© 2016 IJSRSET | Volume 2 | Issue 1 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

FPGA SPARTAN 3A Implementation of message based Arbitration in CAN Protocol

Sandeep Kumar Sharma, Abhijit Ray, Yogesh Khandagre

[1,3]ECE Department, Trinity Institute of Technology & Research, Bhopal, India
[2]VLSI Trainer, HCL-CDC, Bhopal, India

ABSTRACT

CAN is a protocol used in Automobiles Industry. The Electronic Control Units (ECUs) in automobiles need to communicate with each other. CAN bus is used for this purpose. In a multi ECU environment, arbiter plays the most important role in CAN bus. The arbiter prioritises and synchronises the transmission of different frames of CAN bus. These frames are continuous coming from different ECUs. CAN is a message based protocol. This means, unlike other arbitration mechanisms where request signals and grant signals are incorporated to provide the arbitration, here, in this type of protocol the message ID is the arbitration field. This field is scanned and the decision of arbitration is taken. With CAN protocol, the problems of point-to-point wiring connection made for ECU communication is removed. Also, the problem of delay with alternative technologies like Ethernet is resolved by using CAN protocol. The CAN arbiter is designed using EDA tools. The tools that are used in preliminary phase of the designing are simulation and synthesis tools. The design is targeted for FPGA technology. The device family used for FPGA technology is SPARTAN 3A. FPGA implementation offers better performance with respect to speed. Also, it provides efficient arbiter hardware, thereby reducing the size and volume.

Keywords : CAN, Arbitration, FPGA, SPARTAN 3A

I. INTRODUCTION

CAN Bus can be connected to multiple ECUs. The ECUs are different Electronic Control Units inside an automobile. The proposed CAN Arbiter scan input frames coming from different ECUs. Each dedicated port of the CAN arbiter is connected to each ECU in the system. In the current work, we have kept CAN frame size[1] of 60 bits. This includes 2 bytes, (i.e. 8 bits) of data frame. The data frame is the actual data which is to be communicated. The maximum size of data frame can be 8 bytes, which is limited to 2 bytes in this current work.

The CAN arbiter[2] is designed using EDA tools. The language that is being used to describe the functionality of the CAN arbiter is VHDL. VHDL is one of the few standard and efficient hardware description language, which can be used the EDA tools to simulate and synthesize the design work done in VHDL. The EDA tool used for simulation purpose is Modelsim from Mentor Graphics. The EDA tool used for synthesis purpose is Xilinx ISE. The design of CAN Arbiter is targeted for FPGA platform. FPGA is an multiple time programmable platform. The synthesis tool has to be specified with the target technology while performing the synthesis process. The FPGA target technology is SPARTAN 3A.

II. METHODS AND MATERIAL

CAN Arbiter

In this dissertation, message based Arbiter of CAN Protocol was designed using Xilinx FPGA SPARTAN 3A technology. CAN is a protocol used in Automobiles Industry. The Electronic Control Units [3] (ECUs) in automobiles need to communicate with each other. CAN bus is used for this purpose. CAN is a successful industry standard protocol which is robust, reliable and effective. It is a message-based protocol[4] , designed originally for multiplex electrical wiring within automobiles, but is also used in many other contexts. Here, in this current work, we have attempted to design an hardware efficient[5] CAN arbiter which works on message based protocol. CAN arbiter takes frames coming from different ECUs parallelly, and prioritises

the frames[6] and transmits frames accordingly, thereby providing an efficient communication.

CAN Bus can be connected to multiple ECUs . The ECUs are different Electronic Control Units inside an automobile. The proposed CAN Arbiter scan input frames coming from different ECUs. Each dedicated port[7] of the CAN arbiter is connected to each ECU in the system. In the current work, we have kept CAN frame size of 60 bits. This includes 2 bytes, (i.e. 8 bits) of data frame. The data frame is the actual data which is to be communicated. The maximum size [8] of data frame can be 8 bytes, which is limited to 2 bytes in this current work.

The CAN arbiter is designed using EDA tools. The language that is being used to describe the functionality of the CAN arbiter is VHDL. VHDL is one of the few standard and efficient hardware description language, which can be used the EDA tools to simulate and synthesize the design work done in VHDL. The EDA tool used for simulation purpose is Modelsim from Mentor Graphics. The EDA tool used for synthesis purpose is Xilinx ISE. The design of CAN Arbiter is targeted for FPGA platform. FPGA is an multiple time programmable platform. The synthesis tool has to be specified with the target technology while performing the synthesis process. The FPGA target technology is SPARTAN 3A.

A general Input-Output pin diagram of Message Based CAN Arbiter is shown in Figure 1.1. This arbiter is different from conventional arbiters, as it prioritises the incoming frames based on the message ID. Here, we have 8 ports of 60 bits wide, a clock signal and a reset signal at the input end. At the output end, we have only a single port of 60 bit wide. The frame which the arbiter decides to have maximum priority is placed on the output port.

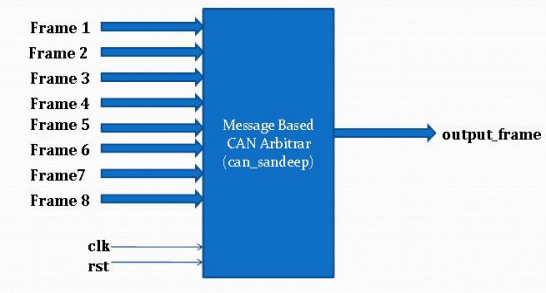

Figure 1 : General I/O Pin Diagram of Message Based CAN Arbiter

CAN Frames

The CAN frame is designed here to be of 60 bits. It starts with Start-of-frame (SOF) of 1 bit. This SOF denotes the start of frame transmission. Next, we have Identifier field. It is of 11 bits. A identifier which represents the message priority. So, this field has to be unique. This is followed by Remote transmission request (RTR). This is of 1 bit. Must be dominant (0) for data frames and recessive (1) for remote request frames. Identifier extension bit (IDE) is of 1 bit. It must be dominant (0) for base frame format with 11-bit identifiers. Reserved bit (r0) is of 1 bit. It is reserved bit. Must be dominant (0), but accepted as either dominant or recessive.

Data length code (DLC) is of 4 bit. Data field is of 0–64 bits (0-8 bytes). In this work , we have data field of 16 bits. Data to be transmitted (length in bytes dictated by DLC field) CRC stands for Cyclic redundancy check. CRC is of 15 bits. CRC delimiter is of 1 bit. This bit must be recessive (1). ACK stands for acknowledgement. It is of 1 bit. Transmitter sends recessive (1) and any receiver can assert a dominant[9] (0). ACK delimiter[10] is of 1 bit . It must be recessive (1). EOF stands for End-of –Frame. It is of 7 bits. It must be recessive(1).

Arbiter Ports and Logic

The CAN arbiter has multiple input ports and one output port. In this work, we have limited our ECUs to 8. So, there will be 8 ports in CAN arbiter having width of 60 bits, out of which 16 bit is the data bits since we have optionally kept data frame of two bytes.

CAN Arbiter reads input frames coming from different ECUs. Each dedicated port of the CAN arbiter is connected to each ECU in the system. We have kept CAN frame size of 60 bits. This includes 2 bytes, (i.e. 8 bits) of data frame. The data frame is the actual data which is to be communicated. The maximum size of data frame can be 8 bytes, which is limited to 2 bytes in this current work. The CAN arbiter reads the message ID of all frames simultaneously, and the frame corresponding to the most dominant bits (i.e. least binary value) is passed further giving it the highest priority.

The message ID of first two frame is compared. The frame with least message ID (more dominant bits) is of

higher priority. So, it is selected to be compared with the third frame. In this way, the comparison of message ID any two frames goes till all the frames are exhausted. The final frame selected till the end is the frame with highest priority.

III. RESULTS AND DISCUSSION

Simulation Result

Input Output Waveforms:-

The designed CAN Arbiter scan input frames coming from different ECUs. Each dedicated port of the CAN arbiter is connected to each ECU in the system. We have kept CAN frame size of 60 bits. This includes 2 bytes, (i.e. 8 bits) of data frame. The data frame is the actual data which is to be communicated. The maximum size of data frame can be 8 bytes, which is limited to 2 bytes in this current work. The CAN arbiter reads the message ID of all frames simultaneously, and the frame corresponding to the most dominant bits (i.e. least binary value) is passed further giving it the highest priority.

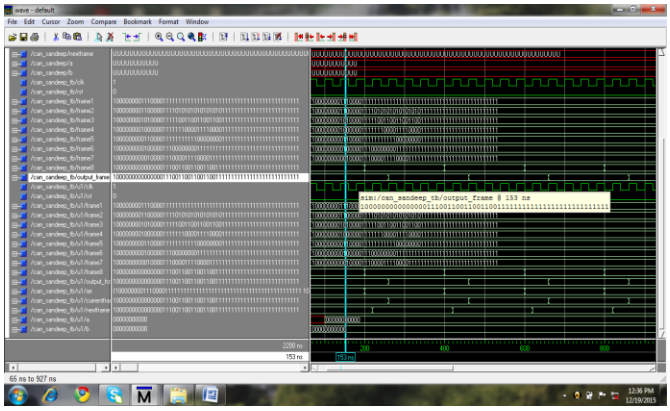

Figure 2 : I/O Waveform of CAN Arbiter at Simulation Time 153 ns

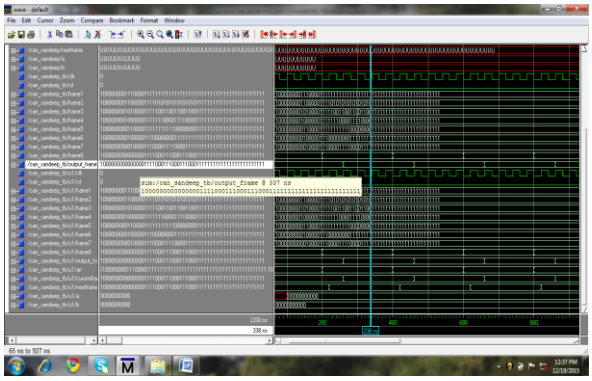

Figure 3 : I/O Waveform of CAN Arbiter at Simulation Time 337 ns

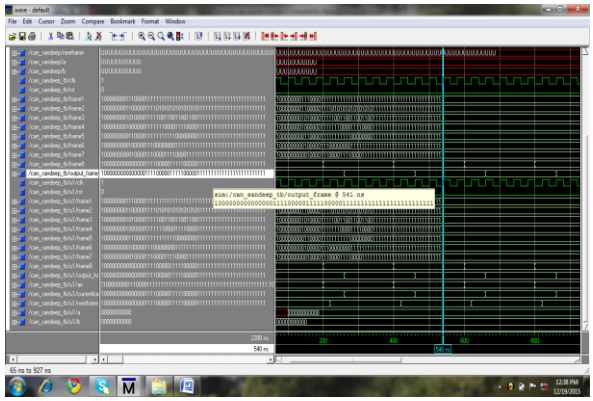

Figure 4 : I/O Waveform of CAN Arbiter at Simulation Time 541 ns

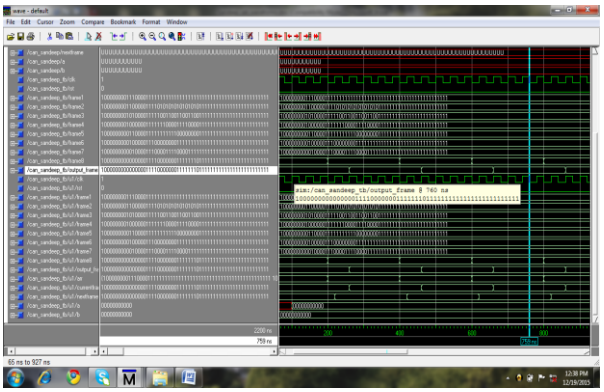

Figure 5 : I/O Figure 5Waveform of CAN arbiter at Simulation Time 760 ns

SYNTHESIS RESULTS

RTL Schematics

RTL Schematic is the output of synthesis tool corresponding to the code that is being synthesized. It is pictorial representation of the synthesized circuit. RTL View generated initially can be used to obtain further inner level view. This is obtained by clicking on each of the representations obtained. We have obtained five RTL views from outer to inner views.

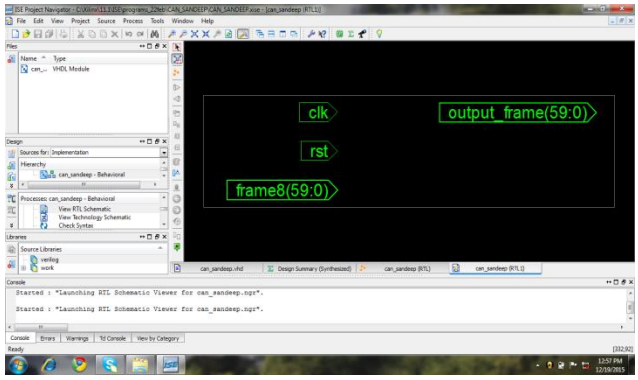

Figure 6 : RTL Schematics (Outermost level)

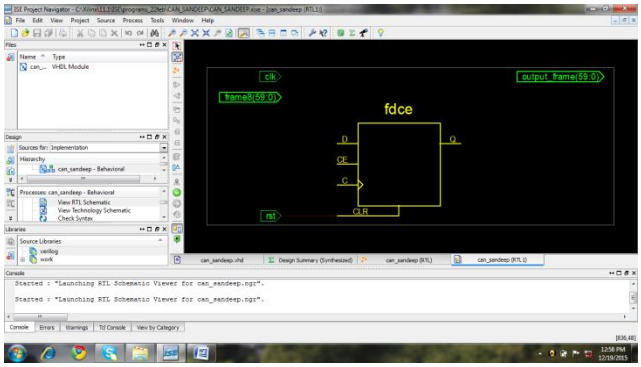

Figure 7 : RTL Schematics (Level-1)

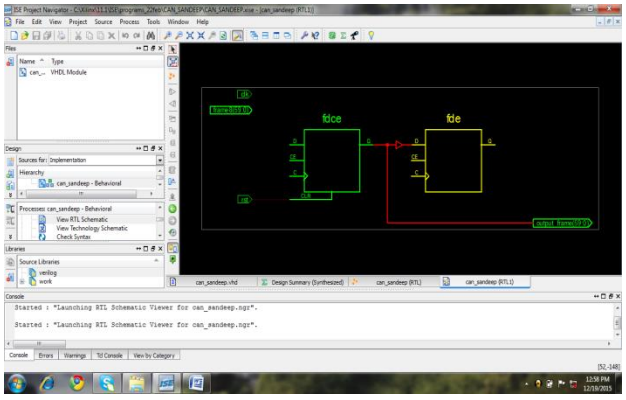

Figure 8 : RTL Schematics (Level-2)

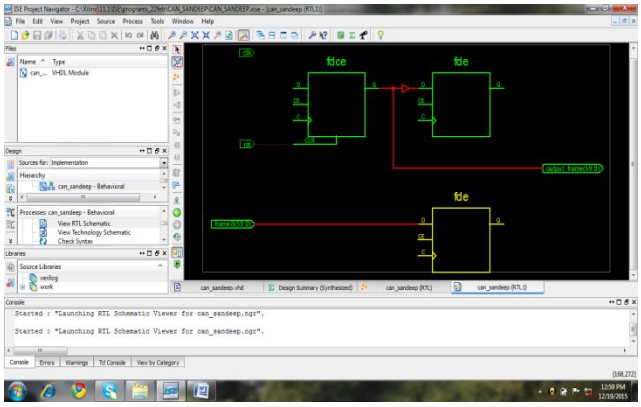

Figure 9 : RTL Schematics (Level-3)

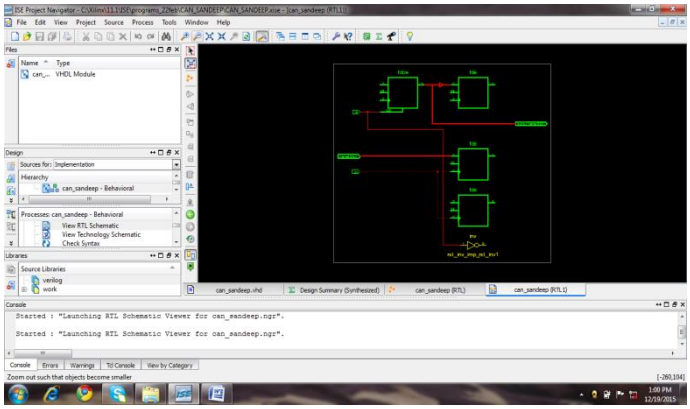

Figure 10 : RTL Schematics (Innermost level)

Design Summary

The design of CAN arbiter in VHDL using Xilinx ISE has the follows results. The number of Slices used in the FPGA device is 83, which is only 2% of the available Slices. The number of Slice Flipflop is 142, which is only 1% of the available Slice Flipflop. The number of 4 input LUTs (Look up table) is 12, which is negligible percentage of the total LUT , i.e. 7168. The number of bonded IOBs is 122, which is 62% of the available IOBs. The number of GCLKs is 1, which is 4% of available GCLKs.

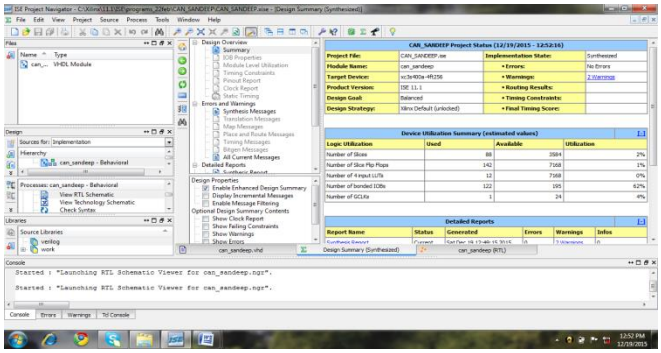

Figure 11 : Design Suumary

Table 1: Device Summary of CAN Arbiter

Number of slice	83 out of 3584
Number of slice Flipflop	142 out of 7168
Number of 4 input LUTs	12 out of 7168
Number of Bonded IOBs	122 out of 195
Number of GCLKs	1 out of 124
Minimum input arrival time	4.358 ns
Maximum output required time	5.558 ns

IV. CONCLUSION

We have designed and implemented message based Arbiter of CAN Protocol in VHDL. We have simulated it using Mentor Graphics Modelsim tool and synthesised it using Xilinx ISE. The design is targeted for FPGA SPARTAN 3A technology. The FPGA implementation offers better speed and effective and optimised hardware. FPGA device utilisation also shows minimum device utilisation.

Controller Area Network bus (CAN bus) is a standard that allows microcontrollers and other devices to communicate with each other within a vehicle (or other network) without a host processor. Its robust differential signals are carried on small-gauge twisted-pair cabling that is much lighter and less costly than the bulky wiring harnesses it usually replaces. CAN bus's low cost and high reliability helped make it the de facto control interface for the growing number of digitally-controlled powertrain, body electronics, safety and "infotainment" subsystems in conventional vehicles.

V. REFERENCES

[1] Xiaohong Ren, Chenghua Fu, Tianwen Wang and Shuxiang Jia, "CAN bus network design based on bluetooth technology," in Electrical and Control Engineering (ICECE), 2010 International Conference On, 2010, pp. 560-564.

[2] K. Pazul, "Controller Area Network (CAN) Basics," Microchip Technology Inc.Preliminary DS00713A-Page, vol. 1, 1999.

[3] Kumar, M. A.Verma, and A. Srividya, Response-Time "Modeling of Controller Area Network (CAN). Distributed Computing and Networking, Lecture Notes in Computer Science Volume 5408, p 163-174, 2009.

[4] Tindell, K., A. Burns, and A.J. Wellings, Calculating controller area network (CAN) message response times. Control Engineering Practice, 3(8): p. 1163-1169, 2005.

[5] Li, M., Design of Embedded Remote Temperature Monitoring System based on Advanced RISC Machine. Electrotechnics Electric, 06, p. 273,2009.

[6] Pazul, "Controller Area Network (CAN) Basics", Microchip technology Inc., AN713, May 1999.

[7] Prodanov, W., M. Valle, and R. Buzas, A controller area network bus transceiver behavioral model for network design

[8] Transactions on Industrial Electronics, 56(9): p. 3762-377, 2009. ISO (1993). Road Vehicles: Interchange of Digital Information: Controller Area Network (CAN) for High Speed Communication. ISO 11898:1993.

[9] B.Gmbh, "CAN specification" vol 1 Version 2.0, 1991.

[10] Wilfried Voss, A comprehensive guide to controller area network, Copperhill Media Corporation, 2005-2008.

AUTHOR'S PROFILE

Sandeep Kumar Sharma has completed B.E. (Bachelor of Engineering) in Electronics & Communication from Rajiv Gandhi Technological University, Bhopal, India .This paper is published as part of his Final year dissertation work for M.Tech. in VLSI Technology from Rajiv Gandhi Technological University, Bhopal India.

Abhijit Ray has done B.E. (Bachelor of Engineering) in Electronics from Mumbai University, India and Post Graduate Diploma in VLSI Design from C-Dac, Mumbai. He has also completed his M.Tech. in VLSI Technology from Bhagwant University, Ajmer, India. He is currently associated with HCL-CDC,Bhopal, India. He has a total work experience of about 13 years in Industry and Academics.

Yogesh Khandagre has completed B.E. (Bachelor of Engineering) in Electronics & Communication from Rajiv Gandhi Technological University, Bhopal, India He has completed his M.Tech. in Digital Communication Technology from NIIST Bhopal, India. He is currently working as an Asst. Professor in Trinity Institute of Technology & Research, Bhopal, India. He has a Teaching experience of 8 years. He has published several papers in International journals.