# Design Of I2C Master With Multiple Slave

Hardik Kaneriya
VLSI & Embedded System Design
Gujarat Technological University PG School
Ahmedabad, India
*Hardikec08@gmail.com*

Santosh Jagtap
VLSI Design
WIPRO
Pune, India
*sansjagtap@gmail.com*

***Abstract* —** This paper implements serial data communication using I2C (Inter-Integrated Circuit) master/slave bus controller. The I2C master/slave bus controller was designed, which act as either master or slave as per the requirement. This module was designed in Verilog HDL and simulated and synthesized in Questasim 10.0c. I2C master initiates data transmission and in order of operation slave responds to it. It can be used to interface low speed peripherals like motherboard, embedded system, mobile phones, set top boxes, DVD, PDA's or other electronic devices.

***Keywords-*** *I2C (or IIC, inter integrated circuit), Master, Questasim, Serial Data Communication, Slave, SDA (serial data line), SCL (serial clock line).*

*****

## I. INTRODUCTION

In the world of serial data communication [8], there are protocols like RS-232, RS-422, RS-485, SPI (Serial peripheral interface), and Microwire for interfacing high speed and low speed peripherals. These protocols require more pin connection in the IC(Integrated Circuit) for serial data communication to take place, as the physical size of IC have decreased over the years, we require less amount of pin connection for serial data transfer. USB/SPI/Microwire and mostly UARTS are all just 'one point to one point' data transfer bus systems. They use multiplexing of the data path and forwarding of messages to service multiple devices. To overcome this problem, the I2C [6] protocol was introduced by Phillips which requires only two lines for communication with two or more chips and can control a network of device chips with just a two general purpose I/O pins whereas, other bus protocols require more pins and signals to connect devices.

In this paper, we are implementing I2C bus protocol for interfacing low speed peripheral devices on FPGA [3]. It is also the best bus for the control applications, where devices may have to be added or removed from the system. I2C protocol can also be used for communication between multiple circuit boards in equipment with or without using a shielded cable depending on the distance and speed of data transfer.

I2C bus is a medium for communication where master controller [9] is used to send and receive data to and from the addressed slave. The low speed peripheral, as a slave is interfaced with I2C master bus and synthesized on Questasim. Fig-1 shows the I2C bus system with the I2C master controller implemented and the real time clock device acting as the multiple slave.

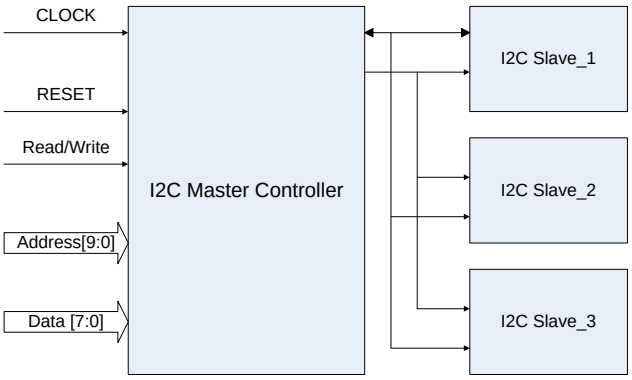


Figure 1. I/O diagram of I2C master controller interfaced with slave devices.

The synopsis of the paper is as follows: In section II, we discussed I2C protocol of our proposed design which also presents module description for our proposed system. In section III, we present the software implementation along with algorithm and flow chart. Section IV, holds the simulated result of I2C master/slave data transmission with different


Figure 2. START & STOP Condition of I2C.

 ISSN: 2321-8169
2890 - 2893


Figure 3. ACK pulse for I2C.

working frequency. Finally, concluded with future scale up in section V.

## II. PROPOSED WORK

### A. *I2C Protocol*

I2C is a two wire, bidirectional serial data communication protocol. I2C may support multiple devices where each of that device can be addressed by its unique address. As we can see in fig. 1 data line is of 8-bit where address line can vary from 7-bit or 10-bit depending on addressing scheme we selects. Here $R/\overline{W}$ signal will decide that whether it's a reading operation or writing.

I2C bus has only two wires and they are SDA (serial data line) and SCL (serial clock line). SCL acts as a clock line for I2C bus where SDA acts as a data line.

### B. *7-bit addressing mode*

I2C bus is in IDLE state when SDA and SCL both are HIGH. Whenever master wants to transmit the data, it will generate START condition by pulling SDA line at LOW level and keeping SCL at HIGH. After START condition bus will be considered as BUSY. Once the START condition is generated master will send 8-bit of data out of that 8

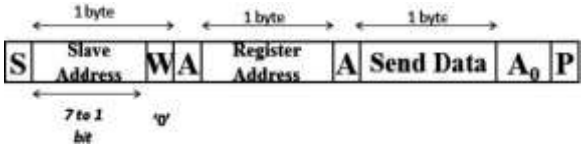


Figure 4. 7-bit addressing mode.

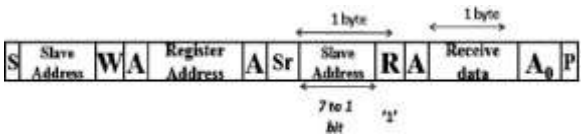


Figure 5. 10-bit addressing mode.

bits first 7 bits are slave address and eighth bit is read or write bit. When master will sent this 8 bits all slaves will listening to it and whichever slave have the same address will generate and send ACK pulse to master. Then master will send register address which is located on that slave, will be used for data transmission whether to write or read data from that that register. Again one ACK pulse is generated by slave. Now depending on $R/\overline{W}$ bit data will be transferred either from master to slave or slave to master. Whoever sending data will send 8-bit and wait for the ACK pulse from the other, so this is a 9 bit of communication. When all data transmission is over master will generate STOP condition, SCL is HIGH and SDA is from LOW to HIGH. We can see the START and STOP condition in fig 2.

### C. *10-bit addressing mode*

10-bit addressing mode is bit different then 7-bit addressing. 10-bit addressing will be known by considering first two bytes after START condition. When first five bits in first byte is 11110 then slave will come to know that it is 10-bit addressing mode. Then $6^{th}$ and $7^{th}$ bits will be MSB of 10 bit address and $8^{th}$ bit will be $R/\overline{W}$. After this first byte whichever slave have the same address will send ACK to master. In second byte all 8 bits will be of remaining 10-bit addressing and now only one slave will send ACK to master for perfect match. Then after master will send register address and now data transmission in this mode also will be as same as 7-bit.

## III. SOFTWARE IMPLEMENTATION

I2C master controller is designed using Verilog HDL [5] based on Finite State Machine (FSM) [13]. FSM is a sequential circuit that uses a finite number of states to keep track of its history of operations, and based on history of operation and current input, determines the next state. There are several states in obtaining the result.

### A. *Algorithm*

*1)* IDLE: When SDA and SCL are HIGH it will stay in idle and will not perform any operation.

*2)* START: To start I2C operation master will generate it by transmittiing SDA from HIGH to LOW when SCL is HIGH.

*3)* WRITE: Master will send 8 bits of data from that 8 bits, 7 bits will be address and $8^{th}$ bit will be zero as it is a read operation.

*4)* If the address sent by master will be matched with slave address then slave will generate ACK pulse.

*5)* After reception of ACK pulse master will send 8 bit register address, if again it is matched then slave again generate ACK pulse.

*6)* Now DATA which needs to be write will be sent by master in a packet of 8 bit. After reception of each packet slave will send ACK.

*7)* STOP: Once data transmission is over maste will generate it by trnasmitting SDA from LOW to HIGH when SCL is HIGH.

*8)* If I2C wants to perform READ operation then aslo it will first go for WRITE operation and once register is

 ISSN: 2321-8169
2890 - 2893

selected, after generating repeated START it will go into READ operation. For that 8th bit of first data byte will be one.

*9)* Master will send slave address for read operation to the slave.

*10)* Master will receive data from slave and acknowledges the slave.

*11)* Master will generate STOP condition for termination.

### *B. FSM*

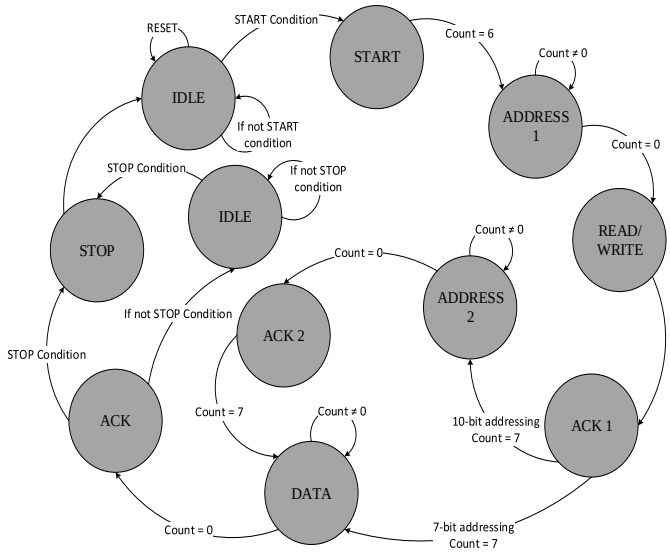


Figure 6. FSM of proposed I2C design

## IV. SIMULATION RESULT

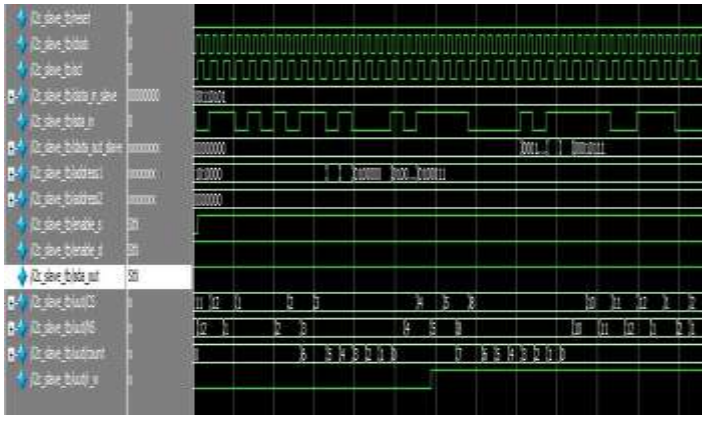

Figure 7. I2C Master/Slave Read Operation

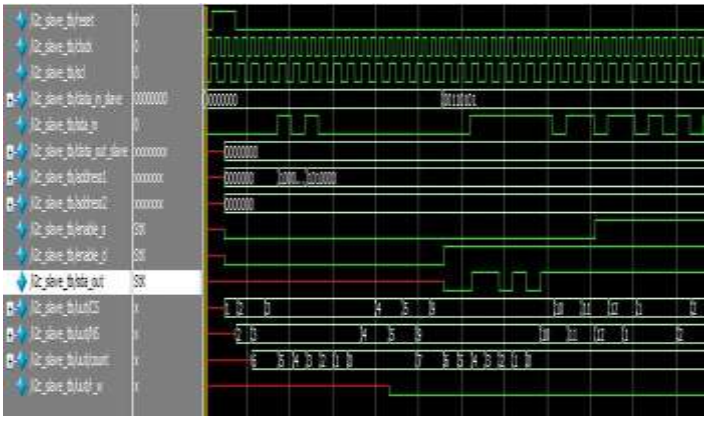

Figure 8. I2C Master/Slave Write Operation

## V. CONCLUSION

This results shows that how I2C master controller, where I2C master will sends and receives data from the salve. We can say that any low speed peripherals can be used with I2C bus. In future multiple masters can be implemented in real time clock domain using different arbitration and clock synchronization techniques.

## ACKNOWLEDGMENT

I would like to express my deepest appreciation to all those who provided me the possibility to complete this paper. A special gratitude I give to my guide Mr. Santosh S. Jagtap and course coordinator Mr. Suresh Sikka whose contribution in stimulating suggestions and encouragement helped me to coordinate my seminar especially in writing this paper.

## REFERENCES

[1]. Peter Corcoran, "Two-Wires and 30Years: A Tribute and Introductory Tutorial to the I2C Two-Wire Bus", IEEE consumer electronics magazine.

[2]. P.Venkateswaran, Madhumita Mukherjee, Arindam Sanyal, Snehasish Das and R.Nandi, "Design and Implementation of FPGA Based Interface Model for Scale-Free Network using I2C Bus Protocol on Quartus II 6.0", IEEE, 2009.

[3]. Shen-Fu Hsiao, Chi-Guang Lin, Po-Han Wu and Chia-Sheng Wen, "Asynchronous AHB Bus Interface Designs in a Multiple-Clock-Domain Graphics System", IEEE, 2012.

[4]. Bollam Eswari, N.Ponmagal, K.Preethi, S.G.Sreejeesh, "Implementation of I2C Master Bus Controller on FPGA", IEEE, 2013.

[5]. P.Venkateswaran, A. Saynal, S. Das, S.K Saynal and R. Nandi , "FPGA Based Efficient Interface Model for Scale-Free Computer Networking using I2C Protocol", 15th international conference on computing- CIC 2006, proc. Research in computer science: special issue – advances in computer science &eng., ISSN 1870-406, pub. National polytechnic institute, Mexico, vol. 23, pp 191-198, Nov. 21-24, 2006.

[6]. M.Krstic, et al., "Globally Asynchronous, Locally Synchronous Circuits: Overview and Outlook", IEEE

Design & Test of Computers, vol. 24, no 5, pp. 430-441, Sept.-Oct. 2007.

[7]. P.Teehan, et al., "A Survey and Taxonomy of GALS Design Styles", IEEE Design & Test of Computers, vol. 24, no. 5, pp. 418-428, Sept.-Oct. 2007.

[8]. R.W.Apperson, et al., "A Scalable Dual-Clock FIFO for Data Transfers Between Arbitrary and Haltable Clock Domains", IEEE Trans. VLSI Systems, vol. 15, no. 10, pp. 1125-1133, Oct. 2007.

[9]. Prof. Jai Karan Singh "Design and Implementation of I2C master controller on FPGA using VHDL", IJET, Vol. 4 No. 4 Aug-Sep 2012.

[10]. Fred Eady, "Networking and Internetworking with Microcontrollers", Elsevier, 2004.

[11]. "Clock Domain Crossing: Closing the Loop on Clock Domain Functional Implementation Problems," The Cadence India Newsletter, May 2005.

[12]. C.E.Cummings, "Simulation and Synthesis Techniques for Asynchronous FIFO Design with Asynchronous Pointer Comparisons", Synopsys Users Group, 2002.

[13]. "Open Core Protocol Specification", OCP-IP Association, 2001.

[14]. "I2C bus specification" Philips semiconductor, version 2.1, January, 2000.

[15]. "AN10216–01 I2C Manual" Philips Semiconductor, March, 2003.

[16]. "I2C bus specification" by NXP semiconductor, version 6.

[17]. Regu Archana, M.V. Rao “Implementation of I2C Master bus protocol on FPGA”, Regu Archana Int. Journal of Engineering Research and Application, p.p 06-10, October 2014.

[18]. A. S. Tadkal, P. Patil “Design of serial data communication using dual master ans slave I2C bus controller”, volume-2, issue-3, International Journal of Electrical, Electronics and Computer System, 2014.

[19]. B. Eswari, N. Ponmagal, K.Preethi, S.G.Sreejeesh “Implementation of I2C Master Bus Controller on FPGA”, International conference on communication and signal processing, April 3-5, 2013.

[20]. “UM10204 I2C-bus specification and user manual” by NXP semiconductor, Rev. 6, 4 April 2014.