P2-14

Design of a Multi-Context FPGA Using a Floating-Gate-MOS Functional Pass-Gate

Masanori Hariyama Sho Ogata Michitaka Kameyama
Graduate School of Information Sciences, Tohoku University
Aoba 6-6-05, Aramaki, Aoba, Sendai, Miyagi,980-8579, Japan
Email: {hariyama@, ogata@kameyama., kameyama@}ecei.tohoku.ac.jp

Yasutoshi Morita
Nippon Precision Circuits Inc.
4-3, Fukuzumi 2-chome, Koto-ku
Tokyo 135-8430,Japan

***Abstract*— Multi-context FPGAs (MC-FPGAs) have multiple memory bits per configuration bit forming configuration planes for fast switching between contexts. The additional memory planes cause significant overhead in area and power consumption. To overcome the overhead, a fine-grained MC-FPGA architecture using a floating-gate-MOS functional pass gate(FGFP) is presented which merges threshold operation and storage function on a single floating-gate MOS transistor. The transistor count of the proposed multi-context switch (MC-switch) is reduced to 10% in comparison with SRAM-based one. The transistor count of the proposed MC-switch is also reduced to 20% in comparison with the MC-switch that uses floating-gate MOS transistors just as storage device. The test chip is designed using a 0.35μm EPROM technology, and the area of the proposed MC-FPGA is reduced to about 50% of that of a conventional MC-FPGA.**

I. Introduction

Dynamically-programmable gate arrays (DPGAs) provide more cost-effective implementations than conventional FPGAs where hardware resources are dedicated to a single context[1]. A DPGA is sequentially configured as different processors in real time, and efficiently re-uses the limited hardware resources in time. One typical DPGA architecture is a multi-context one. Multi-context FPGAs (MC-FPGAs) have multiple memory bits per configuration bit forming configuration planes for fast switching contexts. However, the additional memory planes cause significant overhead in area and power consumption.

To reduce the overhead of configuration memory in MC-FPGAs, this paper proposes two concepts as follows:

Fine-grained architecture: An FPGA has lookup tables (LUTs) to serve as programmable computation elements (logic blocks). Logic blocks are connected to each other through programmable switch blocks. Each switch block consists of many programmable switches. In conventional FPGAs such as Xilinx XC4000, relatively large logic blocks are used for flexibility. For example, two 4-input LUTs are used in XC4000. The major disadvantage of the large-size logic block is that the logic block is underutilized to a great degree. To solve this problem, fine-grained architecture [2] is used to increase the LUT utilization.

Floating-Gate-MOS functional pass gate using a floting-gate MOS transistor: Conventional FPGA uses a large number of SRAM cells to store configuration data in logic blocks and switch blocks. The area of the configuration memory is dominant in area. Moreover, the large number of SRAM cells will cause serious problem in static power dissipation in a deci-micron process because of their leakage current. To solve this problem, this paper presents a multi-context switch (MC-switch) using a non-volatile functional pass gate. The functional pass gate using a floating-gate MOS transistor(FGMOS) is the device that can merge a logic operation and storage in a single FGMOS[3]. The MC-FPGA can be constructed by the MC-switch switches that can generate an arbitrary ON/OFF pattern according to a context switching signal. The arbitrary ON/OFF pattern can be represented by combining logic functions called “window literals”, each of which is efficiently implemented by using the functional pass gates.

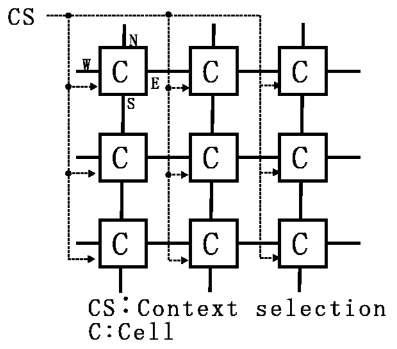

Fig. 1. Overall structure of an MC-FPGA.

By using the FGFP, the transistor count of the MC-switch is reduced to 10% in comparison with SRAM-based one. The transistor count of an MC-switch is also reduced to 20% in comparison with the MC-switch that uses floating-gate MOS transistors just as storage device. The test chip is fabricated in a 0.35μm CMOS EPROM process. The area of the MC-FPGA using the non-volatile functional pass gate is reduced to about 50% of the MC-FPGA using SRAM cells.

II. Fine-grained architecture

As shown in Fig. 1, the FPVLSI consists of a mesh-connected cellular array based on a bit-serial architecture to reduce the complexity of the switch block. Each cell is connected to only four adjacent cells, and the number of programmable switches is reduced in comparison with the typical FPGA. The specification of the functionality of a logic block provides a great impact on the complexity of the switch

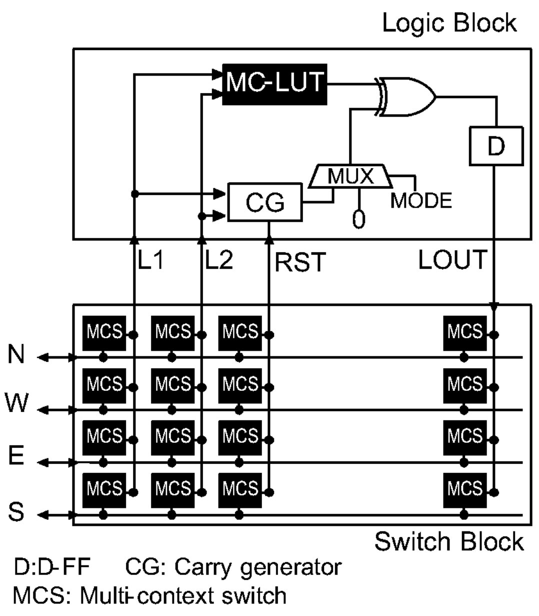

Fig. 2. Structure of a cell.

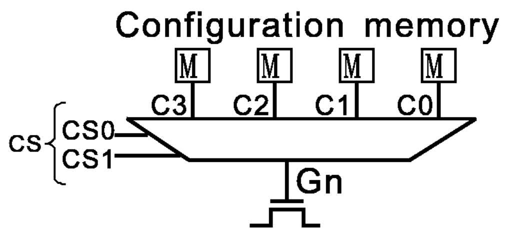

Fig. 3. Equivalent circuit of multi-context switch (4 contexts).

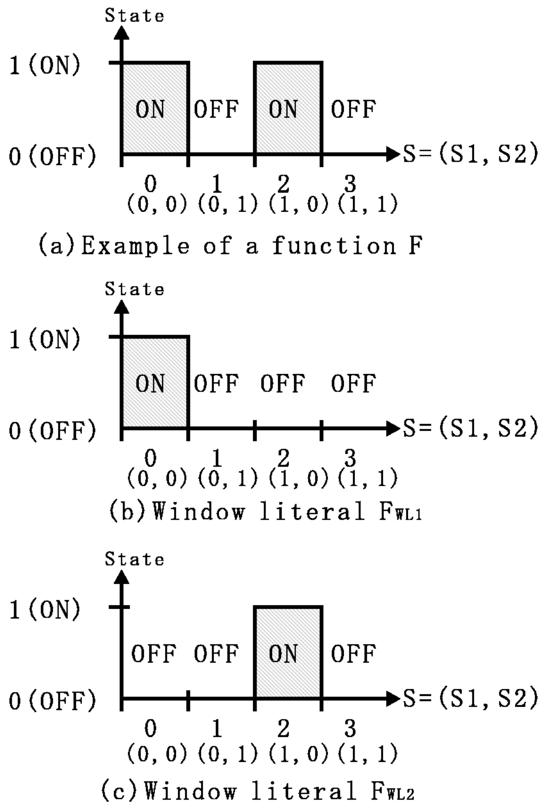

Fig. 4. Function of the multi-context switch (4 contexts).

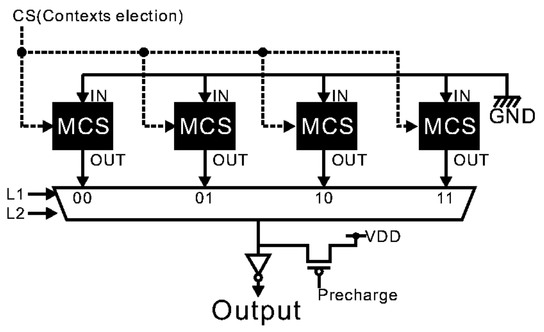

Fig. 5. Block diagram of the MC-LUT.

block, the area of the logic block, and the delay of the logic block. The higher functionality of the logic block requires the larger number of inputs and outputs of a logic block. This increases the complexity of the switch block. On the other hand, the higher functionality of the logic block reduces the number of the logic blocks required for implementing a target function, and reduces the total delay. Based on this observation, the functionalities of the fine-grain logic block are specified as follows.

- 1-bit addition with carry storage.
- Arbitrary logic function of two inputs.
- 1-bit storage.

The specification allows to perform a bit-serial addition on a single cell without carry ripple, and the high functionality can be achieved with the minimum number of inputs and outputs of the logic block. This results in further reduction of interconnection complexity. Figure 2 shows the block diagram of the cell. A cell consists of a logic block (LB) and a switch block (SB). The context selection signal CS is commonly distributed to the multi-context LUT(MC-LUT), and multi-context(MC) switch, but it is omitted to be drawn to simplify the figure. The LB consists of an MC-LUT, a carry generator (CG), an XOR gate, a pipeline register (D-FF) and a 2-to-1 multiplexer. Use of the small-size LUT increases the utilization of hardware resource. The CG consists of a 2-in-1 MUX and a D-FF area-efficiently based on the inverting and symmetrical property of the full-adder function[4]. The CG has a dedicated control signal, RST that indicates word boundaries of bit-serial inputs. The LB has three modes: logic, delay and arithmetic modes, one of which is selected by a signal $MODE$. For example, in the arithmetic mode, the LB is configured as an adder or a subtracter for bit-serial operations. The LUT is configured as an XOR gate, which generates a sum output of a half adder, and the output carry generator is selected by $MODE$. In the case of a single context, the performance of the fine-grained bit-serial architecture is more than two times higher than that of the conventional FPGA architecture under a constraint of the same chip area[2].

Let us explain the MC switch that is the most essential unit in the switch block and the MC-LUT. Figures 3 and 4(a) show an equivalent circuit and an example of the function of the MC switch for 4 contexts, respectively. The context signal CS=(CS0, CS1) is used to select the configuration bit G_n. The configuration bits $C0$,$C1$,$C2$,$C3$ are stored in the configuration memory M in advance. The configuration memory M is programmed to 0 or 1 in advance, and One of C0, C1, C2, and C3 is selected as the configuration bit corresponding to $(S1, S0)$=$(0,0)$,$(0,1)$,$(1,0)$ and $(0,0)$, respectively. The MC switch generates an arbitrary ON/OFF pattern according to the context selection signal. For the case of N contexts, the MC switch requires the configuration memory with N bits and the context selection signal with $log_2 N$ bits.

Figure 5 shows the block diagram of the MC-LUT, where the pre-charge-evaluate circuit is used. The input $L1$ and $L2$ are used as input variables of the logic function implemented by the MC-LUT. When the context selection signal is given, the ON/OFF state of each MC switch is determined, that is, configuration bit of each MC switches is determined. The output of the MC-LUT corresponding to the $(L1, L2)$ is selected from the configuration bits.

III. Implementation of an MC switch using an FGMOS functional pass gate

Let us consider the function F shown in Fig. 4(a). The function can be given by *OR*-ing the functions F_{WL1} (Fig.4(b)) and F_{WL2}(Fig.4(c)) called window literals. Given $S1$ and $S2$ $(S1 \leq S2)$, a window literal is defined as follows.

$$F_{WL}(S, S1, S2) = \begin{cases} 1 & S1 \leq S < S2 \\ 0 & otherwise \end{cases}$$

The function F is given by *OR*-ing 2 window literals as follows:

$$F(S) = F_{WL}(S, 0, 1) + F_{WL}(S, 2, 3) \qquad (1)$$

For the case of N contexts, the function of an MC switch can be given by *OR*-ing $N/2$ window literals at most.

Let us consider the window literal F_{WL2} shown in Fig. 6(a). The window literal can be *AND*-ing the functions F_{UL} and F_{DL} called “up-literal” and “down-literal” respectively. An up-literal is a monotone increasing function as shown in Fig. 6(b). Given the threshold value T, an up-literal $F_{UL}(S, T)$ is given by

$$F_{UL}(S, T) = \begin{cases} 1 & T \leq S \\ 0 & otherwise \end{cases}$$

A down-literal is a monotone decreasing function as shown in Fig.6(c). Given the threshold value T, a down-literal $F_{DL}(S, T)$ is given by

$$F_{DL}(S, T) = \begin{cases} 1 & S \leq T \\ 0 & otherwise \end{cases}$$

Hence, the window literal F_{WL2} is expressed as

$$F_{WL}(S, 2, 3) = F_{UL}(S, 2) \cdot FL_{DL}(S, 2)$$

Finally, Eq.(1) can be rewritten as

$$\begin{aligned} F(S) \;\; = & \;\; F_{UL}(S, 0) \cdot FL_{DL}(S, 0) \\ & + F_{UL}(S, 2) \cdot FL_{DL}(S, 2) \end{aligned}$$

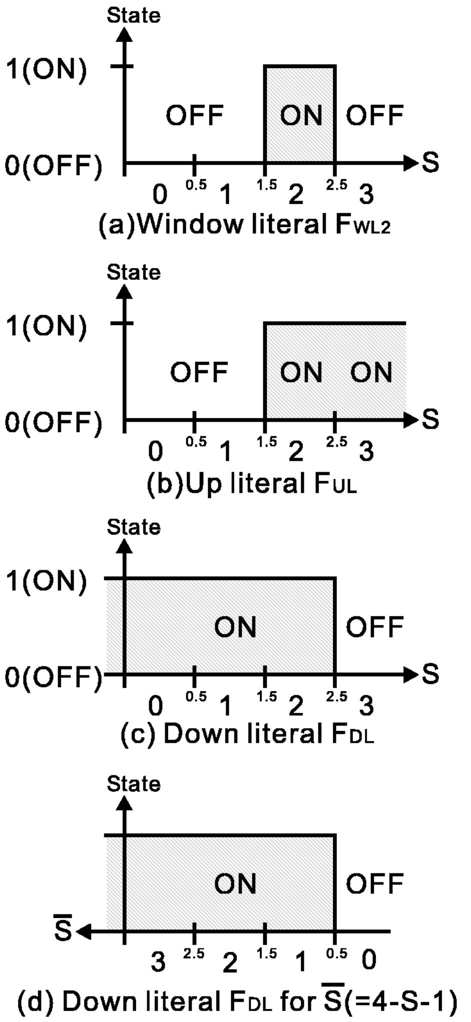

Fig. 6. Decomposition of a window literal into an up-literal and a down-literal(4 contexts).

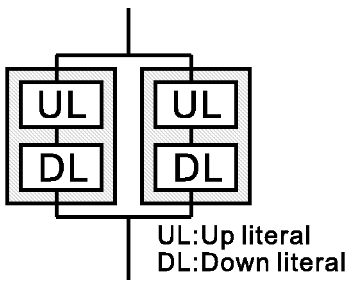

Fig. 7. Circuit of an MC switch (4 contexts).

From this equation, the circuit of the MC switch for N contexts is provided as shown in Fig. 7. The function of the 4-context MC switch is generated by wired-*OR*-ing the outputs of 2 window literals. In general, the function of the N-context MC switch is generated by wired-*OR*-ing the outputs of less than $N/2$ window literals. The output of each window literals is generated by wired-*AND*-ing the outputs of an up-literal and a down-literal. If we use the FGMOS as a 4-valued device, each of an up-literal and a down-literal is implemented by a single FGPG[3] where an FGMOS is used not only as a storage device but also pass transistor. The threshold value of an up-literal or a down-literal is programmed by

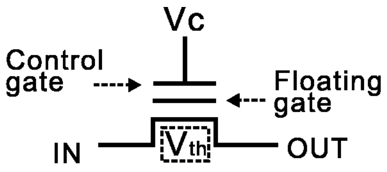

Fig. 8. Floating-Gate MOS transistor.

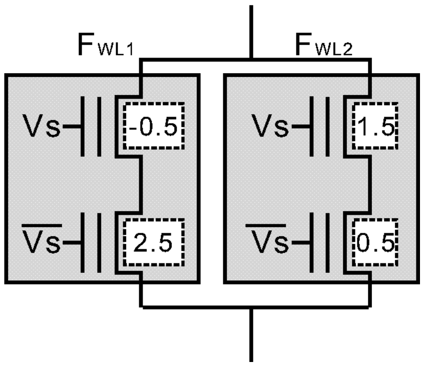

Fig. 9. Implementation of the MC switch using FGFPs.

injecting a controlled amount of electrons into the floating gate. Figure 8 shows the symbol of the FGFP, where threshold voltage is denoted by V_{th} bounded by a broken line. Figure 9 shows the implementation of the switch function F shown in Fig.4, where V_S and $\bar{V}_S$ denotes the control-gate voltages corresponding to S and $\bar{S}$. In N-valued logic, $\bar{S}$ is defined as $N - S - 1$. Hence, $\bar{S} = 3 - S$ for $N = 4$. Note that the down literal F_{DL} for S(Fig.6) can be implemented by the up literal for $\bar{S}$ as shown in Fig.6. By using the FGFP, the transistor count of a multi-context switch(MC-switch) is reduced to 10% in comparison with SRAM-based one. The transistor count of an MC-switch is also reduced to 20% in comparison with the MC-switch that uses floating-gate MOS transistors just as storage device.

IV. Implementation and Evaluation

As a first step, the proposed MC-FPGA is implemented for 2 contexts using an FGMOS as a binary device. Figure 7 shows the MC switch for 2 context. Figure 11 shows the micrograph of the MC-FPGA designed using a 0.35μm EPROM process of Chartered semiconductor manufacturing. The chip contains 4×4 cells, and circuit for writing test on $5mm \times 5mm$ chip. The supply voltages at operating and writing modes are 3.3V and 6.0V, respectively. Figure 12 shows the layouts of the proposed cell and the SRAM-based cell. By using the FGMOS functional-pass gate, the cell area is reduced to 56% of the SRAM-based cell.

V. Conclusion

This paper presents a novel MC-FPGA architecture. The key technologies are the fine-grained architecture to increase the utilized ratio of LUTs, and an FGMOS functional pass gate to implement an MC switch area-efficiently. As a next step, multiple-valued version is undergoing. If we use the FGMOS as a multiple-valued device, the proposed MC-FPGA becomes more efficient in area.

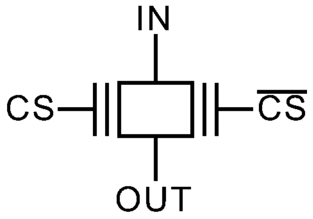

Fig. 10. MC switch for 2 contexts using FGMOSs.

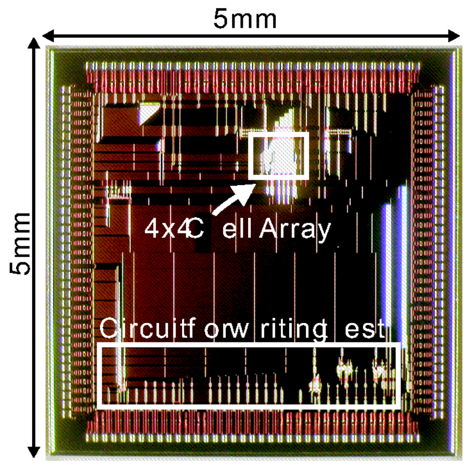

Fig. 11. Micrograph of the proposed MC-FPGA.

References

[1] A. Dehon. Dynamically programmable gate arrays: A step toward increased computational density. In *the Fourth Canadian Workshop on Field-Programmable Devices*, pages 47–54, 1996.

[2] M. Kameyama M. Hariyama, W. Chong. Field-programmable vlsi based on a bit-serial fine-grain architecture. *IEICE Trans. Electron.*, E87-C(11):1897–1902, 11 2004.

[3] M. Kameyama T. Hanyu. Multiple-valued logic-in-memory vlsi architecture based on floating-gate-mos pass-transistor logic. *IEICE Trans. Electron.*, E82-C(9):1662–1668, 9 1999.

[4] Katarzyana Leijten-Nowak. An fpga architecture with enhanced datapath functionality. In *FPGA03*, pages 195–204, 2003.

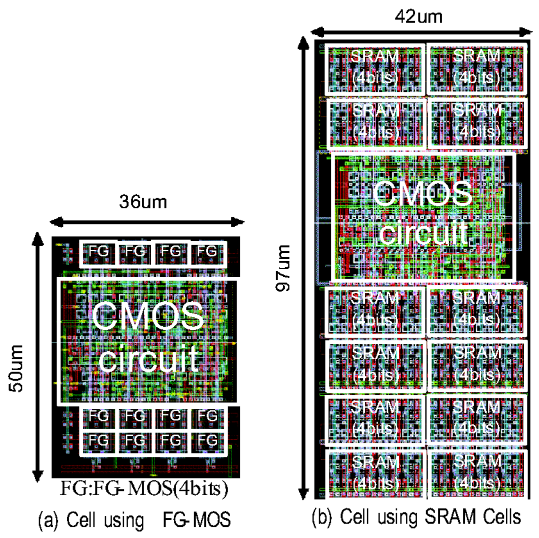

Fig. 12. Layouts of MC-FPGA cell using the FGFP and SRAM cells.