MLCHIP HW3

Implement NoC by SystemC

Implementation

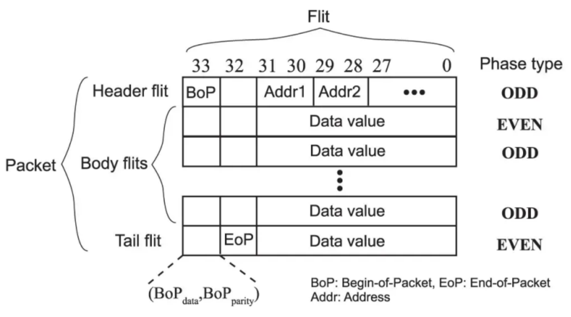
Architecture	4x4 mesh-based NoC architecture
Routing algorithm	XY routing (deadlock free)
Switching	Wormhole switching
Buffer size	one flit (34 bits), depth=1
Virtual channel	No implementation
Flow control	ack/nack protocol (buffered flow control)

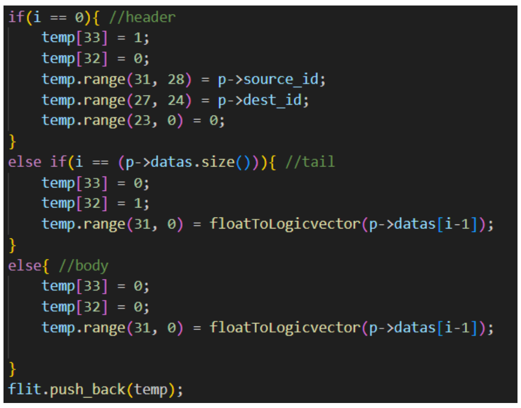
Core Design (NI)

1. Transmit

First, to ensure get_packet is called when the tx channel is free, which means that the previous packet has been transmitted or it is the begin of the program to get the first packet, check if the pointer is nullptr before calling get_packet and reset the pointer to nullptr when the tail flit has been transmitted.

If a packet is returned, then write it to the flit vector. The flit format is described in Fig. 1. BoP is set 1 for header and EoP is set 1 for tail.





```
if(i == 0){ //header
    temp[33] = 1;
    temp[32] = 0;
    temp.range(31, 28) = p->source_id;
    temp.range(27, 24) = p->dest_id;
    temp.range(23, 0) = 0;
}
else if(i == (p->datas.size())){ //tail
    temp[33] = 0;
    temp[32] = 1;
    temp.range(31, 0) = floatToLogicvector(p->datas[i-1]);
}
else{ //body
    temp[33] = 0;
    temp[32] = 0;
    temp.range(31, 0) = floatToLogicvector(p->datas[i-1]);

}
flit.push_back(temp);
```

Fig. 1: Flit format definitions

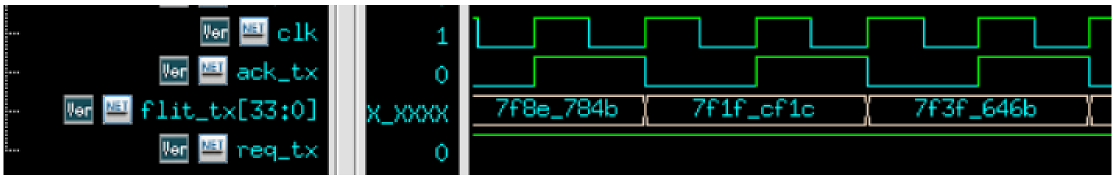


Fig. 2: Transmitting flits

2. Receive

Receive the flit and put it into Packet when req_rx and ack_rx are both high. Note that ack_rx cannot be always 1 but must lower down for one cycle before next handshake.

Once the tail flit is received, call check_packet to check the answer. The congratulation message will be printed only when all cores received their packets. The error message will be printed once the received packet is wrong, and the simulation will stop immediately.

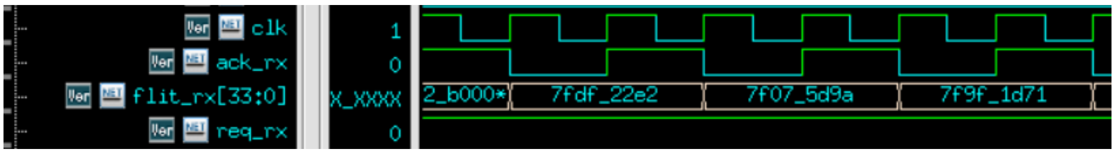


Fig. 3: Receiving flits

Router Design

For each input channel, if in_req arrives and the buffer is available, read the in_flit into the buffer and set the buffer as full.

If the flit inside the buffer is header, calculate the output direction by comparing the destination id and the router id. Since the NoC topology is Torus and routing algorithm is XY routing, the horizontal directions will be chosen first, followed by vertical directions. The crossbar should choose the shortest path to go to the destination.

When there are more than one input buffer intending to go to the same direction, the priority is fixed to NORTH > SOUTH > WEST > EAST > LOCAL. Only the one who gets the occupation of output channel can output. The channel is occupied during the transmitting of a packet, from header to tail.

When in_ack arrives, the buffer should be clear to become available again to next input. The out_req will become low for one cycle because it needs time for buffer to grab a new value even though the input is continuous.

The simulation completes at 400th cycle from the design described above using pattern given by TA.

Optimization

I slightly change the order of the code in router to make out_req continuous if it is possible. That is, the output will only stop when there is no flit in input buffer. The final version completes the simulation at 275th cycles.

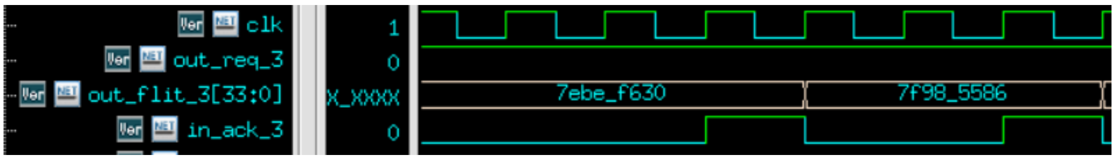


Fig. 4: Continuous out_req, wait until handshake with in_ack

Main function

1. Pointer and Connection

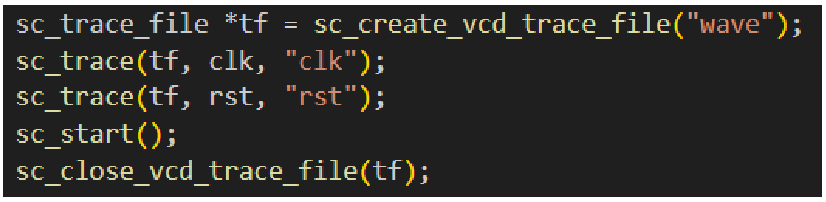
   Use pointer to create Core and Router in a concise style.

   The connections between routers should be taken carefully. I connected them from the view of output ports:

   signal_of_router[direction][output_router_id]

2. Dump Waveform

   By adding the following lines in the main function, the wave.vcd file is dumped. Type yes when logging to the server and open nWave, the file can be transferred to fsdb file and we can trace the signals in a visual style, which is more convenient to debug. I also separate the sc_trace into Router and Core modules so that it is easier to trace signals in each module.

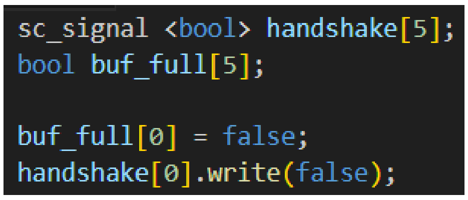


```
sc_trace_file *tf = sc_create_vcd_trace_file("wave");
sc_trace(tf, clk, "clk");
sc_trace(tf, rst, "rst");
sc_start();
sc_close_vcd_trace_file(tf);
```

Fig. 5: Dump the trace file

Observation

1. The initialization of PE cannot be called at the reset stage because rst is high for two cycles and PE will be initialized two times.
2. Since I need to send id to initialize and send tf to dump the waveform in each core and router, it is suggested to use SC_HAS_PROCESS instead of SC_CTOR to define the argument of the constructor by myself.
3. The C++ type variables change their values when they are assigned immediately, while sc_signal and sc_buffer change their values at the next triggered time. Variables can be assigned using the = operator, but sc_ type needs write() function to assign values. Also, sc_ type's value should be read by read() function.
4. We can declare a vector (length: 5) by adding [5] after its name.



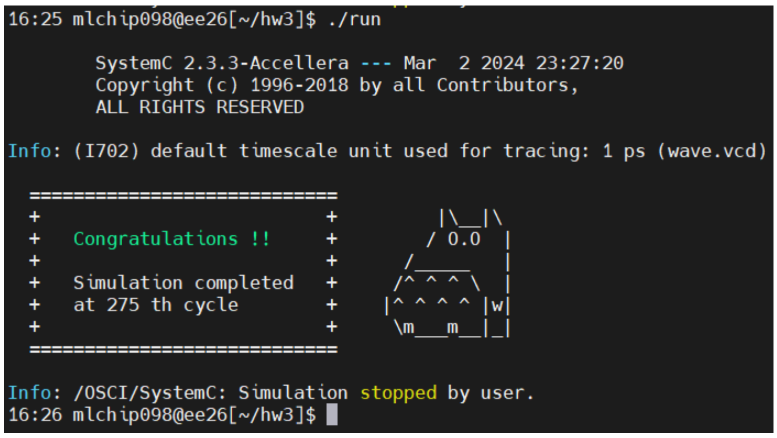
```
sc_signal <bool> handshake[5];
bool buf_full[5];

buf_full[0] = false;
handshake[0].write(false);
```

Fig. 6: Example for observation 2 and 3

5. The program is executed from top to bottom in the void function. Therefore, it is important to take care of value of every variable line by line.
6. Use the built-in functions of sc_dt::scfx_ieee_float datatype to transfer between float and sc_lv.

Result



```
16:25 mlchip098@ee26[~/hw3]$ ./run

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

Info: (I702) default timescale unit used for tracing: 1 ps (wave.vcd)

  ============================
  +                          +         |\__|\
  +   Congratulations !!     +        / 0.0  |
  +                          +      /_____   |
  +   Simulation completed   +     /^ ^ ^ \  |
  +   at 275 th cycle        +    |^ ^ ^ ^ |w|
  +                          +     \m___m__|_|
  ============================

Info: /OSCI/SystemC: Simulation stopped by user.
16:26 mlchip098@ee26[~/hw3]$ 
```

Fig. 7: Simulation result