International Journal of Electronics and Communication Engineering.
ISSN 0974-2166 Volume 10, Number 2 (2017), pp. 63-70

http://www.irphouse.com

Optimal Implementation of IP based Router with Shortest Path Algorithm Using VLSI Technology

M. Sri Venkat Rami Reddy [1] and Dr. D. Nageswara Rao [2]

[1] Assistant Professor and [2] Professor
[1, 2] Department of Electronics and Communication Engineering
[1, 2] TKR College of Engineering and Technology

Abstract

In this paper our motto is to give a simple and optimal networking solution by the integration of the VLSI technology with the networking domain, a router is the vital device in networking domain so the focus remains on router itself to get a better control over the network without sacrifice latency in favor of other goals, to enhance the network we go for the bridging loops which effect the latency and security concerns. The other is of multiple protocols being used in the industry today. With this paper the attempt is to overcome the security and latency issues with protocol switching technique embedded in the router engine itself. This paper is based on the hardware coding which will give a great impact on the latency issue as the hardware itself will be designed according to the need. In this paper our attempt is to provide a multi-purpose router by means of Verilog code, with this we can maintain the same switching speed with more secured way of approach we have even the packet storage buffer on chip being generated by code in this design. so we call this as the IP based router. This paper has the main focus on optimal implementation of hardware IP router. The approach here is that router will process multiple incoming IP packets with different versions of protocols simultaneously and even it is going to hold true for the IPv4 as well as for IPv6. This paper is going to be an enhancement in the domain of networking with VLSI technology.

Keywords: IPv4, IP packet, OSPF, Router.

I. INTRODUCTION

In this paper we focused on optimal implementation of IP based router with shortest path algorithm. A high level view of generic router architecture is shown in Figure-1.

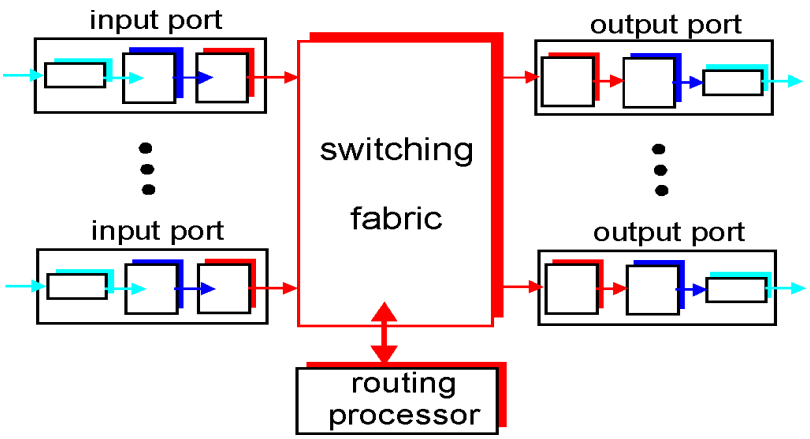

Figure 1: A high level view of generic router architecture.

Four major components of a router can be identified:

1. Input ports: The input port performs several functions. It performs the physical layer functionality of terminating an incoming physical link to a router. It performs the data link layer functionality (shown in dark blue) needed to interoperate with the data link layer functionality on the other side of the incoming link. It also performs a lookup and forwarding function (shown in red) so that a datagram forwarded into the switching fabric of the router emerges at the appropriate output port. Control packets (e.g., packets carrying routing protocol information such as RIP, OSPF) are forwarded from the input port to the routing processor. In practice, multiple ports are often gathered on a single line card within a router.

2. Switching fabric: The switching fabric connects the router's input ports to its output ports. This switching fabric is completely contained with the router - a network inside of a network router.

3. Output ports: An output port stores the datagrams that have been forwarded to it through the switching fabric, and then transmits the datagrams on the outgoing link. The output port thus performs the reverse data link and physical layer functionality as the input port.

4. Routing processor: The outing processor executes the routing protocols maintains the routing tables, and performs network management functions within the router.

II. DIJKSTRA'S SHORTEST PATH ALGORITHM

One of the key problems in systems modeling by means of graphs is the determination of shortest paths between given nodes on a graph. It is one of the most important basic problems in operations research. It is known that essentially any combinatorial optimization problem can be formulated as a shortest path problem. That is why the class of shortest paths problems can be used to model numerous practical problems that are not really shortest path problems. In many practical applications the computation of the shortest path has to be done in real time. There are a number of shortest path algorithms but there is no clear answer as to which algorithm is the best. Many considerable empirical studies on the performance of shortest path algorithms have been reported in the literature. There are two basic types of shortest path problems: those which find minimum paths from a given vertex and those which allow the determination of minimum paths between every vertex pair in the graph. The shortest path problems exists both in directed and undirected graphs. Dijkstra's algorithm solves the single-source shortest path problem while the Bellman-Ford algorithm solves the single-source problem if edge weights may be negative. Floyd-Warshall algorithm solves all pairs shortest paths, Johnson's algorithm solves all pairs shortest paths too, and may be faster than Floyd-Warshall on sparse graphs. Viterbi algorithm solves the shortest stochastic path problem with an additional probabilistic weight on each node. Nevertheless of results of similar studies the practice shows that the most widely studied and used is the original Dijkstra's algorithm. This algorithm is often used in routing and as a subroutine in other graph algorithms. For a given source vertex (node) in the graph, the Dijkstra's algorithm finds the path with lowest cost (i.e. the shortest path) between that vertex and every other vertex. It can also be used for finding costs of shortest paths from a single vertex to a single destination vertex by stopping the algorithm once the shortest path to the destination vertex has been determined. For example, let the starting node is X and the distance of node Y is the distance from the starting node X to Y. Dijkstra's algorithm assign some initial distance values and will try to improve them step by step as follows:

1. Assign for every node a tentative distance value. The value for initial node is set to zero and infinity for all other nodes.

2. Mark all nodes unvisited and set the initial node as current. Create a set of the unvisited nodes called the unvisited set consisting of all the nodes.

3. For the current node, consider all of its unvisited neighbors and calculate their tentative distances. Compare the newly calculated tentative distance to the current assigned value and assign the smaller one.

4. If all of the neighbors of the current node are considered the current node is mark as visited and is remove from the unvisited set. A visited node will never be checked again.

5. In case when the destination node has been marked or if the smallest tentative distance among the nodes in the unvisited set is infinity, then stop, i.e. the algorithm has finished.

6. Choice of unvisited node with the smallest tentative distance, and set it as the new "current node" then go back to step-3.

In resume, shortest path algorithms have many practical applications to automatically find directions between physical locations, such as driving directions, urban traffic planning, network routing protocols, subroutine in advanced algorithms, etc. This is why the proposed tool for shortest paths visualization is a subject of e-learning for many students and practitioners.

III. SYSTEM

Flow Diagram: The system flow diagram is as shown below which makes us to understand the flow of the signals through the system from each block by block and transaction carried between the blocks to accomplish the task of the robust router. The flow diagram described here is a brief one, which helps us to understand the flow of every block. Every block have the state machine cycle included in them to enhance the system logical transaction to the level of parallelism. First the packet is received from the ingress channel ring to the input interface block the packet is parsed to data packet and header packet, the data packet is stored in the parser queue and the header is sent to the filter block. The flier block then checks weather the packet is IPv4 or IPv6 and accordingly send the request to the filer table to router the packet to required destination. The filer table cross verifies the egress ring channel with it Destination IP address and send the egress ring ID to the filer block. The filer block send and enables the particular egress ring in egress blocks and gives the command to the particular egress ring in egress block. Then in egress block the stored data packet in the parser queue is added back with header and is sent out with the specified egress ring channel. In this way the every packet is processed and routed in robustrouter.

Simulation: The Net List is RTL level of the robust router system, which is synthesizable and can be extracted on the Xilinx tool. By which we can get preface look of the system and a transition from the frontend of the VLSI designing to backend of the VLSI designing. Which means the same can run on FPGA kit and test its robustness and errors of the system can be debugged before it is taken to SOC Level and to Fabrication Laboratories.

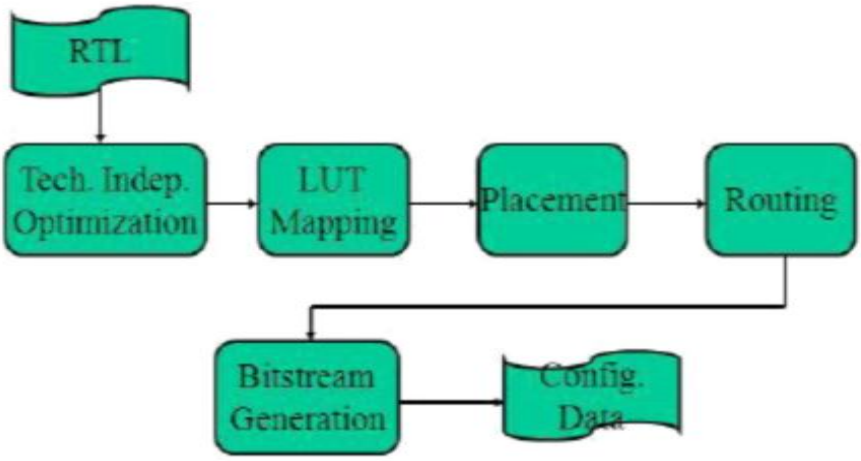

Figure 2: System flow diagram

The RTL level of design which we get from the Net List of the system will have gate delay, propagation delay and wire delays included in them. These are all calculated and made into an optimization level. Then the design is fixed into LUT'S and the mapped between the LUT'S further the placement of the LUT'S are prissily done keeping mind the power utilization and the delay calculated earlier. Then the routing is done between the CLB'S. Further the bit- stream is generated to test the system and verification done across the Net List output to get the exact design. Then the system design is masked and made to the GDSSI Level further to be sent on to the Fab-Labs for fabrication.

Design Under Test: The design under test [DUT] is made to test the system robustness under different cases. The DUT architecture includes the Test case which will define the test. The input driver block will generate the test input signals for the system testing. The input and output transacted will make the system get the input and output according to the system core requirement. The input and out monitor are placed to compare the system testing. At last the Response checker is to give the system testing pass or fail. The DUT is as shown below.

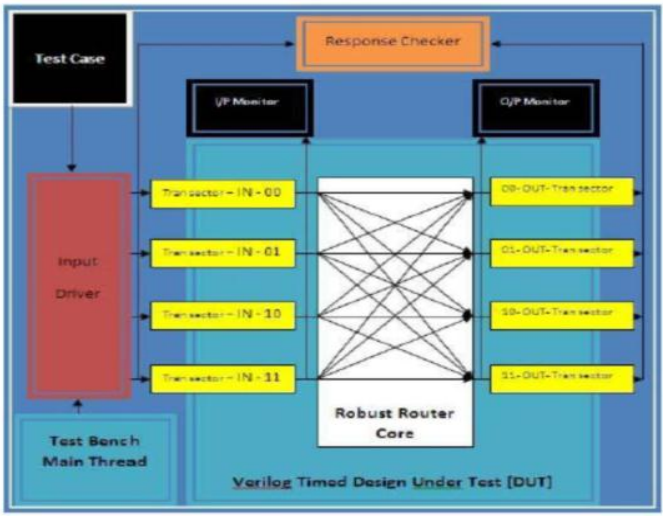

Figure 3: Design Under Test

In this paper we design a router which has a one input port from which the packet enters. It has five output ports where the packet is driven out. Packet contains 2 parts. They are header and data. The switch drives the packet to respective ports based on the destination address of the packets. Each output port has 3-bit unique port address. If the destination address of the packet matches the port address, then switch drives the packet to the output port. The RTL Schematic and Simulation wave forms are shown in figure-4 and 5 respectively.

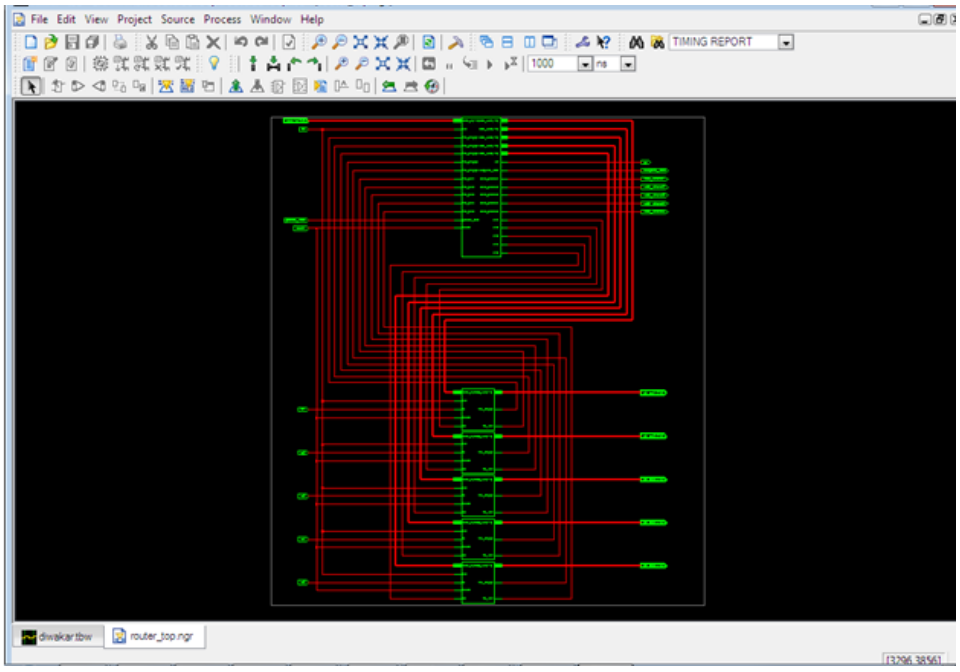

Figure 4: RTL Schematic

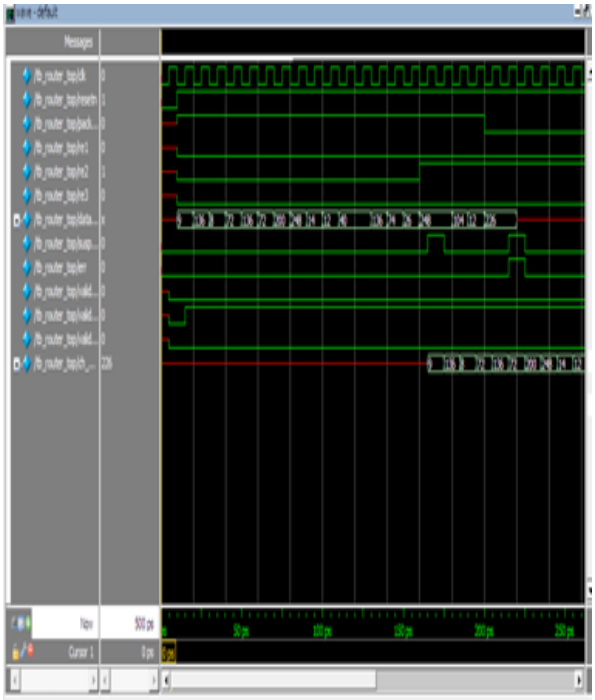

Figure 5: Simulation Wave form

IV. CONCLUSIONS

Here the code given in the output for IPv4 and IPv6 packets is put in the router at a time. The router will route both packets at the same time at the same speed without scarifying other performance parameters like latency, band with etc. The Robustness is simulated with Model-Simtool with different Test Cases and the same code's Net List is extracted with Xilinx Tool for the synthesizable code. The same can be taken to the System on Chip level with Cadences Encounter Tool. The same Verilog code design can be taken to the implementation of MPLS (Multi-Protocol Label Switching). The some code design can be taken to the System on chip level and can be implemented as the Ethernet Stand-a-lone System Router. The same code can be made variable with TCP and UDPProtocols.

REFERENCES

[1] RFC 2178 OSPF Version 2 was published in 1977. John Moy Cascade Communications Corp. Carlisle Road Westford, MA 01886

[2] Y. Katsube, K. Nagami, and H. Esaki, "Toshiba's Router Architecture Extensions for ATM: Overview," IETF RFC2098, April 1997.

[3] Cherkassky B. V., Goldberg A. V. &Radzik T. (1996). Shortest Paths Algorithms: Theory and Experimental Evaluation. *Mathematical*

Programming, *Ser.* A73(2), 129-174.

[4] Goldberg A. V. &Radzik T. (1993). A Heuristic Improvement of the Bellman-Ford Algorithm.*Applied Mathematics Letter*, **6**(3), 3-6.

[5] James, Balfour and William, J. Dally. (2006).Design tradeoffs for tiled cmpon-chip networks.In ICS ’06: Proceedings of the 20th annual international conference on Supercomputing, pages187–198.

[6] Ahuja R. K., Magnanti, T. L. &Orlin, J. B. (1993). Network Flows: Theory, Algorithms and Applications. Englewood Cliffs, NJ: Prentice Hall.

[7] Tobias, Bjerregaard and Shankar, Mahadevan 2006. A survey of research and practices of network-on-chip.ACM Comput. Surv., 38(1):1,2006.

ABOUT AUTHORS:

Mr. M. Sri Venkat Rami Reddy working as Assistant Professor of ECE in TKR College of Engineering & Technology, Hyderabad. He received the B.Tech. degree in Electronics & Communications Engineering from the J.N.T. University, Hyderabad, India, in 2004, and the M.Tech. degree in VLSI System Design from the J.N.T. University, Hyderabad, India, in 2012.

His current research interests include VLSI based Systems Design, Network-On-Chip (NoC), Low Power VLSI. He is a Life Member of the Indian Society for Technical Education (ISTE).

Dr. D. Nageswara Rao working as Professor of ECE in TKR College of Engineering & Technology, Hyderabad. He received the B.Tech. degree in Electronics & Communications Engineering from the S.R.T.M. University, Nanded, India, in 1999, and the M.Tech. degree in VLSI System Design from the J.N.T. University, Hyderabad, India, in 2004. He received Ph.D in VLSI domain from GITAM University in 2014.

His current research interests include VLSI, Image Processing, SOC, Low Power VLSI. He is a Life Member of the Indian Society for Technical Education (ISTE).