High-Density Quaternary Logic Array Chip for Knowledge Information Processing Systems

doi: 10.1109/4.34071

High-Density Quaternary Logic Array Chip for Knowledge Information Processing Systems

TAKAHIRO HANYU, STUDENT MEMBER, IEEE, AND TATSUO HIGUCHI, SENIOR MEMBER, IEEE

Abstract — **A new high-density NMOS logic array chip based on quaternary logic is implemented for high-speed parallel pattern matching in a knowledge information processing system. The logic array can be exploited in real-time applications when the rules are fixed. Based on the appropriate quaternary coding for the contents of working memory and production memory, a double-pattern-matching algorithm is proposed for achieving a high-density chip. One of four states for 2-bit information concerning two elements of a rule is stored into a pattern-matching cell by multiple ion implants, so that the pattern-matching cell is implemented using only a single transistor. It is demonstrated that the chip area for pattern matching is reduced by 30 percent compared with the corresponding binary logic array.**

I. Introduction

THE knowledge information processing system has some possibilities that exhibit intelligent behavior. Conventionally, it has been well-known that pattern matching is a basic component of the knowledge information processing system interpreters. For instance, production systems have been observed to spend more than nine-tenths of their total run time performing pattern matching [1]. If high-speed reasoning is available, it will be more widely used in the applications of complex and sophisticated real-time systems such as the autonomous land rover [2]–[4]. In order to achieve real-time applications, the knowledge information processing system requires high computational power and large memory capacity at low cost [5], [6].

This paper presents a new high-density logic array for high-speed pattern matching based on a totally parallel structure [7]. In particular, a double-pattern-matching algorithm is effectively employed for achieving greater densities on smaller semiconductor chips [8]. The appropriate quaternary encoding for the contents of working memory and production memory makes this double pattern matching very simple.

Manuscript received December 7, 1988; revised March 10, 1989.
The authors are with the Department of Electronic Engineering, Tohoku University, Aoba, Aramaki, Sendai 980, Japan.
IEEE Log Number 8928812.

One of four states corresponding to the 2-bit information in the rules is stored into a pattern matching cell by using the multiple ion-implant technique, which makes the threshold of the transistor programmable [9]. Moreover, the basic operation in the double-pattern-matching procedure is equivalent to the operation of the "threshold literal function" [10], [11] where the threshold corresponds to the quaternary-coded information of each element in the rules. As a result, the pattern-matching cell is implemented using a single transistor.

Since the 2-bit information in a single transistor is stored by using multiple ion implants and is processed simultaneously, the processing capability per unit cell is greatly increased. In fact, the number of cells and transistors is reduced to 50 percent of the corresponding binary implementation because of the double pattern matching. Moreover, the pattern-matching operation is performed in parallel, so that the processing time on each reasoning cycle is provided by the propagation delay time of a single transistor.

II. Hardware Algorithm for Quaternary Pattern Matching

Production systems (PS's) discussed here which are frequently used in knowledge information processing systems have the following components:

1) a working memory (WM): a set of data structures or elements C_i representing the current state of the system;
2) a production memory (PM): a set of rules of the form, if ⟨the logical product terms of elements $C_1 \wedge C_2 \wedge \cdots \wedge C_n$⟩ then ⟨do action or working memory changes⟩, where the symbol $\wedge$ indicates the logic product AND;
3) a rule interpreter: which applies the production rules to the working memory.

The production interpreter repeatedly looks for production rules whose left-hand sides match working memory.

0018-9200/89/0800-0916$01.00

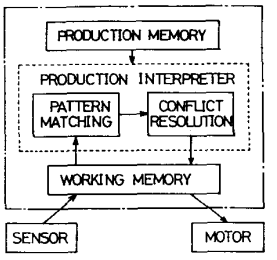


Fig. 1. Functional blocks in production systems.

TABLE I
ENCODINGS FOR x_i AND X_i

BINARY-CODED INFORMATION	x_1	x_2	x_3	x_4	x_5	x_6
	1	0	1	1	0	1
QUATERNARY-CODED INFORMATION	X_1		X_2		X_3	
	1		3		2	

On each cycle, it picks up a rule, and does what its right-hand side dictates. By repeating the above cycle, the forward reasoning in PS can be completed for solving the problem [12]. Fig. 1 shows a schematic of the forward reasoning in PS.

In the following section, new encoding of the left-hand sides in WM and PM are presented for quaternary pattern matching.

A. Method of Quaternary Encoding in WM

Let us denote the information on the ith element in WM as x_i. This variable x_i indicates whether the ith element C_i exists in WM or not, and is defined by

$$x_i = \begin{cases} 0, & \text{if } C_i \text{ does not exist in WM} \\ 1, & \text{if } C_i \text{ exists in WM} \end{cases} \tag{1}$$

where $x_i \in \{0,1\}$.

Furthermore, let us express the ith quaternary-coded information in WM as X_i. X_i can be obtained from the binary-coded digits x_{2i-1} and x_{2i} in WM which indicates the existence of the two elements C_{2i-1} and C_{2i}, and is defined as follows:

$$X_i = \begin{cases} 1, & \text{if only the element } C_{2i-1} \text{ exists} \\ 2, & \text{if only the element } C_{2i} \text{ exists} \\ 3, & \text{if both of the elements } C_{2i-1} \text{ and } C_{2i} \text{ exist} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

For example, Table I shows both the binary and quaternary encodings in WM when the following elements are contained in WM:

$$\mathrm{WM} = \{C_1, C_3, C_4, C_6\}. \tag{3}$$

TABLE II
ENCODINGS FOR y_i AND Y_i

	y_1	y_2	y_3	y_4	y_5	y_6
	Y_1		Y_2		Y_3	
RULE 1	1	0	0	0	1	0
	1		0		1	
RULE 2	0	1	0	1	1	0
	2		2		1	
RULE 3	1	0	1	1	0	1
	1		3		2	

B. Method of Quaternary Encoding in PM

It is also necessary to express the left-hand side of each production rule as a set of quaternary-coded information in order to perform the double pattern matching. Let us denote the information on the ith element in PM as y_i. This also indicates whether the ith element C_i exists or not in the left-hand side of each rule, and is defined by

$$y_i = \begin{cases} 0, & \text{if } C_i \text{ does not exist in PM} \\ 1, & \text{if } C_i \text{ exists in PM} \end{cases} \tag{4}$$

where $y_i \in \{0,1\}$.

Moreover, let us denote the ith quaternary-coded information in the left-hand side of the kth rule as Y_i. Y_i can be obtained from the binary-coded information y_{2i-1} and y_{2i} in PM. The variable Y_i indicates the existence of the two elements C_{2i-1} and C_{2i}, and is defined by

$$Y_i = \begin{cases} 1, & \text{if only the element } C_{2i-1} \text{ exists} \\ 2, & \text{if only the element } C_{2i} \text{ exists} \\ 3, & \text{if both of the elements } C_{2i-1} \text{ and } C_{2i} \text{ exist} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

For example, Table II shows both of the binary and quaternary encodings in PM when PM consists of the following rules:

$$\begin{aligned} &\text{Rule 1:} && C_1 \wedge C_5 \longrightarrow A_1 \\ &\text{Rule 2:} && C_2 \wedge C_4 \wedge C_5 \longrightarrow A_2 \\ &\text{Rule 3:} && C_1 \wedge C_3 \wedge C_4 \wedge C_6 \longrightarrow A_3. \end{aligned} \tag{6}$$

C. Formulation of Quaternary Pattern Matching

Table III shows the truth table of quaternary pattern matching between the quaternary-coded information X_i and its corresponding information Y_i where the output values 3 and 0 correspond to "match" and "no match," respectively.

Now, consider the matched result U_i and $X_i = 0$ and $Y_i \in \{0,1,2,3\}$. The case of $X_i = 0$ and $Y_i = 0$ indicates "neither of the elements C_{2i-1} nor C_{2i} exists" in WM and PM. Hence, X_i matches Y_i, so that the matched result U_i becomes 3. The case of $Y_i \in \{1,2,3\}$ indicates "either of the elements C_{2i-1} or C_{2i} exists" in PM. In this case, Y_i does not match X_i, so U_i becomes 0.

TABLE III
DOUBLE PATTERN MATCHING

(x_{2i-1}, x_{2i})		(0,0)	(1,0)	(0,1)	(1,1)
X_i		0	1	2	3
Y_i	0	3	3	3	3
	1	0	3	0	3
	2	0	0	3	3
	3	0	0	0	3

From Table III, the double-pattern-matching operation can be obtained as

$$U_i = \begin{cases} X_i^{Y_i 3}, & \text{if } Y_i \in \{0,2,3\} \\ 0, & \text{if } Y_i \in \{1\} \text{ and } x_{2i-1} \in \{0\} \\ 3, & \text{if } Y_i \in \{1\} \text{ and } x_{2i-1} \in \{1\} \end{cases} \quad (7)$$

where x_{2i-1}, X_i, and Y_i are defined in (1), (2), and (5), respectively, and where $U_i \in \{0,3\}$ expresses the ith matched result defined by

$$U_i = \begin{cases} 3, & \text{if } X_i \text{ matches } Y_i \\ 0, & \text{if } X_i \text{ does not match } Y_i. \end{cases} \quad (8)$$

Moreover, X^{ab} in (7) denotes the literal function defined by

$$X^{ab} = \begin{cases} 3, & \text{if } a \leqslant X \leqslant b \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

where $a, b \in L = \{0,1,2,3\}$.

According to the above definition, the matched result W_h of the hth rule can be generally written as

$$\begin{aligned} W_h &= U_1 \wedge U_2 \wedge \cdots \wedge U_k \\ &= X_m^{Y_m 3} \wedge \cdots \wedge X_i^{Y_i 3} \wedge \cdots \wedge X_n^{Y_n 3} \\ &\quad \times x_{2m'-1} \wedge \cdots \wedge x_{2j-1} \wedge \cdots \wedge x_{2n'-1} \end{aligned} \quad (10)$$

where k, m, n, m', n', i, and j indicate appropriately selected natural numbers and $i \neq j$. The literal X_i is selected in (10) if $Y_i \in \{0,2,3\}$, and the binary variable x_{2j-1} is selected in (10) if $Y_j \in \{1\}$. The expansion shown in (10) makes the gate array highly compact, because the literal function can be easily designed by a single transistor with one of four different thresholds as shown in Section III.

For example, when WM has the elements shown in (3), the matched result W_3 of the third rule in (6) can be formulated as

$$\begin{aligned} W_3 &= U_1 \wedge U_2 \wedge U_3 \\ &= x_1 \wedge X_2^{33} \wedge X_3^{23}. \end{aligned} \quad (11)$$

In (11), U_1 is expressed as the binary variable x_1 because of $Y_1 = 1$. U_2 and U_3 are expressed as the literals X_2^{33} and X_3^{23} because of $Y_2 = 3$ and $Y_3 = 2$, respectively.

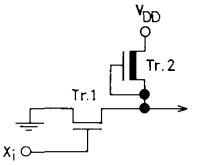


Fig. 2. Literal $X_i^{Y_i 3}$ circuit.

TABLE IV
QUATERNARY LOGIC LEVELS

LOGICAL VALUE	0	1	2	3
VOLTAGE (V)	5.0	3.3	1.7	0.0

TABLE V
TRANSISTOR PARAMETERS FOR $Y_i \in \{0,2,3\}$

Y_i	0	2	3
V_{th} (V)	5.5	2.5	0.9
V_{dep} (V)	-3.0		

III. DESIGN OF QUATERNARY LOGIC ARRAY FOR HIGH-SPEED PATTERN MATCHING

As shown in (10), the matched result W_h of the hth rule consists of the literal function $X_i^{Y_i 3}$ and the logical product $\wedge$.

A. Design of the Literal Function $X_i^{Y_i 3}$

According to the definition of (9), $X_i^{Y_i 3}$ can be expressed as

$$X_i^{Y_i 3} = \begin{cases} 3, & Y_i \leqslant X_i \leqslant 3 \\ 0, & \text{otherwise} \end{cases} \quad (12)$$

where $Y_i \in \{0,2,3\}$. X_i is a "threshold literal" function whose logical threshold corresponds to Y_i. Therefore, $X_i^{Y_i 3}$ can be easily implemented as shown in the circuit diagram of Fig. 2. The quaternary voltage levels of the input signal X_i can be determined according to Table IV. Y_i corresponds to the threshold voltages in each transistor as shown in Table V, where the symbols V_{th} and V_{dep} indicate the threshold voltages of a enhancement-mode MOS transistor (Tr.1) and a depletion-mode MOS transistor (Tr.2) in Fig. 2, respectively.

B. Design of the Logical Product $\wedge$

The logical product $X_i^{Y_i 3} \wedge X_j^{Y_j 3}$ can be easily designed by wire-ORing the outputs of the literal circuits as shown in Fig. 3. The output of $X_i^{Y_i 3} \wedge X_j^{Y_j 3}$ becomes a binary signal. Similarly, the logical product $x_i \wedge x_j$ of the binary signals can be designed. Table VI shows the relation between the logical values and the voltages for x_i. The threshold voltages for V_{th} and V_{dep} can be determined as shown in Table VII.

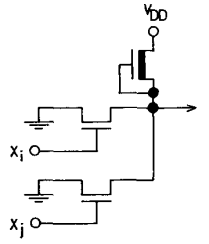


Fig. 3. Circuit diagram of a logical product term $X_i^{Y_i 3} \wedge X_j^{Y_j 3}$.

TABLE VI
BINARY LOGIC LEVELS

LOGICAL VALUE	0	1
VOLTAGE (V)	5.0	0.0

TABLE VII
TRANSISTOR PARAMETERS FOR $Y_i \in \{1\}$

Y_i	1
V_{th} (V)	2.5
V_{dep} (V)	-3.0

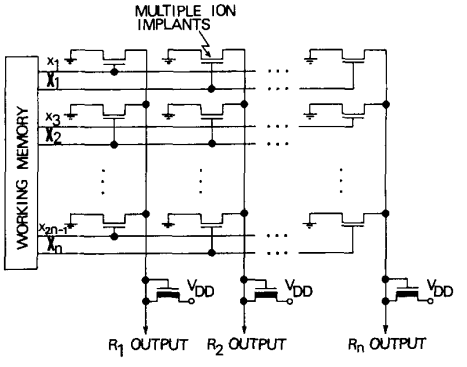


Fig. 4. Structure of a quaternary logic array for pattern matching.

C. Design of a Logic Array for Pattern Matching

Fig. 4 shows a new logic array based on quaternary logic. The binary information x_{2i-1} and x_{2i} from WM is converted into the quaternary information X_i according to (2). The encoder circuit can be easily realized. The function of the encoder circuit is summarized in Table III. For example, Fig. 5 shows the circuit diagram of the encoder with the binary inverters and the NMOS pass transistors.

Either of the input signals x_{2i-1} or X_i provided in each transistor cell is selected and connected to the gate of the NMOS transistor according to (7). Moreover, the threshold voltage of the connected transistor cell is programmed by using multiple ion implants according to the value of Y_i. Tables V and VII show the relation between the value of Y_i and the threshold voltage of the transistor cell. The output of each transistor becomes zero if the voltage level

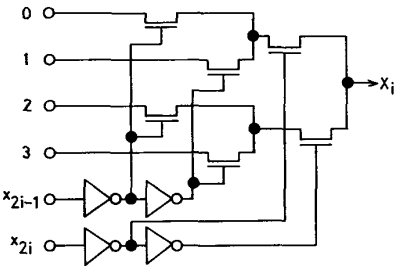


Fig. 5. Circuit diagram of a encoder.

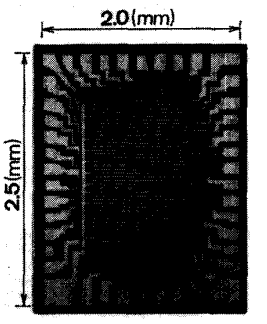


Fig. 6. Chip photomicrograph.

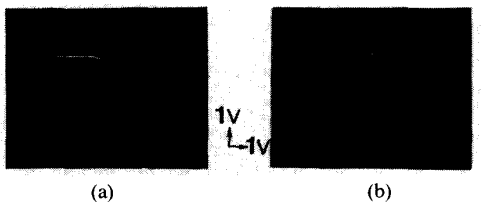


Fig. 7. DC transfer characteristics of literal circuits. (a) Characteristic of literal X^{33} circuit. (b) Characteristic of literal X^{23} circuit.

of the input value x_{2i-1} or X_i is greater than the threshold voltage of Y_i. Otherwise, the transistor cell is off. The matched result of each rule can be obtained by wire-ORing the output of each transistor.

IV. Implementation of a Quaternary NMOS Logic Array Chip

In order to confirm the principle operation of the quaternary logic array, a test chip was fabricated in the Custom LSI Design and Development System at Tohoku University as shown in Fig. 6. The design is based on 10-μm single-metal NMOS technology [13]. The chip size is 2.0×2.5 mm^2 with a total of 210 transistors. The chip performs the pattern-matching operations for 15 rules with 26 elements simultaneously. According to (7) and Table V, two kinds of literal functions are also implemented in this chip. Fig 7(a) and (b) shows the dc transfer characteristics of literal functions X_i^{33} and X_i^{23}, respectively. The dose control technique enables us to obtain the threshold voltage with high precision [14]. If such precise control is available, the desired threshold voltages become uniform as expected.

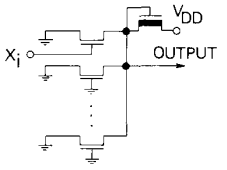


Fig. 8. Circuit diagram of an experimented logic array.

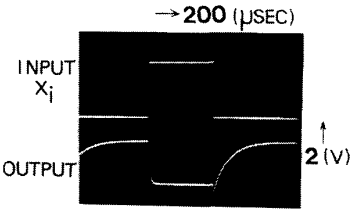
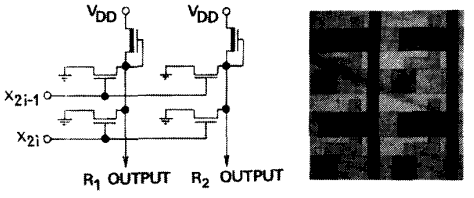
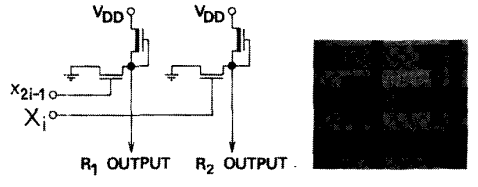


Fig. 9. Input–output waveforms of Fig. 8.



(a) (b)

Fig. 10. Binary logic array. (a) Circuit diagram. (b) Layout.



(a) (b)

Fig. 11. Quaternary logic array corresponding to Fig. 10. (a) Circuit diagram. (b) Layout.

Figs. 8 and 9 show the circuit diagram and the input–output waveforms of the matching result for a certain rule, respectively. In this case, the rule consists of 13 pattern matching cells. As shown in Fig. 9, the rise and fall times are slow because of the large capacitive load connected to the output node in the experiment. However, the substantial capacitive load in this test chip is almost equal to 2 pF, so that the worst-case propagation delay time is estimated at about 1.4 μs using SPICE2 simulation.

Fig. 10 shows the circuit diagram and layout of a binary logic array based on binary codings of WM and PM, where the binary logic array for high-speed pattern matching is also proposed here. The function of the binary logic array is equivalent to that of a quaternary logic array in Fig. 11. The number of pattern-matching cells based on binary logic is a factor of 2 times more than quaternary pattern-matching cells because of the parallel pattern-matching procedure.

TABLE VIII
PERFORMANCE OF THE QUATERNARY LOGIC ARRAY

	32-BIT MICRO-PROCESSOR	LOGIC ARRAY CHIP	
		BINARY CODING	QUATERNARY CODING
EXECUTE SPEED (SEC/TOTAL PATTERN MATCHING OPERATIONS)	1) > 3.2 (msec)	2) ≈ 1.4 (μsec)	2) ≈ 1.4 (μsec)
NUMBER OF TRANSISTORS		10×10^5	5×10^5
CHIP AREA (1/CELL)		390 λ^2 3) (390 λ^2) 4)	273 λ^2 3) (195 λ^2) 4)

1) 10 MIPS PERFORMANCE
2) 10-μm NMOS PROCESS
3) ONE-LEVEL INTERCONNECTIONS
4) MULTILEVEL INTERCONNECTIONS

From the above discussion, it is clear that the quaternary logic array becomes useful in large-scale production systems with many rules and elements. Table VIII summarizes the features of the proposed logic array for pattern matching when 1000 rules with 1000 elements are included in the system. In spite of using 10-μm design technology, the execution speed for pattern-matching operations is much faster than that of a corresponding software system which uses the same encoding algorithms of WM and PM as shown in (1) and (4).

It is assumed in Table VIII that the surface mobility of transistors and the interconnection line capacitances of input and output nodes are constant, so that the processing speed of both logic arrays becomes almost equal. The mobility is decreased with multiple ion implants and total line capacitance is proportional to the number of pattern-matching cells. Even if the above effects are estimated, the processing speed of the quaternary logic array will not be slower than that of the binary logic array.

Moreover, the number of transistors is reduced to 50 percent in comparison with the corresponding binary implementation. However, the chip area is reduced to only 30 percent because of single-metal layer technology. If multilayered metal interconnections are available, the chip area can be reduced to 50 percent.

V. Conclusion

This paper has reported on the design and implementation of a new high-density logic array chip for high-speed pattern matching which is performed by the parallel processing scheme. This chip can be implemented not only using 10-μm NMOS technology but also using submicrometer process technology. In this case, the use of advanced VLSI processing technology enables us to further increase the operating speed and the density in this quaternary logic array chip, so that the advantage of the quaternary chip is greatly enhanced. In fact, one of the most important features for the implemented chip is that the double-pattern-matching algorithm is employed by the appropriate quaternary codings of WM and PM. Moreover, the

double-pattern-matching cell is implemented using only a single transistor with multiple ion implants. As a result, the chip area has been reduced to 30 percent compared with the binary VLSI implementation.

A new architecture of a high-performance logic array chip based on quaternary logic offers very attractive features and good trade-off for high-density, high-performance logic-in memories. In the future, not only logic-in memories with higher bit density, but also symbol processors will be effectively realized by using this quaternary logic array chip.

ACKNOWLEDGMENT

The authors would like to acknowledge Associate Prof. M. Kameyama of Tohoku University for very thoughtful and helpful suggestions.

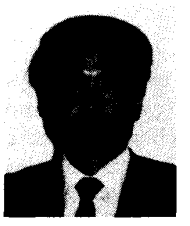
REFERENCES

[1] C. L. Forgy, "Rete: A fast algorithm for the many pattern/many object pattern match problem," *Artificial Intelligence*, vol. 19, pp. 17–37, 1982.
[2] F. Hayes-Roth, D. A. Waterman, and D. B. Lenat, *Building Expert Systems*. Reading, MA: Addison-Wesley, 1983.
[3] R. Forsyth, *Expert Systems: Principles and Case Studies*. London: Chapman and Hall, 1984.
[4] J. J. Nitao and A. M. Parodi, "A real-time reflexive pilot for an autonomous land vehicle," *IEEE Control System Mag.*, pp. 14–23, Feb. 1986.
[5] R. Reddy, "Super chips for artificial intelligence," in *ISSCC Dig. Tech. Papers*, WPM 5.1, Feb. 1985, pp. 54–55, 315.
[6] J. A. Stankovic, "Misconceptions about real-time computing," *IEEE Computer*, vol. 21, no. 10, pp. 10–19, Oct. 1988.
[7] T. Hanyu, M. Kameyama, and T. Higuchi, "Quaternary gate array for pattern matching and its application to knowledge information processing system," in *IEEE Proc. 17th Int. Symp. ISMVL*, May 1987, pp. 181–187.
[8] A. Mukhopadhyay, "Hardware algorithm for non numeric computation," *IEEE Trans. Comput.*, vol. C-28, no. 6, pp. 384–394, June 1979.
[9] M. Kameyama, T. Hanyu, and T. Higuchi, "Design and implementation of quaternary NMOS integrated circuits for pipelined image processing," *IEEE J. Solid-State Circuits*, vol. SC-22, no. 1, pp. 20–27, Feb. 1987.
[10] K. C. Smith, "Multiple-valued logic: Tutorial and appreciation," *IEEE Computer*, vol. 21, no. 4, pp. 17–27, Apr. 1988.
[11] D. C. Rine, Ed., *Computer Science and Multiple-Valued Logic, Theory and Applications*, 2nd ed. Amsterdam: North-Holland, 1984.
[12] E. Charniak and D. McDermatt, *Introduction to Artificial Intelligence*. Reading, MA: Addison-Wesley, 1985.
[13] C. A. Mead and L. A. Conway, *Introduction to VLSI Systems*. Reading, MA: Addison-Wesley, 1980.
[14] A. Wittkower, P. H. Rose, and G. Ryding, "Advances in ion implantation production equipment," *Solid State Technol.*, vol. 18, no. 12, p. 41–45, Dec. 1975.

Takahiro Hanyu (S'86) was born in Hokkaido, Japan, on May 28, 1961. He received the B.E., M.E., and D.E. degrees in electronic engineering from Tohoku University, Sendai, Japan, in 1984, 1986, and 1989, respectively.

He is currently a Research Associate with the Department of Electronic Engineering, Tohoku University. His main interests and activities are in the multiple-valued logic system and its applications to artificial intelligence.

Dr. Hanyu is a member of the Institute of Electronics, Information and Communication Engineers of Japan, and the Information Processing Society of Japan. He received the Awards for Excellence at the 1985 and 1987 IEEE International Symposiums on Multiple-Valued Logic (with T. Higuchi *et al.*).



Tatsuo Higuchi (M'70–SM'83) was born in Sendai, Japan, on March 30, 1940. He received the B.E., M.E., and D.E. degrees in electronic engineering from Tohoku University, Sendai, Japan, in 1962, 1964, and 1969, respectively.

He is currently a Professor with the Department of Electronic Engineering, Tohoku University. His research interests include design of one-dimensional and two-dimensional finite word-length digital filters, multiple-valued logic systems, fault-tolerant computing, and VLSI computing structures for signal processing and image processing.

Dr. Higuchi is a member of the Institute of Electrical Engineers of Japan, the Institute of Electronics, Information and Communication Engineers of Japan, and the Society of Instrument and Control Engineers of Japan. He received the Awards for Excellence at the 1984, 1985, and 1987 IEEE International Symposiums on Multiple-Valued Logic (with M. Kameyama *et al.*), the Outstanding Transactions Paper Award from the Society of Instrument and Control Engineers of Japan in 1984 (with M. Kameyama), and the Technically Excellent Award from the Society of Instrument and Control Engineers of Japan in 1986 (with M. Kameyama *et al.*). He was the Program Chairman of the 1983 IEEE International Symposium on Multiple-Valued Logic, and the Chairman of the Japan Research Group on Multiple-Valued Logic from 1983 to 1985.