International Journal of Innovative Computing, Information and Control
Volume **7**, Number **10**, October **2011**

 ISSN 1349-4198
pp. **0–0**

THE IMPLEMENTATION OF SPEECH RECOGNITION SYSTEMS ON FPGA-BASED EMBEDDED SYSTEMS WITH SOC ARCHITECTURE

Shing-Tai Pan

Department of Computer Science and Information Engineering
National University of Kaohsiung
Kaohsiung 81148, Taiwan
stpan@nuk.edu.tw

Chih-Chin Lai

Department of Electrical Engineering
National University of Kaohsiung
Kaohsiung 81148, Taiwan
cclai@nuk.edu.tw

Bo-Yu Tsai

Department of Electrical Engineering
National University of Kaohsiung
Kaohsiung 81148, Taiwan

Abstract. *An implementation of Artificial-Neural-Network (ANN) based speech recognition systems on the embedded platform is explored in this paper. An FPGA chip is adopted as the hardware of the embedded platform with the architecture of SOC. This makes the speech recognition systems applicable on the voice activated systems in toys, games, smart phones, office devices, vehicular communications, etc. Because the fast Fourier transform (FFT) is an important operation in speech recognition, in which a great deal of floating-point arithmetic operations are performed, it takes a long time for speech recognition due to the limitation on the computation capability of the embedded platform. In this paper, we use Integer FFT to replace the Floating FFT so that the speed of speech recognition is enhanced without influencing the recognition rate. The experimental results from the FPGA platform reveal that the speech recognition rate of the proposed hardware implementation methods is better than that in existing literatures.*
Keywords: Speech recognition, Embedded system, SOC architecture, FPGA

1. **Introduction.** The development of speech recognition attracts a lot research in recent years [1, 2, 3, 4, 5, 6]. With the development and maturity of speech recognition technique, speech becomes an important tool of man-machine interface. This speech interface is increasingly used in office automation, factory automation, and home automation devices [2, 3]. The methods of speech recognition are mainly categorized as three ways: the earliest one is the Dynamic Time Warping (DTW) method [7], which uses time difference in a speech sound frame to achieve the speech recognition. Then, the Artificial Neural Network (ANN) [8] is applied on the speech recognition and then replaces DTW in most of the applications of speech recognition. Finally, Hidden Markov Model (HMM) [9, 10] appears, which uses the statistical method to enhance the speech recognition rate.

Duo to the vast improvement on the manufacture of ICs in recent years, the capacity and the calculation ability of a chip is dramatically increased. Consequently, the speech recognition can be implemented in a chip rather than in a computer. This makes the embedded speech recognition systems in many consuming electronics realizable and then applicable on the voice activated systems in toys, games, consumer electronics, and office devices [3]. Especially, due to the trend of the recent development in consumer electronics, the speech recognition systems are always embedded in smart phone, vehicular communications and intelligent houses, etc. In [3], a micro-controller 8051 chip is adopted for implementation of a speech recognition system. However, in order to have more extensive applications of speech recognition in consuming electronics, a platform with better computational ability and flexibility, such as embedded systems with SOC structure, is necessary for the implementation of the systems. On the embedded platform, the speech recognition systems can perform much more applications by combing the other units on the platform. This is the reason why the embedded platform is adopted in this paper to realize the speech recognition systems.

Because the speed of speech recognition by ANN surpasses other methods and the calculation burdens required for the recognition are fewer, ANN is more suitable to be implemented on a chip with lower computation capacity. Therefore, this study adopts ANN to be the structure of speech recognition system. In this paper, the back-propagation algorithm is adopted to train the ANN. For the ANN speech recognition process, the pre-processing procedures such as sound recording, effective End-Point Detection (EPD), pre-emphasis, the calculation of speech signal features should be first addressed, and then the speech signal will be recognized by using ANN according to the obtained features. As for the calculation for feature, this paper adopts Mel-Frequency Cepstrum Coefficient (MFCC) duo to its excellent performance comparing to other methods in many applications. However, it needs a great deal of process for digital signals to perform MFCC, which will result in great burdens on real-time required by embedded system. In general, the design of embedded system has two requirements: one is the request of real-time, while the other is the simplification of memory usage. However, quite a few research aim at the two aspects which are available already [11]. MFCC is calculated in the frequency domain; hence, the signal represented in time domain should be converted into frequency-domain representation by FFT in advance. So far, almost all the existing literatures apply Floating FFT to transfer the speech signal from time-domain representation to frequency-domain representation. This will increase the computation time enormously. Hence, this paper uses Integer FFT for the domain transformation of speech signal to avoid the floating-point arithmetic operations. As long as the problem of overflow is prevented, the original wave shape of time domain can be reserved. And hence the speech recognition rate wouldn't be reduced.

The rest of this paper is organized as follows. Section 2 gives some introductions about the pre-processes for speech signals before effective speech recognition, such as EPD, pre-emphasis, and the calculation of speech signal feature. In Section 3, we will briefly describe the architecture of ANN. The Integer FFT is introduced in Section 4 which explains how to use integer multiplication to replace floating-point arithmetic multiplication in FFT. Section 5 shows the experimental results of speech recognition on the embedded system with SOC architecture. Finally, some conclusions are given in Section 6.

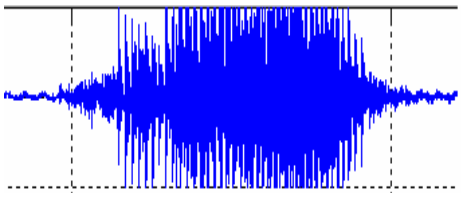

FIGURE 1. End-point detection

2. **Pre-Processing For Speech Signal.** Before doing speech recognition on the recognition platform, we need to do some processing of the speech signal, such as end-point detection, pre-emphasis, multiplication of Hamming window and retrieval of feature, etc, which are called pre-processing for speech signal here.

Since the speech signal belongs to time-varying signal which is complicated and hardly to find its rule, the signal need to be sliced into many small frames (audio frame). When a speech signal is cut into small frames, its rules can be seen, and the frame can be viewed as time-invariant signal [12]. Consequently, the speech signals are divided into several audio frames, and then the features according to its property of rules are retrieved for speech recognition. The steps of pre-processing for the speech signals are described as follows.

2.1. **Fixed-size frame and dynamic-size frame.** Because the length of every speech signal is different, if we adopt the length of fixed-size frame, the number of audio frames will be different depending on the different speed of talking for every recording. There is no problem for the speech recognition platform that uses DTW. However, for ANN used in this article, fixed-size frame can not be applied duo to the constant input of ANN. Hence, instead of fixed-size frame, the dynamic-size frame is used to acquire the fixed number of frames, in order to match the requirement of ANN recognition system. Besides, there are two ways to achieve fixed number of frames: (1) the use of dynamic-size frame sampling points (2) adjusts the overlap rate of dynamic-size frame [13]. Adjusting the number of dynamic-size sampling points or dynamic-size overlap rate, we can get a consistent number of audio frames to achieve the requirement for ANN speech recognition system.

2.2. **End-point detection (EPD).** The recorded speech signals always include speech segments, silence segments and background noise. The process of identification between speech segments and silence segments is called EPD, shown as the Figure 1. If the non-speech parts, i.e., silence segments and background noise, are removed, the number of timeframe for comparison will decrease, and then the computation time for speech recognition will speed up and recognition rate will be improved.

There are many algorithms for the EPD of speech sound signal, which can be mainly divided into three types according to the domain of analysis: (1) time-domain EPD, (2) frequency-domain EPD, and (3) mixed parameters EPD. Among them, time-domain EPD is one of the simplest and the most popular ways with, however, the disadvantage of less anti-noise capacity. As for frequency-domain EPD and mixed parameter EPD, they have stronger anti-noise capacity and are more precise, but the disadvantage is that the calculation for frequency-domain transform from time-domain signal is more complex.

2.3. **Pre-emphasis.** Spreading via air, the magnitude of speech signal will reduce as the frequency rises. In order to compensate the attenuated speech signal, we put the signal through into a high-pass filter to recover the signal. The difference equation governing high-pass filter is shown as the following equation.

$$S(n) = X(n) - 0.95X(n-1), 1 \leq n \leq L. \tag{1}$$

In the Eq.(1), $S(n)$ represents the signal that has been processed with high-pass filter, while $X(n)$ represents the original signal, and L is the length (number of sampling) of each audio frame.

2.4. **Hamming window.** The purpose of applying Hamming window to the signal is to prevent the non-continuity in the two ends of the audio frame, and to avoid the influence of front and back audio frames when analyzed. The non-continuity of the audio frames can be eliminated by multiplying the Hamming window with the audio frames, because this process can make each audio frame more centered on the frequency spectrum. The following Eqs.(2) and (3) are the function of Hamming window [14] and the result of the multiplication from Hamming window and audio frame, respectively.

$$W(n) = \begin{cases} 0.54 - 0.46\cos(\frac{2n\pi}{L-1}), 0 \leq n \leq L-1, \\ 0, \text{otherwise.} \end{cases} \tag{2}$$

$$F(n) = W(n) \times S(n), \tag{3}$$

in which $S(n)$ is a frame of speech signal, $W(n)$ is the Hamming window, and $F(n)$ is the result of audio frame multiplied by Hamming window.

2.5. **Retrieval of feature.** Suppose the speech sound signal is sampled by the rate of 8k, there are 8,000 sampling points to be processed per second. Such tremendous data and the complexity in speech signal make the process for the speech recognition hard. Therefore, the properties of a speech signal, i.e., the features of a speech signal, are extracted for the speech recognition. And then the speech recognition can be performed according to these features. This will massively reduce the number of data to be processed.

In speech recognition, the methods commonly used for extracting the feature of speech signal are time-domain analysis and frequency-domain analysis. The way of time-domain analysis is more direct and time-saving doe to the fewer operations. On the other hand, in frequency-domain analysis, the signal has to undergo Fourier transform first. So, it needs more operations and is more complicated and hence leads to the requirement of much more computation time compared to time-domain analysis. The most popular methods for features extraction are Linear Predict Coding in time domain and, in frequency domain, are Cepstrum Coefficient and MFCC [14]. Because the distinction by MFCC is more close to that made by human ears toward speech signal, we use it to extract the feature for speech signal in this paper. The processes of MFCC are described as follows. First, each audio frame is transformed to frequency domain, says $|X(k)|$, by FFT. Due to masking effect in sound, the energy in each frequency domain $|X(k)|$ will be multiplied by a triangle

filter as follows,

$$B_m(k) = \begin{cases} 0, k < f_{m-1}, \\ \dfrac{k - f_{m-1}}{f_m - f_{m-1}}, f_{m-1} \le k \le f_m, \\ \dfrac{f_{m+1} - k}{f_{m+1} - f_m}, f_m \le k \le f_{m+1}, \\ 0, f_{m+1} < k, \end{cases} \tag{4}$$

where $1 \le m \le M$ and M is the number of the filters. After accumulating and applying the log(.) function, we can get a energy function

$$Y(m) = \log \left[\sum_{k=f_{m-1}}^{f_{m+1}} |X(k)| B_m(k) \right]. \tag{5}$$

Finally, the MFCC can be obtained by applying the Discrete Cosine Transform on M pieces of $Y(m)$ as

$$c_x(n) = \frac{1}{M} \sum_{m=1}^{M} Y(m) \cos \left[\frac{(m - \frac{1}{2})n\pi}{M} \right], \tag{6}$$

in which $c_x(n)$ is MFCC. In order to decrease the input number of ANN, only the preceding 10 coefficients are used.

3. **Speech Recognition Platform.** After the feature of speech signal is retrieved, the speech signal can then be recognized. As that we have introduced in the Introduction part, there are many ways to do speech recognition by making comparisons through the feature parameters. For the recognition by DTW, we have to compare the data in the data base of speech signal samples with the tested speech sounds one by one. Consequently, the more the tested speech sounds are, the more the computation time for recognition costs. Moreover, the process of the method HMM needs much statistical computation for speech recognition. It is seen that the more speech signals to be recognized, the more statistical computation must be done. In contrast, as long as the training of ANN is finished, the speech recognition on ANN is relatively rapid. This is because that the dimension of ANN is fixed after it had been trained and is regardless to the amount of the signal to be recognized. Consequently, the advantage of the method ANN for speech recognition is that it can get a faster recognition speed. Hence, we adopt the method of ANN in this paper for the purpose of chip realization of speech recognition system. In addition to the advantage in recognition rate, ANN has another advantage on fault-tolerant capacity, which is a specific feature of ANN with respect to other methods.

3.1. **Back-propagation neural network.** The principles of ANN are all based on multiple-layer perceptron as its systematic structure. The back-propagation algorithm is used for the training of ANN. Such a system is then called back-propagation neural network. The multiple layer in the model of multiple-layer perceptron means that it comprises multiple-layered neurons, in which the way to transmit signals between every two neurons is just like the that in single layer. In this paper, we adopt the three-layered structure (i.e., input layer, hidden layer, and output layer) which is shown in Figure 2 to be the structure of ANN for speech recognition. .

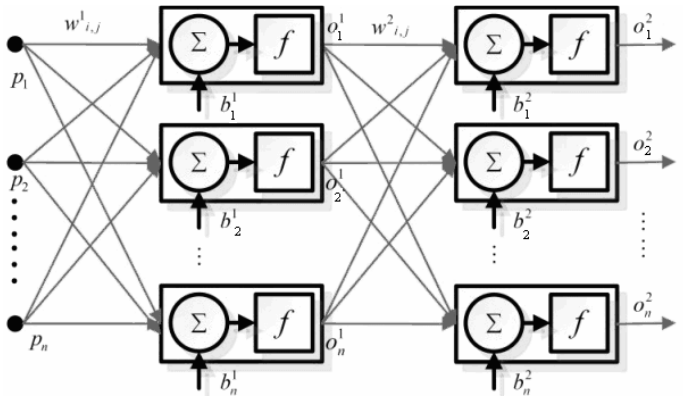

FIGURE 2. The structure of an Artificial Neural Network

In Figure 2, p_n means the n-th input, $w^k_{i,j}$ represents the weight value of the k layer, i indicates the number of the neuron in the $(k-1)$-th layer, j is the number of neutron in the k-th layer, o^k_n represents the n-th output of the k-th layer, and b^k_n is the bias in the k-th layer of n-th neuron. The most important goal of back-propagation algorithm is to adjust the weight of ANN by means of the error function between the output and the target. And then the modified value will be transmitted to the neuron in the front layer. This process will continue until the ideal target is achieved. The Eqs.(7) and (8) define the output function of the ANN in this research, in which the variable i represents the number of the neuron in the $(k-1)$-th layer.

$$o^k_n = f\left(\sum_{\forall i} w^k_{i,n} o^{k-1}_i - b^k_n\right), \tag{7}$$

$$f(x) = \frac{1}{1+e^{-x}}. \tag{8}$$

The back-propagation algorithm is then shown in the following Eqs.(9)-(14).

$$\delta_n = (T_n - o^2_n) \times f'\left(\sum_i w^2_{i,n} o^1_i - b^2_n\right), \tag{9}$$

$$\Delta W^2_{i,n} = \eta \times \delta_n \times o^1_i, \tag{10}$$

$$\Delta b^2_n = -\eta \times \delta_n, \tag{11}$$

$$\delta_i = \left(\sum_n \delta_n W^2_{i,n}\right) \times f'\left(\sum_i w^1_{r,i} p_r - b^1_i\right), \tag{12}$$

$$\Delta W^1_{r,i} = \eta \times \delta_i \times p_r, \tag{13}$$

$$\Delta b^1_i = -\eta \times \delta_i. \tag{14}$$

The Eqs.(9)-(11) are used for the calculation of the deviation for weights and bias in the layer 2, while Eqs.(12)-(14) are for those in layer 1 for ANN. The symbols T_n is the target output value, η is the step size of each iteration, r is the number of neurons in the input layer, i is the number of neurons in the hidden layer, and n is the number of neurons in the output layer.

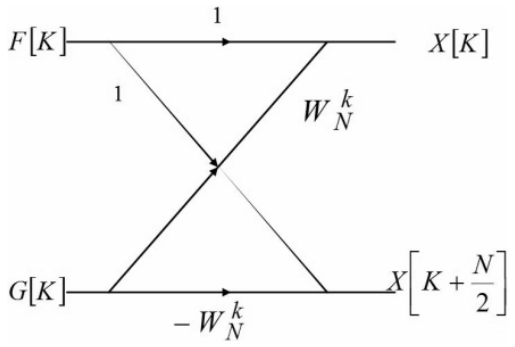

FIGURE 3. The time division for FFT

4. **Fast Fourier Transform (FFT).** As for the application on the embedded platform, if Discrete Fourier Transform (DFT) is used to transform time-domain signal to frequency-domain signal in the calculation of MFCC, the burdens on computation time will be too huge to have a real-time application. So, we use FFT to increase the speed. However, due to the limitation of FFT, the sampling points of each audio frame should be limited in 2^n times.

4.1. **The principle of fast Fourier transform.** To transform the discrete signal from time domain to frequency domain by DFT is described as follows [15]:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, 0 \leq k \leq N-1, \tag{15}$$

where $W_N = e^{\frac{-j2\pi}{N}}$, N is the number of sampling points in an audio frame.

For the calculation of DFT, a higher efficiency can be obtained by decomposing and calculating Eq.(15) to be many serial small DFT and then evaluate. During this process, both the symmetric and periodic properties of the complex number index $W_N^{kn} = e^{-j(\frac{2\pi}{N})kn}$ are used. The decomposition of the algorithm is based on decomposing the sequence $x[n]$ into many small sequences. Hence, it is called Time Division Algorithm. First, the DFT in Eq.(15) is decomposed as

$$X[k] = \sum_{n=0}^{\frac{N}{2}-1} f[n] W_{\frac{N}{2}}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} g[n] W_{\frac{N}{2}}^{nk} = F[k] + W_N^k G[k], \tag{16}$$

in which $f[n] = x[2n]$ and $g[n] = x[2n+1]$ are the even sampling and odd sampling of $x[n]$, respectively. Figure 3 shows the time division for FFT. The multiplication complexity N^2 for original DFT can then be reduced to be $\frac{N}{2}\log_2 N$.

4.2. **Integer FFT.** Although FFT can be used to speed up the computation time of DFT; however, the floating-point arithmetic operation always costs much time. Unfortunately, the FFT operation is an important step in the calculation of MFCC for speech recognition. Consequently, the wasted time on floating FFT operation will make the real-time application of speech recognition on the embedded platform impossible. Therefore, instead of the multiplication and addition for floating-point numbers, integer operations

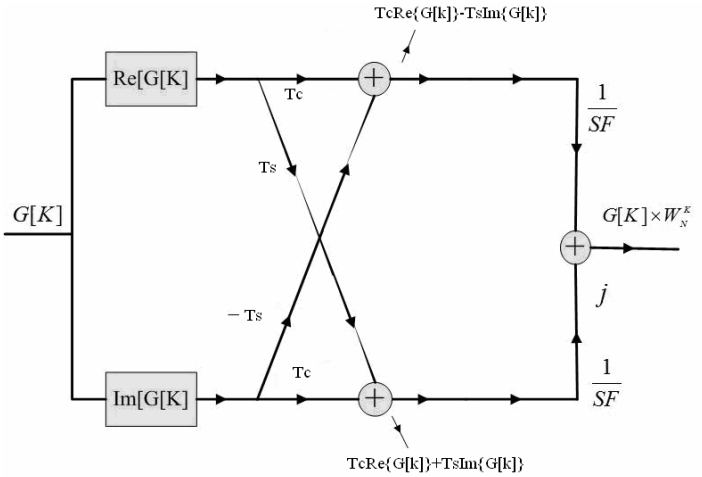

FIGURE 4. An approximation of the operation $W_N^k G[k]$

are used. The Integer FFT is used in this paper to improve the computing efficient of speech recognition on embedded platform.

The Integer FFT is implemented in this paper as follows. The function W_N^k in Figure 3 is expanded as

$$W_N^k = e^{-j(\frac{2\pi}{N})k} = \cos(\frac{-2\pi k}{N}) + j\sin(\frac{-2\pi k}{N}) = c + js, \tag{17}$$

in which $c \equiv \cos(\frac{-2\pi k}{N})$ and $s \equiv \sin(\frac{-2\pi k}{N})$. The real and imaginary part of W_N^k, c and s, respectively, will be amplified by a factor of SF and truncated to a integer before FFT operation. These results are then tabulated to accelerate the computation time. In the last stage of the operation of $W_N^k G[k]$, the factor $\frac{1}{SF}$ will be multiplied for recovering the original magnitude. We can then obtain an approximation of $W_N^k G[k]$. The approximation of the operation $W_N^k G[k]$ is then derived as follows and is shown in Figure 4.

$$G[K] \times W_N^k \cong \frac{1}{SF}(T_c Re[G[K]] - T_s Im[G[K]]) + j\left\{\frac{1}{SF}(T_c Re[G[K]] + T_s Im[G[K]])\right\}, \tag{18}$$

in which, $T_c = trunscate(c \times SF)$ and $T_s = trunscate(s \times SF)$.

In hardware realization of the above algorithm, the functions of cos(.) and sin(.) which are stored in the registers of the embedded platform, are shifted to the left for n bit to perform the multiplication of SF. And it is shifted back (to the right) to perform the multiplication of $1/SF$ when the operation is completed. The digits after the decimal point are neglected. This process will be repeated for the calculation of $\log_2 N$ times to complete the calculation of FFT [16]. During the process of FFT calculation, every addition may increase one bit in length. Hence, for the variables of real numbers and imaginary numbers stored inside FFT, enough storage space is indispensable for keeping from overflow.

5. **Experimental Results.** The developed speech recognition system is implemented on an FPGA-based SOC embedded platform. The block diagram is shown in Figure 5. An Altera develop board DE2-70 in which a Cyclone II FPGA chip is included is used for this

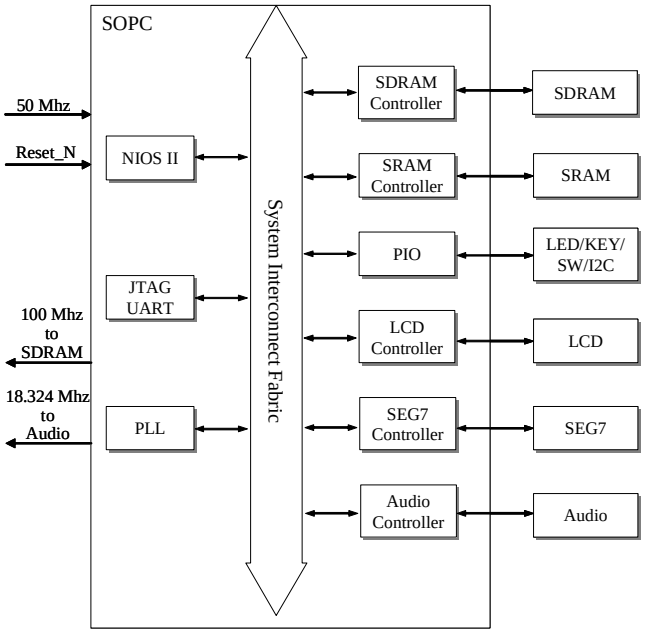

FIGURE 5. Block diagram of the SOC system in this experiment

experiment. The condition on the development board for this experiment is that there is no operation system in the FPGA chip, only single-threaded procedure is available. This will slow down the computation speed of the speech recognition systems. On the board a push button is used for the control of the starting and ending of the voice record and a Toggle switch is used for controlling the sampling rate of AUDIO codec. The EDA tools Quartus II, SOPC Builder and Nios II are used to develop and simulate the system. In the hardware implementation, SRAM and Flash RAM are used for the storage of source code and testing signal, respectively. The I2C Protocol is used to control the register of the platform. Besides, AUDIO Controller is used to receive speech data and SEG7 is used for the display of recognition results. The standard control IPs which are supported by SOC Builder are adopted for driving the necessary elements SDRAM, SRAM and LCD. The push button and Toggle switch are connected by the built-in PIO. Moreover, SEG7 Controller and AUDIO Controller are user-defined. In the experiment, PLL is adjusted to have a frequency of 100Mhz and a delay of 3ns, and then support the system's clock.

After the introduction of the hardware and software of the SOC embedded platform used in this experiment, the procedures for this experiment are then described as follows.

5.1. **Pre-processing of the speech signal.** As for the number of speech signal frame, because we use ANN recognition platform, the same numbers of frames are necessary with various length of speech segment. Also, when FFT is used in the calculation for feature, the sampling point of each frame should be fixed. Therefore, this paper adopts the dynamic overlap rate as the Eq.(19) [13]:

$$R = \frac{1}{l_F}\left[l_F - floor\left(\frac{l_S - l_F}{N_F - 1}\right)\right], \tag{19}$$

in which R is the overlap rate (%) of speech signal frame, l_F is the width of each frame, l_S is the total effective sampling length, N_F is the number of frames, and the function $floor(x)$ is the maximum integer that is smaller than x.

Besides, the time-domain EPD is adopted in the experiment due to its property of fewer calculations loading and hence of short computation time. The process of time-domain EPD is first to acquire the average energy value of the preceding several silence frames which contains background noise. Then, the average energy value is added with the value which ranges between 5% and 10% of the maximum energy value of audio frame to get the threshold value for EPD. This process is governed by the Eq.(20) [13, 17]:

$$\text{Threshold} = 7.5\% \times \max[E(n)] + \frac{1}{K}\sum_{i=1}^{K} E(i), 1 \le n \le N, \tag{20}$$

in which "Threshold" is the the threshold value for EPD, $E(n)$ is the energy sum of sampling points in the n audio frame before EPD, N is the number of audio frames before EPD, K means the number of retrieved silence frames, and $E(i)$ is the energy sum of sampling points in the i-th silence frame. In this study, the 7.5% of the maximum energy value in audio frames ($\max[E(n)]$) is added with 5 average energy values in silence frames ($K = 5$) to obtain the threshold value for EPD is finished.

The Integer FFT can express smaller range of value than Floating-point FFT, and hence overflow appears frequently. During the process of the experiment, speech recognition has poor tolerance on overflow, and there is a possibility of carry which arises in the addition in every stage of FFT. So, the storage space should be reserved in advance to keep form overflow. However, in order to get a tradeoff between the computation time and accuracy, the length of register for storing the data is specified as 32 bit in the adopted SOC embedded platform. In this experiment, we would like to compare the difference between the results by Integer FFT and Float FFT to show that the results of the Integer FFT implemented is very close to those of Float FFT. Hence, the Float FFT can be replaced by Integer FFT with little influence on the recognition rate. Let the error between the Integer FFT and the Float FFT for each frequency K is defined as

$$E[K] = \frac{1}{25}\left(\sum_{i=1}^{25} integerFFT_i[K] - \sum_{i=1}^{25} floatFFT_i[K]\right), 0 \le K \le 255, \tag{21}$$

in which $integerFFT_i[K]$ and $floatFFT_i[K]$ are the Integer FFT and Float FFT of i-th frame, respectively. The spectrum of $E[K]$ is shown in Figure 6. Besides, the spectrum of $integerFFT_i[K]$ and $floatFFT_i[K]$ are shown in Figure 7. From Figure 6 and Figure 7, it can be seen that the magnitude of the error $E[K]$ between the spectrum of $integerFFT_i[K]$ and $floatFFT_i[K]$ are very small relative the those of the spectrum of $integerFFT_i[K]$ or $floatFFT_i[K]$. This implies that the Float FFT can be approximated by Integer FFT with little influence on the speech recognition rate.

5.2. **Training the ANN.** In the experiment, a speech signal is assigned to be 25 audio frames, while an audio frame has 256 sampling points and 10 features. This means that there will be 250 inputs in the input layer of ANN with the hidden layer of 30 neurons. As for the output layer, because 10 speech signals ($0 \sim 9$) are to be recognized, there are 10 output neurons in the ANN. As for the speech signal samples used in the training

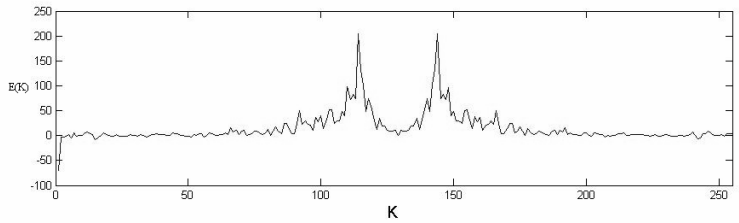

FIGURE 6. The spectrum of $E[K]$

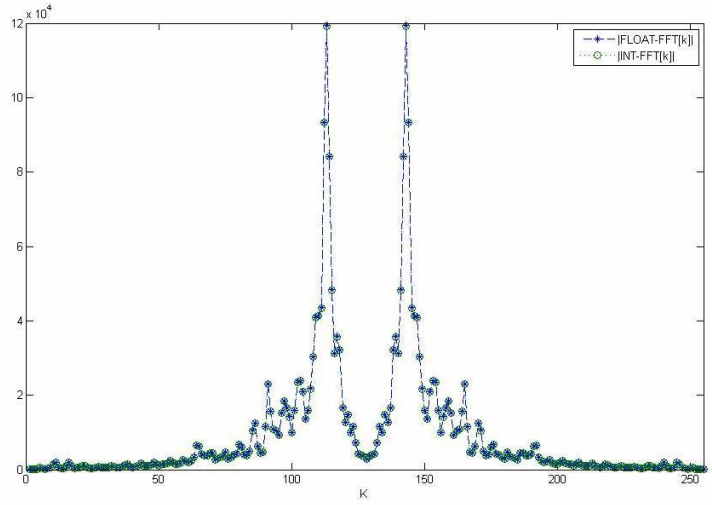

FIGURE 7. The spectrum of $integerFFT_i[K]$ and $floatFFT_i[K]$

phase of ANN in this experiment, each speech has 7 groups training samples. Moreover, the formats for recording the speech signals are 16 bit, 8 kHz, and mono. The methods in each stage of the pre-processing for speech samples and the architecture of the recognition platform ANN are illustrated in Table 1. It is worthwhile to mention that to keep a faster computation time, a condition that the length of the registers in the FPGA platform are fixed at 32 bits is imposed on the development of the recognition systems. For the specs of MFCC used to retrieve the features of speech signals, the highest frequency, lowest frequency, sampling rate and number of filters of the filter bank in Eq.(22) are set as $f_h = 4k$, $f_l = 133$, $f_s = 8k$, and $M = 40$, respectively. The frame number and frame size for the speech signal is $N = 25$ and $L = 256$, respectively. The center frequency for m-th filter in the filter bank is then be computed as

$$f_m = \frac{N}{f_s} \times fre\left[mel(f_l) + m \times \frac{mel(f_h) - mel(f_l)}{M+1}\right], \tag{22}$$

$$mel(f) = 2595 \times \log_{10}(1 + \frac{f}{700}). \tag{23}$$

After the computation of the center frequency in Eq.(22), the feature can then be retrieved from the procedure described in Section 2.5. The feature number for each frame of speech signal is set as 10.

TABLE 1. The pre-processing and architecture of ANN for Speech Recognition

Pre-processing for speech sound	Audio Frame: 256 sampling points by using Hamming Window EPD: Energy EPD Calculation for feature: Mel-Frequency Cepstrum Coefficient
Speech recognition (using ANN)	input layer: 250 neurons hidden layer: 30 neurons output layer: 10 neurons

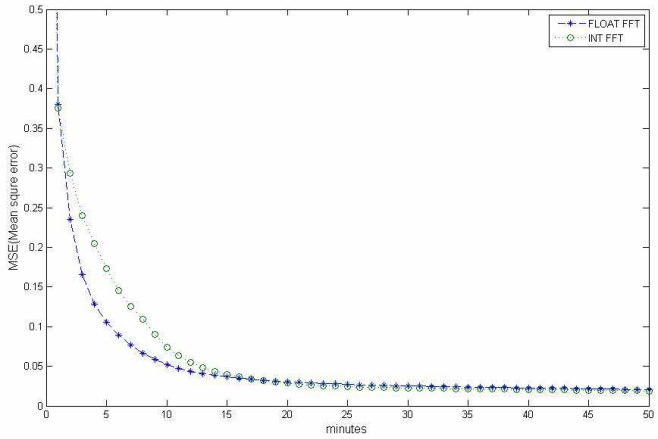

FIGURE 8. The comparison of Float FFT-ANN and Integer FFT-ANN with respects to times

The Float_ANN and Integer_ANN are then trained by back-propagation algorithm in Section 3. The step size η is tuned adaptively with initial value of 0.1 and is multiplied by a factor of 0.999 till its value up to 0.01. This adaptive η can make a faster learning rate in the experiment. The initial value of the weights in ANN are all randomized as the value between $-3.5 \sim 3.5$. In our experiment, these initial values can avoid the output of hidden layers becomes 0 or 1 which will slow down the learning speed. The error function of the output of ANN are defined as follows:

$$ERROR = \frac{1}{7 \times 9 \times 4} \sum_{k=1}^{7} \sum_{i=1}^{9} \sum_{j=1}^{4} E_{i,k,j}, \tag{24}$$

in which $E_{i,k,j}$ means the recorded voice of the j-th record of the i-th literal by the k-th person.

The comparisons between the performances of ANN trained by the features obtained from MFCC with Float FFT and Integer FFT are shown in Figure 8 and Figure 9 with respects to times and iterations of the training process, respectively. Both the two figures reveal that the convergence rate is not affected by the replace of Float FFT with Integer FFT. This means that the Integer FFT is suitable used in MFCC to reduce the computation time without any influence on the performance of ANN and hence the overall recognition systems. This fact can also be found in the Table 2 which is the comparison of MSE between ANN output by using Integer FFT and Float FFT.

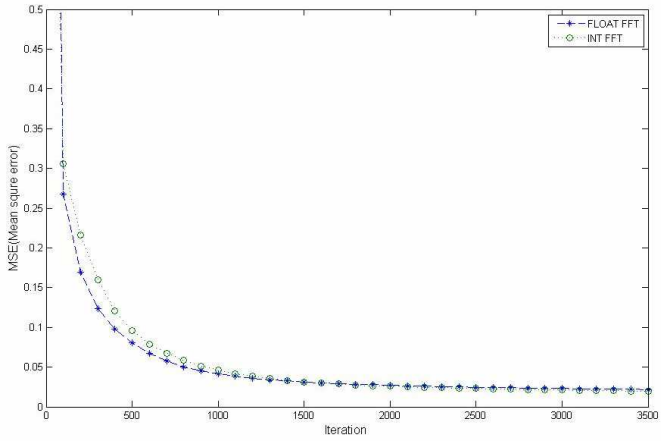

FIGURE 9. The comparison of Float FFT-ANN and Integer FFT-ANN with respects to iterations

TABLE 2. The comparison of MSE for Float FFT-ANN and Integer FFT-ANN

Iterations	MSE (Integer FFT)	MSE (Float-FFT)
200	0.216073	0.169689
400	0.120789	0.097950
600	0.078738	0.067565
800	0.058213	0.050320
1000	0.045995	0.041379
1200	0.038290	0.036114
1400	0.033277	0.032655
1600	0.029913	0.030201
1800	0.027542	0.028399
2000	0.025794	0.027040
2200	0.024464	0.025981
2400	0.023484	0.025139
2600	0.022679	0.024413
2800	0.021933	0.023750
3000	0.021236	0.023137

5.3. **Realization on embedded hardware.** In this paper, an FPGA develop platform, as shown in Figure 10, is applied to realize real-time speech recognition on the embedded system. The CPU is 100MHZ, and a SOC Builder is used for connecting the audio codec required in the experiment to control IP, CPU, SSRAM, SDRAM, and downloading all the softwares to FPGA. Finally, the Nios II is used for compiling the program.

Because the quality of recording on the develop platform may be somewhat different from that in a PC, during the process of the experiment, the recognition rate on the platform will decrease, if the speech signal samples recorded on PC are used to train the ANN. In order to prevent such a difference, we must use directly the speech signal samples recorded by the platform to be the speech signal samples for training ANN.

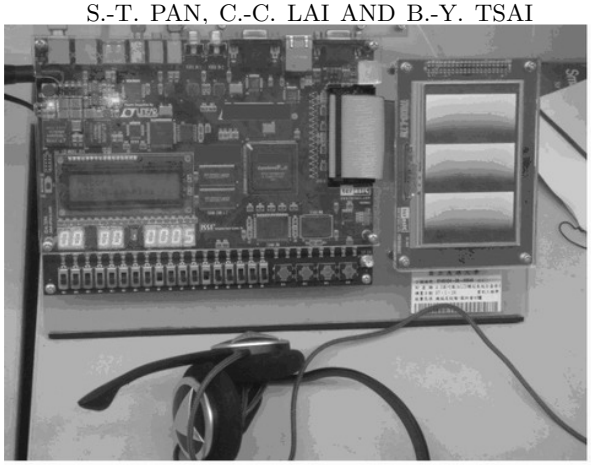

FIGURE 10. The Embedded Develop Platform used in the laboratory

TABLE 3. The comparison between the recognition time by using Float FFT and Integer FFT

Process	Time (sec.)	
	Float FFT	Integer FFT
Point detection and Pre-emphasis	0.11468	0.11468
Hamming window	0.02497	0.02497
FFT	0.88087	0.38
Feature capture	0.75311	0.75311
ANN Recognition	0.04770	0.04770
Total	1.82	1.32

The required trigonometric functions cos(.) and sin(.) for the operations in FFT are operated by means of a look-up table. For comparison of the performance of the two FFT algorithms, two look-up tables are constructed: one is for floating points, and another is for integer. Table 3 shows comparison between the recognition time by using Float FFT and Integer FFT. Obviously, we can see that the speed of the calculation of Integer FFT is faster substantially compare to that of Float FFT. The computation time for FFT is improved by using Integer FFT at the percentage of (0.88087-0.38)/0.88087 = 57%. This hence reduces the total recognition time enormously.

Table 4 shows the speech recognition rate performed directly on the embedded platform. Each speech signal is tested for 100 times, and the average recognition rate reaches 0.919, which means that the implemented embedded speech recognition system by Integer FFT can maintain a good recognition rate while the recognition time is improved. Moreover, the recognition rate of this paper is much better than that in [18] in which only 56.8% of recognition rate is reached on the FPGA platform. This means that the hardware implementation by the proposed methods in this paper outperforms that in [18].

6. **Conclusions.** This paper has shown three advantages of the proposed speech recognition system. The first is that the system is implemented on an embedded platform with FPGA chip and a SOC architecture to make the systems flexible to corporate with other

TABLE 4. The results of speech recognition rate tested via 100 groups speech

Speech sound	Recognition Correctness	Recognition Failure	Recognition Rate (%)	Average Recognition Rate (%)
0	84	16	0.84	
1	95	5	0.95	
2	97	3	0.97	
3	96	4	0.96	
4	96	4	0.96	
5	95	5	0.95	0.919
6	94	6	0.94	
7	75	25	0.75	
8	98	2	0.98	
9	89	11	0.89	

digital systems. Second, for the computation of FFT on hardware, the Integer FFT is adopted to replace Float FFT. A realization algorithm of Integer FFT on the hardware is also proposed. Through the usage of Integer FFT, the calculation time for FFT can be decreased by a factor of 57%. The calculation time decreases substantially, which conforms to the application of real-time on an embedded platform. The third advantage is that the experimental results show that the speech recognition rate is better than that of the existing papers. However, in spite of the improvement on the computation time for FFT, the recognition time of the proposed systems is still too long to be used for real-time applications. Multi-core and parallel processing for the speech recognition algorithm are necessary to further improve the recognition time and is worthwhile to examine in the future research. Besides, the simplification of the ANN architecture without reducing the recognition rate can also speed up the recognition time and is also an important topic of the future research.

Acknowledgment. This work was supported by the National Science Council, Taiwan, R.O.C., under grant NSC 99-2221-E-390-027. The authors would like to thank anonymous referees for their valuable comments and suggestions.

REFERENCES

[1] X. Wang, J. Lin, Y. Sun, H. Gan and L. Yao, Applying feature extraction of speech recognition on VoIP auditing, *International Journal of Innovative Computing, Information and Control*, vol.5, no.7, pp.1851-1856, 2009.

[2] M. Nakayama and S. Ishimitsu, Speech support system using body-conducted speech recognition for disorders, *International Journal of Innovative Computing, Information and Control*, vol.5, no.11(B), pp.4255-4266, 2009.

[3] Y. Shi, J. Liu, and R. Liu, Single-chip speech recognition system based on 8051 microcontroller core, *IEEE Trans. on Consumer Electronics*, vol.47, no.1, pp.149-153, 2001.

[4] L. D. Persia, D. Milone, H. L. Rufiner, and M. Yanagida, Perceptual evaluation of blind source separation for robust speech recognition, *Signal Processing*, vol.88, no.10, pp.2578-2583, 2008.

[5] T. Guan and Q. Gong, A study on the effects of spectral information encoding in Mandarin speech recognition in white noise, *ICIC Express Letters*, vol.3, no.3(A), pp.415-420, 2009.

[6] C. Y. Wan and L. S. Lee, Histogram-based quantization for robust and/or distributed speech recognition, *IEEE Trans. on Audio, Speech, and Language Processing*, vol.16, no.4, pp.859-873 2008.
[7] H. Sakoe and S. Chiba, Dynamic programming optimization for spoken word recognition, *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol.26, no.1, pp.43-49, 1978.
[8] M. T. Hagon, H. B. Demuth, and M. Beale, *Neural Network Design*, Thomson Learning, 1996.
[9] J. Tao, L. Xin and P. Yin, Realistic visual speech synthesis based on hybrid concatenation method, *IEEE Trans. on Audio, Speech, and Language Processing*, vol.17, no.3, pp.469-477, 2009.
[10] S. Kwong and C. W. Chau, Analysis of parallel genetic algorithms on HMM based speech recognition system, *IEEE Trans. on Consumer Electronics*, vol.43, no.4, pp.1229-1233, 1997.
[11] J. Liang and T. Tran, Fast multiplierless approximation of the DCT, *IEEE Trans. on Signal Processing*, vol.49, no.12, pp.3032-3044, 2001.
[12] W. C. Chu, *Speech Coding Algorithms*, John Wiley & Sons, Wiley-IEEE, 2003.
[13] F. Runstein and F. Violaro, An Isolated-Word speech recognition system using neural networks, *Proc. of the 38th Midwest Symposium on Circuit and Systems*, vol.1, pp. 550-553, 1995.
[14] X. Huang, A. Acero, and H. Wuenon, *Spoken Language Processing: A Guide to Theory, Algorithm and System Development*, Pearson, 2005.
[15] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, Pearson, 1999.
[16] S. Oraintara, Y. J. Chen, and T. Q. Nguyen, Integer fast fourier transform, *IEEE Trans. on Signal Processing*, vol.50, no.3, pp.607-618, 2002.
[17] F. Sadaoki and M. Dekker, *Digital Speech Processing, Synthesis, and Recognition*, Marcel Dekker, 2001.
[18] S. J. Melnikoff, S. F. Quigley, and M. J. Russell, Implementing a simple continuous speech recognition system on an FPGA, *Proc. of the 10th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, pp.275-276, 2002.