Interrupt Enabled Priority Based Master Slave Communication using SPI Protocol

Deepika, Jayanthi K Murthy

Abstract*: Serial Peripheral Interface or SPI is a synchronous serial communication protocol that provides full – duplex communication at very high speeds. It is a master – slave type protocol that provides a simple and low-cost interface between a microcontroller and its peripherals. This paper proposes the design of a priority-based master slave communication system using SPI Protocol that enables the system to operate using interrupts. The design mainly emphasizes on priority-based communication where the slaves will generate an interrupt over a newly defined interrupt pin when some data transfer needs to happen. When the master receives an interrupt from the slave it establishes communication with one slave at a time based on the priority and the priority to each slave is assigned by the arbiter or priority control block. The highest priority slave is served first. Shift register is used to store and transfer the data bit by bit and it resets every time a data transfer is complete. The design proposed here can be implemented in different applications which involve the peripherals that can support SPI protocol for communication such as LCDs, Analog to Digital Converter, Digital to Analog Converter, Memory Cards, Temperature Sensor, Pressure Sensor. In this work, a single master multi slave architecture is considered. The design given in this paper can be scaled up to support more than four slaves.

Keywords: Arbiter, Master-slave communication, Priority based communication, Serial peripheral interface (SPI).

I. INTRODUCTION

For serial communication two protocols namely Serial Peripheral Interface (SPI) and Inter-Integrated Circuit (I2C) are widely used [1]. Both protocols are well-suited for communications between integrated circuits and with on-board peripherals. The I2C bus uses two signals, a serial clock signal SCL and a serial data signal SDA, to transfer data among multiple devices. As compared to I2C, SPI uses four signals for the data transfer among multiple devices, the signals are Master Out Slave In (MOSI), Master In Slave Out (MISO), Serial Clock (SCLK) and Slave Select or Chip Select (SS/CS) which is usually an active low signal. The devices using SPI for communication are in a Master-slave relationship. The master device is usually a microcontroller and the slave can be any of the peripherals. SPI protocol also provides high speed data transfers at the rate of 10Mbps and it is faster than I2C and UART. SPI supports full duplex communication where the data on MOSI and MISO can be transferred simultaneously, in other words a two-way communication can take place at the same time. In SPI protocol, usually the master always initiates the communication. The master generates clock signal which is of a frequency the slave devices support. The slave device with which the communication must be started is selected by pulling the slave select pin to low state. The slave devices which are not selected by the slave select signal will reject the clock signal. Devices operating using SPI can be in single master, single slave configuration which is the simplest form or can have single master multiple slave configuration. In the second type, they can be connected using daisy chain configuration or parallel configuration (independent slave configuration) in which each slave will have independent slave select line. To design a priority-based master slave communication and in order to generate interrupts from the slaves a new pin called interrupt pin can be defined. This design can be used in application which require master to communicate with a slave with the highest priority. As an example of use case, consider a system which has a master/microcontroller and is connected to different slaves. Assuming the slaves are temperature sensor and an alarm. The system works using SPI interface. In this system the temperature needs to be continuously monitored and once it crosses a threshold value, an alarm should sound. In such a case the master/microcontroller should switch the communication with alarm since it has the higher priority because the temperature reached a threshold value. In this way the priority-based master slave communication using SPI interface can be used in different applications according to the requirement.

II. LITERATURE SURVEY

In the past, works have been carried out in communication protocol field for better performance and efficiency. SPI protocol is one of the major evolving topics when it comes to short distance communication with high speed transmission. The main features of SPI protocol have been a topic of research under light weight communication protocols such as UART and I2C. Full duplex communication with high speed transmission and multiple slave handling capability of SPI protocol contributes to the research areas.

Since light weight protocols share similar features for communication between two devices, some works emphasize on combining two protocols for different peripherals supporting different protocols.

Revised Manuscript Received on June 30, 2020.

Deepika, Student, Master of Technology, Department of Electronics and Communication, B.M.S College of Engineering, Bengaluru, India.

Dr. Jayanthi K Murthy, Associate Professor, Department of Electronics and Communication, B.M.S College of Engineering, Bengaluru, India.

This has been the main aim of the work discussed in paper [1]. The purpose is to design and simulate a Protocol Conversion Unit (PCU) for seamless communication between the two widely accepted serial communication protocols SPI and I2C. The design in this paper can be upgraded to support large number of peripherals by providing a First In First Out (FIFO) Queue for storing commands and data along with corresponding addresses of peripheral devices in the Protocol Conversion Unit (PCU). Communication protocols play an important role in IoT applications and this paper [2] proposes the enhancement of Universal Asynchronous Transmitter Receiver (UART) and Serial Peripheral Interface (SPI) protocols for improving the node performance in the Internet of Things (IoT) platform. This design is capable of handling scalability problems of up to two SPI Slaves. In paper [3], a data acquisition-based method is proposed to analyze serial bus communication and the information transmitted through serial bus with standalone clock signals. When using SPI protocol, the internal functionalities also need to be considered such as modes, clock speeds, register fields or flags and other signals. The design given in paper [4] allows communication with user specified number of slaves, which may require independent SPI modes, data widths, and serial clock speeds. This provides the capability to extend the number of slaves that can be supported in the design. Designs with multiple slave handling capability can be implemented using different approaches but SPI supports master to communicate with only one slave at a time. In all the existing works, only the master can initiate the communication with a slave by selecting that slave using SS pin. A discussion on the design approaches that can offer effective ways of controlling SPI-bus, incorporating the flexibility of handling two slaves at a time has been carried out in paper [5] where the design is implemented on FPGA. The simulations in this work show considerable improvement in speed. The proposed design also has multi slave handling capability with no performance degradation.

Communication protocols play a key role in FPGA implementation which is discussed in paper [6] where a novel architecture for an I2C/SPI interface for APB AMBA bus is presented, showing how it is possible to merge flexibility and reduced area occupation. Large part of this work is centered on the testing and FPGA prototyping of the IP. ASIC/SoC applications need some type of communication protocols for data transmission, be it on-chip or off-chip communications. SPI is used in many off-chip communication applications involving ASIC/SoC. This paper [7] introduces high-quality SPI Master/Slave IPs that incorporate all necessary features required by modern ASIC/SoC applications. The purpose of this paper is to provide a full description of an up-to-date SPI Master/Slave FPGA implementation.

III. IMPLEMENTATION DETAILS

In order to implement the priority based master slave communication system, there are two approaches with the existing SPI protocol. One way to generate the interrupt signal is to use the existing pin MISO over which the slave can request the master for communication with some defined commands using wishbone interface. This method has the disadvantage that MISO becomes dedicated to send commands when slave initiates any communication and the system cannot be called as full duplex anymore. Another approach is to define a new pin connection between the master and the slave called “interrupt pin”. This can be used by the slave for requesting the master and after receiving the request, the master can send interrupt grant response to establish a connection for data transfer depending on the mode and state of operation. In this case, the master works in two modes i.e., Idle and Wait for interrupt and the state of operation could be Read or Write.

To assign priority to the slave an arbiter module is defined wherein each slave is assigned a particular priority from highest to lowest. The assignment of the priority depends on the user requirement and this module is reconfigurable. The slave with highest priority is served first and then the order of priority is followed. During a communication, if the master receives an interrupt from another slave, it first checks for the priority of the slave. If it has got a higher priority the master establishes communication with that slave. Otherwise it finishes the current data transfer and then checks if the slave with second priority has some data to transfer.

The use of SPI interface for communication between master and slave with various registers and signals is shown in Fig. 1. The interrupt signal INTR is described in the figure which sends the interrupt request signal from slave to master.

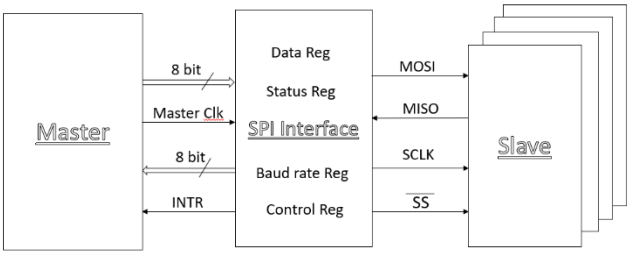

Fig. 1. Master-Slave Design with SPI Interface

In general, the following events occur when data transfer happens using SPI:

- The SS line is turned down to select the slave.
- Then as soon as the SS turns down the clock starts ticking the SCLK line.
- Then with the first positive edge of SCLK line read-write bit along with address is sent on the line. The MOSI lines remain null until the SCLK line is fired up. MISO line is used if slave sends some data.
- Then the data bits are sent one by one.

It is necessary to design the interface for communication between master and slaves based on priority. To design various blocks, RTL code for each module and test bench has been written using Verilog code, synthesis and simulation is done using Xilinx Vivado tool. The following steps need to be done to generate and control the interrupts and priorities.

- Design of Priority Controller that assigns priorities to different slaves.
- Design of Top Module which includes all the functional modules that is master module, slave module and the priority controller or router.

- Design of Master module which provides communication access to the slave module by checking the priority order and sending response to the interrupt request signal by interrupt grant signal.
- Design of Slave module which requests the master to establish communication by sending interrupt request signal.

Fig. 2 shows the design of the SPI interface along with the internal functional blocks that will be essential for implementing the priority-based communication.

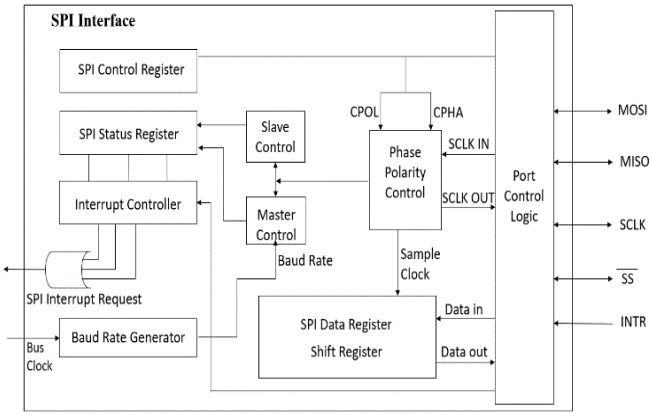

Fig. 2. SPI Interface with internal functional blocks

The master and each slave will have the interface through the signals MOSI, MISO, SCLK and SS. These signals interface with the port control logic that drives the transitions. The Phase Polarity Control block is mainly used to select the mode based on CPHA and CPOL value that gives out four modes of operations. It also has control over the clock synchronization. SPI Control Register is connected to the Port Control Logic to control the interfacing signals and to control the data transfer of the SPI. This includes enabling the SPI, configuring the SPI mode, setting the sampling of data, configuring clock polarity. The Slave control and Master control block are connected to the Status Register that has the status of operation and status of each slave. The Interrupt Generator is used to generate interrupts when any slave wants to communicate with the master. Baud Rate Generator consists of a series of divider stages with divisors ranging from 2 to 128 that helps in generation of clock frequency. The SPI Data Register is basically a shift register that is used to store and move the data bit by bit for data transfer between master and slave, during READ or WRITE operation. Once the operation is completed the content is erased and it is reset.

The implementation is done using Verilog Code in Xilinx Vivado synthesis tool for schematic, waveforms and power report and utilization report generation. Simulation can be done using Xilinx Vivado tool or Mentor's ModelSim tool.
The design can be implemented with major blocks that is spi_master, spi_slave, spi_router and top module along with testbench. The top module block includes all the remaining blocks and establishes the communication between master and slave accordingly. It has interrupt request and interrupt grant signals to provide communication based on the priority of the slave. These priorities are assigned in the router or priority control module that acts as an arbiter giving different priority to each slave.

The master block is the one that acts as a controller that can work in two states i.e., Idle or Wait-for-interrupt. The slave module has all the slaves defined in it, here four slaves are used. The slave module consists of transmit buffer to transmit the data to the master, receive buffer to receive the data from the master and shift register which holds the data during operation. Once the operation is completed, the shift register is reset and the content/data is erased.

The design is reconfigurable as changes can be made in the number of slaves, assigning priority to the slaves and the order of priority, the data to be read or written, the amount of time required for a full duplex communication. These parameters can change according to the user application. The sequence of operations can be checked in the generated waveforms.

IV. RESULT AND DISCUSSION

The schematic layout of the proposed design is shown in Fig. 3. It can be observed that there is one master and four slaves. Also, there is a priority control block that controls the priorities and interrupts. The interrupt pin on the slaves and the master can also be seen in the figure through which request and response takes place. In this way data transfer happens between master and slave based on priority of the slave and the interrupt that is generated.

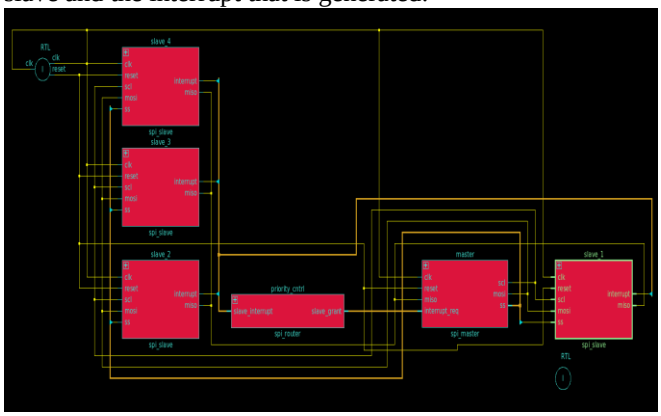

Fig. 3. Schematic layout of the Master-slave communication with priority control block

Fig. 4 shows the synthesized design of the master slave communication along with the priority control block. Various signals that will be required for the communication along with interrupts can be seen in the figure.

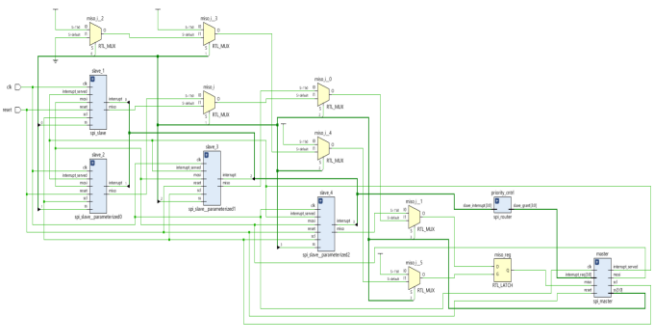

Fig. 4. Synthesized design of the Master-slave communication with priority control block

Fig. 5 shows the resultant waveform with the interrupt request and grant signals. This is generated using the Vivado Simulator tool. In the figure, it can be observed that the serial clock has started and reset signal is low. All the slaves have interrupt request signal high, but only the slave with the highest priority is given the grant. In this case the slave 3 has highest priority hence that signal is high. The order of priority for different slaves depends on the user requirement.

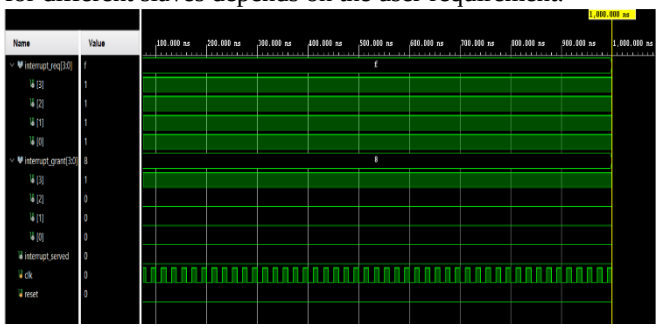

Fig. 5. Interrupt request and interrupt grant signals

The utilization report and the power report are generated using the Vivado tool after the simulation and synthesis. Table I shows utilization report for the IO resource and it is evident from the table that the % utilization is less than 10% i.e., 8.67 % of the total available resources. It can be observed from Table II that the total on-chip power is 0.058 W, which is very less when compared to other complex designs and the static power accounts for the total on-chip power since the dynamic power is 0 W.

Table-I: Utilization Report

Resource	IO
Utilization	13
Available	150
Utilization %	8.67

Table-II: Power Report

Total On-Chip Power	0.058 W
On-Chip Static Power	0.058 W
On-Chip Dynamic Power	0.000 W
Junction Temperature	25.3 ^{o}C
Thermal Margin	59.7 ^{o}C (11.3 W)
Power supplied to off-chip devices	0 W

V. CONCLUSION

In this paper, the design and implementation of interrupt enabled priority-based master slave communication using SPI protocol is discussed. The simulations show the priority-based data transfer between master and slave where four slaves are used. The slave priorities are assigned by the arbiter and the master communicates with a slave based on the priority when an interrupt occurs over the newly defined interrupt pin. The design is reconfigurable in terms of the number of slaves and the assignment of the priority. The design consumes very less power and utilizes fewer resources when compared to other complex designs involving interrupts.

VI. FUTURE SCOPE

The design described here can be implemented for multiple slaves, more than four. The shift register used in this design resets every time a communication is completed. When a data transfer is taking place and if the master gets an interrupt from a slave of higher priority, it abruptly ends the current data transfer and establishes a communication with the recently selected slave. In such a case the data in the shift register is replaced by the data of the latest transmission. This is a disadvantage in terms of loss of data and can be avoided by storing the data of the previous transmission in some buffer or other register so that once the current transmission completes with the higher priority slave the master can resume the data transfer with the previous slave from where it left off.

REFERENCES

1. Dvijen Trivedi, Aniruddha Khade, Kashish Jain, Ruchira Jadhav, Dept. of EXTC Engineering K.J. Somaiya College of Engineering Mumbai, “SPI to I2C Protocol Conversion using Verilog’’, IEEE 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA).
2. Rizka Reza Pahlevi, Aji Gautama Putrada S and Maman Abdurohman, “Fast UART and SPI Protocol for Scalable IoT Platform”, IEEE 2018 6th International Conference on Information and Communication Technology (ICoICT).
3. Lei Feng, Xingzhe Song, Xinyu Zhao, Wenbin Zheng and Yue Liu, “A Method for Analyzing Serial Bus Protocols with Standalone Clock Signals”, IEEE 2017 First International Conference on Electronics Instrumentation & Information Systems (EIIS).
4. V. D. Veda Patil, “Implementation of SPI Protocol in FPGA,” International Journal of Computational Engineering Research (IJCER), vol. 3, no. 2, pp. 142–147, 2013.
5. Anand N, George Joseph, Suwin Sam Oommen, and R Dhanabal School of Electronics Engineering, VIT University, Vellore, India, “Design and Implementation of a High-Speed Serial Peripheral Interface”, IEEE 2014 International Conference on Advances in Electrical Engineering (ICAEE).
6. L. Bacciarelli, G. Lucia, S. Saponara, L. Fanucci, M. Forliti, “Design, testing and prototyping of a Software Programmable 12C/SPI IP on AMBA bus”, IEEE 2006 Conference Paper.
7. A.K. Oudjida, M.L. Berrandjia, A. Liacha, R. Tiar, K. Tahraoui & Y.N. Alhoumays, “Design and Test of General-Purpose SPI Master/Slave IPs on OPB Bus”, IEEE 2010 7th International Multi-Conference on Systems, Signals and Devices.

AUTHORS PROFILE

Deepika, Student, pursuing Master of Technology in Department of Electronics and Communication at B.M.S College of Engineering, Bengaluru, India.

Dr. Jayanthi K Murthy is presently serving as an Associate Professor in Department of Electronics and Communication, B.M.S College of Engineering, Bengaluru, India with 25 years of experience. She has authored more than 20 research papers in International Conferences and repute journals. Her research interests span Wireless Communication, Computer Networking and its application in AI/ML.