The tmemon Program

Dave Galloway

Edward S. Rogers Sr. Department of Electrical and Computer Engineering
University of Toronto

January 2014

Introduction

The `tmemon` program is a daemon process that acts as a front end for the demonstration board. It runs on the workstation that is connected to the demonstration board (currently skynet.eecg, November 2011).

Tmemon opens two network TCP stream sockets, and sits waiting for connections to those sockets. On one of the sockets (the housekeeping socket) tmemon will accept requests to change or report on the state of the demonstration board. On the other socket (the ports socket) tmemon will accept data to be transferred to or from a user's design inside the demonstration board. See *The TME Ports Package* for a description of how the ports socket is used.

A second program called `tm` is usually used to talk to the housekeeping socket of tmemon. The tm program can be run on any host, and will transfer one or more requests to tmemon over the network. A single request can be given as the arguments to tm. For example,

```
tm getdesigndir
```

will ask tmemon for the name of the directory containing the design most recently loaded into the demonstration board.

Multiple requests can be passed to the tm program on the standard input by giving it a "-" argument:

```
tm -
getdesigndir
getdesigndate
^D
```

Starting Tmemon

To start the tmemon daemon, sign on to the computer that has the demonstration board board connected to it, and type:

```
cd /
tmemon &
```

Tmemon Housekeeping Socket Requests

This is a list of the requests that tmemon understands on the housekeeping socket:

acquire P N	Ask for the demonstration board at priority P for N seconds. Returns 0 if successful, or the current priority and the number of seconds left in the current reservation if not.
debug N	Set the debug level to N. Debugging output will go to a file called /tmp/TMmon.debugout on the machine that is running tmemon.
EXIT	Causes tmemon to exit.
getarch	return TME as a string
getdesigndir	return the arguments of the last setdesigndir command
getportlistdir	return the arguments of the last setportlistdir command
getstatus	return a string of 1s and 0s that describe the current status of the board
getstatus1	return the arguments of the last setstatus1 command
getstatus2	return the arguments of the last setstatus2 command
release	release the demonstration board so others can acquire it
setdesigndir dirname	set the name of the directory containing the design
setportlistdir dirname	set the name of the directory containing the port lists for the current design
setstatus1 args	set the string returned by getstatus1
setstatus2 args	set the string returned by getstatus2
stop	kill off any ports sockets connected to programs that are talking to the current design, and stop talking to the FPGA.

Tmemon Ports Socket Format

The ports socket is normally used by the tmports package to send data to and from a user’s circuit in the demonstration board. Data is sent to the ports socket as a series of packets. Each packet consists of a 32 bit packet header, followed by optional data bytes. Each header specifies a chip and port number where the data is to be transferred. The 32 bit header is sent in network byte order (most significant byte first). The most significant bit is labelled 31 in this table:

Bit Number	Meaning
31	1 for a write (data transferring to circuit); 0 for read
30-28	board id (0 is the default board, 1 is DE-4, 2 is DE-5)
27-26	must be 0
25-18	port number inside the chip
17-0	packet data length in bytes (max length is 262143 bytes)

A header requesting a write must be followed by the number of bytes of data that were specified in the header. A read request header will cause tmemon to block until the specified number of bytes are received from the design.

Each request (read or write) will be answered by a 32-bit word containing the number of bytes transferred to or from the circuit in the case of a successful transfer, or -1 in the case of an error. If it was a successful read request, the word will then be followed by the number of bytes requested.