International Journal of Control and Automation
Vol.8, No.12 (2015), pp.425-434
http://dx.doi.org/10.14257/ijca.2015.8.12.39

Research on the Design of Multi-Core Embedded System Based on Microblaze

Yang Nie [1, 2], Zhenhuan Ma[1] and Lili Jing[1]

[1]*Department of Physics, Jining Normal University, Inner Mongolia, China*
[2]*Digital Engineering Center, Communication University Of China, Beijing, China*
Nieyangwork@163.Com

Abstract

In this paper, MicroBlaze processor is adopted to multi-core embedded systems. The whole system is connected with FSL bus, and the hybrid structure of mutex and mailbox is adopted in the communication between the two cores. Embedded multi-core system is mapped into FPGA, and the feasibility and practicability of the system are verified. Experimental results show that the system not only has good practicability, and has high-speed communication performance between the two processors.

Keywords: *Embedded System, MicroBlaze, Multi-core Communications, FPGA*

1. Introduction

With the rapid development of information technology, the development of embedded system has become increasingly mature. However, as the product power consumption continues to grow, it is an indication that the processing capacity of a single core processor has been close to the limit [1]. As a nuclear treatment technology which is just at the stage of germination, it can reduce the difficulties faced by the single atomic by multi nuclear structure. However, the performance of the multi-core processor is not simply increasing with the increase of the number of cores. The performance of multi-core processor is given in many factors [2], such as the transmission mechanism, the design of the operating system, the software application development. For example, the speed improvement is not noticeable on a dual core processor running on a single processor. This is explained by the fact that in most cases only one processor core is used, and a nuclear communication transmission mechanism is not effectively supported by the multi-core applications. Therefore, the transmission mechanism of multi-core communication has a direct influence on the performance of the multi processor.

At present, there are two kinds of transmission mechanism of multi-core communication [3, 4]. One is cache structure based on a shared bus. Each processor core on the chip shares a two or three cache on the chip, and then achieves the sharing of data and inter-core communication. The advantages of this communication mechanism are simple structure, high communication speed, and the shortcomings of the system are based on the shared bus mode of an embedded system can be expanded. The other is based on chip interconnect structure. Each processing core has its own independent processing unit and storage medium, and each processing core is connected with each other by means of cross switch or on chip bus network. The advantage of this structure is the fact that it can be well expanded, and the data bandwidth is guaranteed. The disadvantage is that the hardware structure is complex and the design process is complex.

The rest of this paper is structured as follows. Section 2 describes the basic principle of MicroBlaze, the architecture of MicroBlaze and Fast Simplex Link (FSL) bus are discussed in detail. Section 3 demonstrates the mechanism of multi-core communication, and the structure of mutex and mailbox is analyzed and studied. In Section 4, MicroBlaze is used to construct a multi-core embedded system, and the multi-core communication

ISSN: 2005-4297 IJCA

mechanism is adopted by mutex and mailbox. Finally, concluding remarks are drawn in Section 5.

2. The Architecture of Microblaze

MicroBlaze is 32-bit RISC Harvard architecture soft processor core with a rich instruction set optimized for embedded applications [5, 6]. The MicroBlaze soft processor solution delivers complete flexibility to select the combination of peripheral, memory and interface features that will give you the exact system you need at the lowest cost possible on a single FPGA. The backbone of the architecture is a single-issue, 3-stage pipeline with 32 general-purpose registers, an Arithmetic Logic Unit (ALU), a shift unit, and two levels of interrupt. This basic design can then be configured with more advanced features to tailor to the exact needs of the target embedded application such as: barrel shifter, divider, multiplier, single precision floating-point unit (FPU), instruction and data caches, exception handling, debug logic, Fast Simplex Link (FSL) interfaces and others. This flexibility allows the user to balance the required performance of the target application against the logic area cost of the soft processor. Figure 1 shows a functional block diagram of the MicroBlaze core. Because MicroBlaze is a soft-core microprocessor, any optional features not used will not be implemented and will not take up any of the FPGA resources.

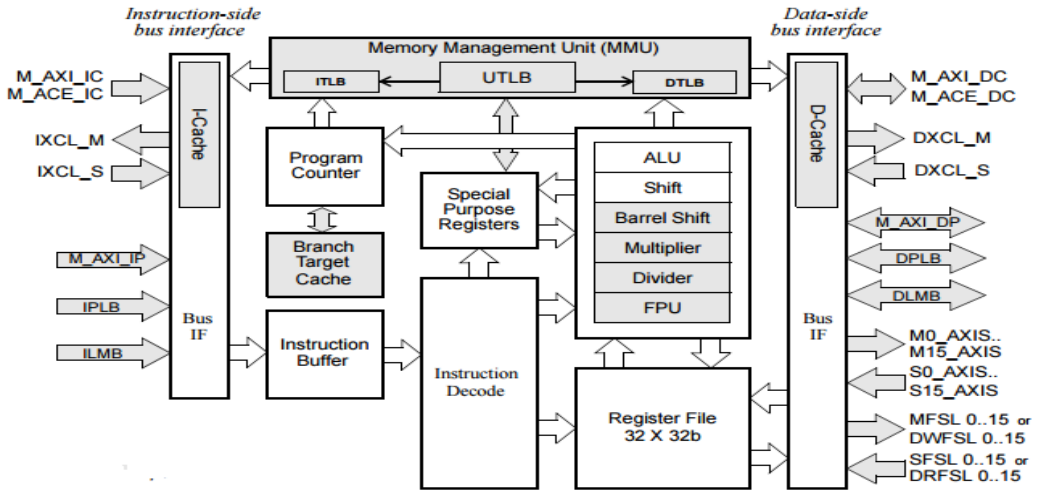

Figure 1. Microblaze Soft Processor Core Block Diagram

The MicroBlaze core is organized as Harvard architecture with separate bus interface units for data and instruction accesses. The following four memory interfaces are supported: Local Memory Bus (LMB), the AMBA AXI4 interface (AXI4) and ACE interface (ACE), the IBM Processor Local Bus (PLB), and Xilinx Cache Link (XCL). The LMB provides single-cycle access to on-chip dual port block RAM. The AXI4 and PLB interfaces provide a connection to both on-chip and off-chip peripherals and memory. The ACE interfaces provide cache coherent connections to memory. The Cache Link interface is intended for use with specialized external memory controllers. MicroBlaze also supports up to 16 Fast Simplex Link (FSL) or AXI4-Stream interface ports, each with one master and one slave interface.

Fast Simplex Link (FSL) bus [7] is a uni-directional point-to-point communication channel bus used to perform fast communication between any two design elements on the FPGA when implementing an interface to the FSL bus. The FSL interface is available on the MicroBlaze processor. The interfaces are used to transfer data to and from the register file on the processor to hardware running on the FPGA. The performance of the FSL

interface can reach up to 300MB/sec. This throughput depends on the target device itself. The FSL bus system is ideal for MicroBlaze-to-MicroBlaze or streaming I/O communications. The FSL bus is driven by one Master and drives one Slave. Figure 2 shows the principle of the FSL bus system and the available signals.

Figure 2. FSL Bus Block Diagram

FSL peripherals may be created as a Master or a Slave to the FSL bus. A peripheral connected to the master ports of the FSL bus pushes data and control signals onto the FSL. The put and get instructions of MicroBlaze can be used to transfer the contents of a MicroBlaze register onto the FSL bus and vice-versa. The FSL bus configuration of MicroBlaze can be used in conjunction with any of the other bus configurations.

3. Inter-Processor Communication Mechanism of Mailbox and Mutex

In a multiprocessor environment, the processors need to communicate data with each other. The easiest method is to set up inter-processor communication. Mailbox and mutex are two main inter-processor communication mechanisms. Mutex core provides a mechanism for mutual exclusion to enable one process to gain exclusive access to a particular resource. The Mailbox core features a bidirectional communication channel between two processors.

The mutex core contains a configurable number of mutex [8]. Each mutex can be associated with a 32-bit user configuration register to store arbitrary data. In a multi-processor environment, the processors share common resources. The mutex provides a mechanism for mutual exclusion to enable one process to gain exclusive access to a particular resource. The mutex core in a typical AXI4-Lite system [9] is shown in the top-level block diagram in Figure 3.

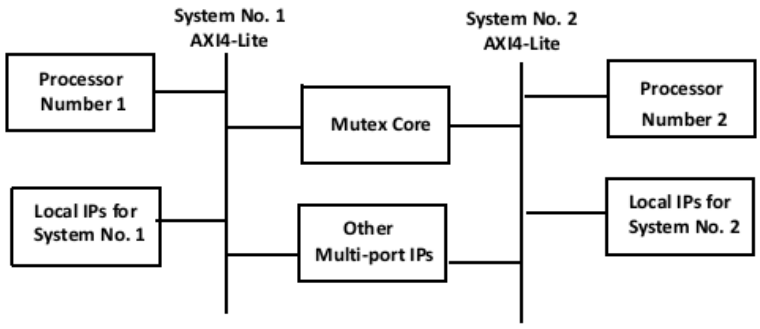

Figure 3. Mutex Core by an AXI4-Lite System

The mutex core adheres to the AXI4-Lite standard defined in the AXI and ACE. The mutex core supports AXI4-Lite interfaces, and the number of interfaces is independently

configured from 0 to 8. The frequency and latency of the mutex core are optimized for use with MicroBlaze. This means that the frequency targets are aligned to MicroBlaze targets.

The mailbox core is used for bidirectional inter-processor communication. A mailbox is a link between two otherwise separate processor systems. Other multi-port IP blocks, such as a memory controller, can also be shared by the two sub systems. In addition to sending the actual data between processors, the Mailbox core can be used to generate interrupts between the processors. The mailbox core in a typical AXI4-Lite system [10] is shown in the top-level block diagram in Figure 4.

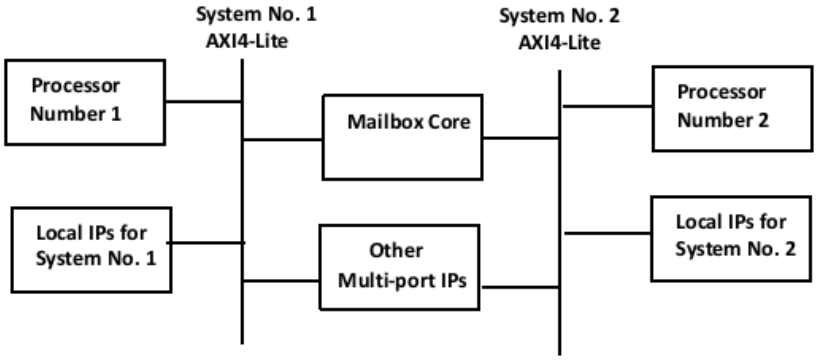

Figure 4. Mailbox Core by an AXI4-Lite System

The mailbox core has two bus interfaces to access the internal resources, usually connected to different processors in a multi-processor system [11]. Both interfaces can be independently configured to use an AXI4-Lite or AXI4-Stream interface. The interfaces are available for connection to any IP that supports them, for example MicroBlaze.

4. Design of Multi-Core Embedded System Based on Microblaz

Embedded system design involves the comprehensive development of software and hardware. In this section, we build a multi-core embedded system based on MicroBlaz using Embedded Development Kit (EDK).

4.1. EDK Introduction and its Development Process

EDK system tools enable you to design a complete embedded processor system for implementation in a Xilinx FPGA device [12]. EDK is a component of the Integrated Software Environment (ISE) Design Suite. ISE is a Xilinx development system product that is required to implement designs into Xilinx programmable logic devices. EDK includes Xilinx Platform Studio (XPS), Software Development Kit (SDK) and Intellectual Property (IP). XPS system tools suite with which you can develop your embedded processor hardware. SDK is based on the Eclipse open-source framework, which you can use to develop your embedded software application. SDK is also available as a standalone program. Embedded processing IP cores include processors and peripherals.

While the EDK environment supports creating and implementing designs, the recommended flow is to begin with an ISE project, and then add an embedded processor source to the ISE project. EDK depends on ISE components to synthesize the microprocessor hardware design, to map that design to an FPGA target, and to generate and download the bit stream. The tools provided with EDK are designed to assist in all phases of the embedded design process [13], as illustrated in Figure 5.

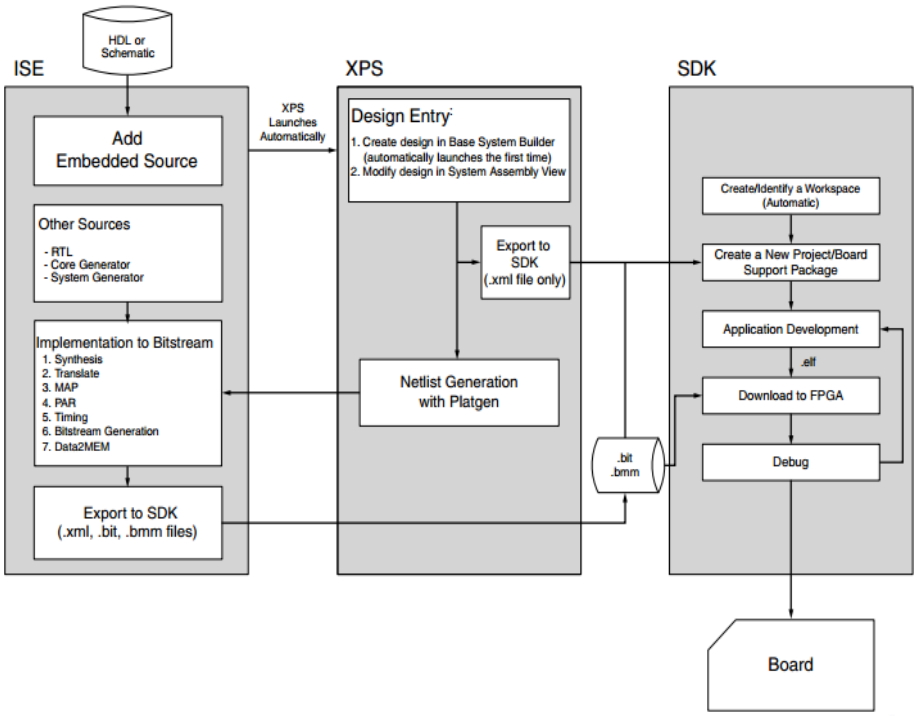

Figure 5. Embedded Design Process Flow with EDK

The hardware platform consists of one or more processors and peripherals connected to the processor buses. XPS captures the hardware platform description in the Microprocessor Hardware Specification (MHS) file. The MHS file is the principal source file that maintains the hardware platform description and represents in ASCII text the hardware components of your embedded system. Microprocessor Software Specification (MSS) file describes the content and configuration of the software platform for a particular processor. Components are instantiated as blocks in the MSS file, and configuration is specified using parameters.

4.2. Multi-Core Embedded System Architecture Design

The multi-core embedded system is constructed by three 32 bit MicroBlaze core, and the overall structure of the system is shown in Figure 6.

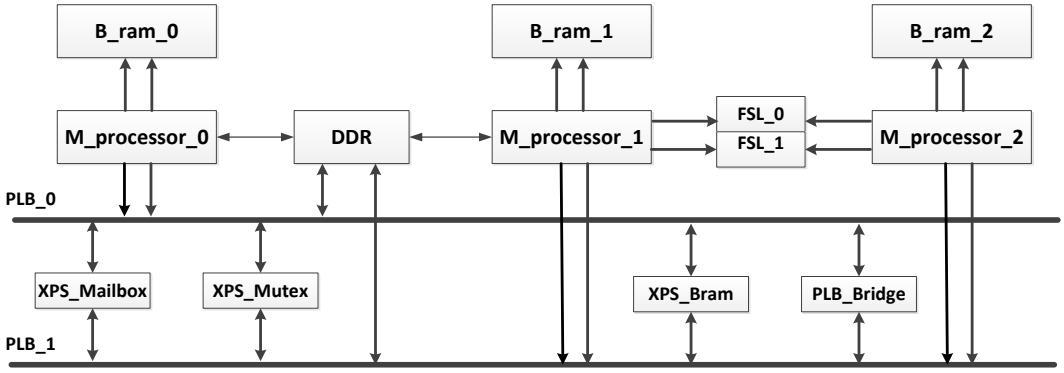

Figure 6. Architecture of Multi-Core Embedded System

In this system, two MicroBlaze complete collaborative work. PLBv46 bus is used to interconnect the multi-core, and mailbox and mutex are used for inter-core

communication mechanism. The data is transmitted to the first core. Next, first cores and second cores complete the communication, and the data is transmitted to the third core by the second core. In the end, the super terminal receives the output of UART and gets the output information. By analyzing and comparing, the system can be debugged and the final result is verified.

4.3. Multi-Core Embedded System Software Design

In the software design of a multi-core processor, any shared memory must be mapped out. Shared code is placed in the BRAM of the system FPGA. The application interface and peripherals are driven by the platform's studio (XPS) device, and the link script files are set reasonably. The system link script files are configured as shown in Table 1. The experimental results [14] show that the configuration can meet the practical requirements of the system and code execution.

Table 1. Configuration Space of Programming Section

Section name	Configuration Storage Location	Description
.text	DDR_SDRAM_SP MD_BASEADDER	Store executable code
.rodata	slmb_cntlr_dlmb_cntlr	Store read-only data
.data	slmb_cntlr_dlmb_cntlr	Storage read and write data
.sdata2	slmb_cntlr_dlmb_cntlr	Storage Read and write data with high access frequency
.bss	DDR_SDRAM_SP MD_BASEADDER	Storage Read and write data of the stack section
.vectors.reset	slmb_cntlr_dlmb_cntlr	Reset segment address space
.vectors.interupt	slmb_cntlr_dlmb_cntlr	Interrupt address space

When the two processors deal with shared resources, the mutex core is used to synchronize the work of the processor, which avoid the conflicts in common resources. When mutex successfully mounted to the system, the program starts with a "#include <xmutex.h>" header file. Using "mutex XMutex", mutex shows a mutual exclusion between mb_1 and mb_0. For the initialization of the mutex kernel, the use of statements is as follows:

“cfg =XMutex_LookupConfig (XPAR_XPS_MUTEX_0_DEVICE_ID);

XMutex_LookupConfig (XPAR_XPS_MUTEX_0_DEVICE_ID);”

Mailbox core is used to transfer information and handshake signals required for communication with two processors. Mailbox is a scheme for transmitting information from one or more sender to a receiver. It forms a channel that is arranged in a FIFO stream from the sender to the receiver via this channel information and then to the receiving end. It can be viewed as a simple TCP/IP form of the two cores. When the message arrives at Mailbox, the Mailbox receiver sends

an interrupt; when the Mailbox is unable to receive the message, the interrupt is invalid. Interrupt controller can make the interrupt invalid, and can choose to open or shut down. So, the sender and receiver of the communication can be synchronous, and also can be asynchronous. When the Mailbox configuration is successful, the system will appear "#include and #include". The former is a Mailbox function library and the latter is the interrupt function library. For the Mailbox initialization, the use of the statement is as follows:

"cfg =XMbox_LookupConfig (MBOX_INTR_ID);

XMbox_CfgInitialize (&mhox, cfg, cf→ BaseAddress);"

4.4. Debugging Of Multi-Core Embedded System

In the system engineering, the bit stream is updated. The bit stream is downloaded to the target FPGA chip. After synthesis and mapping, the application is run. Through the serial port and board debugging, the results can be observed in the super terminal window. Figure 7 and Figure 8 are the results of the display of the processor sending and receiving in the super terminal. Figure 9 and figure 10 is the information display of the inter-core communication by mailbox in the super terminal.

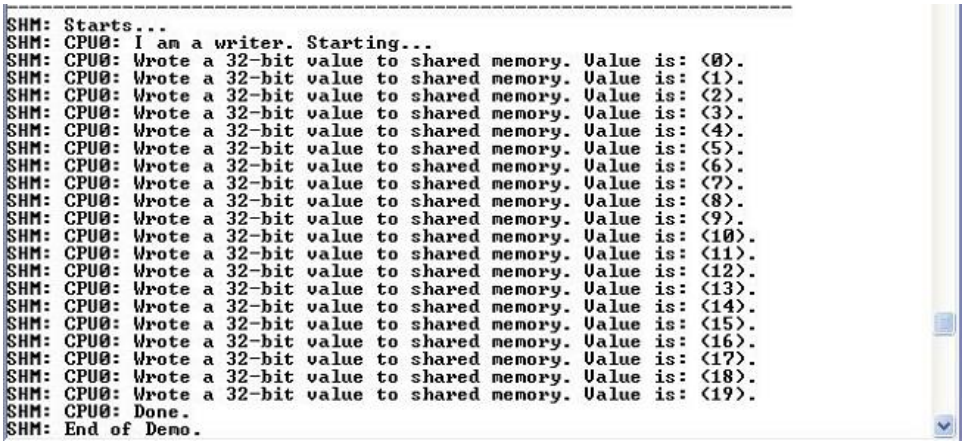

Figure 7. Display Result of Processor Sending Data

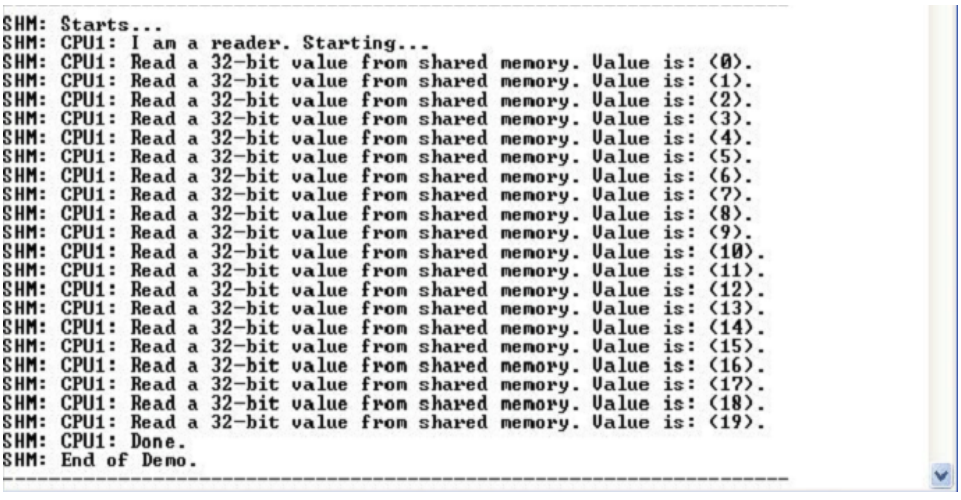

Figure 8. Display Result of Processor Receiving Data

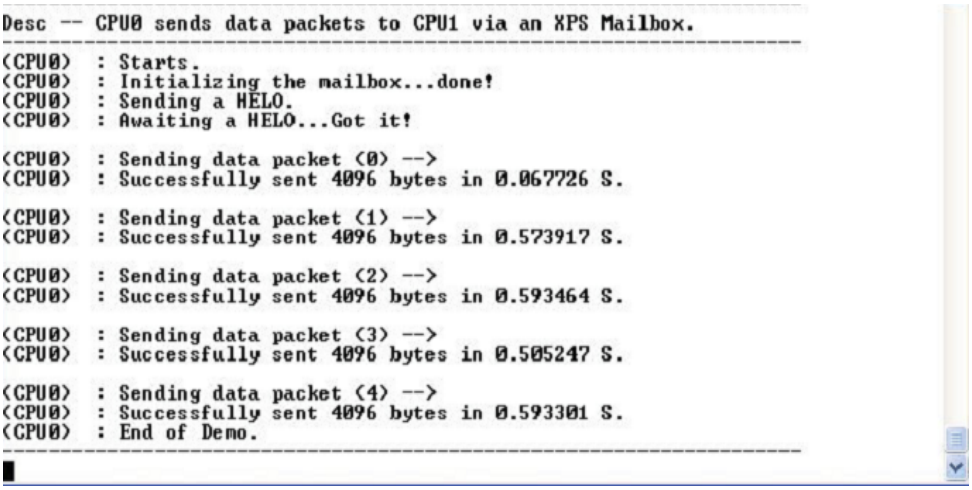

Figure 9. Output from Mailbox Message Passing by Producer

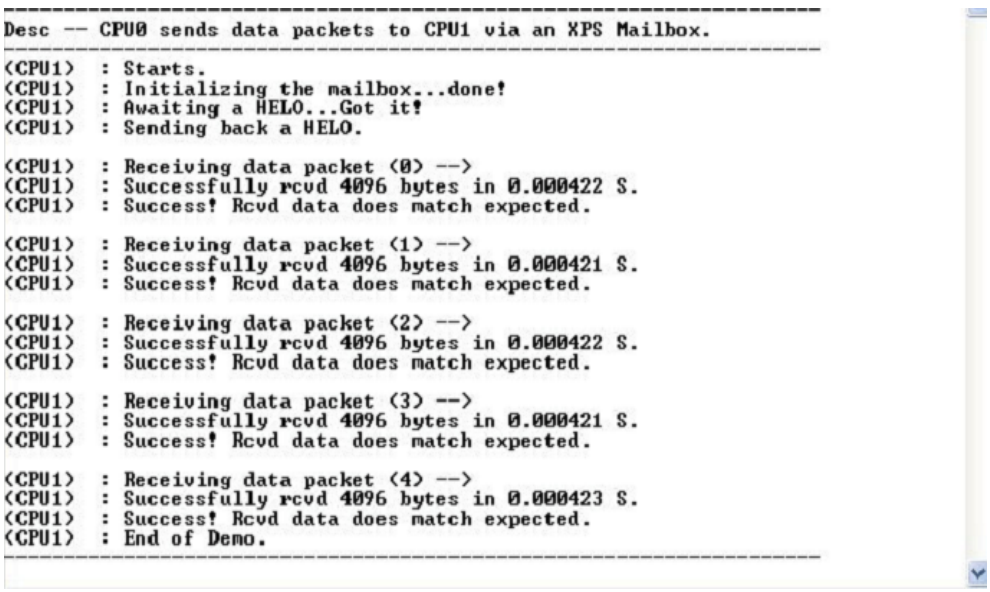

Figure 10. Output from Mailbox Message Passing by Consumer

5. Conclusion

The results of the experiment show that the design of the multi core embedded system based on MicroBlaze has improved the performance of the system. Because of the portability of the MicroBlaze core, which increases the flexibility of the system design? In this paper, when a plurality of MicroBlaze core embedded in FPGA chip, multi-core than single core performance has been significantly improved. In addition, the system not only verifies the feasibility of the PLB bus and FSL bus, but also improves the communication speed, which shows that the system has a certain practicality.

Acknowledgments

We would like to thank professor HU and professor GE for stimulating discussions with respect to the topic of this paper and laboratory equipment. Moreover, we greatly appreciate the reviewers' comments that lead to an improved presentation of the results. This work was supported by Research Program of science and technology at Universities of Inner Mongolia Autonomous Region (NJZY282, NJZZ14288).

References

[1] H. Kim and R. Bond, "Multi-core software technologies", IEEE Signal Processing Magazine, vol. 26, no. 6, **(2009)**, pp.80 -89.
[2] A. Munir, A. Gordon-Ross and S. Ranka, "Multi-Core Embedded Wireless Sensor Networks: Architecture and Applications", IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 6, **(2014)**, pp.1553 -1562.
[3] L.Yan, Q. Shi, T. Chen and G. Chen, "An On-chip Communication Mechanism Design in the Embedded Heterogeneous Multi-core Architecture", Proceedings of 2008 IEEE International Conference on Networking, Sensing and Control, Sanya, China, **(2008)**.
[4] X. Xu, N. Center and L. Wang, "Task assignments based on shared memory multi-core communication", Proceedings of the 2nd International Conference on Systems and Informatics, Shanghai, China, **(2014)**.
[5] Xilinx, "MicroBlaze Processor Reference Guide", **(2006)**.
[6] Xilinx, "Embedded System Tools Reference Manual", **(2006)**.
[7] Xilinx, "Data Sheet: Fast Simplex Link (FSL) Bus", **(2005)**.
[8] A. Yamawaki and M. Iwane, "An FPGA Implementation of a Snoop Cache With Synchronization for A Multiprocessor System-On-chip", Proceedings of the 13th International Conference on Parallel and Distributed Systems, Washington, DC, USA, **(2007)**.
[9] Xilinx, "LogiCORE IP Mutex", (**2010)**.
[10] Xilinx, "LogiCORE IP Mailbox", **(2010)**.
[11] Y.H. Lin, C. Tu, C.S. Shih and S.H. Hung, "Zero-Buffer Inter-core Process Communication Protocol for Heterogeneous Multi-core Platforms", Proceedings of the 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, Beijing, China, **(2009)**.
[12] Xilinx, "Embedded System Tools Reference Guide", **(2011)**.
[13] Xilinx, "EDK Concepts, Tools, and Techniques", **(2011)**.
[14] P. Fogarty, C. MacNamee and D. Heffernan, "On-chip support for software verification and debug in multi-core embedded systems", IET Software, vol.7, no. 1, **(2013)**, pp.56 -64.

Authors

Yang Nie is a Ph.D. in the Digital Engineering Center of Communication University of China and is a lecturer of Jining Normal University, China. His main research interests include Compressive Sensing, High-performance DSP Algorithms and VLSI Architectures.

Zhenhuan Ma is a lecturer of Jining Normal University, China. Her main research interests include PLC system design, High-performance DSP Algorithms and VLSI Architectures.

Lili Jing is a lecturer of Jining Normal University, China. Her main research interests include sensor detection technology, High-performance DSP Algorithms and VLSI Architectures.