DESCRIPTION

This directory and its contents contain a project that ties a zedboard design to a SystemC ESL design. This was described in a presentation entitled "A SystemC Technology Demonstrator" at the SystemC Tutorial at DVCon 2013 in San Jose. The goal of the project was to demonstrate multiple technologies including:

- ZedBoard containing Xylinx Zynq-7000 FPGA technology (contains dual Cortex A9's and many peripherals)
- How to have OS threads safely communicate with SystemC
- Use of TCPIP sockets, POSIX threads, and mutexes in C
- SystemC code using both IEEE 1666-2011 (C++) and ISO-IEC 14882-2011 (C++) features. These included:
  - SystemC async_request_update
  - SystemC ensuring sc_stop is called
  - C++ user-defined literals
  - C++ override keyword for safety
  - C++ threads
  - C++ mutexes
  - C++ unique_ptr<>
- Other features incidentally used:
  - SystemC dumping a netlist
  - Managing sc_reporting

This directory contains two subdirectories: sysc/ and zedboard/ representing the component parts. There are also a few supporting directories: docs/ contains information about the project. bin/ contains some supporting scripts for one of two makefile flows (systemc related). NOTE: You probably do not need to understand the two makefile flows in any detail.

The zedboard/ directory contains C-code for a zedboard; although, it can be compiled for execution any any standard linux environment that supports POSIX sockets and pthreads. main simply sends a variety of random data values to a driver that communicates with sockets to a remote SystemC design.

The sysc/ directory contains C++ code for a SystemC adaptor that takes TLMX packets and converts them to TLM 2.0 payloads, which are then sent to the TLM target.

REQUIREMENTS

- A C++11 compliant compiler such as g++ 4.7 or clang++ 3.2
- A C99 compliant cross-compiler for the zedboard's Cortex-A9 linux environment if you wish to use the zedboard. A simple C99 compliant compiler for linux will suffice if only targeting another linux process.
- SystemC installation compliant to IEEE-1666-2011 such as ASI's proof-of-concept version 2.3.
- Zedboard (around 400 $US at time of writing) -- go to zedboard.org for more information

---

HOW TO BUILD

Make sure you meet the general requirements. See NOTES.md for additional information.

To build the software for systemc, type:

- pushd sysc && make clean exe && popd

To build the software for zedboard, type:

- pushd zedboard && make TARGET_ARCH=ZEDBOARD clean exe && popd

---

HOW TO RUN

Before running this software, it is neccesary to connect the ethernet to a host that is running the SystemC server/simulator software called zynq.x. On the remote host, type:

- sysc/async_adaptor-linux64.x -port=PORTNUMBER

It should output half a page of info and then pause.

To execute the initiator software in the zedboard directory type:

- zedboard/software.x HOSTNAME PORTNUMBER

Where HOSTNAME should designate a host running the SystemC simulator on the ethernet and the PORTNUMBER should match the sysc number. You can specify either a DNS name or the IP address (e.g. 198.168.1.12)

Port numbers should be number greater than 2000 to avoid collisions with standard OS ports (e.g. mail or ssh). Suggest using 4000.

ABOUT THE SOURCE

SystemC code for simulated portion:

- sc_literals.cpp -- C++11 feature to make SystemC feel more natural
- netlist.cpp -- displays a simple netlist of the design
- report.cpp -- convenience features to improve reporting
- tlmx_packet.cpp -- TLM-like class used over sockets. Includes serialization.
- tlmx_channel.cpp -- Thread-safe SystemC channel
- async_adaptor.cpp -- OS thread receiving TCP/IP traffic to forward to SystemC
- dev.cpp -- dummy "device" used as target
- top.cpp -- top-level netlist
- main.cpp -- SystemC main including report summary

C-code for embedded system:

- random.c -- implements standard srandom/random
- creport.c -- simplifies error reporting
- tlmx_packet.c -- describes the TLM-like structure used over sockets. Includes serialization.
- driver.c -- where the driver lives
- software.c -- main

TAF!