Multi-Mode Trace Signal Selection for Post-Silicon Debug

Min Li and **Azadeh Davoodi**
Department of Electrical and Computer Engineering
University of Wisconsin-Madison

WISCAD
Electronic Design Automation Lab
http://wiscad.ece.wisc.edu/

Outline

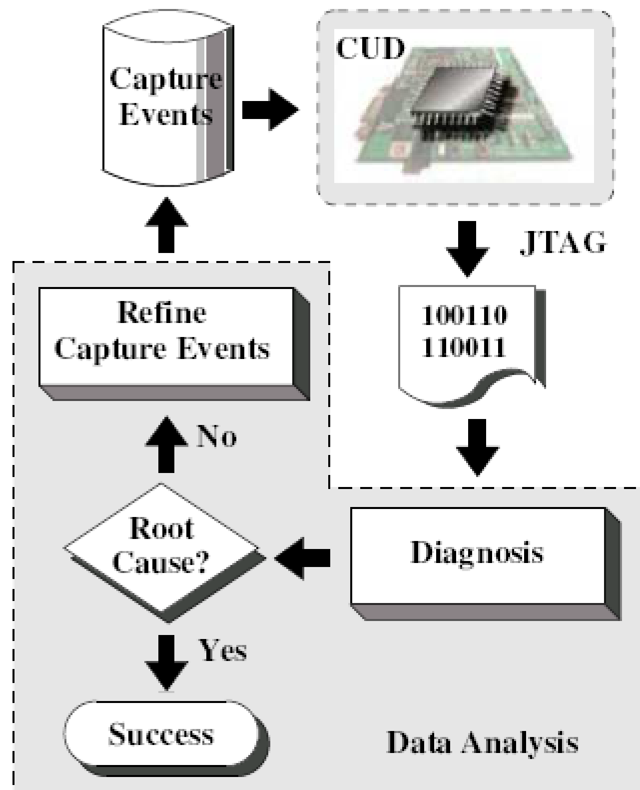
- Preliminaries
  - Overview of post-silicon debug (PSD) using trace buffers
  - Review of restoration process and previous works
- Introduction to restoration considering control signals and operation modes
- Motivation of multi-mode trace selection (MMTS)
- An iterative MMTS algorithm
  - New metrics for trace signal selection
  - Mode merging and iterative selection process
- Experimental results

Post-Silicon Debug

- Real-time operation of a few manufactured chips with real-world stimulus
- Involves finding errors causing malfunctions
  - Fix through multiple rounds of Silicon Stepping/Revision
- Has become significantly time-consuming and expensive
  - Tight Time-to-Market requirement
  - Formal verification and simulation tools do not scale as technology scales
  - Poor visibility inside the chip

Debug using Trace Buffers

- Use trace buffer technology
  1. Trigger an event in the CUD
  2. Real-time capture traces of a few selected flipflops through on-chip buffer
  3. Extract and analyze
- Reveals internal states at real-time operation
- Off-chip state restoration

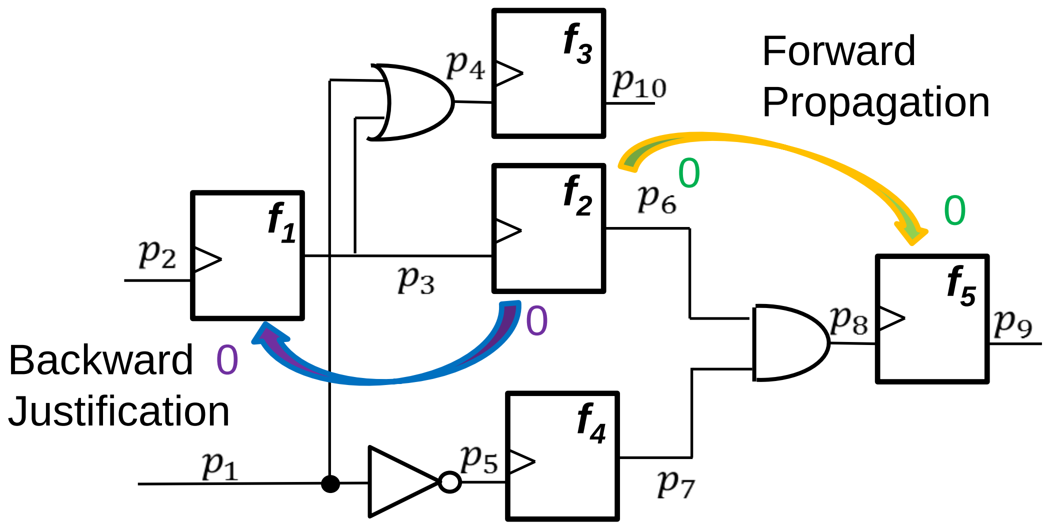


[Figure from Yang, et al DATE09]

Trace Signal Selection Problem

- Limitation
  - Small trace buffer width (8~32 bits) and depth (1K~8K clock cycles)
  - Limited on-chip white spaces
- Automated trace selection
  - Needs to select a subset of flipflops to trace such that the visibility of internal signals is maximized
  - Based on algorithms rather than hints and experiences

Restoration Using Traced Signals



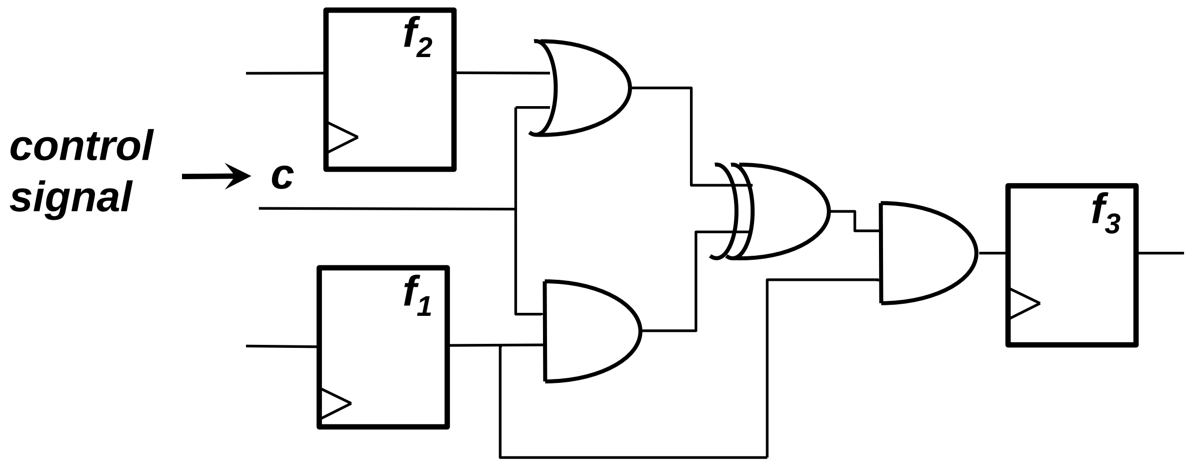
State Restoration Ratio (SRR):

$$\frac{\text{\# tot. FF restored}}{\text{\# tot. FF traced}}$$

e.g. in this example 11/4=2.75

DFF\Cycle	0	1	2	3	4
F1	1	1	0	X	X
F2	0	1	1	0	X
F3	X	1	1	X	X
F4	X	X	X	X	X
F5	X	0	X	X	0

Control Signals and Modes



- Examples of control signals include signals for mode selection, scan enable, power gating and clock gating, encryption, etc.
- n control signals result up to 2^n number of modes
- A single mode is defined as each control signal taking a constant value “0” or “1” throughout the debugging process
- “Multi-mode” refers to the combination of all single modes

Need to Consider Multiple Modes

	SRR^0	SRR^1
$SMTS^0$	17.0	4.3
$SMTS^1$	14.3	8.2

- Case study of S38584
  - Ran single mode trace selection (SMTS) procedure (*of Li & Davoodi [DATE'13]*) twice, for modes 0 and 1, to select two different sets of trace signals
  - For each solution evaluated the SRR for mode 0 and 1
- Observations
  - SRR of SMTS solution in each mode is higher in that mode
    - For example $SMTS^0$ is higher for solution of $SMTS^0$compared to $SMTS^1$
  - Therefore solving SMTS for one mode may result in poor restoration in the other modes. This can be a problem during the debugging process since the operation mode when a bug occurs is not a-priori known

Previous Works

- Previous works give procedures for solving the trace selection problem for a single operation mode
  - *Ko & Nicolici [DATE'08]*
  - *Liu & Xu [DATE'09]*
  - *Prabhakar & Xiao [ATS'09]*
  - *Basu & Mishra [VLSI'11]*
  - *D. Chatterjee [ICCAD'11]:* pure simulation with backward elimination
  - *Li & Davoodi [DATE'13]:* a hybrid algorithm combining fast metric and accurate simulation
- Two major drawbacks of applying single mode trace selection algorithms for multi-modes
  - Traces selected for different modes may have little in common
  - Traces selected for one mode may not be good for other modes

Contributions

- A new metric and problem definition when considering multiple modes
  - Multi-mode State Restoration Ratio (MSRR)
  - Multi-Mode Trace Selection problem (MMTS)

- Algorithms for solving MMTS including
  - A procedure to reduce the number of modes by merging the modes with “similar” restoration maps
  - A procedure based on perturbing an initial single-mode optimized solution (selected from a suitable “start” mode) to improve the restorability over all the modes
  - Our procedure is fast and has a non-greedy nature

Multi-mode Trace Selection Problem

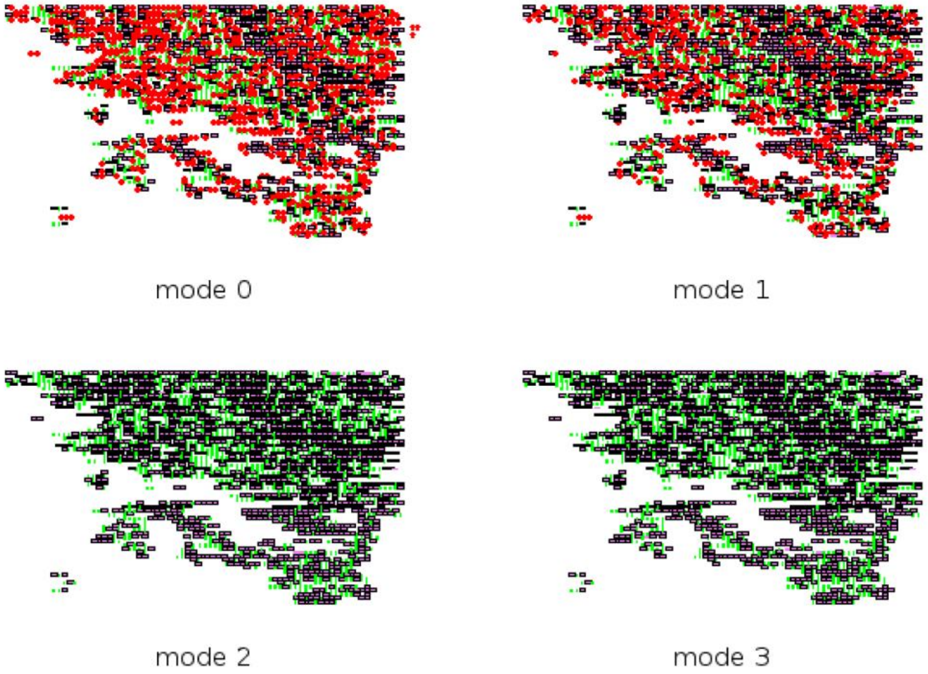
- Multi-mode State Restoration Ratio (MSRR)
  - Defined as summation of state restoration ratios (SRRs) of different modes obtained from a given set of selected trace signals

$$MSRR = \sum_{m=1}^{M} SRR^{m}$$

- Multi-mode Trace Selection problem (MMTS)
  - Given a trace buffer of size $B \times N$, and a set of control signals defining M operation modes, the Multi-mode Trace Selection (MMTS) problem selects B flipflops, in order to maximize MSRR over a debugging window of N cycles.

Mode Merging: Motivation

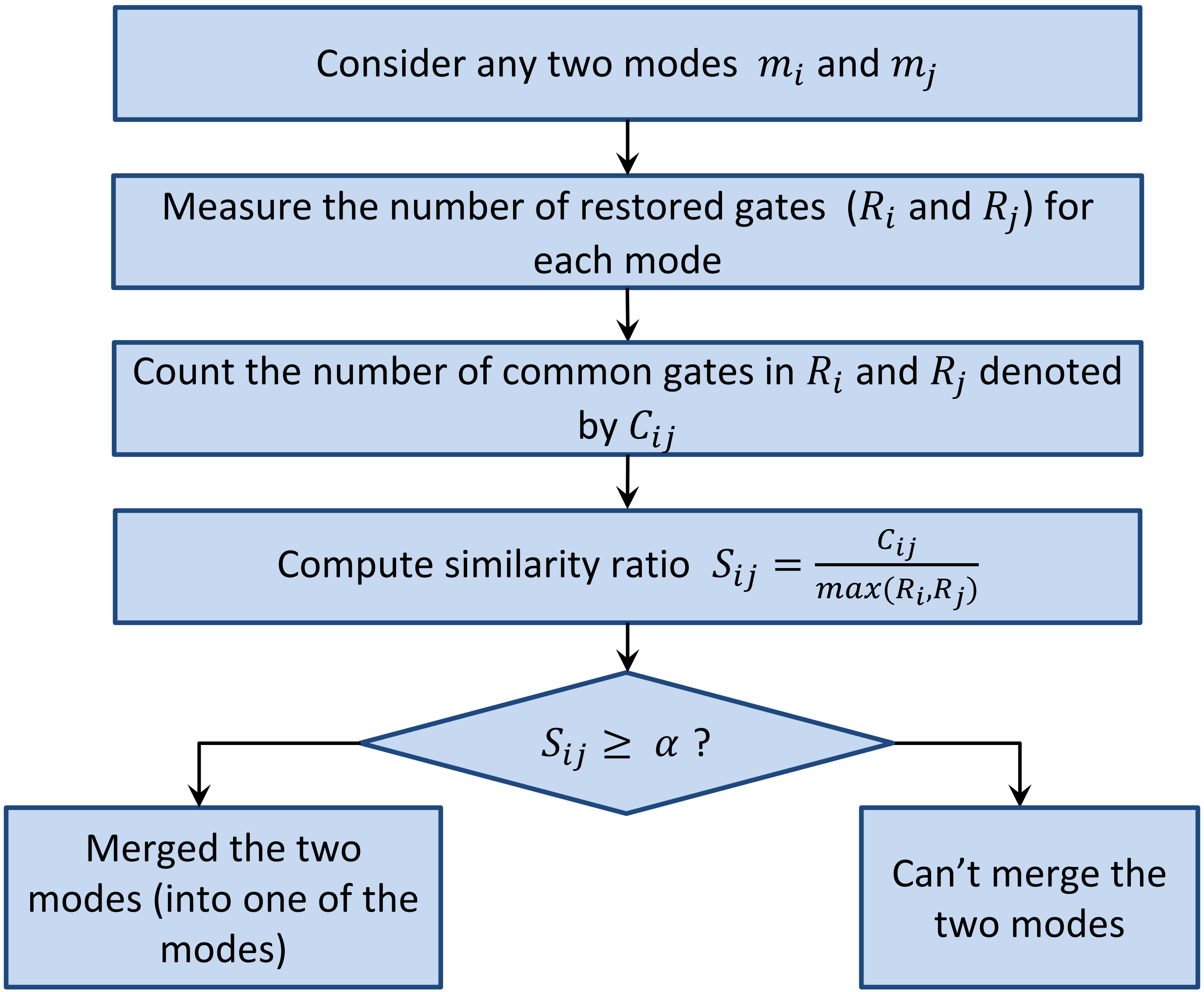
- For S35932 we plotted four restoration maps when each of its four operation modes are set to the corresponding values (when no trace signal is selected yet)



- In each restoration map
  - Green pixel: gate restored to 0
  - Black pixel: gate restored to 1
  - Red pixel: unrestored gate

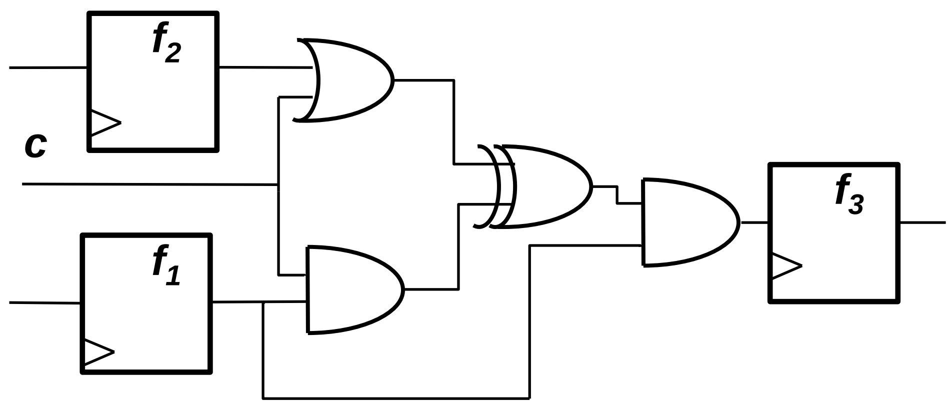
- Observations
  - Modes with similar restoration maps can be merged into a single mode
  - In this case, modes 0 and 1 can be merged, so is modes 2 and 3

Mode Merging: Procedure



MMTS Procedure: Metrics

- $L_{f^v}^m$: **reachability list in mode m**
  - The set of flipflops which can be restored solely by flipflop *f* when *f* takes value *v* ("0" or "1") and control signals take constant values corresponding to mode *m*
  - Example: $L_{f_3^0}^{c=0} = \emptyset \quad L_{f_3^1}^{c=0} = \{f_1, f_2\} \quad L_{f_3^0}^{c=1} = \emptyset \quad L_{f_3^1}^{c=1} = \{f_1\}$



MMTS Procedure: Metrics

- $\boldsymbol{d^m_{i,f^v}}$: **restoration demand in mode m**
  - $d^m_{i,f^v} = \min\left(1 - r^m_i, a^m_{f^v}\right), \forall i \in L^m_{f^v}$
    - r^m_i: restoration rate of flipflop *i* in mode *m* using the traces selected so far
    - $a^m_{f^v}$: rate that flipflop *f* takes value *v* in mode *m*
  - Approximates how much of the restoration of flipflop *i* can be provided by flipflop *f* when *f* takes value *v*
    - $1 - r^m_i$: how much more is needed for full restoration of *i*
    - $a^m_{f^v}$: how much *f* can offer to the restoration of *i*

MMTS Procedure: Metrics

- $\boldsymbol{W_f^m}$: **Impact Weight in mode *m***
  - The impact weight reflects how much flipflop *f* can contribute to restoring the untraced flipflops in its reachability list $L_{f^v}^m$

$$W_f^m = \sum_{v=0,1} \sum_{\forall i \in L_{f^v}^m} d_{i,f^v}^m$$

- $\boldsymbol{MW_f}$: **Multi-mode impact weight of flipflop *f***
  - Measures the contribution of *f* to MSRR if it is selected as a trace signal

$$MW_f = \sum_{m=1}^{M} W_f^m$$

IteM: Iterative Multi-mode Trace Selection

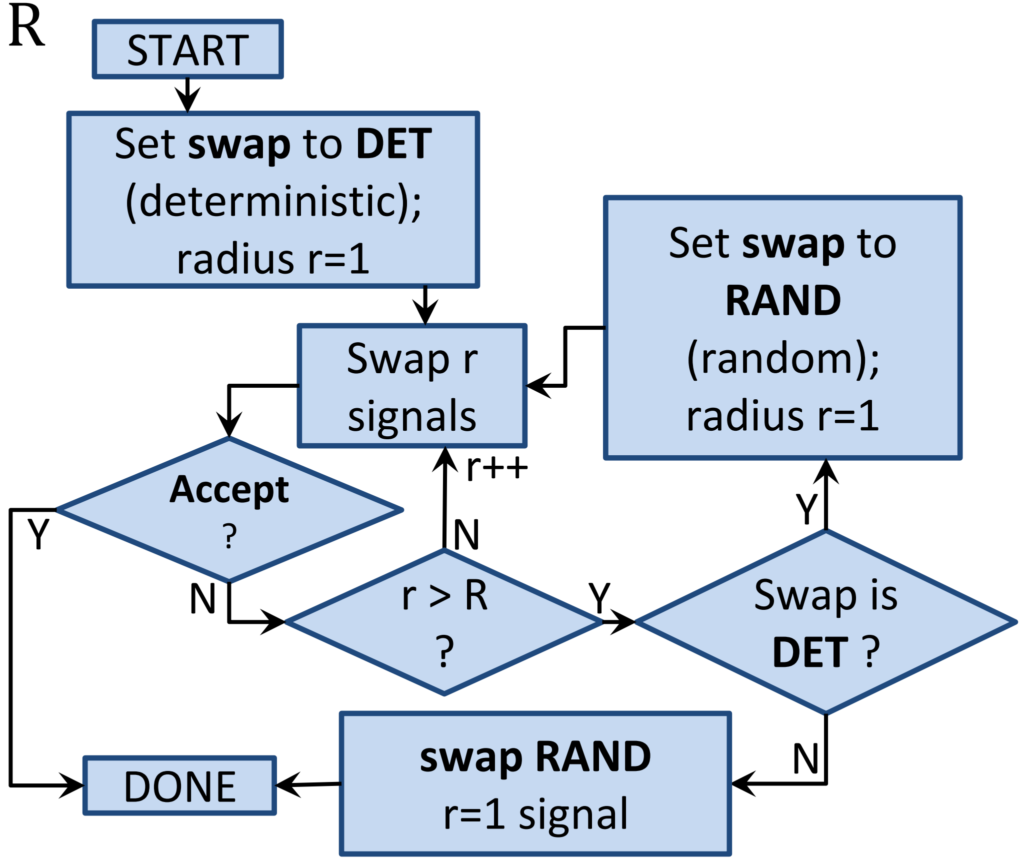
Overview of our procedure:

1. Identify a suitable start mode m_{init}
   - For each mode compute a set representing the union of the reachability lists for all the flipflops in that mode, and then let the start mode to be the one which has the largest size among all sets
2. Find an initial solution to maximize the SRR in mode m_{init}
   - Generated using *Li & Davoodi [DATE'13]*
3. Iteratively perturb the current solution for better multi-mode restoration
   - Has a a non-greedy nature
   - Is based on a gradually-increasing perturbation radius r within each iteration
   - Specifically, at each iteration, up to $R=3$ number of trace signals in the current solution may be swapped
4. The process terminates upon observing no improvements in MSSR in 20 consecutive iterations

ItеM: Iterative Multi-mode Trace Selection

Overview of swap for $r \le R$ **trace signals:**

- Gradually increases the perturbation radius from $r=1$ to $r=R$
- Uses a probabilistic acceptance criteria similar to simulated annealing to probabilistically accept the swaps when there is no improvement in MSRR



ItеM: Iterative Multi-mode Trace Selection

- Each swap consists of the following two basic steps:
  1. Eliminate r trace signals which are least promising
     - Evaluate how much each currently-selected trace signal contributes to MSRR using simulation and eliminate r trace signals with the least contribution
     - If the above deterministic elimination does not lead to improvement, randomly eliminate r trace signals

  2. Add r most promising trace signal
     - Identify the top 3% of the flipflops with the highest value of the proposed multi-mode impact weight MW_f
     - Use simulation to compute MSRR for the identified top flipflops and pick the one with the highest MSRR as the trace signal to add
     - The above procedure is similar to *Li & Davoodi [DATE'13]*

Experimental Results

Benchmark Information

Bench	#FF	#Gates	M	M_{merged}	Suite
S38584	1166	10552	2	2	ISCAS89
S35932	1728	11032	4	2	ISCAS89
b17	1317	33888	4	4	IWLS05
b18	3020	119762	2	2	IWLS05
dsp	3605	54730	8	2	IWLS05
DMA	2192	36556	8	4	ISPD12
des_perf	8802	149066	2	2	ISPD12

- All benchmarks (excluding S38584 and S35932) are much larger compared to the ISCAS’89 used in prior works
- dsp has the maximum reduction in the number of modes, from 8 to 2, due to mode merging

Implemented Approaches

- **RATS:** implemented the single-mode procedure of *Basu & Mishra [VLSI'11]*
- **HYBR:** single-mode procedure of *Li & Davoodi [DATE'13]* (our previous work)
- **SimF:** single-mode forward-greedy selection based on simulation
- **HYBRM:** simple extension of HYBR for multi-mode signal selection
- **IteM:** the proposed iterative multi-mode selection procedure (this work)
- **REF:** upper bound computed by adding the highest attainable SRR per mode
  - Highest SRR/mode computed by solving the single-mode trace selection in that mode using various algorithms

Comparison of MSRR

Bench	REF	RATS	HYBR	SimF	HYBRM	IteM
S38584	25.20	0.86	0.85	0.95	0.95	0.99
S35932	66.40	0.64	0.74	0.65	0.91	0.91
b17	7.90	N/A	0.62	0.58	0.76	0.94
b18	5.90	N/A	0.50	0.92	0.61	0.80
dsp	42.80	N/A	0.41	0.88	0.37	0.92
DMA	50.67	0.76	0.88	0.89	0.84	0.92
des_perf	77.60	N/A	0.97	0.98	0.98	0.99
Average	**1.00**	**N/A**	**0.71**	**0.83**	**0.77**	**0.93**

- REF column reports MSRR and the remaining columns are normalized with respect to REF
- Observations: IteM consistently performs better than other methods

Comparison of Runtime

Bench	RATS	HYBR	SimF	HYBRM	IteM
S38584	0.1	2	19	4	13
S35932	0.1	2	14	5	15
b17	> 24hrs	1	19	4	24
b18	> 24hrs	4	2151	119	90
dsp	> 24hrs	2	92	28	251
DMA	5	7	99	38	125
des_perf	> 24hrs	16	469	24	94

- Runtime is reported in minutes
  - RATS, although fast for the ISCAS89 benchmarks, didn't scale for the large benchmarks (took more than 24hrs)
- The runtime of IteM is reasonable given the large size of the benches and comparable with HYBRM which is based on simple extension of the very fast HYBR

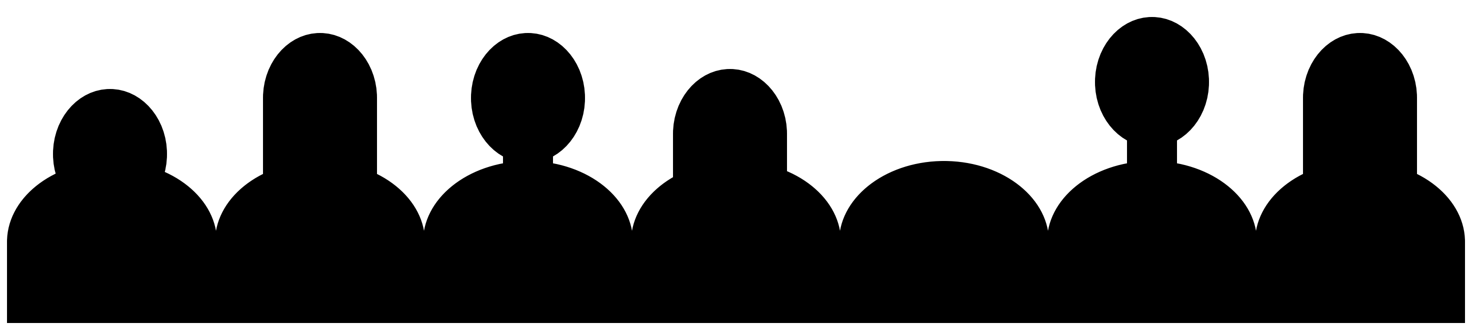
Conclusions

- We proposed the multi-mode trace signal selection problem (MMTS)
- We introduced a strategy to merge the modes with similar restoration maps
- We proposed an algorithm to solve the problem based on iterative perturbation of an initial solution obtained from a single but suitable start mode
- Experimental results showed that the iterative algorithm performs better than various single-mode or multi-mode algorithms, with a high solution quality comparable to the reference case

References

1) K. Basu and P. Mishra. **RATS: restoration-aware trace signal selection for post-silicon validation**. In *IEEE TVLSI,* 2013
2) D. Chatterjee, C. McCarter, and V. Bertacco. **Simulation-based signal selection for state restoration in silicon debug**. In *ICCAD,* 2011
3) M. Li and A. Davoodi. **A hybrid approach for fast and accurate trace signal selection for post-silicon debug**. In *DATE,* 2013
4) H. F. Ko and N. Nicolici. **Algorithms for state restoration and trace signal selection for data acquisition in silicon debug**. In *IEEE Trans. On CAD,* 2009
5) X. Liu and Q. Xu. **On signal selection for visibility enhancement in trace-based post-silicon validation**. In *IEEE Trans. on CAD,* 2012

Thank You!



Comparison of Merged/Unmerged Cases

- Apply the iterative algorithm on the three benches having mode reduction (merged)
- For “W Merge” case, randomly pick one mode to represent the modes merged
- For MSRR comparison, raw numbers are shown (without normalization). For runtime comparison, results of the “W/O Merge” case are normalized to “W Merge”
- **Mode merging can significantly reduce the runtime while obtaining a comparable solution quality**