



Design and Implementation of Speech Recognition System Based on Field Programmable Gate Array

Haitao Zhou

Information and Communication Department

Tianjin Polytechnic University

Tianjin 300160, China

E-mail: zhouanni@126.com

Xiaojun Han

Information and Communication Department

Tianjin Polytechnic University

Tianjin 300160, China

The research is financed by Applied Program of Basic Research of Tianjin (08JCYBJC14700)

Abstract

In this paper, a Hidden Markov Model (HMM) speech recognition system which is based on Field Programmable Gate Array (FPGA) is designed. It introduces the principle of speech recognition algorithm and deduces the hardware frameworks accordingly. In terms of HMM recognition module, the conventional Viterbi algorithm has been improved and recognition speed has been increased. The core part of the hardware is EP2S60F1020C3 FPGA chip. The experimental result of this system shows that the speech recognition accuracy reaches 94% when ten numbers are being recognized, and the average recognition time is 0.669s.

Keywords: Field Programmable Gate Array, Hidden Markov Model, Speech Recognition, Viterbi Algorithm

1. Introduction

As a new convenient means of human-machine interaction, speech recognition is widely applied to many portable embed speech products. The ultimate aim of speech recognition is to make machine understand natural language. It is of great significance not only in practical application but scientific research. The research on speech recognition technology mainly concentrates on two aspects. One is the software running on computer, the other is embedded systems. The advantages of Embedded systems are high-performance, convenience, cheap and they have huge potential for development.

FPGA has advantages of short development cycle, low-cost design and low-risk. In recent years, FPGA has become the key components in high-performance digital signal processing systems in digital communication, network, video and image fields. In this paper, the design was implemented on an EP2S60F1020C3 FPGA, sitting on stratix II development board.

2. Speech Recognition Basics

Fig.1 shows the speech recognition algorithm flow. A typical speech recognition system starts with the Mel Frequency Cepstrum Coefficient (MFCC) feature analysis stage, which is composed of the following items: 1) Pre-emphasis. 2) Divide the speech signal into frames. 3) Apply the hamming window. 4) Compute the MFCC feature. The second stage is vector quantization stage. In this stage, codebook is used to quantize the MFCC feature and get MFCC feature vector. The codebook is generated on compute via LBG arithmetic, and is downloaded to ROM. The last stage is recognition, which is performed by using a set of statistical models i.e. hidden Markov models (HMM). In this stage, the probability

of MFCC feature vector has been generated by each model and the result is the model which generated the largest probability.

2.1 MFCC Feature analysis

Figure 2 shows the process of creating MFCC features. The first step is to be taken the Discrete Fourier Transform (DFT) of each frame. Certain amount of 0s are added to the end of Time-domain signal $s(n)$ of each frame, in order to form the sequence of N-length. And then the DFT of each frame is taken to get the linear spectrum $X(k)$. In the second step, linear spectrum $X(k)$ is multiplied by the Mel frequency filter banks and converted to Mel spectrum. Mel frequency filter banks are several band pass filters $H_m(k)$, and each band pass filter is defined as follows:

$$H_m(k) = \begin{cases} 0 & (k < f(m-1)) \\ \frac{k - f(m-1)}{f(m) - f(m-1)} & (f(m-1) \leq k \leq f(m)) \\ \frac{f(m+1) - k}{f(m+1) - f(m)} & (f(m) \leq k \leq f(m+1)) \\ 0 & (k > f(m+1)) \end{cases} \quad (0 \leq m < M) \quad (1)$$

Where $0 \leq m < M$, M is the number of the band pass filters, and $f(m)$ is the central frequency.

The third step is to be taken the logarithm of Mel spectrum to get logarithmic spectrum $S(m)$. Thus, the transfer function from linear spectrum $X(k)$ to logarithmic spectrum $S(m)$ is

$$S(m) = \ln \left(\sum_{k=0}^{N-1} |X(k)|^2 H_m(k) \right) \quad (0 \leq m < M) \quad (2)$$

In the last step, logarithmic spectrum $S(m)$ is transformed into cepstrum frequency by Discrete cosine Transform (DCT) in order to yield MFCC feature.

2.2 Vector Quantization

In this paper, due to the discrete hidden markov model is used, it is necessary to transform continuous MFCC feature which has been yielded into discrete MFCC feature.

Vector quantization is to map one K dimensional vector $X \in \tilde{X} \subset R^K$ to another K dimensional quantize vector $Y \in \tilde{Y}_N = \{Y_1, Y_2, \dots, Y_N | Y_i \in R^K\}$, in where X is input vector, Y is quantize vector or codeword, \tilde{X} is source space, \tilde{Y}_N is output space, N is the size of codebook, and R^K is K dimensional Euclidean space.

The process of quantizing vector X is to search a codeword which is the nearest one from the vector X in codebook \tilde{Y}_N . In this paper, square distortion measure is applied to calculate distortion, which is defined as

$$d(X, Y) = \|X - Y\|^2 \quad (3)$$

2.3 HMM Recognition

The role of HMM Recognition is to find out the maximum probability of the HMM which has generated the feature vector, according to the given feature vector. In this paper, viterbi algorithm is used to solve the problem, and an improved algorithm is proposed based on the original algorithm.

The given HMM parameters $\lambda = \{\pi, A, B\}$ ($\pi = \{\pi_i\}, A = \{a_{ij}\}, B = \{b_{jk}\}$), and the observation sequence $O = O_1, O_2, \dots, O_T$, in where N is the number of HMM states, $\delta_i(j)$ is the highest probability along with a single path, at time t , which accounts for the first observations and ends in state j , $\varphi_t(j)$ is the HMM state at time t . The detailed algorithm is defined as follow:

1) Initialization

$$\begin{aligned} \delta_1(j) &= \pi_j b_j(O_1) \quad (1 \leq j \leq N) \\ \varphi_1(j) &= 0 \end{aligned} \quad (4)$$

2) Recursion

$$\begin{aligned} \delta_t(j) &= [\max_i \{\delta_{t-1}(i) a_{ij}\}] b_j(O_{t+1}) \quad (1 \leq j \leq N) \\ \varphi_t(j) &= \arg \max_i \{\delta_{t-1}(i) a_{ij}\} \quad (1 \leq j \leq N) \end{aligned} \quad (5)$$

3) Termination

$$P = \max_j[\delta_T(j)] \quad (1 \leq j \leq N)$$

$$q_t = \arg \max_j[\delta_T(j)] \quad (6)$$

4) Path backtracking

$$q_t = \varphi_{t+1}(q_{t+1}) \quad (1 \leq t \leq T-1) \quad (7)$$

5) Algorithm improving

In practice, π , A and B are decimal fractions between 0 and 1. It is not conducive for FPGA to implement decimal fraction operation, because decimal fraction multiplication may cause the problem of gross underflow when T is larger than a threshold. So it is important to take the logarithm of π , A and B before operation. When π , A and B are transformed to logarithmic probability π' , A' and B' , floating point numbers multiply operation is transformed to integer addition operation. In addition, considering taking out the sign bit before operation, (4) and (5) should be changed to

$$\begin{aligned} \delta_1(j) &= \pi_j' b_j'(O_1) \quad (1 \leq i \leq N) \\ \varphi_1(j) &= 0 \end{aligned} \quad (8)$$

$$\delta_t(j) = [\min_i\{\delta_{t-1}(i) + a_{ij}'\}] + b_j'(O_{t+1}) \quad (1 \leq i \leq N)$$

$$\varphi_t(j) = \arg \min_i\{\delta_{t-1}(i) + a_{ij}'\} \quad (1 \leq j \leq N) \quad (9)$$

Thus (8) and (9) are improved algorithm expression.

3. Design of Speech recognition hardware

3.1 Design of MFCC module hardware

As shown in Fig.3, MFCC module is consisted of DFT module, Mel filter banks, endpoint detection module, logarithm operation module, DCT module, output control module and control module.

Speech signal is sampled at a sample rate of 8 k. Each speech frame is composed of 256 24-bit sample points. Data will be sent to the Mel filter bank under the control of the control element, and the result after calculation will be told to control element. The output of the Mel filter bank will be exported to logarithm computation unit and DCT module to calculate the MFCC parameter. Meanwhile, the point detection will be executed: control element determine whether put out the MFCC parameter according to the output of speech endpoint module. Data get through the module in a pipeline mode, which enhance the system processing speed.

3.2 Design of Vector quantization module hardware

Vector quantization module hardware is designed as Fig.4. The order number is stored in counter1. The index of codebook is stored in counter2. The index of the nearest codebook is stored in register2. The value of the distance between ROM (codebook) and RAM (MFCC) is stored in address module.

The work flow is shown as follows:

- 1) Under the control of the controller, counter1 starts counting. The MFCC of each frame and codebook are read, subtracted, and send to accumulator.
- 2) To compare the value of register2 and the output of accumulator: if the output of accumulator is larger than the value of register2, the controller stops the compute of current codebook and tends to next codebook. Counter1 and accumulator are cleared. The value of counter2 plus 1.
- 3) If the output of accumulator is less than the value of register2 when the value of counter 1 is 12. The output of accumulator is stored in counter 2. The current value of counter 2 is stored in register 1. The index of the nearest codebook and the index of codebook are renewed. Counter1 and accumulator are cleared. The value of counter2 plus 1.
- 4) To repeat above process, until the value of counter 2 is 256. Then the vector quantization of a speech frame is accomplished.

3.3 Design of HMM recognition module hardware

A 4 state left-to-right HMM without skipping is adopted in this paper.

The design of HMM recognition module hardware is shown in Fig.5. FSM is the controller of state machine. The observation sequence is stored in RAM O. The value of initial probability is stored in RAM Pi. State transition probability A is stored in RAM A. Output probability B is stored in RAM B. The address of RAM A and REM B are generated from GENaddrA and GENaddrB respectively. CurrentMin is used for preserving the smallest probability of the recognition model until the current model, CounterIndex is used for saving the model label of the smallest

probability,

The key point of the Viterbi algorithm is seeking $\delta_t(j)$ via type (9), as a result, the PE unit has been designed for calculating $\delta_t(j)$ in this paper. As shown in Figure 6, PE unit is consisted of three adders and two data selectors. First, to calculate the value of $\delta_{t-1}(j-1) + a'_{j-1,j}$ and $\delta_{t-1}(j) + a'_{j,j}$. Second, to choose the smallest value through data selector, and add a value of $b'_j(O_t)$ on it to get $\delta_t(j)$. In the initial state, if $j=1$, it only needs to compare the value of $\delta_{t-1}(j) + a'_{j,j}$ and π'_j , and to take the smaller value as the smallest value.

4. Implementation and Results

It was achieved the entire voice training and the recognition process by using Stratix II EP2S60 DSP development board as the hardware platform of Voice processing module. Fig.7 is the RTL view.

Acquire the voice signal through the microphones and PC-in tape recorder. The sample rate was 11025KHz, and the sample precision was 16bits. Gain 50 samples for each mandarin digit from 1 to 10 as the experiment subjects.

The experimental results were shown in table 1. The average recognition accuracy of speaker-independent mandarin digits reaches 94% and the average recognition time is 0.669s in this system, which achieves the recognition rate and real-time requirements.

5. Conclusions

In this paper, a FPGA-based Hidden Markov Model speech recognition system was designed. It completes the acquisition of voice by microphone and PC-in tape recorder and the generation of code book and training data. In the system, calculate the MFCC feature vector was calculated, quantized and recognized by Vertibi algorithms. In the HMM recognition, the traditional Viterbi algorithm was improved to enhance the recognition speed, which was able to meet the needs for real-time voice recognition systems and the requirements of the recognition accuracy.

References

Altera Corporation. (2006). *Nios II Processor Reference Handbook*, 1-1.
 Altera Corporation. (2006). *Nios II Software Developer's Handbook*, 4-1.
 Bok-Gue Park, Koon-shik Cho, & Jun-Dong Cho. (2002). Low power VLSI architecture of vertibi scorer for HMM-based isolated word recognition. *International Symposium on Quality Electronic Design*, 235-39.
 Elmisery, F, A, Khalil, A, H, Salama, A, E, & Hamed, H, E. (2003). A FPGA-Based HMM for a discrete Arabic speech recognition system. *Proceedings of the 15th International Conference on 9-10 Dec*, 322-325
 Lawrence, R, Rabiner. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, VOL.77, NO 2, February.
 Lawrence, Rabiner, & Biing-Hwang, Juang. (1999). *Fundamentals of speech recognition*. Beijing: Prentice-Hall International, Inc.
 Melnikoff, S, J, Quigley, S, F, & Russell, M, J. (2002). Implementing a simple continuous speech recognition system on an FPGA. *Field-Programmable Custom Computing Machines, Proceedings.10th annual IEEE Symposium*, 275-276
 Nedeveschi, S, Patra, R, K, & Brewer, E, A. (2005) Hardware speech recognition for user interfaces in low cost, low power devices. *Design, Automation Conference. Proceedings. 42nd13-17 June*, 684-689.
 Yoshizawa, S, Miynamaga, Y, & Wada, N. (2002). A low-power VLSI design of an HMM based speech recognition system. *Circuit sand Systems. Midwest Symposium on Volume 2, II-489-II-49292*.

Table 1. Experiment result

Number	1	2	3	4	5	6	7	8	9	10
Correct rate (%)	96	94	94	92	96	92	94	92	96	94
Time (μS)	0.67	0.69	0.65	0.70	0.66	0.65	0.68	0.65	0.64	0.70

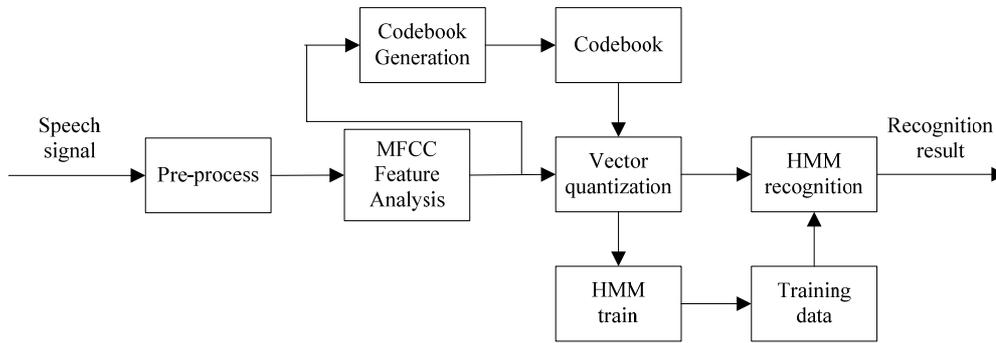


Figure 1. Speech recognition algorithm flow

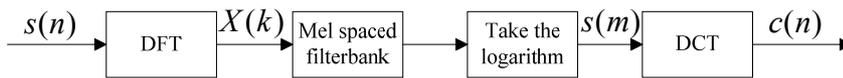


Figure 2. MFCC Feature analysis algorithm flow

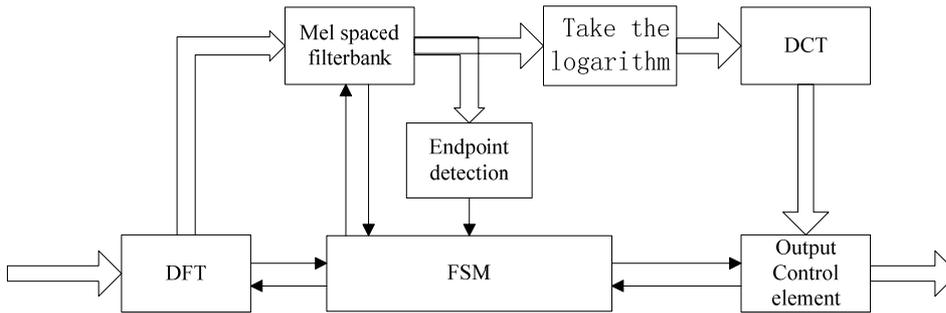


Figure 3. MFCC feature analysis hardware structure

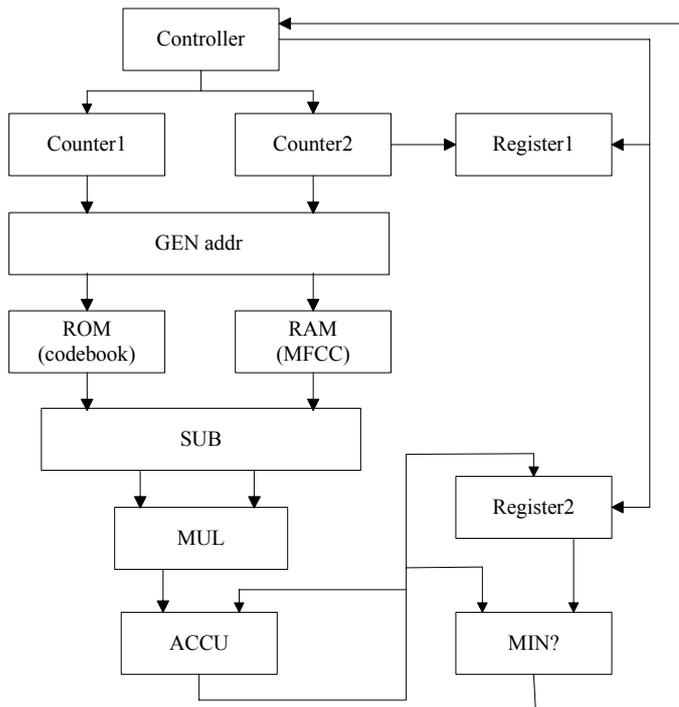


Figure 4. Vector quantization hardware structure

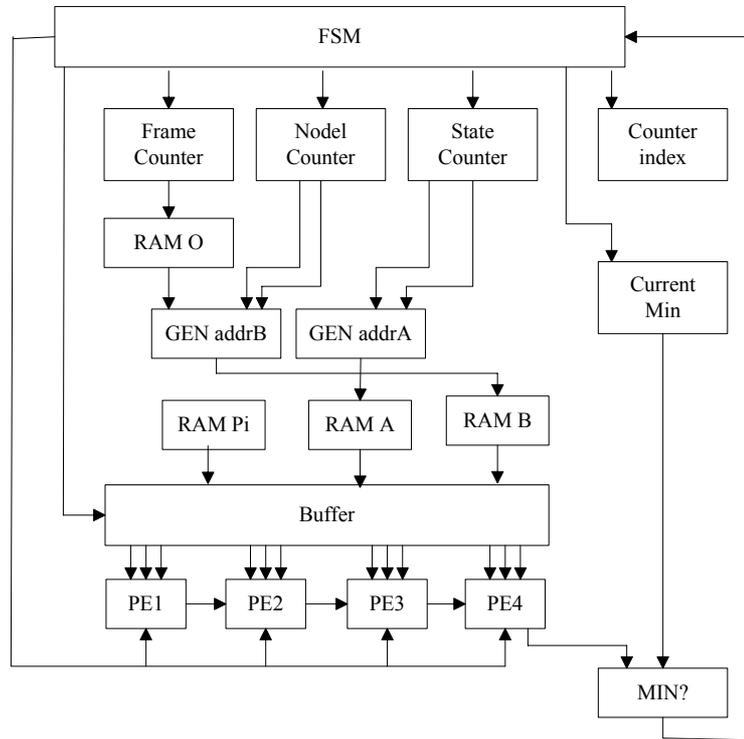


Figure 5. HMM recognition hardware structure

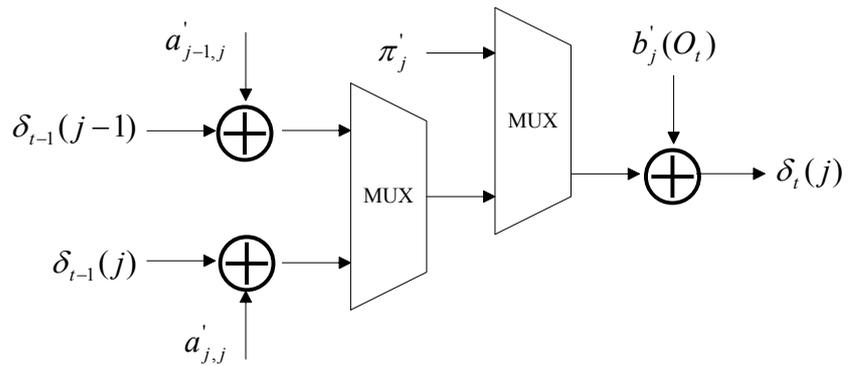


Figure 6. processing element

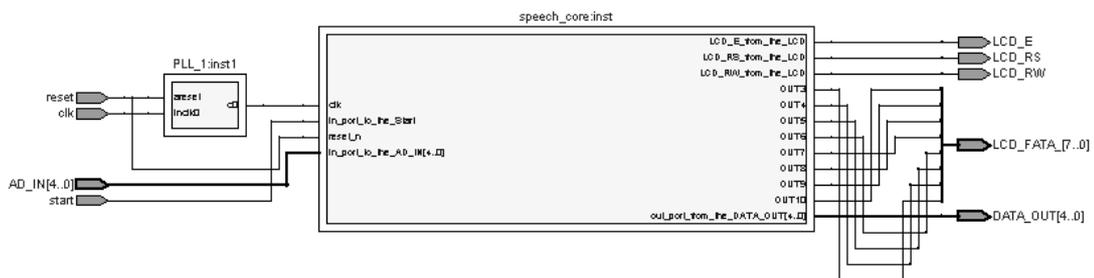


Figure 7. RTL view