A Real-Time Implementation of Isolated Word Recognition System Based on a Hardware-Efficient Viterbi Scorer

효율적인 하드웨어 구조의 Viterbi Scorer를 이용한 실시간 격리단어 인식 시스템의 구현

Yun-Seok Cho*, Jin-Yul Kim**, Kwang-Sok Oh***, and Hwang-Soo Lee****
조 윤 석*, 김 진 율**, 오 광 석***, 이 황 수****

Abstract

Hidden Markov Model (HMM)-based algorithms have been used successfully in many speech recognition systems, especially large vocabulary systems. Although general purpose processors can be employed for the system, they inevitably suffer from the computational complexity and enormous data. Therefore, it is essential for real-time speech recognition to develop specialized hardware to accelerate the recognition steps.

This paper concerns with a real-time implementation of an isolated word recognition system based on HMM. The speech recognition system consists of a host computer (PC), a DSP board, and a prototype Viterbi scoring board. The DSP board extracts feature vectors of speech signal. The Viterbi scoring board has been implemented using three field-programmable gate array chips. It employs a hardware-efficient Viterbi scoring architecture and performs the Viterbi algorithm for HMM-based speech recognition. At the clock rate of 10 MHz, the system can update about 100,000 states within a single frame of 10 *ms*.

요 약

HMM을 이용한 알고리즘은 대용량 음성인식 시스템을 비롯하여 많은 시스템에 적용되어 왔다. 음성인식 시스템을 범용의 프로세서들을 가지고 구현할 경우 많은 계산량과 데이타들로 말미암아 실시간의 성능을 얻을 수 없다. 따라서 실시간 음성인식을 위해서는 인식을 가속화 시키기 위한 전용 하드웨어를 개발하는 것이 요구되어진다.

본 논문에서는 HMM을 이용한 격리단어 인식 시스템을 구현하는 내용을 다루고 있다. 음성인식 시스템은 호스트 컴퓨터와 DSP 보드 그리고 프로토타입 Viterbi scoring 보드로 이루어져 있다. 음성신호로부터 특징 벡터를 추출하는 과정은 DSP 보드에서 이루어지고, Viterbi scoring 보드는 세개의 field-programmable gate array 칩들을 사용하여 설계되었다. Viterbi scoring 보드는 하드웨어적으로 효율적인 Viterbi scoring 구조를 채택하고 있고 음성인식을 위한 Viterbi 알고리즘을 수행한다. 제작된 시스템은 10 MHz로 동작하고, 한 프레임 즉 10 ms 동안에 100,000 스테이트를 처리할 수 있다.

*Department of Electronic Engineering, Handong University
**Department of Electronic Engineering The University of Suwon
***Samsung Electronics Co., LTD
****Department of Information and Communication Engineering KAIST
접수일자 : 1994년 9월 8일

I. Introduction

The hidden Markov model (HMM)-based algorithm has been successfully applied to speech recognition since the hidden Markov modeling method provides a robust modeling capability of speech signal and still maintains high recognition accuracy. An isolated word recognition system using HMM consists of two phases : training phase and scoring(recognition) phase. The training phase estimates the HMM parameters and results in a distinct HMM for each word of the vocabulary by using training set of observations. The scoring phase consists of computing the probability of generating the test word with each word model, and choosing a word model that gives the highest probability as the recognized word. The Viterbi algorithm [1] is usually employed in this phase, which is an efficient technique to find the best matching word by comparing received utterance with models in memory.

In order to process the HMM-based algorithm in real-time, we need to design a VLSI architecture which is dedicated to the Viterbi algorithm employed in the scoring phase. Yet only a few VLSI architectures have been reported [2, 3, 4]. One approach is to recognize that the Viterbi algorithm is a kind of dynamic programming (DP) problem and to design architectures that can solve a class of general DP problems [4, 5]. Although they are flexible in their use and applicable to a wide range of DP problems, they inevitably suffer from inefficiency in hardware utilization when applied to the Viterbi scoring procedure used in speech recognition, due to the following properties of HMM :

1. Most states are locally connected to only three or fewer preceding states.

2. The transitions between states exhibit quite irregular structures.

3. The amount of data required during the processing is huge.

Property 1 states that the state transition matrix becomes banded and upper-triangular, and that the hardware utilization in the architectures optimized for general DP problems in which the state transition matrices are very dense or complete will be very low if applied to speech recognition. For example, the ring-connected systolic array [5] is efficient only when the state transition matrix is dense ; almost all of processing elements (PEs) will do meaningless works when the state transition matrix is very sparse. Therefore, the architecture design for the Viterbi scoring procedure used in speech recognition should be considered in a quite different basis from general DP problems.

There are some previous works that have reflected, in part, the properties of HMM on the architecture design [2, 3]. In [2], a single customized VLSI processor is employed for a Viterbi scoring data path. The data path uses three dual-ported memories to overcome the irregularity of the transitions between states. Three identical copies of a state metric are stored in three different dual-ported memories so that three interim state metrics can be computed at the same time. Therefore, three relative addresses to the preceding states should be maintained. Also the single processor is responsible for serially computing all of the calculations required to generate the state metrics.

A parallel architecture for the Viterbi scoring procedure was reported in [3]. They insist that only seven clock cycles are required to process an observation symbol no matter how many states are in HMM. However when this architecture is actually applied to speech recognition, a tremendous amount of hardware will be required to maintain the computational speed of seven clock cycles because a state processor is assigned for each state composing a word.

This paper describes the implementation of an HMM-based real-time speech recognition system which employs our proposed architecture [6], and thus the system can expedite the HMM scoring

steps for large vocabulary recognition systems.

Ⅱ. Overview of Isolated Word Recognition

The block diagram of an isolated word recognition system is shown in Fig 1. The input speech signal from a microphone is filtered with a low pass filter for anti-aliasing and line noise reduction. Then the filtered signal is sampled at 8 kHz by an analog-to-digital converter, and the endpoint detection to find the end points of the sampled speech signal is performed. The endpoint detection algorithm uses the short-time energy and the zero-crossing rate of the sampled speech signal whose thresholds are adjusted during the silence period. Given the end points of the speech signal, the next step is to filter the input signal with a pre-emphasis filter to flatten spectrum by emphasizing the high frequency components of the input signal.

In order to extract features, the pre-emphasized signals are blocked into frames of 30 *ms* (240 points) with each consecutive frame spaced at 10 *ms* (80 points) apart, where time interval of one frame is 10 *ms*. Each frame is multiplied by the Hamming window, and then the autocorrelation analysis is performed for obtaining LPC coefficients. From the LPC coefficients, we derive the LPC cepstral coefficients and apply the bilinear transform for mel-scaling. This mel-scaled LPC cepstral vector is used as the input feature vector of HMM. An input feature vector is obtained every 10 *ms*. Finally, the feature vectors are vector quantized, and the index that is the output of vector quantization (VQ) is used as the output symbol.

With the coming VQ indices, the Viterbi scorer shown in Fig 1 computes the state metrics for all states composing each candidate word in the vocabulary and generates the word index for which the state metric is maximum among candidate words.

Ⅲ. Viterbi Scoring Procedure

The major task in speech recognition is to find the best matching word by comparing input utterance with speech models in memory. Given an observation symbol sequence $\{O_1, O_2, \cdots, O_T\}$, the following logarithm-integer version of the original Viterbi algorithm [7] is performed for each word model λ^v, $1 \le v \le V$

(1) Initialization :

$$S_1(1) = b_1(O_1),\ S_1(i) = -\infty,\ 2 \le i \le N,$$

(2) Recursion :

$$S_t(j) = \max_{1 \le i \le j} \{S_{t-1}(i) + a_{ij} + b_{ij}(O_t)\},\ 2 \le t \le T,\ 1 \le j \le N,$$

(3) Termination :

$$P_v = S_T(N),$$

where N denotes the number of states in the model, and $\{a_{ij}\}$ and $\{b_{ij}(O_t)\}$ are obtained from the state transition and the output probabilities, respectively, by taking logarithmic transformation followed by quantization, as explained below. We choose a word with the highest P_v as the recognized word.

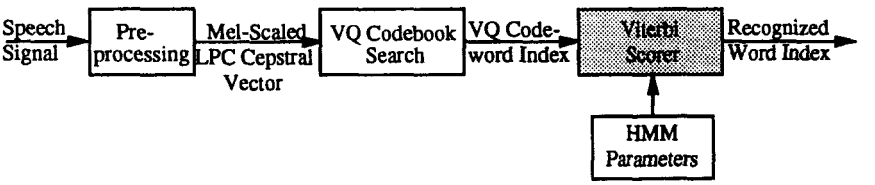


Fig 1. An HMM-based isolated word recognition system.

Let $\{S(j)\}$, $\{a_{ij}\}$, and $\{b_{ij}(O_t)\}$ denote the state metric, the transition metric, and the output metric, respectively. Since multiplications in the original Viterbi scoring procedure are time consuming operations, they are converted to additions by taking logarithms of the probabilities, and underflow is avoided. Floating-point numbers are converted into integers for finite precision hardware implementation. These steps are performed by using the following rules :

$$a_{ij} = ulog(\alpha_{ij}),$$

$$b_{ij}(O_t) = ulog(\beta_{ij}(O_t)),$$

where $ulog(x) = C \cdot Int(log_{10}(x)) + \Delta$.

The function $ulog(x)$ maps a real number x $(0 \leq x \leq 1)$ into an integer number in the range controlled by coefficients C and Δ.

We used the trained HMM parameters which showed that the recognition accuracy was 96% in floating-point simulation [8]. To examine the effect of quantization, a 16 bit number is used for representing each state metric and a 8 bit number is used for representing each of the transition and the output metrics. The value of Δ is chosen to be 2^8-1 since the transition and output metrics are represented by 8 bit numbers. The exact value of C depends on the actual vocabulary employed for test, and it must be determined so as to maximize the dynamic range of the transition metrics and the output metrics under finite precision. For our simulation, C was chosen to be 51. Simulations using these quantized data showed that the recognition accuracy was 96%. Only a small degradation in recognition rate was observed in spite of logarithmic compression and quantization.

IV. Hardware-Efficient Viterbi Scorer

This section describes the design procedure of the proposed PE, which has been reported by ours in [6]. We briefly review it again since a prototype isolated word recognition system developed here employs it. The design procedure of PE is illustrated in Fig 2.

The key observation is that, in many HMM topologies, most states stay only for a few time steps due to the local connection and thus the number of states to be stored is very small. In Fig 2(a) the states that are required to evaluate the next state(denoted by the superscript +) are listed on the right side of each state : state 2^+ requires the metrics of state 1 and 2, and state 7^+

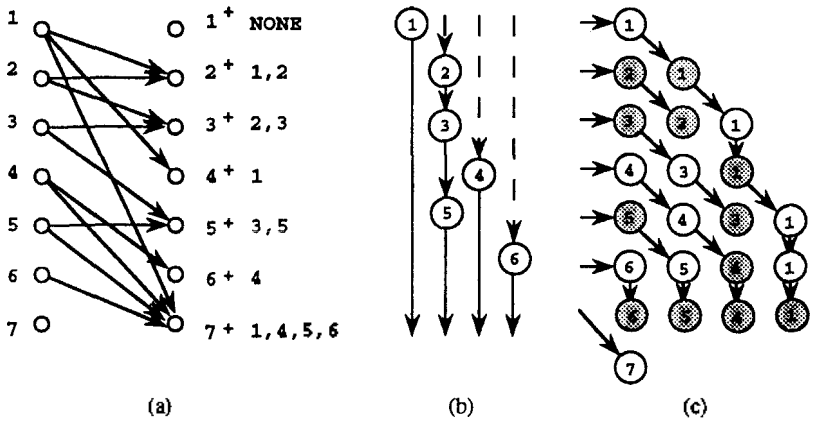


Fig 2. (a) An trellis, (b) Life-time analysis, (c) Management of the passing list.

requires the metrics of state 1, 4, 5, and 6 only. Based on this list, we can analyze the life time of the states (Fig 2(b)) and construct a scheme to pass the *live* states downward for computing the next remaining states (Fig 2(c)). In Fig 2(c), the shaded states participate in evaluating the next state, while the unshaded ones are just passed downward for later use. If a state is identified not to be used any longer, then the state is removed from the passing list which is defined to be the set of state metrics passed downward for next computation : state 1 is maintained in the passing list until step 7 since it participates in computing the metric of state 7, however state 2 is removed from the passing list after step 3.

A simple PE results from these observations and is shown in Fig 3. PE consists of an elastic storage and an add compare-select (ACS) circuit. The elastic storage is implemented using four pairs of multiplexers (MUXs) and D type registers, and the ACS circuit is composed of eight adders and three maximizers. MUXs control the flow of state metrics and D-type registers store the metrics in the passing list. The state metrics are fed to the top-left sequentially, and they are either passed to the adjacent D registers, or recirculated in the current registers, or removed from the passing list according to the control signals applied to MUXs. Each MUX is controlled by a 1 bit control signal. The proposed PE can accommodate various HMM topologies by modifying the 4-bit signals that control the flow of the state metrics in the elastic storage. The state metrics maintained in the elastic storage, i.e., those in the passing list, are supplied to the ACS circuit and the corresponding transition metrics and output metrics are added to make four interim state metrics. Of these interim state metrics, the maximum value is selected and this state metric is passed to the adjacent trellis stage for the next recursion.

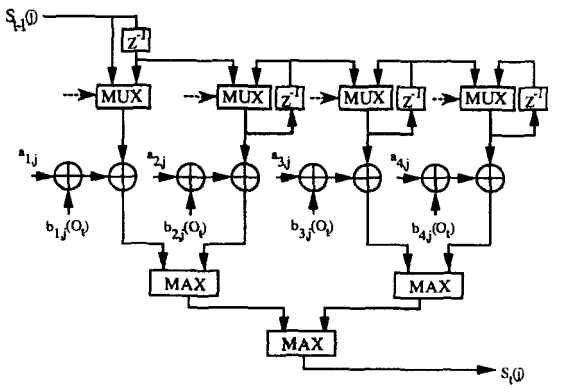


Fig 3. Basic PE structure for the Viterbi scorer.

V. A Prototype Isolated Word Recognition System

A prototype system capable of processing 100,000 state metrics in real-time has been implemented. It is dedicated to isolated word recognition based on HMM. A DSP board is employed for the front-end signal processing and a prototype Viterbi scoring board is designed for HMM scoring. We adopt the processing element shown in Fig 3 as the core processor for processing the logarithm integer Viterbi algorithm. Also, we use 16-bit values for state metrics and 8-bit values for transition and output metrics.

5.1 Overall System Structure

The system includes a personal computer (PC), an Elf DSP board from Atlanta Signal Processors Inc., and a dedicated Viterbi scoring board. This is depicted in Fig 4. Two added boards exchange the data through PC ISA bus. Before starting the speech recognition, PC initializes the Viterbi scoring board. The Viterbi scoring board contains reprogrammable filed-programmable gate array (FPGA) chips and memory blocks. The FPGA chips are configured with designed data, and the HMM parameters are downloaded to the memory blocks. Then a DSP program is run on the DSP board in the Ashell environment, and the Ashell environment enables the program to access the

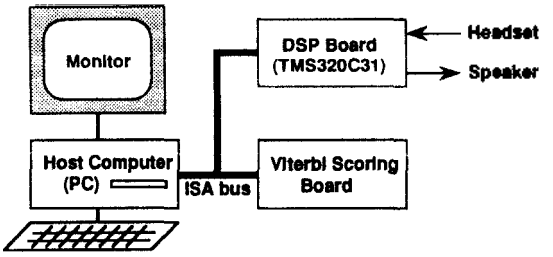


Fig 4. Overall configuration of the dedicated speech recognition system.

host's disk, keyboard, and display screen.

A/D and D/A converters are built in the DSP board. DSP board reads the sampled speech signal from a headset and performs the detection of end-points. Then a linear predictive coding (LPC) vector is computed from autocorrelation coefficients using the Levinson-Durbin algorithm [9], and a mel-scaled cepstral feature vector is derived from it. With the feature vector, DSP generates the corresponding VQ index and sends it to the Viterbi scoring board. The Viterbi scoring board scores all state metrics for all the vocabulary words with the VQ index. This procedure continues until the end-point of speech signal is detected. At the end of speech signal, the Viterbi scoring board reports an index to the word that is most probable among all the candidate words.

5.2 Front-End Signal Processing

The DSP board is responsible for the front-end part of speech recognition, that is, from the A/D conversion to the generation of VQ indices. The speech signal from a headset is digitized by A/D converter, built-in DSP board, at 8 kHz sampling frequency. Then the board detects the end points of speech signal. Once the start point of speech is detected, the input signal is pre-emphasized by a filter with the transfer function of $1-0.95z^{-1}$. The pre-emphasized signal is blocked into three frames and the autocorrelation analysis is performed to obtain the LPC coefficients. An LPC cepstral vector is then computed, and is mel-scaled by bilinear transform. The mel-scaled cepstral vector is vector quantized, and then the index that is the output of vector quantization is sent to the Viterbi scoring board. This procedure repeats every 10 *ms* (duration of a single frame) until end-point signal indicating the end of speech is detected.

Here, the major works to be performed in the DSP board are summarized.

FrontEnd_Signal_Proc :

1. Initialize the data structure and A/D session.
2. Extract the feature vectors :
   (a) Wait until start-point of speech is found.
   (b) Get one frame of data.
   (c) Preemphasize the data overlapped by two frames.
   (d) Multiply the coefficients corresponding to Hamming window to each sample.
   (e) Perform autocorrelation analysis and computed the LPC coefficients.
   (f) Transform the LPC coefficients to the cepstral coefficients.
   (g) Mel-scale the coefficients.
3. Map the mel-scaled cepstrum coefficients to VQ index.
4. Go to step 2.(b) unless the end-point signal is found.
5. Go to step 2.(a) in order to restart.

5.3 Design of the Viterbi Scoring Board

The overall block diagram of the board is shown in Fig 5. The Viterbi scoring board is a slave device attached to the PC ISA bus. It consists of three FPGA chips, four memory blocks, and a clock generator. Among several FPGAs, we have selected XC4010 FPGA to implement the processing element and the control logic. One XC4010 device has about 10,000 gates, 160 user I/O pins, and 20×20 configurable logic blocks (CLBs). Three FPGA chips contain almost all digital logic ; one

processing element and the associated control logic. Each of them is labeled as FPGA1, FPGA2, and FPGA3.

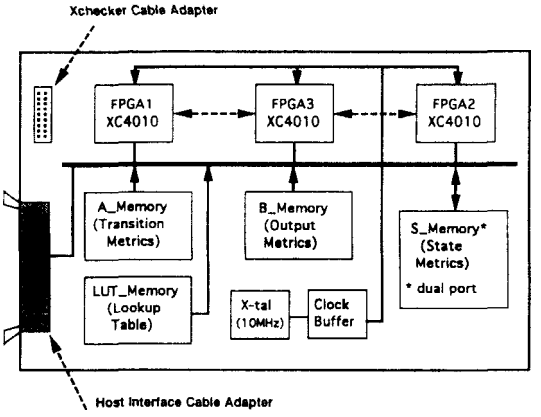


Fig 5. Overall block diagram of the Viterbi scoring board.

FPGA1 contains the register ports for communicating to the host computer, the logic for downloading the data from the host computer to memories, and the circuits for generating the address for LUT_Memory and S_Memory. Also it generates most control signals which control the main operation of the Viterbi scoring board along with the signals set by host computer in the recognition phase. FPGA2 generates the output metrics when multiple codebooks are employed for output observation probability, and distributes the output metrics. It also has a ROM which contains the information on the HMM topology. The addresses for A_Memory and B_Memory are also generated in FPGA2. FPGA3 implements a processing element with pipelined registers and a normalization circuit that prevents an arithmetic overflow.

Each of four memory blocks is labeled as A_Memory, B_Memory, S_Memory, and LUT_Memory. A_Memory, B_Memory, and S_Memory contain the transition metrics, the observation metrics, and the state metrics, respectively. LUT_Memory stores the information on all the vocabulary words.

Configuration of FPGAs in the Viterbi Scoring Board

The configuration of FPGAs is the process of loading design-specific programming data into one or more FPGA devices to define the functional operation of the internal blocks and their interconnections. The XC4000 family uses about 350 bits of configuration data per CLB and its associated interconnects. The XC4000 family has six configuration modes selected by a 3-bit input code applied to the M0, M1, and M2 inputs. There are three self-loading master modes, two peripheral modes and the serial slave mode used primarily for daisy-chained devices. In this design, each of FPGAs has been configured as the serial slave mode by applying all ones to the M0, M1, and M2 inputs. The configuration data are transmitted through Xchecker cable [10].

Startup of the Viterbi Scoring Board

The behavior of the Viterbi scoring board is controlled by a register port CTLPORT in the board. The function of each bit in CTLPORT is summarized in Table 1. All bits are set to 0 at reset.

Table 1. Definition of control port register

Bit	Name	Function
0	Download	When this bit is set to 1, the download of HMM parameters to memory is performed.
1	Start	The Viterbi scoring board is now ready state when this bit is set to 1.
2	FrameSync	Whenever feature vectors extracted and their VQ indices are obtained, this bit is set to 1. At the first frame of speech, FrameSync bit must be set to 1 along with Start.
3	End	This bit is set to 1 when the end-point signal is detected.

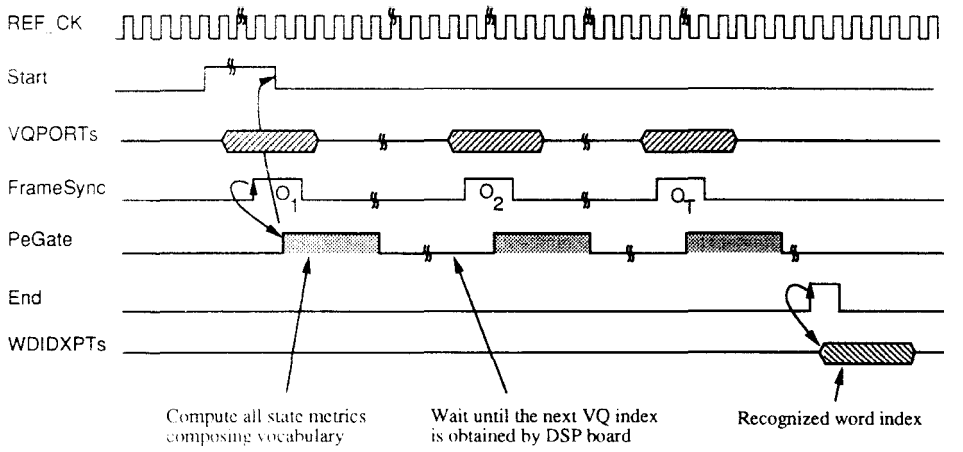


Fig 6. Overall timing diagram of the Viterbi scoring board.

The overall timing diagram of the Viterbi scoring board is shown in Fig 6. In order to start up the board, we must configure the board with an FPGA configuration file and HMM parameters. The next step is to set the Start bit of CTLPORT to 1, and VQ indices are written to the board, and then set the FrameSync bit of CTLPORT to 1. With these Start and FrameSync signals, the Viterbi scoring board updates all state metrics in vocabulary. At the end of speech signal which is informed by the DSP board, we set the End bit of CTLPORT to 1. Finally the recognized word index can be read from the Viterbi scoring board. The procedure to be performed by the system is summarized below.

START_SpeechRecSystem_OnLine :

1. Initialize candidate words with predefined vocabulary.
2. Set Start bit of the control register to 1.
3. Recognition :
   (a) Wait until the starting point of speech signal is found.
   (b) Extract feature vectors and map them onto VQ indices.
   (c) Send VQ indices to the Viterbi scoring board.
   (d) Set the FrameSync bit to 1.
   (e) If the end point of speech signal is not found, go to 3.(b).
4. Get the recognized word index :
   (a) Set the End bit of the control register to 1.
   (b) Get a recognized word index from the board.
   (c) Display or sound the result and go to step 2.

5.4 Performance

The prototype Viterbi scoring board is shown in Fig 7. The board can operate at 10 MHz clock rate. Since a feature vector is obtained every 10 *ms*, if the clock period for the Viterbi scoring board is t_p μs and the number of states is 50 per word, the throughput of the dedicated board is estimated as :

$$\frac{10\,ms}{50\,states/word \times t_p \mu s/state} = \frac{200}{t_p}\ words.$$

Therefore, with $(t_p \mu s)^{-1} = 10$ MHz, the board can process 2,000 words or equivalently 100,000 states in real-time.

VI. Conclusions

In this paper, we present a real-time isolated word recognition system based on a hardware-efficient Viterbi scorer. The system is composed of

a host computer (PC), a DSP board, and a prototype Viterbi scoring board. The data exchange between the boards is done through the PC ISA bus. The DSP board does the front-end processing of speech signal. The Viterbi scoring board, which has been implemented using three FPGA chips and memory blocks, performs the Viterbi algorithm for HMM-based speech recognition. The system is capable of computing 100,000 states in real-time, or equivalently, 2,000 words when the average number of states per word is 50.

References

1. G. D. Forney, Jr., "The Viterbi algorithm," *Proc. of the IEEE*, vol. 61, No. 3, pp. 268-279, Mar. 1973.
2. H. Murveit *et al.*, "A large-vocabulary real-time continuous speech recognition system," *Proc. ICASSP 89*, pp. 789-792, 1989.
3. W. J. Yang and H. C. Wang, "Parallel architecture for HMM scoring procedure," *Signal Processing II: Theories and Applications*, Elsevier Science Publishers, pp. 1251-1254, 1988.
4. G. M. Quenot *et al.*, "A dynamic programming processor for speech recognition," *IEEE J. Solid-State Circuits*, vol. 24, No. 2, pp. 349-357, Apr. 1989.
5. W. G. Bliss and L. L. Scharf, "Algorithms and architectures for dynamic programming on Markov chains," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, No. 6, pp. 900-912, Jun. 1989.
6. Y. S. Cho, J. Y. Kim and H. S. Lee, "Efficient Viterbi scoring architecture for HMM-based speech recognition systems," *IEE Electronics Letters*, vol. 28, no. 25, pp. 2338-2340, Dec. 1992.
7. L. R. Rabiner, S. E. Levinson and M. M. Sondhi, "On the application of vector quantization and hidden Markov models to speaker-independent, isolated word recognition," *AT&T Bell Syst. Tech. Jour.*, vol. 62, No. 4, pp. 1075-1105, Apr. 1983.
8. M. W. Koo, C. K. Un, H. S. Lee, J. K. Koo and H. R. Kim, "A comparative study of speaker adaptation methods for HMM-based speech recognition," *Proc. International Conference on Spoken Language Processing*, Kobe, Japan, Nov. 1990.
9. L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*, Prentice Hall, 1978.
10. Xilinx, *XACT User's Guide*, 1992.

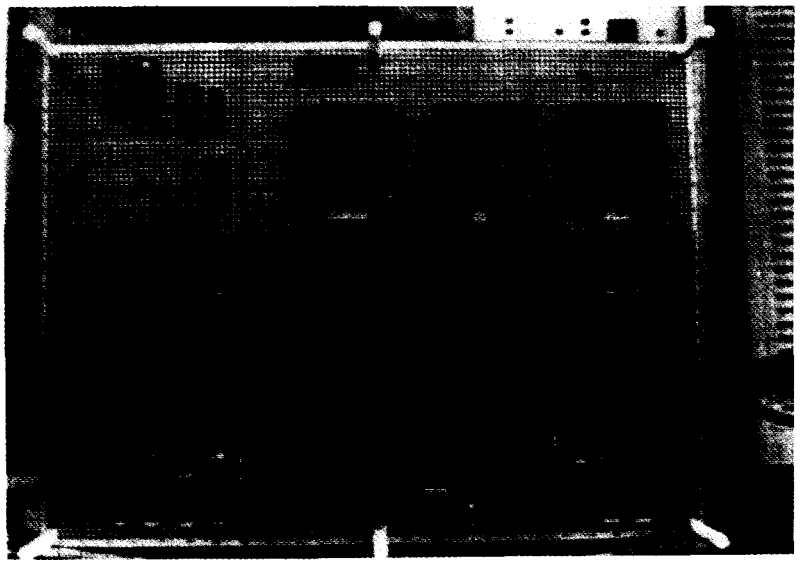
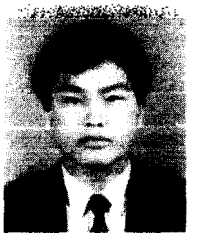


Fig 7. A prototype Viterbi scoring board.

▲Yun-Seok Cho

Yun-Seok Cho received the B.S. degree in Electronic Engineering from Kyungpook National University, Taegu, in 1987, the M.S. degree in Electrical Engineering from KAIST, Seoul in 1989, and the Ph.D. degree in Electrical Engineering from KAIST, Daejeon in 1994. Since graduation, he has been with the Department of Information and Communication Engineering at KAIST Seoul Campus, where he has served as an engineer for the development of speech recognition system. His research interests include speech recognition, VLSI architectures, and digital signal processing.

▲Jin-Yul Kim



Jin-Yul Kim received the B.S. degree in the department of electronics engineering from the Seoul National University, in 1986, and the M.S. and Ph.D. degrees in the department of electrical engineering both from the Korea Advanced Institute of Science and Technology (KAIST), in 1988 and 1993, respectively. In 1993 he was a research engineer in the department of information and communications engineering at KAIST. He is currently with the department of electronic engineering at the University of Suwon, as an instructor. His research interests include applications of digital signal processing, digital communications, and parallel processing architectures and computer-aided design for digital signal processing.

▲Kwang-Sok Oh

Kwang-Sok Oh was born in Boun, Korea on March 6, 1963. He received the B.S. degree in avionics from Korea Aviation College in 1985 and the M.S. degree in Information and Communication Engineering from KAIST Seoul Campus. He has been a research engineer at Samsung Electronics Co., LTD. since 1984. His current research interests are the area of speech signal processing and VLSI architecture for digital signal processing.

▲Hwang-Soo Lee

Hwang-Soo Lee was born in Seoul, Korea, in 1952. He received the B.S. degree in electrical engineering from Seoul National University, Seoul, in 1975, and the M.S. and Ph.D. degrees in electrical engineering from the KAIST, Seoul in 1978 and 1983, respectively. In 1975 he was with the Hyundai Heavy Industries Co., where he was involved in designing marine instrumentation and automatic navigation systems. In March 1983, he joined KAIST as an Assistant Professor of electrical engineering. He is currently a Professor in the Department of Information and Communication Engineering at KAIST Seoul Campus. His current research interests include spoken language processing, digital data transmission, and design of mobile communication systems.