ECE | ELECTRICAL & COMPUTER ENGINEERING DEPARTMENT

Beyond Worst-Case Design for Energy-Conscious Dependable Architectures

Georgios Karakonstantis

Associate Professor

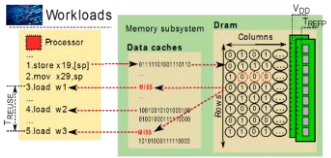
Dept. of Electrical and Computer Engineering

Twin-Relect Summer Workshop – July 2026

Research Overview

AI based System Behaviour Learning & Prediction

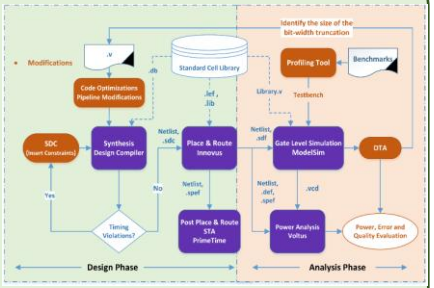
- Data & micro-architecture aware power and reliability characterization
- Data-aware AI based failure prediction models
- Failure prone code identification using genetic algorithms and AI



*DATE '20 Best Paper, MICRO '20 BP Nm, TC '21, TC '23, ISLPED'24, VTS'26

Energy-Scalable, Error-Resilient & Secure Cores

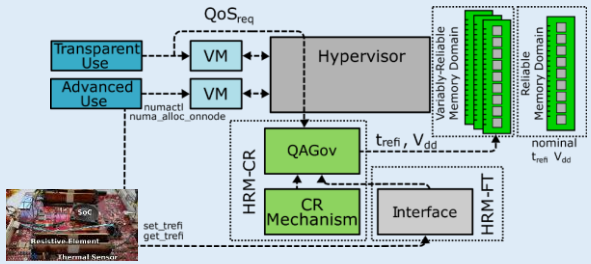
- **Error-resilient** processing cores via **in-house circuit-architecture design** flow with path shaping for error-avoidance
- **Dynamic data-aware precision scaling** for better energy-quality trade-offs
- Processor **Physical Unclonable Functions**
- **Application Specific Accelerators** – Biomedical, DSP, Multimedia Wireless, NNs, ML



*DAC '16, DATE' 19, DATE' 20, TDMR'19, MICRO '21, TCAS '22, ICCAD '23, 'TVLSI 24, TBioCAS '25

Adaptive System Design & Evaluation

- **QoS Aware Power-Reliability Management** with OS/virtualization support
  - **8% system energy savings** on real servers
- Scheduling for **Efficient Resource Sharing of FPGAs** in Multi-Tenant Environments
- Cross-layer tools for **Total-Cost of Ownership** estimation and early evaluation of **application resilience** to HW faults or approximations



*IEEE TC '23, JSPS '21, ASPLOS '20, TC'20, CCGRID '20, CAL '19, FPT '19, FPL '18

Contents

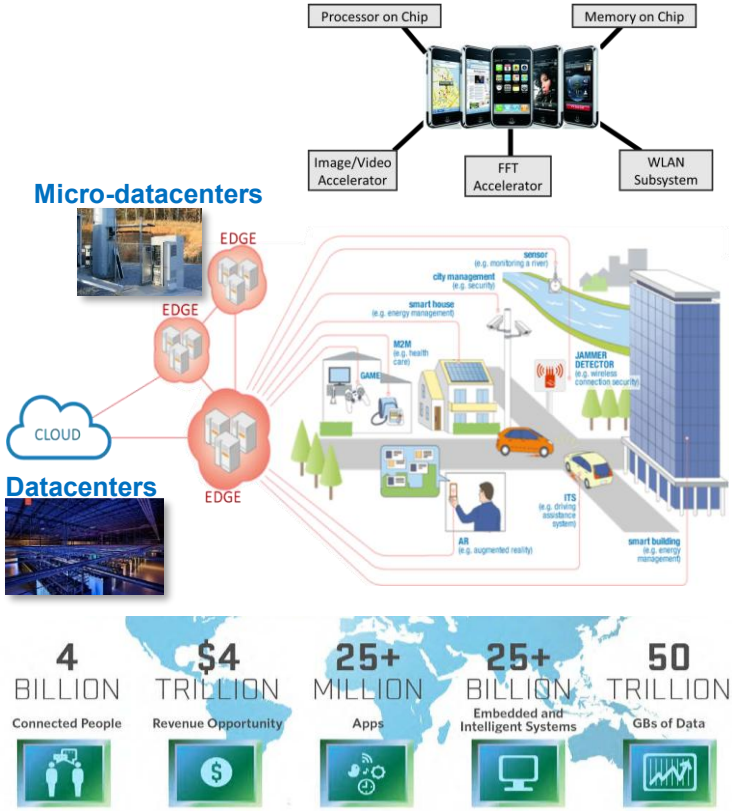
Challenges

- The advent of Internet of Things
  - Smart- city, home, cars, Industry 4.0

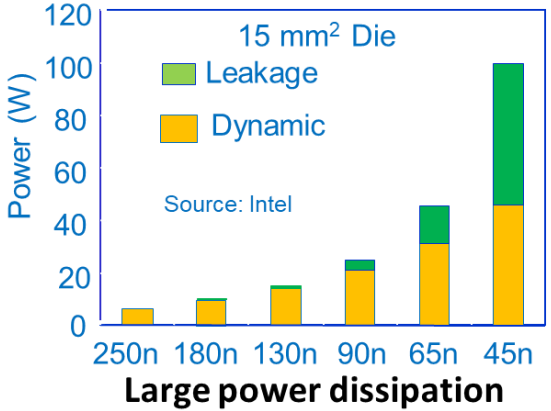
- Growing dependence on:
  - Signal-, Image-processing algorithms and AI based data analysis at the Edge and Cloud
  - Distributed embedded devices operating under strict power budgets and dynamically changing conditions
  - Logic/processor and memory more prone to faults

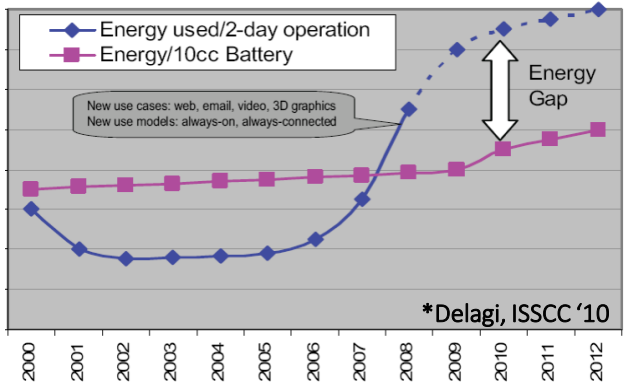
    → difficult to maintain

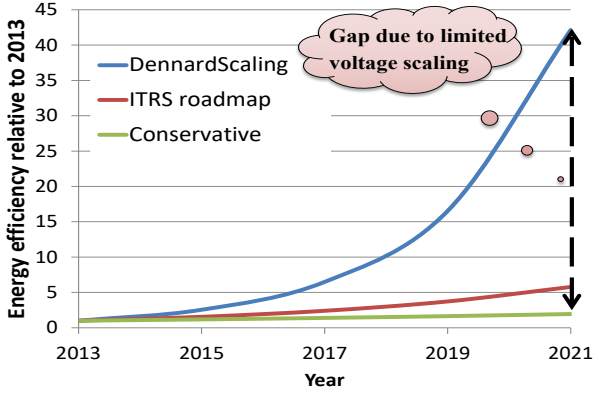
➢ **Main Challenge:** Need energy-efficient and dependable systems able to adapt to varying operating conditions and user-requirements



Design Challenges: Power Consumption





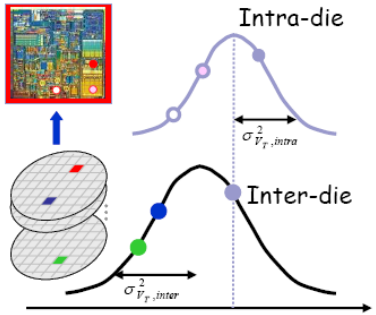


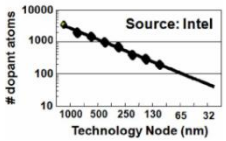
- **Stagnant Power Scaling and Increasing battery gap**
  - Rapidly increasing static and dynamic power consumption
  - Slowly increasing battery capacity

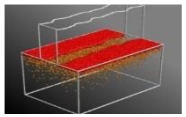
Need for power limiting techniques and efficient use of available energy

Design Challenges: Parametric Variations

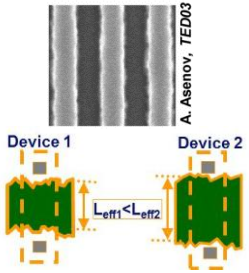
Static variations







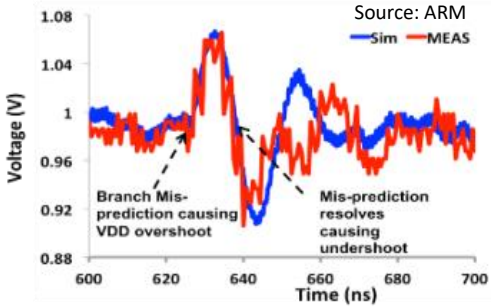
Random dopant fluctuation

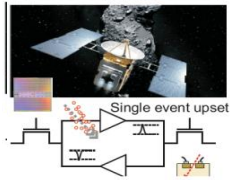


Line-edge roughness

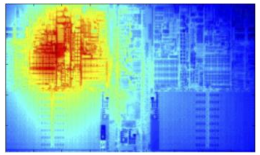
Dynamic variations/Runtime

Voltage variation



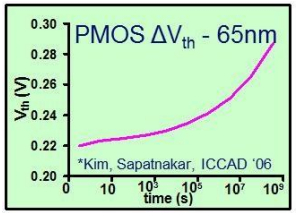


Single event upsets



Temperature fluctuations

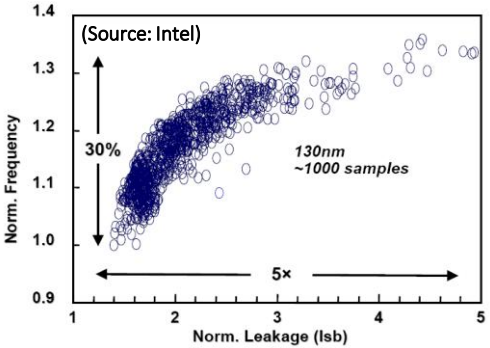
1010010
0100100
Data dependencies



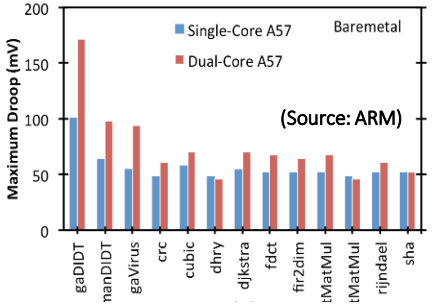
NBTI/Aging

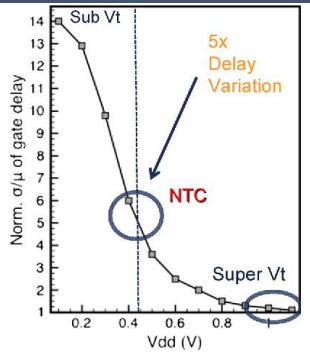
Device parameters are no longer deterministic, changing over time

Design Challenge: Delay variations/Failures

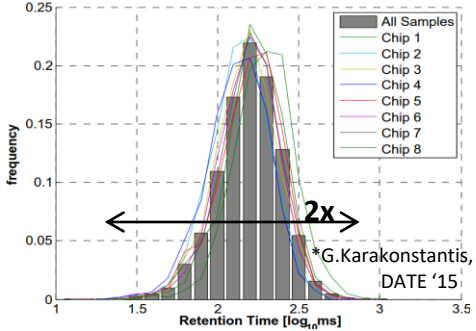


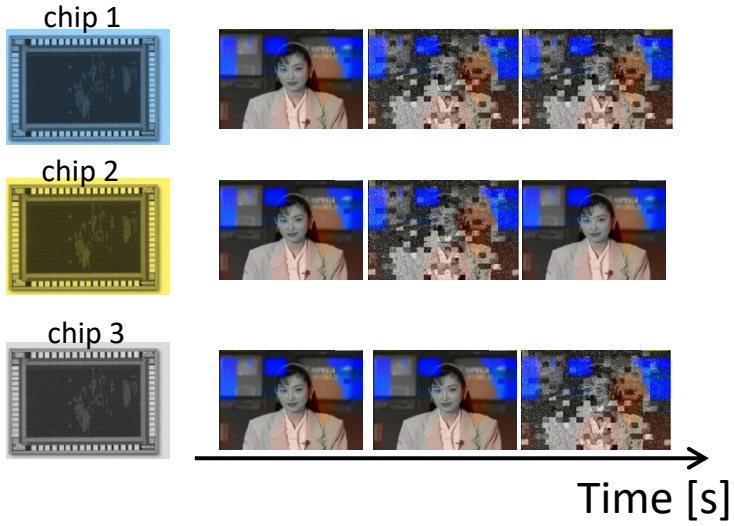
Leakage and Delay variation





DRAM cell Retention Time





Delay variations and failures affecting each chip and delay path differently depending on the operating conditions, workload/data

Contents

Conventional Low Power Design

- Various techniques have been proposed at various levels of design abstraction
  - Pruning operations, scaling down the precision
  - Turning off inactive modules at architecture level
  - Clock gating at circuit level
  - Frequency scaling when lower speed is allowed

- One of the most effective technique turns out to be Voltage scaling
  - Quadratic power savings - $P = C_{eff} \ V_{dd}^2 \ f$

    Example: Reducing V_{dd} by 20% from 1V to 0.8V leads to 36% savings

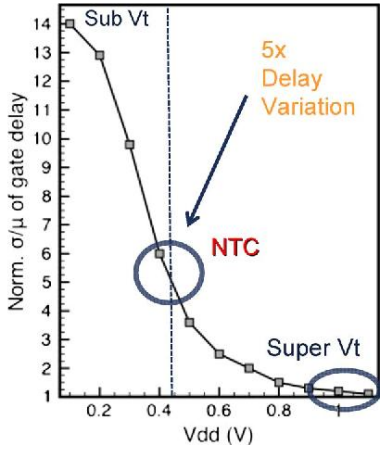
… Voltage Over Scaling

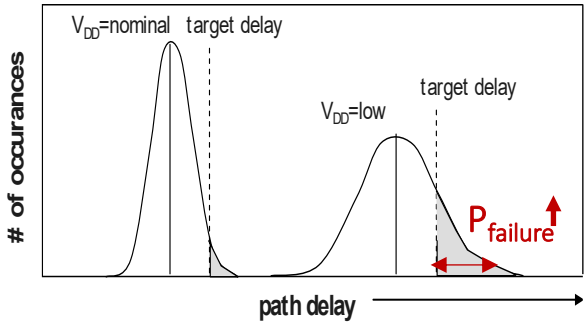
- Voltage scaling may lead to quadratic power savings

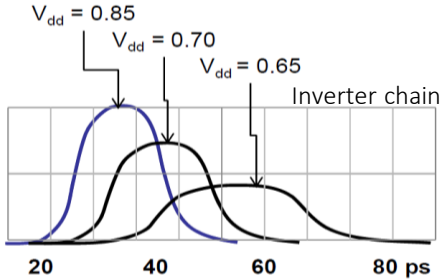
  BUT

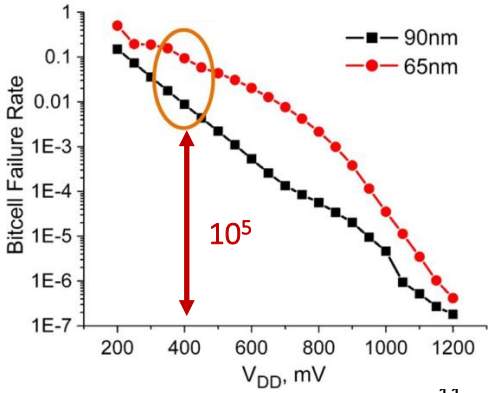
⚠ Increases mean delay - $d \propto \frac{1}{V_{dd}}$

⚠ Worsens variability making circuits more prone to faults and more difficult to meet design specifications and timing









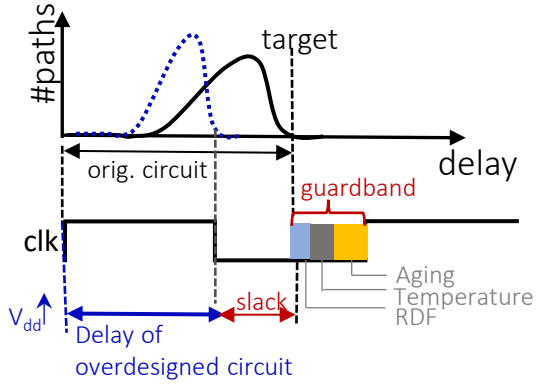
Conventional Variation Aware Design

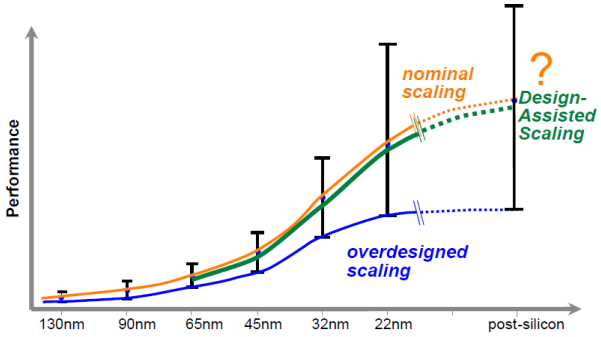
- **Overdesign** to compensate any performance loss induced by various sources of variations
  - Guardbands, Timing redundancy, create slack by up-scaling voltage or by up-sizing transistors, space redundancy/ECC

⚠ Worst case Performance.
Diminishing returns from technology scaling

⚠ Power, area overheads for all chips, even the good ones

⚠ Limit voltage down-scaling.
Contradict with low power requirements



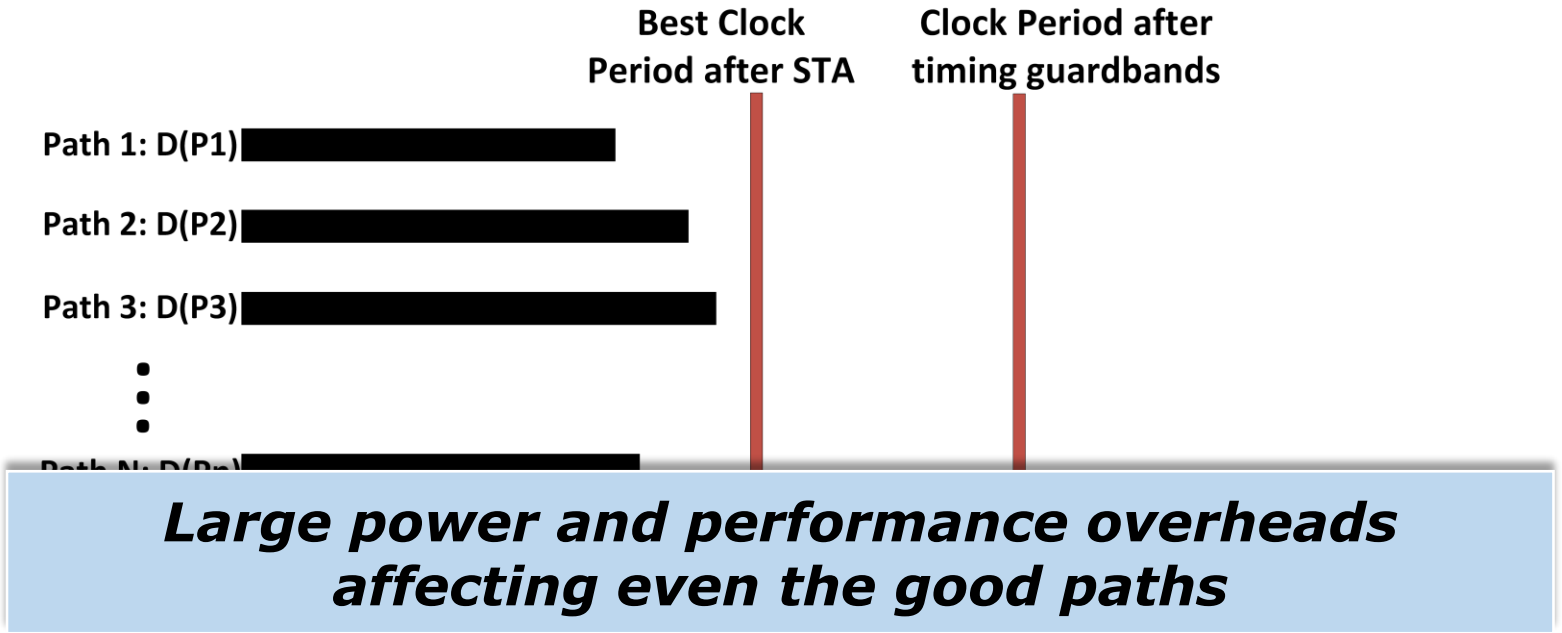


Dreslinski et al. IEEE, 2010

Gupta et al. TCAD, '13

Conventional Error Mitigation Schemes

- Designers address delay variations / timing errors by:
  - Using pessimistic timing/voltage guardbands and schemes
    - Based on Static Timing Analysis (STA) and worst case assumptions (workload, conditions)

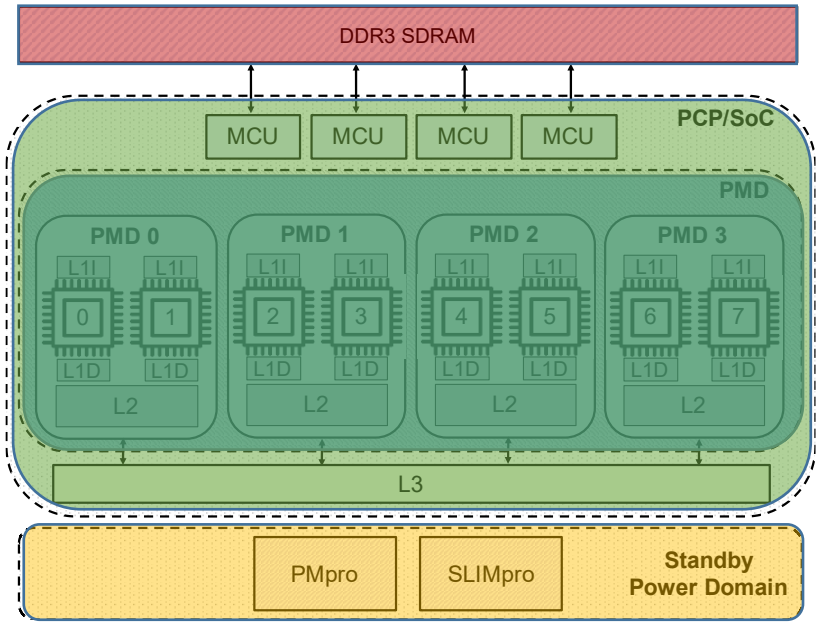


Large power and performance overheads affecting even the good paths

Contents

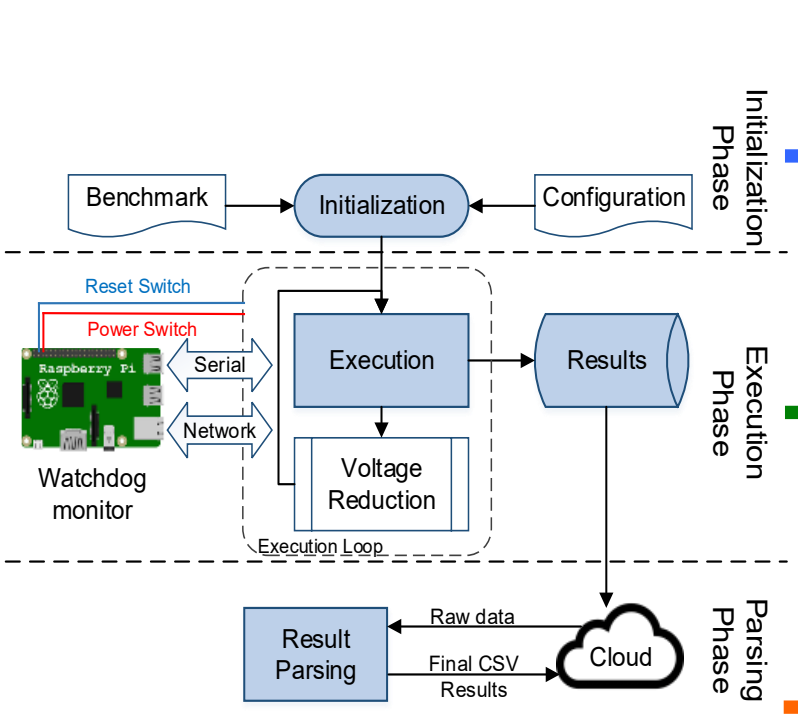
Exposing Pessimism in ARM based Micro-servers

- ARMv8 AArch64 based X-Gene2 Tigershark
  - Socketed and Soldered boards



Parameter	Configuration
ISA	ARMv8 AArch64
CPU	8 cores
Core clock	2.4 GHz
L1I $	32KB per core (Parity)
L1D $	32KB per core (Parity)
L2 $	256KB per PMD (SECDED)
L3 $	8MB (SECDED)
Technology	28 nm
Max TDP	35 W
DRAM	4 x 8GB DDR3-1866 ECC

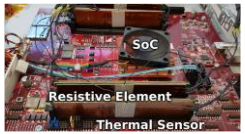
Characterization Framework



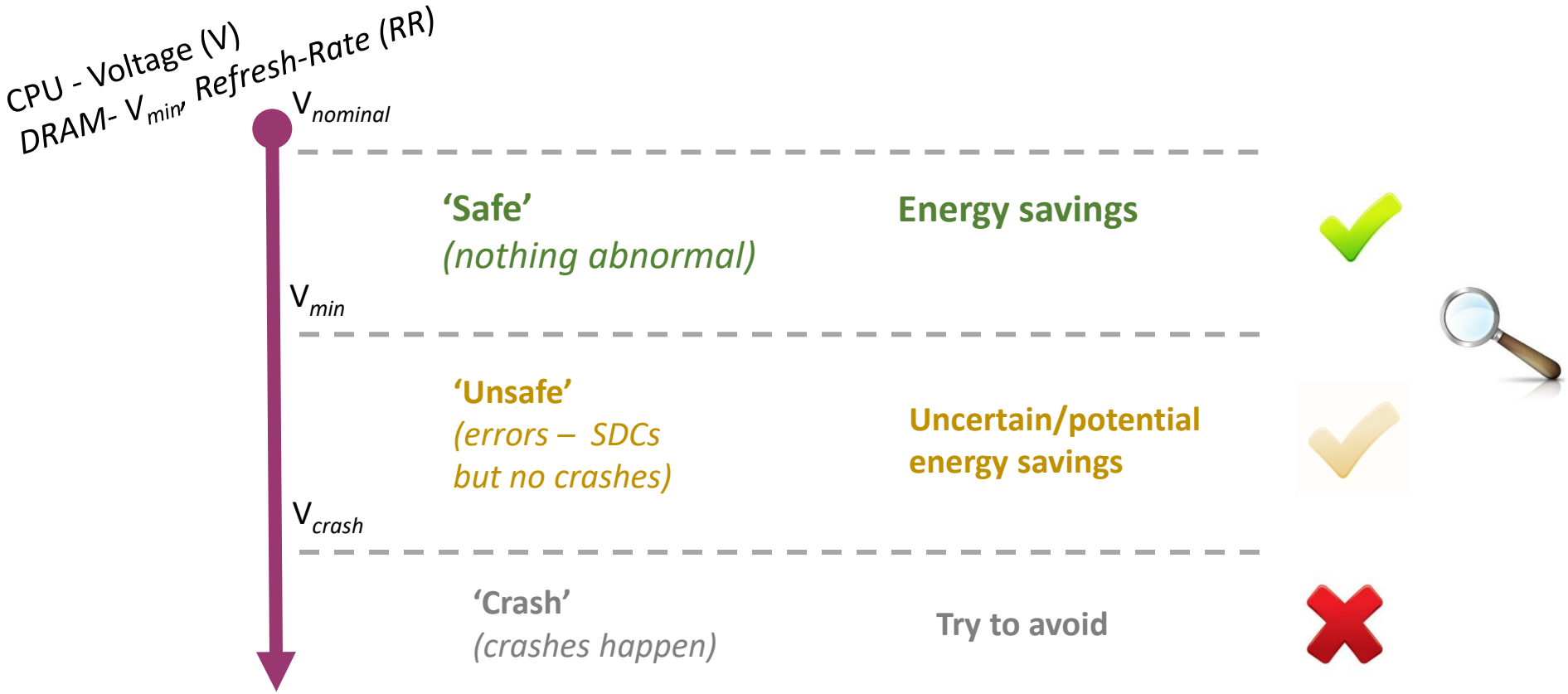
- Any **workload (benchmark or synthetic micro-vius)**
- Any **configuration** (**individual core** or **PMD**, **all cores**)
- **Any voltage and frequency** combination

- Multiple execution of the workload under test with a **step reduction of 5mV**
- **Raspberry Pi** monitors the system for crashes, and if so, it reboots the X-Gene 2.
- **Store the results** in nominal conditions

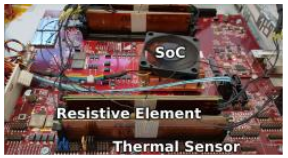
- Fine grain effect classification (**SDC**, **system** or **application crash**, **corrected** and **uncorrected errors**)
- Identification of **Safe (Vmin)**, **Unsafe** and **Crash Regions (System crash)**



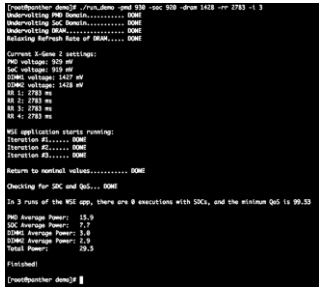
Characterization Target



Cross-System Parameter Variability



- Wireless Denial of Service/Jammer detector
- 4 parallel instances of Jammer Detector, >90% system utilization
  - QOS Constraint: 99%, Attack detection time requirement: 240 ms

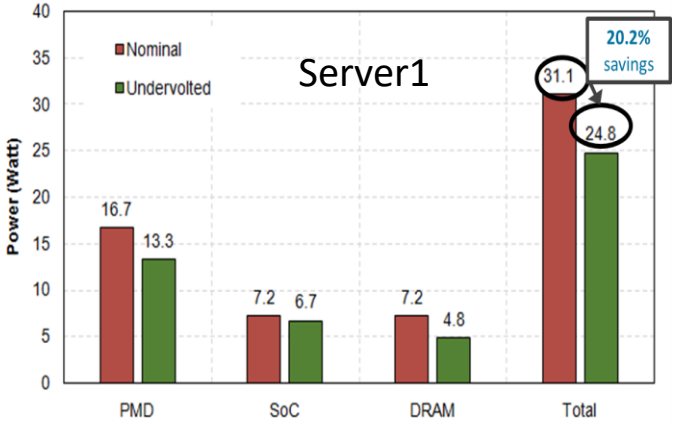


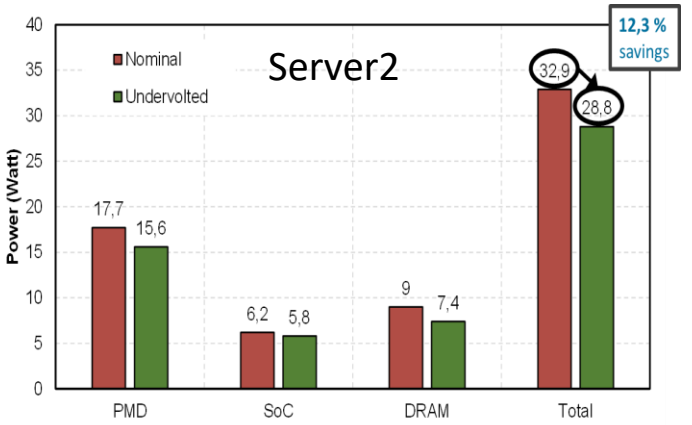
- **Safe Vmin and RRmin** – variation among 3 servers

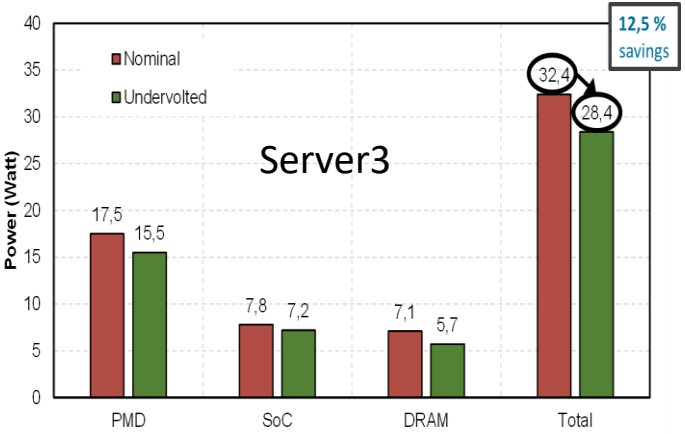
	Nominal	Server1	Server2	Server3
Vmin – PMD	980 mV	930 mV	910 mV	930 mV
Vmin – SoC	950 mV	920 mV	870 mV	900 mV
Vmin – DRAM	1500 mV	1428 mV	1428 mV	1428 mV
RRmin – DRAM	64 ms	2279 ms	2279 ms	2279 ms

K. Tovletoglou, et al., Measuring and Exploiting Guardbands of Server-Grade ARMv8 CPU Cores and DRAMs, IEEE DSN 2018

Cross-System Power Savings Variability



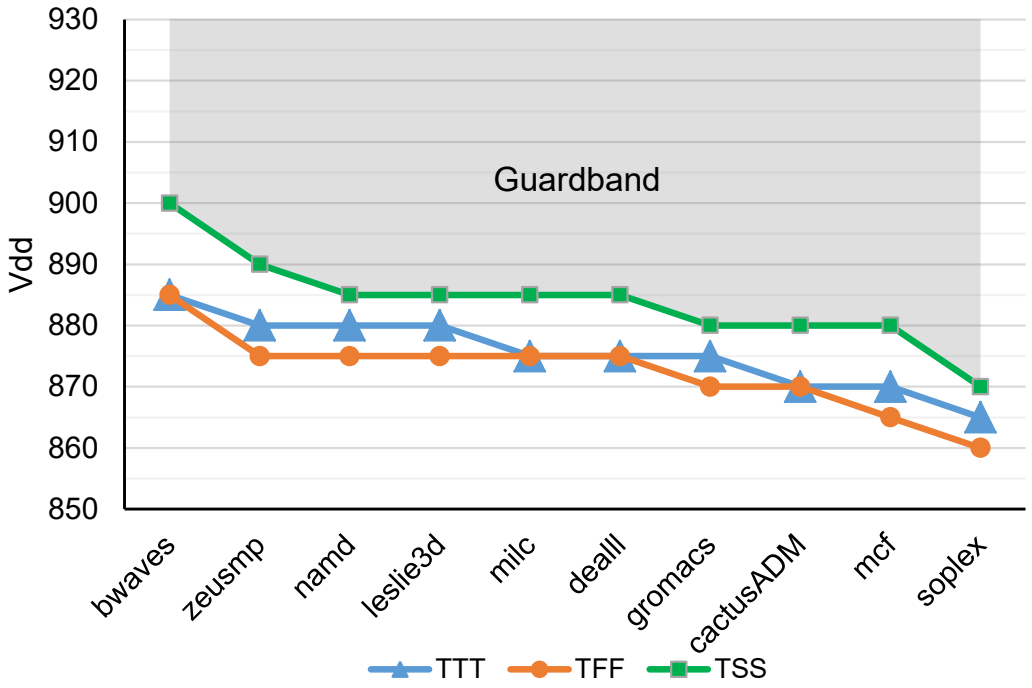




- Up-to 20.2% Power savings
  can be achieved by exploiting the device variability in Tigershark, ARM based servers

*P. Nikolaou,…, 'On the Evaluation of the Total-Cost-of-Ownership Trade-offs in Edge vs Cloud deployments: A Wireless-Denial-of-Service Case Study,' IEEE TSC, 2022

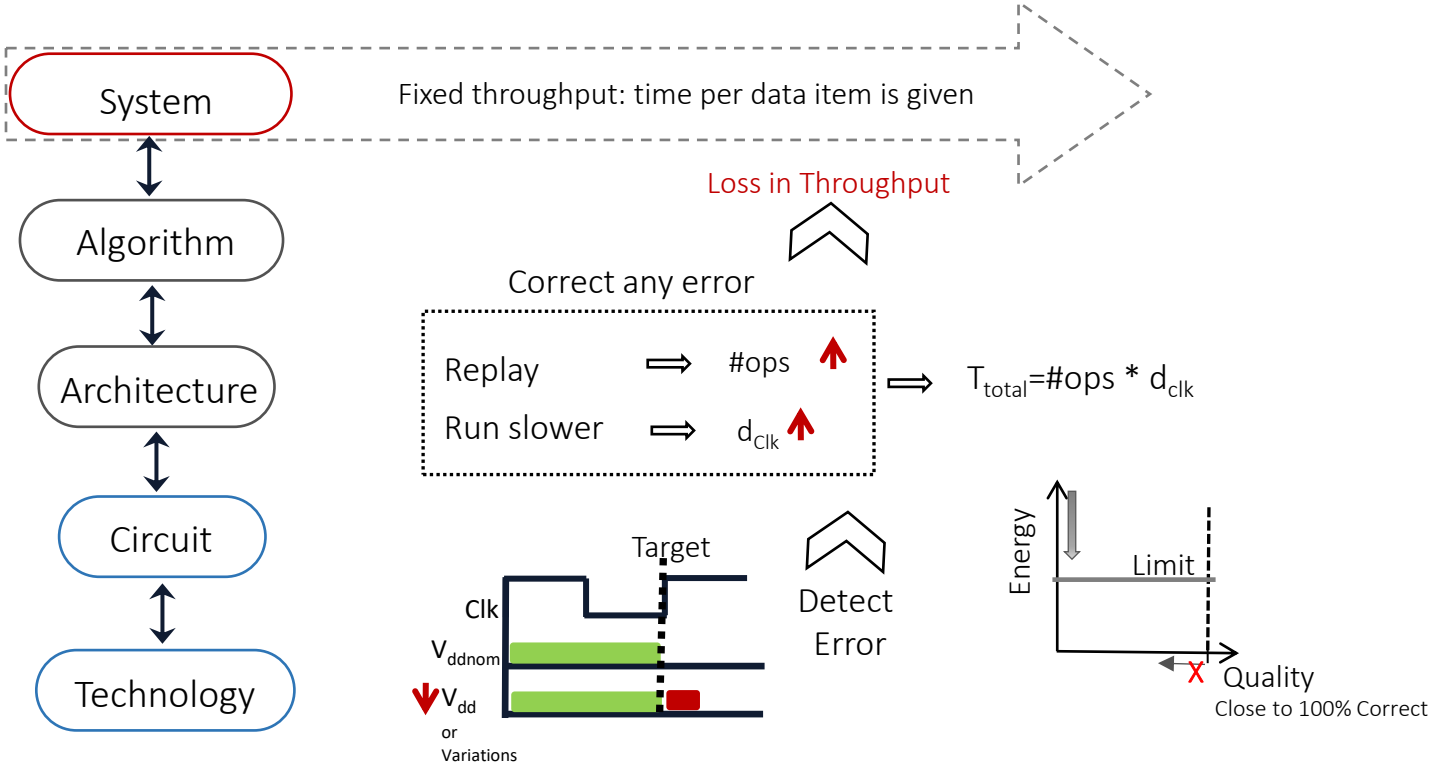
V_{min} results @ 2.4 GHz for 10 SPEC CPU2006



- Voltage guardband: the **safest margin** between *Vnominal* (980mV) and the *Vmin*
  - *Vmin*: lowest voltage without any abnormal behavior)
- There is a relatively **large variation among the chips**
  - V_{min} range = 900mV to 860mV
  - Voltage can be reduced by 18.4% for TTT/TFF and 15.7% for TSS chips
- The workload-to-workload variation is similar across the 3 chips, but large (900mv to 870mV) within each chip
  - Vmin dependency on workloads
    - Instructions and operands

Contents

Beyond Conservative Design

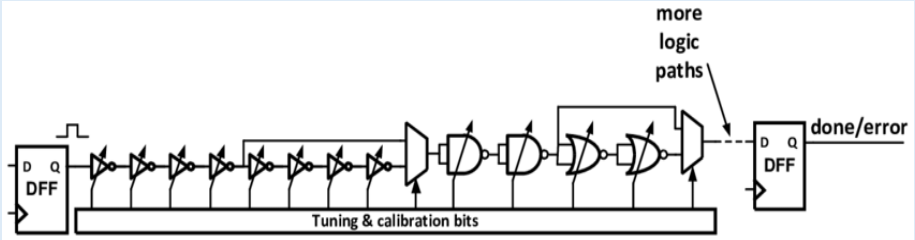


Better-than-worst-case Designs

- Better-than-worst-case designs allow operation beyond the always correct critical timing/voltage margins

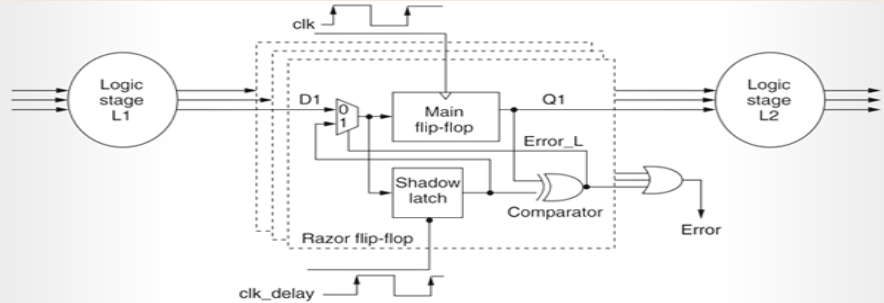
Tunable Replica Circuits

- Mirrors delay of critical paths
- Monitors for errors over voltage/frequency changes



In-situ Error Detection

- Special circuit for dynamic error detection



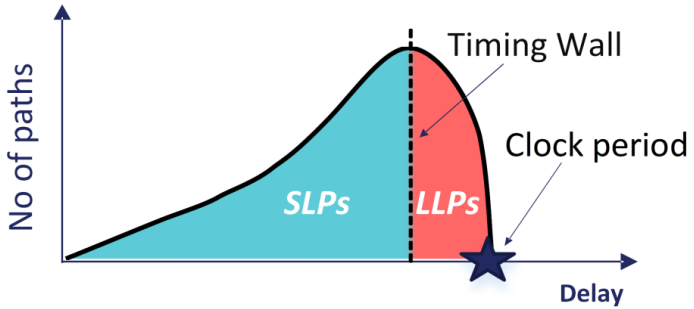
- Different error recovery mechanisms:
  - Pipeline stall
  - Instruction replay
  - Extra clock cycle(s)

High cost error recovery in case of many timing errors

Classical HW Design Flow

- Design flows are classically performance centric and lead to a "timing wall"
  - The inherently fast paths (SLPs) are allowed to become slow (for secondary power optimization)
  - Many long paths from every pipeline stage are close to the target clock period

  - ➢ Under any delay variation there is an increased probability for massive timing failures
  - ➢ Conservative design: require error avoidance schemes in many stages/paths
  - ➢ Increased redundancy and overheads

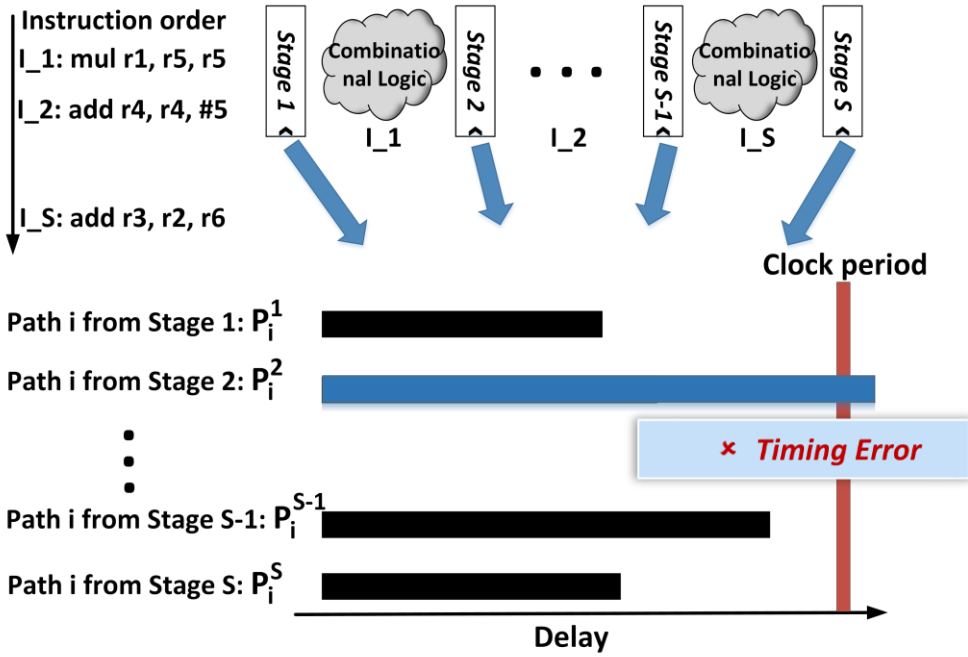


*LLPs: Long Latency paths
*SLPs: Short Latency paths

*I. Tsiokanos, et al, *Hardware Level Approximations*, Book Chapter in Book: Approximate Computing Techniques, Springer, 2022

Error Manifestation

- Challenging to identify where error may occur
- Long latency paths distributed across stages in pipelined cores
  - Require error mitigation mechanisms within each stage
- Long latency paths are activated by few instructions and certain operands
  - →difficult to identify/predict
  - →No workload awareness



*I. Tsiokanos, et al, *Hardware Level Approximations*, Book Chapter in Book: Approximate Computing Techniques, Springer, 2022

Contents

Proposed Approach

- I. Core redesign for Error Resilience
  - Re-design the target circuit to obtain a path distribution with less LLPs
  - Isolate the Long latency paths (LLPs) to few (single) pipeline stage(s)

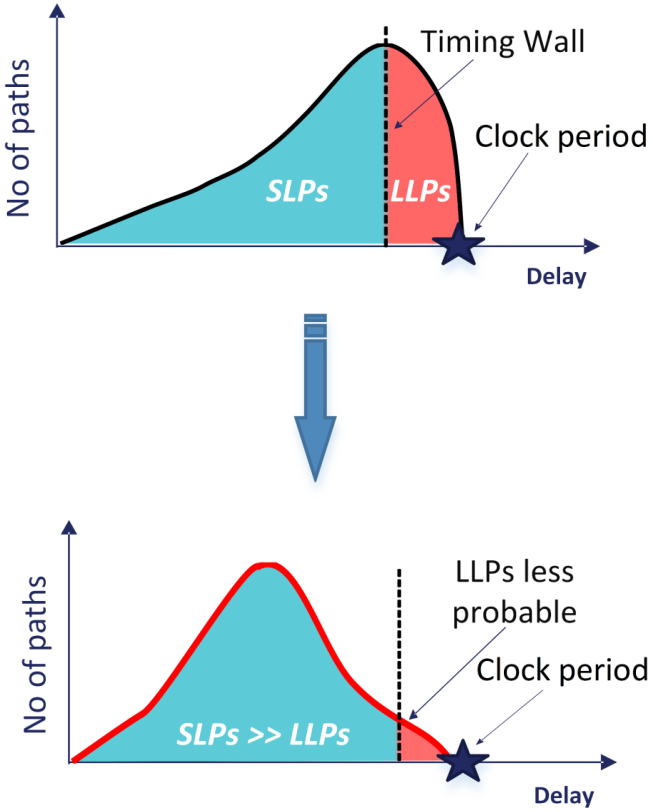
- II. Dynamic data-aware prediction of the error-prone Long Latency Paths (LLPs)
  - Identify instructions and operands that activate LLPs at runtime with low cost mechanisms

- III. Dynamic Error Mitigation
  - Dynamic Cycle Adjustment: Provide an extra cycle for the completion of the long-latency paths that are predicted to be excited
  - Dynamic Operand Truncation: Dynamically set to 0 only the less significant bits (LSB) of the operands of the specific instruction(s) that activate(s) the error prone paths

- **Target:** Minimization of timing failures while avoiding the use of pessimistic guardbands with low cost error mitigation applied only where/when needed

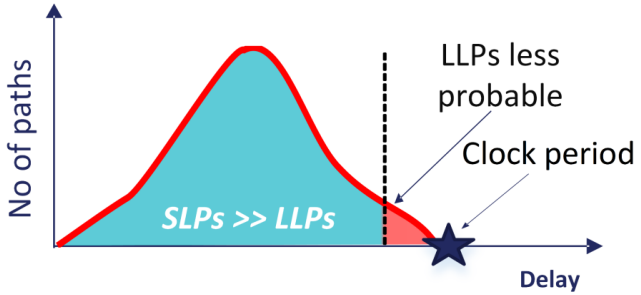
I. Core Re-design for Resilience

- Apply path shaping at design time
  - Re-design the target circuit to obtain a path distribution with less LLPs
    - path group constraints
  - Limit the LLPs on specific pipeline stages for facilitating the use of any error mitigation scheme in few points
    - Understand the target architecture and move logic to ‘relaxed’ pipeline stages, while avoiding any delay penalty



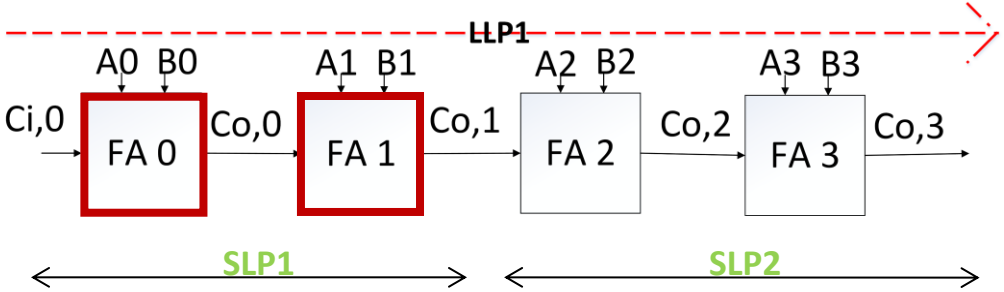
II. Dynamic Prediction of LLPs

- **Observation:** the error-prone long latency paths (LLPs) are excited by specific instructions and data



- **Main Idea:** Detect instructions and operands that excite the LLPs
  - ➢ by monitoring the carry propagate signals

- Ripple carry adder example

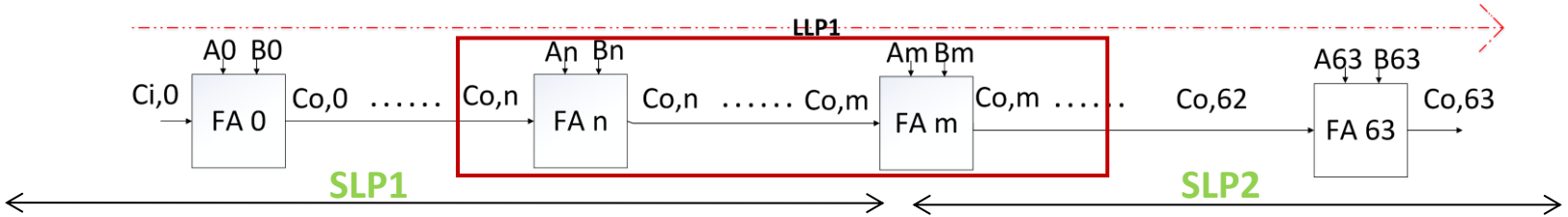


- Most timing critical path (LLP1): Carry propagation from Ci,0 to Co,3
- Identify LLPs activation by monitoring Carry propagation at FA0 & FA1:

$F(0,1) = (A0 \oplus B0) \bullet (A1 \oplus B1)$

- $F(0,1) = 1$ Activation of Long Latency Paths (LLPS)
- $F(0,1) = 0$ Activation of Short Latency Paths (SLPs)

II. Dynamic Data Aware Prediction of LLPs

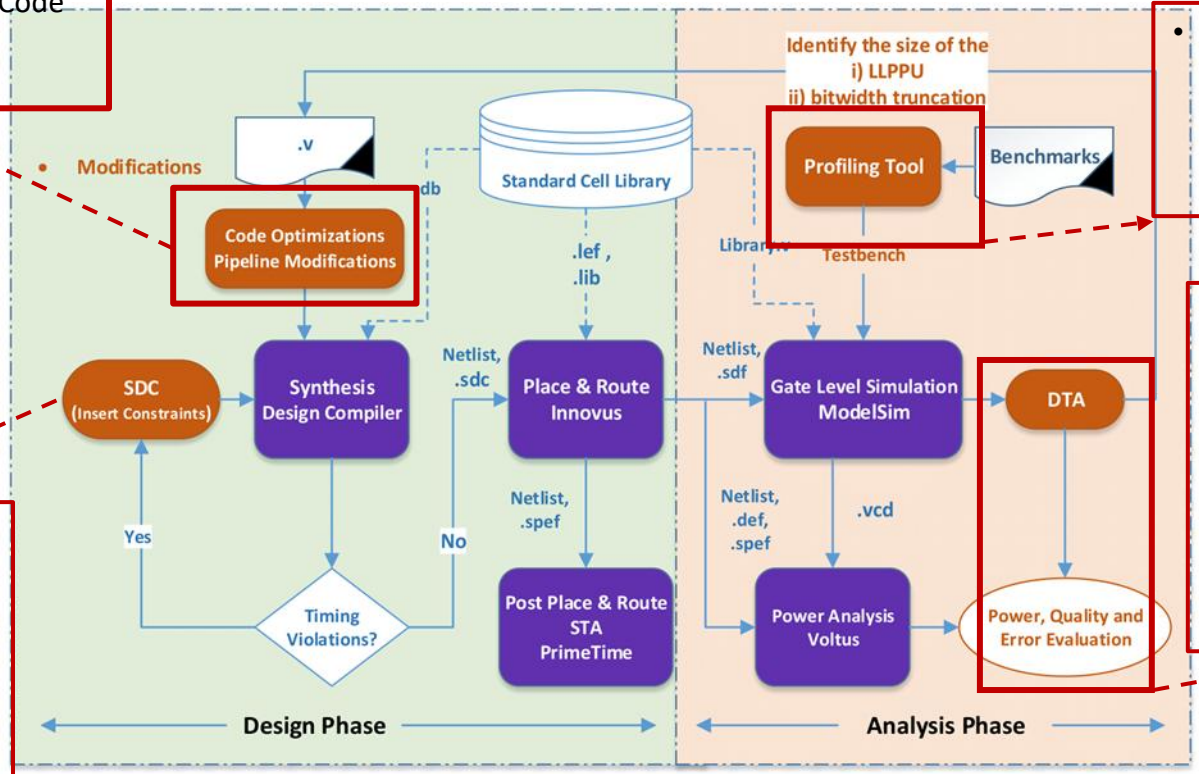


- Identify LLPs activation by monitoring m - n FAs :
  - F(n,m) = (An⊕Bn) ● (An+1⊕Bn+1) ● ... • (Am-1⊕Bm-1) • (Am⊕Bm)

- Tradeoffs in such a circuit:
  - Increase the number of (m-n) bits → Decrease the activation probability of LLPs
    - Carry propagation of 1 FA = $2(1-p)p$
    - Carry propagation of 4 FAs = $(2(1-p)p)^4$
  - Increase the number of monitored bits → Increase the area/power dissipation

Design and Analysis Flow

- Microarchitectural and Code optimizations
- Extra optimization step



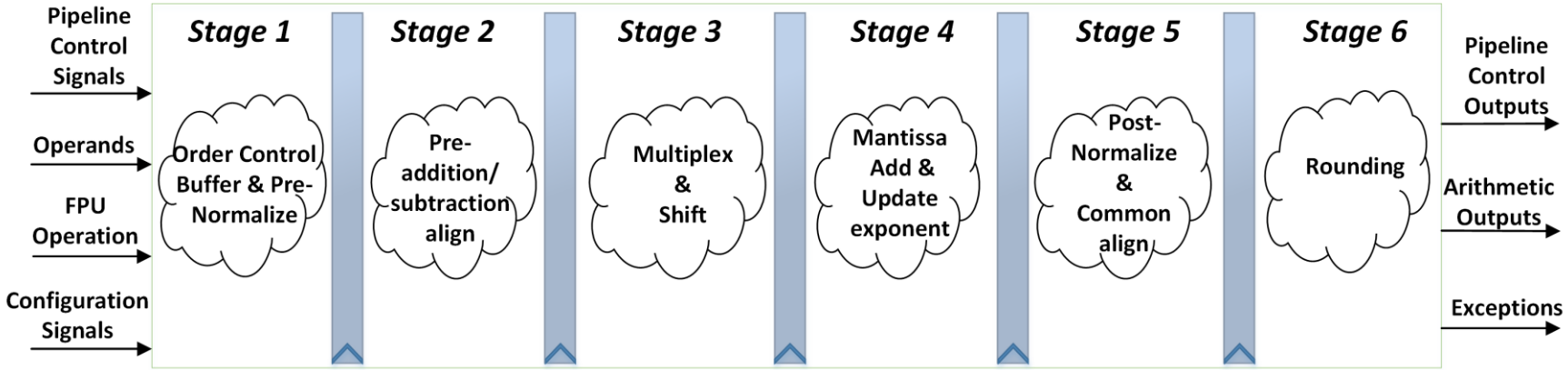
- A profiling tool for extracting instruction traces and operands as are being executed on real processors

- Exploit the dynamic data-dependent path activation
- Simulate the design under potential timing variations
- Power, quality, error and dynamic power measurements

- Introduction of path-group constraints
- Make long paths smaller in number
- Not allow the tool to make the naturally fast paths slower

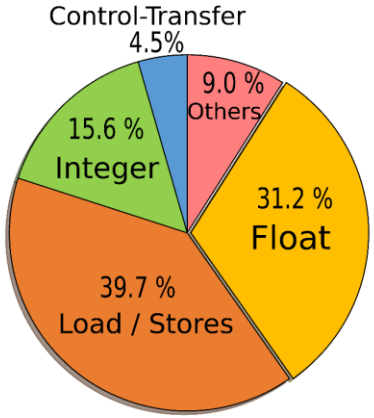
Case Study: *Application to an FPU*

- Application to the mor1kx marocchino pipeline FPU (single/double precision)
  - based on OpenRISC ISA
  - 6 pipeline stages

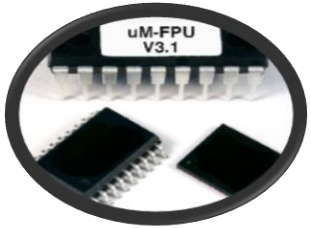


Case Study: *Application to an FPU*

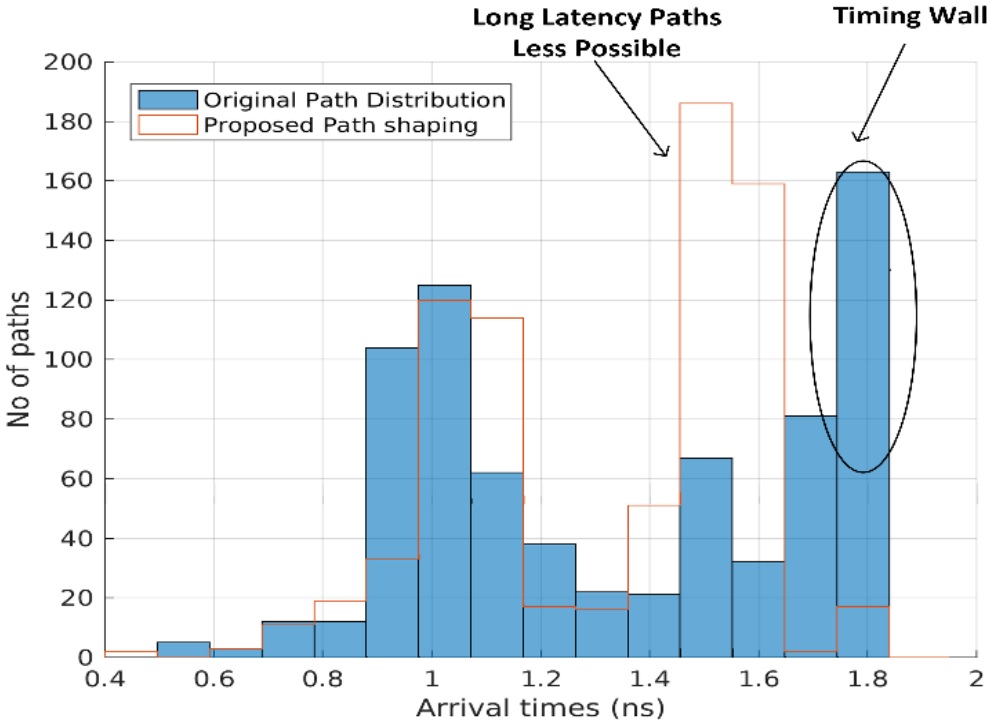
- FPUs Excellent representatives of modern RISC pipelined designs, implemented as co-processors
- Among the most time consuming with long latency paths
- Floating point operations dominant in various apps
  - Considered apps:
    - K-means, CFD and Heartwall from Rodinia Suite,
    - Raytrace, Face-detection from Parsec Suite



Average instruction distribution



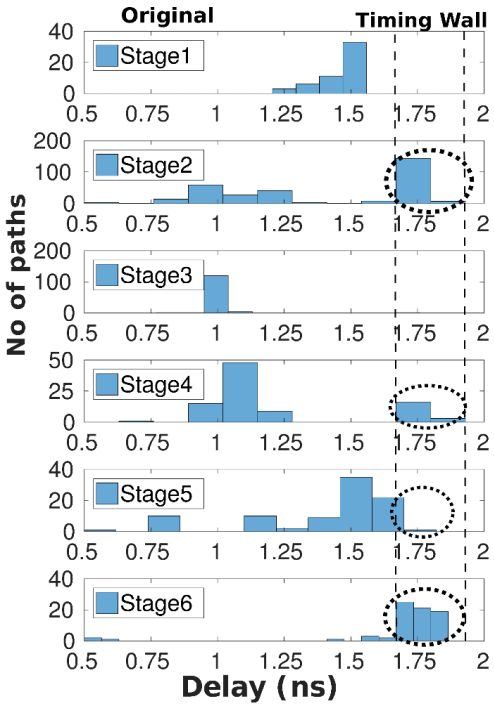
Path Distribution of the Re-designed FPU



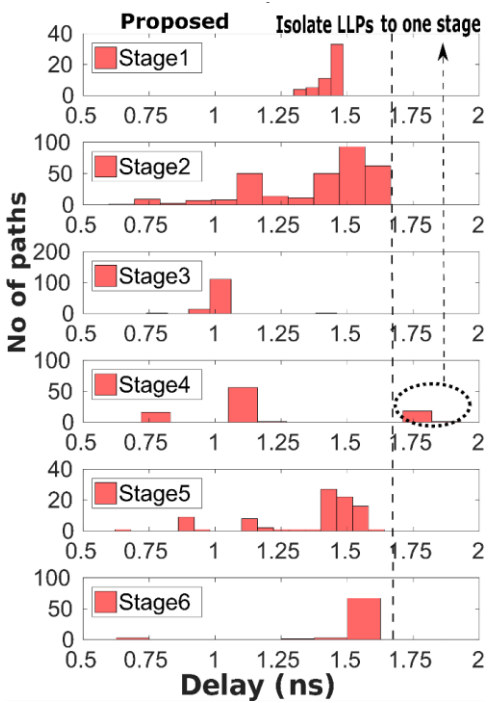
Path Distribution of the Re-designed FPU

- Original Design
  - Timing wall
  - LLPs in 4 out of 6 stages
  - Increase probability of failures
- Proposed design
  - Number and probability of LLPs have been reduced
  - LLPs restricted only to Stage 4
  - Stage 4: mantissa addition

Original path distribution among the stages



Proposed path distribution



Long Latency Paths Prediction Unit (LLPPU)

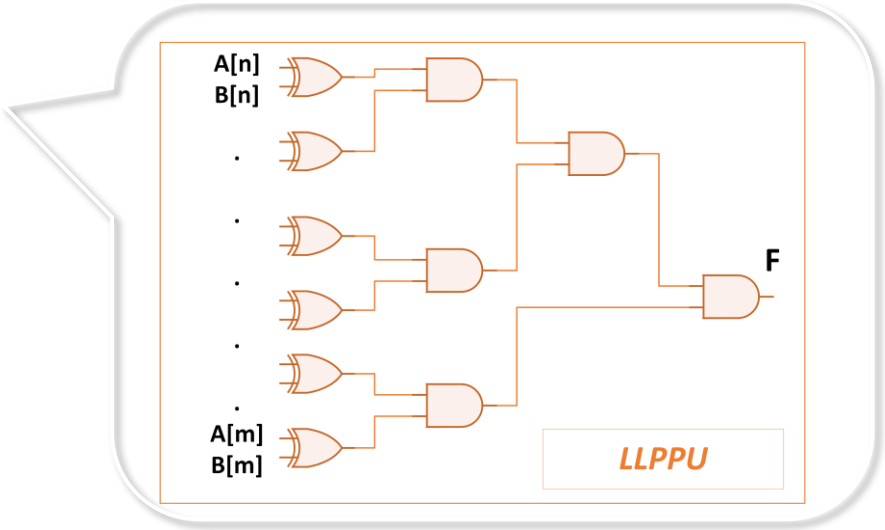
- Adoption of a prediction unit for detecting LLPs activation at Stage 4

- Low complexity unit:
  - F(n,m) = (An⊕Bn) • … • (Am⊕Bm)

number of (m-n) bits ↑

↓

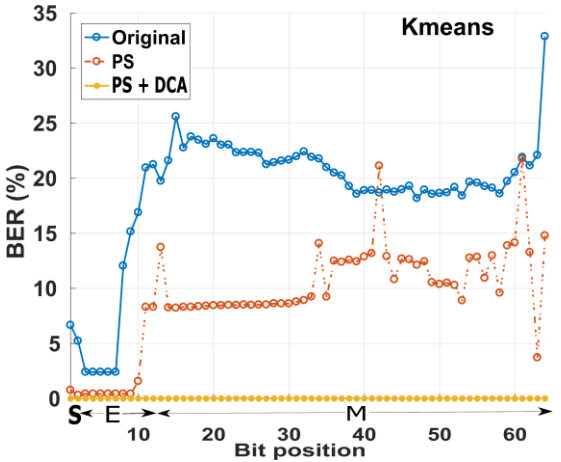
activation probability of LLPs (P(L)) ↓

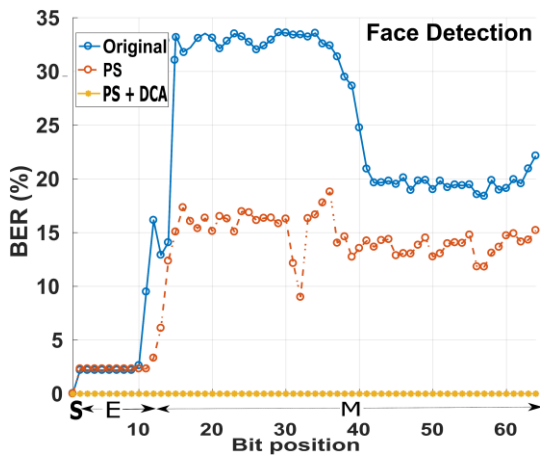


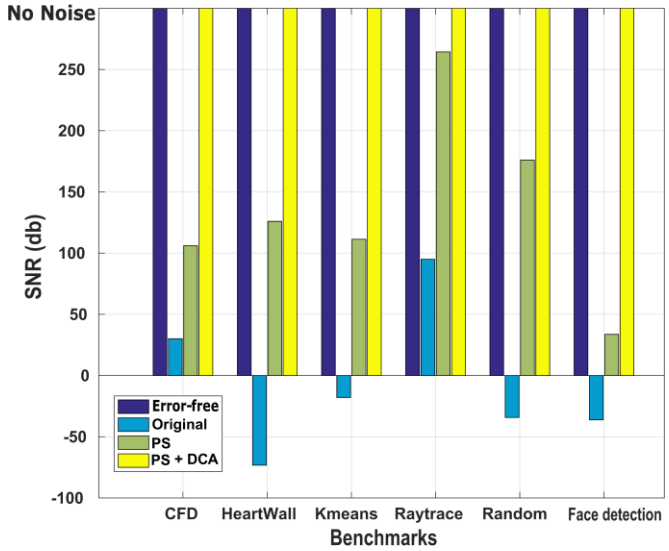
(n,m)	(27 , 34)	(27 , 33)	(28 , 33)	(28 , 32)
P(L) (%)	0.62	1.1	2.71	4.98

III. Dynamic Cycle Adjustment

- In case of path activation prediction, allow an extra clock cycle for the completion of the error-prone instruction
  - Provides enough time for the potential long latency instruction to complete correctly under any delay increase (e.g. 8%) due to VoS and/or process variations







III. Dynamic Cycle Adjustment

- Throughout penalty due to extra clock cycles
  - Limited (0.61% on average for different applications) due to rare activation of critical paths that are restricted in one stage

- Area penalty due to the error prediction unit
  - Limited (0.31%) due to low cost by design and integration only in one stage

- In comparison with voltage based guardbands can obtain large power savings by allowing operation at lower voltage (0.95V)

Benchmarks	Kmeans	CFD	Heartwall	Raytrace	Face Detection	Ranodm
Power gains (%)	38.1	39.02	30.65	43.1	42.22	34.63

Contents

Proposed Approach

- I. Core redesign for Error Resilience
  - Re-design the target circuit to obtain a path distribution with less LLPs
  - Isolate the Long latency paths (LLPs) to few (single) pipeline stage(s)

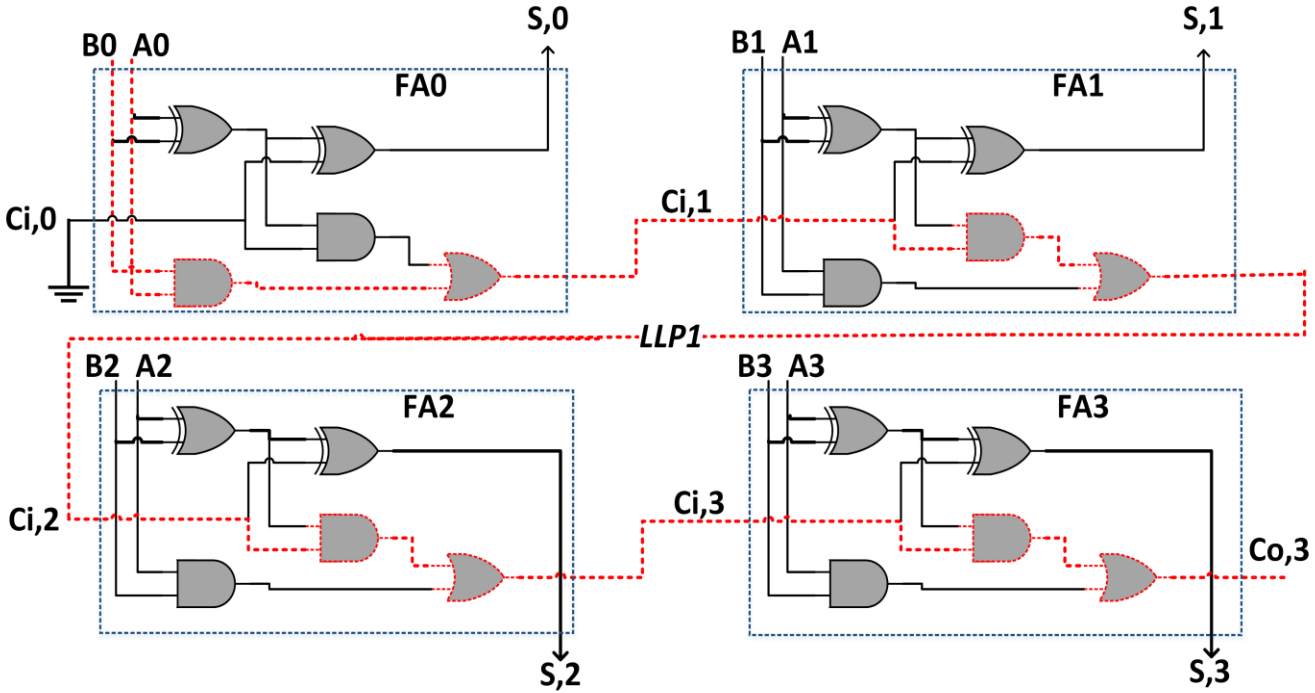
- II. Dynamic data-aware prediction of the error-prone Long Latency Paths (LLPs)
  - Identify instructions and operands that activate LLPs at runtime with low cost mechanisms

- III. Dynamic Error Mitigation
  - Dynamic Cycle Adjustment: Provide an extra cycle for the completion of the long-latency paths that are predicted to be excited
  - Dynamic Operand Truncation: Dynamically set to 0 only the less significant bits (LSB) of the operands of the specific instruction(s) that activate(s) the error prone paths

- **Target:** Minimization of timing failures while avoiding the use of pessimistic guardbands with low cost error mitigation only where needed

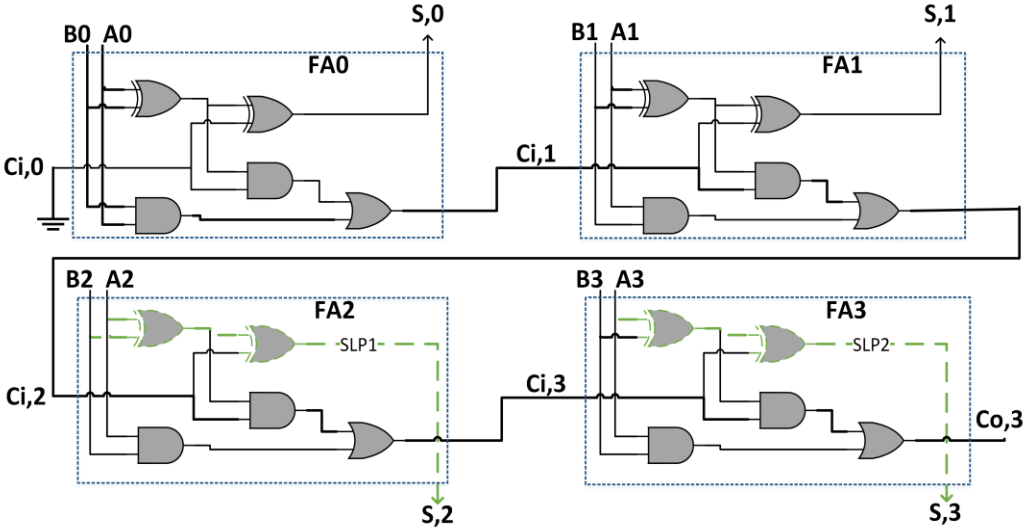
Impact of Bitwidth Truncation on Delay

- 4-bit Ripple Carry Adder (RCA)
- Longest path: LLP1
  - Carry propagation from Ci,0 to Co,3
- Assuming Gate delay: T
  - ***Delay of LLP1: 8T to complete***
- Worst case delay activated by only few inputs, e.g.
  - A = 1111
  - B = 0001



Impact of Bitwidth Truncation on Delay

- 4-bit Ripple Carry Adder (RCA)
- If we truncate 2 LSBs:
  - A = 1100
  - B = 0000
- Only short latency paths excitation (SLP1, SLP2)
- **Delay of SLP1 (or SLP2) reduced to = 2T**



- ❑ *Delay of original data: 8T*
  A= 1111, B = 0001
- ❑ *Delay of truncated data: 2T*
  A=1100, B = 0001
- ✓ *Safety Timing margins*
- × *Deterministic quality loss*

Random vs Deterministic Quality Loss

- Timing Errors → Random bit errors incur high quality loss
- Data truncation → Deterministic errors induced by setting to "0" some LSBs
- Addition: A = 11110 (decimal: 30) + B = 00001 (decimal: 1)
  - Output C = 11111 (decimal: 31)

Quality loss – Timing Errors

- ***Random bit errors can affect MSBs***
- ***Quality loss is expected to be high***
- ***Timing error at MSB of output C:***

C = 01111 (decimal 15)

Reference Output: 31

Quality loss – Bitwidth Truncation

- ***Minimizes timing errors at MSBs***
- ***Any quality loss can be controlled***
- ***2 LSBs truncation: A = 11100, B = 00000***
- ***No timing errors in the output C:***

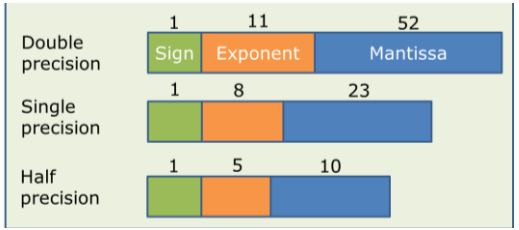
C = 11100 (decimal 28)

Reference Output: 31

Applying Approximation Dynamically

- Truncation and reduced precision can help avoid any potentially catastrophic random errors, under VoS/variations by providing timing slack against the original design

- Existing schemes have tried to apply precision scaling:
  - statically on all input/output data
  - agnostically of the underlying hardware

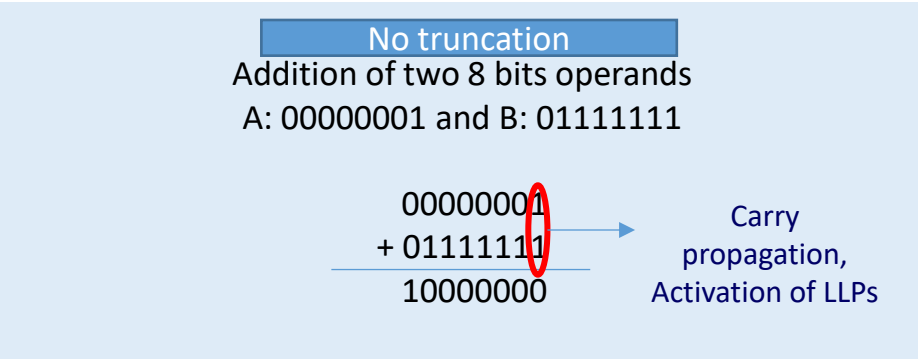


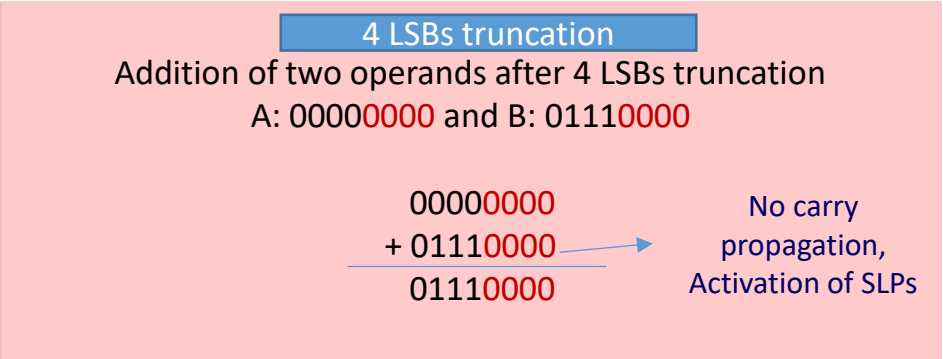
→Approximating all data/variables can lead to unnecessary loss since only few (rarely executed) data activate the (few) critical paths of a design

→Approximation is really needed (to obtain timing slack/avoid errors) only in few cases when certain operands are processed and activate the critical path

III. Proposed Dynamic Operand Truncation

- Utilize the proposed mechanisms to detect instructions and operands that excite the long latency paths by monitoring the carry propagate signals (simple low cost circuitry (a xor b))
- Truncate the operands by setting to 0 only the LSBs of the operands of the specific instruction(s) that activate the LLPs

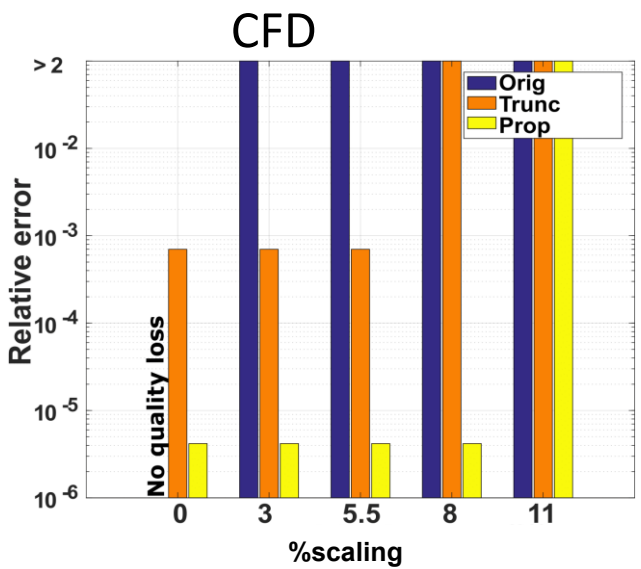


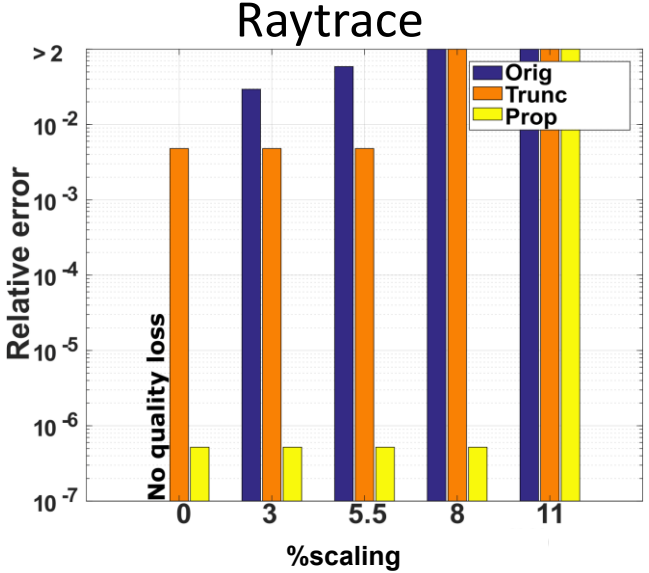


- The accuracy due to truncation is minimized by dynamically identifying the instruction and operands that activate the long latency paths

Evaluation Results – Quality loss

- Relative Error (RE)
  - $RE = \left|\frac{Ogold - Osim}{Ogold}\right|$
- Orig:
  - Unacceptable quality loss even under small delay increase/voltage decrease
- Trunc: static bit truncation in the mantisa
  - Considerable, unnecessary quality loss
- Prop:
  - Minimization of quality loss up-to 8% delay increase (7 orders of magnitude lower RE than static Trunc)
- Under 11% scaling all designs incur unacceptable quality loss
  - Design choice (number of truncated bits), can be altered at design phase





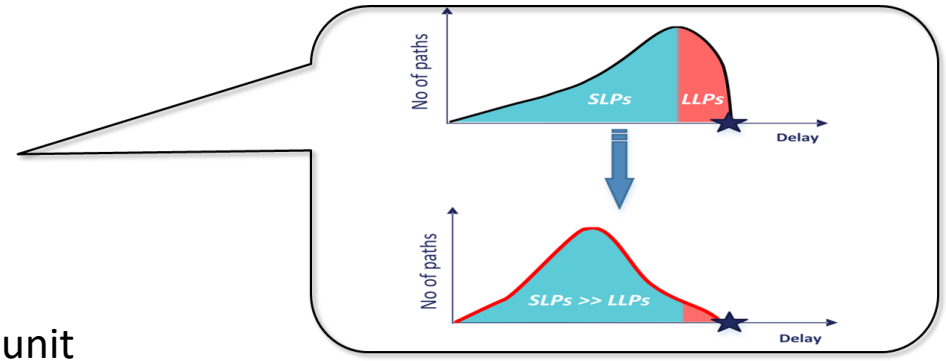
*I. Tsiokanos, L. Mukhanov, G. Karakonstantis, "Low-Power Variation-Aware Cores based on Dynamic Data-Dependent Bitwidth Truncation", IEEE Conference on Design Automation & Test in Europe (DATE), 2019

*I. Tsiokanos, L. Mukhanov, G. Karakonstantis, "Significance-Driven Data Truncation for Preventing Timing Failures", IEEE Trans on Reliability (TDMR), 2020

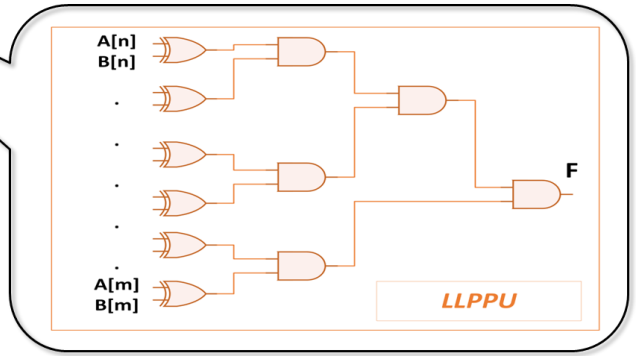
Area & Power Penalties

- Integration of :

1. Path shaping technique



2. Long latency paths prediction unit



- Area overhead: 0.34%
- Power overhead (at nominal): 4.48% on average

Comparison with a Guardband based scheme

- Guardband based design: up-scale voltage to avoid failures
  - Fast corner cell library (1.25V)

- Proposed: up-to 44% power gains and minimal quality loss

Applications		Power Gains in Prop vs Guardbands (%)
K-means		41.73
Heartwall		44.3
Raytrace		38.76
CFD		38.78

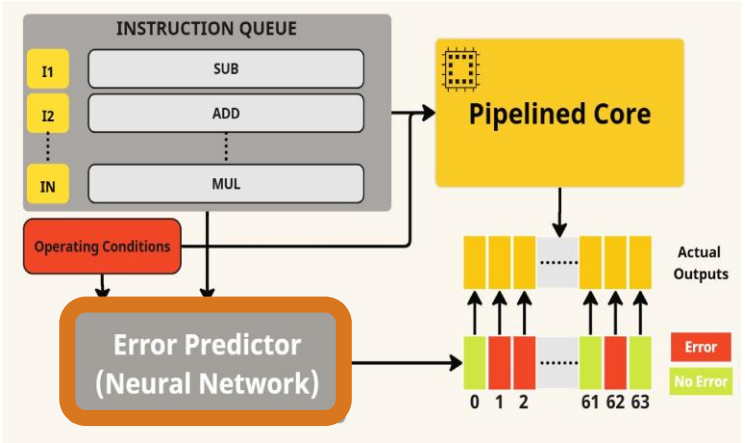
Contents

- Challenges
  - Low Power
  - Variations, Faults
- Conventional Design
- Exposing Pessimism in ARM Processors
- Beyond Conservative Design

- **Proposed Approach**
  - Circuit-Architecture Redesign for Data Aware Voltage Scaling and Resilience
  - Dynamic Cycle Adjustment
  - Dynamic Operand Truncation
  - **AI based Error Prediction**

AI Based Error Prediction

- Utilize AI methods and aim at the prediction of errors at the output of each executed instruction on the monitored core
- Consider a short instruction history, the opcode and operands of each instruction, the voltage point and bitwidth (32bits or 64bits)
- The developed model can be added as a co-processor next to the monitored core



- Main Challenges
  - Development of a compressed and accurate model with low hardware cost
  - Appropriate datasets with instruction and data aware error labels

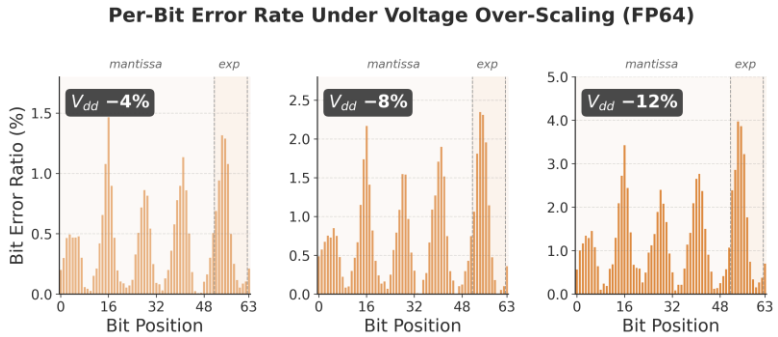
*G. Chatzitsompanis, G. Karakonstantis, 'ePredictNet: Low Cost Error Prediction Neural Network', IEEE ISLPED 2024
S. Tompazi, I. Tsiokanos, J. Martínez del Rincón, G.Karakonstantis, 'Estimating Code Vulnerability to Timing Errors Via Microarchitecture-Aware ML', IEEE Design & Test 2023

Proposed ePredBitNet Framework

- As use case focused on the prediction of bit-level errors in vulnerable FP instructions of a pipelined FPU of the (post-P&R) OpenRISC Mor1kx Marocchino

1. Generate a new Dataset

- Error estimation (labels) based on gate level timing simulations of ~9M instructions from various benchmarks under 3 scaled voltages
- Error rate **increase under VOS** — **non-uniformly** across bits



2. Model Training - Teacher

- **Full-precision Multi-Layer Perceptron (MLP)** D→64→64→64 + residual
- **Class-weighted Binary Cross-Entropy (BCE)** training

3. Knowledge Distillation

- **1-bit Quantization (BNN)** with ~30k parameters
- **KD + Exponential Moving Average** calibration

4. FINN design

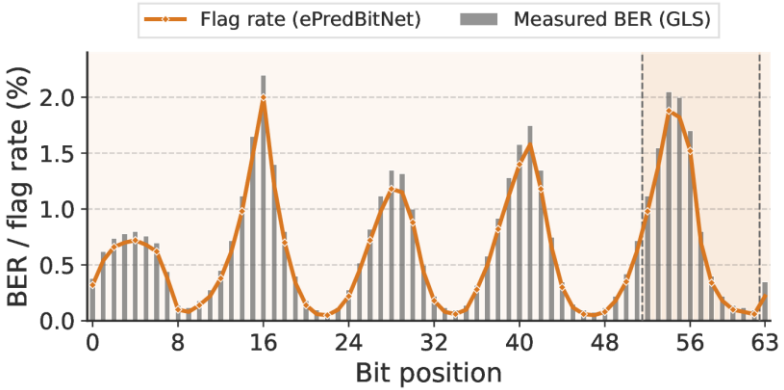
- **Streaming bit-op RTL** via FINN
- **XNOR + POPCOUNT** — LUT-only

*G. Chatzitsompanis, N. Kostakis, G. Karakonstantis, 'ePredBitNet, *Compressed Bit-LevelTiming-Error Predictors via Binary Neural Networks',* IEEE VTS 2026

Results — Prediction Quality & Hardware Cost

- The developed model achieves high bit-level accuracy
  - precision **≥ 0.97**
  - micro-F1 = **0.81–0.86**
  - ROC-AUC ≈ **0.9998**
  - False-Positive Rate = **0.01%**

- On a Xilinx Zynq Ultra Scale+ ZCU-104 FPGA,it occupies small number of resources
  - 72% less than Dual lock step and 85% less than Triple modular redundancy (TMR)
  - 27.9% of the OpenRISC core

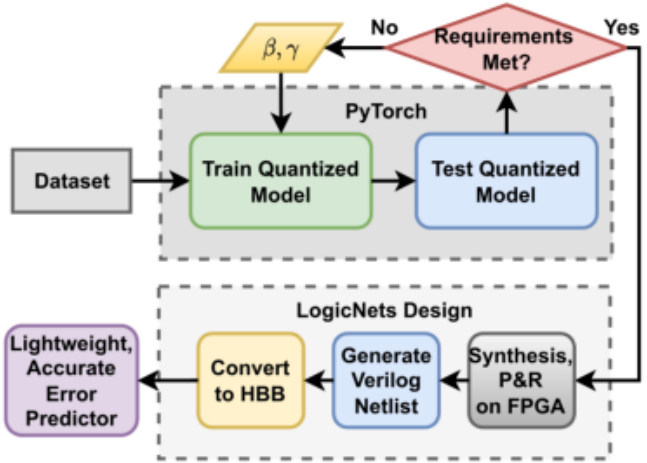


	ePredBitNet	DCLS	TMR
LUTs	**9,647**	~34k	~3× FPU
FFs	**12,721**	—	—
BRAM / DSP	**0 / 0**	—	—
Core overhead	**27.9%**	~100%	≥200%
Mode	**Proactive**	Reactive	Reactive

ZCU-104 FPGA • 250 MHz • 3.42 M inf/s • 0.246 mW

Proposed Value–Level Error Predictor

- Simplify the model by aiming at prediction of erroneous FP (rather than complex bit-level predictions)
- 3-layer Full-precision Multi-Layer Perceptron (MLP)
  D→64→32→32 with Relu activations
- Quantized model training with LogicNets framework
- Average accuracy 99.39% and ROC-AUC ≈ **0.9910**



- 960MHz with a 6 clock cycles latency
- Only 270 LUTs and 219 FFs with only 2.1% area and 5% power overhead compared to the overall OpenRISC core

*G. Chatzitsompanis, G. Karakonstantis, 'ePredictNet: Low Cost Error Prediction Neural Network', IEEE ISLPED 2024

Conclusions

- Need to depart from conventional worst case performance centric hardware design flows
- Re-design circuits and architectures targeting resilience-by-design
- Framework for avoiding timing errors in pipelined cores by:
  - Detecting/predicting the excitation of the error-prone long latency paths
  - Redesigning the target circuit in a way that isolate the LLPs in a single stage
  - Dynamically adjusting the clock or truncating the bitwidth only of the operands that activate the isolated LLPs
- Up-to ~40% power savings at 0.34% area overhead and minimal quality loss
- AI enables bit-level and instruction-level error prediction and opens the avenues for proactive error protection schemes that can complement existing mechanisms for more efficient fine-grain error mitigation schemes

Thank You!

Acknowledgements: H.F.R.I. Project - IoTHealth (no. 25267)

EU Project - COIN-3D (no. 101159667)

Hellenic Chips Competence Centre (HCCC)

Contact
Georgios Karakonstantis - gkarakon@uth.gr