Design Implementation of UART and SPI in single FGPA

M.Poorani
ECE Department,
Sri Manakula Vinayagar Engineering College,
Puducherry, India
pooranimuthukumarasamy.10@gmail.com

Mrs.R.Kurunjimalar
Associate Professor, ECE Department,
Sri Manakula Vinayagar Engineering College,Puducherry,
Research Scholar, SCSVMV University,
Kancheepuram , India
kurunjirajmeera13.03@gmail.com

Abstract— **The UART(Universal Asynchronous Receiver and transmitter) controller is the key component of the serial communications subsystem of a computer. The UART takes bytes of data and transmits the individual bits in a sequential fashion. SPI is a common technology used nowadays for communication with peripheral devices where we want to transfer data speedily and with in real time constraints. In the existing work data acquisition system for underground has been designed for counters triggered by surface detectors and they have used UART and SPI protocol for the communication with dedicated processor. In this paper we have used Xilinx 12.4i and SPARTAN 3E FPGA to implement the whole system. The hyper terminal is used to check the UART protocol and 12 bit DAC MCP4922 used for check SPI protocol. The proposed work can provides both protocol effectively for the wireless serial communication. These serial protocols mainly used in zigbee wireless technology.**

Key words: **UART, SPI, Xilinx, Hyper link, Zigbee**

I. INTRODUCTION

An Universal Asynchronous Receiver and Transmitter is a device that allows transmission and reception of information, in an asynchronous and serial way. UART are used in asynchronous serial data communication between remote embedded systems. The UART can be used to control the process of breaking parallel data from the PC down into serial data that can be transmitted. It consists of one transmitter module and receiver module. UART has been an important input/output tool for decades and it is widely used. SPI stands for Serial Peripheral Interface. It is a synchronous protocol that allows a master device to initiate communication with slave devices.

SPI is a full duplex, serial bus commonly used because of its simple hardware protocol flexibility and interface requirements. SPI consists of two blocks ,the SPI master and the SPI slave.SPI protocol specifies four signal wires MISO -Master Out Slave In (output from master), MISO – Master In Slave Out (output from slave), SCLK - Serial Clock (clock output from master) and SS - Slave Select (active low, output from master.The SPI Master block generates the control signals to interface to external slave devices using the serial data out port (MOSI), serial data in port (MISO), output clock (SCLK) and slave select (SS)..The SS signal must be used if more than one slave exists in the system. This signal is most often active low, so a low on this line will indicate the SPI is active, while a high will signal inactivity.

The main advantage of the UART- SPI interface is that it can fit in any application where an SPI device has to be used. As the UART-SPI interface can be used to communicate to SPI slave devices from a PC with UART port it can be used for typical applications like interfacing of EEPROM, flash memories and sensors.

II. UART

An UART (Universal Asynchronous Receiver/ Transmitter) is the microchip with programming that controls a computer's interface to its attached serial devices. UART is an integrated circuit designed for implementing the interface for serial communications. It provides the computer with the RS-232C Data Terminal Equipment (DTE) interface so that it can "talk" to and exchange data with modems and other serial devices . As part of this interface, the UART also: Converts the bytes it receives from the system along parallel circuits into a single serial bit stream for outbound transmission On inbound transmission, converts the serial bit stream into the bytes that the system handles. Adds a parity bit (if it's been selected) on outbound transmissions and checks the parity of incoming bytes (if selected) and discards the parity bit Adds start and stop delineators on outbound and strips them from inbound transmissions May handle other kinds of interrupt and device management that require

coordinating the on-chip communication of operation with high speed devices. Wait until the incoming signal becomes '0 ' (the start bit) and then start the sampling tick center. When the center reaches 7, the incoming signal reaches the middle position of the start bit. Clear the center and restart.

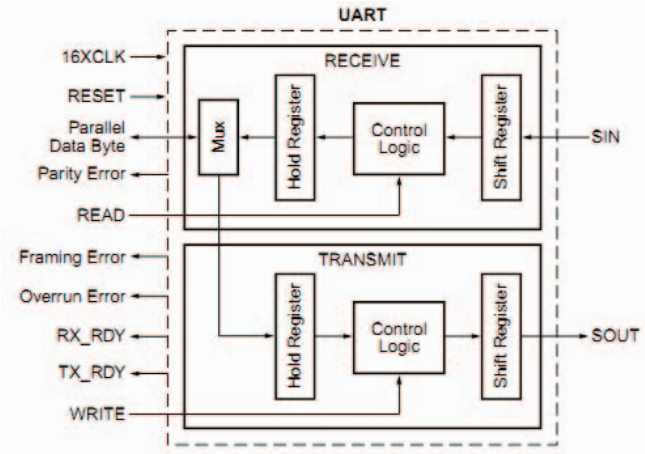

Figure1. UART Block diagram

The UART includes both transmitter and receiver. The transmitter is a special shift register that loads data in parallel and then shifts it out bit-by-bit. The receiver shifts in data bit-by-bit and reassembles the data byte • Wait until the incoming signal becomes '0 ' (the start bit) and then start the sampling tick center. When the center reaches 7, the incoming signal reaches the middle position of the start bit. Clear the center and restart.

III. SPI

SPI stands for Serial Peripheral Interface.SPI is a synchronous protocol that allows a master device to initiate communication with a slave device. Data is exchanged between these devices. SPI is implemented by a hardware module called the Synchronous Serial Port or the Master Synchronous Serial Port. This module is built into many different micro devices. It allows serial communication between two or more devices at a high speed and is reasonably easy to implement. SPI is a Synchronous protocol.

The clock signal is provided by the master to provide synchronization. The clock signal controls when data can change and when it is valid for reading . Since SPI is synchronous, it has a clock pulse along with the data. RS-232 and other asynchronous protocols do not use a clock pulse, but the data must be timed very accurately. Since SPI has a clock signal, the clock can vary without disrupting the data. The data rate will simply change along with the changes in the clock rate. This makes SPI ideal when the microcontroller is being clocked imprecisely, such as by a RC oscillator.

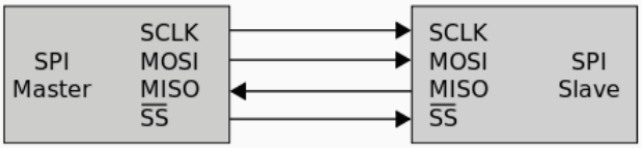

Figure 2. SPI Block Diagram

- SI - Serial Data Input (used to transfer data into the SPI device).
- SO - Serial Data Output (used to transfer data out of the SPI device).
- CS - Chip Select Input (for enabling device operation).
- W- Write Protect Input (used to guard against program/erase instructions).
- HOLD - Hold Input (to pause SPI transaction).

SPI is a Master-Slave protocol. Only the master device can control the clock line SCK. No data will be transferred unless the clock is manipulated. All slaves are controlled by the clock which is manipulated by the master device. The slaves may not manipulate the clock. The SSP configuration registers will control how a device will respond to the clock input. SPI is a Data Exchange protocol . As data is being clocked out, new data is also being clocked in. When one "transmits" data, the incoming data must be read before attempting to transmit again. If the incoming data is not read, then the data will be lost and the SPI module may become disabled as a result. Always read the data after a transfer has taken place, even if the data has no use in your application. Data is always “exchanged” between devices. No device can just be a "transmitter" or just a "receiver" in SPI. However, each device has two data lines, one for input and one for output. These data exchanges are controlled by the clock line, SCK, which is controlled by the master device. Often a slave select signal will control when a device is accessed. This signal must be used for when more than one slave exists in a system, but can be optional when only one slave exists in the circuit.

IV. THE EXISTING TECHNIQUE

In this method synchronous (Master/Slave) data acquisition system for underground muon counters triggered by surface detectors. A water Cherenkov surface detector (Master) when hit by extensive air showers generates and transmits the trigger to the underground system (Slave) to register the muon fraction of the showers which survived the propagation through 3 m layer of soil. The surface and underground segments are built on the

Altera field-programmable gate array (FPGA) CycloneIII/CycloneIV platforms, respectively. Both FPGAs are equipped with NIOS processors, which makes previously used external microcontrollers unnecessary. They also generate necessary interfaces: SDRAM controller, UART,SPI,DMA; interfaces which were previously implemented by means of logic elements. Moving several time-consuming tasks from the logic block (coded in the Altera Hardware Description Language) to the NIOS (coded in "C") dramatically simplifies the system and increases its flexibility. The time margin for all processes managed by the soft-core NIOS for the 100 Hz T1 trigger rate remains sufficient. NIOS processors communicate with each other via UART protocol and by the RS485 standard. The underground CycloneIV FPGA is programmed remotely via additional MAXII CPLD with nonvolatile programmable memory.

Tests have shown that a full string of data processing goes smoothly. The above-mentioned string involves the following processes: the transfer of the trigger with a time stamp from the surface detector into the underground segment via a dedicated line with a galvanic barrier, freezing data from 64 channels at 320 MHz sampling in internal DPRAMs, writing/reading data into/from external SDRAM, extracting physical data identified by GPS time stamps sent from Central Data Acquisition System (CDAS) and the data transfer from the underground NIOS via the surface NIOS to CDAS.

IV. THE PROPOSED TECHNIQUE

In the proposed system we are going to implement both UART and SPI communication in a single FGPA board. Xilinx code is being used for implementing these protocols in the FPGA kit.

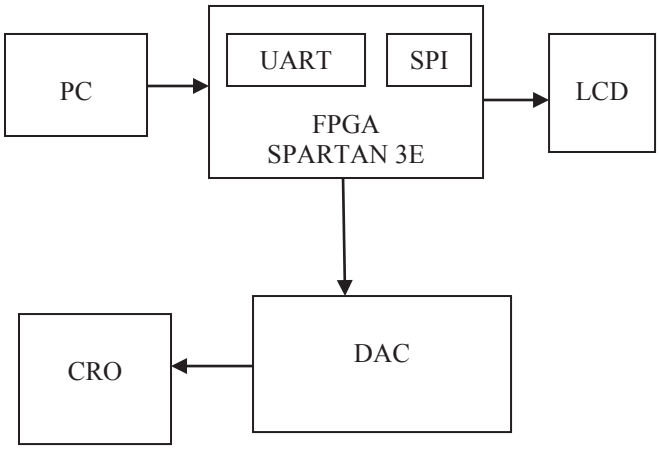

Figure 3. Block Diagram of Design Implementation of UART and SPI

The key is being given to the FPGA kit through the PC by opening the Hyper terminal. The key is being sent to the FPGA by using a RS 232 Cable. In the FGPA kit UART and SPI is being implemented using Xilinx code. Through the RS 232 the data is being sent bit by bit to the UART that is being present in the FPGA. To verify whether the data is being transmitted correctly the values is being send to DAC. Here it is important that we chose the DAC in such a way that it will support the SPI protocol. The DAC converts the analog data into digital data and it sends it to the CRO.

For the SPI the data from UART is being sent. To verify the implementation of SPI in the FGPA kit a LCD is being connected. The value will be displayed in the LCD.

V. HARDWARE DESCRIPTION

A field-programmable gate array (FPGA) is an integrated circuit designed to be configured by a customer or a designer after manufacturing – hence "field-programmable". The FPGA configuration is generally specified using a hardware description language(HDL), similar to that used for an application-specific integrated circuit (ASIC).FPGAs contain an array of programmable logic blocks, and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together", like many logic gates that can be inter-wired in different configurations. Logic blocks can be configured to perform complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory.

This Design Implementation of UART and SPI in Single FGPA is implemented using the hardware FPGA Spartan 3E as shown in figure 4.

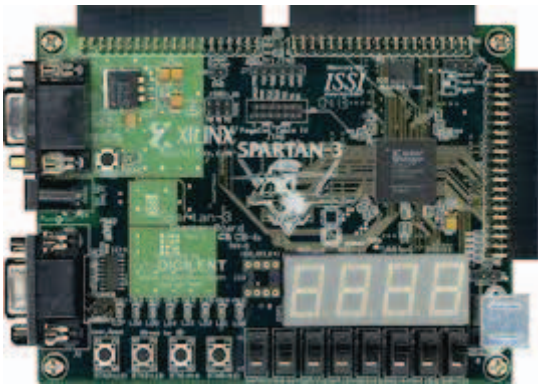

Figure 4. FPGA Spartan 3

Spartan-3E FPGAs are programmed by loading configuration data into robust, reprogrammable, static CMOS configuration latches (CCLs) that collectively control all functional elements and routing resources. The FPGA's configuration data is stored externally in a PROM or some other non-volatile medium, either on or off the board. After applying power, the configuration data is written to the FPGA using any of seven different modes:

- Master Serial from a Xilinx Platform Flash PROM
- Serial Peripheral Interface (SPI) from an industry-standard SPI serial Flash
- Byte Peripheral Interface (BPI) Up or Down from an industry-standard x8 or x8/x16 parallel NOR Flash
- Slave Serial, typically downloaded from a processor
- Slave Parallel, typically downloaded from a processor
- Boundary Scan (JTAG), typically downloaded from a processor or system tester.

Furthermore, Spartan-3E FPGAs support MultiBoot configuration, allowing two or more FPGA configuration bitstreams to be stored in a single parallel NOR Flash. The FPGA application controls which configuration to load next and when to load it.

Here we also use an LCD interfaced to an FPGA to display the Letter.

Figure5. Liquid Crystal Display

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters animations and so on. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data. The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.

5. EXPERIMENTAL RESULTS

Thus the binary data values of the ASCII values are generated and can be viewed through the Xilinx software as shown in fig 6.

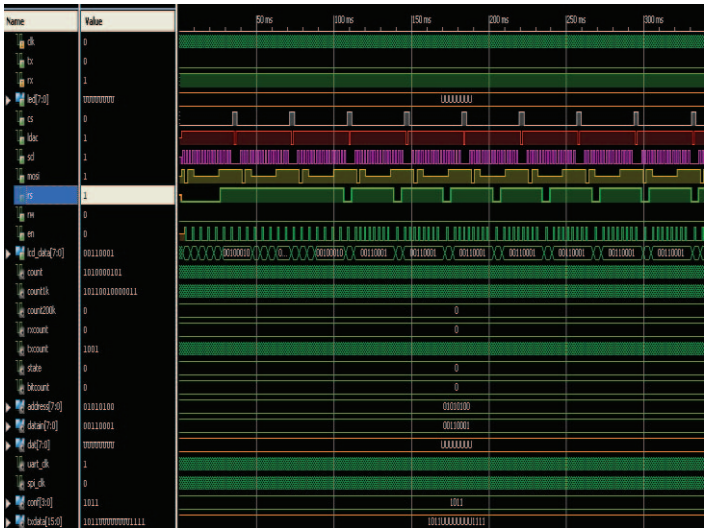

Figure6. Simulation using Xilinx software

5. Conclusion

The design implementation of UART and SPI is being done using Xilinx code in a single FPGA. This helps us to minimize the number of processors and by doing this the cost of the system is also minimized. By doing this serial communication can be effectively done in wireless sensor networks. This FPGA can be used for many application oriented processors.

Authors

M.POORANI completed her B.Tech in Electronics and communication Engineering from Sri Manakula Vinayagar Engineering College, Puducherry, India in 2013. She is now pursuing her Master of Technology (M.Tech) in VLSI & Embedded Systems at Sri Manakula Vinayagar Engineering College, Puducherry, India.

.

R. KURINJIMALAR received the B.E. (ECE) degree from Bharadhidasan University, Thiruchirapalli, Tamilnadu, INDIA, in 1997, then joined as Plant Manager in VENUS INDUSTRIES, Hosur, Tamilnadu, INDIA. Then joined as Lecturer in Bharathiyar College of Engineering & Technology, Karaikal, Puducherry State, INDIA by 1999. She completed MBA (HR) by 2005 and M.E (Communication Systems) degree from Vinayaga Mission University, Salem, Tamilnadu, INDIA, in 2007.She is currently a Research scholar SCSVMV University, Kancheepuram. She is currently Purusing her Ph.d in She is currently Assistant Professor with Sri Manakula Vinayagar Engineering College, Puducherry State, INDIA. Her current interests include Wireless Communication Networks and Security. She serves as a member of ISTE & Organized two FDPs.

Acknowledgment

I would like to thank Mrs.R.Kurinjimalar, Associate Professor, ECE Department who had been guiding me throughout the project and supporting me in giving technical ideas about the paper and motivating me to complete the work efficiently and successfully.

References

[1] Dr G.B.Wakle, I. Aggarwal and S. Gaba, “Synthesis and Implementation of UART using VHDL Codes”, International Symposium on Computer, Consumer and Control, pp.1-3, June 2012.

[2]T.P.Blessington, B.B.Murthy, G.V.Ganesh and T.S.R Prasad, “Optimal Implementation of UART-SPI Interface in SOC”, Devices, Circuits and Systems (ICDCS), International Conference, pp.673-67, 2012.

[3] Design and simulation of UART serial communication module based on VHDL - Fang Yi-Yuan, Chen Xue, IEEE Explore, may 2011.

[4] Y. Wang and K. Song, “A New Approach to Realize UART,” in Electronic and Mechanical Engineering and Information Technology (EMEIT), International Conference, vol. 5, pp. 2749 –2752, 2011.

[5] Design and test of general purpose SPI master/slave IPs on OPB Bus- systems signals and devices, 7th international multi conference, 2010.

[6] A.K Oudjida et ai, Master-Slave wrapper communication protocol: A case-study, Proceedings of the 1'1 IEEE International Computer Systems and Information Technology Conference ICSIT'05, PP 461-467, 19-21 July 2006.

[7] F. Leens, "An Introduction to SPI Protocols,"IEEE Instrumentation & Measurement Magazine, pp. 8-13, February 2009.

[8] A.K. Oudjida et ai, FPGA Implementation of I2C & SPI protocols A Comparative Study". Proceedings of the 16th edition of the IEEE International Conference on Electronics Circuits and Systems ICECS, pp.507 -510, Dec 13-16 2009.

[9] REN Yu-fei,ZHANG Xiang,CHENG Nai-ping (Department of Optical and Electrical Academy of Equipment Command &Tech, Beijing 101416, China); Design and Realization of Two-way Transmission SPI Interface; Telecommunication Engineering; 2009.

[10] Zhang Rui;A Method to Realize DSP Communicating with Other Device by SPI Interface Protocol [J]; International Electronic Elements; 2003-08.

[10] A micro- FT- UART for safety critical SOC basedApplications, www.doi.ieeecomputersociety.org.

[11] www.xilinx.com/support/documentation/ipdocumentation/xpspi.pdf