Design of Feature Extraction Circuit for Speech Recognition Applications

Saambhavi.V.B.[*], S.S.S.P.Rao[†] and P.Rajalakshmi[‡]
[*†‡]Indian Institute of Technology Hyderabad
[*]Email: ee10m09@iith.ac.in
[†]Email: sssp.rao@cmcltd.com
[‡]Email: raji@iith.ac.in

***Abstract*—This paper presents a hardware-software co-design implementation of feature extraction circuit which can be used for speech recognition applications. Mel-frequency cepstral coefficients are used to represent the features of the speech. A comparison between a complete software implementation and a co-design with both hardware and software components is brought out for the same circuit. The advantage of the hardware-software co-design is brought out by showing that the delay of execution has decreased to 0.0184 seconds from 17.29 seconds for the complete software implementation approach.The MicroBlaze soft-core processor from Xilinx is used in the hardware-software co-design. The processor frequency is chosen to be 66.67MHz. The Xilinx EDK software is used to design the circuit. The entire work is implemented on Atlys Spartan-6 development board.**

I. Introduction

Feature extraction is the process of taking out linguistic information from an uttered speech signal for utilizing in recognition. Short sections of the speech signal are isolated and given for processing. This processing is repeated for the entire duration of the waveform. The result of this operation is a new sequence of features along the time axis, representing the speech signal [1]. Mel-scale frequency cepstral coefficients (MFCC) are the most frequently used for speech recognition. This is because MFCCs considers observation sensitivity of human ear at different frequencies, and hence, is appropriate for speech recognition.

Feature Extraction plays a major part in the speech recognition algorithm. The proficient implementation of the design for feature extraction leads to efficient calculation of the speech features. Mostly its implementation is achieved as complete software implementations in literature. In [9], MFCC computation is implemented fully in software and it amounts to 22.82% (0.34 sec) of the whole decoding time required to recognize 2.515 seconds of speech at 120MHz. Similarly, [10] presents a mid-sized vocabulary system implementing the MFCC features extraction step in software. MFCC calculation takes 10% of the total computation load in their implementation, which is a significant percentage. Although the pure software based approach is easier to implement and deploy, the time taken by the software routines to do the complex digital speech signal processing is high. Such high computation time makes recognition of voice-command in real-time difficult, in case of implementations on processors which are suitable for low cost embedded devices. There are also pure hardware implementations present. In [11], an ASIC containing approximately 10,000 gates is designed which calculates the MFCC features of dimension order 12 in hardware. The area is efficiently used by exploiting the symmetric property of cosine function, minimizing the size of the look-up table and decreasing the computational load. The design works at 50 MHz and has an area of 3.2x3.3 sq.mm. The design takes 3670 clock cycles to compute the MFCC without considering the FFT operation. An ASIC design is presented to calculate the MFC coefficients in [12]. A novel design is presented to reduce the computations involved in finding the Mel-filtered energy spectrum. The total number of computation cycles reported is 260 without considering the FFT operation. The design works at 100MHz. Implementing the entire algorithm in hardware increases the response rate. Even though this approach gives good timing performance, there are certain disadvantages. They are

1) The resources needed for implementation of the algorithm is high.
2) Separate hardware designs must be made for different embedded applications as the vocabulary for every kind of application varies.
3) If further additional capabilities are to be added, the dedicated architecture cannot be modified to include them as they are designed only for speech recognition application.

An optimization between the pure hardware-based and pure software-based approaches is to include both the hardware and software components in the system and partition the tasks such that the resources are optimized and the delay is not compromised a lot. This hardware-software co-design approach has the combined advantage of being adaptable and fast enough. Also the area needed would be lesser than the pure hardware implementation. This paper presents a hardware-software co-design implementation of the feature extraction circuit and brings out the comparison between the complete software-based implementation and the co-design implementation.

II. Mel-Frequency Cepstral Coefficient Extraction

The step-by-step calculation of MFCC is shown as a block diagram in Fig 1 and is explained as follows.

- The speech signal is sampled and quantized.

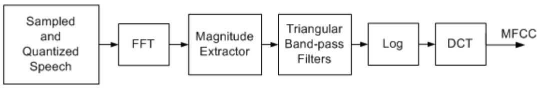

Fig. 1. Feature Extraction - Block Diagram

- Pre-emphasis - The speech samples are sent through a high-pass filter to amplify the frequencies above 1 KHz in the spectrum because hearing is more perceptive in this region [2]
- Frame blocking: The speech samples are blocked into frames of N samples (amounting to a time period of 10-30 ms) with an overlap of some samples between frames.
- Fast Fourier Transform (FFT): FFT is performed on each of the frames to obtain the magnitude values of the frequency response.
- Triangular Band-pass Filtering: The magnitude frequency response is multiplied by a set of triangular band-pass filters to get the log energy value of each filter. The positions of these filters are evenly spaced along the Mel-frequency scale.
- Discrete cosine transform or DCT: The DCT is applied on the logarithm of the energy obtained from the triangular band pass filters. The result gives the Mel-scale cepstral coefficients or MFCCs.

III. Hardware-software co-design of Feature extraction block

The prototyping is carried out on the Atlys development board containing the Spartan-6 FPGA.

A. Acquisition

The speech is collected through the AC'97 codec with the help of the controller circuit. These speech samples are stored in the memory to be retrieved by the feature extraction circuitry. The steps involved in acquisition are as follows.

- The AC'97 codec is initialized - the codec and the controller are synchronized
- The microphone input is selected, and the volume and gain of the input channels are set by giving appropriate control information in the serial-data-out line going to the codec.
- The sampling rate is set to 8000 Hz and the input coming through the microphone is recorded for a particular interval and stored in the DDR2 memory.

B. Implementation of Fast Fourier Transform and Magnitude extractor

The fast Fourier transform is implemented with the help of the LogiCORE IP core FFT v8.0. The FFT core is configured to compute a 256-point forward DFT in pipelined streaming I/O architecture for every frame of the input. The IP core is interfaced with two first-in-first-out (FIFO) memory interfaces on either sides for collecting the input and output. The input frame is written into the FIFO named as the Write-FIFO by the processor. The FIFO streams all the data in the frame into the IP core if the core is ready. The IP core collects and processes the data and after a latency of 862 clock cycles, streams its output data. Also the IP is configured to output the samples in natural order. This output data is collected into a FIFO named as the Read-FIFO. The interface between the FIFOs and the FFT core is implemented as asynchronous handshakes. The FFT is made to be operating on fixed-point data. The output samples are scaled by a value of 256. The processor retrieves the FFT data from the Read-FIFO for further calculating the magnitude of the output complex numbers.

C. Triangular Mel frequency filtering

1) Band-pass Filtering: Mel is a measuring unit of perceived pitch of a sound. The Mel scale was developed based on the hearing perception of human beings. On an approximation, the scale is linear below 1 KHz and it is logarithmic after 1 KHz. This is because human ear can discern comparable pitch increments only with bigger and bigger intervals of frequency above 1 KHz [7]. There are several relations relating the Mel frequency to the linear frequency obtained through varied trial and analysis. The prevalently used relation is given by the Eq.(1): [7]

$$Mel(f) = 1127 * ln(1 + f/700) \tag{1}$$

Only certain frequencies of the spectrum of the speech signal is needed to characterize it effectively. Hence, the spectrum of each speech frame is passed through many band-pass filters to capture the important Mel-frequencies in the speech. This is called Mel-bank filtering. The result produces magnitudes of power at the frequencies that can be heard discretely by the human ear. Hence, these values are called ear magnitudes.

2) Hardware Design: The calculation of the ear-magnitudes is implemented in hardware. It is designed in such a way that it can be called as an instruction by the processor. Forty overlapping triangular band-pass filters centered on forty critical frequencies are constructed. All the triangular filters have unit area. Hence, the filters have decreasing magnitude and have larger frequency bandwidths at larger frequencies. The coefficients of the filters are calculated beforehand and are stored in the memory. Each frame of the speech signal is passed through the filters and the resulting power spectrum values on Mel frequency axis are stored back in the memory. This filtering operation can be represented by the means of a multiplication between a matrix containing the filter coefficients and another matrix containing the speech frame [8]. This can also be inferred as a matrix multiplication between a sparse matrix and the speech matrix as shown in Eq.(2), where $f_{x,y}$, $a_{x,y}$, $emag_{x,y}$ represent the filter coefficients, absolute magnitude of the FFT values and ear-magnitudes respectively in the x^{th} row and y^{th} column of the respective matrices.

$$\begin{pmatrix} f_{1,1} & .. & .. & f_{1,256} \\ . & .. & .. & .. \\ . & .. & .. & .. \\ f_{40,1} & .. & .. & f_{40,256} \end{pmatrix} * \begin{pmatrix} a_{1,1} \\ . \\ . \\ a_{256,1} \end{pmatrix} = \begin{pmatrix} emag_{1,1} \\ . \\ . \\ emag_{40,1} \end{pmatrix} \tag{2}$$

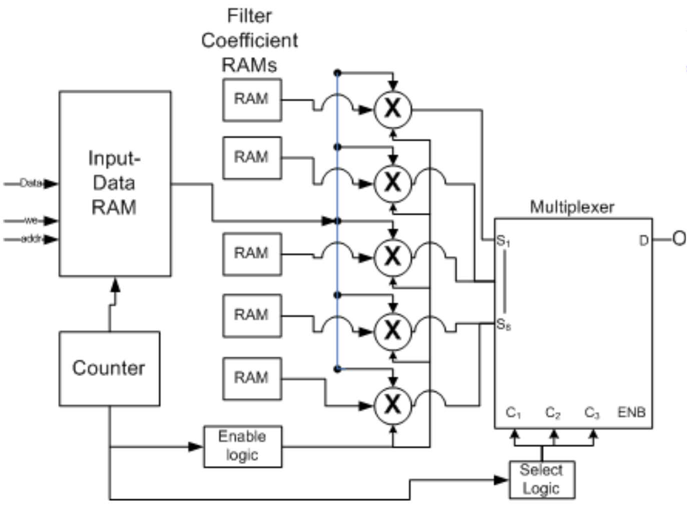

Fig. 2. Design of the Ear Magnitude Extractor

The hardware implementation designed is as shown in Fig. 2. The input data RAM represents the memory where the input speech frames are stored. There are 40 filter-coefficient RAMs which store the coefficients of each filter. In the filter coefficient matrix, there are 413 non-zero values out of the total of 10240 elements in the matrix. Thus there are 413 numbers of multiplications and several additions to be done per speech frame. After filtering, each frame produces 40 values, stored as a column in the emag matrix. Each element of the emag matrix, say emag$_{x,1}$ is produced by multiplying each element of the x^{th} row of filter-coefficient matrix with corresponding elements in the column matrix of the speech frame and accumulating the results of the same. An array of 5 MACs is reused for all the 40 MAC operations that go on in parallel when two matrices are multiplied. Once the 1st MAC operation is done, it is relieved and reset to be used for the 6^{th} MAC operation. It is again used for the 11^{th}, 16^{th}, 21^{st}, 26^{th}, 31^{st} and 36^{th} MAC operations. Similarly the 2^{nd} MAC is used for the 7^{th}, then 12^{th} and so on. A counter circuit and additional control signals are designed in hardware on FPGA. These control circuits reset the MACs at regular intervals and are used for collecting the output values in order from the MAC units. The implementation is done for fixed point numbers with appropriate scaling. The filter weights are scaled by a value of 10000 to include the first four numbers after the radix point. This scaling is approximated later by subtracting a value of 4 from the logarithm of these values. A LUT for the logarithm implementation is used and the hardware multiplier units are used for the data manipulation purposes. The design uses only 5 MACs and some control logic for sequencing and resetting purposes. Figure. 3 shows the RTL schematic implemented by the Xilinx synthesizer. Part of the figure is expanded and shown at the right of the main RTL figure, which depicts the use of the five MAC units.

3) Logarithm circuit implementation: One of the ways of designing the hardware unit for logarithm calculation is through the look-up table (LUT) implementation, which is followed in this project. This method has been detailed in [6]. Any number, a, can be written as shown in Eq. (3), where p is an integer (called the power-value) and N is called the normalized value which is between 0.5 and 1.0.

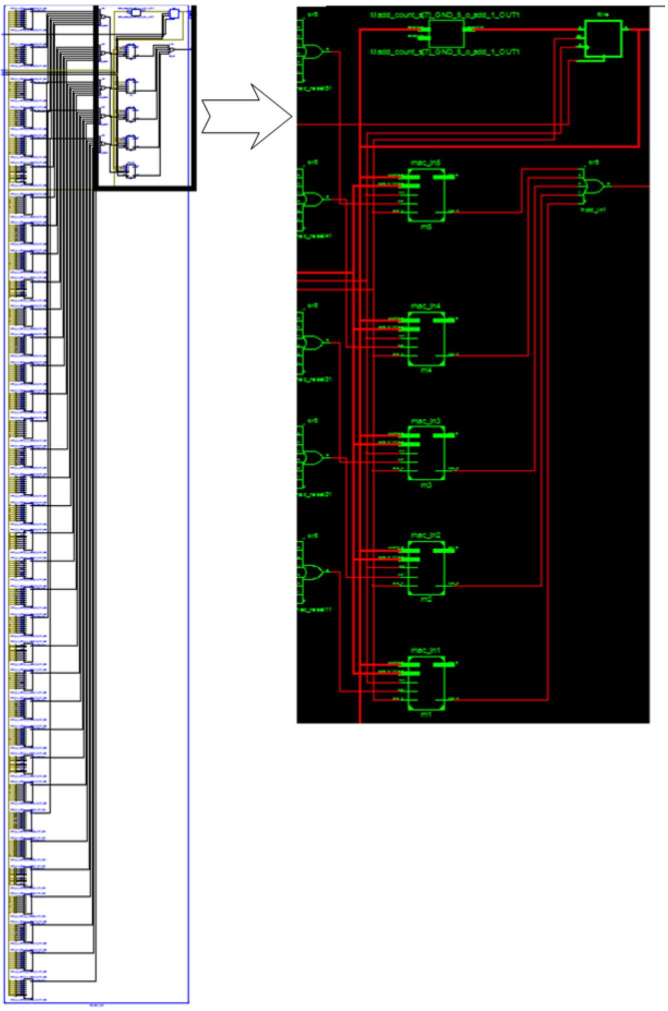

Fig. 3. RTL Figure of Ear-magnitude Extractor

$$a = 2^p * N \tag{3}$$

Taking log on both the sides, Eq.(3) can be written as

$$log_2 a = p + log_2 N \tag{4}$$

Also, the relation between logarithm to the base 2 and natural logarithm can be expressed as Eq.(5).

$$log_2 a = log_e a / log_e 2 \tag{5}$$

Using Eq.(4) and Eq.(5),

$$log_e a = (p + log_2 N) * log_e 2 \tag{6}$$

$log_e 2$ is a floating point constant. Once p and $log_2 N$ are calculated the final logarithm value can be calculated by adding them and later multiplying with the $log_e 2$ value. p can be found out by finding the position of the most-significant bit in the binary representation of the number a that is of logic 1. The value of $log_2 N$ is found out from the look-up table. The table contains 256 numbers of logarithms of values that are equally spaced with an interval of $0.5/256 = 0.001953$ from 0.5 to 1.0. Only the logarithms are stored in the LUT. The indexing is accomplished by manipulating the input number. That is, if x is the input, then the index is calculated as $(a - 0.5)/0.001953$. Finally the scaled values (by 10000) of the Mel filter coefficients are corrected by subtracting the value 4. All the values involved in the logarithm calculations and the values stored in the LUT are scaled by 10000 for fixed-point calculations. This scaling includes additional precision because of including the first four numbers after the decimal point. A hardware block including the LUT is created which can be used as a tailored instruction by the processor. The LUT is created as a single port ROM (using BRAMs). The index of the LUT is given by the address of the ROM.

D. Implementation of the Discrete Cosine Transform

The logarithms of the Mel-scaled spectral coefficients are stored in the memory and accessed by the processor for the calculation of the final Mel frequency cepstral coefficients. This is done by taking the discrete cosine transform of the logarithm of the ear-magnitudes. It can be expressed by the Eq.(7):

$$cep_i = \frac{2}{40} * \sum_{1}^{40} (log_e(mag_j)) * \cos(\pi * 2 * 40 * (j-1) * i) \quad (7)$$

where i = 0 to (C-1). C in the above equation is an integer referring to the number of cepstral coefficients, chosen to be 13. The Eq.(7) can be rewritten as

$$cep_i = \frac{2}{40} * \sum_{1}^{40} (log_e(mag_j)) * m_{i,j} \quad (8)$$

where,

$$m_{i,j} = \cos(\pi * 2 * 40 * (j-1) * i) \quad (9)$$

where i = 0 to 12 and j= 1 to 40 . Eq.(9) can also be represented as the matrix shown in Eq.(10).

$$m_{i,j} = \begin{pmatrix} \cos(\pi*2*40*0*0) & .. \; .. & \cos(\pi*2*40*0*39) \\ \vdots & \vdots \; \vdots & \vdots \\ \cos(\pi*2*40*0*12) & .. \; .. & \cos(\pi*2*40*39*12) \end{pmatrix} \quad (10)$$

Hence, the Eq.(8) can be represented as

$$\begin{pmatrix} cep_1 \\ cep_2 \\ cep_3 \\ . \\ . \\ cep_1 3 \end{pmatrix} = \frac{2}{40} * \left(m_{i,j}\right)_{(13X40)} * \begin{pmatrix} emag_{0,0} \\ . \\ . \\ emag_{39,0} \end{pmatrix}_{(40X1)} \quad (11)$$

This is equivalent to a multiplication between two matrices. The first matrix is a constant matrix and the second is the frame matrix containing the logarithm of the ear-magnitudes. The result gives a column matrix with 13 cepstral coefficients. An array of 13 multiplier-accumulators is used to compute each cepstral coefficient. This avoids the repeated memory access times required for computing each dimension of the MFCCs. The MACs are built using the LogiCORE IP of Xilinx. The MAC design uses the DSP slices available in Spartan-6 and hence, it offers good speed. The cosine values in the equation are stored in memory. A control unit has been built, which keeps track of the memory addresses needed to access the cosine values and the input ear-magnitudes. In this implementation, the ear-magnitudes are accessed from the memory only once. This implementation of the module in hardware is as shown in Fig. 4. The *Input Data RAM* contains the ear-magnitudes. The *Filter Coefficient RAMs* contains the cosine values. The result of the computation produces the Mel-frequency cepstral coefficients or MFCC. The RTL schematic generated by the software is shown in Fig. 5. The blocks with the cross on top of it indicate the multiply-accumulate units. The 13 cepstral coefficients are drawn from this block as outputs simultaneously. These outputs are written in user registers incorporated in the custom user logic of the IP interface module. The processor is designed to read these outputs after the validity flag are set by the hardware showing that new output data has arrived. The processor keeps checking on the flag after giving a frame of ear-magnitudes as the input.

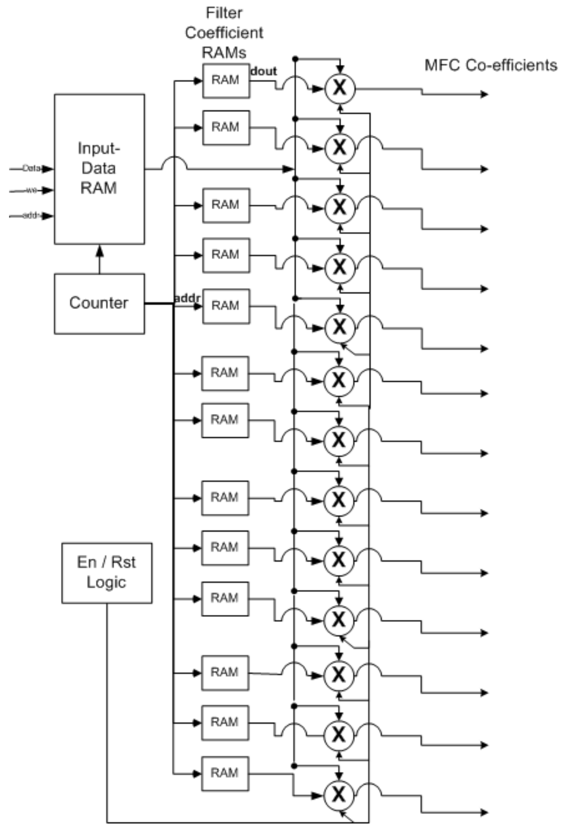

Fig. 4. Design of the Cepstral Coefficient Extractor

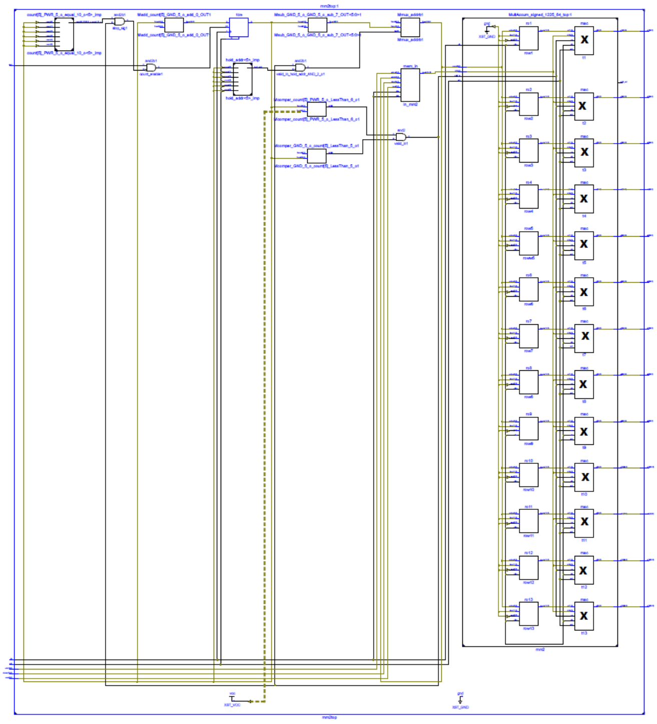

Fig. 5. RTL of the Cepstral Coefficient Extractor

TABLE I
COMPARISON OF THE CO-DESIGN AGAINST SOFTWARE IMPLEMENTATION

Parameter	Software	Co-design
Execution time (s)	17.2987	0.0184
No of slices registers	0	1010
No of slice LUTs	0	1788
No of BRAMs	0	27
No of DSP slices	0	5
No of fully used LUT-FF pairs	0	998

IV. ANALYSIS OF THE IMPLEMENTATION

The area occupied by the designs (excluding the area occupied by the processor) and the execution time for the feature extractions from the speech are tabulated as shown in Table I. As can be observed, the software implementation uses the least amount of resources, however takes a lot of time to finish the MFCC computation.

V. CONCLUSION

The paper presents a hardware-software co-design for the implementation of the Mel-frequency cepstral coefficient extraction from speech signal. This circuit is used in speech recognition applications. The advantage of the hardware-software co-design is brought out by comparing the design with a complete software design implemented on the same platform. The delay is found to have reduced compared to the software implementation because of using exclusive hardware resources, which increases the area.

REFERENCES

[1] Piero Cosi, Giovanni De Poli, and Giampaolo Lauzzana. Auditory modelliing and self-organizing neural networks for timbre classification. *New Music Research*

[2] Vergin, R, O'Shaughnessy, D, and Gupta, V.. Compensated mel frequency cepstrum coefficients. *International Conference on Acoustics, Speech, and Signal Processing* 1, (1996) 323–326.

[3] http://mirlab.org/jang/books/audiosignalprocessing

[4] L.R.Rabiner and R.W.Schafer. Digital Processing of Speech Signals, Pearson Education. Pearson education, 2009.

[5] Hyunjin Lim, Kisun You and Wonyong Sung. Design and Implementation of Speech Recognition on a Softcore Based Fpga *International Conference on Acoustics, Speech and Signal Processing* 3,III.2006.

[6] Rajesh Sharma. Log Approximation Fixed Point Arithmetic.

[7] Terri Kamm, Hynek Hermansky and Andreas Andreou. Learning the Mel-scale and optimal VTN mapping.1998

[8] Developing an Isolated word recognition system in MATLAB. *MATLAB Digest* 2010.

[9] Cheng. O, Abdulla. W and Salcic. Z. HardwareSoftware Codesign of Automatic Speech Recognition System for Embedded Real-Time Applications. *Industrial Electronics, IEEE Transactions on* 58, (March 2011) 850-859.

[10] Peng Li. Design of a Low-Power Coprocessor for Mid-Size Vocabulary Speech Recognition Systems. *Circuits and Systems I: Regular Papers, IEEE Transactions on* 58, (May 2011) 961-970.

[11] Jia-Ching Wang. Chip design of mel frequency cepstral coefficients for speech recognition. *Acoustics, Speech, and Signal Processing, 2000. ICASSP '00. Proceedings. 2000 IEEE International Conference on* 6, (2000) 3658-3661.

[12] Hyunjin Lim. Design and Implementation of Speech Recognition on a Softcore Based Fpga . *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on* 3, (May 2006) III.