Readme File for TELS

Dept. of Electrical Engineering, Princeton University

March 30, 2006

1 ABOUT TELS

TELS is a tool that does threshold logic synthesis. It can be used for circuits made of resonaunt-tunneling diode (RTDs) and quantum-cellular automata (QCA) based nanotechnologies. It is developed on top of SIS and distributed freely for research purpose. PLEASE DO NOT REDISTRIBUTE TELS OR MALS in any other manner. It comes as-is with no warranties.

2 INSTALL & RUNNING TELS

2.1 INSTALL

Before installing, you should install student versions of CPLEX and AMPL (www.ampl.com) on your machine and add these two programs in your ${PATH} variable.

The tool has been tested on Red Hat Linux 8.0 and above only. The installation instructions are as follows:

1. Type 'tar zvxf sis-1.2.tar.gz' to uncompress the file.
2. Type 'cd sis-1.2; make' to compile everything
3. Add '${PATH}/sis-1.2/bin' to your ${PATH} variable

Note that you may have to modify the variables in the top level Makefile a little bit to get the tool to compile. If you are having some major problems, send a detailed report to one of the authors and they will try to get back to you.

Here are some problems you may encounter during compiling PROBLEMS:

1. Make sure /bin/make exists. Create a soft link if you have to.
2. In newer versions of linux (Red Hat 8.0 and above), you have to modify /usr/include/bits/dirent.h to get SIS to compile. Here is what the file looks like. Add "int d_namlen;" as shown below. If you don't, you will get a bunch of compile errors.

```
#ifndef _DIRENT_H
# error "Never use <bits/dirent.h> directly; include <dirent.h> instead."
#endif

struct dirent
  {
#ifndef __USE_FILE_OFFSET64
    __ino_t d_ino;
    __off_t d_off;
#else
    __ino64_t d_ino;
    __off64_t d_off;
#endif
    unsigned short int d_reclen;
    unsigned char d_type;
    int d_namlen; /* pallav, for sis to work */
    char d_name[256];           /* We must not include limits.h! */
  };
...
```

2.2 HELP

Run-time help can be obtained by typing "help" or "man" while in sis, which will list all of the available commands (including TELS commands). Help for each command can be obtained by typing "help command_name" or "man command_name" while in sis.

2.3 TELS COMMANDS

TELS provides the following new commands: *thatpg, thbuffer, thconv, thdot, thfsim, thprint, thprint_stats, thsim, thsyn, thsyn_red_remove, thvcg, thfactor*

usages:

- **thatpg** [-d] [-c] [-n num] [-r limit] [-b limit] [-t time] [-y num] [-z num] [-o ⟨file⟩]

  -d Perform fault dropping

  -c Do not perform fault collapsing

  -n ⟨num⟩ Number of vectors to generate for each pass of random test generation (default is 32)

  -r ⟨limit⟩ Fault coverage limit during random test generation (default is 0.75)

  -b ⟨limit⟩ Backtrack limit during deterministic test generation (default is 1000)

  -t ⟨time⟩ Timeout limit for random test generation (default is 120s)

-y ⟨num⟩ δ_{on} (default is 0).

-z ⟨num⟩ δ_{off} (default is 0).

-o ⟨file⟩ Write test vectors to ⟨file⟩

- **thbuffer** [-h]

  -h: For help information.

  This command does technology mapping for RTD and Single-Electron Boxes based ciruits by inserting buffers to the circuit wherever needed. The circuit's opera tion is based on the four-phase or three-phase clock scheme for RTDs and Single- electron boxes respectively.

- **thconv** [-a and] [-o or] [-n num1] [-f num2]

  -a and And gate with fanin limit 'and' (default is 2)

  -o or Or gate with fanin limit 'or' (default is 2)

  -n num1: Assign the integer num2 to δ_{on}, default value is 0.

  -f num2: Assign the integer num3 to δ_{off}, default value is 1.

  This command mapps a given netowrk to AND/OR gates composed threshold networks.

- **thdot** [-o file] [-m int1] [-h]

  -o *langle*file⟩ Write the dot file to ¡file¿, default write to thdot_output.dot.

  -m ⟨int1⟩ Assign the printing mode to int1. Print weight on the edge when int1=1. Print weight-threshold vector inside of the node when int1=2 (default is 1).

  -h Print out help information

- **thfsim** [-d] [-c] [-o file] -i ⟨file⟩

  -d Perform fault dropping

  -c Do not perform fault collapsing

  -i ⟨file⟩ Read test vectors from ⟨file⟩

  -o ⟨file⟩ Write all detected faults to ⟨file⟩

- **thprint** -h

  usage: thprint [-d] [n1 n2 ...]

  -d Print in negative form

  This command prints out the threshold nodes information of given threshold networks (threshold and weights values and Boolean functions).

- **thprint_stats** -h

  usage: thprint_stats [n1 n2 ...]

  This command prints out the threshold network statitics, such as number of nodes, number of levels, network area, etc.

- **thsim** [-c] [-s] [-o ¡file¿] -i ⟨file⟩

  -c Do not do combinational simulation

  -s Do not do STG simulation (default is NO) (TODO)

  -o ⟨file⟩ Dump results to file

  -i ⟨file⟩ Read vectors from file

- **thsyn** [-c num1] [-n num2] [-f num3] [-o file] [-h]

  -c num1: Assign the integer num1 to fanin restriction, default value is 6.

  -n num2: Assign the integer num2 to δ_{on}, default value is 0.

  -f num3: Assign the integer num3 to δ_{off}, default value is 1.

  -o file: Write the linear program problem into file.mod and file.dat, default write to thsyn_ampl.mod and thsyn_ampl.dat.

  -h: For help information.

  This command does the threshold logic synthesis job. Given a network written in blif format, it outputs the synthesized functionally equivalent threshold network.

- **thsyn_red_remove** [-c num1] [-n num2] [-f num3] [-o file] [-h]

  -c num1: Assign the integer num1 to fanin restriction, default value is 6.

  -n num2: Assign the integer num2 to δ_{on}, default value is 0.

  -f num3: Assign the integer num3 to δ_{off}, default value is 1.

  -o file: Write the linear program problem into file.mod and file.dat, default write to thsyn_ampl.mod and thsyn_ampl.dat.

  -h: For help information.

  This command does the threshold logic synthesis job. Given a network written in blif format, it outputs the synthesized functionally equivalent threshold network that is irredundant.

- **thvcg** [-o file] [-m int1] [-s int2] [-h]

  -o ⟨file⟩ Write the vcg file to ⟨file⟩, default write to thvcg_output.vcg.

  -m ⟨int1⟩ Assign the printing mode to int1. Print weight on the edge when int1=1. Print weight-threshold vector inside of the node when int1=2 (default is 1).

  -s ⟨int2⟩ Shrink the node size into $(1/int2) * 100$ percent (default is 1).

  -h Print out help information

- **thfactor** [-c num1] [-n num2] [-f num3] [-p num4] [-r] [-o file] [-h]

  -c num1: Assign the integer num1 to fanin restriction, default value is 6.

  -n num2: Assign the integer num2 to δ_{on}, default value is 0.

  -f num3: Assign the integer num3 to δ_{off}, default value is 1.

-r : Split each node in the network recursively until num1 is reached or no kernels (default is recursively).

-p num4: Consider the num4/100 portion of the candidate kernels (0 ¡= num4 ¡=100, default is 60).

-o file: Write the linear program problem into file.mod and file.dat, default write to thsyn_ampl.mod and thsyn_ampl.dat.

-h: For help information.

This command uses new objective function to do factorization. It can be used as a preprocess procedure before doing threshold logic synthesis.

2.4 MODIFICATIONS TO SIS

The network structure has a new field: int threshold. For non-threshold network, $network \rightarrow threshold = 0$. For threshold network, $network \rightarrow threshold = 1$.

The node structure, node_t *, has a new field: char *threshold. It is reserved for threshold package.

2.5 INPUT to TELS

Input to TELS should be a combinational circuit. For example, the circuit can be written in *.blif* format. Preprocessing (such as algebraic decompositon, boolean decomposition, technology mapping to AND/OR gates, etc) can be done before calling TELS. Some TELS commands, such as *thconv, thsyn*, set $network \rightarrow threshold$ to 1 after execution. Some TELS commands, such as thprint, thprint_stats, thsim ..., needs to make sure $network \rightarrow threshold$ is 1 before execution.

2.6 EXAMPLE

First run SIS by typing command "sis". In SIS, first read in a nework by using some SIS commands, such as "read_blif xxx.blif". Preprocessing, such as algebraic decompositon, can then be done by some script file or SIS commands. After this, type TELS command "thsyn" or "thconv" with correct options to generate a valid threshold network.

Once we got the threshold network, more TELS commands can be used. For example, "thprint" is used to print out the node information with weights and threshold values. "thsim" is used for threshold network simulation. "thprint_stats" can print out the threshold network overall information, such as number of nodes, number of levels.

One example is given below:

```
[sispurnea mlex]$ sis
UC Berkeley, SIS 1.3 (compiled 13-Jul-04 at 3:15 PM)
sis> read_blif majority.blif /* read in the network */
sis> print /* print out the network */
f = h'
h = a'b'd' + a'c'd' + a'd'e' + b'c'd' + b'd'e' + c'd'e'
sis> thsyn /* do threshold network synthesis */
sis> thprint /* print out the threshold network */
f = 3.0; d = 3.0; a = 1.0; b = 1.0; c = 1.0; e = 1.0;
f = abc + abe + ace + bce + d
sis> thprint_stats /* print out the status of the threshold network */
# of pi: 5
# of po: 1
# of levels: 2
# of nodes: 1
area: 10.00
network name: traffic_cl
sis>
```

3 MORE INFORMATION

For more information, refer to our paper:

Rui Zhang, Pallav Gupta, Lin Zhong, and Niraj K. Jha, "Threshold network synthesis and optimization and its application to nanotechnologies," in IEEE Trans. on Computer-Aided Design, vol. 24, no. 1, pp. 107-118, Jan. 2005.

Please reference the above paper in work which has significant use of TELS.

4 CONTACTS

All bugs and problems should be mailed to pgupta@princeton.edu or rzhang@princeton.edu. Please send a detailed description of the bugs and problems.