A design technology for re-configurable MPU and software on FPGA

H.Araki[1] K.Harashima[2] T.Kutsuwa[2]
[1] Graduate School of Engineering [2]Faculty of Engineering,
Osaka Institute of Technology
5-16-1, Omiya Asahi-ku, Osaka, 535-8585 Japan
e-mail araki@kesakoy.elc.oit.ac.jp, kutuwa@elc.oit.ac.jp

Abstract: FPGA is the necessary device to design of hardware at present, it is researched on many ways of applying to design caused by expansion of capacity in recent years. One of these applying ways is SoC (System on a Chip) that is proposed for realizing the basic function of a system. For realizing SoC efficiently, IP (Intellectual property) is very important and developed for re-use of hardware. A MPU for built-in exists as an IP. But almost of MPUs at present as an IPs are lengthy and lar ge-scale for using embedded-application. Furthermore, the function of executing specific treatment critically is required to embedded MPU. We propose a flexible and small scale MPU and its design method.

1. Introduction

At present, the research for applying FPGA is more active. And the possibility of applying FPGA is expanded with expansion of FPGA capacity. There are branches to research, "For Description", "For Synthesize", "For Design Device" and so on. In this paper, we propose about "For Description". Many approach are tried to many application. One of the researches is an approach to include a MPU to FPGA. In this research, some approaches are tried and applied.

1) FPGA has a MPU as a hardware macro.
(Fig.1 (a)) (Hardware macro cannot be changed.)
2) FPGA has a MPU as a software macro.
(Fig.1 (c)) (Software macro can be changed a little.)
3) FPGA has a MPU as a description with other functions.
(Fig.1 (c)) (This MPU can be changed in full.)

We aim at the 3rd method. Because the MPU that is made by the method is most extensive and possible. But applying the MPU, there are some advantages and some faults.

- We can build best MPU for an application. (advantage)
- We can use the hardware efficiently. (advantage)
- We can get a necessary speed. (advantage)
- To build the MPU is difficult. (fault)
- There is no software development tool for the MPU.(fault)

To conquer the faults, we propose that specification description language expresses the MPU. The proposing includes following that.

Describe the MPU and the other hardware by the specification description language.
Describe the software for the MPU by the specification description language.
Improve the system re-configuration by these descriptions.

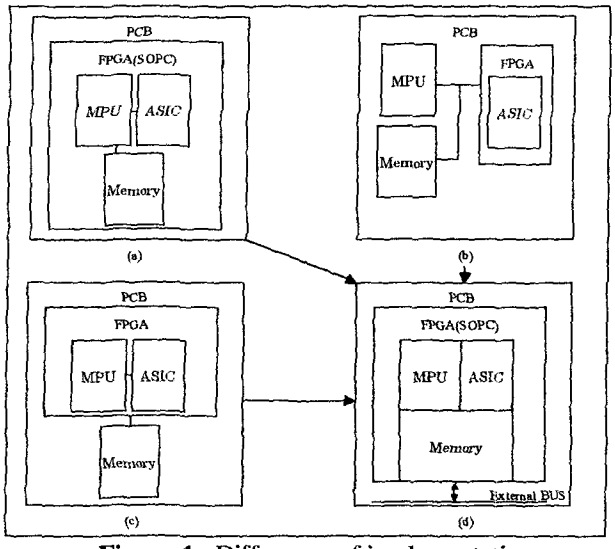

Figure 1 Difference of implementation..

2. Technology

As described above, expansion of capacity and speed-up of FPGA are realized in recent years. And, the basic composition of logic-cell is based on SRAM technology. Furthermore, logic-cells for memory and random logic are mounted separately. The effective method of characteristic of separate logic-cells is that functional implementation is executed by not hardware-logic but software-logic. However, the implementation of critical function is executed effectively by hardware-logic than software-logic. The way of implementation is decided when specification are described. To use a specific description language, we can change easily the way. It is a good technique for designing re-configurable system with a FPGA. For the small-scale system, there are often several changes of specifications and requirement, it is important that a change can be easily. So, we implement a re-configurable MPU in FPGA, and apply it to a programmable controller. Furthermore, MPU has structural modules, and the modules are able to change independently. Fig.1 illustrate the ways of design with MPU in FPGA. We propose the structure of Fig1.(d) from them. We think that the structure of FPGA is efficiently and does not lose flexibility. In FPGA, the MPU accedes to the ASIC by asynchronous single direction

registers to do without bus-arbitration. But the ASIC can connect an internal bus of MPU by necessity. We can select the way to accede from MPU to the ASIC with considering a tradeoff between forwarding speed and transaction speed. We offer the frame like the above. With applying the frame, the software and the hardware can convert efficiently between each other. The MPU that is re-configurable structure and expressed by specifications description language has following characters.

1) Modifying the command decoder block, a hardware that is specific for applications can be controlled as an additional command from software.
2) Adding some ALU, the MPU can calculate with parallel.
3) The MPU can obtain a specific calculation unit by changing an ALU.
4) External unit can access the register file directly.
5) External unit can access the program counter unit directly.

Applying the characters, MPU application system can be designed easily and quickly. Besides, the MPU can be applied and extended flexible than other MPU that is distributed as IP. This program for the MPU is written by specifications description language. So the system that includes a program can be modified uniformly. And a program memory for the MPU is on FPGA, there is a single controller on a printed circuit board (PCB).

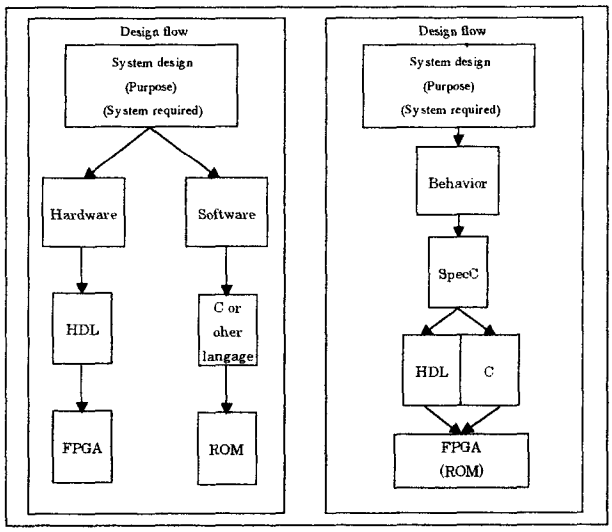

Figure 2 Description flow.

We use "SpecC" which is a specifications description language to realize this method. With applying "SpecC", we can change a system to a request by a minimum modification. "SpecC" is based on "ANSI C language" and surpasses. On the behavior level evaluation, we can use "C language code" is generated from "SpecC language code". Fig.2 illustrate how difference between usual design flow and our propose design flow. The proposal can be confirmed delaying the decision to implement software and hardware. To design a re-configurable system, it is important. The codes that are described in "SpecC" for the re-configurable system are divided to HDL (Hardware Description Language) codes and C program codes. Each code generate each object file by each compiler. Finally, the object file for programming to FPGA is generated by all object files.

We develop a computer simulator based on this design technique, and we experiment with this simulator.

3. Experiment

We develop a system controller for experiment of the evaluation that is 8Bit MPU with motor control unit on this simulator.

This experiment system performs following movement.

(1) A user indicates a speed from general-purpose input.
(2) The detection of the speed is realized by measuring electromotive force of the motor.
(3) The control of the motor is performed so that indicated speed and a measuring speed may become the same.
(4) The motor speed is controlled as a motion with constant angle acceleration.

Our MPU is designed suitable configuration for FPGA. And the MPU's inside is composed of function modules. Function modules have command-decoder, command-controller, register-files, program-counter, stack-unit, ALU (Arithmetic Logic Unit), command-bus, data-bus, and so on.

The command-bus connecting program-memory and command-decoder reads a program at the address indicated by program-counter.

The command-decoder analyzes a program which is read-out on command-bus.

The command-controller takes control of the other modules according to instruction of command-decoder.

The register-files write data on data-bus to any register of oneself and output data on any register of oneself to data-bus according to command-controller.

The program-counter is increasing the PC-register (program counter register) or set a value of the data-bus to PC-register according to command-controller.

The stack-unit records the value of PC-register according to command-controller.

The ALU outputs to data-bus the value that is calculated according to inside register and data-bus.

The data-bus is connected to register-file, ALU, and program-counter.

The modules exchange a value using the data-bus.

The motor control unit executes control of speed by PWM. This control unit is connected to register-file of MPU, and it controls motor at the non-synchronism. MPU can access the control unit through the three registers that are speed control register, timer register and setup register. The value written into the registers reflects on the system in real time except for the timer register. The motor control unit has the input of 8Bit, and the value reflects on the register of MPU in real time.

Additionally, MPU has general-purpose I/O of 8bit, and it is connected to the register of MPU directly.

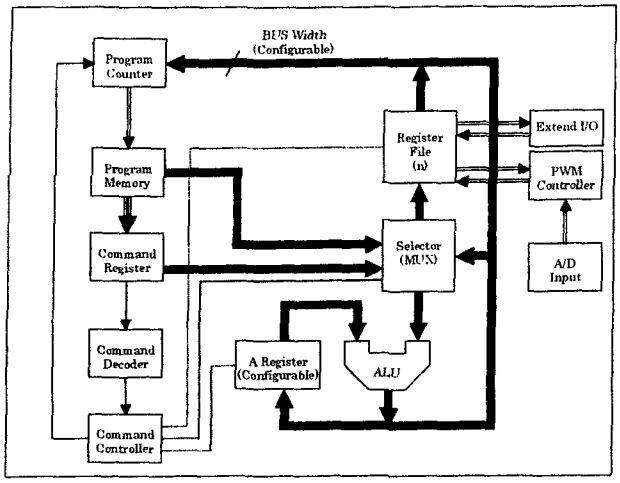

Figure 3 MPU block for experiment.

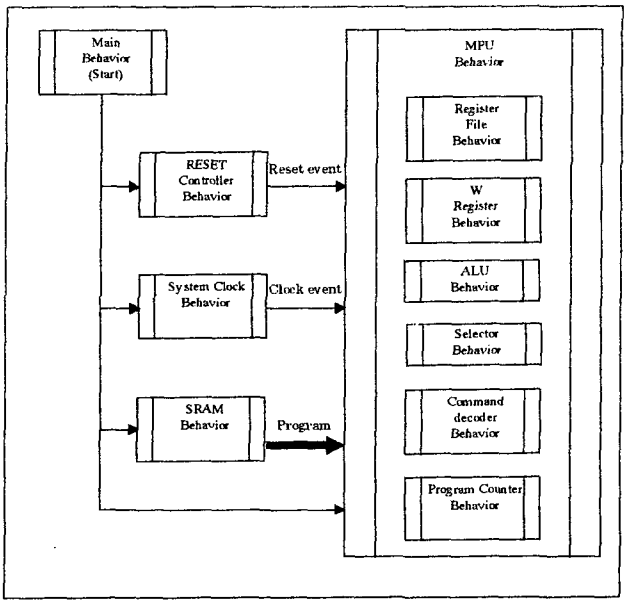

Figure 4 Execution of simulation flow.

Next, we express about software to mount on the system.

(1) All registers are initialized.
(2) MPU reads the value of indicated speed (Wi) from general-purpose I/O, and sets a value to MPU register.
(3) MPU reads the value of detected speed from motor control unit, and sets a value to MPU register (Wd).
(4) PWM is controlled following the rules.
(Wd-Wi) > 0 (accelerate)
(Wd-Wi) >8 (revolution accelerate increase by 1)
(Wd-Wi) <8 (revolution accelerate increase by(Wd-Wi)/2)
(Wd-Wi) = 0 (fixed speed)
(Wd-Wi) < 0 (deceleration)
(Wd-Wi) >-8 (revolution accelerate decrease by 1)
(Wd-Wi) <-8 (revolution accelerate decrease by(Wd-Wi)/2)
(5) MPU calculates the revolution acceleration (Aw).
(6) MPU writes the value of (Wd+Aw) to PWM register (Wo).
(7) Repeat (2) to (6).

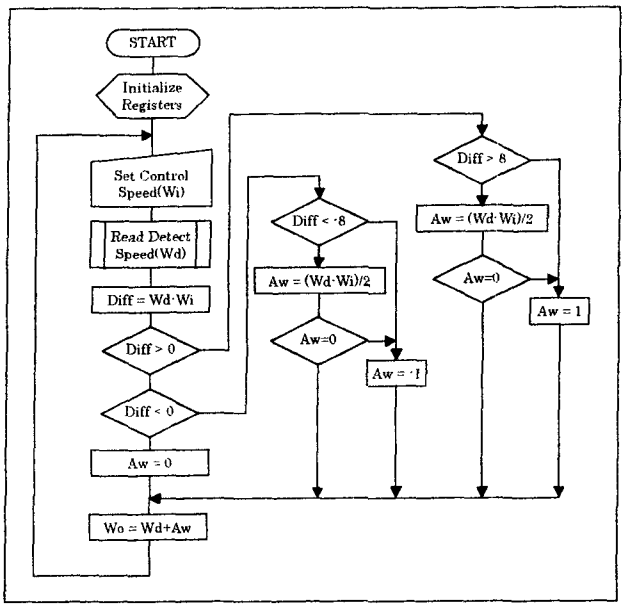

Figure 5 Control program chart for experiment.

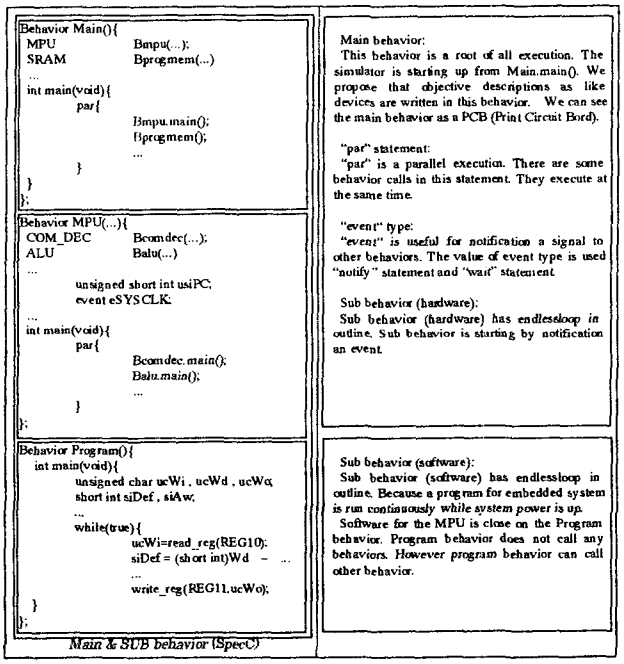

Figure 6 Description examples of behavior.

We implement hardware and software of the specifications to the MPU emulator. We make two models. One is mounted acceleration calculating function by hardware. The other is calculation module is mounted by software. And we simulate about these two models.

We explain about the modules and their description for experiment. The modules are written for "SpecC reference compiler V1.2". This compiler is built a "C++ source code" from "SpecC source code". The simulator written by

"SpecC" starts from main function of main behavior The system which is built for experiment starts from this main function. The following, we explain the descriptions about the MPU and the software for the MPU. The codes that are the experiment system written by SpecC are compiled to the codes that are written by C++ owing to SpecC compiler. Usually we compile the C++ code to generate a simulation object code. However we cut off the codes about the program for the MPU. The codes for the program are compiled to generate object codes for the MPU. The other codes are compiled to generate simulator codes. The simulator loads a object code for the MPU, and executes the code. When to build programming file for FPGA, the object codes for the MPU merge these files. The two codes are independent of each other and there are no problems to compiling and executing. At the SpecC, description of a behavior is closed and it is easy to make software a capsule.

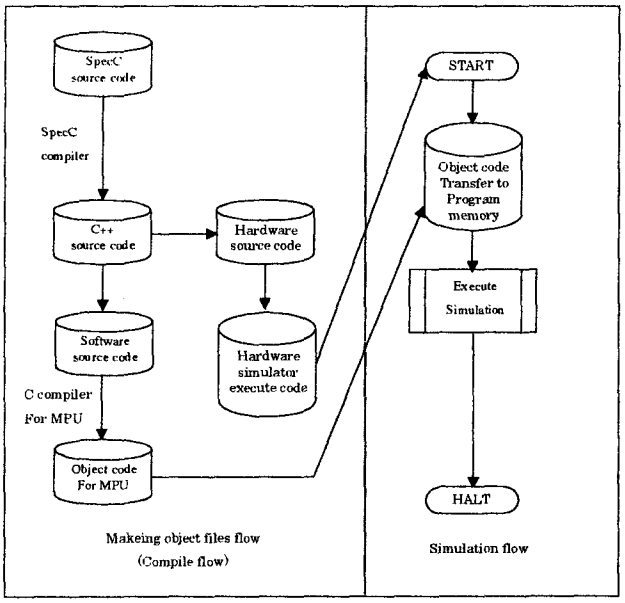

Figure 7 Simulation executing flow.

The simulator behaves as hardware, the software is executed by the MPU on the simulator. The hardware has some independence modules and the modules are executed independent on the simulator. The modules look like behaviors styles. But they are described with RTL and synchronous each other. The synchronous signals and data paths are through the union behavior. When the modules are added or changed, we change the union behavior and describe controls for the module. As the above, we show how to design a application system and design with the MPU.

Next experiment, we compare the descriptive styles of the MPU and each MPU that has a same style but is described by VHDL. At first, it is impossible that to describe hardware system include software by VHDL. If there is software described by VHDL, it must be converted to C language. Although that, the differences of the execution to describe the languages appear strikingly. When to describe software by VHDL, we must describe carefully to avoid the problem. At the following explanation, we do not compare the descriptive style for software. Accordingly, We compare only re-configurable MPU description. To evaluate about the descriptive styles, we compare our MPU with a MPU that is decripted by VHDL. VHDL MPU is composed of some component block. On our proposed design method, all control signals and data path are through a union behavior for re-configurable MPU. They are through root architecture in described VHDL as well. We add an ALU to each MPUs. For this addition, the code volume of SpecC is increase about 200bytes. The code of VHDL is increase about 400Bytes as well. Further, the number of parts to change necessary are 3 parts for SpecC, and 8 parts for VHDL. In our purposed design method, adding functions are complete changing main behavior. The description of SpecC is abstractly and we can complete changing smaller than VHDL.

Table 1 Increase of description for adding ALU.

	SpecC (Main behavior)	VHDL (root Architecture)	note
Line	257	152	All line number of files.
Size	+200Kbyte	+400Kbyte	To add ALU
Change	3	8	Points. (Not Line)

4. Experiment result and consideration

We have shown and described how to compose MPU that is built by modules. It has been shown that the change and addition of each module is easily executed. We have changed the trade-off relations between the performing speed by the hardware and the description-easiness by the software. Furthermore, we have confirmed that we can realize a user demand in consideration of this trade-off. In this paper, we have expressed that our proposed MPU and its design technique are effective.

References

[1] D. Gajski, J. Zhu, R. Domer, A. Gerstlauer, and S. Zhao, Spec C: Specification Language and Methodlogy, *Kluer Academic Publishers*, 2000.

[2] K. Wakabayashi, T. Okamoto,"C-Based Soc Design Flow and EDA Tools: An ASIC and System Vendor Perspective" in Proc. IEEE Trans. Computer-Aided Design, Vol.19, No.12, 2000, pp.1507-pp.1522.

[3] K. Okada, A. Yamada, T. Kambe,"Hardware Alogorithm Optimization Using Bach C" in Proc. IEICE Trans. Fundmentals, Vol.E85-A, No.4, 2002, pp.835-pp.841.