Machine Learning Intelligent Chip Design

[HW3] Implement NoC by SystemC

Description

NoC (Network-o[illegible]g architecture that can help overcome communication l[illegible]ormance limitations in modern computer systems. It decouples computing resources from communication resources, allowing for large-scale parallel processing and highly flexible communication channel configurations that can be optimized based on specific application requirements. Additionally, NoC is highly fault-tolerant and scalable, providing a powerful foundation for future integrated circuit and system architectures.

Implementation Details

In HW3, you are required to implement a 4x4 mesh-based NoC architecture as shown in Figure 1. The system architecture includes the following two types of modules:

- **Router**: The routers will be responsible for routing flits between different components within the network.
- **Core**: Each router will be connected to a core module, which includes the Processing Element (PE) and the Network Interface (NI). The PE generates data packets, while the NI manages communication between the PE and the router.

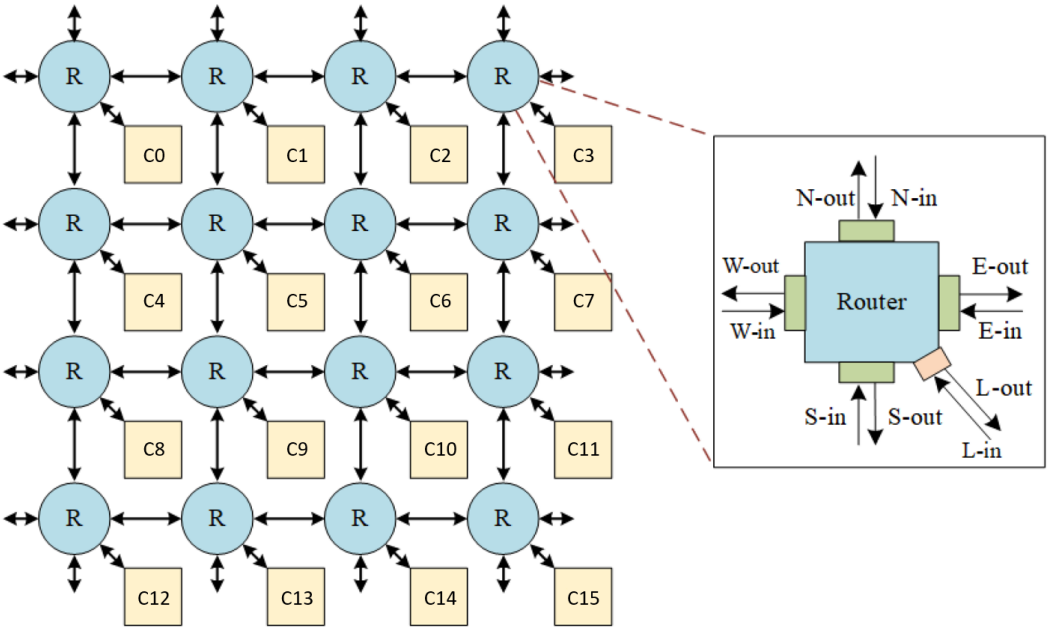

Figure1. 4x4 mesh-based NoC architecture

To simplify the complexity of system design for this assignment, TA will provide a pre-written PE. The PE will be encapsulated within the core module and mainly consists of three functions:

- **`void init(`**
  You need to [illegible]he beginning of the simulation. The `pe_id` is numbered s[illegible]rom 0 in the upper-left corner, as shown in Figure 1.
- **`Packet* ge`**
  Each time you [illegible] you can obtain a send packet. If the PE has no more packets to send, this function will return `nullptr`.
  The definition of a packet is shown in Figure 2, each packet contains a `source_id` and a `dest_id`, along with a floating-point vector `datas`. The length of the vector in each packet is different.


```
struct Packet {
    int source_id;
    int dest_id;
    vector<float> datas;
};
```

Figure 2. Packet structure

- **`void check_packet(Packet* p)`**
  When all flits of a packet are received, you need to pack these flits into a packet and send it to the PE by calling this function. The PE will verify whether the packet is correct. When all PEs receive the correct packets, the simulation will stop immediately and display the following screen.


```
0:59 mlchipTA02@ee21[~/hw3_student]$ ./run

        SystemC 2.3.3-Accellera --- Mar  2 2024 23:27:20
        Copyright (c) 1996-2018 by all Contributors,
        ALL RIGHTS RESERVED

  ============================
  +                          +          |\__|\
  +   Congratulations !!     +         / 0.0  |
  +                          +       /_____   |
  +   Simulation completed   +      /^ ^ ^ \  |
  +   at 220 th cycle        +     |^ ^ ^ ^ |w|
  +                          +      \m___m__|_|
  ============================
```

Figure 3. Screenshot of successful simulation

Please take a screenshot and place it in your report, ensuring that your workstation account is in the picture.

Additionally, TA also provides the port definitions of Core and Router modules (Figure 4). You need to connect the core to the router, and the router to the top, bottom, left, and right routers, as shown in the architecture in Figure 1. As a reminder, since [illegible]s limited to 34 bits, the packet should be decomposed be[illegible]outer.


```
SC_MODULE( Co
    sc_in  <
    sc_in  <
    // receiv
    sc_in  < sc_lv<34> > flit_rx;
    sc_in  < bool > req_rx;
    sc_out < bool > ack_rx;
    // transmi
    sc_out < sc_lv<34> > flit_tx;
    sc_out < bool > req_tx;
    sc_in  < bool > ack_tx;
```

```
SC_MODULE( Router ) {
    sc_in  < bool >  rst;
    sc_in  < bool >  clk;

    sc_out < sc_lv<34> >  out_flit[5];
    sc_out < bool >  out_req[5];
    sc_in  < bool >  in_ack[5];

    sc_in  < sc_lv<34> >  in_flit[5];
    sc_in  < bool >  in_req[5];
    sc_out < bool >  out_ack[5];
```


Figure 4. Port definitions of Core module and Router module

Figure 5 is an example of flit format definitions, the first two bits are used to identify the header, body or tail flit. You can reference the definition example or customize the flit format and even modify the port definition. Please explain your design considerations (such as latency, bandwidth, complexity, etc.) in detail in the report.

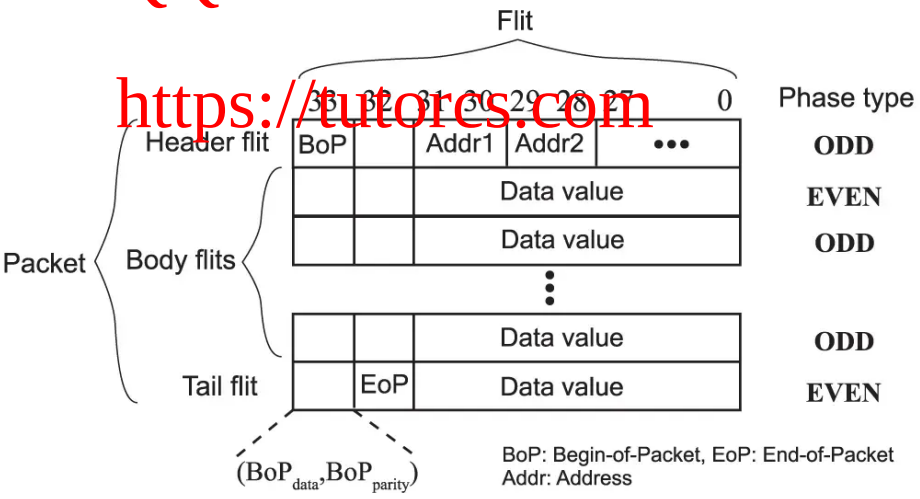

Figure 5. Example of flit format definitions

In the main function, there are three separate parts. The signals declaration, modules declaration and modules connection. You can reference Figure 4 to declare all the signal you need in the main function and interconnect these routers and cores to construct your network.

For the pattern files, the data format is “TO <dest_id> <data length> <data>” or “FROM <source_id> <data length> <data>”. Each PE will read the corresponding file in the pattern folder according to its `id`. You don't need to process them yourself, but understanding th bug.

Implement N

A key aspect of will be the choice of routing policy employed by the routers (e.g. st adaptive routing, etc.). This policy will determine how data packets are transmitted through the network and affects simulation time.

It is important to note that in this assignment, only `sc_in` and `sc_out` can be used for ports. Channels and interfaces that were utilized in HW2 are not allowed. Additionally, using pointer in port definition is also forbidden as it doesn't make sense in hardware design.

Submission Guidelines

- **Please compress a folder named HW<ID>_<studend-ID> into a zip file with the same name and upload it to E3.**

- **The folder should include:**
  - Report (Name: HW<ID>_<student ID>.pdf)
  - Codes
  - Makefile
  - pattern folder

- **Example:**

```
HW3_123456789.zip
└── HW3_123456789
    ├── HW3_123456789.pdf
    ├── ...
    ├── main.cpp
    ├── Makefile
    └── pattern
```

- **Ensure that your code is well-commented and organized for clarity and understanding.**
- **Plagiarism is forbidden, otherwise you will get 0 point!!!**

Deliverables

- **SystemC Implementation:**
  Use SystemC to implement the 4x4 mesh-based NoC architecture.
- **Report:**
  A brief repor ng
  - Simulat workstation account.
  - How do er and NI? What routing algorithm do you use? What is fer? Do you use virtual channels?
  - Your implementation approach, challenges faced, and any observations or insights gained during the implementation and simulation process.