DiFuzzRTL: Differential Fuzz Testing to Find CPU Bugs

Jaewon Hur, Suhwan Song, Dongup Kwon, Eunjin Baek, Jangwoo Kim, Byoungyoung Lee

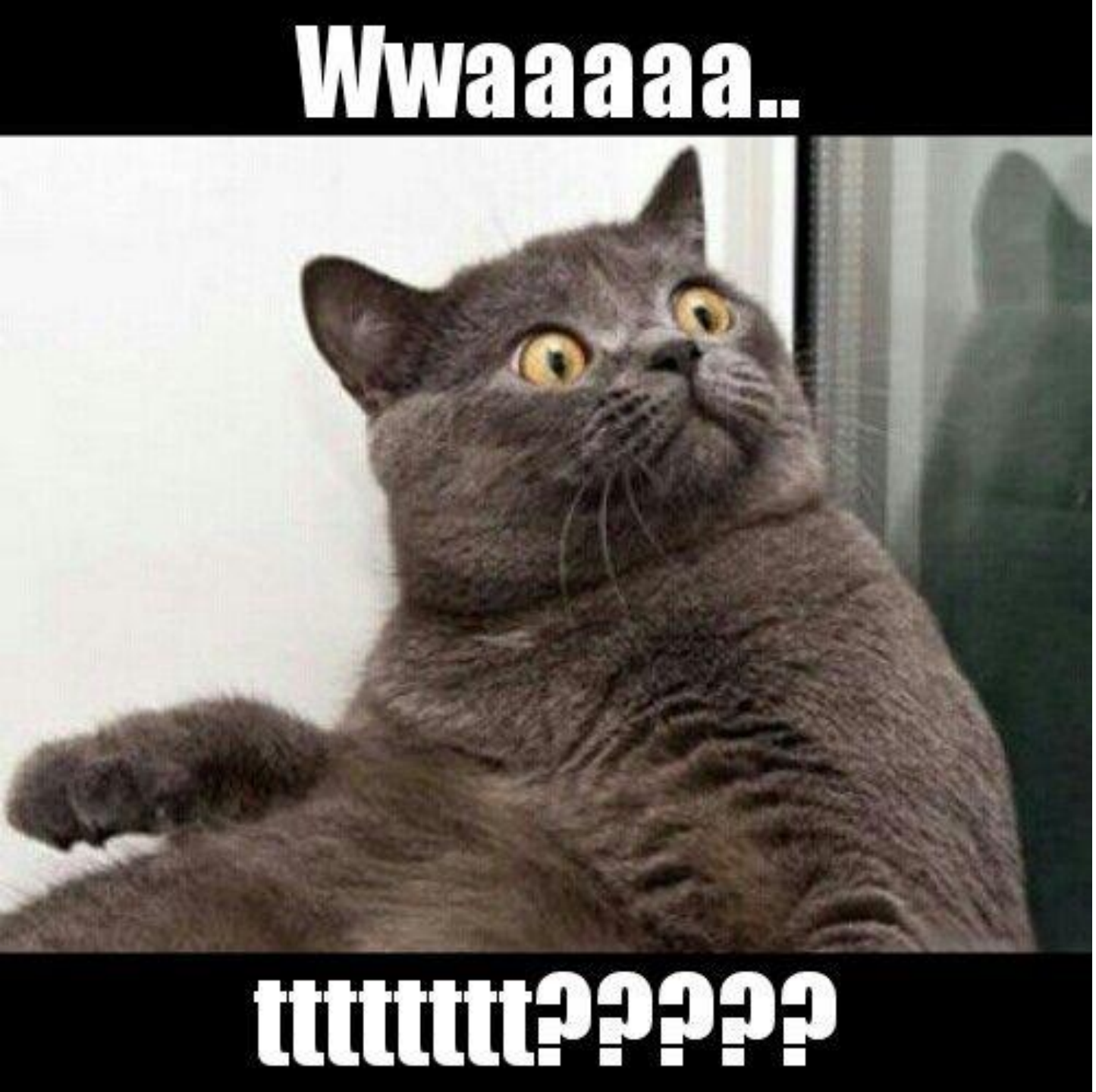
Computer Security Lab & High Performance Computer System Lab

Seoul National University

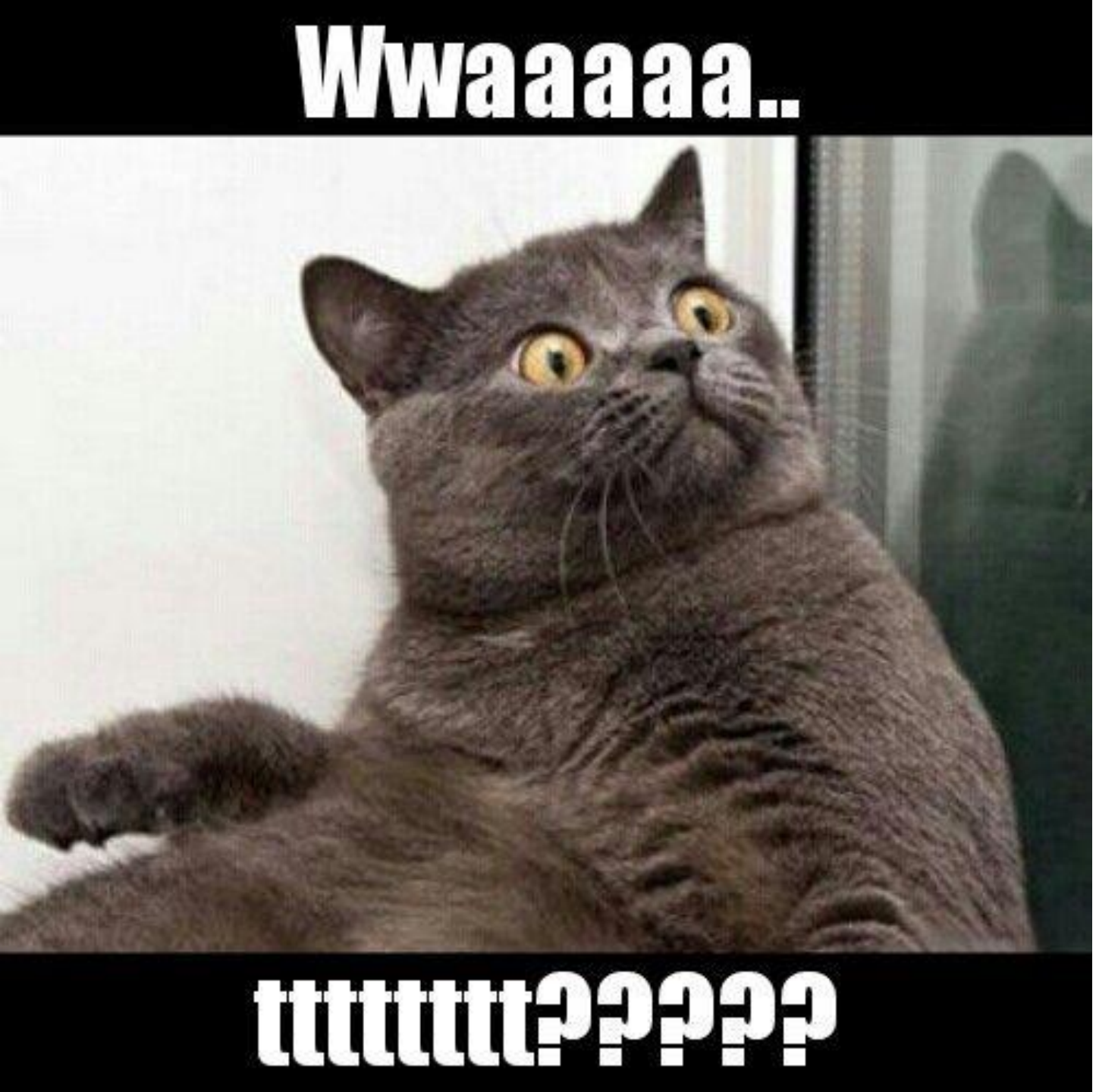
{hurjaewon, sshkeb96, dongup, ebaek, jangwoo, byoungyoung} @snu.ac.kr

서 울 대 학 교

SEOUL NATIONAL UNIVERSITY

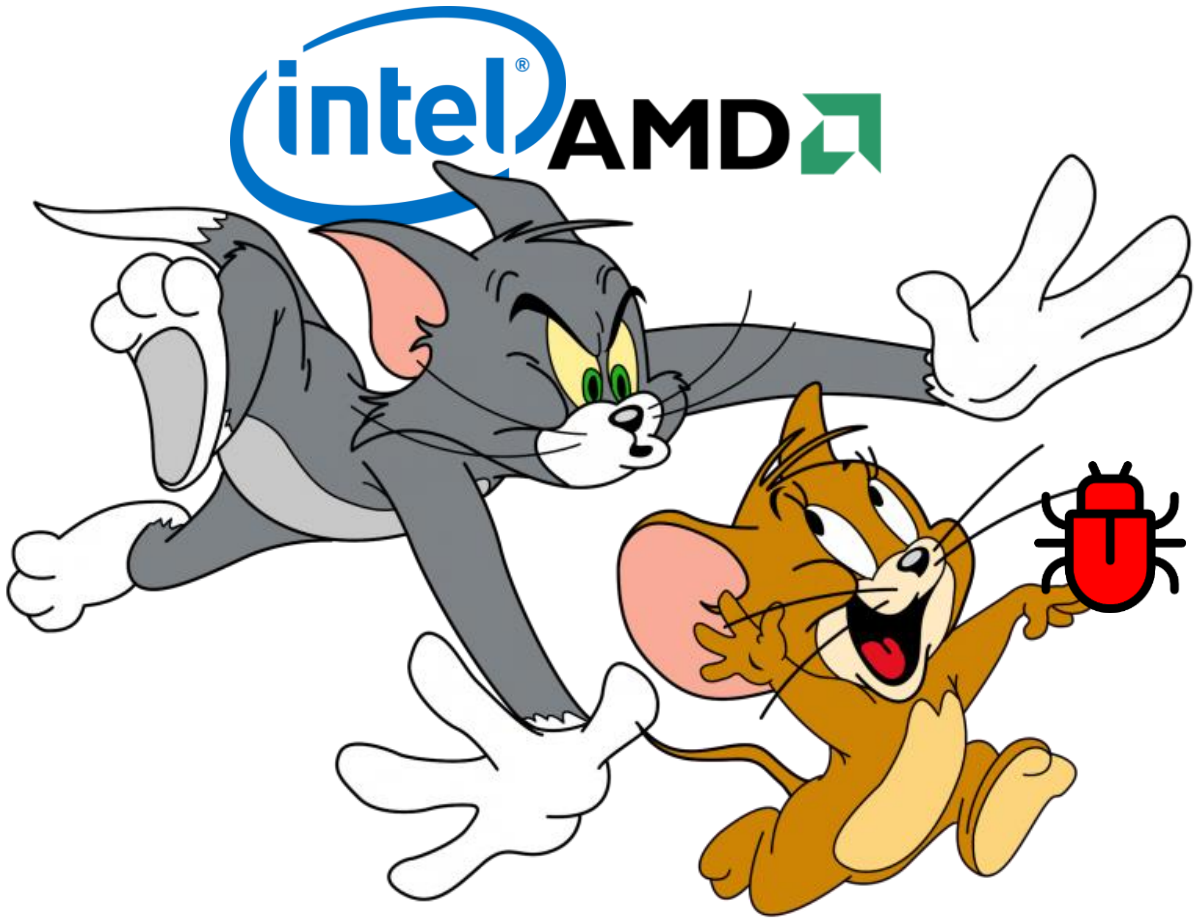


Pentium FDIV bug



$475 million

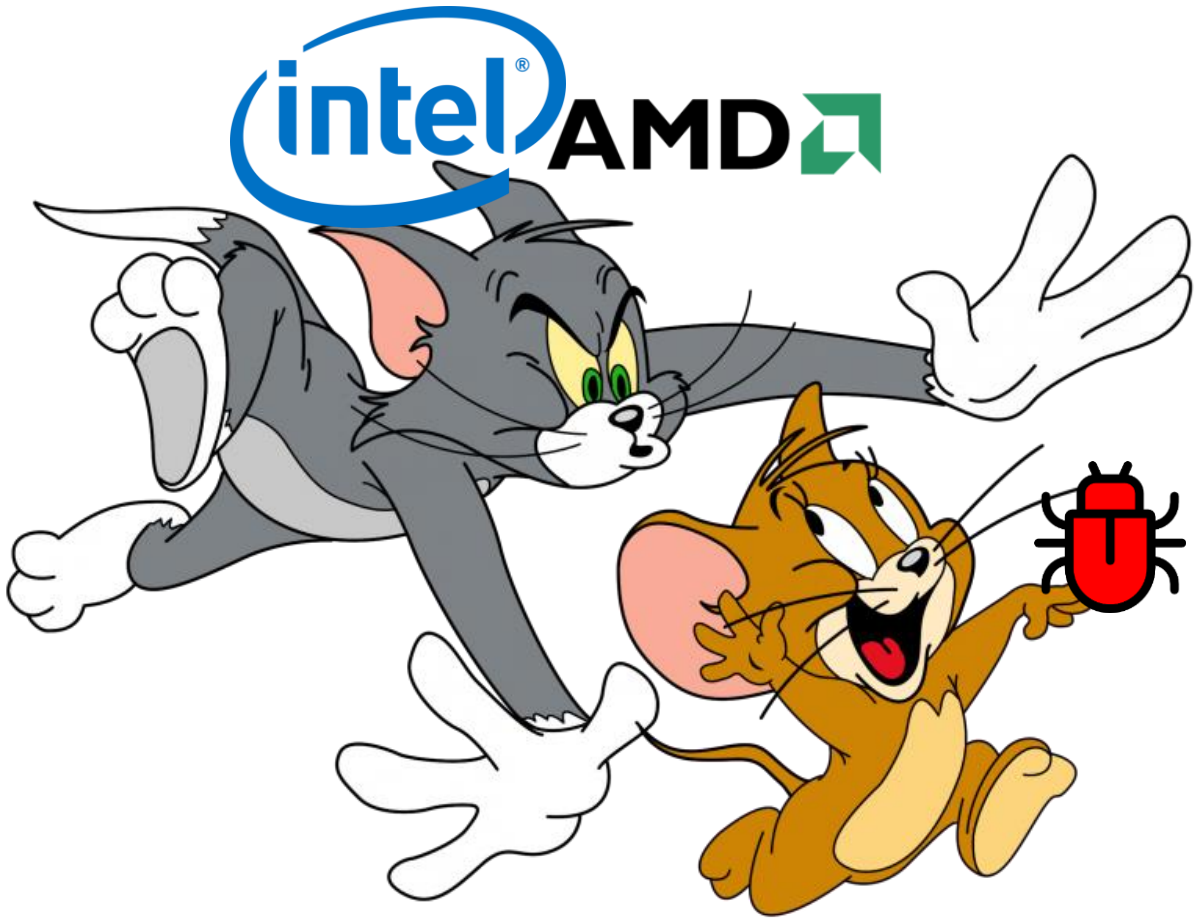
Continued Verification



CPU vendors invest huge efforts into the **Functional verification**

Check if **CPU RTL design** correctly follows **ISA**

Complete Verification is Difficult

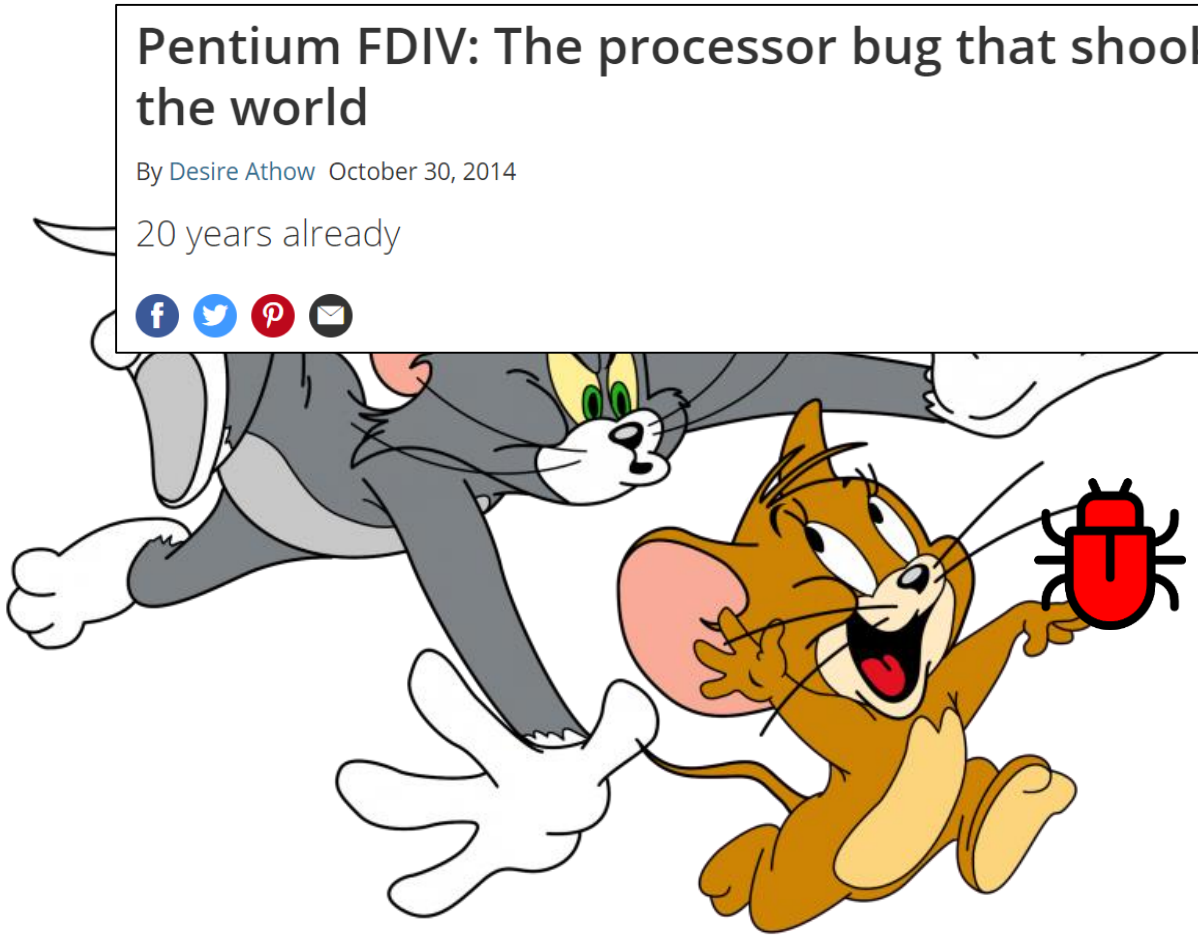


CPU vendors invest huge efforts into the **Functional verification**

Check if **CPU RTL design** correctly follows **ISA**

So we keep observing CPU bugs

Complete Verification is Difficult

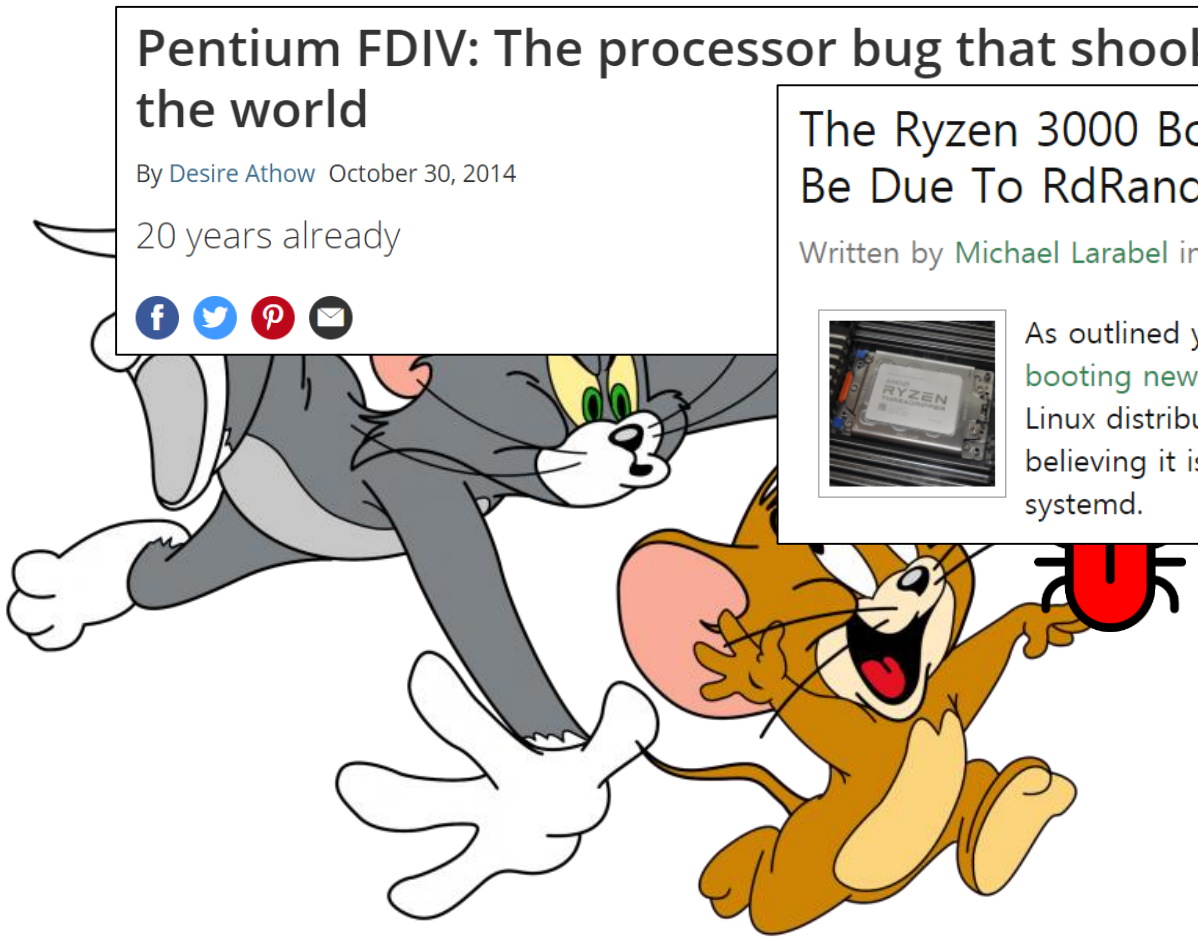


CPU vendors invest huge efforts into the **Functional verification**

Check if **CPU RTL design** correctly follows **ISA**

So we keep observing CPU bugs

Complete Verification is Difficult



Pentium FDIV: The processor bug that shook the world

By Desire Athow October 30, 2014

20 years already

The Ryzen 3000 Boot Problem With Newer Linux Distros Might Be Due To RdRand Issue

Written by Michael Larabel in AMD on 8 July 2019 at 09:42 AM EDT. 121 Comments

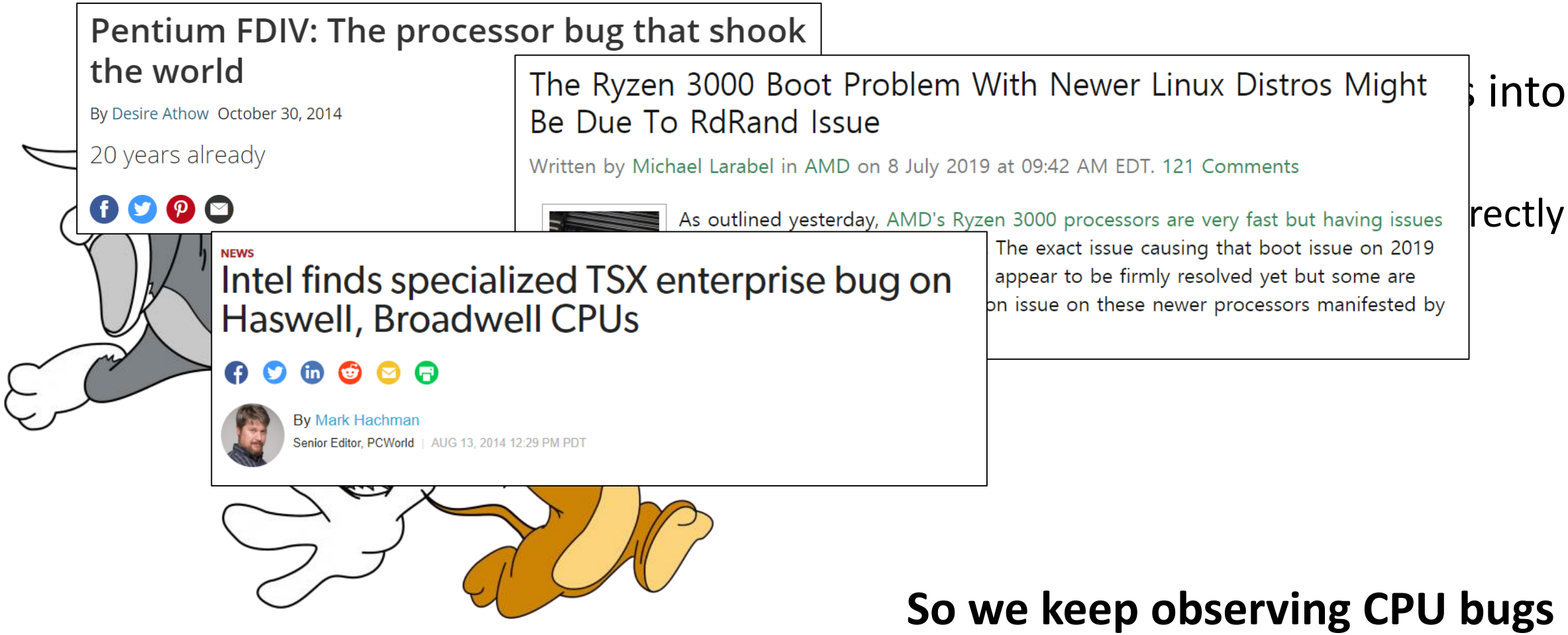
As outlined yesterday, AMD's Ryzen 3000 processors are very fast but having issues booting newer Linux distributions. The exact issue causing that boot issue on 2019 Linux distribution releases doesn't appear to be firmly resolved yet but some are believing it is an RdRand instruction issue on these newer processors manifested by systemd.

s into

rectly

So we keep observing CPU bugs

Complete Verification is Difficult

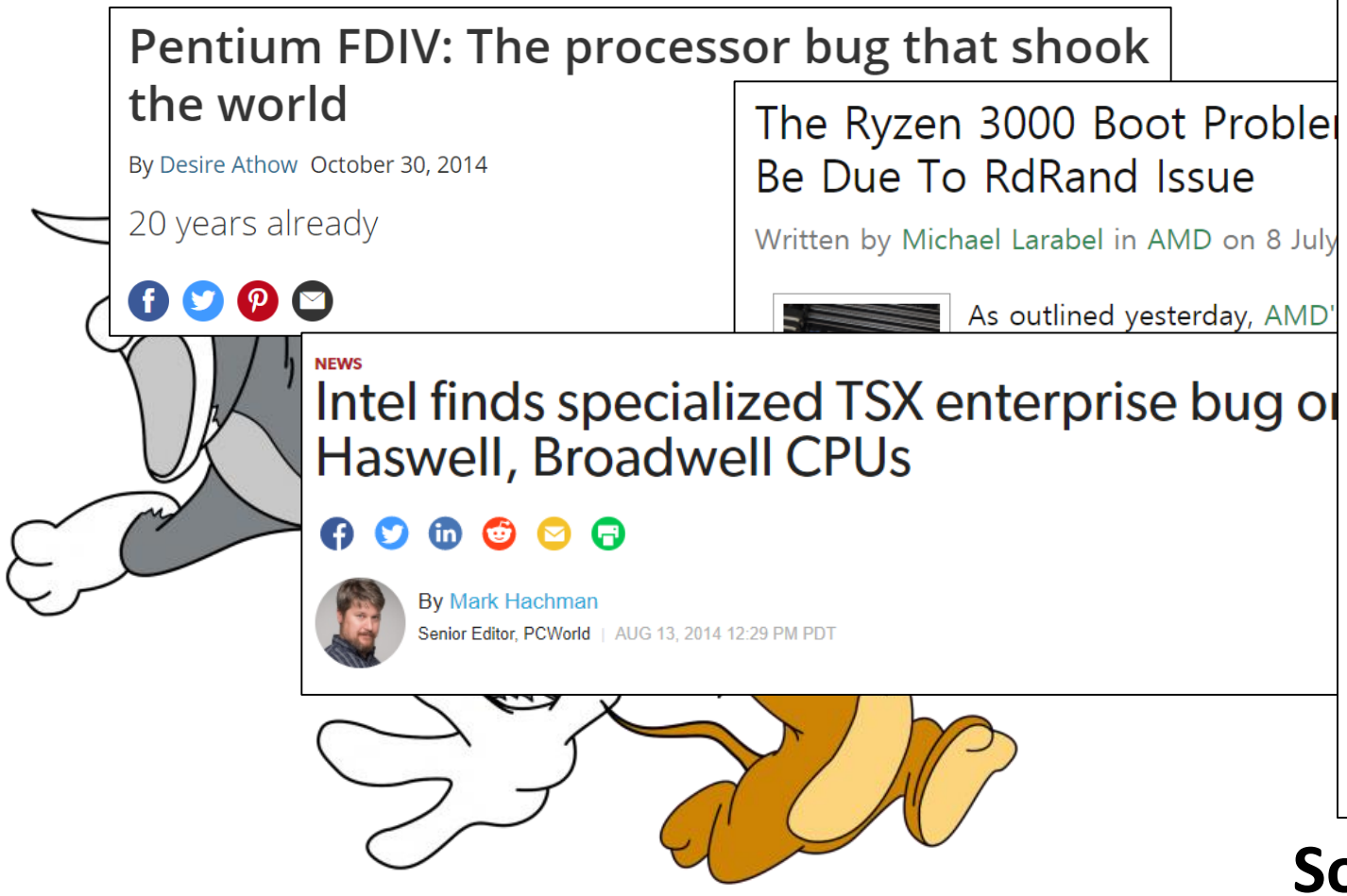


s into

rectly

So we keep observing CPU bugs

Complete Verification is Difficult



Pentium FDIV: The processor bug that shook the world

By Desire Athow October 30, 2014

20 years already

The Ryzen 3000 Boot Proble
Be Due To RdRand Issue

Written by Michael Larabel in AMD on 8 July

As outlined yesterday, AMD'

NEWS

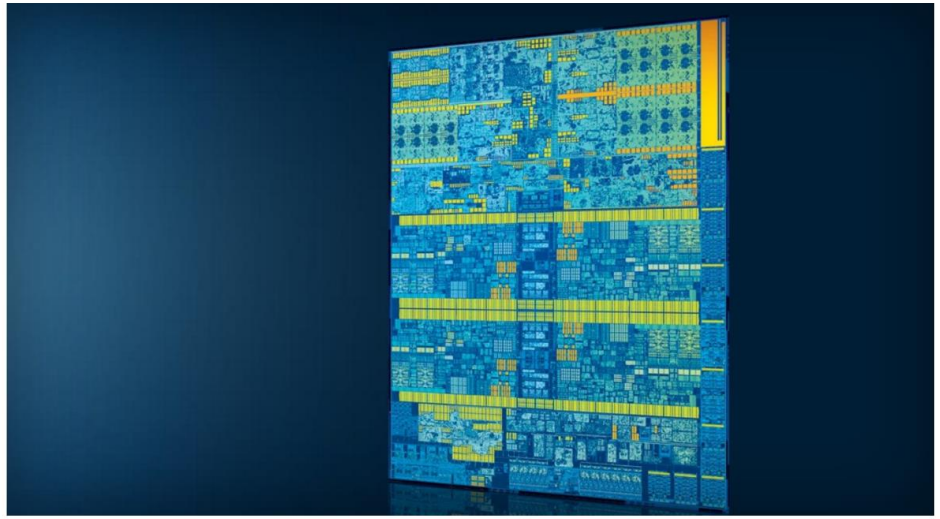
Intel finds specialized TSX enterprise bug o
Haswell, Broadwell CPUs

By Mark Hachman

Senior Editor, PCWorld | AUG 13, 2014 12:29 PM PDT

Skylake bug causes Intel chips to freeze under 'complex workloads'

By Joel Hruska on January 11, 2016 at 4:22 pm Comments



Intel has disclosed that its sixth-generation Core products (known as Skylake) suffer from a CPU bug that can cause a system to hang. The company has only publicly identified one application family that causes it, Prime95.

So we keep observing CPU bugs

DiFuzzRTL: Differential Fuzz Testing to Find CPU Bugs

DiFuzzRTL Found Real-world CPU Bugs

We found **16 real-world bugs** in **OpenRISC** and **RISC-V** CPUs

DiFuzzRTL Fuzzes CPU RTL Designs

What does the Fuzzer do ?

DiFuzzRTL Fuzzes CPU RTL Designs

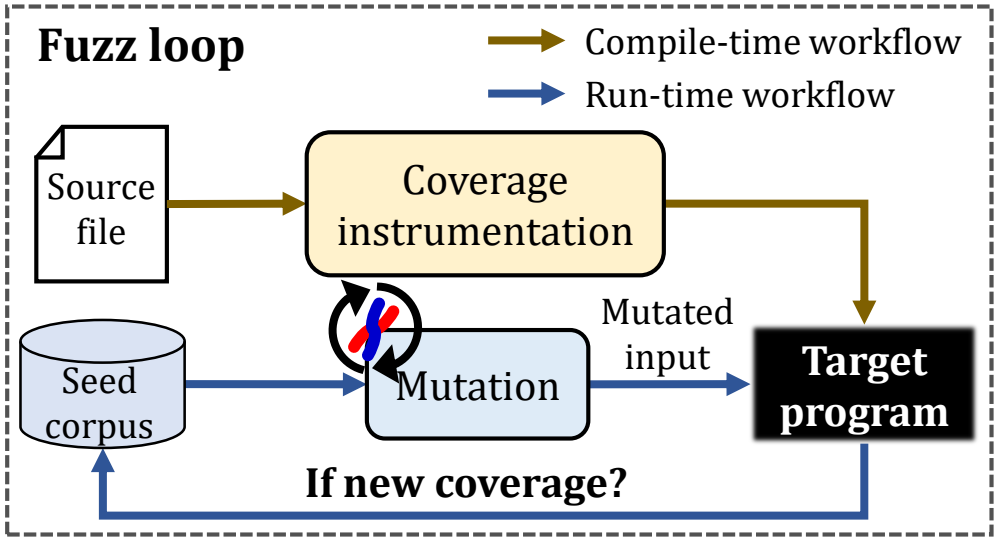
What does the Fuzzer do ?

Mutate inputs guided by a coverage!

DiFuzzRTL Fuzzes CPU RTL Designs

What does the Fuzzer do ?

Mutate inputs guided by a coverage!



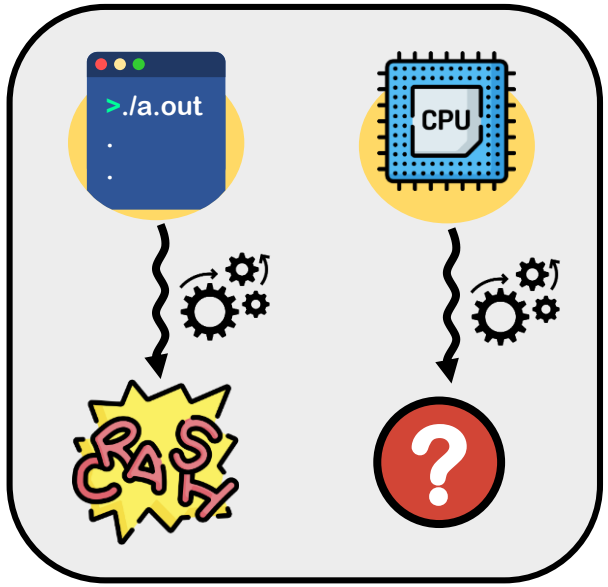
DiFuzzRTL Fuzzes CPU RTL Designs

How to fuzz CPU RTL designs ?

DiFuzzRTL Fuzzes CPU RTL Designs

How to fuzz CPU RTL designs ?

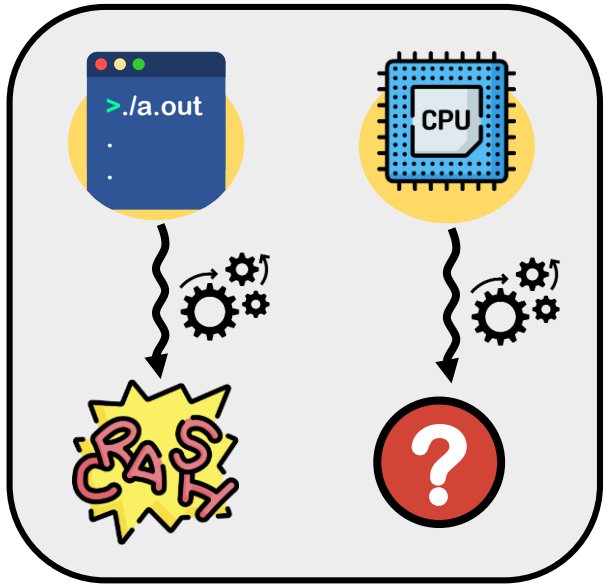
✓ **Requirement 1.** Framework for detecting the CPU bugs

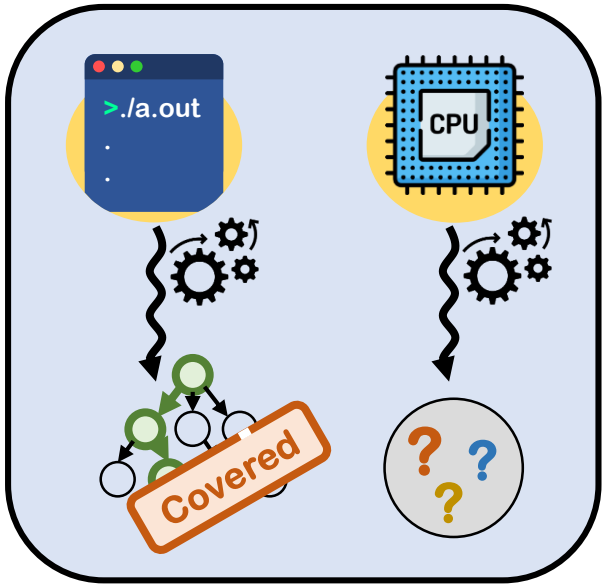


DiFuzzRTL Fuzzes CPU RTL Designs

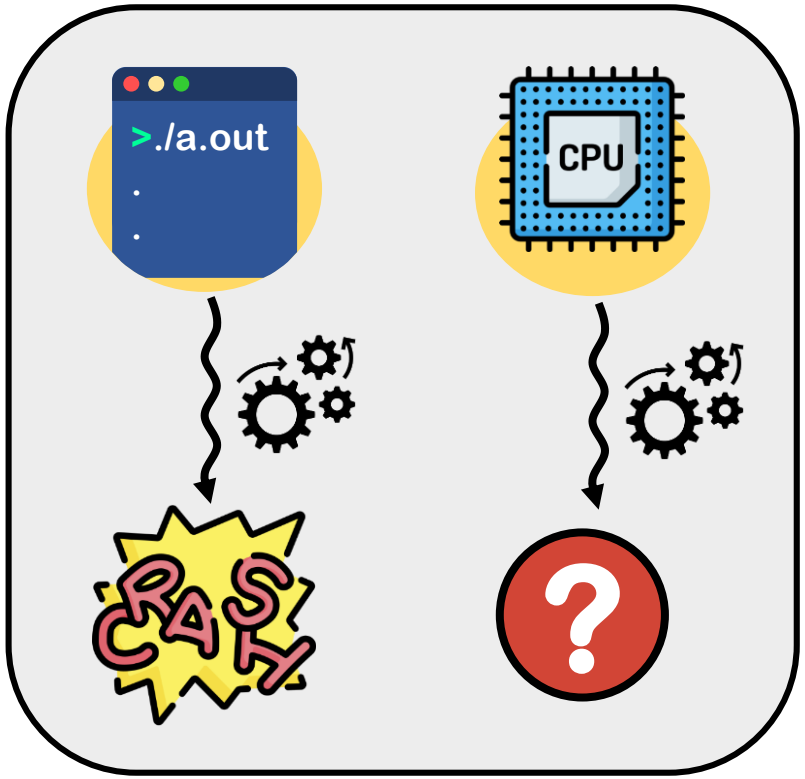
How to fuzz CPU RTL designs ?

- ✓ **Requirement 1.** Framework for detecting the CPU bugs
- ✓ **Requirement 2.** New coverage definition for the RTL designs





[1] Framework for detecting the CPU bugs

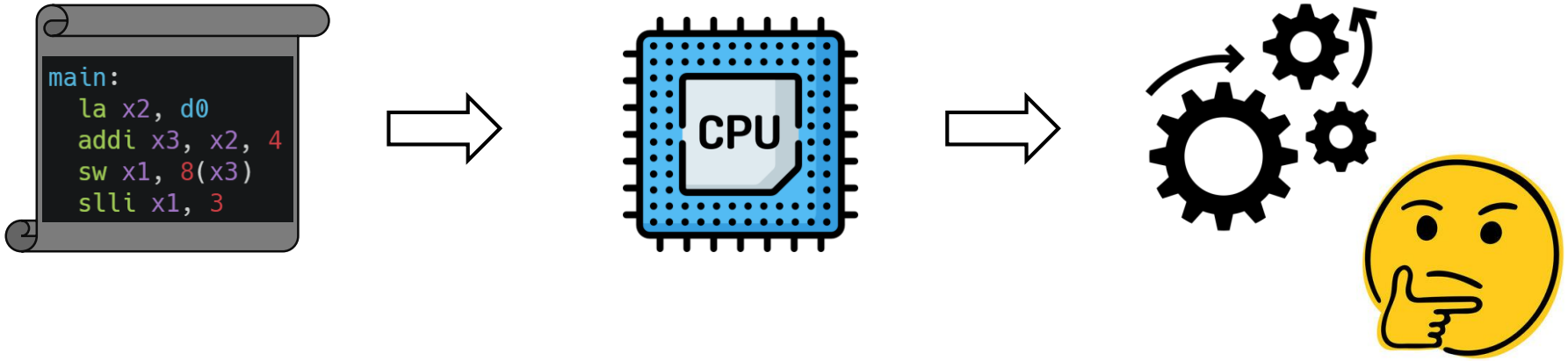


Detecting CPU Bugs

CPU bug

Abnormal CPU behavior

Different from a predefined ISA



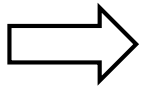
Framework for detecting the CPU bugs

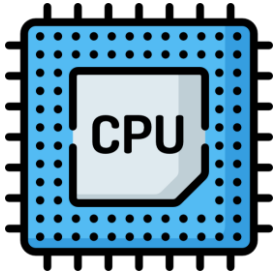
Detecting CPU Bugs

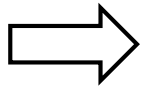
CPU bug

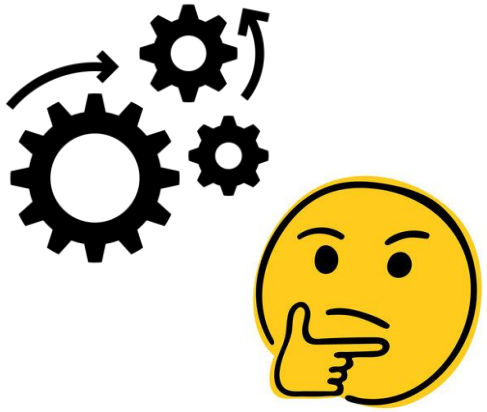
Abnormal CPU behavior

Different from a predefined ISA









✓ ***misaligned lr instruction***

Framework for detecting the CPU bugs

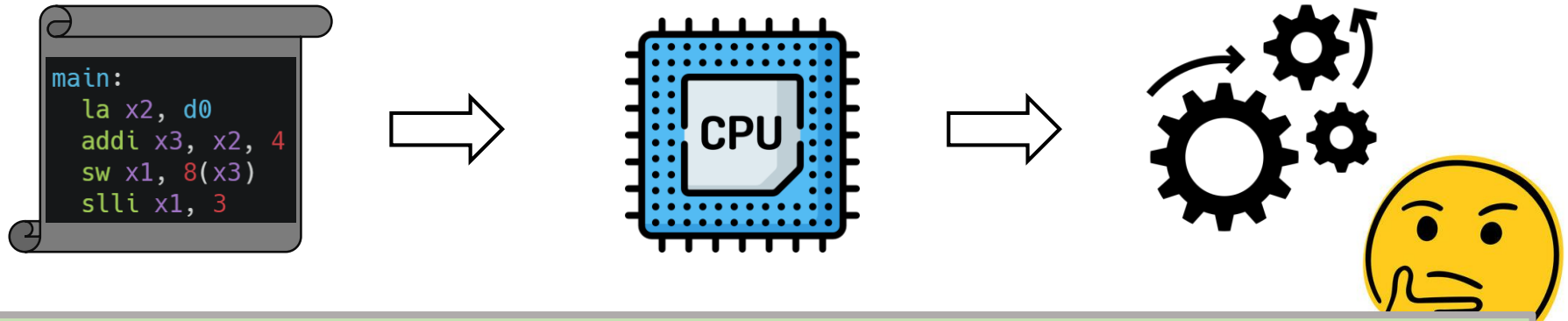
Detecting CPU Bugs

CPU bug

Abnormal CPU behavior

Different from a predefined ISA

✓ ***misaligned lr instruction***



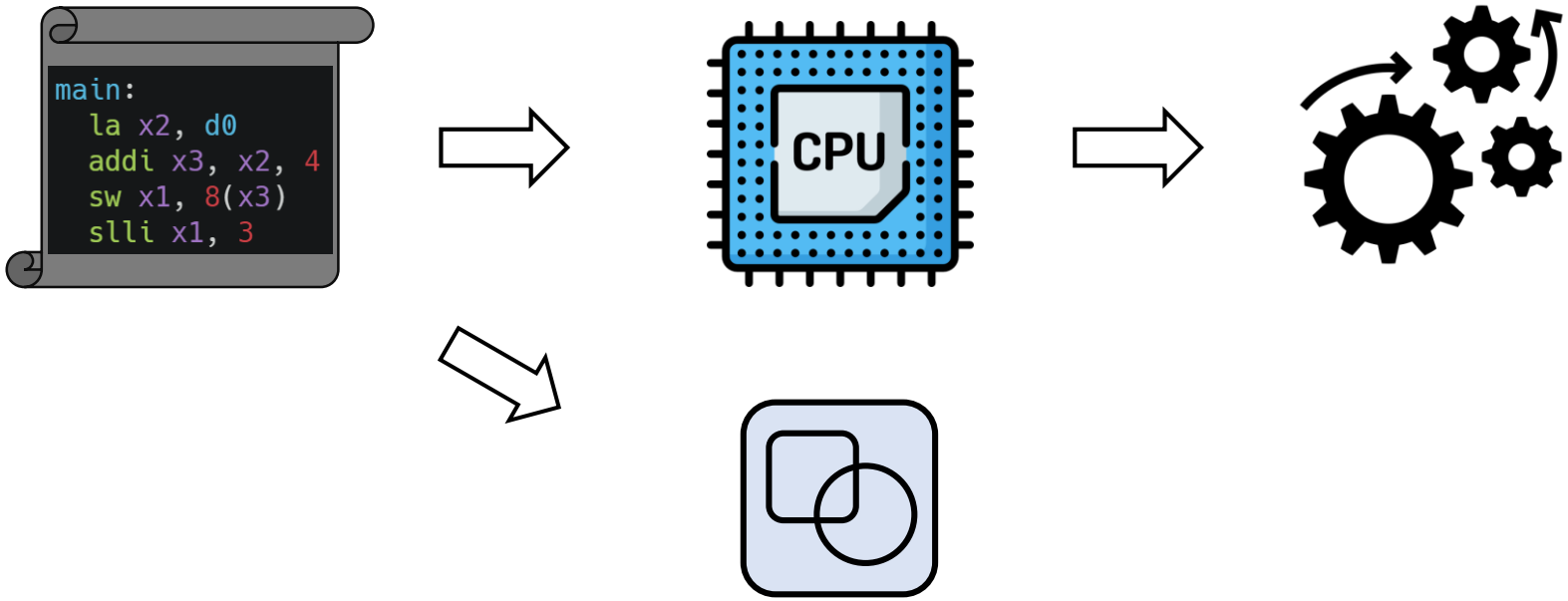
Testing CPU RTL design alone cannot detect bugs

Framework for detecting the CPU bugs

Differential Testing Framework

Detects CPU bugs by comparing with the ISA simulator

ISA simulator – Software implementation of the ISA

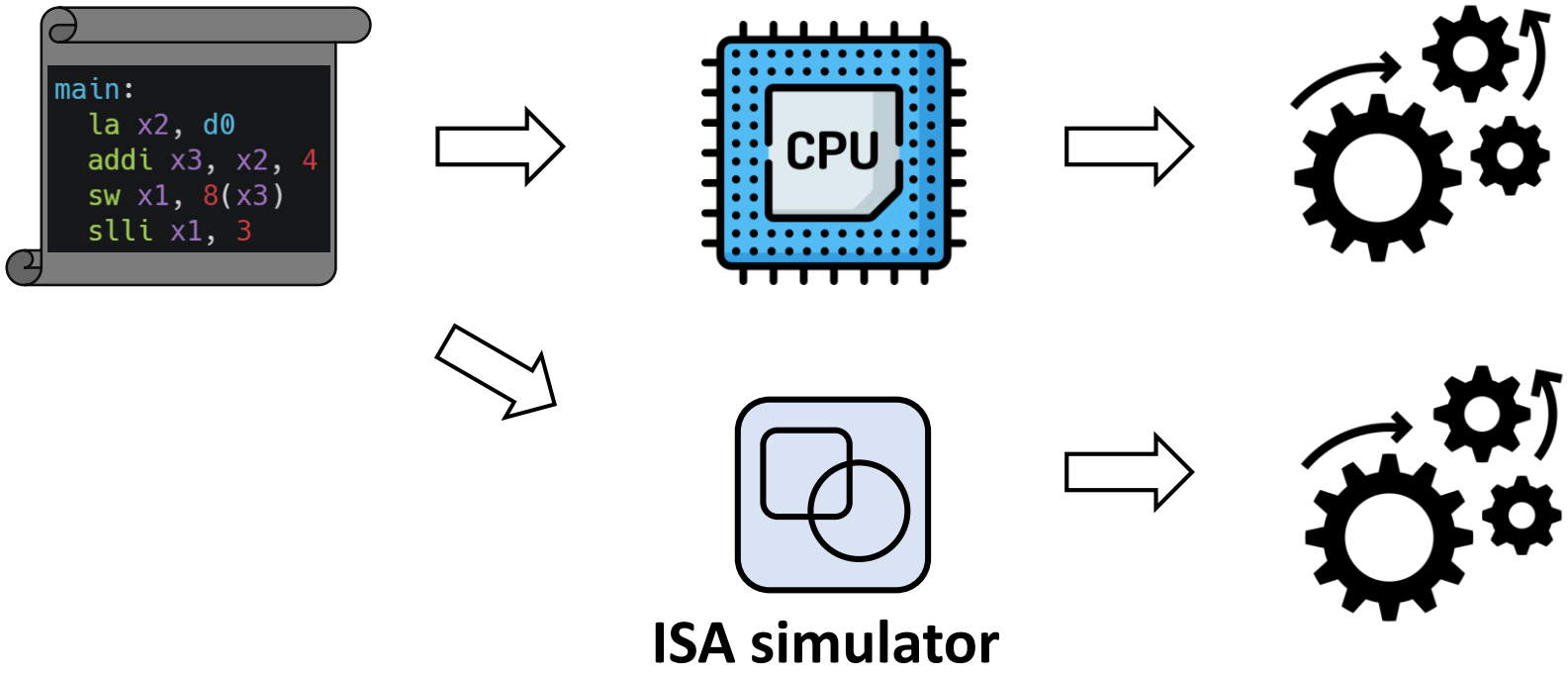


Framework for detecting the CPU bugs

Differential Testing Framework

Detects CPU bugs by comparing with the ISA simulator

ISA simulator – Software implementation of the ISA



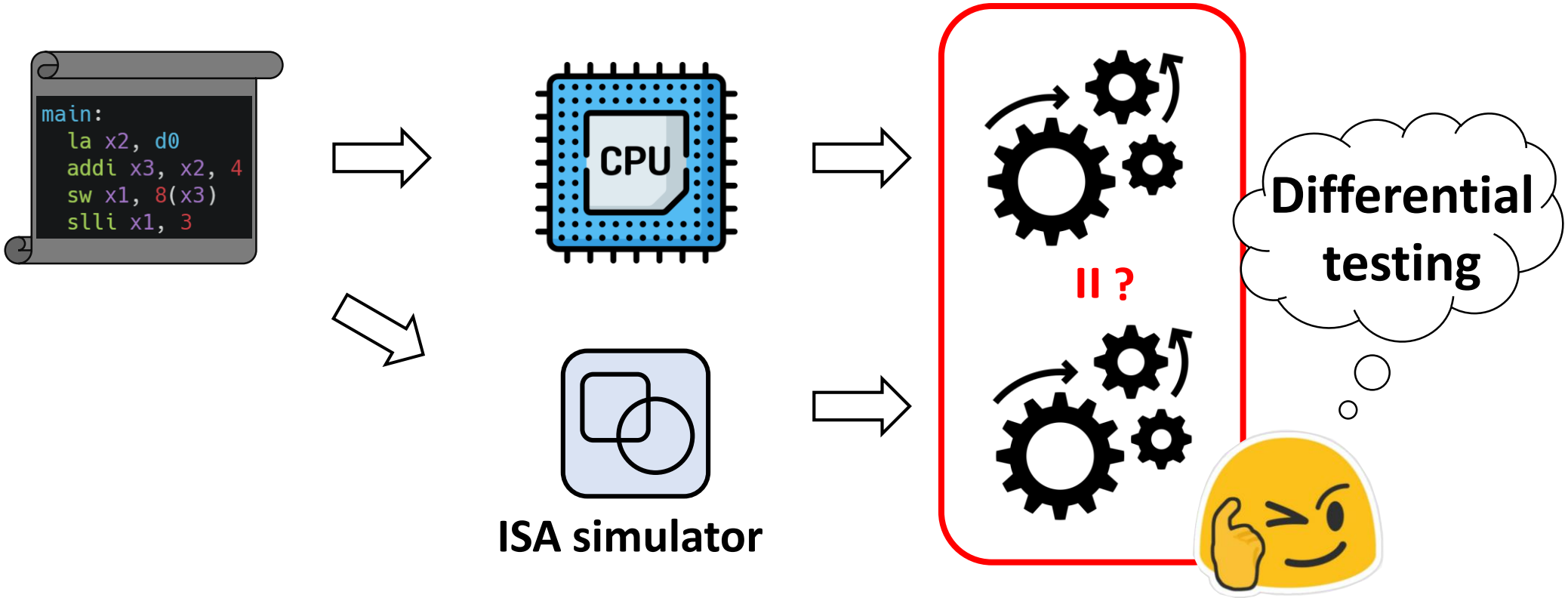
Framework for detecting the CPU bugs

Results should be same

Differential Testing Framework

Detects CPU bugs by comparing with the ISA simulator

ISA simulator – Software implementation of the ISA

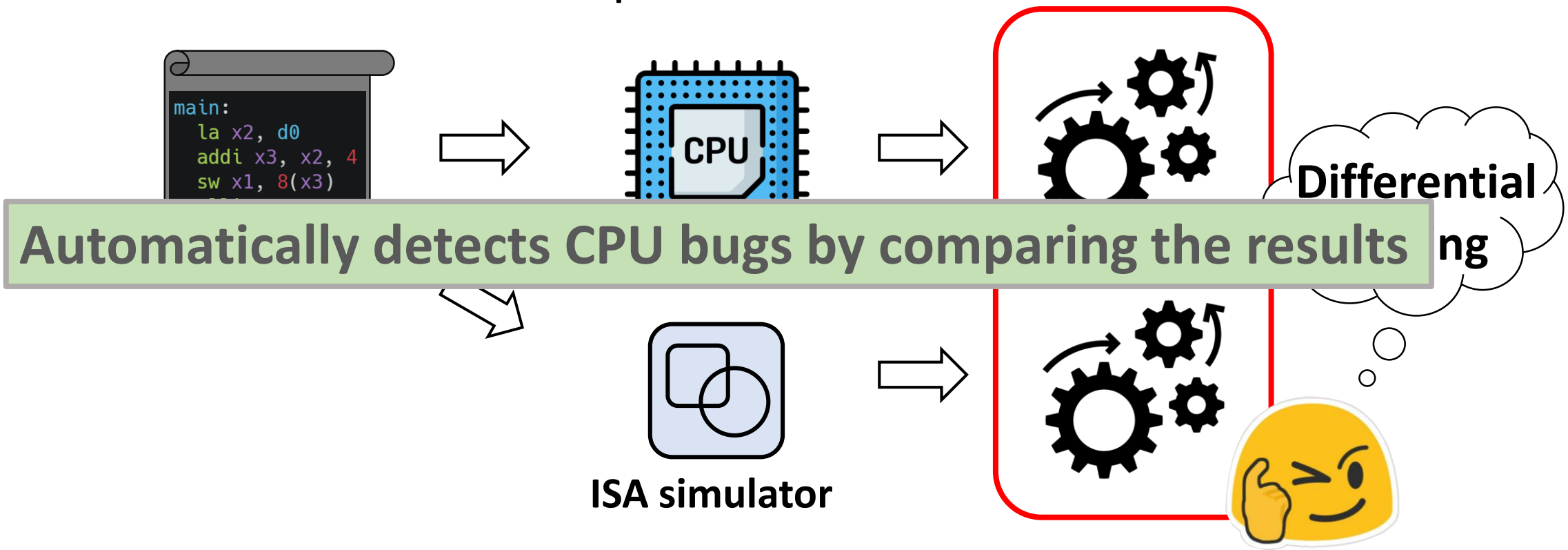


Framework for detecting the CPU bugs

Differential Testing Framework

Detects CPU bugs by comparing with the ISA simulator

ISA simulator – Software implementation of the ISA

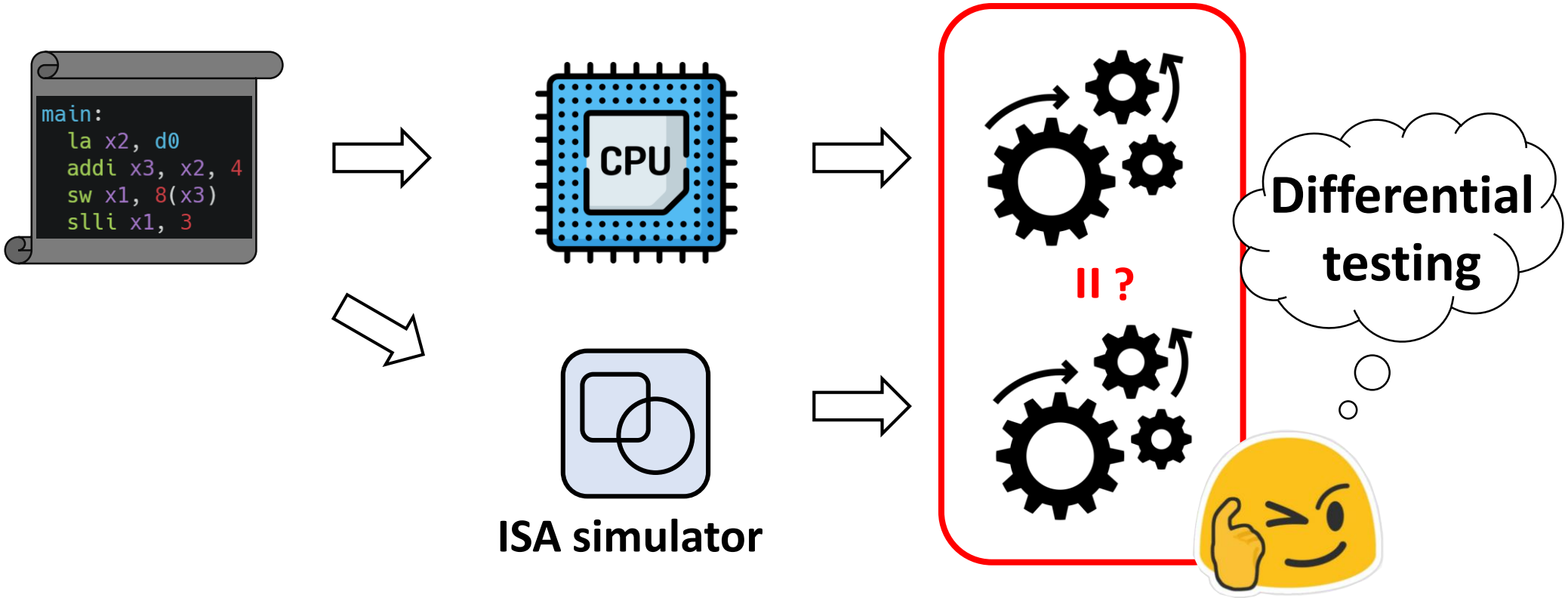


Framework for detecting the CPU bugs

Testing CPU RTL Design and ISA simulator

Defines SimInput for a unified input space

SimInput – Fuzz input containing instruction, data, and interrupt

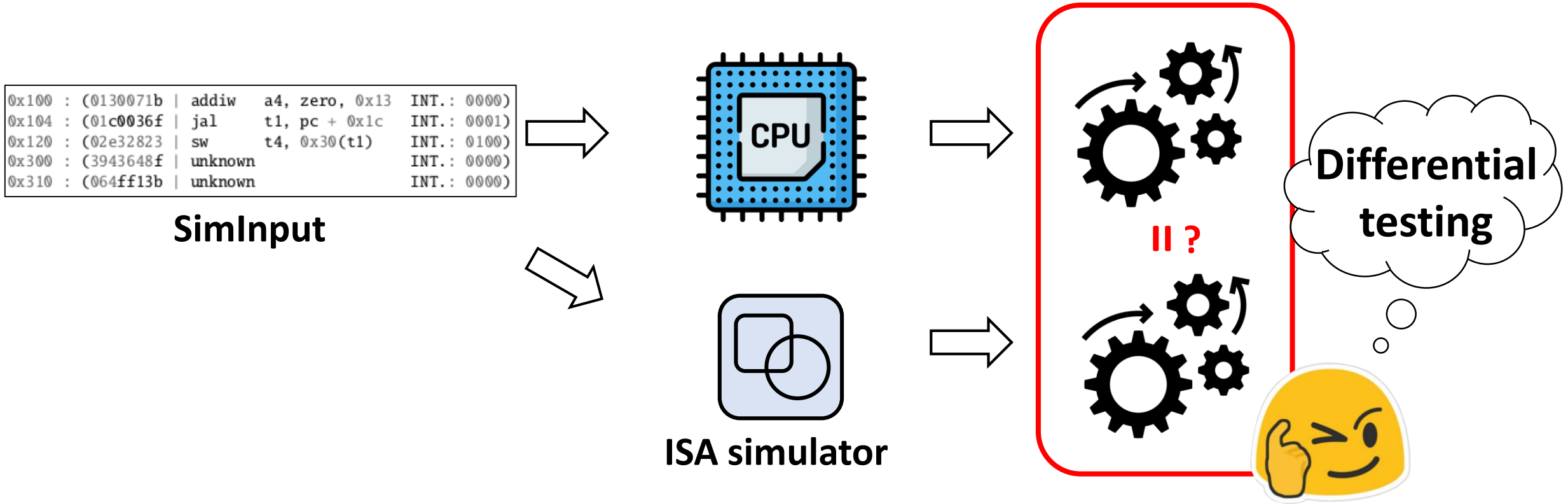


Framework for detecting the CPU bugs

Testing CPU RTL Design and ISA simulator

Defines SimInput for a unified input space

SimInput – Fuzz input containing instruction, data, and interrupt

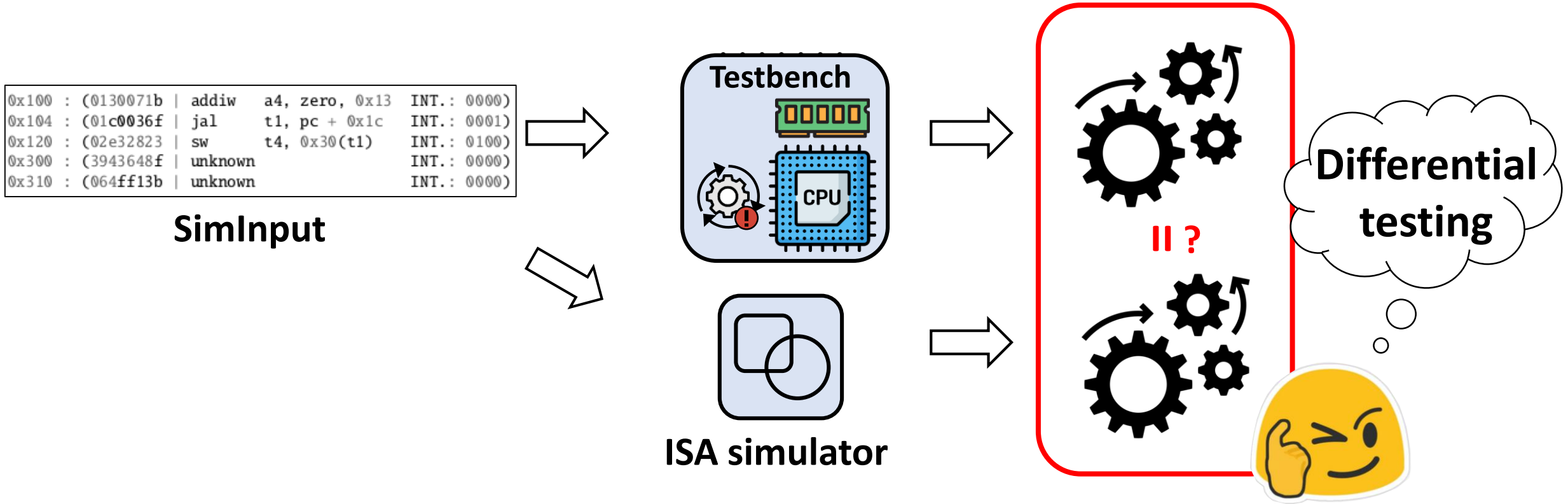


Framework for detecting the CPU bugs

Testing CPU RTL Design and ISA simulator

Defines SimInput for a unified input space

SimInput – Fuzz input containing instruction, data, and interrupt

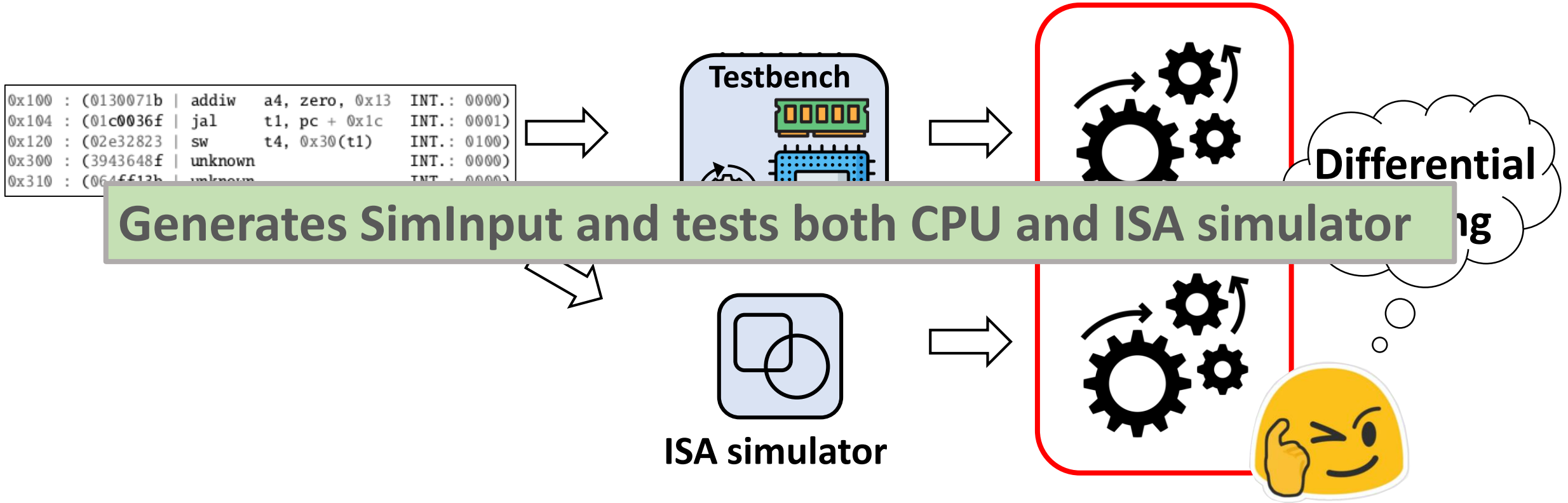


Framework for detecting the CPU bugs

Testing CPU RTL Design and ISA simulator

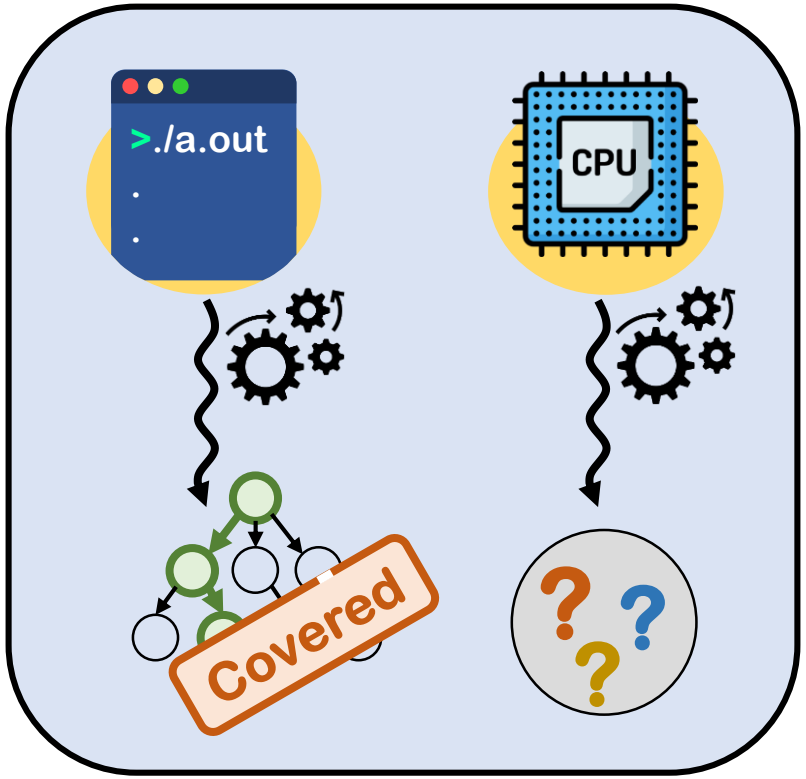
Defines SimInput for a unified input space

SimInput – Fuzz input containing instruction, data, and interrupt



Framework for detecting the CPU bugs

[2] New coverage definition for the RTL designs



Coverage for RTL Design

RTL (Register Transfer Level) ?

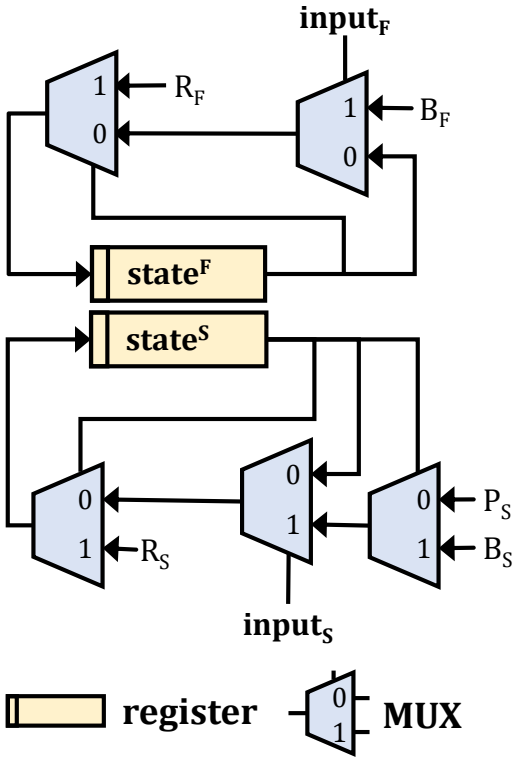
Abstraction to describe hardware circuit implementation

New coverage definition for the RTL designs

Coverage for RTL Design

RTL (Register Transfer Level) ?

Abstraction to describe hardware circuit implementation

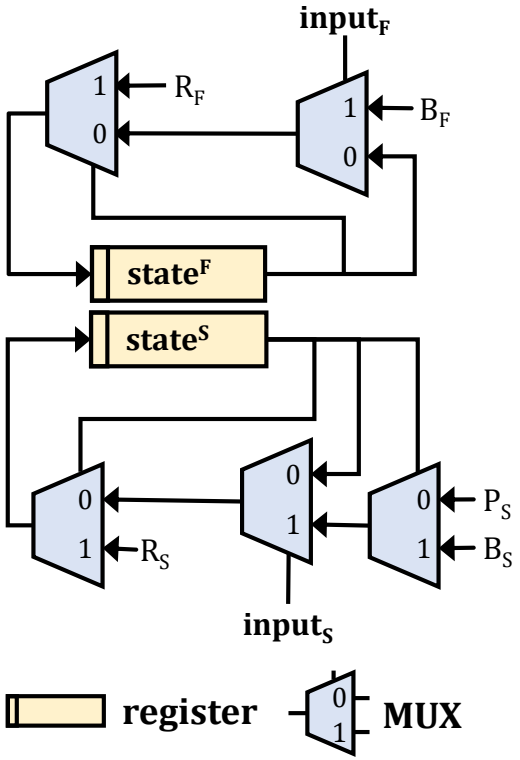


New coverage definition for the RTL designs

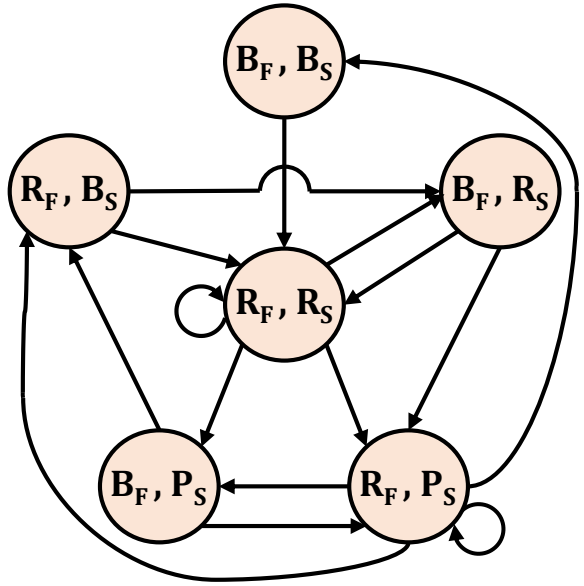
Coverage for RTL Design

RTL (Register Transfer Level) ?

Abstraction to describe hardware circuit implementation



FSM Modeling



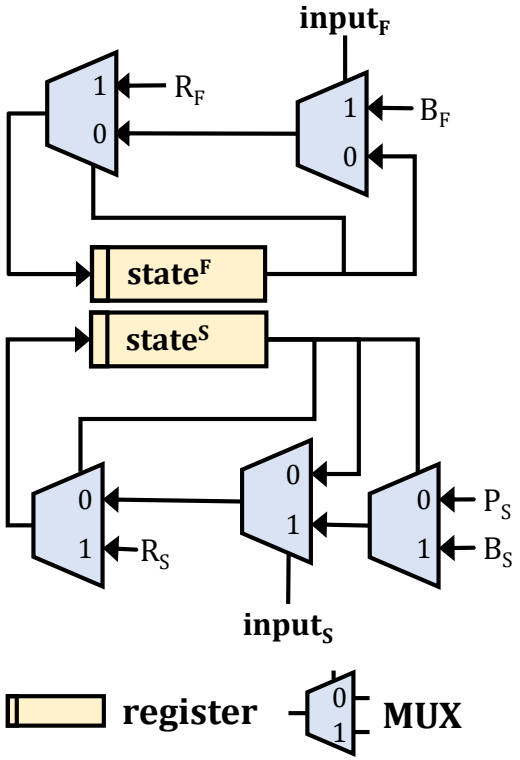
Finite State Machine (FSM)

New coverage definition for the RTL designs

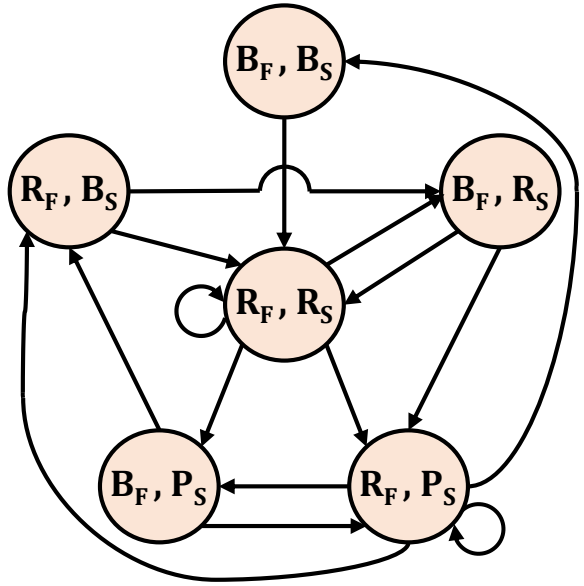
Coverage for RTL Design

Verification goal ?

Explore as many states in the FSM



FSM Modeling



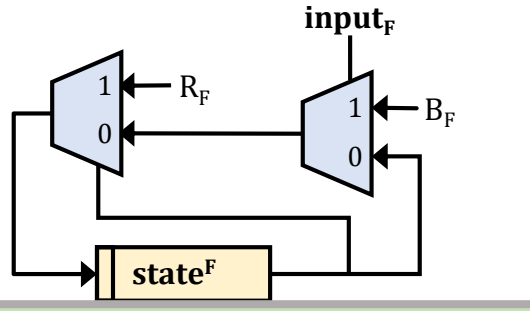
Finite State Machine (FSM)

New coverage definition for the RTL designs

Coverage for RTL Design

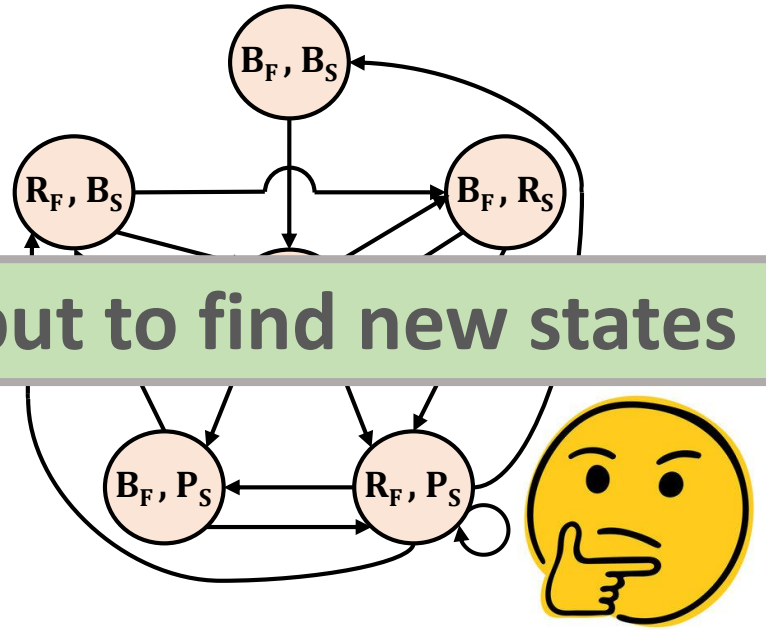
Verification goal ?

Explore as many states in the FSM



Coverage should guide the input to find new states

FSM Modeling

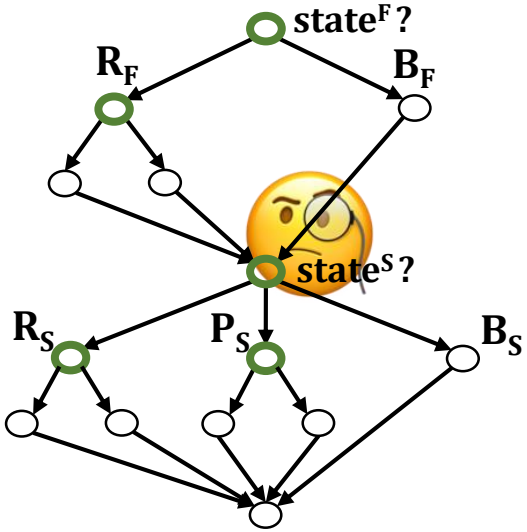


Finite State Machine (FSM)

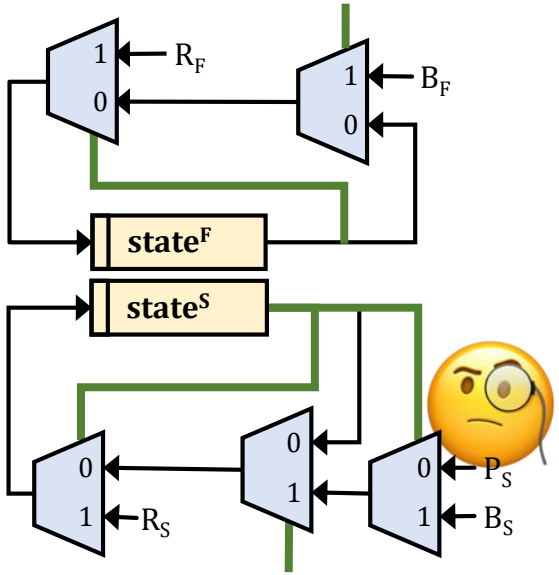
New coverage definition for the RTL designs

Limitation of Previous Coverage Measures

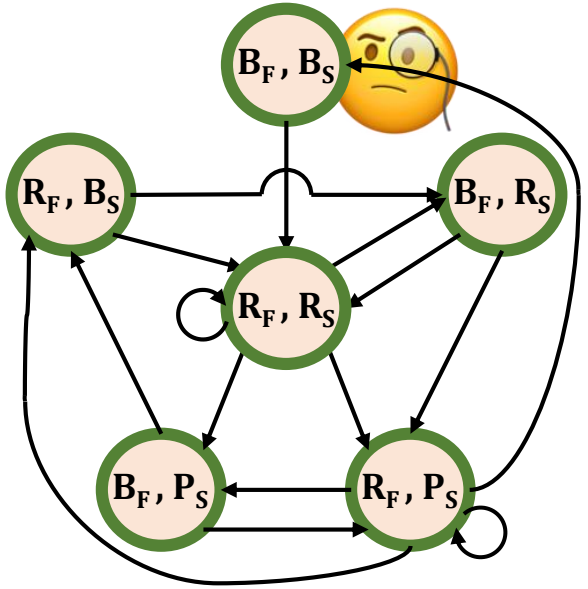
- **Branch coverage** [Vineeth et al. ETS'15], [Alif et al. DATE'18]
- **MUX control coverage** [Kevin et al. ICCAD'18]
- **FSM coverage** [Dinos et al. TC'98], [Jian et al. TCAD'15]



Branch coverage



MUX control coverage

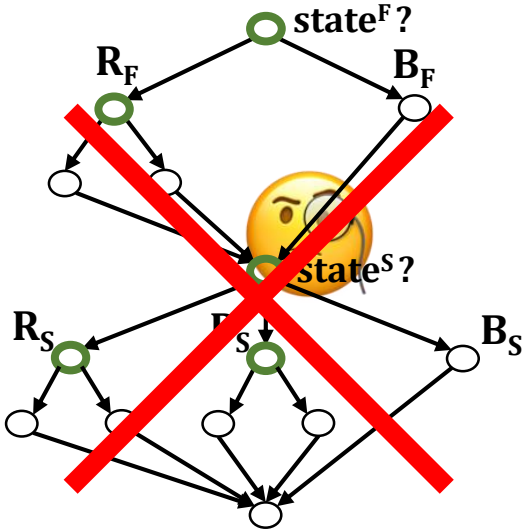


FSM coverage

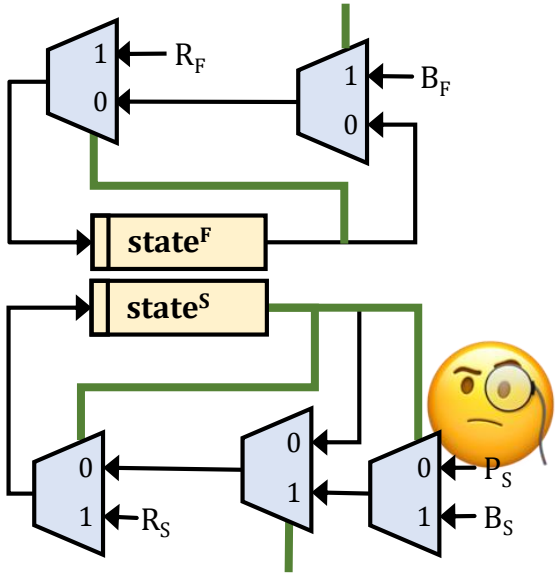
New coverage definition for the RTL designs

Limitation of Previous Coverage Measures

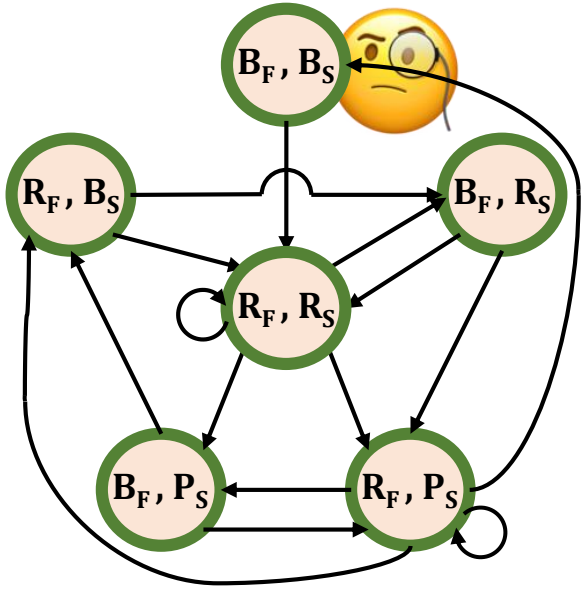
- ~~**Branch coverage** [Vineeth et al. ETS'15], [Alif et al. DATE'18]~~ ***Not accurate***
- **MUX control coverage** [Kevin et al. ICCAD'18]
- **FSM coverage** [Dinos et al. TC'98], [Jian et al. TCAD'15]



Cannot capture FSM



MUX control coverage



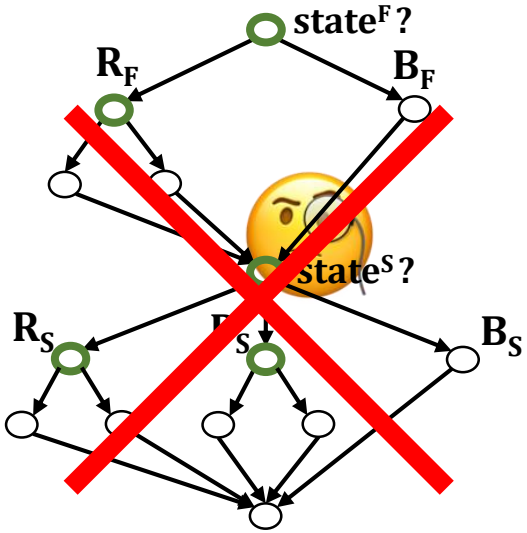
FSM coverage

New coverage definition for the RTL designs

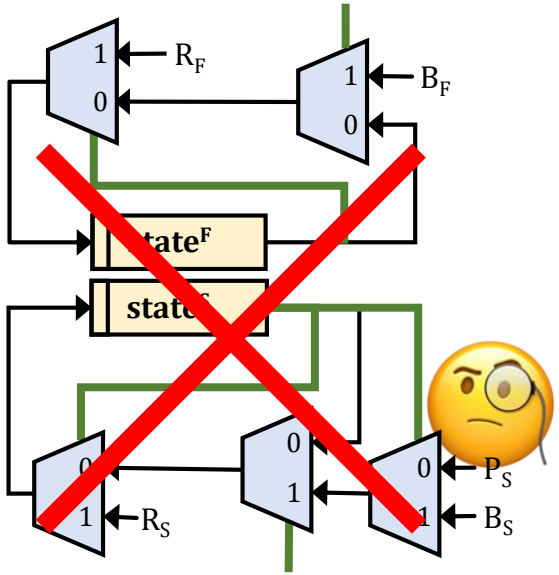
Limitation of Previous Coverage Measures

- ~~**Branch coverage** [Vineeth et al. ETS'15], [Alif et al. DATE'18]~~
- ~~**MUX control coverage** [Kevin et al. ICCAD'18]~~
- **FSM coverage** [Dinos et al. TC'98], [Jian et al. TCAD'15]

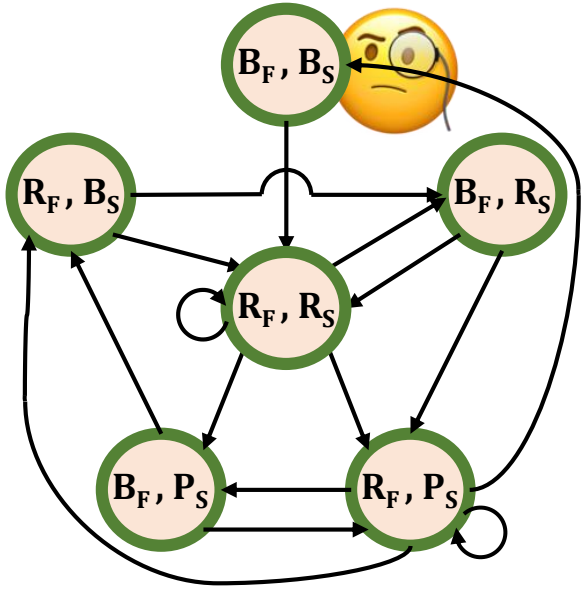
Not accurate
Not efficient



Cannot capture FSM



Incurs large instrument overhead



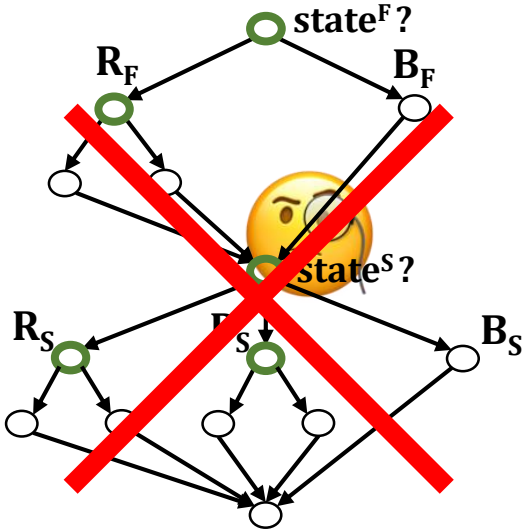
FSM coverage

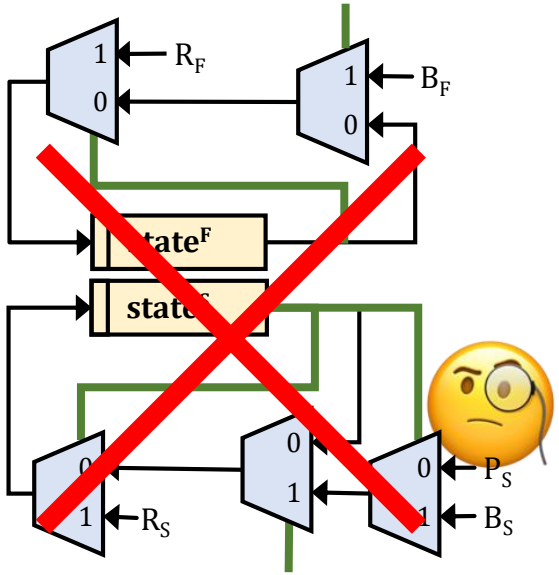
New coverage definition for the RTL designs

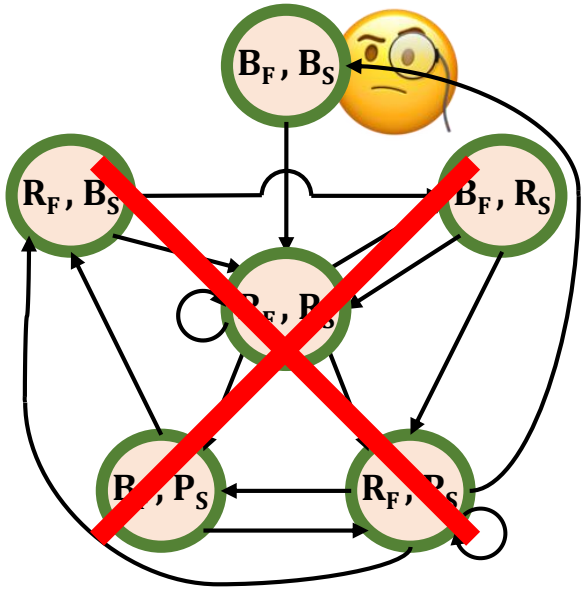
Limitation of Previous Coverage Measures

- ~~**Branch coverage** [Vineeth et al. ETS'15], [Alif et al. DATE'18]~~
- ~~**MUX control coverage** [Kevin et al. ICCAD'18]~~
- ~~**FSM coverage** [Dinos et al. TC'98], [Jian et al. TCAD'15]~~

Not accurate
Not efficient
Not automatic







Cannot capture FSM **Incurs large instrument overhead** **Needs manual efforts**

New coverage definition for the RTL designs

Register Coverage: New Coverage for RTL

New coverage definition for the RTL designs

Register Coverage: New Coverage for RTL

Accurate: correctly captures FSM exploration

New coverage definition for the RTL designs

Register Coverage: New Coverage for RTL

Accurate: correctly captures FSM exploration

Efficient: incurs only 7% runtime overhead

New coverage definition for the RTL designs

Register Coverage: New Coverage for RTL

Accurate: correctly captures FSM exploration

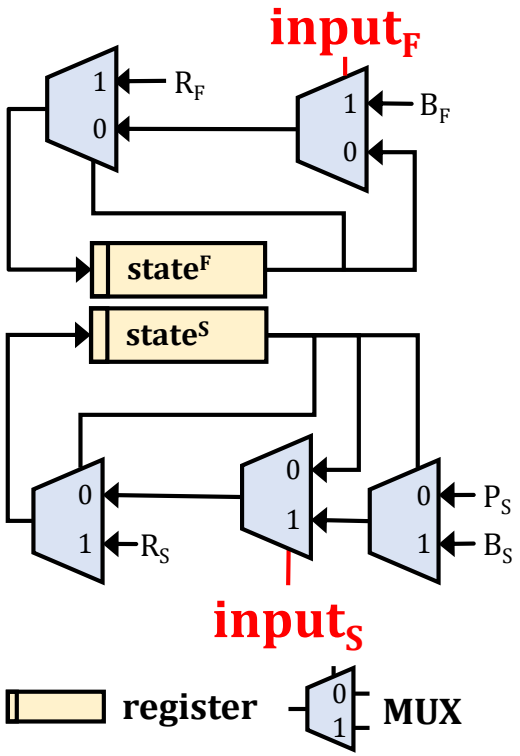
Efficient: incurs only 7% runtime overhead

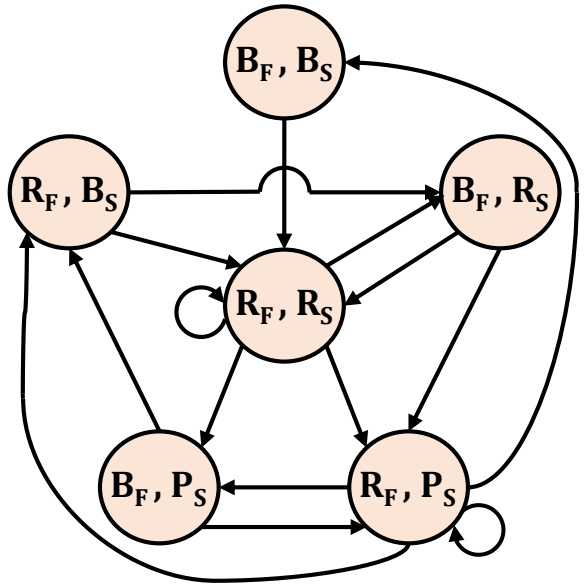
Automatic: requires no manual effort from developers

New coverage definition for the RTL designs

Capturing FSM Exploration

Monitors registers to correctly capture the FSM exploration



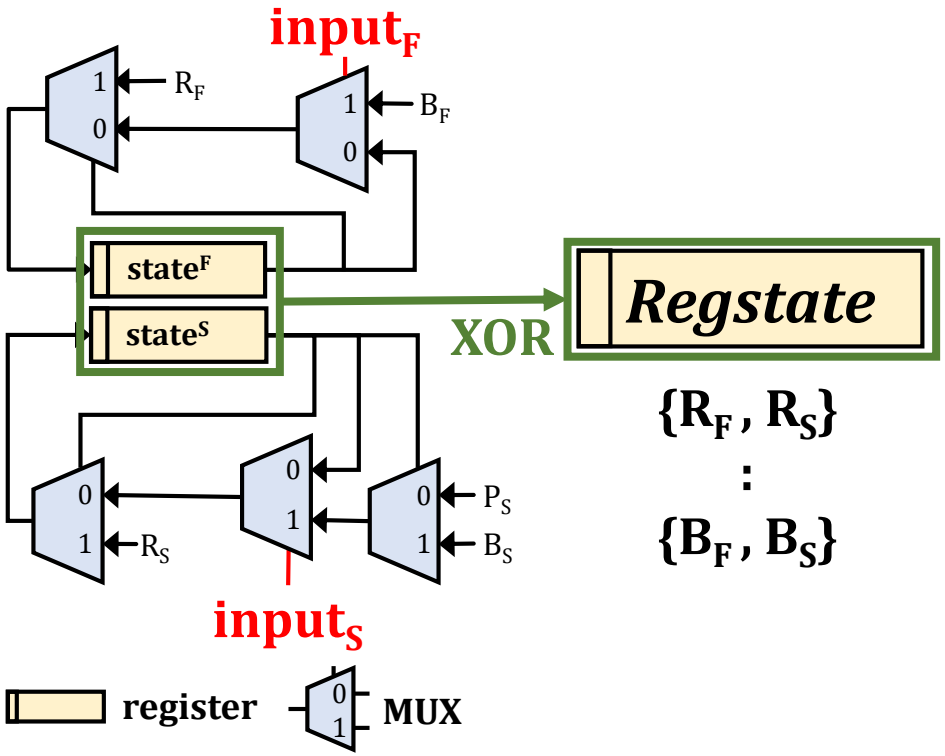


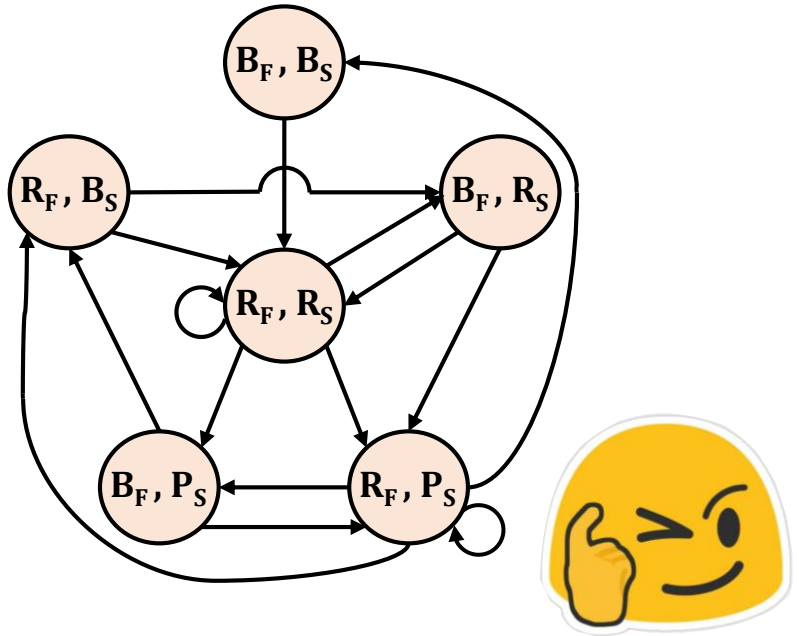
Finite State Machine (FSM)

New coverage definition for the RTL designs

Capturing FSM Exploration

Monitors registers to correctly capture the FSM exploration





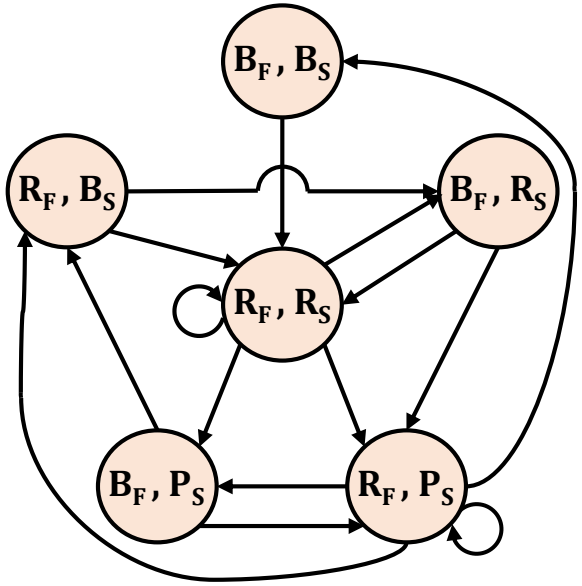
Finite State Machine (FSM)

New coverage definition for the RTL designs

Capturing FSM Exploration

Monitors registers to correctly capture the FSM exploration

Cycle	0	1	2
$input_F$	1	0	0
$input_S$	1	1	0



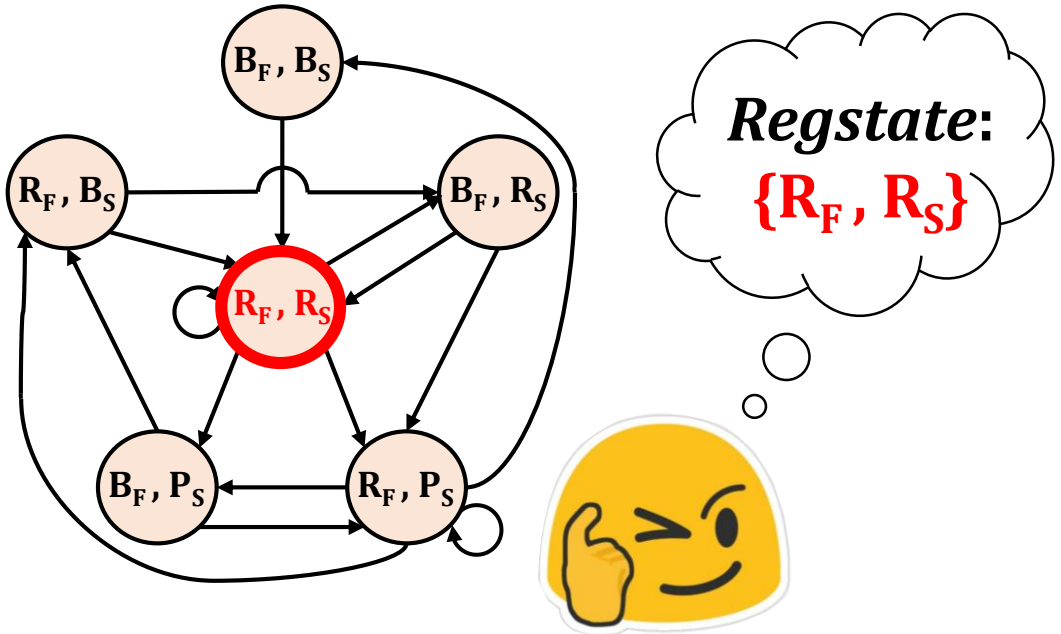
Finite State Machine (FSM)

New coverage definition for the RTL designs

Capturing FSM Exploration

Monitors registers to correctly capture the FSM exploration

Cycle	0	1	2
$input_F$	1	0	0
$input_S$	1	1	0



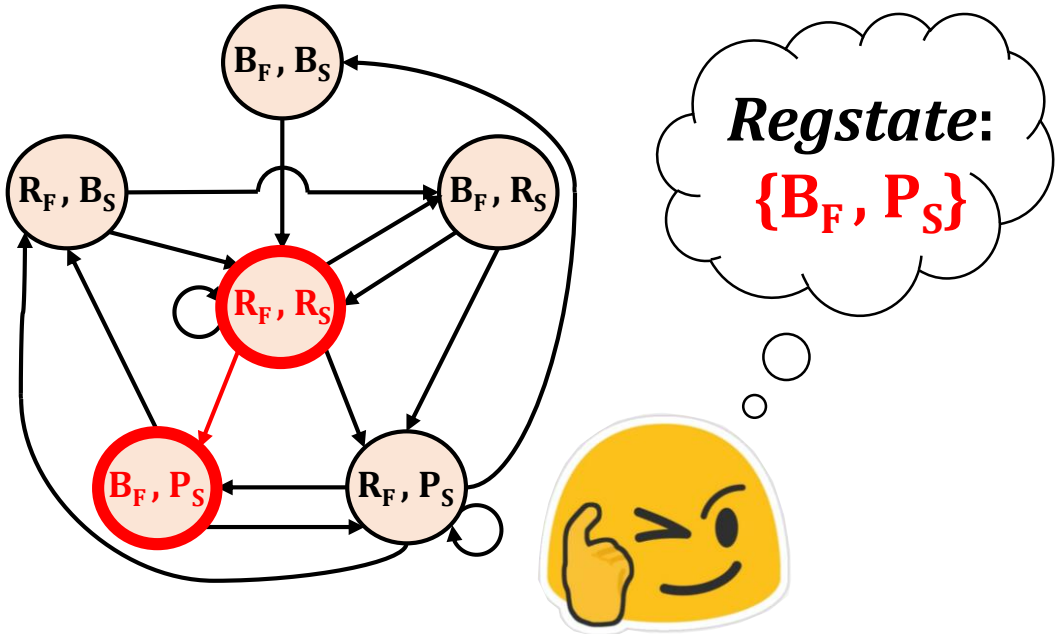
Finite State Machine (FSM)

New coverage definition for the RTL designs

Capturing FSM Exploration

Monitors registers to correctly capture the FSM exploration

Cycle	0	1	2
$input_F$	1	0	0
$input_S$	1	1	0



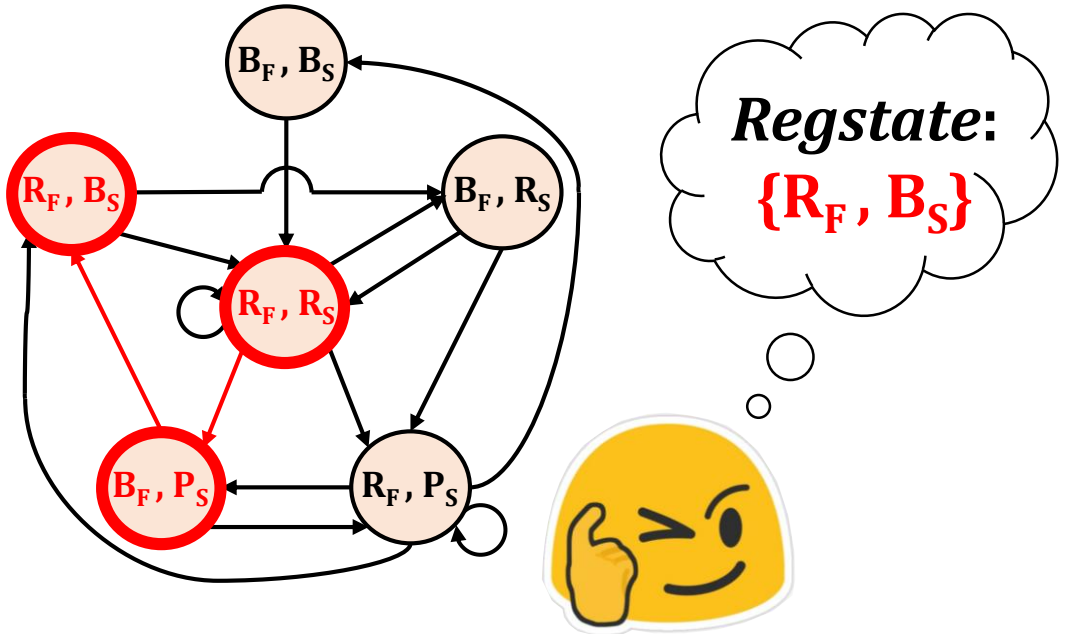
Finite State Machine (FSM)

New coverage definition for the RTL designs

Capturing FSM Exploration

Monitors registers to correctly capture the FSM exploration

Cycle	0	1	2
$input_F$	1	0	0
$input_S$	1	1	0



Finite State Machine (FSM)

New coverage definition for the RTL designs

Monitoring Control Register 

Improves efficiency by monitoring only control registers

Control register – Registers wired into MUX select signal

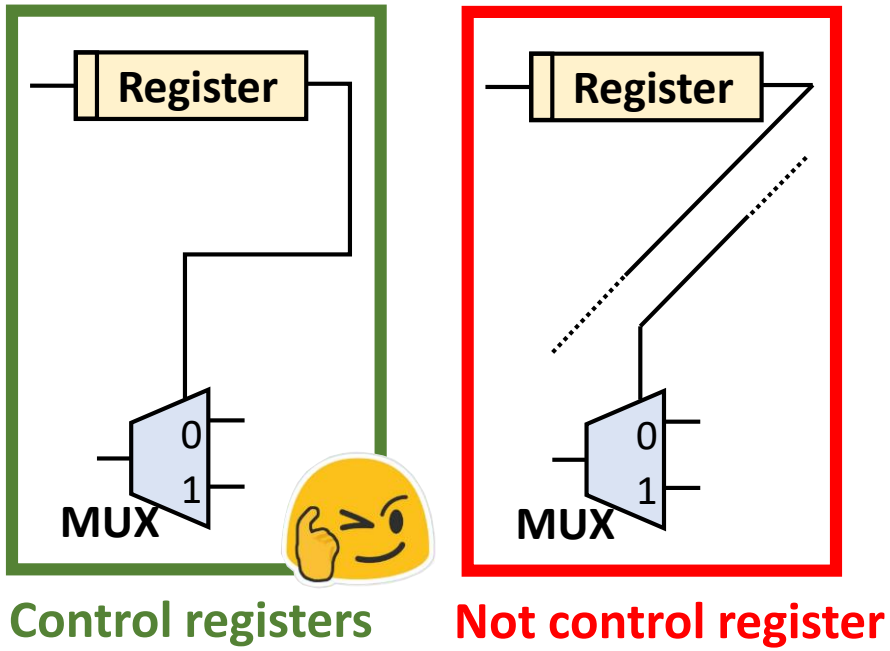
CPU	Number of registers
Mor1kx	258
Rocket	1,300
Boom	4,900

Monitoring Control Register

Improves efficiency by monitoring only control registers

Control register – Registers wired into MUX select signal

CPU	Number of registers
Mor1kx	258
Rocket	1,300
Boom	4,900



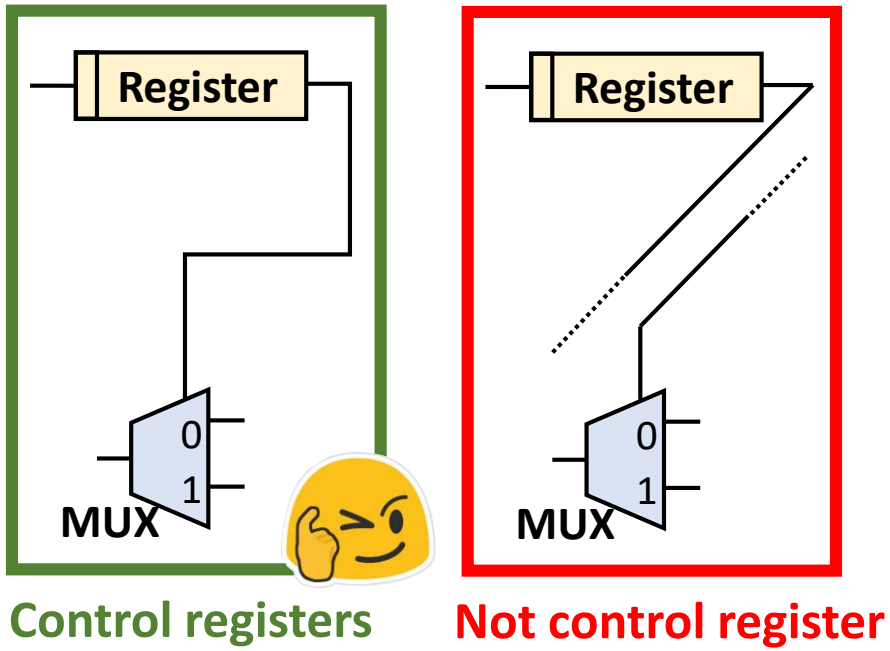
New coverage definition for the RTL designs

Monitoring Control Register

Improves efficiency by monitoring only control registers

Control register – Registers wired into MUX select signal

CPU	Number of registers
Mor1kx	258
Rocket	1,300
Boom	4,900



Control registers

Not control register

DiFuzzRTL static analyzer

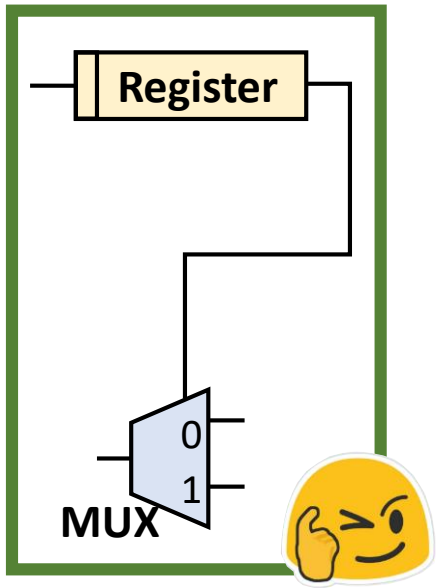
New coverage definition for the RTL designs

Monitoring Control Register

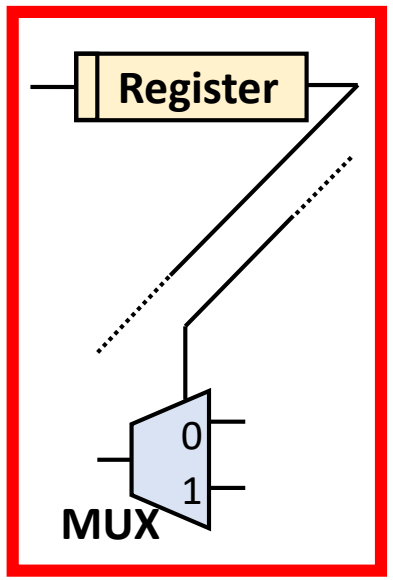
Improves efficiency by monitoring only control registers

Control register – Registers wired into MUX select signal

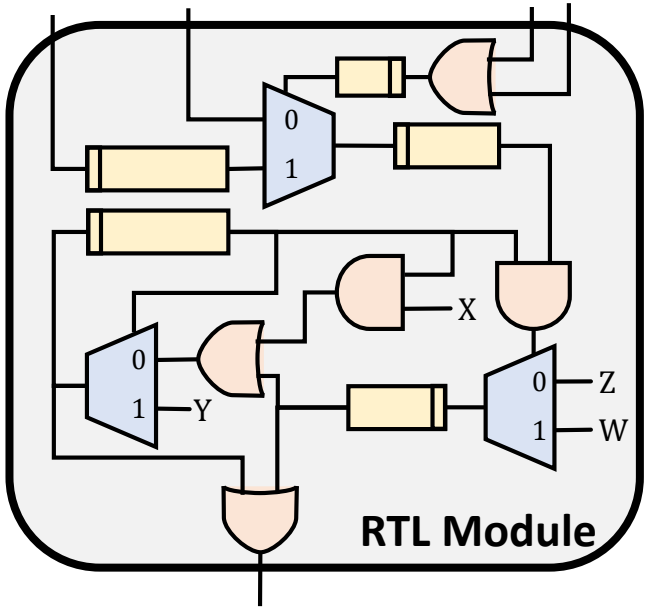
CPU	Number of registers
Mor1kx	258
Rocket	1,300
Boom	4,900



Control registers



Not control register



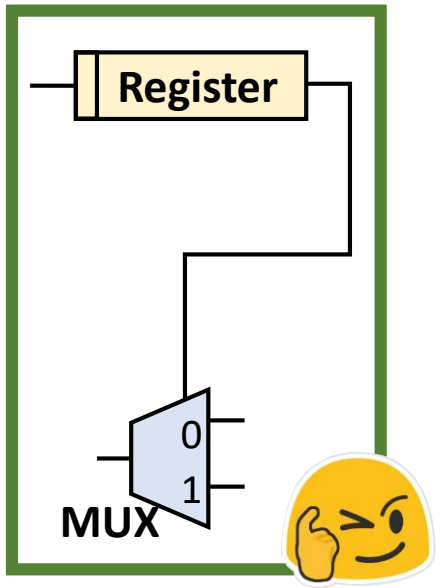
New coverage definition for the RTL designs

Monitoring Control Register

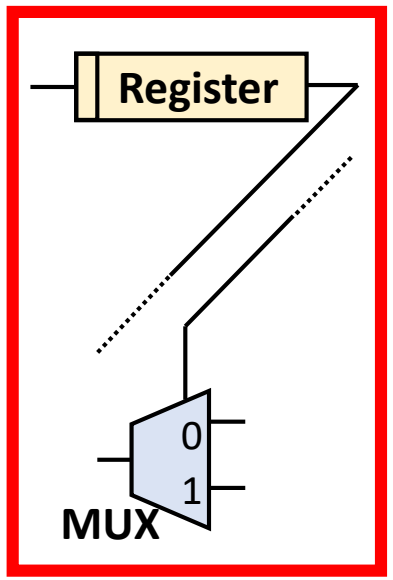
Improves efficiency by monitoring only control registers

Control register – Registers wired into MUX select signal

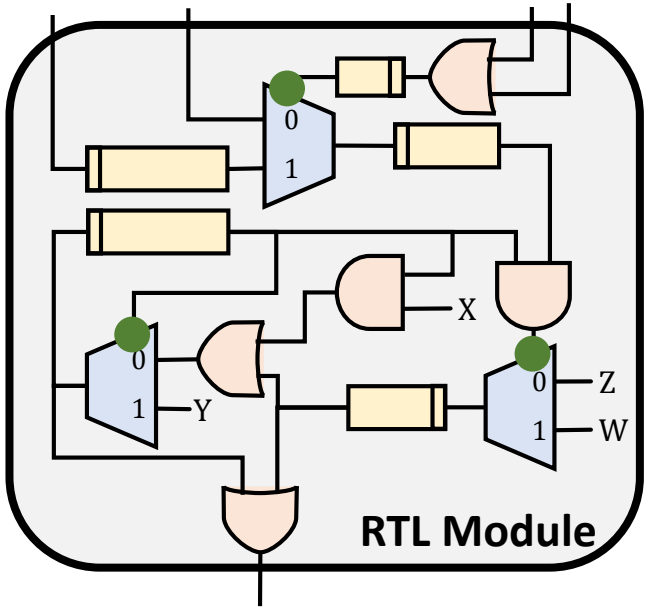
CPU	Number of registers
Mor1kx	258
Rocket	1,300
Boom	4,900



Control registers



Not control register



1. MUX select signal identification

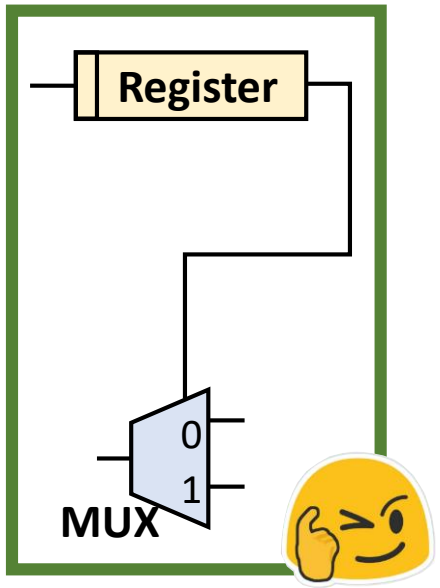
New coverage definition for the RTL designs

Monitoring Control Register

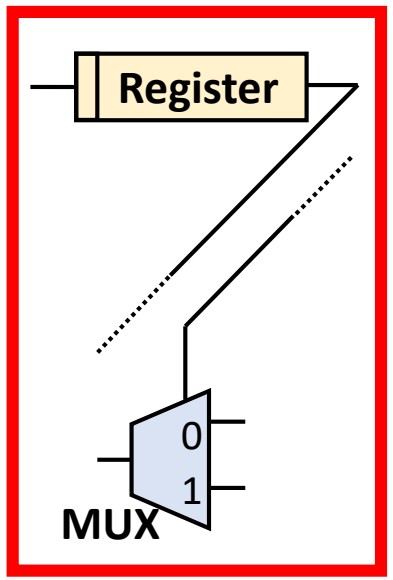
Improves efficiency by monitoring only control registers

Control register – Registers wired into MUX select signal

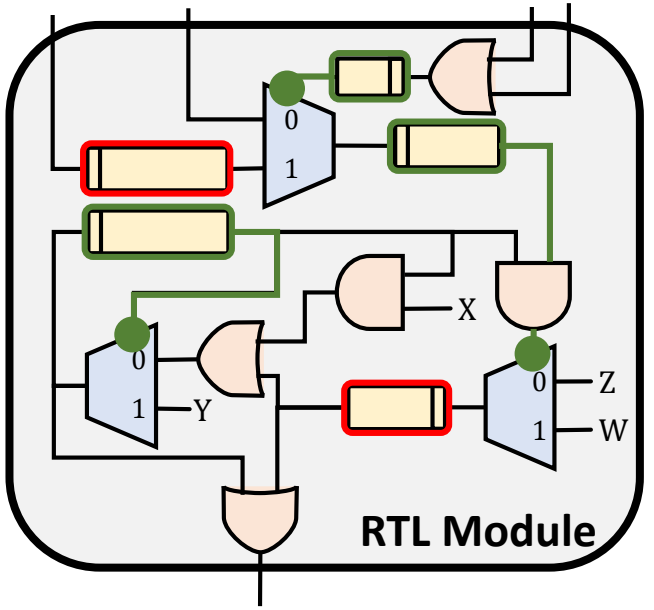
CPU	Number of registers
Mor1kx	258
Rocket	1,300
Boom	4,900



Control registers



Not control register



2. Control register identification

New coverage definition for the RTL designs

Monitoring Control Register 

Improves efficiency by monitoring only control registers

Control register – Registers wired into MUX select signal

CPU	Number of registers	Number of control registers
Mor1kx	258	90
Rocket	1,300	207
Boom	4,900	330

New coverage definition for the RTL designs

Monitoring Control Register

Improves efficiency by monitoring only control registers

Control register – Registers wired into MUX select signal

CPU	Number of registers	Number of control registers
Mor1kx		
Rocket	1,300	207
Boom	4,900	330

Automatically identifies the control registers

New coverage definition for the RTL designs

RTL-based Coverage Instrumentation 

Efficiently computes the number of new state explorations

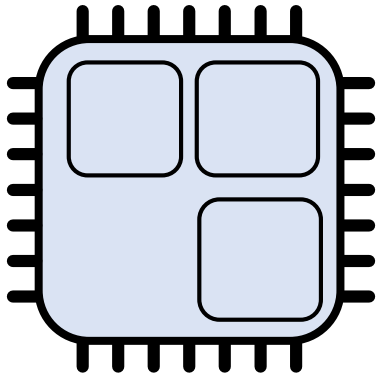
Module-based coverage map instrumentation

New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation

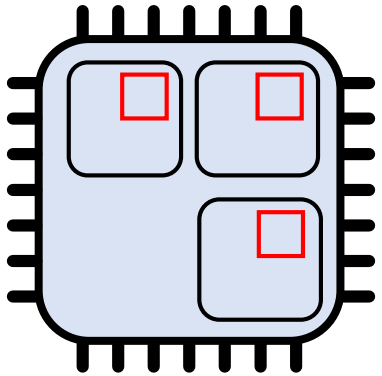


New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation

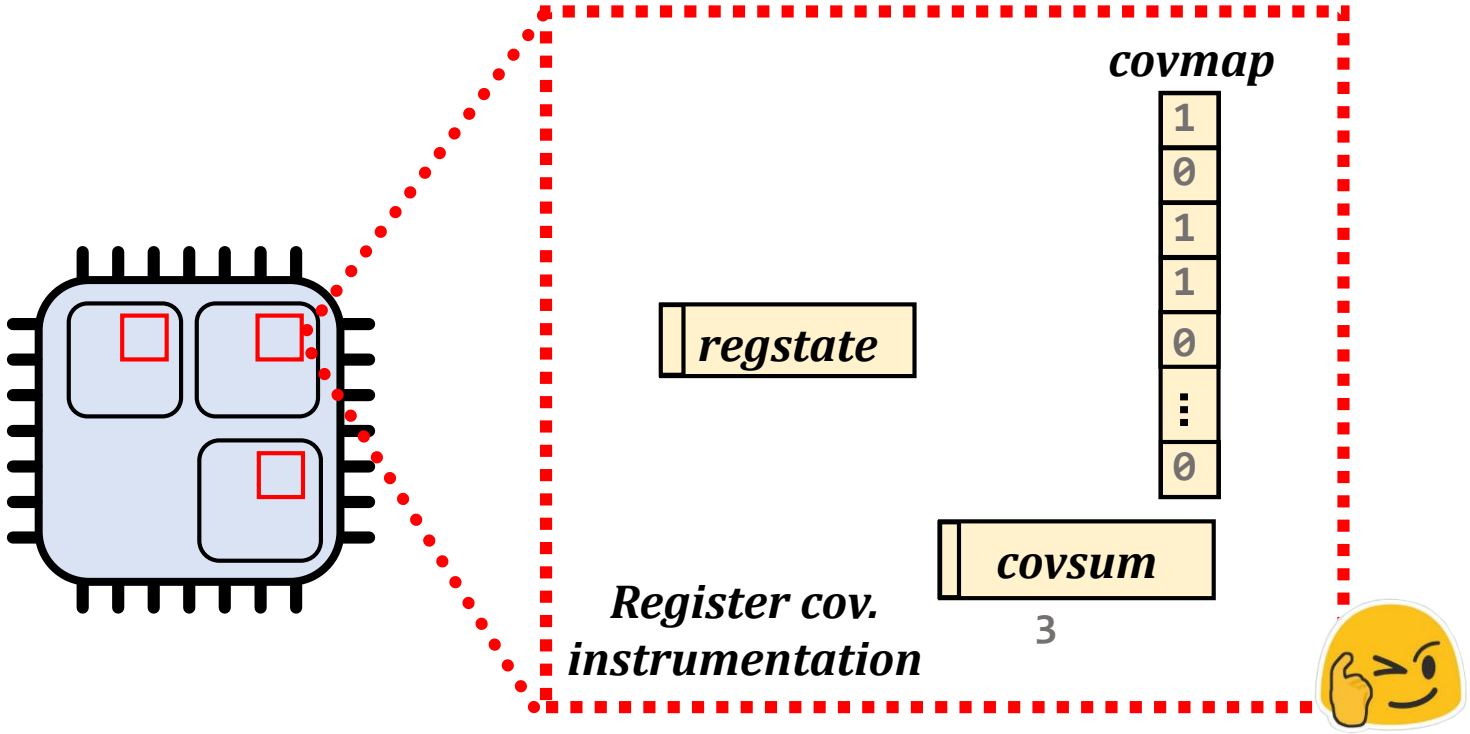


New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation



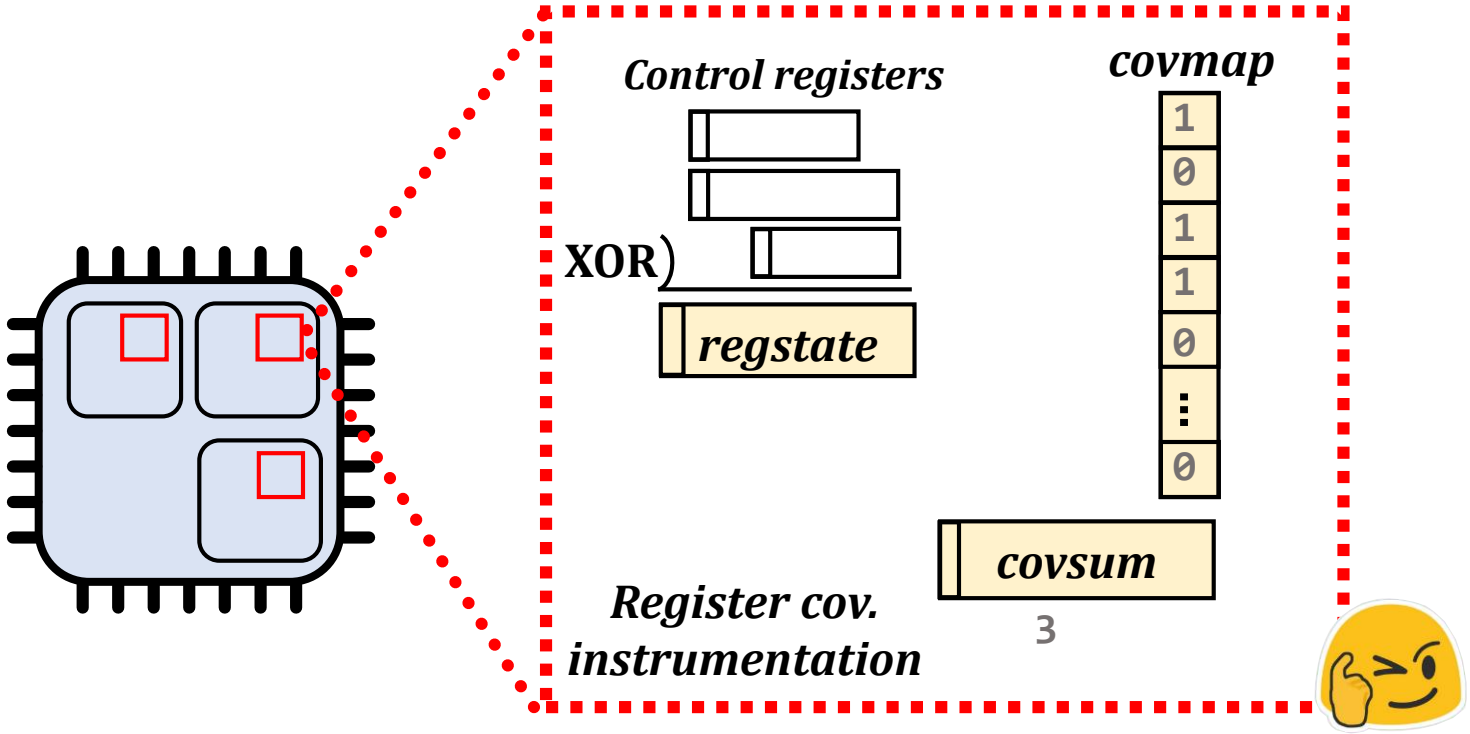
1. Efficient new state identification

New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation



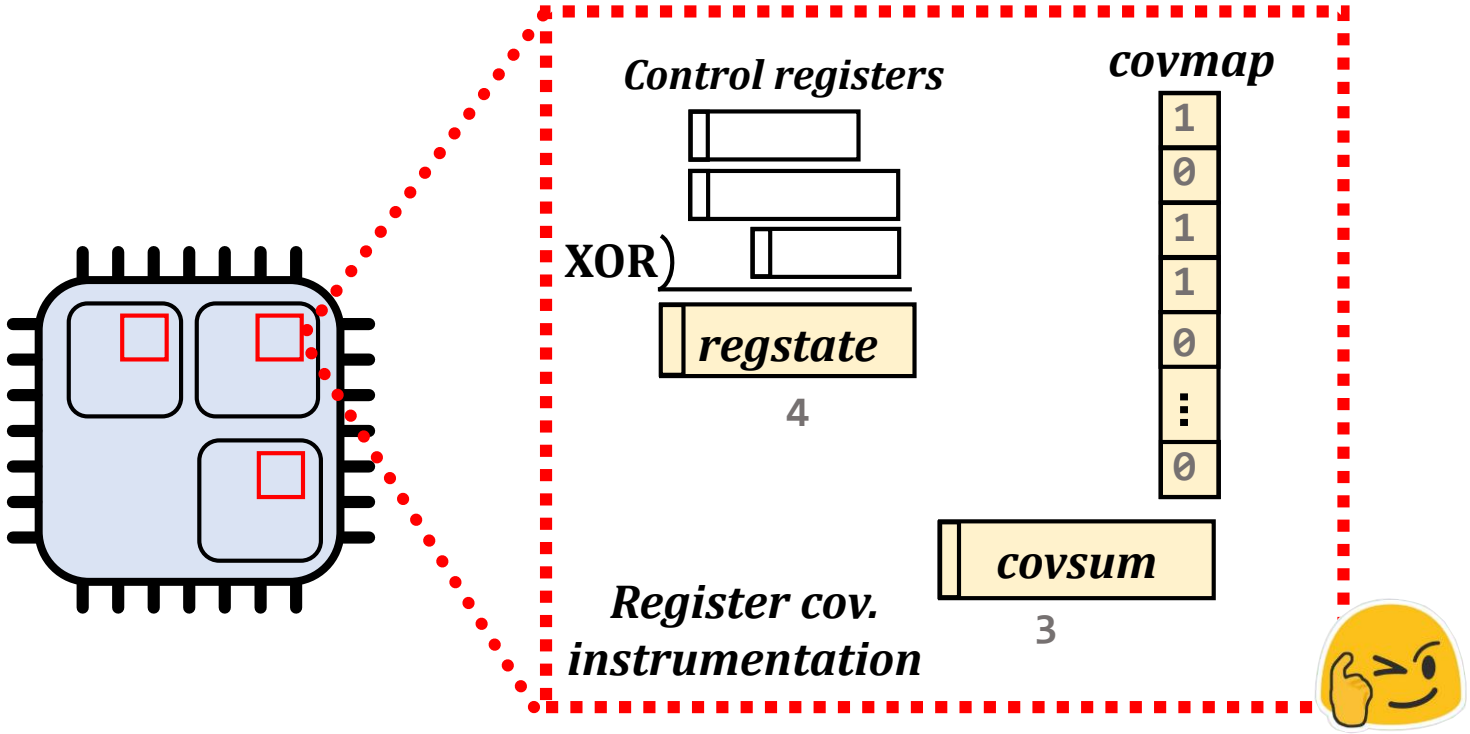
1. Efficient new state identification

New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation



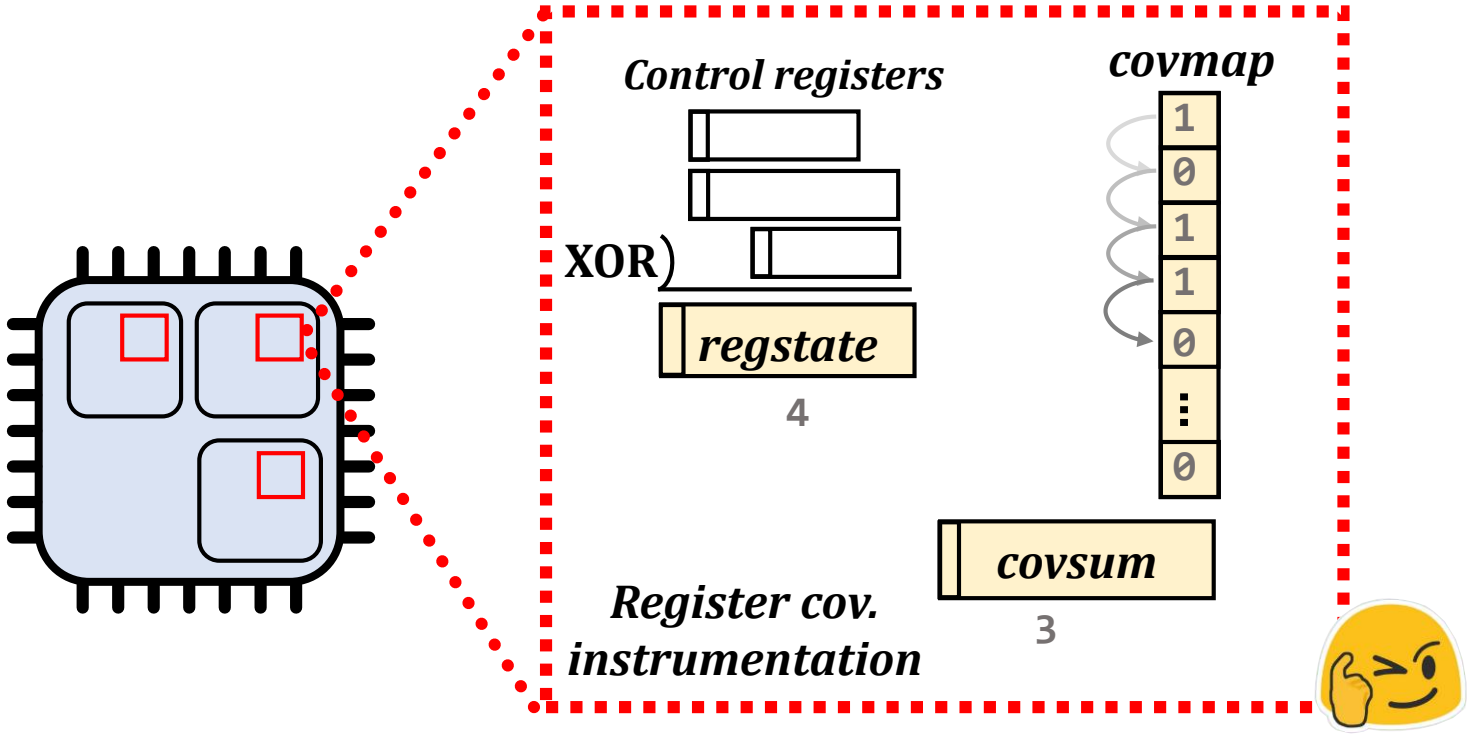
1. Efficient new state identification

New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation



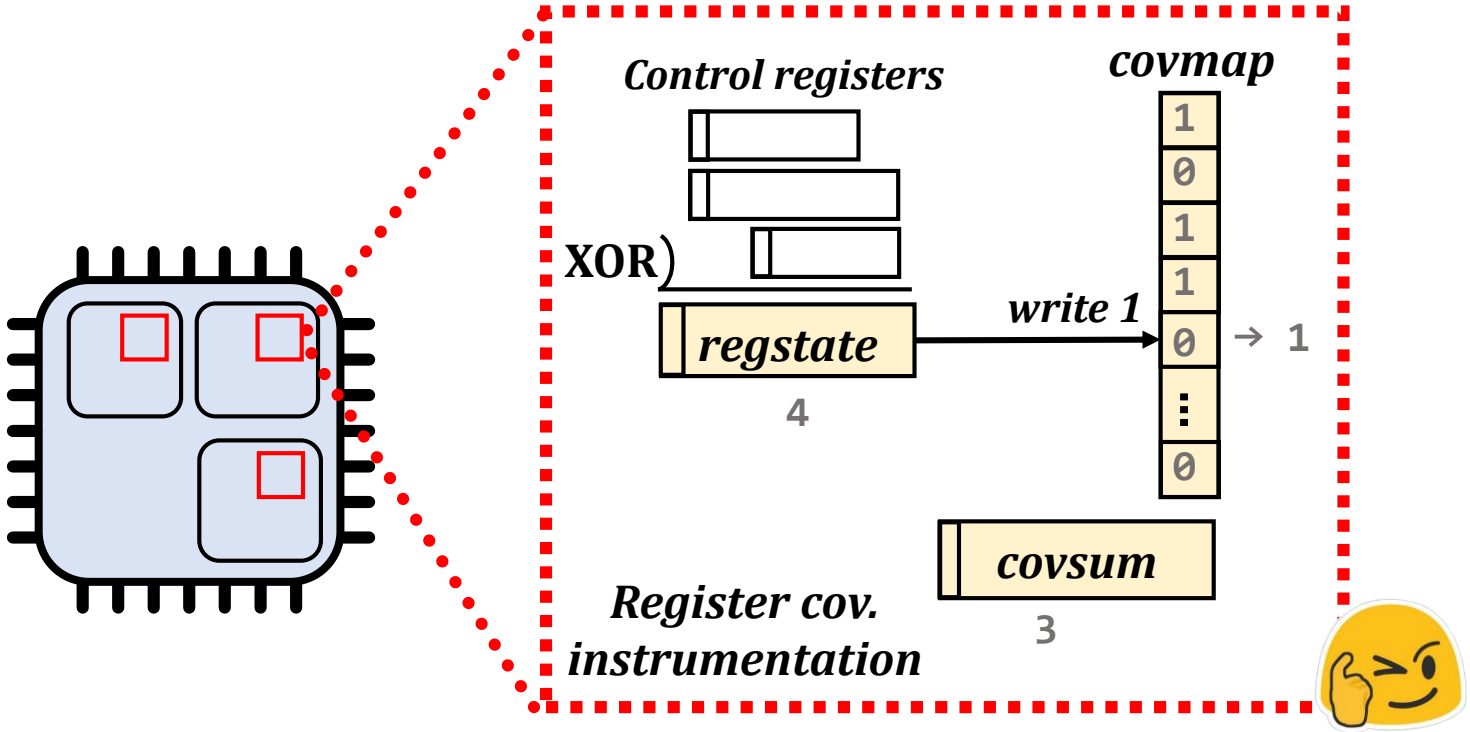
1. Efficient new state identification

New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation



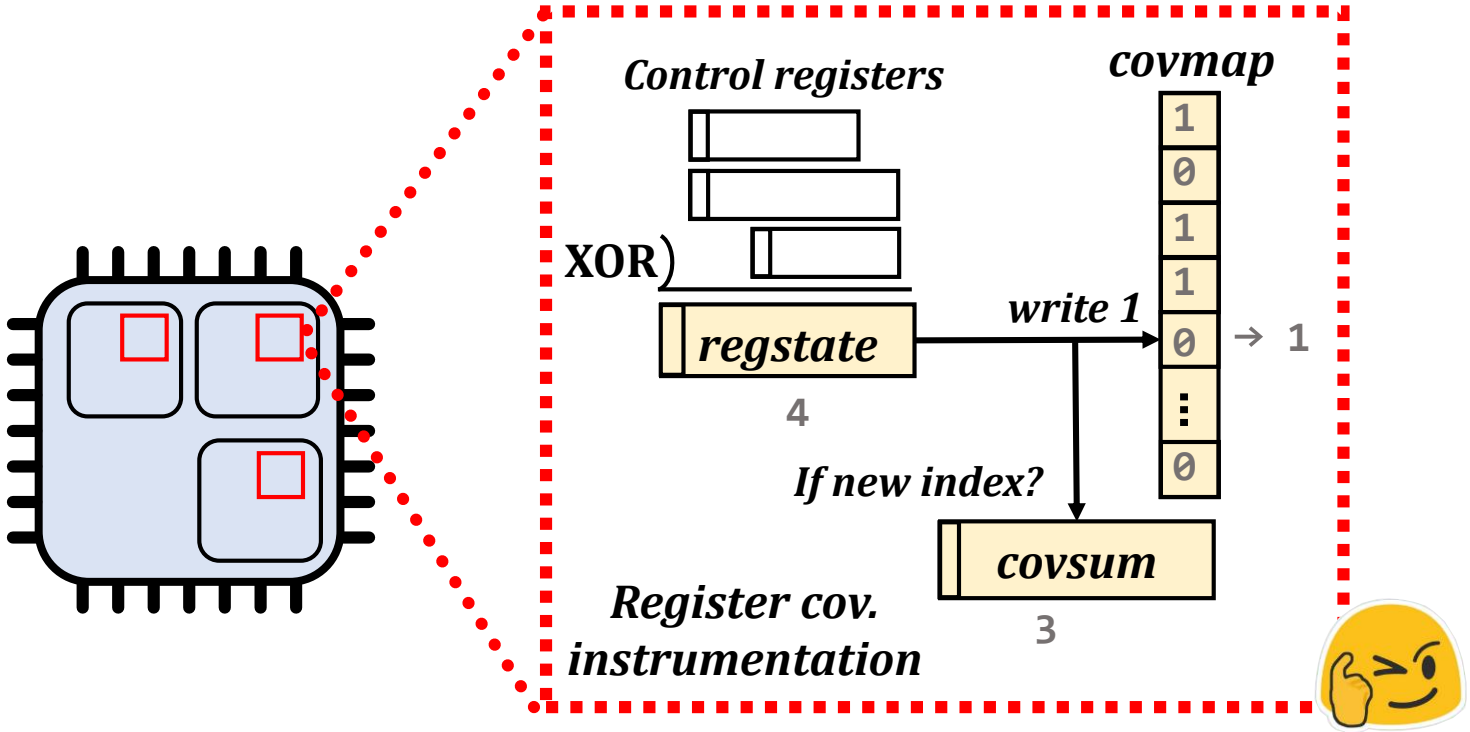
1. Efficient new state identification

New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation



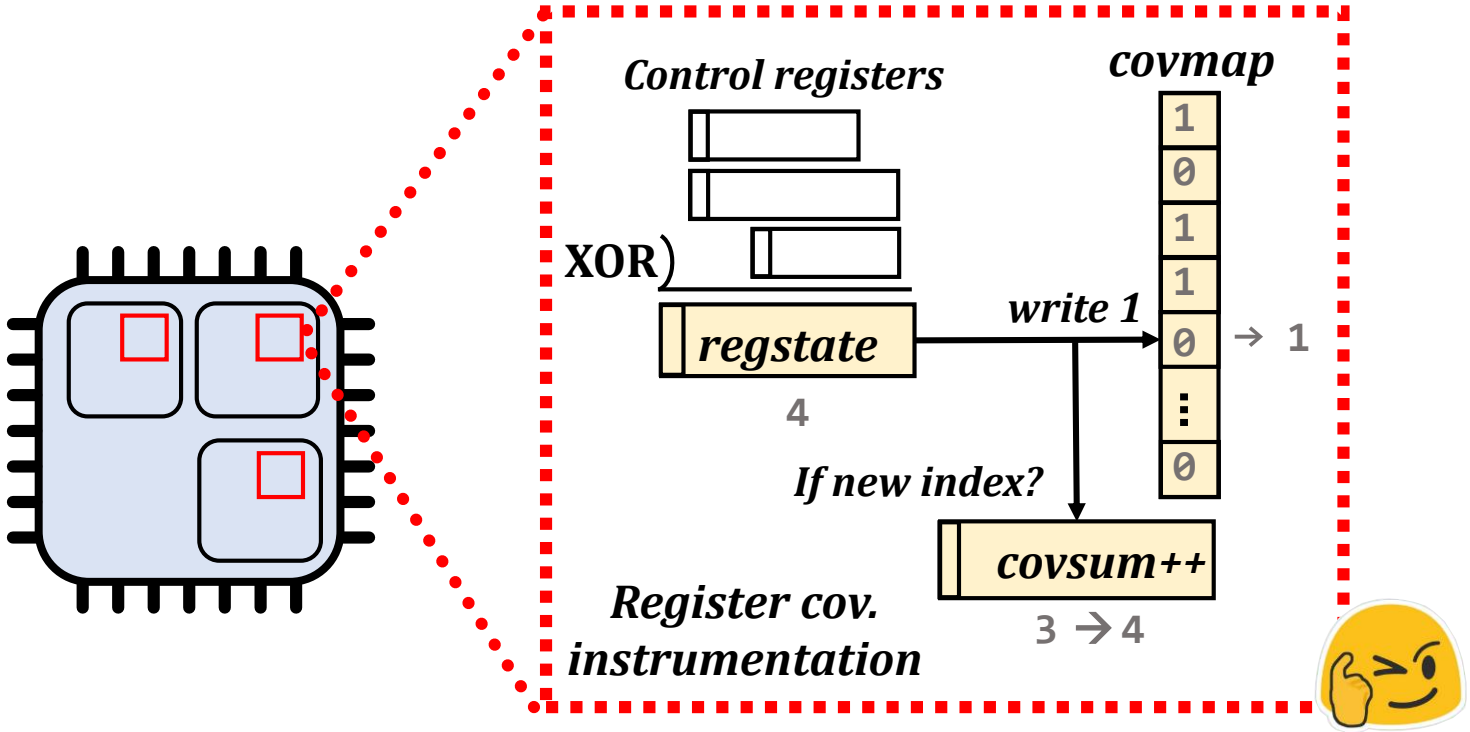
1. Efficient new state identification

New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation



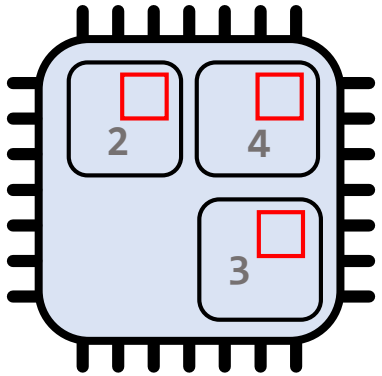
1. Efficient new state identification

New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation



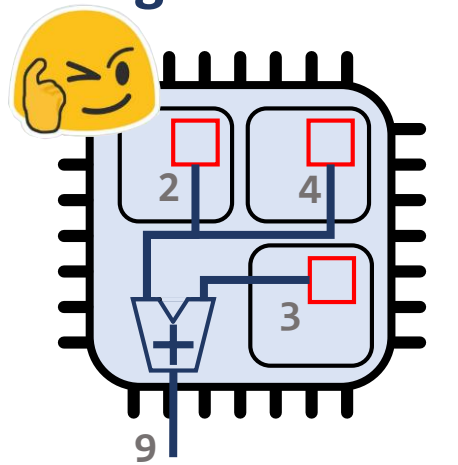
New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation

2. Scalable coverage computation



Sum of *covsum*

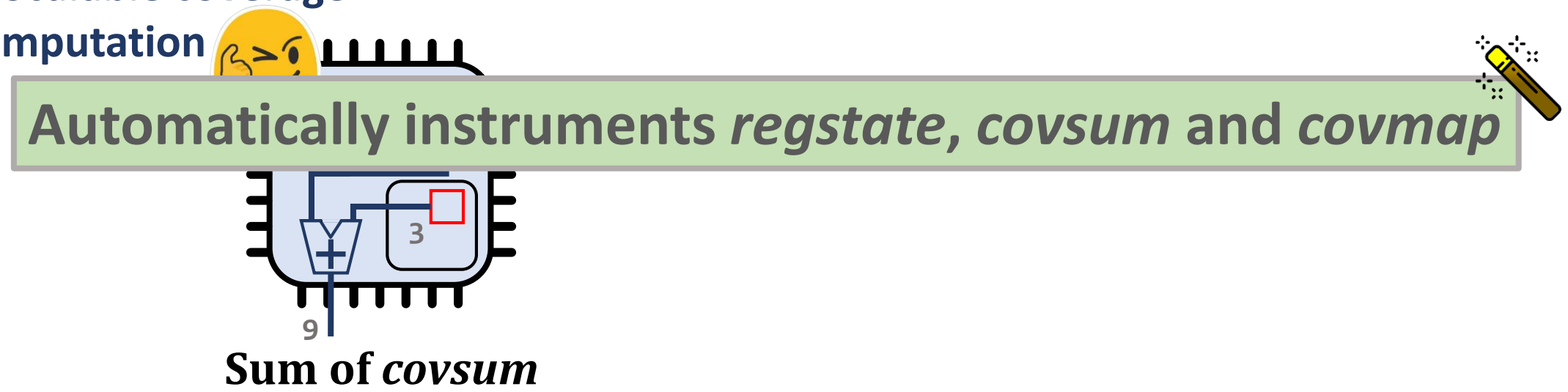
New coverage definition for the RTL designs

RTL-based Coverage Instrumentation

Efficiently computes the number of new state explorations

Module-based coverage map instrumentation

2. Scalable coverage computation



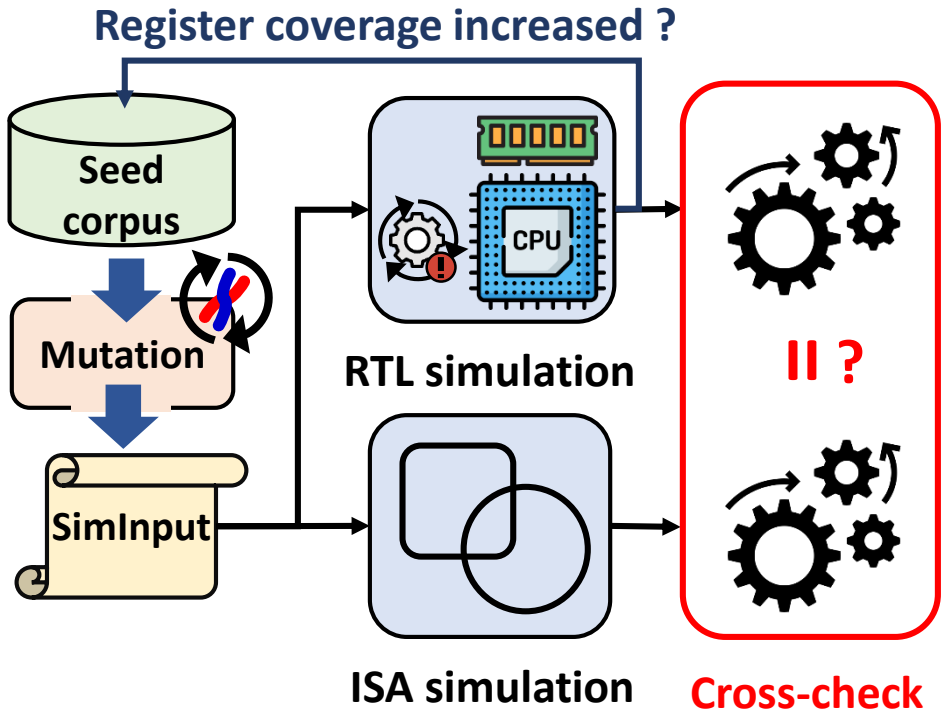
New coverage definition for the RTL designs

DiFuzzRTL

Accurate, **Efficient**, and **Automatic** fuzzer to find CPU bugs

Coverage-guided input generation

Automatic testing and bug detection



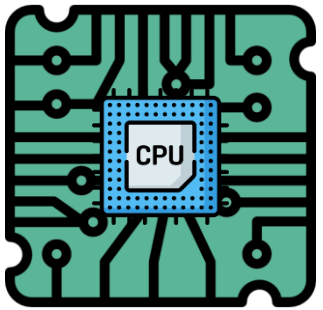
Implementation & Evaluation Setup

- Prototype with three CPU RTL designs:
  Mor1kx (OpenRISC),
  Rocket, and **Boom (RISC-V)**

Implementation & Evaluation Setup

- Prototype with three CPU RTL designs:
  Mor1kx (OpenRISC),
  Rocket, and **Boom (RISC-V)**

- RTL testing environments:
  Software simulation, and **FPGA prototyping**



What DifuzzRTL Found?

Project	ISA	Bug ID	Description	Confirmed	Fixed
Mork1x	OpenRISC	CVE-2020-13455	Reservation is not cancelled when there is snooping hit between lwa and swa	✓	pending
		CVE-2020-13454	Jump to link register does not assert illegal instruction exception	✓	pending
		CVE-2020-13453	Misaligned swa raise exception when reservation is not set	✓	pending
		Issue #114	`l.fl1`, `l.ff1` instruction decoding bug	✓	✓
		Issue #99	`eear` register not saving instruction virtual address when illegal instruction exception	✓	✓
Rocket chip	RISCV	Issue #2345	Instruction retired count not increased when ebreak	✓	pending
Boom	RISCV	CVE-2020-13251	Source field in `ProbeAckData` does not match the sink field of `ProbeRequest`	✓	✓
		Issue #458	Floating point instruction which has invalid `rm` field does not raise exception	✓	✓
		Issue #454	`FS` bits in `mstatus` register is set after `fle.d` instruction	✓	pending
		Issue #492	When `frm` is set `DYN`, floating point instruction with `DYN` `rm` field should raise exception	✓	✓
		Issue #493	Rounding mode in `fsqrt` instruction does not work	✓	✓
		Issue #503	`invalid operation` flag is not set after invalid `fdiv` instruction	✓	✓
		CVE-2020-29561	Misaligned `lr` instruction on a cached line set the reservation	✓	✓
Spike	RISCV	CVE-2020-13456	Misaligned `lr.d` should not set load reservation	✓	✓
		Issue #2390	Reading `dpc` register should raise exception in machine mode	✓	✓
		Issue #426	Faulting virtual address should not be written to `mtval` when `ebreak`	✓	✓

- Found **16 new CPU bugs**
  - **6 of those were assigned with CVE numbers**.
- Showed the effectiveness of DiFuzzRTL
  - Case study with ***Issue #492 (invalid rm bug)*** and ***CVE-2020-29561 (misaligned lr bug)***

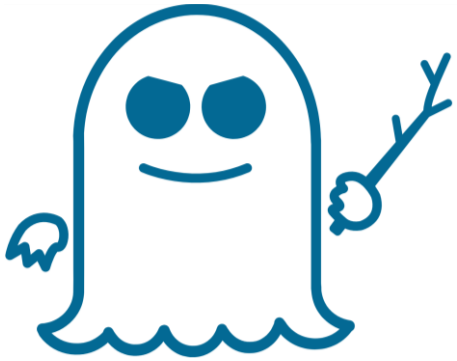
Bug ID	Elapsed time (h)		
	riscv-torture	`mux-cov`	`reg-cov`
Issue #458	118	✗	20.3
Issue #504	✗	✗	31.7

✗Not able to reproduce bug

Future Use Cases

Future Use Cases

- Detecting micro-architectural side channels, e.g., Spectre, Meltdown

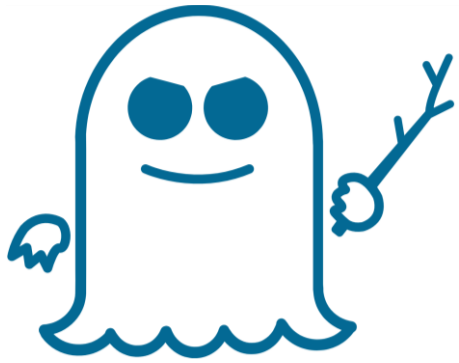


Spectre

Meltdown

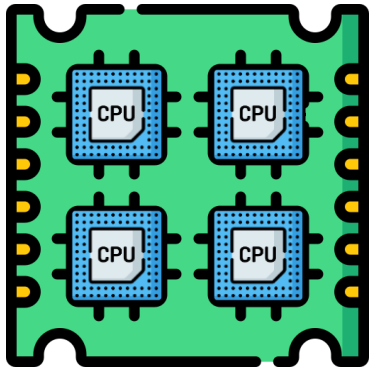
Future Use Cases

- Detecting micro-architectural side channels, e.g., Spectre, Meltdown
- Fuzzing an entire SoC with DiFuzzRTL, e.g., memory consistency bug



Spectre

Meltdown

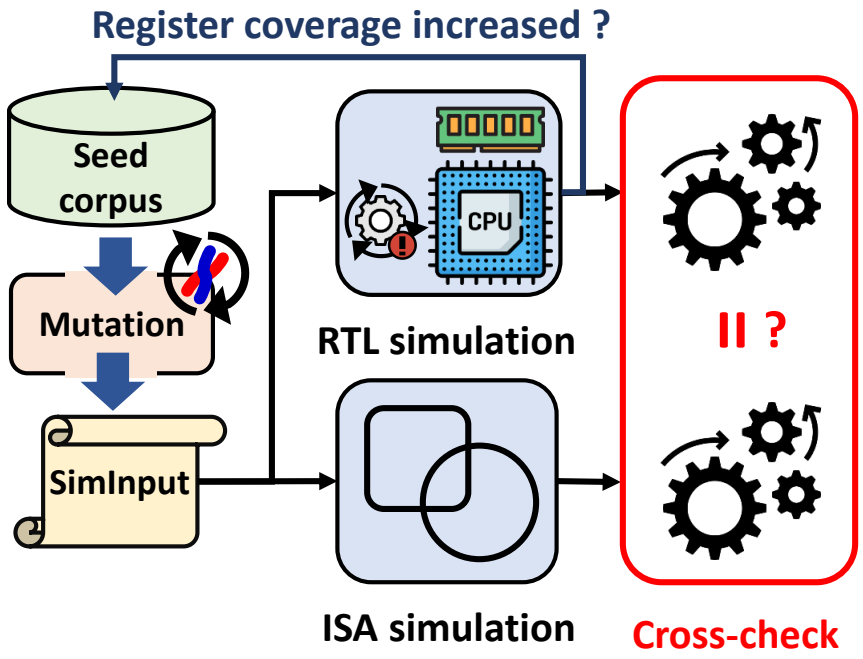


Multicore SoC

Conclusion

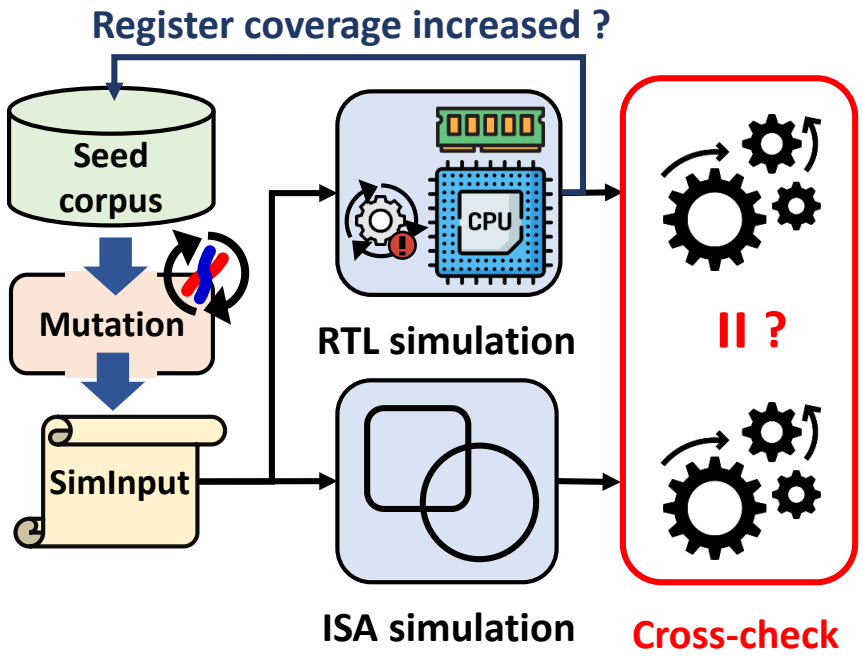
Conclusion

- DiFuzzRTL, an accurate, efficient, and automatic fuzzer for CPU RTL designs



Conclusion

- DiFuzzRTL, an accurate, efficient, and automatic fuzzer for CPU RTL designs
- We found several real-world bugs with DiFuzzRTL



CPU	Bug ID
Mor1kx	CVE-2020-13455, 2020-13453, 2020-13454 Issue 114, 99
Rocket	Issue 2345
Boom	CVE 2020-13251, 2020-29561 Issue 458, 454, 492, 493, 503
Spike	CVE-2020-13456 Issue 426, 2390

Thank you