System C

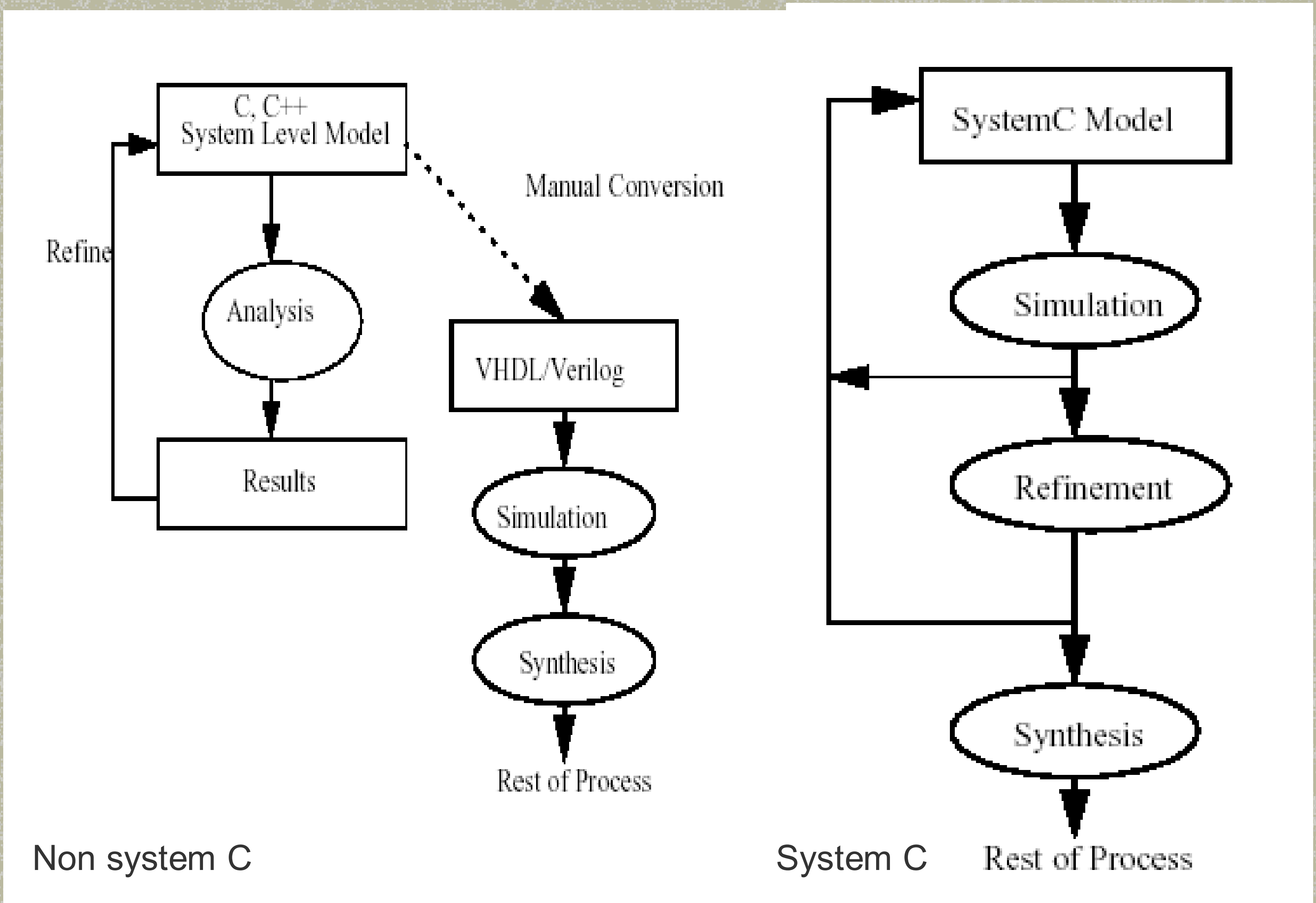
ECE 652

Tutorial by

Ashwin Balakrishnan

SystemC – A brief note

- SystemC is based on C++ programming language.
- C++ is an extensible object oriented modeling language.SystemC extends the capabilities of C++ by enabling modeling of hardware descriptions
- SystemC adds important concepts of C++ as concurrency,timed events and data types.
- SystemC describes System level design,Software algorithms and Hardware architecture.



SystemC vs. Non-SystemC

Basic Definitions

- **Modules**

  SystemC has a notion of a container class called a module.This is a hierarchical entity that can have other modules or processes contained in it.

- **Processes**

  Processes are used to describe functionality.They are contained inside modules.

- **Ports**

  Modules have ports through which they connect to other modules.SystemC supports single-direction and bi-directional ports.

- **Signals**

  SystemC supports resolved and unresolved signals.Resolved signals can have more than one driver(a bus) while unresolved signals can have only one driver.

- **Data Types**

  SystemC has a rich set of data types to support multiple design domains and abstraction levels.

Module

Modules are the basic building block within

SystemC to partition a design.

- allow designers to break complex systems into smaller more manageable parts.

- allow designers to hide internal data representation and algorithms from other modules and the entire system becomes easier to change and maintain.

- Modules are declared with the SystemC keyword SC_MODULE as shown in the example
- SC_MODULE(transmit) {

- The identifier after the SC_MODULE keyword is the name of the module, which is transmit in this example.

- A module can contain a number of other elements such as ports, local signals,local data ,other modules ,processes and constructors.These elements implement the required functionality of the module

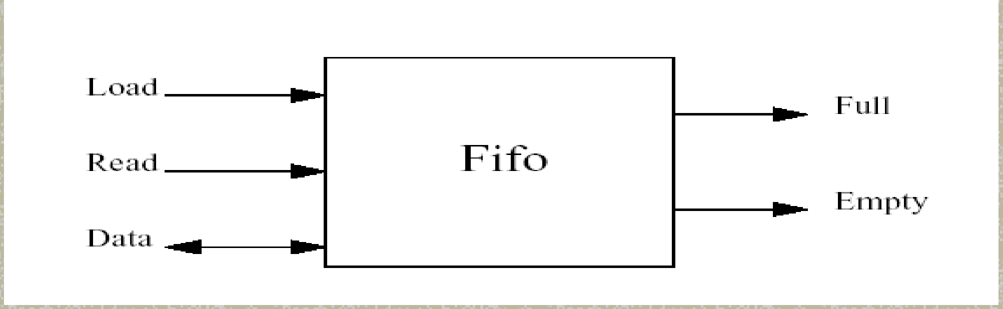
Ports

- Ports of a module are the external interface that pass information to and from a module, and trigger actions within the module.

- A port can have three different modes of operation:* Input * Output * InOut

- A port can be declared using one of the sc_in,sc_out or sc_inout declaration.

- The declarations are of the form:
  - sc_in<type> input_name1,input_name2
  - sc_out<type> output_name1,output_name2
  - sc_inout<type> inout_name1,inout_name2

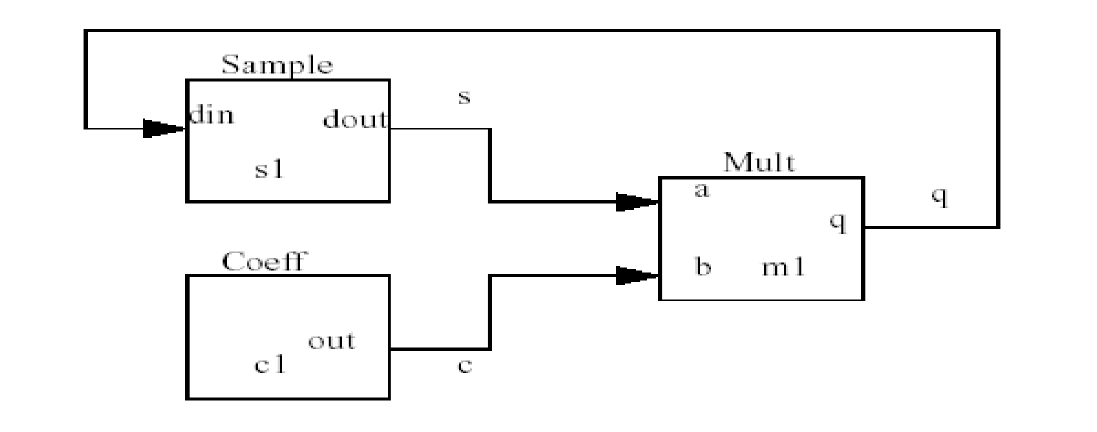


- The figure shows a fifo module with a number of ports.The ports on the left are the input ports or inout ports while the ports on the right are output ports.
- Each port has an identifying name.Each port on the block diagram has a matching port statement in the SystemC description.Port modules *sc_in* ,*sc_out* and *sc_inout* are predefined by the SystemC class library.

Signal

- Signals are used for interprocess communication and for connecting module instances.
- Signals can be local to a module, and are used to connect ports of lower level modules together represent the physical wires that interconnect devices on the physical implementation of the design.
- A signal is declared using the sc_signal declaration.The declaration is of the form

  sc_signal<type> signal_name1,signal_name2

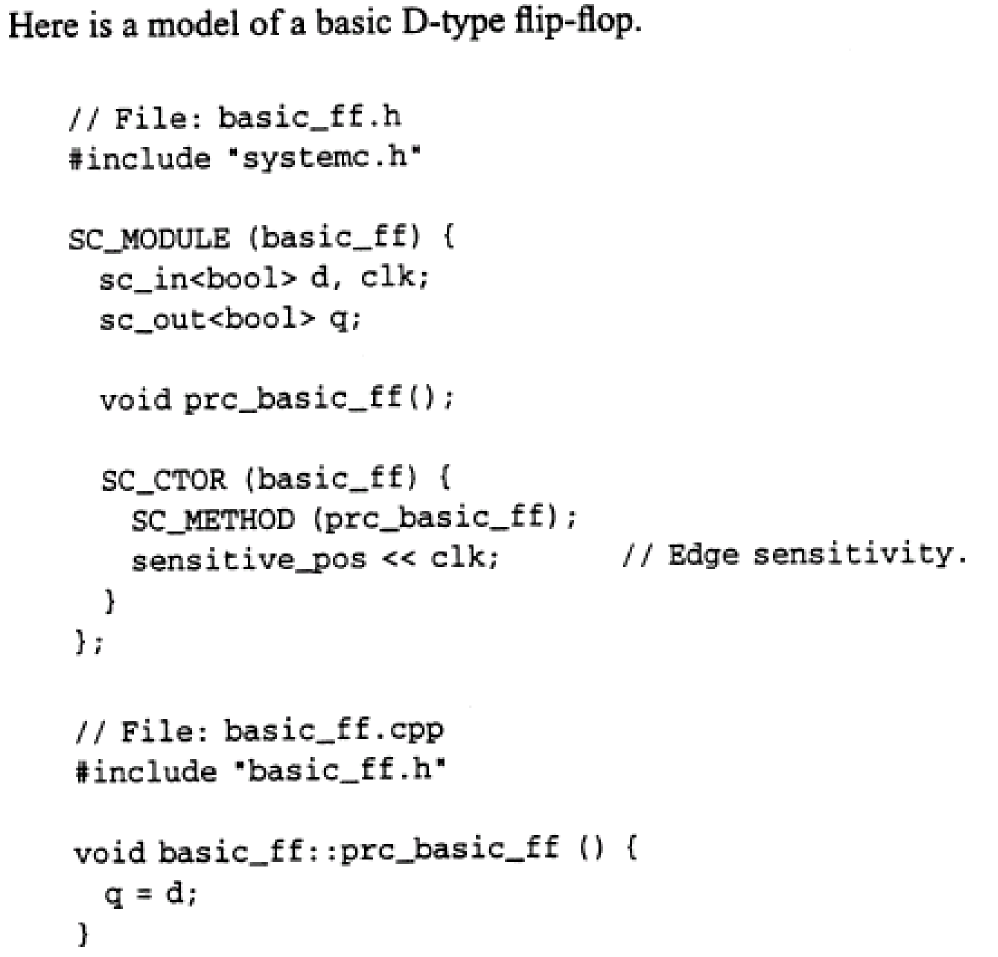


- The example shows the data path of a simple filter design.
- There are three lower level modules instantiated in the filter design, sample,coeff and mult modules.
- The module ports are connected by local signals q,s and c

Tutorial

- Tutorial describes how to model synchronous logic using SystemC.
- We take an example of building a D-flip flop and test bench to simulate the program.

- To get explanation of the syntax involved in SystemC programming , do refer to SystemC user guide .
- The next slide gives an example of a D-flip flop and an explanation for the same.



Here is a model of a basic D-type flip-flop.

```
// File: basic_ff.h
#include "systemc.h"

SC_MODULE (basic_ff) {
  sc_in<bool> d, clk;
  sc_out<bool> q;

  void prc_basic_ff();

  SC_CTOR (basic_ff) {
    SC_METHOD (prc_basic_ff);
    sensitive_pos << clk;        // Edge sensitivity.
  }
};

// File: basic_ff.cpp
#include "basic_ff.h"

void basic_ff::prc_basic_ff () {
  q = d;
}
```

<u>Courtesy:</u> SystemC primer
by J.Bhasker

To model synchronous logic the `SC_MODEL` process must be used with edge sensitivity.

The sensitivity list contains the edge sensitivity `sensitive_pos` specified on port `clk` , indicates that only on the rising edge of port `clk` does the data input `d` gets transferred to the output `q`.

Flip flop with Asynchronous Preset and Clear , these signals additionally specified as part of edge sensitivity list

A module can contain any number of processes , with each process either being a combinational process or a synchronous process.

Model of D-type flip-flop

Here is the model for a D-type flip-flop followed by a test bench

***Models**:*

a) ff.cpp & ff.h

b) ff_tb.cpp & ff_tb.h

c) Main.cpp

A brief note on all the programs are given in the program itself.

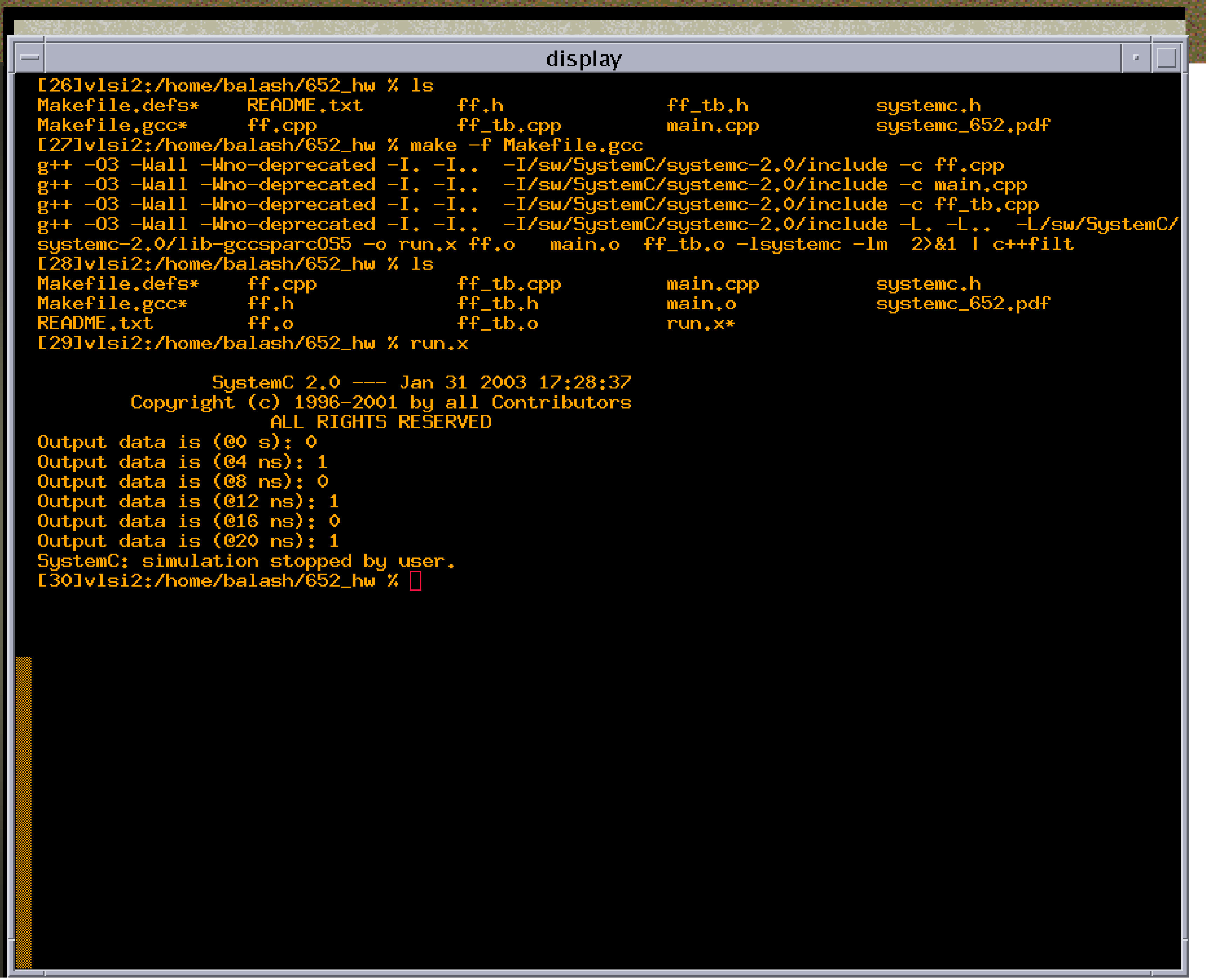
Compiling the Design

- Untar the file “652_hw.tar” .
- A directory will be created with the following files
  - Makefile.defs
  - Makefile.gcc
  - ff.cpp & ff.h
  - ff._tb.cpp & ff_tb.h
  - main.cpp

To compile all the files give the command

```
make –f Makefile.gcc
```

Next slide , gives you the desired output



```
[26]vlsi2:/home/balash/652_hw % ls
Makefile.defs*    README.txt        ff.h              ff_tb.h           systemc.h
Makefile.gcc*     ff.cpp            ff_tb.cpp         main.cpp          systemc_652.pdf
[27]vlsi2:/home/balash/652_hw % make -f Makefile.gcc
g++ -O3 -Wall -Wno-deprecated -I. -I..  -I/sw/SystemC/systemc-2.0/include -c ff.cpp
g++ -O3 -Wall -Wno-deprecated -I. -I..  -I/sw/SystemC/systemc-2.0/include -c main.cpp
g++ -O3 -Wall -Wno-deprecated -I. -I..  -I/sw/SystemC/systemc-2.0/include -c ff_tb.cpp
g++ -O3 -Wall -Wno-deprecated -I. -I..  -I/sw/SystemC/systemc-2.0/include -L. -L..  -L/sw/SystemC/
systemc-2.0/lib-gccsparcOS5 -o run.x ff.o   main.o  ff_tb.o -lsystemc -lm  2>&1 | c++filt
[28]vlsi2:/home/balash/652_hw % ls
Makefile.defs*    ff.cpp            ff_tb.cpp         main.cpp          systemc.h
Makefile.gcc*     ff.h              ff_tb.h           main.o            systemc_652.pdf
README.txt        ff.o              ff_tb.o           run.x*
[29]vlsi2:/home/balash/652_hw % run.x

               SystemC 2.0 --- Jan 31 2003 17:28:37
        Copyright (c) 1996-2001 by all Contributors
                    ALL RIGHTS RESERVED
Output data is (@0 s): 0
Output data is (@4 ns): 1
Output data is (@8 ns): 0
Output data is (@12 ns): 1
Output data is (@16 ns): 0
Output data is (@20 ns): 1
SystemC: simulation stopped by user.
[30]vlsi2:/home/balash/652_hw % 
```