Development of Memory Mapped Register and Bus Switch Circuit for Hardware Virtualization in GEDY

Sota Kono, Satoshi Yonemitsu , Akira Yamawaki*

Kyushu Institute of Technology, Kitakyushu, 804-8550 Japan

Corresponding Author: yama@ecs.kyutech.ac.jp

Abstract

Truly general purpose microcomputer which can be applied to any kinds of electronic products is difficult to develop. Because truly general purpose microcomputer has problems such as circuit scale, power consumption, unit price, and so on. GEDY (General purpose microcomputer with Dynamic partial reconfiguration) is a general purpose microcomputer architecture using dynamic partial reconfiguration of circuits. The GEDY can deal with many embedded systems. The virtual GEDY has all peripheral devices on the uniform memory space including their own memory mapped registers. Developers have only to write the traditional microcomputer program that cooperates with the peripheral devices via the memory mapped registers on virtual GEDY. The physical GEDY realizes the function of virtual GEDY by dynamically partial reconfiguration for the peripheral devices on the reconfiguration unit. To realize such execution behavior, the reconfiguration unit on physical GEDY needs to be virtualized for many kinds of peripherals used on the virtual GEDY. This virtualization must be performed without conscious of circuit switching for programmers. This paper develops the virtualization hardware mechanism, transparent memory mapped register (tMMR) and flexible bus switch (FBS) circuit. The tMMR and FBS are circuits for hardware virtualization in the physical GEDY. Experimental results show that the developed tMMR and FBS perform their functionalities correctly with unified pin assignment.

Keywords: DPR, hardware virtualization, FPGA.

1. Introduction

Microcomputers (MCUs) are used in many kinds of the embedded devices. Developers of the microcomputer need to meet the requests from the makers of the embedded

products. Tailoring to each application leads to the higher development cost. However truly general purpose microcomputer which can be applied to any kinds of electronic products is difficult to develop. Because, truly general purpose microcomputer has problems such as circuit scale, power consumption, unit price, and so on. GEDY (General purpose microcomputer with dynamic partial reconfiguration) is the general purpose microcomputer architecture using dynamic partial reconfiguration (DPR) ⁽¹⁻⁴⁾ of circuit. GEDY can deal with many embedded systems. The virtual GEDY has all peripheral devices on the uniform memory space including their own memory mapped registers. Developers have only to write the traditional microcomputer program that cooperates with the peripheral devices via the memory mapped registers on virtual GEDY. The physical GEDY realizes the function of virtual GEDY by dynamically partial reconfiguration the peripheral devices on the reconfiguration unit. To realize such execution behavior, the reconfiguration unit on physical GEDY needs to be virtualized for many kinds of peripherals used on the virtual GEDY. This virtualization must be performed without conscious of circuit switching for programmers. This paper develops the virtualization hardware mechanism, transparent memory mapped register (tMMR) and flexible bus switch (FBS) circuit.

2. Overview of GEDY

2.1 Structure of GEDY

Fig. 1 shows an overview of the physical GEDY organization. The physical GEDY consists of CPU, RCP, MMR, FBS, RP, and OFCM.

Fig. 1. Physical GEDY Organization

The CPU is a processor to manage the whole of the physical GEDY. It also handles the dynamic partial reconfiguration on the reconfigurable parts (RPs). The CPU reconfigures the dedicated RP by storing the circuit configuration bitstream into the reconfiguration port (RCP).

The tMMR is a characteristic component of the physical GEDY. The tMMR decouples the on-chip interconnect and the reconfigurable circuit reconfigured on a RP by the unified memory mapped register interface. Thus, the circuit designer have only to consider the simple memory mapped registers when designing the interface to the CPU. From the viewpoint of the CPU, the CPU has only to access the memory mapped registers to manage the RP similar to the conventional I/O peripherals. In addition, the memory map for each hardware to be configured on a RP are remapped according to the hardware swapping. Thus, the CPU can uniformly access the hardware through the fixed memory map even if many hardware modules are dynamically on the different RPs.

The FBS is a circuit for switching the connection with the external peripheral circuit according to the circuit of the RPs. In the physical GEDY, Peripheral circuits are configured on different RPs. Regardless of which RP the peripheral circuit is configured, the bus must be connected to the real peripheral. The FBS realizes switching of bus by multiplexing input and output with control signal.

“the mailbox (MB) is a register file to communicate any data between the reconfigurable part and the CPU. The mailbox hides the complexity of the interconnect among the RP and CPU by the simple readable/writable register file. The CPU has only to load/store the data to the MB to manage the hardware module on the RP. The hardware module has only to read/write the dedicated registers in the MB to execute following the request of the CPU. The base address of the MB can be remapped dynamically on the MMR according to the hardware module to be reconfigured into the RP as mentioned above.

The off-chip memory (OFCM) holds many bitstreams

Fig. 2. MMR Organization

(a) Read Control

(b) Write Control

Fig. 3. State Transition Diagram

to be configured on the RPs. The other useful data can be stored into the OFCM.”⁽⁵⁾

3. Hardware Virtualization

3.1 Memory Mapped Register

MMR is a register in which circuit data is mapped in a unified memory. The processor and the circuit of

reconfigurable part are separated by MMR. Therefore, when designing the interface to the CPU, the programmer only need to consider the memory map. In other words, the programmer only needs to write a conventional program that cooperates with peripheral devices via the memory map.

Fig. 2 shows the overview of the MMR organization. The MMR has AXI_BRIDGE circuit and ADDR_DEC circuit. The bus with the processor is using the AXI bus. The AXI bus has read address channel, read data channel, write address channel, write data channel and write response channel.

AXI_BRIDGE is used when the processor accesses memory. AXI_BRIDGE performs read control and write control. Fig. 3. Shows a state transition diagram of read control and write control.

In the read control, there are three states, IDLE, READ_REG, and SEND_READ_VAL. IDLE state is waiting for operation start signal. When the processor sends a read request, the state transitions to READ_REG. READ_REG state saves the read data in a register. When the reading is completed, the state transitions to SEND_READ_VAL. SEND_READ_VAL state returns the read data to the processor. When the processor receives the data, the state returns to the IDLE state.

In the write control, there are five states, IDLE, RESP_AWREADY, RECB_REG_VAL, WRITE_REG, RET_RESP. IDLE state is waiting for operation start signal. When the processor sends a write request, the state transitions to RESP_AWREADY. RESP_AWREADY state waits for response of receiving write address. RECV_REG_VAL state saves the write data in a register. WRITE_REG state writes the data. RET_RESP state waits

Fig. 4. FBS Organization

Fig. 5. GEDY-2 Organization

for a processor write response. When the processor returns a response, the state returns to the IDLE state.

ADDR_DEC converts the address accessed from the processor to the MB number. Data can be exchanged with MB by ADDR_DEC.

3.2 Flexible Bus Switch

FBS is a circuit that switches the connection between the external device and the RPs. In GEDY, Peripheral circuits are configured on different RPs. However, the bus connected to the physical real peripheral is fixed. Regardless of which RP the peripheral circuit is configured, the bus must be connected to the real peripheral. The FBS switches the connection of the bus to the real peripheral device. FBS is realized the bus connection switching by multiplexing FBS receives the control signal from the MMR and multiplexes the external input and the internal output.

Fig. 4. Shows the overview of the FBS organization. As shown in Fig. 4, the FBS can be connected to the bus between the peripheral circuit on the RPs and corresponding real peripheral device by the control signal from the MMR.

4. Prototype of GEDY, GEDY-2

4.1 Overview of Prototype

We develop the prototype of GEDY (GEDY-2) for verify the functionality of the MMR and FBS. We implemented GEDY-2 to the FPGA (XC7Z010CLG400-1) on the Digilent zybo board. Fig. 5 shows an overview of the GEDY-2.

Table. 1. Utilization of FF in the FP

Module		FF
GEDY2_TOP	Zynq7 processing System	0
	Axi_hwicap	952
	Axi_interconenct	730
	Proc_sys_reset	40
MMR	ADEC0	5
	ADEC1	5
	MMR_AXI	133
FBS		13
TOTAL		1878

Table. 2. Utilization of LUT in the FP

Module		LUT
GEDY2_TOP	Zynq7 processing System	112
	Axi_hwicap	532
	Axi_interconenct	675
	Proc_sys_reset	25
MMR	ADEC0	1
	ADEC1	1
	MMR_AXI	341
FBS		4
TOTAL		1691

In order to check the operation of MMR and FBS, we designed a circuit that uses memory mapped LED and SW and a serial communication circuit.

4.2 Implementation of Prototype

Table.1 shows the utilization of FF in the FP. Table. 2. Shows the utilization of LUT in the FP. As shown in Table.1 and Table. 2, the amounts of FF and LUT in MMR and FBS are small. Further, the total resource utilization of the FP is about 5.3 % for FF and about 9.6 % for LUT. That is, since the resource utilization of MMR and FBS is small, it hardly affects the resource amount allocated to the RP.

4.3 Operation verification of Prototype

First, using the LED lighting circuit to check the operation of the MMR. The LED lighting circuit has a SW register at address 0x43c00000 and an LED register at address 0x43c00004. Confirm whether it can communicate data with the MB of the LED lighting circuit on the RP only by memory access. Therefore, the program of the processor assigns the address of SW to the address of the LED (Fig. 6).

Secondary, using a serial communication circuit to check the operation of the FBS. Fix port for serial communication and configure serial communication circuit to RP0 and RP1. Whether serial communication can be performed regardless of which RP is configured (Fig.7).

Fig. 6. Confirmation of MMR operation

Fig. 7. Confirmation of FBS operation

5. Conclusion

This paper has developed MMR and FBS for hardware virtualization in GEDY. We verified the operation on the developed prototype system and confirmed that the MMR and FBS circuits operate correctly. Resource utilization of MMR and FBS is small, influence on resources in FPGA is small.

This paper has developed the developed the hardware section of the virtualization architecture. However, with this alone, programmers cannot freely write programs and operate. In future work, we will develop software to manage the hardware unit developed this paper.

References

- (1) M.Taher and T.El-Ghazawi : "Virtual Configuration Management: A Technique for Partial Runtime Reconfiguration", IEEE Transactions on Computers, vol.58, No.10, pp.1398-1410, 2009
- (2) M.Majer and J.Teich : "The Erlangen Slot Machine: A Dynamically Reconfigurable FPGA-based Computer", Journal of VLSI Signal Pro-cessing, vol.47, No.1, pp. 15-31,2007
- (3) Yosuke Shiga, Haruki Imanaka, Takahumi Ueno : "PCI interface technology by FPGA of dynamic reconfigurable Hw/Sw mixed system", Information Processing Society of Japan Vol.2002, No.5, pp. 85-90, 2002

- (4) Shuhei Igari, Junji Kitamichi, Yuichi Okuyama :
“Implementation of Multi-accelerator type dynamically reconfigurable processor”, The Institute of Electronics Information and Communication Engineers, Vol.110, No.360, pp.163-168, 2011
- (5) Sota Kono, Gen Nishizawa, and Akira Yamawaki :
“Development of Reconfiguration unit for General Purpose Microcomputer with Dynamic Partial Reconfiguration”, International Conference on Intelligent Systems and Image Processing, 2016