A 168-mW 2.4×-Real-Time 60-k Word Continuous Speech Recognition Processor VLSI

He, Guangji ; Sugahara, Takanobu ; Miyamoto, Yuki ; Izumi, Shintaro ; Kawaguchi, Hiroshi ; Yoshimoto, Masahiko

(Citation)
IEICE Transactions on Electronics,96(4):444-453

(Issue Date)
2013-04-01

(Resource Type)
journal article

(Version)
Version of Record

(URL)
https://hdl.handle.net/20.500.14094/90002979

IEICE TRANS. ELECTRON., VOL.E96–C, NO.4 APRIL 2013

PAPER Special Section on Solid-State Circuit Design—Architecture, Circuit, Device and Design Methodology

A 168-mW 2.4×-Real-Time 60-kWord Continuous Speech Recognition Processor VLSI*

Guangji HE[†a)], ***Student Member***, **Takanobu SUGAHARA**[†], **Yuki MIYAMOTO**[†], ***Nonmembers***, **Shintaro IZUMI**[†], **Hiroshi KAWAGUCHI**[†], ***and*** **Masahiko YOSHIMOTO**[†], ***Members***

SUMMARY 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). It features 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), 48.5% power consumption reduction (74.14 mW) for 60 k-Word real-time continuous speech recognition compared to the previous work while 30% of the area is saved with recognition accuracy of 90.9%. This chip can maximally process 2.4× faster than real-time at 200 MHz and 1.1 V with power consumption of 168 mW. By increasing the beam width, better recognition accuracy (91.45%) can be achieved. In that case, the power consumption for real-time processing is increased to 97.4 mW and the max-performance is decreased to 2.08× because of the increased computation workload.

key words: *40 nm VLSI, hidden Markov model (HMM), large vocabulary continuous recognition (LVCSR)*

1. 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 Image and Vidio processing system, the DRAM only acts as a buffer between camera and chip, the pixels are read from DRAM orderly and saved to the on-chip memory. However, in large vocabulary continuous speech recognition (LVCSR) system, the DRAM functions as a data-base which saves the dictionary parameters and language models. These data will be accessed randomly during the processing. According to the characteristics of DRAM, there are several cycles of latency caused by pre-charge every time before we read from the DRAM, therefore if the required data are not saved sequentially, the access-efficiency is bad. As a result, speech recognition needs much higher IO frequency than video processing to get the same amount of data from DRAM. Especially, with the number of vocabulary increase, the external memory bandwidth become enormous which causes two problems, firstly, real-time processing is impossible to be achieved because of the I/O frequency limitation. Secondly, large amount of power is consumed by I/O because of the high supply voltage (3.3 V). Consequently, reducing external memory bandwidth is one of the most important things to implement a low-power speech recognition system.

In the prior work [9], [10], we focus on reducing the external memory bandwidth, we presented a VLSI processor (HMM_1) for real-time continuous 60-kWord continuous speech recognition. It employs some algorithm optimization such as two-stage language model (LM) search to reduce the cross-word transitions and beam pruning using a dynamic threshold. A variable-frame look-ahead scheme, highly GMM parallel architecture and a specialized Viterbi cache architecture using locality of speech recognition. 40 nm process is used because large mount of cache memory is needed to be implemented on a limited chip area. As a result, 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

Manuscript received August 7, 2012.
Manuscript revised November 5, 2012.

[†]The authors are with Kobe University, Kobe-shi, 657-8501 Japan.

*This paper is the extended version of the original paper presented at the 2012 Custom integrated Circuits Conference [11].

a) E-mail: achilles@cs28.cs.kobe-u.ac.jp

DOI: 10.1587/transele.E96.C.444

internal RAM size reaches 7.8 Mbit, occupying a large area.

Although the external memory bandwidth was reduced adequately in the previous work, it needed a large amount of on-chip memory and caused long latency. Therefore in this chip (HMM_2), we try to find another way to reduce the external memory bandwidth. As described herein, we optimize the number of look-ahead frames cooperating with a compression decoding scheme to reduce the external memory bandwidth for Gaussian Mixture Model (GMM) computation, which cuts down the required GMM result RAM. Furthermore, we proposed multi-path Viterbi transition units to hide the DRAM latency when a mis-hit occurs to accelerate the processing. We use the max-approximation GMM calculation to save the add-log table 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, 48.5% power consumption reduction and 30% area reduction compared to HMM_1 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) with power consumption of 168 mW. A comparison of the vocabulary size and processing speed among recently announced hardware-based speech recognizers is shown in Fig. 1.

The rest of this paper is organized as follows. The speech recognition algorithm is explained in Sect. 2. Section 3 describes the proposed architecture. Section 4 presents the VLSI implementation and its measurement results. Finally, Sect. 5 offer s concluding marks.

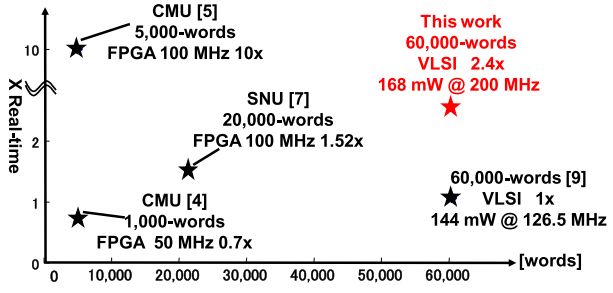


Fig. 1 Vocabulary versus speed.

2. Algorithm Overview

Figure 2 presents the speech recognition flow with the HMM algorithm [12]. **Step 1:** Feature vector extraction — The input speech signal is sampled using a A/D converter and the mel frequency cepstral coefficients (MFCC) feature vectors 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:** Sort — 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.

2.1 GMM Computation

We calculate the log probability density function (PDF) by its max approximation instead of the previous log-table based one [10], which saves about 1 Mbit of on-chip memory while cause 049% degradation in regocnition accuracy.

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

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.

2.2 Time-Synchronous Viterbi Beam Search

The following formulas show a time-synchronous Viterbi beam search algorithm [13], which 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).

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

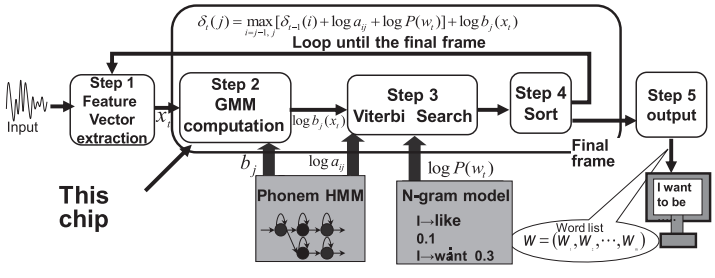


Fig. 2 Speech recognition flow with HMM algorithm.

Where 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. Once an internal word transition reach a word-end state, cross-word transition will be treated, a bi-gram (2-gram) model is used in this chip, where the transition probability of a word depends on the immediately preceding word. DP recursion for this part is shown in (4).

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

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.

3. Architecture

3.1 Speech Recognition System

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 [14] custom test board to construct a speech recognition system with a test chip. The MFCC feature vectors are extracted using a PC. The reason why we separate the feature extraction part from the recognizer is as follows:
1) Firstly, the use of fixed-point computation in the feature extraction part will cause big degradation [15], [16] in recognition accuracy (3–5%);
2) Secondly, the computation workload for feature extraction is small and can be easily handled by PC or an embedded soft-core [4], [7];
3) Lastly, it is flexible for adopting other capabilities like noise reduction or speaker adaptation [7].

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 48 pin is available for the test chip according to the pin limitation. The data-path for GMM computation (16 pin) and Viterbi search (32 pin) 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 16 pin. 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:

$$4B\times52\times16\times1987\times100/n\times2\times2.4\,\text{MB/S} < 100\,\text{MB/S},$$

which requires roughly 3.2 Mbit of result RAM and causes latency of 0.32 S. For further optimization, We proposed a compression-decoding scheme. 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

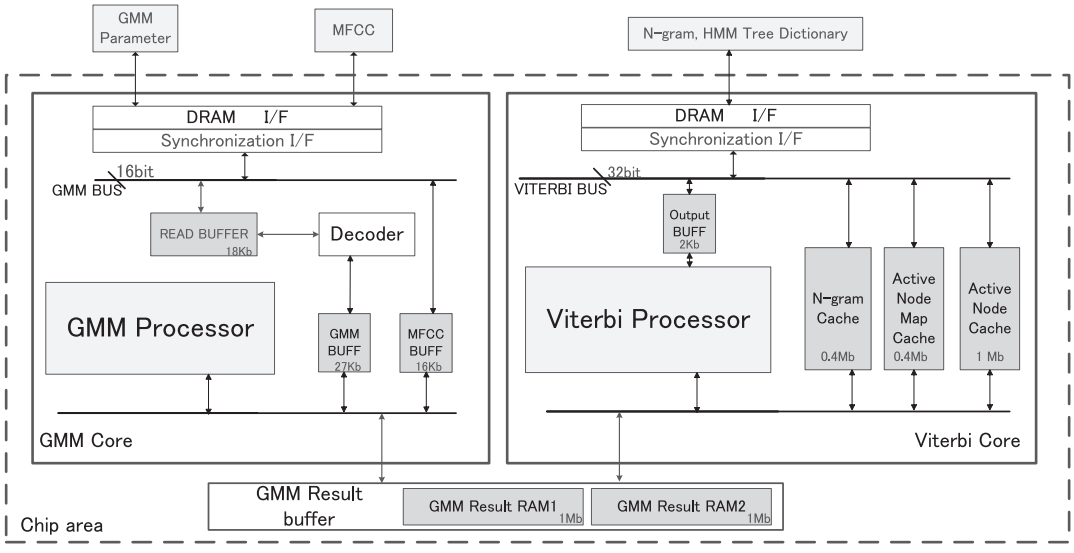


Fig. 3 Proposed speech recognition architecture.

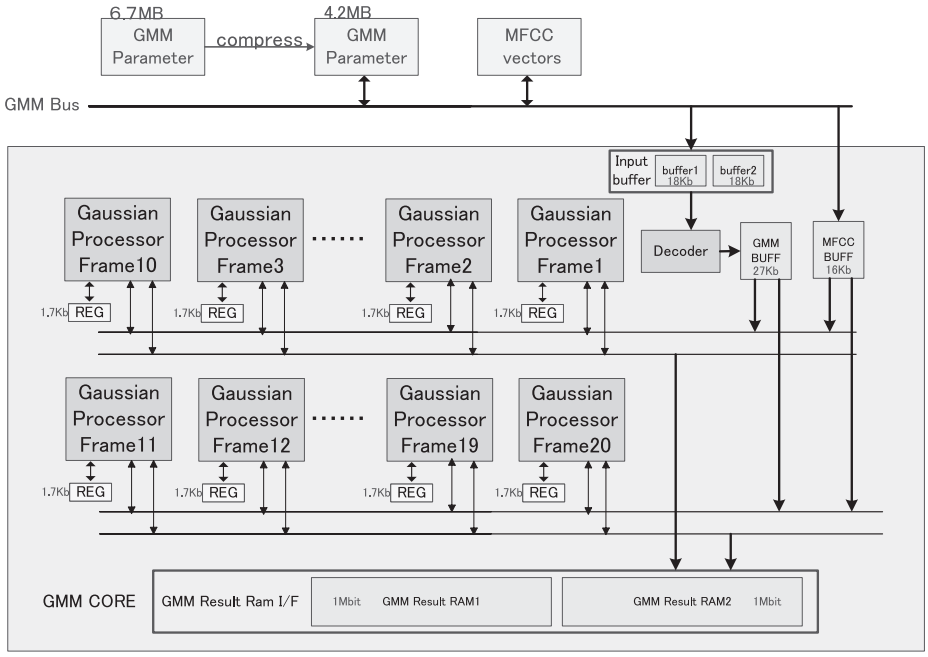


Fig. 4 GMM architecture with compression-decoding scheme.

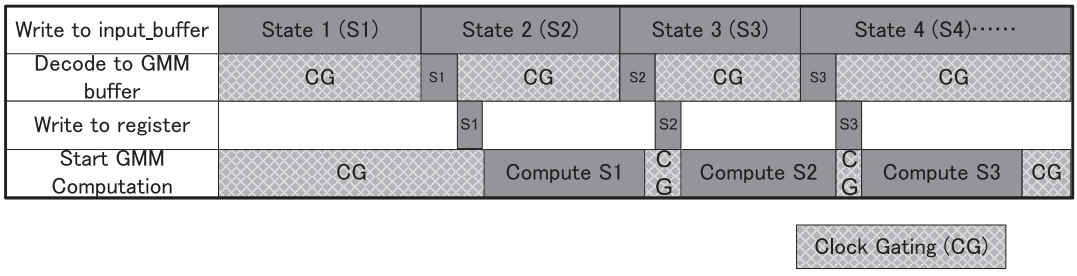


Fig. 5 Pipeline operation in GMM core.

inside the chip and then decoded. The parameters are rebuilt completely and therefore have no degradation to recognition accuracy. Then the compression-decoding scheme can reduce the required external memory bandwidth by the compression ratio.

The GMM architecture and pipeline operation are portrayed respectively in Fig. 4 and Fig. 5. The GMM core comprise a MFCC buffer for feature vectors, two input buffers for loading the compressed data, a GMM buffer for the decoded parameters, one decoder and 20 GMM computation processors. As shown in Fig. 5, the compressed data of state n is loaded to the input buffer while the decoding and the computation for state n-1 is treated.

Clock gating is implemented in the decoder and the GMM processors. After decoding state n, the decoder will be clock gated until the loading for state n+1 is accomplished. GMM processors are also clock gated in the same way. The power consumption reduction by clock gating is measured by simulation with 40 nm library and power compiler. Because of the high leakage current in 40 nm process, only 1.25% power reduction is confirmed.

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 modified the RLE operation, instead of compressing all bit, we merely compressed the top n bit for one parameter. There are three cases for decoding as shown in Fig. 7. In Case one or case two 2 steps are needed to generate one parameter while in case three only one step is needed. As there are 832 parameters for one state, the total number of operation is less than 832*2 = 1664. The average compression ratio is 37.5% with only 0.01% extra computation workload and 0.005% area overhead of the GMM core. Consequently, the number of look-

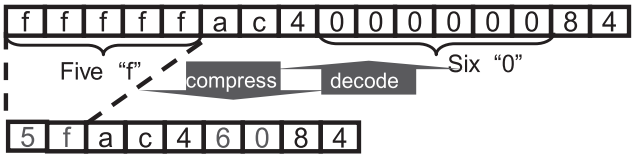


Fig. 6 Run Length Encoding (RLE) for GMM parameters.

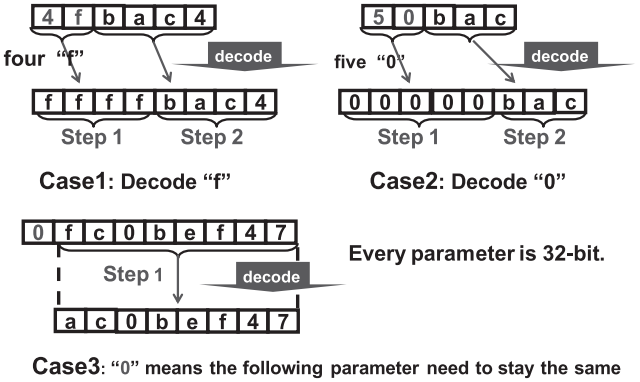


Fig. 7 Three cases for decoding operation.

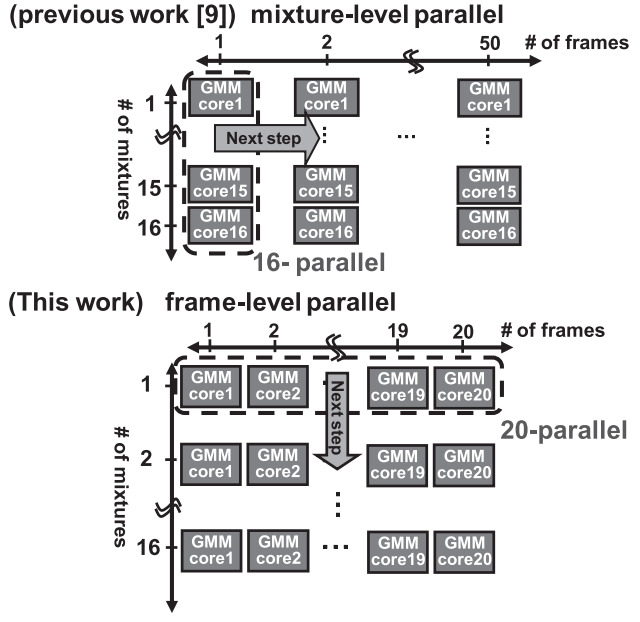


Fig. 8 20-frame parallel processing vs 16-mixture parallel processing.

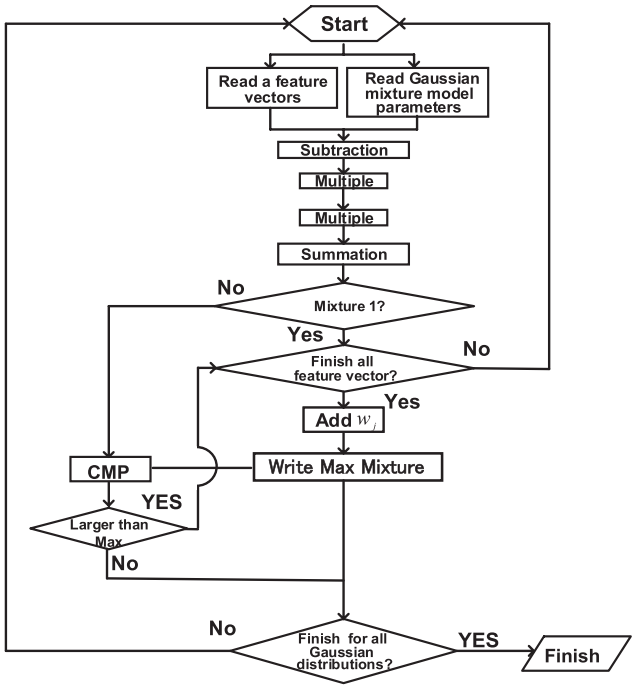


Fig. 9 GMM Computation flow.

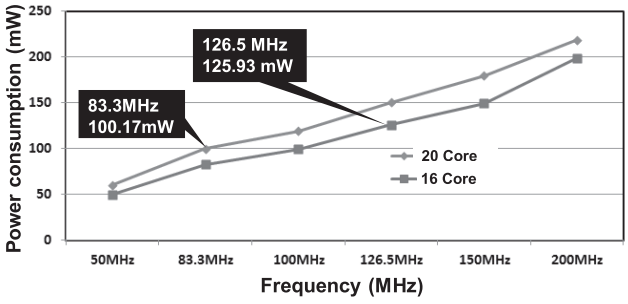


Fig. 10 Power comsumption for real-time GMM computation of 16-core and 20-core measured by simulation.

ahead frames is reduced further to 20.

According to the required processing speed of this chip, it is needed to increase the degree of parallelism. However, mixture-level parallel implementation is not flexible because only limited degrees can be utilized (16, 32 etc.), which will cause unnecessary increment of logic elements. Therefore we need to implement the parallel architecture at the frame level and it is just suitable for the 20-frame look-ahead scheme. As described herein, we implement a 20-frame parallel architecture for GMM computation to reduce the cycle count for the computation part. Figure 8 presents both the 16-parallel processing at mixture level of the previous chip [9] and the 20-parallel processing at frame level of this chip. The GMM computation flow using the max-approximation algorithm is shown in Fig. 9. As the value of one mixture is getting smaller with the calculation going on, the computation for one mixture can be stopped when its value is less than the max result of the previous mixtures. Around 40% cycle count reduction is achieved by the above changes. The power consumption variety of 16-core and 20-core measured by simulation are shown in Fig. 10. Power consumption for real-time processing is reduced by 20%.

3.2 Multi-Path Viterbi Transition Unit

The external memory bandwidth required in Viterbi processing has been reduced greatly by two-stage language model (LM) search, the 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 or active-node map during cross-word transition, it will take two IO cycles, which is eight internal cycles to access the external database. In the previous chip [9], the processing is stopped to wait until the required data is got from the DRAM, this latency strongly delays the Viterbi processing. Although the internal data-path to the caches are available

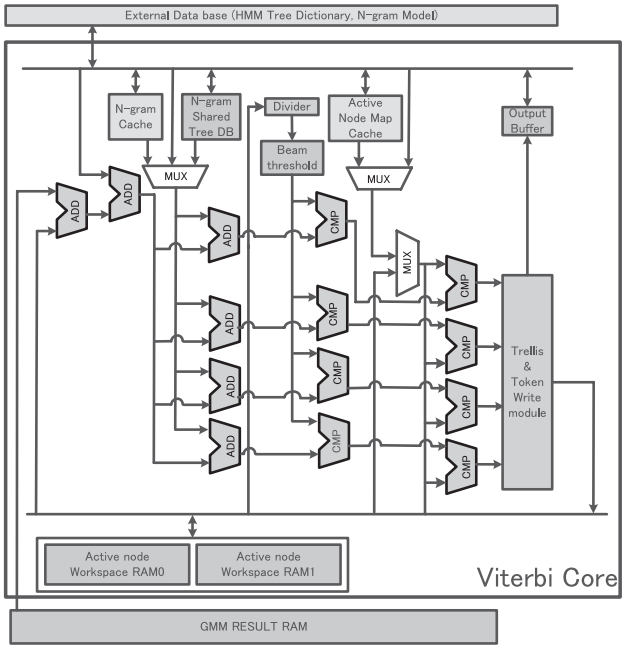


Fig. 11 Multi-path Viterbi architecture.

Path 1:	N-hit	ADD	CMP	Map-hit	CMP	write	N-hit	ADD	...
Path 2:		N-hit	ADD	CMP	Map-hit	CMP	write	N-hit	...
Path 3:			N-hit	ADD	CMP	Map-hit	CMP	write	...
Path 4:				N-hit	ADD	CMP	Map-hit	CMP	...
Clock count	1	2	3	4	5	6	7	8	...

Path 1:	Miss-hit	Access external DRAM								...
Path 2:		N-hit	ADD	CMP	Map-hit	CMP	write	N-hit	ADD	...
Path 3:			N-hit	ADD	CMP	Map-hit	CMP	write	N-hit	...
Path 4:				N-hit	ADD	CMP	Map-hit	CMP	write	...
Clock count	1	2	3	4	5	6	7	8	9	...

N-hit : N-gram cache Map-hit : Active node map cache

Fig. 12 Pipeline operation for cross-word transition.

at this time, it is impossible to process the next transition.

Consequently in this chip, we proposed a four-path Viterbi-processing units to hide the latency as portrayed in Fig. 11. Figure 12 shows the pipeline operation. When a mis-hit occurs at one of the transition paths, the other units will access the cache memory and continue processing other transitions. This scheme eliminates 34.2% of the cycle counts for the Viterbi transition. Because the computation time for Viterbi is the neck of the GMM-Viterbi pipeline processing, the total computation time is reduced by the same rate.

4. Implementation

The chip, which was fabricated in 40 nm CMOS technology as shown in Fig. 13, occupies 1.77× 2.18 mm^2 containing 2.52 M transistors for Logic and 4.29 Mbit on-chip SRAM. A summary of the chip statistics is shown in Table 1. The breakdown of the internal memory is shown in Table 2.

We implement a software prototype profiling with Microsoft Visual C++ and a referential hardware using hardware descrption language (HDL) to check the required memory bandwidth and frequency for real-time operation. The reduction of both frequency and external memory band-

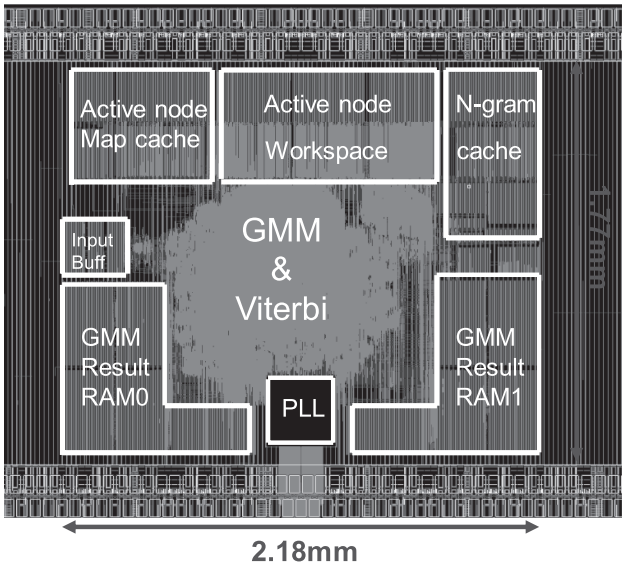


Fig. 13 Chip layout.

Table 1 Summary of chip implementation.

Process Technology		40–nm CMOS
Core area		2.18 mm × 1.77 mm
Chip area		5 mm × 2.5 mm
Transister Count (Logic)	GMM	1.58 M
	Viterbi	0.43 M
	Other	0.51 M
	Total	2.52 M
On–Chip Memory (SRAM)		4.29 Mbit
Supply voltage		1.1V
I/O voltage		3.3V
Evaluation environment		SOC Tester System
Operating Frequency		83.3 MHz for 60 kWord real–time processing
Measured Power (without IO)		74.14 mW
Leakage current		1.35 mA

Table 2 Breakdown of Internal memory.

Description	# of SRAM	Size of SRAM (Kb)
gmm_result_ram	10	2000
gmm_buffer	1	26
Input_buffer	2	36
MFCC_buffer	1	16
active_node_work_space	6	1248
shared Tree DB	1	14
bi–gram	4	468
active_node_map	6	476
initial parameter	1	4
output buffer	1	2
Total	50	4290 (536kB)

width are presented in Fig. 14 and Fig. 15. The required frequency for real-time processing is reduced by 34.2% compared to HMM_1. This rate is just the same as Viterbi part because the computation time for Viterbi transition is the neck of the GMM-Viterbi pipeline operation. Although the external memory bandwidth was reduced greatly in HMM_1, it needed a large amount of SRAM and caused long latency. Therefore in HMM_2, we optimize the number of look-ahead frames cooperating with a compression

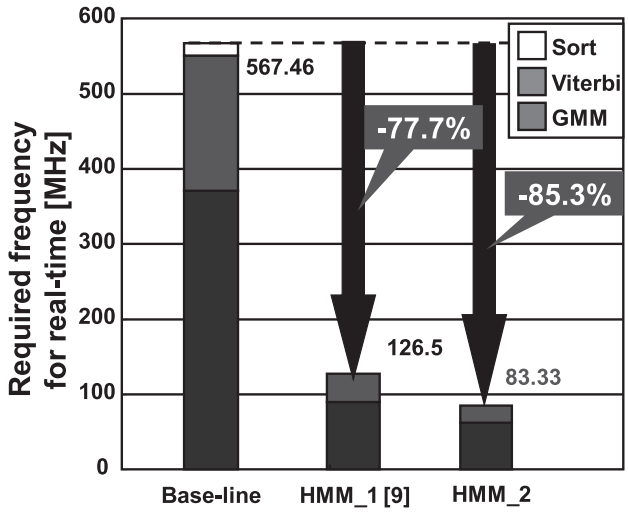


Fig. 14 Required frequency reduction.

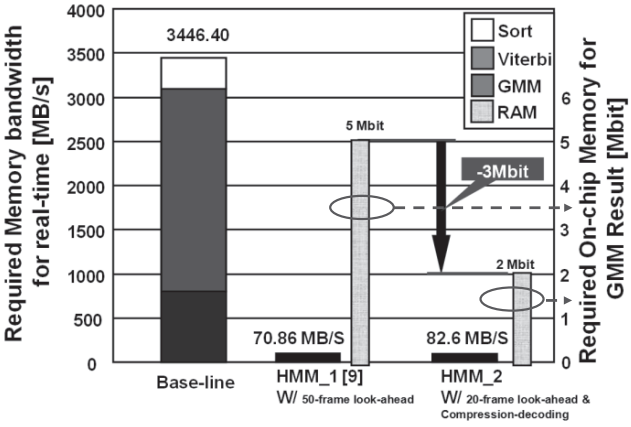


Fig. 15 Required memory bandwidth reduction & GMM result RAM reduction.

decoding scheme to reduce the external memory bandwidth for GMM computation, which cuts down 3 Mbit of GMM result RAM as shown in Fig. 15.

We evaluated the test chip with a logic tester. The generated Shmoo plot is presented in Fig. 16. Figure 17 shows the power consumption at different frequency. 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 which is only half of the power consumed by HMM_1. 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. 18). Table 3 presents a comparison between this work and some recently announced works in terms of the vocabulary size, GMM model, language model, beam-width, recognition accuracy, real-time factor, operation frequency, internal memory, external memory bandwidth, area and the logic element. Power consumption of both core and IO are also included.

The beam-width is the number of active node we treat every frame, as described in [9], [10], larger beam-width can

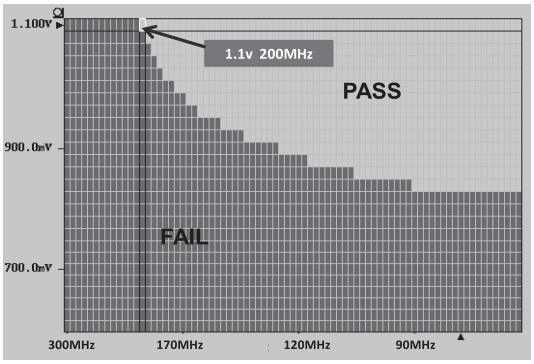


Fig. 16 Shmoo plot generated using a logic tester.

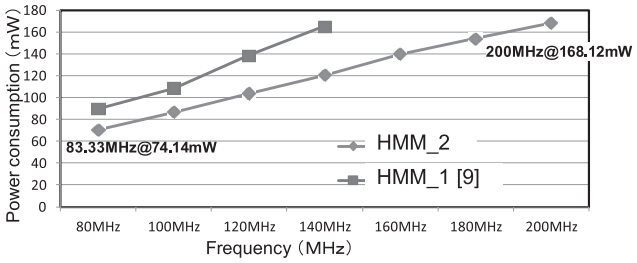


Fig. 17 Power consumption versus frequency.

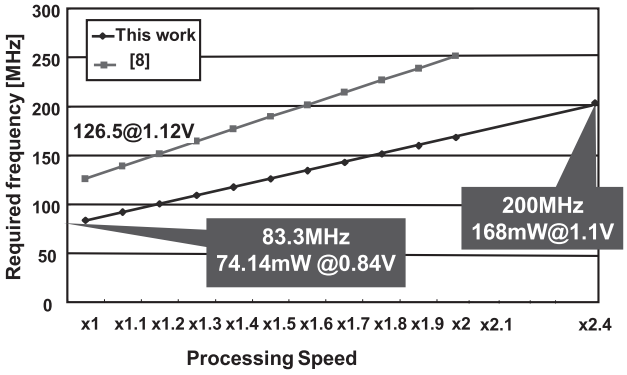


Fig. 18 Processing speed versus required frequency.

Table 3 Comparison with recently reported works.

		[8]	[7]	[9] (HMM_1)	This work (HMM_2)			
Vocabulary (k)		5	20	60	60			
Technology		VLSI (0.18 um)	FPGA	VLSI (40 nm)	VLSI (40 nm)			
GMM Mod	# of states	3,001	3,001	2,000	2,000			
	# of distributions	16	16	16	16			
	# of dimensions	39	39	25	25			
LM	# of unigram	5000	19,983	60,001	60,001			
	# of bigram	835,000	1,440,272	4,000,273	4,000,273			
Viterbi beam width		NA	500	3000	3000		4000	
Accuracy (%)		89	88	91.39	90.9		91.45	
Real-time factor		2.38x	1.51x	1x	1x	2.4x	1x	2.08x
Internal Frequency (MHz)		100	100	126.5	83	200	96	200
IO Frequency (MHz)		100	100	31.625	21	50	24	50
External memory BW (MB/		NA	800	70.86	83	198	96	199
Power consumpt	Core (mW)	NA	NA	144	74	168	97	172
	IO (mW)	NA	NA	58.2	39	94	45	94
Core area (mm^2)		15.47	NA	5.5	3.86			
Logic elements		NA	13,835 slices	1.9 MTr	2.52 MTr.			
Internal memory (KB)		140.1	416	975	536			

afford higher recognition accuracy while increase the computation workload. Consequently, although the use of GMM approximation computation cause 0.49% accuracy degradation comparing to HMM_1, because of the performance-improvement, this chip can recognize speech with a larger beam-width of 4000 which provide a better recognition accuracy of 91.45% as shown in Table 3. In that case, the power consumption for real-time processing is increased to 97.4 mW and the max-performance is decreased from 2.4× to 2.08×.

5. Conclusion

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), 48.5% power consumption reduction (74.14 mW) and 30% area reduction compared to the previous work for 60 k-Word real-time continuous speech recognition with recognition accuracy of 90.9%. This chip can maximally process 2.4× faster than real-time at 200 MHz and 1.1 V with power consumption of 168 mW. By increasing the beam width, better accuracy (91.45%) can be achieved. In that case, the power consumption for real-time processing is increased to 97.4 mW and the max-performance is decreased to 2.08×. because of the increased computation workload.

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. Eur. Conf. Speech Communication Tech. (EUROSPEECH), Aalborg, pp.1691–1694, Denmark, Sept. 2001.

[2] K. Yu and R. Rutenbar, “Profiling large-vocabulary continuous speech recognition on embedded device: A hardware resource sensitivity analysis,” Proc. 10th Conf. Interspeech, pp.995–998, Brighton, UK, Sept. 2009.

[3] E.C. Lin, K. Yu., R. Rutenbar, and T. Chen, “In silico vox: Towards speech recognition in silicon,” “HOTCHIPS”, Aug. 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,” Proc. 15th ACM/SIGDA Int. Symp. FPGA, Monterey, CA, 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. 17th ACM/SIGDA Intl. Symp FPGA, pp.83–92, Montery, CA, Feb. 2009.

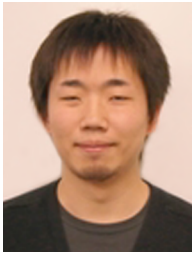
[6] S. Yoshizawa, N. Wada, N. Hayasaka, and Y. Miyanaga, “Scalable

architecture for word HMM-based speech recognition and implementation in complete system," 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, vol.57, no.8, pp.2119–2131, Aug. 2010.
[8] K. You, Y. Choi, J. Choi, and W. Sung, "Memory access optimized VLSI for 5000-word speech recognition," J. Signal Process. Syst., vol.63, no.1, pp.95–105, April 2011.
[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 Sept. 2011.
[10] 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," IEEE Trans. Circuits Syst. I, Reg. Papers, vol.59, no.8, pp.1656–1666, Aug. 2012.
[11] G. He, T. Sugahara,Y. Miyamoto, S. Izumi, H. Kawaguchi, and M. Yoshimoto, "A 40-nm 168-mW 2.4×-real-time VLSI processor for 60-kWord continuous speech recognition," Proc. IEEE Custom Integrated Circuits Conference (CICC), Sept. 2012.
[12] X. Huang, A. Acero, and H.W. Hon, Spoken Language Processing-A Guide to Theory, Algorithm, and System Development, Prentice Hall, Englewood Cliffs, NJ, 2001.
[13] H. Ney and S. Ortmanns, "Dynamic programming search for continuous speech recognition," IEEE Signal Process. Mag., vol.16, no.5, pp.64–83, Sept. 1999.
[14] http://www.mms.co.jp/powermedusa/concept/index.htl
[15] T.W. Kohler, C. Fugen, S. Stuker, and A. Waibel, "Rapid porting of ASR-systems to mobile devices," Proc. Interspeech, pp.233–236, Sept. 2005.
[16] D. Huggins-Daines, K. Mohit, and C. Arthur, "Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices," Proc. ICASSP, pp.185–188, May 2006.

Guangji He received a B.Eng. degree in Electronic Engineering from the Beijing Institute of Technology, Beijing, China, in 2007, and an M.Eng. degree in Computer Science and Systems Engineering in 2010 from Kobe University, Hyogo, Japan, where he is currently pursuing the Ph.D. degree. His current research interests include speech recognition, speech processing and low-power multimedia VLSI design. He is a student member of IEEE.

Takanobu Sugahara received a B.Eng. degree in Computer and Systems Engineering in 2010 from Kobe University, Hyogo, Japan, where he is currently working in the M.E. course. His current research is speech recognition algorithms and their hardware implementation for wearable devices.

Yuki Miyamoto earned a B.Eng. degree in Computer and Systems Engineering in 2011 from Kobe University, Hyogo, Japan. Currently, he is a master course student at Kobe University. Since 2009, he has been involved in the research and development of low-power speech recognition designs.



Shintaro Izumi received his B.Eng. and M.Eng. degrees in Computer Science and Systems Engineering from Kobe University, Hyogo, Japan, in 2007 and 2008, respectively. He received his Ph.D. degree in Engineering from Kobe University in 2011. He was a JSPS research fellow at Kobe University from 2009 to 2011. Since 2011, he has been an Assistant Professor in the Organization of Advanced Science and Technology at Kobe University. His current research interests include biomedical signal processing, communication protocols, low-power VLSI design, and sensor networks. He is a member of the IEEE and IPSJ.

Hiroshi Kawaguchi received B.Eng. and M.Eng. degrees in electronic engineering from Chiba University, Chiba, Japan, in 1991 and 1993, respectively, and earned a Ph.D. degree in electronic engineering from The University of Tokyo, Tokyo, Japan, in 2006. He joined Konami Corporation, Kobe, Japan, in 1993, where he developed arcade entertainment systems. He moved to The Institute of Industrial Science, The University of Tokyo, as a Technical Associate in 1996, and was appointed as a Research Associate in 2003. In 2005, he moved to Kobe University, Kobe, Japan. Since 2007, he has been an Associate Professor with The Department of Information Science at that university. He is also a Collaborative Researcher with The Institute of Industrial Science, The University of Tokyo. His current research interests include low-voltage SRAM, RF circuits, and ubiquitous sensor networks. Dr. Kawaguchi was a recipient of the IEEE ISSCC 2004 Takuo Sugano Outstanding Paper Award and the IEEE Kansai Section 2006 Gold Award. He has served as a Design and Implementation of Signal Processing Systems (DISPS) Technical Committee Member for IEEE Signal Processing Society, as a Program Committee Member for IEEE Custom Integrated Circuits Conference (CICC) and IEEE Symposium on Low-Power and High-Speed Chips (COOL Chips), and as an Associate Editor of IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences and IPSJ Transactions on System LSI Design Methodology (TSLDM). He is a member of the IEEE, ACM, and IPSJ.

Masahiko Yoshimoto earned a B.S. degree in electronic engineering from Nagoya Institute of Technology, Nagoya, Japan, in 1975, and an M.S. degree in electronic engineering from Nagoya University, Nagoya, Japan, in 1977. He earned a Ph.D. degree in Electrical Engineering from Nagoya University, Nagoya, Japan in 1998. He joined the LSI Laboratory, Mitsubishi Electric Corp., Itami, Japan, in April 1977. During 1978–1983 he was engaged in the design of NMOS and CMOS static RAM including a 64 K full CMOS RAM with the world's first divided-wordline structure. From 1984, he was involved in research and development of multimedia ULSI systems for digital broadcasting and digital communication systems based on MPEG2 and MPEG4 Codec LSI core technology. Since 2000, he has been a Professor of the Dept. of Electrical and Electronic Systems Engineering at Kanazawa University, Japan. Since 2004, he has been a Professor of the Dept. of Computer and Systems Engineering at Kobe University, Japan. His current activity is focused on the research and development of an ultra low power multimedia and ubiquitous media VLSI systems and a dependable SRAM circuit. He holds on 70 registered patents. He has served on the program committee of the IEEE International Solid State Circuit Conference from 1991 to 1993. Also he served as Guest Editor for special issues on Low-Power System LSI ,IP and Related Technologies of IEICE Transactions in 2004. He was a chair of IEEE SSCS(Solid State Circuits Society)Kansai Chapter from 2009 to 2010. He is also a chair of The IEICE Electronics Society Technical Committee on Integrated Circuits and Devices from 2011–2012. He received the R&D100 awards from the R&D magazine for the development of the DISP and the development of the realtime MPEG2 video encoder chipset in 1990 and 1996, respectively. He also received 21st TELECOM System Technology Award in 2006.