Reviving Erroneous Stability-based Clock-gating using Partial Max-SAT

Bao Le
Dipanjan Sengupta
Andreas Veneris

University of Toronto

Outline

- Introduction
  - Partial Max-SAT Debugging
  - Clock-gating
- Pre-Processing
  - Components in a Clock-gating Design.
  - Identifying the Erroneous Component(s).
- Debugging
  - Clock-gating Debugging
  - Rectification
- Results and Final Remarks
  - Experimental Results

Outline

Introduction
- Partial Max-SAT Debugging
- Clock-gating

Pre-Processing
- Components in a Clock-gating Design.
- Identifying the Erroneous Component(s).

Debugging
- Clock-gating Debugging
- Rectification

Results and Final Remarks
- Experimental Results

Introduction

Clock-gating

-Popular low-power technique

-ODC and STC-based

STC-based Clock-gating

-Register output is stable

-Complex and susceptible to errors

Debugging

-Common practice is to remove clock-gating

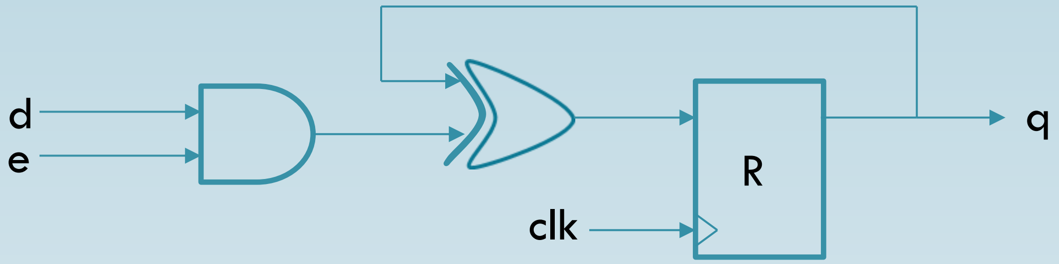
-Power saving is reduced

Contribution:

- Reduce debugging time by using Partial Max-SAT.
- While fixing the error(s) still retain power savings.
- Overall, 12% improvement in runtime and 98% power savings retained.

Introduction: Stability Condition

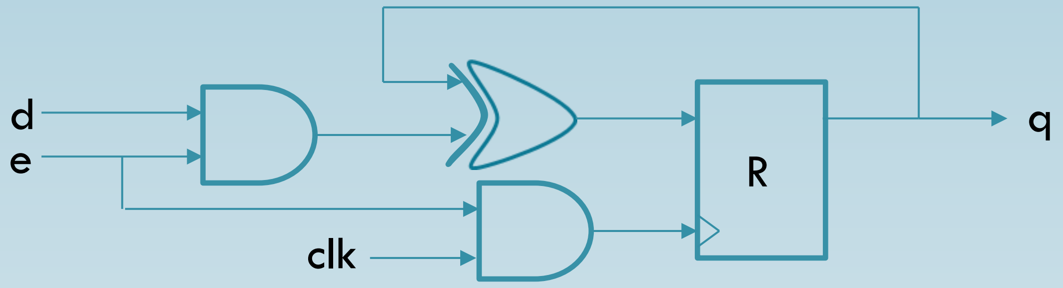
A register is said to be stable when its output does not change for two or more consecutive clock-cycles, i.e $q^t = qt^{-1}$.



R is stable when e = 0

Introduction: STC Clock-gating

- The clock is turned off when e = 0 by using an AND gate.
- e is called an enable signal for clock of R.



ODC

Automatically extracted

Implemented automatically

STC

Hard to identify

Implemented manually

Introduction: Partial Max-SAT

Max-SAT

- Given an unsatisfiable CNF instance, Max-SAT solver returns an assignment that maximizes the number of satisfiable clauses.

Partial Max-SAT

- In Partial Max-SAT, the instance is organized as a set of hard clauses which must be satisfied and a set of soft clauses which may or may not be satisfied.
- The goal is to satisfy all hard clauses while maximizing the number of satisfied soft clauses.

Introduction: Design Debugging

Given an erroneous circuit, a counter example of length k, and error cardinality N.

Goal: Return shortlist of potentially buggy gates (solutions)

- Gates that can be modified to fix counter-example

SAT-based Design Debugging

Fault diagnosis and logic debugging using Boolean Satisfiability [Smith, Veneris, Ali, Viglas-TCAD2005]

Introduction: Partial Max-SAT Debugging

Design debugging problem can be encoded as a Partial Max-SAT instance.

- In this instance, constraints are encoded as hard clauses while the transition relation T_{en} are encoded as soft clauses.
- The **complement** of a Partial Max-SAT solution presents a set of clauses that correspond to potentially erroneous gates.
- All solutions with cardinality $\leq N$ are found by using blocking clauses.

Outline

Introduction
- Partial Max-SAT Debugging
- Clock-gating

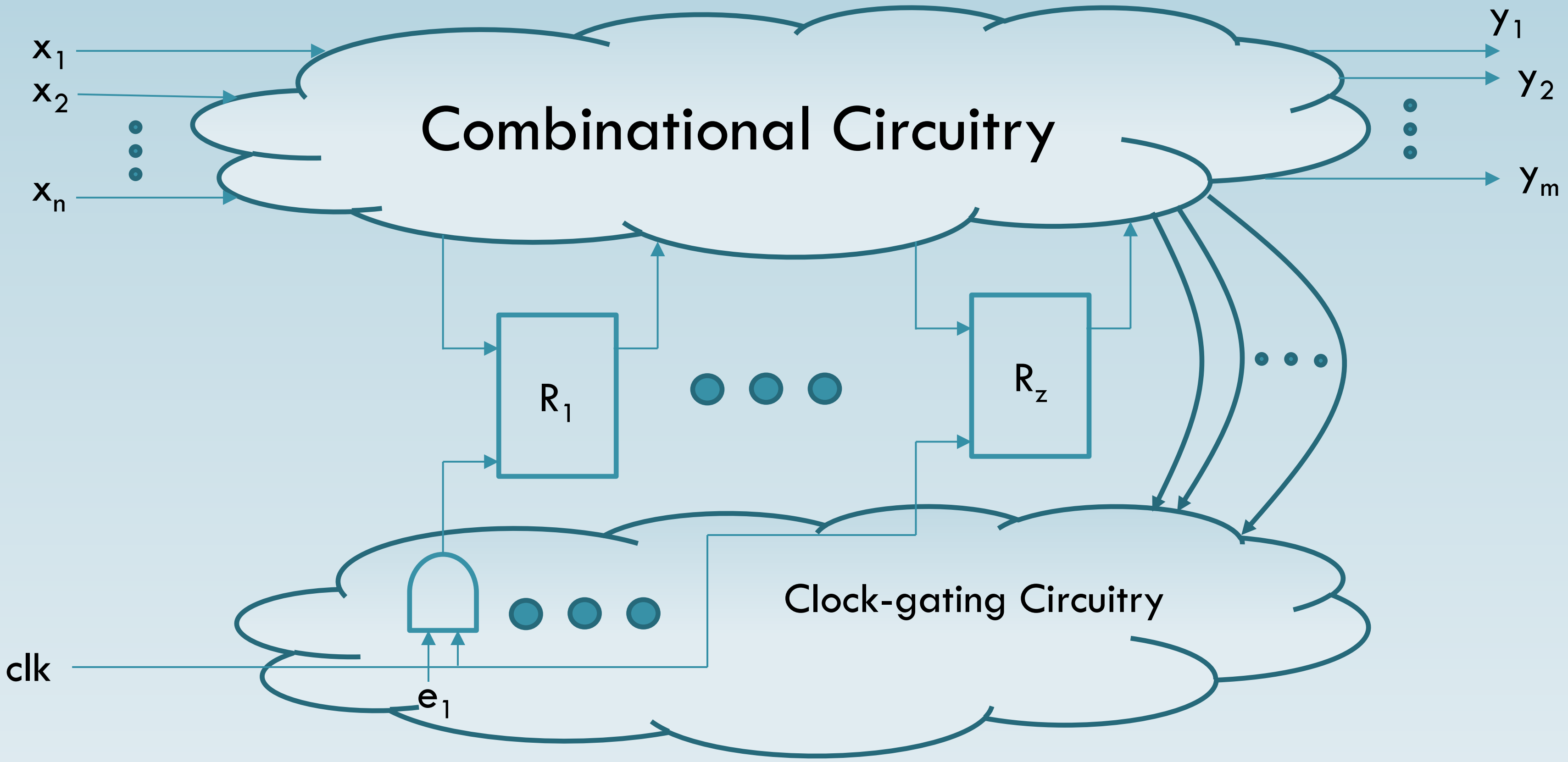
Pre-Processing
- Components in Clock-gating design.
- Identifying the Erroneous Component(s).

Debugging
- Clock-gating Debugging
- Rectification

Results and Final Remarks
- Experimental Results

Clock-gating Design

- Typically, there are three components in a clock-gating design C.



Clock-gating Design

Clock-gating Circuitry

- Includes all nodes that are only used for clock-gating (i.e only used to compute enable signals).

Combinational Circuitry

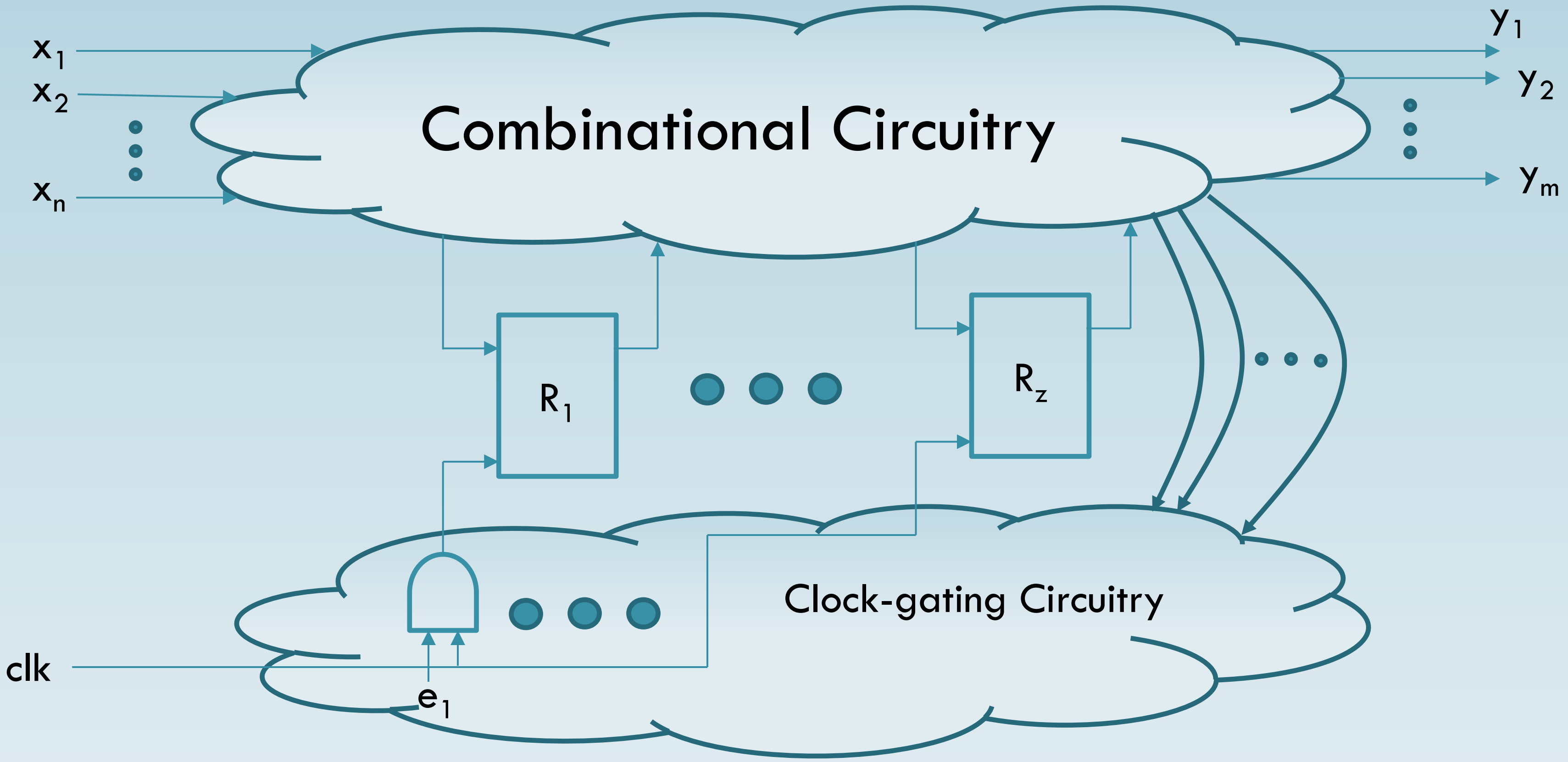
- All other nodes that are not registers.

State Elements

- All registers in the design.

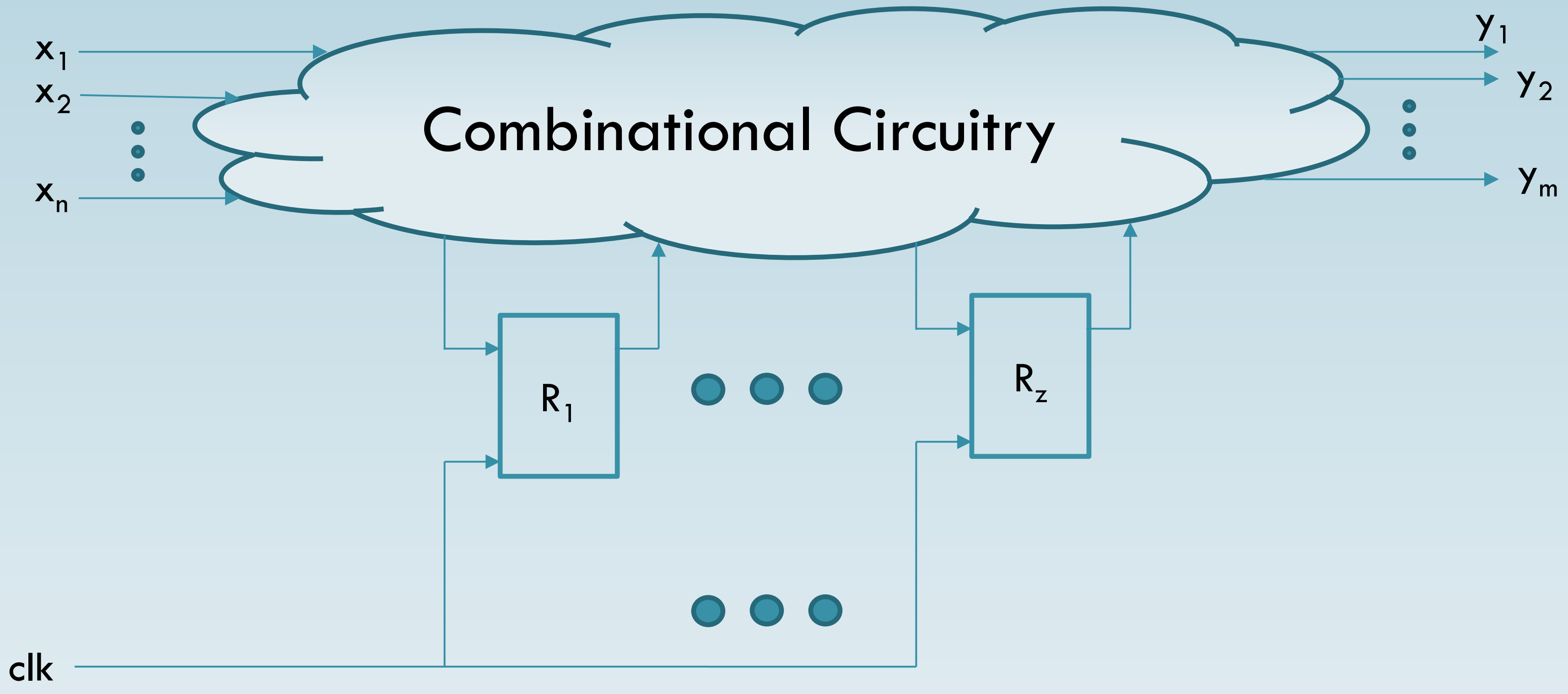
Pre-processing

- We first construct an enhanced version of C called C_{en} by removing all clock-gating.



Pre-processing

- We first construct an enhanced version of C called C_{en} by removing all clock-gating.



Pre-processing

Given the same trace of length k, if C and C_{en} produce the same states and outputs for all time-frame $t \leq k$, the erroneous gate is in the combinational circuitry.

After the combination circuitry is identified as erroneous component:

- All other components are bug-free (**reliable**) components
- These components can be set as hard clauses.
- The increase in number of hard clauses helps reducing the complexity of the Partial Max-SAT instance.

Erroneous Component Identification

Construct C_{en} from C by removing all clock-gating and connect clk to all register clocks.

↓

Simulate C and C_{en} using the counter-example

↓

If the output and state of C and C_{en} at all time-frame are the same, the erroneous gate is in the combinational circuitry

Outline

Introduction
- Partial Max-SAT Debugging
- Clock-gating

Pre-Processing
- Components in a Clock-gating Design.
- Identifying the Erroneous Component(s).

Debugging
- Clock-gating Debugging
- Rectification

Results and Final Remarks
- Experimental Results

Partial Max-SAT Debugging

Encode $Debug$ as a CNF instance. The instance is unsatisfiable since C is erroneous

Using information from the pre-processing step, all clauses that belong to reliable components are set as hard clauses.

The Max-SAT solver finds all solutions with cardinality $\leq N$ by using blocking clauses.

To increase the number of hard clauses in each call to the Partial Max-SAT solver, we also apply sliding window technique.

Rectification

Designer usually removes clock-gating to fix the error(s) introduced during clock-gating implementation.

- This is due to stringent time-to-market and finding error at gate-level is an arduous task.
- However, this is undesirable as it reduces power savings

Clock-gating circuitry is usually small in number of gates involved and constructed from a limited set of gates

- We derive a dictionary-based rectification technique for nodes in clock-gating circuitry.
- Our fix corrects the erroneous behavior presented by the counter-example while maintaining certain level of power saving in the design.

Potential Fixes

It must correct the erroneous behavior presented by the counter-example

- This can be verified by simulating the circuit after applying the fix.

It must not consume more power than removing clock-gating

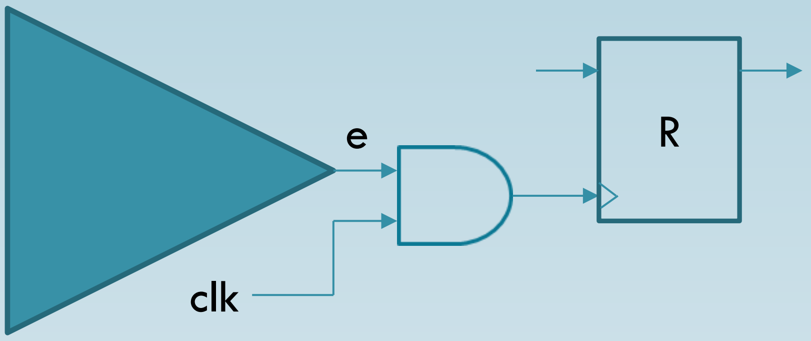
- This is verified by using power estimation tool.

It must not introduce new error(s)

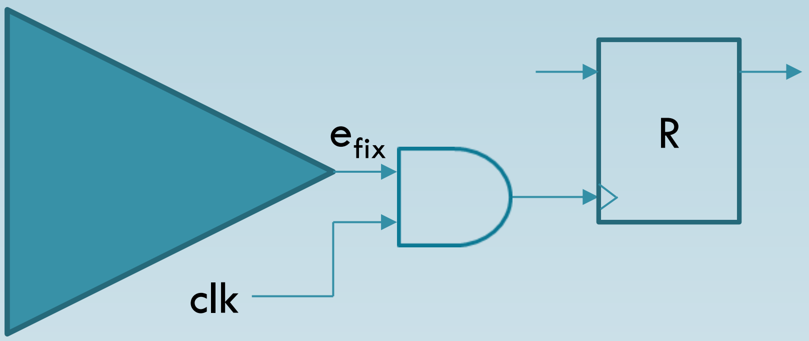
- This can be done by using Sequential Equivalent Checking (SEC). However, this method is costly.
- We present a light-weight verification method for this property.

Rectifications (cont.)

Recall that the clock of a clock-gated register is controlled by enable signal “e”. The following property ensures that no new error(s) is introduced:



Erroneous enable signal



Enable signal after the fix is applied

$$\boxed{e = 1 \rightarrow e_{fix} = 1}$$

Overall Algorithm

1
- Run pre-processing step to identify reliable components.

2
- Encode the problem in CNF and solve it as a Partial Max-SAT instance.
- Find all solutions with cardinality $\leq N$

3
- Rectification is run for erroneous gate in clock-gating circuitry.
- Only potential fixes are returned to the user.

Outline

Introduction
- Partial Max-SAT Debugging
- Clock-gating

Pre-Processing
- Components in a Clock-gating Design.
- Identifying the Erroneous Component(s).

Debugging
- Clock-gating Debugging
- Rectification

Results and Final Remarks
- Experimental Results

Experimental Results

- Platform: i5 3.1Ghz, 8GB memory, 2 hour time-limit.
- Benchmarks: 8 ISCAS-89 and 7 ITC-99 circuits. For each, several bugs are injected to generate debugging instances.
- We run each instance with and without preprocessing step. Rectifications are computed for clock-gating gates.
- We compare to a state-of-the-art Max-SAT-based debugger [Chen, etal-TCAD'10]

Experimental Results (cont.)

Instance Name	Preprocessing time (s)
s382_1	<1
s298_1	<1
[illegible]	<1
[illegible]	<1
s9234_1	<1
s1423_1	64
s838_1	1
s420_1	1
b03_1	1
b04_1	<1
b05_1	4
b07_1	1
b08_1	<1
b09_1	<1
b12_1	1

Preprocessing overhead is neglectable.

Experimental Results (cont.)

The increase in the number of hard clauses reduces the complexity of the problem.

12 % reduction in debugging time is observed.

Instance Name	w/o Prepro(s)	With Prepro (s)	HC Improv(x)	Runtime Improv (%)
		2	9.5	33.3
		10	19.8	23
		20	6.1	9
		9	5	0
s9234_1	467	378	1.9	19
s1423_1	TO	TO	36.5	-
s838_1	TO	2135	17.2	∞
s420_1	TO	TO	15.4	-
		24	8.5	31.4
		105	4.5	8.7
		161	2.9	1.8
b07_1	1336	1282	11.3	4
b08_1	127	126	6	0.7
b09_1	240	230	10.4	4.1
b12_1	1665	1571	72.5	5.6

Experimental Results (cont.)

Instance Name	Orig(microW)	w/o cg(microW)	With rect	Improv (%)
s382_2	8.8	13.6	9.0	95.8
s2		5.5	5.34	22.2
s3		6.02	5.8	100
s349_2	5.8	6.0	5.8	100
s9234_2	69.7	69.9	69.7	100
s1423_2	38.1	40.7	NA	-
s838_2	14.1	14.4	14.36	13.3
s420_2	7.45	8.33	7.46	98.8
b03_2	12.47	14.99	13.9	43.2
b04_2	57.03	62.9	57.03	100
b05_2	9.8	12.4	9.8	100
b0		18.3	15.8	100
b0		11.2	10.1	97.3
b0		15.3	15.16	70
b12_2	52.07	52.4	NA	-

Overall, 80% of the power-savings are retained.

Our rectification technique is able to retain all power-savings in most cases.

Summary

Overview

- A SAT-based debugging methodology for clock-gated circuit is introduced, includes a preprocessing, a debugging and a rectification step.
- The debugging time is reduced and the engineers do not have to sacrifice power-savings to correct the error(s).

Future Work

- Improve the preprocessing step.
- Develop ODC condition debugging.

Questions/Discussions