TRANSACTION GENERATOR 2 USAGE

Updated: July 8, 2011

Lasse Lehtonen, Esko Pekkarinen
Department of Computer Systems
Tampere University of Technology

1 PACKAGE CONTENTS

1.1 Directory structure

`bin/`	Location for executables
`doc/`	Documentation
`examples/`	Example XML models
`execution_monitor/`	Execution Monitor source files
`hw_lib/`	Network-on-Chip hardware models
`adapters/`	Example OSCI TLM adapter
`fifo/`	Fifos used by Mesh_2D
`mesh_2d/`	Mesh_2D NoC models
`noc_factory/`	Source code to modify for adding NoC models
`packet_codec/`	Packet codec used by Mesh_2D
`simple_bus/`	Simplified OSCI TLM shared bus model
`java_tool_installer/`	Java libraries for Execution Monitor
`lib/`	Support libraries for executables
`scripts/`	Scripts for Modelsim
`transaction_generator_2/`	Main simulator source files
`work_libs/`	Working directory for Modelsim
`Makefile`	Main makefile

1.2 Included Network-on-Chips

Class	Type	Description
mesh_2d	vhd	Synthesizeable VHDL RTL model of a two dimensional mesh.
	sc_rtl_1	SystemC RTL model. Uses sc_logic and sc_lv for logic.
	sc_rtl_2	SystemC RTL model. Uses bool and sc_bv for logic.
	sc_tlm_1	SystemC OSCI TLM model.
simple_bus	sc_tlm_1	SystemC OSCI TLM model.

2 PREREQUISITES

2.1 Mandatory prerequisites

Transaction Generator 2 needs a terminal environment with the following tools and libraries. Tested in Windows XP with Cygwin (32-bit) and in Debian Squeeze (64-bit).

Make	Tested with version 3.80
gcc	Tested with version 3.4.3
Boost c++ libraries	Transaction Generator 2 utilizes many libraries from Boost. At least version 1.42.0 is needed. Header files are sufficient if not using Execution Monitor (www.boost.org)
SystemC	Tested with version 2.2.0 (www.systemc.org)
OSCI TLM	Tested with version 2.0.1 (www.systemc.org)
OCP-IP TLM Kit	Tested with version 2.0 (www.ocpip.org)

2.2 Optional prerequisites for mixed-language simulation

Transaction Generator 2 can be used easily with Network-on-Chips described in other languages than SystemC. Package's makefile has the necessary commands for example simulation with Modelsim SE.

Modelsim SE	Tested with versions 6.4c and 6.6a

2.3 Optional prerequisites for Execution Monitor

Execution Monitor is a Java program used to visualize utilization of processing elements and the status of the application model. Package includes precompiled version but to compile it one needs:

Java Development Kit (JDK)	Tested with version 1.6.0_16
Apache Ant	Tested with version 1.7.1
Boost c++ libraries	TG uses Boost.Asio which requires compiled installation

3 USAGE

3.1 Setting environment

Transaction Generator 2 is makefile driven and it's designed to be used from the directory it is extracted to. Modify Makefile's contents to suit the used environment.

```
# Boost's header files
BOOST_INC      = ???/boost_1_42_0/include
#Boost's libraries
BOOST_LIBS     = ???/boost_1_42_0/lib
# OSCI TLM headers
TLM_INC        = ???/tlm-2009-07-15/include/tlm
# SystemC header files
SC_INC         = ???/systemC/include
```

3.2 Compilation

Makefile's default objective is to compile all SystemC only files and create executeable file `sctg` (or `sctg.exe`) to package root.

```
$ make clean
$ make
```

3.3 Running simulation

`sctg` requires at least one parameter which locates the xml source file containing the application model.

```
$ ./sctg -i examples/test_mesh.xml
```

Other possible parameters and their descriptions can be viewed using the −−help parameter.

```
$ ./sctg --help
Transaction Generator 2
Allowed options:
  --help                     This message
  -i [ --input-file ] arg    Input file
  -r [ --replay-file ] arg   Recorded file to replay
  -d [ --delay ] arg          Delay (ms) to slow down replay
  -s [ --save-directory ] arg Directory to save logs
  -e [ --execution-monitor ]  Use Execution Monitor
```

3.4 Mixed language simulation with Modelsim

Makefile assumes Modelsim's binaries are found in $PATH.

Create a DO-file for Modelsim which initializes working libraries and compiles all source files.

```
$ make modelsim
```

Open Modelsim and run the DO-file in transcript window.

```
ModelSim> do sctg.do
```

Run the example simulation for mesh or simple bus network.

```
$ make sim_mesh
```

```
$ make sim_sbus
```

3.5 Simulation with Execution Monitor

Modify `Makefile.env` and uncomment the following variables. Values of `ASIO_FLAGS` (given to compiler) and `ASIO_LINK` (for linker) depend on the used system. Refer to Boost.Asio documentation. Example values below are for Windows XP with Cygwin.

```
USE_EXECMON    = -DSCTG_USE_EXECMON
ASIO_FLAGS     = -D_WIN32_WINNT=0x0501 -D__USE_W32_SOCKETS
ASIO_LINK      = -lboost_system -lws2_32
```

```
$ make clean
$ make
```

If you want to compile Execution Monitor again set environment variable `JAVA_HOME` to point JDK and `ANT_HOME` to its home directory. Remove previous installation and compile it.

```
$ make clean_javas
$ make execmon
```

Launch Execution Monitor and configure its user interface (*File->Open*) with file `execmon_conf.xml` (stored in examples directory).

```
$ bin/execution_monitor &
```

Start Transaction Generator 2 with -e parameter. It will wait for Execution Monitor to connect to it (use reconnect button in Execution Monitor's lower left corner).

```
$ ./sctg -e -i examples/test_mesh.xml &
```

Mixed language simulation with Execution Monitor requires the use of simulator's replay-mode. After running the simulation start Transaction Generator 2 using log_execmon.txt as a record file.

```
$ ./sctg -r log_execmon.txt &
```