IOSR Journal of Agriculture and Veterinary Science (IOSR-JAVS)
e-ISSN: 2319-2380, p-ISSN: 2319-2372.Volume 3, Issue 4 (May. - Jun. 2013), PP 85-89
www.iosrjournals.org

Implementation of PCI Target Controller Interfacing with Asynchronous SRAM

Kodam Latha[1], M.Srilatha[2], K.Swathi[3], Sangeeta Singh[4]
[12 3 4]*Department Of Electronics and Communication Engineering*
[1234]*Vardhaman College of Engineering, Hyderabad, A.P-India*

Abstract: *In this paper, we present design of a PCI (Peripheral Component Interconnect) target controller which is interfacing with asynchronous SRAM 64kX8 memory. The target controller provides the control signals to the SRAM for read and writes cycles. The master sends the address, data and other control signals. Based on these signals the controller initiates the read and write cycles we have designed PCI block diagram which represents how the master controls target and target interfaces with memory. We also designed state machine to generate control signals for target controller by which the controller initiates the read and write cycles. PCI implements a 32-bit multiplexed Address and Data bus (AD [31:0]).The simulation results presented in this paper represents read and write transactions between slave and memory according to commands generated by controller. We have been used Xilinx ISE project navigator 0.40d to simulate project code which is written in Verilog Hardware Description Language. We have been tested our functionality by writing test bench and then compared that results with actual functionality.*
Keywords – *Asynchronous SRAM, PCI, PCI connector*

I. Introduction

PCI [1] is a computer bus for attaching hardware devices in a computer. These devices can take either the form of an IC fitted onto the motherboard itself, called a planar device in the PCI specification, or an expansion card that fits into a slot.. It was originated at IntelInc in the early 1990's as standard methods of interconnecting chips on a board. It was later adopted as an industry standard administered by the PCI Special Interest Group or the PCI SIG. Typical PCI cards used in PCs include: network cards, sound cards, modems, extra ports such as USB or serial, TV tuner cards and disk controllers. PCI video cards replaced ISA and VESA cards, until growing bandwidth requirements outgrew the capabilities of PCI; the preferred interface for video cards became AGP, and then PCI Express [2].

In this paper we designed PCI slave or Target Controller which will communicate with both Master and Asynchronous SRAM 64kX8Memory. The master will give commands to the slave and the slave will respond to according to master commands which are generated from target state machine.PCI provides direct access to system memory for connected devices [3], but uses a bridge to connect to the front side bus and therefore to the CPU, that means it is higher performance bus and eliminates the potential for interference with the CPU.PCI cards use 47 pins to connect provided CPU [3]. The PCI bus is able to work with so few pins because of hardware multiplexing, which means that the device sends more than one signal over a single pin. The connectors at the end of the card are connected to the mother board slot and are called gold fingers.

II. PCI TARGET DATA PATH

In Fig1 the target controller [4] provides the control signals to the SRAM for read and writes cycles. The master sends the address, data and other control signals. Based on these signals the controller initiates the read and writes cycles.

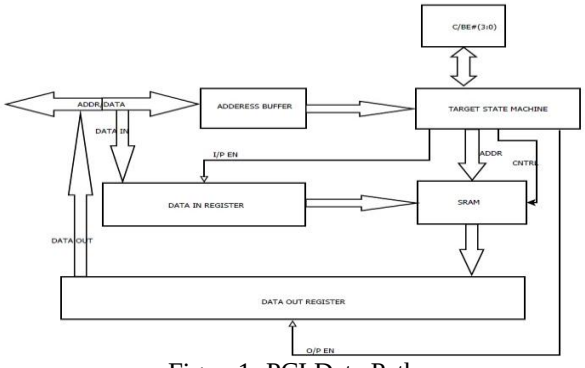

Figure1: PCI Data Path

Firstly, the address and data are stored in buffers [5] and the control signals are provided to the controller. Based on the control signal it provides the enables and multiplexer select input. For read signal the outgoing data is stored in the out buffers and then transmitted on the multiplexed bidirectional multiplexed data bus. The input and output enables are activated while data transferring between data buffers and slave.

2.1Asynchronous SRAM:

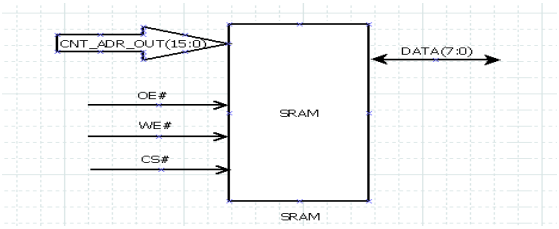

Figure2: Asynchronous SRAM

The SRAM 64kX8 as shown in Fig 2 will active when chip select (cs#) is asserted high. When cs# is asserted and we# is also asserted then the input data to the SRAM is written into CNT_ADR_OUT [15:0] location in asynchronous SRAM. When cs# and oe# both are asserted and we# is reasserted then the data from the SRAM [6] is read into data [7:0] bus. When both we# and oe# asserted at the same time then invalid operation.

2.2 ADDRESS BUFFER

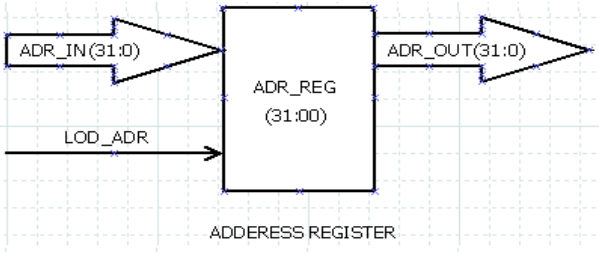

Figure 3: Address Buffer

The address buffer shown in Fig 3 is used to store the input address sent by the master on whom transaction is performed when load_adr is activated.

III. Target State Machine

Fig 4 shows the FSM of Target Controller which consists of different states:

Idle state: It indicates that an idle state [7] is on the PCI bus; no master is implementing any PCI transaction. As soon as FRAME# is sample asserted, i.e. there is an address phase the State Machine goes to Address.

Address state: The address has been decode by the Address decoder in the Configuration Block and Target can check the Hit lines to understand if it is the actual slave of the current transaction. The address decoder compares the PCI address with the BAR value; the Target State Machine decodes the command and enables the I/O, Configuration or Memory hit signals. A memory access to a locked BAR is terminated with an automatic Retry (path to Targ_Term). When Target is not involved in the current transaction (hit signals de-asserted) the machine goes to Not_Inv or Idle (if FRAME# has just been de-asserted).

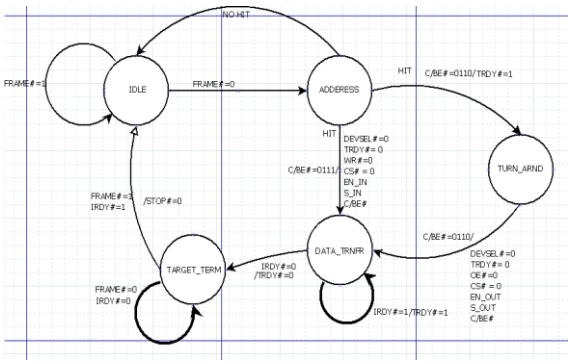

Figure4: Target State machine

Data_tranfr state: This state implements the PCI data phase where data transfers occur. At any time the transaction can be stopped (path to Targ_Term) because of Abort or Stop request from back-end application. A Disconnect occurs after the expiration of the timer for the Data Latency rules (16-clock for the first data and 8-

clock for the subsequent one).A combinatory block called Addr_Checker checks the correct value of the two bits AD [1..0]: linear incrementing mode is supported and the Wrap mode when wrap enable from the Configuration region has value of '1'. If an error is detected a Disconnect after the first data transfer occurs.

Turn_Arnd state: In this state Controller implements turnaround cycle. The path to Addr is for decoding fast back-to-back protocols.

Target_Term state: In this state STOP# is asserted to implement a target termination. When FRAME# is '1' and IRDY# is '0' the turnaround cycle can occur.

3.1 TARGET CONTROLLER

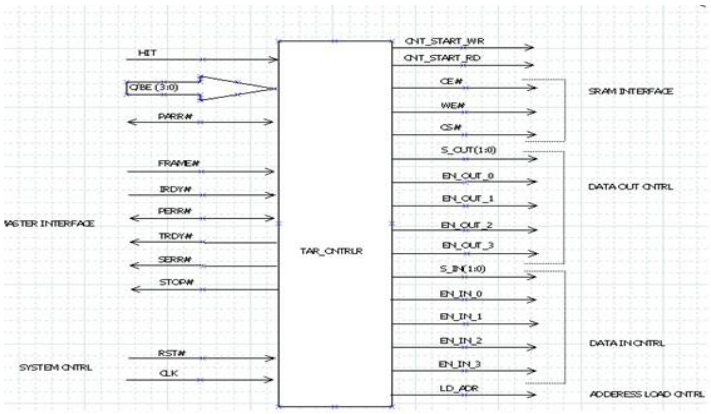

Figure5: Target Controller

Fig5 shows the pin diagram of PCI Target Controller [8], in which all the control signals, interfacing signals and data output represented with appropriate signal names.

IV. Experimental Results

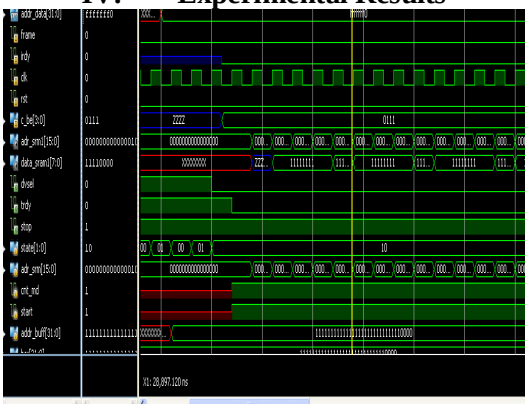

Figure 6: Data write from Slave to SRAM

The Fig6 shows how data can be written into SRAM from the Slave or Target according to the control signal generated by Target Controller Which is acting as Master [9].

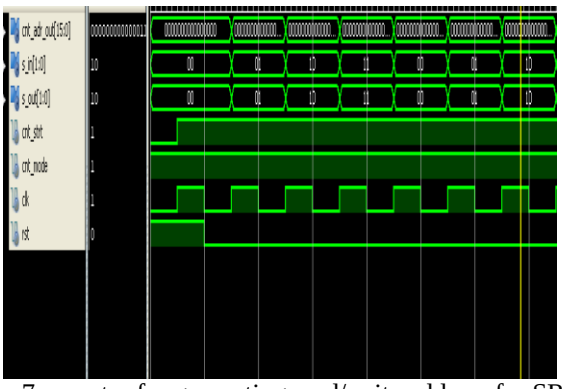

Figure7: counter for generating read/write address for SRAM

Fig 7 shows the result of a counter by which the next 8 bit data slots are reached. The counter will count from 0 to 3 because we require 2-bit counter to count four 8-bit data frames of 32-bit data.

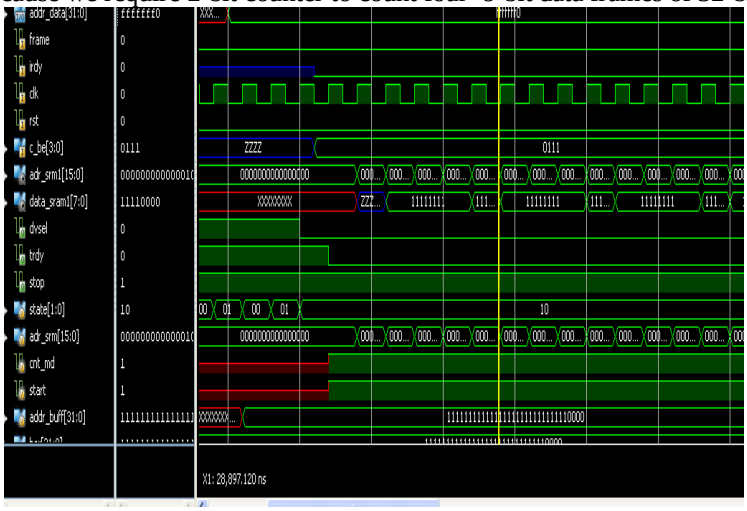

Figure8: Target controller results

Fig8 shows results for Target controller, by selecting corresponding control word particular operation will be performed like memory write or memory read etc..

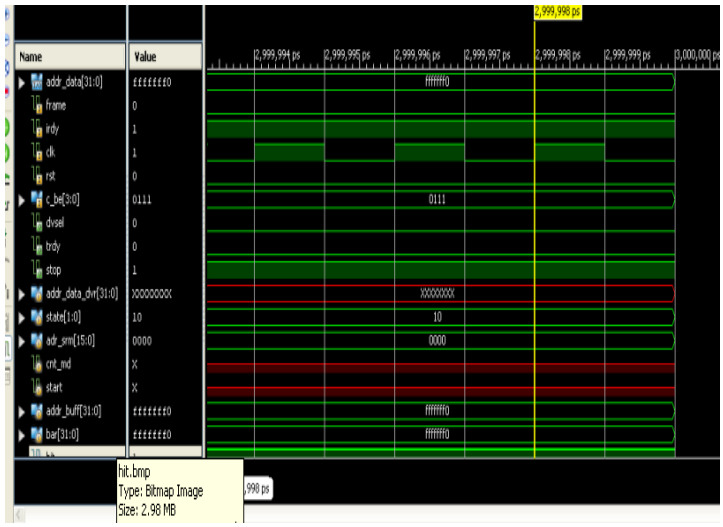

Figure9: Address comparator result

Fig9 shows address comparator which will compare Base Address Register value ADR_IN and LOD_ADR i.e. loading value If both are matched hit signal is generated.

4.1PCI Technical Information

TABLE 1: Technical information

Data Transfer Rate	132Mbytes/
Bus Width	64-bit or 32-bit
Theoretical Max. Transfer Rate	1 Gbytes/sec (8 Gbits/sec)
Bus Clock Signal	33 MHz

V. Conclusion

We have been successfully modelled PCI target controller and verified its functional behaviour with the help of simulation results. The design is working properly without any latency. The command signals of Master are generated through the test bench and the Target is responding to the commands generated by the master. The read and write transactions implemented by interfacing with SRAM 64*8 memory and the simulation results are shown. We compared the simulation results with the expected results and those results are matched. The data transfer rate of PCI is 132Mbytes which is good factor and demanding data rates of multimedia, network disk drives and other peripherals. PCI has the potential to transfer of 64 bit data words at a faster rate, 66MHz clock rate.

References

[1] PCI Special Interest Group, Portland, OR 97214, PCI Local Bus Specification, 2.1 ed., June 1995.
[2] "PCI SIG, PCI Local Bus Specification Revision 2.0, April, 1993".
[3] Fully Compliant PCI Interface in an XC3164A-2, 1994: Xilinx Inc.
[4] Designing Flexible PCI Interfaces with Xilinx EPLDs, 1994
[5] M. A. Song, "System level assertion-based verification environment for PCI/PCI-X and PCI-express," inProc. Conf. CIS, 2007, pp. 1035–1038.
[6] PC Card Standard, Personal Computer Memory CardInt'l Assoc. (PCMCIA), San Jose, Calif., 1997;
[7] CompactPCI Specification, Revision 1.0, PCI Industrialand Computer Manufacturers Group (PICMG), Wake-field, Mass., 1995;
[8] Ravi Budruk, Don Anderson, Tom Shanley, PCI Express System Architecture. Addison Wesley, 2004.
[9] Microsoft, PCI Express and Windows. WinHEC, http://www.microsoft.com, 2004.
[10] PCI Express - An Overview of the PCI Express Standard. National Instruments White Paper
[11] PCI SIG, PCI Express Base Specifications Revision 1.0a, PCI SIG, 2003.
[12] Ravi Budruk, Don Anderson, and Tom Shanley, PCI Express System Architecture, MindShare, 200.
[13**]** P. Bohm and T. Melham, "A refinement approach to design and verification of on-chip communication protocols," inProc. FMCAD, 2008, pp. 136–143.
[14] PCI Express Base Specification Revision 2.0, PCI-SIG, Beaverton, OR, Dec. 2006.