VLSI Implementation of Reducing Area By Using 1x5 Robust Router Architecture

N. KESAVAMMA[1], MR.P. SURESH[2]
[1]PG Scholar, Dept of ECE, GLOBAL College of Engineering &Technology, Kadapa , AP, India.
[2]Assistant Professor, Dept of ECE, GLOBAL College of Engineering & Technology,Kadapa, AP, India.

Abstract: The use of router provides better conservation against hacking than a software firewall because no computer Internet Protocol address directly exposed to the Internet, this makes port scans essentially impossible. Router device or in some cases software in a computer determines the network point to which a packet should be forwarded toward its destination. In existing many routers used to developed, have some problems among those routers the number of input should be large and they require large area more power consumption and starvation effect. One of the method index based round robin blocks used to reduce the starvation effect and to increase speed but it require large area and system complexity. The 1x5robust router used has single input and five outputs internally contain register flip-flop and finite state machine. Xilinx ISE EDA Tool used for synthesis and Modelsim used for simulation.

Keywords: Router_1x5 Architecture, Network on Chip, Flip-Flop Synchronization, NOC, RIP, ROP.

I. INTRODUCTION

Router performs traffic direct functions on the Internet. A data packet typically sent from one router to another through the networks that constitute the internal network until it reaches its destination node.

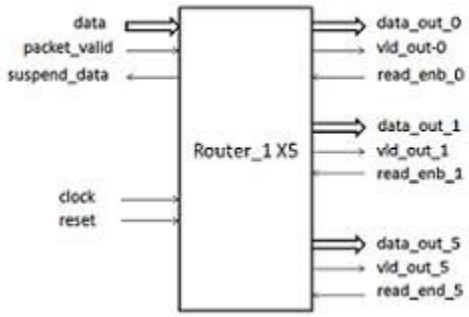

Figure.1 Block Diagram of Router_1X5.

We research the blocks used in NOC system router packet based protocol. It has one input port from which the packet enters and 5 output ports where the packet driven_out. Active low synchronous input reserten which reset the router**.** Packet contains three parts Header, payload and parity Packet width is eight bits.

II. EXISTING SYSTEM

The routeris a " Network Router " has a one input port from which the packet enters. It has three output ports where the packet is driven out. Packet contains 3 patrs. They are Header, data and frame check sequence. Packet width is 8 bits and the length of the packet can be between 1 bytes. Destination address(DA) of the packet is of 8 bits. The switch drives the packet to respective ports based on this destination address of the packets. Each output port has 8 bit unique port address. If the destination address of the packet matches the port address, then switch drives the packet to the output port, Length of the data is of 8 bits. In this paper the Xilinx ISE EDA Tool is used for synthesis and Modelsim is used for simulation.

III. PROPOSED SYSTEM

A. Router Input Protocol:

The characteristics of input protocol:

1. All input signals active high and synchronized to falling edge of the clock. But driving input signals on the falling edge ensures setup and hold time, therefore the signals driven on the rising edge of the clock.
2. The packet valid signal asserted on the same clock when packet (the header byte) driven onto the data bus.
3. The header byte contains address to which output channel packet routed(data_out_0, data_out_1/ data_ out_2).
4. The packet valid signal asserted same clock when the first byte of packet driven onto data bus.
5. The header byte contains address to output channel (data_out_0, data_out_1, or data_out_2) data bus.
6. The width of suspend data signal assertion should not exceed 100 cycles. The error signal asserts when packet with bad parity detected in router, within 1 to 10 cycles of packet completion.

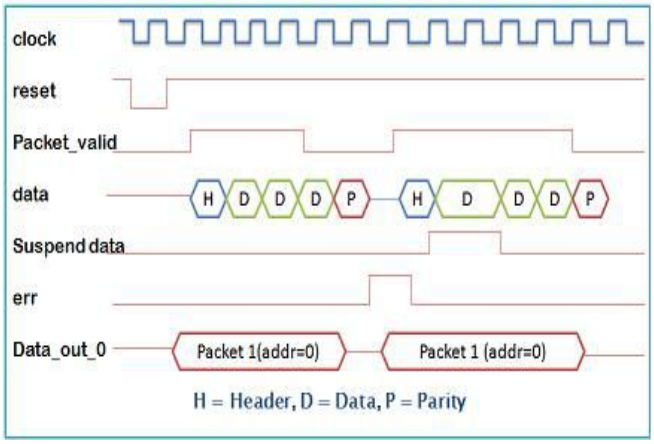

Figure.2 Router Input Protocol signal diagram.

B. Router Output Protocol:

The characteristics of output protocol: All output signals active high and synchronized to rising or falling edge of the clock, Receiver drive sample data at rising or falling edge of the clock and router drive sample data at rising string of clock.

1 To each output port data_out_x (data_out_0 to data_out_4) internally buffered by FIFO of 1 byte width and 16 location depth.
2 Router asserts vld_out_x (vld_out_0 to vld_out_4) signal when valid data appears on the vld_out_x (data_out_0 to data_out_4) output bus Signal to packet receiver, valid data available on a particular router.
3 The packet receiver wait until enough space to hold bytes of packet and then respond with assertion of read_enb_x (read_enb_0 to read_enb_2) signal input to the router.
4 The read_enb_x (read_enb0 to read_enb_2) input signal asserted on rising/falling clock edge in which data read from the data_out_x(data_out_0 to data_out_4) bus. As long as read_enb_x (read_enb_0to read_enb_2) signal remains active high the data_out_x(data_out_0 to data_out_2) bus drives a valid packet byte on each rising clock edge.
5 The packet receiver cannot request the router for suspend data transmission while middle of the packet. Therefore packet receiver must assert the read_enb_x (read_enb_0 to read_enb_4) signal and it ensures adequate space to hold the entire packet.
6 The read_enb_x (read_enb_0 to read_enb_4) asserted within 30 clock cycles of the vld_out_x (vld_out_0, to vld_out_4), otherwise congestion present in the packet receiver.

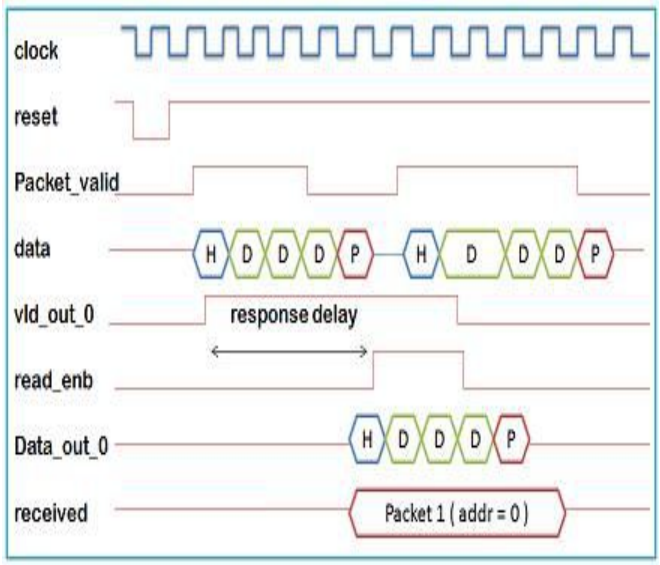

Figure.3 Router Output Protocol signal diagram

Router_1X5 Architecture: The proposed design mainly consists of eight blocks fsm-router, fsm-reg, fsm-sync and 5-FIFO's.The fsm-router provides necessary signal to control fsm-reg and fifo's. Status data and parity registers for router_1x5 contained the fsm-reg. The fsm-router provides necessary control signal to registers that latches to new status. The five-FIFO's for each output port stores the data from input port based on the control signals donated by fsm_router module. Synchroni-zation provided between fsm_router module and 5-FIFO's provided by fsm sync, such single input port and 5 output ports communicate faithfully.

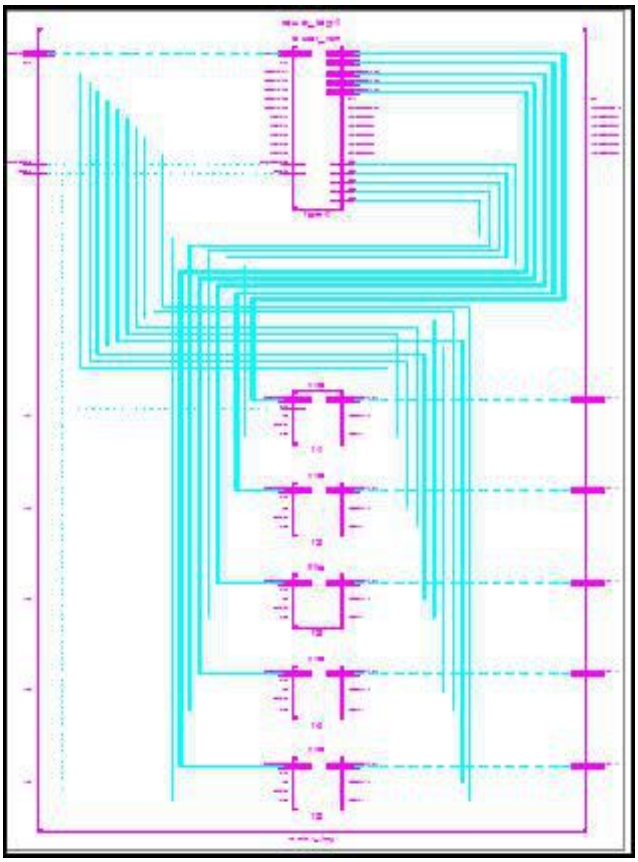

Figure4. Architecture of Router_1X5

C. FIFO Block:

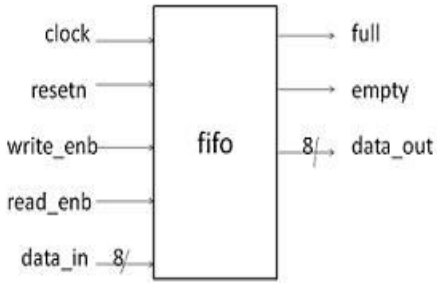

Figure5 : FIFO Block Diagram

5-FIFO's used for router design an eight bit width and sixteen bit depth FIFO works on system clock. Its synchronous input signal reset, If resetn low then full =0, empty = 1 and data out =0

Write operation: When input wrt_enb high and FIFO not full, the data from input data_in sampled at rising edge of the clock.

Read Operation: When read_enb high and FIFO not empty, the data read from output data_out at rising edge of the clock.

Full – It indicates that all locations inside FIFO have been written.

Empty –It indicates that all the locations of FIFO empty.

FF_sync block:

Synchronization provided between fsm_router module and 5 FIFO's provided by fsm-sync such single input port and five output ports communicate faithfully.

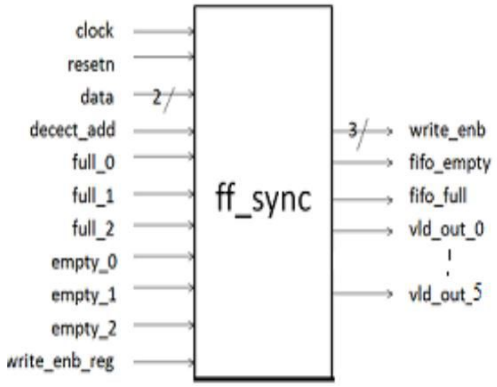

Figure6 : Flip-Flop Synchronisation Block

It find address of the channel and it latch till packet valid asserted, address and write enable used for latching the incoming data into the FIFO of particular channel. A fifo_full output signal generated, when present FIFO full, and fifo_empty output signal generated when the present FIFO empty.

- If data = 00 at that time fifo_empty = empty_0 and fifo_full = full_0
- If data = 01 at that time fifo_empty = empty_1 and fifo_full = full_1
- If data = 10 at that time fifo_empty = empty_2 and fifo_full = full_2
- Else fifo_empty =0 and fifo_full = 1

Output Vld_out signal generated when empty of present fifo_ low means present FIFO ready to read.

Vld_out_0 = ~empty_0
Vld_out_1 = ~empty_1
Vld_out_2 = ~empty_2
Vld_out_3 = ~empty_3
Vld_out_4 = ~empty_4

Write_enb_reg signal which comes from the fsm used to generate write_enb signal for present fifo which selected by present address.

D. Router_reg block:

Module contains status, data and parity registers. Rising edge of the clock latches all the registers based on state and status of control signals. Data registers latches data from data input and latched data sent to FIFO for storage. Apart from it, data also latched into parity registers for parity calculation and it compared with parity byte of packet. An error signal generated if packet parity not equal to the calculated parity. If resetn low then output (dout, err, parity_done and low_packet_valid) low. The output parity_done goes high.

- when the input ld_state high, fifo-full and packet_valid is low
- When the input laf_state and output low_packet_valid both are high and the previous value of parity_done low. It reseted to low value by reset_int_reg signal. The output low_packet_valid is high
- The input ld_state goes high and packet_valid go to low.It reseted to low by reset_int_reg signal.

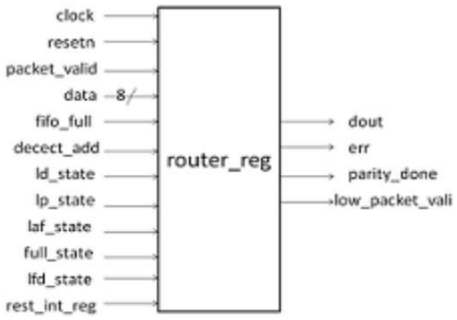

Figure7 : Router Register Block

First data byte header latched inside the internal register detect-add, packet valid signals latched to output dout when load full data state signal high, Then input data payload latched to output dout if input data state signals high and fifo_full low and the input data parity latched to output dout if load parity state signals high and fifo_full low. When load full_state and fifo_full high the input data latched to internal register full_state byte. The parity calculated packet stored in data internal parity register, when packet transmitted fully. Internal calculated parity compared with parity byte of the packet, error signal generated packet parity is not equal to calculated parity.

E. FSM Block

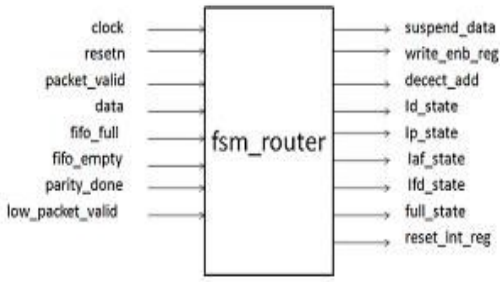

Figure8 : FSM Block

Fsm_router module acts controller circuit for the router. This module generates all control signals when new packet sent to router. These control signals used by other modules to send data at output, writing data into the FIFO.

F. FSM State Diagram

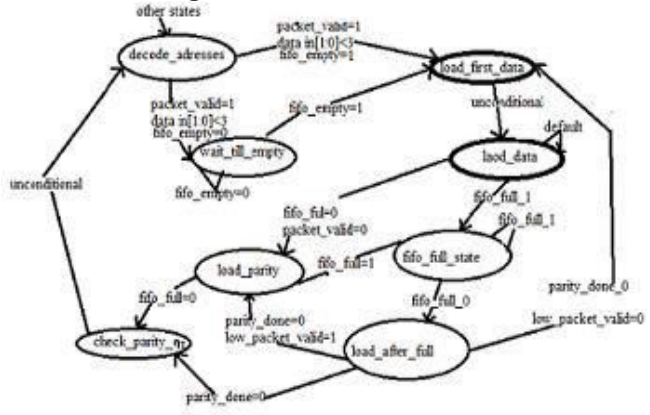

Figure9 : FSM State Diagram

STATE - DECODE_ADDRESS: The default state wait for packet_valid assertion, after packet_valid signal high if the address valid and fifo for that address empty (fifo_empty

signal high) data loaded so it goes to next state load first data, If fifo shouldn't empty the wait_till_empty state so new data couldn't be accepted till fifo ready. The output signal detect_add made high ff_sync module detect address of fifo used. detect_add signal also used by router_reg module, to latch the first byte in internal register.

STATE -LOAD_FIRST_DATA: Load_full data state signal generated which indicates router_reg module first data byte latched, at the same time suspend data signal made high so that first data byte faithfully latched inside the output data register in router_reg module. The next clock edge unconditional state changed to load data.

STATE - LOAD_DATA: This state data latched inside the data registers of router_reg module, for this ld_state signal generated for router_reg module. suspend_data signal made low, so that router can accept the new data from input simultaneously, latched data sent to the fifo and write_enb_reg is generated for writing into present fifo. If fifo_full input goes high then no more data can be accepted by router so it goes to fifo_full_state. Data is latched till the packet_valid signalasserted, when it is de-asserted in load_data state, it goes to load_parity state, where last parity byte latched.

STATE – LOAD_PARITY: This state last byte latched which is parity byte. If fifo_full high, data cannot be latched, so it goes to fifo_full_state else if fifo_full low it goes to state check_parity_error and Signal lp_state generated for router_reg module. suspend_data signal made high so now router couldnot accepts any new data and write_enb_reg made high for latching the last byte. lp_state signal is generated for the router_reg module, so that last byte can be latched and the parity bytes can be compared.

STATE -FIFO_FULL_STATE: Neither new data accepted nor any data latched, so suspend_data signal made high and write_enb_reg signal made low. Full_state signal generated for router_reg module, state changes to load_after_full state when fifo_full becomes low.

STATE -LOAD_AFTER_FULL: This state laf_state signal generated for router_reg so that it can latch the data after fifo_full_state, no new data accepted so suspend_data kept high and last data is latched in router_reg module for that write_enb_reg made high. It checks for parity_done register which if high shows that load_parity state has passed, if parity_done high it goes to the last state check_parity_error. Then it checks for low_packet_valid register, which if high shows that packet_valid for present packet has been deasserted, if low_packet_valid high it goes to load_parity state otherwise it goes back to the load_data state.

STATE–WAIT_TILL_EMPTY: Neither new data accepted nor data latched by router_reg module so suspend_data signal Made high and write_enb_reg signal made low. It waits for fifo_empty signal high to load_first_data state.

STATE–CHECK_PARITY_ERROR: Reset_int_reg signal generated to reset the status and parity registers inside neither the router_reg module, neither any data latched nor any input data accepted. Router_reg compares the data parity from packet with calculated parity during state and State changes to default state decode_adress with next clock edge.

Table1. Device Utilization Summary:

Logic Utilization	Proposed Utilization	Existed Utilization	Available
Number of Slices	134	1287	4656
Number of Slice Flip Flops	126	1203	9312
Number of 4 input luts	253	724	9312
Number of bonded iobs	43	148	232
Number of gclks	3	7	24

IV. CONCLUSION

In this paper reduced area (Number of LUT'S) of a five port router presented, The proposed router structure functionality implemented in Verilog HDL and proven that architecture consumes less resources in terms of number of LUT"S, slices and number of IO Buffers. In this project the Xilinx ISE EDA Tool used for synthesis and Modelsim used for simulation, the data which can be send through the router reached the destination with 9.375ns latency, In future chance to estimate the power consumpation also.

V. REFERENCES

[1]. Bluespec Inc http://www.bluespec.com

[2]. Xilinx, "ML605Hardware UserGuide", **i**http://www.xilinx.com/ support/ documentation/ boards&kits/ug54.pdf.

[3]. Xilinx, "logicore IP Processor Local Bus(PLB) v4.6", http://www.p.com/support/documentaion/ipdocumentation/plb v46.pdf.

[4]. P. Wolkotte, P.Holzenspies, and G.Smit, "Fast, Accurate and Desailed NOCSimulations", NOCS, 2007

[5]. M.Pellauer, M.Adler, M.Kinsy, A.Parashar, and J.Emer," HAsim:FPGA-BasedHigh-DetailMulticore Simultion Using Time-Division Multiplexing", HPCA, 2011.

Nukala Keshavamma, M.TECH in very large scale system design at Global College of Engineering & Technology at Kadapa,A.P.

Potladurthi Suresh, Head of the Department of ECE in Global College of Engineering&Technology At Kadapa, A.P.