Nepal Journal of Science and Technology Vol. 15, No.1 (2014) 89-98

Implementation of Audio Effect Generator in FPGA

Sujit Rokka Chhetri[1], Bikash Poudel[1], Sandesh Ghimire[2], Shaswot Shresthamali[2] and Dinesh Kumar Sharma[3]

[1]Department of Electronics and Communication, Thapathali Campus, Institute of Engineering, Kathmandu, Nepal

[2]Sagarmatha Engineering College, Institute of Engineering, Lalitpur, Nepal

[3]Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering, Lalitpur, Nepal

e-mail: sujitchhetri@gmail.com

Abstract

This paper describes the theory and implementation of audio effects such as echo, distortion and pitch-shift in Field Programmable Gate Array (FPGA). At first the mathematical formulation for generation of such effects is explained and then the algorithm is described for its implementation in FPGA using Very high speed integrated circuit hardware descriptive language (VHDL). The digital system being designed, which is synthesizable and reconfigurable, offers a great flexibility and scalability in designing and prototyping in FPGAs. The system is divided into three HDL blocks, each for echo, distortion, and pitch-shift effect generation, which are multiplexed in order to share the common ADC and DAC. The audio effect generator designed in this paper was successfully implemented in Spartan-3E FPGA utilizing the resources available effectively. There has been tremendous research being carried out in the field of IP core. Efficient IP cores designed to carry out digital signal processing are implemented in every modern device using configurable logics. This trend hasn't yet been realized in Nepal. Through the design and implementation of audio effect generator, this paper also aims at bringing the field of IP core development to limelight among scholars of Nepal.

Key words: audio processing, echo, distortion, pitch shift, FPGA.

Introduction

Audio effect generation is the process of artificially creating or enhancing sounds in order to emphasize artistic or other content of live performance, music, etc. The audio effect generator described in this paper is designed using VHDL and is implemented in Spartan-3E Starter Kit using Xilinx ISE 13.2. FPGA is a programmable chip where one can design and implement any kind of digital system ranging from simple system like multiplexer to a complex system like a processor core itself (Xilinx 2006). The various effects described in this paper are echo, distortion and pitch-shift. Echo effect is to simulate the effect of reverberation in a large hall or cavern. Generating the distortion effect involves clipping-off the peaks of the signal at both the peaks thus introducing the noise and creating the distortion in regular sound. Pitch-shift is generating by simply modulating the audio samples with samples of sine wave. Three VHDL modules or components are designed which take audio samples, performs audio processing, and produces output samples which are fed to the DAC to regenerate the audio signal with effect. The overall modules are controlled by a Finite State Machine (FSM) which determines the flow of data from one module to the other. The functional verification of each module is checked using ISIM simulator of Xilinx ISE 13.2.

Methodology

This section describes the mathematical formulation and implementation of digital signal processing algorithms to generate different effects described earlier.

Echo

Digital generation of echo is based on philosophy of storing input signal for significant time interval and mixing it with newly arrived signal. Here, echo effect is generated using the circular buffer.

Circular buffer

From Fig.1., data are originally written to the very first memory location. As new data comes in, they are written to the next available memory address. Once the address pointer reaches the end of the circular buffer, it immediately wraps around and starts writing to the first memory address again. Storing data in this manner, 2^n samples are always available in the buffer, and all that is necessary in controlling the pointers. This can be most easily done with a simple *n*-bit counter (Richard & Schultz 2007).

After generating delay, two outputs can be generated:
$y1\ [n] = x[n]$
$y2\ [n] = x[n - \Gamma]$, With Γ being a value corresponding to the time difference of the two output values.

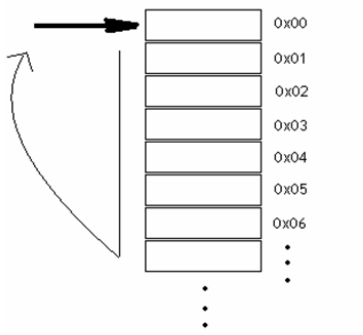

Fig.1. Circular buffer

Generation of echo

Now, since delay is successfully generated, echo is implemented by the following equation:

$$y_{echo} = \alpha * x[n] + \beta * x[n - \Gamma_{echo}] \quad (1)$$

where, α, β are real constants and Γ_{echo} is constant delay added to the original signal.
Essentially, original signal x[n] is scaled by some factor and added to the delayed, scaled version of original signal.

Algorithm for echo

1) In the rising edge of clock, receive the 14 bit digital data from ADC and store it in circular buffer.
2) Retrieve the data from the location which is delayed by Γ
3) Mix the scaled version of the present digital data from the ADC with the delayed data from the circular buffer.
4) Send the mixed data to DAC.

Distortion

Distortion effects create "warm", "dirty", and "fuzzy" sounds by compressing the peaks of a musical instrument's sound wave and adding overtones (Wikipedia 2012). Distortion effect has been created by clipping the original digital signal. The theory is based on the fact that clipping creates some high frequency components which are responsible for the fuzzy sound. The mathematical formulation and waveform follows next.

Assume an unclipped signal $x_u(t)$ defined as

$$x_u(t) = \cos(2\pi f_o t) \quad (2)$$

Taking Fourier transform,

$$Xu(f) = 0.5*\delta(f - f_o) + 0.5*\delta(f + f_o) \quad (3)$$

The time domain and frequency domain representation of signals are shown in Fig. 2.and Fig. 3. Now, the signal is clipped when amplitude is greater than B. Then the clipped signal can be defined as

$$x(t) = \begin{cases} B & if \quad \cos(2\pi f_0 t) \geq B \\ \cos(2\pi f_0 t) & otherwise \end{cases} \quad (4)$$

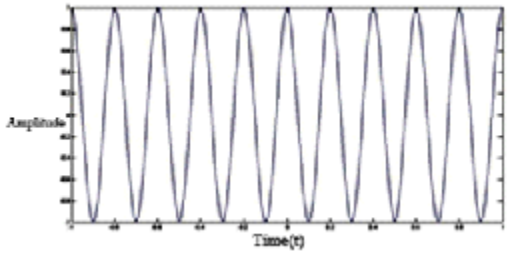

Fig.2. Time domain representation of xu(t)

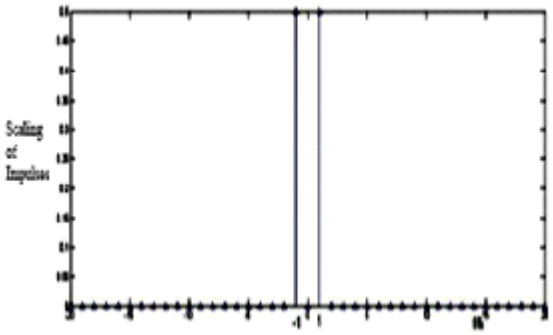

Fig. 3. Frequency spectrum of Xu(f/f0)

In order to calculate Fourier transform of this periodic infinitely extended signal, first, calculation of Fourier series coefficients a_k is done as

$$a_k = \left(\frac{1}{T}\right)\int_{-\frac{T}{2}}^{\frac{T}{2}} x(t) * e^{-j*k*2\pi f_0 t}\, dt \quad (5)$$

$$or, a_k = \left(\frac{2}{T}\right)\int_0^{tb} B * \cos(k * w_0 t)\, dt +$$

$$\left(\frac{2}{T}\right)\int_{tb}^{\frac{T}{2}} \cos(w_0 t) * \cos(k * w_0 t)\, dt$$

$$where, w_0 = 2\pi f_0$$

$$tb = (1/w_0) * cos^{-1}(B)$$

If k=0, then

$$a_k = \left(\frac{2}{T}\right)\int_0^{tb} B\, dt + \left(\frac{2}{T}\right)\int_{tb}^{\frac{T}{2}} \cos(w_0 t)\, dt$$

$$= \left(\frac{2}{T}\right) * B * tb - \left(\frac{2}{Tw_0}\right)\sin(w_0 tb) \quad (6)$$

If $k = 1\ or\ -1$,

$$a_k = \left(\frac{2}{T}\right)\int_0^{tb} B * \cos(w_0 t)\, dt +$$

$$\left(\frac{2}{T}\right)\int_{tb}^{\frac{T}{2}} \cos(w_0 t) * \cos(w_0 t)\, dt$$

$$= \left(\frac{2}{Tw_0}\right) * B * sin(k * w_0 tb) +$$

$$\left(\frac{1}{2Tw_0}\right)\left(2w_0\left(\frac{T}{2} - tb\right) - \sin(2w_0 tb)\right) \quad (7)$$

If k≠-1,1,0,then

$$a_k = \left(\frac{2}{T}\right)\int_0^{tb} B * \cos(k * w_0 t)\, dt$$

$$+\left(\frac{2}{T}\right)\int_{tb}^{\frac{T}{2}} \cos(w_0 t) * \cos(k * w_0 t)\, dt$$

On integration,

$$a_k = \left(\frac{B}{k\pi}\right) * \sin(k * w_0 tb) -$$

$$\left(\frac{1}{T}\right) * \left[\frac{\sin((k+1)w_0 tb)}{(k+1)w_0} + \frac{\sin((k-1)w_0 tb)}{(k-1)w_0}\right] \quad (8)$$

After the calculation of Fourier series coefficients, following relation to calculate Fourier transform can be used.

$$X(f) = \sum_{k=-\infty}^{\infty} a_k \delta(f - f_0) \quad (9)$$

Using relation (9) along with (6),(7) and (8), we obtain Fourier transform of clipped signals. The time domain and frequency domain plot of those signals are shown in Fig. 4.and Fig. 5.

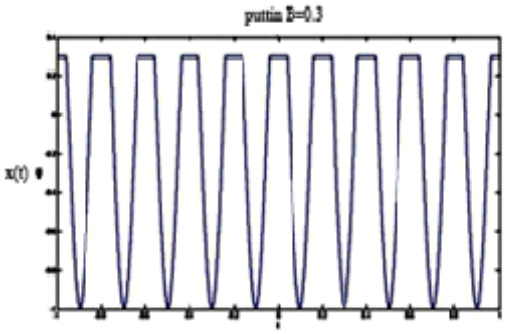

Fig.4. Time domain plot of clipped signal

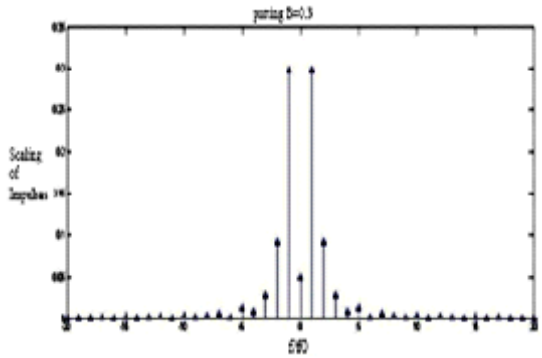

Fig. 5. Fourier transform of clipped signal (B=0.3)

The mathematics and waveform above shows that clipping of a signal in the time domain results in the spreading of frequency components in the frequency domain. In real application, single tone frequency is rarely used as input signal. However, any kind of signal can be thought as being composed of combination of different single tone frequencies scaled by some factor. Thus, each frequency component of input signal undergoes spreading as shown in Fig. 5.when the signal is clipped about some value. It is this spreading of frequency spectrum and introduction of higher components that creates coarse sound in audio input signals introducing distortion.

Algorithm for distortion

1. Set the lower and upper limits of clipping: clip_low and clip_high
2. In the rising edge of clock, take the input signal and store in x
3. If distortion_enable ='1'then goto(4) else, goto (5)
4. i. If x is less than clip_low then x_out = clip_low

else if x is greater than clip_high then x_out = clip_high.
elsex_out = x.
ii. Goto (5)
5. x_out = x,Goto (2)

Pitch shifting

A pitch shifter raises or lowers each note a performer plays by a pre-set interval. For example, a pitch shifter set to increase the pitch by a fourth will raise each note four diatonic intervals above the notes actually played. Pitch shifter is essentially a frequency shifter.In this paper, a frequency shifter of complementary nature has been designed. A complementary shifter doesn't shift all the frequency by same quantity neither in same ratio but in a complementary form. In the paper,any single tone signal of frequency f_{old} gets transformed into a new signal having frequency f_{new} given by

$f_{new} = f_0 - f_{old}$

Mathematical formulation

Assume an input audio signal x(t)= $2A\cos(2\pi wt)$, and let its Fourier transform be *X(f)*, which can be written symbolically as

$$x(t) \xleftrightarrow{Fourier transform} X(f)$$

Now, taking a carrier signal *c(t)* defined as

$$c(t) = \cos(2\pi f_0 t) \qquad (10)$$

Taking the Fourier transform of equation (10),

$C(f) = 0.5 * \delta(f - f_0) + 0.5 * \delta(f + f_0)$

Multiplying *x(t)* by *c(t)*,*y(t)*can be obtained as

$y(t) = c(t) * x(t)$

In frequency domain,

$Y(f) = X(f) \aleph C(f)$ where, $\aleph$ denotes convolution

$$or, \quad Y(f) = X(f) \aleph [0.5 * \delta(f - f_0) + 0.5 * \delta(f + f_0)]$$
$$= 0.5 * X(f - f_0) + 0.5 * X(f + f_0) \qquad (11)$$

After obtaining a modulated signal, it is passed through a low pass filter. Due to this, higher component is suppressed and only lower frequency component is retained. As shown in Fig.6., since *x(t)* has been taken as a single tone frequency signal, this lower frequency component will be nothing but a single tone frequency component having amplitude same as the input amplitude and frequency equal to $(f_0 - w)$.

Performing inverse Fourier transform of this signal, an output signal *y_out* is given by

$y\_out = A\cos(2\pi(f0 - w)t)$ (12)

Thus, a single tone input frequency results in single tone output signal with frequency shifted in a complementary fashion. If, however, input signal is not a single tone but possess a complex frequency spectrum, each component of frequency spectrum is shifted in complementary fashion and thus output frequency spectrum will be shifted version of input spectrum flipped around about y-axis.

It is important to mention that all the frequency domain operations of convolution and filtering are performed digitally in FPGA using DSP algorithms. In the mathematical formulation and the figure, analog to digital transformation and vice versa were not included for the purpose of simplicity and brevity. The figures and equations give the gist of technique at a glance, and, therefore, analog conventions are used to represent filter and carrier signals in frequency domain. However, these operations are actually performed digitally.

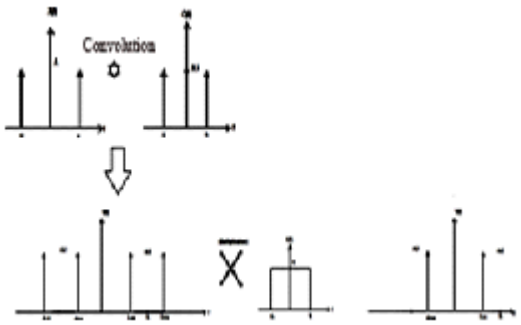

Fig.6. Frequency domain representation of operations used in pitch shifting

Carrier signal generation

The carrier signal used above *c(t)* is a continuous time signal. However, a digital signal $c_d[n]$ which corresponds to the signal *c(t)* in continuous time is generated. This transformation is related in time domain as

$$c_d[n] = c(nT) \qquad (13)$$

where, T= sampling period
fs=1/T= sampling frequency

Let $C_d(e^{j*2\pi f})$ be the Fourier transform of $c_d(n)$ and *C(f)* be the Fourier transform of *c(t)*, then from (11) it follows that

$C_d(e^{j*2\pi f}) = C(f/T) = C(f * f_s), -0.5 \leq f \leq 0.5$ (14)

Inherent in equation 14 is the fact that $C_d(e^{j\cdot 2\pi f})$ is periodic in f with period of 1. In other words, the frequency in digital domain cannot exceed unity. $C_d(e^{j\cdot 2\pi f})$ is unique in the frequency interval f=[0.5,0.5] and starts repeating after that. The frequency axis contracts by the factor of f_s when a continuous domain signal is converted into discrete by sampling at a frequency of f_s. This relation can be used to digitally construct a carrier signal similar to that of the analog domain. For example, let's suppose f_s=60 KHz and we want to shift an audio signal by 1 KHz i.e. were require a carrier frequency of 1 KHz .
Then, $f*f_s$=1 KHz
or, f=1kHz/f_s
or, f=(1/60) unit
Therefore, period of digital signal is N=1/f=60.
For the purpose of shift by 1 KHz, digital cosine carrier signal is given by

$$c_d[n] = \cos(2\pi n/60) \qquad (15)$$

Algorithm for modulation

The algorithm for the implementation of modulation is as follows:

1. Store the the values of digital carrier in an array x() (or a look up table), Counter=0
2. Input the data and convert into digital signal x_in(n)
3. If modulation is enabled goto (4) else, goto (5)
4.
   i. Find the product x(counter)*x_in(n)
   ii. Normalize the product of 4(i) and assign to x_out(n)
   iii. Increment the value of counter by 1.If the value exceeds maximum value, set counter=0
5. Goto (6)
6. x_out=x_in, Goto (2)

Filtering

As mentioned earlier, a digital low pass filter has been. Specifically, the low pass filter that has been used is an FIR (Finite Impulse Response) Kaiser window filter. An FIR with coefficients is an LTI digital filter. The output of an FIR of order or length L, to an input time-series x[n], is given by the convolution

$$y[n] = x[n] * h[n] = \sum_0^{L-1} x[k] \cdot h[n-k] \qquad (16)$$
$$\text{or, } y[n] = \sum_0^{L-1} h[k] \cdot x[n-k] \qquad (17)$$

where, *h[0]* through *h[L-1]* are non-zero filter coefficients.

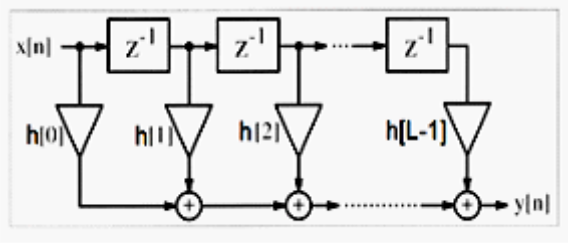

Fig. 7. Direct form I FIR filter

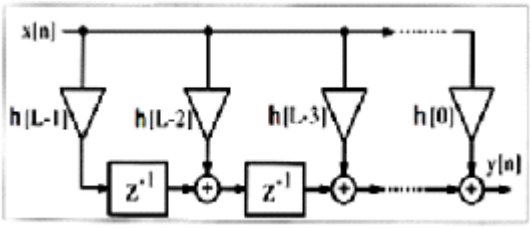

Fig. 8. FIR filter in transposed form

Fig. 7.and fig. 8.show direct and transposed form of FIR filter. Transposed structure is more popular for the implementation of FIR filters because it minimizes extra pipeline stages and shift registers (Meyer-Baese 2004). It can be obtained by using transposition theorem in DSP.

The transposed form of FIR filter has been used in this paper to construct low pass filter. The design of FIR low pass filter, there are two major steps:

Calculation of coefficients

The process of transformation of analog calculation into digital is useful in designing digital filters. If a low pass filter having cutoff frequency f_a, is required, then a digital filter of cutoff frequency f_a/f_s should be designed, where f_s is the sampling frequency of analog to digital conversion. Similarly, the fact that digital filter has a transfer function that is periodic with a period equal to one in frequency domain has to be considered while designing a low pass filter. The coefficients of the low pass filter were calculated as required by using MATLAB.After the calculation of coefficients i.e. *h[n]*, the transformed structure of Fig. 8. was used to implement the design in FPGA.

Algorithm of FIR filter implementation

This algorithm has two parallel processes P1 and P2. All the coefficients are stored in an array c[n],n=0 to L-1. Thus h[n] is stored in array c[n]. Each element in c[n] is also a standard logic vector.

1. In process P1, input signal x_in is multiplied with each coefficient c[i] to obtain product p[i], p[i]=x_in *c[i]
2. In process P2,
   i) a[L]=0
   ii) for i =0 to L-1, calculate a[i] by the relation
   iii) a[i]= a[i +1]+p[i]
   iv) end for Output signal is given by, x_out=a[0]
3. Steps (2), (3) and (4) are repeated if filter is enabled.

It is important to note that the output of FIR filter is obtained in only one cycle by the use of transposed structure and therefore filter can be serially introduced without significant increment in sampling frequency.

Algorithm for pitch shifting

The algorithm of pitch shift is similar to modulation in many ways. Given below is the algorithm for pitch shifting.

1. Store the the values of digital carrier $c_d[n]$ in an array x() (or look up table), Counter=0
2. Input the data and convert into digital signal x_in(n)
3. If pitch shift is enabled goto (4)
   i. Else, goto (5)
4.
   i. Find the product x(counter)*x_in(n)
   ii. Normalize the product of 4(i) and assign to x_temp(n)
   iii. Increment the value of counter by 1.If the value exceeds maximum value, set counter to 0
   iv. Assign the intermediate value x_temp to the input of filter
   v. The output of filter is assigned to output value x_out, Goto (5)
5. x_out=x_in, Goto 2

Implementation

This section proceeds to describe how the algorithms described have been implemented in FPGA using VHDL. The block diagram of arrangement of the components designed to implement the effects in FPGA is shown in Fig. 9.

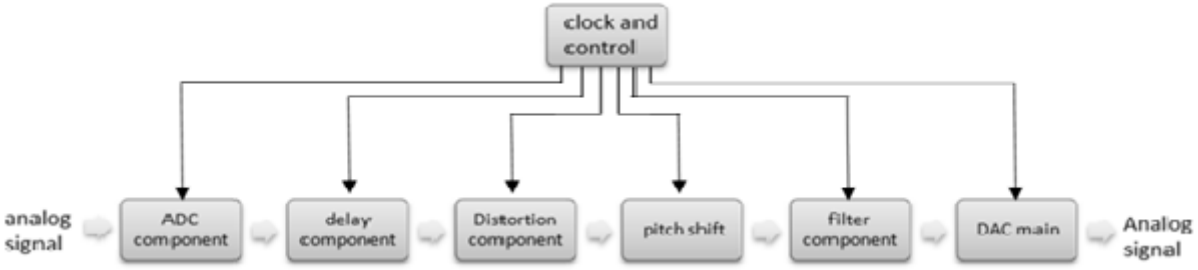

Fig. 9. Block diagram of arrangement of the all the components

The block diagram of the delay component can be shown in the Fig. 10.

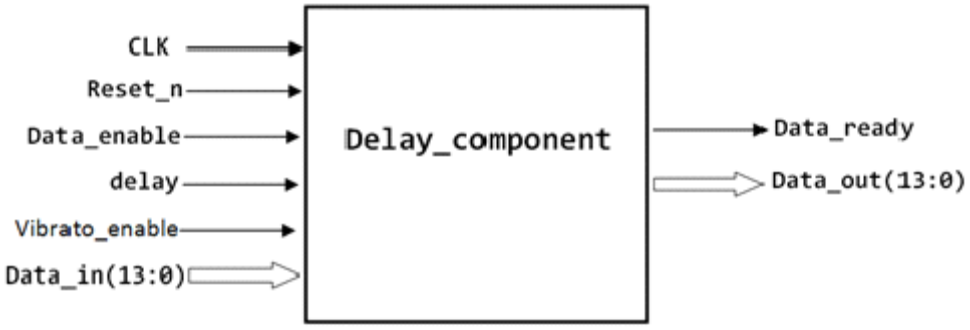

Fig. 10. Block diagram of delay_component

The component is implemented using VHDL. We define the input and output ports of the component under its entity.

The delay component will take 14 bit data and then store it in the cyclic ram. The output data is given on the basis of the control signal provided to it. The output data may or may not be delayed data. Delay component will work on the basis of a finite state machine. In order to store the 14 bit data, we've created a RAM component inside the delay component. The sequential programming of the finite state machine of the delay component is done under the process. Everything written inside the process will have sequential behavior. Inside the process there is a seven state finite cycle. If the delay component is enabled then only it will save the current data in the memory and retrieve the delayed data from the memory. If it is disabled then the original data is passed and the delay component is bypassed.

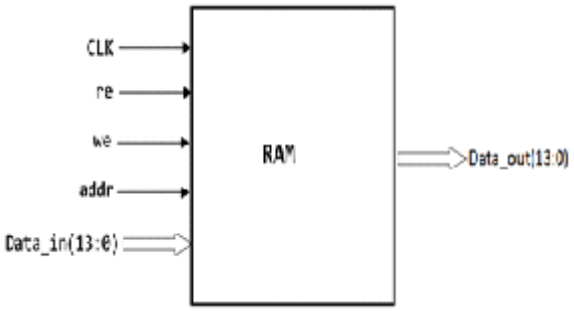

Fig.11. RAM component block diagram

Distortion component

It is responsible for taking the data and manipulating the data to produce the distortion effect. The block diagram of the component is shown in Fig. 12.

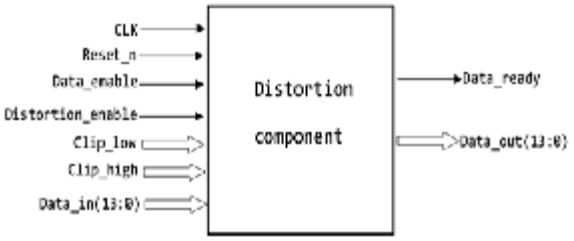

Fig. 12. Block diagram of distortion component

In order to filter the signal, to condition it, we've made a filter component. The block diagram of the filter component is given in Fig. 13.

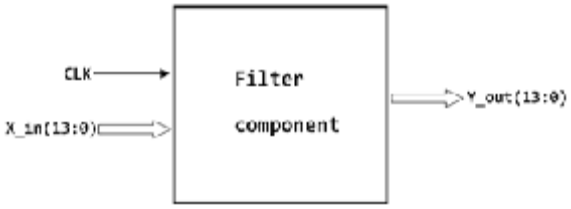

Fig. 13. Block diagram of filter component

Results and Discussion

A digital oscilloscope was used to observe the outputs in both time and frequency domain. This section deals with the analysis of those outputs.

Distortion

observed in oscilloscope

Theoretical output values:

Lower clipping value= 1.05 V corresponding to binary value 01101011100001

Higher clipping value= 1.45 V corresponding to binary value 10010100011110

Observed output values:

Lower clipping value = 1.03 V

Higher clipping value = 1.45 V

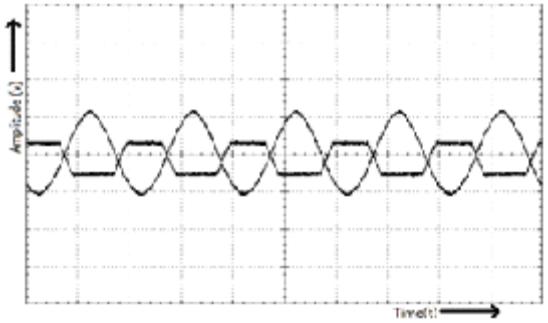

Fig. 14. Input sinusoid and distorted output signal as observed in oscilloscope

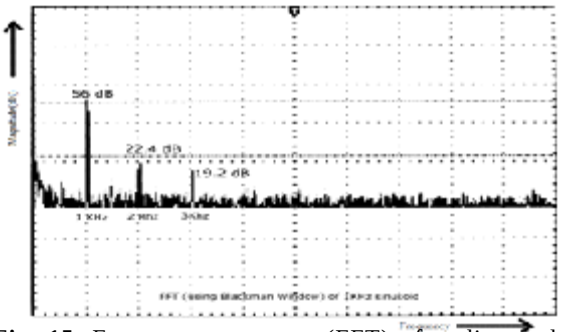

Fig. 15. Frequency spectrum (FFT) of undistorted sinusoidal signal

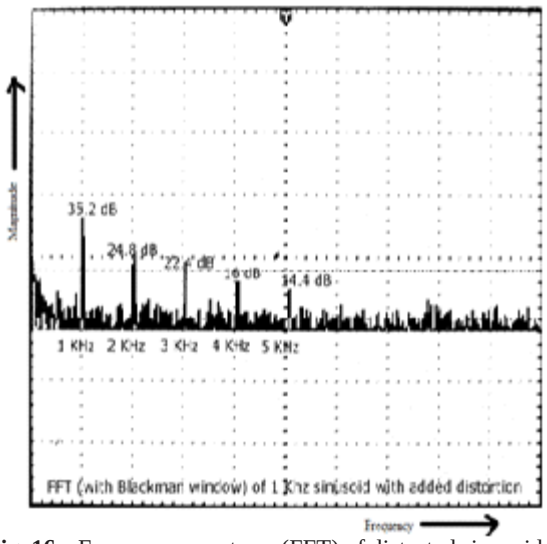

Fig. 16. Frequency spectrum (FFT) of distorted sinusoidal signal

Theoretically speaking, a single tone signal (undistorted) should have only one frequency component. Looking at Fig. 16., higher frequency components were also introduced too even in undistorted signal. This is because digital oscilloscope was used, which performed Fast Fourier Transform (FFT) of the digitized signal using a Blackman window. If finite number of signal in time domain is taken, convolution of actual frequency spectrum has to be performed with the frequency response of Blackman window. While doing this, higher frequency components were attenuated by the magnitude comparable to the attenuation of side lobes of Blackman window was obtained. The fundamental frequency is 1 kHz and corresponding to that frequency signal of 56 dB was obtained, but the signal of 2 kHz was attenuated by 33.6 dB and that of higher frequency was attenuated even more. On the other hand, theoretically the frequency response of clipped (distorted) signal must contain higher frequency components. In Fig. 16., frequency domain plot contained a number of higher frequency components with decreasing magnitude. One important distinction in frequency domain plot of undistorted and distorted signal is the difference in the magnitude of fundamental harmonics and higher harmonics. Unlike in the undistorted signal, the 2 kHz frequency component is only about 10 dB below the 1 kHz component. Such a high magnitude signal could have been introduced by the side lobe of Blackman window.

Modulation and pitch shift

The outputs for modulation and pitch shift effects were observed as plots of digital oscilloscope. Firstly, a sinusoid signal was taken, the frequency domain plot of which is shown in Fig.17. Internally a 1 KHz carrier signal was generated and modulated with an input signal of 180 Hz. The sampling frequency is 60 kHz and signal used is

$$c_d[n] = \cos(2\pi n/60)$$

As stated in earlier theory section, the analog equivalent signal corresponding to this carrier signal should be 1 KHz. Understanding of the theory and calculations was verified by the frequency domain plot of modulated signal in Fig. 19.When 1000 Hz signal is modulated by 180Hz signal, the output should be a signal containing frequency components 820Hz and 1180Hz as explained in earlier theory section.

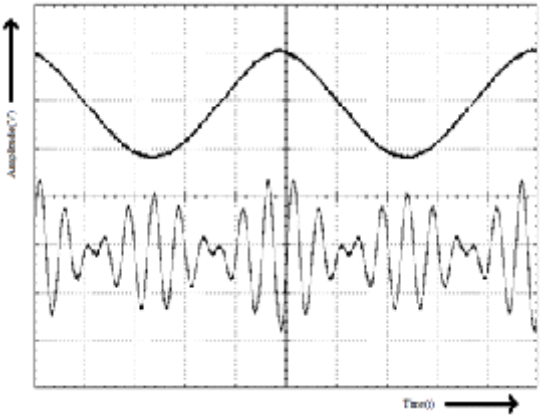

Fig. 17. Time domain modulated signal (carrier signal = 1 KHz, modulating signal = 180 Hz)

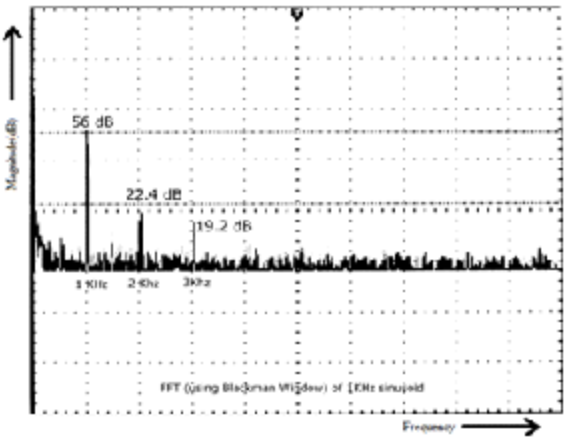

Fig. 18. Frequency spectrum (FFT) of input signal

From Fig.19., we see that frequency components are at 810 Hz and 1170 Hz which are very close to theoretically calculated values.

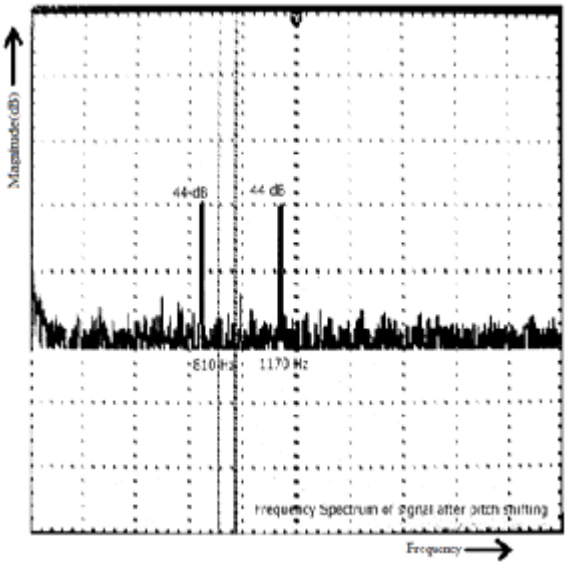

Fig.19. Frequency spectrum of 1 KHz signal modulated by 180 Hz

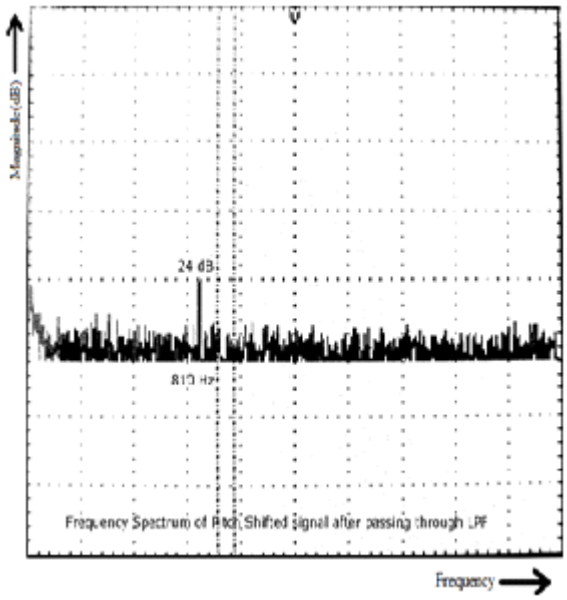

Fig. 20. Pitch shifted signal (180Hz signal shifted to 810Hz)

In order to isolate only the lower frequency of the two sidebands, a low pass filter was designed whose stopband is at 1KHz. However, an FIR filter with sharp transition band requires filter of very high order and that posed great difficulty in implementation. Therefore, constraint of sharp transition band was overlooked and a low pass filter having transition band starting from 500Hz to 1 KHz was designed. The side effect of widening transition band is observed. In Fig.20., it is seen that frequency component of 1170Hz is successfully filtered out, but it is found that 810Hz component is also attenuated. This is not a serious disadvantage, because the signal is higher than noise level and thus can be amplified to obtain clear sound.

Echo effect

The fundamental requirement for echo effect is delay generation. After implementing algorithm the observed data and analysis of are given as follows:

Theoretical calculation
Sampling frequency= 61.54 KHz
Sampling period = (1/61.54) ms = 16.25 μs
Number of cycles delayed= 14900
Delay = 14900*16.25 μs =242.125 ms

Table 1. Observed Delays

Frequency of single tone signal in Hz (f)	Observed effective delay (O) (in ms)	N	Observed actual delay, (N*1000/f) + O (in ms)
0.63	240	0	240
92	3	22	242.13
292.7	0.4	71	242.97
1000	0.163	242	242.163
5450	0.01	1319	242.02
11600	0.03	2807	242.01
21560	0.002	5217	241.998

Comparing the theoretical values and the observed output delays, the delays are in close agreement to the theoretical delay. There is, however, one important feature that needs elaboration. As the frequency of signal increases, the time period decreases. When the time period is less than the expected delay, then there will be confusion about the peak which is to be compared with original signal so as to calculate the delay. Multiple cycles already pass between a peak point in original signal and the peak point in its delayed signal. In such cases if delay between nearest peaks in input and output waveforms is calculated, then it would be an observed delay but not the actual perceived delay. The number of complete cycles that

has already passed is denoted by N in above table. With this correction of delay, observed actual delay has been calculated in the above table.

Delay component

Device utilization summary:
Number of Slices: 136 out of 4656
Number of IOs: 65
Number of BRAMs: 14 out of 20

Distortion component

Number of Slices: 28 out of 4656
Number of IOs: 61

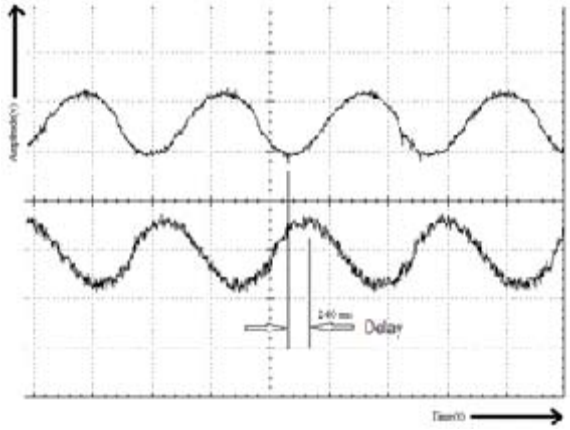

Fig. 21. Input and output time domain waveforms with delay

Filter component

Number of Slices: 1327 out of 4656
Number of IOs: 29
Number of MULT18X18SIOs: 16 out of 20

Since the Audio effect generator is synthesizable and reconfigurable digital system, one can upgrade the system by incorporating number of other various effects in the current code. Additional effects as flanger, phaser, chorus, equalization, filtering, resonator, robotic voice generator, etc can be easily added in the present system thus making this audio generator scalable and easy to implement.

References

Meyer-Baese. U. 2004. *Digital signal processing with field programming gate arrays.* Springer. 81 pp.

Schultz, R. 2007. *FPGA implementation of audio effects - An EE 552 Student Application Note.* pp. 1-2.

Serrano, J. 2008. Digital signal processing using field programmable gate arrays. In: *13th BEAM instrumentation workshop* (May 4-8, 2008), Lake Tahoe, California. pp 29-38.

Wikipedia. 2012. Distortion(Music).http://en.wikipedia.org/wiki/Distortion_(music).

Xilinx. 2006. Spartan-3E starter kit board user guide, UG230 (v1.0).