

Lab2

Scan Chain Insertion and ATPG Using DFTADVISOR and FASTSCAN

Prof: Chia-Tso Chao

TA: Yu-Teng Nien

2019-05-31

Outline

- Introduction
- DFTADVISOR
- FASTSCAN
- Mixed Flow
- Lab

Outline

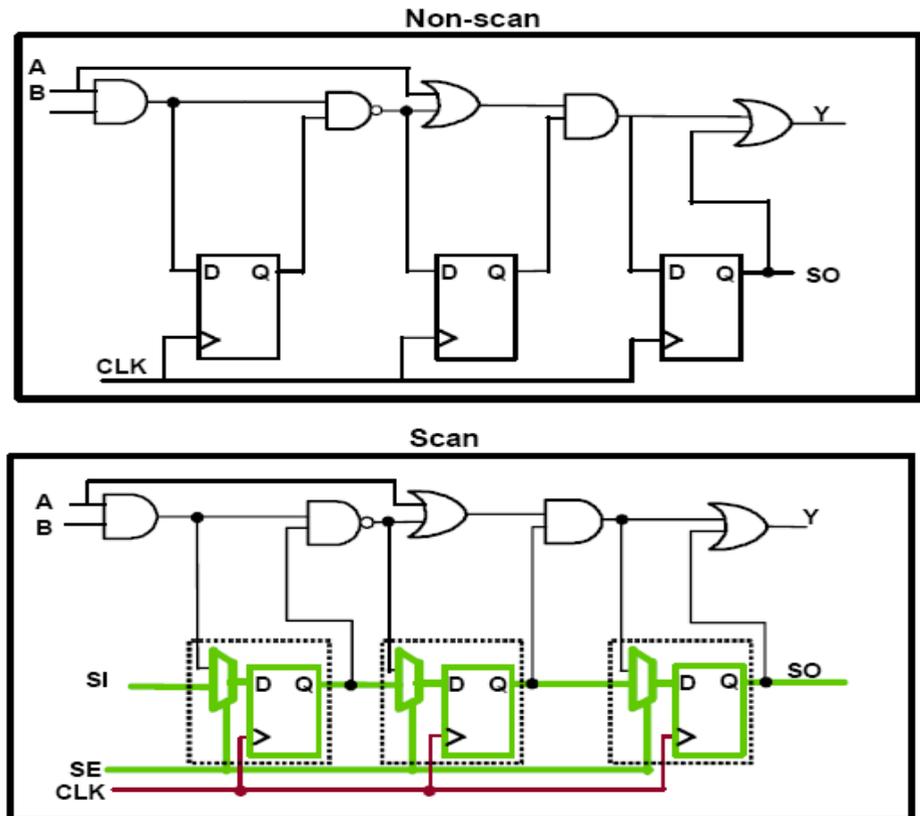
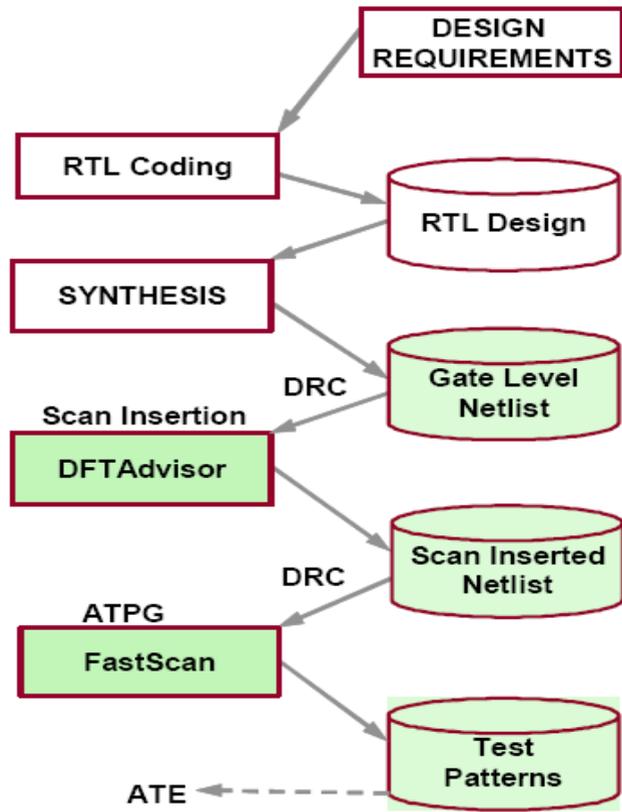
- Introduction
- DFTADVISOR
- FASTSCAN
- Mixed Flow
- Lab

Introduction

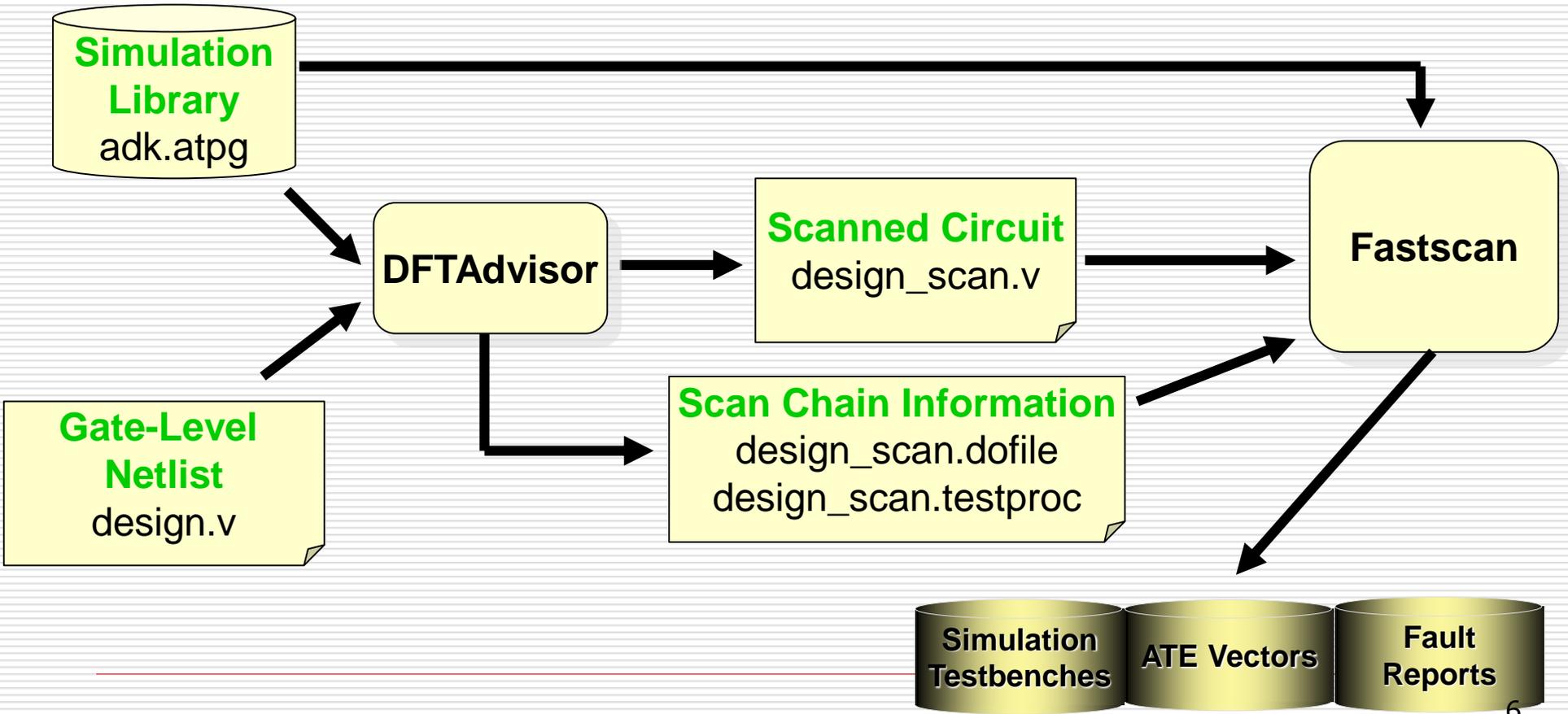
- This lab focuses on ATPG with tools from 2 different EDA vendors
 - Synopsys
 - Mentor Graphics
- DFTAdvisor inserts scan chain
 - Basically replace FFs with scan FFs
- Fastscan performs ATPG and fault simulation

Insert Scan and ATPG Flow

Tool Flow



Input/Output Files



Outline

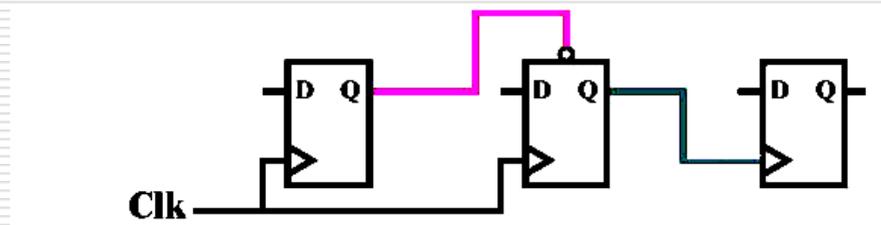
- Introduction
- DFTADVISOR
- FASTSCAN
- Mixed Flow
- Lab

Invoke DFTADVISOR

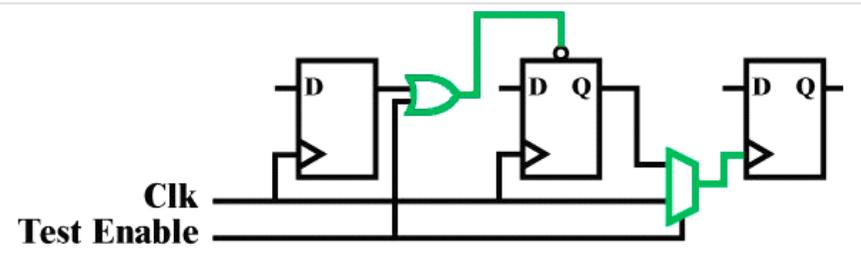
- Read in Verilog source file and assign ATPG library file
 - `$ dftadvisor pre_norm_noscan.v -verilog -lib l90sprvt.atpg -nogui`
- Default system mode is "SETUP"
 - `SETUP>`

Setup Test Logic Configuration

- ❑ Set scan style design
 - Mux_scan: mux-DFF
 - Lssd: level sensitive
 - Clocked_scan: clocked-signal
 - ❑ `SETUP> set_scan_type m`
- ❑ Test logic options make clock lines controllable to get a scannable design
 - `SETUP> set_test_logic`
 - clock on
 - reset on
- ❑ Verify with `report_environment`



Non-scannable



Scannable after test logic insertion

Enter Scan Insertion System Mode

- Enter Scan Insertion system mode (DFT) and perform scan identification
 - `SETUP> set_system_mode dft`
- Report detailed statistical report of scan identification
 - #Sequential instances
 - #Scannable instances
 - `DFT> report_statistics`

Insert Scan-Chain and View Report

- Set # of scan-chains to insert and do so
 - DFT> insert_test_logic -number 10
- Report scan-chain information
 - DFT> report_scan_chains
 - DFT> report_test_logic

Output Scanned Design for ATPG

- Write out files and exit DFTADVISOR
 - DFT> write netlist pre_norm_scan.v -verilog
-replace
 - DFT> write_atpg_setup pre_norm_scan
-replace
 - DFT> exit

↑
.dofile : setup information
.testproc : procedure file

Outline

- Introduction
- DFTADVISOR
- FASTSCAN**
- Mixed Flow
- Lab

Invoke Fastscan

- Specify scanned Verilog file and ATPG library file
 - `$ fastscan pre_norm_scan.v -verilog -lib l90sprvt.atpg -nogui`

Read Setup Information

- ❑ Read setup information from DFTAdvisor
 - `SETUP> dofile pre_norm_scan.dofile`
- ❑ Enter ATPG mode
 - `SETUP> set_system_mode atpg`
- ❑ Select fault type: stuck, IDDDQ, transition, path_delay, bridge, etc
 - `ATPG> set_fault_type stuck`

Generate Patterns (1/2)

- Use “-auto” option to
 - Suggest the best settings possible to generate the most compact patterns with the highest coverage within the lowest time
 - `ATPG> create_patterns -auto`

Generate Patterns (2/2)

- Without the "auto" option, you can specify your own configurations using these commands
 - ATPG> set_atpg_limits
 - cpu_seconds [integer]
 - test_coverage [real]
 - pattern_count [integer]
 - ATPG> set_abort_limit [integer]
 - ATPG> create_patterns

During ATPG

- ATPG is performed pass after pass

```
Simulation performed for #gates = 3420 #faults = 5542
system mode = ATPG    pattern source = internal patterns
```

```
-----
#patterns  test      #faults  #faults  # eff.   # test    process      RE/AU/AAB
simulated  coverage  in list  detected  patterns patterns  CPU time
deterministic ATPG invoked with comb/seq abort limit = 300/100
---        -
64         87.16%    781      4761     60       60        1.15 sec     0/0/15
---        -
128        95.93%    243      450      51       111       1.23 sec     87/1/91
---        -
192        98.81%    70       143      50       161       2.33 sec
---        -
229        99.73%    15       55       31       192       2.33 sec     117/1/103
---        -
229        99.73%    15       55       31       192       2.58 sec     117/1/103
---        -
229        99.73%    15       55       31       192       2.59 sec     117/1/103
---        -
229        99.73%    15       55       31       192       2.59 sec
```

ATPG Result

- 4 main parts
 - Fault number (#FU)
 - Test/Fault coverage
 - Pattern count
 - Runtime
- Print ATPG statistics report
 - `ATPG> report_statistics`

Statistics Report Stuck-at Faults	
Fault Classes	#faults (total)

FU (full)	6098

U0 (unobserved)	15 (0.25%)
DS (det_simulation)	5409 (88.70%)
DI (det_implication)	540 (8.86%)
UU (unused)	16 (0.26%)
RE (redundant)	117 (1.92%)
AU (atpg_untestable)	1 (0.02%)

Coverage	

test_coverage	99.73%
fault_coverage	97.56%
atpg_effectiveness	99.75%

#test_patterns	192
#simulated_patterns	229
CPU_time (secs)	300.5

View Report

- Display fault information
 - `ATPG> report_faults -all`
- Each fault is associated with a fault class/code

Fault value:
Either 0 (for stuck-at-0)
or 1 (for stuck-at-1)

Fault code

Fault site

```
ATPG> REPort FAults -class
ATPG_UNTESTABLE
0  AU  /IS7/OUT
1  EQ  /IS7/IN
0  EQ  /IS1/en
1  AU  /IS7/OUT
0  EQ  /IS7/IN
1  EQ  /IS1/en
0  AU  /IS4/i1
0  AU  /IS20/en
1  AU  /IS20/en
0  AU  /IS2/en
1  AU  /IS2/en
```

Fault Classes - Full (FU)

- $FU = TE + UT$
- TE: Testable
- UT: Untestable
 - Faults which no pattern can exist to either detect or possible-detect
 - Cannot cause functional failures, so they are excluded from test coverage calculation

Fault Classes - Testable (TE)

- DT: Detected
 - UD: Undetected
 - Faults that cannot be proven untestable or ATPG_untestable
 - Initial class for testable faults
 - AU: Atpg_untestable
 - Due to pin constraint or insufficient sequential depth placed on Fastscan
 - PD: Possible-detected
 - Faults with good-machine value being either 0 or 1 and faulty machine value being X in simulation
-

Fault Classes - Untestable (UT)

- UU: Unused
 - Faults not connected to any circuit observation point
- BL: Blocked
 - Faults blocked by logic on all paths
- TI: Tied
 - Point of the fault value is always same (e.g. AND2 with complementary inputs)
- RE: Redundant
 - Faults undetectable after exhausting all patterns and need dedicated analysis to verify redundancy
 - `ATPG> identify_redundant_faults`

Test Coverage Formula Comparison

□ TetraMAX

$$\text{test_coverage} = \frac{DT + (PT * \text{posdet_credit})}{\text{all_faults} - (UD + AU * \text{au_credit})}$$

Annotations for TetraMAX formula:
- *possible detected* points to *PT*
- *default 50%* points to *posdet_credit*
- *default 0* points to *au_credit*

□ Fastscan

$$\text{test_coverage} = \frac{DT + (PD * \text{posdet_credit})}{\text{testable}} * 100$$

Annotation for Fastscan formula:
- *default 50%* points to *posdet_credit*

$$\text{fault_coverage} = \frac{DT + (PD * \text{posdet_credit})}{\text{full}} * 100$$

$$\text{ATPG_effectiveness} = \frac{DT + UT + AU + PU + (PT * \text{posdet_credit})}{\text{full}} * 100$$

Save Patterns

- Save patterns that are generated via ATPG
- Various formats including binwgl, ctl2005, stil2005, stil999, Verilog, VHDL, wgl, zycad, tstl2, utic
 - ATPG> save patterns pre_norm_scan.pat
-verilog -proc -replace
 - ATPG> save patterns pre_norm_scan_tstl2.pat
-TSTL2 -replace
 - ATPG> exit

Toshiba Standard Tester Interface
Language 2



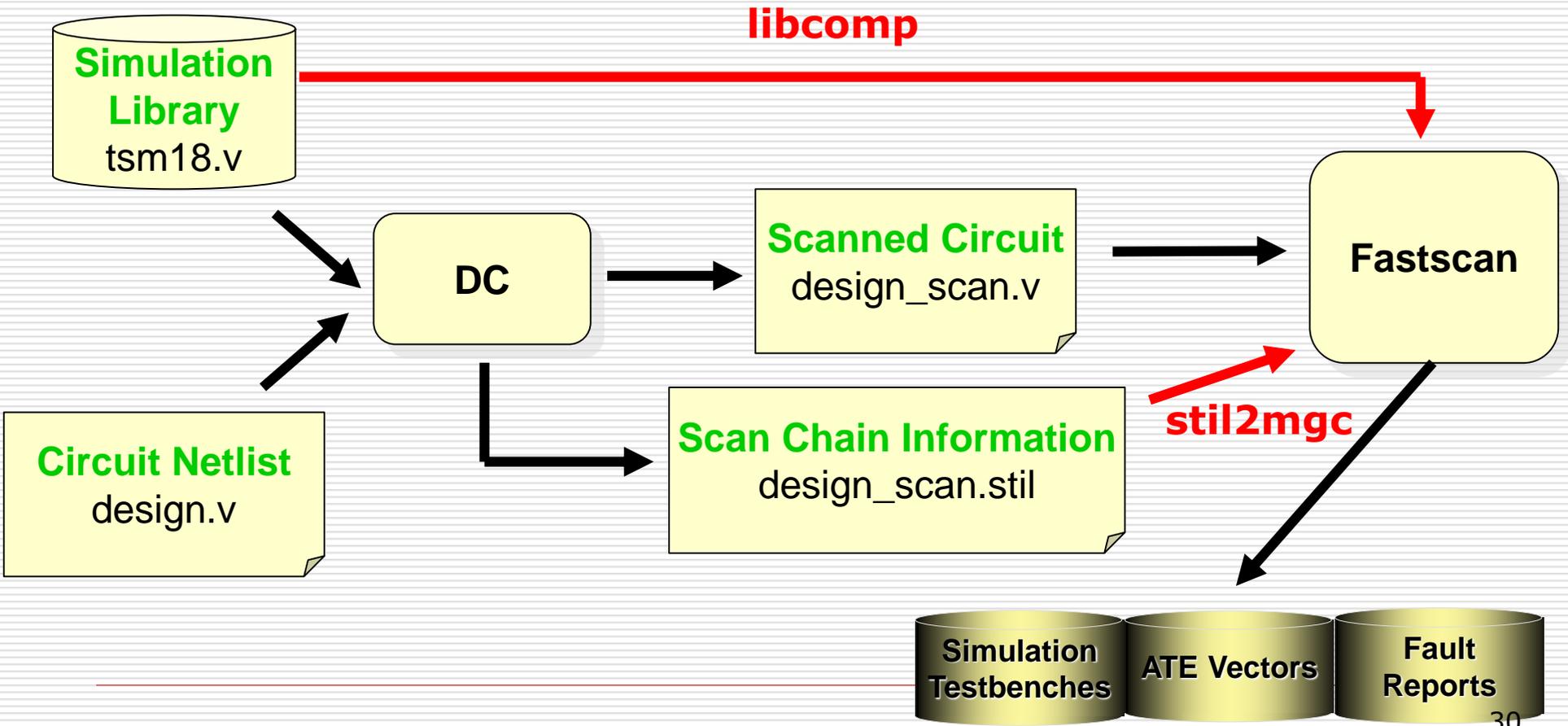
Outline

- Introduction
- DFTADVISOR
- FASTSCAN
- Mixed Flow
- Lab

Mixed Flow

- ❑ Synopsys Design Compiler is way better at mapping from RTL code to gate-level netlist
- ❑ Some practices in industrial project hence adopt
 - Design Compiler to synthesize gate-level netlist and do scan-chain insertion
 - Fastscan to perform ATPG

Input/Output Files



Input Files Required in Mixed Flow

- Library file needs to be converted
 - From .v to .atpg
- The detailed information as to scan chain needs to be converted
 - From .stil to .dofile
- Scanned design

Convert Library File

□ Fastscan uses different library from TetraMAX, so use '`libcomp`' command to convert `I90sprvt.v` to `I90sprvt.atpg`

- `$ libcomp I90sprvt.v`
 - `SETUP> add model -all`
 - `SETUP> set optimization on`
 - `SETUP> set learning on`
 - `SETUP> set sys mode tran`
 - `TRANSLATION> run`
 - `TRANSLATION> write lib I90sprvt.atpg -rep`
 - `TRANSLATION> exit`
-

Convert STIL File

- Use '`stil2mgc`' to convert STIL file from Design Compiler into Fastscan-compatible dofile and test procedure file
 - `$ stil2mgc pre_norm_scan.stil`
 - It generates both files
 - `pre_norm_scan.stil.do`
 - `pre_norm_scan.stil.proc`

Perform ATPG using FASTSCAN

- Read scanned circuit from Design Compiler to perform ATPG
 - `$ fastscan pre_norm_scan.v -verilog -lib l90sprvt.atpg -nogui`
 - `SETUP> dofile pre_norm_scan.stil.do`
 - `SETUP> set_system_mode atpg`
 - `ATPG> create_patterns -auto`
 - `ATPG> report_statistics`

Outline

- Introduction
- DFTADVISOR
- FASTSCAN
- Mixed Flow
- Lab

Lab Goal

- Compare the following during ATPG using the DC+TMAX, DFTA+FS and DC+FS flows
 - Total fault number
 - Test coverage
 - Pattern count
 - Run time (s)
- Run on circuit “pre_norm_noscan.v”, and show a table like next slide

Example of Lab Result

Flow	#Faults	Test Coverage	#Patterns	Run time
DC+TMAX	71298	100%	138	0.83s
DFT+FS	122072	100%	201	0.66s
DC+FS	75208	100%	203	0.51s

References

□ Mentor Graphics

- DFTAdvisor Reference Manual, v8.6_4
- Tessent Scan and ATPG User's Manual, v2014.1

□ Synopsys

- TetraMAX ATPG User Guide, J-2014.09-SP1