Novel Switch Block Architecture Using Non-Volatile Functional Pass-gate for Multi-Context FPGAs

Masanori Hariyama, Weisheng Chong, Sho Ogata and Michitaka Kameyama
Graduate School of Information Sciences
Tohoku University
Aoba 6-6-05, Aramaki, Aoba, Sendai, 980-8579, Japan
hariyama@ecei.tohoku.ac.jp

Abstract

Dynamically-programmable gate arrays (DPGAs) promise lower-cost implementations than conventional FPGAs since they efficiently reuse limited hardware resources in time. One of typical DPGA architectures is a multi-context one. 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 reconfigurable architecture called reconfigurable context memory (RCM) is presented based on the fact that there are redundancy and regularity in configuration bits between different contexts. A floating-MOS functional pass-gate, where storage and switch functions are merged, is used to construct the RCM area-efficiently.

1 Introduction

Dynamically-programmable gate arrays (DPGAs) provide more cost-effective implementations than conventional FPGAs where hardware resources are dedicated to a single context[1],[2]. A DPGA can be sequentially configured as different processors in real time, and efficiently reuse the limited hardware resouces in time. One of typical DPGA architectures is a multi-context one. Multi-context FPGAs (MC-FPGAs) have multiple memory bits per configuration bit forming configuration planes for fast switching between contexts. However, the additional memory planes cause significant overhead in area and power consumption [3]. Especially, switch blocks require a much larger memory capacity than look-up tables.

Figure 1 shows the overall structure of an MC-FPGA. Each cell consists of a programmable logic block and a programmable switch block. Figure 2 shows the structure of a conventional multi-context switch. The switch has multiple memory bits for multi-contexts and its contexts are selected from the memory bits according to a context ID. In the conventional approach, each switch requires n bits to store n contexts. Most previous works for DPGAs reduce the overhead using only device-level solutions. That is, compact memory devices such as DRAM were used to store configuration data [1].

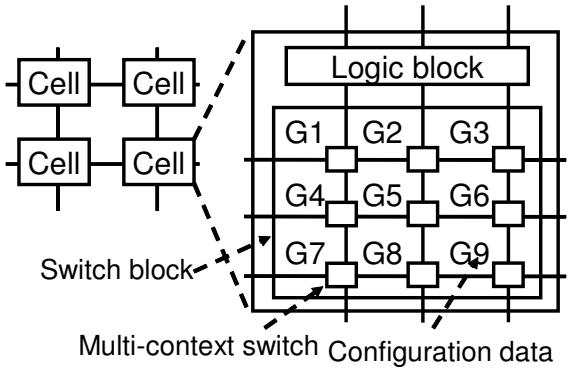

Figure 1. Overall structure of an MC-FPGA

To reduce the overhead of configuration memory in MC-FPGAs, this paper proposes an architectural-level solution based on the fact that there are redundancy and regularity in configuration bits between contexts. To illustrate the redundancy and regularity, Table 1 shows an example of configuration data of the switch block shown in Fig. 1. Each row denotes configuration data of each switch. The configuration data G3 and G9 have redundancy in themselves. That is, there is no change in their configuration bits. It is said that less than 3% of configuration data are changed when contexts are switched [4]. There is another type of redundancy between configuration data of different switches. For example, G2 and G4 have the same configuration data. Moreover, there is regularity in configuration data such as G2 and G4. The configuration data G2 and G4 can be represented by repeating bits in an order of (0,1). To exploit the

Proceedings of the IEEE Computer Society Annual Symposium on VLSI
New Frontiers in VLSI Design
0-7695-2365-X/05 $20.00 © 2005 **IEEE**

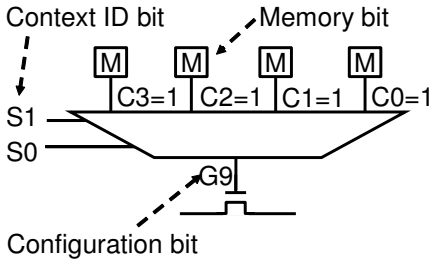

Figure 2. Conventional multi-context switch (four contexts)

Table 1. Redundancy and regularity in configuration data

	Context 3 (C3)	Context 2 (C2)	Context 1 (C1)	Context 0 (C0)
G1	0	0	0	1
G2	1	0	1	0
G3	0	0	0	0
G4	1	0	1	0
G9	1	1	1	1

redundancy and regularity, a reconfigurable context memory is proposed based on a floating-gate MOS functional pass-gate (FGFP). The FGFP is the device that can merge a logic operation and storage in a single floating-gate MOS transistor[5]. An arbitrary switch function can be decomposed into switch functions called "window literals", each of which is efficiently implemented by using the FGFP. The number of the window literals corresponds to the number of configuration bits for the switch function. In the FGFP-based RCM, the number of window literals can be flexibly changed by reconfiguring the FGFP network connection. By using the FGFP-based RCM, the number of transistors of switch block can be reduced to 10% in comparison with conventional SRAM-based switch block. The use of FGFPs will be efficient in static power in comparison with the SRAM-based implementation because no supply voltage is required to keep the storage.

2 Switch Block Architecture Using the Reconfigurable Context Memory

Redundancy and regularity in configuration data can be used to reduce the area of the context memory. In this paper, an architecture with four contexts is considered as an example although our approach is also applicable to architectures with other number of contexts. Contexts are switched by a 2-bit context ID (bit S1 and bit S0) as shown in Table 2.

Figure 3 shows configuration-bit patterns that are independent from the context ID because the switch is programmed to be always turned on or off. A single memory bit is sufficient to control the switch, while four memory bits are required for the conventional switch shown in Fig. 2. Figure 4 shows configuration-bit patterns that depend on a single context-ID bit. Note that each bit pattern is same as the bit patterns of S1 (or $\overline{S1}$) or S0 (or $\overline{S0}$) shown in Table 2. A switch using a single context-ID bit is smaller than the conventional switch which uses two context-ID bits. The other configuration-bit patterns depend on S1 and S0. Each bit pattern can be generated using a 2-to-1 multiplexer. The multiplexer is slightly larger than the hardware shown in Figs. 3 and 4. However, the bit patterns are not frequently used in a multi-context architecture since less than 3% of configuration data change when contexts are switched [4].

Table 2. Relations between contexts and context ID bits

	Context 3	Context 2	Context 1	Context 0
S1	1	1	0	0
S0	1	0	1	0

Configuration bit (G)				Hardware generation of G
Context 3 (C3)	Context 2 (C2)	Context 1 (C1)	Context 0 (C0)	
0	0	0	0	
1	1	1	1	

Figure 3. Configuration-bit patterns that are independent of a context ID

Figure 5 shows an MC-FPGA architecture that uses reconfigurable context memory (RCM) as switch blocks. A logic block is connected to a RCM block. RCMs are connected by 2 types of interconnections: single-length lines and double-length lines to achieve both of flexibility and high speed data transfer. Single-length lines connect neighbor RCMs. Double-length lines connect RCMs every 2 RCMs. To simplify the figure, only single-length lines are illustrated. The single-length lines allow flexible connections between RCMs, but it may decrease speed of data transfer between distant LBs. The double-length lines allow high-speed data transfer between distant LBs through less RCMs.

Proceedings of the IEEE Computer Society Annual Symposium on VLSI
New Frontiers in VLSI Design
0-7695-2365-X/05 $20.00 © 2005 **IEEE**

Configuration bit (G)				Hardware generation of G
Context 3 (C3)	Context 2 (C2)	Context 1 (C1)	Context 0 (C0)	
0	0	1	1	G= $\overline{S1}$
0	1	0	1	G= $\overline{S0}$
1	0	1	0	G= S0
1	1	0	0	G= S1

Figure 4. Configuration-bit patterns that depend on a single context-ID bit

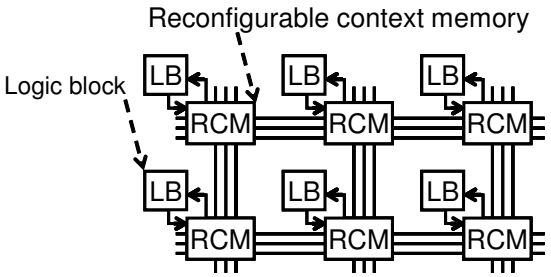

Figure 5. MC-FPGA architecture using the reconfigurable context memory as switch blocks

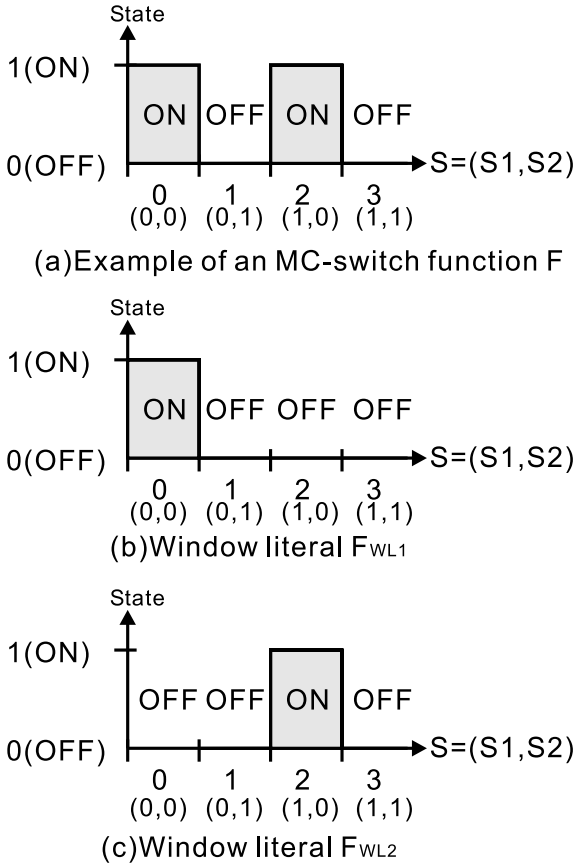

Figure 6. Function of the multi-context switch (4 contexts).

3 Implementation of an RCM using an FGPG

Let us consider the function F shown in Fig. 6(a). The function can be given by *OR*-ing the functions F_{WL1} (Fig.6(b)) and F_{WL2}(Fig.6(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. 7(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. 7(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.7(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

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

Proceedings of the IEEE Computer Society Annual Symposium on VLSI
New Frontiers in VLSI Design
0-7695-2365-X/05 $20.00 © 2005 **IEEE**

From this equation, the circuit of the MC switch for N contexts is provided as shown in Fig. 8. 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[5] 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 injecting a controlled amount of electrons into the floating gate. Figure 9 shows the symbol of the FGFP, where threshold voltage is denoted by V_{th} bounded by a broken line. Figure 10 shows the implementation of the switch function F shown in Fig.6, 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 $F_{DL}forS$(Fig.7) can be implemented by the up literal for $\bar{S}$ as shown in Fig.7. Based on this observation, we propose the RCM as shown in Fig. 11. To simplify the figure, only 4 tracks are illustrated. The RCM consists of three types of FGFPs. The FGFPs with the control-gate voltage V_S and $\bar{V_S}$ are used to implement up-literals and down-literals, respectively. The FGFPs denoted by small squares are used to connect(or disconnect) horizontal and vertical tracks. This type of FGFPs can implement the configuration-bit patterns with high redundancy (Fig. 3) area-efficiently. For example, implementation of MC-switches shown in Fig.12 is denoted by thick lines in Fig.11. For the case of 4 contexts and 4 tracks, the number of transistor is reduced to about 10% and 60% in comparison with the SRAM-based implementation and the FGFP-based implementation without redundancy.

4 Conclusion

This paper presents novel switch block architecture for multi-context FPGAs. The key technologies are architecture to exploit redundancy between context data and FGFPs to implement configuration circuits area-efficiently. The use of FGFPs will be efficient in static power in comparison with the SRAM-based implementation because no supply voltage is required to keep the storage. The implementation of the test chip is undergoing using 0.35μ EPROM technology.

References

[1] A. Dehon, "Dynamically Programmable Gate Arrays: A Step Toward Increased Computational Density", in Proc. the Fourth Canadian Workshop on Field-Programmable Devices, pp. 47-54(1996)

[2] S.M.Scalera and J.R.Vazquez, "The design and implementation of a context switching FPGA", in Proc. IEEE Symposium on FPGAs for Custom Computing Machines, pp.78-85(1998)

[3] S. Trimberger, et al. "A Time-Multiplexed FPGA", in Proc. of FCCM, pp. 22-28(1997)

[4] I. Kennedy, "Exploiting Redundancy To Speedup Reconfiguration of An FPGA", in Proc. FPL, pp. 262-171(2003)

[5] T. Hanyu, 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(1999)

[6] Katarzyana Leijten-Nowak, "An FPGA Architecture with Enhanced Datapath Functionality", in Proc. FPGA03, pp.195-204(2003)

Proceedings of the IEEE Computer Society Annual Symposium on VLSI
New Frontiers in VLSI Design
0-7695-2365-X/05 $20.00 © 2005 **IEEE**

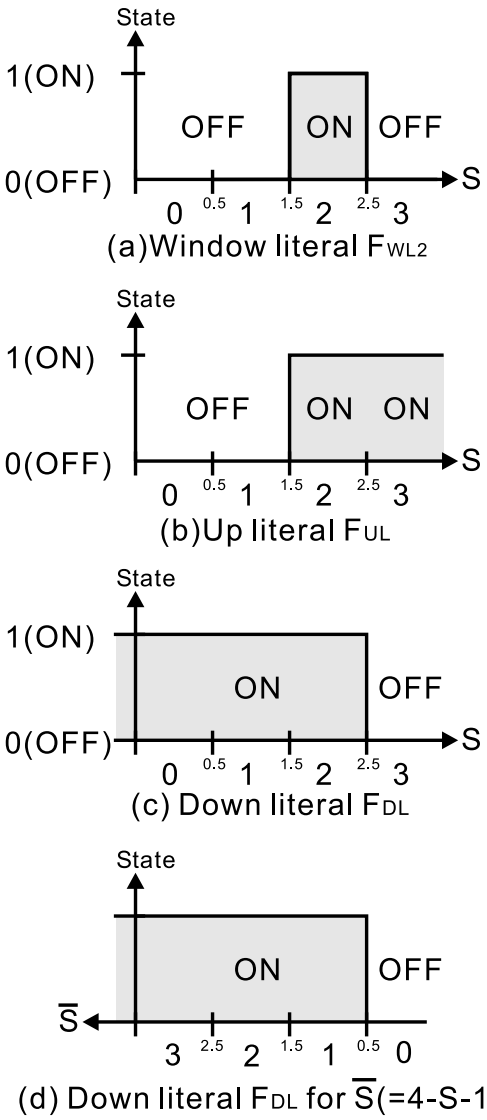

Figure 7. Decomposition of a window literal into an up-literal and a down-literal(4 contexts).

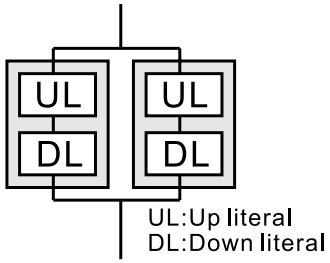

Figure 8. Circuit of an MC switch (4 contexts).

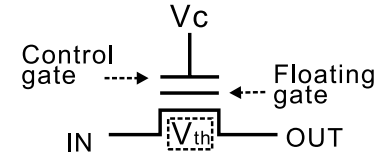

Figure 9. Floating-Gate MOS transistor.

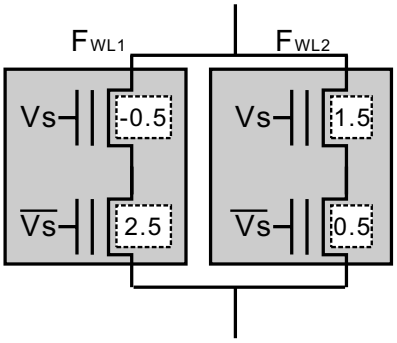

Figure 10. Implementation of the MC switch using FGFPs.

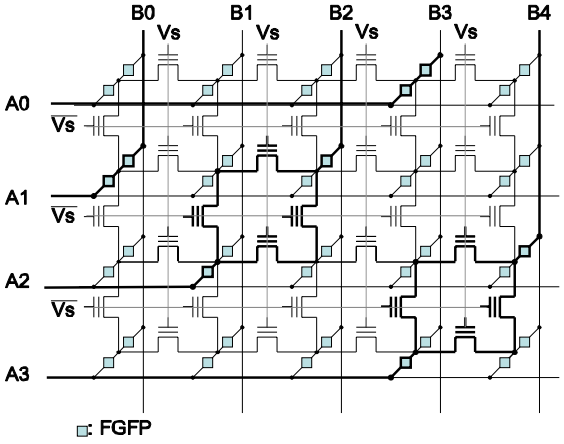

Figure 11. RCM using FGFPs.

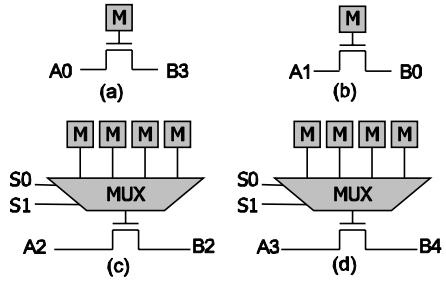

Figure 12. Example of MC switches.

Proceedings of the IEEE Computer Society Annual Symposium on VLSI
New Frontiers in VLSI Design
0-7695-2365-X/05 $20.00 © 2005 **IEEE**