Middle-East Journal of Scientific Research 23 (Sensing, Signal Processing and Security): 409-417, 2015
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2015.23.ssps.213

Single Wire Common Protocol for Multiple Masters

[1]M. Jayapriya, [2]T.N. Rajvignesh and [3]A. Nandini

[1]Department of ECE, Saveetha Engineering College, Chennai, India
[2]Salieabs Electronics Engineers LLP, Salem, India
[3]Department of ECE, Saveetha Engineering College, Chennai, India

Abstract: Electronic communication is defined as the exchange of data in the form of digital electrical signals. For proper and speedy working of Electronic systems, communication between the peripherals and the core is to be secure, fast and error free. A good protocol is characterized by simplicity of hardware (less number of wires to communicate), faster data rates, less error rate, more number of nodes, multiple master provision etc. Arbitration mechanism is the one that prevents multiple peripherals from accessing the bus simultaneously. 1-Wire protocol invented by Maxim Semiconductors is a simple communication system that involves the use of a single wire for data transfers between multiple nodes on the bus. However, the 1-wire bus does not allow existence of multiple masters on the bus as it is a single master multiple slave system. A 1-Wire bus with provisions for existence of Multiple Masters is designed and arrived as an arbitration protocol. Thus the Arbitration protocol is coded in C Language and simulated using Proteus VSM on Arduino microcontroller. In VLSI, the 1-Wire Single Master is coded in VHDL and simulated using Quartus II software. The area occupation is very less comparing to the existing protocol. For the future work multi master algorithm is going to be design in VHDL and implementation in FPGA.

Key words: 1-wire · Priority based arbitration mechanism

INTRODUCTION

Communication is the process of data transfer between the transmitter and the receiver with the help of digital electrical signals. Communication within an electronic system happens between the peripherals such as sensors and actuators and the processing core like micro controller or microprocessor. For proper and speedy working of electronic systems, communication between the peripherals and the core is to be secure, fast and error free. A set of rules that define how communication should happen between the peripherals and the core is called as communication protocol. The channel used for communication is called as communication bus. A good protocol is characterized by simplicity of hardware (less number of wires to communicate), faster data rates, less error rate, more number of nodes, multiple master provision etc. Arbitration mechanism is an important part of an algorithm that supports multiple masters on the communication bus. This mechanism is the one that prevents multiple peripherals from accessing the bus simultaneously. It also ensures fair allocation of the bus to all the peripherals. The existing protocol like UART has 3 modules [1], SPI uses 4 wires for communication the hardware complexity is increased [2], I^2C uses 2 serial lines [3], etc. But there are notable disadvantages in each of them either in being too much hardware complex, or slower data rates, lesser number of nodes, etc.

To establish effective communication between on board peripherals in an embedded system, the communication bus should be of simple architecture with lesser number of lines, so as to reduce cross talk and interference.1-wire protocol invented by Maxim Semiconductors is a simple communication system that involves the use of a single wire for data transfers between multiple nodes on the bus [4]. However, the 1-wire bus does not allow existence of multiple masters on the bus as it is a single master multiple slave system.

Objective: The main objective is to provide the multi master connection within the communication protocol through a single wire. Hence it requires less area in terms of VLSI design and also simplicity in Hardware because of using Single Wire.

Corresponding Author: M. Jayapriya, Department of ECE, Saveetha Engineering College, Chennai, India.

Specification for Single Wire Multi Master

Wire: The bus should have single wire for bidirectional data transfer.

Read and Write: All the nodes on the bus should be able to read and write the bus.

Priority Based Arbitration: An arbitration mechanism to prioritize the access of nodes to the bus should exist.

Nominal Overhead: Frame format is to be chosen in such a way that there should be nominal overhead when compared to the payload in each frame.

Functional Requirements:

FR1: The nodes on the bus should be able to read and write the bus simultaneously.
FR2: Higher priority nodes should be gives priority in access to the bus.
FR3: Lower priority nodes should have guaranteed delay or should not have infinite starvation.
FR4: The overhead in a frame should be nominal that time taken to parse one frame on the bus is on par with existing protocols.
FR5: On loosing arbitration, a node should become a slave.
FR6: Any node that wants to transmit should listen to the bus to know if it is idle or busy.
FR7: When a bus is identified by a node to be busy, it should not transmit anything.
FR8: The frame format should accommodate the priority of the node in order for arbitration to take place on the bus.
FR9: One mode should not preempt the other node because of its high priority.

Non Functional Requirements:

NR1: The nodes should have a dedicated read and write line.
NR2: The protocol must be scalable to accommodate different payload sizes.
NR3: Each frame should have distinct delimiters.
NR4: None of the nodes should force an electrical short circuit on the bus.

Single Wire Multi Master Bus Design: In order to achieve the requirements labelled FR1 and NR1, the 1-wire bus is designed in such a way that it has single line for bidirectional data transfers, on which all the nodes, connect using dedicated, read and write lines.

The design of the bus is depicted in Figure 1

Design of Frame Format: The design of frame format involves critical analysis of the frame formats of the continuing protocols so as to examine the amount of overhead imposed on the payload. Analysis carried out on the frame formats of the continuing protocols namely UART, SPI, I^2C, LIN, CAN and FlexRay, are tabulated in Table 1. The table explains the total number of bits in each frame, the maximum number of data bits (payload) and hence the calculation of the number of overhead bits.

In order to be feasible to assess the relation amid the size of payload and the size of overhead, a ratio amid the overhead and payload size is calculated. This ratio of overhead size and the payload size, termed as Overhead-Payload Ratio (OPR) is an indication of the relation amid the overhead and the payload size. The lesser the value of OPR, the less is the overhead compared to the payload size.

Analysis as presented in Table 1 indicates that FlexRay has the best overhead payload ratio among all other protocols. However, the OPR is not the only measure that would decide on the performance, data rate and throughput of the protocol, because each protocol has its own bit timing (time duration of each bit). It also depends on whether the protocol is a synchronous one or asynchronous one, because in asynchronous protocols, some bits and time are being spent for synchronization and frame delimiting.

In order to comply with the requirements labelled FR4, FR8 and NR3, the frame format of the designed protocol is set to consist of the following fields.

- Transmitting Master‘s Priority Number (MPN)
- Start of frame (SOF)
- Receiving slave ID (SID)
- Read/Write Bit (RW)
- 64 bit data

Deriving inspiration from I^2C, the current implementation is limited to a maximum of 1024 nodes to reside on the bus. Thus, the slave ID and master‘s priority number are to be 10 bits each. Frame delimiters are set to consume 8 bits. Thus Table 2 is formulated to explain the frame weight i.e. number of bits in a frame. Hence, the frame format of the designed protocol is arrived at and presented as Table 3.

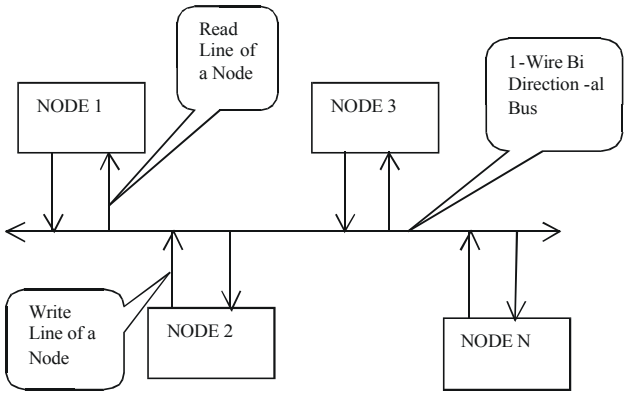

Fig. 1: Single Wire Multi Master Bus Architecture Design

Table 1: Analysis of Frame Size, Payload Size, Overhead size and OPR

S.No	Protocol Name	Frame Size(Bits)	Maximum Payload Size(Bits)	Overhead(Bits) =Frame Size –Maximum Payload	Overhead-Payload Ratio (Lesser The Better)
1	UART	11	8	3	0.38
2	SPI	18	12	6	0.50
3	I^2C	27	8	19	2.38
4	LIN	104	64	40	0.63
5	CAN	118	64	54	0.84.
6	FlexRay	2096	2016	80	0.04
7	1-Wire Single Master	88	8	80	10.00

Table 2: Frame Contents of the Design 1-Wire Multi Master Protocol

Payload/ Data	Master‘s Priority Number (MPN)	Slave/Node Address/ID (SID)	Read/Write‘	Frame Delimiter
64	10	10	1	8

Total Number of Bits = 93

When a master wants to send data to a slave, the entire frame duration is occupied and sent by it. In such a case, the fourth field is set to logic 0. If a master wants to command a slave to send data, then it sends the first three fields namely.

- SOF
- Transmitting Master‘s Priority No (MPN),
- Slave ID (SID)
- Read/Write (1 to Read)

Then the slave occupies the duration of the last field and sends the 64 bits data as requested by the master. In this case, the frame transmitted by the master is called a command frame.

Calculation of OPR for the frame format as shown in Table 3 is furnished below,

Number of Payload Bits = 64
Number of Overhead Bits = 29
OPR= Number of Overhead Bits / Number of Payload Bits = 29/64
OPR of 1-Wire Multi Master Protocol = 0.45

Comparing this value with that of asynchronous protocols in Table 1, it can be said that the OPR of the designed protocol is better than single master 1-wire protocol and CAN and as good as that of UART, but not as good as that of FlexRay. This is justified by the fact that FlexRay has a maximum capacity of transmitting 2016 bits with an overhead of 80 bits, whereas the designed protocol is limited to sending 64 bits as payload.

Priority Based Arbitration Mechanism: The arbitration mechanism should perform following functions as FR2, FR3, FR5, FR6, FR7 and FR9. The design of arbitration mechanism assumes that priority is a factor of the frequency in which a node will request to access the bus. The features of the designed arbitration mechanism are listed as follows.

- The number of masters existing on the 1-wire bus is fixed.
- Each master node on the bus is assigned a unique 10 bit priority number (MPN) with a fixed number of shares from which it can gain the access of bus.
- Each slave node has a unique 10 bit slave ID (SID).

Table 3: Frame Format for a Transmitting Master Data Frame Transmitted by a Master

SOF	PRIORITY NO	SLAVE ID	R/W	PAYLOAD/ DATA
8 Bits	10 Bits	10 Bits	1 Bits	64 Bits

Command Frame Transmitted by a Master

SOF	PRIORITY NO	SLAVE ID	R/W
8 Bits	10 Bits	10 Bits	1 Bits

Data Response from Slave for Command Frame Sent by Master

PAYLOAD/DATA
64 Bits

- If the bus is found idle, a node may start sending its frame of data or command
- A transmitting master understands that it has lost arbitration if it finds that data on the bus is not that as it is transmitting. Otherwise it understands that it has won arbitration
- After each successful access to the bus, the share value is decremented.
- A node of higher priority will not pre-empt an ongoing transmission on the bus.
- All slaves listen to the bus constantly and send back data if they find a command frame addressed to them.

Assignment of Shares: Assignment of bus access shares to the masters residing on the 1-wire bus is done by means of a static mathematical procedure. The terminologies used in the mathematical manipulations are listed below.

- N – Number of master nodes on the bus
- F – Time taken for one frame to be put on the bus
- K – A constant
- Cycle – One trip in a round robin scheme

Each node is to be assigned a fixed amount of shares, which limits infinite access to the bus by any master. This is done to prevent infinite starving of any lower priority node. The assignment of shares is done in such a way that there is at least one share guaranteed to all the nodes.

- A cycle is a period of time over which the nodes can gain access to the bus at least one time. This period is defined by the constant K.

Thus, Cycle Period = N * K (1)

- In one cycle slot, there is time for N*K frames to get transmitted. Thus there are N*K opportunities in a cycle that the nodes can use to send data.
- To ensure that no node starves for bus access, one frame slot is guaranteed to each node in a cycle
- Out of N*K frame slots, N frame slots are reserved as each node is guaranteed one frame slot.

Hence the residual frame slots is given by R = N*K –N (2)

Therefore residue, R = N*(K-1) (3)

- The arbitration scheme allots the residual frame slots to all the nodes depending on their priority.
- When residue is distributed on all the nodes, each node gets a share,

S = R/N (4)

- To maintain fairness, the node with higher priority is to be given higher share
- Thus each node looses one share to all of its higher priority ones
- Mathematically a node of priority i will lose (N-i) frame slots equally to the nodes of higher priority.
- This when generalized, narrows down to a mathematical formula.
- Nodes receive a share denoted by the formula in order of their descending priority S+(N-1), S+(N-3), S+(N-5), S+(N-7), S+(N-9).....

Arbitration Scheme: The arbitration procedure that will be followed by a master node on the bus is depicted in Figure 2. If a master node has a frame to be sent and if the share allocated to it is not exhausted yet, then it enters the arbitration scheme. Otherwise, it turns to be a slave. On turning to be a slave, a node will follow a process depicted in Figure 3.

Master Procedure: Once a master enters the arbitration scheme, it checks if the bus is busy. It waits till the bus becomes idle. Once the bus becomes idle, the master starts transmitting the frame bit by bit. Once each bit is

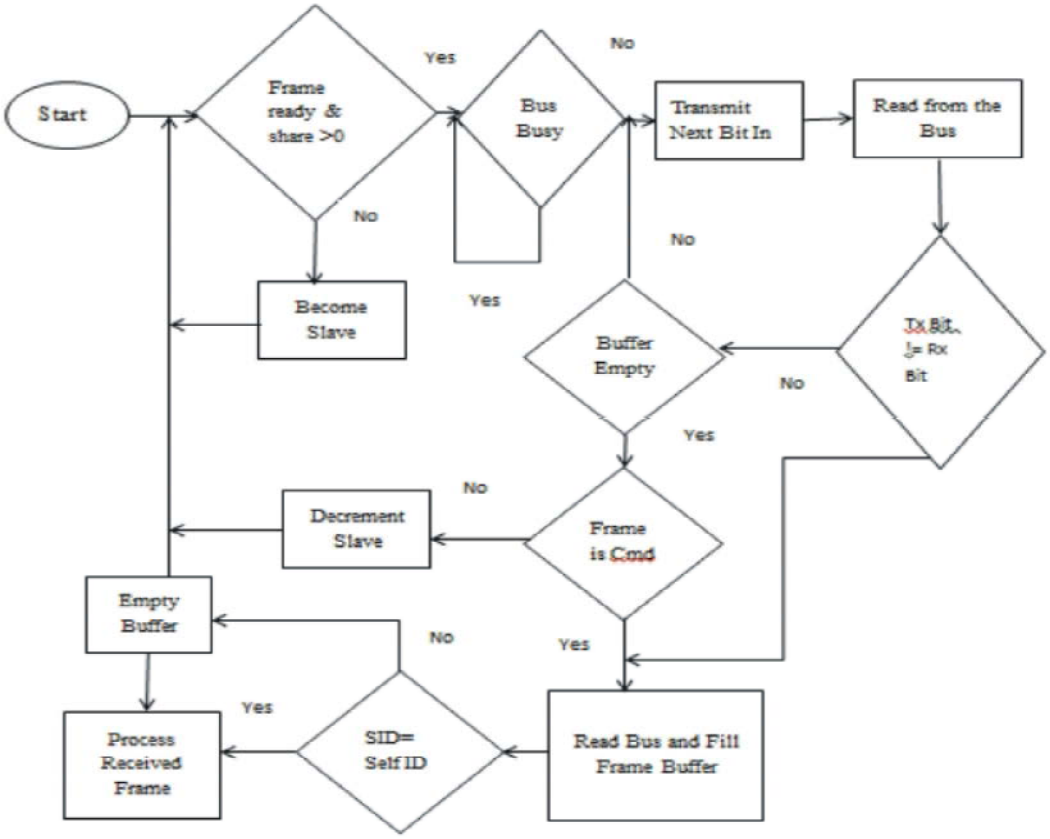

Fig. 2: Master Flow Chart Mechanisms

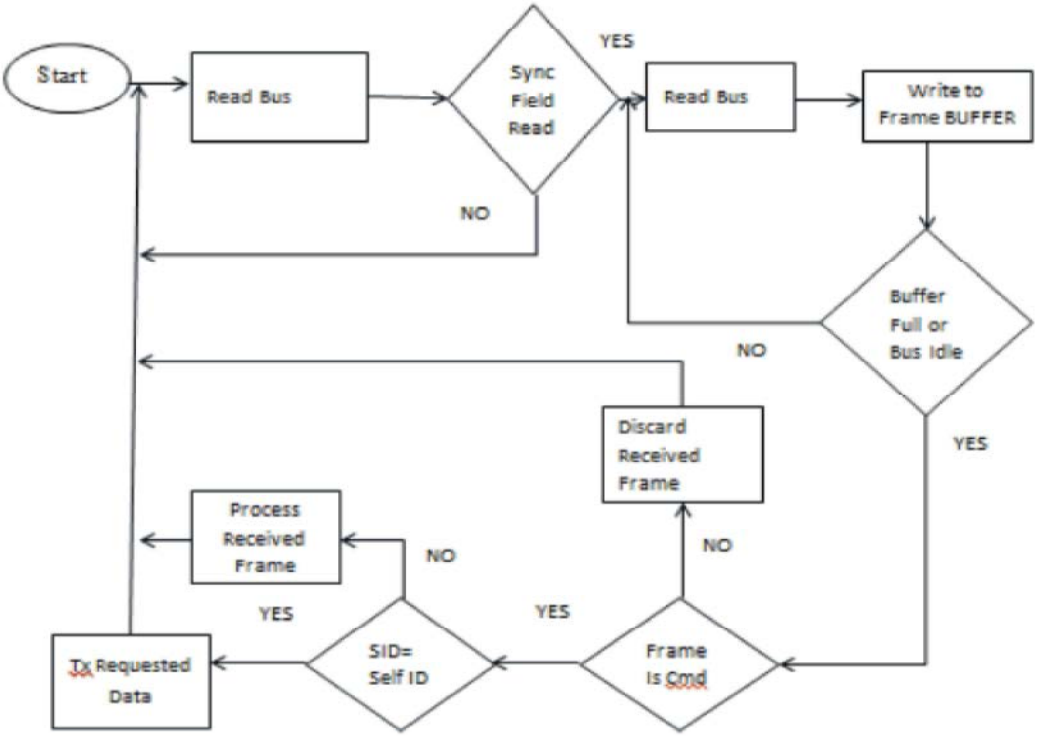

Fig. 3: Slave Flow Chart Mechanisms

transmitted, the node also reads the bus to find if what it has transmitted is on the bus. This is done till the last bit of the frame is transmitted. If in the due course it finds that what it has transmitted is not available on the bus, then it understands loss of arbitration. On loosing arbitration, the node reads from the bus and fills the remaining bits in the frame buffer. If the frame buffer contains the slave ID that corresponds to the ID of the receiving node, then it takes the frame for processing. Otherwise it discards the frame. On winning arbitration, the allotted share is getting decremented.

Slave Procedure: A slave node starts reading the bus as soon as it comes to act on the bus. It repeatedly looks for the sync field on the bus. If the sync field is found on the bus, the node continues to read the bus and fill its frame buffer. In the due course, if the slave finds the bus to be idle or if it finds that the frame buffer is filled, then it stops reading the bus and looks for the slave ID field and matches with its own ID. If both of them match, then the frame is analysed to see if it is a command frame or a data frame. If it is a data frame, the data is extracted from the frame buffer and taken for further processing at the application level. If it is a command frame, the slave node transmits the data held within it to the master.

RESULTS AND DISCUSSION

Thus the Single wire multi master communication protocol arbitration mechanism is coded using C language and simulated with the help of proteus VSM simulator. The setup was developed with the help of Arduino library for proteus VSM. The setup clearly shows the 1-Wire bus to which all the nodes cling with a read and Write line. Each node has a virtual terminal to shows the debug messages from arduino. And also existing protocol of single wire protocol area is analysed using Quartus II simulator.

Arduino Output in Proteus Simulator: The thick line is the Single Wire and four Arduinos are acting as the nodes. Each Arduino has two digital IO lines employed, one for reading and writing. The bus is pulled down with the help of a 120 Ohm resistor to avoid level transitions due to external noise. This is shown in Figure 4.

Share allotment of Single Wire Multi Master is displayed in the virtual terminal. Data transmission from master to slave and that acknoweldegement and the share allomentis displayed in the virtual terminal.this is shown in the Figure 5.

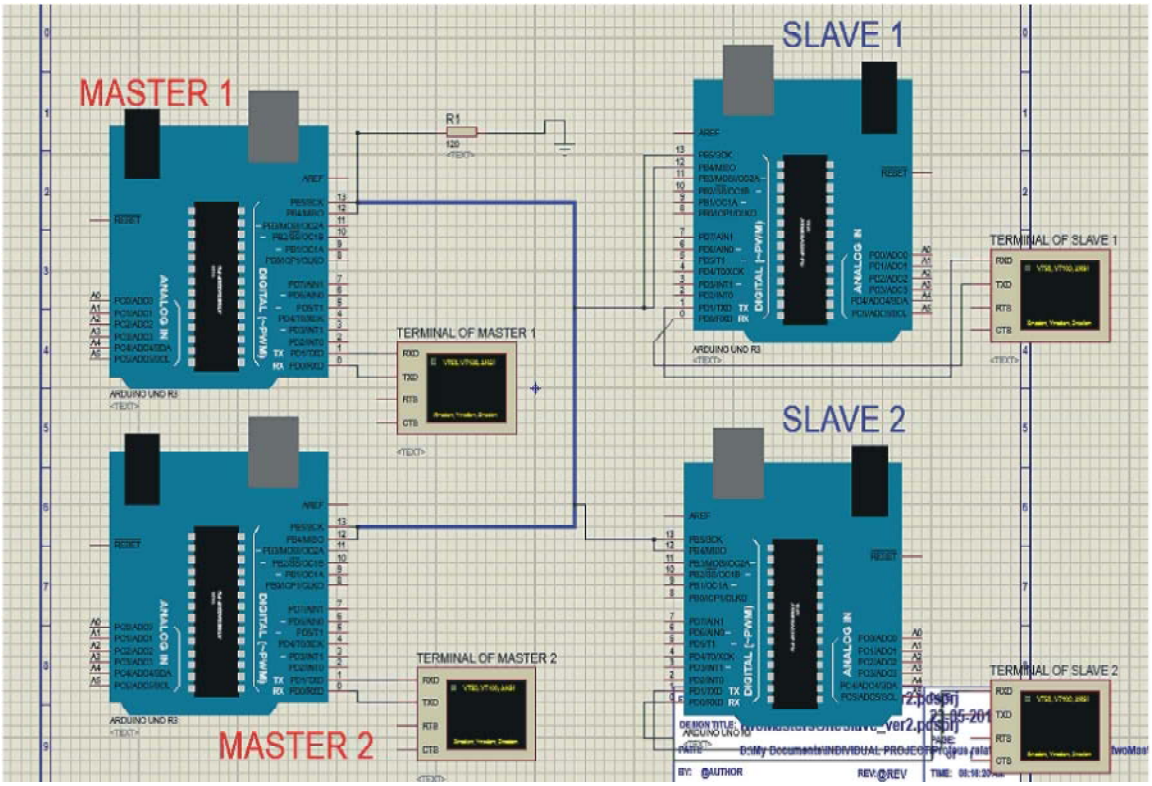

Fig. 4: Single Wire Communication Protocol For Multiple Master Connection

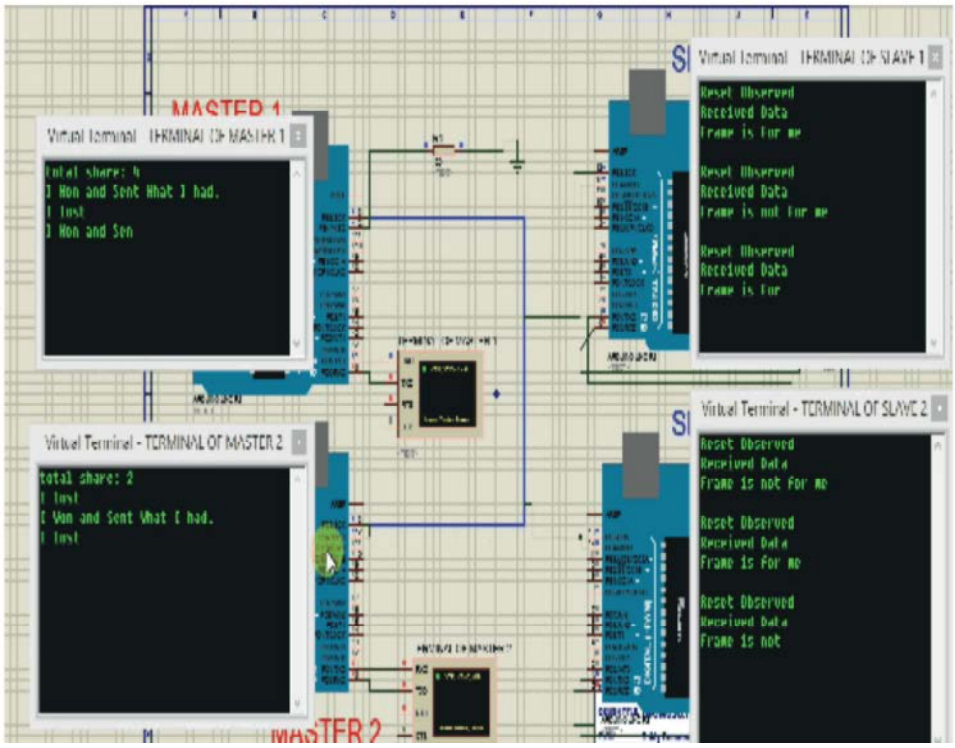

Fig. 5: Share Allotment and Acknowledgement displayed in Virtual Terminal

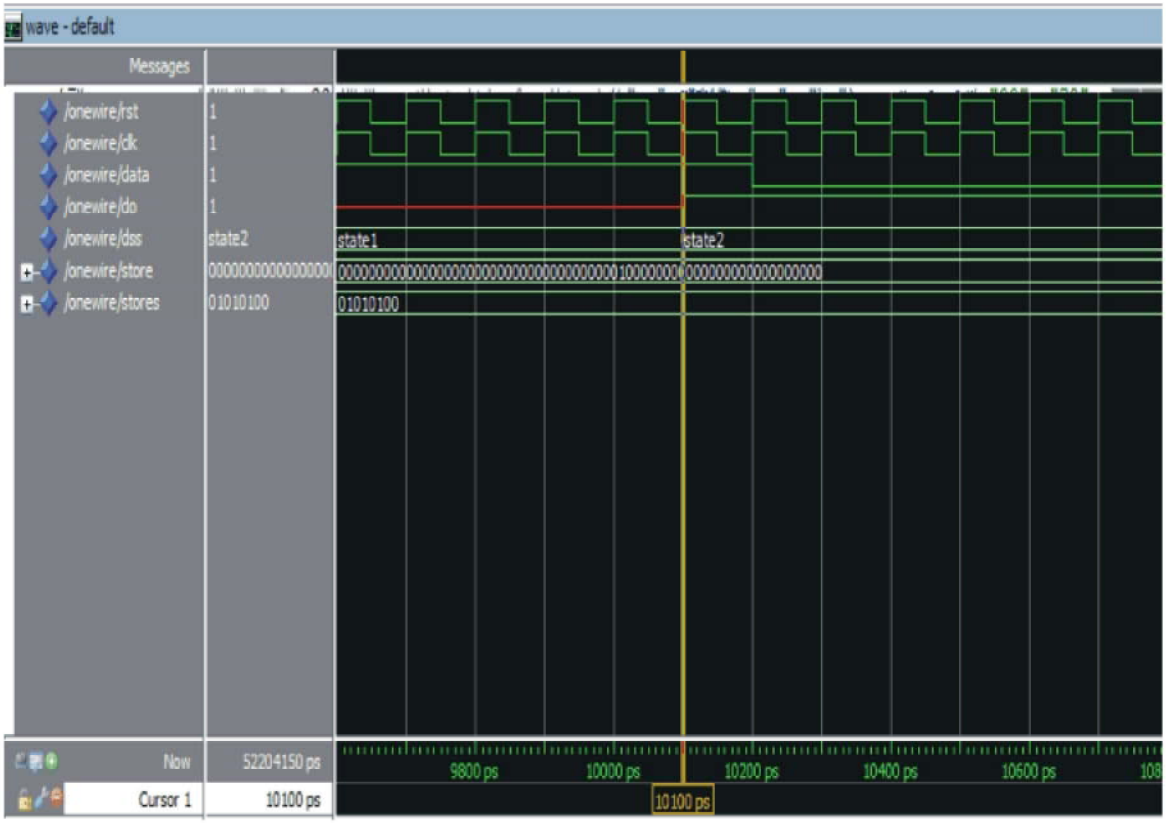

Fig. 6: Single Wire Single Master Output in FPGA

Fig. 7: Area Occupation of 1-Wire Single Master

Modelsim Output: The existing Data output of 1-wire Single Master. Data being transmitted from data line(data) to do (data out)line at an 10100ps when both the reset(rst) and clock(clk) is in HIGH which is shown in the Figure 6.

Area occupied for the 1-Wire Single master in Altera Cyclone II DE1 Kit. The Area occupation of 1-wire single master is very less when compare to other protocols like I^2C. Only 3% utilization of Total logic elements. This is shown in the Figure 7.

CONCLUSION

Communication protocol is designed for the purpose of providing appropriate speed, less error rate and simplicity in hardware, etc. The 1-Wire multi master protocol is designed and analysed by using proteus VSM simulator. Hence this protocol has simplicity in hardware, less error rate(less OPR).The Existing 1-wire single master is coded in VHDL and its area also very less compare to other protocols like I^2C.For the future work multi master algorithm is going to be embedded in Single Master FPGA Coding and implementation in FPGA.

REFERENCES

1. Laddha, N.R. and A.P. Thakare, 2013. International Journal Of Advanced Research In Computer Science and Software Engineering A Review on Serial Communication By UART, 3(1): 366-369
2. Shanker, Shreejith and Suhaib A. Fahmy, 2013. Extensible Flexray Communication Controller for FPGA-Based Automotive Systems IEEE Transaction on Vehicular Technology.
3. Phillips The 87LPC76x In Multi-Master I^2C Applications Application Note 2000.
4. Atmel AVR:318 Dallas 1-Wire Master Application Note 2004.
5. Bertran B. Consel C. Jouve Inria W. Guan H. Kadionik K SIP as A Universal Communication Bus:A Methodology and an Experimental Study IEEE Communications Society 2010.
6. Bajaj, P. and D. Padole, 2013. Arbitration Schemes For Multiprocessor Shared Bus Application Note 2013, 395-410.
7. Goyal, U. and N.R. Prakash, 2013. Interfacing Can Bus With PIC32 Microcontroller for Embedded Networking' International Journal Of Engineering Research And Applications, 3(4): 689-693.
8. Inseok Park and Myoungho Sunwoo Flex Ray, 2011. Network Parameter Optimization Method for Automotive Applications IEEE Transactions on Industrial Electronics, 58(4).
9. Kwak, J., J. Nam and D.H. Kim, 2002. A Modified Dynamic Weighted Round Robin Cell Scheduling Algorithm Etri Journal, 24(5): 360-372.
10. Lázaro, J. *et al*., 2010. I^2Csec: A Secure Serial Chip-To-Chip Communication Protocol Elsevier Ltd-Journal of Systems Architecture, 57: 206-213.

11. Micea, *et al*., 2010. Parsecs: Communication System for Smart Sensors and Hard Real-Time Applications IEEE Transactions on Instrumentation and Measurement, 59(11).
12. Mohan, S. and A. Joseph, 2013. A Dynamic Priority Based Arbitration Algorithm International Journal of Innovative Technology and Exploring Engineering, 3(1): 232-234.
13. Peter Corcoran Two Wires and 30 Years from IEEE Consumer Electronics Magazine July 2013.
14. Sharma M. and D. Kumar, 2013. Wishbone Bus Architecture - A Survey And Comparison International Journal of VLSI Design & Communication Systems, 3(2): 107-122.
15. Shingare, T. D. and R.T. Patil, 2013. SPI Implementation on FPGA International Journal of Innovative Technology and Exploring Engineering, 2(2): 7-9.
16. Yeon, K.B. and J.W. Chong, 2013. An efficient LIN MCU design for In-Vehicle Networks Journal of Semiconductor Technology and Sci., 13(5): 451-458.