitch is basically the same.

Design and Implementation of the Plug-in Software

We designed our plugin in a way that it works on different DAW platforms which support the VSTi plug-in format. The "Kirar VSTi" first version got two basic features which are the SoundFont oscillator and Release DSP feature.

The features are powered with a simple graphical user interface which includes a loader button for the sound font file along with graphical slider for controlling the release DSP feature. The plugin works seamlessly on Cubase 5. The GUI of our plug-in inside the Cubase DAW is shown in Figures 6 to 8.

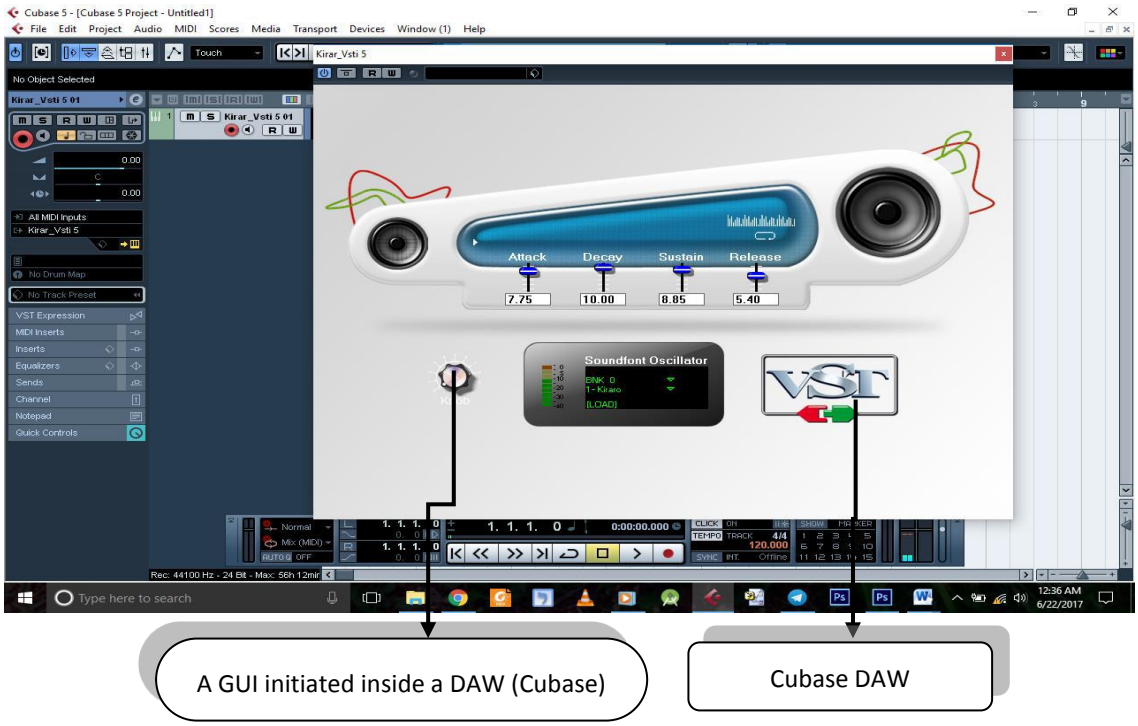

Figure 6: System Design

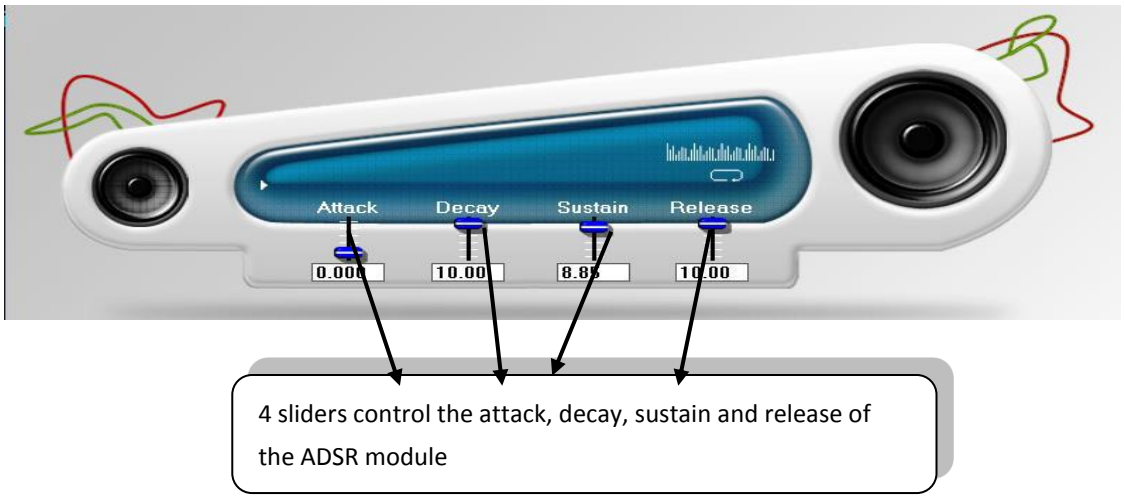

Figure 7: ADSR module

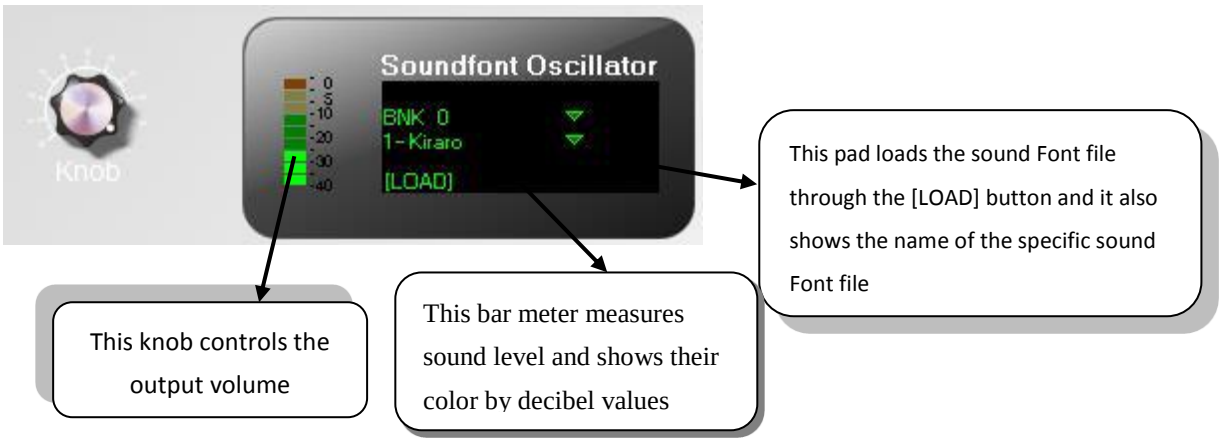

Figure 8: GUI Description

The designed plug-in has the following components:

Keyboard module: This module defines and identifies each of the keyboard key notes independently and it makes the system ready to be played through a computer and MIDI keyboards. Its inputs and outputs are MIDI data.

IoMod module: This module takes care of the inputs received from the keyboard and sends the output to the "Sound Out" module.

Midi Filter module: This module receives the input received through "IoMod" and filters it by root key/note and velocity and sends to the "Midi To Cv" module.

Midi To Cv module: After receiving a filtered input from "Midi Filter" module, this module controls the voltage and multiplexes the gate to the "Oscillator" and "ADSR" module. In addition, it sends the pitch and velocity of the input to the "Oscillator".

Kirar Oscillator module: After receiving a controlled volume input from "Midi To Cv" module, it generates/produces a pure tone by shaping the waveform of the input.

ADSR module: This module takes care of supporting the basic four VSTi standard digital signal process effects namely Attack, Delay, Sustain and Release given from number 7 to 10 below.

String Release module: This module controls and sets the degree of "release" working in cooperation with the "ADSR" module. With a slider supported with GUI, it increases the release from the value of [0.00 to 10.00] with a double precision.

Attack module: This module controls and sets the degree of "attack" working in cooperation with the "ADSR" module. With a slider supported with GUI, it increases the attack from the value of [0.00 to 10.00] with a double precision.

Decay module: This module controls and sets the degree of "decay" working in cooperation with the "ADSR" module. With a slider supported with GUI, it increases the decay from the value of [0.00 to 10.00] with a double precision.

Sustain module: This module controls and sets the degree of „sustain' working in cooperation with the "ADSR" module. With a slider supported with GUI, it increases the sustain from the value of [0.00 to 10.00] with a double precision.

Peak Meter: A LED Bar Graph style volume meter, calibrated in decibels. It uses colored LED bars; with green for normal volume (40dB-10dB), yellow for high volumes (10dB-5dB) and red for maximum volume (5dB-0dB).

VCA module: This module receives input from the "Oscillator" and "ADSR" modules and lets the DAW control the signal and the volume of the sound and sends it to the "Sound Out" module ***through IO mod*** for further process.

Sound Out module: This module receives the final output from the VCA through its left and right pins and handles the communication between the DAW's soundcard in order to handle playing the sounds.

GUI module: This module takes care of the simple GUI we managed to build for demo purposes. We have two GUI functions which are the "StringRelease" slider which helps to control the DSP process and the sound font loader button.

The procedural programming design and components are shown below on Figure 9.

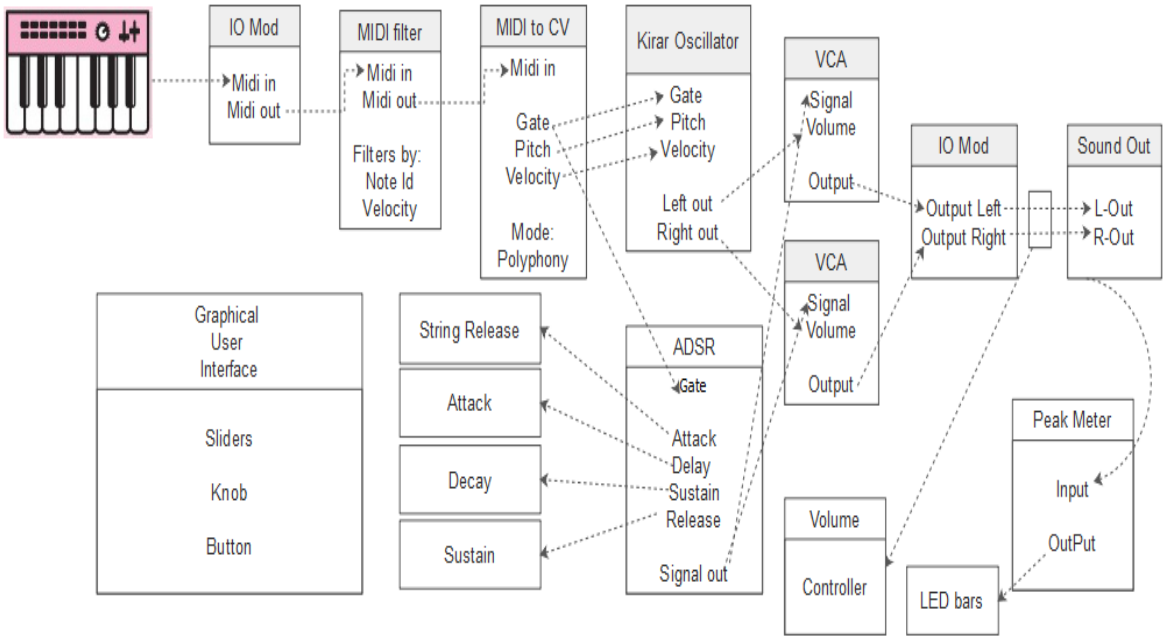

Figure 9: Procedural programming design and components diagram

Integration and Testing in VST Systems

To integrate the code and the samples, we used Sound Font file system architecture (.sf2). This was the biggest improvement we added on our initial research project plan.

What the Sound Font file system architecture does here is that it completely separates the coding of basic classes from the file system architecture which will later let us to improve the quality of WAV samples along with recording more of them without changing the programming classes, method and the code.

The tool we used for creating the Sound Font file system (.sf2) is Vienna programming language.

Using the results we got from the sampling process, we managed to create a single Sound Font (.sf2) file, which replaces and holds all 36 samples as one.

By classifying the making process of the plug-in into main modules, we managed to create 14 independent modules, which are put together to create a system which takes a Sound Font file as an input and gives an output of an oscillated audio stream.

We have also taken care of implementing the 3-level MIDI mapping which resulted in creating an intensity based volume creation for the plugin.

Creating an appealing genuine GUI was also implemented for the plug-in so that we can get an easy and interactive program platform.
Finally a window installer .exe file was created in order to simplify the hardship of installing the plug-in.
As shown in section 4.3 above, the plug-in has been tested with Cubase 5.x as well as FL Studio DAW platforms and it worked flawlessly and was recognized as VST instrument plug-in in these platforms.
It could also initiate its own GUI in these DAW platforms.

The plug-in also managed to work seamlessly with both mouse and computer keyboard controls of the GUI.
Our plugin managed to be successfully played, recorded, re-shuffled and quantized inside the DAW.
The 3-level MIDI mapping worked to initiate intensity based sound as we tried playing it using a professional MIDI controller keyboard.

CONCLUSIONS

We have successfully designed and implemented the plug-in for Kirar that can work in DAW platforms so that music composers can easily use the Ethiopian Kirar in their VST systems of soft music compositions.

We have collected samples for the third octave keys, made processing of audio signals and achieved the 3-level MIDI mapping. Our plug-in has been developed, integrated with Cubase 5.x and FL Studio DAW platforms and we have made a thorough test on the plug-in.

Our test result shows that our plug-in works flawlessly in these DAW platforms. We believe that our contribution in making the software models of our traditional music instruments is immense in delivering VSTi Plug-in for Kirar even though very much work is left undone in developing similar plug-ins for other Ethiopic traditional music instruments that our contribution in making the software models of our traditional music instruments is immense in delivering VSTi Plug-in for Kirar even though very much work is left undone in developing similar plug-ins for other Ethiopic traditional music instruments.

Future Works

Even though we have addressed several challenges, our design and development has some issues which are not still addressed. These limitations of the current research project can be interesting future research directions. Some of these limitations are:

Numbers of samples are still 36 which is a lot less when we compare it to professional VSTi plugin software platforms.

This is because of the lack of equipment and standard sound proof studios which can give us more precise and quality audios.

By increasing the number of samples, the plug-in spectral diversity and hence its effects can be improved.

The sound Font files loads all of the sampled audio files altogether on the RAM, which creates problems on computers with limited RAM resources.

So by taking the average of keys a kirar player could play at a time, we limited the polyphony of the plugin to 3 keys at a time. Which means it can only play 3 notes at a certain time. By

making optimizations of memory usage, all the keys can be played and used at a time.

REFERENCES

[1]. Steinberg. VST3: New Standard for Virtual Studio Technology. Steinberg (2013), http://www.steinberg.net/en/company/technologies/vst3.html (Accessed on: June 24, 2017)

[2]. Tanev, George, and Adrijan Božinovski. "Virtual Studio Technology inside Music Production." *ICT Innovations 2013*. Springer International Publishing, 2014. 231-241.

[3]. Hsu, Mu Hsen, Timothy K. Shih, and Jen Shiun Chiang. "Real-time finger tracking for virtual instruments." *Ubi-Media Computing and Workshops (UMEDIA), 2014 7th International Conference on*. IEEE, 2014.

[4]. Tanev, George, and Adrijan Bozhinovski. "Virtual Studio Technology and Its Application in Digital Music Production." (2013): 182-186.

[5]. DeVane, Charlie, and Gabriele Bunkheila. "Automatically Generating VST Plugins from MATLAB Code." *Audio Engineering Society Convention 140*. Audio Engineering Society, 2016.

[6]. Han, Jinseung. *Digitally Processed Music Creation (DPMC): Music composition approach utilizing music technology*. Diss. Teachers College, Columbia University, 2011.

[7]. Claude Borne. The Audio Plugins, Claude Borne (2003), http://www.macmusic.org/articles/view.php/lang/en/id/62/The-Audio-Plug-ins (Accessed on: June 27, 2017)

[8]. Hitsquad. Guitar Rig 5.2.0 for Windows XP/7/Vista. Hitsquad Pty Ltd. (2013), http://www.hitsquad.com/smm/programs/GuitarRigWin/screenshot.shtml (Accessed on: June 27, 2017)

[9]. Plectra Series 4 Plug-in for Turkish Oud, https://impactsoundworks.com/product/plectra-series-4-turkish-oud/#details (Accessed on: June 27, 2017)

[10]. Physics of Music, Frequencies for equal-tempered scale, http://www.phy.mtu.edu/~suits/notefreqs.html (Accessed on June 28, 2017)

[11]. Wikibooks. Sound Synthesis Theory, Oscillators and Wavetables - Wikibooks, open books for an open world, https://en.wikibooks.org/wiki/Sound_Synthesis_Theory/Oscillators_and_Wavetables (Accessed on: June 28, 2017)

[12]. Puckette, Miller. *The theory and technique of electronic music*. World Scientific Publishing Co Inc, 2007.

[13]. De Leon, Phillip L. "Computer Music in Undergraduate Digital Signal Processing." *American Society for Engineering Education/Gulf Southwestern Region (Las Cruces, NM.)* (2000).