Project 1 – Designing a Cycle Accurate Simulator for Network-on-Chip (NoC) router and mesh with provision to study the impact of Process Variations

What is an NoC?

As Systems-on-Chips become more demanding, complex, and elaborate, interconnection via traditional bus-based networks became highly inefficient and slow, giving rise to the development of network-on-chip architectures. The basic and major advantage of NoCs over bus communication is that multiple processing elements (PEs) like CPU, GPU, ADC, Memory, or any other IP, can communicate with one another simultaneously (parallelly) with the help of “Routers” attached to every peripheral. As the name also suggests, Routers decide to direct the data to various network peripherals (connected to other routers). Every router is connected to at least two other routers and one PE.

In a mesh design, a router has five ports - four for other routers (North, South, East, and West) and one for PE. Note that not all ports of the router need to be connected. For example, in the NoC shown below (Fig. 1), the northern ports of routers A and B are not connected to anything.

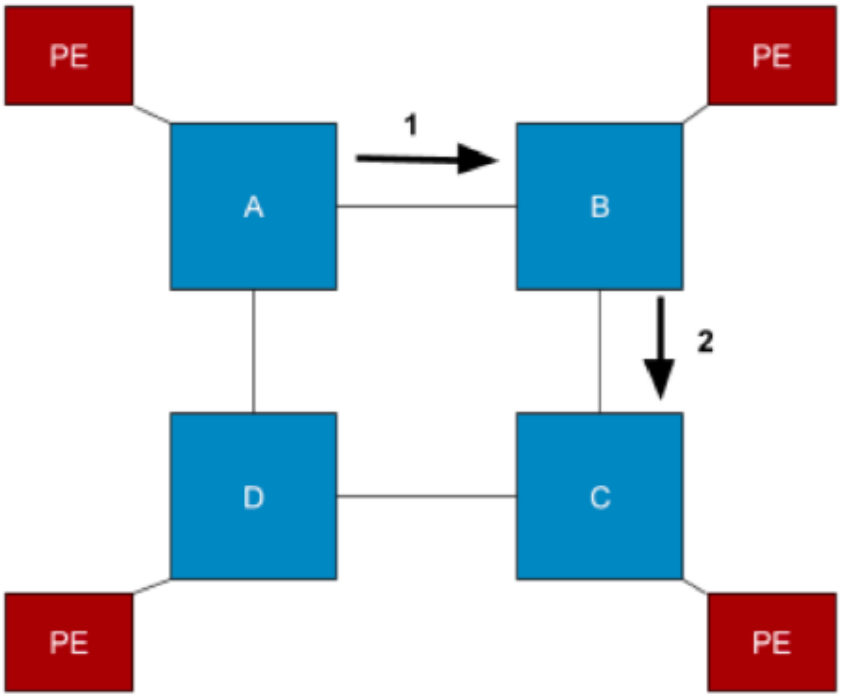

Fig. 1. Simple 2X2 mesh NoC Architecture with Processing Elements

Now, let us consider that router A has to send some data to router C. Two possible paths include ABC and ADC. To remove ambiguity in the path selection, “routing algorithms” are used. “XY routing” is one such algorithm that prioritizes horizontal movement (X-axis) over vertical (Y-axis) one. In this example, XY routing allows movement from A to B but not through A to D. So, the XY routing path in the above example is through ABC. This path is demonstrated in Fig. 1 using arrows. Since we are using XY routing, the packet first travels in the X direction (arrow number 1) and then travels in the Y direction (arrow number 2). Note that if no horizontal path exists, only vertical movement is done. For instance, vertical movement from A to D is done if the data transfer between A and D is needed.

A “packet” is a basic data transfer unit in an NoC. However, the packets cannot be transmitted from one node (source) to another (destination) in a single go. To aid transmission, the packet is broken down into flits. The flits can be transferred over a single go; hence, the packet is transmitted as a series of flits. Flits are of three types - “header flit (HF), body flit (BF), and tail flit (TF).” The header flit contains the control information (source, destination, etc.) and is responsible for creating the path for the network. The body flits contain the data to be transferred, and the tail flits indicate the end of the packet.
The following Fig. 2 shows a simple NoC router microarchitecture. The router has one input and one output port in each direction (North, South, East, and West). The “crossbar (XBAR)” connects every input to every output of a NoC (except in the same directions). The “switch allocator (SA)” configures the crossbar according to the header flit and routing algorithm. Whenever a packet (or flit) enters a router, it is initially stored in the “input buffers” corresponding to the respective “input port”. For instance, in the above Fig. 1, router A’s PE will inject the packet (or flit) at the local input port, which will stay in buffer till it is decoded as the type of flit, then switch allocator will configure the XBAR to connect local input port to east output port which will be received by router B’s west input port followed by the buffer. The router B’s switch allocator will configure the crossbar to connect the west input port to the south output port. Further details about NoC can be found in the attached book.

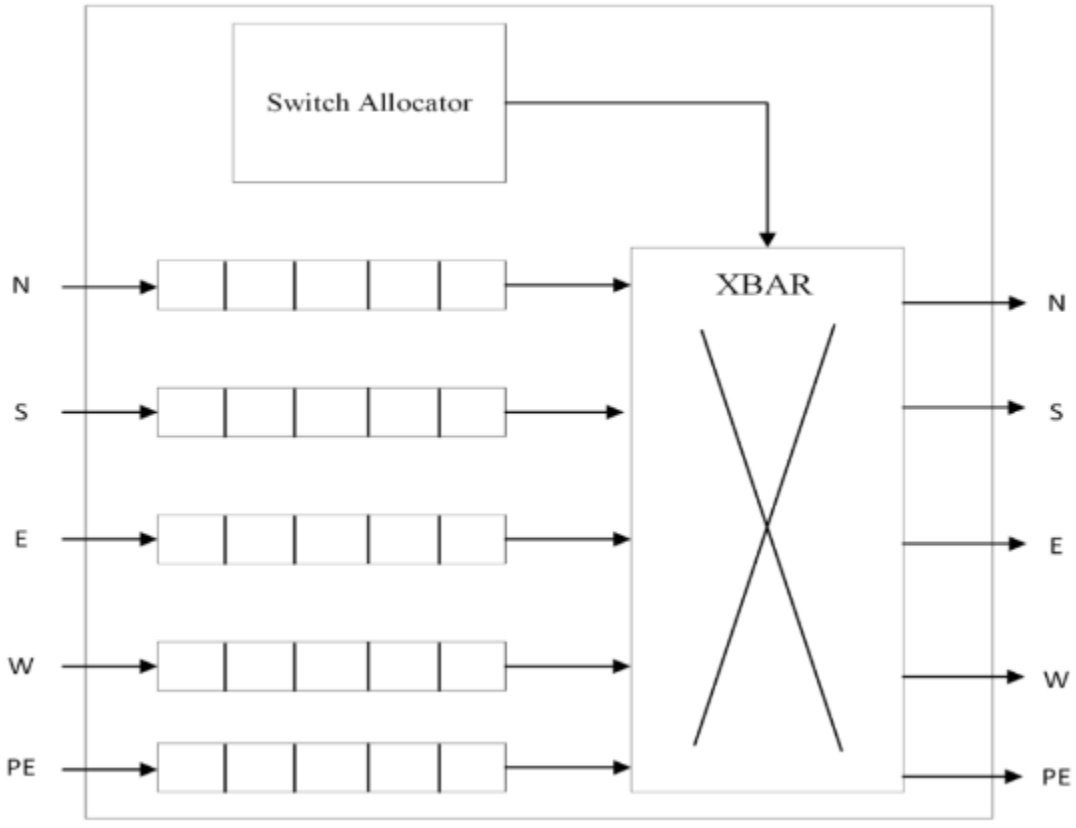

Fig. 2. Router Microarchitecture

Project Description:

In this project, I created a NoC simulator in python language.

You have to simulate a NoC router and a 3x3 NoC mesh containing nine routers. As described above, each router needs to have five ports (north, south, east, west, and local) to connect to its neighbors and connect to its local PE. Some ports of the routers may be left unconnected if there is no neighbor (for example, the north port of router B in Fig. 1).

Your router must support XY routing as described above. Moreover, your router must also support YX routing, in which the packet travels first in the Y direction and then in the X direction. The type of routing to be used will be passed to your simulator via the command line arguments.

The simulator needs to support reading a “traffic file” that describes which packets (split into three types of flits: HF, BF,& TF) are inserted in the NoC at which respective clock cycles. The traffic file is a text file with four values mentioned in each line, separated by spaces. The value in the first column describes the clock cycle where a packet/flit is inserted into the NoC. The second and third columns’ values represent the source and the destination for the packet, respectively. The fourth value is a 32-bit value representing the flit which needs to be injected in NoC at the clock cycle (the first column).

→ The Bits of the flit-types of a Packet are defined as follows:

a. Head Flit (Flit Type = 00) :

Destination Node ID															Source Node ID															Type	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

b. Body Flit (Flit Type = 01) :

Payload (30 Bits Wide)																													Type	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

c. Tail Flit (Flit Type = 10) :

Unused (any value)																													Type	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The simulator must be cycle accurate, which means that your simulator must be able to simulate clock cycles. Your routers' data transfer and state update must functionally happen at the positive clock edges.

Each router must have a crossbar, a switch allocator, input buffers, and I/O ports. You can find the details of these sub-components in the aforementioned description. These sub-components must be visible in the code.

Every router in your NoC must accept the packets (split into flits) via the local input port (ideally connected to PE) at the specified clock cycle in the “traffic file”.

The simulator must read the delay of individual router elements (SA, XBAR & buffers) via a "delays file". Nominal Delays to the routers' elements described in Fig. 1. must be defined in the delays file for all of the nine routers (3x3 NoC mesh) elements. The clock frequency has to be derived from these delays assuming that the router works in a pipelined fashion, i.e. first, the flit enters the buffers, then if it is a header-flit, it gets decoded by the SA unit. Then, the crossbar gets switched accordingly, and the flit passes through XBAR to the next node. These three operations in buffer, SA, and XBar must be assumed to be pipelined for a packet containing three flits: head, body, and tail. So, the final time period of the clock depends on the element of the router having maximum delay.
For this project, you may assume there will be no congestion issues, i.e., no flit will have to wait in the buffer for the previous flit to be handled.
You may implement the connection as an array in which the sender writes the data to the receiver's array.

You have to enable two types of Simulation Modes for simulation, analysis, and reporting: Process-Variation-Agnostic (PVA) and Process-Variation-Supported (PVS):

a. PVA: This is the FIRST simulation mode. The chosen nominal delays of individual router element(s) (buffer, SA, and XBar) must remain the same across the complete NoC. The clock frequency is derived using the delay file's nominal delays. For example, if you choose buffer, SA, and Xbar delay to be x, y & z, respectively, then these, "x, y & z" values should be applied to all the routers in NoC.

b. PVS: This is the SECOND simulation mode. Following a Gaussian Normal Distribution, the above delays should be randomly assigned to the different routers' elements in this simulation mode by your simulator only. However, the clock frequency decided during PVA will remain the same without any modification in this mode. The mean value of the delay distribution will be the nominal delays considered in PVA, and sigma (standard deviation) must be taken as 10% of those nominal delay values. You must ensure that delays ranging from "mean + 3*sigma" and "mean – 3*sigma" must be spread according to the Gaussian Bell curve of standard normal distribution across the complete NoC Mesh. For example, the delay "x" of the buffer element must be varied from "x + 3*sigma_x" to "x - 3*sigma_x" where "sigma_x = 10% of x", across the complete NoC. The same applies to other element delays as well.

The simulator must generate a log file that indicates which flit is received at each cycle (cycle number should be displayed), at what router, and at what element of that router so that we can trace the flit flow at every clock cycle. Your log file must indicate the contents of the flit received at every clock cycle. The log file should indicate the clock cycle(s) at which different flits were inserted in the NoC.

Additionally, a separate report file must be generated from the simulator, which indicates the impact of process variation applied in PVS mode, i.e., if any node cannot receive a particular flit sent (in case the flit gets delayed than expected time in nominal simulation mode) or not or if

any node receives the flit earlier than the expected delay time. So, in addition to clock cycle-wise tracking of packets, your report has to indicate in terms of delay units, which flit took what exact delay units to reach which exact element of which exact router in the 3x3 NoC first for PVA simulation mode and then for PVS simulation mode for an apple-to-apple comparison.

The user (here, the evaluator TA) here gives three inputs to the simulator,

a. Traffic file
b. Delays file
c. Choice of routing algorithm XY or YX

The simulator is expected to generate two output files and two graphs:

a. Log file (as explained in Point #12.)
b. Report file (as explained in Point #13.)
c. A graph that plots the number of flits sent over a connection. A connection is a link between two routers or between a router and its PE. So in your case, you will have eight links. Your graphs must show the number of flits sent over each of these eight links as a bar graph. Plot for both PVS and PVA modes.
d. A graph showing the packet transfer latency for each packet. Each packet transferred via the NoC will take some clock cycles to go from the source to the destination. You must plot this latency as a function of packets sent. Plot for both, PVS and PVA modes.