Design of data transmission for a portable DAQ system (Postprint)

Authors: ZHOU Wen-Xiong, WANG Yan-Yu, NAN Gang-Yang, ZHANG Jian-Chuan

Date: 2023-06-18T00:00:00+00:00

Abstract

Field Programmable Gate Array (FPGA), combined with ARM (Advanced RISC Machines) is increasingly employed in the portable data acquisition (DAQ) system for nuclear experiments to reduce the system volume and achieve powerful and multifunctional capacity. High-speed data transmission between FPGA and ARM is one of the most challenging issues for system implementation. In this paper, we propose a method to realize the high-speed data transmission by using the FPGA to acquire massive data from FEE (Front-end electronics) and send it to the ARM whilst the ARM to transmit the data to the remote computer through the TCP/IP protocol for later process. This paper mainly introduces the interface design of the high-speed transmission method between the FPGA and the ARM, the transmission logic of the FPGA, and the program design of the ARM. The theoretical research shows that the maximal transmission speed between the FPGA and the ARM through this way can reach 50 MB/s. In a realistic nuclear physics experiment, this portable DAQ system achieved 2.2 MB/s data acquisition speed.

Full Text

Preamble

ChinaXiv Cooperative Journal NUCLEAR SCIENCE AND TECHNIQUES 25, 010404 (2014)

Design of Data Transmission for a Portable DAQ System

ZHOU Wen-Xiong[1],[2], WANG Yan-Yu[1],∗, NAN Gang-Yang[1],[2], and ZHANG Jian-Chuan[1],[2]

[1]Institute of Modern Physics, Chinese Academy of Sciences, Lanzhou, 730000, China

[2]University of Chinese Academy of Sciences, Beijing, 100039, China

(Received July 24, 2013; accepted in revised form December 20, 2013; published online February 20, 2014)

Abstract: Field Programmable Gate Array (FPGA) combined with ARM (Advanced RISC Machines) is increasingly employed in portable data acquisition (DAQ) systems for nuclear experiments to reduce system volume while achieving powerful and multifunctional capabilities. High-speed data transmission between FPGA and ARM represents one of the most challenging issues for system implementation. This paper proposes a method to realize high-speed data transmission by using the FPGA to acquire massive data from front-end electronics (FEE) and send it to the ARM, whilst the ARM transmits the data to a remote computer through the TCP/IP protocol for later processing. This work mainly introduces the interface design of the high-speed transmission method between FPGA and ARM, the transmission logic of the FPGA, and the program design of the ARM. Theoretical research shows that the maximum transmission speed between FPGA and ARM through this approach can reach 50 MB/s. In a realistic nuclear physics experiment, this portable DAQ system achieved a data acquisition speed of 2.2 MB/s.

Keywords: Field Programmable Gate Array (FPGA), Advanced RISC Machines (ARM), High-speed transmission, Driver, Data acquisition (DAQ)

DOI: 10.13538/j.1001-8042/nst.25.010404

Introduction

To understand the mechanisms of nuclear reactions, multiple groups of parameters must be measured simultaneously during nuclear physics experiments [1]. Even for small-scale experiments, several parameters need to be acquired due to the correlations among particles produced in accelerator collisions. To meet these requirements, a portable DAQ (PDAQ) system based on Field Programmable Gate Array (FPGA) and Advanced RISC Machines (ARM) has been developed. In this system, the FPGA controls the front-end electronics (FEE) and transmits data converted by ADCs to the ARM, while the ARM sends the data to another computer for real-time storage and processing. Given that the maximum DAQ rate of the FEE for this system is 8 MB/s, the data transmission speed from FPGA to ARM must exceed 8 MB/s.

Many DAQ systems currently used in nuclear physics experiments are based on the CAMAC (Computer Automated Measurement And Control) bus, which has a maximum transmission rate of 3 MB/s [2]. However, this bus cannot meet the speed requirements of the PDAQ system [3]. Additionally, numerous DAQ systems are based on VME (VERSA-Module-Eurocard), FASTBUS, PCI (Peripheral Component Interconnect), or PXI (PCI eXtensions for Instrumentation) buses [4]. Although these buses can achieve speeds that meet the

PDAQ system requirements [5, 6], they are unsuitable for the PDAQ system because their protocols are very complicated and the ARM's I/O ports are not compatible with those protocols.

Some portable devices are based on serial buses such as SPI and I2C [7]. However, due to speed limitations, these are also not suitable for this system. Therefore, we chose the high-speed parallel communication method, not only because of its simple protocols but also due to its advantages of low cost and high speed. Compared to other methods, it is easier to design and can fully meet the requirements of the PDAQ system.

There are two key aspects to this communication method: First, the FPGA is designed as a memory connected to the ARM's I/O bus; second, the FPGA is controlled by the SROM (Static Read-Only Memory) controller (SROMC) integrated in the ARM. This data communication mechanism can theoretically raise the transmission rate between FPGA and ARM up to 50 MB/s, meeting the design requirements of the PDAQ system. In a nuclear experiment, the system was used to obtain the energy spectrum of Na22, which matched the spectrum obtained using the PHILLIPS7164 CAMAC module.

II. PDAQ System Architecture

The system consists of DAQ modules developed completely in-house. Each DAQ module is designed using FPGA and ARM and can operate independently or together with other identical modules, making it very convenient to extend ADC modules according to experimental requirements. [Figure 1: see original paper] shows the system architecture. The DAQ module can operate in master mode or client mode. The FPGA controls four local ADCs through ADC-BUS and communicates with other DAQ modules through Back-BUS. Only in master mode does the DAQ module need to transmit data to the ARM through SROM-BUS. All data is sent to another computer for processing via Ethernet.

III. Communication Methods Between FPGA and ARM

Communication methods between FPGA and ARM include serial communication, custom parallel communication, and high-speed parallel communication [8]. The speeds of these methods are shown in .

TABLE 1. The speeds of different communication methods [9]

Communication Method	Protocol	Speed
Serial communication	Custom	0.4 MB/s
Custom parallel communication	Custom	0.4 MB/s

Communication Method	Protocol	Speed
High-speed parallel communication	-	50 MB/s

Owing to protocol limitations and ARM characteristics, the serial communication method cannot meet the minimum requirements of the PDAQ system. For the custom parallel communication method, the data and address bus must consist of GPIO ports. An experiment was designed to determine the maximum electrical level change rate of GPIO ports in the Linux operating system, which showed that the minimum period of electrical level change is 5 µs. Thus, if the custom data bus width is 16 bits, the transmission speed can be calculated using the following formula:

$$S = \frac{W}{T}$$

where $W = 16$ bits and $T = 5$ µs, yielding a transmission speed (S) of approximately 0.4 MB/s (tested under Linux). This still cannot meet the PDAQ system requirements.

In comparison, the ARM has a special memory bus with a very high data exchange rate, as its clock frequency is 133 MHz and the data bus width is 16 bits. Because each memory access requires only a few cycles, the transmission speed can reach about 50 MB/s. Overall, the high-speed transmission method is fast and can meet the PDAQ system requirements.

Implementation of the High-Speed Parallel Communication Method

A. Logic Design The ARM (S3C6410) used in the PDAQ system includes an SROM controller (SROMC). A 32 KB FIFO is implemented in the FPGA and controlled by the SROMC [10]. [Figure 2: see original paper] shows the connection between the FPGA and ARM in the PDAQ system.

The connection consists of a 16-bit data bus, a 16-bit address bus, read control signal lines (OEN), write control signal lines (WEN), and an interrupt signal line (IRQ). Because the control logic of the FIFO and SROMC differs, the SROMC control signals must first be processed in the FPGA to enable correct data read/write operations in the FIFO. The following figures illustrate the different control logic of the FIFO and SROMC. [Figure 3: see original paper] shows a timing block diagram of the FIFO in the FPGA, while [Figure 4: see original paper] illustrates the timing block diagram of the SROMC in the ARM.

As shown in [Figure 3: see original paper] and [Figure 4: see original paper], the two components operate differently regarding data input and output. While the FIFO is controlled by read and write clocks, the SROMC is controlled through read and write control signals. Therefore, we must convert the SROMC's read

and write control signals into read and write clocks for the FIFO. After study and analysis, the following logic formulas are available for conversion:

$$\text{reclk} = \text{CSN4\&OEN}$$
$$\text{wrclk} = \text{CSN4\&WEN}$$

Special attention must be paid to synchronizing the "rdclk" and "wrclk" signals with the system clock to avoid conflicts in the sequential circuit. As the FPGA system clock frequency is 50 MHz, the cycle of the rdclk and wrclk signals must be more than 40 ns for synchronization [11, 12]. For example, in [Figure 4: see original paper], with the HCLK frequency at 133 MHz, we set Tacp to 0 and Tacs, Tcos, Tcoh, and Tcah to 1, and Tacc to 3. Thus, the cycle of rdclk and wrclk is 53 ns according to the following equation:

$$T = (T_{acs} + T_{cos} + T_{acc} + T_{acp} + T_{coh} + T_{cah}) \times T_{HCLK}$$

Additionally, the SROMC has different control modes. In page control mode, Tacp is used to read data circularly, while in normal mode, Tacp is inactive or ineffective. With this configuration, we can convert the read and write signals to read and write clocks for the FIFO.

However, because the ARM's data bus and address bus are shared by the FIFO and other devices, we must check the CSN signal before any read or write operation. Otherwise, the read/write logic will produce incorrect results and the entire system will crash. [Figure 5: see original paper] shows an rdclk signal obtained from the FPGA using this logic.

B. Design of the Driver for the ARM [Figure 6: see original paper] shows the driver and application architecture.

In nuclear physics experiments, event signals are generated randomly and unpredictably, making it impossible for the PDAQ system to determine when to read data. To solve this problem, an interrupt mechanism is implemented to achieve efficient operation [13, 14]. The FPGA is configured to notify the ARM to read data when the cumulative number of events (Multi-Event mode) reaches a preset value. This method reduces the ARM's processing burden, and the driver is designed to process the interrupt signal and notify applications to handle it.

The interrupt processor module responds to hardware interrupts [15]. When the number of data reaches the preset value, the FPGA transmits an interrupt message to the ARM, which then reads the data and writes it into a buffer. Furthermore, to decrease the event loss rate and dead time (the time during which a second pulse is not detected), we apply ping-pong buffer technology in the driver [16]. While the application program (APP) reads data from one of the two buffers, the interrupt service routine can simultaneously respond to hardware interrupts and write data into the other buffer.

Additionally, the ARM application is based on multi-threading, allowing signals and data to be processed in different threads. The signal processing thread handles data reading requirements from the driver and writes data into the APP buffer, while the data processing thread analyzes the data obtained by the signal processing thread. Similarly, ping-pong buffer technology is also applied in the APP, reducing dead time risk and increasing data throughput.

V. Experiments and Results

The PDAQ system implementing the high-speed data transmission method has been tested in two experiments.

First, in the laboratory, a P1010 NIM module used to generate pulses and gates produced two signal types: pulse signals and gate signals. Pulse signals were sent directly to the PDAQ system, while gate signals were first processed by a GG8000 NIM module before transmission. Experimental results showed a maximum data transmission rate of approximately 2 MB/s with an event generation frequency of 250 kHz.

In the second experiment, the PDAQ system was used to process the energy spectrum of a Na22 radioactive source using a LaBr3 detector. The energy signal was converted to pulse and gate signals after amplification by the detector's photomultiplier. Experimental results illustrated in [Figure 7: see original paper] showed that the energy spectrum acquired using this PDAQ system matches that obtained through PHILLIPS7164.

VI. Conclusion

The high-speed transmission method successfully solved the problem of massive data transmission between FPGA and ARM in the PDAQ system. The first experimental results demonstrated a maximum data transmission rate of 2 MB/s with an event generation rate of 250 kHz. The second energy spectrum test confirmed that this method meets the PDAQ system requirements. However, due to minor flaws in other system components, the entire system cannot yet run at the full 8 MB/s speed. Once all system parts are optimized, the system will be capable of running at full speed.

References

[1] Wang Y Y and Ma X L. Nucl Electron Detect Technol, 2007, 27: 18–19. [2] Hao J, Ojaruega M, Becchetti F D, et al. Nucl Instrum Meth A, 2011, 652: 483–486. [3] Lu W, Cheng Z Q, Kong J, et al. Nucl Phys Rev, 2012, 28:

464– [4] Kong J, Su H, Chen Z Q, et al. Nucl Instrum Meth A, 2010, 622: [5] Monica T, Jon A, Micrea B, et al. Phys Procedia, 2012, 37: [6] Bauer D A, Burke S, Cooley J, et al. Nucl Instrum Meth A, 2011, 638: 127–133. [7] He Q, Tang B, Du D M, 2009 International Conference on Measuring Technology and Mechatronics Automation, 2009, 370– [8] Liao Y K. ARM and FPGA design and application, China electric power press (in Chinese): 238–240. [9] Shen H and Wang Y X. Aeronaut Comp Tech, 2007, 37: 109– [10] Zhu X P, Xiao T J, Zhao H, et al. Comp Eng Des, 2009, 30: [11] Liu J, Yang Y, Yu X Z. Mod Electron Tech, 2007, 20: 182–186 [12] Zhang Q. Research and Exploration in Laboratory, 2003, 22: [13] Zhang J C, Wang Y Y, Nan G Y, et al. High power laser and particle beams, 2012, 24: 2727–2730. [14] Nan G Y, Wang Y Y, Zhang Y C. Nucl Electron Detect Technol, 2011, 31: 1250–1254. [15] Corbet J, Rubini A, Kroah-Hartman G. Linux device drivers, China electric power press (in Chinese): 46–74. [16] Li W S, Chi Z Y, Chen W J. Optoelectronic Technology and Information, 2005, 18: 76–79.

Note: Figure translations are in progress. See original paper for figures.