



MICRO-50, Boston, October 2017

Tutorial:

Microarchitecture Level Reliability Assessment

Throughput and Accuracy
<http://micro50-tutorial.di.uoa.gr/>

Organizers/Presenters:

Athanasios Chatzidimitriou, Manolis Kaliorakis, Dimitris Gizopoulos



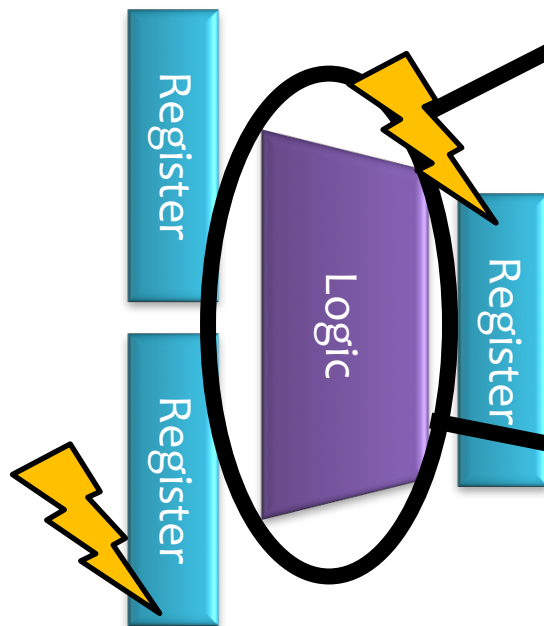
MICRO-50, Boston, October 2017

Part 2:

Microarchitecture-level Fault Injection

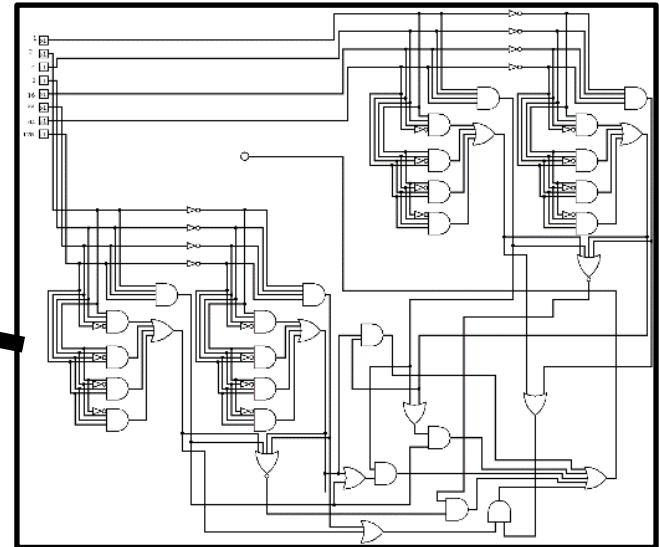
Microarchitecture-level simulation

- **Performance** models
- **Pipeline** & hardware description
- **Memory** elements (arrays, flops)



```
template<class InputString, class OutputString>
bool unhexlify(const InputString& input, OutputString& output) {
    if (input.size() % 2 != 0) {
        return false;
    }
    output.resize(input.size() / 2);
    int j = 0;
    auto unhex = [](char c) -> int {
        return c >= '0' && c <= '9' ? c - '0' :
               c >= 'A' && c <= 'F' ? c - 'A' + 10 :
               c >= 'a' && c <= 'f' ? c - 'a' + 10 :
               -1;
    };
    for (size_t i = 0; i < input.size(); i += 2) {
        int highBits = unhex(input[i]);
        int lowBits = unhex(input[i + 1]);
        if (highBits < 0 || lowBits < 0) {
            return false;
        }
        output[j++] = (highBits << 4) + lowBits;
    }
    return true;
}
```

μarch



RTL

RTL simulation

- Accurate **RTL** model
- **Slow** throughput
 - Small **simulation window**
 - (100K-200K)
- Hardware **observation points**
 - Hardware layer observability
 - CPU as a **hardware block**

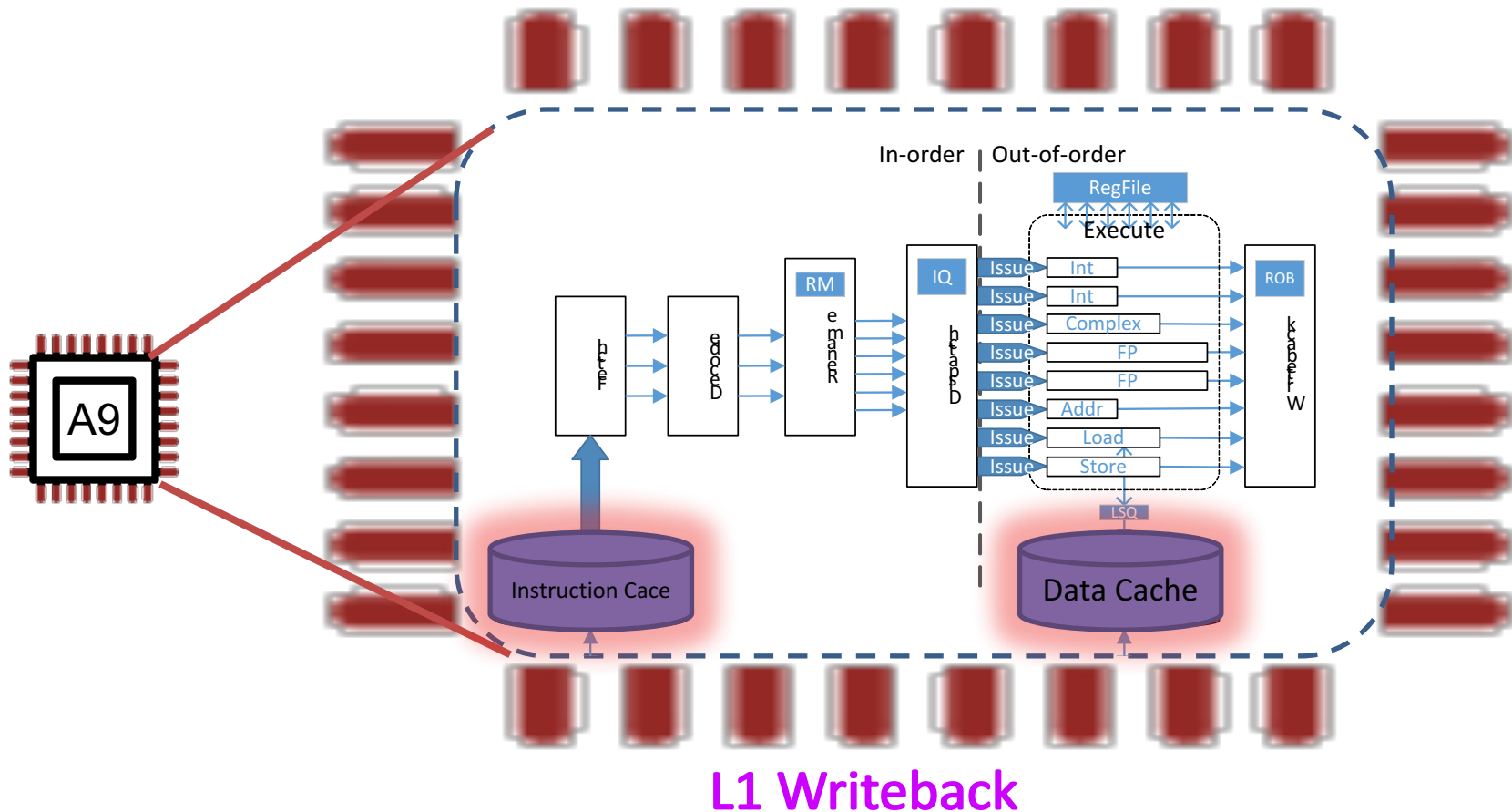
Benchmark	Speed (seconds/run)		
	RTL	GeFIN	Ratio
FFT	7001	39.1	179
qsort	3157	23.9	132
sha	3421	8	427
s corners	1019	3.2	315
s edges	874	3.6	242
s smooth	893	3.7	241
average			256

* RTL model: ARM Cortex-A9
Microarchitecture model: Gem5 Cortex-A9

* A.Chatzidimitriou, M.Kaliorakis, D.Gizopoulos, M.Iacaruso, M.Pipponzi, R.Mariani, S.Di Carlo, "RT Level vs. Microarchitecture Level Reliability Assessment: Case Study on ARM Cortex-A9 CPU", DSN-w, 2017

RTL observation points

- Cortex A9 example



Microarchitecture-level simulation

Choosing the simulator

Simulator	x86 ISA ?	ARM ISA ?	Full system	Public?	Maintained?
Flexus	✓	✗	✓	✓	✗
Gem5	✓	✓	✓	✓	✓
GEMS	✗	✗	✓	✓	✗
MARSS	✓	✗	✓	✓	✓
OVPsim	✓	✓	✗	✓	✓
PTLsim	✓	✗	✓	✓	✗
Simics	✓	✓	✗	✗	✓



Fault Injection Frameworks

- **MaFIN:** MARSS-based Fault Injector ^[1]
- **GeFIN:** Gem5-based Fault Injector ^[2]

	GeFIN	MaFIN
ARM ISA	✓	✗
x86 ISA	✓	✓
Cycle accurate	✓	✓
Detailed CPU model	✓	✓
Full system	✓	✓
Deterministic	✓	✗
Active maintenance	✓	✗
Availability of data in components	✓	✗

[1] Foutris, et al., "Versatile architecture-level fault injection framework for reliability evaluation: A first report", IOLTS, 2014

[2] Kaliorakis, et. al., "Differential Fault Injection on Microarchitectural Simulators", IISWC, 2015



Microarchitecture-level simulation

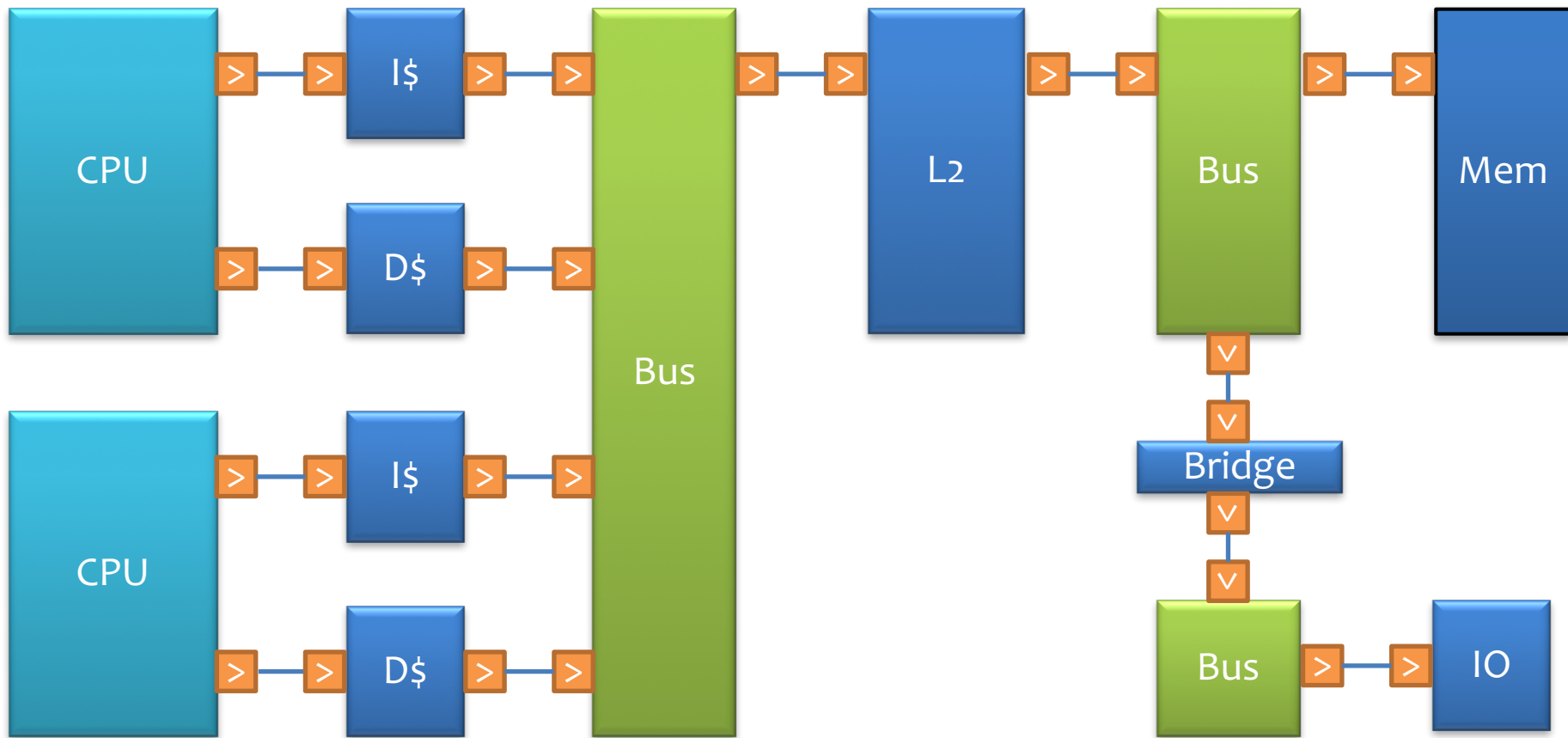
The Gem5 Simulator



Configurable CPU models	Functional, In-order, Out-of-order
Pluggable memory system	Flexible system design
Device models	Real full system modelling
Multiple ISAs	ARM, x86, Alpha, Sparc, PowerPC, Mips, RiscV
Boot real operating systems	Linux, Android, FreeBSD

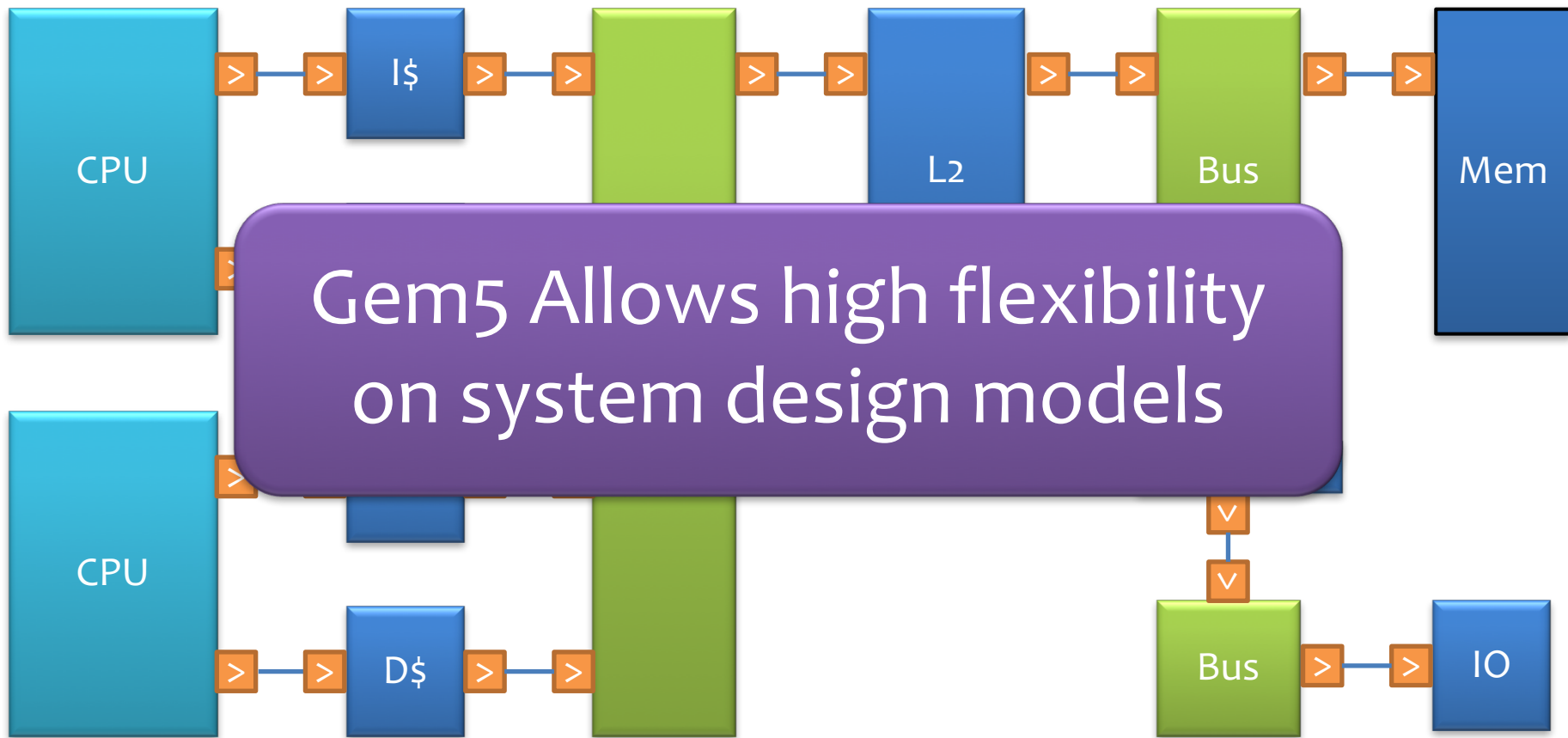
Gem5 modular scheme

Building the system



Gem5 modular scheme

Building the system



Full system microarchitectural simulation

A simulation consists of the following **layers**:


- A simulated full-system **hardware**
 - Detailed CPU microarchitecture
 - Peripheral devices
- An **operating system**
- An **application**




Towards reliability assessment

How to assess **reliability**?

Given an **functional full-system** modeling:

- **ACE** analysis
- Statistical **Fault Injection**
- **Analytical** methods



METHOD	ACE Analysis	Statistical Fault Injection	Analytical methods
Speed	Fast	Medium (<u>Adjustable</u>)	Fast
Accuracy	 Low	High (<u>Adjustable</u>)	 Very Low
Complexity	 High	Medium	Low

ACE analysis

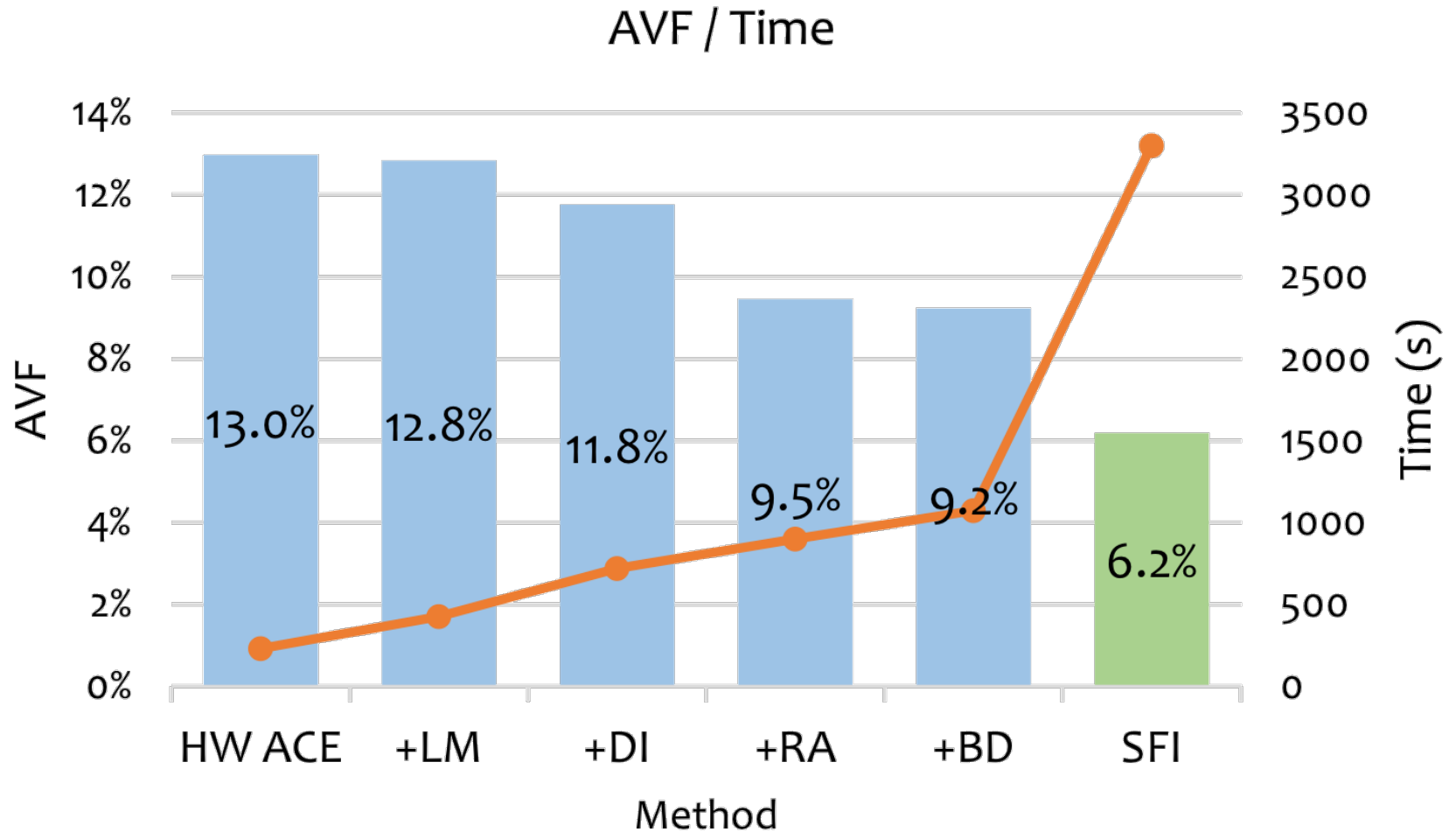
- **Lifetime** analysis
- **Logical masking**
- **Dead instruction** resolution
- **Output** dependencies
- **Complex modifications of simulator**
 - **Effort/Analysis time/Inaccuracy** tradeoff



MICRO-50, Boston



ACE analysis tradeoff



Fault modeling

- **Transient** faults (soft errors)
 - Single event bit-flips
- **Intermittent** faults
 - Stuck-at bits for a certain period of time
- **Permanent** faults
 - Permanently stuck-at bits
- **Multiple** faults – Combined faults
 - Cycles
 - Bits
 - Components
 - Models



MICRO-50, Boston



Fault modeling

Fault description

- Target **component**
- **Location** (bit granularity)
- **Model** (transient, intermittent, permanent)
- **Type** (bit flip, stuck-at 0, stuck-at 1)
- **Time** (cycles relative to simulation start)
- **Duration**



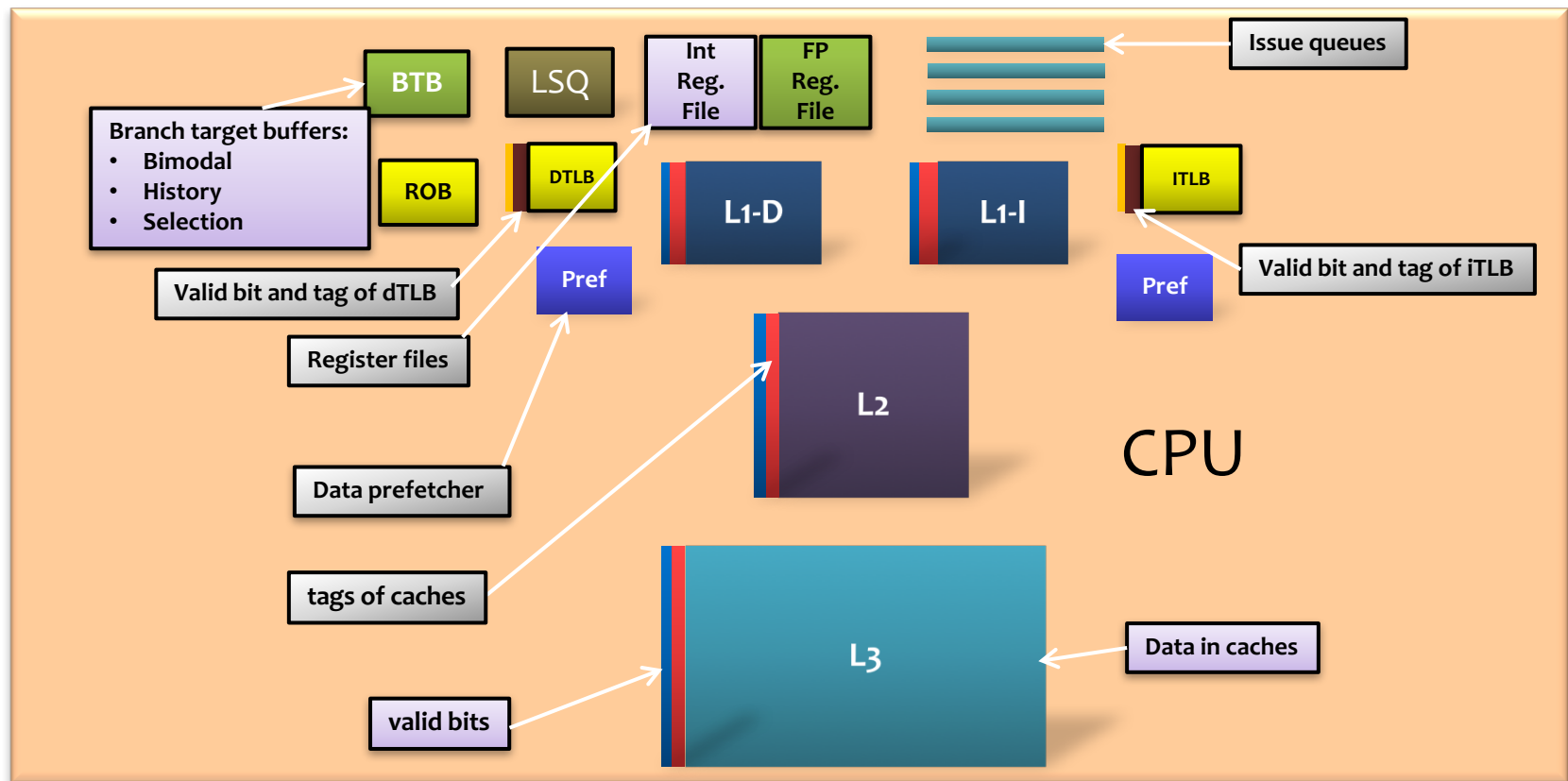
MICRO-50, Boston



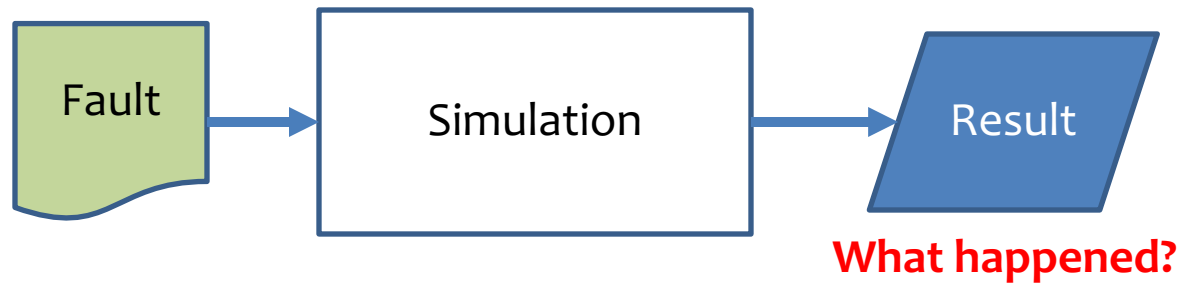
Target components

More than 50 hardware structures can be targeted:

Caches, RAM, Registers, Register Files, Buffers, Queues, etc. (arrays)



Fault injection simulation



Fault effect classification

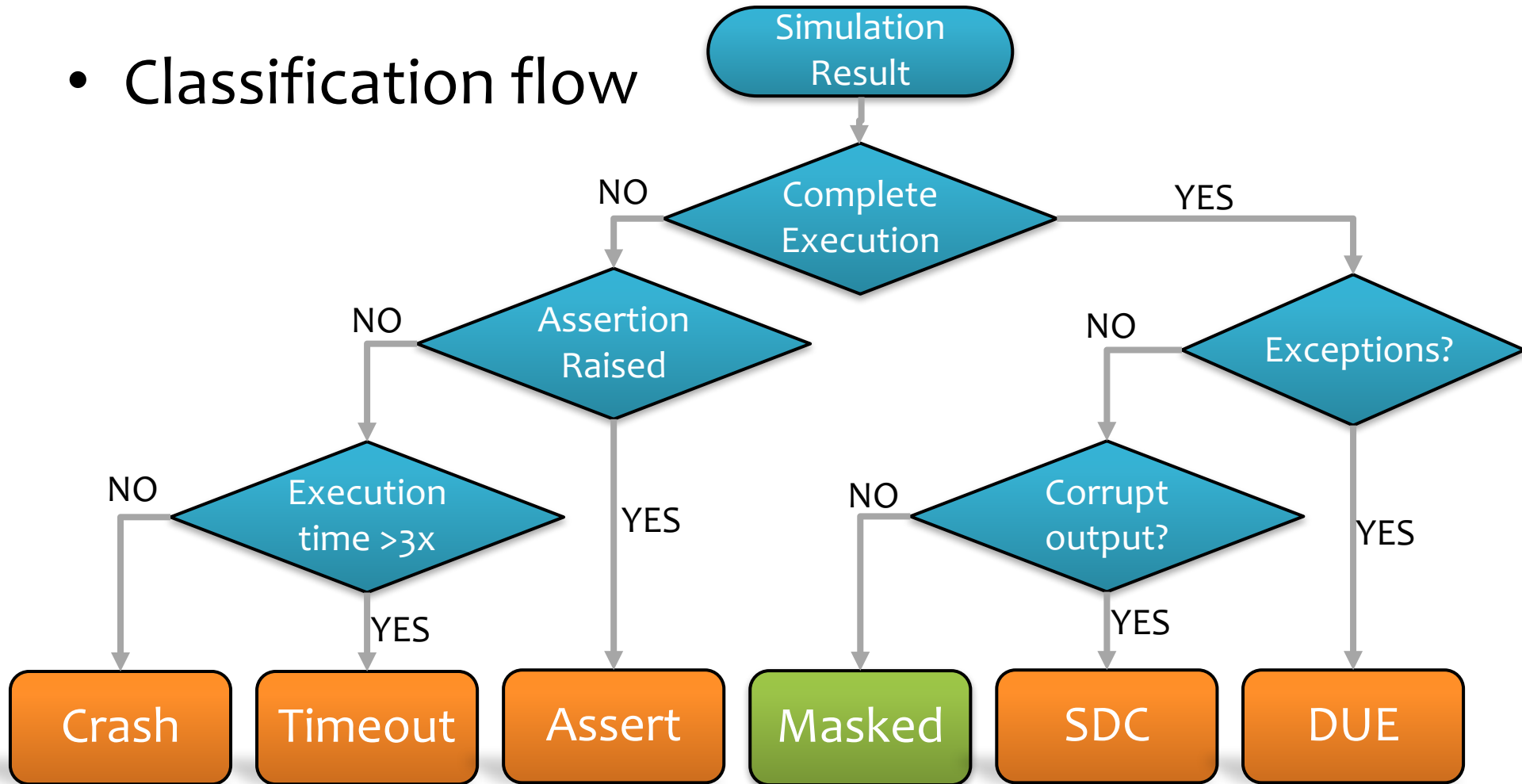
Fault classes, based on the simulation outcome

- **Masked**
 - Correct output without any abnormality
- **SDC** (silent data corruption)
 - Corrupted output without any abnormality
- **DUE** (detected unrecoverable error)
 - Exception abnormal behavior
- **Crash**
 - Application or System crash
- **Assertion**
 - Simulator failure
- **Timeout**
 - Simulation did not finish within the given time window (3x normal execution time)



Fault effect classification

- Classification flow



Metrics

- **Architectural Vulnerability Factor (AVF)**
 - Vulnerability metric
 - Probability of a fault to corrupt correct operation
- **Failures In Time (FIT)**
 - Reliability metric
 - Number of failures per 1 billion hours of operation
 - **Raw** FIT (per bit) vs **Actual** FIT
 - FIT rate of a **component**

$$FIT_{component} = FIT_{raw} \times \#bits \times AVF$$



MICRO-50, Boston



Vulnerability estimation

- **Statistical** sampling (by experiment)
 - Multiple **fault injection simulations**
 - **Classification** of each simulation, based on the outcome
 - Quantify **vulnerability** by measuring non-masked cases

$$AVF = \frac{\text{Vulnerable cases}}{\text{Total simulations}}$$

Statistical Fault Injection (SFI)



Statistical Fault Injection

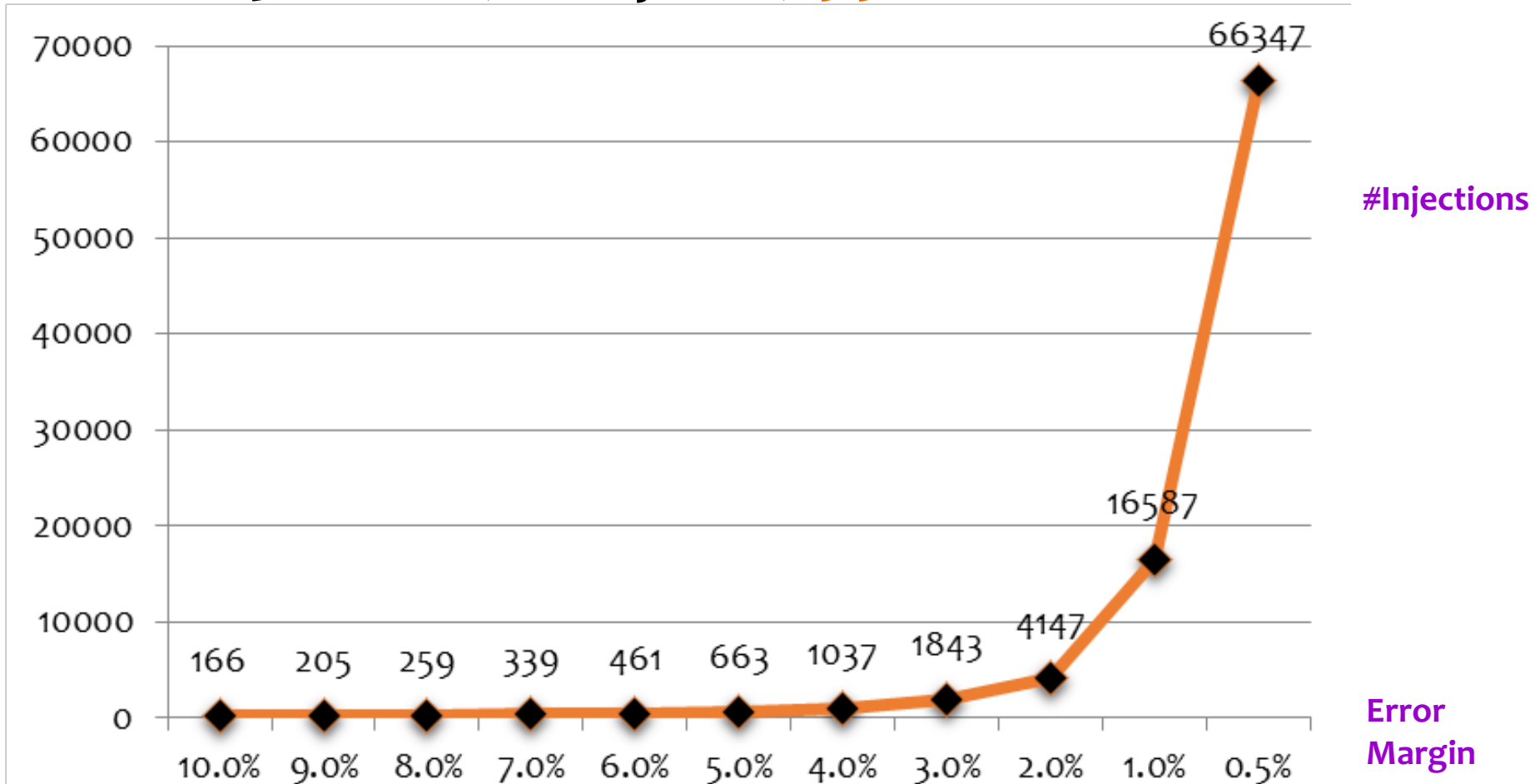
- How many simulations?
 - Quantifying statistical sample **significance** *
 - **Number** of faults (n) is a result of :
 - Workload **duration** (d)
 - Hardware **size** in bits (s)
 - **Error margin** (e)
 - **Confidence** (t)
- Initial **population** (N) = $d \times s$
- Exp. proportion (p) = 0.5 *

$$n = \frac{N}{1 + e^2 \times \frac{N - 1}{t^2 \times p \times (1 - p)}}$$

* Leveugle, et. al., "Statistical Fault Injection: Quantified Error and Confidence", DATE, 2009

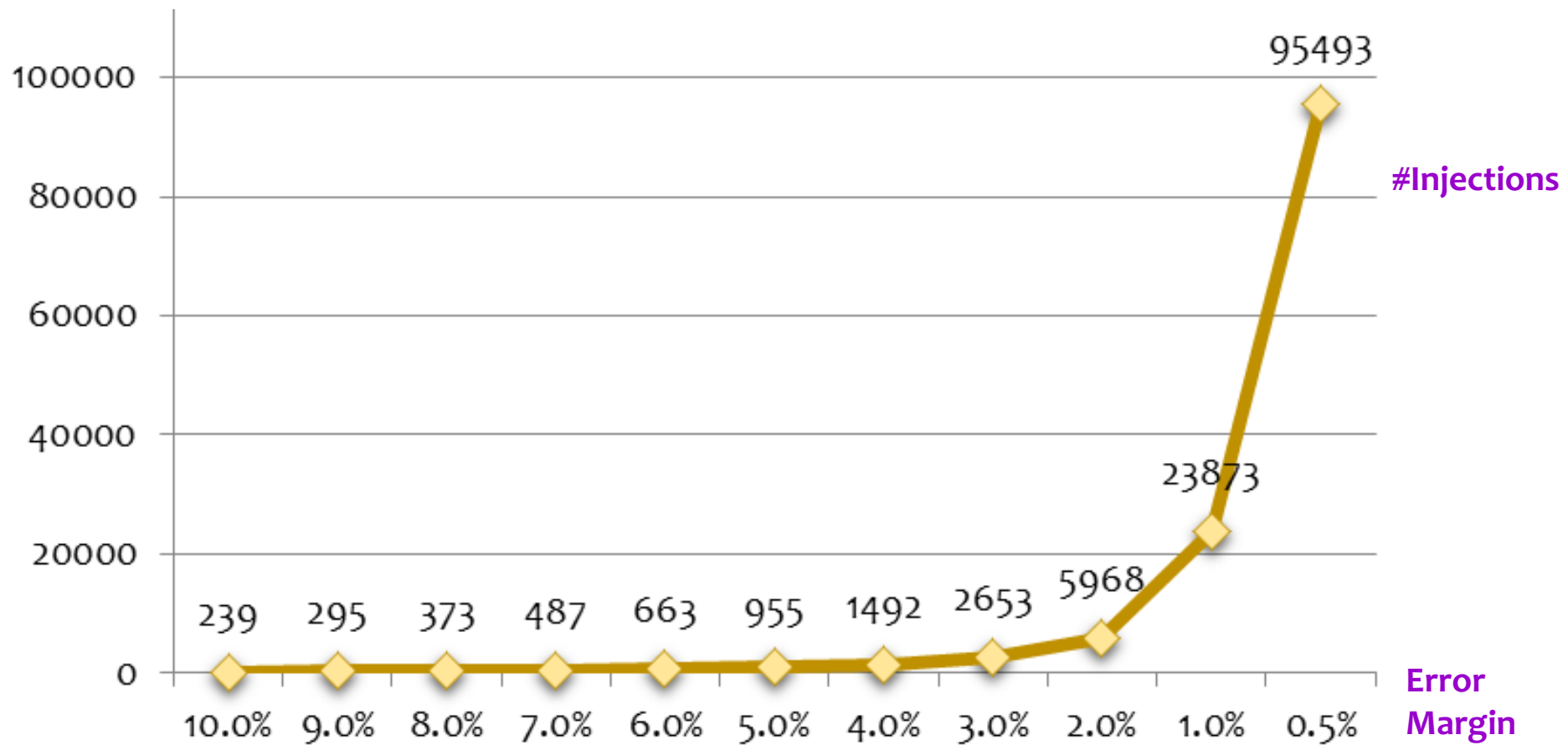
#Fault Injections vs. Error Margin

- For 32K bits, 1B cycles, **99%** confidence



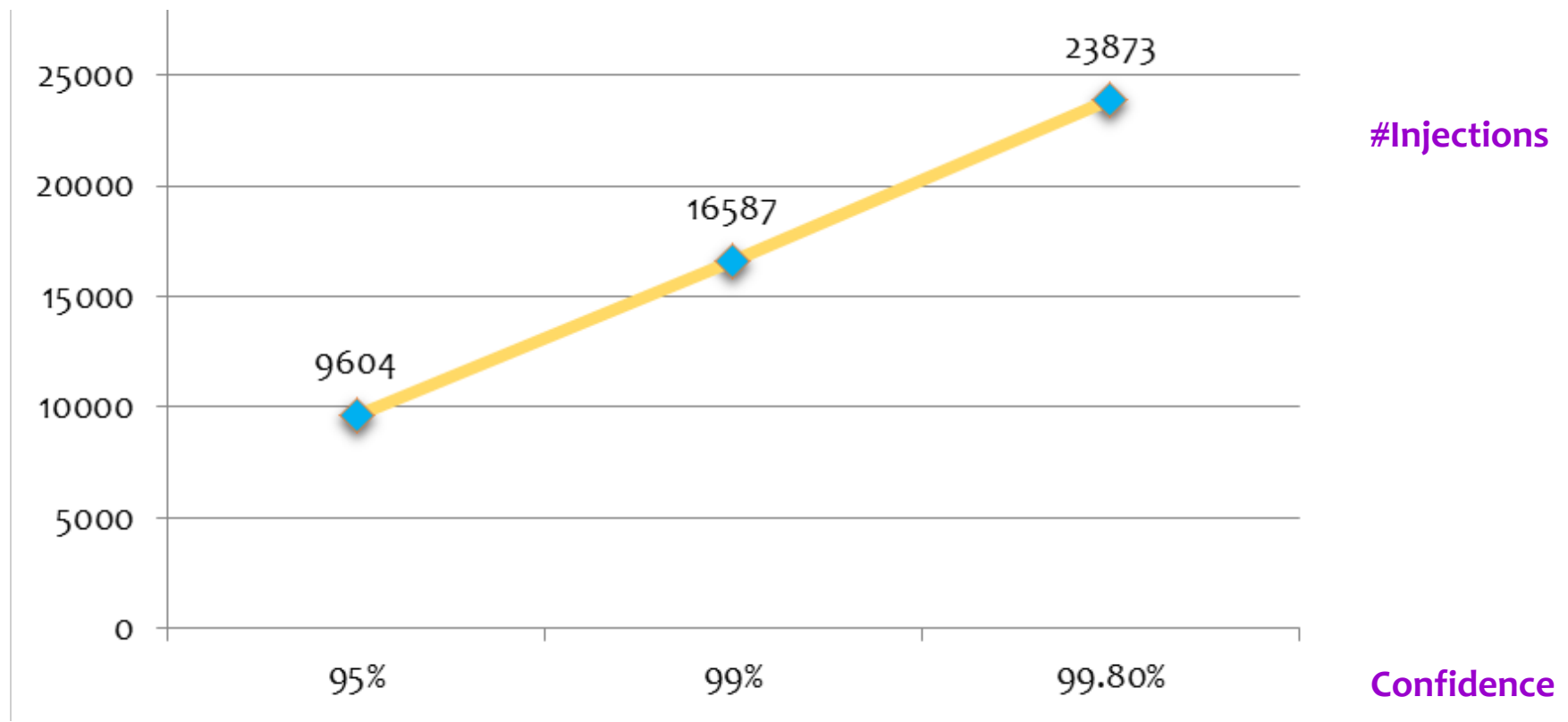
#Fault Injections vs. Error Margin

- For 32K bits, 1B cycles, **99.8%** confidence

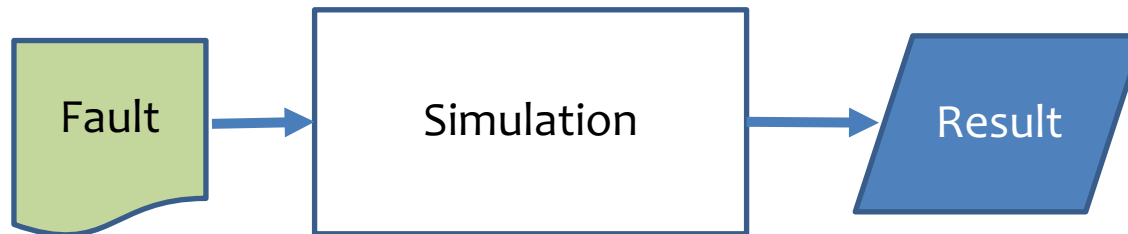


#Fault Injections vs. Confidence

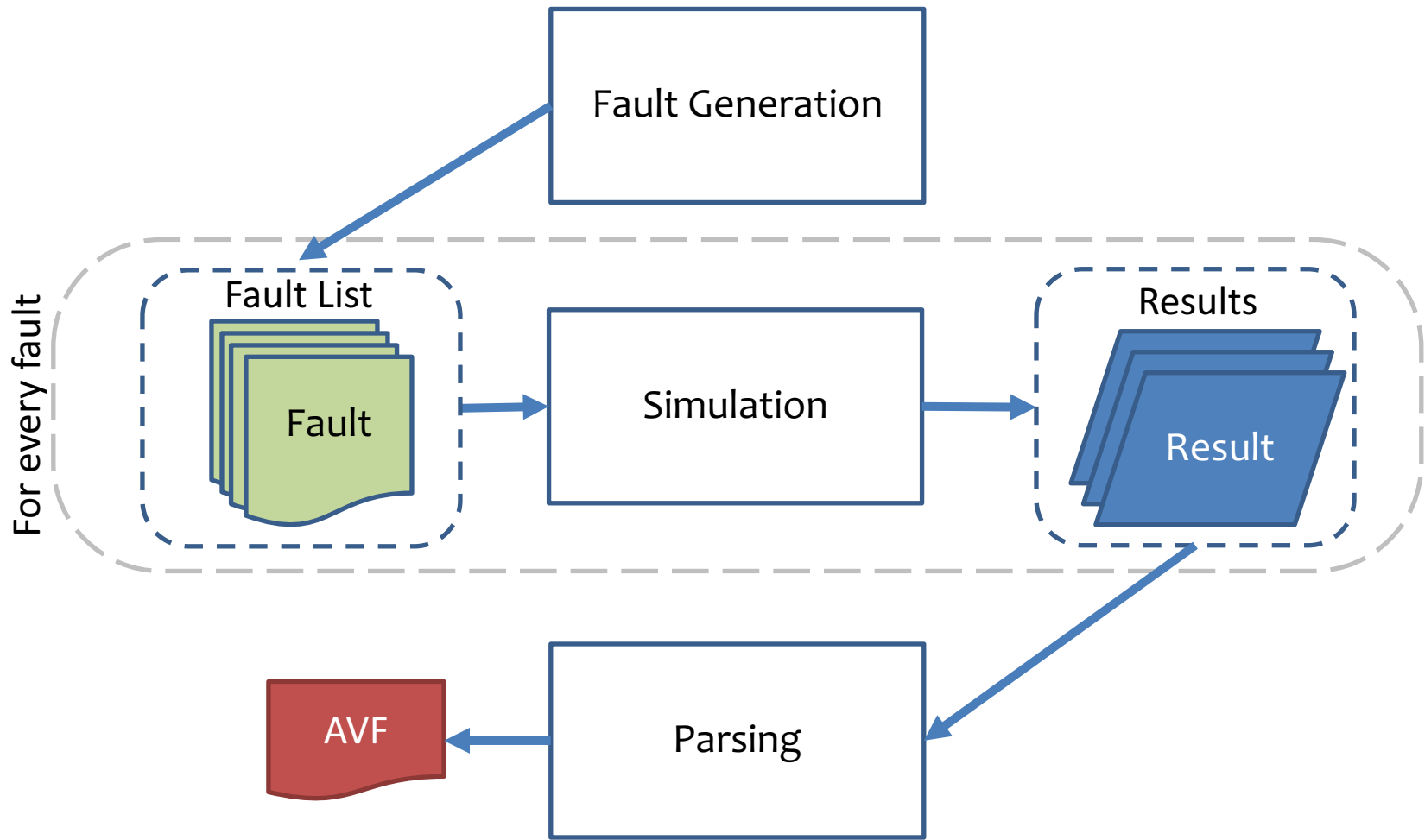
- For 32K bits, 1B cycles, 1% error margin



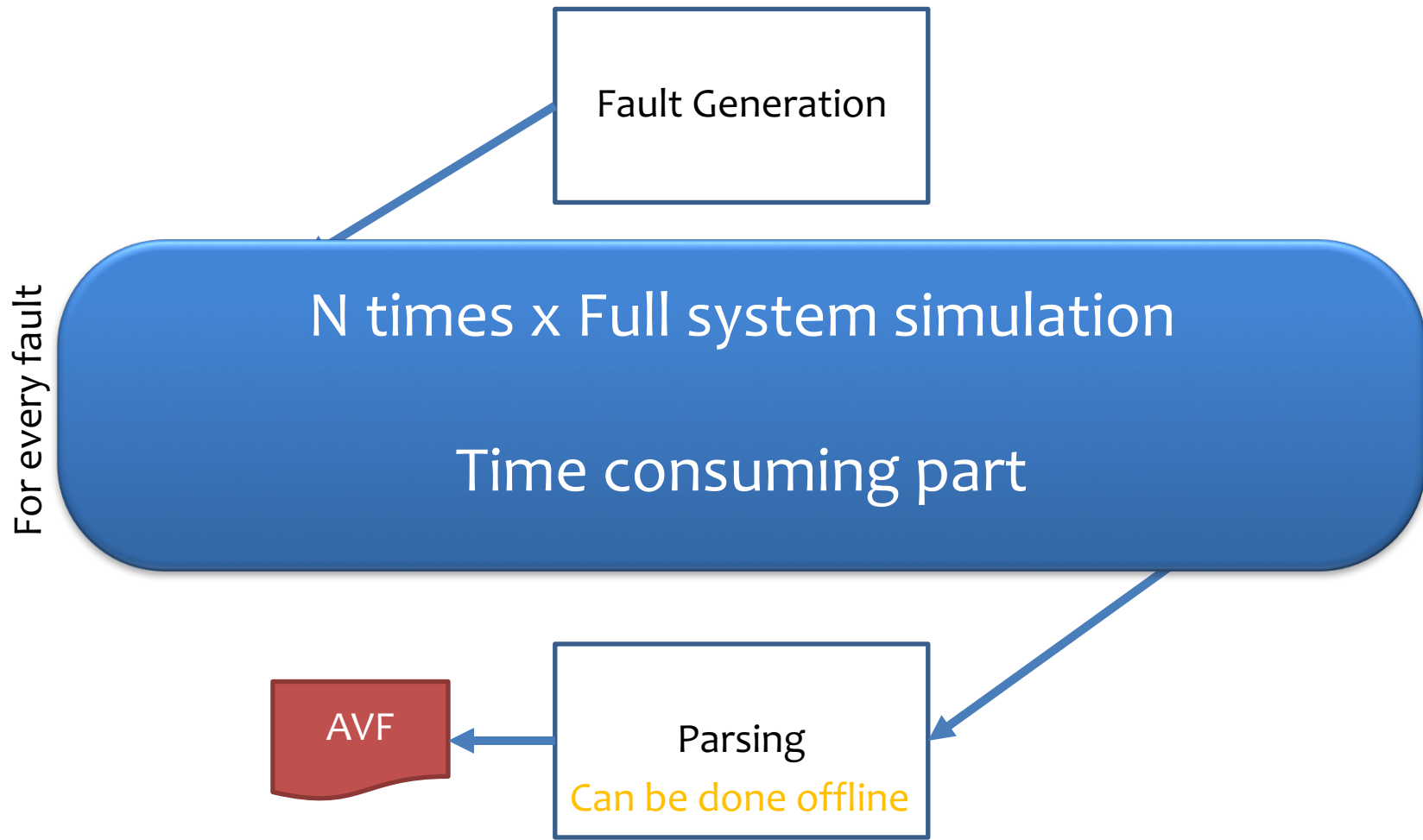
Fault injection simulation



Fault Injection Campaign



Fault Injection Campaign



Injection Throughput Calculation

- Hardware + Software

parameter	value
program execution time	1B (10^9) dynamic instructions
hardware structure	10K bits (reg.file)
simulation throughput	300K instructions/sec
equipment	10 servers

- Injection Statistics

confidence, error margin		#Injections	Fault injection campaign time
95%	5%	384	1.5 day
99%	3%	1843	1 week
99.8%	1%	23,873	3 months
99.8%	0.5%	95,493	1 year

Workloads

- **MiBench** benchmark suite
 - Automotive
 - Consumer
 - Network
 - Office
 - Security
 - Telecommunication
- **Short** execution
- Widely used in **reliability studies**




MICRO-50, Boston



Hardware configuration

Examples of commercial **microarchitectural** configurations on which GeFIN was applied

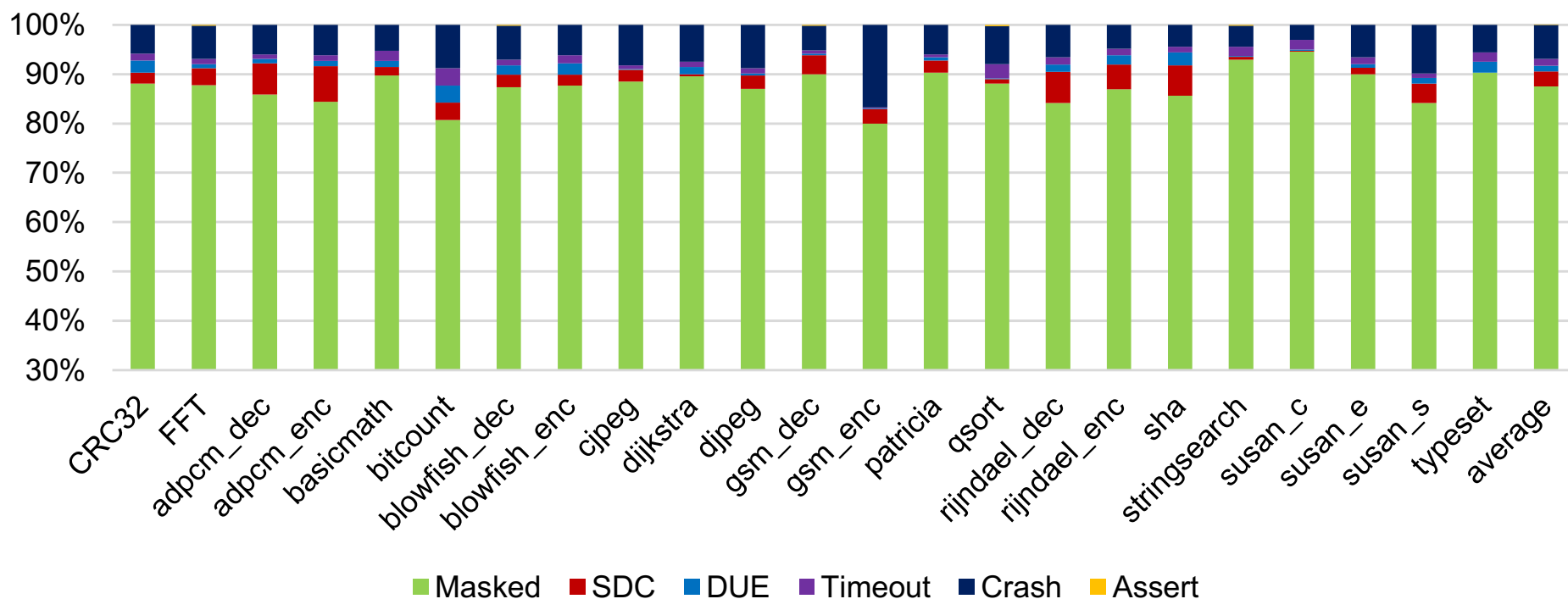
Configuration preset	Vendor
Cortex A9	
Cortex A15	

Chip failure rates (FIT) derived from
26 hardware structures

AVF/Fault Effects – A9

ARM®
Cortex-A9

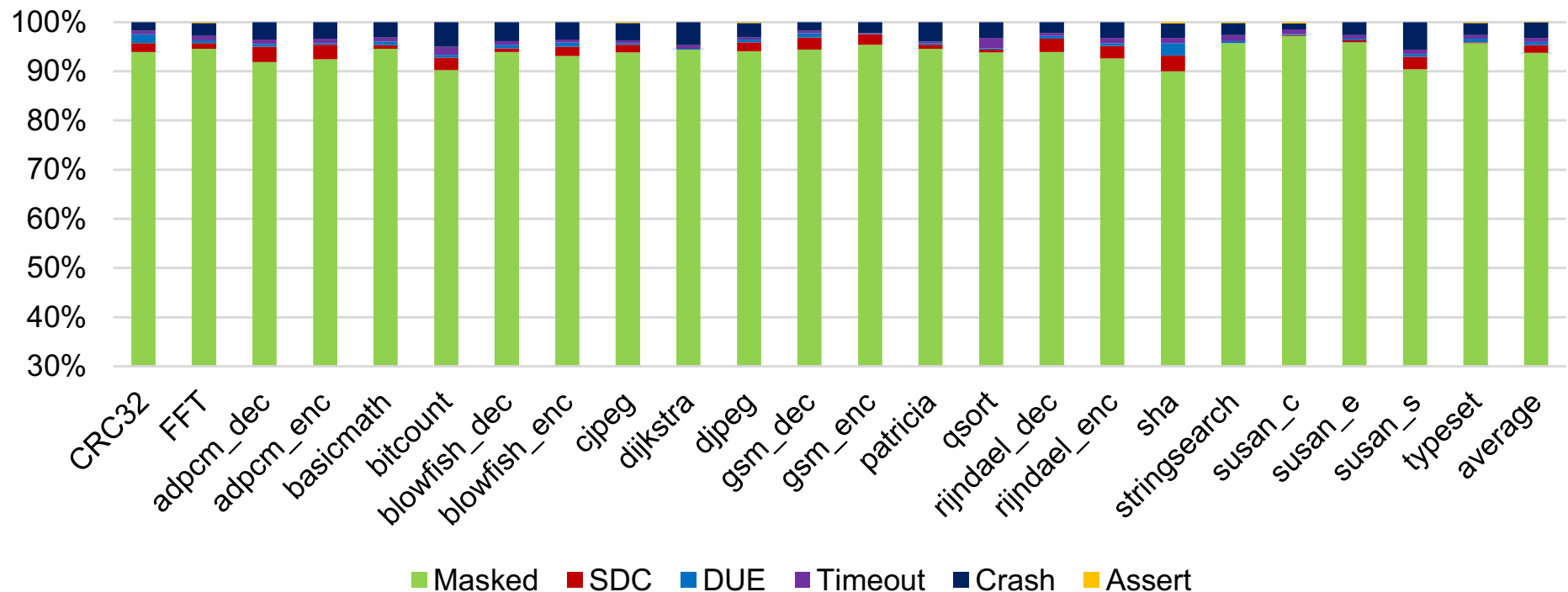
Integer Register file



AVF/Fault Effects – A15



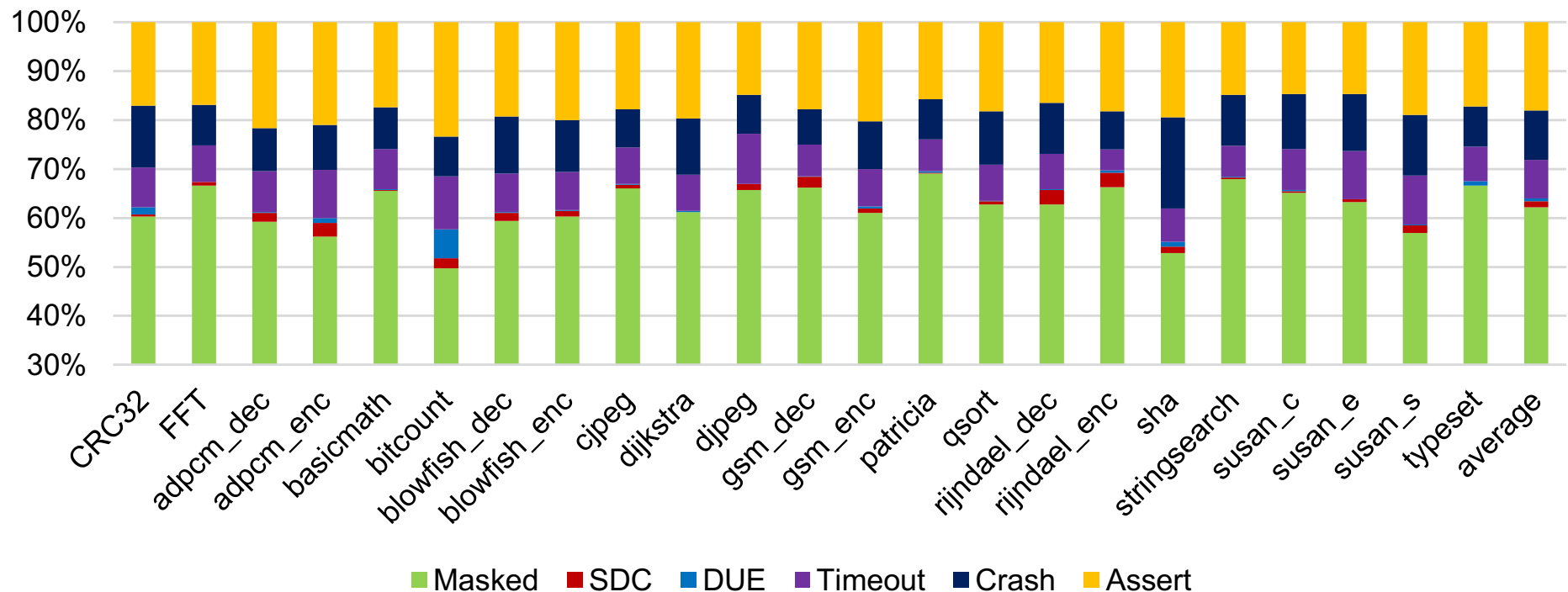
Integer Register file



AVF/Fault Effects – A9

ARM®
Cortex-A9

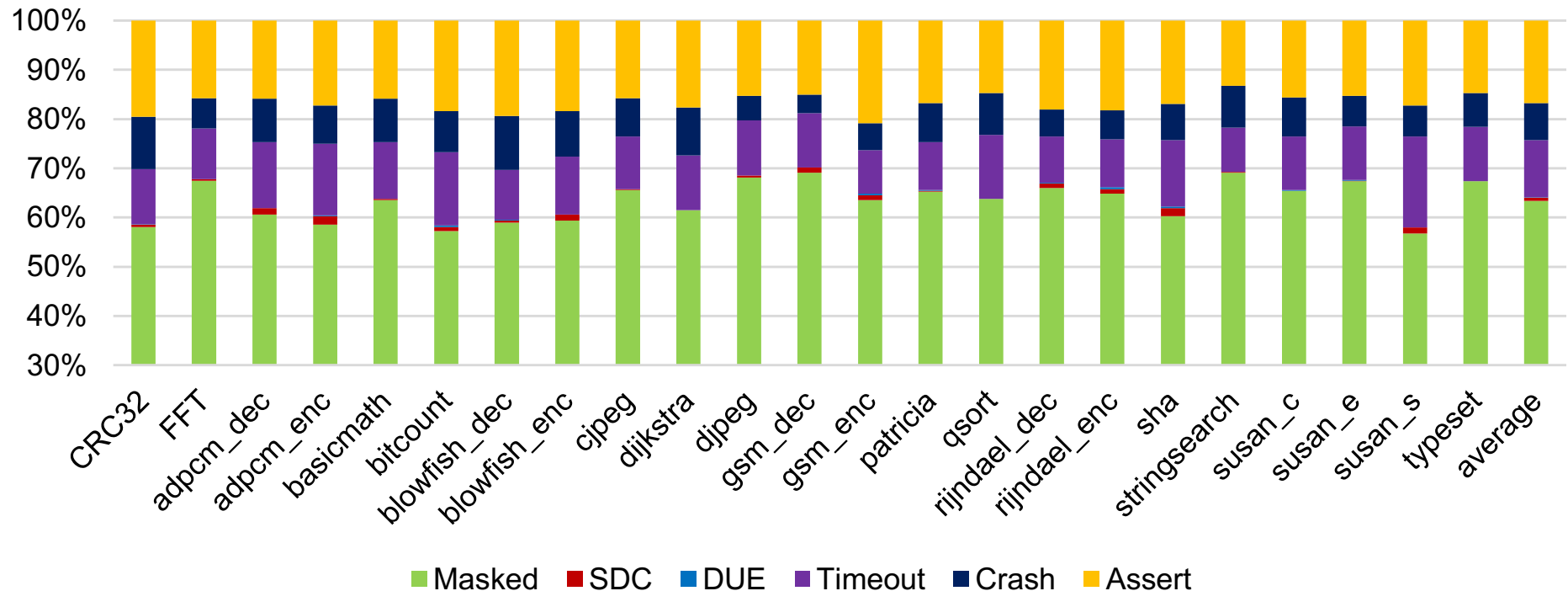
Integer Rename map



AVF/Fault Effects – A15



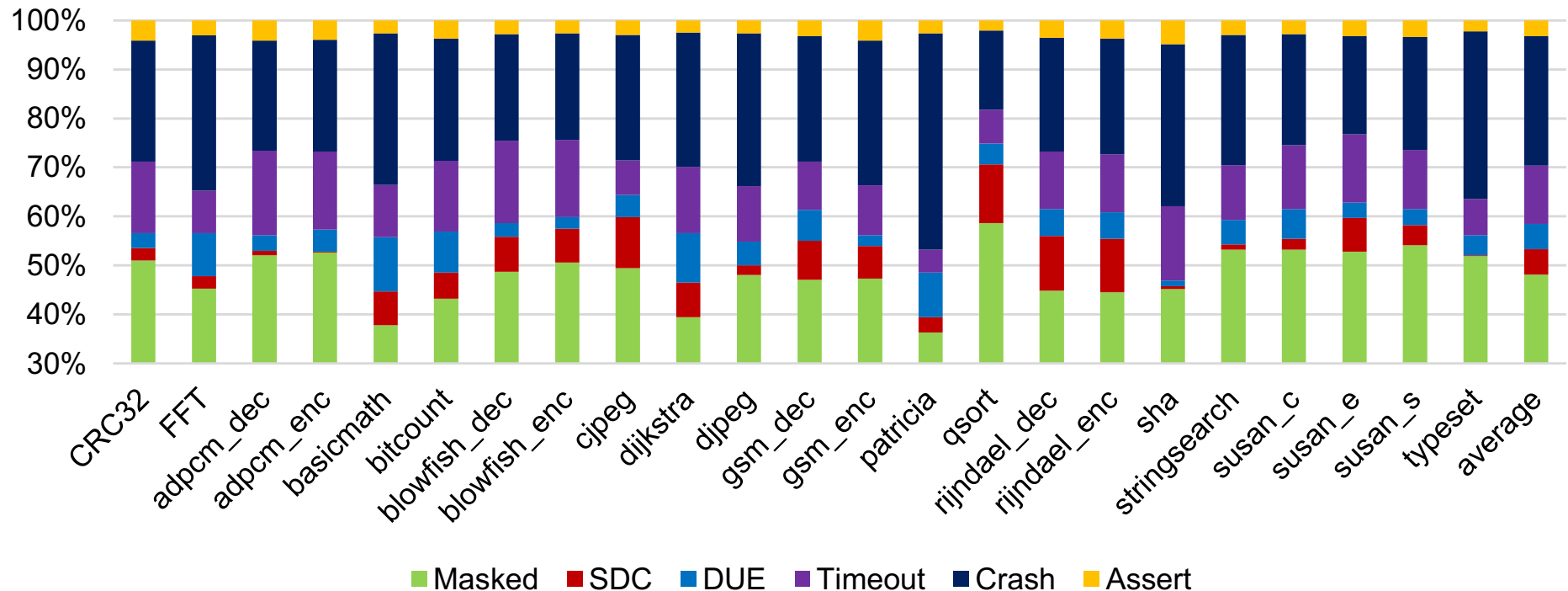
Integer Rename map



AVF/Fault Effects – A9

ARM®
Cortex-A9

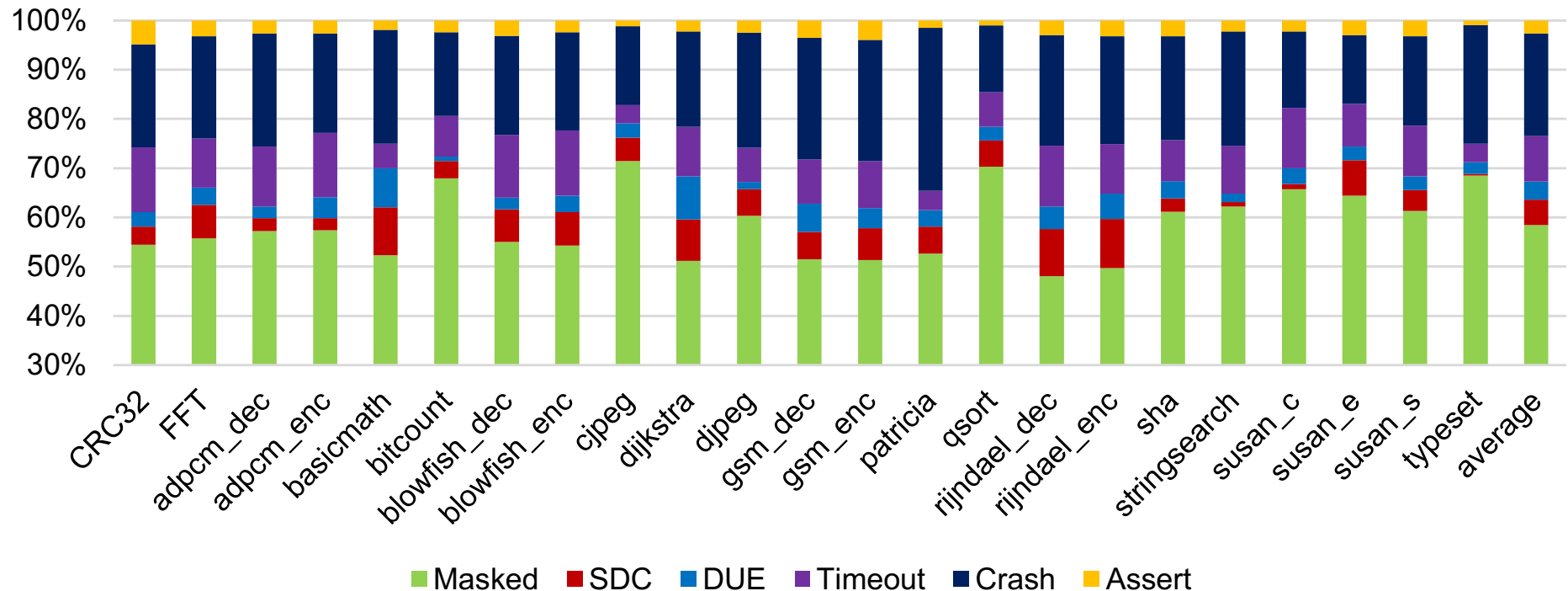
Data translation buffer physical



AVF/Fault Effects – A15



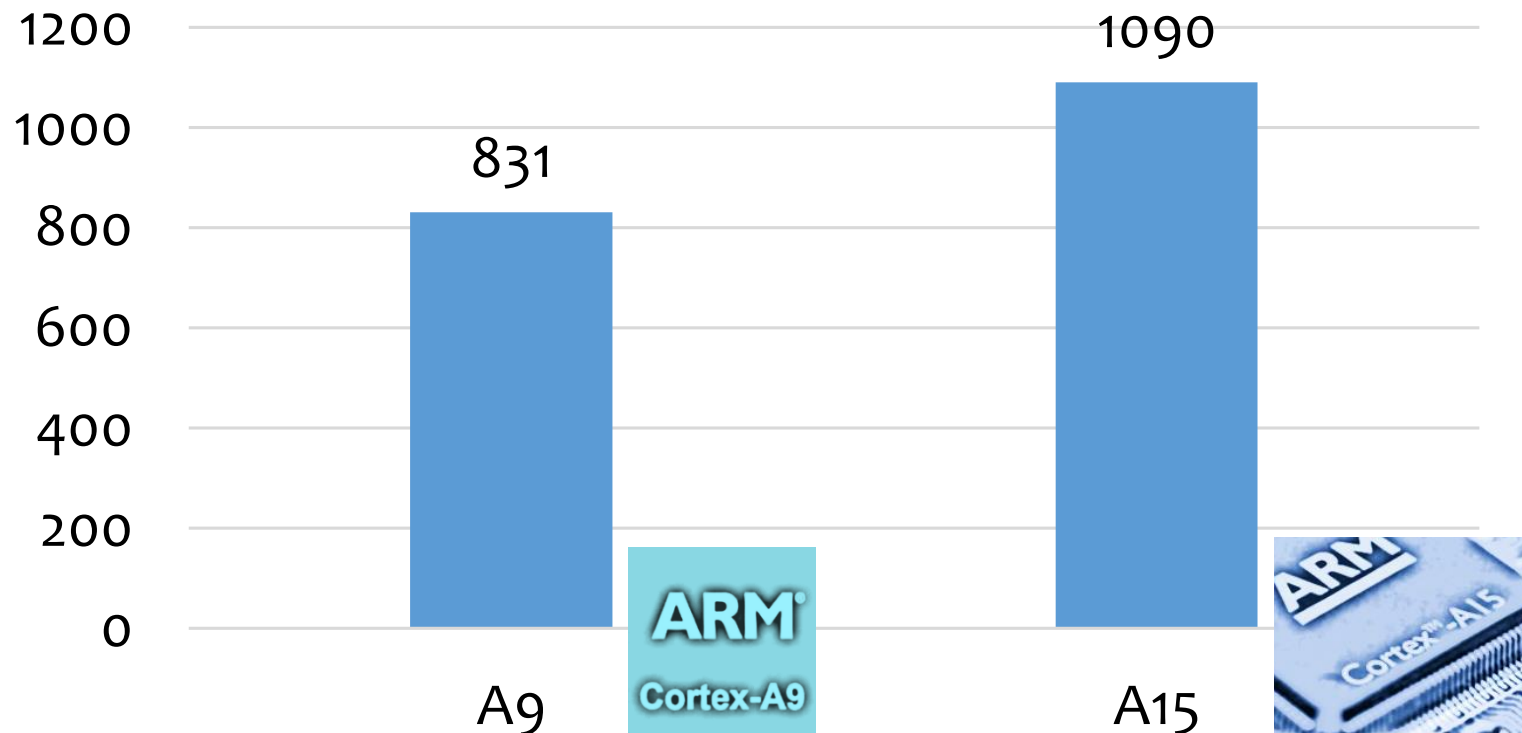
Data translation buffer physical



Reliability assessment

- Technology **Raw FIT = 0.001**
- **26** hardware components

FIT



Uses of Fault Injection – Case Studies

- **Design exploration**
 - Microarchitectural attributes / vulnerability impact
- **Performance** studies
 - Performance component tolerance
- **Differential** studies
 - Different architectures
- **Error Protection** evaluation

Design exploration for Reliability

- Microarchitectural attributes
 - Impact on **vulnerability**
 - **Design exploration**
- Different **size**
- Different **associativity**
- Write back/write through **policy**

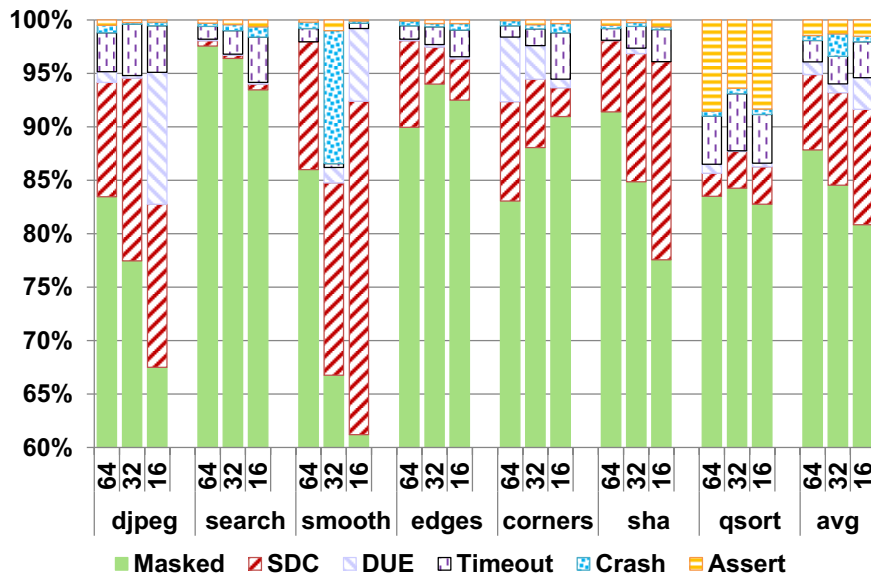


MICRO-50, Boston

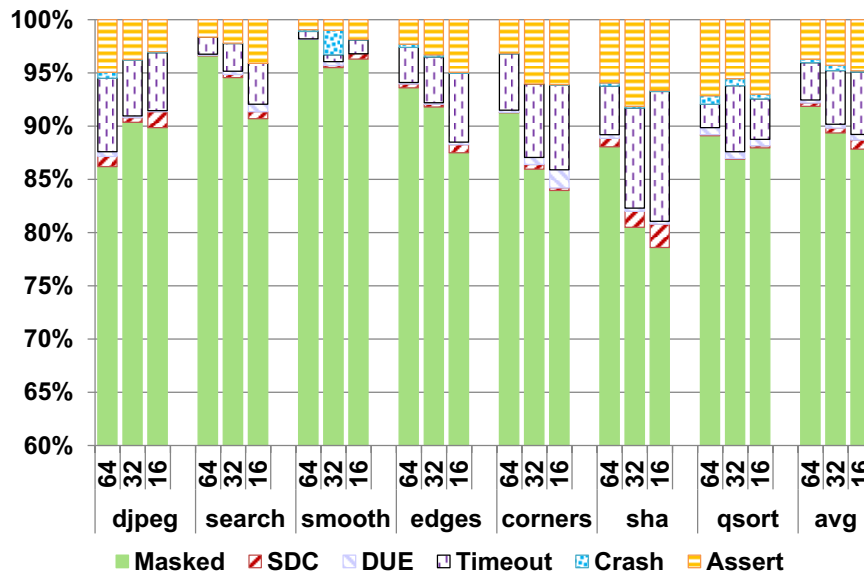


Cache size

L1 Data Cache across different size (KB)



L1 Instruction Cache across different size (KB)

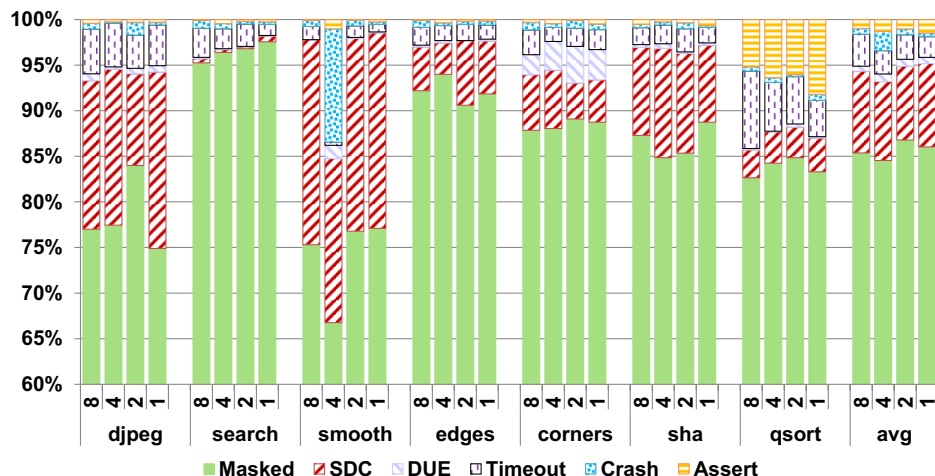


- Increase in masked class from 16KB to 64KB
 - ~7 percentile points in L1 Data cache
 - ~5 percentile points in L1 Instruction cache
- More reliable for smaller sizes:
 - 251.1 FIT for 16KB L1 Data cache
 - 159.4 FIT for 16KB L1 Instruction cache

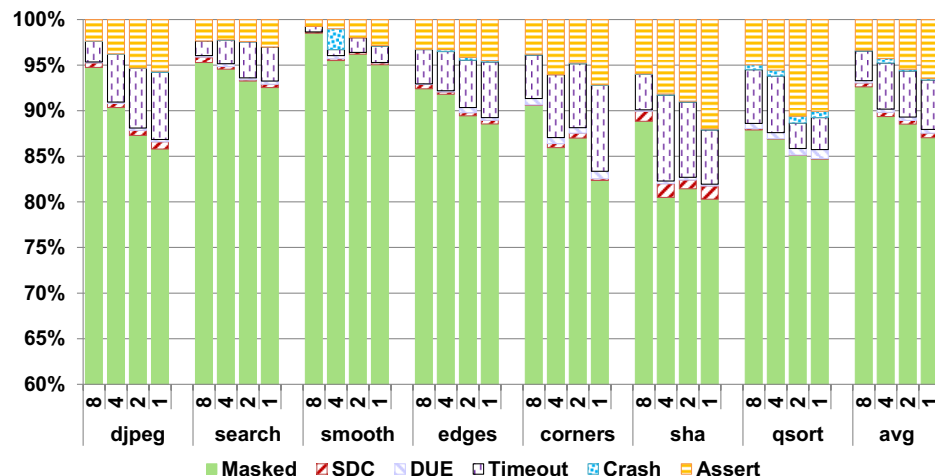
* S.Tselonis, M.Kaliorakis, N.Fourtris, G.Papadimitriou, D.Gizopoulos, "Microprocessor Reliability-Performance Tradeoffs Assessment at the Microarchitecture Level", VTS, 2016

Associativity

L1 Data Cache across different associativity (number of ways)



L1 Instruction Cache across different associativity (number of ways)



- Average trend of Masked category:
 - almost insensitive in L1 Data cache (apart from djpeg and smooth)
 - ~6 percentile points increase from a direct-mapped to 8-way set associative L1 Instruction cache
- Most reliable cases observed:
 - 346.4 FIT for 2-way set associative L1 Data cache
 - 193.6 FIT for 8-way set associative L1 Instruction cache

* S.Tselonis, M.Kaliorakis, N.Foutris, G.Papadimitriou, D.Gizopoulos, "Microprocessor Reliability-Performance Tradeoffs Assessment at the Microarchitecture Level", VTS, 2016

Performance Faults Studies

- Capturing performance impact of speculative **performance components**
 - Branch predictors
 - Prefetchers
 - Branch target buffers
 - Return address stack
- Use case:
Emulating **under-volting** on BPU SRAM arrays



MICRO-50, Boston



Under-voltage on BPU

BPU

- Tournament BP
- BTB

Using **14nm FinFET** near-threshold SRAM failure distributions *

Array	Size (bits)
Global BP	65,536 bits
Local BP	32,768
Local BPHT	22,528
Select Comp	16,384
BTB	307,200
Total	421,888

BPU

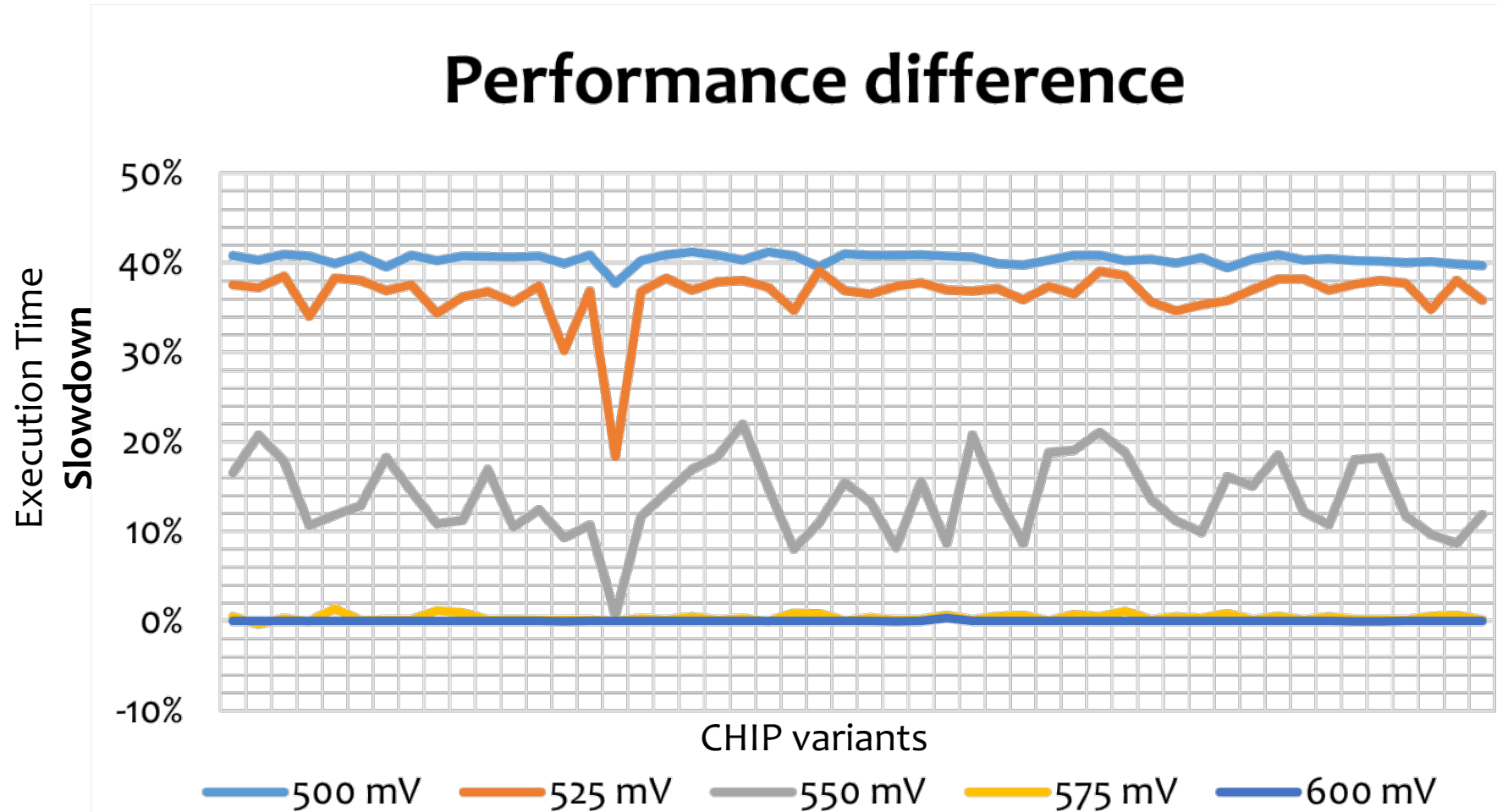
- Generation of **50 “different” BPUs**
- Average number of permanent **faults per voltage**

Voltage	Avg. #faults
500 mV	369 K
525 mV	136 K
550 mV	10 K
575 mV	198
600 mV	2

* S. Ganapathy, J. Kalamatianos, K. Kasprak, S. Raasch, "On Characterizing Near-Threshold SRAM Failures in FinFET Technology", DAC, 2017

Permanent Faults Performance Impact

JPEG decode benchmark **slowdown**



Differential studies

- Assess reliability of
 - Different **microarchitectures**
 - Different **architectures**
 - Different **systems**
 - Different **simulators**
- **Same programs**
- Different probabilities of **correct execution**

[1] A.Chatzidimitriou, M.Kaliorakis, S.Tselonis, D.Gizopoulos, "Performance-aware reliability assessment of heterogeneous chips", VTS, 2017

[2] M. Kaliorakis, S. Tselonis, A. Chatzidimitriou, N. Foutris, D. Gizopoulos,, "Differential Fault Injection on Microarchitectural Simulators", IISWC, 2015

Differential study – CPUs vs. GPUs

- **Case study:**
 - **Register file** of:
 - x86-64 CPU
 - Cortex A9 CPU
 - Cortex A15 CPU
 - Nvidia GT200 GPU

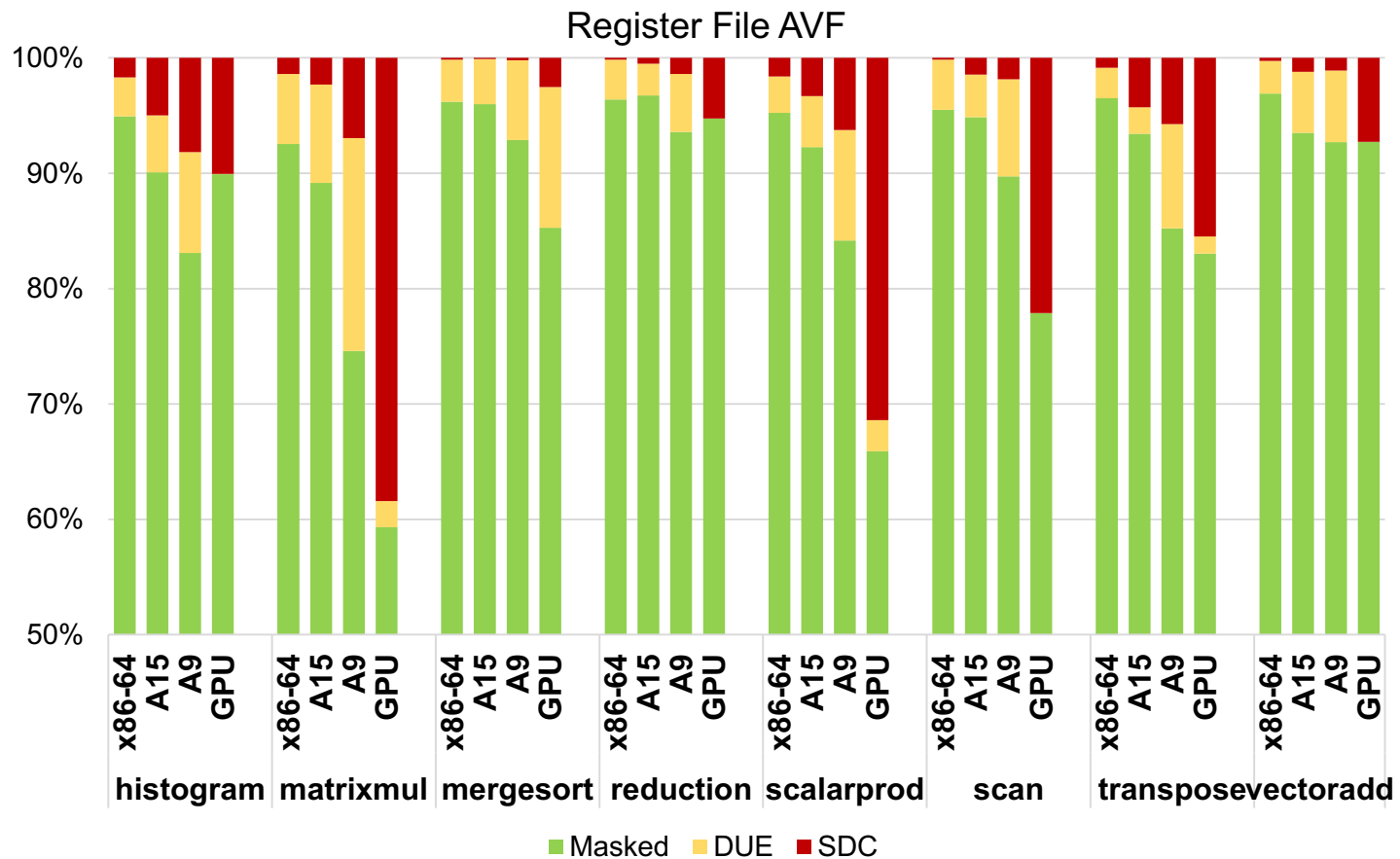


MICRO-50, Boston



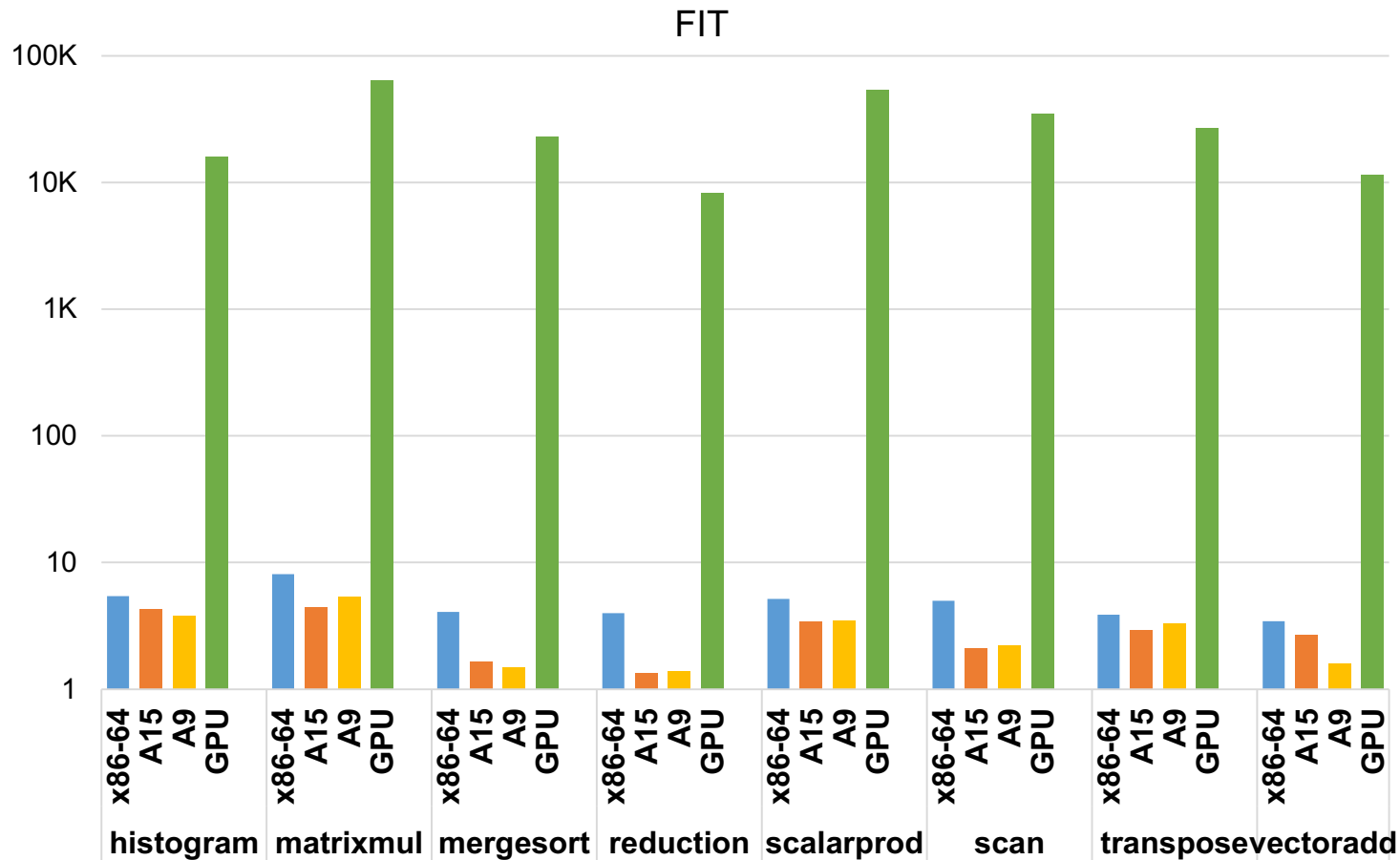
CPU vs. GPU – AVF

- Fault effects classes



CPU vs. GPU – FIT

- FIT rates



MICRO-50, Boston



Error Protection Evaluation

- Assessment of protection mechanisms
 - Protection **scheme**
 - **Performance** impact
 - **Scrubbing** techniques
 - **Granularity**
- Exploration of **different** mechanisms



MICRO-50, Boston

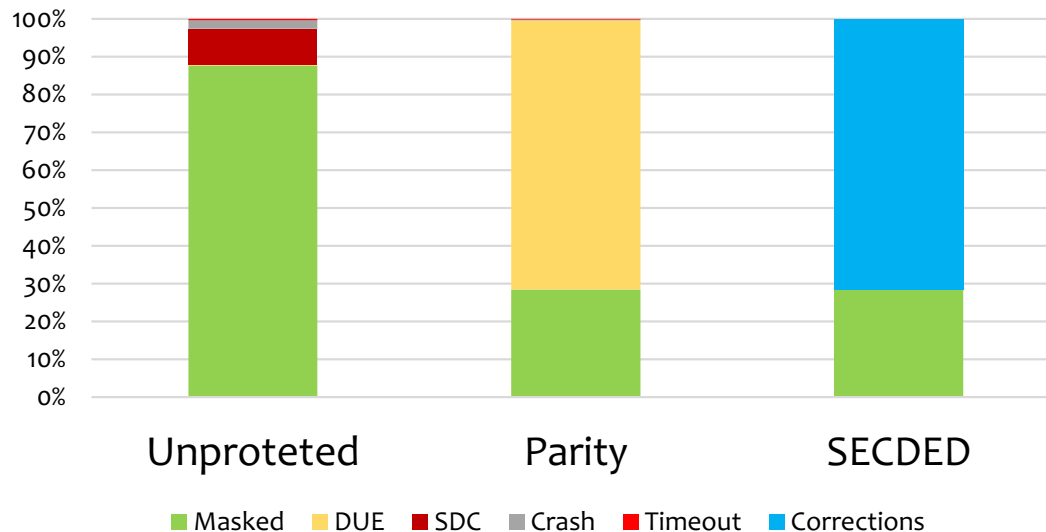


Error Protection Evaluation

Use case:

- L1 **Data Cache**
- **Parity** EDC / **SECDED** ECC
- **Transient** faults (single/double)

Single transient faults



Double transient faults





MICRO-50, Boston, October 2017

Next ...
Part 3

Tutorial:

Microarchitecture Level Reliability Assessment

Throughput and Accuracy
<http://micro50-tutorial.di.uoa.gr/>

Organizers/Presenters:

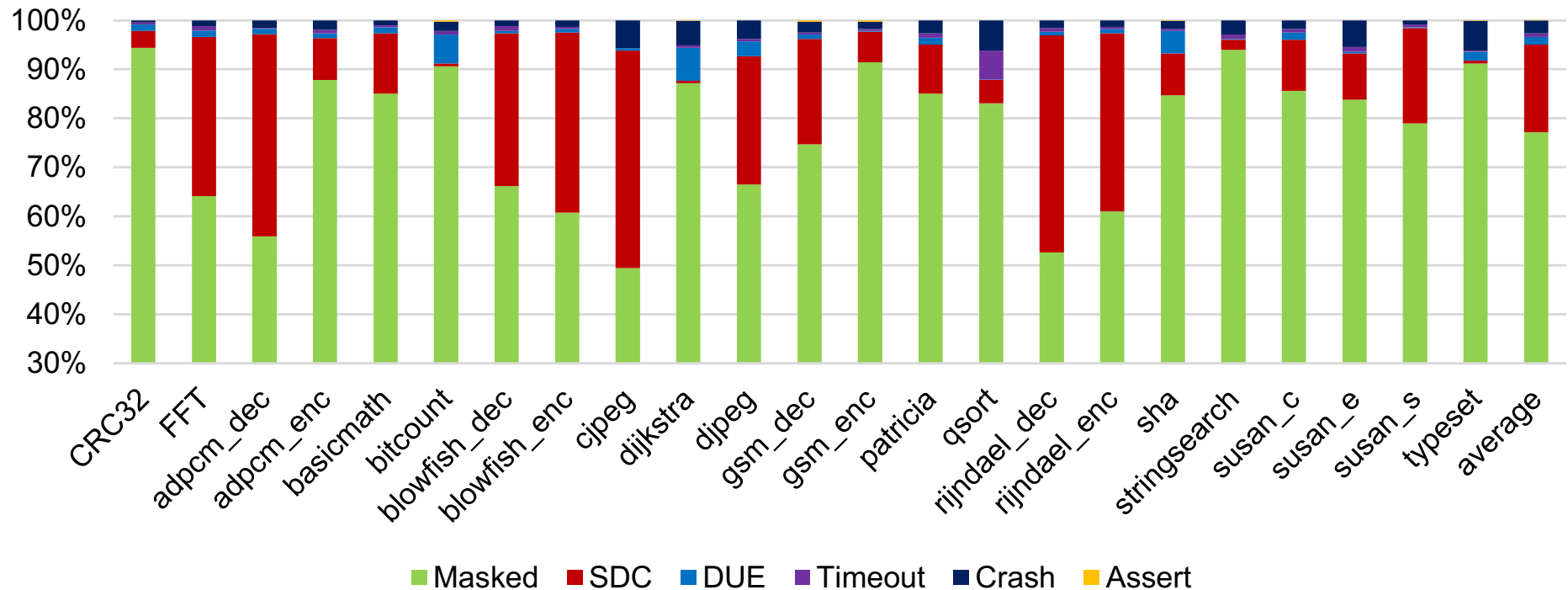
Athanasios Chatzidimitriou, Manolis Kaliorakis, Dimitris Gizopoulos

More A9/A15 Components Fault Effects

AVF/Fault Effects – A9

ARM®
Cortex-A9

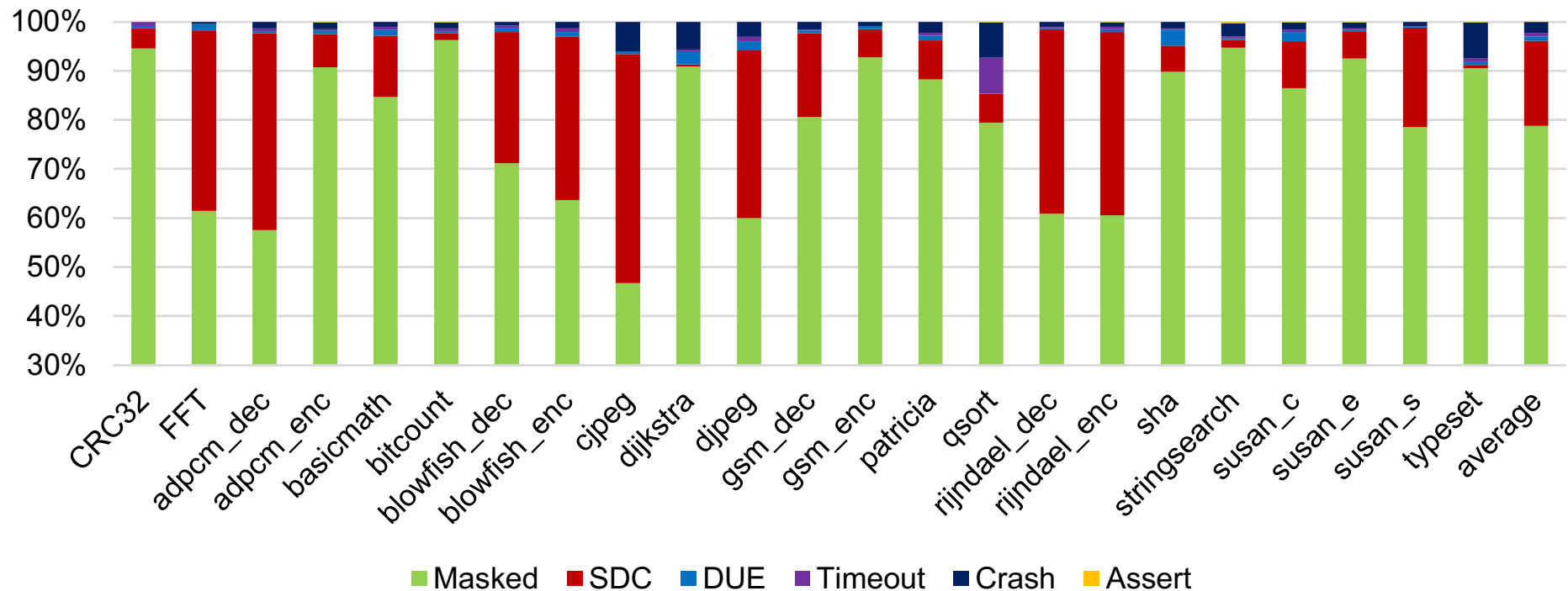
L1 Data cache - data



AVF/Fault Effects – A15



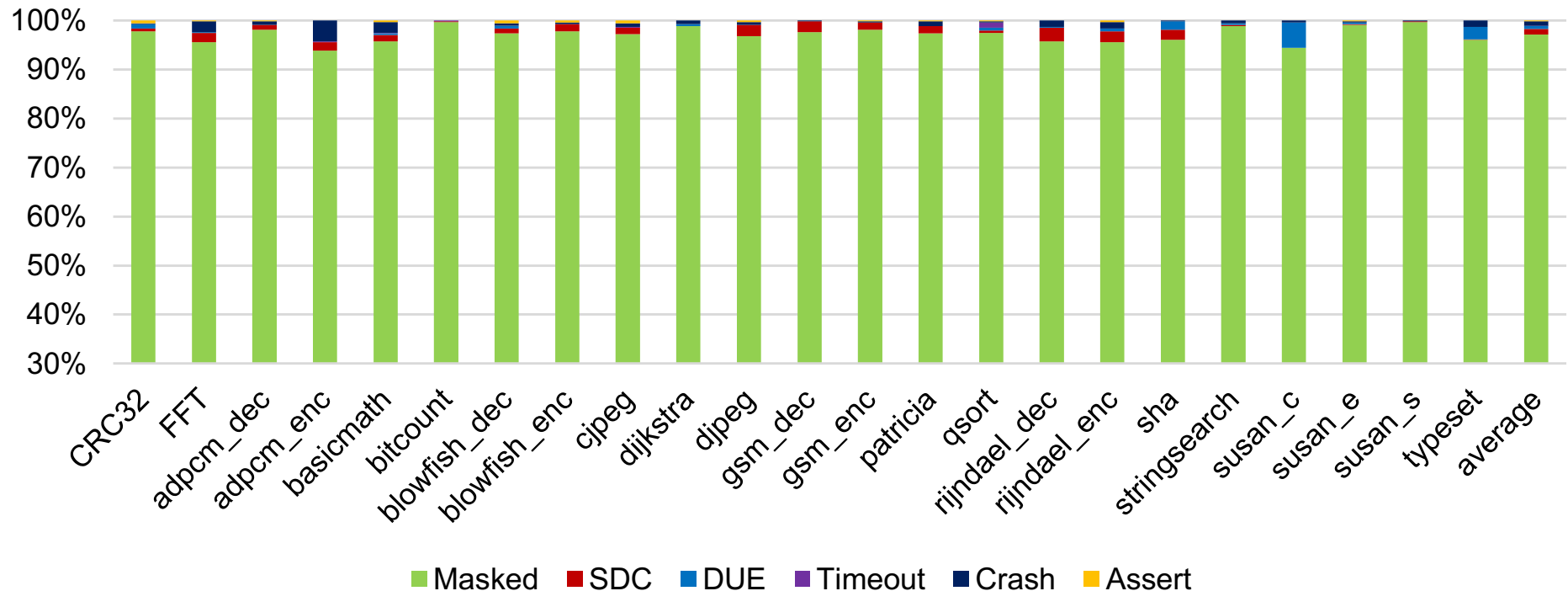
L1 Data cache - data



AVF/Fault Effects – A9

ARM®
Cortex-A9

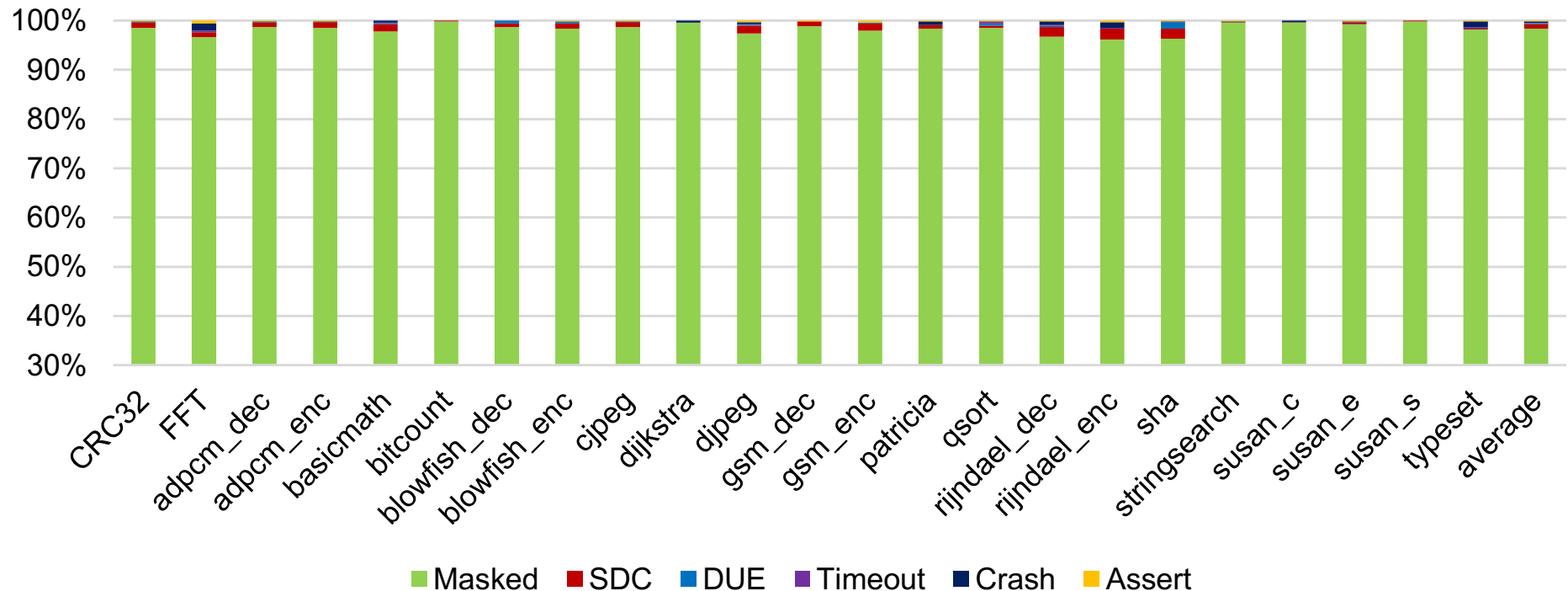
SQ Physical address



AVF/Fault Effects – A15



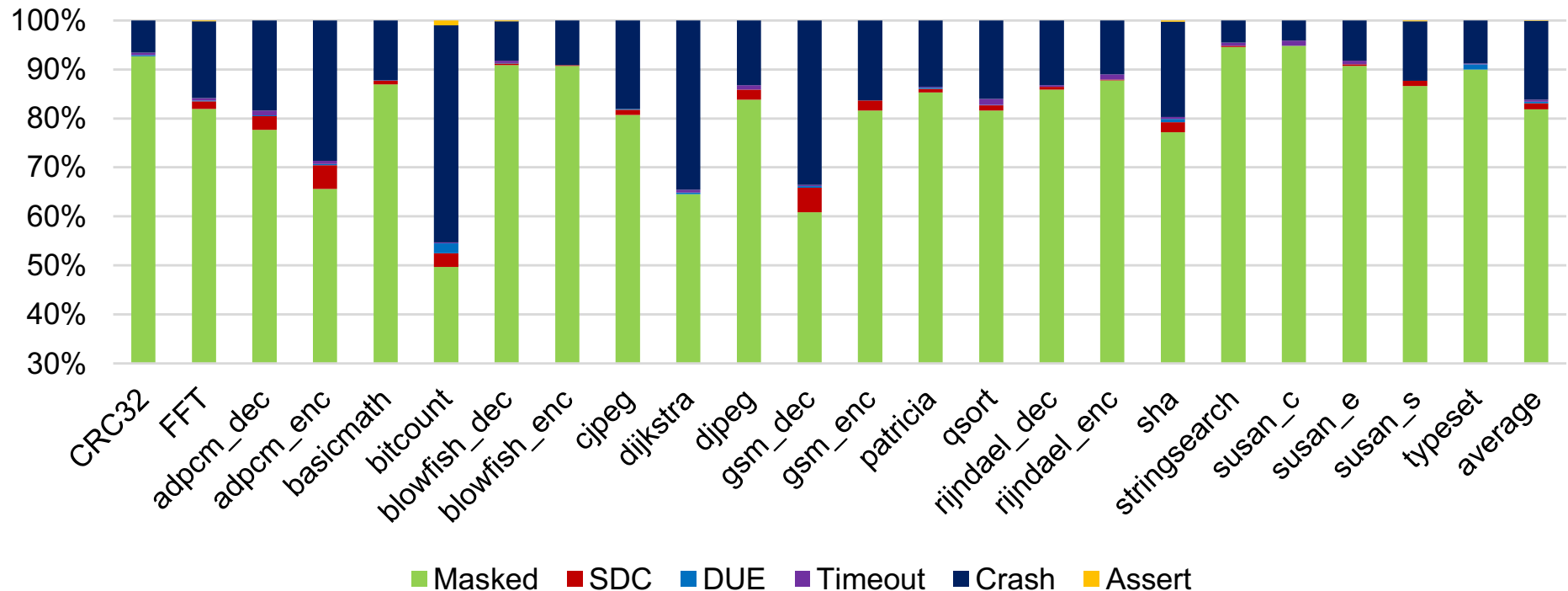
SQ Physical address



AVF/Fault Effects – A9

ARM®
Cortex-A9

Reorder Buffer PC state field



AVF/Fault Effects – A15



Reorder Buffer PC state field

