**CMPE/ENEE 691: Hardware Security**
**Spring 2023**
**University of Maryland Baltimore County**

# Tutorial: Using TetraMAX to find SCOAP Testability (Controllability and Observability) Metrics

How to use TetraMAX?

TetraMAX is a high-speed, high-capacity automatic test pattern generation (ATPG) tool. It can generate test patterns that maximize test coverage while using a minimum number of test vectors for a wide variety of design types and design flows. It is well suited for designs of all sizes up to millions of gates. Information about the tool is available in the TetraMAX User Guide. The main components of the GUI are shown in Figure 1.

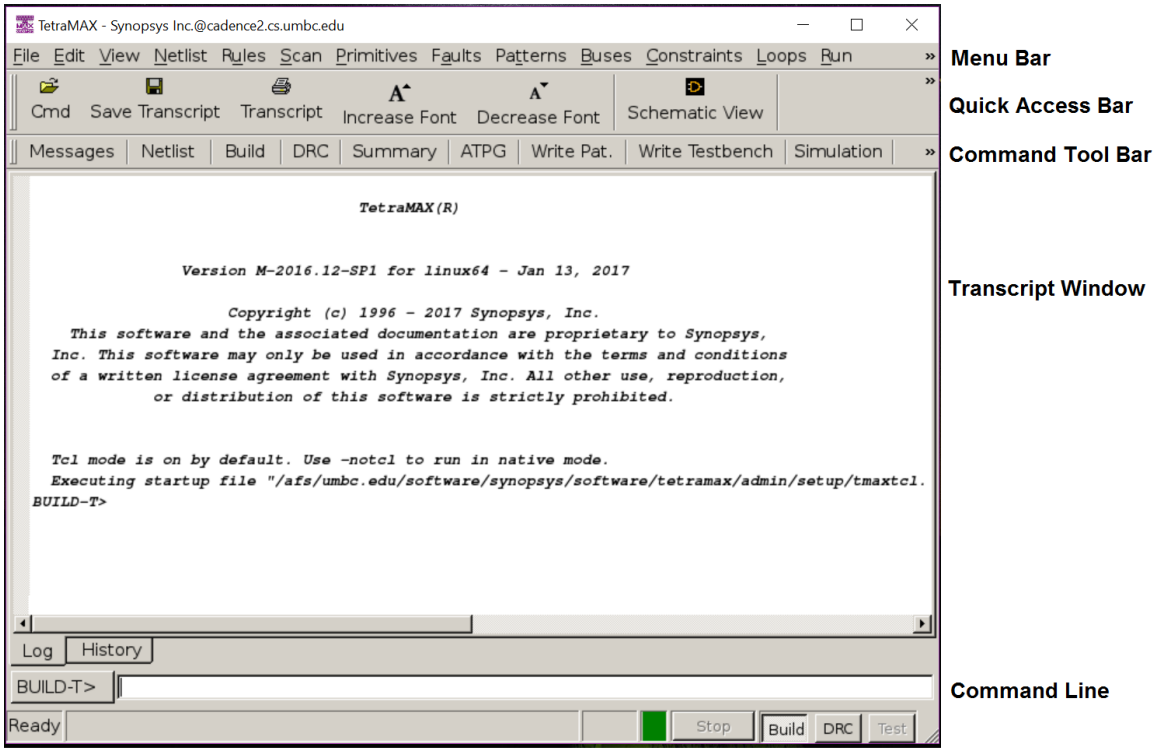



**Figure 1: TetraMAX Main Window**

Whenever you want to work on a tmax project, it is recommended to have all your files in the same directory. Create a working directory as follows:

$ mkdir lab4
$ cd lab4

Copy the 3 files you need for the HW4b from BB to the folder you asked to create in the previous step.

You should now have 3 files in your work directory:

1. **mycircuit.v**: Verilog netlist of the circuit we want to examine
2. **lib.v**: Verilog library file containing the gate primitives used in **mycircuit.v**
3. **scoap.tcl**: TCL script that you will run on TetraMAX to generate the testability measures for your circuit. The script is described in Figure 2.

```
# set the log file as ./scoap-test.log
set_messages -log ./scoap-test.log -replace
# read the source and library files and build the circuit
read_netlist ./lib.v
read_netlist ./mycircuit.v
run_build_model circuit

# Add the scan chains in the circuit and constraining the scan chain inputs - these steps are not within the scope of this lab but necessary to calculate scoap data for a sequential circuit
add_clocks 0 { CK } -shift -timing { 100 50 80 40 }
add_scan_chains chain1 test_si1 test_so1
add_scan_chains chain2 test_si2 test_so2
add_scan_chains chain3 test_si3 test_so3
add_scan_chains chain4 test_si4 test_so4
add_scan_chains chain5 test_si5 test_so5
add_scan_chains chain6 test_si6 test_so6
add_scan_chains chain7 test_si7 test_so7
add_scan_chains chain8 test_si8 test_so8
add_scan_chains chain9 test_si9 test_so9
add_scan_chains chain10 test_si10 test_so10
add_scan_enables 1 test_se
add_nofaults -module SDFFX1
add_nofaults -module DFFNX2
add_pi_constraint 0 test_si1
add_pi_constraint 0 test_si2
add_pi_constraint 0 test_si3
add_pi_constraint 0 test_si4
add_pi_constraint 0 test_si5
add_pi_constraint 0 test_si6
add_pi_constraint 0 test_si7
add_pi_constraint 0 test_si8
add_pi_constraint 0 test_si9
add_pi_constraint 0 test_si10
add_pi_constraint 0 test_se

# run design rule check
run_drc

# set fault model and run automatic test pattern generation - these steps are not within the scope of this lab but are necessary to calculate scoap data
set_faults -model stuck
remove_faults -all
add_faults -all
run_atpg -ndetects  1

# add scoap data to pin data to be reported
set_pindata -scoap_data
# write pin data (which includes scoap data) to the output file "out.txt"
report_primitives -all > out.txt
```

**Figure 2: TCL Script for Calculating Testability Measures**

From your working directory, start TetraMax by running:
$ tmax &

In the Quick Access Bar, click on "Cmd", find your script and run it, as shown in Figure 3.

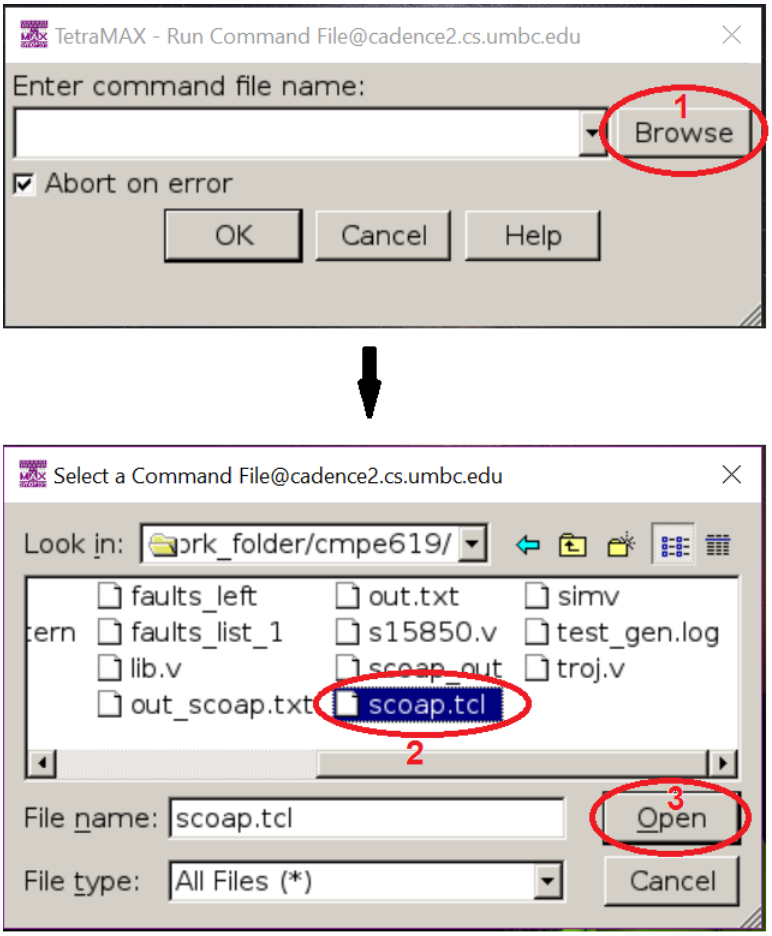
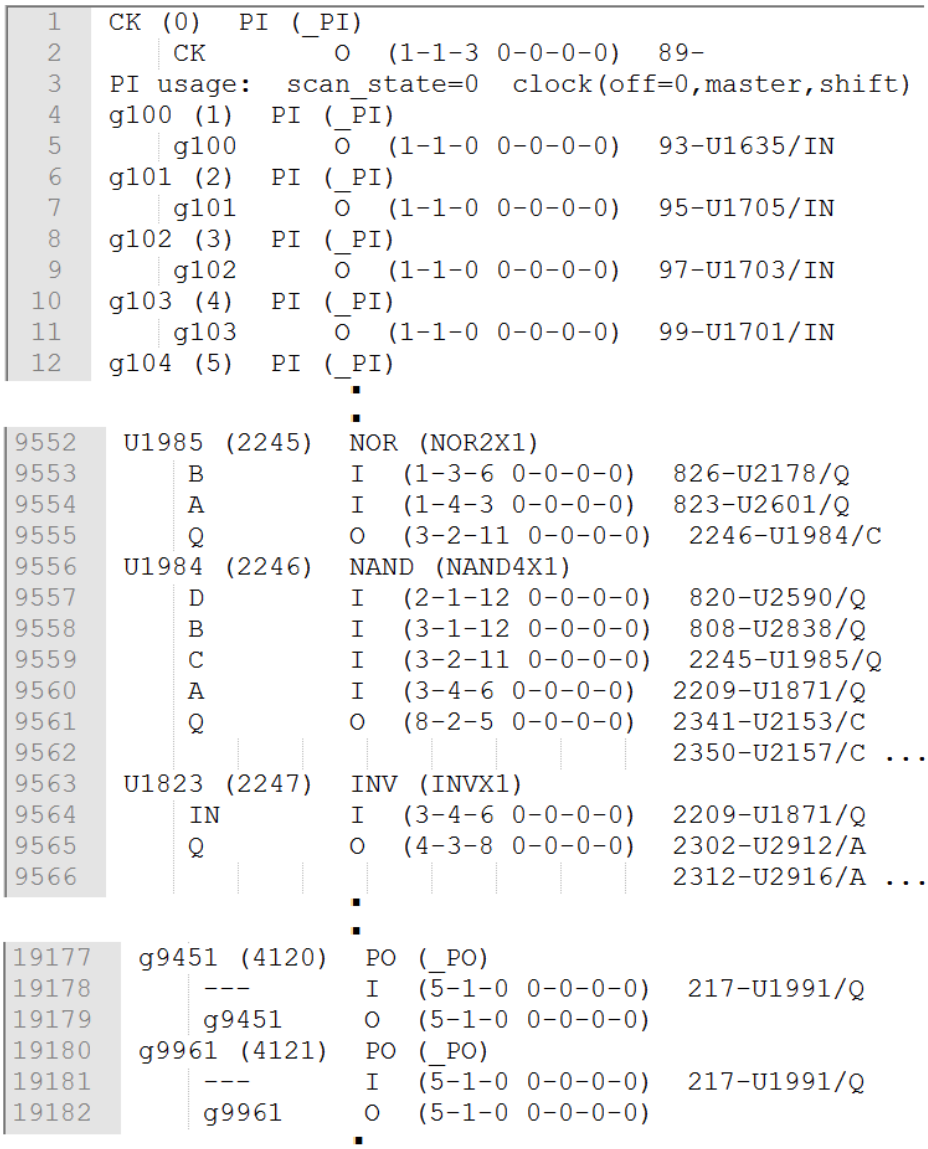



**Figure 3: Running the TCL script**

You can also run the script be typing the command in the TetraMAX Command Line box.
**$** source <path to scoap.tcl>
E.g., assuming you invoked TetraMAX from your working directory to which you previously copied "scoap.tcl"
**$** source ./scoap.tcl

When you run "scoap.tcl", an output file called "out.txt" which contains the net descriptions will be saved to your working directory. Figure 4 shows some sections of the output file.



```
 1   CK (0)   PI (_PI)
 2       CK          O   (1-1-3 0-0-0-0)   89-
 3   PI usage:  scan_state=0  clock(off=0,master,shift)
 4   g100 (1)   PI (_PI)
 5       g100        O   (1-1-0 0-0-0-0)   93-U1635/IN
 6   g101 (2)   PI (_PI)
 7       g101        O   (1-1-0 0-0-0-0)   95-U1705/IN
 8   g102 (3)   PI (_PI)
 9       g102        O   (1-1-0 0-0-0-0)   97-U1703/IN
10   g103 (4)   PI (_PI)
11       g103        O   (1-1-0 0-0-0-0)   99-U1701/IN
12   g104 (5)   PI (_PI)
                  .
                  .
9552  U1985 (2245)   NOR (NOR2X1)
9553      B          I   (1-3-6 0-0-0-0)   826-U2178/Q
9554      A          I   (1-4-3 0-0-0-0)   823-U2601/Q
9555      Q          O   (3-2-11 0-0-0-0)   2246-U1984/C
9556  U1984 (2246)   NAND (NAND4X1)
9557      D          I   (2-1-12 0-0-0-0)   820-U2590/Q
9558      B          I   (3-1-12 0-0-0-0)   808-U2838/Q
9559      C          I   (3-2-11 0-0-0-0)   2245-U1985/Q
9560      A          I   (3-4-6 0-0-0-0)   2209-U1871/Q
9561      Q          O   (8-2-5 0-0-0-0)   2341-U2153/C
9562                                        2350-U2157/C ...
9563  U1823 (2247)   INV (INVX1)
9564      IN         I   (3-4-6 0-0-0-0)   2209-U1871/Q
9565      Q          O   (4-3-8 0-0-0-0)   2302-U2912/A
9566                                        2312-U2916/A ...
                  .
                  .
19177  g9451 (4120)   PO (_PO)
19178      ---        I   (5-1-0 0-0-0-0)   217-U1991/Q
19179      g9451      O   (5-1-0 0-0-0-0)
19180  g9961 (4121)   PO (_PO)
19181      ---        I   (5-1-0 0-0-0-0)   217-U1991/Q
19182      g9961      O   (5-1-0 0-0-0-0)
                  .
                  .
```

**Figure 4: Output File**

The output file lists an index, then gate data, and lastly primary output data

We are concerned with the gate inputs and outputs only, as shown in the middle section of Figure 4.

Each gate (module) is described as follows:

**<module name> (<module number>) <description> (<gate primitive>)**

For example:

**U2280** (2864) OR (**OR2X1**)

Below the gate description, each pin is described as follows:
**<pin name>  <pin type>  (CC0-CC1-CO  CC0s-CC1s-COs-Cs)  <connected net>**

For example below the line shows U1984 (2246) NAND (NAND4X1), you see 5 lines as below:

```
D          I   (2-1-12 0-0-0-0)   820-U2590/Q
B          I   (3-1-12 0-0-0-0)   808-U2838/Q
C          I   (3-2-11 0-0-0-0)   2245-U1985/Q
A          I   (3-4-6 0-0-0-0)   2209-U1871/Q
Q          O   (8-2-5 0-0-0-0)   2341-U2153/C
                                 2350-U2157/C ...
```

The first 3 numbers in the braces are the combinational testability parameters.
**CC0**: controllability to 0
**CC1**: controllability to 1
**CO**: observability

The last 4 numbers are the sequential testability parameters (we do not deal with sequential testability parameters in this lab).

You can ignore any pin named "---".

Note1: TetraMAX has a modified version of SCOAP parameters, and it does not add **1** to controllability and observability whenever a gate is traversed.

Note2: A controllability or observability value of '*' means that the value exceeds 254 which is the maximum value the program can track. It is safe to replace '*' with 255 in your calculations.