Machine Learning Intelligent Chip Design

Homework3　Implement NoC by SystemC

Description

NoC (Network-on-Chip) is a promising architecture that can help overcome communication bottlenecks and performance 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.

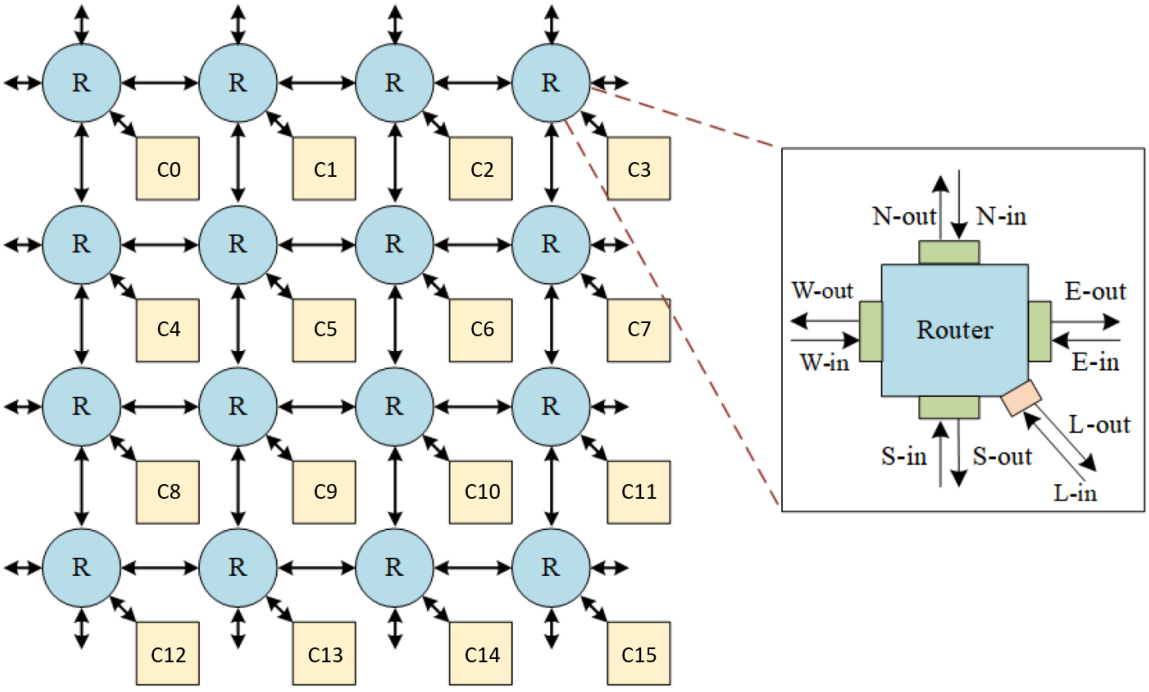

Figure 1. 4x4 mesh-based NoC architecture

To simplify the complexity of system design for this assignment, TA will provide a pre-written PE, it should be noted that **the PE cannot be modified**. The PE will be encapsulated within the core module and mainly consists of three functions:

- **void init(int pe_id)**

  You need to call this function at the beginning of the simulation. The pe_id is numbered sequentially, starting from 0 in the upper-left corner, as shown in Figure 1.

- **Packet* get_packet()**

  Each time you call this function, you can obtain a send packet. If the PE has no more packets to send, this function will return **NULL**. 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. The value 220 in the figure represents the number of execution cycles of the program and does not need to be the same as in the example.

```
01:47 mlchipTA05@ee21[~/hw3]% make
g++ -I . -I /RAID2/COURSE/2025_Spring/mlchip/mlchipTA01
inux64 -o run *.cpp -lsystemc-2.3.3 -Wl,-rpath,/RAID2/C
./run

        SystemC 2.3.3-Accellera --- Mar  4 2025 01:46:3
        Copyright (c) 1996-2018 by all Contributors,
        ALL RIGHTS RESERVED

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

Info: /OSCI/SystemC: Simulation stopped by user.
```

Figure 3. Correct simulation result

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 the size of each flit is limited to 34 bits, the packet should be decomposed before it is sent to the router.

```
SC_MODULE( Core ) {
    sc_in  < bool >  rst;
    sc_in  < bool >  clk;
    // receive
    sc_in  < sc_lv<34> > flit_rx;
    sc_in  < bool > req_rx;
    sc_out < bool > ack_rx;
    // transmit
    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.

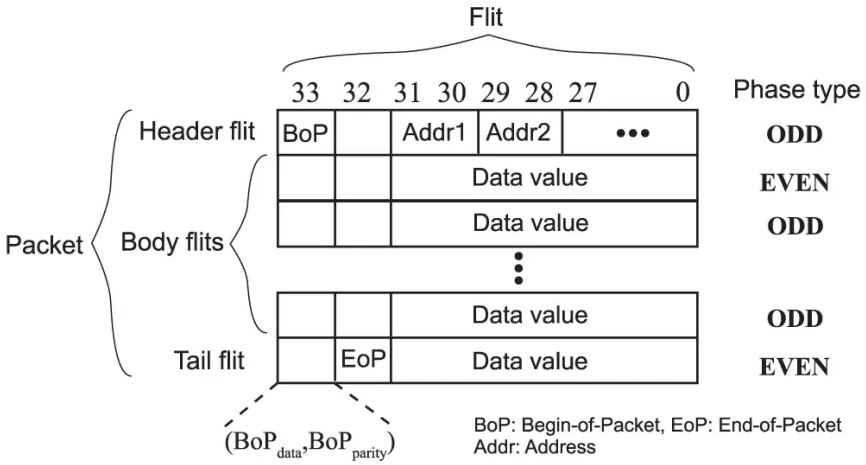

Figure 5. Example of flit format definitions

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 are not allowed to modify pattern**, and you also don't need to process them yourself, but understanding them will help you debug.

Implement Notes

A key aspect of the implementation will be the choice of routing policy employed by the routers (e.g., XY routing, west-first 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、sc_out、sc_signal 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 & Grading Policy

- **The grading breakdown for this assignment is as follows:**
  - **Report : 30%**
    - Simulation results demonstrate the predicted output for the provided input data.
    - How do you design the router and NI? What routing algorithm do you use? What is the depth of the buffer? Do you use virtual channels?
    - Your implementation approach, challenges faced, and any observations or insights gained during the implementation and simulation process.
    - If the code is not submitted, **no points will be awarded for the report!**
  - **Simulation Result : 70%**
    - You need to complete a NoC architecture similar to Figure 1. After executing the 'make' command, the terminal must display a 'Congratulations' message as shown in Figure 3. Note that the number of execution cycles does not need to be the same.
- **For the code submission, please use the compression and submission command provided by the TA.**
- **Please submit the report file to the new E3. The name of your report file is report_mlchipXXX.pdf (XXX is your account ID). If the file violates the naming rule and the file format, 10 points will be deducted.**
- **Ensure that your code is well-commented and organized for clarity and understanding.**
- **The following are rules that must be absolutely adhered to. Any violation will result in 0 points.**
  - **Plagiarism is forbidden.**
  - **You cannot modify the provided Makefile, PE (pe.cpp、pe.h), or pattern files.**
  - **Any words with “error”, “fail” or “Congratulations” can’t be used**

in your assignment.

- You cannot use any Chinese characters in the assignment, including within comments.
- Only sc_in、sc_out、sc_signal can be used for ports. Channels and interfaces that were utilized in HW2 are not allowed.
- Using pointer in port definition is forbidden.

Submission & demo command

- **Please use make commands to execute your SystemC code.**
- **Follow following command in 09_SUBMIT folder to submit your code and demo.**

1. **./00_tar**

```
07:16 mlchipTA05@ee21[~/hw3/09_SUBMIT]% ./00_tar
[Info] Directory exists. Removing old directory and creating a new one.
[Info] Directory created: hw3_mlchipTA05
[Info] Copied: ../clockreset.h -> hw3_mlchipTA05/clockreset_mlchipTA05.h
[Info] Copied: ../core.h -> hw3_mlchipTA05/core_mlchipTA05.h
[Info] Copied: ../define.h -> hw3_mlchipTA05/define_mlchipTA05.h
[Info] Copied: ../pe.h -> hw3_mlchipTA05/pe_mlchipTA05.h
[Info] Copied: ../router.h -> hw3_mlchipTA05/router_mlchipTA05.h
[Info] Copied: ../clockreset.cpp -> hw3_mlchipTA05/clockreset_mlchipTA05.cpp
[Info] Copied: ../core.cpp -> hw3_mlchipTA05/core_mlchipTA05.cpp
[Info] Copied: ../main.cpp -> hw3_mlchipTA05/main_mlchipTA05.cpp
[Info] Copied: ../pe.cpp -> hw3_mlchipTA05/pe_mlchipTA05.cpp
[Info] Copied: ../router.cpp -> hw3_mlchipTA05/router_mlchipTA05.cpp
[Info] Creating tar archive: hw3_mlchipTA05.tar.gz
hw3_mlchipTA05/
hw3_mlchipTA05/clockreset_mlchipTA05.h
hw3_mlchipTA05/core_mlchipTA05.h
hw3_mlchipTA05/define_mlchipTA05.h
hw3_mlchipTA05/pe_mlchipTA05.h
hw3_mlchipTA05/router_mlchipTA05.h
hw3_mlchipTA05/clockreset_mlchipTA05.cpp
hw3_mlchipTA05/core_mlchipTA05.cpp
hw3_mlchipTA05/main_mlchipTA05.cpp
hw3_mlchipTA05/pe_mlchipTA05.cpp
hw3_mlchipTA05/router_mlchipTA05.cpp
[Success] hw3_mlchipTA05.tar.gz created successfully.
```

2. **./01_submit**

- **Simulation examples of correct and incorrect results are shown below.**

```
07:16 mlchipTA05@ee21[~/hw3/09_SUBMIT]% ./01_submit
[Info] Deadline check OK ...
[Info] File check OK ...
[Info] Copying TA's Makefile to your folder.
[Info] mlchipTA05 SystemC start
[Info] Checking source code for channels and interfaces...
[Info] Starting compilation...
[Info] Match Golden Result


Server_Account  mlchipTA05
Demo_Result     PASS
Error_Message   No_Error
Submiss_Date    2025/04/26
Submiss_Time    19:19:47
Sim_Time (s)    2.62


[Info] Your file will be submitted to:  TA folder
[Warning]  demo has been submitted.
[Warning]  It will overwrite your original file.
[Info] Now submit hw3_mlchipTA05.tar.gz file to system.
[Success] Copying Sucessfully.
============================================================
                    Submit Report
============================================================
Result          :   has been submitted.
Submission time :  2025/04/26 19:19:47
============================================================
        ----------------------------
        --                        --       |\__||
        --  Congratulations !!    --      / 0.0  |
        --                        --    /_____   |
        --  Submission Sucessful!!--   /^ ^ ^ \  |
        --                        --  |^ ^ ^ ^ |w|
        ----------------------------   \m___m__|_|
   Please remember to check your submission with ./02_check !!
   Please remember to check your submission with ./02_check !!
   Please remember to check your submission with ./02_check !!
============================================================
```

Correct example

```
07:31 mlchipTA05@ee21[~/hw3/09_SUBMIT]% ./01_submit
[Info] Deadline check OK ...
[Info] File check OK ...
[Info] Copying TA's Makefile to your folder.
[Info] mlchipTA05 SystemC start
[Info] Checking source code for channels and interfaces...
[Info] Starting compilation...
[Error] Make failed: Command 'make' returned non-zero exit status 2.


Server_Account  mlchipTA05
Demo_Result     FAIL
Error_Message   Compile Error
Submiss_Date    2025/04/26
Submiss_Time    19:34:48
Sim_Time (s)    2.19


[Info] Your file will be submitted to:  TA folder
[Warning]  demo has been submitted.
[Warning]  It will overwrite your original file.
[Info] Now submit hw3_mlchipTA05.tar.gz file to system.
[Success] Copying Sucessfully.
============================================================
                    Submit Report
============================================================
Result          :   has been submitted.
Submission time :  2025/04/26 19:34:48
============================================================
        ----------------------------
        --                        --       |\__||
        --  Congratulations !!    --      / 0.0  |
        --                        --    /_____   |
        --  Submission Sucessful!!--   /^ ^ ^ \  |
        --                        --  |^ ^ ^ ^ |w|
        ----------------------------   \m___m__|_|
   Please remember to check your submission with ./02_check !!
   Please remember to check your submission with ./02_check !!
   Please remember to check your submission with ./02_check !!
============================================================
```

Incorrect example

3. **./02_check**

```
07:16 mlchipTA05@ee21[~/hw3/09_SUBMIT]% ./02_check
 hw3_mlchipTA05.tar.gz  has been downloaded!
 demo_result_hw3_mlchipTA05.csv  has been downloaded!
Server_Account,Demo_Result,Error_Message,Submiss_Date,Submiss_Time,Sim_Time (s)
mlchipTA05,PASS,No_Error,2025/04/26,19:19:47,2.62
```