Putrevu Voshanavi, 2024, 12:5
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology

An Open Access Journal

Review of FPGA based Design of Low-Power Router

Putrevu Voshanavi, Yenni Ajay, Ponnana Triveni, Padala Sri Sai Nandu, Sanapathi V V S S Siddartha, Mohd Rizwan Uddin Shaikh

Electronics and Communication
GMR Institute of Technology

Abstract- The paper discusses an FPGA-based approach to designing routers for efficient data packet forwarding across networks. It introduces the router's architecture, which uses Verilog HDL and Universal Verification Methodology (UVM) for design verification. The proposed design focuses on reducing power consumption while maintaining high performance through the use of synchronous FIFO buffers, Finite State Machines (FSM), and a register-based approach for packet handling. The router operates at the network layer, ensuring data integrity through error checking mechanisms such as parity verification. The survey also outlines a verification strategy involving UVM-based test benches and RTL linting to identify functional bugs early, ensuring a robust, low-power router design suitable for Network-on-Chip (NoC) platforms.

Keywords- Top Module, Xilinx FPGA Vivado, Verilog HDL, Routing, Router, and Universal Verification Methodology (UVM).

I. INTRODUCTION

Network equipment that moves data packets between computer networks is called a router. Unlike network switches, routers link two or more data connections from several networks together. A data packet's final destination is determined by the router using the address information when it arrives on one of the lines. An overlay internetwork is created when it routes the packet to the following network along its path using data from its routing table or policy.

Routers are in charge of "traffic directing" on the Internet. Typically, data packets are sent from router to router across the network until they arrive at their destination node. At least two networks are connected to by routers, usually two LANs or WANs, or a LAN and the network of its ISP. At gateways—the intersections of two or more networks—they can be located. Routers are devices that use OSI layer 3 routing. They use the address information in the packet header to determine which output channel to send incoming packets to. The best route to follow for packet forwarding is decided by routers using forwarding tables and headers. Additionally, they use protocols to communicate with each other and figure out the best route between any two hosts.

1. Encapsulation

The process of encapsulation begins at the application layer. Here, data generated by user applications is organized into packets or segments. These packets include the application data itself, along with any relevant metadata, such as headers containing information. The OSI model's layers are as follows: L1 is the Physical Layer; L2 is the MAC/Ethernet/Data Link Layer; L3 is the Network Layer; L4 is the Transport Layer; and L5 is the Application Layer. TCP is the L4 header, IP is the L3 header, and MAC is the L2 header. As we move down the protocol stack, the transport layer adds its own headers to the application layer's packets. This includes details like sequence numbers for data ordering, acknowledgment mechanisms, error-checking codes, and flow control information. The

result is a segmented unit containing both the application data and the transport layer headers. The network layer takes these segments and encapsulates them into packets, adding network layer headers. These headers include the source and destination IP addresses, routing information, and possibly Quality of Service (QoS) parameters. This packet forms the basic unit that will traverse the network. As the encapsulated packet goes down further, the data link layer adds its own headers and sometimes trailers. These headers include the MAC addresses of the source and destination devices, allowing for proper addressing within the local network segment. Additionally, error- checking information may be added to ensure data integrity. The entire package is now referred to as a frame. At the lowest layer of the network stack, the frame undergoes a process of serialization. The frame's binary data is converted into a format suitable for transmission over the physical medium, such as electrical signals, light pulses, or radio waves. The physical layer handles the modulation and encoding of this data, ensuring it can traverse the network's physical infrastructure.

2. De-encapsulation

De-encapsulation begins at the physical layer upon receiving the transmitted signal. The physical layer processes the incoming signal, which could be electrical, optical, or radio waves, and converts it into binary data. The data is subsequently routed to the data link layer. The data link layer receives binary data from the physical layer. It examines the data to determine where the frame begins and ends, and then extracts its contents. The data link layer checks for errors to maintain data integrity.

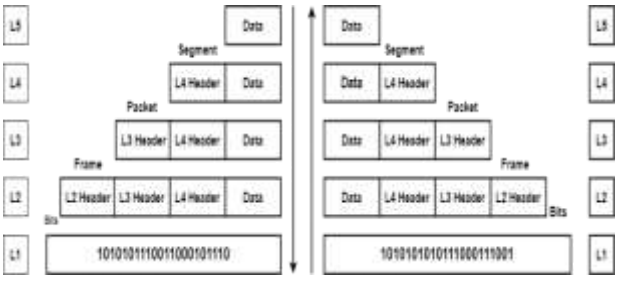

Fig 1: Process of Encapsulation and De-encapsulation in Networking

It removes the trailer and perhaps the data link layer header. The resultant packet is transmitted to the network layer without the data link layer headers. The packet is sent from the data connection layer to the network layer. To ascertain the packet's path and destination IP address, it looks at the packet's header. The transport layer segment or packet is revealed when the network layer removes its own header. After that, the transport layer receives the data and headers. The network layer sends the packet or segment to the transport layer. It verifies that flow management, error correction, and sequencing are done correctly. The original application layer message or data segment is revealed when the transport layer removes its own header. After that, the application layer receives the application data. Lastly, the original data is sent from the transport layer to the application layer. After processing the data, it shows it to the program or user. At this layer, any application-specific processing or interpretation is completed, and the recipient can now use the data.

3. Packet Format

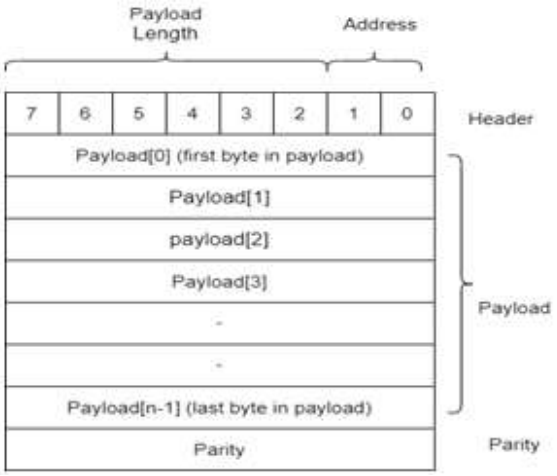

Fig 2: Data Packet Format

- **Header:** It includes the destination address and length of the data to be sent. i.e. FIFO address.
- **Destination Address:** The header's first two bits provide the destination address. It can display the address 00, 01, and 10. Here, address 11 is not valid. These two bits identify the address and route the data, regardless of the packet value provided, according to the test scenario.

- **Length:** The data length is the remaining six bits from the header. This packet may contain 1 byte or 63 bytes (26).
- **Payload:** Payload consists of data values that ought to be stored in bytes.
- **Parity:** Parity checks the accuracy of the data. There is evidence of data corruption if the parity at the source address differs from the parity at the client address. Here, it is computed by bitwise XOR operation of payload and header.

II. PROPOSED SYSTEM

1. Design Methodology

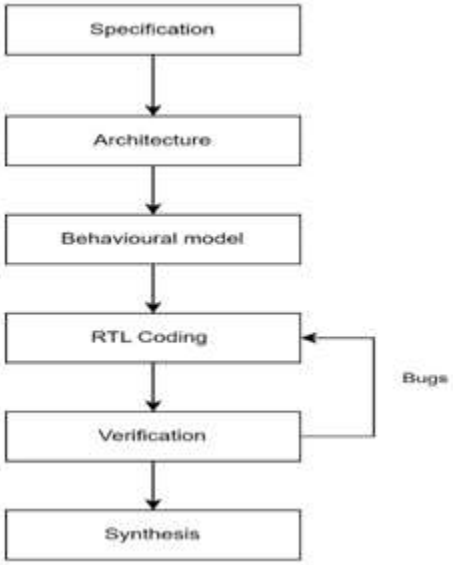

Fig 3: Design Methodology

The router has been developed and verified, the front-end—which begins with specification and ends with synthesis—is all that is left. To create the router that will route the provided source data to one of the destinations, requirements must first be created. The inputs and outputs of the packet, which are utilized to drive the input and achieve the output, are the functional aspects to take into account. The architecture of the router is then created as a block diagram, and the synthesized model is subsequently constructed using RTL coding in Verilog. Lastly, stimuli are sent to complete verification. Utilizing industry standards, the UVM Methodology is employed for validation. A test bench, known as the Design Under Test (DUT), is constructed, and the developed test bench model is used to validate the DUT. The code must be corrected in the design and the individual defect must be addressed if errors are found during verification. This process must be repeated until all errors have been debugged. The backend team then processes the netlist that is produced by Quartus Prime's implementation of synthesis.

2. Architecture

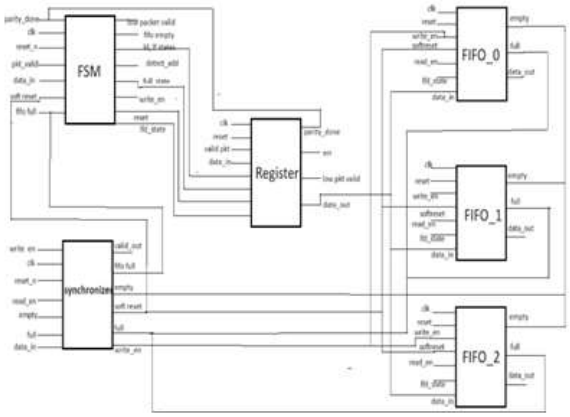

Fig 4: Internal Blocks of Router

A router is a device used for packet-based information transmission. Every time a data packet arrives on one of the lines, the router reads the network address information contained within to ascertain its ultimate destination.

Input Register: The Register acts as a loadable register and uses control signals from the FSM Controller to load packets in a specific order: Header, Payload, Parity using the data_out bus. It also generates an error signal based on the source parity and internal parity.

FSM: The FSM Controller is the central component of the architecture, generating multiple control signals to manage the other sub-blocks.

Synchronizer: The Synchronizer captures the packet address and uses it to generate a write enable signal, which is connected to the write_enb pins of all FIFOs. It also controls the router's timer logic.

FIFO: The primary function of a FIFO is to store a packet based on the relevant address driven by the source LAN.

3. Router Top Block

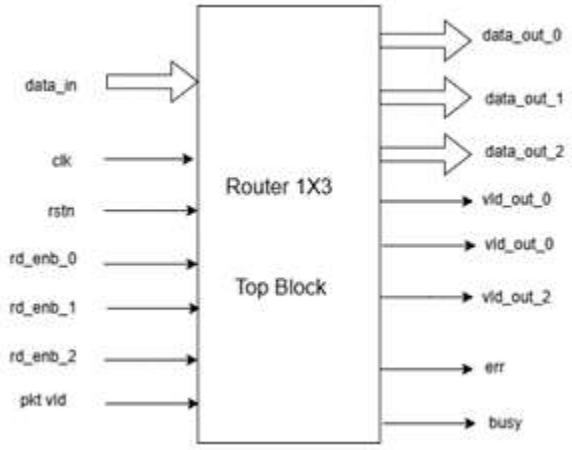

Fig 5: Router Top Block

The system has an active high clock in addition to an active low synchronous reset. This indicates that the flip-flop's state will only be changed or reset by the reset signal when the clock's active edge is reached. As part of the combinational logic that creates the flip-flop's d input, the flip-flop can be reset.

The active high input signal Pkt-vld detects a new packet from the source network. An 8-bit input data bus transports the packet from the source network to the router. To read data from the output data bus, all FIFOs maintain the read enable (read- en) pin, an active high signal similar to the data input (data-in) pin, high.

The data-out pin, sometimes referred to as the data-out, is an 8- bit data bus that transports data packets from the router to the destination client networks. Three data-out pins are present for each of the three destination networks in order to determine whether or not the data being transferred is legitimate. The valid_out pin is turned on and the signal is active high when the data being sent is legitimate. Each destination network has three valid_out pins that are used to identify a valid byte. The busy signal is an active high signal that alerts the router to a busy state. This pin causes the router to cease taking in new bytes. Furthermore, an active high signal known as the error pin is used to identify discrepancies between internal and packet parity. The router correctly receives the data by comparing the internal parity generated in the register with the packet parity, which is delivered as the final byte in the packet.

The header byte begins when pkt-vld is high, and the parity byte begins when pkt-vld is low, as the waveform shows. When the busy signal is strong, the subsequent input data byte is not transmitted from the source.

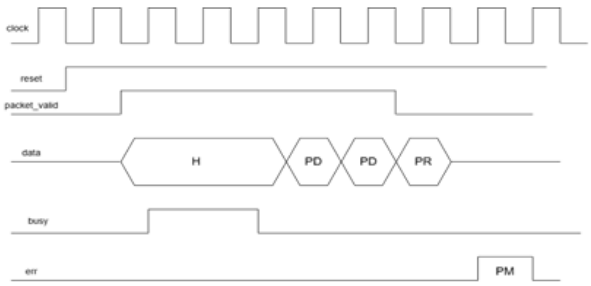

Fig 6: Router Input Protocol

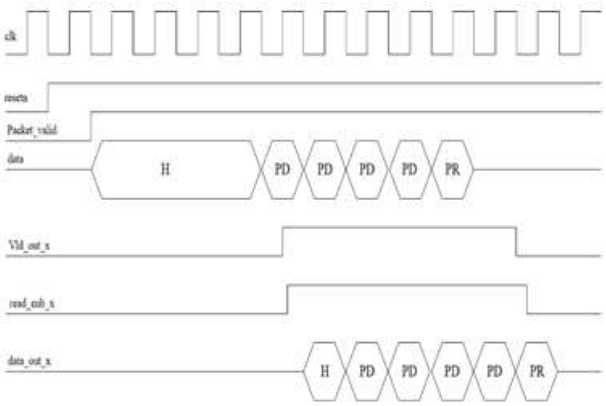

Fig 7: Router output Protocol

The waveform illustrates that a high vld-out indicates that the router has legitimate data that the destination ought to be able to access. To be readable, the destination must raise the rd-enb pin.

Once rd-enb is high, the data is read by the data-out pin. Furthermore, if vld-out is high and rd-enb is not made high within 30 clock cycles, the router will be reset by the soft reset pin.

4. FIFO Design

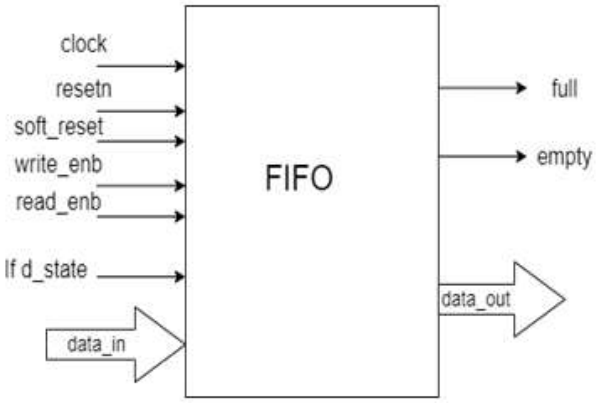

Fig 8: FIFO

Data is frequently written to and read from it using the First-In, First- Out (FIFO) protocol. FIFO is considered synchronous when read and write operations are performed using the same clock signal in this case; it is considered asynchronous when they are performed using different clock signals. The read and write operations in this synchronous FIFO use the same clock. It is 9x16 in width and depth. Data-in, clk, reset, soft-reset, lfd-state, read-enable, and write-enable are among the input pins. While read-enable and write-enable are sent by the synchronizer and are utilized for read and write operations, the soft-reset is the same as the reset. The top block's output pin is called data-out, and the FIFO outputs are empty and full. The data-in and data-out busses are both 8- bit. When the FIFO is empty, the empty pin goes high, and when the FIFO is filled, the full pin goes high.

Write Operation:

- The write operation involves sampling data_in when wr_en is high, which occurs at the clock's rising edge.
- Write operations only take place when the FIFO is not overflowing.
- Read Operation:
- When rd_en is high, the read operation reads data from data_out at the clock's rising edge.
- Reading only takes place when the FIFO is not empty.

When a header byte is read, the packet payload length plus the parity byte "1" are placed into an internal counter, which begins to decrease with each clock cycle until it approaches 0. When the FIFO memory is fully read or the router is in the timeout state, Data_out is driven to a high-impedance state.

Synchronizer

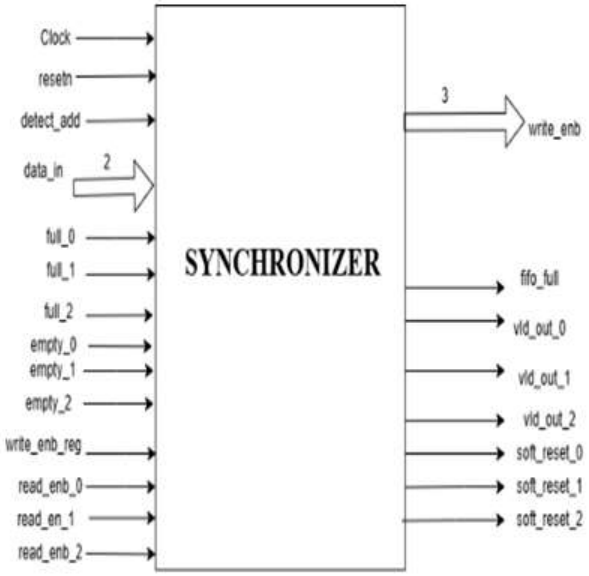

Fig 9: Synchronizer

Digital circuits are frequently subject to metastability. Circuits must be protected from its damaging effects by synchronizers. They are also utilized to provide the appropriate write enable signal for the FIFO and decode the current packet's address. As a result, only the final two bitsof data are provided. The synchronizer decodes the address of the current packet and creates the write enable signal for the FIFO by using the detect-add pin to find the address in the header byte. When using a one-hot encoder, the write enable signal is 001, and if the address is 00, FIFO1 is picked; if the address is 01, FIFO2 is selected; and if the address is 10, FIFO3 is selected. The synchronizer inputs are a number of pins on each FIFO. When either of these pins is set to high, the synchronizer's output fifo-full pin is also set to high. The destination receives the empty pins, and the validout pins are the pins that are the complement of those pins. When the destination's valid-out value is high, the matching read-enb pin should rise. When empty-0 is low, it means that there is data in FIFO1, and validout-0 becomes high. To read data from the FIFO, the destination

needs set read-enb-0 to high. Otherwise, once the synchronizer's internal timer has started counting down to 29, soft-reset-0 is set to high on the next clock pulse. As with valid-out-1 and valid-out-2, the timer starts when they are high, and soft-reset-1 and soft-reset-2 are set high if the destination does not set read-enb-1 and read-enb-2 high.

FSM

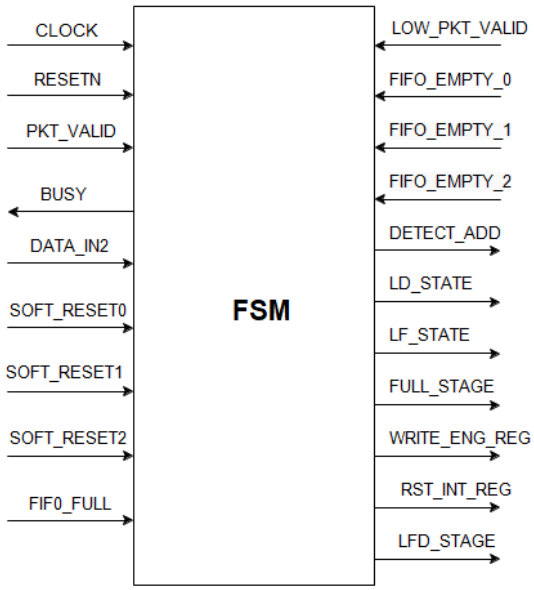

Fig 10: FSM

The Finite State Machine (FSM) is crucial for controlling complex circuits with many blocks, as it generates the required control signals. This FSM has 8 states and switches between them based on inputs, producing corresponding outputs. Similar to a synchronizer, the FSM has a 2-bit data-in to determine the destination network address and receives fifo-empty from FIFOs and soft-reset from the synchronizer. It also regulates important inputs to the Synchronizer (such as write- enb-reg and detect-add) and Register (such as ld-state, lfd-state, laf- state, full-state, and rest-int-reg). During the "Decode Address" and "Load Data" phases, when the router receives data, the busy signal stays high. The busy signal is managed by the FSM. By controlling

data flow through signals to other blocks, the FSM modifies states in response to the router's processing of input data.

Register

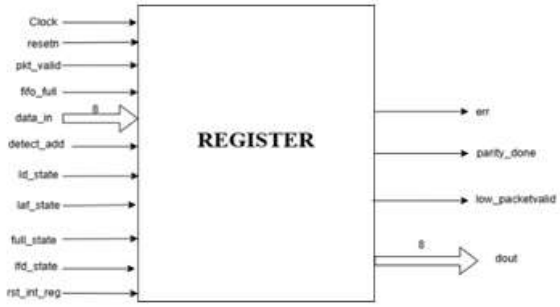

Fig 11: Register

There are eight-bit data-in and data-out pins on the register. Before being moved to the FIFO via the data-out pin, data supplied from the source to the router is first stored in the Register. Internal registers like FFB (FIFO full byte) and HHB (hold header byte) are present. In addition, the Register determines the internal parity and, after learning the parity, sets the parity-done signal high. Additionally, it checks the internal and received parity signals; if they differ, the error pin is turned high. This error pin in the Top Block is an output to the source network. The data is saved in the Register when the FIFO is full or the preceding transfer has not been finished. If it is a header byte, the detect add signal is used to send it to the FIFO after it has been placed in HHB. Data is saved in FFB and transmitted to the FIFO when the laf-state pin becomes high if the FIFO is full and data-in sends data.

5. UVM TestBench (TB) Infrastructure

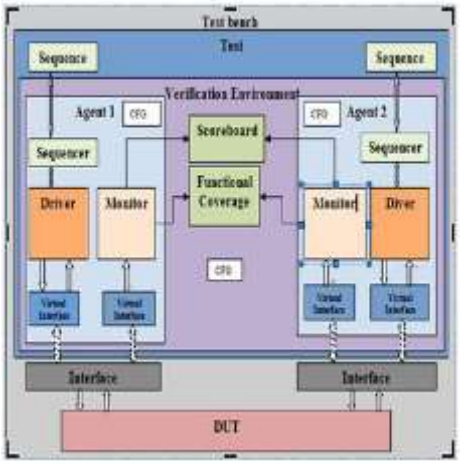

Fig 12: Test Bench Architecture

The initial stage of verification is the TB Infrastructure. The UVM classes required to construct the UVM TB Architecture components are suggested, along with an outline of those components. The DUT's interface protocol determines whether the environment contains at least one agent. Every agent ought to have the same architecture, which consists of a sequencer, monitor, and driver. The UVM scoreboard is used to verify data. The write sequence is given an input packet that is made up of a series of transactions. The sequence items are sent to the driver from the sequence via the sequencer. Only one transaction can be worked on by the driver at once. TLM ports are used for communication between the sequencer and driver. Sequence item ports are incorporated into the driver by default, and the sequencer has a built-in port named sequence item export. These ports are connected. The data is taken from the same interface by the monitor, stored in a transaction, and then transmitted to the scoreboard for coverage analysis and comparison. In the meantime, the synchronizer sends the packet to FIFO after decoding the address and utilizing FSM to control it. The read enable signal, which needs to be driven to the FIFO, should be used to read the data once it gets there. The packet must be collected from the FIFO via an interface and sent to the scoreboard via an analysis port by a monitor after the read enable signal is asserted.

III. RESULTS

The cadence program incorporates the Top-Block Level Architecture depicted above in Figure 4. Power, delay, and area are calculated when the router design circuit is put into practice.

Model	Power	Area	Density (million gates per $mm^{2)}$
1x3 Router	100mw	0.1-1mm^2	1-1.5
1x5 Router	100mw-2w	0.2-2mm^2	1.5-2.5

IV. CONCLUSION

The paper highlights the critical role of routers in directing data between networks by making efficient routing decisions based on IP addresses. It emphasizes the importance of designing low-power routers to improve energy efficiency in network systems. Through the FPGA-based approach, the proposed router was successfully designed and verified, addressing various components such as FIFOs, FSMs, synchronizers, and registers. The use of Verilog HDL for modeling and UVM for verification ensured the design's accuracy and robustness. The implementation results showed that the design met power and performance requirements, making it suitable for Network-on-Chip (NoC) applications. RTL analysis was utilized to proactively identify and resolve synthesis issues early in the design process, enhancing the reliability and optimization of the router.

REFERENCES

1. V. G. K. Sista, S. S. S. Keerthy, M. Shiva Prasad, P. R. Reddy and V. S. B. Telluri, "Design and Verification of 1X3 Router," 2023 International Conference on the Confluence of Advancements in Robotics, Vision and Interdisciplinary Technology Management (IC-RVITM), Bangalore, India, 2023, pp. 1-13, 2023.
2. S. Madhavan and H. P. V, "Design and Verification of 1X5 ROUTER," 2022 IEEE 2nd Mysore Sub Section International Conference (MysuruCon), Mysuru, India, pp. 1-6,2022.
3. E. M. Choudhari and P. K. Dakhole, "Design and verification of five port router for network on chip," 2014 International Conference on Communication and Signal Processing, Melmaruvathur, India, pp. 607-611, 2014.
4. Tsung Lee and Yao-Yi Huang, "Analysis and synthesis techniques of router circuits," 2015 4th International Conference on Computer Science and Network Technology (ICCSNT), Harbin, China, pp. 838- 841, 2015.
5. C. Mattihalli, S. Ron and N. Kolla, "VLSI Based Robust Router Architecture," 2012 Third International Conference on Intelligent Systems

Modelling and Simulation, Kota Kinabalu, Malaysia, pp. 43- 48, 2012.
6. An Wei, Yuqiang Chen and Jinhua Wu, "Simulation Study of TCP/IP Communication Based on Networked Control Systems," 2006 6th World Congress on Intelligent Control and Automation, Dalian, pp. 4479-4483, 2006.
7. M. A. Aslam, S. Kumar and R. Holsmark, "An Efficient Router Architecture and Its FPGA Prototyping to Support Junction Based Routing in NoC Platforms," 2013 Euromicro Conference on Digital System Design, Los Alamitos, CA, USA, pp. 297-300, 2013.
8. G. Feng, Y. Xu, K. Li, D. Zhou and L. Wang, "Design and Implementation of On-orbit Reconfigurable Inter-satellite Links Router," 2019 IEEE 11th International Conference on Communication Software and Networks (ICCSN), Chongqing, China, pp. 599-602, 2019.
9. C. N. Ranjitha and B. K. Sujatha, "Power Aware design and Convergence of Router using Unified Power Format Standards," 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), Bangalore, India, pp. 307-312, 2019.
10. A. Bhanwala, M. Kumar and Y. Kumar, "FPGA based design of low power reconfigurable router for Network on Chip (NoC)," International Conference on Computing, Communication & Automation, Greater Noida, India, pp. 1320-1326, 2015.
11. K. Goossens, J. Dielissen, O. P. Gangwal, S. G. Pestana, A. Radulescu and E. Rijpkema, "A Design Flow for Application-Specific Networks on Chip with Guaranteed Performance to Accelerate SOC Design and Verification," Design, Automation and Test in Europe, Munich, Germany, pp. 1182-1187, 2005.
12. B.Tasneem and M. Wahid, "A Review of Secure Routing Challenges in Low Power and Lossy Networks," 2021 International Conference on Communication Technologies (ComTech), Rawalpindi, Pakistan, pp. 120-125, 2021.
13. A. Bhanwala, M. Kumar and Y. Kumar, "FPGA based design of low power reconfigurable router for Network on Chip (NoC)," International Conference on Computing, Communication & Automation, Greater Noida, India, pp. 1320-1326, 2015.
14. S. Mondal and S. O. Memik, "A low power FPGA routing architecture," 2005 IEEE International Symposium on Circuits and Systems, Kobe, Japan, pp. 1222-1225 Vol. 2, 2005.