I2C Master Protocol Implementation in VHDL for Data Acquisition from CDC

Pavan Kumar K, Daniel Raju Paga, V. Bagyaveereswaran

Abstract: ***Communication between devices plays an important role in data acquisition. Proper selection of communication method or protocols is an important task. For any process, hardware and software components are used to perform many tasks like controlling the process and actuators, data conversion, data acquisition etc. A large amount of hardware or software components cannot be used for data acquisition. A proper and efficient approach is intended to be made such that they are used in least possible numbers, also keeping in mind that the different existing devices are compatible without any major modifications. This paper presents an approach to acquire data through I2C protocol in an efficient way. The AD7746 which is capacitance to digital convertor uses I2C protocol for providing the digital data. The digitized information is acquired from the AD7746 by a FPGA which is designed using modular flow method.***

Keywords: AD7746, FPGA, I2C protocol, Master, FSM, Serialdata communication.

I. INTRODUCTION

The recent trend in fabrication is the miniaturisation of physical size in Integrated circuits (ICs). The reduction in size could be attained by reducing the number of pins needed for interconnections between other ICs. Serial data communication protocols comprise of some commonly used protocols such as RS 232, RS485, RS422, Microwire and SPI(Serial Peripheral Interface) [1],[11].These kind of protocols required more number of pins in their respective ICs. Most data transfer buses like Microwave, SPI, USB and especially UARTs are implemented in point to point data transfer. These buses are responsible for multiplexing data and forwarding messages to multiple devices. With the intention of reducing the number of connection pins, an I2C protocol was pioneered by Philips, which requires a minimum of two I/O pins that can overcome the drawbacks in other buses like number of pin connections to devices and even complexity [6].

The main objective of this proposed work is to implement a I2C master protocol for acquiring data from many slaves by using VHDL language in FPGA. The design, coding and simulation was carried out using MODELSIM software by Mentor Graphics [8],I2C protocol[14]. Multiple circuit (ckt) boards can be communicated using this protocol, with or without shielded cables [7].

The AD7746 is interfaced with I2C master bus. Figure 1 depicts the I2C bus system which comprises of the I2C master controller implemented on a FPGA and slave which is AD7746[9],[12],[15].

Revised Version Manuscript Received on February 11, 2016.

Pavankumar K, Department of Control and Automation, SELECT, Vellore Institute of Technology, Vellore, Tamil Nadu-632014, India.

Daniel Raju Paga, Department of Control and Automation, SELECT, Vellore Institute of Technology, Vellore, Tamil Nadu-632014, India.

Prof. V. Bagyaveereswaran, Department of Control and Automation, SELECT, Vellore Institute of Technology, Vellore, Tamil Nadu-632014, India.

This paper illustrates the proposed design along with its modular description, a software implementation with flow charts, algorithms, and simulation waveforms. Conclusion for future scale up is also described.

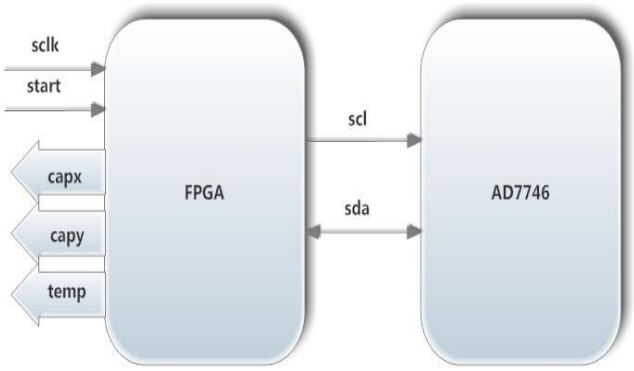

Figure1. Diagram of I2C Master Controller (FPGA) interfaced with AD7746

II. CHARACTERISTICS OF I2C AND AD7746

A. I2C Protocol

I2C protocol uses just two wires and a bidirectional serial bus which provides decisive data communication between any two devices. The abbreviation I2C refers to a standard Inter-Integrated Circuit(IC).The I2C bus network can possess many masters and slave devices(up to 128 devices) . The respective device is recognized based on its unique address. As in Figure 1, temp, capX, capY are 24 bit address of output data received from the slave. The input lines Start and SCLK are used to trigger and initiate the bus controller and the bus controller process.

The I2C bus basically has 2 active wires which are Serial Data Line (SDA) and Serial Clock Line (SCL). Both of them are bi-directional. The SCL is used for synchronizing any data transfers over the bus.SDA and SCL are connected to all the devices on the bus. These are "open drain" drivers. Pull up resistors which pull up these lines are also used. When both SCL and SDA are of high logic(logic 1),then the bus is said to be in an idle mode. If the master i.e. controller wants to transmit data to any one of the slave (AD7746), the slave responds by initiating a start condition through the I2C bus, which follows higher to lower transition on the SDA line when the SCL line is high which can be seen in Figure2(a).This signal acts as an 'Attention signal' to all of its connected devices. During start condition, the bus is busy. The master immediately sends the desired address it wants to gain access to, with an indication if it needs to access it with a 'read' or ' write' operation. After receiving this address, the addresses are compared within each ICs.

In the Figure2 (b) which has the STOP condition, the SDA line is pulled low while SCL is high. The transaction with the slave is ended by this condition.If none of the addresses match, they wait till the bus releases the stop condition .However if it matches, the slave produces a response called ACKNOWLEDGE signal 'ACK' which is done by pulling the SDA to low. When the master receives the signal 'ACK ', the data transfer is in a sequence of 8 bit byte data. On the SDA line, each bit is placed starting with the Most Significant Bit(MSB).For each 8 bit transfer, the slave device which receives the data, sends an acknowledge bit back, so now there are 9 SCL clock pulses which transfers each 8 bit byte data. If the slave(receiving device) sends a low ACK bit back, then it is known that it has received it and is now ready to accept the next byte. If a high signal is sent, then it is indicated that it cannot accept any more data and the master must terminate this transfer which is done by sending a 'STOP' sequence. Figure2 (b) shows the STOP sequence and how the SDA line is driven low and SCL line is driven high, these signals show the end of transaction within the slave device.

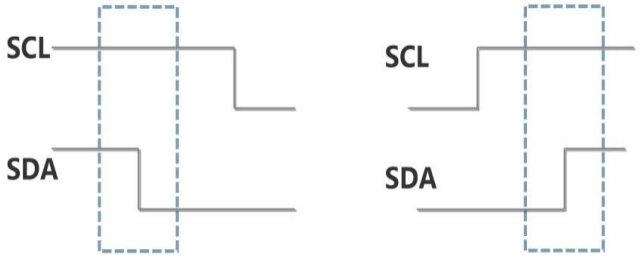

Figure 2(a) Start condition, Figure 2(b) Stop condition

B. Serial Data Communication

They are two modes of operations for I2C bus namely, Master Transmitter and Master Receivers [4]. The I2C master bus can initiate both data transfer and drive SCL and SDA lines. The Slave deviceis addressed when the master initiates the transfers [3]. It issues only data through the SDA line.

The first mode of operation is the Master Transmission mode, where the master after initiating the START sequence, sends a slave address. The slave address byte contains a 7 bit AD7746 address which is followed by a direction bit(logic '0')used for transmission. After decoding the address byte of their slave outputs, an acknowledge signal is sent on the SDA line. After the slave address is acknowledged by AD7746 with a write bit, the master then transmits an address to the slave. This slave will then set up a register pointer pointing to the slave. Now the master will begin to start transmitting each of the byte data it received where the slave acknowledges each byte it has received. Hence, a STOP condition is generated by the master which will terminate the data write. The write cycle procedure is shown inFigure 3[13].

The second mode is the master receiver mode which comprises of transmission and receiving data bytes. The address of the slave and the direction bit send a bit '0' which is received by a master. The master then sends an acknowledge signal to the slave which in turn sends the data to be 'read'. The START sequence is once again initiated and the respective address of the slave along with direction bit as '1' is sent and receives an acknowledge from the slave. The SDA acquires the data from the register for the next 8 cycles of the SCL and the slave should receive a "not acknowledged" from its master in order to end the read. The Read/Receive cycle is given in Figure 4 [13].

C. Slave Device - AD7746

AD7746 possess a high resolution and a capacitance to digital converter (CDC).It has 2 capacitive input channels and an inbuilt temperature sensor. The communication is achieved using a bi-directional, a data transmission protocol and a 2 wire bus. On a 2-wire bus, it operates generally as a slave.

S	Slave Address	0	A	Register Number	A	Send Data	A	P

S - START condition , A - acknowledge , P - STOP condition

Figure 3. Write cycle

S	Slave Address	0	A	Register Number	A	Sr	Slave Address	1	A	Receive Data	A*	P

S - START condition , Sr - Repeated START condition , A - acknowledge , A* - not acknowledge , P - STOP condition ,

Figure 4. Read/Receive cycle

III. SOFTWARE IMPLEMENTATION

The I2C controller is designed using VHDL [4] which interlinks several small modules to a top module. These modules were designed based on finite state machines (FSM)[2]. To track its operational history FSM uses a finite number of states in sequential order, the next state is determined by the operational history and the present input. Many states are involved to obtain the result.

There are two types of modules (lower level and middle level) that are linked to the Top module. The four lower level modules being start conversion, stop conversion, read and write. These four lower modules are linked to the four middle level modules which are Power on reset, register initiate, read status and read register data modules respectively. The top level module links all the components of these modules through port mapping.

The corresponding middle level modules are initiated by the Top module when certain procedures are required to be executed. After the execution of the respective middle level module that was initiated, it gives an end of module indication to the Top module. Similarly the lower level modules are initiated by the middle level module whenever a certain lower level module has to be executed.

The lower level module gives an end of module indication to the middle level module as soon as the respective module finishes execution. Figure 5 shows the handshaking between the lower and the top module. The lower level modules get required data by the middle level module or from the Top module through the middle level module. Likewise the data from lower level module is sent to Top module through the middle level module.

Figure 6 shows the project's module flow diagram. The dir is used to indicate if the data is being transferred (being 1) or received (being 0) from the master. The SCL is bi-directional bus but the current project was designed where the master has full control of the SCL bus making it unidirectional. The SDA on the other hand requires to be bidirectional to transmit and receive the data from the slave [10].

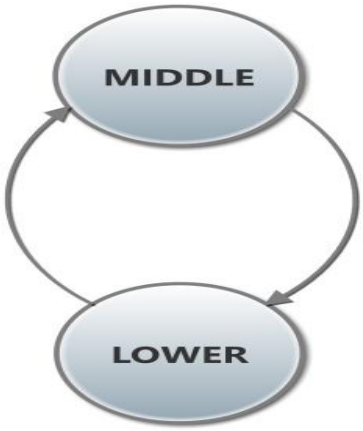

Figure 5 Handshaking between modules

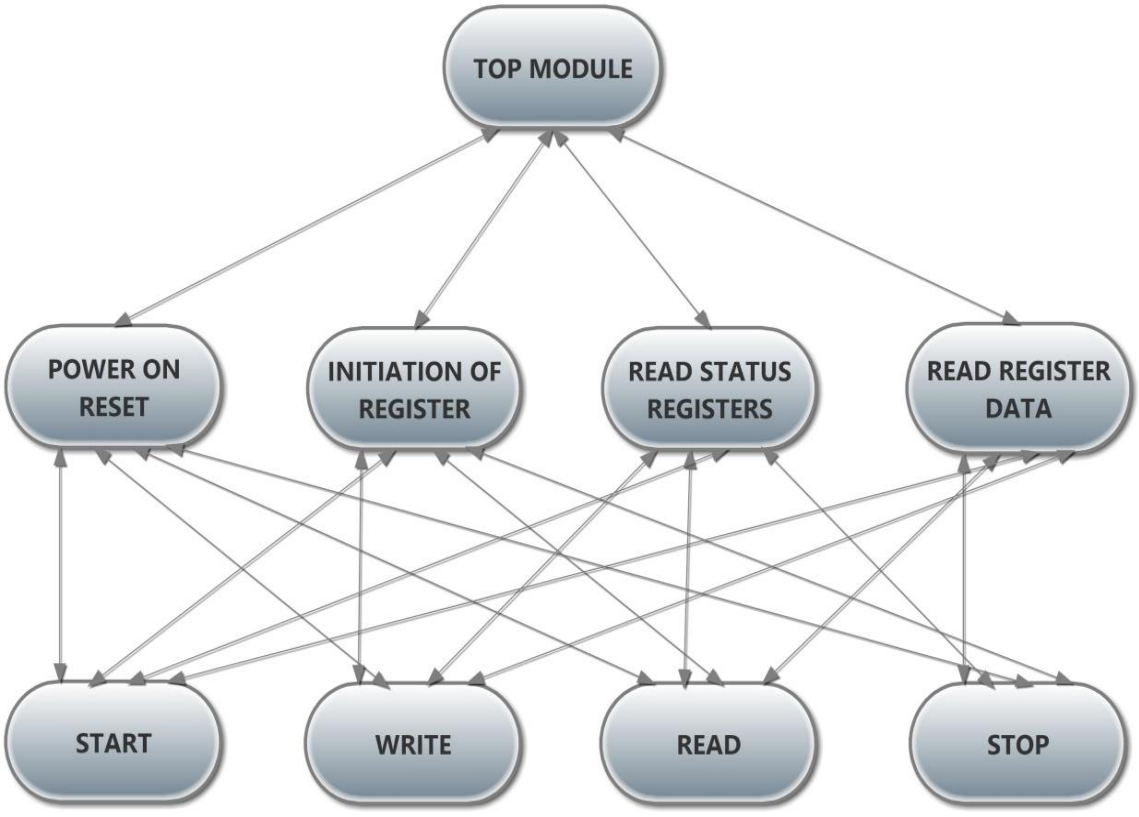

Figure 6 Module flow diagram

Table 1 I2C bus control and direction

I2C condition	Control of bus		Direction of bus	
	SCL	SDA	SCL	SDA
Start	Master	Master	Output	Output
Send	Master	Master	Output	Output
Ack by slave	Master	Slave	Output	Input
Receive	Master	Slave	Output	Input
Ack by master	Master	Master	Output	Output
Stop	Master	Master	Output	Output

LOW LEVEL MODULES

1. **Start module** - initiates the start condition of the I2C protocol by making the SDA low while SCL is high. The dir is logical 1 as the master initiates the start cycle.
2. **Send module** - sends the data acquired from the middle level module through SDA for 8 pulses clock from SCL and gets an acknowledge from the slave for the 9th clock pulse. The dir is logical 1 for the 1st 8 pulses as the data is send by master and the dir is set to 0 for the 9th pulse as the master awaits an acknowledge from the slave.
3. **Read module** - receives the data from the slave for 8 pulses of clock given through SCL bus by master. Their is set logical 0 as the data is being received .The 9th clock pulse makes dir 1 and pulls down the SDA bus indicating the slave that data was received.
4. **Stop module** - stops the conversion process by making the SDA low to high when the SCL is high.

MIDDLE LEVEL MODULE

1. **Power on reset module** - is used to reset the AD7746.This doesn't reset the entire I2C bus except the AD7746.This is done by
2. sending the reset command address word i.e 0xBE to the slave.
3. **Initialization of registers module** - is used to initialize different control registers of AD7746 which controls functions of the AD7746.
4. **Read status register module** - reads the status register and compares the EOC (end of conversion) bit that will indicate whether the data to be acquired from the slave is ready.
5. **Read register data module** - reads the 24 bit data in three 8 bit read cycle from the respective address given by the Top module.

TOP LEVEL MODULE

1. **Top module** - has inputs which are start which is used to initialize the data acquisition and system clock sck , whereas the outputs are the capX, capY and the temp data.

Figure7 shows the flowchart for the top level I2C master communication with slave. Figure 8 shows the Finite state machine diagram for the top module. Figure 9 shows the Modelsim simulation result for start module, which initializes the communication in the I2C bus[8]. Figure10 shows the Modelsim simulation for stop module which terminates the communication with the slave.Figure11 shows the simulation result for write module which transmits the data to slave. Figure12 shows the reading of data from the slave device.

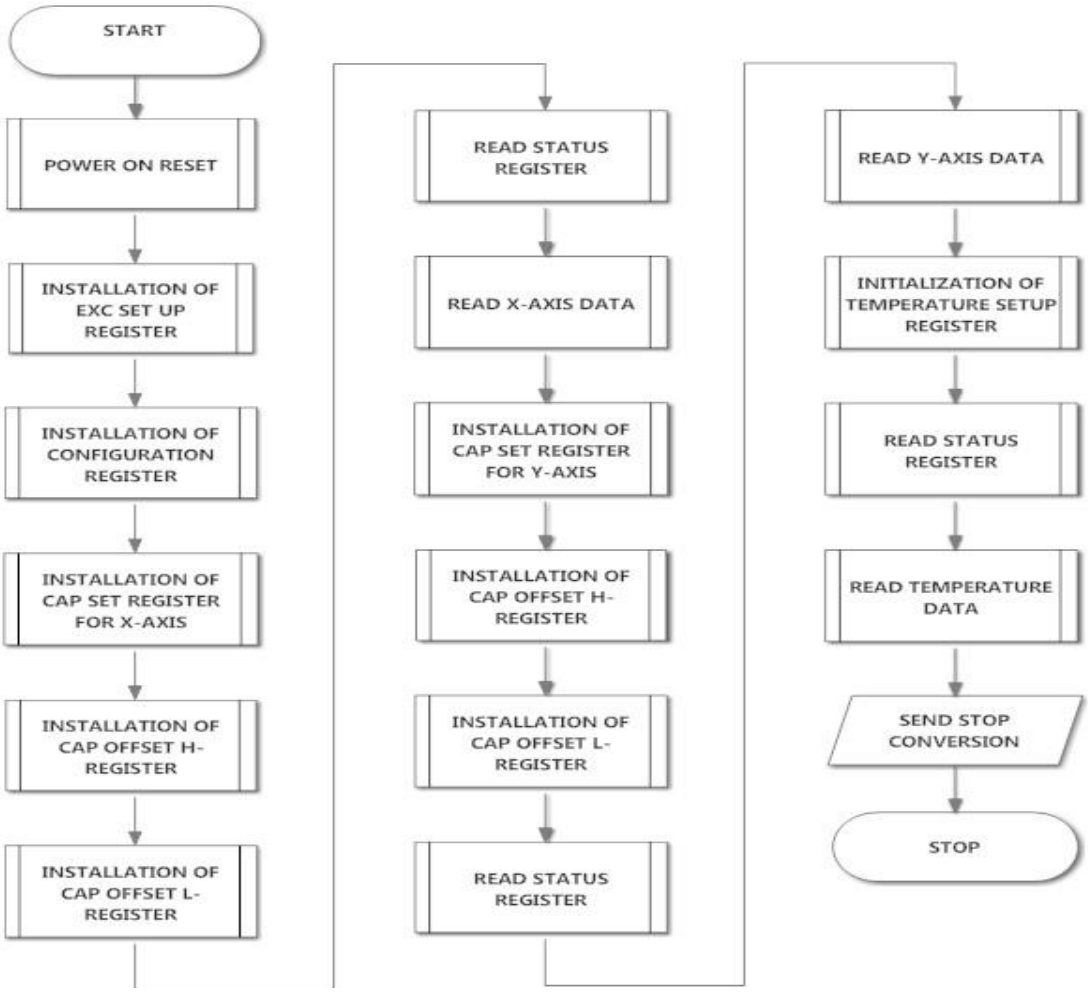

Figure 7 Flow chart of top module

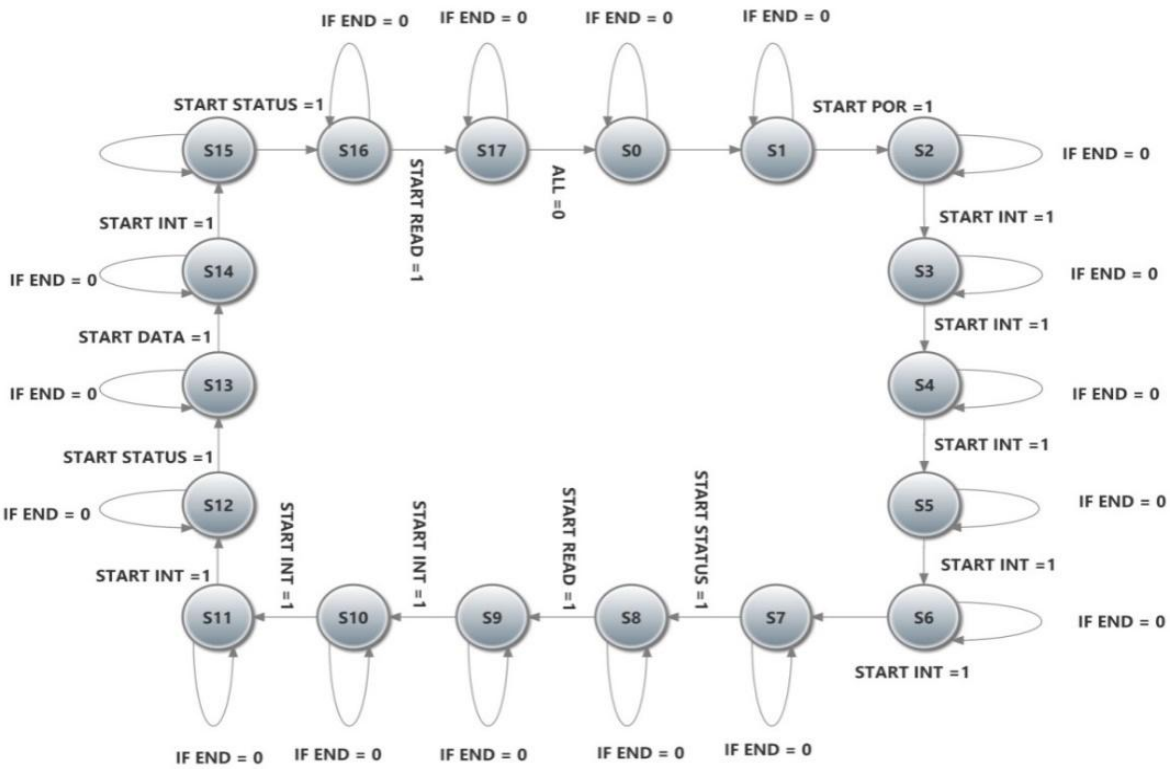

Figure 8 Finite state machine diagram

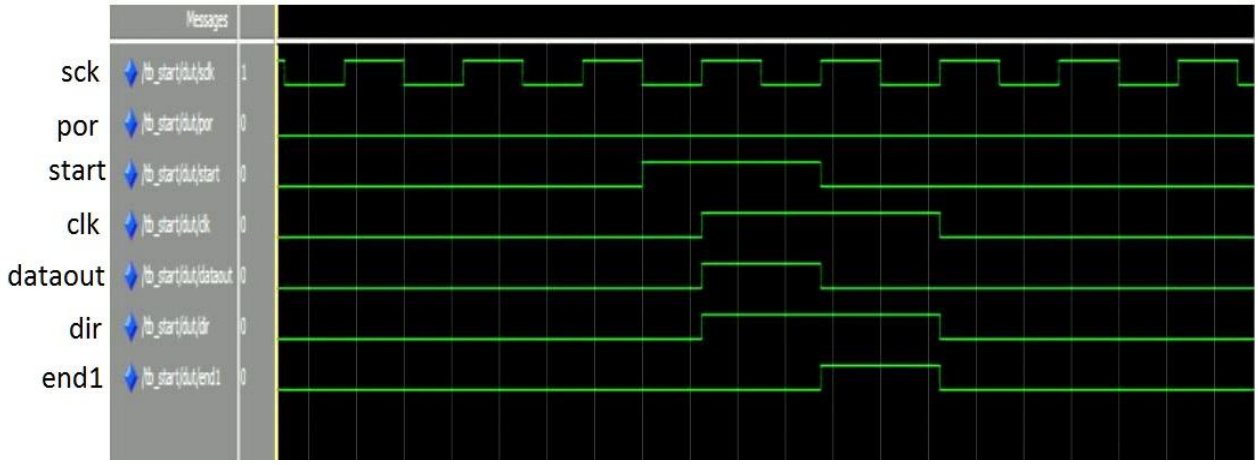

Figure 9 Waveform for Start module

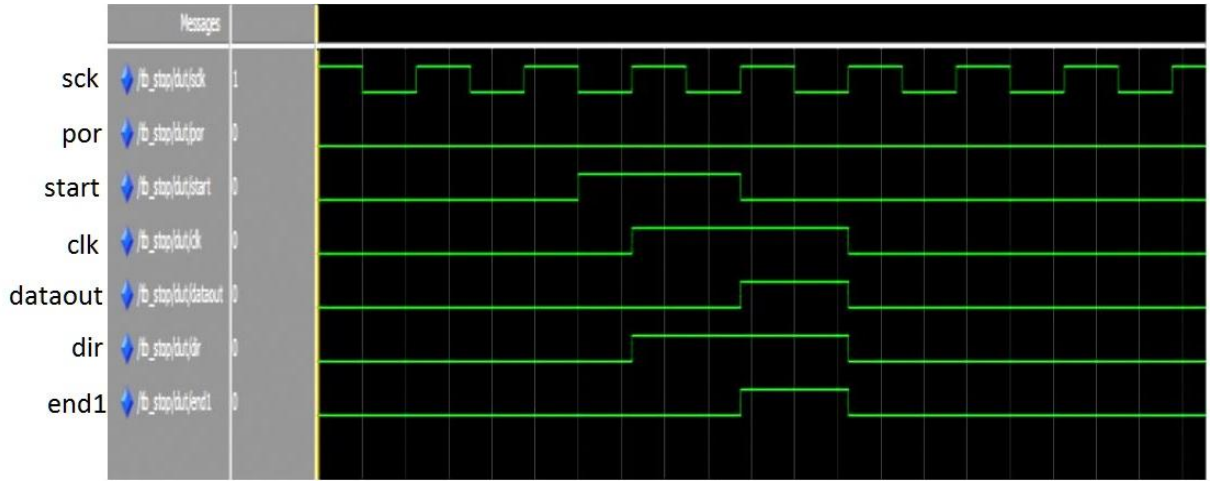

Figure 10 Waveform for stop module

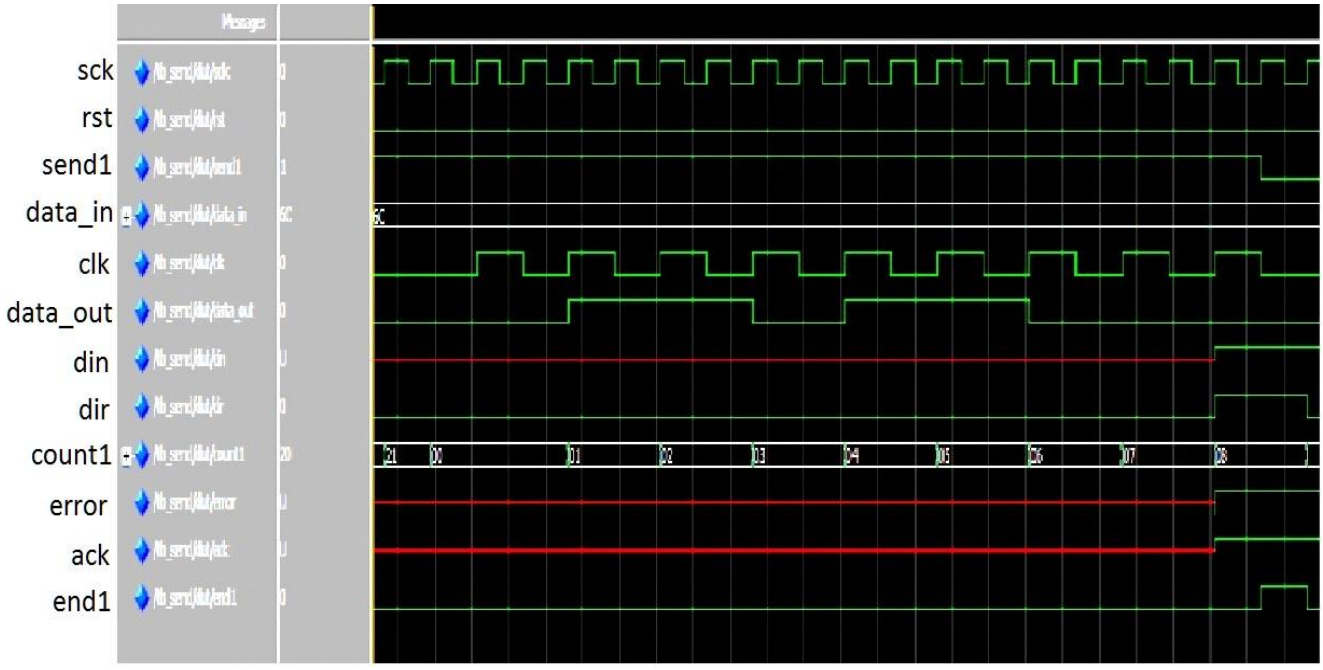

Figure 11 Waveform for write module

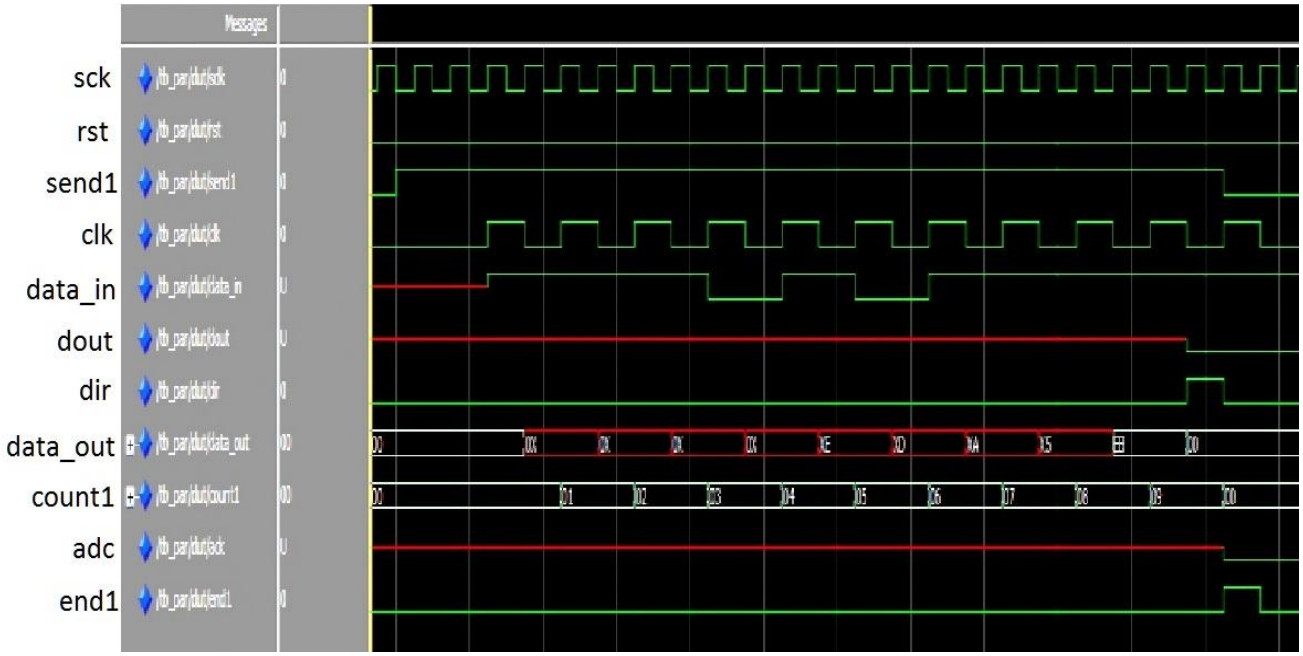

Figure 12 Waveform for read module

IV. CONCLUSION

In this paper we demonstrated how I2C Master Controller (Master) transmits the data which initializes the capacitance to digital conversion of the slave. The digitalized data is received back by the master which can be used for processing. These two tasks of transmitting and receiving data between master and slave is facilitated by the usage of many small modules that are linked to the main module. This reduces the complexity of the coding, so that we acquire the required data by just initializing the top module. In future, this can be implemented as networks that contain multiple masters and multiple slaves to co-ordinate the entire system of sensors.

REFERENCES

1. Designing Embedded Systems with PIC Microcontrollers Principles and applications by Tim Wilmshurst Amsterdam, Newnespublications 2nd edition, 2009
2. Digital Design Principles and Practices 4th Edition by John F Wakerly, Prentice Hall publication, 2005
3. AD7746 data sheet by Analog Devices
4. HDL Programming VHDLand Verilog byNazeih M Botros, Charles River Media, 2005
5. Circuit Design and Simulation with VHDL second edition Volnei A. Pedroni, The MIT press publication, 2010
6. AN10216-01 I2C Manual, Philips Semiconductor, Mar 2003
7. I2C Bus Specification, Philips Semiconductor, version 2.1, January 2000.
8. Model sim SE user manual by model technology.

9. Design and implementation of FPGA based interface model for scale-free network using I2C bus protocol on Quartus II 6.0 by VenkateswaranP.,Dept. of Electron. & Tele-Commun. Eng., JadavpurUniv.,Kolkata,India, Mukherjee M.; SanyalA.;Das S., Computers and Devices for communication ,2009 .CODEC 2009.
10. Design and implementation of a BIST embedded inter-integrated circuit bus protocol over FPGA by Saha, S. ; Dept. of Electron. &Commun. Eng., Khulna Univ. of Eng. & Technol., Khulna, Bangladesh ; Rahman, M.A. ; Thakur, A., Electrical Information and Communication Technology, (EICT) 2013.
11. A review on effectuation of serial communication Inter-IC Protocol by Ashwin Bhandekar, S.S.Thakare, D.S.Chaudhar, International Journal of Advanced Research in Computer Science and Software Engineering ,Volume 3, Issue 3, March 2013.
12. Design and Implementation of I2C Master Controller Interfaced with RAM Using VHDL by Sansar Chand Sankhyan, Research Article, International Journal of Engineering Research and Applications, Vol.4, Issue 7, July 2014, pp.67-70.
13. Designing of Inter Integrated Circuit using Verilog by Disha Malik, International Journal of Science, Engineering and Technology, Volume 02, Issue 6, July 2014.
14. Design and Simulation of I2C Protocol by Tripti Singh, International Journal for Research in Applied Science and Engineering Technology(IJRASET) , Volume 2, Issue XII, December 2014.
15. Design and Implementation of I2C Bus Controller using Verilog by Mr. J.J.Patel, Prof. B.H.Soni, Journal of Information ,Knowledge and Research in Electronics and Communication Engineering, Vol. 2, Issue 2, November 12 to October 13.