On digital VLSI circuits exploiting collision-based fusion gates

Kazuhito Yamada[1], Tetsuya Asai[1], Ikuko N. Motoike[2], and Yoshihito Amemiya[1]

[1] Graduate School of Information Science and Technology, Hokkaido University, Kita 14, Nishi 9, Kita-ku, Sapporo, 060-0814 Japan
[2] Future University - Hakodate, 116-2 Kamedanakano-cho Hakodate Hokkaido, 041-8655 Japan

Abstract. Collision-based reaction-diffusion computing (RDC) represents information quanta as traveling chemical wave fragments on an excitable medium. Although the medium's computational ability is certainly increased by utilizing its spatial degrees of freedom [2], our interpretation of collision-based RDC in this paper is that wave fragments travel along 'limited directions' 'instantaneously' as a result of the 'fusion of particles'. We do not deal with collision-based computing here, but will deal with conventional silicon architectures of a 'fusion gate' inspired by collision-based RDC. The hardware is constructed of a population of collision points, i.e., fusion gates, of electrically equivalent wave fragments and physical wires that connect the fusion gates to each other. We show that i) fundamental logic gates can be constructed by a small number of fusion gates, ii) multiple-input logic gates are constructed in a systematic manner, and iii) the number of transistors in specific logic gates constructed by the proposed method is significantly smaller than that of conventional logic gates while maintaining high-speed and low-power operations.

1 Introduction

Present digital VLSI systems consist of a number of combinational and sequential logic circuits as well as related peripheral circuits. A well-known basic logic circuit is a two-input NAND circuit that consists of four metal-oxide semiconductor field-effect transistors (MOS FETs) where three transistors are on the current path between the power supply and the ground. Many complex logic circuits can be constructed by not only populations of a large number of NAND circuits but also special logic circuits with a small number of transistors (there are more than three transistors on the current path) compared with NAND-based circuits.

A straight-forward way to construct low-power digital VLSIs is to decrease the power-supply voltage because the power consumption of digital circuits is proportional to the square of the supply voltage. In complex logic circuits, where many transistors are on the current paths, the supply voltage cannot be decreased due to stacking effects of transistors' threshold voltages, even though

the threshold voltage is decreasing as LSI fabrication technology advances year by year. On the other hand, if two-input basic gates that have the minimum number of transistors (three or less) on the current path are used to decrease the supply voltage, a large number of the gates will be required for constructing complex logic circuits.

The Reed-Muller expansion [14, 13], which expands logical functions into combinations of AND and XOR logic, enables us to design 'specific' arithmetic functions with a small number of gates, but it is not suitable for arbitrary arithmetic computation. Pass-transistor logic (PTL) circuits use a small number of transistors for basic logic functions but additional level-restoring circuits are required for every unit [18]. Moreover, the acceptance of PTL circuits into mainstream digital design critically depends on the availability of tools for logic, physical synthesis, and optimization. Current-mode logic circuits also use a small number of transistors for basic logic, but their power consumption is very high due to the continuous current flow in turn-on states [5]. Subthreshold logic circuits where all the transistors operate under their threshold voltage are expected to exhibit ultra-low power consumption, but the operation speed is extremely slow [16, 17]. Binary decision diagram logic circuits are suitable for next-generation semiconductor devices such as single-electron transistors [15, 20, 6], but not for present digital VLSIs because of the use of PTL circuits.

To address the problems above concerning low-power and high-speed operation in digital VLSIs, we describe a method of designing logic circuits with collision-based fusion gates, which is inspired by collision-based reaction-diffusion computing (RDC) [2, 12, 11]. This paper is organized as follows. In section 2, we briefly overview collision-based RDC. Then, in section 3, we introduce a new interpretation of collision-based RDC, especially concerning directions and speeds of propagating information quanta. We also show basic logical functions constructed by simple unit operators, i.e., fusion gates, and demonstrate a circuit's operation by using a simulation program with integrated circuit emphasis (SPICE) with typical device parameters. Then, a reconfigurable architecture for the proposed fusion-gate structure and a possible construction of D-type flip flop circuits for constructing sequential circuits are presented. Section 4 is a summary.

2 Collision-Based Logical Computation

Dynamic, or collision-based, computers employ mobile self-localizations, which travel in space and execute computation when they collide with each other. Truth values of logical variables are represented by the absence or presence of the traveling information quanta. There are no pre-determined wires: patterns can travel anywhere in the medium, a trajectory of a pattern motion is analogous to a momentarily wire [1–3]. A typical interaction gate has two input 'wires' (trajectories of colliding mobile localizations) and, typically, three output 'wires' (two 'wires' represent localization trajectories when they continue their motion undisturbed, the third output gives a trajectory of a new localization, formed in the collision of two incoming localizations). The traveling of patterns is analogous

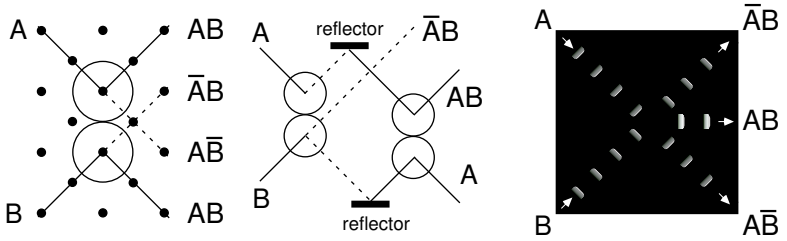

Fig. 1. Collision-based computing models (a) conservative billiard-ball logic for AND and partial XOR computation [8, 9], and (b) nonconservative (dissipative) reaction-diffusion logic that has the same function as that of (a) [3].

to information transfer while collision is an act of computation; thus, we call the set up 'collision-based computing'. There are three sources of collision-based computing: proof of the universality of Conway's Game of Life via collisions of glider streams [7], conservative logic [8], cellular automaton implementation of the billiard ball model [9], and particle machine [19] (a concept of computation in cellular automata with soliton-like patterns); see overviews in [2].

The main purpose of collision-based computing is to perform computation in an 'empty space', i.e., a medium without geometrical constraints. Basic toy models of collision-based computing are shown in Fig. 1. In the billiard ball logic shown in Fig. 1(a), a set of billiard balls are fired into a set of immovable reflectors at a fixed speed. As the billiard balls bounce off each other and off the reflectors, they perform a reversible computation. Provided that the collisions between the billiard balls and between the billiard balls and the reflectors are perfectly elastic, the computation can proceed at a fixed finite speed with no energy loss.

Adamatzky demonstrated that a similar computation can be performed on excitable reaction diffusion systems [2, 3]. Figure 1(b) illustrates basic logic gates where instead of billiard ball wave fragments (white localizations in the figure) travel in an excitable reaction-diffusion medium. In typical excitable media, localized wave fragments facing each other disappear when they collide. With a special setup described in [3], those excitable waves do not disappear, but they do produce subsequent excitable waves.

3 New interpretation of collision-based computing for digital VLSIs

Adamatzky proposed how to realize arithmetical scheme using wave fragments traveling in a homogeneous medium [2, 3]. The sub-excitable Belousov-Zhabotinsky (BZ) system was emulated by a 2^+-medium [1], which was a 2D cellular automaton, where each cell took three states — resting, excited, and refractory, and updated its state depending on the number of excited cells in its eight-cell

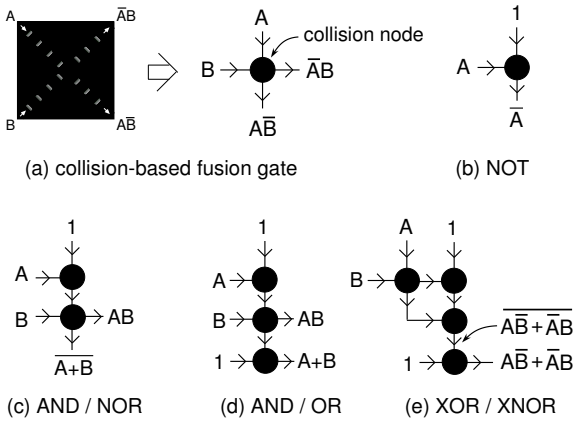

Fig. 2. Definition of collision-based fusion gate (a) and basic logical circuits using several fusion gates [(b)-(e)] that produce multiple logical functions.

neighborhood. A resting cell become excited only if it had exactly two excited neighbors. An excited cell took the refractory state and refractory cell took the resting state unconditionally, i.e., independently of its neighborhood. The model exhibited localized excitations traveling along columns and rows of the lattice and along diagonals. The particles represented values of logical variables. Logical operations were implemented when particles collided and were annihilated or reflected as a result of the collision. Thus one can achieve basic logical gates in the cellular-automaton model of a sub-excitable BZ medium and build an arithmetic circuit using the gates.

A cellular automaton LSI that implements an excitable lattice for BZ systems has been implemented by one of the authors [10, 4]. Each cell consisted of several tens of transistors and was regularly arranged on a 2D chip surface. To implement a one-bit adder, for example, by collision-based cellular automata, at least several tens of cells are required to allocate sufficient space for the collision of wave fragments [2]. This implies several hundreds of transistors are required for constructing just a one-bit adder. Direct implementation of the cellular automaton model is therefore a waste of chip space, as long as the single cell space is decreased to the same degree of chemical compounds in spatially-continuous reaction-diffusion processors.

What happens if wave fragments travel in limited directions instantaneously? When such wave fragments are generated at the top and end of a pipe (not an empty space) filled with excitable chemicals, for example, these waves may disappear at the center of the pipe instantaneously. When two pipes are perpendicularly arranged and connected, wave fragments generated at the tops of the two pipes may also disappear at the connected point. If only one wave fragment (A or B) is generated at the top of one pipe, it can reach the end of the pipe [$A\overline{B}$ or $\overline{A}B$ in Fig. 1(b)]. A schematic model of this operation is shown in Fig. 2. In Fig. 2(a) (left), an excitable reaction-diffusion medium, where excitable

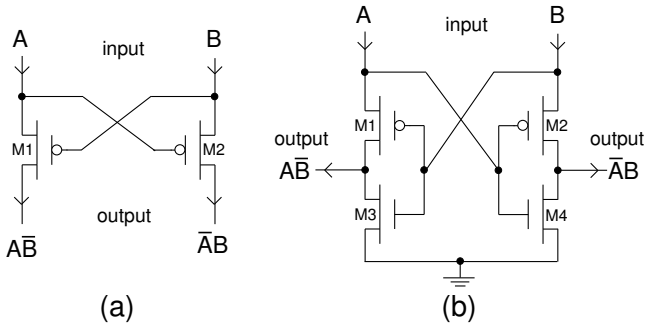

Fig. 3. Circuit construction of collision-based fusion-gate for digital VLSIs. (a) compact-but-slow circuit (two-transistor circuit) and (b) complex-but-fast circuit (four-transistor circuit).

waves (A and B) may disappear when they collide, is illustrated. In Fig. 2(a) (right), an equivalent model of two perpendicular directions of wave fragments, i.e., North-South and West-East fragments, is depicted. The input fragments are represented by values A and B where A (or B) = '1' represents the existence of a wave fragment traveling North-South (or West-East), and A (or B) = '0' represents the absence of wave fragments. When A = B = '1' wave fragments collide at the center position (black circle) and then disappear. Thus, East and South outputs are '0' because of the disappearance. If A = B = '0', the outputs will be '0' as well because of the absence of the fragments. When A = '1' and B = '0', a wave fragment can travel to the South because it does not collide with a fragment traveling West-East. The East and South outputs are thus '0' and '1', respectively, whereas they are '1' and '0', respectively, when A = '0' and B = '1'. Consequently, logical functions of this simple 'operator' are represented by $\overline{A}B$ and $A\overline{B}$, as shown in Fig. 2(a) (right). We call this operator a 'collision-based fusion gate', where two inputs correspond to perpendicular wave fragments, and two outputs represent the results of collisions (transparent or disappear) along the perpendicular axes. Notice that in this configuration the computation is performed with geometrical constraints. Figures 2(b) to (e) represent basic logic circuits constructed by combining several fusion gates. The simplest example is shown in Fig. 2(b) where the NOT function is implemented by a single fusion gate. The North input is always '1', whereas the West is the input (A) of the NOT function. The output appears on South node ($\overline{A}$). Figure 2(c) represents a combinational circuit of two fusion gates that produces AND and NOR functions. An OR function is thus obtained by combining NOT and AND/NOR fusion gates in Figs. 2(b) and (c), respectively, as shown in Fig. 2(d). Exclusive logic functions are produced by three (for XNOR) or four (for XOR) fusion gates as shown in Fig. 2(e).

A collision-based fusion gate receives two logical inputs (A and B) and produces two logical outputs ($\overline{A}B$ and $A\overline{B}$). CMOS circuits for this gate are shown in Fig. 3. They receive logical (voltage) inputs (A and B) and produce the logic

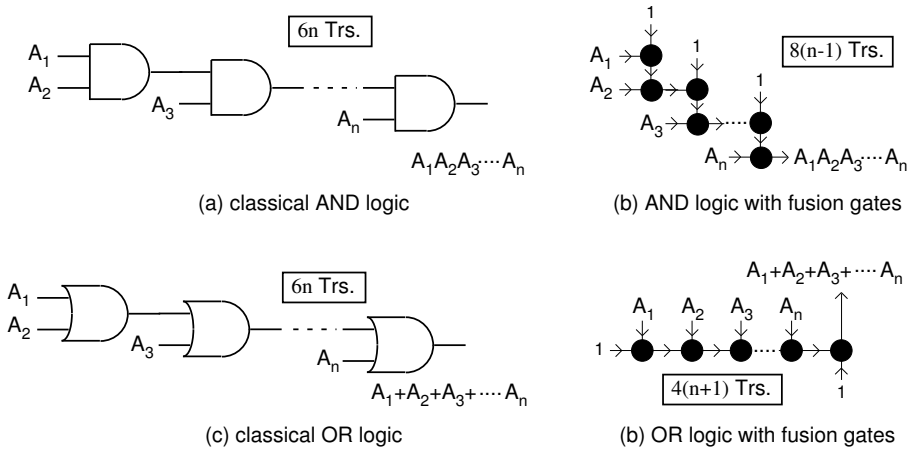

Fig. 4. Classical and fusion-gate logic architectures of multiple-input AND and OR circuits. In classical circuits [(a) and (c)], each logical unit consists of six transistors with constraints for low-power operation; three transistors are on current path between power supply and ground. In fusion-gate logic [(b) and (d)], each fusion gate consists of four transistors as shown in Fig. 3(b).

function. The minimum circuit structure based on PTL circuits is shown in Fig. 3(a), where a single-transistor AND logic is fully utilized. When an nMOS transistor receives voltages A and B at its gate and drain, respectively, the source voltage is given by AB at equilibrium; in the case of pMOS, that is given by $\overline{A}B$. Although there are just two transistors in this construction, there is a severe problem, i.e., the operation speed. When a pMOS transistor is turned off, the output node's parasitic capacitance is discharged by the leak current of the pMOS transistor in the off state. Therefore, the transition time will be rather long, e.g., typically within a few tens of milliseconds when the conventional CMOS process is used, which implies the circuit operates very slowly compared with conventional digital circuits. Additional resistive devices for the discharging may improve the upper bound of the clock frequency. However, we need a breakthrough while satisfying the constraints of a small number of transistors in a fusion gate. One solution to the operation-speed problem is shown in Fig. 3(b). The circuit has two additional nMOS transistors just beneath the pMOS transistors in the two-transistor circuits. If a pMOS transistor is turned off, an nMOS transistor connected between the pMOS transistor and the ground discharges the output node, which significantly increases the upper bound of the operation frequency.

A multiple-input AND and OR implementation with classical and fusion-gate logic is shown in Fig. 4. In classical circuits (a) and (c), each logical unit (two-input AND and OR) consists of six transistors. As introduced in section 1, to decrease the power supply voltage for low-power operation, a minimum number of transistors (three or less) should be on each unit's current path. Each unit

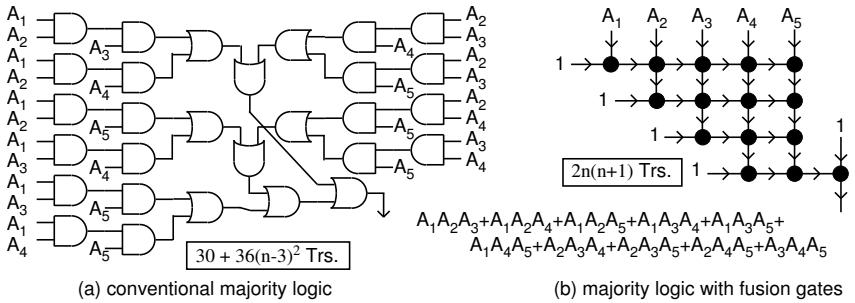

Fig. 5. Classical and fusion-gate logic architectures of majority logic circuits.

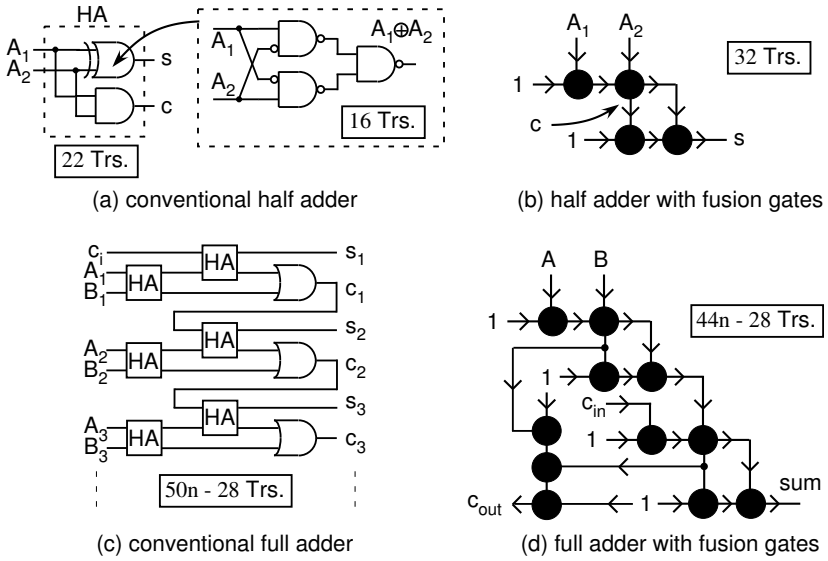

Fig. 6. Classical and fusion-gate logic architectures of half- and full-adder circuits.

circuit has six transistors, so n-input AND and OR gates consist of $6(n-1)$ transistors ($n \geq 2$). On the other hand, in fusion-gate logic (b) and (d), an n-input AND gate consists of $8(n-1)$ transistors [$2(n-1)$ fusion gates], whereas $4(n+1)$ transistors will be used in an n-input OR gate. Therefore, in the case of AND logic, the number of transistors in classical circuits is smaller than that of fusion-gate circuits. However, in the case of OR circuits, the number of transistors in fusion-gate circuits is smaller than that of classical circuits, and the difference will be significantly expanded as n increases.

Classical and fusion-gate implementations of majority logic circuits with multiple inputs are shown in Fig. 5. Again, in classical circuits, the number of transistors on each unit's current path is fixed to three. Five-input majority gates are illustrated in the figure. For n-bit inputs (n must be an odd number larger than

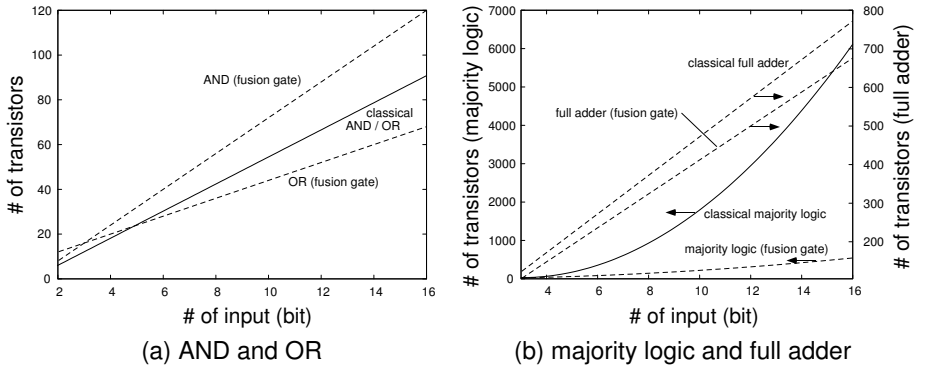

Fig. 7. Comparison of the number of transistors between classical and fusion-gate logic.

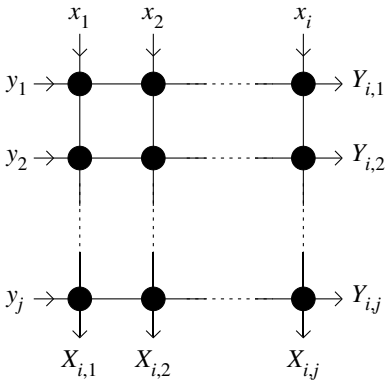

Fig. 8. Fusion-gate array of fusion gates on 2D rectangular grid where x_i and y_i represent the horizontal and vertical inputs and X_i and Y_i the horizontal and vertical outputs.

3), the number of transistors in the classical circuit was $30 + 36(n-3)^2$, while in the fusion-gate circuit, it was $2n(n+1)$, which indicates that the fusion-gate circuit has a significantly smaller number of transistors.

Half and full adders constructed by classical and fusion-gate logic are illustrated in Fig. 6. There were 22 transistors in a classical half adder [Fig. 6(a)], whereas they were was 32 in a fusion-gate half adder [Fig. 6(b)]. For n-bit full adders ($n \geq 1$), there were $50n - 28$ transistors in a classical circuit [Fig. 6(c)], whereas there were $44n-28$ in a fusion-gate circuit [Fig. 6(d)]. Again, the fusion-gate circuit has a little advantage in the number of transistors, but the difference will increase as n increases.

The comparison of the number of transistors between classical and fusion-gate logic is summarized in Fig. 7. Except for an AND logic ($n \geq 4$), the number of transistors in fusion-gate logic is always smaller than that of transistors in classical logic circuits, especially in majority logic gates.

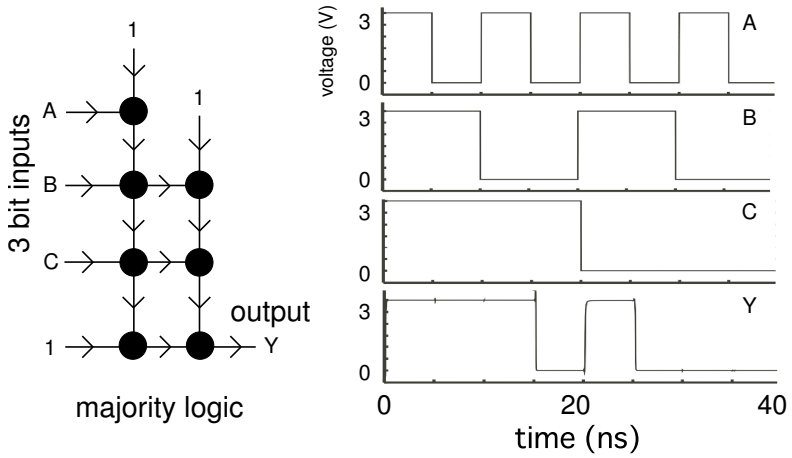

Fig. 9. Example construction of majority logic gate and its simulation results with 0.35-μm digital CMOS parameters (MOSIS, Vendor TSMC).

The computational ability of the proposed methodology can be evaluated by calculating logical functions of a 2D array of fusion gates at each output node. The 2D computing matrix, where x_i and y_i represent the horizontal and vertical inputs and X_i and Y_i the horizontal and vertical outputs, is shown in Fig. 8. For simplicity, here we assume y_i = logical '1'. Then, the horizontal and vertical outputs are represented by the following difference equations:

$$Y_{i,1} = \overline{\sum_{k=1}^{i} X_k}, \quad (i \geq 1) \tag{1}$$

$$Y_{i,j} = \overline{\sum_{k=1}^{i-(j-1)} Y_{i,j-1}}, \quad (i \geq j) \tag{2}$$

$$X_{i,1} = X_i \sum_{k=1}^{i-1} X_k, \quad (i \geq 2) \tag{3}$$

$$X_{i,j} = X_i \sum_{k=1}^{i-j} X_{i,j-1}, \quad (i \geq j). \tag{4}$$

An example of implementation and simulations of three-input majority logic gates is shown in Fig. 9. Seven fusion gates, i.e., 28 transistors for high-frequency operation, are required for this function, whereas in conventional architecture, 30 transistors are necessary for the same function. Using the fusion-gate circuit shown in Fig. 3(b), we simulated the majority logic circuit by using SPICE with 0.35-μm digital CMOS parameters (MOSIS, Vendor TSMC, with minimum-sized transistors). The results obtained where A, B, and C represent the input, while Y represents the output are shown in Fig. 9 (right). The clock frequency was

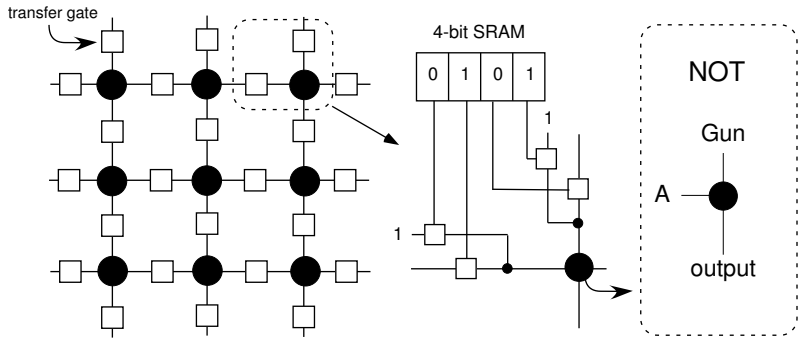

Fig. 10. Possible construction of reconfigurable logic architecture.

100 MHz. The rise time of the output was 0.3 ns for this parameter set. The operation speed is thus significantly faster than that of the two-transistor circuit.

Let us consider the number of transistors per function shown in Figs. 2(c)-(e). A two-transistor fusion gate [Fig. 3(a)] is used for the time being. In the case of NOT, two transistors are necessary; that is the same number of transistors as that of a conventional inverter circuit. For AND and NOR functions, four transistors are required, which is half the number of transistors in a combinational circuit of conventional AND and NOR circuits. In the case of AND and OR, six transistors are required, whereas ten are required in conventional circuits. Therefore, for low clock-frequency applications, fusion-gate logical computing with two-transistor fusion gates certainly decreases the number of transistors in the circuit network. For high-speed applications, four-transistor fusion gates have to be used; however, the number of transistors is doubled in this case.

What are the merits of fusion-gate architecture for high-speed applications? There are two types of answers: introducing yet another device and completely new functions. In the former case, a single-electron reaction-diffusion device [4] is a possible candidate. For the latter case, let us consider 'reconfigurable' functions. A basic idea of the reconfigurable logic architecture using fusion gates is shown in Fig. 10. Each fusion gate is regularly arranged on a 2D chip surface and is locally connected to other fusion gates via transfer gates. In this construction a new unit gate consists of four transfer gates, a single fusion gate circuit, and a four-bit memory circuit in which two bits give static inputs to the gate, and the remainder is for selecting the signal flow. As an example, a NOT function is depicted in the figure.

Arbitrary combinational circuits for arithmetic modules can be constructed by the proposed fusion-gate circuits, but how about sequential circuits for practical computation? In other words, how can we implement static memory, e.g., flip-flop, circuits? A possible answer to this question is shown in Fig. 11. Remember that the fusion gate circuit shown in Fig. 3(b) consists of two inverters. Therefore, by rewiring the fusion gate circuit, one can construct a D-type flip-flop (D-FF) circuit, as shown in Fig. 11. Although four additional transfer-gates

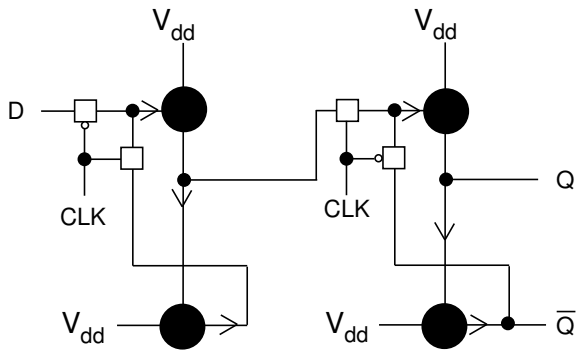

Fig. 11. Construction of D-type flip flop for sequential circuits.

are required, D-FF circuits can be constructed while maintaining the basic construction of a 2D array of fusion gate circuits.

4 Summary

We described a method of designing logic circuits with fusion gates which is inspired by collision-based reaction-diffusion computing. First, we introduced a new interpretation of collision-based computing, especially concerning a limited direction of wave fragments and infinite transition speed. This simplified construction of the computing media significantly. Second, we showed that basic logical functions could be represented in terms of our unit operator that calculated both $\overline{A}B$ and $A\overline{B}$ for inputs A and B. Third, two basic MOS circuits were introduced; one consisted of two transistors but operated very slowly, whereas the other consisted of four transistors but operated much faster than the two-transistor circuit. In the constructions of basic logic functions, the number of transistors in the two-transistor circuit was smaller than that of the corresponding conventional circuits. However, the number in the four-transistor circuit was larger than them. The combination of the fusion gates produces multiple functions, e.g., an AND circuit can compute NOR simultaneously, so we should build optimization theories for generating multiple-input arbitrary functions. Finally, we introduced a possible construction of D-type flip-flop circuits for constructing sequential circuits.

Acknowledgments

The authors wish to thank Professor Andrew Adamatzky of the University of the West of England for valuable discussions and suggestions during the research and Professor Masayuki Ikebe of Hokkaido University for suggestions concerning various CMOS circuits. This study was supported in part by the Industrial Technology Research Grant Program in 2004 from the New Energy and Industrial

Technology Development Organization (NEDO) of Japan and by a grant-in-aid for young scientists [(B)17760269] from the Ministry of Education, Culture, Sports, Science, and Technology (MEXT) of Japan.

References

1. A. Adamatzky, "Universal dynamical computation in multi-dimensional excitable lattices", *Int. J. Theor. Phys.*, vol. 37, pp. 3069–3108, 1998.
2. A. Adamatzky, Editor, *Collision-Based Computing*, Springer-Verlag, 2002.
3. A. Adamatzky, "Computing with waves in chemical media: Massively parallel reaction-diffusion processors," *IEICE Trans. Electron.*, vol. E87-C, no. 11, pp. 1748–1756, 2004.
4. A. Adamatzky A., B. De Lacy Costello, and T. Asai, *Reaction-Diffusion Computers.* Elsevier, UK, 2005.
5. M. Alioto and G. Palumbo, *Model and Design of Bipolar and MOS Current-Mode Logic: CML, ECL and SCL Digital Circuits*, Springer, 2005.
6. N. Asahi, M. Akazawa, and Y. Amemiya, "Single-electron logic systems based on the binary decision diagram," *IEICE Trans. Electronics*, vol. E81-C, no. 1, pp. 49–56, 1998.
7. E. R. Berlekamp, J. H. Conway, and R. L. Guy, *Winning Ways for your Mathematical Plays.* Vol. 2. Academic Press, 1982.
8. F. Fredkin and T. Toffoli, "Conservative logic", *Int. J. Theor. Phys.*, vol. 21, pp. 219–253, 1982.
9. N. Margolus, "Physics-like models of computation", *Physica D*, vol. 10, pp. 81–95, 1984.
10. Y. Matsubara, T. Asai, T. Hirose, and Y. Amemiya, "Reaction-diffusion chip implementing excitable lattices with multiple-valued cellular automata," *IEICE ELEX*, vol. 1, no. 9, pp. 248–252, 2004.
11. I. N. Motoike and K. Yoshikawa, "Information operations with multiple pulses on an excitable field," *Chaos, Solitons and Fractals*, vol. 17, pp. 455–461, 2003.
12. I. Motoike and K. Yoshikawa, "Information operations with an excitable field," *Phy. Rev. E*, vol. 59, no. 5, pp. 5354–5360, 1999.
13. D. E. Muller, "Application of Boolean Algebra to Switching Circuit Design and to Error Detection", *IRE Trans. on Electr. Comp.*, vol. EC-3, pp. 6–12, 1954.
14. I. S. Reed, "A Class of Multiple-Error-Correcting Codes and Their Decoding Scheme", *IRE Trans. on Inform. Th.*, vol. PGIT-4, pp. 38–49, 1954.
15. R. S. Shelar and S. S. Sapatnekar, "BDD decomposition for the synthesis of high performance PTL circuits," *Workshop Notes of IEEE IWLS 2001*, pp. 298–303.
16. H. Soeleman and K. Roy, "Ultra-low power digital subthreshold logic circuits," in *Proc. 1999 Int. Symp. on low power electronics and design*, pp. 94–96.
17. H. Soeleman, K. Roy, and B. C. Paul, "Robust subthreshold logic for ultra-low power operation," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 1, pp. 90–99, 2001.
18. M. Song and K. Asada, "Design of low power digital VLSI circuits based on a novel pass-transistor logic," *IEICE Trans. Electronics*, vol. E81-C, no. 11, pp. 1740–1749, 1998.
19. K. Steiglitz, I. Kamal, and A. Watson, "Embedded computation in one–dimensional automata by phase coding solitons", *IEEE Trans. Comp.*, vol. 37, pp. 138–145, 1988.

20. T. Yamada, Y. Kinoshita, S. Kasai, H. Hasegawa, and Y. Amemiya, "Quantum-dot logic circuits based on the shared binary decision diagram," *Jpn. J. Appl. Phys.*, vol. 40, no. 7, pp. 4485–4488, 2001.