SystemC Channels (03A)

SystemC

Young Won Lim
05/28/2012

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Young Won Lim
05/28/2012

Based on the following original work

[1] Aleksandar Milenkovic, 2002
CPE 626 The SystemC Language – VHDL, Verilog Designer's Guide
http://www.ece.uah.edu/~milenka/ce626-02S/lectures/cpe626-SystemC-L2.ppt

[2] Alexander de Graaf, EEMCS/ME/CAS, 2010
SystemC: an overview ET 4351
ens.ewi.tudelft.nl/Education/courses/et4351/SystemC-2010v1.pdf

[3] Joachim Gerlach, 2001
System-on-Chip Design with Systent of Computer Engineering
http://www2.cs.uni-paderborn.de/cs/ag-hardt/Forschung/Data/SystemC-Tutorial.pdf

[4] Martino Ruggiero, 2008
SystemC
polimage.polito.it/~lavagno/codes/SystemC_Lezione.pdf

[5] Deepak Kumar Tal, 1998-2012
SystemC Tutorial
http://www.asic-world.com/systemc/index.html

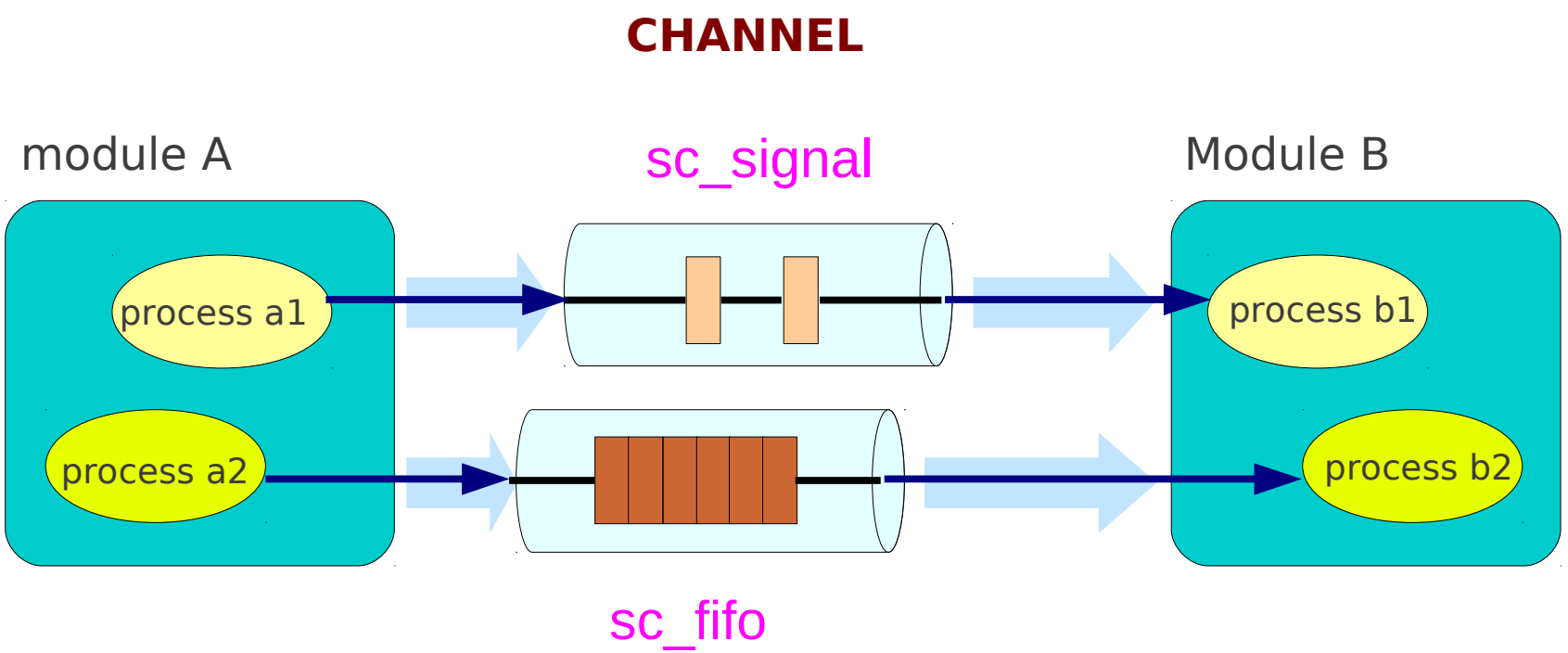
Channel

Channels provide communication between two modules
- inter- module
- intra-module

Two Types of Channel

- Primitive Channel
- Hierarchical Channel

Channel



single direction

Primitive Channel

- Channels that do not have hierarchy or process
- Supports the evaluate-update method of access:
  for simulating concurrency
- Ex. concurrent write & read from a channel
- Base class - sc_prim_channel
- Several built-in channels.
  - sc_signal
  - sc_fifo
  - sc_mutex
  - sc_semaphore

sc_signal

Primitive channel that uses evaluate-update paradigm

write() : cotains the evaluate portion of the evaluate-update

read() : reads the current value not the new value for update

event() : called to see if the channel issued an event

sc_fifo

sc_fifo has following predefined methods.

write() : if full, then it waits till fifo is available

nb_write() : does not wait. It returns with false.

read() : If empty, then it waits till data is available.

nb_read() :does not wait. It returns with false.

num_available() : returns the numbers of data available.

num_free() : returns the number of free slots available

sc_mutex

- Shares a common resource without colliding
- Must lock the mutex to get exclusive access to the resource
- Unlocks the mutex to make the resource available again
- Drawback: Repeated trylock

  → lack of an event that tells when an sc_mutex is freed

-

int lock() : Lock the mutex if it is free, else wait

int unlock() : Unlock the mutex

int trylock() : Check if mutex is free,

if free then lock, else return -1.

char* kind() : Return string "sc_mutex"

sc_semaphore

- A semaphore has an integer value - is the permitted number of concurrent accesses for the shared resources

int wait() : If the semaphore is 0, it waits until the semaphore value is incremented (by another process).

int trywait() : If the semaphore value is 0, it returns immediately the value -1.

int post(); increments the semaphore value.

int get_value() : returns the semaphore value.

char* kind() : Return string "sc_semaphore"

TMP

References

[1] Aleksandar Milenkovic, 2002
CPE 626 The SystemC Language – VHDL, Verilog Designer's Guide
http://www.ece.uah.edu/~milenka/ce626-02S/lectures/cpe626-SystemC-L2.ppt

[2] Alexander de Graaf, EEMCS/ME/CAS, 2010
SystemC: an overview ET 4351
ens.ewi.tudelft.nl/Education/courses/et4351/SystemC-2010v1.pdf

[3] Joachim Gerlach, 2001
System-on-Chip Design with Systent of Computer Engineering
http://www2.cs.uni-paderborn.de/cs/ag-hardt/Forschung/Data/SystemC-Tutorial.pdf

[4] Martino Ruggiero, 2008
SystemC
polimage.polito.it/~lavagno/codes/SystemC_Lezione.pdf

[5] Deepak Kumar Tal, 1998-2012
SystemC Tutorial
http://www.asic-world.com/systemc/index.html