VLSI Based Robust Router Architecture

M. SOWMYA
Dept. of ECE

C. SHIREESHA
Dept. of ECE

G. SWETHA
Associate Professor,
Dept. of ECE

PRAKASH J. PATIL
Head of Dept.ECE
Vijay Rural Engineering College, Nizamabad, Jntu-H

Abstract:
Through this paper our attempt is to give a onetime networking solution by the means of merging the VLSI field with the networking field as now a days the router is the key player in networking domain so the focus remains on that itself to get a good control over the network, Networking router today are with minimum pins and 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. Through 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 multipurpose networking router by means of Verilog code, by 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 our design in the so we call this as the self-independent router called as the VLSI Based router. This paper has the main focus on the 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. With the approach of increasing switching speed of a routing per packet for both the current trend protocols. This paper thus is going to be a revolutionary enhancement in the domain of networking.

Keywords: *IPv4, VLSI, VLSI Based router*

1. Introduction

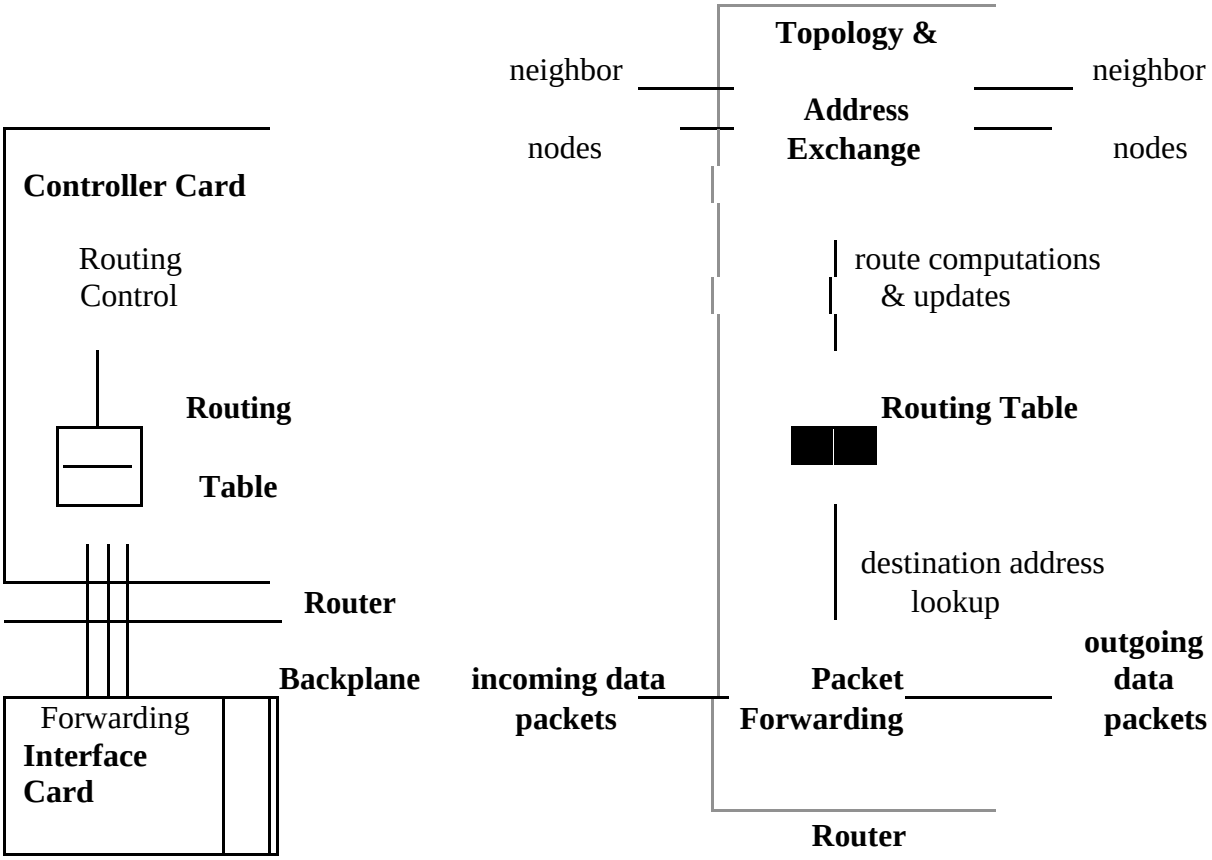
The popularity of the Internet has caused the traffic on the Internet to grow drastically every year for the last several years. It has also spurred the emergence of many Internet Service Providers (ISPs).Our approach here is to design a variable hardware router code by using Verilog and the same to be implemented for the SOC (System On Chip) level router. In this paper we are making a VLSI design for the implementation at the synthesizable level the same can be further enhanced to SOC level, but our main aim is limited to the NetList generation level which would give the result prediction and workable module vision. Our focus being in this is to make this router as much variable as we can which will give the robustness for the design to be called even as a Robust Router in which we can make the same router to not only go for N number of connections but also to detect all variety of packets and route the same. To do so we have to add

the code with specific case's for every type of packets we want to add to our router to route, with this paper of hardware code our approach is to get the basic packets routing with multiple protocols starting with the IPv4 and IPv6.

2. Basic IP Router Functionalities

Generally, routers consist of the following basic components: several network interfaces to the attached networks, processing module(s), buffering module(s), and an internal interconnection unit (or switch fabric). Typically, packets are received at an inbound network interface, processed by the processing module and, possibly, stored in the buffering module. Then, they are forwarded through the internal interconnection unit to the outbound interface that transmits them on the next hop on the journey to their final destination. The aggregate packet rate of all attached network interfaces needs to be processed, buffered and relayed. Therefore, the processing and memory modules may be replicated either fully or partially on the network interfaces to allow for concurrent operations.

A generic architecture of an IP router is given the basic architecture of a typical router: the



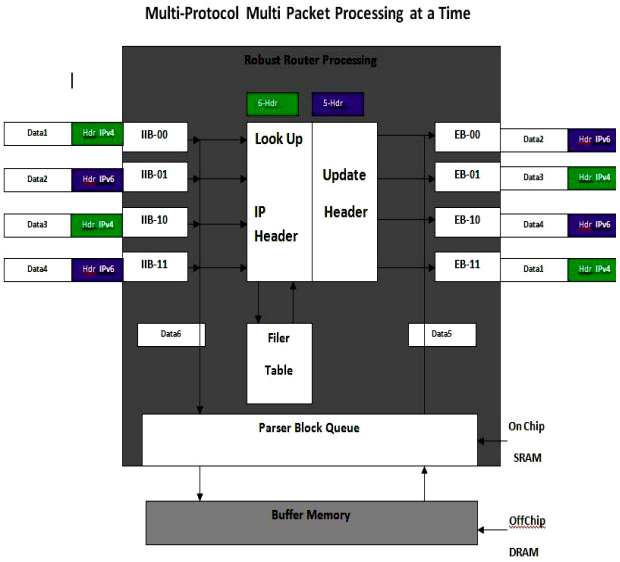
a). Basic architecture. b). Routing components.

controller card (which holds the CPU), the router backplane, and interface cards. The CPU in the router typically performs such functions as path computations, routing table maintenance, and reachability propagation. It runs which ever routing protocols is needed in the router. The interface cards consists of adapters that perform inbound and outbound packet forwarding (and may even cache routing table entries or have extensive packet processing capabilities). The router backplane is responsible for transferring packets between the cards. The basic functionalities in an IP router can be categorized as: route processing, packet forwarding, and

router special services. The two key functionalities are route processing (i.e., path computation, routing table maintenance, and reachability propagation) and packet forwarding. We discuss the three functionalities in more detail below.

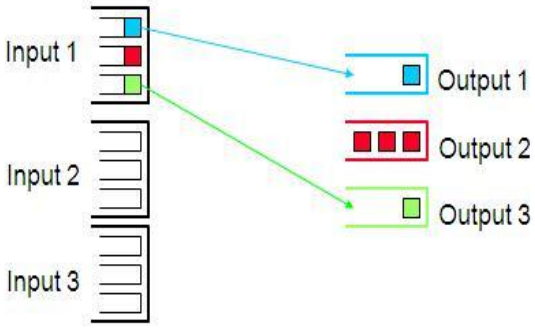
3. A New Robust Router Architecture

The architecture of robust router is totally based on the Verilog code which would enable our design in the implementation of parallel packet processing for N number of channels. This intern enables the multi packet processing at the same time. With the Verilog code being the base of design we have an option for the addition of protocol case and respective look-up table makes us go for the Multi-protocol processing at the same time. By which we are unable to provide the multi-packet multi-protocol routing at the same time with same speed.



Multi-protocol Multi packet processing at a tree

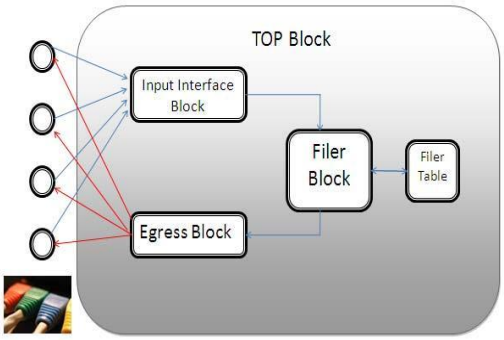
While designing the robust router a special concern is kept in the mind of the switching speed issue to give the maximum speed with parallelism being added. The egress output buffer queuing problem was also solved by providing a separate queue for every ingress channel in the egress channel with N vertical queue by which we can avoid the congestion to a remarkable level which is as shown below.



Congestion flow with vertical queue

The size issue is another special feature of our robust router which makes our robust router a unique system. As discussed earlier in the paper we are trying to make the Robust router on to the chip level design so we further advance it to the level of Ethernet based router which will make the router to be implemented on the standalone systems, which will be a revolutionary enhancement in size matter from room full of router to just the PCI slot operating Router and will make network work more faster. It looks something like this below.

Generally, the router can be interfaced with 'N' number of I/O devices. Block diagram given below.



Block diagram

A. Input Interface Block

This block is mainly responsible for receiving the incoming IP packets over multiple input channels. This block asserts the necessary response signals in order to communicate with the IP packet driver modules. After receiving the IP packets, this block forward the same to the Ingress Block for the further process. This block forwards the same to packet store block as well as parser block.

B. Packet Store Block

This is responsible for storing the error free received packets. This module receives the packet contents from Ingress block and dispatches the same based on the request from Egress block.

C. Parser Block

This block is mainly responsible for parsing the complete packet into multiple set of data according to its field. The parsed contents will be inputted to the filer block.The above three blocks are merged all together as IIB in code to single file.

D. Filer Block

This block is responsible for selecting the egress ring. The block receives the parsed data from the parser block. The parsed data will be forwarded to the filer table. In response to this, the filer table provides the output ring number. Then, the received output from the filer table will be forwarded to the egress block.

E. Filer Table

It is a user configurable table. This table contains a set of data in its each slot, against which the data sent by the filer block will be compared. If the filer block inputted data matches with the data of any slot of filer table, then that slot's data will be used as egress ring through which received packet will be forwarded.

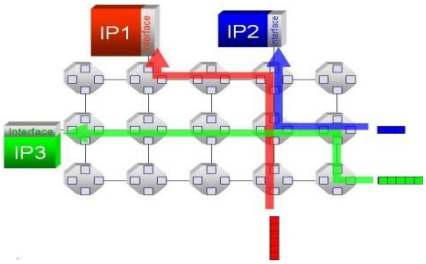
F. Egress Block

This block receives the data from filer block as egress ring number through which the received

packet shall be forwarded. Upon receiving the egress ring number, this block initiates the communication with packet store block to fetch the packet to be forwarded. Then, the fetched IP packet will be forwarded to the output interface block with the output channel details, over which the packet has to be transmitted.

G. Output Interface Block

Upon receiving the packet with output port details from the egress block, this block forwards the IP packet over mentioned output channel. This block is also responsible for asserting all the necessary handshaking signals for the receiving device while transmitting the packets. The Egress Block and Output Interface Block are merged together in code as single file.



Forwarding IP packet

We summarize the advantages and applications below.

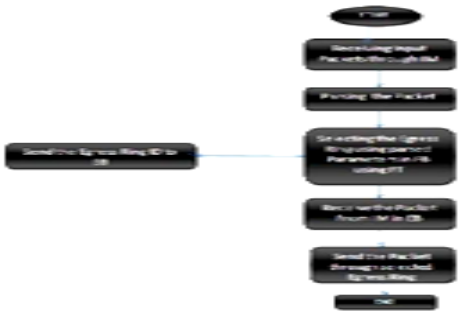
A. Advantages

- General purpose router
- Router hardware code is variable
- High switching speed with any no. of I\o pin connection
- More secured
- Robust router can handle all type of packets (Implemented on IPv4 and IPv6)

4. 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 filer 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 Dest-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 robust router.

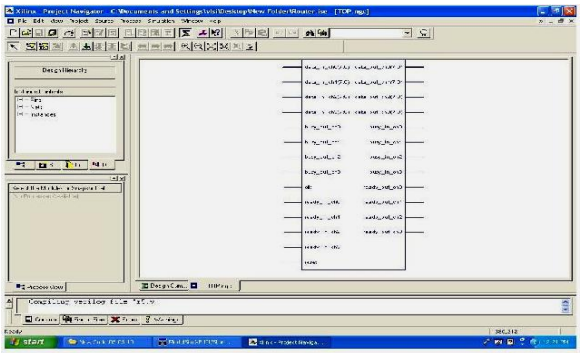


System flow diagram

5. Simulation and Discussion

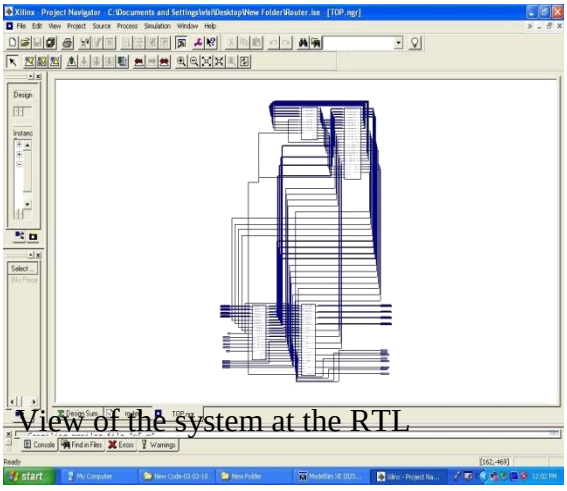
5.1 Net List of the Robust Router

The Net List is RTL level of the robust router system, which is syntasizable 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 Fab-Labs. The snap below is the Pin configuration of the proposed Robust Router.



Net List

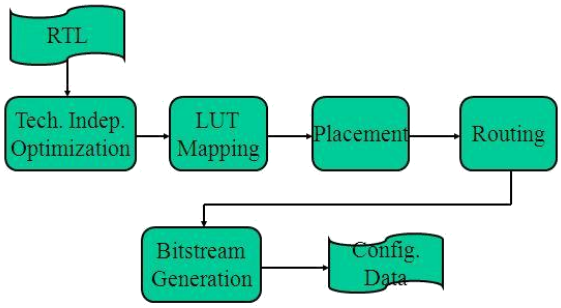
The NetList level can further go after I/O Padding get the exact pin configuration which can be derived accordingly but will be similar one. The Snap below is the Top level system view of the system at the RTL.



View of the system at the RTL

B.SOC Designing

The further movements of the VLSI Designing will require the sharp knowledge of the VLSI backend designing and can be fabricated at the 45 nano technology using the Cadence Encounter Tool which will enable us to take the system to SOC level the steps are as shown below.

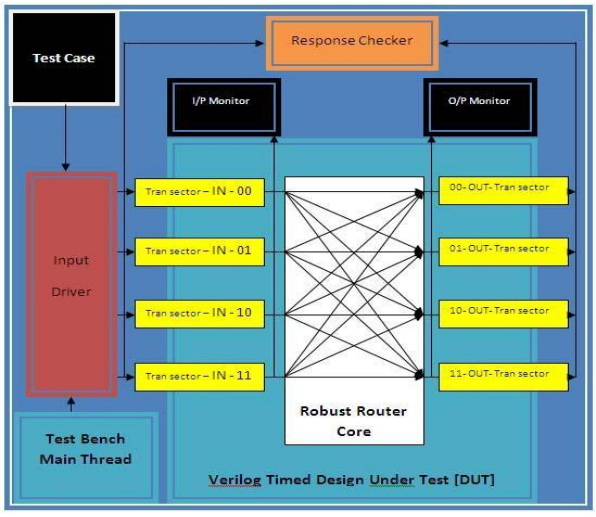


System to SOC level the steps

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.

5.2 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.



DUT architecture

For the testing of the robustness here we are mixing theIPv4 and IPv6 packets and we testing the system in different cases.

6. Conclusions

In this paper the code given in the output for IPv4 and IPv6 packets is put in the router at the same time. The Robust Router will route both packets at the same time at the same speed. The Robustness is simulated with Model-Sim Tool with different Test Cases and the same code's NetList is extracted with Xilinx Tool for the synthesizable code. The same can be taken to the SOC (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 SOC (System on chip) level and can be implemented as the Ethernet Standalone System Router.
- The same code can be made variable with TCP and UDP Protocols.

References

1. Abromovici, M. Breuer Digital System Testing and Testable Design.
2. Charles, H. Roth Jr Digital System Design Using VHDL
3. James, Balfour and William, J. Dally. (2006). Design tradeoffs for tiled cmp on-chip networks. In ICS '06: Proceedings of the 20th annual international conference on Supercomputing, pages 187–198.
4. Jenkins, Jesse H. Designing with FPGAs and CPLDs
5. Moy, J. (1998). OSPF: Anatomy of an Internet Routing Protocol.
6. Tobias, Bjerregaard and Shankar, Mahadevan (2006). A survey of research and practices of network-on-chip. ACM Comput. Surv., 38(1):1, 2006.