Prototyping of On-chip I2C Module for FPGA Spartan 3A series using Verilog

Ramandeep Singh
School of Engineering and Technology,
ITM University, Gurgaon, India

Neeraj Sharma
School of Engineering and Technology,
ITM University, Gurgaon, India

ABSTRACT

Today, at the low end of the communication protocols we find two worldwide standards: I2C and SPI [7]. I2C – commonly known as Inter IC, is a bus protocol. I2C protocol was proposed by Philips Semiconductors to enable faster device to communicate with slower devices without any data loss [1]. In this paper, we will assist the system designers to understand the communication between EEPROM (24C02) and FPGA Spartan 3A. The design is synthesized and simulated using Xilinx ISE and 7.1 and 12.4. Programmed FPGA acts as a master where as EEPROM acts as a Slave.

Keywords: FPGA, I2C Bus, Verilog.

1. INTRODUCTION TO SYSTEM

I2C is a simple bus protocol, developed by Philips semiconductors, which is efficient for inter IC-control. This I2C bus has simple interface,, easy to operate, which more and more engineers designers [2]. It consists of two bus lines; named as serial data line and a serial clock line, also referred as SDA and SCL respectively. I2C enjoys a great advantage of error detection during the transmission of recession bits. In this paper, the design of I2C bus and EEPROM module through FPGA is discussed; it can make better interconnection between different function chips, and provide the basis for future design.

1.1 Understanding the basic terminologies

Whenever we talk about I2C, we usually come across the term such as Master and Slave. Before going further, we must understand these terminologies.

Master is that block in the circuit, which has the capability of initiating the process of sending and receiving the data by issuing the clock and addressing the Slave. Slave acts as a receptor which receives the clock line and the addresses from Master block as show Figure.1

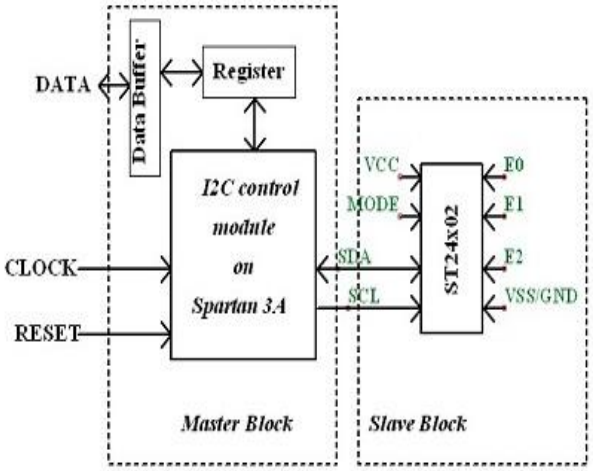

Figure 1: Block diagram showing the Master and Slave Blocks

2. BUS ARCHITECTURE OF I2C PROTOCOL

Although, parallel buses have high throughput as compared to serial buses, but they need less wiring and less IC connecting pins. A bus also concretizes all the format and protocols for communication.

Devices communicating on a serial bus have some form of protocols to avoid confusion. I2C also have certain start and stop conditions and other formats which avoid data loss and information blockage.

2.1 Start and Stop condition for I2C

In I2C, usually the bus is very much idle, for SDA= '1' and SCL='1' condition. As the Master issues a START condition, which is nothing but just a High to Low transition on SDA line, while SCL is still high, data transfer begins as shown in Figure 2.

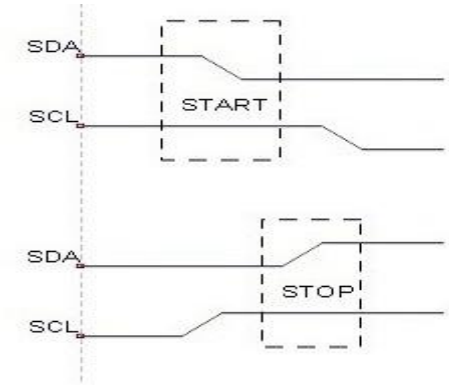

Figure 2: Shows START and STOP condition.

2.2 Data Transfer

Once the Start condition is achieved, it makes the bus status busy, as it is considered that bus is busy in carrying the 8-bits long packet.7-bits of which is considered to be a slave address. The 8th bit is a read/write (R/W bar) bit, where '0' indicates the write operation from Master to slave whereas '1' indicates read operation from Slave to Master. Master sends only one bit on SDA for every clock cycle. The most significant bit is always sent first.

2.2.1 Acknowledgment

The slave device, for which the address matches, gets selected. This selected slave device then sends back the acknowledgment. The acknowledgment is sent by pulling the SDA line low, for 9th clock cycle on SCL line. The concept of acknowledgment is very much imperious for correct transfer of information. During the ninth clock cycle, this can also be referred as acknowledgment clock pulse, the transmitter tends to release the SDA line (High), while the receiver should pull down (Low) it. As a result of which, SDA line can remain low during the high period of the acknowledgment clock pulse. Usually when any receiver is addressed, it is induced to generate an acknowledgment after each byte has been received. In case the slave does not acknowledge the slave address, SDA line is left High by the slave.

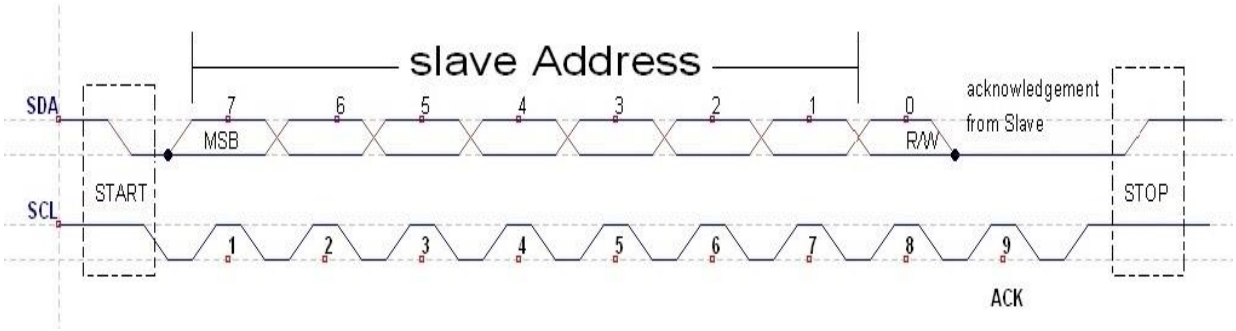

Figure 3: Shows the Slave Address bits and acknowledgment

3. DIGITAL MODEL FOR I2C IMPLEMENTATION

.

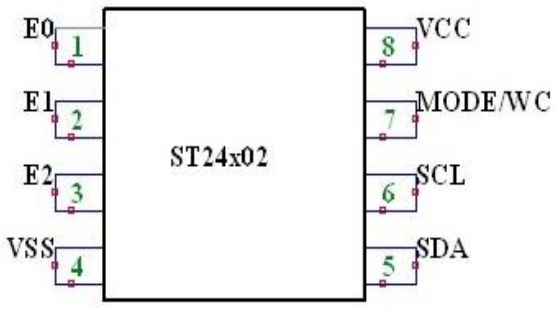

Figure 4: Pin Connection for EEPROM

We are designing module in which programmed FPGA acts like a Master whereas EEPROM (24C02) acts as a slave. For simulation, purpose we are considering a case where Master performs the write operation on-to EEPROM

3.1 Understanding the working of EEPROM

Usually market covers the range of ST24/25x02 EEPROM products. Here x can be C for Standard Version and W for hardware write control version. For this paper, we are considering the 24C02 Standard Version as a Slave. ST24C02 are 2k (256x8) bits electrically erasable memories with the very high endurance of one million erase/write cycles with the data retention of almost 40 years.

Table 1: Signal Names

Pin no.	Name	Function
1-3	E0-E2	Chip Enable Inputs
4	Vss	Ground
5	SDA	Serial Data Address Input/ Output
6	SCL	Serial Clock
7	MODE/WC bar	Multibyte or Page Write Mode(C version)/Write Control(W version)
8	Vcc	Supply voltage.

The memory is compatible with the I2C standard and carries a built-in 4 bits unique identification code for device selection (i.e. 1010). This is used with 3 chip enable inputs (E0, E1, E2) so that 8 such memories can be used with I2C bus and can be selected individually. The memories act as slave device with all memory operations synchronized by the Serial Clock. The Read/Write operation is initiated by START condition. The START condition is followed by the 7-bits device address, where 4 bits are unique i.e. 1010 and next 3 chip enable bits provides 8 different combinations plus one read/write bit and terminated by an acknowledge bit shown in Table 2.

Table 2: Device Address

B7	B6	B5	B4	B3	B2	B1	B0
1	0	1	0	E2	E1	E0	R/W

NOTE: MSB B7 is first.

B3, B2, B1 are known as chip enable bits

B0 is read/write bit. '1'corresponds to read where '0' corresponds to write.

3.2 Device Operation

The ST24C02 supports I2C protocol, so the START and STOP conditions are same as mentioned above. Now let us discuss its operation in more details. Usually the memory supports three modes for write operation.

1. Multibyte Write mode.

2. Byte Write mode.

3. Page Write mode.

3.2.1 Multibyte Write mode

The Multibyte Write mode is only available on ST24C02/R and ST25C02 versions. The MODE (7^{th}) pin is set at V_{IH} for Multibyte mode. The START condition is followed by the device select code along with R/W bit reset to '0'. The memory acknowledges and waits for a byte address. The 8-bits address is capable of providing access to 256 bytes of memory. The device again responds with an acknowledgement, after receiving the address byte. The write operation can start from any memory address. In Multibyte Write mode, Master can send data ranging from one byte to four bytes. Memory gives the acknowledgment for individual byte. Master can terminate the transfer of data by generating a STOP condition as shown in Figure 6.

3.2.2 Byte Write mode

In Byte Write mode, a single data byte is written on to the memory followed by the termination of the transfer of data by Master by generating the STOP condition. This mode is independent of MODE pin. This pin can have V_{IL} or V_{IH} for Byte mode, where as for Multibyte it must be V_{IH}. However this mode is least recommended operating mode as shown in Figure 5. For the address read mode, Master sends the byte address with R/W bit set to '1'.

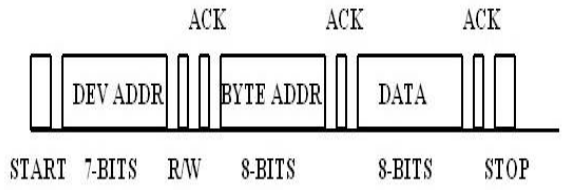

Figure 5: Shows the Byte write mode

3.2.3 Page Write mode

In Page Write mode Master can send up to 8 bytes in one single Write cycle before terminating the transfer. For Page Write mode, MODE pin is at VIL. As similar to Multibyte Write mode, each byte is acknowledged by the memory in Page Write mode. Care must be taken to avoid confusion for the address of each byte. This can be done by using an internal counter. This counter can b used to store the initial address for the first byte can be incremented for each up-coming byte of data. It very important to understand that, after the generation of the STOP condition by the Master, an internal program cycle starts, as the result of which all the inputs are disabled until the completion of this cycle and memory give no response to any other request shown in Figure 6.

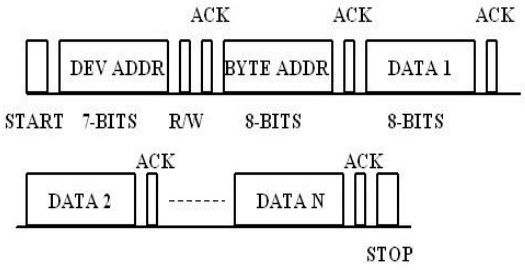

Figure 6: Shows Multibyte Write mode and Page Write mode

4. EXPERIMENT AND RESULTS

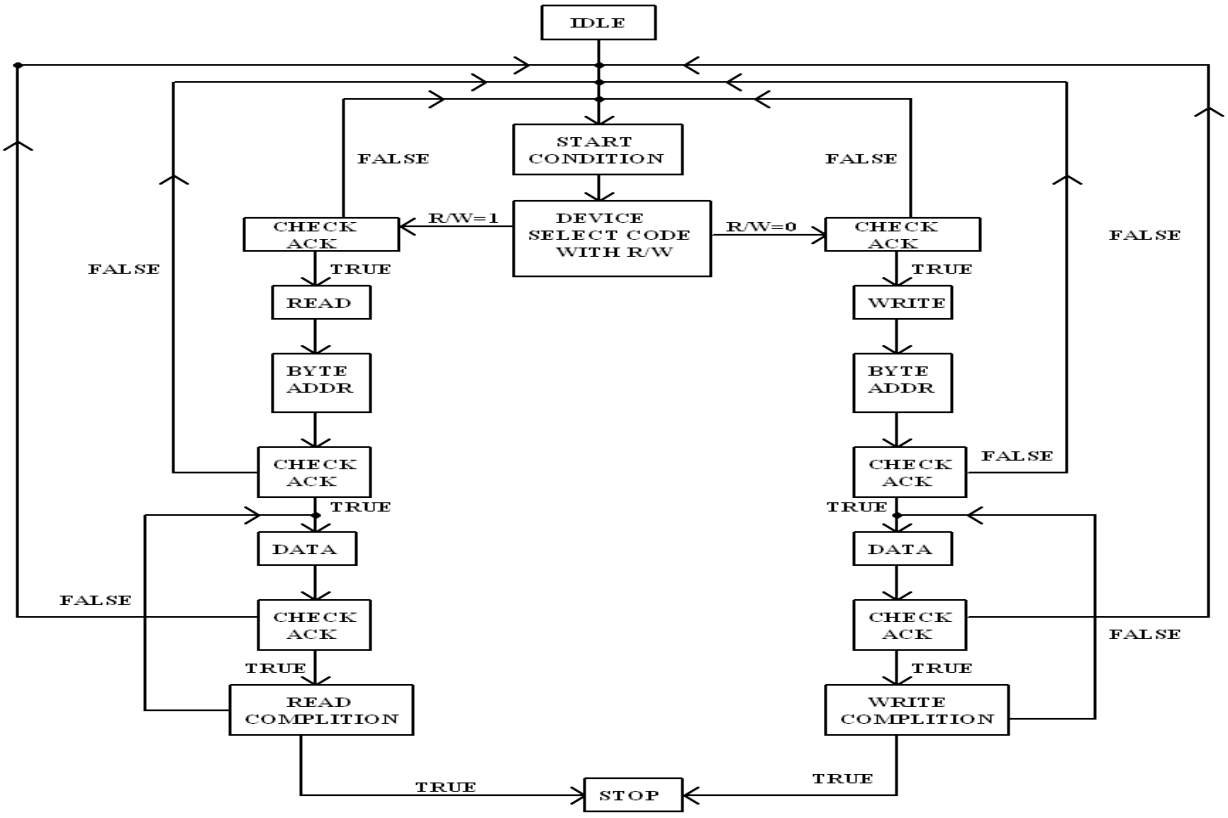

Figure 7: Shows the general flow chart for the read or write operation.

We have seen ST24C02 transmit data through two connecting line (SDA and SCL). The complete network consists of Master and Slave. The device can act as a Master or Slave depending on stimuli provided to generate SCL. In this design, for the sake of simplicity, we are considering a programmed FPGA, which acts as a prototype for Master. On other hand, we are using EEPROM 24c02, which acts as Slave. Figure 7 shows the general flow chart for the read or write operation. Whenever Master need to perform data transfer, it senses whether the SDA is free. If it is free it pulls it to 'low' and enables the clock generator and address block. The address block sends the address byte serially. That is, each bit is sent on SDA line on the positive edge of SCL. If the sent address matches with the Slave address correctly, Slave acknowledges the Master by pulling SDA line to 'low', followed by transfer of byte address and 8-bits data. Each byte address and data byte is followed by the positive acknowledgment. In case slave fails to provide positive acknowledgement, Master is forced to lose the control over the bus, and hence bus is made free and can be used by other Masters.

5. CONCLUSION

In todays world of communication, I2C bus protocol is considered as "little" communication protocols when compared with protocols such as Ethernet, USB,SATA,etc. [6]. But one should always remember that protocols such as Ethernet and SATA are use for the outside the box communication. Whenever there is a need to communicate between integrated circuits, such as two or more microcontrollers and few other peripherals, protocols such as I2C and SPI are used to avoid complexity.

This paper presents a method of evolving hardware design based on I2C bus and ST24C02. Experiments shows stable, accurate and satifactory results. It acts as a basis for other designs to achieve data transmission between I2C bus and other chips.

Figure 8, shows 7-bits device address along with 8th R/W. Since we are using EEPROM 24C02 as a slave peripheral, we have fixed 7:4 bits as 1010. (Explained in Section 3.1), 3:0 bits can be changed as per the embedded system designer wish. In our experiment device address is 1010001(7:0). 8th bit is 0, which indicates that the write operation on EEPROM is being performed. Value 'low' on 9th clock indicates the acknowledgement from the slave. Once, the acknowledgment is received by the receiver, master send the Byte address on SDA line. Byte address indicates the address where our actual data would be stored.

Figure 9, shows 8-bits Data In along with device address and Byte address. Data In represents the actual data to be stored in binary format.

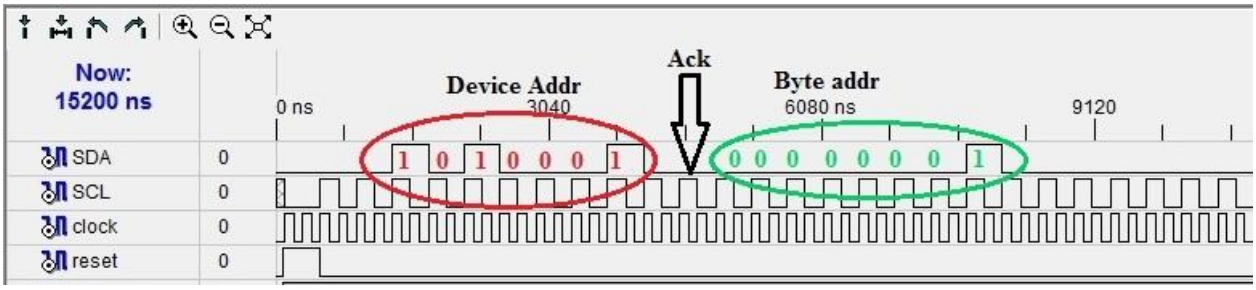

Figure 8: Shows Device Address and Byte Address for Write Operation.

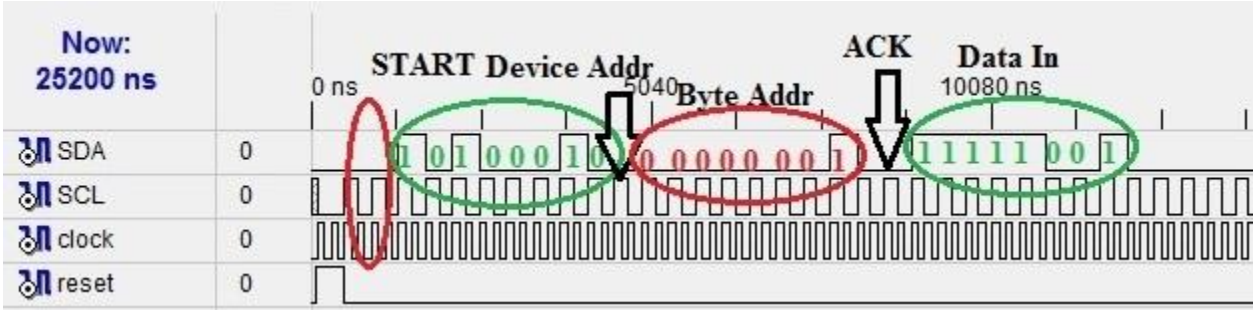

Figure 9: Shows Device Address, Byte Address for and Data In for Write Operation.

6. ACKNOWLEDGEMENT

This work is supported by highly equiped Electonics Lab of ITM University, Gurgaon.

7. REFERENCES

[1] 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. 2009 International Conference on Computers and Devices for Communication

[2] Kangshun Li1,2, Yan Chen1 and Hezuan Liu2. A New Method of Evolving Hardware Design Based on IIC Bus and AT24C02. Proceedings of the 10th World Congress on Intelligent Control and Automation July 6-8, 2012, Beijing, China

[3] Peter Wilson,Du Sheng translate. Design and Practice Based on FPGA [M]. Beijing: Posts&Telecom Press. 2009. Jul.

[4] ST24C02, user manual by ST MICROELECTRONICS

[5] I2C-Bus Specification, Version 2.1 January 2000

[6] Frederic Leens. An Introduction to I2C and SPI Protocols. IEEE Instrumentation & Measurement Magazine February 2009

[7] J.M. Irazabel & S. Blozis, Philips Semiconductors, "I2C Manual," Application Note, ref. AN10216-0, March 24, 2003.

[8] Abinesh R. Bharghava, M.B.Srinivas. Transition Inversion based Low Power Data Coding Scheme for Synchronous Serial Communication. 2009 IEEE Computer Society Annual Symposium on VLSI

[9] Seung-Hun Park, Doo-Hwan Bae. Tailoring a large-sized software process using process slicing and case-based reasoning technique. September 2012.

[10] Pasi Virtanen, Samuli Pekkola, Tero Päivärinta. Why SPI Initiative Failed: Contextual Factors and Changing Software Development Environment. 2013 46th Hawaii International Conference on System Sciences

[11] Manish Kumar Saxena, Ekansh Bhatnagar, Nitin Jaiswal,Milind Parab, Samir Nagesh Kulkarni. Reconfigurable Architecture for IP Peripherals. The International Conference on Signals and Electronics System Gliwice, Poland, September 7-10, 2010.