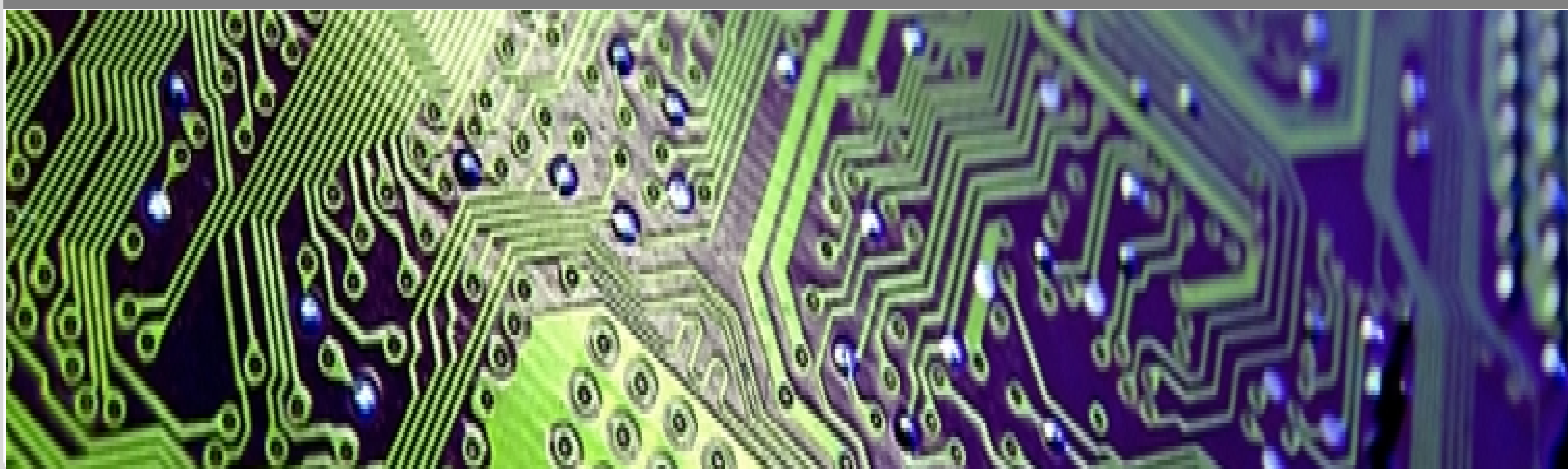
Workload-aware Static Aging Monitoring of Timing-critical Flip-flops

Arunkumar Vijayan, S. Kiamehr, F. Oboril, K. Chakrabarty, and M. B. Tahoori

INSTITUTE OF COMPUTER ENGINEERING (ITEC) – CHAIR FOR DEPENDABLE NANO COMPUTING (CDNC)

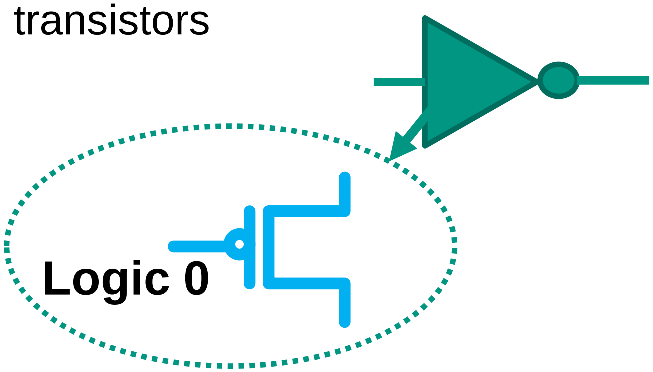


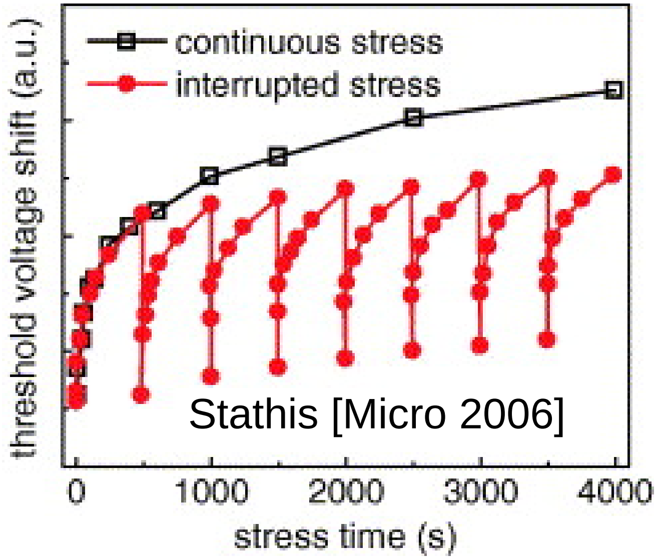
Outline

- Purpose and Motivation
- Background and Related Work
- Main Idea and Methodology
- Experimental Setup & Results
- Summary and Conclusion

Purpose and Motivation

- Technology scaling advances → Aggravates aging effect in transistors
- Aging
  - Bias Temperature Instability (**BTI**)
  - Increases threshold voltage of transistors
  - Increase in propagation delay of critical paths
  - Leads to timing failures in aging paths
- BTI based on type of stress
  - Dynamic BTI (**DBTI**)
  - Static BTI (**SBTI**)
- Static BTI
  - Long phases of inactivity in flip-flops
  - Accelerated aging stress during active operation
  - One year stress of **DBTI** = A few hour stress of **SBTI**
- Proposed Approach
  - Track **SBTI** by online monitoring
  - Trigger switching of critical flip-flops for relaxation



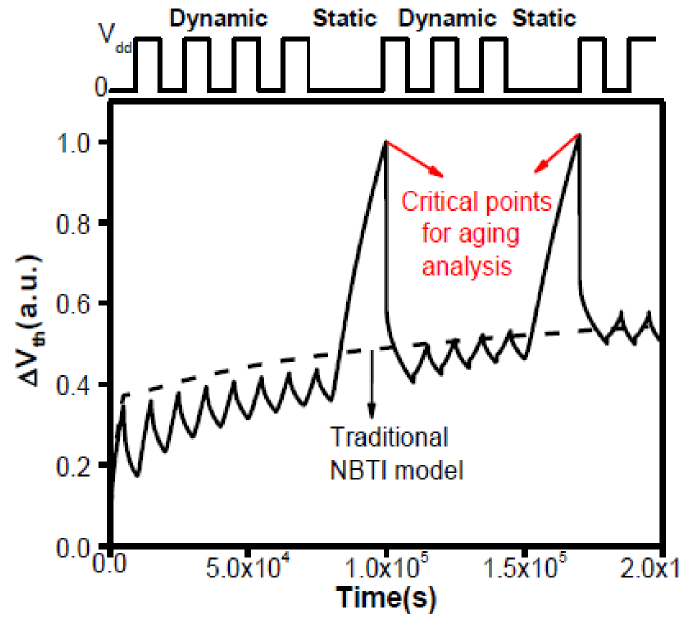


Outline

- Purpose and Motivation
- Background and Related Work
- Main Idea and Methodology
- Experimental Setup & Results
- Summary and Conclusion

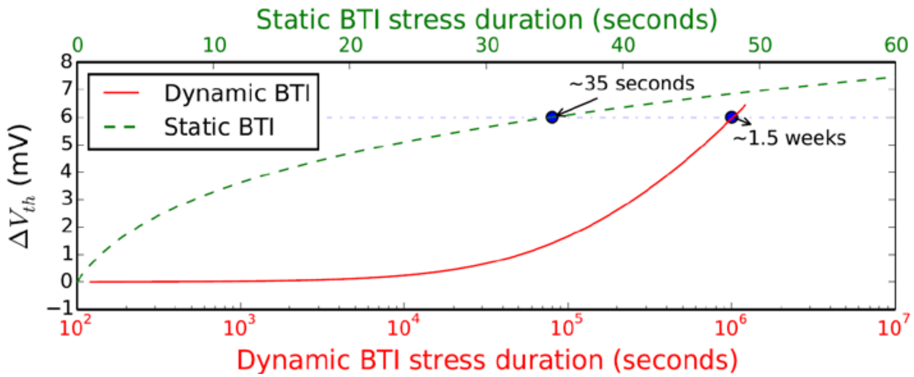
Background

- Bias Temperature Instability (BTI)
  - Major reliability concern
  - Increases threshold voltage of transistors
  - Translates to circuit-delay increase



Ref: Velamala [ICCAD 2011]

- Dynamic BTI (DBTI)
  - Stress phase and recovery phase
  - Overall aging effect is non-critical in a short period of time



- Static BTI (SBTI)
  - Continuous stress phase
  - Accelerated aging effect

Related Work

- Only **DBTI** is considered [Rao 2011 ICCD], [Bild 2012 TODAES], [Wang 2009 DATE]
  - Ignores workload profile of flip-flops in processors

- **SBTI** is considered only in specific scenarios.
  - Stand-by mode and sleep mode [Velamala 2011 ICCAD]
  - Ignore workload-specific stress scenarios in flip-flops

- **SBTI** in flip-flops [Golanbari 2015 ETS]
  - Ignores runtime variation in **SBTI** stress durations
  - Pessimistic approach

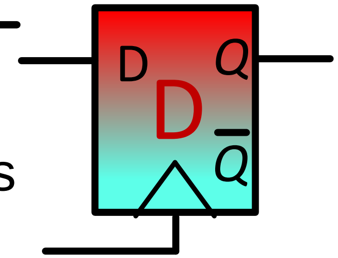
Overall Inference:
Runtime dependency in **SBTI** stress due to workload profile of flip-flops is largely ignored.

Outline

- Purpose and Motivation
- Background and Related Work
- Main Idea and Methodology
- Experimental Setup & Results
- Summary and Conclusion

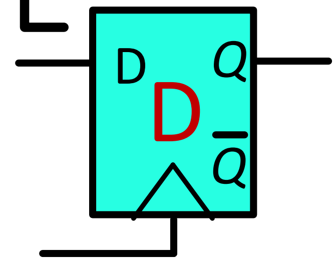
Main Idea

Static Aging Phase (SAP)

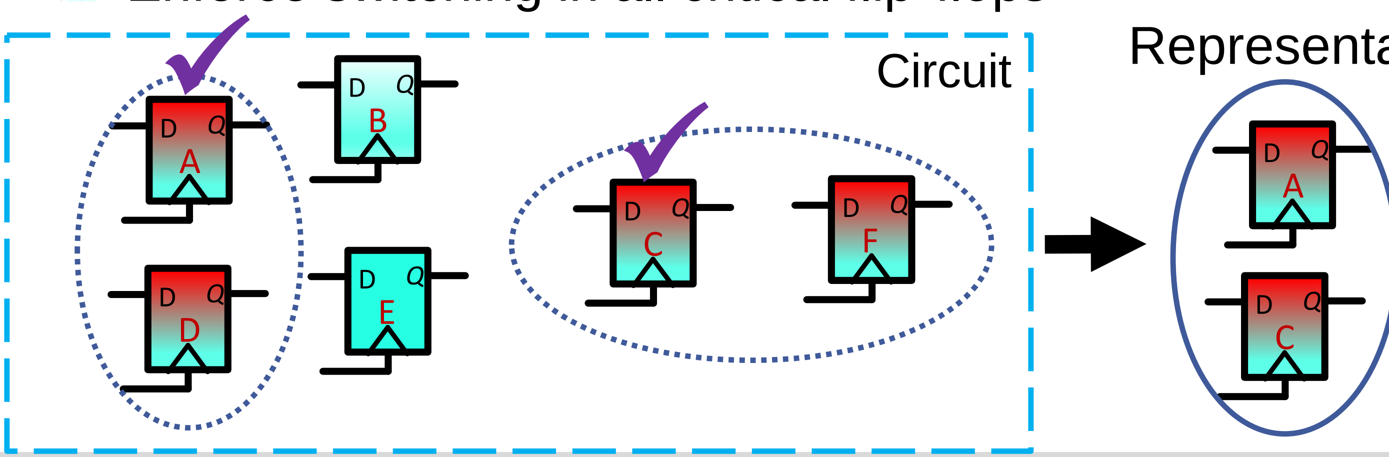


- Monitor flip-flops and check the workload profile
- Detect long periods of inactivity in critical flip-flops
- Enforce recovery

(Recovery)



- Reduce overhead for monitoring
  - Correlate Static Aging Phases
  - Monitor only a smaller subset of flip-flops as representative
  - Enforce switching in all critical flip-flops

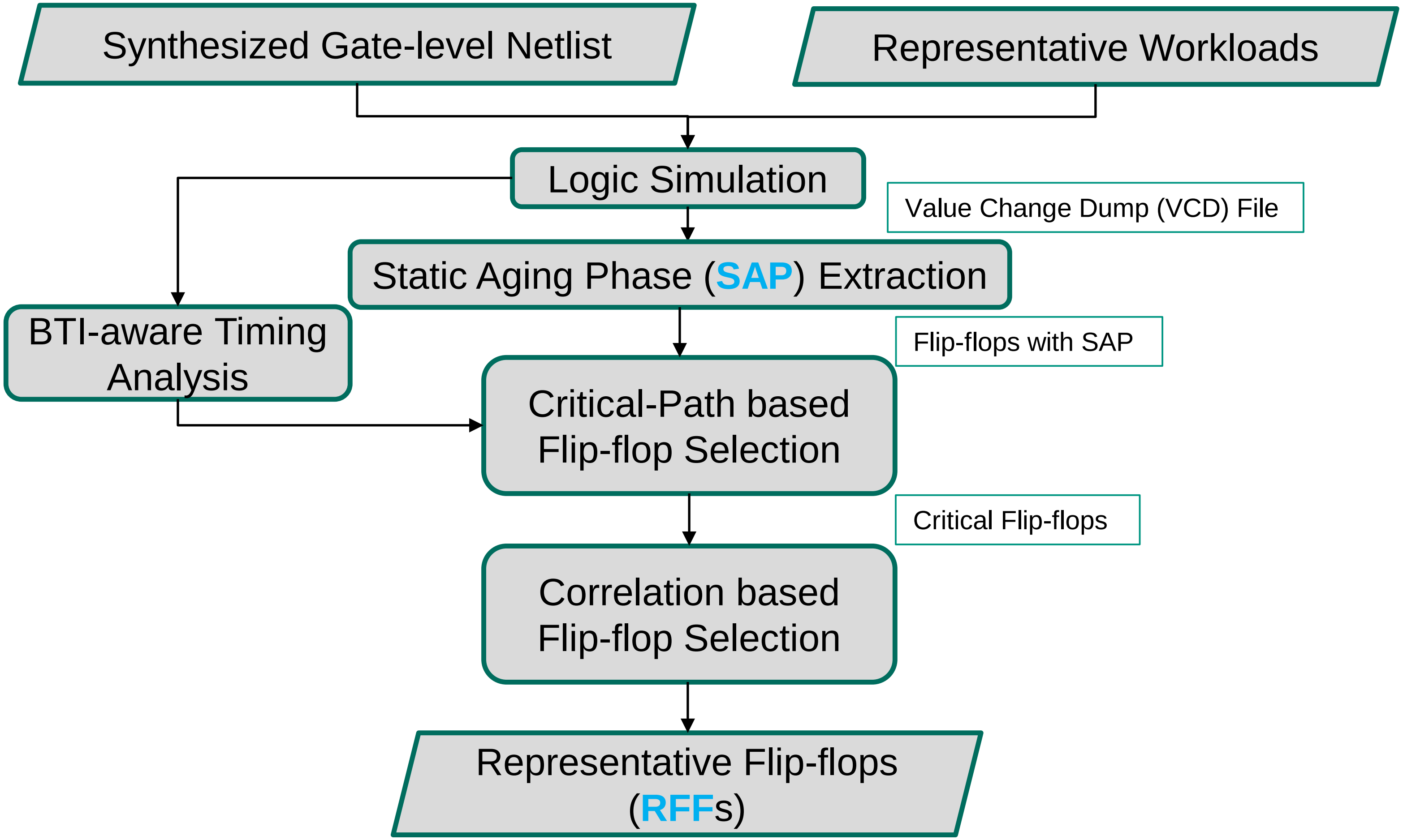


Methodology

- Offline Characterization
  - Post-synthesis simulation
  - Using gate-level netlist of processor cores
  - Executing real workloads
  - Dumping logic values of circuit nodes for millions of clock cycles.
- Offline Correlation Analysis
  - Analysis of Static Aging Phases (**SAPs**) in simulation dump
  - Correlation of **SAPs** for timing-critical flip-flops
  - Grouping correlated flip-flops
  - Obtaining representative flip-flops

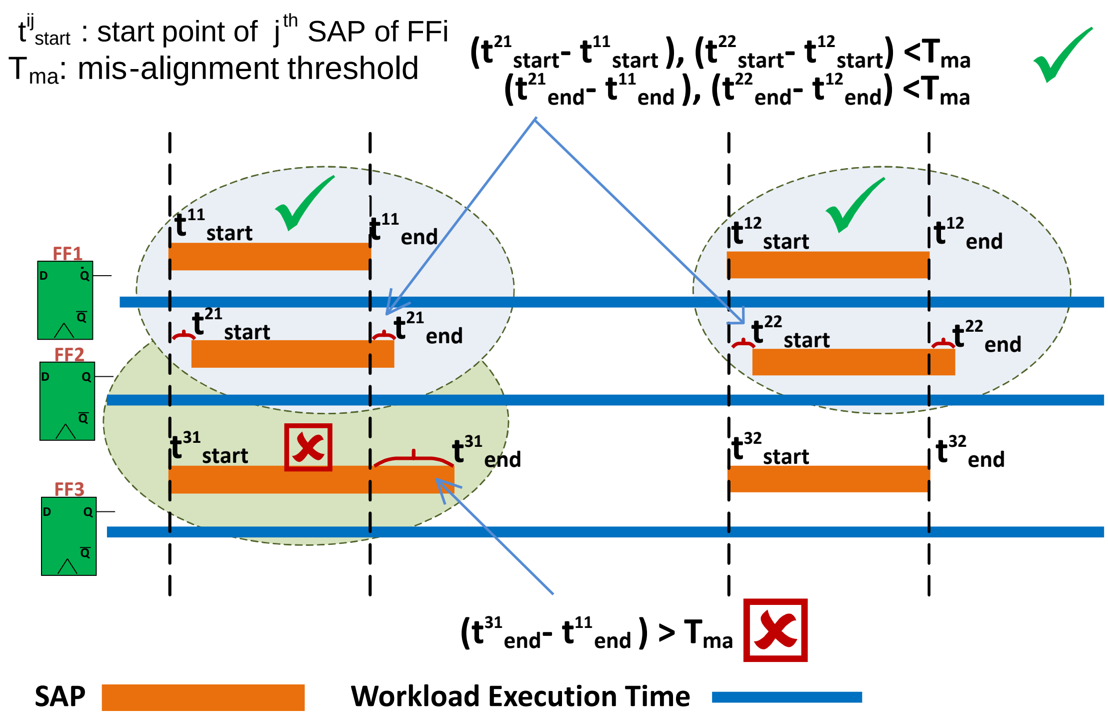
- Online Monitoring
  - Monitor representative flip-flops to find **SAP**s
  - Trigger alarm signal in critical scenarios
- Mitigation
  - Relax/switch flip-flops under severe **SBTI**

Offline Characterization



- Offline analysis of Static Aging Phases (**SAP**s) for different workloads
- Extracting correlation between **SAP**s across flip-flops
- Finding a small set of flip-flops to represent the circuit aging stress

Correlation-based Flip-flop Grouping

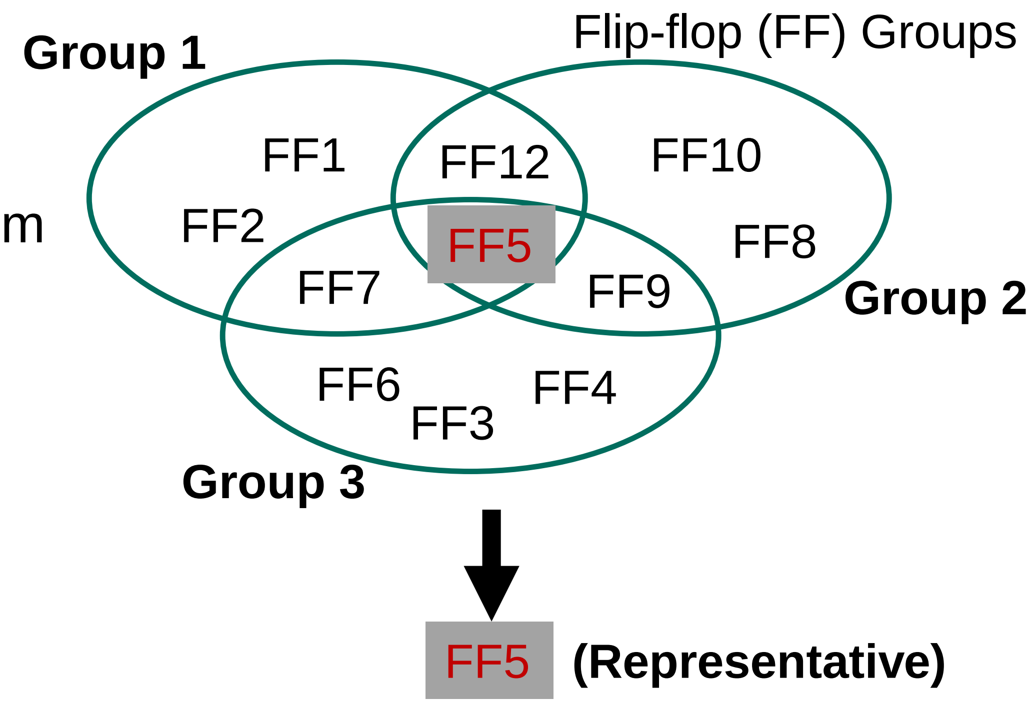


- Exploiting correlation in Static Aging Phases (**SAP**s) to reduce the number of **RFF**s
- Uses the time-points at which a flip-flop enters/leaves an **SAP**
- Time-points correlate → **SAP**s correlate
- **SAP** observed on a flip-flop → criticality to the correlated flip-flops

Correlation-based Flip-flop Selection

- Selection of flip-flops
  - Obtain correlated flip-flop groups
  - Select one flip-flop from each group
  - Representative flip-flops → union of selected flip-flops

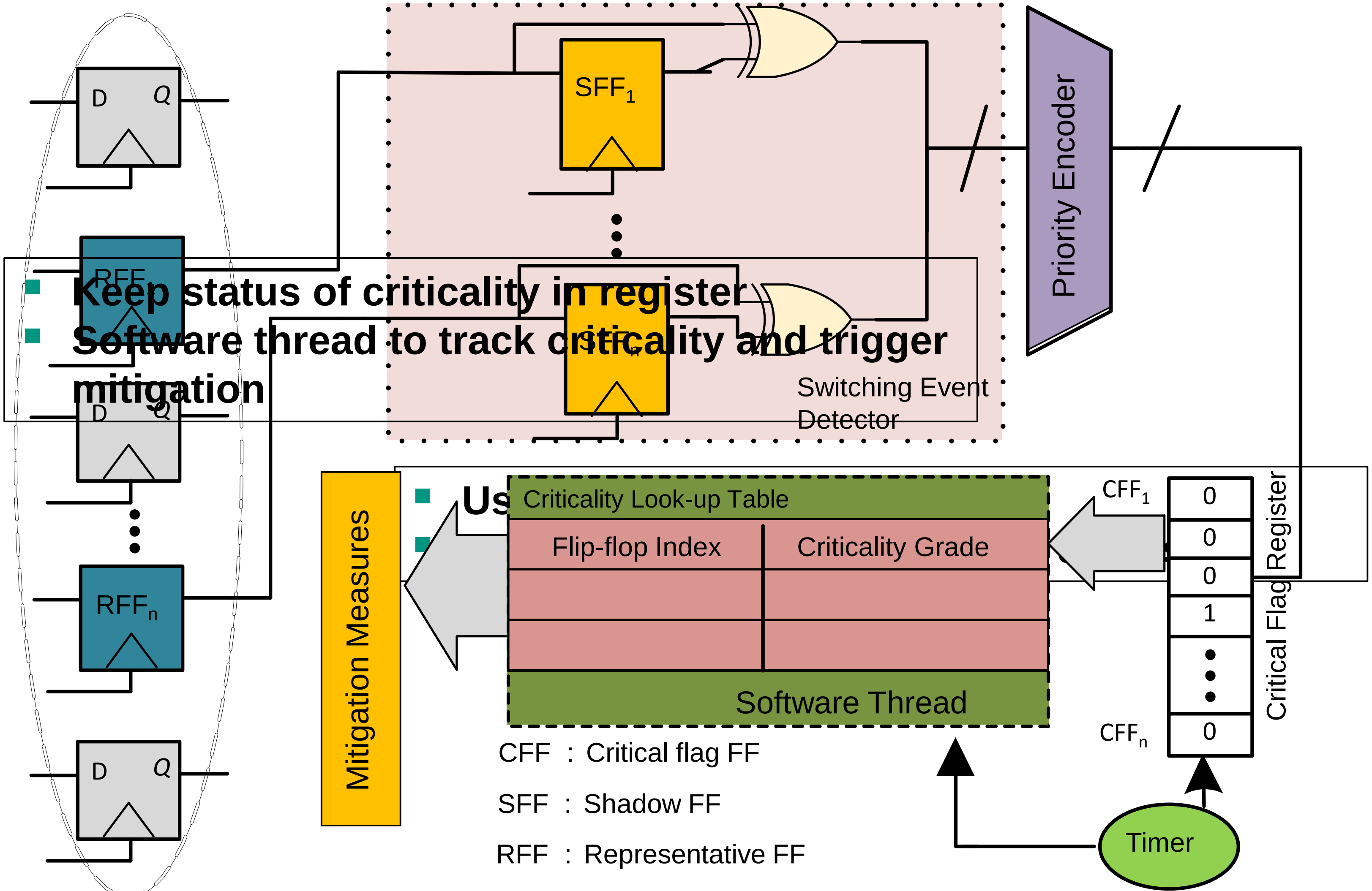
- Optimum selection
  - Minimal Hitting Set Problem
  - Prefer common flip-flops
  between different groups



- Minimal Hitting Set
  - NP-complete
  - Used greedy algorithm

Online Monitoring

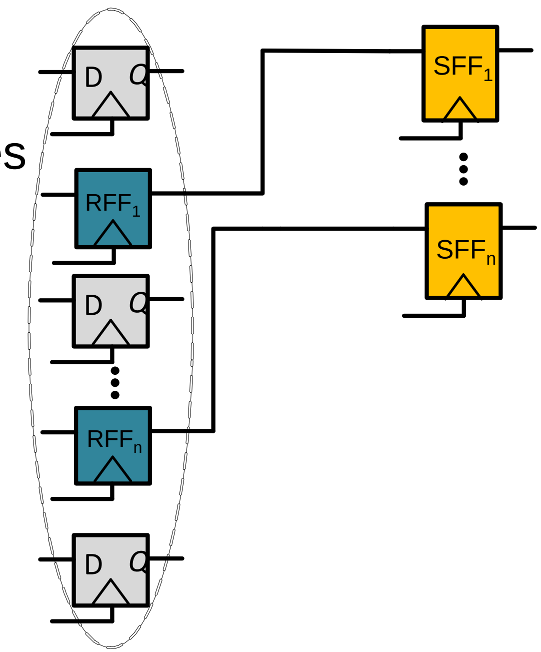
- Keep status of criticality in register
- Software thread to track criticality and trigger mitigation



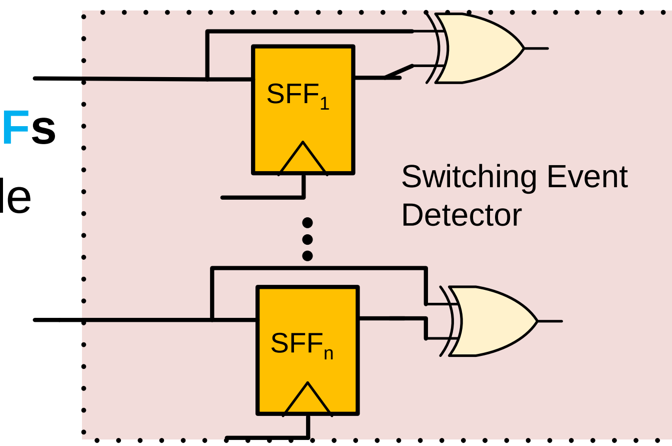
- Monitoring **Representative Flip-flops** (**RFF**s)

Online Monitoring: Critical Phase Detection

- Using Shadow Flip-flops (**SFF**s)
  - Each **RFF** is shadowed
  - Track **RFF**s online to identify critical workload phases



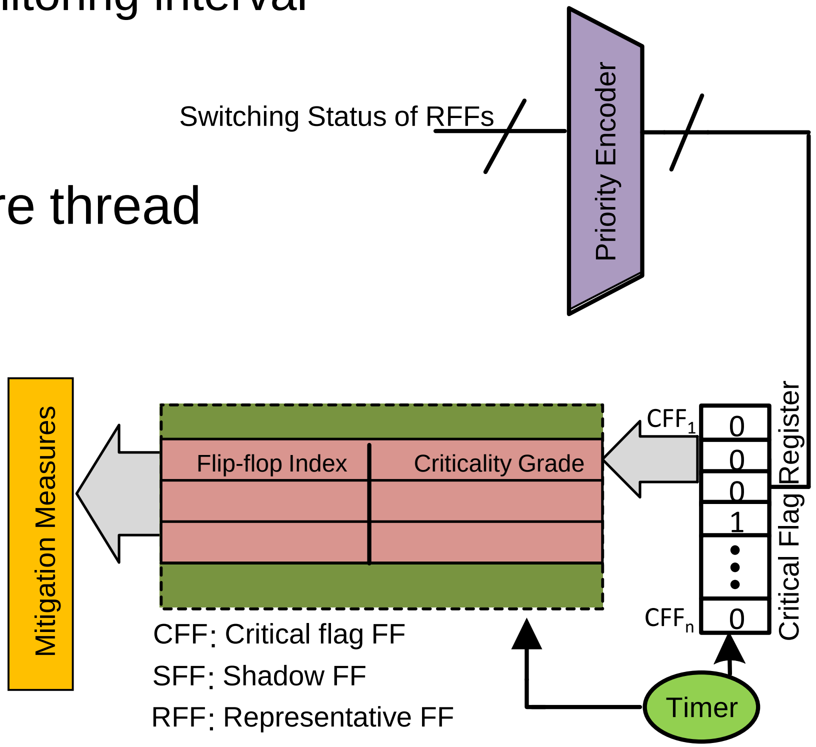
- Switching Event Detection
  - Using XOR gates
  - Tracks and reports switching events in **RFFs**
  - XOR gate: comparing value in current cycle and previous cycle
  - Generates a logic ‘1’ for a switching event



Online Monitoring: Stress analysis and Recovery

- Records **RFFs** under static aging in each monitoring interval
  - Use critical flag register
  - Encode the aging status
  - One bit reserved for each **RFF**
  - Start with all 0s and set to 1 for the first switching event in an **RFF**
  - Send the criticality report for each monitoring interval

- Track critical static aging using software thread
  - Maintains a criticality Look-up Table
  - Receives criticality report
  - Trigger activation signal for mitigation



Mitigation Measures

- Switching event in Flip-flops
  - Can trigger an immediate recovery from static aging
  - Releases stress of both flip-flops and logic gates
- Propose software-level solutions
  - Adds appropriate instructions in the execution queue (Eg: Pseudo-NOPs)
  - Exercise critically aged FFs
  - Reverses action and leaves no foot-print
- Overhead
  - Software-level solution
    - Minimal performance and area overhead

Outline

- Purpose and Motivation
- Background and Related Work
- Main Idea and Methodology
- Experimental Setup & Results
- Summary and Conclusion

Experimental Setup

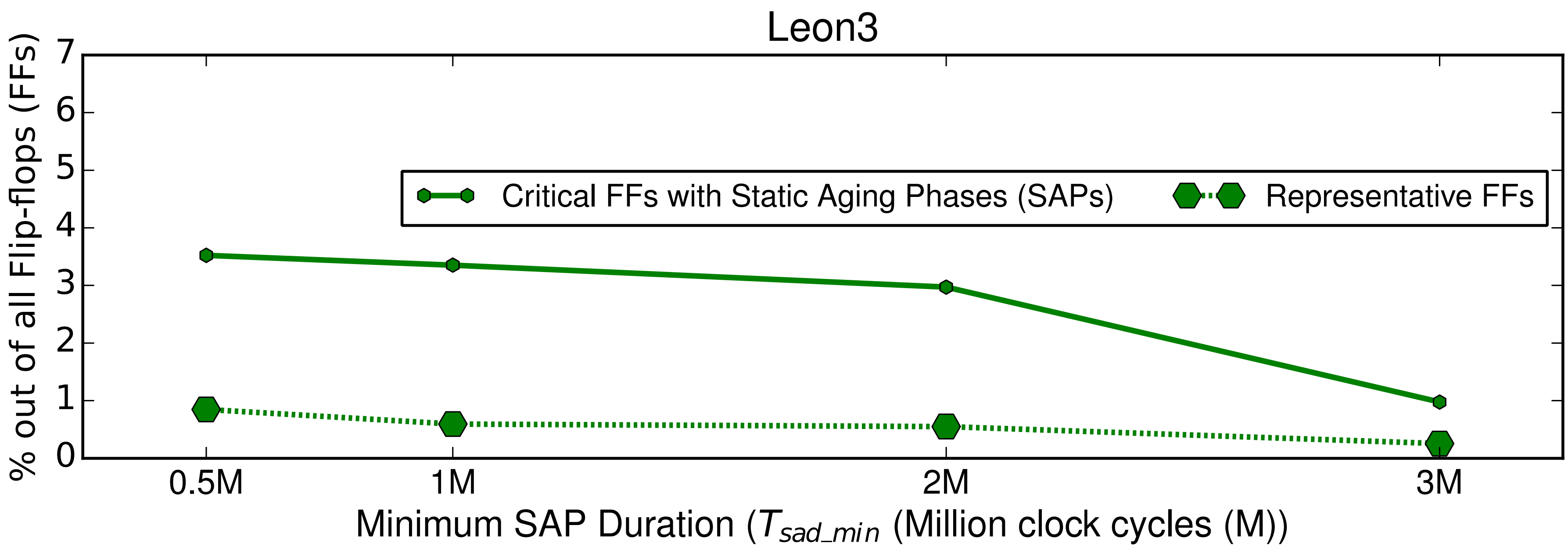
- Processors
  - Leon3
    - 32 bit embedded processor
    - 7 stage pipeline
    - SPRAC-V8 Instruction Set Architecture (ISA)
    - single core
  - Fabscalar
    - Superscalar out-of-order processor
    - 11 stage pipeline
    - Portable ISA (PISA)
    - single core
- Programs
  - six Mibench workloads for Leon3
  - six SPEC workloads for Fabscalar
- Library
  - Nangate 45nm

Results and Discussion

	Leon3	Fabscalar
Total FFs	2356	7563
FFs on critical path (<10% of maximum slack)	451 (19%)	536 (7%)
No. of critical FFs with static aging phases (Union)	42 (1.8%)	450 (6%)
No. of Representative FFs	7 (0.30%)	36 (0.48%)

- Significant number of flip-flops are under static aging stress
- Need to monitor only 7 flip-flops for Leon3 and 36 for Fabscalar

Impact of inactivity phase on monitoring overhead



- Threshold (T_{sad_min}): minimum duration of inactivity considered for **SBTI**
- Higher threshold → longer critical **SAPs** → fewer **RFFs** to monitor

Overheads

	Area Overhead	Power Overhead
Leon3	0.22%	0.04%
Fabscalar	0.09%	0.05%

- FFs under **SBTI** switches rarely → leakage power dominates
- Additional load at FFs → no impact on circuit delay

Lifetime Improvement

- Defined as the reliably operating duration of a circuit within margin
- Comparing worst case scenarios with and without **SAP** monitoring
- (Dynamic + Static BTI) vs (Dynamic BTI alone)
- 1.9X lifetime improvement (for a minimum **SAP** of 3 million clock cycles)

Outline

- Purpose and Motivation
- Background and Related Work
- Main Idea and Methodology
- Experimental Setup & Results
- Summary and Conclusion

Summary and Conclusion

- With technology scaling → Significance of BTI
- Static BTI
  - Aggravates during phases of inactivity (**SAP**)
  - Need to be considered in worst-case analysis
- Our approach
  - Offline stage
    - Analyze static aging phases in flip-flops
    - Select timing critical and aging critical flip-flops
    - Correlation analysis to find representative flip-flops
  - Online stage
    - Monitor representative flip-flops to find static aging phases
    - Update the criticality in a software thread
    - Trigger mitigation actions based on severity of BTI
- Overheads
  - Less than 0.25% area and power overheads for Leon3 and fabsaclar
- Lifetime improvement
  - 1.9X lifetime improvement for a minimum SAP of 3M clock cycles.

Thank You!

Q&A

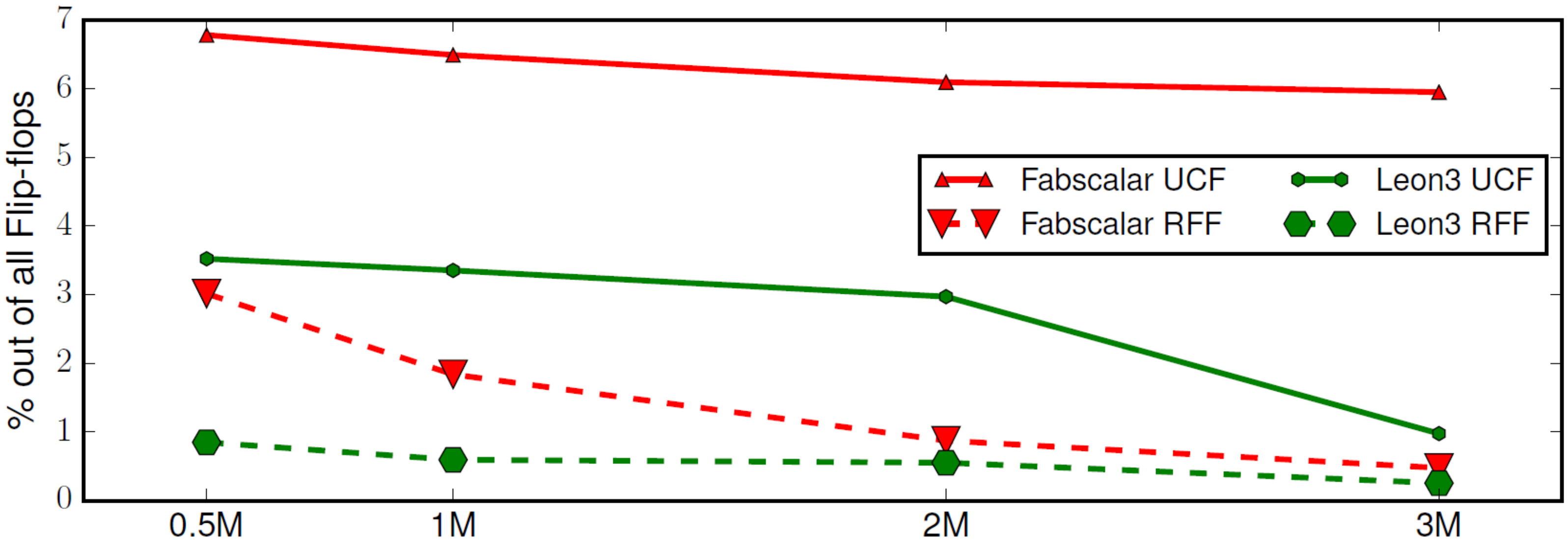
Back-up Slides

Results and Discussion

	Leon3						**Fabscalar**					
Total FFs	2356						7563					
FFs on critical path (<10% of maximum slack)	451						536					
Workloads	String search	qsort	susan	sha	basic math	crc32	gzip	gap	parser	vortex	mcf	bzip
No. of critical FFs	34	37	24	36	21	33	207	7	17	5	376	196
No. of critical FFs (Union)	42						450					
No. of correlated FF groups	2	5	2	1	3	2	17	5	10	3	12	16
No. of RFFs	7						36					

- Significant number of flip-flops are under static aging stress
- Can be represented using a few representative flip-flops

Results and Discussion



UCF (%): $\dfrac{\textit{Union of Critical Flip-flops of all workloads}}{\textit{Total number of Flip-flops}}$ RFF (%): $\dfrac{\textit{Number of Representative Flip-flops}}{\textit{Total number of Flip-flops}}$

- SAP is considered when duration > constant threshold T_{sad_min}
- Shows variation in number of representative flip-flops when SAP duration threshold increases
- To reduce the number of RFFs, concentrate on critical SAPs

Online Monitoring

- Representative Flip-flops
  - Each RFF is shadowed
  - Monitored online to identify critical workload phases
- Switching Event Detector
  - Tracks and reports switching events in RFFs
  - Represents aging stress relaxation
- Critical-Flag Register
  - Encodes flip-flops staying at SAP and flip-flops relaxed
  - Stores criticality report to send to software at regular intervals
- Tracking Software
  - Maintains a criticality Look-up Table
  - Live status of aging stress is stored
  - Trigger activation signal for critical flip-flops
- Mitigation
  - Exercise critical flip-flops in order to relieve stress
  - Suitable instructions (eg: pseudo-NOPs) are executed

Summary and Conclusion

- Existence of critical workload phases for logic designs
- Worst-case workload-specific aging scenarios due to SBTI can cause timing violations
- Reliability requirements can only be met by proper monitoring and mitigation techniques.
- Design of runtime monitoring hardware that raises a flag on criticality
- Achieved by monitoring a few number of representative flip-flops correlated with the critical flip-flops
- 1.9X reliability lifetime improvement with low area and power overhead