IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 5, Issue 3, Ver. II (May - Jun. 2015), PP 59-65
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Real Time Application Depicting the Integration of Microcontrollers and FPGA

BhavyaAlankar[1], Binod Kumar Kanaujia[2]
[1](*Department Of Computer Science, Jamia Hamdard, Research Scholar UTU Dehradun,India)*
[2](*Department Of |Electronics and Comm,Ambedkar Institute Of Technology and Advance Research,India)*

Abstract:-*Microcontrollers and FPGAs are the soul of Digital Circuit Design. Microcontrollers high speed ,low power dissipation and reduced prices make them an obvious choice on one hand and on the other hand, The reconfigurable power, high speed and high density have made FPGA based systems an alternative choice.These both technologies are unmatched in their own domains but it is valuable to identify the different design patterns which can provide canonical solutions to the common problems based on FPGA integrated with Microcontrollers and we have taken a small step towards it by demonstrating the integration of both the technology with the help of an application in which a data sharing architecture (FPGA based) which we have designed earlier integrated with AVR AT90S8515 microcontroller.*
Keywords*: FPGA Master/ Slave Processor, PCI Reconfigurable Computing,Microcontroller*

I. Introduction

Reconfigurable computing or FPGA based computing not only encompasses ability of configuration but also the configured design can be interfaced with other devices for other specific applications[2],[3] and this advancement of reconfigurable computing is shown in our work in which the main focus is on the Real time implementation and integration of Compact Priority based Architecture designed in FPGA with AVR8515 microcontrollers [5] ,[8],[10],[11],[12]acting as Master and Slave respectively and Performing data sharing in Real time[4]. This Paper is the outcome of our previous work in which we have designed an architecture with compact nature allowing prioritized data sharing in a manner in which Master processor will always get priority over the slave processor in data communication[1]. This Architecture was successfully simulated using modelsim 6.0 and synthesized using Xilinx 7.1(ISE)[7],[8].The organization of this paper is as follows: Section 2 discusses the overview of the previous work which includes the FPGA implementation.section3 shows the methodology that we have adopted for integrating our data sharing architecture in real time with AVR AT90S8515 microcontroller[5] including port description of the microcontroller together with the pin description and assignments of pins along with the read/write operation performed by the AVR AT90S8515 micro controller acting as Master and slave processors. Section 4 discusses the testing results and then in the final section we will conclude our work.

II. Previous Works

In our previous work we have designed an architecture which is fully user friendly, flexible and synchronous. In Our architecture One Processor acts as Master and the other Processor acts like a Slave and can intercommunicate with each other[4]. Memory provided via two Rate and fully synchronous ports. Master will always have the priority over slave and we have used multiplexed address data bus for the Master which can be used for slave as well. Moreover in our design we have used the signals of peripheral component interconnect bus[6] for the Master processor to communicate and the slave processor can communicate via busy, address, data and read write signals as shown in Fig 1[1].but depending upon the application you can modify them. As we can see that the whole architecture is divided in to different blocks such as conflict resolving block, Interactive Controller block, Memory block. The working of each of the block together with the simulations of all the blocks along with the architecture has been shown in our previous work[1].This work mainly deals with the real time implementation and integration of our architecture using FPGA and microcontroller

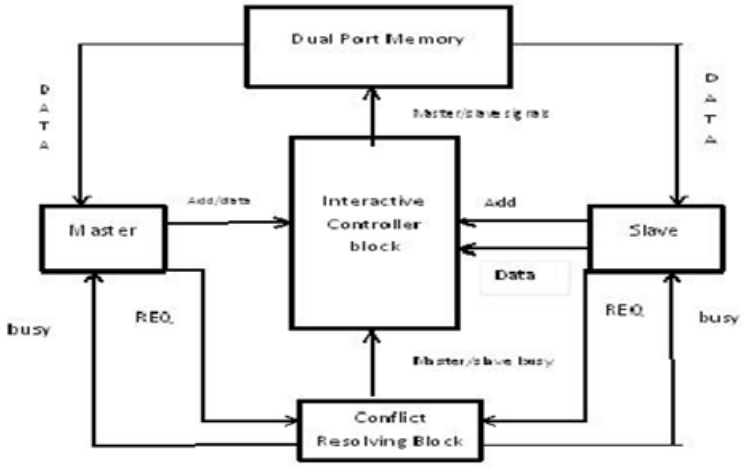

Figure 1: Compact Architecture of Data Sharing between Master and Slave

III. Methodology And Testing In Realtime Enviornment

3.1 Testing of Data Sharing Architecture in Real Time

The data sharing architecture is tested in real time environment by interfacing with two AVR AT90S8515 micro controllers[5],[9],[10] in which one acts as Master, which is inter-faced to the PORTB of the controller and the other acts as Slave processor, which is interfaced to the PORTA of the controller[11],[12]. The AVR used in the place of Master generates the exact signals of PCI, but the operation is slow as com-pared to the original PCIcard. However the testing is perform on the basis of PCI specification timing diagram[6]. The other AVR(Slave) interfaced to PORTA of the controller communicates with the AVR(Master) interfaced to the PORTB of the controller.The AVR acting as Master and Slave respectively generates the address via PORTA, data via PORTD and the control signals via PORTB [2], as shown in Fig 2 below

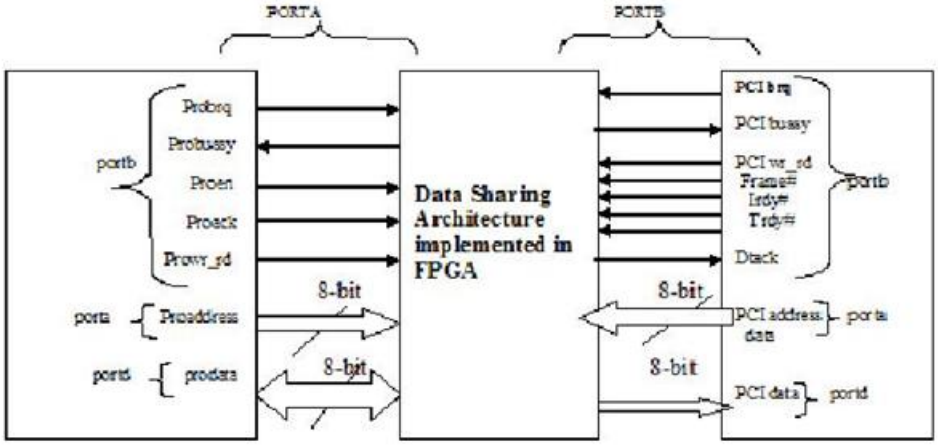

Figure 2: System level diagram

3.2 Testing of Slave Processor in Real Time

One of AVR acting as the Slave processor generates the exact signals of the slave processor through the AVR pins as shown in the Table no.1 The whole operation is started with the event when the Probrq signal is set to low via PORTB of AVR which is interfaced to the left of the FPGA then a check has to be done for Probussy signal whether it is low/high. If high then the processor will go in wait state, if not then the processor will generate 8-bit address via PORTA of AVR and then drives the Prowr_rd signal to low via PORTB of AVR which indicates to the architecture implemented in FPGA that the operation to be performed is Write operation, if drive to high indicates Read operation.

If the operation to be performed is Write then the Slave processor generates the 8-bit data via PORTD of AVR and then drives the Proen signal to low via PORTB and wait until the Proack signal of PORTB goes to low. When the Proack signal is low then the processor drives the Proen to high again and If cycle is last then it

stops operation. Else the address and data is incremented from 00-FF. and the whole process is repeated for each increment of ad-dress and data. Once the address and data reaches the value of FF the process stops

Table 1 :AVR Pins assignment to Slave Processor

AVR pins	PCI bus Interface Controller pins	Description
PINA0-PINA7	Proaddress	Connected to Address pins of the controller
PIND0-PIND7	Prodata	Connected to Bi-directional data pins of the controller
PINB0	Proack	Connected to data acknowledge pin of the controller
PINB1	Probussy	Connected to bussy pin of the controller
PINB5	Proen	Connected to enable pin of the controller
PINB6	Prowr_rd	Connected to write/read pin of the controller
PINB7	Probrq	Connected to bus request pin of the controller
PINC0-PINC7	---	Connected to LED0-LED7 of the AVR
VCC	VCC	Connected to voltage pin of the controller
GND	GND	Connected to the Ground pin of the controller

If the operation to be performed is Read then the processor drives the Proen signal via PORTB of AVR and then waits for the Proack to go low. When the Proack signal is low, the data is displayed on the LEDs via PORTC of AVR. (Since the databus is bidirectional the data is copied to PORTC from PORTD and displayed on 8 LED's ofAVR) then the Slave processor drives the Proen to high again and If cycle is last then it stops operation else the address is incremented from 00 to FF. and the whole process of Read is repeated for each increment of address. Once the address reaches the value of FF the process stops
The Read/Write operation is also depicted in the flow chart shown in Fig 3 and 4

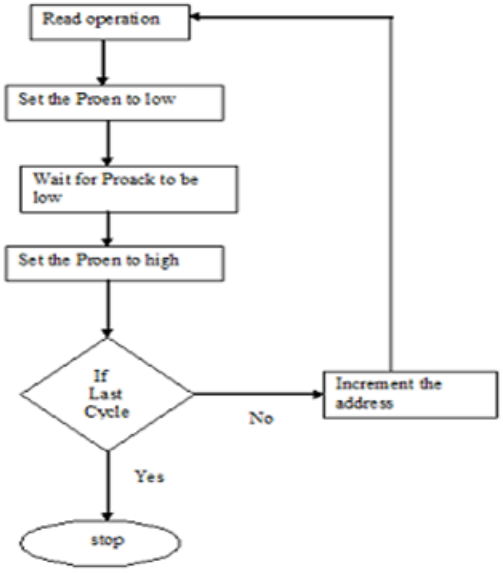

Figure 3 Read operation PORTA(Slave) Testing Flowchart

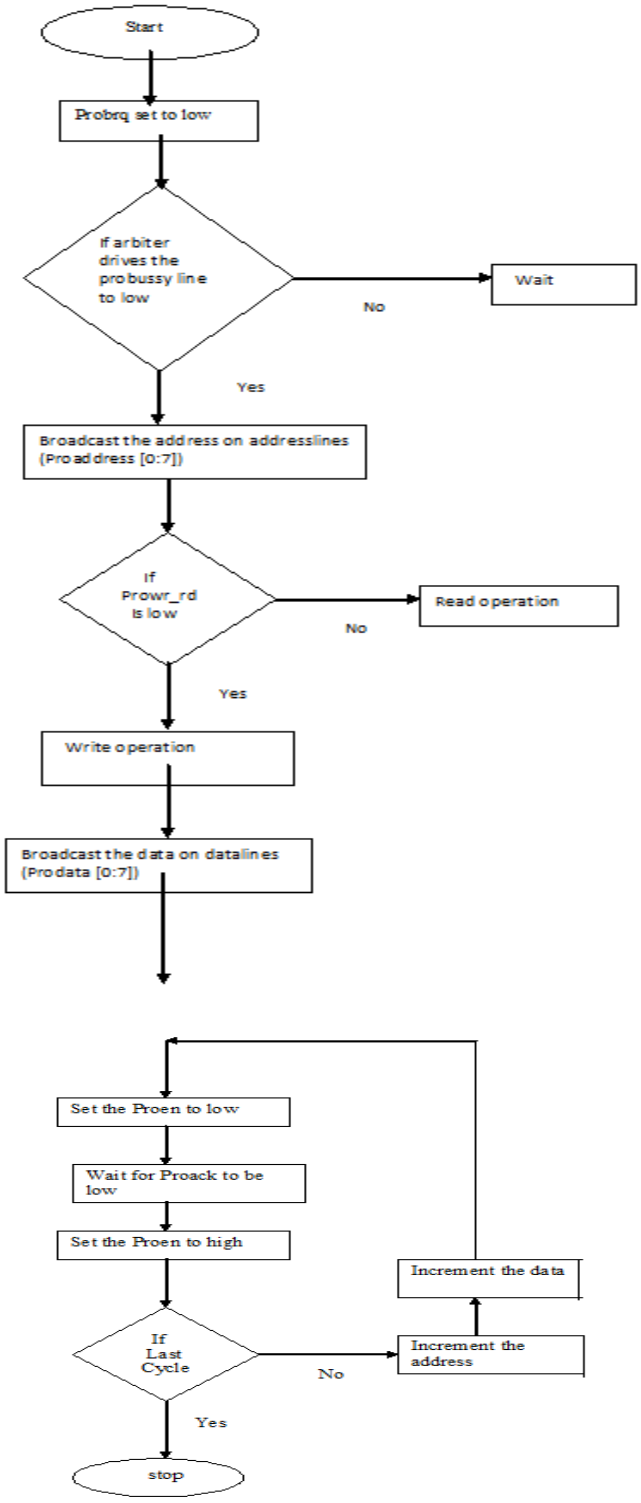

Figure 4 Write operation PORTA(Slave) Testing Flowchart

3.3 Testing of Master Processor in Real Time

The other AVR acting as the Master processor generates the exact signals of the Master processor through the AVR pins as shown in the Table no. 2The whole process started with the event when the PCI brq signal is set to low via PORTB of AVR, which is interfaced to the right of the FPGA, then wait for probussy signal to go low. When the PCIbussy signal is low then the Master will generate 8-bit address via PORTA of AVR and in order to latch the ad-dress it drives the PCI frame# signal low and irdy# and trdy# high through PORTB.After that the data is being sent on the address data line and in order to latch the data the frame# signal

is driven high and the irdy# and trdy# are driven low. As the operation to be performed is Write so the read/write signal is driven through PORTC and asthe PCI en signal is also driven low so that the data ack signal is also get low and which is a input to PORTC and If the cycle is last then it stops operation. Else the address and data is incremented from 00-FF.and the whole process repeated for each increment of address and data. Once the address and data reaches the value of FF the process stops

AVR pins	PCI bus Interface Controller pins	Description
PINA0-PINA7	PCIad-dress data	Connected to Address data pins of the controller
PINB3	Irdy#	Connected to initiator ready pin of the controller
PINB4	Frame#	Connected to frame pin of the controller
PINB5	reset	Connected to reset pin of the controller
PINB6	PCI en	Connected to enable pin of the controller
PINB7	PCI brq	Connected to bus request pin of the controller
PINB2	Trdy#	----------
PINC7	R/w	Connected to read/write pin of controller
PINC6	clk	Connected to clock pin of con-troller
PINC1	PCI bussy	Connected to bussy pin of con-troller
PINC0	dtack	Connected to data acknowl-edgement pin of controller
VCC	VCC	Connected to voltage pin of the controller
GND	GND	Connected to the Ground pin of the controller

1.If the operation to be performed is Read then the PCI drives the read/write signal to high via PORTC of AVR and then waits for the PCI dtack to go low. When the PCI dtack signal is low the data is displayed on the LEDs (Since the output data bus is directly got connected to AVR LEDs) If the cycle is last then the process is stopped

2.Else the address is incremented from00-FF. and the whole process of Read is repeated for each increment of address. Once the address reaches the value of FF the process stops. The whole read/write operation is depicted in the flow chart in Fig 5 and 6

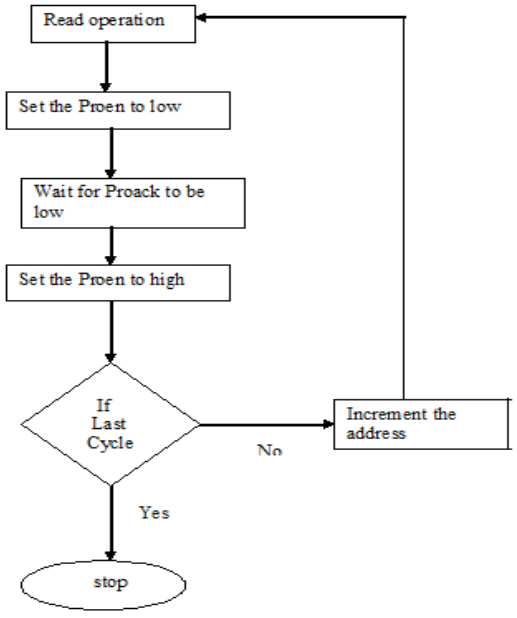

Figure 5 Read operation PORTB(Master) Testing Flowchart

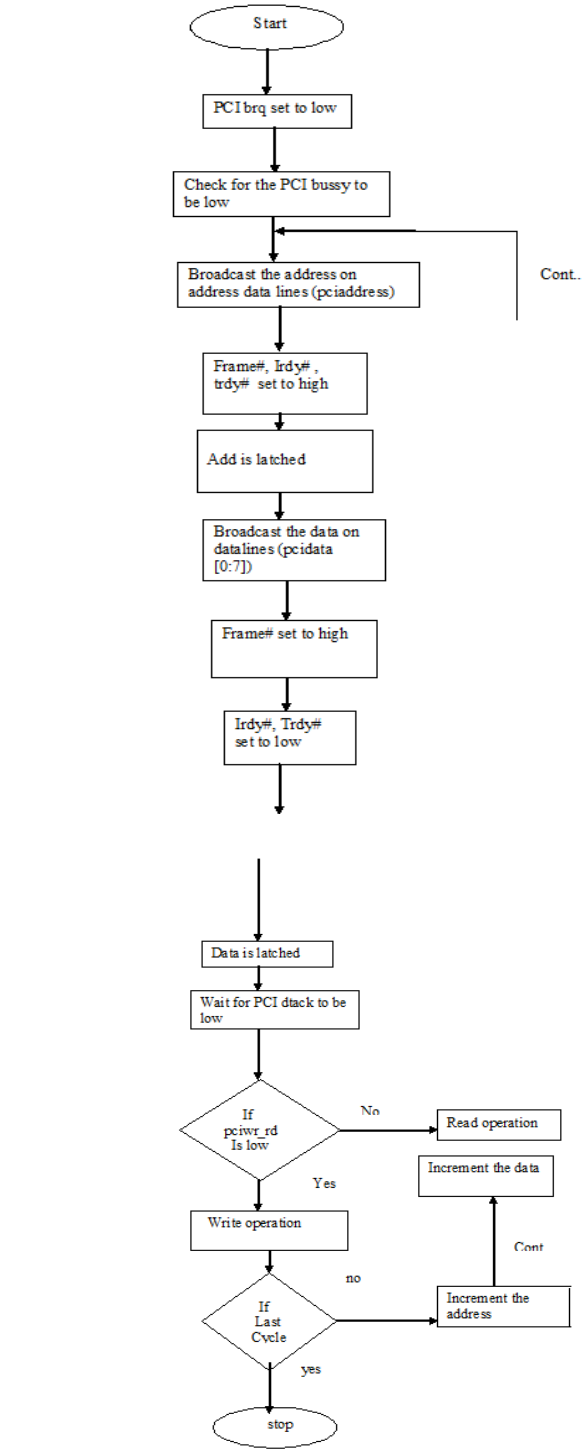

Figure 6 Write operation PORTB(Master) Testing Flowchart

IV. Testing Results

The whole testing setup having FPGA Spartan3 Kit[7],[8] ,[11],[12]acting as our Data sharing architecture and AVR90S8515 microcontrollers acting as Master and Slave respectively is depicted in Figure 7and Testing results are discussed under the following points:

1. **Continuous write and read test: -** Data is written via PORTB continuously from 00 to FF in increasing order and the data is read continuously via PORTA and dis-played on the LED's of the AVR AT90S8515 kit. It is observed that data is read in the same order as it is written in memory by Master.

2. **Master Processor write and read test: -** Data is written via PORTA from 00 to FF in increasing order. Then the data is read via LED's connected to AVR kits. It is ob-served that data is read in the same order as it is written in memory by Master
3. **Slave Processor write and read test: -** Data is written via PORTA from 00 to FF in increasing order. Then the data is read via LED's connected to AVR AT90S8515 kit. It is observed that data is read in the same order as it is written in memory by Slave processor
4. **Master Processor write and Slave Processor read test: -** Data is written via PORTA from 00 to FF in increasing order. Then the data is read via PORTA and displayed on LED's connected to AVR AT90S8515 kit. It is ob-served that data is read in the same order as it is written in memory by Master

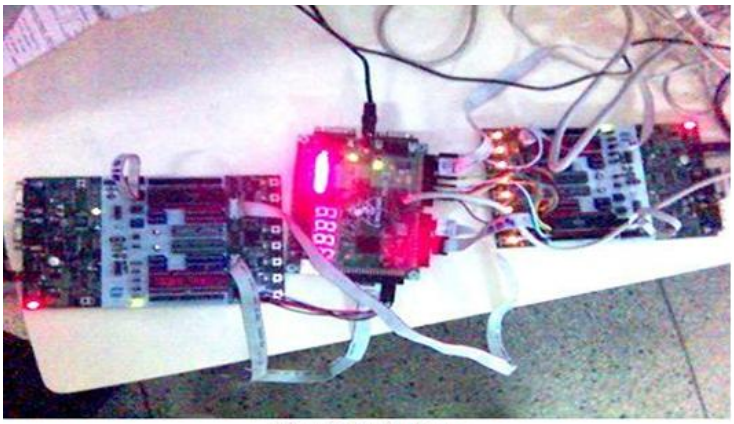

Figure 7: Testing Setup

Slave Processor write and Master Processor read test: - Data is written via PORTA from 00 to FF in increasing order. Then the data is read via PORTB and displayed on LED's connected to AVR AT90S8515 kit. It is ob-served that data is read in the same order as it is written in memory by Master Processor

V. Conclusion

In our work we have designed and demonstrated an application for exploring the integrated world of Reconfigurable computing with Microcontrollers in an easy and refined manner. Our main aim was to give the in-sight of Reconfigurable computing system and how these systems can be further exploited by integrating them with Microcontrollers for different applications. And this is just a beginning because Resconfigurable computing is becoming an important part of research in computer architectures and software systems. It is clear that many fundamental questions remain driven by rapid changes in technology and applications and we have to capture these changes in order to accelerate the performance of our design to the maximum level

References

[1]. Bhavya Alankar and BK Kanaujia,A Compact Priority based architecture designed and Simulated for Data Sharing based On Reconfigurable Computing,Journal of Computing,Volume 4, Issue 4, April 2012.
[2]. Joa O m,. P. Cardoso, Pedro C. Diniz and Markus Wein Hardt, Compiling for Reconfigurable Computing: A Survey, ACM Compu-ting Surveys, Vol.
[3]. T.J. Todman, G.A. Constantinides, S.J.E. Wilton, O. Mencer, W. Luk and P.Y.K. Cheung Reconfigurable Computing:architectures and design methods.published in IEE Proc.-Comput. Digit. Tech., Vol. 152, No. 2, March 2005
[4]. Banerjee,T.P,konar,chaudhary High-speed communication sys-tem developed using FPGA based CAM implementation published in ICETET-2009
[5]. AVR90S8515 microcontroller datasheet website: http://www.atmel.com/Images/doc0841.pdf.
[6]. PCI bus manual website :http://www.pciadda.com/manual/manual1.html.
[7]. Xilinx user guide:http://www.xilinx.com/itp/xilinx10/books/docs/xst/xst.pdf.
[8]. SeeWebsite:: FPGA Spartan-3, http://www.xilinx.com/support/documentation/boards_and_kits/ug2 30.pdf http://www.atmel.com/Images/doc0841.pdf
[9]. Emilio, Maurizio Di Paolo. "Microcontroller Design." In Embedded Systems Design for High-Speed Data Acquisition and Control, pp. 33-48. Springer International Publishing, 2015
[10]. Kanigoro, Bayu, Ricky Efraim Lie, and M. Fitra Kacamarga. "The Framework of Custom Microcontroller using Xilinx Zynq XC7Z020 FPGA." TELKOMNIKA (Telecommunication Computing Electronics and Control) 13, no. 1 (2015).
[11]. Reblewski, Frederic. "Configurable circuits with microcontrollers." U.S. Patent Application 11/311,718, filed December 19, 2005.
[12]. Al-Dhaher, A. H. G. "Development of Microcontroller/FPGA-based systems."International Journal of Engineering Education 20, no. 1 (2004): 52-60.