FPGA Based Sensor System for Biomedical Monitoring Using I2C and UART Protocol

Hemanshi Chugh

Department of Electronics &Telecommunication Engineering

Dr. Akhilesh Das Gupta Institute of Technology and Management, New Delhi, India

E-mail:- *hemanshi.chugh@gmail.com*

DOI: - https://doi.org/10.47531/MANTECH/ECC.2021.24

Abstract

This paper presents the design as well as the simulation of the I2C master-slave interface. The design was outlined using the VHDL (Verilog hardware description language). The I2C Interface is considered to design a sensor system that enables a patient to self-monitor and check in with a trained professional if an abnormal condition is detected (clinical monitoring). From the medical perspective, the accuracy of data, the ability to perform diagnosis and the capability of an observer to assist a patient are critical issues. Collaboration with professionals in the medical field is important to the success and adoption of these systems. Hence, a sensor system is designed to enable a patient to self-monitor and check in with a trained professional if an abnormal condition is detected with the outputs displayed on LCDs. The proposed I2C interface design was prototyped using economical, easily available Xilinx Spartan3E Starter development boards manufactured by Diligent. The prototype system compiled using two I2C "slaves" (sensor and clinical end FPGA) and I2C "master". The "master" acquires results from "slaves" and then transmits the data to a personal computer to display. The prototype system worked at 100 Kbits/sec flawlessly.

Keywords:- *Biomedical monitoring, I2C protocol, FPGA, Verilog.*

INTRODUCTION

Improvements to integrated circuit technology in the form of low-cost, high-density field-programmable gate arrays (FPGAs) with reduced package sizes have broadened microprocessor selection and digital interfacing solutions. Many simple solutions have been developed in this area for specific issues related to the health monitoring of various conditions. One example is a home-monitoring system that notifies a central source when an emergency has occurred. Additionally, the combination of computing and wearable sensors has widely increased the amount of data that can be gathered concerning the status of a patient. A major advantage of a smart sensor system is to enable a patient to self-observe and check in with a trained professional if an abnormal condition is detected. For the rapid prototyping of smart sensor systems, it is favorable to select a versatile hardware platform. The utilization of programmable devices such as field-programmable gate arrays (FPGAs) and hardware design languages such as Verilog enable the rapid prototyping of the wearable portion of these systems for use in both laboratory and clinical environments.

OBJECTIVE

The main objective of the paper is to design a sensor system that enables a user to self-monitor and check in with a trained professional if an abnormal condition is detected (i.e., clinical monitoring). The project flow includes the synchronized sampled serial communication between the pulse sensor, which counts the heartbeat per minute and FPGA (that would be a master) and displaying the pulse rate on the seven-segment display of the board. If the pulse rate exceeds a critical value, it raises a buzzer and an Inter FPGA communication from the user or patient end (USB-UART interface) to the clinical end is made, and the output is displayed on the PC at the clinical end for further monitoring.

Figure 1 illustrates that a heartbeat sensor is used to count the pulses per minute. Whenever the value exceeds a critical value, the alarm is raised and simultaneously, the User End FPGA sends this data to the clinical end FPGA over the I2C serial bus. This would require FPGA at the user end to

be configured in MASTER mode and FPGA at the clinical end to be configured in a SLAVE mode. Also, the software running on the host computer is capable of establishing a serial port connection based on received data via USB-UART communication, displays the data on PC/Laptop.

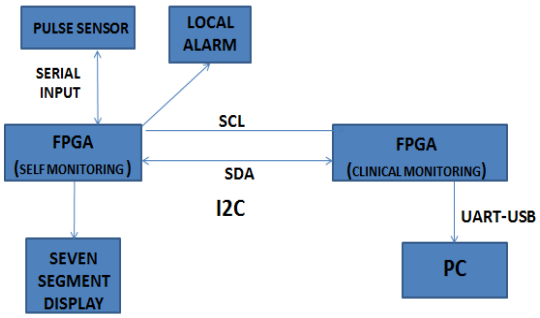

Fig. 1: Basic Building block of the System

OVERVIEW OF THE PROTOCOLS

1. I2C Protocol

I2C is easy to use to link many devices altogether since it has a built-in addressing scheme. It is a low-bandwidth, short-distance protocol.

The I2C interface [Phi:00] was initially invented at the PCB level in the 1980s by Philips Semiconductors Company for data transmission among ICs. In I2C, it takes only two bi-directional lines to carry data between devices, Serial Data (SDA) line and Serial Clock (SCL) line. Every other device is recognizable by a unique address of 7 or 10 bits. The machine undergoes a communication called the Master, and all the other devices on the bus are labeled as Slaves at that moment. I2C is a serial double wire bus, as shown in Figure 2. Chip selection or arbitration logic is not needed, making it cheap and easy to implement in hardware.

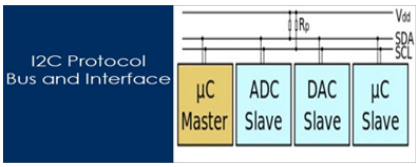

Fig. 2: I2C Bus Configuration

When a Master wants to initiate a communication, it issues a “START” condition (Fig 3). At that time, all devices, including the other Masters, have to listen to the bus for incoming data. After the “START” is issued, the Master sends the “ADDRESS” of the Slave that it wishes to communicate with along with a bit to indicate the direction of the data transfer (either read or write). All Slaves will then compare their addresses with the address received on the bus. If the addresses are identical, the Slave with the matching address will send an “ACKNOWLEDGEMENT” (ACK) to the Master. Slaves whose addresses do not match will not send an ACK. Once communication is established, the two lines are busy. No other device is allowed to control the lines except the Master and the Slave, which was selected. When the Master wants to terminate communication, it will issue a “STOP” signal (Fig 3.2). After that, both the SCL line and SDA line are released and free.

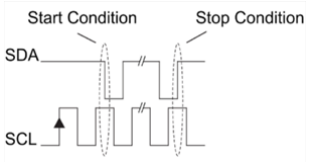

Fig. 3: Initiation and Termination of data

A Complete Data Transfer

Figure 4 illustrates a complete transmission of data from the transmitter to the receiver. The Master must generate the SCL signals, Start and Stop signals, and the first byte. The first byte's Acknowledgement must be created by the Slave when he recognizes his address on the bus. The other Acknowledgements are generated by the receiver. See below figure 4.

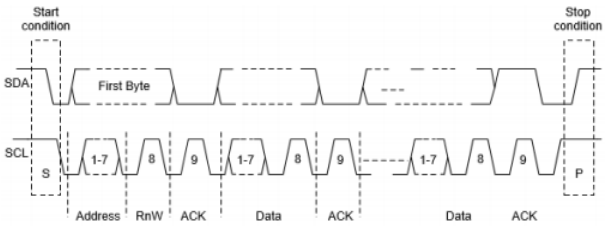

Fig. 4: A complete data transfer in 7-bit addressing mode [Phi:00]

2. UART-USB Protocol

A universal asynchronous receiver and transmitter (UART) is a serial line circuit that sends parallel data. A universal asynchronous receiver and transmitter (UART) is a serial line circuit that sends parallel data. UARTs are often used in conjunction with the RS-232 standard EIA (Electronic Industries Alliance), which specifies the electrical, mechanical, functional and procedural characteristics of two data communication devices. As the voltage level in RS-232 is different from that of FPGA 110, a voltage converter chip is essential between a serial port and an FPGA's 110 pins.

The easiest way to switch a device to USB is to emulate RS-232 over the USB bus. To mimic a COM port, the USB UART system uses a USB interface. When the first time a new device connects to a Windows PC, Windows will ask you to provide a driver. The Nexys4 features a USB-UART FTDI FT2232HQ (attached to J6

connector) bridge that uses PC applications to communicate with the board using normal WindowsCOM port commands. USB-COM port drivers convert UART/serial port data to USB packets. The FPGA exchanges serial port data via a two-wire serial port (TXD/RXD) and optional hardware flow control (RTS/CTS). After the drivers are installed, I/O commands can be used to generate serial data traffic on the C4 and D4 FPGA pins from the PC addressed to the COM port.

SIMULATION

Considering a top-bottom approach, the prototype consists of two modules, namely Patient end (Master) and Clinical end (Slave). The master module further bifurcates into sensor interfacing module, raising alarm module, 7segment module and I2Cmaster transmission module. The slave module divides into an I2C slave reception module, a 7segment module and a USB-UART module.

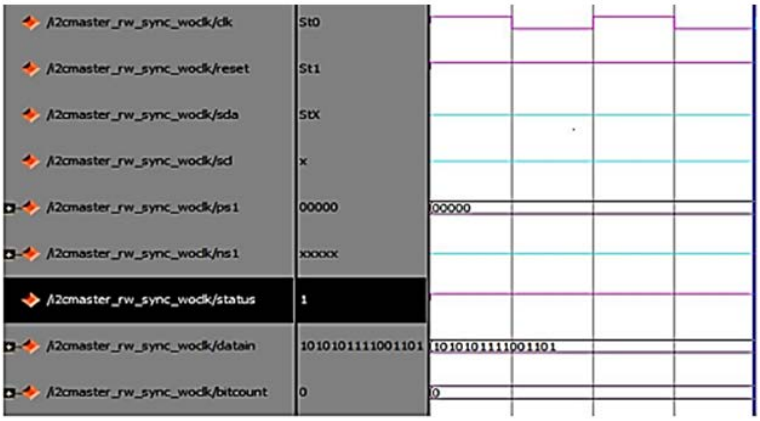

Fig. 5: Simulation Window for I2C Master, Write Mode and Reset High.

Fig. 6: Simulation Window for I2c Master, Write Mode and Reset Low till Ack is received

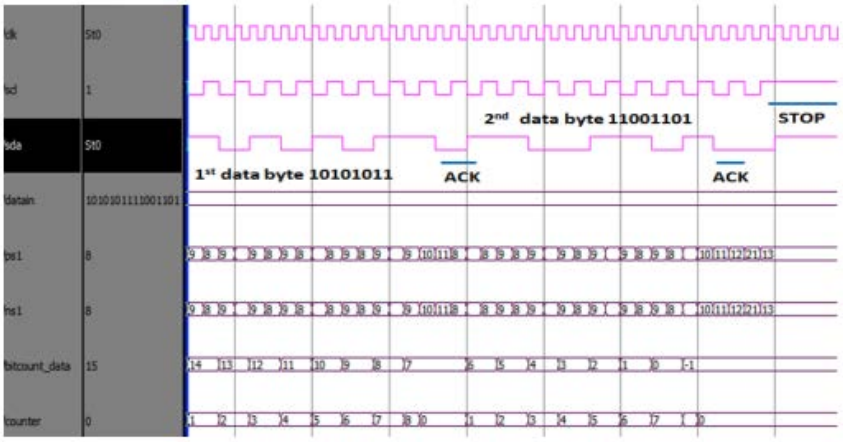

Fig. 7: Simulation Window for I2C Master, Data Transmission after Address Has Been Acknowledged

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	209	1,920	10%	
Number of 4 input LUTs	552	1,920	28%	
Number of occupied Slices	358	960	37%	
Number of Slices containing only related logic	358	358	100%	
Number of Slices containing unrelated logic	0	358	0%	
Total Number of 4 input LUTs	667	1,920	34%	
Number used as logic	552			
Number used as a route-thru	115			
Number of bonded IOBs	26	83	31%	
Number of BUFGMUXs	2	24	8%	
Average Fanout of Non-Clock Nets	3.08			

Fig. 8: Synthesis of Master Controller

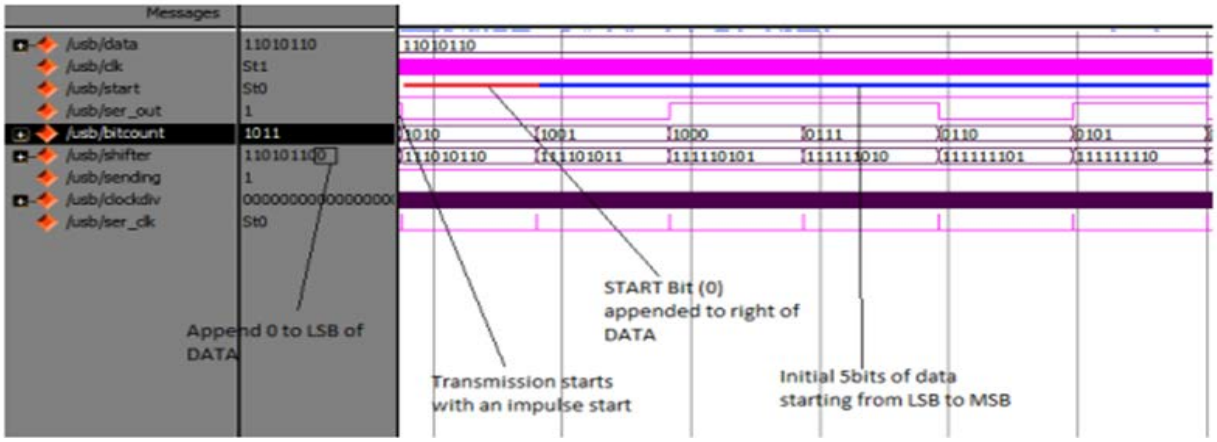

Fig. 9: Simulation Window for UART Transmission with 1 Start and 5 Bits of Data (Continued In Next Figure)

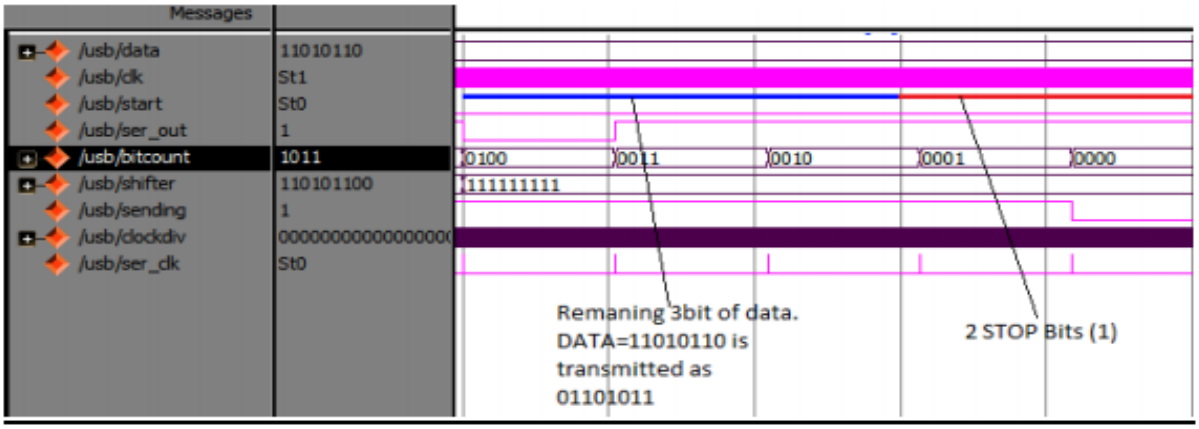

Fig. 10: Simulation Window for UART Transmission with remaining 3 Bits of Data and 2 STOP Bits

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	126	126800	0%
Number of Slice LUTs	231	63400	0%
Number of fully used LUT-FF pairs	116	241	48%
Number of bonded IOBs	24	210	11%
Number of BUFG/BUFGCTRL/BUFHCEs	2	128	1%

Fig. 11: Synthesis of Slave (Clinical End)

THE PROTOTYPE SYSTEM

Figure 5 to Figure 11 illustrates the setup used to test the prototype. It consists of a pulse sensor, Spartan 3E board; for the Master, a buzzer, Artix7 board; for the Slave, and a Laptop connected to it.

STEP 1: Sensor Interfacing With FPGA Master:

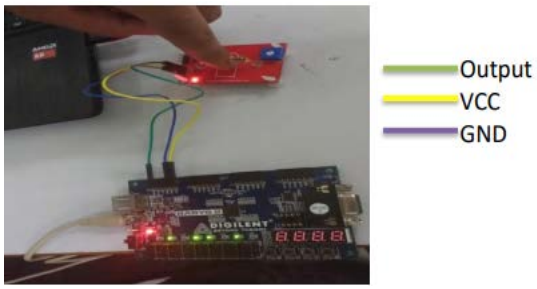

Fig. 12: Sensor Interfacing

STEP 2: Seven Segment Display of the Pulse Sensor:

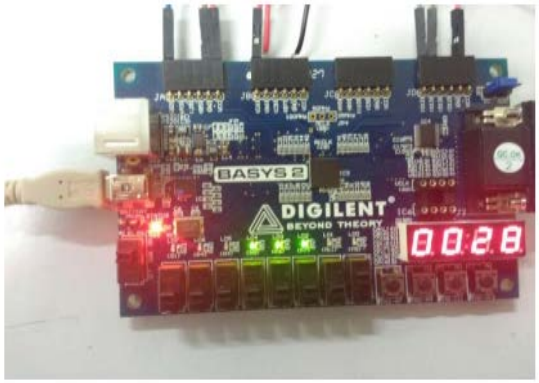

Fig: 13: Seven Segment Display of the Pulse

STEP 3: I2C Master Prototype:

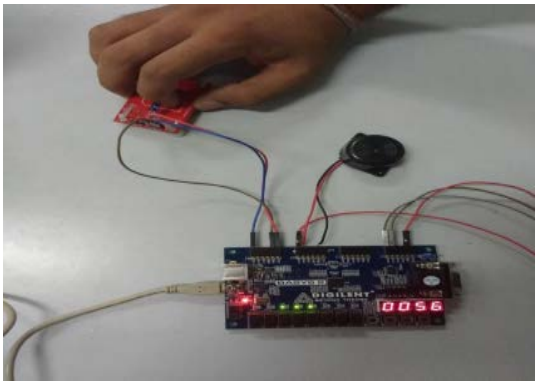

Fig. 14: I2C Master Working

STEP 4: I2C Slave Prototype (I2C Communication):

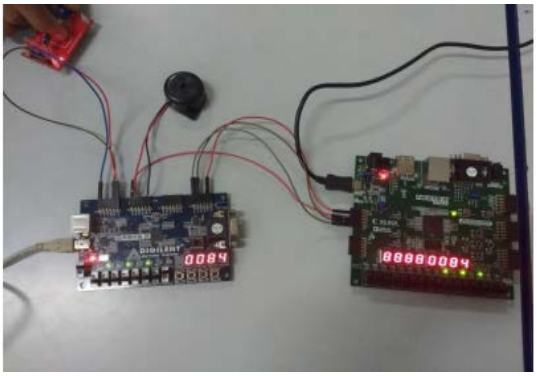

Fig. 15: I2C Communication

STEP 5: UART-USB Communication:

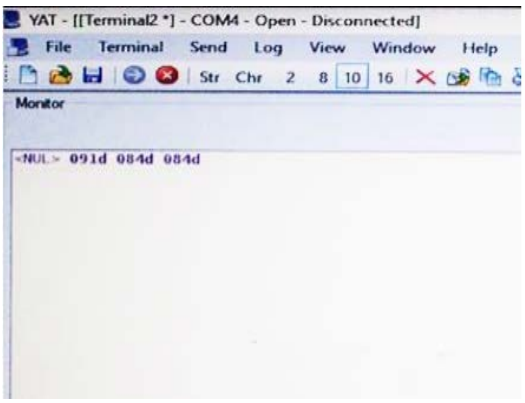

Fig. 16: UART-USB Communication

STEP 6: The Overall Prototype:

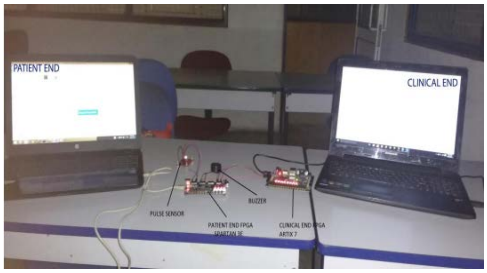

Fig. 17: The Complete Prototype

SUMMARY

The FPGA platform provides an environment for rapid prototyping of sensors for biomedical applications with numerous advantages. An I2C interface (described using Verilog®) and a USB-UART interface have been developed and fully tested on inexpensive, readily available Xilinx FPGA development boards. The I2C interface was tested at an operating frequency of 100 kHz in 7-bit address mode, and the USB-UART interface was tested at a 9600 baud rate. The I2C Interface not only simulates successfully but operates correctly when implemented on a Xilinx FPGA development board.

CONCULSION

The FPGA platform provides an environment for rapid prototyping of sensors for biomedical applications with numerous advantages. An I2C interface (described using Verilog®) and a USB-UART interface have been developed and fully tested on inexpensive, readily available Xilinx FPGA development boards. The I2C interface was tested at an operating frequency of 100 kHz in 7-bit address mode, and the USB-UART interface was tested at a 9600 baud rate. The I2C Interface not only simulates successfully but operates correctly when implemented on a Xilinx FPGA development board. The focus of this thesis is on the PATIENT END of the prototype. The data is collected from the pulse sensor at 8 second time

period. The data is displayed on the 7 segment display of the patient end FPGA, which is a SPARTAN 3E BASYS 2 BOARD. If the receive pulses value exceeds a critical value of (80) and the alarm is raised using a buzzer, and data is transmitted to the CLINICAL END using I2C Protocol at 100KHz. The alarm is pulled down once the value of pulses per minute comes below 80.

REFERENCES

1. Xilinx, “Spartan-3E FPGA Starter Kit Board User Guide”, (June 20, 2008).
2. Phillips Semiconductors, “THE I2C-BUS SPECIFICATION”, Version 2.1, (January 2000).
3. Bollam Eswari, N.Ponmagal, K.Preethi, S.G.Sreejeesh, “Implementation of I2C Master Bus Controller on FPGA”, IEEE, International conference on Communication and Signal Processing, April 3-5, 2013, India.
4. Kimberly Newman, Nathan Laramie and Casey medina, “Rapid Prototyping of an FPGA based sensor system for Biomedical Monitoring”, WSEAS International Conference on Mathematical Biology and Ecology, Miami, Florida, USA, January 18 - 20, 2006.
5. Frédéric Leens, “An Introduction to I2C and SPI Protocols”, IEEE Instrumentation & Measurement Magazine, February 2009.
6. Nikolaos Alachiotis, Simon A. Berger, Alexandros Stamatakis, “EFFICIENT PC-FPGA COMMUNICATION OVER GIGABIT ETHERNET”, 10th IEEE International Conference on Computer and Information Technology (CIT 2010).
7. I2C-bus specification and user manual, NXP Semiconductor, Rev. 6 — 4 April 2014.
8. FPGA PROTOTYPING BY VERILOG EXAMPLES, Pong P. Chu.
9. Regu Archana, Mr. J.V.Rao, “Implementation of I2C Master Bus Protocol on FPGA”, Int. Journal of Engineering Research and Applications, ISSN: 2248-9622, Vol. 4, Issue 10 (Version 5), October 2014, pp.06-10.
10. “Implementation of I2C Master Bus Controller on FPGA” in IEEE, International Conference on Communication and Signal Processing, April 3-5, 2013, India.
11. Wang, Lu, Lisheng Xu, Dazhe Zhao, Yang Yao, and Dan Song. "FPGA-based design and implementation of arterial pulse wave generator using piecewise Gaussian–cosine fitting", Computers in Biology and Medicine, 2015.
12. Uria-Rivas, Rodriguez-Sanchez, Santos, Vaquero, Boticario. "Impact of Physiological Signals Acquisition in the Emotional Support Provided in Learning Scenarios", Sensors, 2019.