ISSN: 2395-7639

International Journal of Multidisciplinary Research in Science, Engineering, Technology & Management (IJMRSETM)

Visit: www.ijmrsetm.com

Volume 2, Issue 10, October 2015

Implementation of Round Robin Arbiter using Verilog

K. Tharun Teja[1], Ponnada Venkata Sai Siva Tarun[2]

Department of ECE, CVR College of Engineering, Hyderabad, Telangana, India[1]

Department of ECE, CVR College of Engineering, Hyderabad, Telangana, India[2]

ABSTRACT: Round Robin technique has been used as a fair (non-starvation) scheduling policy in many computer applications. This paper presents a novel design of a Round Robin Arbiter without any misses. RRA always allows an available resource to one of the justified requests which may be unevenly generated from various sources. The design is being modeled in Verilog, i.e. logically verified, and synthesized. The design of round robin arbiter compared with other designs makes it a promise for performance improvement in systems with potential non uniform requests. Round Robin Arbiter can be used in high speed network switches or routers, microcontrollers in interrupt handling, TDMA, etc.

KEYWORDS: Round Robin Arbiter, scheduling policy, request, priority encoder.

I. INTRODUCTION

Many systems in this universe exist in which a huge number of requesters must access a common resource. The common resource may be a shared memory, a networking switch fabric, a specialized state machine, or a complex computational element. An arbiter is the one which is required to determine how the resource is shared amongst the requesters. When putting an arbiter into a design, many factors are to be considered. The interface between the requesters and the arbiter must be of appropriate size and speed of the arbiter should be fast. Also, the coding style used will usually impact the synthesis results. This paper presents a novel design of a Round Robin Arbiter without any misses. It always grants an available resource to one of the legitimate requests which may be unevenly generated from various sources. The design will be modeled in Verilog to logically verify. For this we have designed a sequential circuit and verified them using Xilinx software after modeling it in Verilog.

II. ROUND ROBIN ARBITER

2.1**Simple Round Robin Arbiter**: As described earlier our idea is to allocate the resource to a user equally. For this we use Round robin mechanism where in a token is passed through users. The token stays with each user for a specific amount of time and he can use the resource during that time (say clock duration). The simple round robin arbiter allocates resources like buses in a round robin fashion where each user is granted the bus for some specific amount of time. Here each user gets equal share to resource, which wastes some cycles if the user doesn't need the resource**,** i.e. when a user doesn't need a resource and still have a token with him that clock cycle or time gets wasted. Our paper now provides an alternate circuit to prevent this wastage i.e. to allocate resource to other user in needwhen a person having tokenis not using the resource.

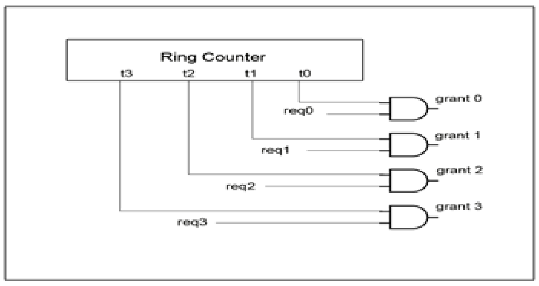

Fig 2.1: Simple Round Robin Arbiter

2.2**Modified Round Robin Arbiter:** As we can see the priority logic block will be enabled in correspondence with the token i.e. if the token is with first user first priority logic 0 is enabled and so on. Based on the person who is accessing the resource the corresponding value of the person comes as output of priority encoder and will be again decoded to get corresponding grant signal. As we can see from the main block diagram each priority logic block has different order of users based on the enable pin (or token) satisfying the basic round robin principle of giving priority to person having token. Also at any value of request any user can be granted resource (to prevent wastage of cycles). So each grant output is preceded by or gates to receive corresponding users grant signals from decoder output.

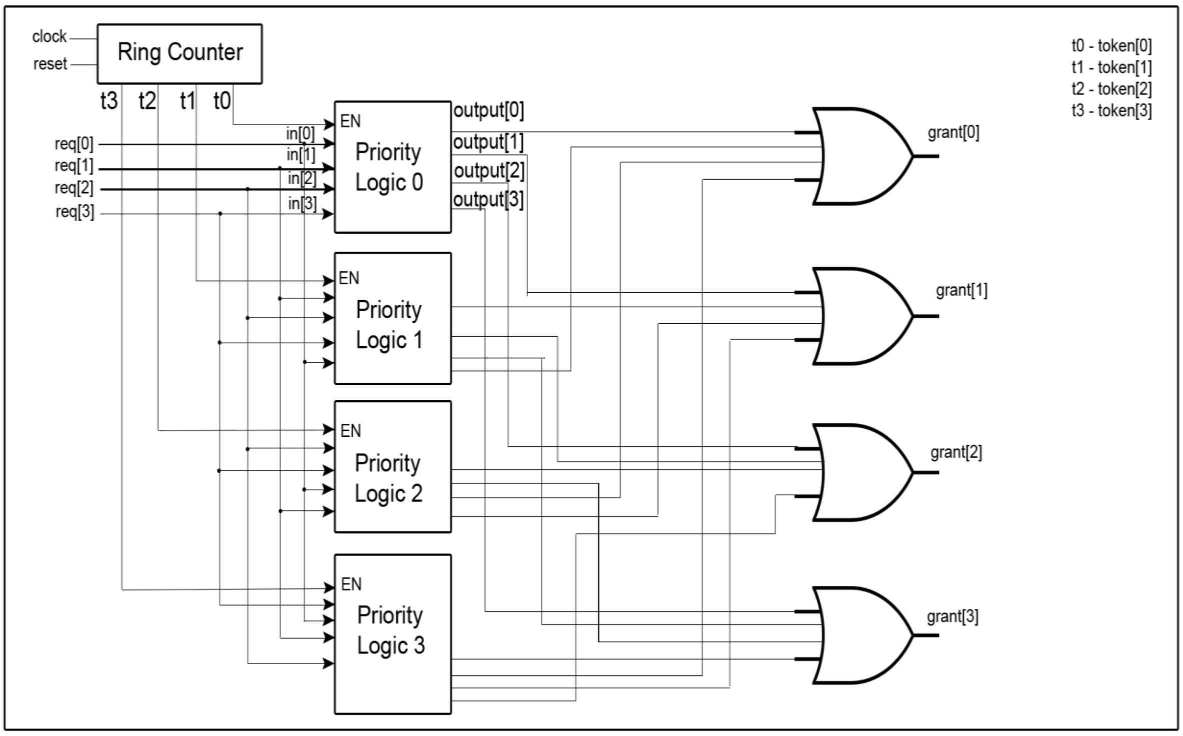

Fig 2.2: Modified Round Robin Arbiter

III. OPERATION OF THE ROUND ROBIN ARBITER

The operation of the block diagram can be explained by taking an examples
Ex 1: Where requests are 1001 i.e. user 1 and four want to access the resource. And let initially the token is in 0001 state. As per the circuit since t0 is 1 only the first priority logic is enabled in which the priority encoder gives output 00 and decoder as 1000. Therefore grant signal should come from first or gate as all its inputs are values which give access to 1st request and only one such input is high. In the next clock pulse the counter is in0010 state giving resource permission to user 2. But since the user is not willing to use the resource we need to allocate it to the next willing user i.e. user 4 which is done by the priority encoder giving output 10 and decoder giving output 0010 and the or gate 4 giving high value as output accessing permission to resource 4.
Ex 2: Where requests are 0110 i.e. user 1 and four want to access the resource. And let initially the token is in 0001 state. As per the circuit since t0 is 1 only thefirst priority logic is enabled in which the priority encoder gives output 01 and decoder as 0100. Therefore grant signal should come from 2nd or gate as all its inputs are values which give access to 2ndrequest and only one such input is high. In the next clock pulse the counter is in0010 state giving resource permission to user 2. As the user is now willing to access the resource 2nd priority logic is enabled and since 2nd user is having high priority grant signal comes from second or gate and resource is allocated to user2.
The same process is continued every time.

IV. RESULT

As seen in the waveforms of the Round Robin Arbiter in fig4.1, only one user is given access to the resource at one instance following the round robin scheme explained as before.

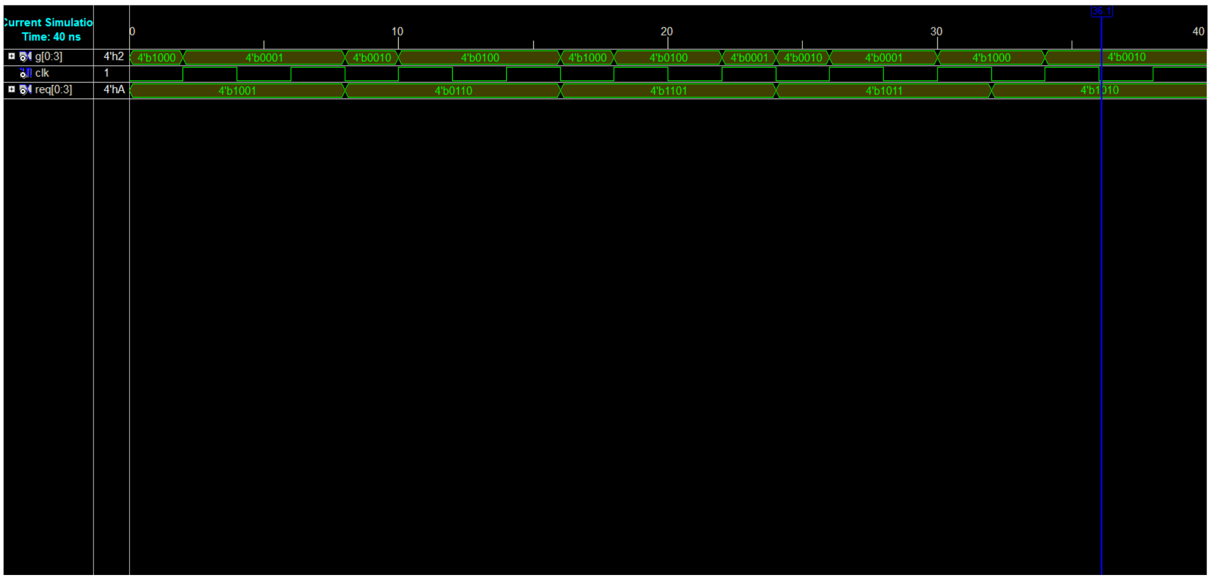

Fig 4.1 Waveforms of the Round Robin Arbiter

V. CONCLUSION

Hence the implementation of round robin arbiter using Verilog had been done taking into consideration the different factors involved in scheduling algorithms. This design fairly uses the clock cycles without any misses. The code is written using Xilinx and the simulation is verified. A complete stand-alone router can be built using this arbiter to users competing for a switch.

REFERENCES

1. Switching Theory and Logic Design by Anand Kumar.
2. A textbook of Electronics Engineering by J. B. Gupta.
3. http://www.electronics-tutorials.ws/