Worcester Polytechnic Institute

ECE 5723: Methodologies for System Level Design and Modeling

SystemC Channels

by

Vladimir Vakhter

Laboratory Report

11/08/2020

Table of Contents

1. Task

In this problem, you are to design a *multiway shared switch*, in which four different targets can be targeted at any one time. The same target should be allowed to be addressed by two initiators at the same time.

2. Analysis and Assumptions

- Four initiators can each send data to their own targets at the same time.
- The target must allow two simultaneous communications. For example, when three initiators send data to the same target (three simultaneous communications), two initiators can simultaneously send data to that target. But the third initiator must wait until the next time the target is available.
- In this solution, the multiway shared switch is implemented as a blocking templated channel. It means that an initiator waits until one of four buses is given to it. When the bus is given, it waits until the transfer is completed. The target waits until the data is available for it. Once the data is available, it notifies the channel that the data is received from a specific initiator.
- All the initiators have the same priority.

2. SystemC Description

A description of the channel (chapter 1) was written in SystemC.

3. Testing

In this chapter, the results of the verification of the designed channel are provided for four scenarios.

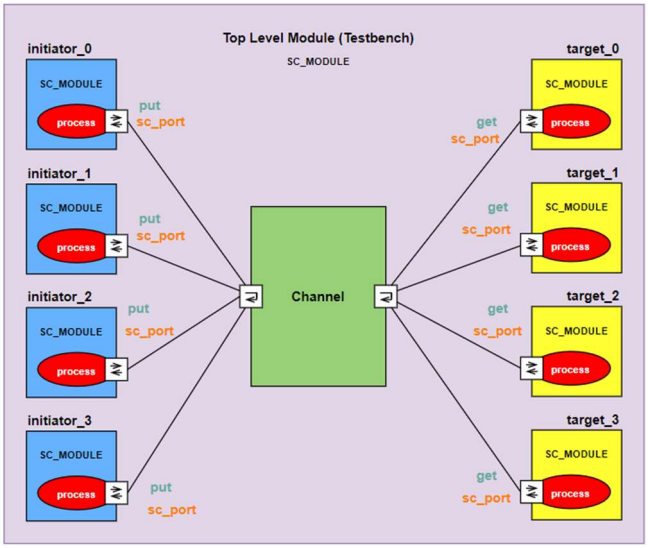

Fig. 1 – Testbench

The project contains two header files with testbenches. To select one of the testbenches, one should uncomment a corresponding line in *main.cpp*.

Scenario 1 (testbench 1)

Initiators 0, 1, 2, and 3 send data to target 1. The data is received only from 2 initiators in one communication (at 4 ns). The rest initiators will transmit the data the next time (at 8 ns) when the target is available.

```
Initiator {0} intends to transmit (00000000) at: 3 ns to: [1]

Initiator {1} intends to transmit (00010000) at: 3 ns to: [1]

Initiator {3} intends to transmit (00110000) at: 3 ns to: [1]

Initiator {2} intends to transmit (00100000) at: 3 ns to: [1]
Target [1] ready to receive something at: 4 ns
Target [1] received (00000000) at: 4 ns from: {0}
Target [1] received (00010000) at: 4 ns from: {1}
Initiator {0} completed transmitting (00000000) at: 4 ns to: [1]
Initiator {1} completed transmitting (00010000) at: 4 ns to: [1]
Target [3] ready to receive something at: 5 ns
Target [2] ready to receive something at: 6 ns

Initiator {0} intends to transmit (00000001) at: 7 ns to: [3]

Initiator {1} intends to transmit (00010001) at: 7 ns to: [3]
Target [0] ready to receive something at: 7 ns
Target [3] received (00000001) at: 7 ns from: {0}
Target [3] received (00010001) at: 7 ns from: {1}
Initiator {0} completed transmitting (00000001) at: 7 ns to: [3]
Initiator {1} completed transmitting (00010001) at: 7 ns to: [3]
Target [1] ready to receive something at: 8 ns
Target [1] received (00110000) at: 8 ns from: {3}
Target [1] received (00100000) at: 8 ns from: {2}
Initiator {3} completed transmitting (00110000) at: 8 ns to: [1]
Initiator {2} completed transmitting (00100000) at: 8 ns to: [1]
```

Scenario 2 (testbench 1)

Initiators 3 and 2 send data to target 2, initiators 0 and 1 send data to target 1. These data is received when the appropriate targets are available.

```
Initiator {1} intends to transmit (00010011) at: 13 ns to: [0]

Initiator {0} intends to transmit (00000011) at: 13 ns to: [0]
Target [0] received (00010011) at: 13 ns from: {1}
Target [0] received (00000011) at: 13 ns from: {0}
Initiator {1} completed transmitting (00010011) at: 13 ns to: [0]
Initiator {0} completed transmitting (00000011) at: 13 ns to: [0]

Initiator {3} intends to transmit (00110010) at: 15 ns to: [2]

Initiator {2} intends to transmit (00100010) at: 15 ns to: [2]

Initiator {1} intends to transmit (00010100) at: 16 ns to: [1]
Target [2] ready to receive something at: 16 ns
Target [2] received (00110010) at: 16 ns from: {3}
Target [2] received (00100010) at: 16 ns from: {2}

Initiator {0} intends to transmit (00000100) at: 16 ns to: [1]
Target [1] received (00010100) at: 16 ns from: {1}
Target [1] received (00000100) at: 16 ns from: {0}
Initiator {3} completed transmitting (00110010) at: 16 ns to: [2]
Initiator {2} completed transmitting (00100010) at: 16 ns to: [2]
Initiator {1} completed transmitting (00010100) at: 16 ns to: [1]
Initiator {0} completed transmitting (00000100) at: 16 ns to: [1]
```

Scenario 3 (testbench 1)

Initiators 0, 1, and 2 send data to targets 1, 0, and 2, accordingly. Each target receives the corresponding data when it is ready for communication.

```
Initiator {2} intends to transmit (00101000) at: 56 ns to: [2]

Initiator {1} intends to transmit (00011001) at: 56 ns to: [0]

Initiator {0} intends to transmit (00001011) at: 56 ns to: [1]
Target [0] ready to receive something at: 57 ns
Target [0] received (00011001) at: 57 ns from: {1}
Target [1] ready to receive something at: 57 ns
Target [1] received (00001011) at: 57 ns from: {0}
Initiator {1} completed transmitting (00011001) at: 57 ns to: [0]
Initiator {0} completed transmitting (00001011) at: 57 ns to: [1]
Target [2] ready to receive something at: 59 ns
Target [2] received (00101000) at: 59 ns from: {2}
Target [2] received (00111000) at: 59 ns from: {3}
Initiator {2} completed transmitting (00101000) at: 59 ns to: [2]
Initiator {3} completed transmitting (00111000) at: 59 ns to: [2]
```

Scenario 4 (testbench 2)

Initiator 2 sends data to target 0, initiators 0, 1, and 3 send data to target 2. Target 0 receives the data when it is ready, target 2 receives 2 out of 3 available data packages in one communication.

```
Initiator {2} intends to transmit (00100101) at: 35 ns to: [0]

Initiator {3} intends to transmit (00110110) at: 37 ns to: [2]

Initiator {1} intends to transmit (00010111) at: 37 ns to: [2]

Initiator {0} intends to transmit (00001000) at: 37 ns to: [2]
Target [2] ready to receive something at: 40 ns
Target [2] received (00110110) at: 40 ns from: {3}
Target [2] received (00010111) at: 40 ns from: {1}
Initiator {3} completed transmitting (00110110) at: 40 ns to: [2]
Initiator {1} completed transmitting (00010111) at: 40 ns to: [2]
Target [0] ready to receive something at: 41 ns
Target [0] received (00100101) at: 41 ns from: {2}
Initiator {2} completed transmitting (00100101) at: 41 ns to: [0]
```

4. Conclusions

In this work, a multi-way shared switch was designed and tested in SystemC.