SystemC Project Part I
System Setup for Network-on-Chip Prototyping

ECE 587, Spring 2016

Student: Yanbo Deng

Student ID: A20337966

Introduction

The goal of this project is to setup the simulation for Fast Fourier Transform (FFT) in SystemC and to understand the behaviors of NoC routers and processing elements (PEs).

In this project, the basic structure of the network has been given. There are two main requirements for the project: 1) to assemble the components, i.e. PEs and interconnects into the 2-D NoC architecture as a 3×3 mesh NoC prototype and complete the XY routing algorithm; 2) to implement the FFT computation on the NoC in the previous step and to implement the communication channels over PE-to-PE transports supported by the NoC architecture. After those, in the project, I will show the maximum throughput and the minimum latency as the performance evaluation.

System Description and Implementation Details

In this section, I will show you the architecture of the 3×3 mesh NoC prototype and introduce some implementation details of this architecture. And how to map this architecture to an 8-bit FFT computation design.

System Description

First of all, I need to build a 3×3 network of routers and each router is connected to one processing element (PE). As shown in Figure 1, we could find that each router has a particular location (x, y) as well as the PE it is connected to and routers only have horizontal and vertical connections which connect all routers together.

Then I will show you the routing algorithm for this network. The XY routing algorithm follows the threes rules below:

- If the destination is the attached PE, then the packet should be delivered to it.
- If the y coordinate of the destination is the same as the current router, then the packet is delivered to the router either on the west or on the east depending on the x coordinate.
- If the y coordinate of the destination is different from that of the current router, then the packet is delivered to the router on the north or on the south depending on the y coordinate.

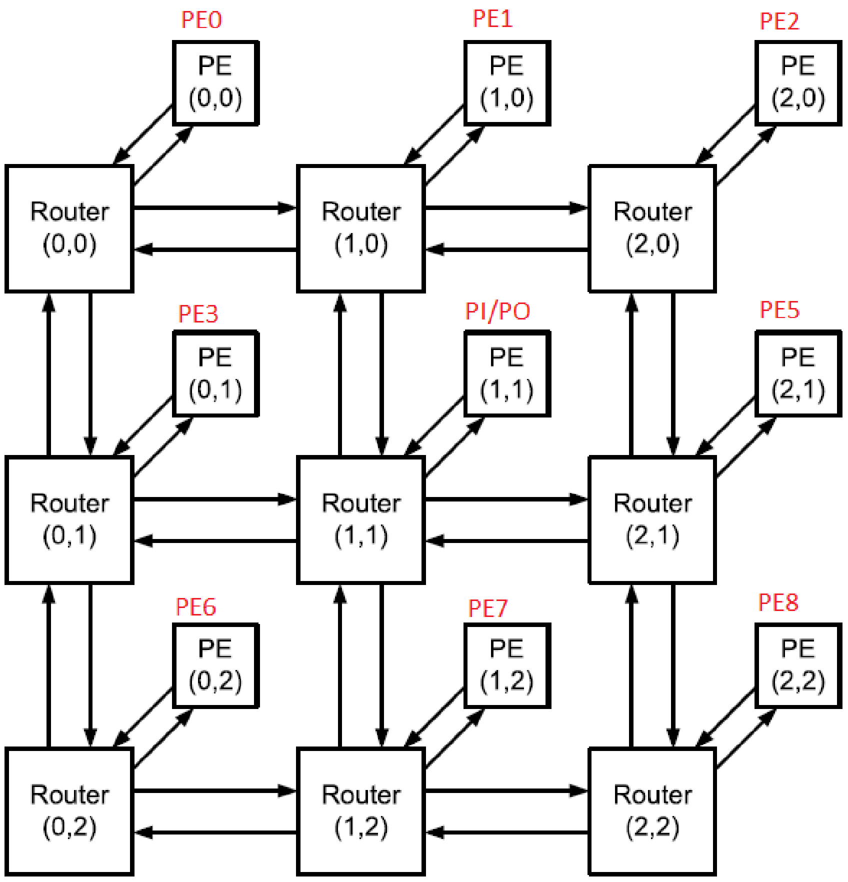


Figure 1: 2-D Mesh NoC Architecture

Then we have to map FFT computation architecture into this 3×3 network. Since an 8-bit FFT computation need 12 complex computations and one input and one output, these 9 PEs in our network have to be multiple used. Here, to simplifier the design, I decide that each PE should have certain output destinations, i.e. even if a PE is multiple used, the output destination is still the same, which is shown in Figure 2. We could find that in Figure 2, PE1 is transmitting its outputs to PE3 and PE5 even if it has different kind of inputs.

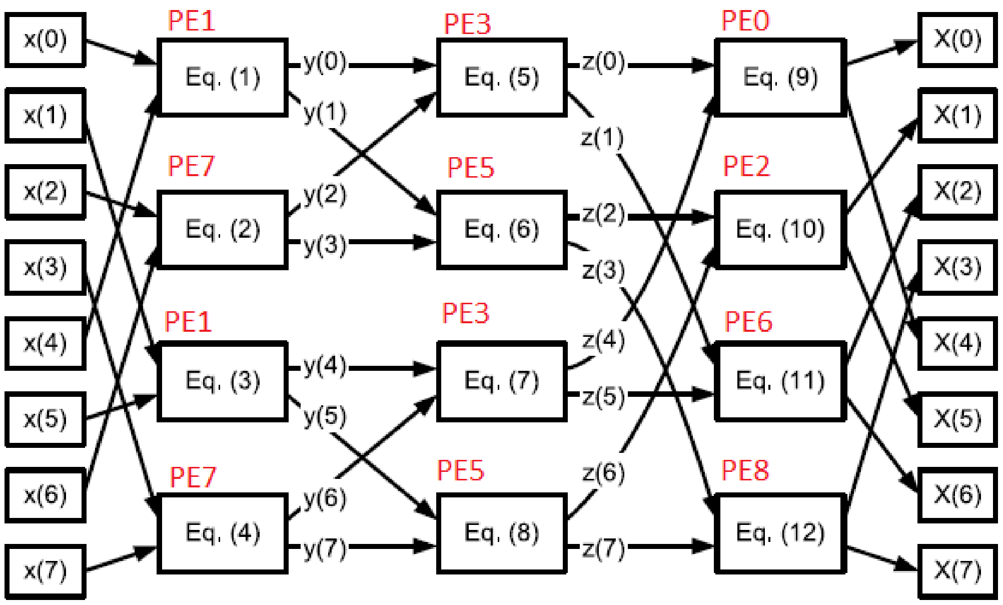


Figure 2: DFG for FFT Computation

Implementation Details

To support my design that all PEs has certain output destinations, we have to divide the 8-bit FFT computations into four 4-bit FFT computation which is so called “butterfly” computation. For example, Eq.(1), Eq.(2) and Eq.(5), Eq.(6) is a butterfly computation. I choose to multiply use PE1, PE7, PE3 and PE5 as it is shown in Figure 2. For better performance and avoiding packets collision, I reordered the inputs into x(0), x(2), x(4), x(6), x(1), x(3), x(5) and x(7) so that each computation could get all of its input as soon as possible.

The communication between each two PEs is built on the packet. The packet has all the information that other PEs could identify and use to do the computation. In my design, the packet contains these four things below:

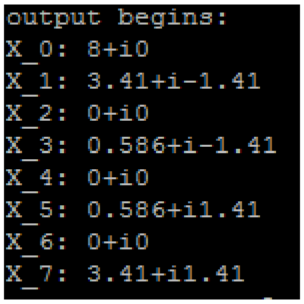
1. Source PE’s location (x, y)
2. Destination PE’s location (x’, y’)
3. A complex number
4. Name of the complex number, i.e. x(0)

Results and Analysis

The inputs I set is [2, 2, 1, 1, 0, 0, 1, 1], and the right output should be

$$[8, 3.41 - i1.41, 0, 0.59 - i1.41, 0, 0.59 + i1.41, 0, 3.41 + i1.41]$$

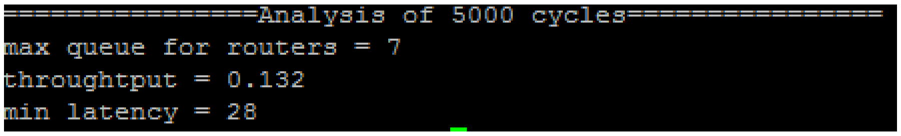
My output is:



```
output begins:
X_0: 8+i0
X_1: 3.41+i-1.41
X_2: 0+i0
X_3: 0.586+i-1.41
X_4: 0+i0
X_5: 0.586+i1.41
X_6: 0+i0
X_7: 3.41+i1.41
```

And it is theoretical correct.

For the performance evaluation, the analysis is shown below:



```
================Analysis of 5000 cycles================
max queue for routers = 7
throughtput = 0.132
min latency = 28
```

As we can see, the maximum queue size in all routers is 7 after 5000 cycles, which means the queue size is infinite and there is no “traffic jam” in my design.

The throughput after 5000 cycles is 0.132 which should be acceptable in this NoC architecture; and the minimum latency is 28.

Acknowledgements

Here I give my great thanks to professor Jia Wang who give me a chance to learn NoC design. Thanks to his kindness, my work could be done after overcoming some troubles.