View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

DRAM-Cell-Based Multiple-Valued Logic-in-Memory VLSI with Charge Addition and Charge Storage

Takahiro Hanyu, Hiromitsu Kimura and Michitaka Kameyama
Graduate School of Information Sciences
Tohoku University
Aoba-yama 05, Sendai 980-8579, Japan
E-mail : kimura@kameyama.ecei.tohoku.ac.jp

Abstract

*multiple-valued logic-in-memory **VLSI** with fast re-programmability is proposed to realize transfer-bottle-neck-free VLSI systems. **A basic** component, in which a dynamic storage function and a multiple-valued thresh-old-literal function are **merged,** can be simply imple-**mented by charge addition and** charge storage with a **DRAM-cell-based** circuit structure. Any logic circuits with multiple-valued inputs **and binary** outputs can **be realized by** the combination of the **basic** components **and** logic-value conversion. **As a** typical example, a fully parallel magnitude comparator **between** three-valued in-put and **stored words is designed by** using the **proposed** logic-in-memory **VLSI** architecture. **Its** performance is superior to that of a corresponding binary implemen-tation **by** using **HSPICE** simulation under a 0.5 – μm CMOS technology.*

1. Introduction

Due to rapid technology scaling, technology generations in the deep submicron regime give us the capability to realize a giga-scaled system-on-a-chip, while accelerating the evolution of chip density causes a communication bottleneck between memories and logic modules [1],[2]. A logic-in-memory structure, where storage functions are distributed over a logic-circuit plane, is one of the key technologies to solve the above interconnection problems [3]. A circuit-level multiple-valued logic-in-memory VLSI architecture using floating-gate MOS transistors or ferroelectric devices has been proposed to realize highly parallel VLSI systems with a single-instruction multiple-data stream such as content-addressable memories [4]-[7]. However, it has been difficult to solve the trade-off between fast reprogrammability and non-volatility without special devices except MOS transistors.

In this paper, a new multiple-valued logic-in-memory VLSI is proposed to realize high-performance combinational circuits with multiple-valued external and stored inputs. One of the basic components in the proposed logic-in-memory VLSI is a ‘functional pass gate’ which is used to perform a one-digit multiple-valued threshold operation [8] and a pass-switch function together with a dynamic data storage. A DRAM-cell circuit structure [9],[10] is embedded in the functional pass gate so that multiple-valued data can be dynamically stored by charge. Since the addition of charge is realized by using capacitor coupling, a multi-level threshold operation between external input data and stored data can be performed by using the combination of a capacitor and a floating-gate MOS transistor whose threshold voltage is programmable.

As a result, the use of charge addition and charge storage due to a DRAM-cell circuit structure makes the functional pass gate compact. In addition to a threshold literal, three other basic operations, such as AND (series connection of functional pass gates), OR (parallel connection of functional pass gates), and logic-value conversion are used to design arbitrary switching functions.

As a typical example of the proposed logic-in-memory VLSI, an n-digit three-valued magnitude comparator is realized. The use of the functional pass gate makes it possible to perform threshold operations in word-parallel and digit-parallel schemes with $4(2n-1)$ transistors. The refresh operation for stored data has been performed in digit-parallel and word-serial schemes by using multi-level sense amplifiers, whose circuit structure is almost same as [8]. As a result, it is demonstrated by HSPICE simulation under a $0.5-\mu m$ CMOS technology that the performance of the proposed circuit is about 3.5-times faster than that of a corresponding binary implementation.

2. Model of a multiple-valued logic-in-memory VLSI

Figure 1 shows a general structure of a combinational logic circuit. It has two kinds of R-valued in-

0-7695-0692-5/00 $10.00 © 2000 IEEE

Authorized licensed use limited to: TOHOKU UNIVERSITY. Downloaded on February 11, 2010 at 20:14 from IEEE Xplore. Restrictions apply.

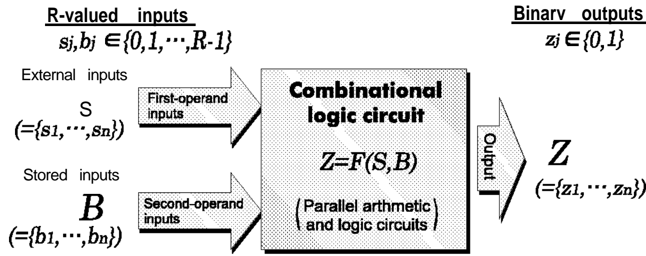

Fig. 1: Combinational logic-circuit model.

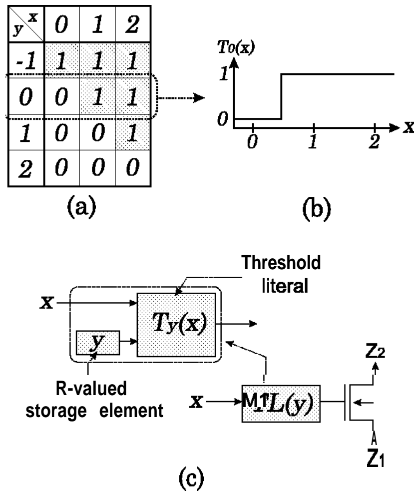

Fig. 2: Schematic of a functional pass gate. (a) Truth table, (b) Threshold literal $T_0(x)$, and (c) Block diagram of $T_y(x)$.

puts, S (n-digit external inputs) and B (n-digit internal (stored) inputs), and binary outputs, Z (m-bit externaloutputs), where $s_j, b_j \in \{0,1,\cdots,R-1\}(1 \leq j \leq n)$ and $z_i \in \{0,1\}(1 \leq i \leq m)$. In the following description, this type of a combinational logic circuit model is discussed.

2.1 Basic components

In the proposed multiple-valued logic-in-memory VLSI architecture, an R-valued-input binary-output logic function is represented by using four kinds of basic operations, AND, OR, a threshold literal and a logic-value conversion (LVC)

A threshold literal is a logic function with two variables x and y, which is defined as

$$T_y(x) = \begin{cases} 1 & \text{if } x > y \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where x $\in \{0,1,\cdots,R-1\}$ and y $\in \{-1,0,\cdots,R-1\}$. For example, Figure 2(a) shows the truth table of the three-valued threshold literal and Figure 2(b) shows $T_0(x)$. The transistor Ml is a pass gate as shown in

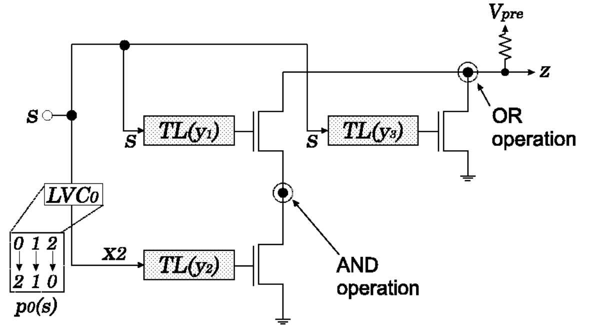

Fig. 3: General structure.

Figure 2(c), whose function is represented as

$$Z_2 = \begin{cases} Z1 & \text{if } T_y(x) = 1 \\ \phi & \text{otherwise} \end{cases} \quad (2)$$

where the symbol ϕ is a high impedance state of Ml.

An LVC is the permutation of a multiple-valued input, where an R-valued input is transformed into an arbitrary R-valued output. The LVC of the input s is represented as p(s) =< p_0, pi, , p_{R-1} > and its output x is defined as

$$x = p(s) \begin{cases} p_0 & \text{if } s = 0 \\ p_1 & \text{if } s = 1 \\ \vdots & \vdots \\ p_{R-1} & \text{if } s = R-1 \end{cases} \quad (3)$$

where $p_i \in \{-1,0,\cdots,R-1\}, x, s \in \{0,1,\cdots,R-1\}$. An arbitrary combinational logic circuit with two R-valued inputs s and b, and a binary output z is designed by using threshold literals, pass-switch functions and LVCs. Figure 3 shows a general structure of a logic circuit with two three-valued inputs and a binary output. In this structure, a stored input b is transformed into three inputs, y_1, y_2 and y_3, which are distributed over three storage elements in $TL(y_1)$, $TL(y_2)$ and $TL(y_3)$, respectively. An external input s is also transformed into x_2 by using $p_0(s)$. The three threshold operations between x_i and y_i are performed by $TL(y_i)$. Consequently, the output z is represented as

$$z = (T_{y_1}(s) \wedge T_{y_2}(x_2)) \vee T_{y_3}(s) \quad (4)$$

where $\wedge$ and $\vee$ indicate AND and OR operations, respectively. AND and OR operations are realized by using series and parallel connections of pass transistors, respectively.

Figure 4 shows a design example of logic circuit shown in Figure 3, where its specification is given by Figure 4 (a). When the storage input b is fixed to a logic value "0", the output z is represented as shown in Figure 4 (b), where three threshold literals T_{y_1} (xi),

0-7695-0692-5/00 $10.00 © 2000 IEEE

Authorized licensed use limited to: TOHOKU UNIVERSITY. Downloaded on February 11, 2010 at 20:14 from IEEE Xplore. Restrictions apply.

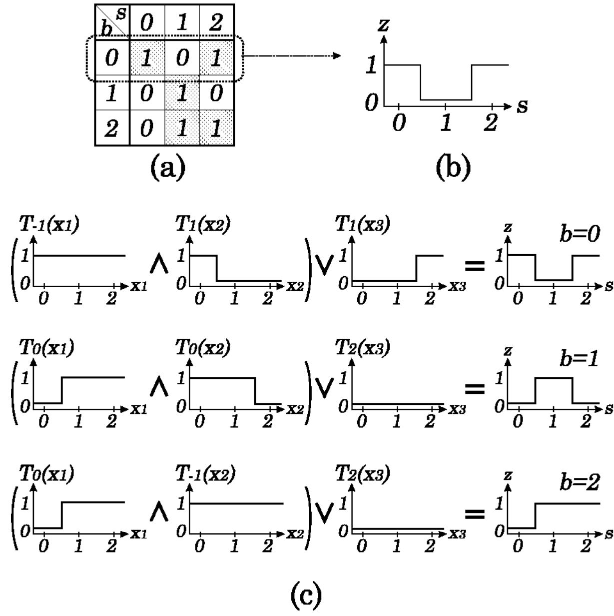

Fig. 4: Design example. (a) Specification, (b) Relationship between s and b when $b = 0$, and (c) Combination of threshold literals.

$T_{y_2}(x_2)$ and $T_{y_3}(x_3)$ are given as shown in Figure 4(c). According to the condition of b = 0, y_1, y_2 and y_3 are set to -1,1 and 1 by three LVCs, $p_1(b)$, $p_2(b)$ and $p_3(b)$. As with b = 0, y_1, y_2 and y_3 which correspond to b = 1 or b = 2 are determined by LVCs as shown in Figure 4(c). C oncequently, the conbinational logic circuit which realize Figure 4(a) is designed by three threshold literals and three LVCs which is defined as

$$y_1 = p_1(b) = \ < \ -1, 0, 0 >$$
$$y_2 = p_2(b) = \ < \ 1, 0, -1 >$$
$$y_3 = p_3(b) = \ < \ 1, 2, 2 \ >,$$

respectively.

2.2 Design of a functional pass gate

The threshold literal $T_0(x)$ whose threshold voltage is fixed to a logic value "0" is realized by using a single floating-gate MOS transistor. Since $T_y(x)$ in Eq.(1) can be rewritten as

$$T_y(x) = T_0(x \qquad y), \tag{5}$$

a functional pass gate can be designed by using four components, an R-valued storage function, the subtraction $(x - \mathrm{y})$, the threshold literal $T_0(x - \mathrm{y})$ and a pass-switch function as shown in Figure 5(a). An R-valued storage function and the subtraction $(x \quad \mathrm{y})$ is realized by charge storage and charge addition, respectively, with a DRAM-cell-based circuit structure. The

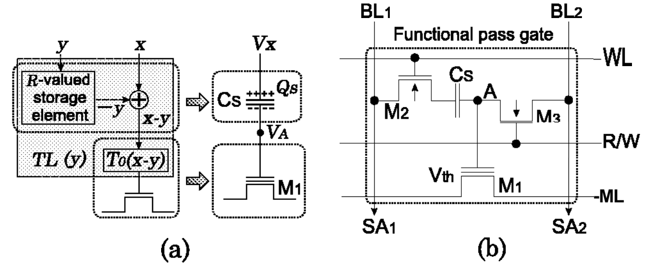

Fig. 5: Functional pass gate. (a) Block diagram, and (b) Circuit diagram.

threshold literal $T_0(x - y)$ can be merged into a pass-switch function by using a single floating-gate MOS transistor. Consequently, a functional pass gate is designed as shown in Figure 5(b). It consists of a floating-gate MOS transistor M_1, two access transistors, M_2 and M_3 and a capacitor C_S.

The transistor M_2 is turned ON by driving the word line WL after the bit line BL_1 is driven by V_x. Then, V_A is determined as

$$V_A = \frac{C_S}{C0 + C_S} V_x - \frac{Q_s}{C0 + C_S} \tag{6}$$

where C_0 is the stray capacitance between the substrate and the point A, and the voltage V_x corresponds to x. The electronic charge Q_s is represented as

$$Q_s = C_S \cdot V_y \tag{7}$$

where the voltage V_y corresponds to y. In the case of $C_0 << C_S$, Eq.(6) is approximately described as

$$V_A \simeq V_x - V_y. \tag{8}$$

Now, assume that V_x and V_y are represented as

$$V_x = V_{th} + (x + 0.5)\Delta V \tag{9}$$
$$V_y = (y + 1)\Delta V \tag{10}$$

where ΔV is a unit voltage, $x = \{0, 1, \ , \ R \quad 1\}$ and y = $\{-1, 0, \cdot\cdot, \ R \quad 1\}$. Then, V_A is given as

$$V_A \simeq V_{th} + (x \quad y \quad 0.5)\Delta V_y \tag{11}$$

by substituting Eqs.(9) and (10) into Eq.(8). Consequently, it is clear in Eq.(11) that the basic operations of a function pass gate can be performed by using the proposed circuit as shown in Figure 5(b).

It is obvious that V_A is always greater than or equal to GND because of eliminating the drain junction leakage at the point A. Therefore, V_x must be greater than or equal to V_y. That is, the relationship between V_{th} and ΔV must satisfy the following equation:

$$\frac{V_{th}}{R - O.5} \geq \Delta V_y \tag{12}$$

0-7695-0692-5/00 $10.00 © 2000 IEEE

Authorized licensed use limited to: TOHOKU UNIVERSITY. Downloaded on February 11, 2010 at 20:14 from IEEE Xplore. Restrictions apply.

Table 1: Relationship between logic values and voltage levels.

Logic value: x	0	1	2	
Input voltage: V_x	2.4V	3.2V	4.0V	
Logic value: y	-1	0	1	2
Stored voltage: V_y	0.0V	0.8V	1.6V	2.4V
$V_{th}+V_y$ (V_{th}=2.0V)	2.0V	2.8V	3.6V	4.4V

For example, V_x and V_y are given by Table 1 in the case of $V_{th} = 2.0V$. The initial threshold voltage V_{th} can be established by using an ion implantation technique[ll].

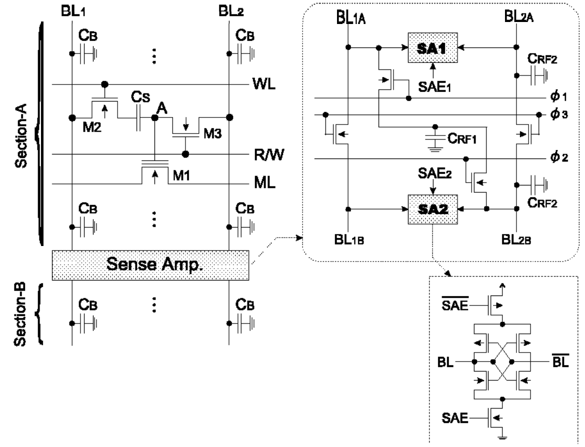

Fig. 6: Multi-level sense amplifier.

3. Read/write operations in the refresh cycle

It is important to realize read/write operations as the refresh cycle in the presented DRAM-cell-based hardware. In the following subsection, the basic behavior of the read/write operations and its HSPICE simulation are discussed.

3.1 Basic behavior

A three-valued stored data in each functional pass gate is updated in the refresh cycle, where a four-level sensing and a restore operation are required as read/write operations. Figure 6 shows the four-level sense amplifier which consists of two sense amplifier, SA1 and SA2, and three reference capacitor, C_{RF1} and two C_{RF2}. BL_1 is divided into BL_{1A} and BL_{1B} which correspond to BL_1 of Section-A and Section-B, respectively. Similarly, BL_2 is also divided into BL_{2A} and BL_{2B}.

The read operation includes four steps, the pre-charge scheme, the data read scheme and two sensing schemes as shown Figure 7. In the precharge scheme, ϕ_3 is set high, and BL_1 and BL_2 are precharged to the half level of maximum-&, V_{hf}. When the voltage level of V_y is given by Table 1, V_{hf} becomes $1.2V$. In the data read scheme, WL is selected and M_2 is turned ON. When the charge stored on C_S is transferred to BL_1, the voltage on BL_1 is represented as

$$V_{BL_{1A}} = V_{BL_{1B}} = V_{hf} + C\Delta V(y - 1/2) \qquad (13)$$
$$\left(C = \frac{C_S}{3C_B + C_S} \right)$$

where y $\in \{-1, 0, 1, 2\}$, C_B is the parasitic capacitance on the bit line and $V_{BL_{1A}}$ and $V_{BL_{2A}}$ are the voltage on BL_{1A} and BL_{2A}, respectively. Then, ϕ_3 is set to GND and BL_{1A} and BL_{1B} are isolated. In the first sensing scheme, SA1 is activated, and $V_{BL_{1A}}$ is sensed using

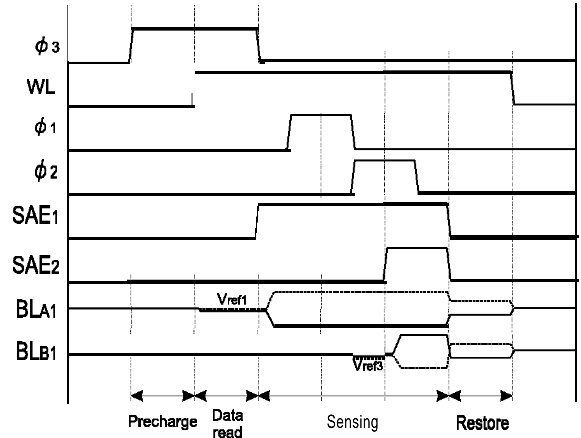

Fig. 7: Timing diagram of a read operation.

V_{hf}. Then, ϕ_1 is set to high and the capacitor C_{RF1} is connected to BL_{1A}. C_{RF1} is charged to maximum-& or GND depending on the voltage level $V_{BL_{A1}}$, where C_{RF1} and C_{RF2} are represented as

$$C_{RF1} = 2/9 \cdot C_S, \quad \text{and} \quad C_{RF2} = 1/9\ C_S, \qquad (14)$$

respectively.

In the second sensing scheme, ϕ_1 is set to GND and ϕ_2 is set to high. The charge on C_{RF1} generates the second reference voltage, V_{REF2} or V_{REF3}, on BL_{2B} which is represented as

$$\begin{aligned} V_{REFi} &= V_{hf} + \frac{2/9C_S}{C_D + 1/3C_S}(V_{BL_{1A}} \quad V_{hf}) \qquad (15) \\ &= V_{hf} \pm C\Delta V. \end{aligned}$$

Figure 8 shows the relationship between the voltage on BL_1 corresponding to the stored charge on C_S and each reference voltage V_{REFi}. Then, SA2 is activated and $V_{BL_{1B}}$ is sensed using V_{REF2} or V_{REF3} and $V_{BL_{1A}}$

0-7695-0692-5/00 $10.00 © 2000 IEEE

Authorized licensed use limited to: TOHOKU UNIVERSITY. Downloaded on February 11, 2010 at 20:14 from IEEE Xplore. Restrictions apply.

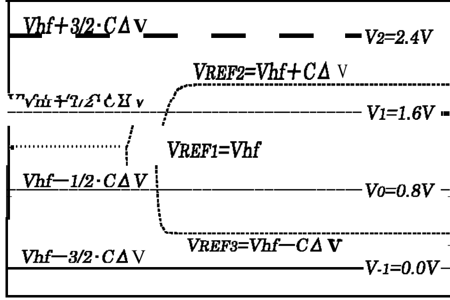

Fig. 8: Relationship between the voltage on BL_1 after the data read scheme and the reference voltage V_{REFi}.

Table 2: Stored data transition after sensing.

Data	V_y	V_{BL1A} after sensing	V_{BL1B} after sensing
-1	0	GND	GND
0	ΔV	GND	$3\Delta V$
1	$2\Delta V$	$3\Delta V$	GND
2	$3\Delta V$	$3\Delta V$	$3\Delta V$

$\Delta V = 0.80V$

becomes maximum-& or GND depending on the stored data. Consequently, the four-level stored charge on C_B is sensed by using two sense amplifier, SA1 and SA2 with time-sharing sensing scheme. The restore operation is performed by using the charge-sharing restore scheme[ll]. $V_{BL_{1A}}$ and $V_{BL_{1B}}$ after sensing scheme is represented as Table 2 depending on the stored data. The capacitance ratio between Section-A and Section-B is two. Therefore, the charge on the BL_1 of Section-A and Section-B are combined after ϕ_3 is set high, and the restore level is generated, where the restore level is represented as

$$V_{BL} = V_{BL_{1B}} + \frac{2}{3}(V_{BL_{1A}} - V_{BL_{1B}}). \qquad (16)$$

3.2 HSPICE simulation

Figure 9 shows the simulation result of the threshold operation and read/write operations when x = 1 and y = 0. In the threshold operation, $3.2V$ corresponding to x = 1 is supplied to BL_1. In this situation, floating-gate-MOS transistor is turned ON and ML is discharged because x is greater than y. The stored voltage on C_S is retained during the threshold operation. In the read/write operation, the stored voltage is read out to the BL_1 of Section-A and Section-B. Then, SA1 is activated, and $V_{BL_{1A}}$ falls to GND. After first sensing scheme, the reference capacitor C_{REF1} is charged by $V_{BL_{1A}}$, and the reference level V_{REF3} for SA2 is generated on $V_{BL_{2B}}$. Simultaneously, SA2 is activated,

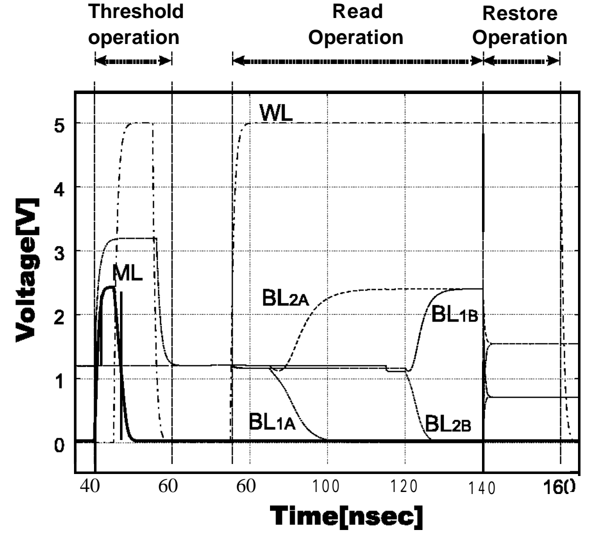

Fig. 9: Simulation result.

and $V_{BL_{1B}}$ rises up to maximum-V_y Finally, BL_{1A} and BL_{1B} are connected, and the restore level ΔV, 0.80V is generated on BL_1.

4. Design example

Figure 10 shows an overall structure of a magnitude comparator with two kinds of three-valued input words, S(an n-digit external input word) and B_i(an ith n-digit stored input word), and generates a binary output word, Z(m-bit output) as its comparison result. In the case of three-valued encoding, an input word S and the ith stored word B_i are expressed as

$$S = \sum_{j=1}^{n} 3^{n-j} \cdot s_j, \qquad (17)$$

$$B_i = \sum_{j=1}^{n} 3^{n-j} \cdot b_{ij} \qquad (18)$$

where s_j and $b_{ij} (1 \leq j \leq n)$ indicate the jth digits of S and B_i, respectively, and s_j, $b_{ij} \in \{0, 1, 2\}$. s_1 and b_{i1} are the most significant digits, and s_n and b_{in} are the least significant digits, respectively. The magnitude comparison between S and B_i is defined as

$$G(S, B_i) = \begin{cases} 1 & \text{if } S > B_i \\ 0 & \text{otherwise.} \end{cases} \qquad (19)$$

The ith binary output $z_i \in Z$ is equal to $G(S, B_i)$

In the following subsection, a hardware algorithm of the magnitude comparison, and its circuit design are discussed.

0-7695-0692-5/00 $10.00 © 2000 IEEE

Authorized licensed use limited to: TOHOKU UNIVERSITY. Downloaded on February 11, 2010 at 20:14 from IEEE Xplore. Restrictions apply.

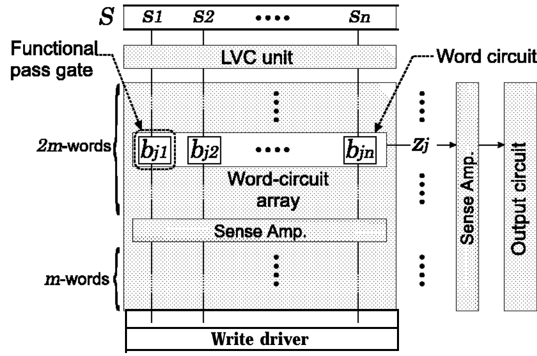

Fig. 10: Overall structure of the three-valued magnitude comparator.

$g_j(s_j, b_{ij})$

b_{ij} \ s_j	0	1	2
0	0	1	1
1	0	0	1
2	0	0	0

$ge_j(s_j, b_{ij})$

b_{ij} \ s_j	0	1	2
0	1	1	1
1	0	1	1
2	0	0	1

Fig. 11: Truth tables of $g_j(s_j, b_{ij})$ and $ge_j(s_j, b_{ij})$

4.1 Hardware algorithm

The magnitude comparison $G(S, B_i)$ is represented by two kinds of threshold operations for each digit. One is the greater-than search operation, g_j, and the other is the greater-than or equal-to search operation, ge_j, between s_j and b_{ij}. These operations are defined as

$$g_j(s_j, b_{ij}) = \begin{cases} 1 & \text{if } s_j > b_{ij} \\ 0 & \text{otherwise} \end{cases} \quad (20)$$

$$ge_j(s_j, b_{ij}) = \begin{cases} 1 & \text{if } s_j \geq b_{ij} \\ 0 & \text{otherwise.} \end{cases} \quad (21)$$

For example, let's consider the magnitude comparison between three-valued three-digit words, S and B as

$$S = (s_1 s_2 s_3) = (211)$$
$$B = (b_1 b_2 b_3) = (210).$$

If B is greater than S, then one of the following conditions:

$$(a) \quad g_1 = 1$$
$$(b) \quad ge_1 \wedge g_2 = 1$$
$$(c) \quad ge_1 \wedge ge_2 \wedge g_3 = 1$$

must be at least satisfied. In this example, the above condition (c) is satisfied because of $(s_1 = b_1)$, $(s_2 = b_2)$ and $(s_3 < b_3)$.

In general, the magnitude comparison between n-digit words can be represented by using g_j and ge_j as

$$G(S, B_i) = g_1 \vee (ge_1 \wedge g_2) \vee (ge_1 \wedge ge_2 \wedge g_3)$$

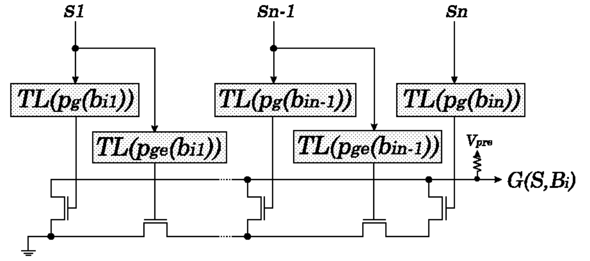

Fig. 12: Word circuit for an n-digit magnitude comparison

$$\vee (ge_1 \wedge ge_2 \wedge \cdots \wedge ge_{n-1} \wedge g_n). \quad (22)$$

where symbol $\vee$ and $\wedge$ indicate binary logic operations, OR and AND, respectively. Eq.(24) is also transformed into

$$G(S, B_i) = g_1 \vee ge_1 \wedge (g_2 \vee ge_2 \wedge (\cdot \cdot \cdot (g_{n-1} \vee ge_{n-1} \vee g_n) \cdot \cdot \cdot)). \quad (23)$$

Therefore, $G(S, B_i)$ can be designed by using four operations, g_j, ge_j, AND and OR.

4.2 Design of an n-digit magnitude comparator

An n-digit magnitude comparator given by Eq.(23) can be designed by using the proposed logic-in-memory VLAI architecture. The truth tables of $g_j(s_j, b_{ij})$ and $ge_j(s_j, b_{ij})$ are represented by Figure 11. The function of $g_j(s_j, b_{ij})$ is realized by using functional pass gates and LVCs which are defined as

$$x = p_g(s_j) = <0, 1, 2> \quad (24)$$
$$y = p_g(b_{ij}) = <0, 1, 2> \quad (25)$$

where $p_g(s)$ and $p_g(b)$ are LVCs for s_j and b_{ij}, respectively. Similarly, LVCs corresponding to $ge_j(s_j, b_{ij})$ are defined as

$$x = p_{ge}(s_j) = <0, 1, 2> \quad (26)$$
$$y = p_{ge}(b_{ij}) = <-1, 0, 1>, \quad (27)$$

where $p_{ge}(s)$ and $p_{ge}(b)$ are LVCs for s_j and b_{ij}, respectively. Therefore, one LVC is required for the one-digit magnitude comparator. Two binary logic operations, AND and OR, can be easily realized by parallel and series connections of functional pass gates. Consequently, a word circuit for the n-digit magnitude comparison circuit is shown in Figure 12, and one-digit magnitude comparator consists of two functional pass gates which correspond to g_j and ge_j, respectively. By the use of the proposed circuit, a magnitude comparison between an input word S and stored word B_i is performed in a word-parallel and a digit-parallel fashions without depending on the number of words.

0-7695-0692-5/00 $10.00 © 2000 IEEE

Authorized licensed use limited to: TOHOKU UNIVERSITY. Downloaded on February 11, 2010 at 20:14 from IEEE Xplore. Restrictions apply.

Table 3: Comparison of S-bit magnitude comparators

	Binary CAM-based	Proposed Digit-serial	Proposed Digit-parallel
Processing time /step	8nsec	6nsec	18nsec
Transistors counts /word	80	20	44
No. of Execution steps	8	6	1
Execution time	64nsec	36nsec	18nsec

4.3 Evaluation of a S-bit magnitude comparator.

To evaluate the performance of the proposed multiple-valued login-in-memory VLSI, we compare the proposed three-valued magnitude comparator with those using a typical binary CAM[12]. Table 3 summarizes the comparison of these magnitude comparator under a 0.5 – um CMOS technology. In the proposed magnitude comparator, a magnitude comparison is executed in digit-parallel and word-parallel fashions, so that, the number of its execution step becomes only one, which is independent of its word length. The use of a digit-parallel comparison scheme together with a new circuit technology makes it possible to design a high-speed magnitude comparator with less transistor counts. In fact, the execution speed of the proposed magnitude comparator is about 3.5-times faster than those of a binary CAM-based implementation and digit-parallel implement ation.

5. Conclusion

A multiple-valued logic-in-memory VLSI, with fast reprogrammability has been proposed by using DRAM-cell-based threshold-literal circuits. By the use of linear summation between input and stored values, a two-variable threshold operation is attributed to a one-variable multiple-valued one, which can be realized by using a single floating-gate MOS transistor. Moreover, the voltage-mode linear summation can be realized by a single capacitor due to a capacitor-coupling technique, which makes the proposed threshold-literal circuit compact. As a typical example, a fully parallel magnitude comparator has also designed, and its execution time and transistor counts have been reduced to 55 percent and 28 percent, respectively, in comparison with those corresponding to the binary implementation.

As a future prospect, it is also important to utilize the logic-in-memory VLSI architecture with fast reprogrammability in the application to fine-grain fully parallel VLSI processors such as stereo-vision VLSI processors.

References

[1] S.Katkoori and P.Maurer, "Challenges for CAD in Deep Sub Micron Regime," *IEEE Computer Society TCVLSI Technical Bulletin,* pp.7-10, 1998.

[2] H.Iwai, "CMOS Technology - Year 2010 and Beyond," *IEEE J.Solid-State Circuits,* SC-34, No.3, pp357-366, Mar. 1999.

[3] W.H.Kautz, "Cellular Logic-in-Memory Arrays," *IEEE Trans. Computers,* Vol. C-18, No.8, pp719-727, Aug. 1969.

[4] T.Hanyu, N.Kanagawa and M.Kameyama, "Design of a One-Transistor-Cell Multiple-Valued CAM," *IEEE J.Solid-State Circuits,* SC-31, No.11, pp.1669-1674, Nov. 1996.

[5] A.Sheikholeslami, P.G.Gulak and T.Hanyu, "A Multiple-Valued Ferroelectric Content-Addressable Memory", *Proc. IEEE International Symposium on Multiple- Valued Logic,* pp.74-79, 1996.

[6] T.Hanyu, K.Teranishi and M.Kameyama, "Multiple-Valued Floating-Gate-MOS Pass Logic and Its Application to Logic-in-Memory VLSI", *Proc. IEEE International Symposium on Multiple- Valued Logic,* pp.270-275, 1998.

[7] T.Hanyu , H.Kimura and M.Kameyama , "Multiple-Valued Content-Addressable Memory Using Melal-Ferroelectric-Semiconductor FETs," *Proc. IEEE International Symposium on Multiple- Valued Logic,* pp.30-35, 1999.

[8] T.Higuchi and M.Kameyama, *Multiple- Valued Digital Processing System,* Shokodo Co. Ltd., Tokyo, 1989(in Japanese).

[9] T.Hanyu and M.Kameyama , "Multiple-Valued Logic-in-Memory VLSI Architecture Based on Floating-Gate-MOS Pass-Transistor Logic," *IEICE Trans. Electron,* Vol.E82-C, No.9, pp.1662-1668, Sep. 1999.

[10] T.Oda and T.Murotani, "A Four-Level Storage 4-Gb DRAM," *IEEE J.Solid-State Circuits,* SC-32, No.11, pp.1743-1747, Nov. 1999.

[11] D.A.Rich, K.C.Naiff and K.G.Smalley, "A Four-State ROM Using Multilevel Process Technology," *IEEE J.Solid-State Circuits,* SC-19, pp.174-179, Apr. 1984.

[12] T.Yamagata, et al., "A 288-kb Fully Parallel Content Addressable Memory Using a Stacked-Capacitor Cell Structure," *IEEE J.Solid-State Circuits,* SC-27, no.12, pp.1927-1933. Dec. 1992.

0-7695-0692-5/00 $10.00 © 2000 IEEE

Authorized licensed use limited to: TOHOKU UNIVERSITY. Downloaded on February 11, 2010 at 20:14 from IEEE Xplore. Restrictions apply.