System Level Languages
IL2452

Lab 3

Transaction Level Modeling

Name:_____

Personal Number:_____

Approval

Date:_____

Sign:_____

1. General Information

- For this lab, you need to install the TLM package on your computers. You can download the archive file "TLM-1.0.tar.gz" from the course web-page. Alternatively, you can download it from the website www.systemc.org, together with much more material. After extracting this archive, you will get the required header files, help documents, as well as a group of examples.

  Note: for this lab, it is important that you have a look and heavily use the code provided in the TLM examples!

- Each Lab event has been thought of as consisting of 3 sections:

  - The first section where students have to accomplish a set of *preparation tasks*. Such tasks should be solved before the actual lab session starts, otherwise there may not be enough time to complete the lab tasks. Their main purpose is to put the students in the condition of being able to solve the lab tasks.
  - The second section where students will be in the lab and will solve the actual lab tasks.
  - The third section will be a discussion with the lab assistant about both the preparation and the lab tasks. To get a Pass, students will have to justify their answers and show that they have understood.

2. Introduction to lab 3

Prerequisites: before starting working on Lab 3, students should have gone through the material shown during Lectures. It will be also useful for you to consult the other materials placed on the course web-page.

Goal: the main goal of Lab 3 is introduce the Transaction Level Modeling using SystemC, and to show how it can be used for design exploration.

Skills acquired: at the end of Lab 3, student will be able to model simple systems at the transaction level and to perform basic design exploration.

3. Preparation tasks (to be done before the actual lab session!)

For this lab, you have add the required include directories to your project as the following:

- Follow the instructions for adjusting your project properties in the "INSTALL" file for SystemC as you done for lab1 and lab2.
1)Select Tools -> Options . . . and the Projects -> VC++ Directories tab
2)Select show directories for: Include files
3)Select the 'New' icon and browse to: ~\TLM-2005-04-08\tlm
4) Repeat these steps in oreder to add the directories "~\TLM-2005-04-08\utils", "~\TLM-2005-04-08\examples\basic_protocol" and "~\TLM-2005-04-08\examples\user".

Task 3.1

1) What is *sc_export*, how is it used? And why?
2) What is the difference between *sc_port* and *sc_export*?
3) What is an *initiator_port*, and how is it used?
4) How is the transaction from initiator to target called?
5) How is the transaction from target to initiator called?
6) What is the main focus of TLM?

In this task, implement a small system composed of a master module and a slave module, as shown in Figure1. The master module has an initiator port and it uses this port to write to/read from the slave module. The slave module is a 4KB memory and it has a target port (export) used to communicate with the master. The master module has a thread that writes the first 150 locations of the memory with the values “50-199”, then it reads the first 150 locations of the memory and prints the results on the screen. The slave module (memory) does not have threads, but it has to count the number of write transactions and read transactions and to show them by the end of simulation.
Hint: study example_3_2 in the TLM examples folder since it represents a similar system; Also, write a member function in the slave module that is responsible to show (print out) the transactions out by the end of simulation.

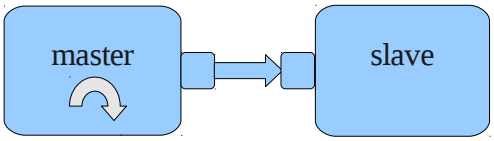

Figure 1

Task 3.2

1) What is the difference between unidirectional and bidirectional interface?
2) Is the “tlm_transport_if” interface unidirectional or bidirectional?
3) What is the difference between blocking and non-blocking interface? And how does each of them deal with request and response?
4) Which kind of interfaces may fail? The blocking or non-blocking ones?
5) Is the “tlm_transport_if” interface blocking or non-blocking?
6) Is there any relation between “tlm_transport_if” and initiator port? If yes, what is it?

In this task, implement a system composed of four modules: master, switch, slave_odd and slave_even. As shown in Figure 2, the master's initiator port is connected to the switch's target port. The switch has two initiator ports, one of them is connected to the target port of the slave_odd, and the other is connected to the target port of the slave_even. Only the master module has a thread, this thread is writing to the switch the values ”200-300” in the addresses “0-100”, then it reads the addresses “0-100” from the switch and prints them on the screen. The duty of the switch is to check addresses in the write/read coming from the master. If the address is odd, the operation will be passed to slave_odd, otherwise the operation will be passed to slave_even. Note that slave_odd and slave_even are 4KB memories and they look like the slave module used in Task 3.1. Each module except the master has to count the number of write transactions and read transactions that it performed and has to show this count by the end of simulation.

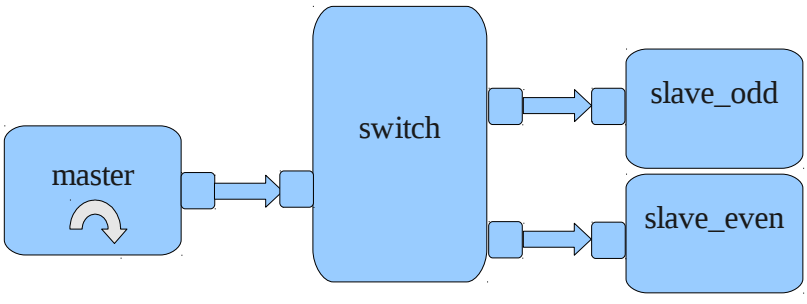

Figure 2

Task 3.3

In this task, implement a master module (master_new) that can be used in place of the master module of Task 3.2. The only difference between master_new and master used in Task 3.2 is that the thread of master_new is writing the a random value “1-20” to the addresses “11,15,12,3,33,35,21,18,55,41”. Then this thread will read the memory content at these addresses and calculate the average. This process has to be repeated 5 times.

Task 3.4

In this task, implement a system composed of four modules: master1, master2, master3 and memory (slave). Master1 is writing the value “1” in the first 20 locations of the memory. Master2 is writing the value ”2” in the first 20 locations of the memory. Master3 is reading the first 20 locations of the memory and printing them on the screen. The memory used in this system accepts a single read or a single write at a time, keeping in mind that the three masters have to work concurrently. See Figure 3.

- Fill in table 4 the expected values that are read by master3.

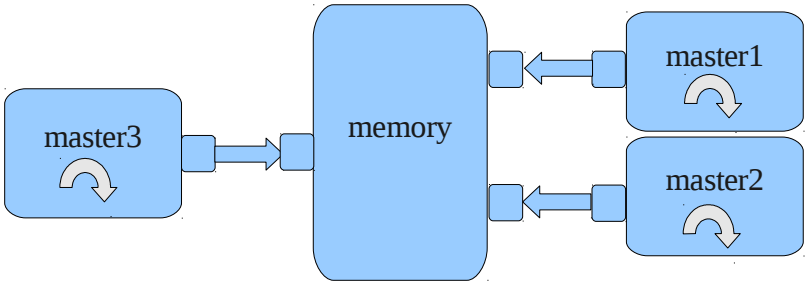

Figure 3

Task 3.5

In this task, implement a TLM for a system composed of 5 modules: node1, node2, node3, node4 and switch. Each of these nodes is able to transmit/receive data to/from other nodes. Each node should have sufficient storage so that it can store the data it receives from other nodes. Nodes are working concurrently and have the addresses 0,1,2 and 3 which are used when nodes transmit data to each other through the switch. Each node should use its corresponding file to get the data and the destinations of each transmission. These files are given and named as node0.txt, node1.txt, node2.txt and node3.txt. This system is shown in Figure 4. Each node should be able to count its receive and transmit transactions and should have a function to display the data that it received by the end of simulation.

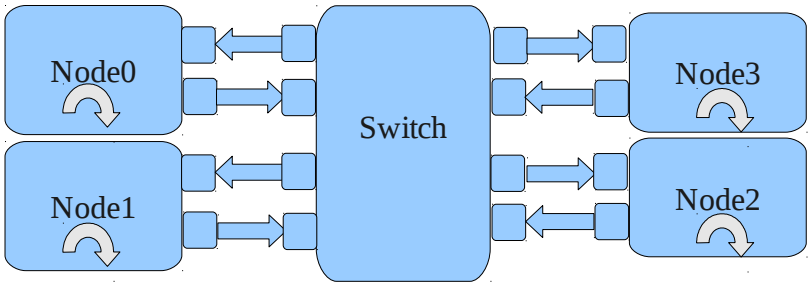

Figure 4

4. Lab tasks

Task 4.1

- Run a Simulation for the model implemented in Task 3.1. Does it work correctly? Fill in Table 1.

Slave Transactions	
Read Transactions	
Write Transactions	

Table 1

Task 4.2

- Run a Simulation for the model implemented in Task 3.2. Does it work correctly? Fill in Table 2.

	Switch Transactions	***Slave_odd Transactions***	***Slave_even Transactions***
Read Transactions			
Write Transactions			

Table2

Task 4.3

- For the system used in Task 3.2, replace the master module with the module “master_new”.
- Run the simulation. Does it work correctly? Fill in Table 3.

	Switch Transactions	***Slave_odd Transactions***	***Slave_even Transactions***
Read Transactions			
Write Transactions			

Table 3

Task 4.4

- Run a Simulation for the model implemented in Task 3.4. Does it work correctly? Fill in Tables 4 and 5.

Location	***Expected Results***	***Simulation Results***
Reading Location 0		
Reading Location 1		
Reading Location 2		
Reading Location 3		
Reading Location 4		
Reading Location 5		
Reading Location 6		
Reading Location 7		
Reading Location 8		
Reading Location 9		
Reading Location 10		
Reading Location 11		
Reading Location 12		
Reading Location 13		
Reading Location 14		
Reading Location 15		
Reading Location 16		
Reading Location 17		
Reading Location 18		
Reading Location 19		

Table 4

Memory Transactions	
Read Transactions	
Write Transactions	

Table 5

- Do the simulation results match the expected results for memory reading? Why?
- If you said that this system has a problem, what do you suggest in order to solve it? Describe in detail.

Task 4.5

- Run a Simulation for the model implemented in Task 3.5. Does it work correctly? Fill in Table 6.

	Node0	***Node1***	***Node2***	***Node3***
Transmit Transactions				
Receive Transactions				

Table 6