A 40-nm 168-mW 2.4×-Real-Time VLSI Processor for 60-kWord Continuous Speech Recognition

Guangji He, Takanobu Sugahara, Shintaro Izumi,

Hiroshi Kawaguchi, and Masahiko Yoshimoto

Kobe University, Kobe, 657-8501 Japan
achilles@cs28.cs.kobe-u.ac.jp

***Abstract*— This paper describes a low-power VLSI chip for speaker-independent 60-kWord continuous speech recognition based on a context-dependent Hidden Markov Model (HMM). Our implementation includes a compression–decoding scheme to reduce the external memory bandwidth for Gaussian Mixture Model (GMM) computation and multi-path Viterbi transition units. We optimize the internal SRAM size using the max-approximation GMM calculation and adjusting the number of look-ahead frames. The test chip, fabricated in 40 nm CMOS technology, occupies 1.77 mm × 2.18 mm containing 2.52 M transistors for logic and 4.29 Mbit on-chip memory. The measured results show that our implementation achieves 34.2% required frequency reduction (83.3 MHz) and reduces 48.5% power consumption (74.14 mW) for 60 k-Word real-time continuous speech recognition compared to the previous work. This chip can maximally process 2.4× faster than real-time at 200 MHz and 1.1 V with power consumption of 168 mW.**

Keywords—40 nm VLSI, hidden Markov model (HMM), large vocabulary continuous speech recognition (LVCSR) 2.4×

I. INTRODUCTION

High-end personal computers can accommodate speech recognition tasks well even with large acoustic and language models [1]. However, such methods are not applicable for mobile systems while considering the physical size and power consumption [2]. Additionally, they are unsuitable for next-generation applications such as audio mining, which request the recognizer to deliver results at rates that are 10×, 100×, or 1000× faster than real-time [3, 4]. Hardware implementation by VLSI or an FPGA is a good approach to satisfy these demands because of its good processing speed and power consumption. Lin et al. reported a Multi-FPGA implementation for 5 k-word continuous speech recognition [5] that achieves 10× faster than real time, but the system is not extendable for larger vocabularies because it is not cost-effective. It needs two FPGAs and two DDR2 DRAMs each with a 64-bit wide data-path. Yoshizawa et al. proposed a scalable architecture for speech recognition [6]. Their chip can have an adjustment between vocabulary size and processing speed, but the system only offers real-time performance with a limited vocabulary of 800 words. Choi et al. developed FPGA and VLSI implementations for 20 k-word speech recognition [7, 8]. They implemented a special memory interface for several parts of the recognition engine to apply optimized DRAM access, which improves the data transfer efficiency, but the numerous external DRAM accesses cause high IO frequency, which requires a high supply voltage and high power consumption in both the FPGA side and DRAM side. In the prior work [9], we presented a VLSI processor for real-time continuous 60-kWord continuous speech recognition. It employs some algorithm optimization such as two-stage language model (LM) search and specialized cache architecture. We reduced 95% of the external memory bandwidth and 78% of required frequency. It is the first hardware-based recognizer that can recognize speech in real-time with 60-kWord models. Nevertheless, its processing speed is limited and the internal RAM size reaches 7.8 Mbit, occupying a large area.

As described herein, we propose a compression–decoding scheme to reduce the external memory bandwidth for Gaussian Mixture Model (GMM) computation and multi-path Viterbi transition units to hide the DRAM latency when a mis-hit occurs during high-speed processing. We use the max-approximation GMM calculation to save the add-log table and for adjusting the result RAM for look-ahead frames. We designed and fabricated a VLSI test chip in 40 nm CMOS technology and measured its performance. Results show that the developed chip achieves 34.2% required frequency reduction and 48.5% power consumption reduction compared to our previous work [9] for performing 60 k-Word continuous real-time speech recognition. This chip can maximally process 2.4× faster than real-time at 200 MHz under standard supply voltage (1.1 V). A comparison of the vocabulary size and processing speed among recently announced hardware-based speech recognizers is shown in Fig. 1.

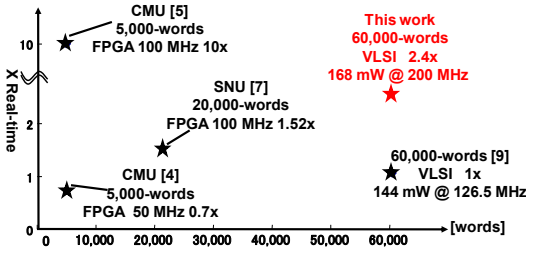

Fig. 1 Vocabulary vs. speed.

II. ALGORITHM OVERVIEW

Figure 2 presents the speech recognition flow with the HMM algorithm [10]. **Step 1:** Feature vector extraction: The speech input is sampled using a A/D converter and the mel frequency cepstral coefficients (MFCC) feature vectors

978-1-4673-1556-2/12/$31.00@2012 IEEE

are extracted from 30 ms length of speech every 10 ms. **Step 2:** GMM computation: State output probabilities are calculated for all possible sounds that could have been pronounced. **Step 3:** Viterbi Search: $\delta_t(j)$ is calculated for all active state nodes using GMM probabilities, transition probabilities and language models. **Step 4:** Beam pruning: according to the beam width, active state nodes having a higher score (accumulated probability) are selected; the others are dumped. **Step 5:** Output sentence: The word list with the maximum score is output as a speech recognition result after final-frame calculation and determination of the transition sequence.

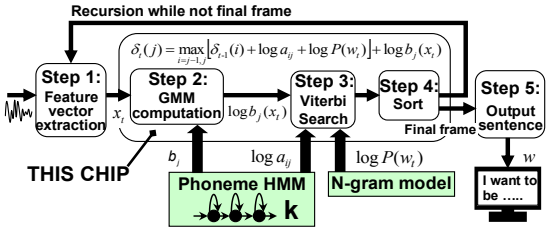

Fig. 2 Speech recognition flow with HMM algorithm.

We calculate the log probability density function (PDF) by its max approximation instead of the previous log-table based one [9], which saves about 1 Mbit of on-chip memory.

$$\log b_s(X_t)=\max_m\{C_m-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_d-\mu_{md})^2}{\sigma_{md}^2}\} \quad (1)$$

Therein, $\log b_s(X_t)$ represents the state output probability of a HMM state s for feature vector X_t at time t; x_d stands for the vector component of the feature vector X_t, D is the feature dimension, and C_m, μ_{md}, σ_{md} respectively denote the constant, the mean, and the standard deviation of Gaussian mixture model.

The Viterbi search is divisible into two parts: internal word transition and cross-word transition. Dynamic programming (DP) recursion for the internal word transition is shown in Eq. (2).

$$\delta_t(s_j;w)=\max_{i=j-1,\,j}[\delta_{t-1}(s_i;w)+\log a_{ij}]+\log b_j(x_t) \quad (2)$$

In that equation, a_{ij} is the transition probability from state s_i to s_j, and $\delta_t(s_j;w)$ stands for the largest accumulated probability of the state sequence reaching state s_j of word w at time t.

In cross-word transition, the bi-gram model is used where the transition probability of a word depends on the immediately preceding word. DP recursion for this part is shown in Eq. (3).

$$\delta_t(s_0;w)=\max_v\{\delta_{t-1}(s_f;v)+\log[p(w|v)]\} \quad (3)$$

Therein, $p(w|v)$ stands for the bi-gram probability from word v to word w, s_0 and s_f respectively denote the start state of word w and the last state of word v.

III. ARCHITECTURE

3.1 Speech recognition system

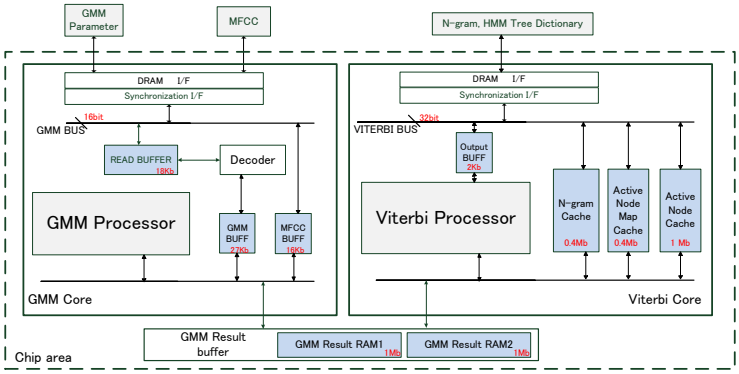

Fig. 3 Proposed speech recognition architecture.

The overall chip architecture, depicted in Fig. 3, comprises the GMM core, Viterbi core, double buffer for the GMM result, and the memory interface. We use a PowerMedusa [11] custom test board to construct a speech recognition system with a test chip. The MFCC feature vectors are extracted using a PC. The input speech data can either be recorded as an audio stream or with real-time speaking. The database is set up at the beginning. The test chip accesses the DRAM through an on-board FPGA. The data-path of the DRAM is 64 bit, but only 48pin is available for the test chip according to the pin limitation. The data-path for GMM computation (16pin) and Viterbi search (32pin) is separated to support pipeline operation.

3.2 GMM Architecture with Compression–Decoding

The external memory bandwidth for GMM computation can be reduced by sharing the parameters for several frames. We can use this scheme to reduce the memory access as much as needed, theoretically. A variable 50-frame-look-ahead scheme was used in [9], which reduced the external memory bandwidth to 13.3 MB/S for GMM computation. However, it becomes cost-ineffective when the look-ahead frames are numerous because the reduction efficiency becomes worse while the on-chip memory for saving the GMM result is proportionally increased. For the proposed method, we adjust the number of look-ahead frames to optimize the area.

The maximal IO frequency of the test chip is 50 MHz and the data pins for GMM computation are 16pin. Therefore, the IO transfer capability is 100 MB/S. The GMM parameters include 1987 states each with 16 mixtures. There are 52 parameters for one mixture. The bit-width of one parameter is 32 bit. Therefore 2 IO cycles are necessary to load one GMM parameter. There are 100 frames in 1S speech. Assuming n frames look-ahead to support 2.4×-real-

time processing (decided by Viterbi), n should be 32 according to the following limitation:

$$4B \times 52 \times 16 \times 1987 \times 100/n \times 2 \times 2.4\ MB/S < 100\ MB/S,$$

which requires roughly 3.2 Mbit of result RAM.

We proposed a compression–decoding scheme for additional optimization. GMM parameters saved in DRAM are compressed state-by-state to a smaller size using a lossless data compression algorithm. When the chip starts processing, the compressed data are transferred to a buffer inside the chip and then decoded. The parameters are rebuilt completely and therefore have no degradation to recognition accuracy. Then the compression ratio of the required external memory bandwidth is reduced.

The architecture and pipeline operation are portrayed respectively in Fig. 4 and Fig. 5. The decoding time for one-compressed-state parameters must be shorter than the loading time for a complete state and the extra computation workload and logic elements for decoder should be small. Therefore, we choose the run-length-encoding (Fig. 6) algorithm for our decoder. To adopt the GMM parameter, we merely compressed the top n bit of the integer part, which gives a compression ratio of 37.5% with only 0.01% extra computation workload and 0.005% area overhead of the GMM core. Consequently, the number of look-ahead frames is reduced further to 20. We implement a 20-frame parallel architecture for GMM computation.

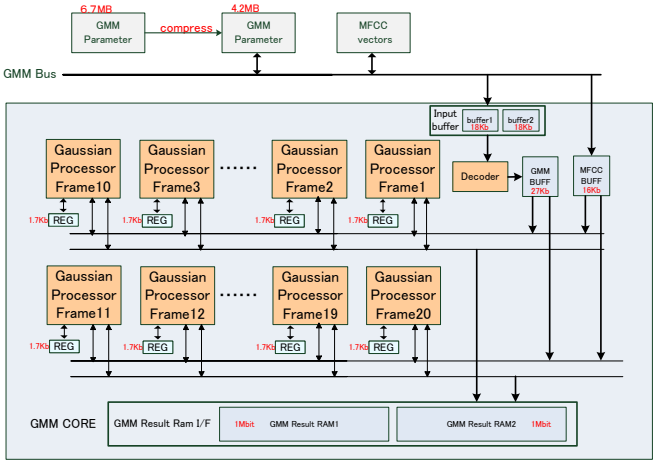

Fig. 4 GMM architecture with a compression–decoding scheme.

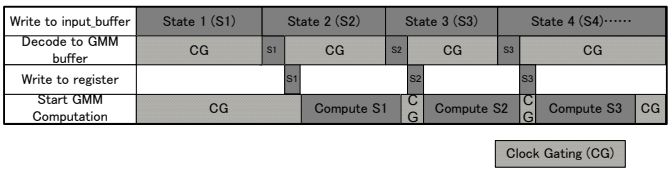

Fig. 5 Pipeline operation in a GMM core.

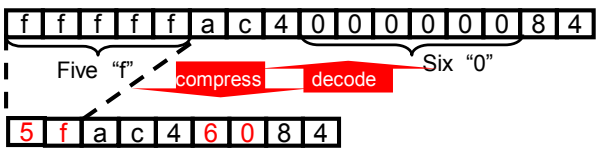

Fig. 6 Run Length Encoding (RLE) for GMM parameters.

3.3 Multi-path Viterbi transition unit

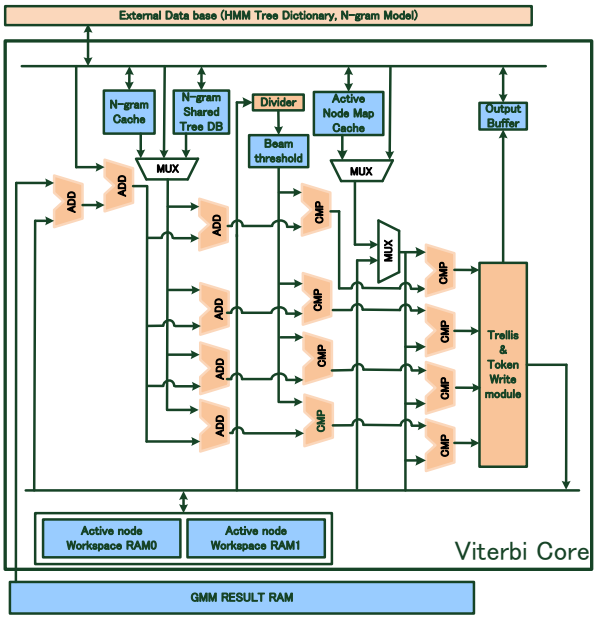

Fig. 7 Multi-path Viterbi architecture.

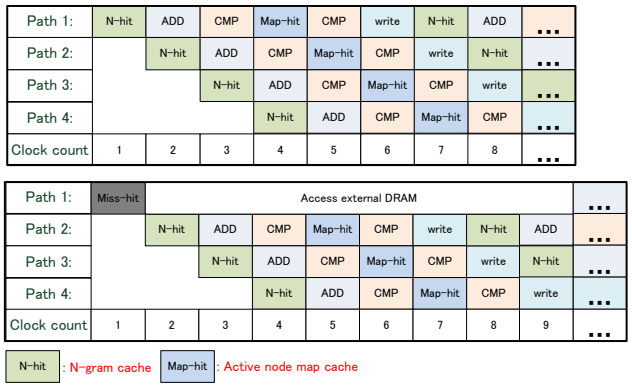

Fig. 8 Pipeline operation for cross-word transition when a mis-hit occurs.

The external memory bandwidth required in Viterbi processing has been reduced greatly by two-stage language model (LM) search, a specialized cache memory, and the elastic pipeline we proposed in [9]. All GMM probabilities and active node information can be read out from the internal RAM. However, when a mis-hit occurs at the n-gram cache and active-node map during cross-word transition, it will take two IO cycles, which is eight internal cycles in this chip, to access the external database. This latency strongly delays the Viterbi processing.

Consequently, we proposed four-path Viterbi-processing units to hide the latency as portrayed in Fig. 7. Figure 8 shows the pipeline operation. When a mis-hit occurs at path-1, the other paths will access the cache memory and continue processing. This scheme eliminates 34.2% of the cycle counts for the Viterbi transition.

IV. IMPLEMENTATION

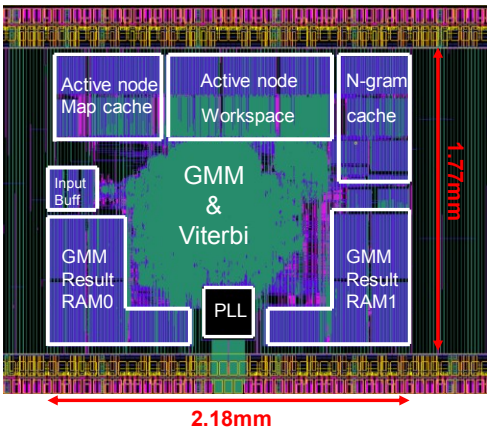

Fig. 9 Chip layout.

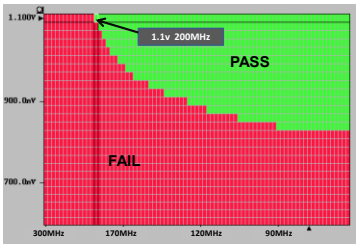

Fig. 10 Shmoo plot generated using a log tester.

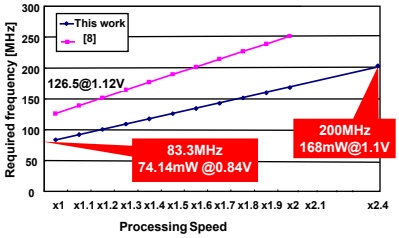

Fig. 11 Processing speed versus required frequency.

The chip, which was fabricated in 40 nm CMOS technology as shown in Fig. 9, occupies 1.77×2.18 mm^2 containing 2.52 M transistors for Logic and 4.29 Mbit on-chip SRAM.

We evaluated the test chip with a logic tester. The generated Shmoo plot is presented in Fig. 10. This chip can process real-time 60-kWord continuous speech recognition at 83.3 MHz and 0.84 V with power consumption of 74.14 mW, it can maximally process at 2.4× faster than real-time at 200 MHz with standard voltage of 1.1 V while dissipating 168 mW (Fig. 11). Table 1 presents a comparison between this work and some recently announced works.

Table 1 Comparison with recently reported works

		[8]	[7]	[9]	This work	
Vocabulary (k)		5	20	60	60	
LM	unigram	5000	19,983	60,001	60,001	
	bigram	835,000	1,440,272	4,000,273	4,000,273	
GMM Module	# of states	3,001	3,001	2,000	2,000	
	# of distributions	16	16	16	16	
	# of dimensions	39	39	25	25	
Viterbi beamwidth		NA	NA	3000	3000	
Process technology		0.18um	FPGA	40nm	40nm	
Real-time factor		2.4x	1.52x	1x	1x	2.4x
Frequency (MHz)		100	100	126.5	83.33	200
External memory BW (MB/S)		NA	800	70.86	82.6	198.24
Core area (mm^2)		15.47	NA	5.5	3.86	
Chip area		NA	NA	5mm×5mm	5mm×2.5mm	
Power consumption(mW)		NA	NA	144	74.14	168.12
Logic elements		NA	13,835 slices	1.9 MTr.	2.52 MTr.	
Internal memory (KB)		140.1	416	975	344	

V. SUMMARY

We have developed a low-power VLSI chip for 60 k-Word real-time continuous speech recognition. For high-speed processing, our implementation includes a compression–decoding scheme to reduce the external memory bandwidth and multi-path Viterbi transition units to hide the DRAM latency when a mis-hit occurs. The measured results show that our implementation achieves 34.2% required frequency reduction (83.3 MHz) and reduces 48.5% power consumption (74.14 mW) for 60 k-Word real-time continuous speech recognition. This chip can maximally process 2.4× faster than real-time at 200 MHz and 1.1 V with power consumption of 168 mW.

ACKNOWLEDGMENTS

The VLSI chip used in this study was fabricated in the chip fabrication program of VLSI Design and Education Center (VDEC), The University of Tokyo. This development was performed by the author for STARC as part of the Japanese Ministry of Economy, Trade and Industry sponsored "Silicon Implementation Support Program for Next Generation Semiconductor Circuit Architectures".

REFERENCES

[1] A. Lee, T. Kawahara and K. Shikano, "Julius – an open source real-time large vocabulary recognition engine," Proc. European Conf. on Speech Communication and Tech. (EUROSPEECH), pp. 1691-1694, Sep. 2001.

[2] K. Yu, and R. Rutenbar, "Profiling Large-Vocabulary Continuous Speech Recognition on Embedded Device: A Hardware Resource Sensitivity Analysis," Proc. ISCA Annual Conf. of Intl. Speech Communication Association (Interspeech), pp. 995-998, Sep. 2009.

[3] E. C. Lin, K. Yu., R. Rutenbar, and T. Chen, "In silico Vox: Towards Speech Recognition in Silicon" HOTCHIPS 18, August, 2006.

[4] E. C. Lin, K. Yu. R. Rutenbar, and T. Chen, " A 1000-Word Vocabulary, Speaker-Independent, Continuous Live-Mode Speech Recognizer Implemented in a Single FPGA", International Symposium on Field-Programmable Gate Arrays (FPGA), Feb. 2007.

[5] E. C. Lin, and R. A. Rutenbar, "A Multi-FPGA 10x-Real-Time High-Speed Search Engine for a 5000-Word Vocabulary Speech Recognizer," Proc. ACM/SIGDA Intl. Symposium on Field Programmable Gate Arrays (FPGA), pp.83-92, Feb. 2009.

[6] S. Yoshizawa, N. Wada, N. Hayasaka, and Y. Miyanaga, "Scalable architecture for word HMM-based speech recognition and implementation in complete system," Proc. IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 53, no. 1, pp. 70-77, Jan. 2006

[7] Y. Choi, K. You, J. Choi, and W. Sung, "A Real-Time FPGA-based 20,000-Word Speech Recognizer with optimized DRAM Access," IEEE Trans. Circuits Syst. I, Reg. Papers, issue 99, Feb. 2010.

[8] K. You, Y.Choi, J. Choi, and W. Sung, "Memory Access Optimized VLSI for 5000-Word Speech Recognition," JOURNAL OF SIGNAL PROCESSING SYSTEMS, vol.63, no. 1, pp. 95-105, Nov. 2009.

[9] G. He, T. Sugahara, T. Fujinaga, Y. Miyamoto, H. Noguchi, S. Izumi, H. Kawaguchi, and M. Yoshimoto, "A 40 nm 144 mW VLSI processor for Realtime 60 kWord Continuous Speech Recognition," Proc. IEEE Custom Integrated Circuits Conference (CICC), pp.1-4 Sep. 2011.

[10] X. Huang, A. Acero, and H. W. Hon, Spoken Language Processing-A Guide to Theory, Algorithm, and System Development. Englewood Cliffs, NJ: Prentice Hall, 2001.

[11] http://www.mms.co.jp/powermedusa/concept/index.htl