Center for Embedded Systems
An NSF Industry/University Cooperative Research Center

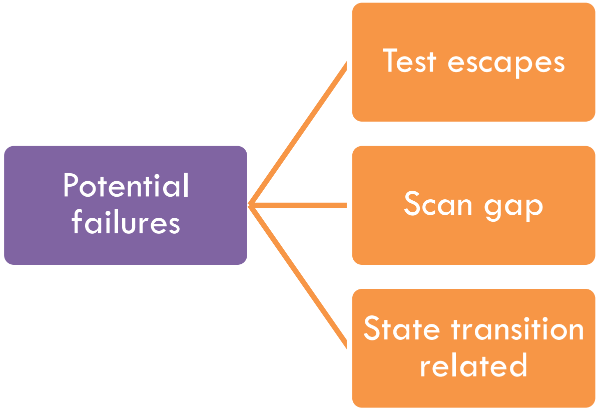
Debugging Errors in Failed Functional Test Sequences

Spyros Tragoudas, SIUC
Themistoklis Haniotakis, SIUC

SIU Southern Illinois University CARBONDALE
NATIONAL SCIENCE FOUNDATION
ASU Ira A. Fulton Schools of Engineering
ARIZONA STATE UNIVERSITY

Project Overview and Description

- A formulation for quick diagnosis of the sources of error in a failing system level test
- Approach the problem at a higher level of abstraction like the Register Transfer (RT) level.
- Model error combinations at the outputs of the hierarchical RT level modules



Potential failures

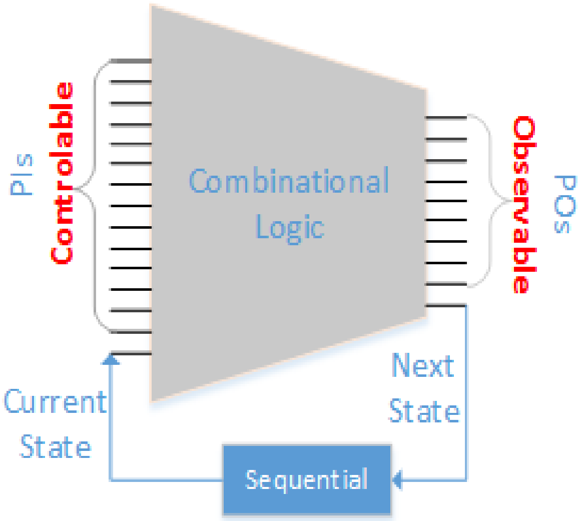
Test escapes

Scan gap

State transition related

Motivation

- Limited controllability/observability of system level tests
- Labor intensive debug for failed system level tests
- High complexity of gate level ATPG



Approach

Novelty

1. Use of function-based methods for input sequence generation
2. Isolation of defects independent of fault model
3. Quick defect isolation due to reduced complexity at the RT level abstraction of the circuit
4. Performs multiple error propagation and justification for a more accurate diagnosis of defects

Benefits

1. Effective techniques to avoid the painstaking manual debug of system-level test failures
2. Quick defect isolation and diagnostic test generation
3. Effort for pattern generation early in product lifecycle

Approach

- Identify the set of all possible input test sequences for a given output functional sequence using function-based methods
- Defect isolation through error analysis on a given set of input and output functional test sequences at the RTL abstraction
- The proposed method generates diagnostic test patterns and provides an error isolation model
- Considers the scenario where a failure may be due to the simultaneous occurrence of multiple errors
- Generated patterns can be applied on ATE or chip level test interfaces such as TAP
- **Approach is different than any existing method at the RTL**

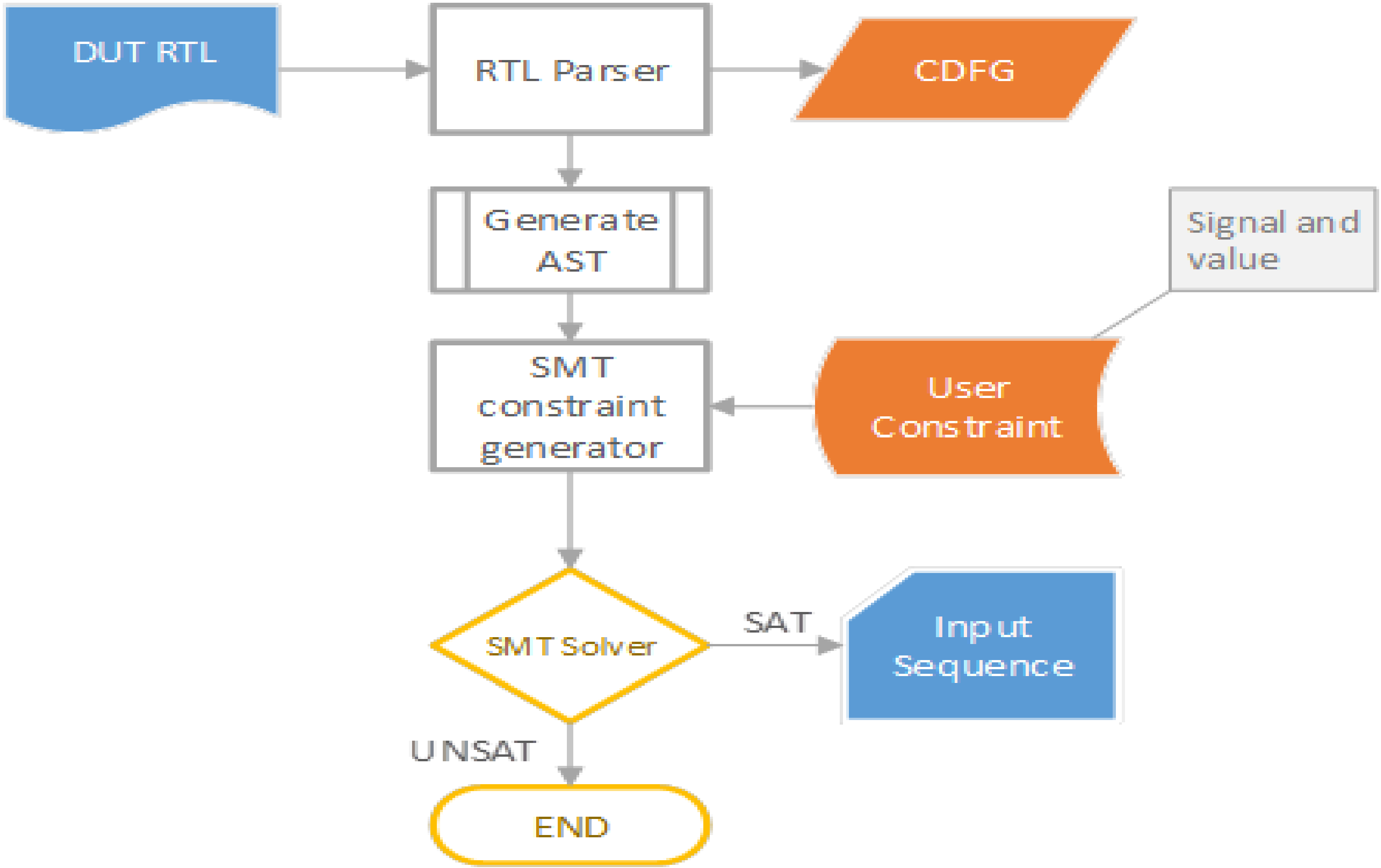
Proposed formulation can be modeled using Satisfiability Modulo Theories (SMT)

Project Tasks/ Deliverables

	Description	Date	Status
1	Implicitly identify input test sequences using SMT	Sep-2015	*Complete*
2	Tool to generate test sequences using SMT **(Phase 1)**	Nov-2015	*Complete*
3	• Development of methodologies to perform defect isolation at the RT level **(Phase 2)** • Implement software tool to automate defect isolation and generate diagnostic test patterns to distinguishably test potential defects **(Phase 2)**	May-2016	*Partially Complete*
4	Procure test cases from member companies and demonstrate the capabilities of the proposed approach **(Phase 3)**	Aug-2016	*Not yet started*

Executive Summary

Phase 1:



Proposed Process Flow

Phase 1:

- Unroll the circuit for a determined bound
  - Based on the longest path in the data dependency tree
- Function-based methods to implicitly generate input test sequences that justify a given output sequence using SMT.

Phase 2:

- Evaluate each of the available input sequences against the erroneous output and identify suspect modules.
- Examine a single module at each hierarchy by modelling errors on its outputs.
- Modelled errors from a potential defect source must justify the observed erroneous output.

Related Work

- Recent work in [1][2] deal with generating functional tests for hard to detect stuck at faults that were not covered by structural tests.
- The approach in [1][2] assumes a fault model and cannot effectively debug arbitrary defects that result in failed system level tests.
- [3] addresses the existence of cases where failing circuits are affected by multiple defects.
- The approach in [3], however, can only be applied to gate level description of circuits and thus, is not scalable for large and complex modern circuits.
- The problem tackled by [5] is very close to the one addressed by the proposed approach.
- The solution in [5], unlike the proposed solution, is based on reachability analysis of states (intractable) and requires on-chip support logic.

Indicative Experimental evidence for Phase 1

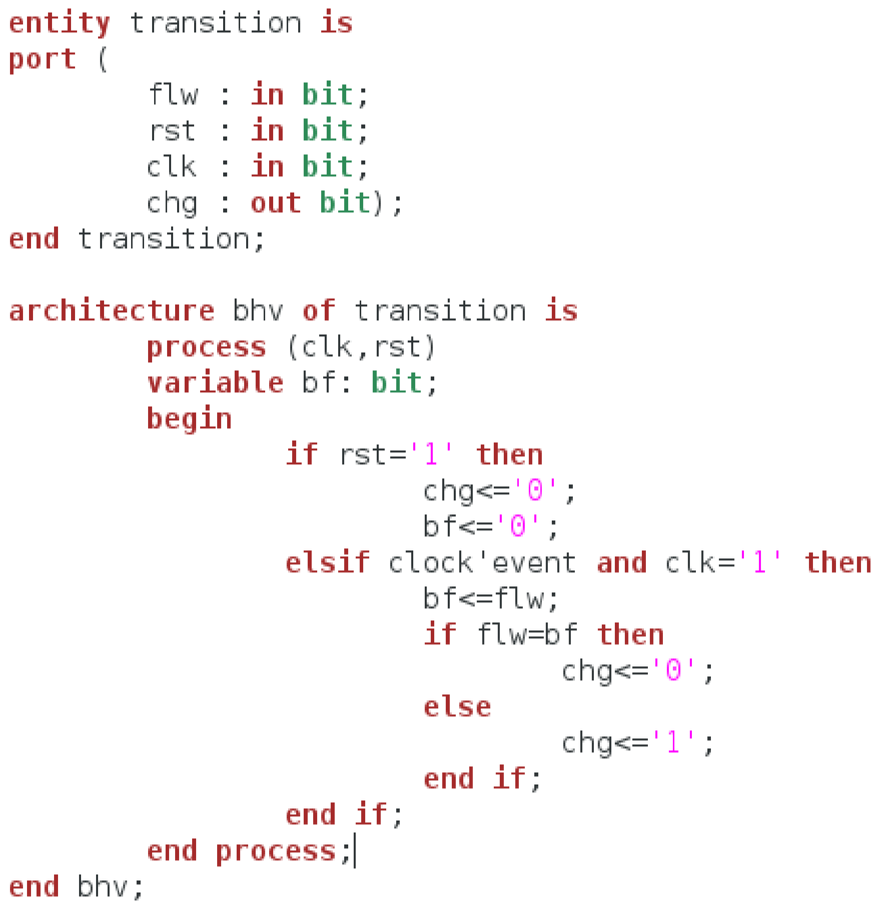
- **Phase 1:** ISCAS '89 Verilog RTL

s35932.v : Parameters that define the size of the circuit

Inputs	Outputs	Flip-flops	Logic Gates
35	320	1728	16065

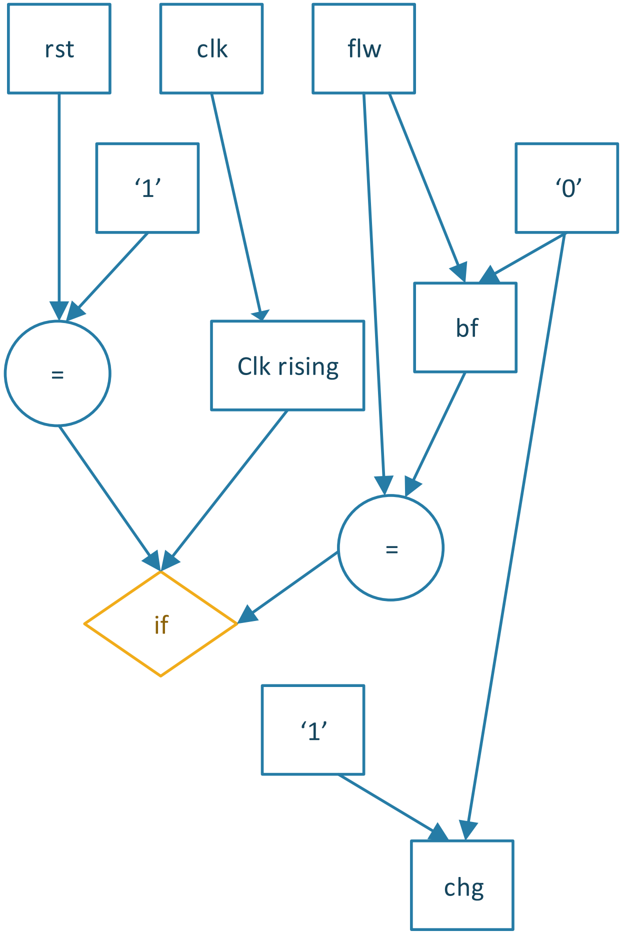
	User Constraints Signal Time Value	Exec. Time in seconds	# Time Frame Expansions
Case 1	● blick_reset_net 1 1 ● DATA_9_31 1 0 ● DATA_9_11 20 1 ● DATA_9_11 30 0 ● DATA_9_11 42 0 ● DATA_9_11 100 1 ● DATA_9_19 160 0 ● DATA_9_9 190 0 ● DATA_9_9 210 0	4935.96	210

RTL to AST



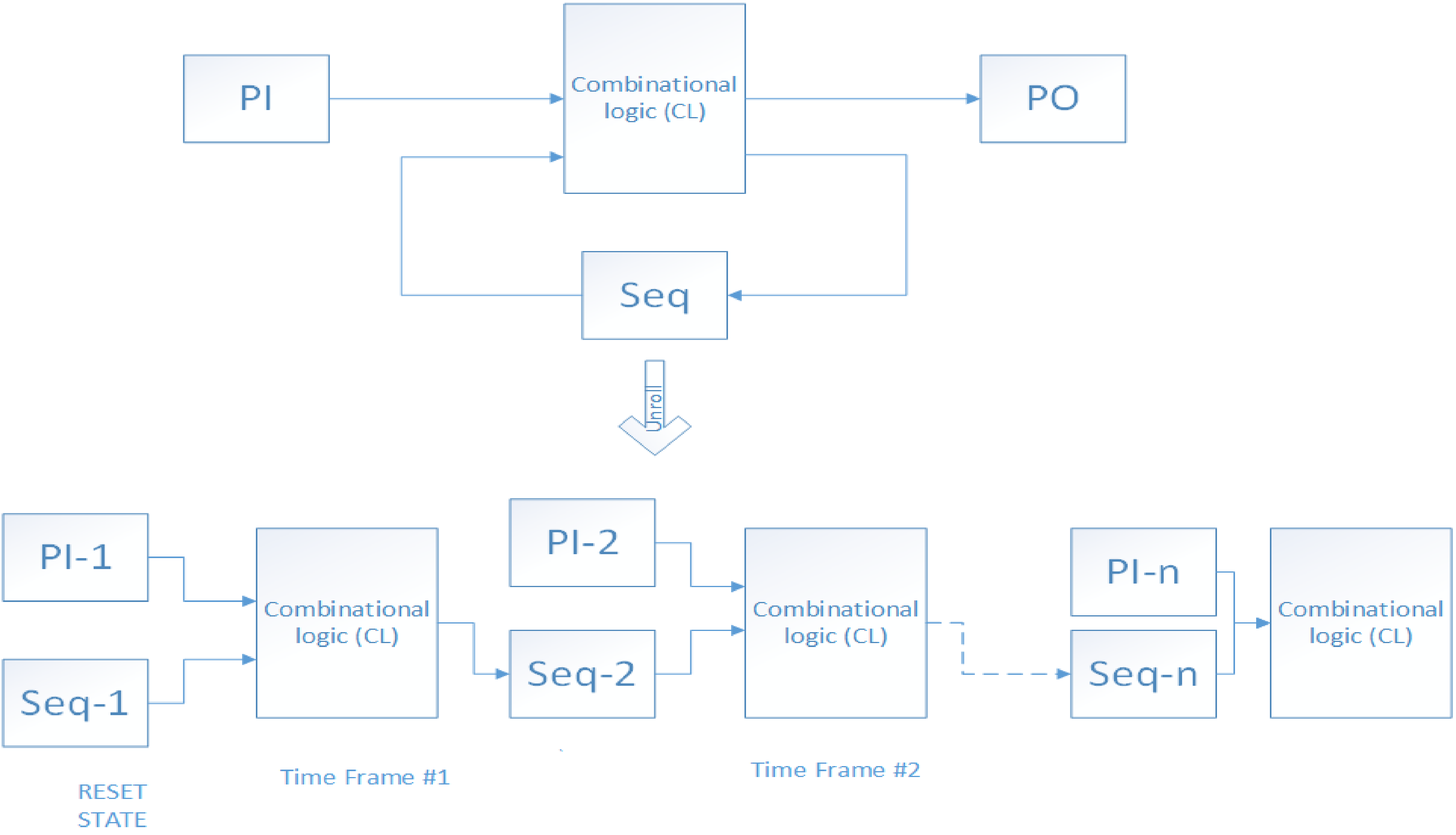
```
entity transition is
port (
        flw : in bit;
        rst : in bit;
        clk : in bit;
        chg : out bit);
end transition;

architecture bhv of transition is
        process (clk,rst)
        variable bf: bit;
        begin
                if rst='1' then
                        chg<='0';
                        bf<='0';
                elsif clock'event and clk='1' then
                        bf<=flw;
                        if flw=bf then
                                chg<='0';
                        else
                                chg<='1';
                        end if;
                end if;
        end process;
end bhv;
```

DUT RTL



Abstract Syntax Tree

Unrolling the DUT for a Given Bound (Number of Time Frames in Expansion)



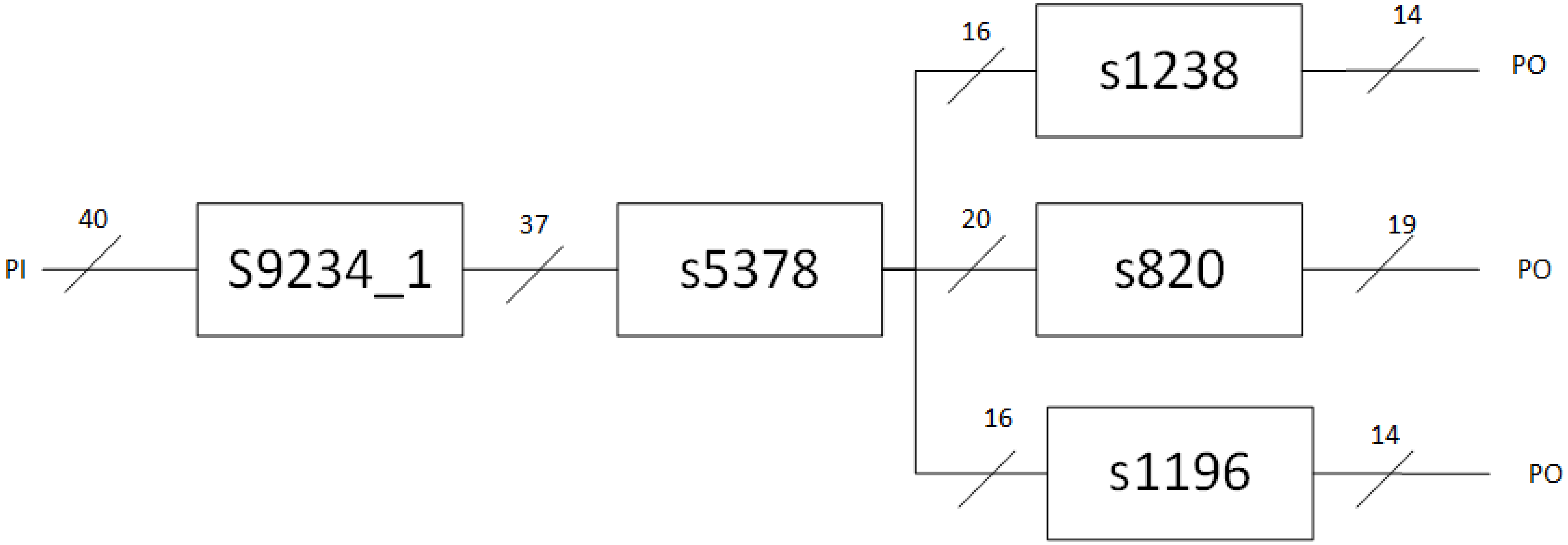
References

1. Prabhu, M.; Abraham, J.A., "Functional test generation for hard to detect stuck-at faults using RTL model checking," *17th IEEE European Test Symposium (ETS), 2012*, vol., no., pp.1,6, 28-31 May 2012
2. Prabhu, M.; Abraham, J.A., "Application of under-approximation techniques to functional test generation targeting hard to detect stuck-at faults," *IEEE International Test Conference (ITC), 2013,* vol., no., pp.1,7, 6-13 Sept. 2013
3. Hongxia Fang; Chakrabarty, K.; Jas, A.; Patil, S.; Tirumurti, C., "Functional Test-Sequence Grading at Register-Transfer Level," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol.20, no.10, pp.1890,1894, Oct. 2012
4. Xiaochun Yu; Blanton, R.D., "An Effective and Flexible Multiple Defect Diagnosis Methodology Using Error Propagation Analysis," *IEEE International Test Conference, 2008. ITC 2008*, vol., no., pp.1,9, 28-30 Oct. 2008
5. Bao Le; Sengupta, D.; Veneris, A.; Poulos, Z., "Accelerating post silicon debug of deep electrical faults," *IEEE 19th International On-Line Testing Symposium (IOLTS), 2013*, vol., no., pp.61,66, 8-10 July 2013

Plan for Phase 2

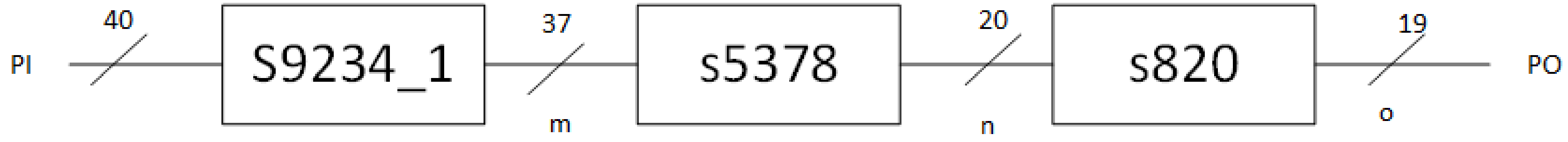
- **Fault Diagnosis :**

Example: Assume s9234.v is faulty (All are ISCAS’89 verilog module)



Example

Error is excited and propagated from output of s9234 → s5378 → s820→PO

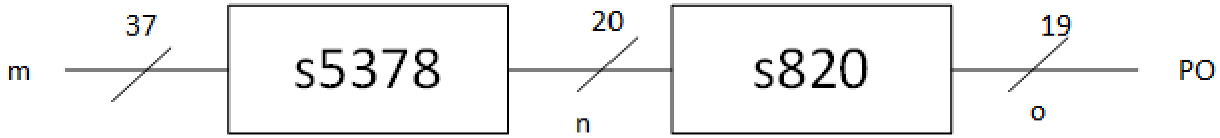


Approach considers an input pattern sequence where faults have been observed at certain time frames

Approach operates one module at a time

For illustration: Assume at most one signal will be faulty at a specific time frame

Remove all modules in the input cone of the fault signal.



Above example assumes defects at s9234.v

Example (assume two time frames in the test pattern sequence)

Condition for the fault to be excited:

Time Frame # 0:

$$\underbrace{\begin{array}{c} \overline{m1[0]}\cdot m2[0]\cdots\cdots m40[0] \\ \text{OR} \\ m1[0]\cdot \overline{m2[0]}\cdots\cdots m40[0] \\ \text{OR} \\ m1[0]\cdot m2[0]\cdots\cdots \overline{m40[0]} \\ \text{OR} \\ m1[0]\cdot m2[0]\cdots\cdots m40[0] \end{array}}_{\text{Time Frame \# 0 Constraint}}$$

Time Frame # 1:

$$\underbrace{\begin{array}{c} \overline{m1[1]}\cdot m2[1]\cdots\cdots m40[1] \\ \text{OR} \\ m1[1]\cdot \overline{m2[1]}\cdots\cdots m40[1] \\ \text{OR} \\ m1[1]\cdot m2[1]\cdots\cdots \overline{m40[1]} \\ \text{OR} \\ m1[1]\cdot m2[1]\cdots\cdots m40[1] \end{array}}_{\text{Time Frame \# 1 Constraint}}$$

Final constraint = Time Frame # 0 Constraint & Time Frame # 1 Constraint.

Example (continued)

- The above constraints must be satisfied along with all observed responses.

If SMT is able to satisfy above constraints, then s9234.v is a suspect module.

Method is capable to return faulty assignment at the output of s9234.v

SSF within s9234 will be verified hierarchically:

Consider the description of s9234

Inject SSFs

Verify SSFs that map to faulty response at s9234 outputs