A PROGRAMMABLE APPLICATION-SPECIFIC VLSI ARCHITECTURE AND IMPLEMENTATION FOR SPEECH WORD-RECOGNIZER

An-Nan Suen, Jhing-Fa Wang, and Tswen-Duh Wang

Institute of Information Engineering
National Cheng Kung University, Tainan, Taiwan, R.O.C.
E-mail suenan@vlsi2.iie.ncku.edu.tw, wangjf@server2.iie.ncku.edu.tw

Abstract

In this paper, the efficient and flexible VLSI architecture and implementation for the voice word-recognizer processor are presented. In order to achieve a flexible and efficient VLSI realization, we use a programmable with specific core design strategy which incorporates the best aspects of both programmable and application specific signal processors to achieve high speed, high accuracy, and efficient hardware realization for the word-recognizer. On the whole, the single chip is fabricated in 0.8 μm double-metal CMOS technology after the physical design and circuit verification. The chip can process 40 MHz sampled data and it contains about 70000 transistors which occupy 0.62.x0.60 cm^2 area.

1. INTRODUCTION

The direct implementation for finite word-recognizer system with ASIC design style is too complex and hard for the designer. Another implementation method is to use the general-purpose DSP chips by programming the speech recognition algorithm in the DSP chip. However, it is computationally intensive and the general-purpose DSP chips are usually not powerful enough to handle such complex speech recognition system [2] [3]. It is obvious that the software-based microprocessor and digital signal processor approaches are very flexible, but often do not meet the performance demand. The ASIC approaches often have high nonrecurring engineering(NRE) costs, and can take considerable time and effort to fabricate and test.

In order to achieve a flexible and efficient IC realization, we use a programmable with specific core design strategy which incorporates the best aspects of both programmable and application specific signal processors to achieve high speed, high accuracy, and efficient hardware realization for the word recognition system. The merits of this architecture are: (1) more flexible due to the capable of programmable structure, (2) high speed operations resulting from the specific core designed for the bayesian neural network operation. These features lead to a high performance circuit design composed of program control modular (PCM) and bayesian neural network (BNN).

This paper is organized as follows. Section II briefly gives an overview of the bayesian neural network . Section III describes the system architecture of the programmable application-specific VLSI architecture for the finite word-recognizer. Section IV discusses the implementation and the circuits design. Finally, we make a conclusion in section V.

2. BAYESIAN NEURAL NETWORK

The BNN used in our speech recognition system contains four slabs: the input slab, the Gaussian slab, the mixture slab and the *a posteriori* slab [1]. Fig. 1 illustrates the structure of the network. Consider an isolated word utterance. The speech data from it are segmented into a sequence of frame vectors. Each frame vector is fed to the input slab in sequence. For the operation of BNN, the input slab fans out the input frame vector to the Gaussian slab. The Gaussian slab is partitioned into groups of processing elements (PEs). There is one group for each frame class which represents one frame or state of speech patterns. Thus, each Gaussian slab PE is assigned to a particular frame class. Each Gaussian slab PE calculates the Euclidean distance between its centroid and input frame class are weighted and accumulated by the mixture slab PE assigned to that frame class. This value forms the mixture density estimate for that frame class. Thereafter, the mixture density estimates from the mixture slab are weighted by the *a priori* probabilities and fed to the *a posteriori* slab. The *a posteriori* slab PEs finally produce the *a posteriori* probabilities with respect to frame classes of all reference patterns.

3. THE SYSTEM ARCHITECTURE

Fig. 2 shows the system block diagram of the programmable application-specific DSP architecture for the speech word-recognizer. The chip is implemented by the programmable application-specific technique. The ASIC direct implementation is not suitable for the finite words recognition algorithm because it is too complex and hard for the ASIC designer. Another implementation method is to adapt the general-purpose DSP chips by programming the recognition algorithm in the DSP chip. However, its computationally intensive and the general-purpose DSP chips are usually not powerful enough to handle such complex coding algorithm [2] [3]. In this paper, the best aspects of both programmable and application-specific signal processors including the performance, design complexity, and flexibility are incorporated in the processor.

The chip is composed of two major components, one is program control unit (PCU), and one is the BNN unit

[1] This work was supported by the National Science Council R.O.C. under Grant NSC84-2622-E-006-006

ASP-DAC '97
0-89791-851-7/$5.00 ©1997 IEEE

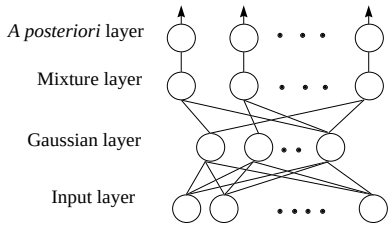

Figure 1. The architecture of the Bayesian neural network. The BNN used in our speech recognition system contains four slabs: the input slab, the Gaussian slab, the mixture slab, and the *a posteriori* slab.

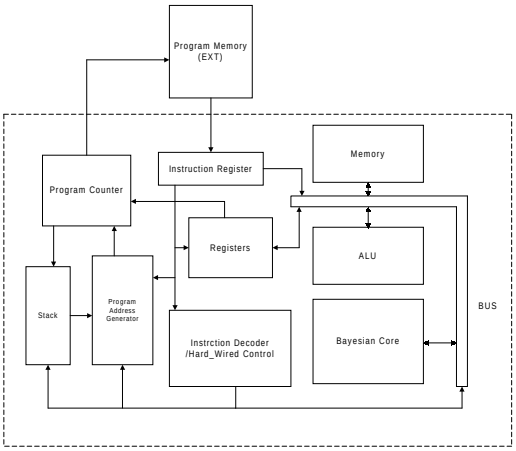

Figure 2. Block diagram of the proposed programmable application-specific DSP architecture for the finite word-recognizer system.

(BNNU). The BNNU is an ASIC core designed for the bayesian neural network operations which occupy most computation time in the recognition mode. Instruction fetching, decoding and datapath handling are done in the PCU. The PCU is composed of program counter (PC), program address generator (PAG), program memory (PM), stack, instruction register (IR), register file and the controller. The BNNU schematic is shown in Fig. 3.

Bayesian Network Unit : The bayesian network (BN) has been widely used as speech recognition template [1] which combines the merits of the Dynamic Programming (DP) and Hidden Markov Model (HMM) methods. The BN architecture is shown in Fig. 1. A Bayesian template is constructed for each reference pattern using the Bayes' rule. Each output of this Bayesian template represents the *a posteriori* probability. Then, the *a posteriori* probability is used to calculate the log distance which represents the distance between the input frame vector and the reference

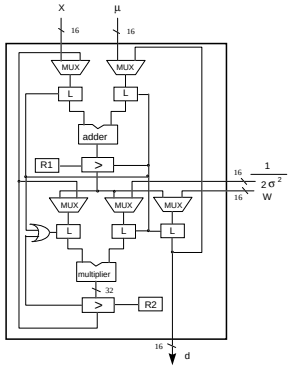

Figure 3. The schematic of the bayesian network processing unit.

pattern. The bayesian network is used as the recognition template. The BNU will output the distance $d^k_{q;i}$, between input vector X_q and the ith frame of reference pattern k. The equation of the Bayesian template can be expressed in the form:

$$d^k_{q;i} = -\log\left[\sum_{i=1}^{j^k_i}\left[P(X_q|C^k_i(j))P(C^k_i(j))\right]\right] \qquad (1)$$

where J^k_i is the number of mixture components on the Gaussian slab for class C^k_i, $P(X_q|C^k_i(j))$ is the cluster conitional probability, and $P(C^k_i(j))$ is the mixture weight between Gaussian slab and mixture slab. Fig. 3 shows the schematic of the BNN.

Pipeline Structure: Basically, our system is built on a DSP (Digital Signal Processor) architecture with its own instruction set, and includes an ASIC core,BNN unit. There are two operation modes in our chip: (1) the normal mode for general-instruction execution, and (2) the specific mode for BNN operation. When the bnn instruction is decoded, the controller stops the PC, saves the current contents of the instruction register (IREG) and simultaneously starts the BNN unit. The BNN core will perform the bayesian network operation till finished. Fig. 4 shows the pipeline structure for this chip Five pipeline stages are described as follows:

- IF: Instruction Fetch.
- ID and RegO: Instruction Decode and Register Output Enable.
- MA: Memory Access.
- EXE and RegW: Execution and Register Write Enable.
- EXE2: Execution 2. In IF stage, a new instruction is fetched out from program memory, and the PC (program counter) will update the next program address generated by PAG (program address generator).

In ID and RegO stage, the fetched instruction is written into IR(instruction register) and decoded into control signals. Besides, the choose register is permitted to output

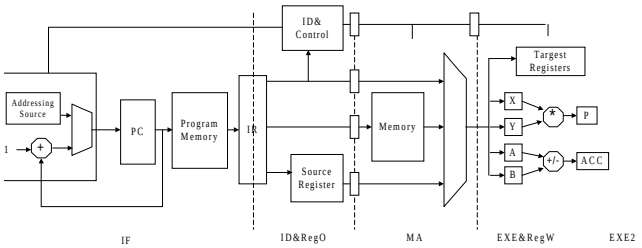

Figure 4. Pipeline structure for the finite words recognition chip.

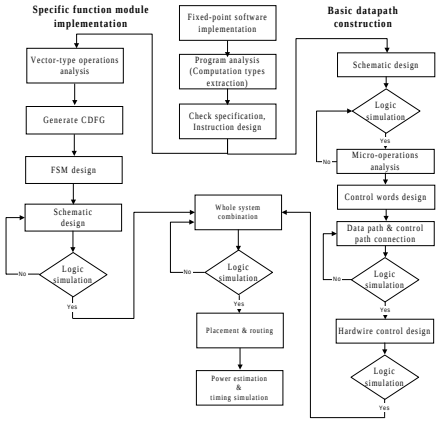

Figure 5. Design flow chart for the speech word-recognizer processor.

4. THE CIRCUIT DESIGN AND IMPLEMENTATION

The design flow chart of the proposed programmable application-specific processor for the speech word-recognizer is shown in Fig. 5. We first implement the vocoder CELP FS1016 using C language with floating-point for testing its objective and subject performance [4]. Using the fixed-point format to instead of the floating-point version is the most important and hard step in the procedure of software simulation, because it should keep the performance as well as the latter under a shorter precision. The processes of software simulation can also be applied for porting such coder algorithm to the fixed-point DSP chip. It's more an art than science for transforming a floating-point program into fixed-point one [9]. The procedure of implementing the fixed-point program will be described as follows:

Step 1. Partition the speech recognition algorithm into n modules, $M_1, M_2, \ldots, M_n$: For example, we divided the whole system into data input , Hamming window, LPC, weighted cepstrum, bayesian neural work , etc.

Step 2. Determining the lower-bound and upper-bound for each module: The speech test database was used to determine the dynamic range, the upper-bound and lower-bound for each module.The internal word length is 16 bits so that can be compatible with most of the general purpose DSP chips. Using the fraction representation, $m.n$ where $m+n=16$ to represent the data format with m-bit integer (including signed bit) and n-bit fraction.

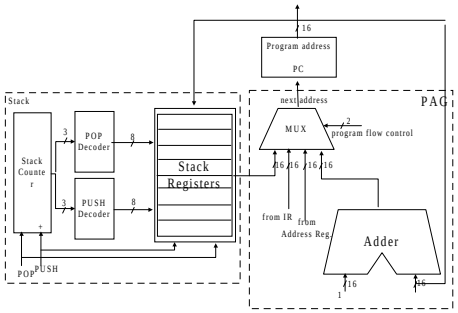

Figure 6. The circuits of PC, PAG, and stack.

Step 3. Overflow and underflow detection and avoidance: In some special case the overflow and underflow will happen, even if the data format is determined in step 2. So we add the ability of detecting the overflow/underflow and avoiding the quantization error propagating to next stage.

Step 4. Error measurement for each Stage (local test) : In the local test, SEGSNR is used for evaluating the quantization error by comparing their output with the output of the corresponding floating-point routines.

Step 5. System performance measurement (global test): In the global test, the accuracy rate are used to evaluate the performance for the fixed-point speech word-recognizer.

To provide an adequate dynamic range for the weight and activation calculations , a wordlength of 16 bits is adopted based on extensive simulations and analysis of the finite wordlength effects. To strike a balance between ease of design and operation speed, a semicustom design is preferred.

The whole circuit which contains 124 pads is designed using the semicustom design. The chip has been fabricated using the 0.8 μm CMOS double-metal technique. The layout

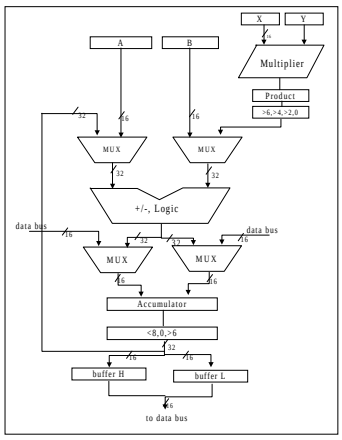

Figure 7. The circuits of processing unit, ALU.

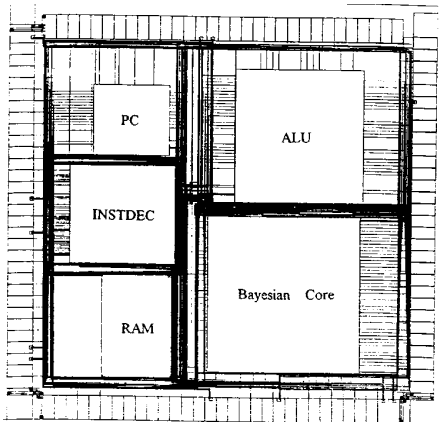

Figure 8. The layout of the single chip of the programmable application-specific finite word-recognizer.

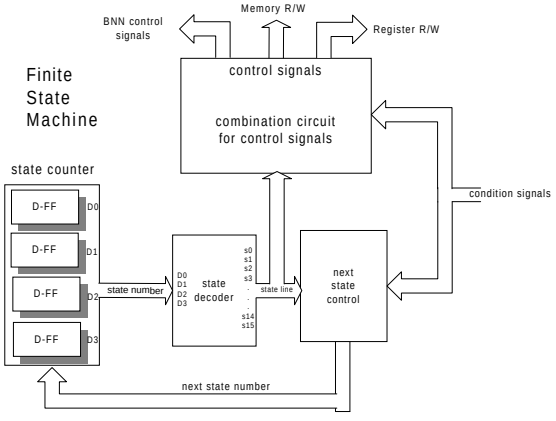

Figure 9. The finite state machine structure for the VAG controllers.

Algorithm Type	Finite words recognition system
Chip Name	Words recognizer
Data Format	Fixed point
Processing Units	BNN processor, Enhanced ALU, Accumulator, Shifter
Fabrication Technology	0.8 μm double-metal CMOS
No. of I/O Pins	124
Instruction Cycle	40 MHz
No. of Transistor	about 70,000
Chip Size	$0.60\text{x}0.62cm^2$

Table 1. Summary of the word-based speech recognition chip.

area of this chip is approximately $0.60\text{x}0.62cm^2$ as shown in Fig. 8. Table 1 summarizes some statistics of this chip.

The program counter(PC), program address generator (PAG), and stack are the main components of the program control modular. PC will update the address value at the negative clock trigger. PAG includes a 4-input mutiplexer and an adder for generating the next program address. Four candidate next addresses are from the default value, address register, instruction register, and stack. The stack is composed of eight registers which support eight-level subroutine call. Fig. 6 shows the circuits of PC, PAG, and stack.

The processor contains three independent, full function processing units: an arithmetic/logic unit (ALU), bayesian neural processing unit and a barrel shifter. Fig. 7 shows the circuits of the enhance ALU performing the basic set of arithmetic and logic operations. The ALU consists of one 16-bit 2's complement multiplier, 32-bit adder/substractor and one logic circuit. If the BNN instruction is decoded, the bayesian processing unit will execute the bayesian neural network operation. The barrel shifter is followed by accumulator which performs the logical and arithmetic shifts, normalization, and denormalization. The barrel shifter can shift right in 8 bits and left in 6 bits.

The BNN is specially designed for the Bayesian Neural Network operations in the speech training and recognition procedures. According to the behavior of BNN operation based on two Bayesian Processing unit, the CDFG and corresponding stage diagram with race free state assignment are derived first. The BNN controller is implemented by the finite state machine (FSM) consisted of state counter and decoder as shown in Fig. 9. The control unit has outputs that specify the next state. These are written into the state counter on the clock edge and become the new state at the beginning of the next clock cycle following the active clock edge.

5. CONCLUSION

In this paper, we have proposed the VLSI architecture for the speech word-recognizer using the programmable application-specific technique which incorporates the best aspects of design complexity, flexibility, and operation speed. The merits of the word-recognizer processor are: (1). more flexible due to the programmable structure, (2) high speed operations resulting from the specific BNU core

design, and (3) training, recognition, and speech synthesis three operating modes are included. Such an implementation strategy, from the software simulation, to the programmable application-specific chip design can also be applied to the speech coding, image processing and some other speed demanded DSP systems.

REFERENCES

[1] Chung-Hsien Wu, Jhing-Fa Wang, Chang-Ching Haung, and Jau-Yien Lee, "Speaker independent recognition of isolated words using concatenated neural networks," *Int. Journal of Pattern Recognition and Artificial Intelligence, vol. 5, no. 5*, 1991, pp. 693-714.

[2] D. C. Chen , and J. M. Rabaey," A Reconfiguratable Multi- processor IC for Rapid Prototyping of Algorithmic-Specific High-Speed DSP Data Paths," *IEEE J. Solid-State Circuits*, vol. 27, no. 12, Dec., 1992, pp.1895-1904.

[3] Kunitoshi Aono et al., "A Video Digital Signal Pressor with a Vector-Pipeline Architecture," *IEEE Journal of Solid -State Circuits*," December 1992, pp. 1886-1893.

[4] J. F. Wang, A. N. Suen, and C. K. Chieh, "A programmable application-specific architecture for real-time speech recognition," in *6th VLSI Design/CAD Symposium*, Aug., 1995, pp.261-264.

[5] J.F. Wang, A. N. Suen, J.R. Lee, and C.H. Wu, "A bayesian neural network chip design for the speech recognition system," in *Proc. IEEE Int. Conf. on Neural Network,* Australia, Nov., 1995.

[6] R. D. Fellman, "Design Issues and an Architecture for the Monolithic Implementation of a Parallel Digital Signal Processor," *IEEE Trans. Acoust. Speech, Signal Processing*, vol, 38, no.5, pp. 839-853, May. 1990.

[7] Edward A. Lee, "Programmable DSP Architecture Part I," *IEEE ASSP Magazine*, Oct. 1988, pp. 4-19.

[8] Edward A. Lee, "Programmable DSP Architecture Part II," *IEEE ASSP Magazine*, Jan. 1989, pp. 4-15.

[9] P.Kroon, and K. Swaminathan,"A high-quality multirate real-time CELP coder,"*IEEE J. Select. Areas commun.*,vol. 10, no. 5, Jun. 1992, pp.850-857.

[10] A. Peled, and B. Liu, "A new hardware realization of digital filters, "*IEEE Trans. Acoust. Speech, Signal Processing* , vol. ASSP-22, no. 6, 1976, pp.456-462.