FPGA Implementation of a Multi-Core System Architecture for Power Adaptive Computing

HUANG Le-tian, Fang Da
University of Electronic Science .and Technology of China
huanglt@uestc.edu.cn, fangdaraul@163.com

Abstract — By analysing the system requirements of Power Adaptive Computing System, a Multi-core system is designed. The system has one master core for management and four slave cores for computing. A sharing bus structure and a special communication mechanism are designed. Based on the sharing bus and the communication mechanism, master CPU could schedule the tasks and control each slave CPU core. The architecture is simulated by modelsim and is verified based on FPGA. The result shows that the system the whole process of can successfully complete from tasks assigned to the result returned in this system.

Keywords — Multi-Core Architecture, FPGA, Power Adaptive Computing, Bus structure

I. INTRODUCTION

Because of the increasingly tight global energy supplies, governments and research institutions around the world are looking for new energy. [1][2] Using a variety of green energy or emerging energy as the source of energy of the electronic system has become the trend of the development of electronic systems.[3][4][5] Unlike traditional power supply such as batteries or DC power supply, emerging energy is not able considered as a stable and sustainable power for the electronic systems. How to protect modern electronic systems to automatically adapt to different power supply a variety of changes and to ensure the stability of the system and meet the requirements that the task should be processed in time should be focused.[4][5][6]

Task scheduling and power consumption conditioning of multi-core systems is the focus of current research, but most of the researchers are seeking for higher energy efficiency and better performance[7][8]. Only a few researchers began to study how to design the electronic system while the maximum output power of the source is time-varying. [9][10] In the other hand, a research work should be verified on a physical platform. So, how to implement a multi-core system which can used to verify different algorithms for power adaptive is one of the most important things.

In this paper, based on the analysis of the needs of power adaptive computing, a multi-core systems architecture is designed. One master RISC CPU core and four slave RISC CPU core are included in this architecture. The architecture of the sharing bus and the communication mechanism are the key points in this paper. Based on the shared bus and the communication mechanism, master CPU could schedule the tasks and control each slave CPU core. In the other hand every slave CPU core could load tasks, read data and return the result of the tasks by itself. The modelsim is used to simulate this system for the procedure from task loading to result returning. This system is also implemented on FPGA, and is s simply verified.

II. SYSTEM REQUIREMENTS ANALYSIS

According to the type of CPU core, a kind of multi-core systems is called homogeneous multi-core system[11] and the other is called heterogeneous multi-core system.[12] By the division of tasks, the cores of heterogeneous multi-core system are designed and optimized for different purpose in order to improve the performance of the system. For the purpose of implementation of power adaptive computing, one CPU core in this system should be used to executive management and scheduling program. However, it is very difficult to design and optimize a special management unit (MU), and is not necessary to that because management and scheduling program can be executed by a common CPU core. So, a common RSIC CPU could be used as a MU to execute.

This system is designed for doing research of power adaptive computing. So, the power consumption of the hardware can be regulated if it is necessary to do. The slave CPU cores which are executing different tasks could be slow down or speed up of the processing speed in order to regulate the power consumption. So, this system must be designed into a global synchronization local asynchronous (GALS) system. In this system, different slave CPU core could be used for different clock frequency. All the slave CPU cores need to increase necessary peripheral module to become independent subsystems, then the system could be more reliable.

In order to manage the system and schedule the tasks, although the MU which is running management program is the same as other RISC CPU cores which are running computing programs, a special communication mechanism and bus structure should be designed. The MU should have the highest priority of communication procedure for controlling the system. So, one Master RISC CPU core which is used to be MU and four Slave RISC CPU cores which are used to processing tasks are designed in this system.

Bus sharing and network on chip (NoC) are two different structure of the on-chip communication system. Bus sharing is a simple way to implement the on-chip communication system among the CPU cores, but the communication efficiency and flexibility is too low because it is too simple. NoC has become the focus of researchers in recent years because of higher scalability, reliability and reusable, but it is very difficult to implement and is no generally accepted conclusion. For this system, the various subsystems are relatively independent. If the tasks could stay in the subsystem long enough and need not be loaded frequently, the improving bus sharing structure is good enough.

In sum, the system should be designed according to the following rules:

[1]. Master-slave communication standard should be used and a master CPU core should be used to be MU to control this system;

[2]. The slave CPU cores need to increase necessary peripheral module to become independent subsystems and could processing the task independently;

[3]. The improving bus sharing structure will be used.

III. THE DESIGN OF MULTI-CORE SYSTEM

System block diagram in Figure 1. Bus sharing structure is used in this System. MU means Management Unit, and is the controller of this system. It is also the scheduler of the tasks. The CPU cores could process the tasks with different power consumption under the scheduling of MU. The CPU cores have necessary peripheral module

to build up subsystem. Wrapper module is the interface of the bus and is used to load tasks and transmit data.

A. Management Unit

MU could schedule the tasks and manage the system under the conditions that the output power is changeable of the sources. By receiving the information about the prediction of the energy source, MU could schedule the tasks and regular the frequent of the clocks of the subsystems for matching the power consumption of the system and output power of the energy sources. MU could control the slave to load tasks and receive the result by using internal bus. It also can transmit other information to control the subsystems. The MU could communicate with other systems by using Ethernet interface.

The bus is followed AMBA 2.0 protocol. MU is the Master of the bus and other module could be seen as peripherals of MU. Other CPU cores should apply to arbiter if they want to access to the bus. DDR2 SDRAM is the external memory of this system, and the tasks and data is stored in the

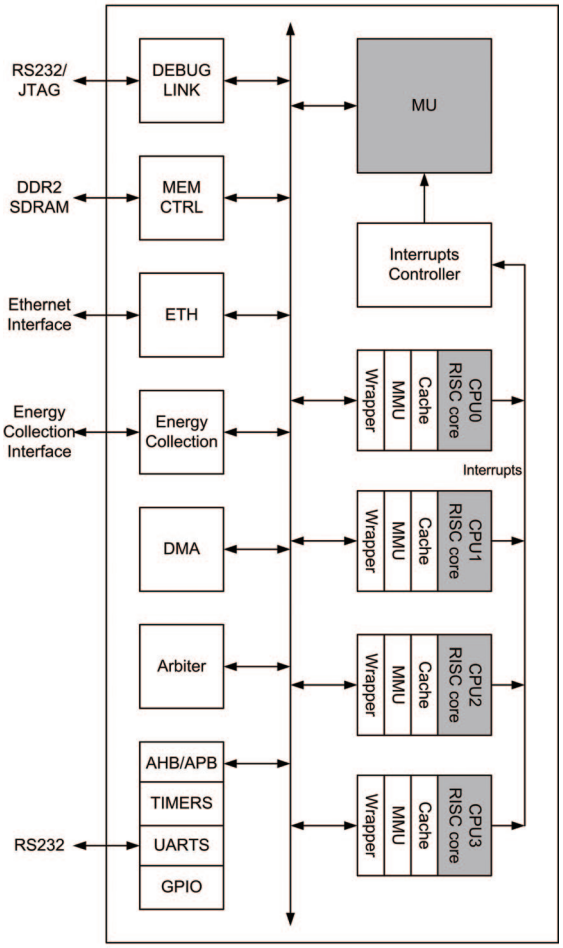

Figure 1. System Block Diagram of This System

memory. The CPU0 to CPU3 could communicate with the DDR2 SDRAM directly by using DMA module. Energy Collection module is the monitor of Power Sources. This module could monitor the change of Power Sources and prediction the trends of power. MU could schedule the tasks and control the whole system followed the information which is got by Energy Collection module. DEBUG LINK is the module which is used to debugging the programmes. This module could be connected with RS232 or JTAG interface, and the user could use this module to monitor each CPU core and the memories. The information could be send out by RS232 or JTAG.

B. Subsystem

A subsystem is build up by a slave CPU core and some necessary peripherals. The block diagram is shown as Figure 2.

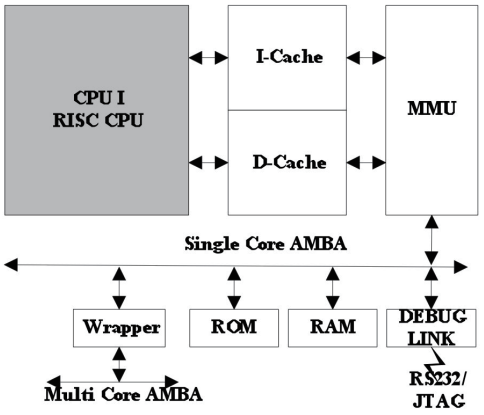

Figure 2. Block Diagram of Subsystem

The subsystem is build up by RISC CPU core, Cache, MMU, ROM, RAM, Wrapper and DEBUG LINK. The subsystem could run programme without any support because it has Cache, MMU, ROM and RAM independently. In other words, the Subsystems have all the characteristics of the smallest computing systems. So, if a task was loaded into the subsystem, it can be executed and need not be loaded frequently. It is very helpful to reduce the needs of bus.

Wrapper is the interface between internal bus of the subsystem and the sharing bus of the whole system. The structure of wrapper will be introduced in part C.

C. Communication Mechanism

Because of the complexity of On-Chip communication, a communication mechanism is defined. The each subsystem is separated from the sharing bus by a wrapper, and the wrapper also can be data buffer of the subsystem. The structure of wrapper is shown as Figure 3

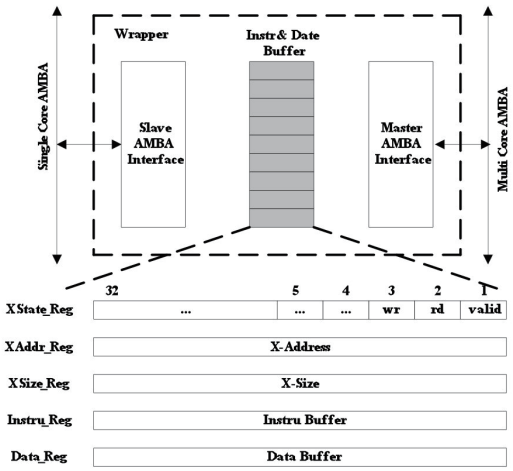

Figure 3. the Structure of Wrapper

The buffer is divided into three types, and the detail of the buffer is shown as Table I.

TABLE I. REGISTER FILE OF BUFFER

Register Name	(Master) W/R	(Slave) W/R	Size
MState_Reg	W/R	R	4Byte
SState_Reg	R	W/R	4Byte
MAddrI_Reg, MSizeI_Reg	W/R	R	4Byte
SAddrI_Reg, MSizeI_Reg	R	W/R	4Byte 4Byte
MAddrD_Reg, MSizeD_Reg	W/R	R	4Byte 4Byte
SAddrD_Reg MSizeD_Reg	R	W/R	4Byte 4Byte
Instru_Reg	W/R	W/R	
Data_Reg	W/R	W/R	

The procedure from loading tasks and returning the result is shown in Figure 4.

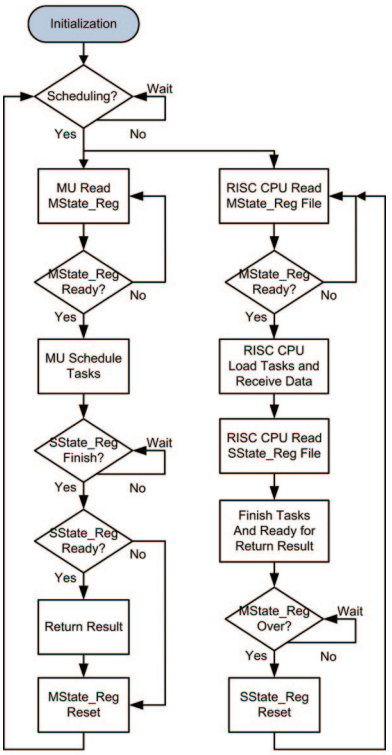

Figure 4. the whole Procedure of Task Scheduling

The register named MState_Reg can only be written by MU, and the other register SState_Reg can only be configured by slave CPU cores. It is very helpful for reducing conflicts of reading and writing of Wrapper.

IV. EXPERIMENTAL RESULTS

The system is simulated by modelsim 6.0 and the simulation results are shown in Figure 5.a and Figure 5.b:

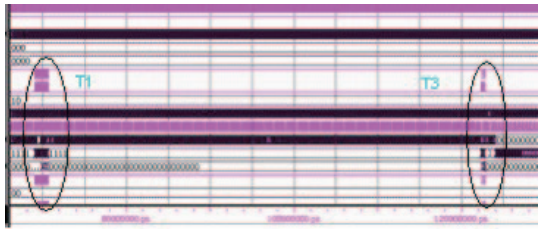

Figure 5.a the simulation results about MU schedule

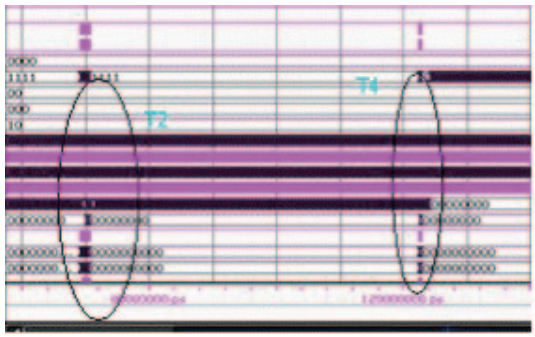

Figure 5.b the simulation results about subsystem loading and executing tasks

The initial software was compiled by Keil and the bin file was used to initial the ROM module by using tools of the software named Quartus II which is designed by Altera Company. The simulation results only show how to load task and return the result in CPU0, because all the subsystem are all the same.

At the time T1, MU sent information to the wrapper of subsystem. CPU0 core read register named MState_Reg at time T2 then got the information from buffer. At the time T4, CPU0

finish the task and return the results, and then the register named SState_Reg was renewed. After that, interrupt signal of MU was triggered by CPU0. MU read the statement about SState_Reg and got the results from the buffer.

The system is implemented based on EP3SE260F1152C2 of DE-3 development board as Figure 6 and the Synthesis Report is shown as Figure 7.

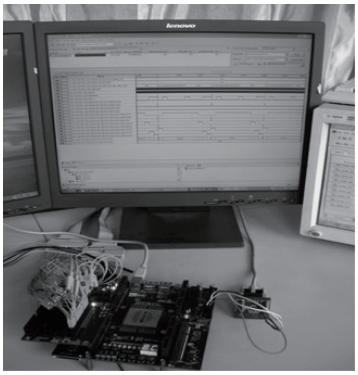

Figure 6. Verification Environment

Fitter Status	Successful - Sun Oct 14 17:41:17 2012
Quartus II Version	9.0 Build 132 02/25/2009 SJ Full Version
Revision Name	arm9_core
Top-level Entity Name	arm9_core_top
Family	Stratix III
Device	EP3SE260F1152C2
Timing Models	Final
Logic utilization	15 %
Combinational ALUTs	19,572 / 203,520 (10 %)
Memory ALUTs	0 / 101,760 (0 %)
Dedicated logic registers	8,513 / 203,520 (4 %)
Total registers	8514
Total pins	7 / 744 (< 1 %)
Total virtual pins	0
Total block memory bits	497,664 / 15,040,512 (3 %)
DSP block 18-bit elements	24 / 768 (3 %)
Total PLLs	2 / 8 (25 %)
Total DLLs	0 / 4 (0 %)

Figure 7. Synthesis Report

CONCLUSIONS

Nowadays, emerging energy is replacing traditional energy. So, to research how to design electronic system when the output power of the energy sources is changeable is a very important significance. Power adaptive computing systems could regulate the power consumption by themselves and could be an important way to solve this trouble. Multi-core system architecture of for power adaptive computing is introduced in this paper. The architecture is simulated by modelsim and is verified based on FPGA. The result shows that the system the whole process of can successfully complete from tasks assigned to the result returned in this system. It could be research platform of power adaptive computing. In the future, the reliability of communications and the corresponding memory consistency problem in the complex case need to research and implementation. In addition, to verify the basis of a variety of scheduling and management algorithm is to improve the system architecture.

ACKNOWLEDGMENT

This work was supported by the Fundamental Research Funds for the Central Universities, National Natural Science Foundation of China (61176025), NLAIC Grants (9140C0901101002 & 9140C0901101003), National Natural Science Foundation of China (61006027), New Century Excellent Talents Program (NCET-10-0297).

REFERENCES

[1]. Mitcheson PD, Yeatman EM, Rao GK, et al, "Energy harvesting from human and machine motion for wireless electronic devices", P IEEE, 2008, Vol:96, pp.1457-1486,

[2]. J.Paradiso and T.Starner, "Energy scavenging for mobile and wireless electronics," IEEE Pervasive Computing, vol. 4, no. 1, pp. 18–27, 2005.

[3]. V. Raghunathan, A. Kansal, J. Hsu, J. Friedman, and M. Srivastava, "Design considerations for solar energy harvesting wireless embedded systems," in Proc. IEEE IPSN' 05, Apr. 2005. pp.457- 462

[4]. Kinget, P. ; Kymissis, I. ; Rubenstein, D. ; Xiaodong Wang ; Zussman, G. "Energy harvesting active networked tags (EnHANTs) for ubiquitous object networking" IEEE Wireless Communications ,2010 ,Volume: 17 , Issue: 6,pp. 18- 25.

[5]. Benjamin Ransford, Jacob Sorber, Kevin Fu "Mementos: system support for long-running computation on RFID-scale devices" ACM SIGPLAN Notices, Apr. 2012.Volume: 47, Issue:4, pp159-170

[6]. A.Kansal, J.Hsu, S.Zahedi, and M.B.Srivastava, "Power management in energy harvesting sensor networks", ACM Trans. Embedded Computing System, 2007, vol. 6, no. 4, pp.32.

[7]. J.-J.Chen, A.Schranzhofer, and L.Thiele, "Energy minimization for periodic real-time tasks on heterogeneous processing units" Parallel and Distributed Processing Symposium, International, pp. 1–12, 2009.

[8]. J.-J.Chen and C.-F.Kuo, "Energy-ef cient scheduling for real-time systems on dynamic voltage scaling (DVS) platforms," 13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, 2007, pp. 28- 38

[9]. Q. Liu, T. Mak, J. Luo, W. Luk, A. Yakovlev, "Power Adaptive Computing System Design in Energy Harvesting Environment", in SAMOS, 2011 PP: 33 – 40

[10]. Kant, K. "Supply and Demand Coordination in Energy Adaptive Computing", 2010 Proceedings of 19th International Conference on Computer Communications and Networks (ICCCN), pp.1- 6, Aug. 2010

[11]. Meilian Xu Thulasiraman, P. "Parallel Algorithm Design and Performance Evaluation of FDTD on 3 Different Architectures: Cluster, Homogeneous Multi-core and Cell/B.E." 10th IEEE International Conference on High Performance Computing and Communications, 2008. HPCC '08. pp. 174- 181

[12]. Tien-Fu Chen, Chia-Ming Hsu, Sun-Rise Wu, "Flexible heterogeneous multi-core architectures for versatile media processing via customized long instruction words" IEEE Transactions on Circuits and Systems for Video Technology, Volume: 15 , Issue: 5, pp.659- 672, May 2005