A Versatile Data String-Search VLSI

MASAKI HIRATA, HACHIRO YAMADA, HAJIME NAGAI, AND KOUSUKE TAKAHASHI

Abstract — **A versatile data string-search VLSI has been fabricated using 1.6-μm CMOS technology. The VLSI consists of an 8K content addressable memory (CAM) and a 20K-gate finite-state automaton logic (FSAL). A number of unique functions, such as strict/approximate-match string search and fixed/variable-length "don't care" operations, were implemented. A total of 217 600 transistors have been integrated on an 8.62 × 12.76-mm die. The unique functions were efficiently tested by the scan path method. The data comparison rate was 5.12 billion characters/s in text search application.**

I. Introduction

RECENTLY, the amount of information to be processed has increased every year. It is very important to be able to pick up necessary information from the information database as fast as possible. Therefore, research for data string-search processors [1]–[6] has become extensive to cope with increasing demands for fast data retrieval. So far, several hardware algorithms for data string search have been presented. However, there are only a few VLSI-oriented hardware architectures. Table I shows the conventional data string-search methods. In the associative memory method, the input data string is concurrently compared with the stored data strings in an associative memory, so a high data comparison rate is obtained. However, this method cannot be used to process variable-length data strings. The cellular array method utilizes the array of logic cells. When an input data matches the stored data in the cell, a match flag is transferred to the adjacent cell. A string match signal is obtained, when the match flag reaches the final cell. This method allows variable-length data string search. However, each cell requires extensive hardware. The finite-state automaton method uses a state transition table stored in a RAM. Although this method can be used to execute complex data search functions, it requires a large memory volume. The dynamic programming method uses a parameter table memory and proximity calculator. This method allows approximate-match search, but the comparison rate is low. As described above, there is no string-search processor which is VLSI oriented and capable of sophisticated string-search operations.

TABLE I
CONVENTIONAL DATA STRING-SEARCH METHODS

METHODS	FLEXIBILITY	HARDWARE VOLUME
ASSOCIATIVE MEMORY	*	○
CELLULAR ARRAY	*	*
FINITE STATE AUTOMATON	○	*
DYNAMIC PROGRAMMING	○	*

○=GOOD *=POOR

A new architecture for a data string-search processor, which is suitable for VLSI implementation and having versatile functions, has been proposed [7]. It combines a content addressable memory (CAM) and a finite-state automaton logic (FSAL) to compare the input data strings with the stored data strings [8]. Such an architecture makes it possible to execute an approximate-match search, in addition to strict-match search, and fixed/variable-length "don't care" operations. The approximate-match search allows single data mismatch. The variable-length "don't care" allows partial string search. These functions were difficult to realize by conventional methods, such as the sort search method [9] and associative memory method [10]. In this paper, some of the unique functions and circuits, the layout method, the test method, and application are described.

II. Functions

A conceptional block diagram of the new data string-search VLSI [11] is illustrated in Fig. 1. In this diagram, character code strings are used as data strings. Basic operation is as follows: character strings "*ABC*," "CUSTOM", "VLSI," etc. are stored in the CAM as reference data. Then, input character data strings "*ABC* ···" are entered in series, character by character. Each character code is compared parallel with the stored reference data in the CAM. Next, the character data string sequence is examined in the FSAL. Finally, if a string sequence match is found, the match signal is generated and the match address is produced through the encoder.

A block diagram for the actual data string-search VLSI is shown in Fig. 2. It consists of an 8K CAM and a 20K-gate FSAL, a priority encoder, and control circuits.

Manuscript received August 7, 1987; revised January 4, 1988.
M. Hirata is with the System LSI Development Division, NEC Corporation, 1120 Shimokuzawa, Sagamihara 229, Japan.
H. Yamada, H. Nagai, and K. Takahashi are with the Microelectronics Research Laboratories, NEC Corporation, 1120 Shimokuzawa, Sagamihara 229, Japan.
IEEE Log Number 8819756.

0018-9200/88/0400-0329$01.00

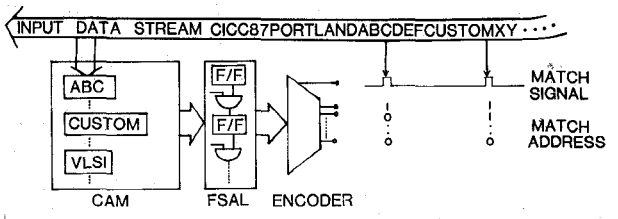


Fig. 1. A conceptional block diagram of the data string-search VLSI.

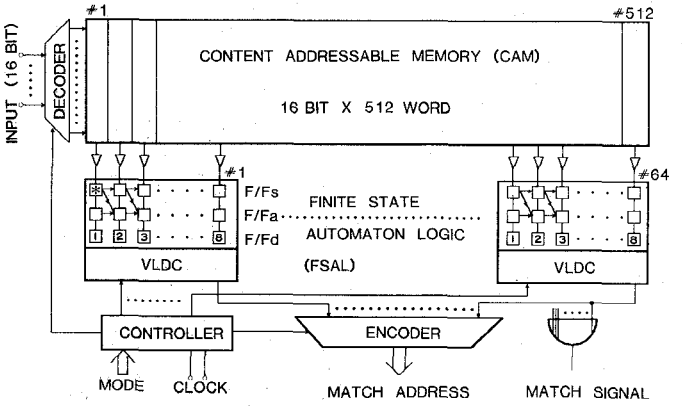


Fig. 2. A block diagram of the data string-search VLSI.

The CAM is made up of the newly developed pair-bit memory structure [8]. The 2-bit CAM cell is composed of four static RAM cells and one store/search circuit. This configuration reduces the logic portion and allows fixed-length "don't care" operation. In the CAM, variable-length data strings can be stored, ranging from a 16-bit×8-word ×64-string format to a 16-bit×512-word×1-string format. Each word has a 16 bit width, and eight words are basically assigned for one data string. A maximum of 64 data strings can be stored in the CAM. The data string length can be extended continuously by connecting adjacent eight CAM words. If all the CAM words are connected in series, 512 word-length data can be stored as one string. Eight delimiter symbols for the anchor mode search and eight masking data for mask function are also stored in the CAM. The CAM executes a single data comparison between series input data and all the stored data in parallel. As a result, high-speed data comparison is achieved. When input data and stored data match, a single data match signal is generated at the CAM output and transferred to the FSAL. The FSAL is composed of flip-flops and logic gates. There are three groups of 512-stage flip-flops. They are strict-match flip-flops (F/Fs), approximate match flip-flops (F/Fa), and delimiter flip-flops (F/Fd). The F/Fs and F/Fa hold the data comparison results. The delimiter flip-flops hold the string end marks for variable-length data strings stored in the CAM. The state transition algorithm for strict- or approximate-match search is executed with the logic gates. The FSAL carries out data string sequence comparison, using the single data word comparison results from the CAM. A string match operation is as follows. If the input data code matches the stored data word, a match flag is set in the strict-match flip-flop. The match flag is transferred in the strict-match flip-flops as long as the following input data matches the stored data. If a data mismatch occurs, the match flag is transferred to the approximate-match flip-flops. Then, the flag is transmitted in the approximate flip-flops, as long as the succeeding input data matches. Finally, when the flag reaches the end mark position in the delimiter flip-flops, the string match signal is obtained. The encoder generates the matched data address codes. When double errors have occurred, the match flag in the approximate-match flip-flops disappears.

TABLE II
UNIQUE FUNCTIONS OF THE DATA STRING-SEARCH VLSI

1. STRICT / APPROXIMATE STRING MATCH SEARCH
2. FIXED / VARIABLE LENGTH DATA STRINGS MATCH
3. ANCHOR / NON-ANCHOR STRING MATCH
4. FIXED / VARIABLE LENGTH DONT CARE MATCH
5. MULTIPLE MATCH ADDRESS RESOLVE
6. MASK FUNCTIONS

The unique functions of this ASIC are listed in Table II. This ASIC executes an approximate string match, in addition to strict string match. The approximate string match allows single data (16-bit code) omission, insertion, and substitution errors in a string data. This function is useful for text search, since texts usually include mistypes and misspellings. Data strings less than eight words are stored in every eight CAM words in the fixed-length mode. For the data exceeding eight words, adjacent CAM cells are sequentially connected for up to 512 words in the variable-length mode. In the nonanchor mode search, data string comparison is executed without delimiting symbols between the strings. This mode is especially useful for Japanese language text search, which has almost no punctuation. Fixed- and variable-length "don't care" (VLDC) searches are also possible. In the VLDC mode, data strings which start from a specified partial string and finish at another specified partial string can be searched. The length of the two specified partial data strings and intermediate data string are variable. The VLDC function enables flexible operations, such as extremely long string data search and search for string data including indefinite string.

Moreover, two kinds of masking functions are available as shown in Fig. 3. One is the bit-masking function (Fig. 3(a)) for the input data. Any bits of a 16-bit width input data can be masked every two bits. Mask data are stored in the mask data register. Masking is carried out by inactivating the decoder, according to the mask data. The comparison for the data bits connected to the inactivated decoder is abandoned. Then, the shaded horizontal data areas in the CAM are masked, as shown in Fig. 3. The other is the data masking function, or so called "wild card" (Fig. 3(b)). Eight kinds of wild cards can be stored in the CAM. When a wild card is detected in an input data

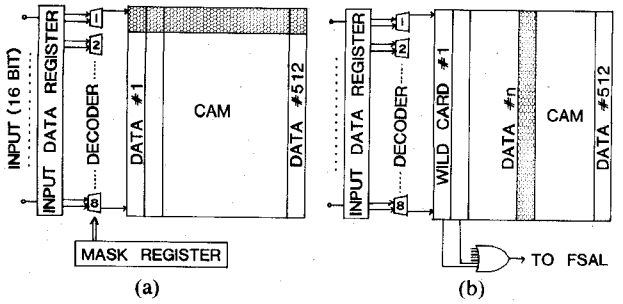


Fig. 3. Masking functions. (a) Bit masking. (b) Wild cards.

string, the data comparison for the shaded vertical data are ignored. For example, suppose that "$" is stored as a wild card and "*ABC*," "*AVC*," and "*BBC*" are stored as reference words in the CAM. If an input data string "*A$C*" is entered, match signals are generated from the CAM for both the stored data "*ABC*" and "*AVC*". In an optical character reader, an unrecognized character is replaced with "$" or "?" marks. The wild-card function allows searching data strings including such indefinite data.

Thus, this ASIC allows flexible search operations. However, in case of the approximate and mask mode search, multiple matches may occur. In this VLSI, the encoder can resolve all individual match address codes in sequence from the lower address. A 6-bit match address ($2^6 = 64$ strings) is serially obtained synchronously with the output control clock.

III. Circuit Design

A. Content Addressable Memory

The CAM is composed of conventional static RAM cells and store/search circuit. The input of the CAM is 16-bit width, which is separated every two bits. Fig. 4 shows the CAM circuit unit, which is capable of 2-bit content addressable operation. The store/search circuit executes the data clear, store, and match search. CELL0 ~ CELL3 are conventional six-transistor static RAM cells. The 4-word × 1-bit static RAM works as a 2-bit CAM. The CAM operation is as follows. First, all the cells are reset by activating the CLR line. Next, reference data is stored. The data themselves are used as address data, and are input into the decoder. The decoder selects one of the RAM cells, and ONE is stored in the selected cell by activating STR line. Two-bit data are stored in the 4-bit cell. Then, in the search mode, bit line B is precharged by *PCH* line, and input data are entered into the decoder. If the input data matches the stored data, the bit line B is maintained to the precharged level. The match output line *SCH* holds a high level. On the other hand, if the input data show a mismatch, the bit line B is discharged. Therefore, the readout signal from the RAM cells represents the data comparison result between the stored data and input data, instead of the stored data itself. This pair-bit CAM can execute "don't care" operation by storing multiple ONE in the RAM cells. If all the four RAM cells hold ONE, the match output is obtained for any input data. In order to compare 16-bit data, eight outputs from eight pair-bit CAM cells are connected to the *SCH* line by wired-AND logic.

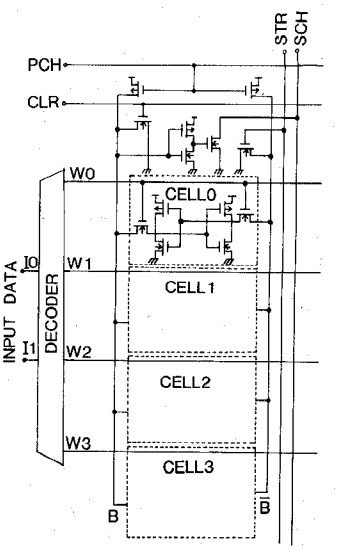


Fig. 4. Schematic of the content addressable memory.

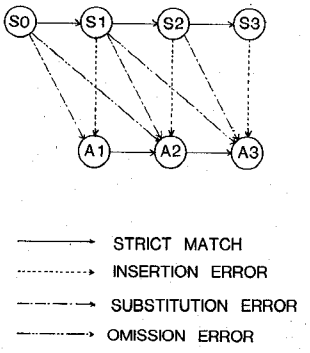


Fig. 5. A state transition diagram for data string search.

B. Finite-State Automaton Logic

The FSAL examines data string sequence match. Fig. 5 shows the state transition diagram for detecting data string match. The circles indicate the states. $S0$–$S3$ are for the strict-match states, and $A1$–$A3$ are for the approximate-match states. Each state holds the match comparison result, that is, ONE is for match and ZERO is for mismatch. $S0$ stands for the initial state, and the state transition starts from this state. The final comparison result is obtained from $S3$ and $A3$. The arrows indicate state transitions, such as strict match, single data insertion, substitution, and omission.

1) If the input data strictly match the stored data, the state transition occurs as $S0 \rightarrow S1 \rightarrow S2 \rightarrow S3$. Then, strict-match output is obtained at $S3$.

2) In case of single data insertion, the state transitions are $S0 \rightarrow S1 \rightarrow A1 \rightarrow A2 \rightarrow A3$, $S0 \rightarrow S1 \rightarrow S2 \rightarrow A2 \rightarrow A3$, and $S0 \rightarrow S1 \rightarrow S2 \rightarrow S3 \rightarrow A3$.

3) For single data substitution error, state transitions are $S0 \rightarrow A1 \rightarrow A2 \rightarrow A3$, $S0 \rightarrow S1 \rightarrow A2 \rightarrow A3$, and $S0 \rightarrow S1 \rightarrow S2 \rightarrow A3$.

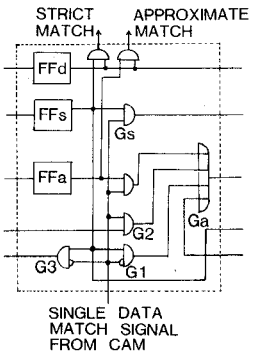


Fig. 6. Schematic of the finite-state automaton logic.

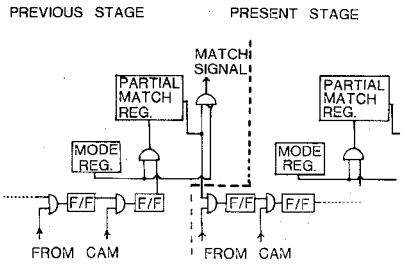


Fig. 7. Organization of the variable-length don't care circuit.

4) When single data is missing, the state transitions are $S0 \rightarrow A2 \rightarrow A3$, and $S0 \rightarrow S1 \rightarrow A3$. The approximate-match output is obtained at $A3$. If double mismatches occur, the match information disappears from the approximate states. Each transition is controlled by the results of single data comparison in the CAM.

The FSAL has been designed using flip-flops and logic gates, as shown in Fig. 6. This block shows one stage of the FSAL. The states were realized by D-type flip-flops and state transitions were implemented by logic gates. The F/Fs and F/Fa hold the strict- and approximate-match states, respectively. The F/Fd stores the data end mark for variable-length data strings. If the single data match signal is supplied from the CAM, the match information in the F/Fs and F/Fa is transferred to the next stage. On the contrary, when the single data match signal is not supplied, the match information in the F/Fs is transferred to the next stage F/Fa. Meanwhile, the match information in the F/Fa drops out. $G1$, $G2$, and $G3$ gates control the substitution, omission, and insertion errors, respectively.

C. Variable-Length Don't Care (VLDC)

One of the unique features of this ASIC is a VLDC mode search operation, in addition to fixed-length don't care (FLDC) mode function. Fig. 7 shows the VLDC circuit. Two stages are indicated, a previous stage and a present stage. Each stage includes eight-stage flip-flops. The flip-flops hold the results of comparison. In this figure, approximate-match flip-flops, delimiter flip-flops, and the logic gates are not shown in order to explain the VLDC operation simply. First, the mode register of the previous stage is set to ONE to suppress the partial match signal. When the header partial string matches the input data string, the partial match register is set to ONE. Then, if the bottom part matches, the match signal of the present stage is generated. The intermediate data string comparisons are neglected. The VLDC function enables flexible operations in a text search.

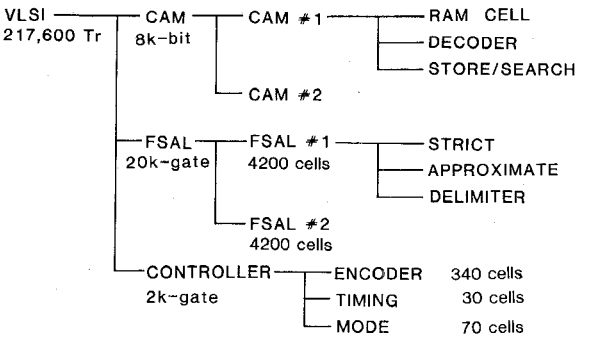


Fig. 8. Hierarchical layout structure of the VLSI.

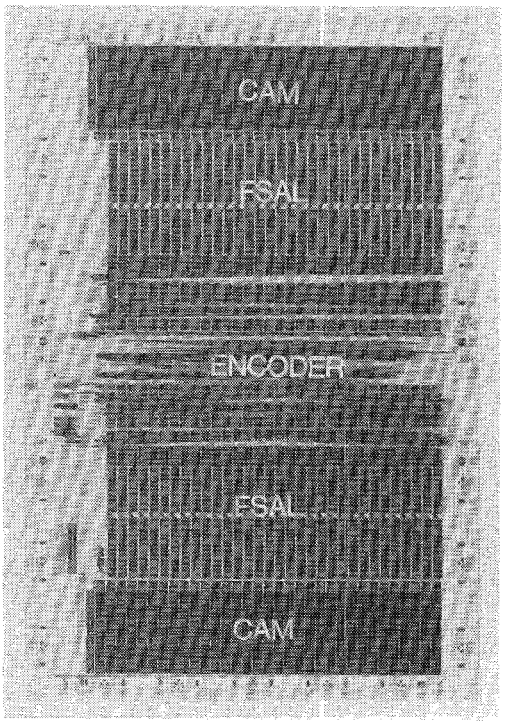


Fig. 9. VLSI layout pattern. The chip size is 8.62×12.76 mm.

IV. Layout Design and Fabrication

The chip layout has been carried out using a hierarchical layout method, as shown in Fig. 8. The CAM is divided into two blocks. Each block has 4K capacity. They were designed in a full-custom manner to minimize the cell area and to obtain a high-speed access. The FSAL is also divided into two blocks. The FSAL blocks and the controller block were designed using 8840 standard cells and a layout CAD tool to attain quick design turnaround time. The VLSI layout pattern is shown in Fig. 9. The CAM and FSAL have regular structures, which are suitable for VLSI implementation. The FSAL block pitch was designed to be the same as the CAM cell pitch, to allow connecting time

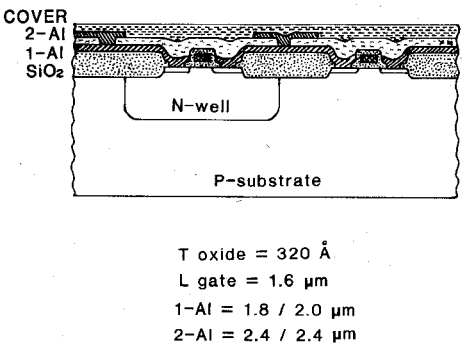


Fig. 10. Cross-sectional view of the CMOS device.

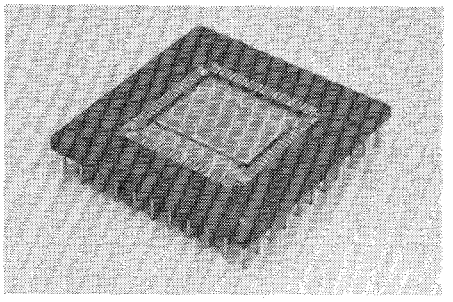


Fig. 11. A microphotograph of the VLSI package.

to each other directly with a short metal line. As a result, a high-speed response was obtained, due to small parasitic capacitance. A total of 217 600 transistors have been integrated on the 8.62×12.76-mm die area.

The VLSI was fabricated using 1.6-μm CMOS double-metal layer technology. The device cross-sectional view is shown in Fig. 10. The gate oxide thickness is 320 Å, and the gate length is 1.6 μm. First metal layer line/space is 1.8/2.0 μm, and second metal layer line/space is 2.4/2.4 μm. The die was mounted on a 72-pin grid array package, as shown in Fig. 11. Table III shows the pin configuration. Sixteen pins are for the data input, 12 pins are for the store address and match address, 11 pins are for the mode control and clocks, and 10 pins are for function tests.

V. Test Method

The number of devices which can be integrated in a silicon VLSI chip is increasing every year. However, the I/O pins are restricted to a small number. Thus, the difficulty involved in function verification is increasing every year. Therefore, testability must be taken into account in the VLSI design to reduce test costs and development time period. Several test methods have been developed for custom LSI's [12], [13]. By including test circuits in a chip, test period and test pattern numbers are reduced. However, the chip area increase is inevitable. Consequently, a test method should be considered to minimize die size increase. The scan-path method has been used in this VLSI design, considering the above mentioned discussions. The test circuit is shown in Fig. 12. In the test mode, the CAM and FSAL are separated by the test control signal. Then, the flip-flops in the FSAL are connected in series. Next, the test patterns are provided directly to the FSAL through test-in pins. The test patterns were supplied to the delimitter flip-flop chain, strict-match flip-flop chain, approximate-match flip-flop chain, and internal control flip-flop chain, separately. The output patterns are obtained from test-out pins at the FSAL final stage, and are compared with the simulated result pattern. Since the test patterns for the FSAL are given through the scan-path circuit instead of through the CAM, high testability was obtained, regardless of CAM defects. In addition, the FSAL was tested in a short time, because the rewrite cycle of the CAM in order to change the test data is not necessary. The VLDC circuit and encoder were also tested using the test pins. The CAM has no separate READ circuit in order to minimize memory area. Then, the CAM was tested as follows: 1) write ONE and ZERO alternately in the cells; 2) transfer the match signal from the CAM to the FSAL in parallel, and 3) read out the FSAL data in series from the test-out pin. The gate increase for the test circuit was less than 3 percent.

VI. Application

Requirements for retrieving necessary texts from the amount of text files by using keywords are many. Generally, text information is filed in magnetic disks or optical disks. Conventionally, text was searched using software and it took much time. Furthermore, approximate-match search and VLDC mode search were not available. Application for text search has been investigated using this VLSI. A text-search system is shown in Fig. 13. Text data, such as technical papers and patents, are stored in the DISK file in unformatted form. Necessary texts are retrieved from the DISK file by using several keywords. First, all the flip-flops and memory cells in the VLSI are cleared by reset pulse. Then, several keywords are stored into the CAM in the store mode. A total of 512-character data can be stored in variable-length formats. Next, input character strings are entered to the VLSI from the DISK file in the search mode. If an input data string matches any of the stored keywords, a match signal and address code are outputted at the encoder. Then, the text file including these keywords is transferred to the host computer from the DISK for further data processing. Multiple VLSI chips can be connected in parallel to expand CAM capacity. Then, the number of keywords that can be stored in the chip is increased. Moreover, a multiple-chip configuration allows parallel text search for many users. The pulse waveforms in text-search operation are shown in Fig. 14. The input data rate was 10^7 characters/s. Since 512-character data in the CAM are concurrently compared, the data comparison rate was 5.12 billion characters/s. This speed is several hundred times faster than that of conventional character data comparison methods.

TABLE III
PIN CONFIGURATION

PIN NO.	PIN NAME	FUNCTION
1	DATA0	INPUT PINS FOR STORE/SEARCH DATA
16	DATA15	
17	PDIN	INPUT CONTROL (0, 0) 16BIT WIDTH/ 8BIT INPUT×2 (0, 1) 8BIT WIDTH/ 8BIT INPUT (1, 0) 16BIT WIDTH/16BIT INPUT (1, 1)
18	SBYT	
19	IMAD0	STORE DATA ADDRESS
20	IMAD5	
21	OMAD0	MATCHED DATA ADDRESS OUTPUT
26	OMAD5	
27	MAEN	MATCHED ADDRESS OUTPUT ENABLE
28	ANCM	1···NON-ANCHOR MODE 0···ANCHOR MODE
29	ENBL	CHIP ENABLE
30	RESET	CAM and FSAL CLEAR
31	MODE0	OPERATION MODE CONTROL (0, 0, 0) STRICT MATCH ADDRESS OUTPUT (0, 0, 1) APPROXIMATE MATCH ADDRESS OUTPUT (0, 1, 0) FLIP-FLOP RESET (0, 1, 1) VLDC SET (1, 0, 0) FIRST DATA STORE (1, 0, 1) NEXT DATA STORE (1, 1, 0) MASK DATA STORE (1, 1, 1) DELIMITER DATA STORE
32	MODE1	
33	MODE2	
34	SMCH	STRICT MATCH SIGNAL
35	OMCH	APPROXIMATE MATCH SIGNAL
36	CLCK	DATA INPUT CONTROL CLOCK
37	MCLK	MATCHED ADDRESS OUTPUT CLOCK
38	AUXCK	AUXILIARY CLOCK
39,40,41,42	VDD	POWER SUPPLY (5V)
43,44,45,46	GND	GND
47	TSTM	TEST CONTROL
48	TSMI	STRICT MATCH F/F TEST INPUT
49	TSMO	STRICT MATCH F/F TEST OUTPUT
50	TAMI	APPROXIMATE MATCH F/F TEST INPUT
51	TAMO	APPROXIMATE MATCH F/F TEST OUTPUT
52	TDLI	DELIMITER F/F TEST INPUT
53	TDLO	DELIMITER F/F TEST OUTPUT
54	TSF0	INTERNAL F/F CONTROL (0, 0) VLDC F/F SET (0, 1) HEADER F/F SET (1, 0) DELIMITER F/F SET (1, 1) MATCH F/F SET
55	TSF1	
56	TMSK	SEPARATE CAM and FSAL

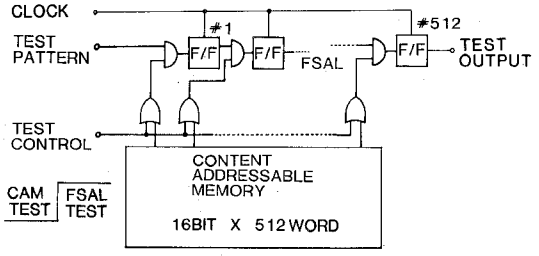


Fig. 12. Block diagram of the test circuits.

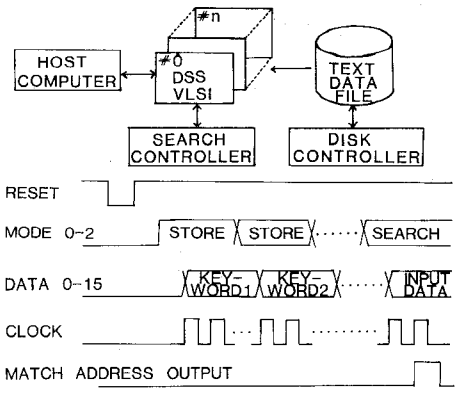


Fig. 13. A text-search configuration.

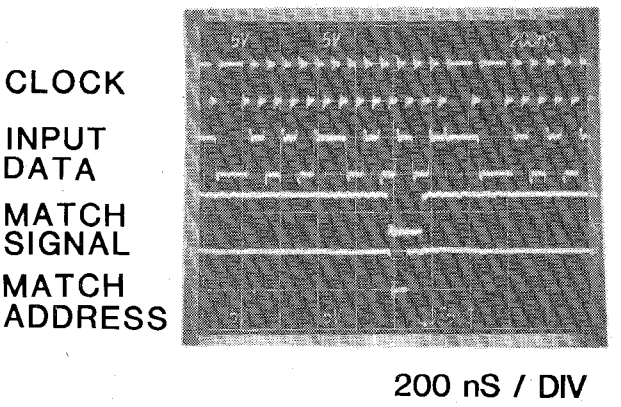


Fig. 14. Waveforms in the text-search application.

VII. CONCLUSIONS

A versatile data string-search VLSI has been described. An 8K content addressable memory and a 20K-gate finite-state automaton logic have been combined to execute data string search. This architecture allowed versatile operations, such as approximate-match and variable-length "don't care" search at high speed. The regularity in the

memory and state transition logics was suitable for VLSI implementation. A total of 217 600 transistors have been integrated by a combination of full-custom design and the standard cell method on a 12.76 × 8.62-mm area. The VLSI was fabricated, using 1.6-μm double-level metal CMOS technology. The unique functions were efficiently verified by the scan-path method, with less than 3-percent gate increase. The data comparison rate was 5.12 billion data/s. The versatile functions implemented on this chip were very useful in text search application.

ACKNOWLEDGMENT

The authors would like to thank K. Sugibuchi, N. Yoshida, Y. Hiki, and K. Matsumoto for their encouragement. They also wish to thank K. Yoshimi, H. Sakuma, and Y. Kitamura for direction. Thanks also go to Y. Ohuchi for CAD support and S. Iwata for device fabrication.

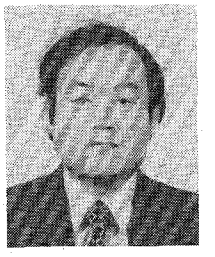
REFERENCES

[1] M. J. Foster and H. T. Kung, "The design of special-purpose VLSI chips," *IEEE Computer*, vol. 13, no. 1, pp. 26–40, Jan. 1980.
[2] A. Mukhopadhyay, "Hardware algorithms for string processing," in *Proc. ICCC'80*, 1980, pp. 508–511.
[3] R. L. Haskin, "Special purpose processors for text retrieval," *Database Eng.*, vol. 4, no. 1, pp. 16–29, Sept. 1981.
[4] P. N. Yianilos, "A dedicated comparator matches symbol strings fast and intelligently," *Electronics*, pp. 113–117, Dec. 1983.
[5] J. P. Wade, P. J. Osler, R. E. Zippel, and C. G. Sodini, "The MIT database accelerator: 2-K TRIT circuit design," in *VLSI Symp. Dig. Tech. Papers*, May 1987, pp. 39–40.
[6] S.-I. Chae, J. T. Walker, C.-C. Fu, and R. F. Pease, "Maskable associative memory design for VLSI pattern inspection," in *VLSI Symp. Dig. Tech. Papers*, May 1987, pp. 47–48.
[7] K. Takahashi, H. Yamada, H. Nagai, and K. Matsumi, "A new string search hardware architecture for VLSI," in *Proc. 13th ISCA*, 1986, pp. 20–27.
[8] H. Yamada, M. Hirata, H. Nagai, and K. Takahashi, "A character string search processor," in *ISSCC Dig. Tech. Papers*, Feb. 1987, pp. 272–273.
[9] M. Segal, "Hardware sorting chip steps up software pace," *Electron. Des.*, pp. 85–91, June 1986.
[10] F. J. Burkowski, "A hardware hashing scheme in the design of a multiterm string comparator," *IEEE Trans. Computers*, vol. C-31, pp. 825–834, Sept. 1982.
[11] M. Hirata, H. Yamada, H. Nagai, and K. Takahashi, "A versatile data string search VLSI," in *Proc. CICC'87*, May 1987, pp. 563–566.
[12] M. J. Y. Williams and J. B. Angell, "Enhancing testability of large-scale integrated circuits via test points and additional logic," *IEEE Trans. Computers*, vol. C-22, pp. 46–60, Jan., 1973.
[13] E. J. McCluskey and S. Bozorgui-Nesbat, "Design for autonomous test," *IEEE Trans. Computers*, vol. C-30, pp. 866–875, Nov. 1981.

Masaki Hirata was born in Hiroshima, Japan, on March 9, 1949. He received the B.S. and M.S. degrees in electronic engineering from the University of Electronics Communication, Tokyo, Japan, in 1972 and 1974, respectively.

He joined NEC Corporation, Kawasaki, Japan, in 1974, where he has been working in MOS IC design. His current research interests are in custom LSI design. He is now a Supervisor in the System LSI Development Division, NEC Corporation.

Mr. Hirata is a member of the Institute of Electronics, Information and Communication Engineers of Japan.



Hachiro Yamada was born in Shizuoka, Japan, on June 6, 1949. He graduated from Shizuoka Technical High School in 1968, and from the Nippon Electric Institute of Technology in 1973.

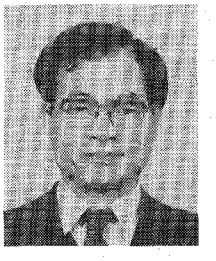
In 1968 he joined NEC Corporation, Kawasaki, Japan, where he was engaged in the research and development of memory hierarchy systems and magnetic bubble memory systems. Since 1985 he has been engaged in the development of VLSI architecture and special-purpose LSI. He is presently a Supervisor in the Microelectronics Research Laboratories, NEC Corporation.

Mr. Yamada is a member of the Institute of Electronics, Information and Communication Engineers of Japan.

Hajime Nagai was born in Hyougo, Japan, on November 1, 1951. He received the B.E. degree in electrical engineering from Saitama University in 1975.

He joined NEC Corporation, Kawasaki, Japan, in 1975 and is now a Supervisor in the Microelectronics Research Laboratories. He has been engaged in the development of a bubble memory system. Currently he is studying AI devices.

Mr. Nagai is a member of the Institute of Electronics, Information and Communication Engineers of Japan.



Kousuke Takahashi was born in Mie, Japan, on November 22, 1940. He received the M.S. degree from Keio University in 1966.

He joined NEC Corporation, Kawasaki, Japan, in 1966 and is now a Senior Research Specialist in the Microelectronics Research Laboratories. He has been engaged in the research and development of plated-wire memory, analog magnetic thin-film memory, and magnetic bubble memory. Now he is researching VLSI architecture for silicon devices.

Mr. Takahashi is a member of the Institute of Electronics, Information and Communication Engineers of Japan, the Information Processing Society of Japan, the Magnetic Society of Japan, and the Audio-Visual Information Research Group (AVIRG).