Race Logic Architecture (RALA): A Novel Logic Concept Using the Race Scheme of Input Variables

Se-Joong Lee and Hoi-Jun Yoo

***Abstract*—A novel logic concept, Race Logic Architecture (RALA), is proposed. RALA is a new logic operation architecture in that the racing between input variables along the interconnection lines functions as an active logic element instead of logic gates, while the logic gates play a simple passive role.**

Logic operations of RALA are based on wired-OR that utilizes shared space and serial-AND that utilizes the triggering sequence of input variables. With these two concepts, RALA can implement arbitrary Boolean operations. Various kinds of combinational circuits are simulated and compared with RALA's. RALA shows the best performance in delay time, area, and power product results. A 64-bit carry–look-ahead adder with RALA is fabricated by 0.25-μm CMOS technology to verify its feasibility and functionality. The area of the adder is 800 μm $\times$ 150 μm, and the delay time from the clock to Sum31 measured 0.9 ns.

***Index Terms*—High-speed integrated circuit, logic design, logic family, race logic, RALA.**

I. Introduction

EVEN IN the age of deep submicron technology, the demand for faster logic circuits has not been tempered, and numerous kinds of high-speed logic families and techniques are constantly being reported. Some of them, such as differential cascode voltage switch logic [1] and sample-set differential logic [2], improve the operation speed by inserting feedback pMOSs or sense amplifiers. Pass gate techniques, wave pipelining, and self-timing are also frequently used for fast logic circuits [3]–[5]. However, these logic families and techniques have fundamental limitations in speed because their logical functions rely entirely on the level-based logic computation. Some other logic families based on analog computation, such as Josephson logic [6], current steering logic [7], and the current-mode circuit [8], have been reported. However, Josephson logic requires integrated inductors that are inadequate for monolithic integrated circuits, current steering logic has been developed for low switching noise rather than high-speed operation, and the current mode circuit technique is hard to integrate with conventional digital logic families.

Moreover, these conventional logic techniques reduce only the gate delay time, while the interconnection delay caused by the coupling capacitance between metal routings causes a more serious delay as fabrication technology develops. In other words, the time during which the signals run along the wires is wasted. As long as transistor logic gates perform logic operations, the delay time caused by interconnection wires is an extra expenditure out of an already limited time budget.

In the proposed new logic architecture, Race Logic Architecture (RALA) abandons transistor logic gates for logic operation. RALA utilizes the racing between input variables to perform logic operations. Sequentially triggered input signals race with each other, and by detecting the winning signal of the race, logic operation can be realized. When a circuit is implemented on the basis of RALA, it can not only overcome the limitations of the transistor delay, but also utilize the interconnection delay to obtain logic operation [9].

In Section II, the concept of RALA is presented. In Section III, a CMOS circuit implementation of RALA is suggested and discussed. The comparisons with the circuits designed by other various logic families follow in Section IV. In Section V, CMOS adder implementation using RALA is shown, and finally, Section VI concludes the paper.

II. Operation of RALA

A. Structure

The conceptual diagram of RALA is illustrated in Fig. 1(a). RALA consists of three parts: the clock distribution line (CDL), the race lines, and the winner-take-all circuit (WTAC). When the clock transits from low to high, the clock travels along the resistor array, and switches (1), (2), (3), and (4) are triggered sequentially. The termination switch, switch (4), is attached at the end of the CDL. The race lines consist of a true-line and a false-line. The race lines perform wired-OR operations. The WTAC determines which line becomes 1 earlier. If the true-line becomes 1 earlier, the OUT becomes 1. On the other hand, if the false-line becomes 1 earlier, the OUT becomes 0. Even in the case of all-0 inputs, one of race lines, e.g., the true-line in this example, is set to 1 by the termination switch. The input of the termination switch is 1, so that there is no situation in which both race lines stay 0. Logic values of each node are depicted in Fig. 1(b) in the case of $A = 1$, $B = 0$, and $C = 0$.

There can be various methods of circuit implementation of RALA. One circuit implementation is shown as an example in Fig. 1(c). To implement the switches by nMOS, negative logic is adopted. During the low clock period, the race lines and OUT/$\overline{\text{OUT}}$ are precharged. If an input variable is 1, the relevant switch is turned on when the clock triggers the switch, and then the race line is discharged. The WTAC only allows the signal that falls faster to pass. The remaining signal that arrives later is blocked by the WTAC. A detailed explanation of the circuit will be shown in Section IV.

Manuscript received November 28, 2000; revised October 16, 2001.

The authors are with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon 305-702, Korea (e-mail: shocktop@eeinfo.kaist.ac.kr).

Publisher Item Identifier S 0018-9200(02)00670-4.

0018–9200/02$17.00 © 2002 IEEE

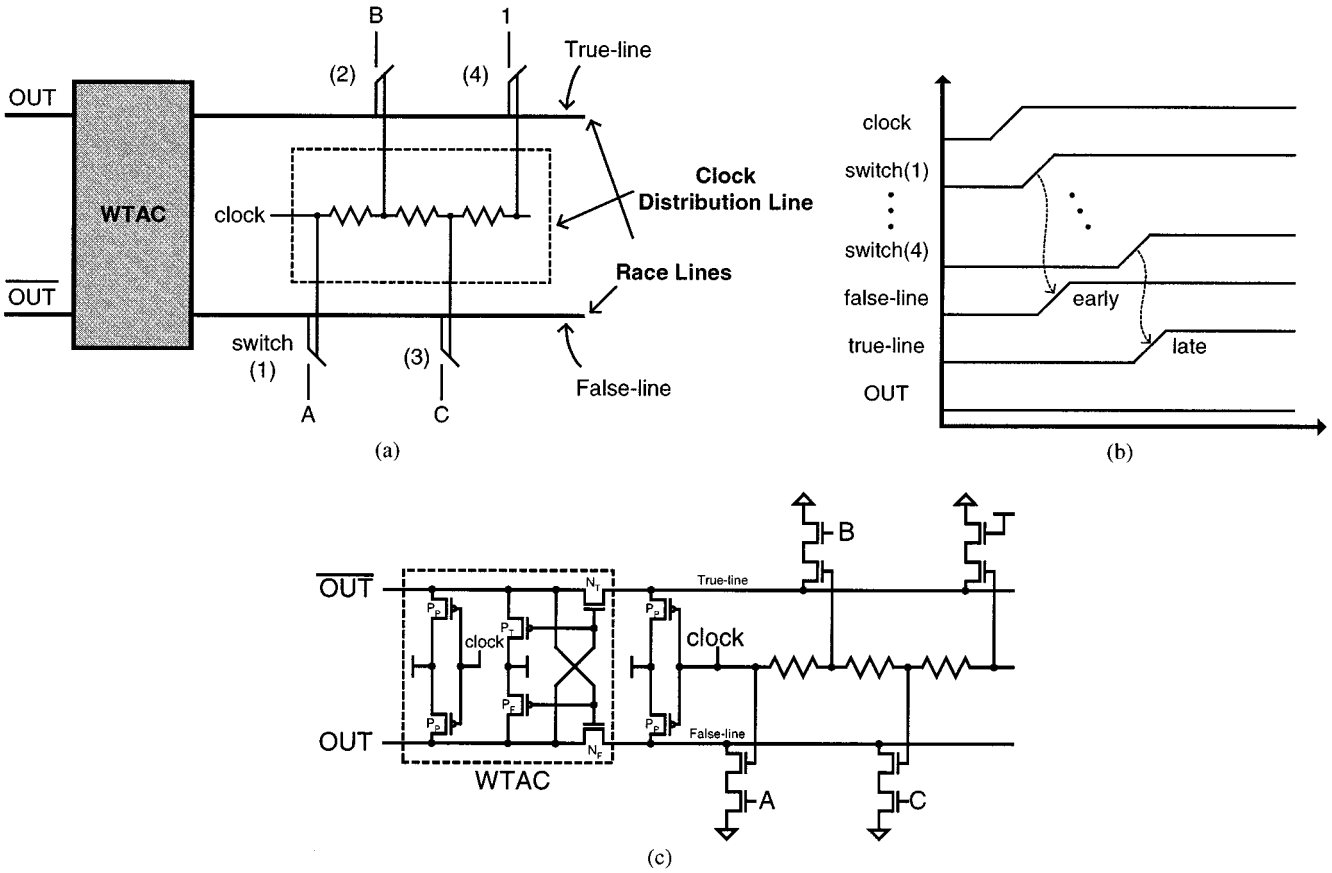


Fig. 1. (a) Race logic architecture. (b) Logical operation of RALA. (c) Example of circuit implementation of RALA.

B. Basic Boolean Operations

Fig. 2(a) shows the AND operation. When the clock goes high, the input $\overline{A}$ is triggered first. If $\overline{A}$ is 1, the output becomes 0. If $\overline{A}$ is 0, OUT depends on the value of input B. If input B is 1, the OUT becomes 1, and if it is 0, the OUT becomes 0 because of the termination that is attached to the false-line. As shown in the truth table of the Fig. 2(a), the race logic performs the AND operation with the depicted circuit configuration. Fig. 2(b) shows another implementation of the AND operation. The OR operation can be implemented with circuit configurations shown in Fig. 3. Implementation of the OR operation is quite similar to that of the AND operation except for the positions of the input variables.

From Figs. 2 and 3, the connection of the first input variable, say A, to the false-line gives an AND operation between the first variable and the following input variable. An OR operation is obtained with the connection of the first input variable to the true-line, regardless of the position of the second input variable. This characteristic is useful when figuring out the functionality of the cascaded multiple race logics.

C. Cascading Race Logics

The race logics can be cascaded to implement more complicated functions. A simple cascading of two race logics is shown in Fig. 4. The logic operations of the left gray block and the right one are AND operations, respectively. The logical relationship between input B and C is an OR operation because input B is connected to the true-line. When two race logics are cascaded, the Boolean function of the cascaded race logic is expected to be (1). However, during cascading the two race logics, an associative operation is created, and the resulting Boolean function is given as (2) instead.

$$\text{OUT} = A \cdot B + C \cdot D \tag{1}$$

$$\text{OUT} = A \cdot (B + C \cdot D). \tag{2}$$

By inserting a WTAC between two race logics, cascading can be performed without an associative operation. Fig. 5(a) and (b) shows the cascading for the case of AND and OR operations, respectively. In the AND operation, the OUT and $\overline{\text{OUT}}$ of the first WTAC are connected to the CDL and the false-line of the second race logic, respectively. In contrast, in the OR operation, OUT and $\overline{\text{OUT}}$ of the first WTAC are connected to the true-line and the CDL, respectively.

D. Design Algorithm of Race Logics

Systematic methodology can be developed for the efficient design of the race logics. For convenience, several symbols are defined as in Table I. T_I and F_I can be expressed as the following equations.

$$T_I = \{a_{n0}, a_{n1}, \ldots, a_{nk}\} \tag{3}$$

$$F_I = \{b_{m0}, b_{m1}, \ldots, b_{ml}\} \tag{4}$$

where the ni and mj are the triggering sequences.

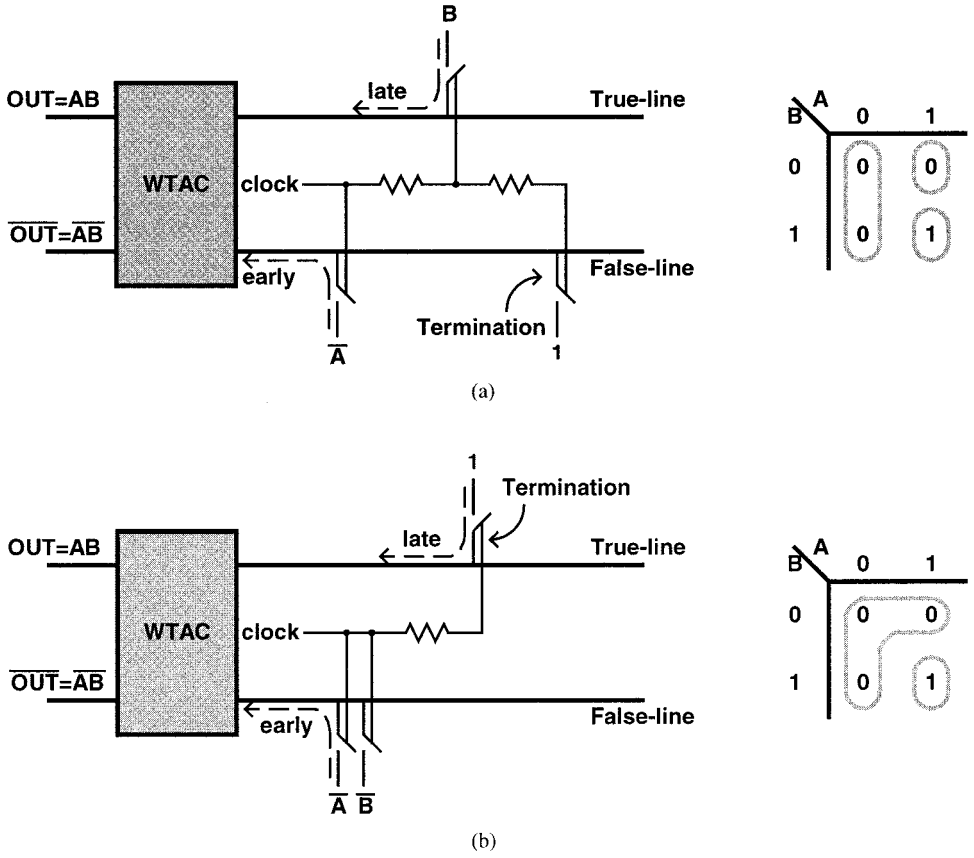


Fig. 2. Two implementations of AND operation by RALA.

There are six design rules for the synthesis of race logic.

Rule 1: The b_{mj} must be negative logic for $0 \le j \le l$.

Rule 2: The S must be a set of integers from 0 to $\mathrm{Max}[nk, ml]$.

Rule 3: To create an AND operator with an associative operation between input x and y:

i) $x \in F_S$
ii) If $y \in T_S$, $\mathrm{SE}(y) = \mathrm{SE}(x) + 1$. If $y \in F_S$, $\mathrm{SE}(y) = \mathrm{SE}(x)$.

Rule 4: To create an OR operator between input x and y:

i) $x \in T_S$
ii) If $y \in F_S$, $\mathrm{SE}(y) = \mathrm{SE}(x) + 1$. If $y \in T_S$, $\mathrm{SE}(y) = \mathrm{SE}(x)$.

Rule 5: To create an AND operator without an associative operation between input x and y:

i) x and y must be in distinct race logic, R_1 and R_2.
ii) W_1 of R_1 must be $W_1(T_{L1}, F_{L1}, \mathrm{CDL}_2, F_{L2})$.

Rule 6: To create an OR operator without an associative operation between input x and y:

i) x and y must be in distinct race logic, R_1 and R_2.
ii) W_1 of R_1 must be $W_1(T_{L1}, F_{L1}, T_{L2}, \mathrm{CDL}_2)$.

To explain the synthesis method of the Boolean function given in (5), an example has been provided. For convenience, the above six rules are summarized in Table II. Methods M-AND1 and M-AND2 create an AND operator with an associative operation, methods M-OR1 and M-OR2 create an OR operator with an associative operation. Methods M-AND3 and M-OR3 create an AND and an OR operator without an associative operation, respectively.

$$\mathrm{OUT} = A \cdot (B + C + \overline{D} \cdot E) + F \cdot G(H + I). \quad (5)$$

The relationship between input A and B is AND with an associative operation; therefore, method M-AND2 in Table II is applied. Method M-OR1 is applied to B and C for an OR operation. When we relate A and B, method M-AND2 is used instead of M-AND1 because input B should be placed in the true-line for the next OR operation. Method M-OR2 is proper for input C and D. Method M-AND1 is used for D and E. Actually, method M-AND1 creates the associative operation. However, the created associative operation has no effect because the next operation will terminate the association. When placing input F after E, the previous associative operation is to be closed and a new OR operation is to be created. Therefore, we apply method M-OR3 when relating input F and E. Inputs G, H, and I are placed by methods M-AND1, M-AND2, and M-OR1, respectively. Finally, the WTAC is placed at the ends of the true-line and the false-line.

When the input vector is given by $\{A, B, C, D, E, F, G, H, I\} = \{1, 0, 0, 1, 1, 1, 1, 0\}$, the operation of the race logic shown in Fig. 6 is as follows. When the clock goes high, input $\overline{A}$, B, C, D, and E are sequentially triggered. When the clock triggers $\overline{A}$, B, and C, true-line1 and false-line1 stay low because all of them are 0. When the clock triggers D, false-line1 is activated, and WTAC1 detects the activation.

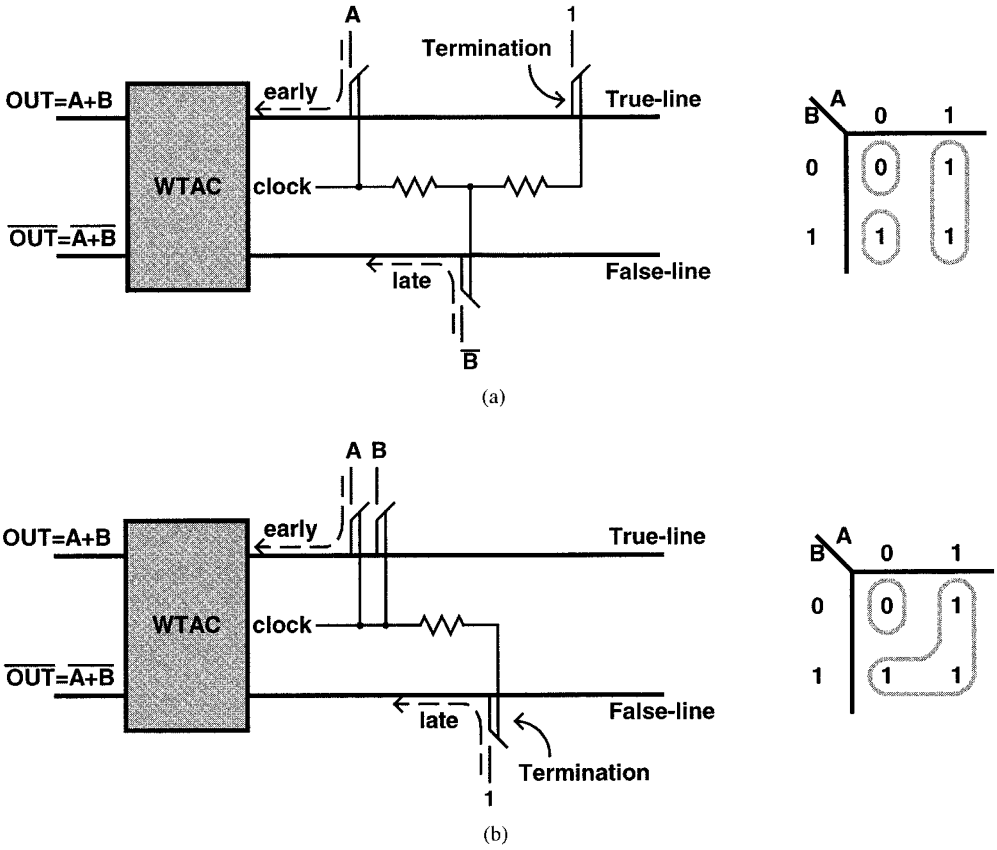


Fig. 3. Two implementations of OR operation by RALA.

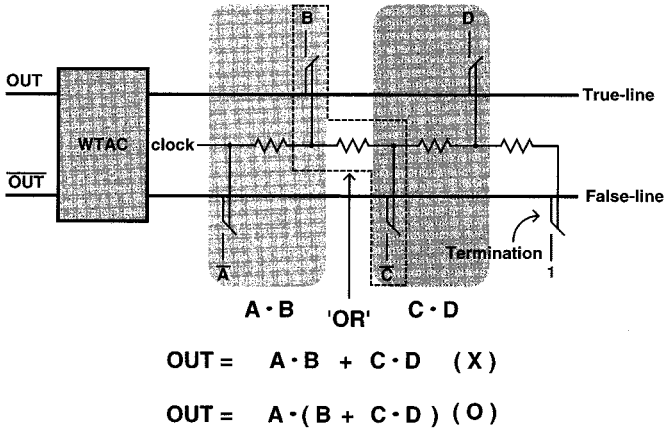


Fig. 4. Direct cascading of two race logics.

Since false-line1 is activated earlier than true-line1, the $\overline{\text{OUT}}$ of WTAC1 becomes high. This operation proceeds regardless of the value of input E. As shown in Fig. 6, the CDL2 is connected to the $\overline{\text{OUT}}$. Therefore, the race logic2 is enabled. If the CDL2 is connected to the OUT, the race logic2 may not be activated. Except for the H, the $\overline{F}$ and $\overline{G}$ are all 0, so that true-line2 is activated earlier than false-line2. Therefore, WTAC2 detects true-line2 and the OUT of WTAC2 becomes 1. Finally, the result of the circuit becomes 1, the same value as that of (5) for the given input vector.

III. CMOS Circuit Implementation

RALA can be realized by various methods, and in this study, a CMOS implementation is exemplified to assert the feasibility of RALA. As shown in Fig. 7(a), the CDL consists of resistors and nMOS gate capacitors. The value of gate capacitance is chosen to be 2 fF, and the value of resistance is determined to satisfy the required timing gap, in this study, 10 ps, between the two neighboring tapped clock signals. The operation of the WTAC is as follows. True-line, false-line, OUT, and $\overline{\text{OUT}}$ are precharged to V_{dd} when the clock is low. During the precharge period, N_T and N_F are turned off, and P_T and P_F are also turned off. If the voltage of the true-line falls down by V_{THN}, N_T is turned on, and $\overline{\text{OUT}}$ starts to fall. N_F, whose gate is connected to $\overline{\text{OUT}}$, is turned off, and P_F is turned on. Therefore, OUT is isolated from the false-line and stays high with the help of P_F.

The simulation results with 0.18-μm CMOS model parameters are shown in Fig. 7(b). Among the inputs, only E is set to 1. Once the clock goes high, the clock signal travels along the CDL, and after 30 ps, the nMOS switch of input E receives the clock signal. The termination switch receives the clock signal after 40 ps. The WTAC detects this 10-ps difference to generate OUT $= 1$ and $\overline{\text{OUT}}$ $=0$. As shown in Fig. 7(b), the WTAC detects the timing difference successfully.

In order to estimate the speed limitations of the RALA, Monte Carlo simulation is performed, taking the process

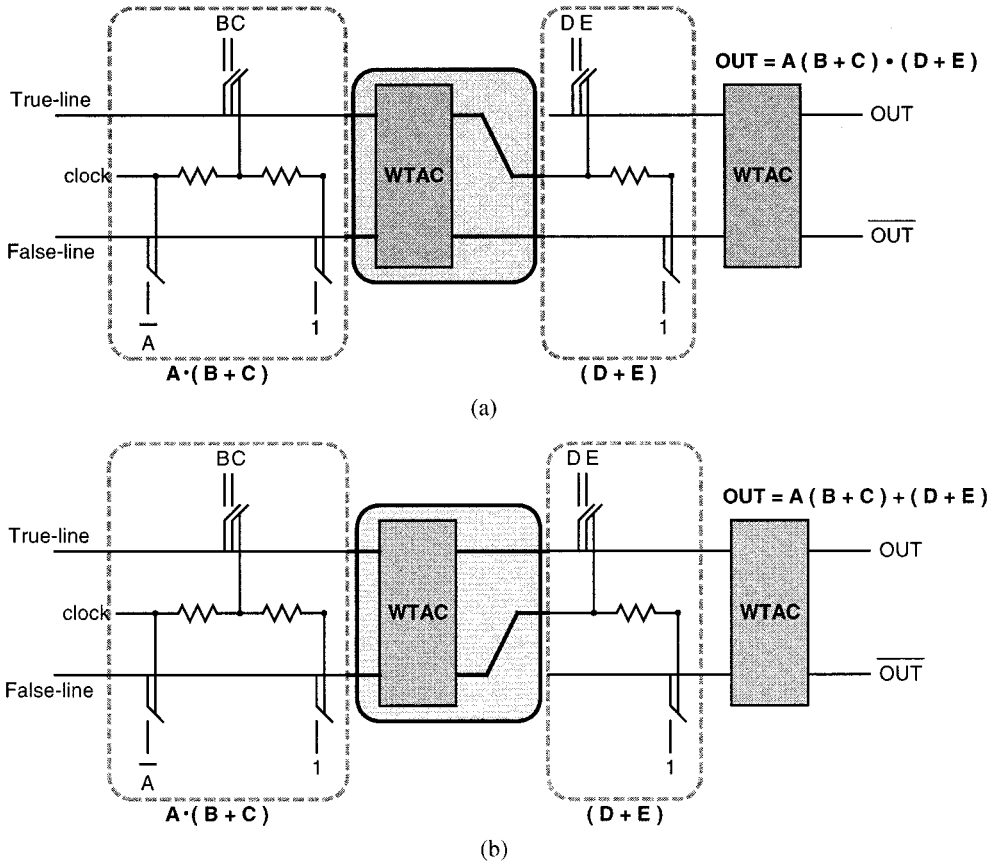


Fig. 5. (a) AND cascading without an associative operation. (b) OR cascading without an associative operation.

TABLE I
DEFINITIONS OF SEVERAL SYMBOLS

Symbols	Description	Example
T_L	True-line	
F_L	False-line	
T_I	Set of inputs on the true-line	$T_I=\{a_0, a_1, a_4, a_7\}$ where a_j is an input variable attached to the true-line. j is a sequence number.
F_I	Set of inputs on the false-line	$F_I=\{b_2, b_3, b_5, b_6\}$ where b_j is an input variable attached to the false-line. j is a sequence number.
S	Set of sequence numbers of T_I and F_I	S={0, 1, 2, 3, 4, 5, 6, 7} for this example.
SE(x)	Function which extracts the sequence number from x	$SE(a_0)=0$ $SE(b_2)=2$
CDL	Clock Distribution Line	
$W(i_1,i_2,o_1,o_2)$	WTAC whose true-line input is i_1, false-line input is i_2, OUT is o_1, and $\overline{OUT}$ is o_2.	$W(T_{L1},F_{L1},CDL_2,F_{L2})$
R	Distinct Race Logic	$R_1=\{T_{L1},F_{L1},T_{I1},F_{I1},CDL_1,W_1\}$

variations into account [10]. The considered process variations include threshold voltage, gate oxide thickness, active layer resistance, channel length, width of MOSFETs, and values of resistors of the CDL. These variations are applied to all of MOSFETs in Fig. 7(a). To highlight the mismatch effect, the input pattern or $\{A, B, C, D, E\} = \{0, 1, 1, 1, 1\}$ is selected, with which three switches pull down the voltage of the true-line and only two switches pull down that of the false-line that

TABLE II
EXPLANATIONS OF SIX METHODS FOR SYNTHESIS

Method Name	Explanation
M-AND1	Put the first input variable in the false-line and the second input variable in the false-line.
M-AND2	Put the first input variable in the false-line and the second input variable in the true-line.
M-OR1	Put the first input variable in the true-line and the second input variable in the true-line.
M-OR2	Put the first input variable in the true-line and the second input variable in the false-line.
M-AND3	Complete the first Race Logic with termination and WTAC. Connect the OUT of the first Race logic to the CDL of the second Race Logic and connect $\overline{OUT}$ to the false-line.
M-OR3	Complete the first Race Logic with termination and WTAC. Connect the $\overline{OUT}$ of the first Race logic to the CDL of the second Race Logic and connect OUT to the true-line.

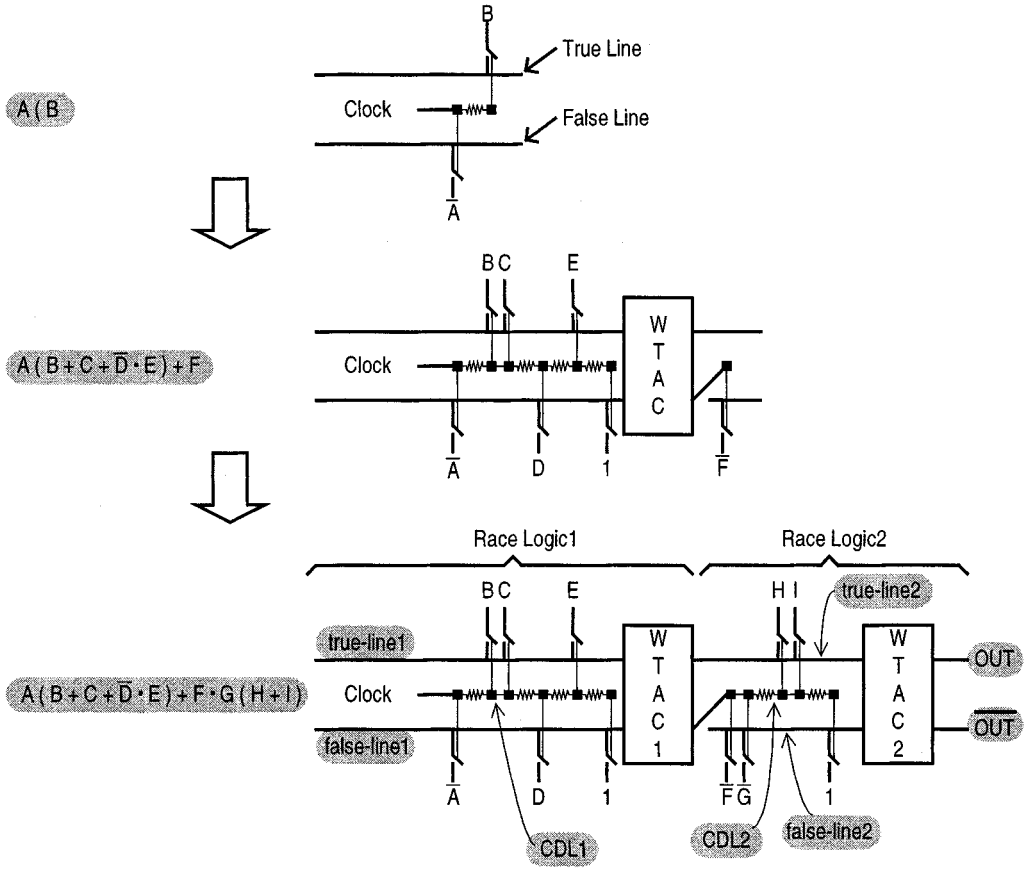
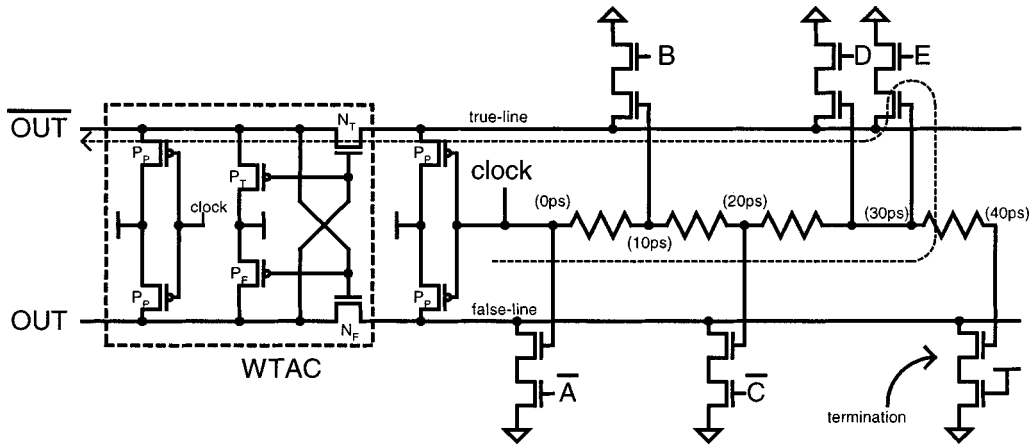


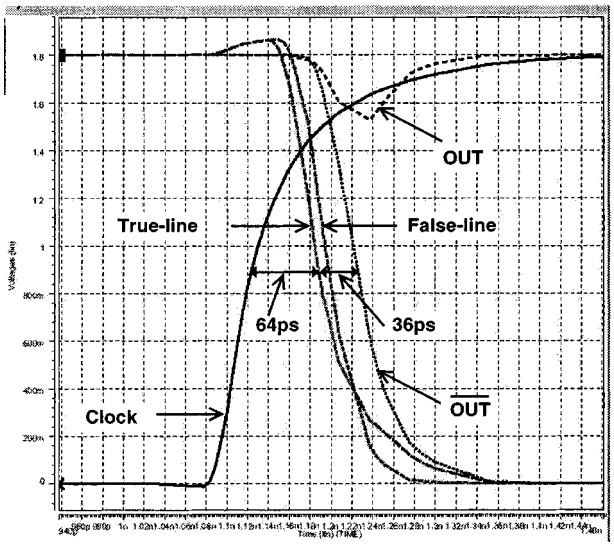
Fig. 6. Example of the synthesis sequence.

must be discharged earlier than the true-line. Fig. 8 shows the malfunction probability measured with respect to the timing gap in the CDL. As shown in this simulation, even 5 ps can be used to maximize its operation speed. In this study, we choose 10 ps for ease of design.

To improve the mismatch robustness and noise immunity, several circuit techniques can be applied as shown in Fig. 9. When the true-line discharges first, the nMOS gate of the false-line, N_F in Fig. 9, is turned off, so that the load capacitance on the false-line is diminished because the junction capacitance of P_1



(a)



(b)

Fig. 7. (a) Example of the circuit implementation of race logic. Boolean function: $\mathrm{OUT} = A\{B + C(D + E)\}$. (b) HSPICE simulation waveform.

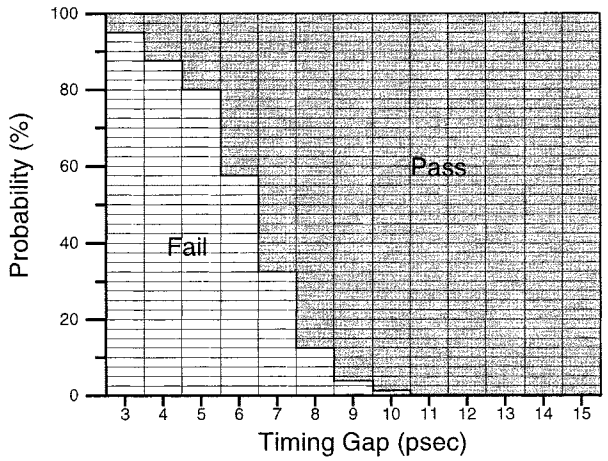


Fig. 8. Monte Carlo simulation result.

and P_2 is isolated from the false-line. This capacitance reduction on the false-line slightly helps the discharging of the false-line, and this can cause the malfunction of the WTAC. To compensate for the isolation, a replica circuit that balances the capacitance of each race line can be added. The optimized sizing of the input switches also improves robustness and noise immunity. In the worst case, the switch A may discharge the false-line alone and all of the true-line switches may discharge the true-line. Even though the discharging of the false-line started earlier than that of the true-line, the discharging of the true-line overrides that of the false-line, which may result in the malfunction of the WTAC. As the number of input switches increases, the situation becomes worse. This problem can be resolved by adjusting the sizes of the input switches. If the size is optimized to make the early-triggered switch discharge the race line earlier and more strongly, the operation of the WTAC can be more stable.

IV. Comparison With Conventional Logic Families

The carry generation circuits of the carry–look-ahead adder are implemented by RALA, dynamic logic, DVCSPG, and SSDL. The comparisons focus on speed, area, and power consumption. 0.18-μm CMOS model parameters are used for the simulation.

In the delay time comparison, no other logic families are found to be faster than the race logic, as shown in Fig. 10(a). Compared to the DCVSPG logic that is found to be the fastest

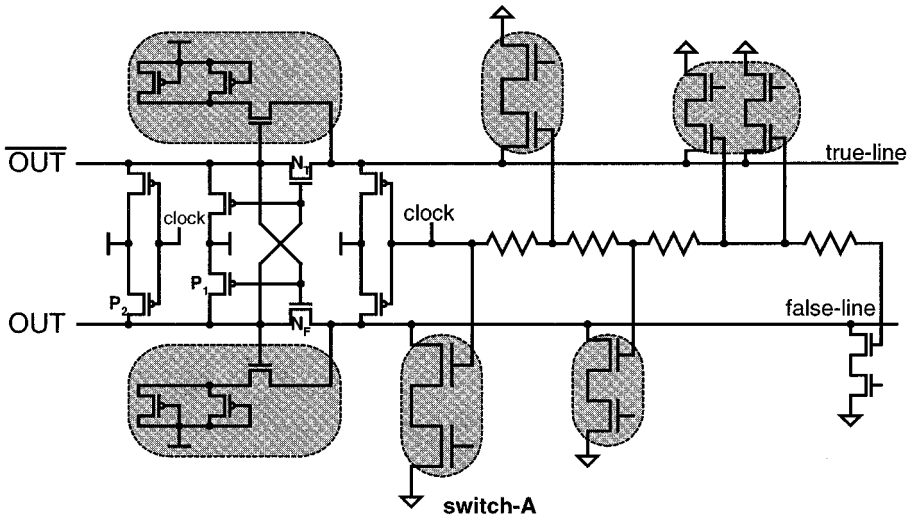


Fig. 9. Load compensation and switch resizing techniques.

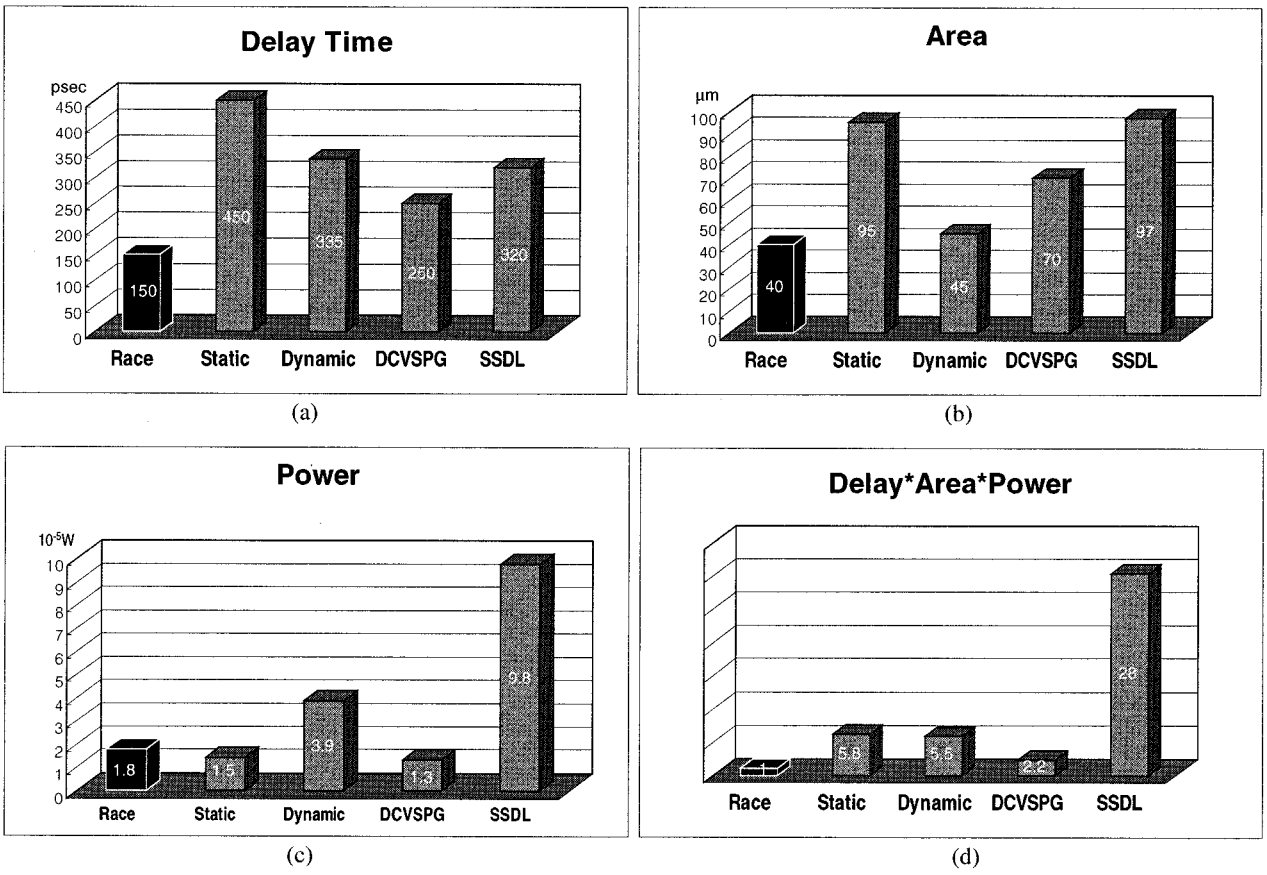


Fig. 10. Result graphs of the performance comparison.

among the conventional schemes in this comparison, the race logic shows 50% speed improvement. The area or the total width of MOSFETs is compared. We assume that the resistor array in the CDL of the race logic is made of poly silicon without salicide. The area occupied by one resistor of the resistor array is calculated as the same as that of the smallest nMOS. In the area comparison, the area of the race logic is similar to that of the dynamic logic, as shown in Fig. 10(b). In the power comparison shown in Fig. 10(c), the race logic consumes 26% more power than the DCVSPG logic that consumes the lowest power. A circuit implemented by race logic has two dynamic nodes, the true-line and the false-line. These two lines are precharged and discharged at every clock cycle, and this consumes dynamic power. However, compared to the dynamic logic, the power consumption of the race logic is smaller than that of the dynamic logic even though it also has dynamic nodes. This is because the size of the transistors related with the dynamic nodes of the race logic is the smallest. The race logic is so fast that it does not

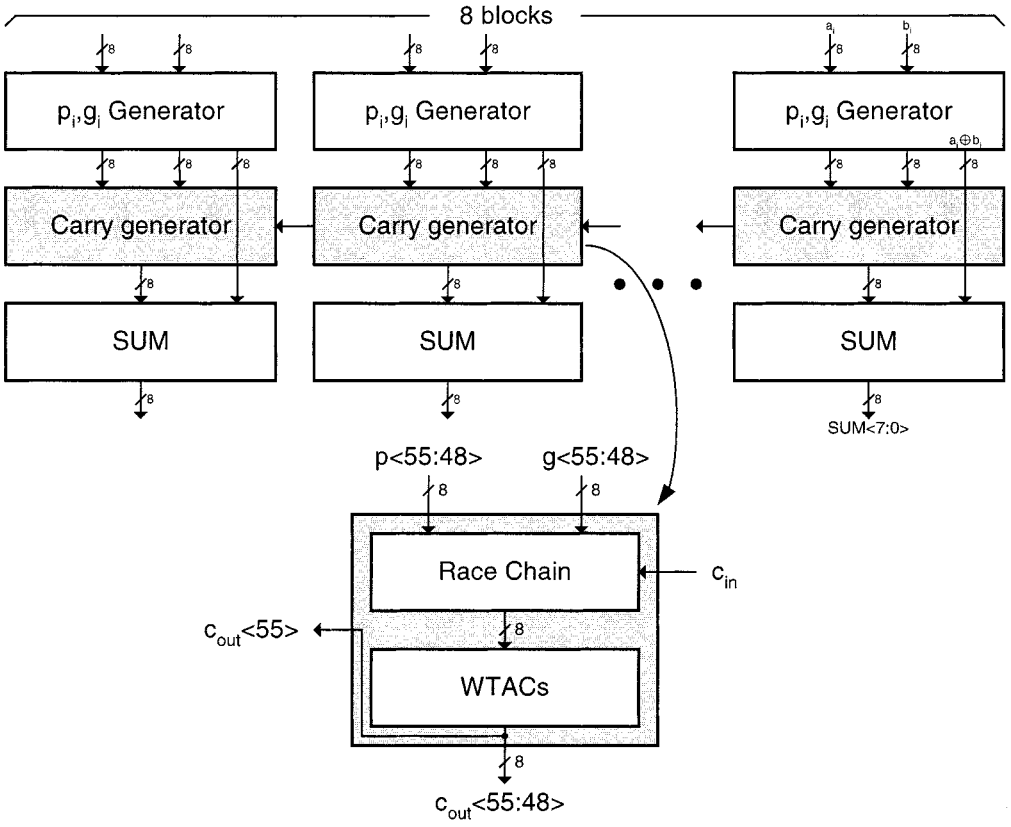


Fig. 11. Block diagram of 64-bit carry–look-ahead adder using RALA.

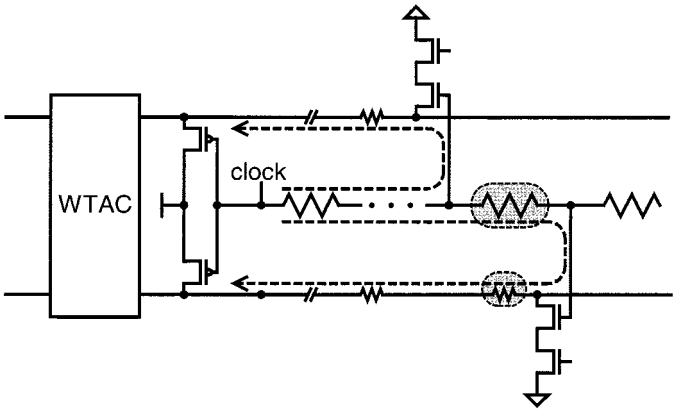


Fig. 12. Signal paths of two adjacently triggered signals.

need to enlarge the transistor width. The delay, area, and power production are shown in Fig. 10(d), and the race logic shows the best performance.

V. CMOS Adder Implementation

To verify the feasibility and functionality of RALA, a 64-bit carry–look-ahead adder is designed and fabricated by 0.25-μm six-metal CMOS technology. The carry generation logic of this adder is designed on the basis of RALA. The architecture of the adder is illustrated in Fig. 11. This carry–look-ahead adder consists of eight 8-bit subadders. Each subadder has one carry generation circuit, which has one gate depth. If the carry generation circuit is implemented by conventional logic styles, it must have several gate stages.

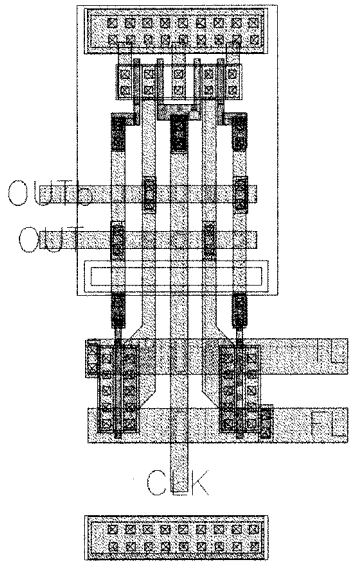


Fig. 13. Layout of WTAC.

The race logic is one type of the clocked nMOS logic, and the inputs of the race logic must be 0 during the precharge phase. The p and g generators, the prestage of the carry generator, are designed by dynamic logic, so that the outputs are precharged to low during the precharge phase. The sum generation circuit is designed by conventional static logic style.

The carry generator designed by race logic certainly does not waste wire delay. As depicted in Fig. 12, the timing difference between two neighboring signals is determined by the clock delay in the CDL plus the wire delay. This means that WTAC

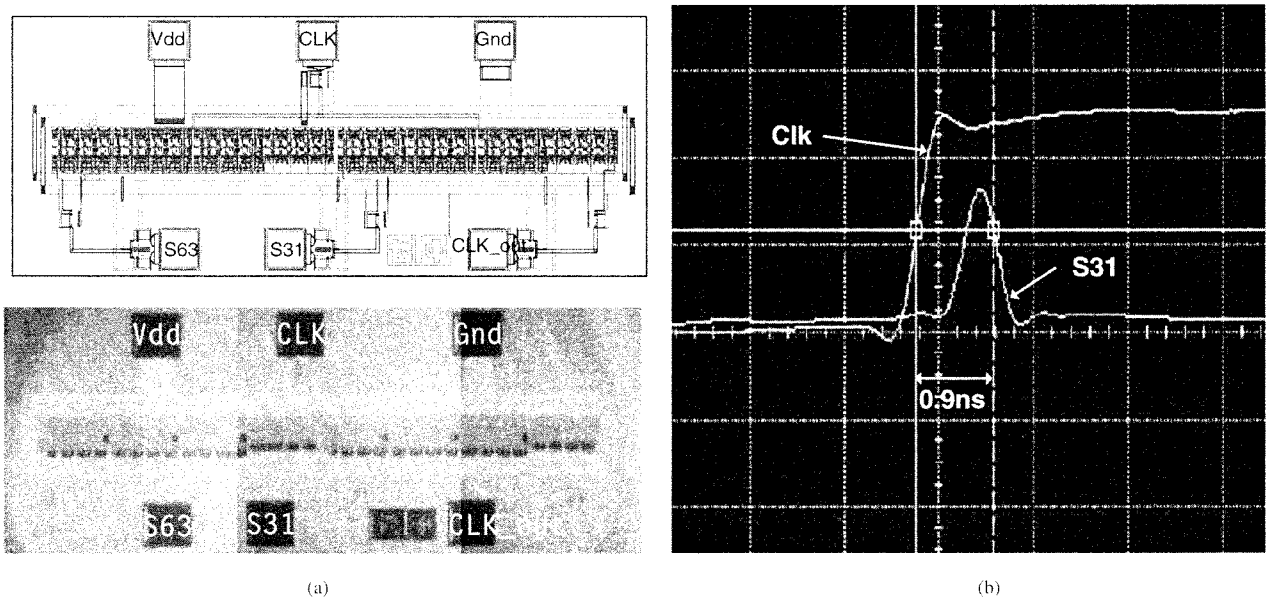


Fig. 14. (a) Photograph and layout of the 64-bit carry–look-ahead adder. (b) Measured waveform.

requires the timing difference and it can be the wire delay in the race lines instead of the clock delay in the CDL. Therefore, the delay time in the race lines that is inevitable because of the geometry of the layout can be used effectively instead of being wasted. In the actual layout, the length of the race lines is about 100 μm in each carry generation circuit.

The layout of WTAC is carefully handcrafted to maintain symmetry as shown in Fig. 13. Polycide with salicide blocking is used as a CDL resistor. The active area of the adder is 800 μm $\times$ 150 μm. The delay time from the clock to Sum31 measures 0.9 ns. The photograph of the adder and measured waveform are shown in Fig. 14(a) and (b), respectively.

VI. Conclusion

A novel logic concept, Race Logic Architecture (RALA), is proposed. RALA realizes logic operations with signal racing between two race lines instead of the actions of logic gates. RALA overcomes transition delay times of logic gates in conventional logic circuits and does not just waste the propagation delay times of routing wires but utilizes it for logic operations. In this regard, RALA is a quite promising concept as fabrication technology develops so that the delay time caused by increased parasitic capacitance of routing wires becomes more serious.

RALA consists of three parts: the clock distribution line, the race lines, and the winner-take-all circuit. The CDL generates sequential trigger signals for input variable switches. Triggered signals start racing on two race lines, the true-line and the false-line. The winner-take-all circuit determines which signal arrives earlier.

CMOS circuits of the carry generation function of a general carry–look-ahead adder are implemented by RALA and by various other logic families. The circuit implemented by RALA shows good performance in terms of speed and area, and also shows reasonable power consumption.

Using RALA, a 64-bit carry–look-ahead adder was fabricated by 0.25-μm CMOS technology to confirm its feasibility. Its measured delay time from the clock to SUM31 was 0.9 ns.

References

[1] F.-S. Lai and W. Hwang, "Design and implementation of differential cascode voltage switch with pass-gate (DCVSPG) logic for high performance digital systems," *IEEE J. Solid-State Circuits*, vol. 32, pp. 563–573, Apr. 1997.

[2] T. A. Grotjohn *et al.*, "Sample-set differential logic (SSDL) for complex high-speed VLSI," *IEEE J. Solid-State Circuits*, vol. SC-21, pp. 367–369, Apr. 1986.

[3] H.-J. Yoo *et al.*, "A 150 MHz 8-Banks 256M synchronous DRAM with wave pipelining methods," in *Int. Solid-State Circuit Conf.*, vol. 38, Feb. 1995, pp. 250–253.

[4] H. J. Yoo, "Dual-V_T self-timed CMOS logic for low subthreshold current multi-gigabit synchronous DRAM," *IEEE Trans.Circuit Syst. II*, vol. 45, pp. 1263–1271, Sept. 1999.

[5] K. Yano *et al.*, "A 3.8 ns CMOS 16 $\times$ 16 multiplier using complementary pass transistor logic," *IEEE J. Solid-State Circuits*, vol. 25, pp. 388–395, Apr. 1990.

[6] H. Hasegawa *et al.*, "A DC-powered Josephson logic family that uses hybrid unlatching flip-flop logic elements (huffles)," *IEEE Trans. Appl. Superconduct.*, vol. 5, pp. 3504–3510, Dec. 1995.

[7] H.-T. Ng *et al.*, "CMOS current steering logic for low-power mixed-signal systems," in *IEEE Symp. Low Power Electronics Dig. Tech. Papers*, 1994, pp. 14–15.

[8] T. Hanyu *et al.*, "Integration of asynchronous and self-checking multiple-valued current-mode circuits based on dual-rail differential logic," in *Proc. Int. Symp. Dependable Computing*, 2000, pp. 27–33.

[9] S. J. Lee and H. J. Yoo, "A novel high speed low power logic family: Race logic," *IEEE Eur. Solid-State Circuit Conf.*, pp. 420–423, Sept. 2000.

[10] M. J. M. Pelgrom *et al.*, "Matching properties of MOS transistors," *IEEE J. Solid-State Circuits*, vol. 24, pp. 1433–1989, Oct. 1989.

Se-Joong Lee was born on January 9, 1978, in Korea. He received the B.S. and M.S. degrees in electrical engineering and computer science from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1999 and 2001, respectively. Since 1999, he has been a Research Assistant at KAIST, where he is currently working toward the Ph.D. degree.

His research activities are related to network switches and high-speed circuit techniques, especially network processor design using embedded-memory logic (EML) technology.

Hoi-Jun Yoo graduated from the Electronic Department of Seoul National University in 1983 and received the M.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), in 1985 and 1988, respectively. His Ph.D. work concerned the fabrication process for GaAs vertical optoelectronic integrated circuits.

From 1988 to 1990, he was with Bell Communications Research, Red Bank, NJ, and invented the two-dimensional phase-locked VCSEL array, the front-surface-emitting laser, and the high-speed lateral HBT. In 1991, he became Manager of a DRAM design group at Hyundai Electronics and designed a family of fast-1M DRAMs and synchronous DRAMs including 256M SDRAM. From 1995 to 1997, he was a faculty member of Kangwon National University. In 1998, he joined the faculty of the Department of Electrical Engineering at KAIST and currently leads a project team on RAMP (RAM Processor). In 2001, he founded a national research center, SIPAC (System Integration and IP Authoring Research Center), funded by the Korean government to promote world-wide IP authoring and its SOC application. His current interests are SOC design, IP authoring, high-speed and low-power memory circuits and architectures, design of embedded memory logic, optoelectronic integrated circuits, and novel devices and circuits. He is the author of the books *DRAM Design* (in Korean, 1996) and *High Performance DRAM* (in Korean, 1999).

Dr. Yoo was the Technical Program chair of the 9th Korean Conference on Semiconductors. He received the 1994 Electronic Industrial Association of Korea Award for his contribution to DRAM technology.