Diagnostic Program Generation Tool for ATE Load Boards

Prof. BRUCE C. KIM

Department of Electrical and Computer Engineering

The University of Alabama

Tuscaloosa, Alabama

USA

Problem Statement

- High test throughput requires trouble-free testers and Load Boards
- Board diagnostic programs are required to maintain good Load Boards
- Load Boards are well structured
- **Is it possible to generate diagnostic programs for Boards automatically ?**

Outline

- ❑ Motivation
- ❑ Scope
- ❑ Methodology
- ❑ Software - DIBPro
- ❑ Performance
- ❑ Conclusion

Motivation

- Mixed-signal load boards are highly populated
  - Hundreds of relays, passives and op amps
  - Thermal cycling and usage limit component life
- Diagnostics
  - Essential for test throughput and debug
  - Manual code development time ~ 3 weeks
  - Tester specific and limited re-use
- Early detection of problem saves money

Scope

- Detect catastrophic failures (not parametric)
- Target components
  - Relays
  - Capacitors
  - Resistors
  - Opamps and Buffers
  - Diodes
  - Transistors
- Device/Board independent
- Tester independent
- Generic solution

Methodology

Step 1: Capture interface information

Step 2: Identify individual circuits

Step 3: Determine relay states

Step 4: Classify circuits based on testability

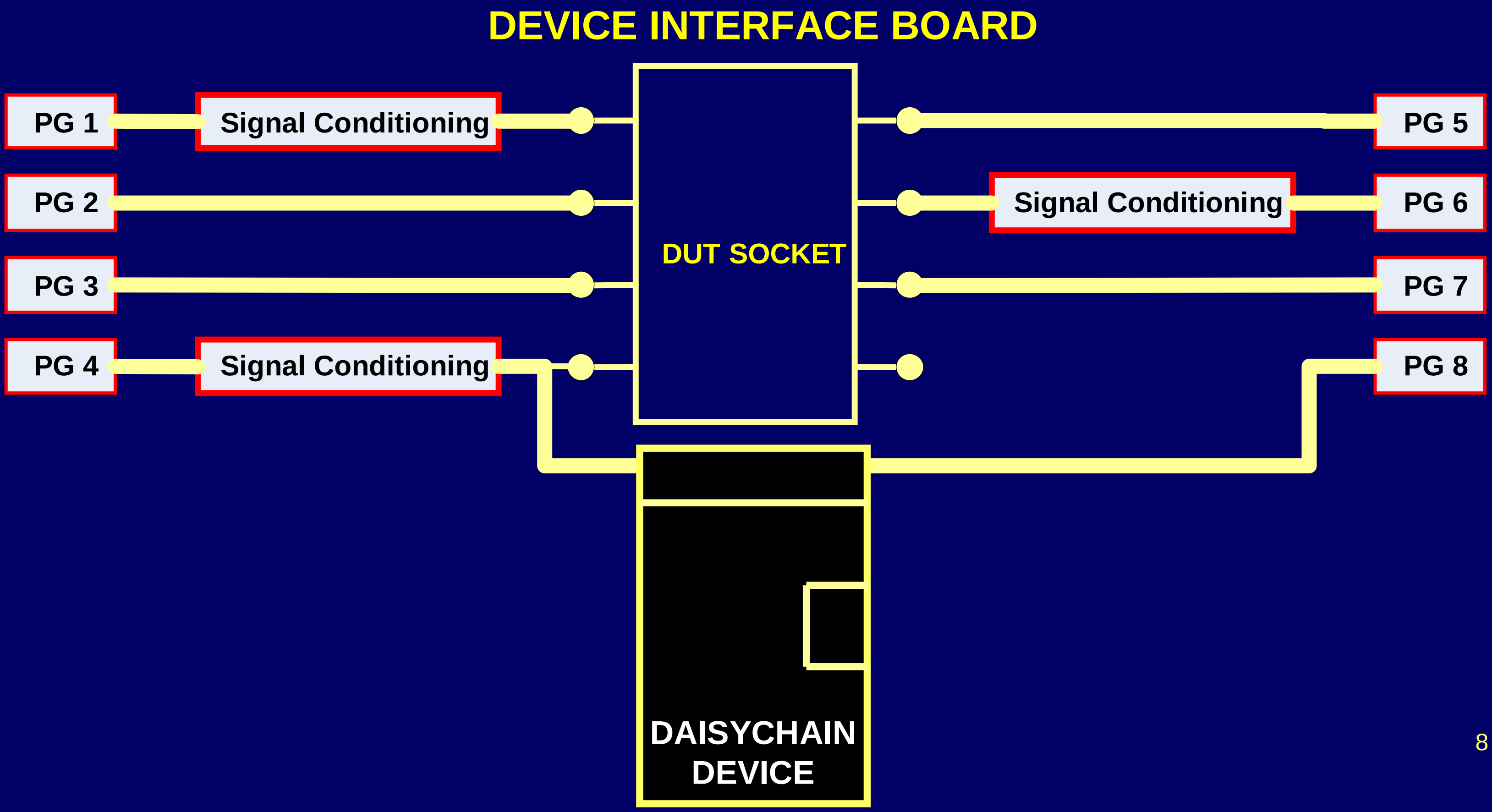
Step 5: Generate test for each circuit

Step 6: Translate pseudo code to tester language

Step 1: Capture Interface Information

❑ Circuits	→ DIB netlist
❑ Component models	→ Model library
❑ Test resources	→ Pinmap
❑ ATE Resource capability	→ Resource file
❑ Socket testability	→ Termination file

What is a Termination Device?



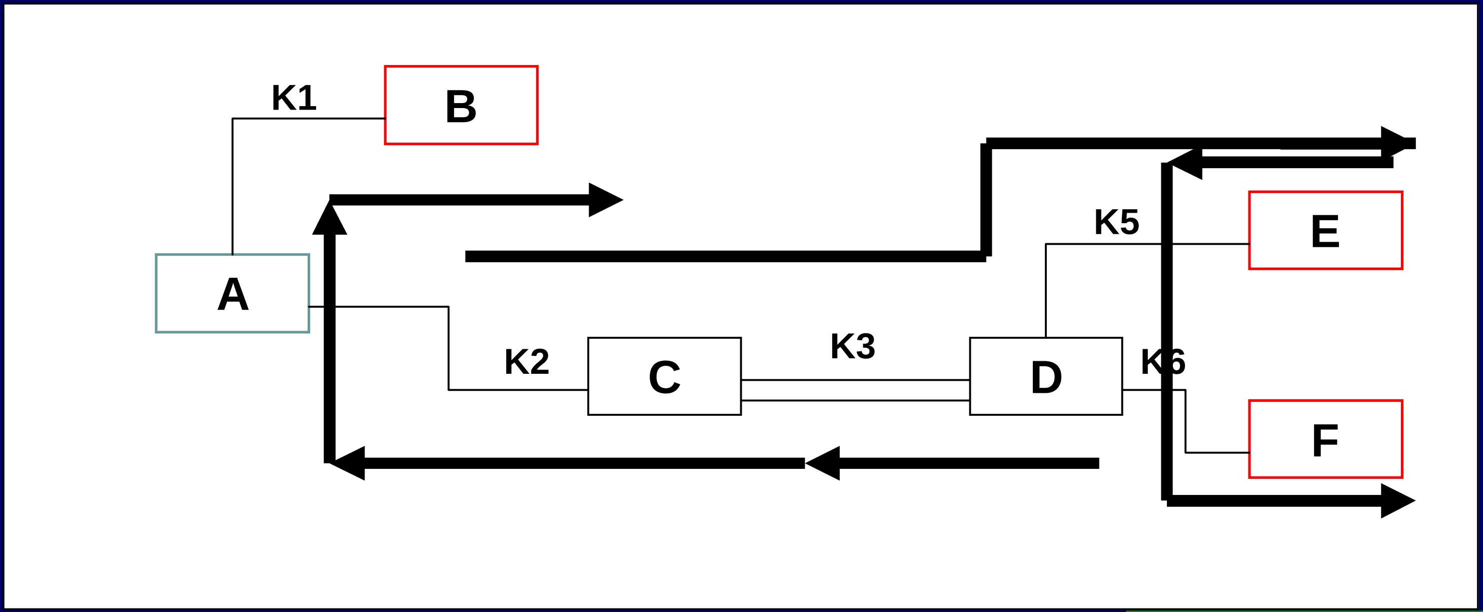
Step 2: Identify Individual Circuits

- Extract all circuits on board, based on connectivity
- Each circuit is independently controllable and/or observable
- Circuit are defined by
  - Start point → Pogo pins
  - Nodes and nets → Components and connections
  - End points → DUT socket pins / Pogo pins
- Connectivity based graph partitioning algorithm

Step 3: Determine Relay States

- 2^k switching states for k relays
- Only few states generate “valid signal path”
- Signal flow based graph partitioning
  - Backtracking algorithm
  - Finds all possible signal paths from start points to different end points by choosing relay states
  - Start/end points have source and/or measure capability
- Build circuit netlist with relay states

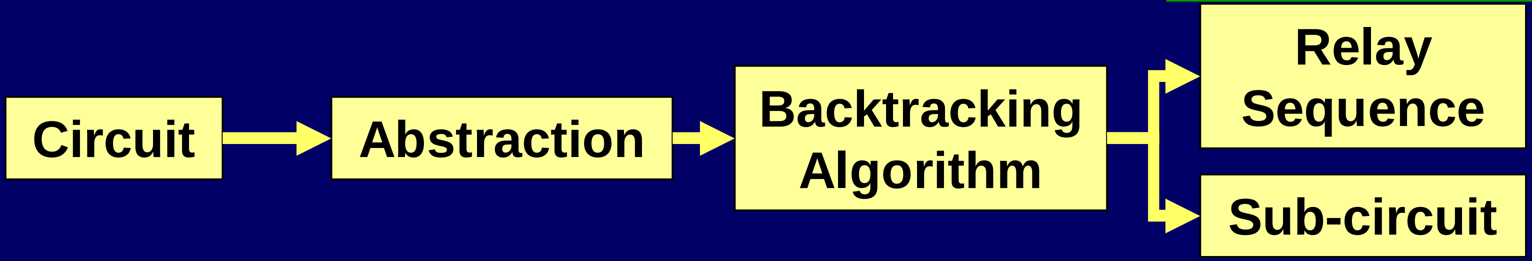
Algorithm



K2, K3*, K5 ON (ACDE)

K2, K3*, K6 ON (ACDF)

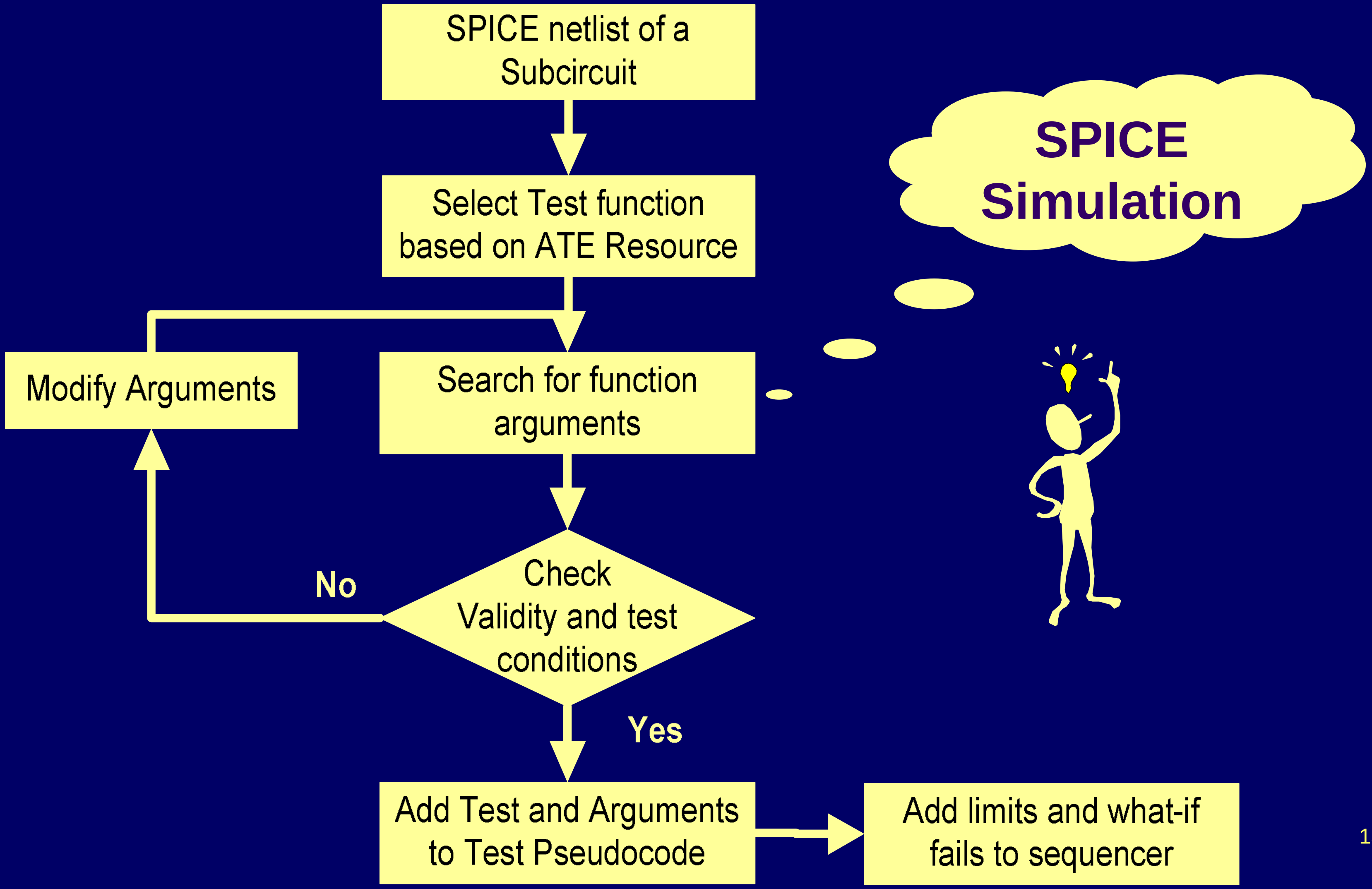
K1 ON (AB)



Step 4: Classify Circuits

- First screen: Circuit testability
  - Availability of test resources at start and end points
  - Availability of component models
- Second screen: Components
  - Capacitors only
  - Resistors only
  - Relays and resistors/capacitors only
- Reduces simulation time
- Build SPICE netlist for the rest of the circuits

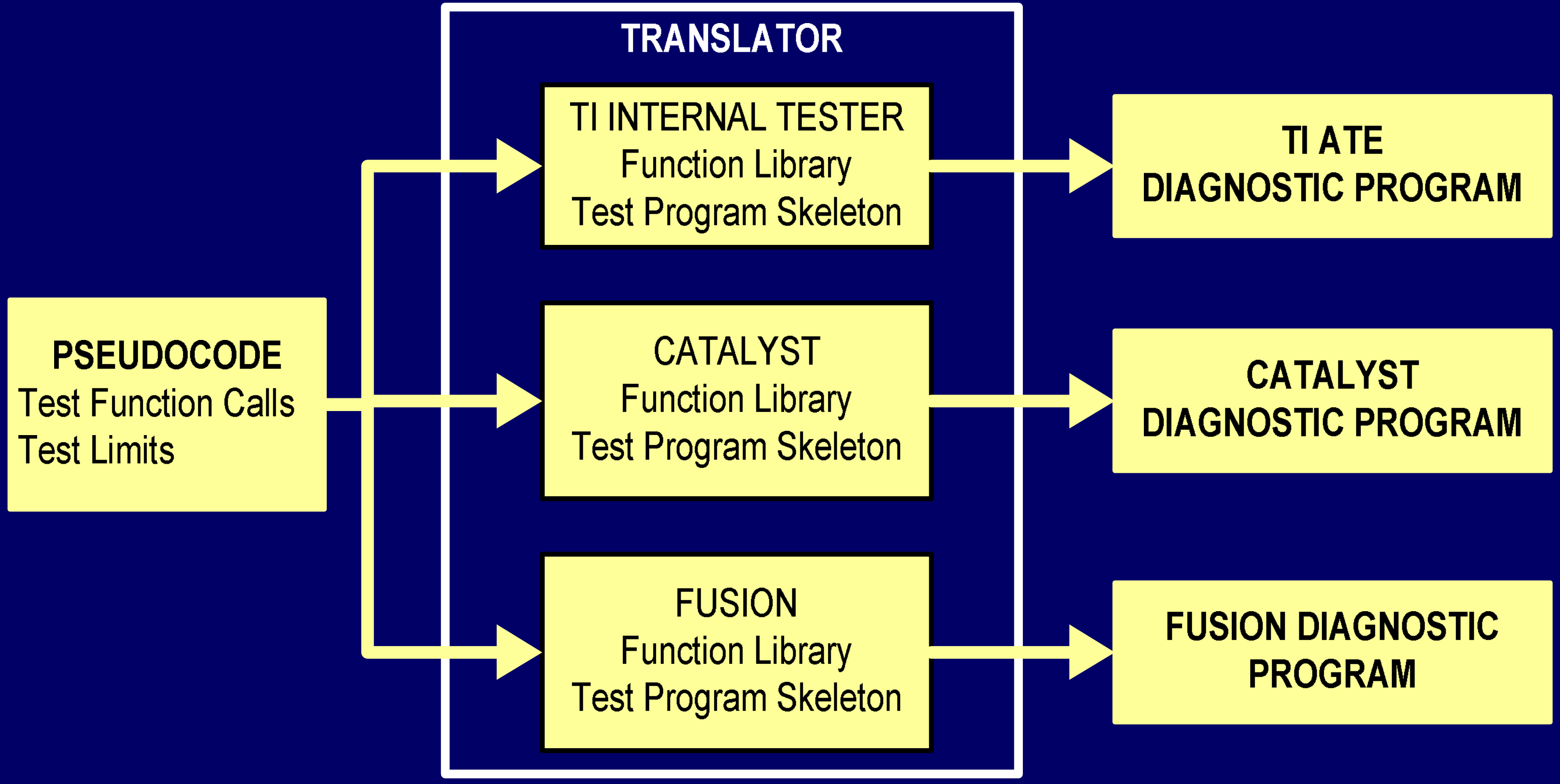
Step 5: Generate Test


SPICE netlist of a Subcircuit
Select Test function based on ATE Resource
Search for function arguments
Modify Arguments
Check Validity and test conditions
No
Yes
Add Test and Arguments to Test Pseudocode
Add limits and what-if fails to sequencer
SPICE Simulation

Function Library

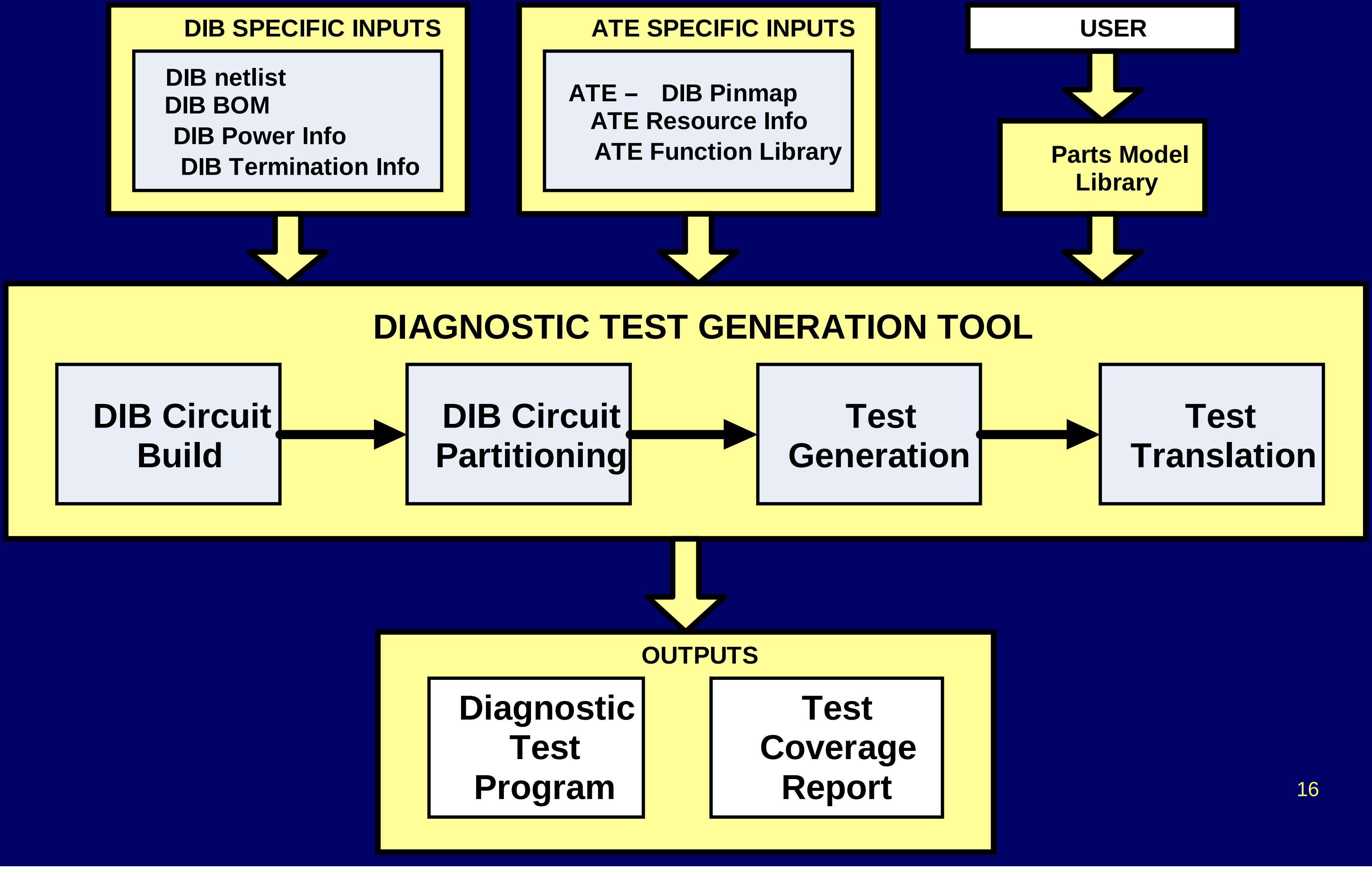
- Force current measure voltage (DC)
- Force voltage measure current (DC)
- Measure DC voltage
- Force current and measure voltage with time Delay
- source single tone
- Digitize (FFT)
- Relay control

Step 6 : Test Translation



Final product can be directly executed on ATE !!!

Software



DIB SPECIFIC INPUTS
DIB netlist
DIB BOM
DIB Power Info
DIB Termination Info
ATE SPECIFIC INPUTS
ATE – DIB Pinmap
ATE Resource Info
ATE Function Library
USER
Parts Model Library
DIAGNOSTIC TEST GENERATION TOOL
DIB Circuit Build
DIB Circuit Partitioning
Test Generation
Test Translation
OUTPUTS
Diagnostic Test Program
Test Coverage Report

Results

Test Pseudocode

```
# CIRCUIT No.115
FCMVDT(PG82-7,0.0016 A,5V,1ms)

# CIRCUIT No.117
FCMV(PG128-8,0.00250 A, 5V)

# CIRCUIT No.119
RELAY(K28_AL1,1)
FCMVDT(PG70-8,0.0424 A,5V,1ms)
RELAY(K28_AL1,0)
RELAY(K27_AL1,1)
FCMV(PG70-8,0.00150 A, 5V)
```

Sequencer

```
115000   >3.9   <4.0   C142,C242
117000   >3.9   <4.0    R361
119000   >2.0   <3.2   R120, K28
119001   >3.2   <4.5   R182, K27
```

Coverage Report

CIRCUIT No.115
TESTED

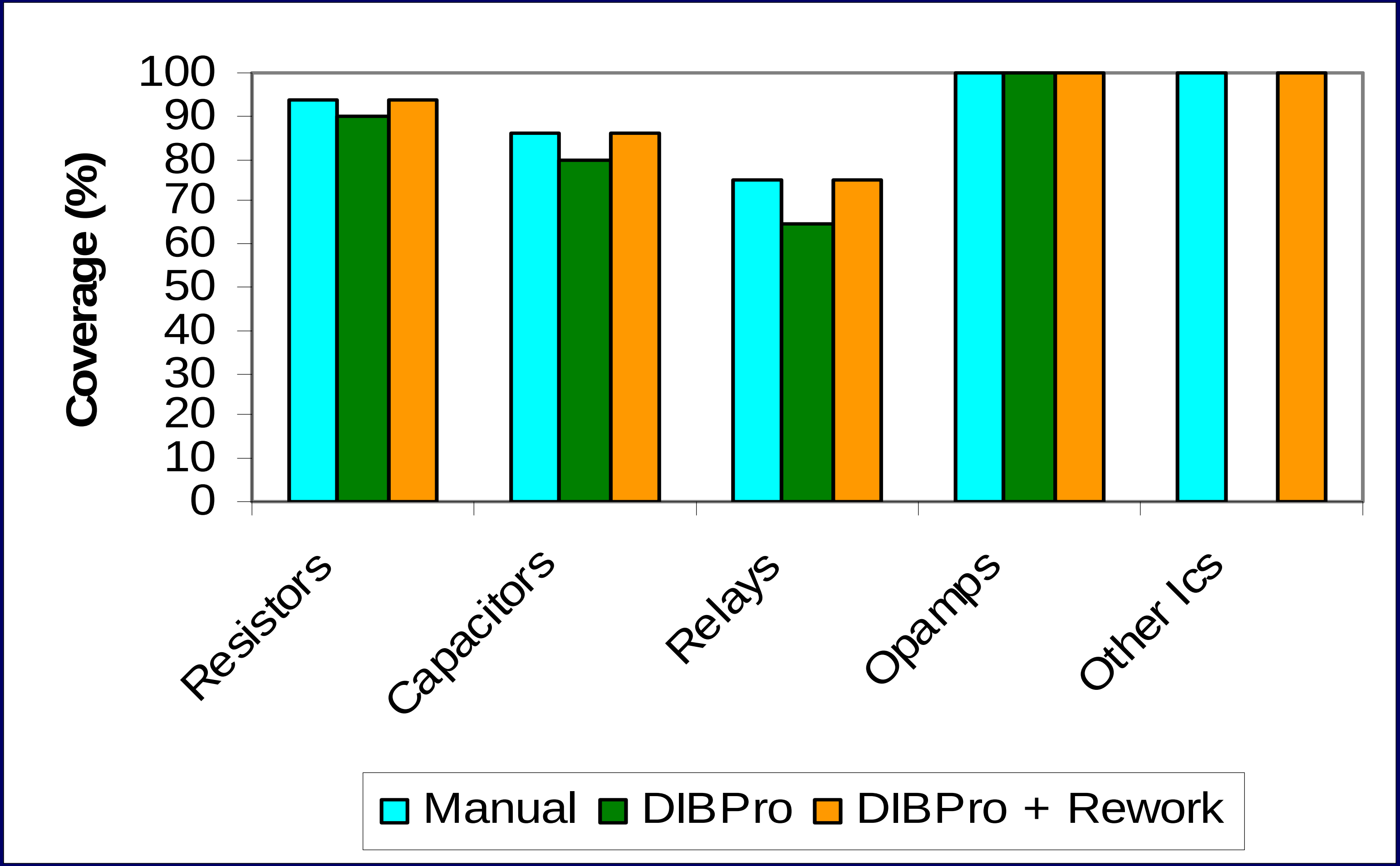
CIRCUIT No.116
NON-TESTABLE component
U24_AL1 of SN75468D found
…

COMPONENTS NOT TESTED
R121, R431, …
C141,C169, …

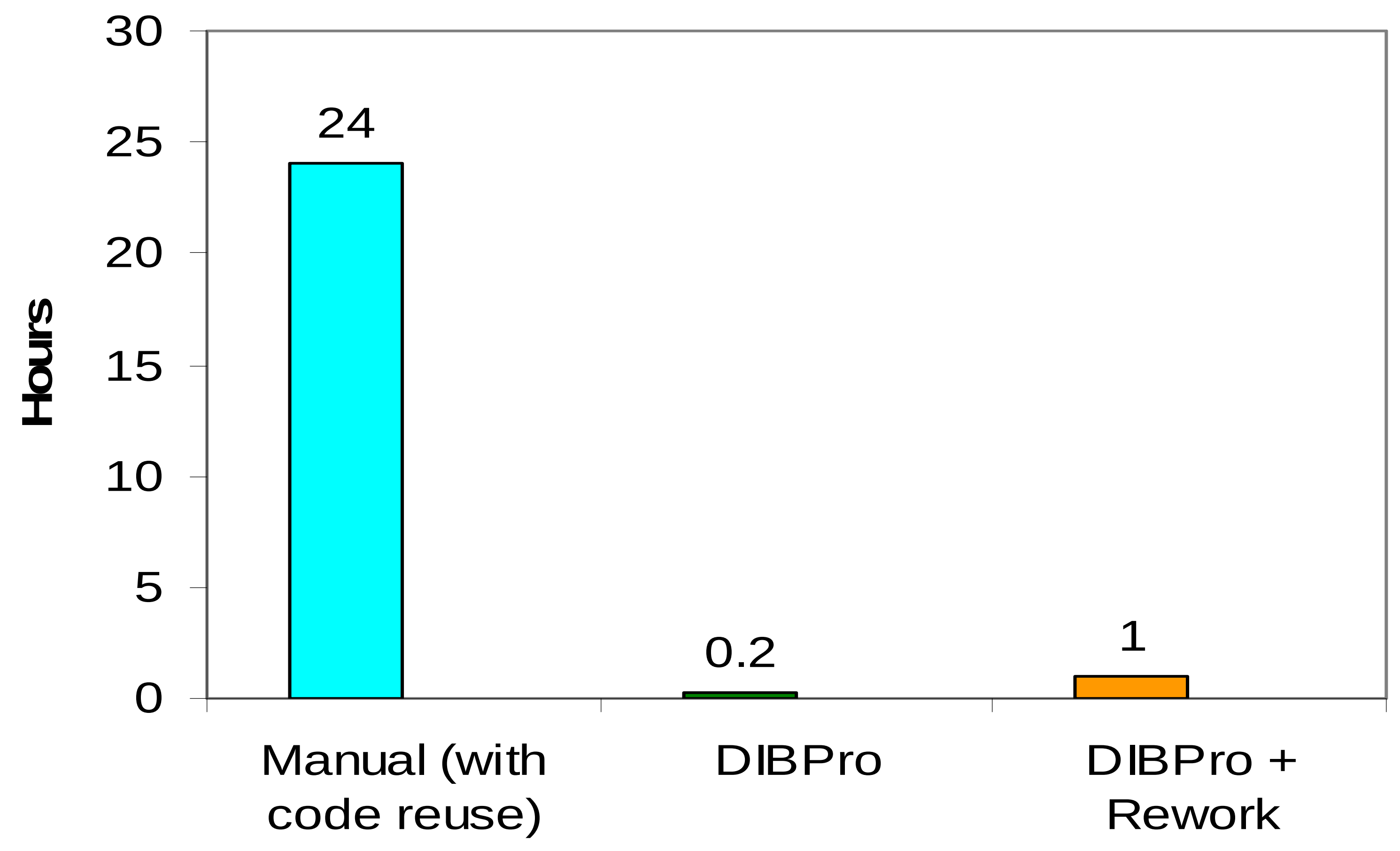
TEST COVERAGE

RESISTORS	210/260
CAPACITORS	195/310
INDUCTORS	0/0
RELAYS	56/80
OPAMPS	8/8
OTHER ICs	0/12

Performance – Test Coverage



Performance – Test Generation Time



Limitations

- Many ICs don’t have a good simulation SPICE model
- Jumpers are tough to comprehend
- Tolerance of components makes it hard to set good limits
- Optional components on netlist (not assembled) cause problems

Impact

- The tool can be used even during board design to determine test coverage
- Reduces hardware verification time
- The same technique can be extended to bench boards with boundary scan (JTAG)

Conclusion

- Yes, it is possible to automatically generate diagnostic program for device interface boards
- The test coverage is very close to that of manual program
- Immense savings in test generation time
  days to minutes

Acknowledgements

TEXAS INSTRUMENTS

SRC Semiconductor Research Corporation