FPGA Design of Speech Compression by Using Discrete Wavelet Transform

J. Pang, S. Chauhan

***Abstract*—This paper presents the Discrete Wavelet Transform (DWT) for real-world speech compression design by using the Field Programmable Gate Array (FPGA) device. Compared with many reports which only focused on the DWT architecture design, this paper gives the speech compression system design detail on how to interface DWT block with SDRAM and audio codec chip. Speech compression was achieved by keeping only the approximation part of the DWT result. The compressed speech signal was read back after upsampling was performed. The resulting compressed speech can be heard clearly with some introduced background noise Future work is to reduce noise.**

***Index Terms*—Audio codec, discrete wavelet, FPGA, speech compression.**

I. Introduction

Most digital audio signals used today are quantized by 16 bits per sample. The CD standard digital audio signals use a sampling frequency of 44.1 KHz and the studio audio signals use a sampling frequency of 48 KHz [1]. In order to reduce the amount of storage for some audio applications especially for portable audio devices, it is necessary to compress the audio data.

The wavelet transform becomes an emerging signal processing technique and it is used to decompose and reconstruct non-stationary signals efficiently. The speech signal is non-periodic and it varies temporally. The wavelet transform can be used to represent speech signals by using the translated and scaled mother wavelets, which are capable to provide multi-resolution of the speech signal. This property of wavelet can be used to compress the speech signal.

The DWT consists of banks of low pass filters, high pass filters and down sampling units. Half of the filter convolution results are discarded because of the down sampling at each DWT decomposition stage. Only the approximation part of the DWT wavelet results is kept so that the number of samples is reduced by half. The level of decomposition is limited by the distortion tolerable from the resulting speech signal.

Several VLSI designs of the discrete wavelet transform have been reported for the discrete wavelet transform implementations [2]-[4]. Custom VLSI circuit implementations of the discrete wavelet transform are costly and time consuming. Conventional audio systems use DSP devices as design platform. The DSP device is usually a general-purpose, fixed-architecture technology. The DSP processor is not capable of configuring the data path width and it may waste hardware resources. Besides, it needs long development cycle, and it may consume more power.

As a low cost fast prototyping tool, FPGA design offers flexibility of re-programmability. The FPGA device allows easy configuration of the design data path, easy modification of the design architecture and also provides high performance at the same time.

Many DWT FPGA designs have been proposed by previous researchers [5]-[7]. Most of them focus on the DWT core design structure. However, this paper concentrates on the whole speech compression system design which interfaces with the real world speech signals and embeds the DWT design in the whole system. This design work was implemented on Altera Cyclone II FPGA device EP2C35F672 [8] by using the Altera DE2 kit [9]. In the whole digital design system, the analog speech signal streams are converted into digital samples through the audio codec chip WM8731 [10], and then stored inside the SDRAM chip [11] on board. Next, the Cyclone II device can read the digital samples from the SDRAM, do the DWT compression, and write the results back into the SDRAM chip. If users want to hear the result, the compressed speech data stored inside the SDRAM chip can be up sampled, and then sent to the audio codec chip to be converted into the analog signal to drive the speaker.

This paper is organized as follows. Part 2 gives an introduction to the basic wavelet computation. Part 3 discusses how to interface SDRAM, audio codec and DWT. Part 4 gives FPGA implementation detail. Part 5 compares several implementation results. Part 6 draws conclusion of the whole design.

II. Discrete Wavelet Transform

The discrete wavelet transform uses translated and scaled mother wavelets as a set of basis functions to represent a signal. The translation factor shifts the original signal in the time domain and the scale factor determines the frequency band. As a result, the discrete wavelet transform gives time-frequency joint representations of the original signal. The discrete wavelet transform is a mathematical function that involves multiplication and addition operation to convolve signal of interest with predefined wavelet

Manuscript received July 30, 2008.

Jing Pang is assistant professor in the Department of Electrical and Electronic Engineering at California State University, Sacramento, Sacramento, CA 95826 USA (phone: 916-278-4549; fax: 916-278-7215; e-mail: pangj@gaia.ecs.csus.edu).

Shitalben Chauhan is with California State University, Sacramento, Sacramento, CA 95826 USA (e-mail: svc34@saclink.csus.edu).

expansion coefficients. It can be represented by low pass and high pass filter banks. The example described below gives more insight to above explanation.

The digital speech signal X[n] is filtered by high pass filter H_1 (z) and low pass filter H_0 (z). The filtered results are down sampled by 2. Since most of the speech energy concentrates on the low frequency band, the low pass filtered signals need to be split again into sub bands by applying the H_1 (z) and H_0 (z) filters. This procedure repeats until the desired decomposition level is reached. At high frequencies, the DWT provides good time resolution and poor frequency resolution. At low frequencies, DWT gives good frequency resolution and poor time resolution.

The reconstruction of the original signal is the inverse discrete wavelet transform (IDWT) process and it needs reconstruction filters $G_0(z)$ and $G_1(z)$. When perfect reconstruction condition is satisfied [2], the reconstructed signal Y[n] will be same with the original signal X[n]. Fig. 1 shows three levels of discrete wavelet decomposition and reconstruction tree.

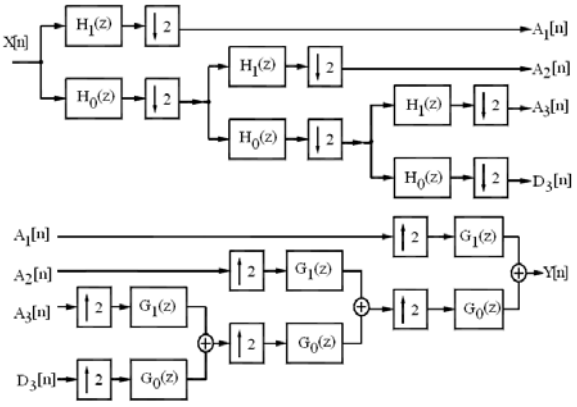

Figure 1. Wavelet Decomposition and Reconstruction

Daubechies wavelets are compact orthogonal filter banks which satisfy the perfect reconstruction condition. In addition, Daubechies wavelets have maximum number of vanishing moments for a given order so that they can be used to provide the good approximation of the original signal. The Daubechies 4-tap (db4) orthogonal filter bank was chosen for this design work.

The coefficients of the db4 filters are shown in the following table.

Table 1. Daubechies 4-tap (db4) wavelet coefficients

Low Pass Filter Coefficients	High Pass Filter Coefficients
a0 = 0.4829629	b0= 0.1294095
a1 = 0.8365163	b1= 0.2241439
a2 = 0.2241439	b2= -0.8365163
a3 = -0.1294095	b3= 0.4829629

For the sake of hardware implementation, these coefficients need to be quantized into binary digits which may result in loss of accuracy. The accuracy is measured in terms of the signal to noise ratio of the reconstructed signal. Therefore, the quantization should be done appropriately so that the quality of the wavelet transform is not compromised.

Matlab simulation shows up to 5-level of speech compression results are still recognizable when floating point db4 coefficients are used. Fig. 2 shows 3-level speech compression by using the db4 DWT which gives a compression ratio of 8:1, and the compressed sound can be heard very clearly.

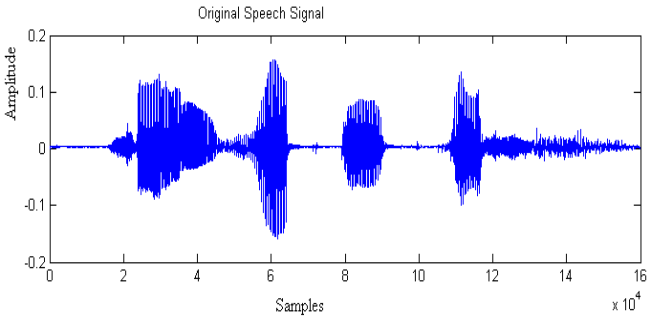

Figure 2. (a) Original speech waveform

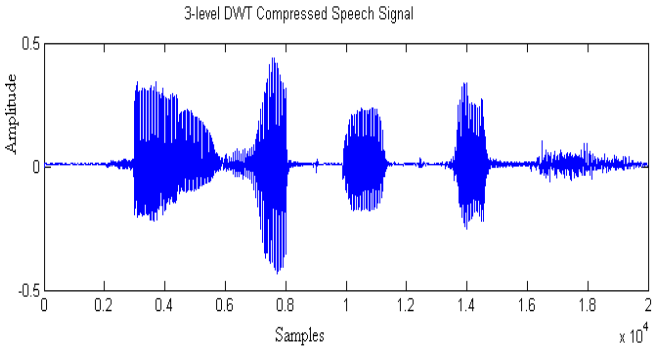

Figure 2. (b) 3-level DWT compressed speech waveform

III. Digital Speech Signal Collection

The Wolfson WM8731 24-bit sigma-delta audio codec is used to convert audio signal between the analog format and the digital format. WM8731 audio codec on DE2 kit has two stereo 24-bit sigma delta ADCs and two stereo 24-bit DACs with built-in oversampling digital interpolation and decimation filters.

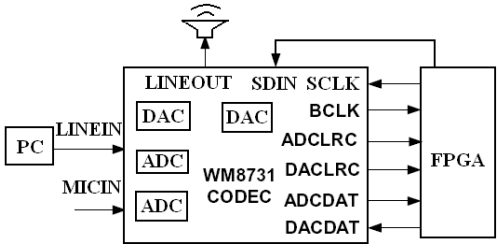

Figure 3. WM8731 interface with FPGA

FPGA is used to configure the internal registers of the WM8731 chip. Fig. 3 shows that PC provides audio speech output signal to WM8731 chip through the **LINEIN** port. Microphone signal can also be used as input **MICIN** to the codec. In master mode, WM8731 generates **BCLK**, **ADCLRC**, **DACLRC**, **ADCDAT** signals and sends them to FPGA. **BCLK** is the digital audio bit clock signal. **ADCLRC** is the left/right channel sampling clock which is used by the WM8731 internal ADC. **DACLRC** is the left/right channel sampling clock which is used by the WM8731 internal DAC. ADC converts analog signal from **LINEIN** or from **MICIN**

into serial digital data **ADCDAT**. Two ADCs are used, one for left channel and one for right channel. **DACDAT** is the digital data sent from FPGA to the WM8731 chip for digital-to-analog conversion. And the final converted analog output is sent out through the **LINEOUT** port of the WM8731 chip to the speaker. Different mode and frequency setup for the WM8731 codec are configured by the 2-wire **SDIN** and **SCLK** signals sent from FPGA. **SDIN** is serial data and **SCLK** is the clock used for serial data transfer.

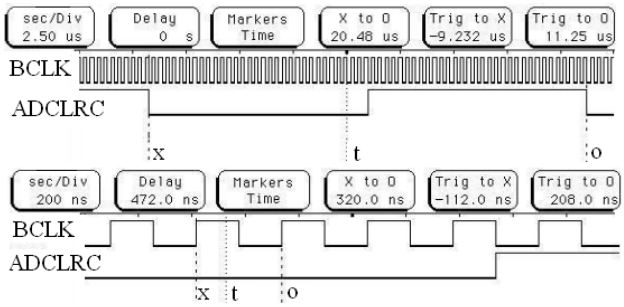

Figure 4. WM8731 output BCLK and ADCLRC signals

Fig. 4 shows the logic analyzer measurement of the I^2S mode **BCLK** and **ADCLRC** signals. The period of the **BCLK** is 320 ns. In I^2S mode, the valid ADC serial output audio digital data of left channel is available one **BCLK** clock cycle later after the **ADCLRC** signal changes value from logic '1' to logic '0'. The serial digital audio data is right justified.

The internal registers of the WM8731 are configured by FPGA so that the WM8731 works at about 48.8 KHz sampling rate and each digital audio sample has 16-bits.

In fig. 5, **wr** signal, **addr** signal and **dbus** signal are the write control, the address bus and the data bus for the synchronous DRAM (SDRAM) memory banks on DE2 board. This design only collects left channel audio data to save the amount of data storage. **count** signal is used to record 32 clock cycles after the **ADCLRC** signal changes value to logic '0'. Only the first 16 clock cycles are used to collect the valid left channel audio data. **tmp** represents 32-bit registers. **tmp** signal is used to convert the left channel right justified serial audio data stream **ADCDAT** into the parallel format.

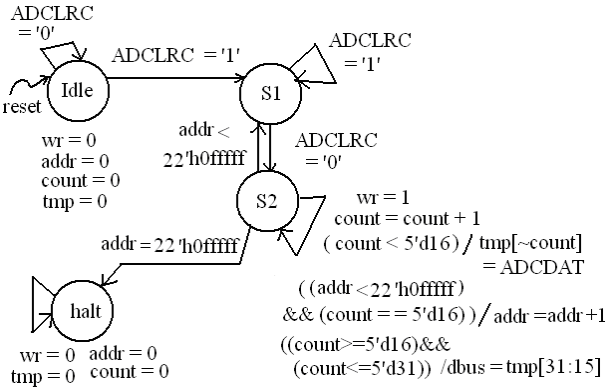

Figure 5. ADC left channel data collection state diagram

DE2 board has 8M byte SDRAM organized in four memory banks, and each bank uses 2M byte. Table 2 shows the name and functionality of each SDRAM pin.

Table 2. SDRAM pins and functionality

Signal Name	Functionality
zs_ba	bank address selection
zs_cas_n	column address strobe
zs_cke	clock enable
zs_cs_n	chip select
zs_dq	data
zs_dqm	data mask
zs_ras_n	row address strobe
zs_we_n	write enable
sdram_clock	clock signal

This design work used the Altera IP core for SDRAM controller. The instantiation of the SDRAM controller is shown below:

```
sdram_0  inst1(
// inputs:
.az_addr(addr1), .az_be_n(2'b00), .az_cs(1'b0),
.az_data(data1),  .az_rd_n(rd_n), .az_wr_n(wr_n),
.clk_50Mhz(clk_50Mhz),          .start(start),
// outputs:
.zs_ba(zs_ba),  .zs_cas_n(zs_cas_n), .zs_cke(zs_cke),
.zs_cs_n(zs_cs_n), .zs_dq(zs_dq), .zs_dqm(zs_dqm),
.zs_ras_n(zs_ras_n), .zs_we_n(zs_we_n),
.sdram_clock(sdram_clock)  );
```

The user address is represented by signal **addr1**. **rd_n** and **wr_n** signals are low active and they are used to control the SDRAM reading or SDRAM writing process. When **rd_n**='0' and **wr_n**='1', signal **data1** is written into logical address **addr1** location of the SDRAM. When **rd_n**='1' and **wr_n**='0', **zs_dq** is the data read out of the SDRAM. The major task of the SDRAM controller is to convert the logical address to the physical address, set appropriate row, column strobe and data mask values, precharge and refresh the SDRAM data stored on capacitors. User can set **rd_n** and **wr_n** values easily to write data into the SDRAM or to read data from the SDRAM.

The SDRAM on DE2 board can work at very high clock frequency up to 133 MHz. In this design work, SDRAM clock frequency is 50 MHz obtained from the on board oscillator.

IV. FPGA Implementation

A. The Overall Digital System

Fig. 6 shows the whole FPGA speech wavelet compression digital design system.

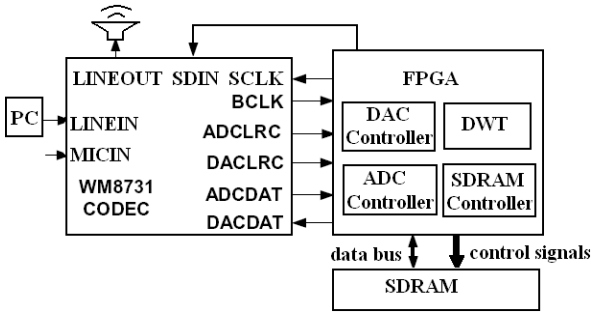

Figure 6. Speech wavelet compress system design block

In fig. 6, FPGA design has four major design blocks, including DAC controller, ADC controller, SDRAM controller and DWT block.

First, PC plays speech signal. At the same time, FPGA ADC controller starts sampling the **ADCDAT** signal sent from the WM8731 codec, and stores each left channel sample into the SDRAM chip. This design work collected more than 10 seconds of speech samples and stored them inside the SDRAM. Each speech sample was converted into 2-byte digital data. The sampling frequency of the WM8731 was configured as 48.8 KHz. It can also be configured as 44.1 KHz.

B. DWT Coefficient Quantization

Daubechies 4 (db4) wavelet [12] was chosen for speech compression in this design work. Db4 coefficients are floating numbers. In order to simplify the digital design, db4 coefficients need to be scaled up and truncated into integers.

Table 3. db4 wavelet coefficients quantization result

cb4 coefficients	Scale up by 2^3	Scale up by 2^8	Scale up by 2^9
a0 = 0.4829629	a0 =4	a0=124	a0=247
a1 = 0.8365163	a1=7	a1=214	a1=428
a2 = 0.2241439	a2=2	a2=57	a2=115
a3 = -0.1294095	a3=-1	a3=-33	a3=-66

When the db4 wavelet coefficients are scaled up by 8, the filter bank convolution results should be shifted right 3 bits to be scaled down by 8. Fig. 7 shows the convolution of db4 low pass filter with the ADC speech sample data **d_in**. Since the approximation part of the DWT results are kept for speech compression purpose, only the DWT low pass filter part is used for speech multi-resolution compression.

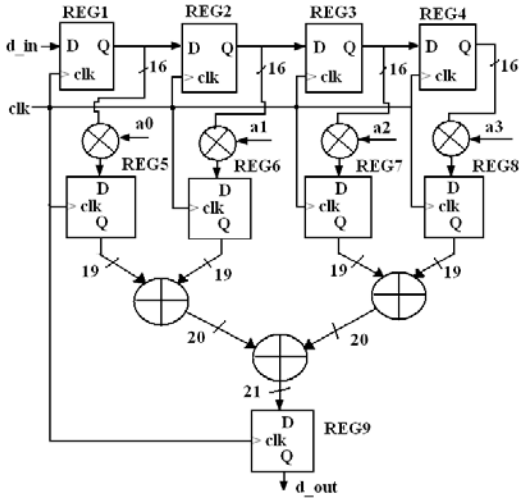

Figure 7. Convolution with db4 low pass filter by using the 8-times scaled up db4 quantized coefficients

In fig. 7, **d_in** signal is signed 16-bit data. a0, a1, a2, a3 are db4 coefficients scaled up by 8. So a0, a1, a2, and a3 are 4-bit signed data. REG1, REG2, REG3, and REG4 are 16-bit registers. REG5, REG6, REG7, and REG8 are 19-bit registers. REG9 represents 21-bit registers. Ripple carry adder structure is used for the adder stages. And signed multipliers are used for the multiplication units. The final result **d_out** needs to be shifted right by 3-bits before it is written into the SDRAM. {d_out[20], d_out[17: 3]} is the final 16-bit data which is written into the SDRAM.

In addition, the signal **flag** is used to control the down sampling of the **d_out** signal by 2. The 1-bit flag value is flipped every clock cycle. When it has value '0', the SDRAM write control signal is turned on to write the scaled 16-bit DWT result into the SDRAM. And the SDRAM address is increased by 1. Otherwise, the SDRAM write control signal is off, the SDRAM address is not changed and the DWT result is discarded.

When scaling the db4 coefficients up by 2^8, or 2^9, wider size multipliers, adders and registers need to be used. To test these three scaling algorithms, three designs were downloaded on the FPGA DE2 board. It turned out the scaling factors from 2^3 to 2^9 did not affect the hearing quality of the compressed speech that much.

C. DAC Conversion

When three levels of DWT are performed on the speech signal, the original signals are compressed 8 times and they are stored inside SDRAM. Since the WM8731 chip is set to work on the Master mode, the sampling frequency for DAC is the same with the ADC. Both used about 48.8 KHz sampling frequency in this design work. When the compressed speech data is read out of the SDRAM chip, each compressed sample needs to be sent to the DAC chip for 8 clock cycles.

Fig. 8 illustrates how to read compressed data samples stored inside the SDRAM chip and convert them into analog signals to be played by speaker.

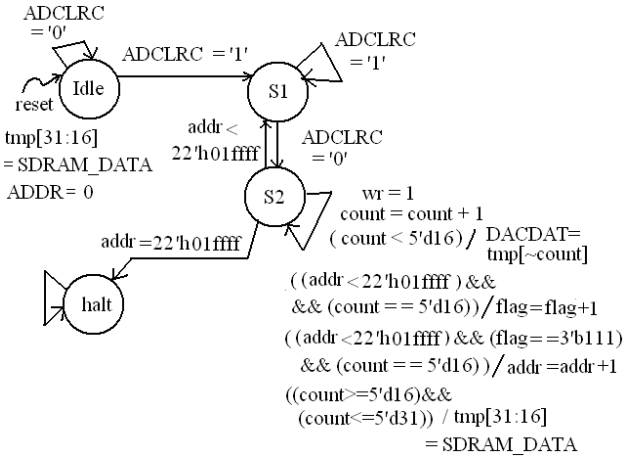

Figure 8. DAC left channel data conversion state diagram

In fig. 8, the **DACDAT** signal converts the parallel SDRAM data into right justified serial format for 16 clock cycles after **ADCLRC** signal has a transition to logic '0'. And another 16 clock cycles are used for the **tmp** signal to hold the parallel SDRAM data. The signal flag is used to control sending the SDRAM data from the same address 8 times before the SDRAM address is increased by 1.

The original speech samples are stored inside SDRAM from address 0 to address 22'h0fffff. The compressed speech samples are stored inside SDRAM from address 0 to address 22'h01ffff. The speech signals are compressed by 8 times.

V. Result Discussion

Table 4 gives a summary of the hardware resource consumption result for the whole speech compression FPGA digital system design. Here the db4 coefficients are scaled up by 8. DE2 kit uses Altera Cyclone II EP2C35F672 FPGA chip.

Table 4. Hardware Resource Consumption

	Availabl e	Used	Percentage of Use
Logic Elements	33216	985	985/33216 =3%
Combinational functions	925	33216	925/33216 =3%
Dedicated logic registers	648	33216	648/33216 =2%
Embedded Multiplier 9-bit elements	70	0	0%

Table 4 shows the overall hardware consumption is about 3%. The DE2 kit has 50 MHz on board oscillator, and it is used as system clock in this design work. DWT block needs to read speech data from SDRAM, do DWT compression and then write results back into SDRAM. 500 KHz clock works very well for this process.

The compressed speech data samples sounds very clear after they are read back from the SDRAM with the up sampling rate of 8:1. The background noise can also be heard although the speech sound is much higher above the noise level. The db4 coefficients quantization is one source of the noise generation. From our hearing testing results, the more accurate quantization did not greatly reduce the background noise. Another source of noise generation is the final up sampling process. Digital filter design should be included before sending the up-sampled data out for digital to analog conversion.

Theoretically, this compression can continue up to 5 levels which will give the compression ratio of 32:1. Matlab simulation shows the final compressed data still gives recognizable speech sound. Due to the background noise introduced in the hardware design, this design work only implemented 3-level compression. However, the same design structure can be used to easily implement 5-level compression in FPGA. The future research effort is to implement noise reducing blocks inside FPGA speech compression design.

VI. Conclusions

This paper presents FPGA design architecture for speech compression by using discrete wavelet transform. Db4 wavelet was chosen in this design work. The speech compression was achieved by keeping only the approximation components of each DWT process and throwing away the high pass filtered components. This can be done because most of the speech energy concentrates on the low frequency part. This design work successfully interfaced with the SDRAM chip, and the WM8731 codec chip. The compressed speech signal was read back after upsampling was performed. 3-level DWT filter banks were implemented and the resulting compressed speech could be heard clearly. At the same time, some background noise was introduced in the final data. Future work is to reduce noise.

References

[1] M. Raad, A. Mertins, and I. Burnett, "Audio compression using the MLT and SPIHT," *Proc. of DSPCS'02*, 2002, pp. 128–132.

[2] P. Y Chen, "VLSI implementation for one-dimensional multilevel lifting-based wavelet transform," *IEEE Trans. on Computers*, vol. 53, no. 4, April 2004, pp. 386-398.

[3] S. Zhuang, J. Carlsson, W. Li, K. Palmkvist, L. Wanhammar, "GALS based approach to the implementation of the DWT filter bank," *Proc. 7th International Conf. on Signal Processing, ICSP'04*, vol. 1, 2004, pp. 567 – 570.

[4] S. Huang, L. Dung, "VLSI implementation for MAC-level DWT architecture," *Proc. IEEE Computer Society Annual Symposium on VLSI*, 2002, pp. 92 – 97.

[5] J.Chilo, T. Lindblad, "Hardware implementation of 1D wavelet transform on an FPGA for infrasound signal classification," *IEEE Trans. on Nuclear Science*, vol. 55, Issue 1, Part 1, 2008, pp. 9 – 13.

[6] T. Sung, H. Hsin; Y. Shieh, C. Yu, "Low-Power multiplierless 2-D DWT and IDWT architectures using 4-tap daubechies filters," *7th International Conf. on Parallel and Distributed Computing, Applications and Technologies (PDCAT'06)*, 2006, pp. 185–190.

[7] W. Fan, Y. Gao, "FPGA design of fast lifting wavelet transform," *2008 Congress on Image and Signal Processing (CISP'08)*, vol. 4, 2008, pp. 362 – 365.

[8] Altera Corporation , "Cyclone II architecture," [Online], pp. 2-1 – 2-62. Available: http://www.altera.com/literature/hb/cyc2/cyc2_cii51002.pdf

[9] Altera corporation, "DE2 development and education board user manual", *Product Datasheet*, 2006, pp. 1– 69.

[10] Wolfsonj Microelectronics Plc. , "WM8731/WM8731L portable internet audio codec with headphone driver and programmable sample rates," *Product Datasheet*, 2004, pp. 1– 59.

[11] Integrated Circuit Solution Inc., "IS42S8800 / IS42S8800L / IS42S16400 / IS4216400L 2(1) M words x 8(16) bits x 4 banks (64-Mbit) synchronous dynamic RAM," *Product Datasheet DR007-0A*, pp. 1-68.

[12] M. I. Mahmoud, M. I. M. Dessouky, S. Deyab, and F. H. Elfouly, "Comparison between haar and daubechies wavelet transformions on FPGA technology, " Proceedings Of World Academy Of Science, Engineering And Technology, Vol. 20 April 2007 ISSN 1307-6884, pp. 68 – 72.