

Design & Development of FPGA Based VME Bus Controller

Himali Patel¹, Poornima Talwai²

¹Vivekanand Institute of Technology, Chembur, Mumbai

² Ramrao Adik Institute of Technology, Nerul, Mumbai

¹Himali.patel@ves.ac.in , ²poornima.talwai@gmail.com

Abstract: The VME bus interface Controller (VIC068A) is used to interface Local CPU bus and VME bus. VME Bus Controller is used in wide application areas where high reliability, good accuracy and high speed are desired to withstand industrial environment like nuclear power plant and process industries. VME Bus controller can configure as Master, Slave, Interrupt Handler, arbiter as well as power monitor. Commercial VME Bus interface controller chips are available from a few vendors and are very expensive. As time goes VME Bus Controller Chips become absolute and vendor support will not be provided. To solve part obsolescence problem and support long term maintainability, it has been implemented by using Field Programmable Gate Array (FPGA). This paper discusses implementation of VME Bus Controller on FPGA development board XC3S200 with functionality coded in VHDL. The design of VME Bus Controller involved implementation of the Master interface module, slave interface module, Interrupt handler module as well as system controller module. All these modules were programmed using VHDL hardware description language. The modules were simulated and implemented using Xilinx ISE 9.2i. By simulation results of the modules functionality of VME Bus Controller was verified correctly. After successful implementation of the VME Bus Controller functionality has been tested in VME Bus setup.

Index Terms: VME bus, VIC, obsolescence, maintainability, FPGA, VHDL

I. INTRODUCTION

The VME bus interface controller (VIC068A) is a single chip designed by cypress semiconductor [6]. The VIC068A was developed through the efforts of a consortium of board vendors, under the auspices of the VME bus International Trade Association (VITA) [7]. The VIC068A is an interface between a local CPU bus and the VME bus. The local bus interface of the VIC068A emulates Motorola's family of 32-bit CISC processor interfaces. Existing system at NPCIL uses Motorola processor based VME bus development board. It uses VIC068A VME bus interface controller that uses Motorola MC68020 processor [5] to communicate. But Commercial VME Bus interface controller chips are available from a few vendors and are very expensive. By using high speed and high density FPGAs it is possible to implement a VME interface controller.

II. SYSTEM DESIGN

1. Overview

The VME bus controller with embedded PC provides everything

necessary for a complete PC/AT computer in a single VME bus slot. In this system implementation, VME bus Controller implemented on FPGA development board XC3S200[1] interfaces with the Motorola Cold Fire CPU[5]. Cold fire CPU's address bus, data bus and control bus interfaces with the FPGA. Signals of FPGA interfaces with the VME Connector (96-pin P1 connector) through transceivers as shown in Figure 1. VME bus Controller can function as system Controller, master, slave, Interrupt Handler, requester, clock generator and utility module.

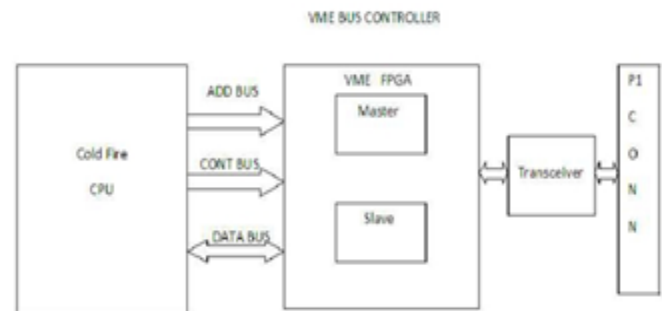


Figure 1 : Block Diagram Of VME Bus Controller

2. Customized Vme Bus Controller Features

Customized VME Bus Controller can configure as system Controller, master, slave, Interrupt Handler, requester, clock generator and utility module.

- **Master Interface:** The addressing supported are A16 and the data transfer supported are D16 and BLT(Block transfer).
- **System Controller:** The on-board VME bus system controller capabilities allow the board to operate as a slot 1 controller. The system controller is programmed to provide single level (SGL) arbitration. The system controller has arbiter, power monitor as well as clock generator.
- **VME bus Requester:** The microprocessor can request and gain control of the bus using any of the VME bus request lines (BR3* to BR0*) under software control. The requester is a Release When Done (RWD) requester.
- **Interrupt Handler:** The interrupt handler monitors and can be programmed to respond to any or all VME bus IRQ* lines. The interrupt handler has a corresponding STATUS/ID register for each IRQ* interrupt. Once the handler receives an IRQ*, it requests the VME bus and, once granted, it performs an IACK cycle for that level. Once the IACK cycle is complete.
- **Slave:** VME Bus controller can also work as a slave module. It responds to transaction initiated by Master Module.

III. DETAILED DESIGN DESCRIPTION

1. VME Bus Controller Board as a System Controller

System controller board must reside in slot 1 of VME backplane. Top level design includes arbiter, clock generator, utility module. It also sets direction and enable signal of various transceivers like data bus transceiver, address bus transceivers etc.

1.1 Clock Generator Block:

VME board has 64MHz. Clock generator block generates 16MHz and 32MHz clock from 64MHz clock. This 16MHz Clock can be used as System Clock for VME bus, 32MHz Clock can be used as an internal Clock for programming.

1.2 Arbiter:

It decides which requester should be granted control of the DTB. A SGL arbiter has been implemented so it accepts bus request on BR3* only. The Arbiter monitors the Bus Request lines BR3* and BBSY*. When Bus request is asserted by any of the Master modules, first it checks status of BBSY* signal if it is high then it issues bus grant by asserting the BGOUT3* line. Arbiter doesn't have onboard requester so it asserts BGOUT3* line instead of BGIN3*. So next slot BGIN3* is asserted if that board's master or interrupt handler doesn't require DTB then it issues no grant by asserting BGOUT3* lines. If that board's Master or Interrupt handler requires DTB requester issues BBSY* Signal. When arbiter detects BBSY* high then it withdraws bus grant. Figure 2 shows simulation results of arbiter module obtained from Xilinx ISE 9.2



Figure 2 : Simulation Results Of Arbiter Module

1.3. Utility Block:

Utility bus contains system clock driver, serial clock driver and Power monitor. Utility module generates SYSRESET* and AC FAIL* signals.

- **System Reset (SYSRESET*):** System reset (SYSRESET*) signal is used in the power-up/power-down sequence. When local RESET* signal goes low it is in output mode. In output mode, a low level on local RESET* signal asserts SYSRESET* for minimum period of 200 milliseconds. When local RESET* signal goes high it is in input mode. In input mode, a low level on this signal resets the internal logic of the VME Bus controller and asserts the local interrupt to the processor.

- **AC Failure (ACFAIL*):** AC failure (ACFAIL*) signal is used in the power-down/power-up sequence. ACFAIL* signal is input to the master when AC power failure occurs. The VME bus controller is enabled to provide a local interrupt when this signal is asserted for 200 microseconds. Master is designed such a way that it will not request the bus for any purpose except power fail activity, after receiving an interrupt.

2. VME Bus Controller Board as a Master/Slave

This Board can work as Master or Slave. If slavemaster jumper is at logic 0 then it works in Master Mode. If slavemaster jumper is at logic 1 then it works in slave mode. Top level design constructs using structural modeling. The top level design includes following modules: Master, InterruptHandler, Requester, Slave, Synchronizer, clock generator and utility module. It also sets direction and enable signal of various transceivers like data bus transceiver, address bus transceivers etc. In Master mode master, interrupt handler, requester, synchronizer, clock generator and utility modules work concurrently. In slave mode slave, synchronizer, clock generator and utility modules work concurrently.

2.1 Master

First it checks signal from processor and goes to master mode. If any interrupt comes then it goes to interrupt handler. If Master needs data transfer bus then it asserts device wants bus (DWB1) signal to the on board requester. If system controller grants bus then on board requester asserts device granted bus (DGB) signal to indicate that bus is available to the master. VME bus data transfer cycles start with the DGB (Device Granted Bus) signal from on-board requester block. It checks read write signal from microprocessor. It drives accordingly write signal high or low. Write signal low indicates data transfer direction from Master to Slave, high indicates data transfer direction from Slave to Master. VME address and AM code information is put on the bus. Master drives address strobe AS* low to indicate slave module that address is stable and can be captured. It drives data strobe DS* to low to indicate that Master placed valid data on the data bus. After completing transaction Master checks DTACK* is coming from slave or not. Master generates VME timeout interrupt if DTACK is not coming from slave board within 2 microseconds after asserting data strobe low. Figure 3 shows simulation results of master module obtained from Xilinx ISE 9.2.

2.2 Requester

The requester asserts Bus Request (BR3*-BR0*) whenever the on-board Master or the Interrupt handler asserts the Device Wants Bus (DWB) signal. The Master drives DWB1 while the interrupt handler drives DWB2. If both the signals arrive together then priority is given to the interrupt handler. The requester, then monitors the BGIN3-0* lines. If BGIN3-0* lines goes low and if the onboard master or interrupt handler needs the bus, then it asserts DGB1 or DGB2 (Device Granted Bus) and Bus busy (BBSY*) signal. If the requester is granted the bus but the on-board master doesn't need the DTB, then it passes the bus grant to the next slot by asserting its BGOUT3-0* low. If the bus grant is accepted by the requester, then it drives BGOUT3-0* high. The bus request level

for each the requester is software programmable. The RWD (Release When Done) requester is implemented. The RWD requester releases the bus when its master negates DWB. Master negates DWB when its data transfer process is over. Figure 4 shows simulation results of requester module obtained from Xilinx ISE 9.2

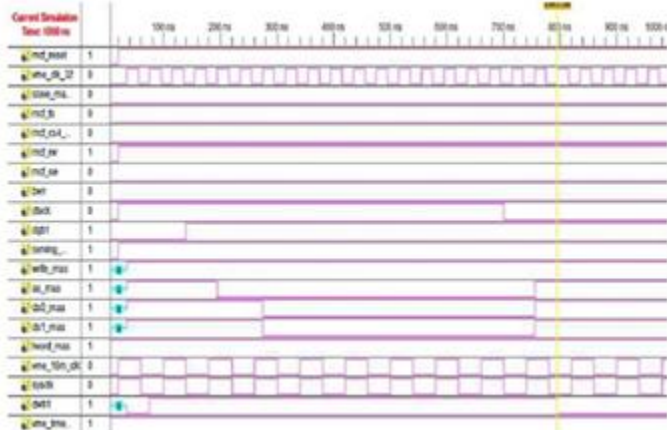


Figure 3: Simulation Results Of Master Module



Figure 4: Simulation Results Of requester Module

2.3 Interrupt Handler

When Slavemaster jumper is at logic low, board works as interrupt handler. The interrupt handler monitors and can be programmed to respond to any or all VME bus IRQ* lines. The interrupt handler has a corresponding STATUS/ID register for each IRQ* interrupt. Once the handler receives an IRQ*, it informs Cold fire processor. It uses on board requester to request the DTB by device wants bus DWB2 when any interrupt comes. If it receive signal from microprocessor and if requester grants bus by asserting device granted bus (DGB) signal then it initiates interrupt acknowledge cycle. It drives LWORD* and DS* low to initiate interrupt acknowledge cycle. It reads a STATUS/ID from the interrupter being acknowledged. It drives IACK* low, Interrupt Handler doesn't drive address modifier lines. It never drives data bus. After receiving DTACK* from interrupter it again monitors IRQ* lines. Figure 5 shows simulation results of Interrupt Handler module obtained from Xilinx ISE 9.2.



Figure 5: Simulation Results Of Interrupt Handler Module

2.4 Slave

It is functional module that detects data transfer bus cycles initiated by a Master and when those cycles specify its participation, transfers data between itself and Master. If Slavemaster jumper is at logic high then VME bus board works as Slave. It checks address modifier codes (AM) sent by Master. If AM code is either 111110, 111001, 111101 or 111010 one of this then it checks address strobe AS*. If it goes low then it indicates address is stable. It checks DPRAM address and whether write signal is high or low. After receiving data form Master it drives DTACK* low. After driving DTACK* low, it wait for data strobe to become low after that it goes to idle state. VME Master can access maximum of 8 VME slave boards each slave board has 3 bit jumper. These jumpers are configured uniquely for each board. The entire DPRAM of each slave board is made available for the master. Master can access any one slave board at a time. The address range for which this slave board's DPRAM available to the Master is 0XFE100000 to 0XFE170000. Figure 6 shows simulation results of slave module obtained from Xilinx ISE 9.2.

2.5 Synchronizer block

Synchronizer module is used to synchronize the VME bus signals between diferent modules. it's main function is when RESET signal is asserted by microprocessor synchronizer gives initial value to signals. e.g when RESET is activated it sets slave- master jumper at logic low, chip select signal at logic high, Out enable signal from the Processor to logic high, AS*, DS*, DTACK*, write*, BERR* and AM* Codes to logic high. When RESET is not activated then it synchronizes signals to diferent Modules.

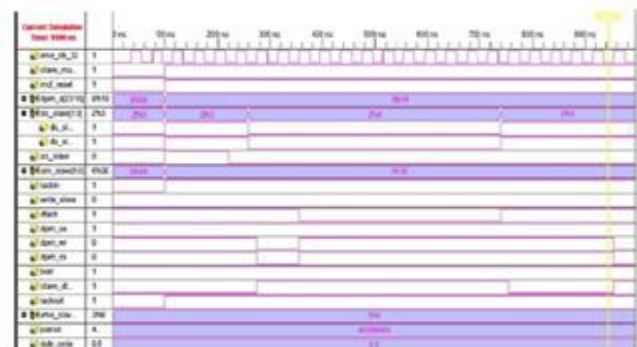


Figure 6 : Simulation Results Of Slave Module

IV. Hardware Test Setup

VME Bus Controller prototype has been implemented on FPGA Development board XC3S200. Previous Section describes various interface blocks with its simulation results which shows logically valid functioning of code in simulation. The hardware setup is required to verify coding practically. Figure 7 shows hardware testing setup. VME Bus is backplane bus[2]. VME bus system contains total 21 slots. In this every VME bus controller board is booted from host PC. Setup contains board with system controller program, that must be reside in slot1. Master 1 and Master 2 reside in slot no. 2 and 3 respectively. I/O I/F (Input Output Interface) acts as a slave, it resides in slot no.5. Relay output Module (ROM) resides in slot no.7 acts as a I/O. It has 16 relays and one Fail LED. As shown in figure one overlapping slot for IOIM (Input output interface module) is given which connects VME Bus to the I/O bus. Another slots given for field connection in which I/O boards can be inserted which acts as slave.

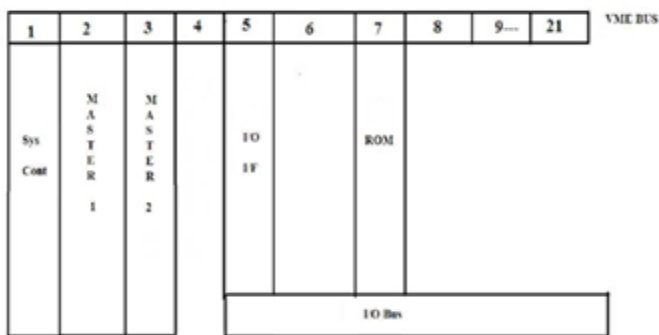


Figure 7: Hardware Testing Setup (I/O I/F: Input Output Interface Module, ROM (Relay Output Module))

V. RESULTS

Hardware implementation of the VME Bus Controller has been tested in VME Bus test setup. Results have been verified by measuring signals from VME Bus backplane to the logic analyzer and digital storage oscilloscope. Figure 8 (practical result from logic analyzer) shows timing diagram of data transfer bus cycle with consequence of Master 1 request. As Master 1 needs data transfer bus, it puts request on Bus Request3 (BR3*) and valid address. System controller grants DTB cycle by asserting BGOUT3* to low unless it is not used by another bus master. As Requester of master 1 detects Bus Grant in Lines (BGIN3*) low, it asserts BBSY* signal low to indicate that now master can use Data transfer bus. Then master asserts first Address strobe (AS*) low to indicate whether valid address is present on address bus, then consequently one of the data strobe (DS*) low to indicate valid data is presented on data bus. Depending on Write signal status master reads or writes to the slave. In this setup Relay output module act as a I/O, its LED goes on or OFF when master accesses slave. Slave responds to DTB by asserting DTACK*. After acknowledgement from slave, Master terminates DTB cycle by deactivate AS* and DS*. Finally requester deactivates BBSY* to indicate system controller that the data transfer bus is available for another Bus Master. Figure shows that complete data transfer bus cycle re-

quire approximately 500ns.

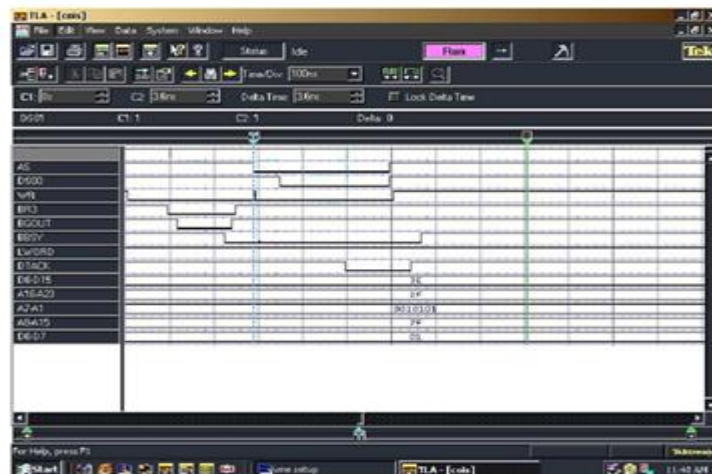


Figure 8: Practical Results when Master resides in slot 1 request for data transfer bus.

Figure 9 shows status of BR3*, BBSY* and I/O board LED when Master reside in slot 2 request for data transfer bus. When Master resides in slot 2 request for DTB, BR3* goes low and system controller grants bus by asserting BGOUT3* low. Requester asserts BBSY* low to indicate the bus is available for data transfer, Master starts to access relay output module's LED. Fail LED goes on to of and viceversa. Master resides in slot 2 takes approximately 500ns to access ROM board.



Figure 9: Practical Results when Master resides in slot 2 request for data transfer bus.

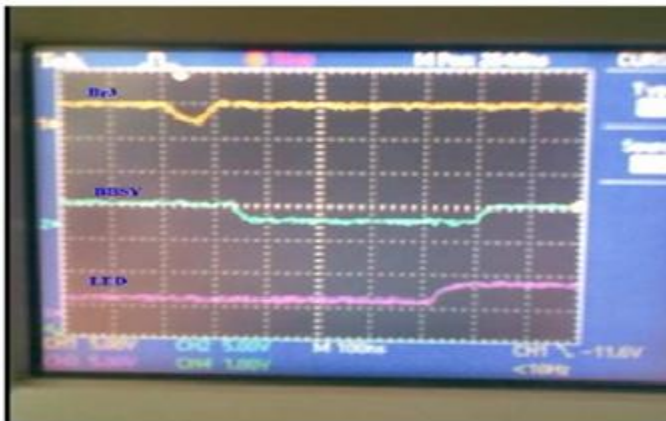


Figure 10: Practical Results when Master resides in slot 3 request for data transfer bus.

Figure10 shows status of BR3*,BBSY* and I/O board LED when Master reside in slot 3 request for data transfer bus.When Master resides in slot 3 request for DTB, BR3* goes low and system controller grants bus by asserting BGOUT3* low.If Master resides in slot 2 doesn't need DTB then it will pass bus grant to the next slot.Requester resides in slot 3 requires bus then it asserts BBSY* signal, Master starts to access Relay output module's LED.Failed LED goes on or off and viceversa.After receiving DTACK*, requester releases data transfer bus.Master resides in slot 3 takes approximately 515ns to access ROM board.Analysis of these two figures shows that daisy chain configuration requires only 10 to 15ns extra to grant the Bus.If two masters request the bus simultaneously then it first grant the bus to slot 2 and after completion of data transfer,bus is granted to slot 3.

VI. CONCLUSION

The designed VME Bus Controller is used for the communication between Motorola cold fire CPU and VME Bus.The design of VME Bus Controller involved implementation of the Master interface module, slave interface module, Interrupt handler module as well as system controller module. All this modules were programmed using VHDL hardware description language. The modules were simulated and implemented using Xilinx ISE 9.2i.By simulation results of the modules functionality of VME Bus Controller was verified correctly. After successful implementation of the VME Bus Controller functionality has been tested in VME Bus setup. By implementing VME Bus Controller using FPGA has many advantages. Which includes, speed increases, time require to perform transaction in terms of nanoseconds. This will be verified using logic analyzer connected to VME Bus setup and also from the simulations obtained from the software. By using XC3S200 FPGA further development of the board can also be possible without changing hardware.

The VME Bus controller board, thus developed will be fully compatible with VME Bus -IEEE 1014-1987 specification [7] and will have application in computer based nuclear systems and in developing VME modules in future. Future work

will involve testing of the board for VME compliance. Some VME bus capabilities like bus timer, arbitration timeout timer and RRS arbitration etc. will be implemented soon.

ACKNOWLEDGMENT

The author is grateful to Shri.A.K.Chandra, Director, Smt. Agilandeswari and Shri.S.G.Bhandarkar for giving an opportunity to carry out work at Nuclear Power Corporation Of India Limited. The author is thankful to Shri Vinod Nagare for his guidance that was continues source of learning as well as motivation.

REFERENCES

- i. *Spartan-3 FPGA Family Datasheet,DS099 December 4, 2009.*
- ii. *Steve Health. The VME Bus user's Handbook. Heinemann Publishing, 1989.*
- iii. *J.Bhaskar. VHDL Primer. Pearson Education third edition, 2004.*
- iv. *Douglas L Perry. VHDL: Programming By Example., McGraw- Hill Publication, Fourth Edition.*
- v. *Free scale Semiconductor. MCF548x reference manual.*
- vi. *Cypress datasheet, 'VIC068A – VME bus Interface Controller', December 1990 - Revised July 23, 1997, pp. 1-15.*
- vii. *VME Bus Specification VITA. ANSI/IEEE Standard 1014-1987, 1987.*
- viii. *Cypress Semiconductor note, 'VME bus Interface Handbook', December 1990, pp. 1-1 to 1-44, 1-53 to 1-55, 1-59 to 1-66, 1-93 to 1-120.*
- ix. *Wade Peterson, 'The VMEbus Handbook', 4th Edition, VMEbus International Trade Association (VITA), 1997, pp. 1-174.*
- x. *Michael John Sebastian Smith, 'Application Specific Integrated Circuits', Pearson Education. Inc., Fifth impression, 2008.*