TheHuzz: Instruction Fuzzing of Processors Using Golden-Reference Models for Finding Software-Exploitable Vulnerabilities

Rahul Kande[†], Addison Crump[†], Garrett Persyn[†], Patrick Jauernig[*],
Ahmad-Reza Sadeghi[*], Aakash Tyagi[†], Jeyavijayan Rajendran[†]

[†]Texas A&M University, College Station, USA,
[*]Technische Universität Darmstadt, Germany

Motivation

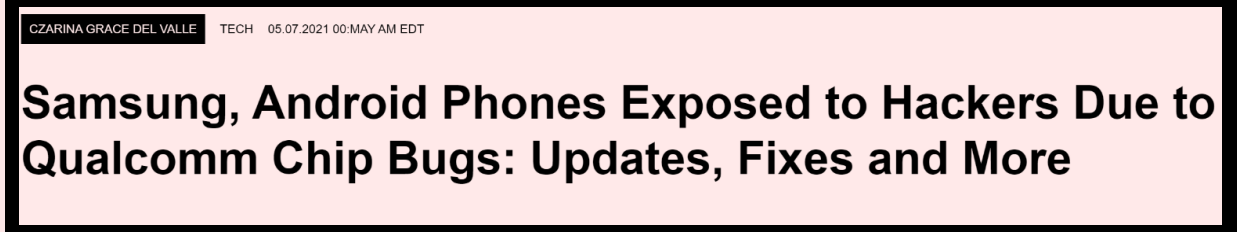

CYBERSECURITY

iOS 14.7.1: Apple Issues Urgent iPhone Update With Important Security Fixes

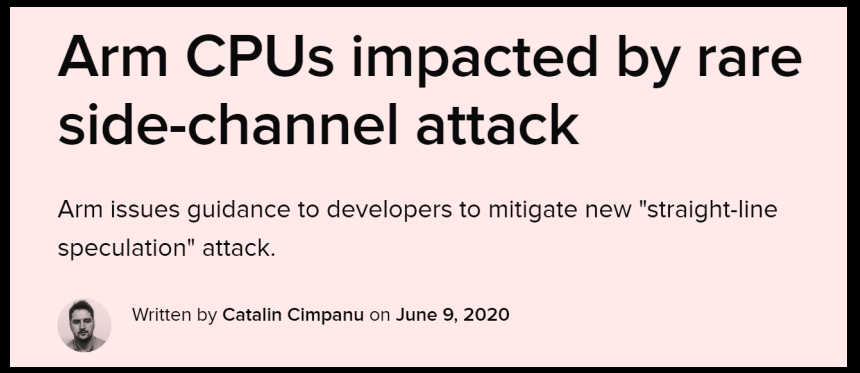
Kate O'Flaherty Senior Contributor ⓘ
Straight Talking Cyber

Follow

Jul 26, 2021, 02:15pm EDT


CZARINA GRACE DEL VALLE TECH 05.07.2021 00:MAY AM EDT

Samsung, Android Phones Exposed to Hackers Due to Qualcomm Chip Bugs: Updates, Fixes and More


Arm CPUs impacted by rare side-channel attack

Arm issues guidance to developers to mitigate new "straight-line speculation" attack.

Written by **Catalin Cimpanu** on **June 9, 2020**

https://www.zdnet.com/article/arm-cpus-impacted-by-rare-side-channel-attack/
https://www.forbes.com/sites/kateoflahertyuk/2021/07/26/ios-1471-apple-issues-urgent-iphone-update-with-important-security-fixes/?sh=9de0071df186
https://www.itechpost.com/articles/105576/20210507/samsung-android-phones-exposed-hackers-due-qualcomm-chip-bugs-updates.htm
https://www.techrepublic.com/article/63-of-organizations-face-security-breaches-due-to-hardware-vulnerabilities/

Motivation

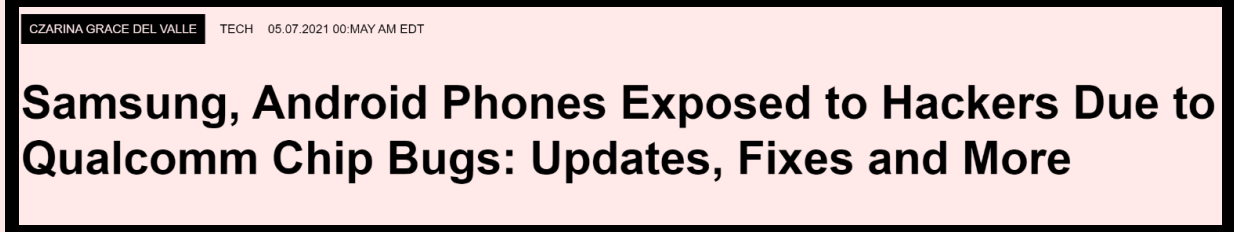

CYBERSECURITY

iOS 14.7.1: Apple Issues Urgent iPhone Update With Important Security Fixes

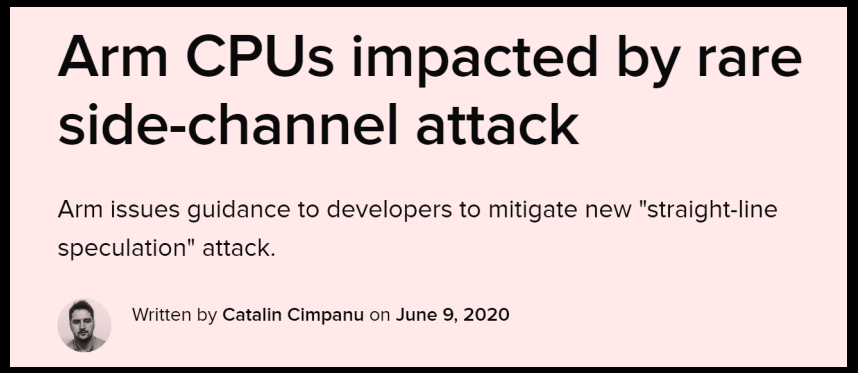
Kate O'Flaherty Senior Contributor
Straight Talking Cyber

Follow

Jul 26, 2021, 02:15pm EDT


CZARINA GRACE DEL VALLE TECH 05.07.2021 00:MAY AM EDT

Samsung, Android Phones Exposed to Hackers Due to Qualcomm Chip Bugs: Updates, Fixes and More


Arm CPUs impacted by rare side-channel attack

Arm issues guidance to developers to mitigate new "straight-line speculation" attack.

Written by **Catalin Cimpanu** on **June 9, 2020**


63% of organizations face security breaches due to hardware vulnerabilities

by **Macy Bayern** in **Security**
on December 11, 2019, 6:00 AM PST

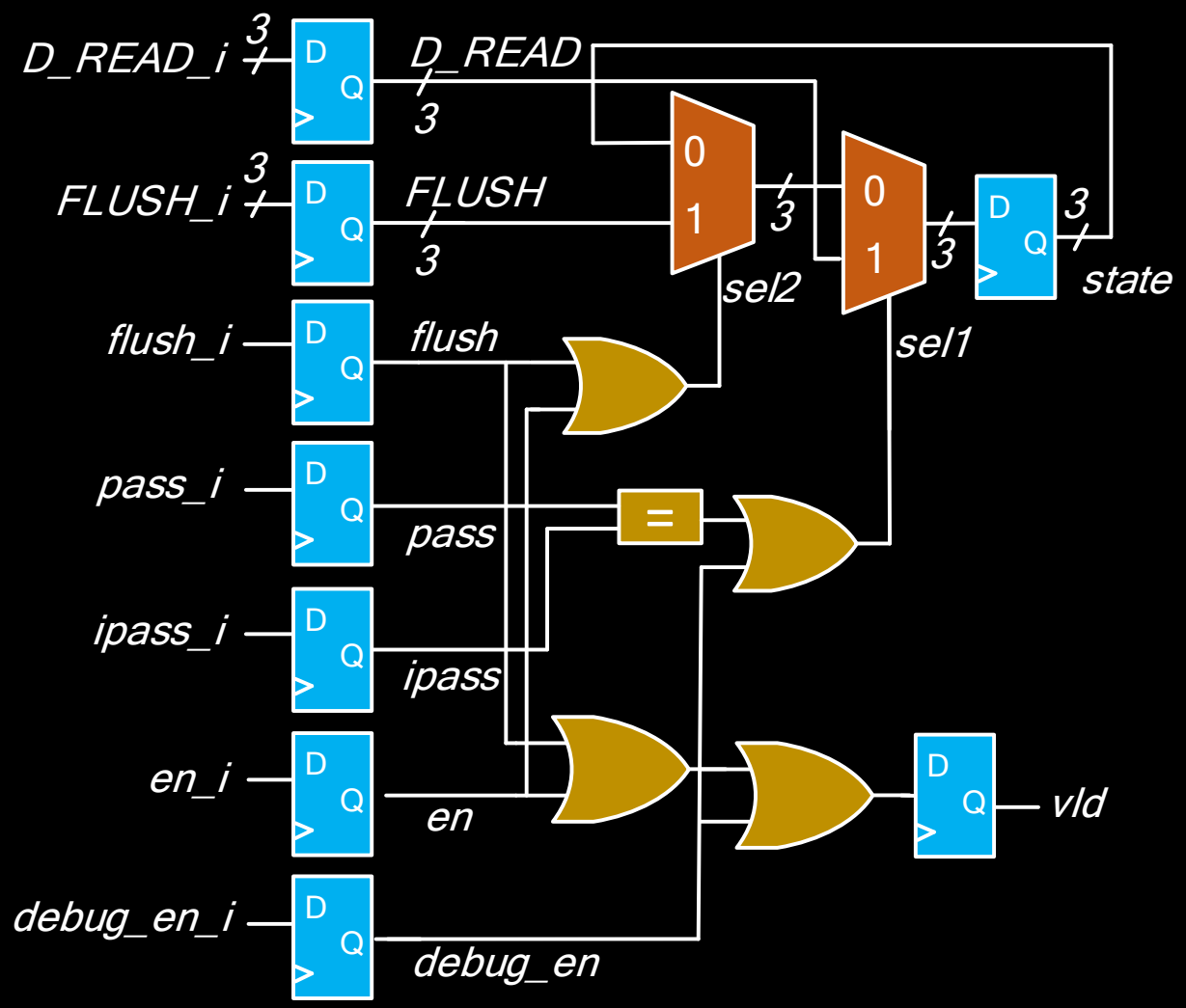
- **113** new H/W CWEs since 2020 by MITRE

https://www.zdnet.com/article/arm-cpus-impacted-by-rare-side-channel-attack/
https://www.forbes.com/sites/kateoflahertyuk/2021/07/26/ios-1471-apple-issues-urgent-iphone-update-with-important-security-fixes/?sh=9de0071df186
https://www.itechpost.com/articles/105576/20210507/samsung-android-phones-exposed-hackers-due-qualcomm-chip-bugs-updates.htm
https://www.techrepublic.com/article/63-of-organizations-face-security-breaches-due-to-hardware-vulnerabilities/

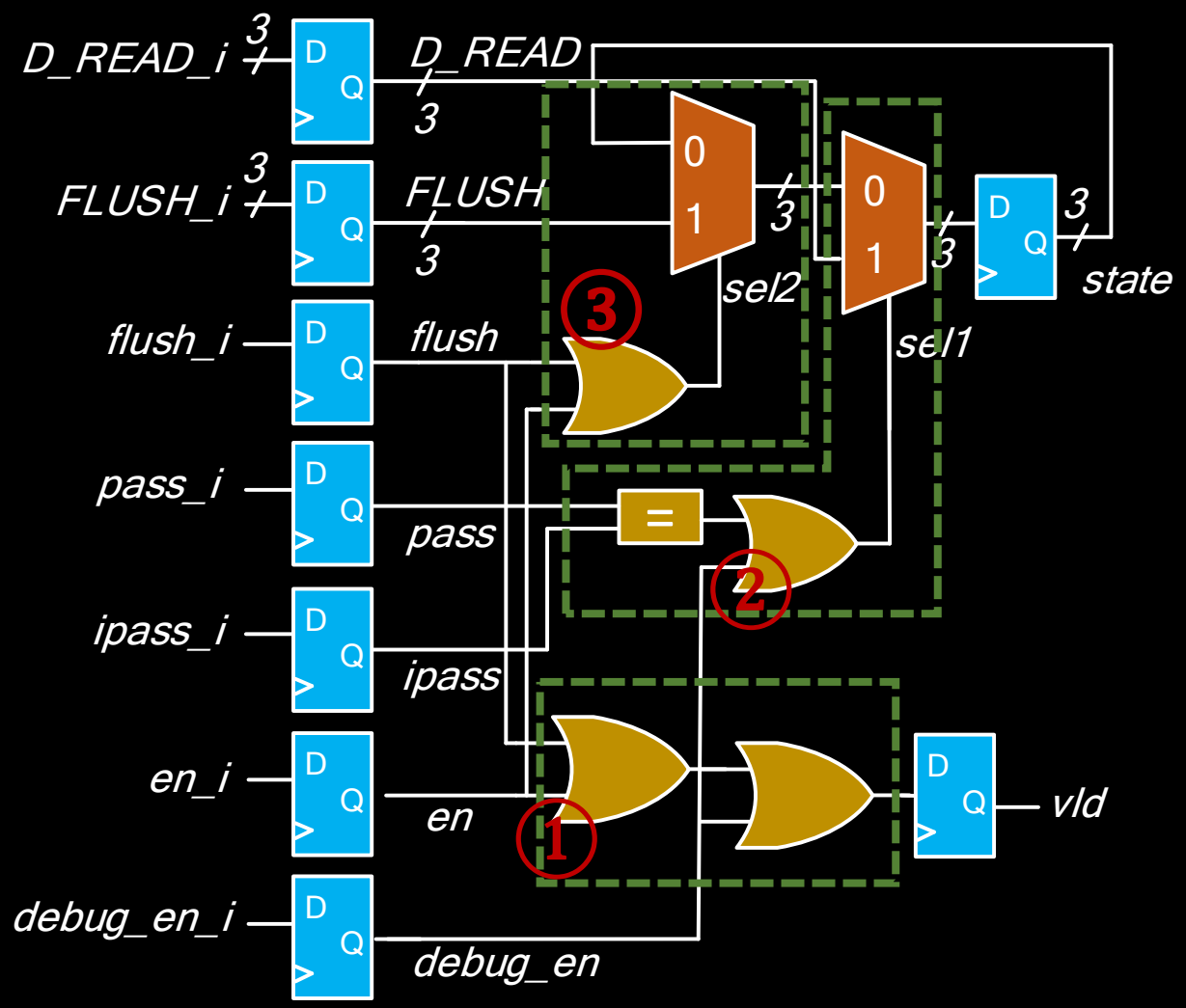
Motivation

Technique	Fast	Coverage	Scalable	Automated
Manual inspection	✗	✗	✗	✗
Formal verification[1]	✗	✓	✗	✗
Regression testing	✗	✗	✓	✓
Hardware Fuzzing	✓	✓	✓	✓

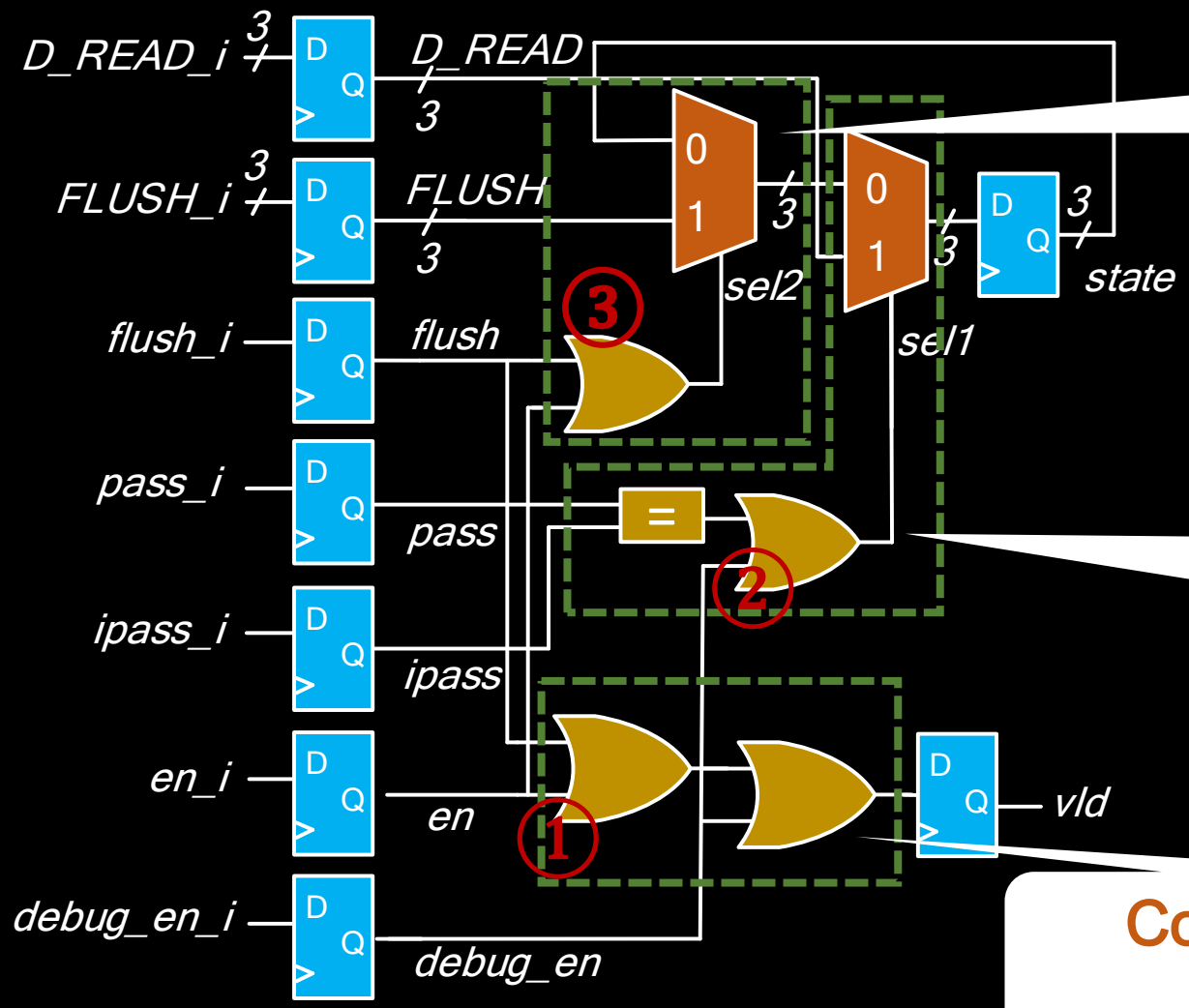
Case Study: Ariane Cache Controller



Case Study: Ariane Cache Controller



Case Study: Ariane Cache Controller

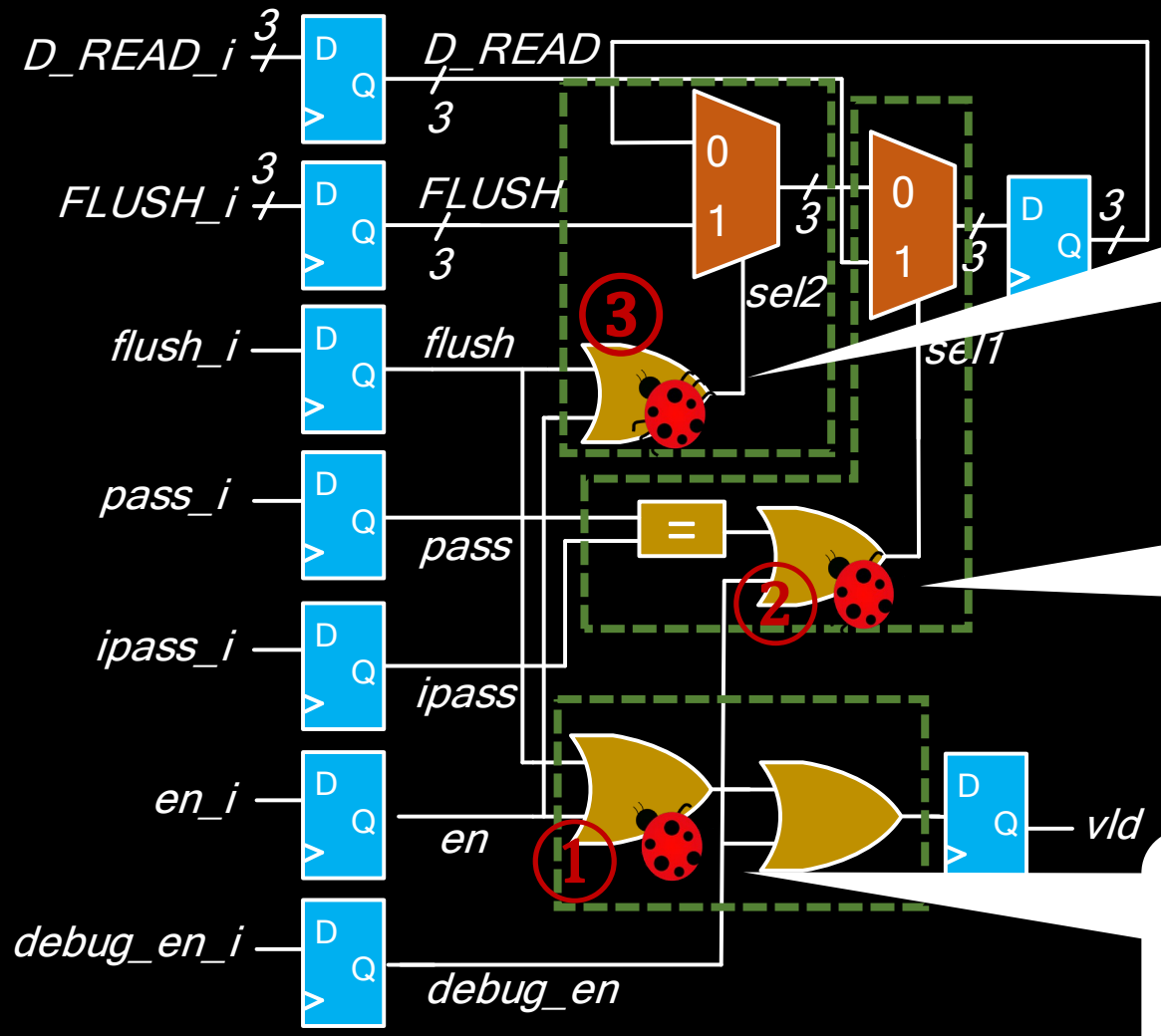


MUX ③ to flush the cache

MUX ② to protect debug reads using password

Combinational logic ① to generate *valid* signal

Case Study: Ariane Cache Controller

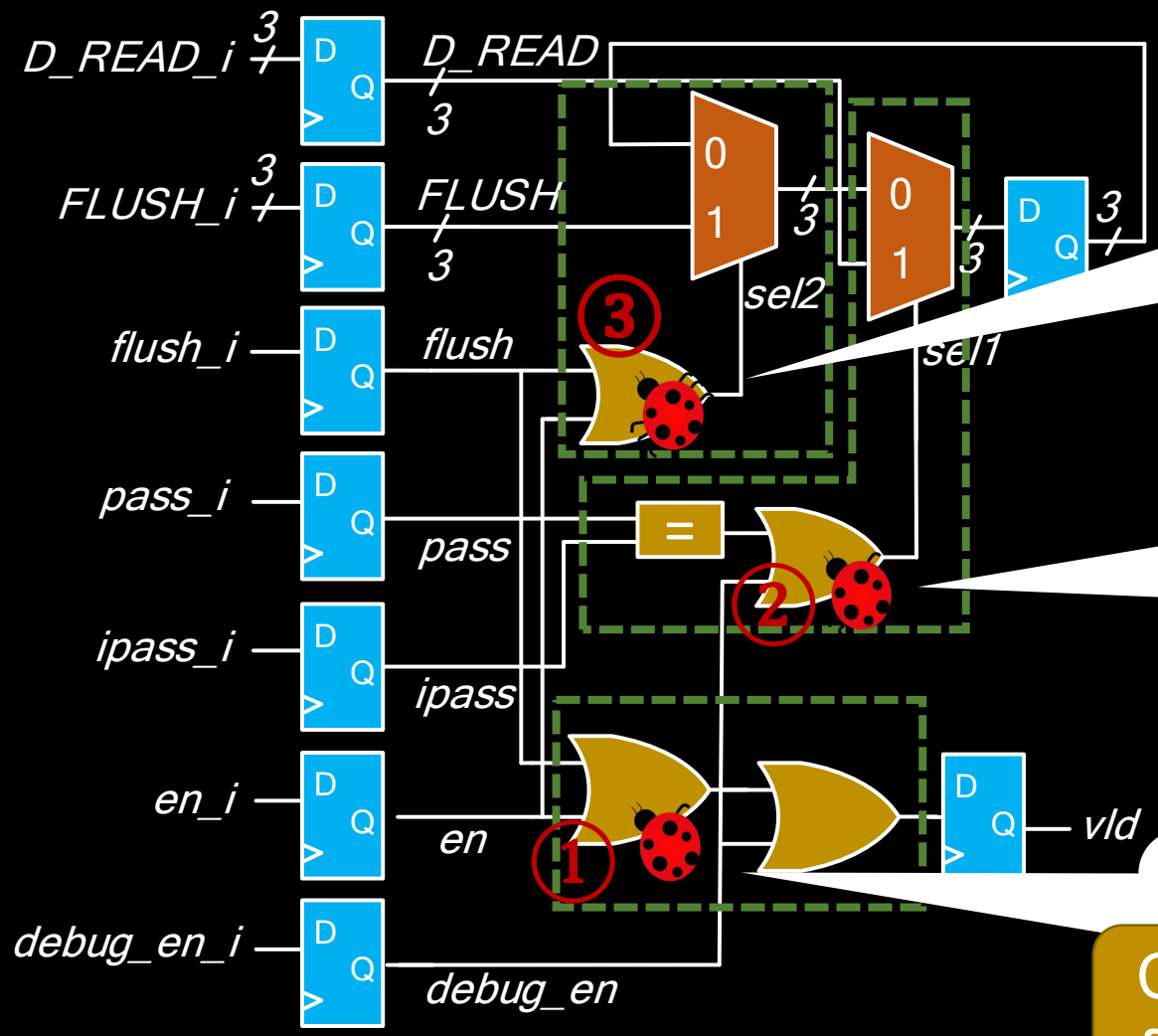


Bug in **MUX** ③ :
Cache can be flushed even when it is not enabled

Bug in **MUX** ②:
Debug password protection is bypassed

Bug in **Combinational logic** ①:
Cache is activated even when the *en_i* signal is not enabled

Case Study: Ariane Cache Controller



Bug in **MUX ③** :
Cache can be flushed even

Cover different combinations of values for **input signals of select logic**

Bug in **MUX ②**:

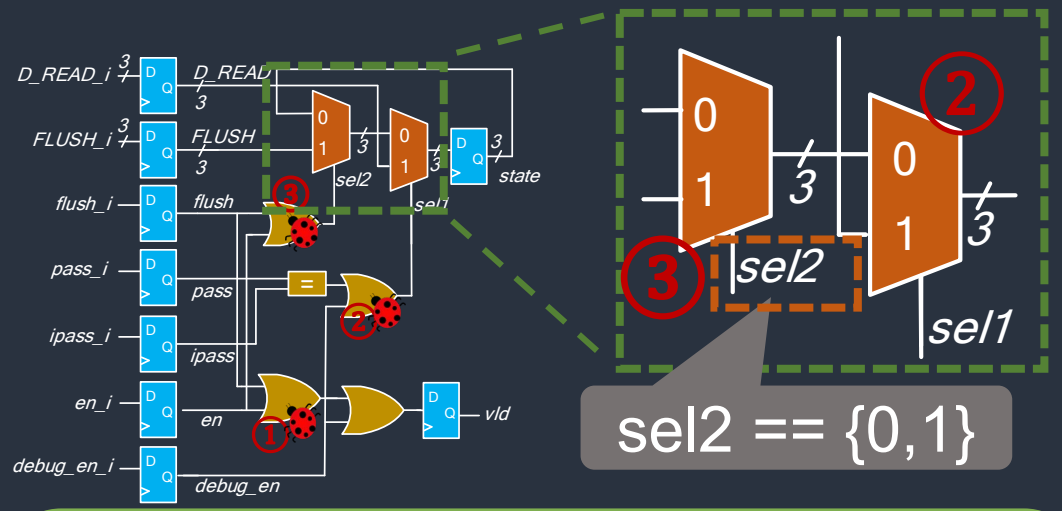
Cover different combinations of values for **input signals of select logic**

Bug in **Combinational logic ①**:

Cover different combinations of values for **input signals of combinational logic**

Existing Hardware Fuzzers

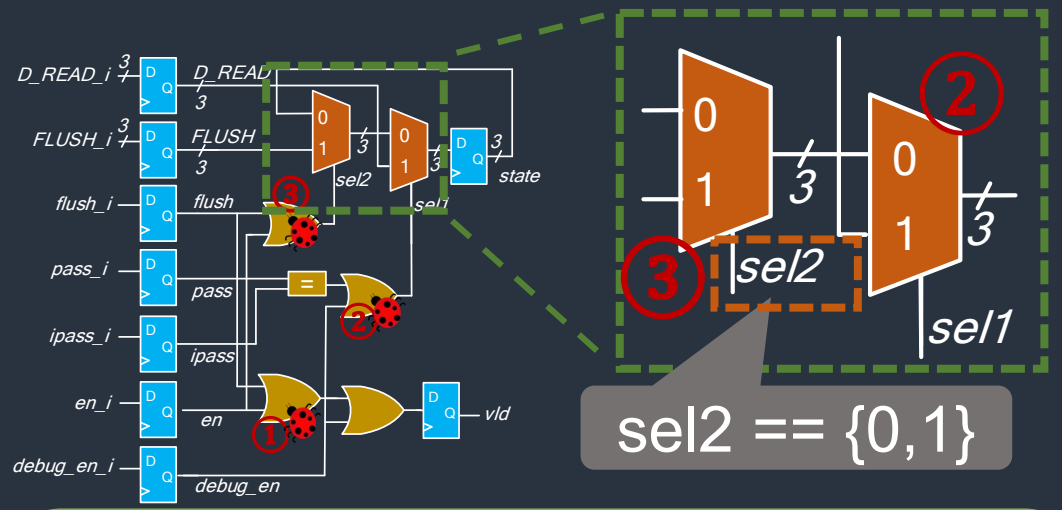
RFUZZ [2]



- Novelty: first hardware fuzzer
- Can fuzz any hardware design
- Covers select signals of MUXs coded as *control logic*

Existing Hardware Fuzzers

RFUZZ [2]

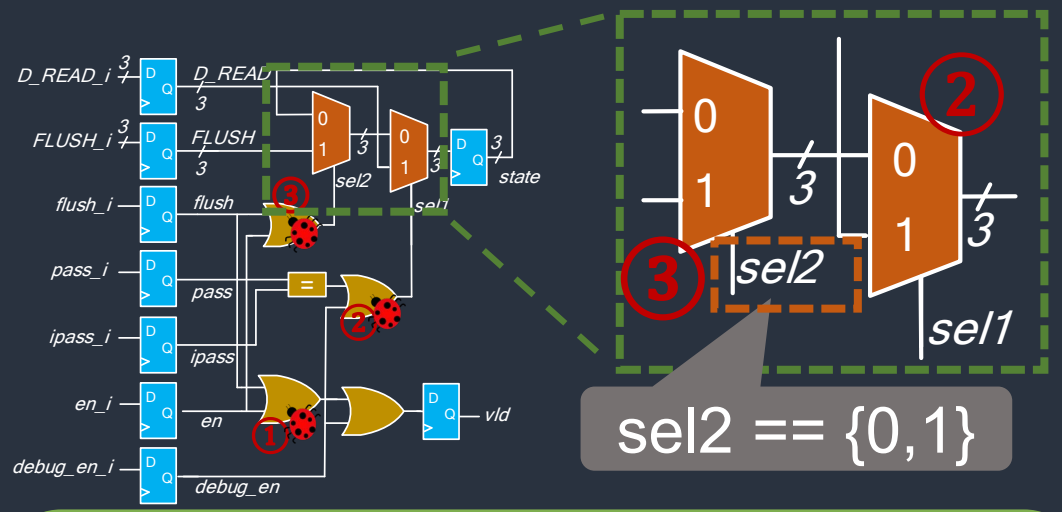


- Novelty: first hardware fuzzer

- Doesn't detect MUX ②
- Doesn't cover activity in combinational logic and flip-flops
- Computationally expensive
- Does not scale to large designs

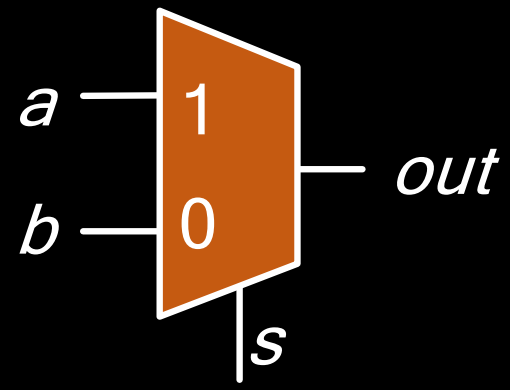
Existing Hardware Fuzzers

RFUZZ [2]



- Novelty: first hardware fuzzer

- Doesn't detect MUX ②
- Doesn't cover activity in combinational logic and flip-flops
- Computationally expensive
- Does not scale to large designs



MUX as control logic

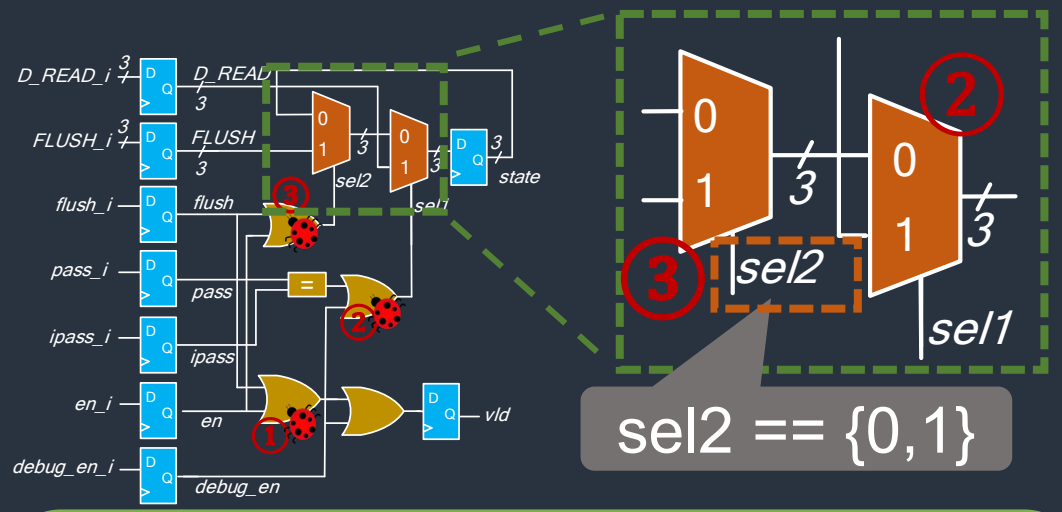
```
if (s)
    out <= a;
else
    out <= b;
```

MUX as combinational logic

```
assign out =>
  (s&a)|(!s&b)
```

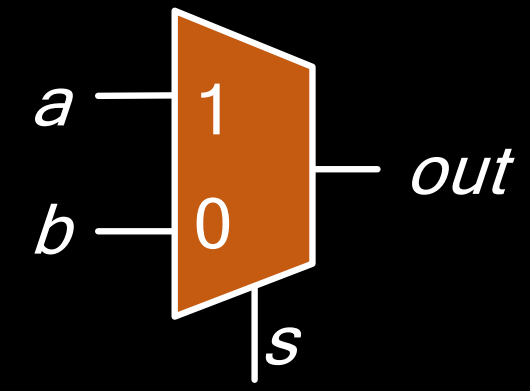
Existing Hardware Fuzzers

RFUZZ [2]



- Novelty: first hardware fuzzer

- Doesn't detect MUX ②
- Doesn't cover activity in combinational logic and flip-flops
- Computationally expensive
- Does not scale to large designs



MUX as control logic

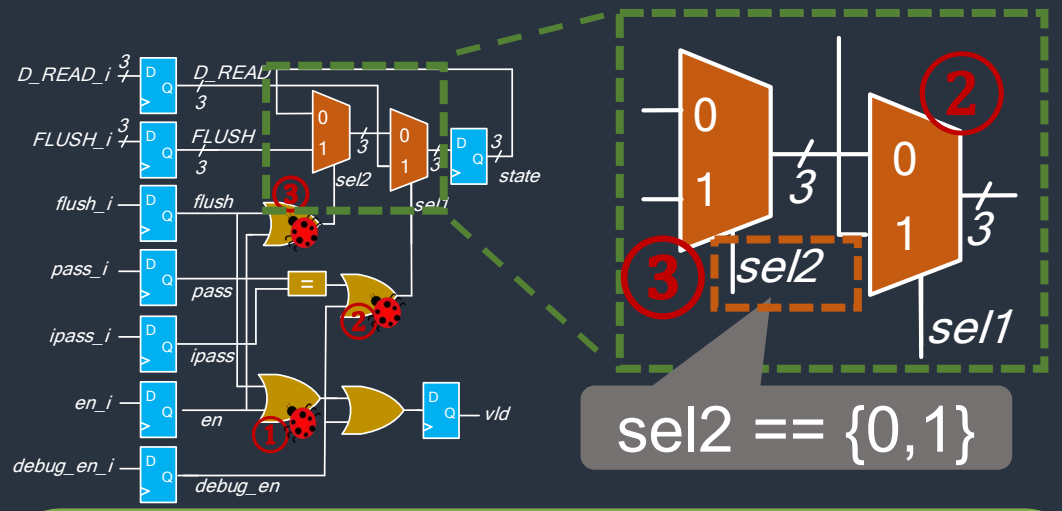
```
if (s)
    out <= a;
else
    out <= b;
```

MUX as combinational logic

```
assign out =>
  (s&a)|(!s&b)
```

Existing Hardware Fuzzers

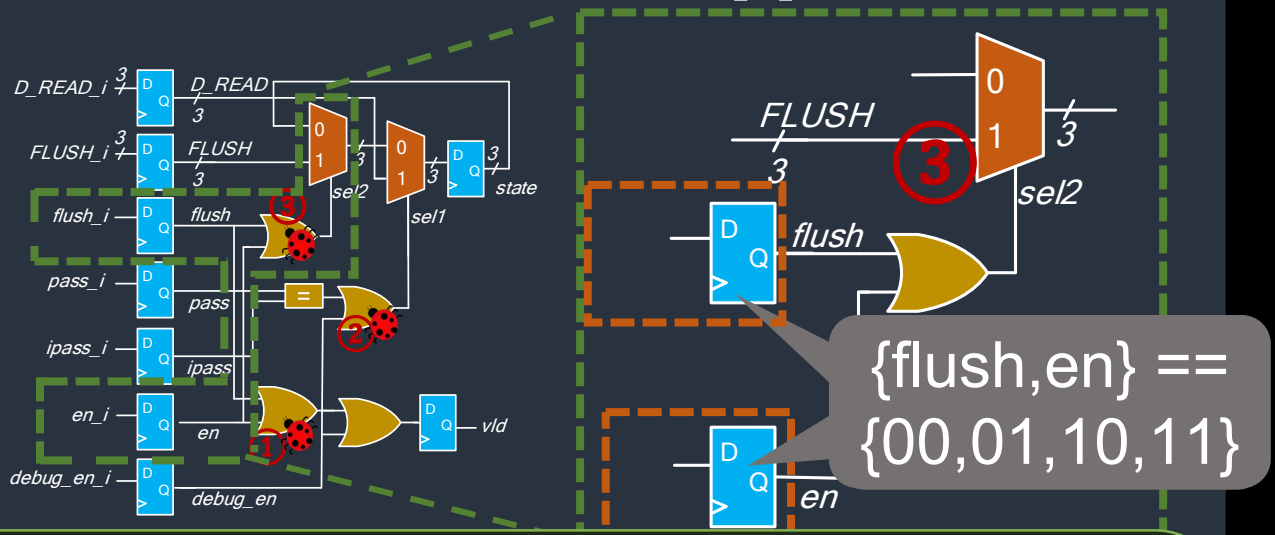
RFUZZ [2]



- Novelty: first hardware fuzzer

- Doesn't detect MUX ②
- Doesn't cover activity in combinational logic and flip-flops
- Computationally expensive
- Does not scale to large designs

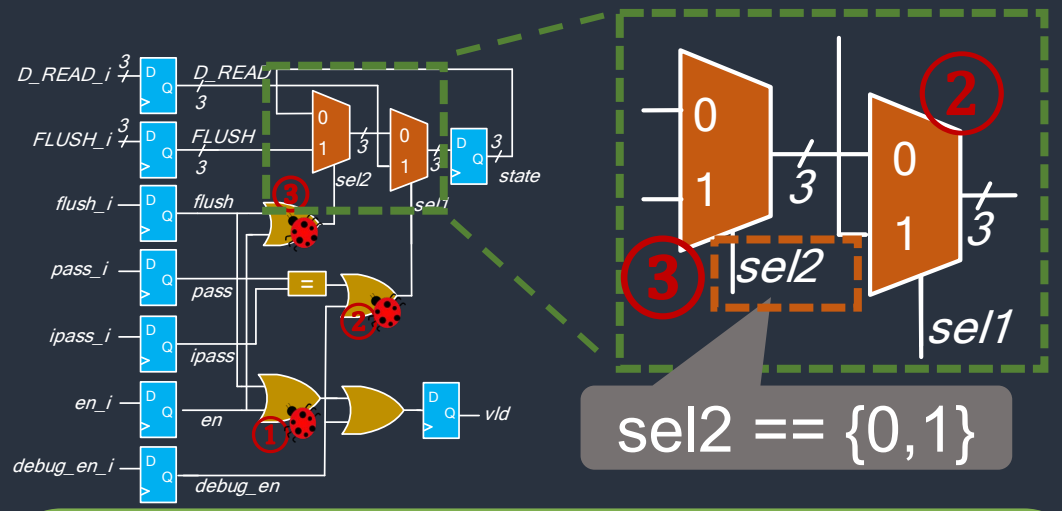
DifuzzRTL [3]



- Novelty: Resolved scalability issue of RFUZZ
- Covers registers driving select logic of MUXs coded as *control logic*
  → covers bug in ③

Existing Hardware Fuzzers

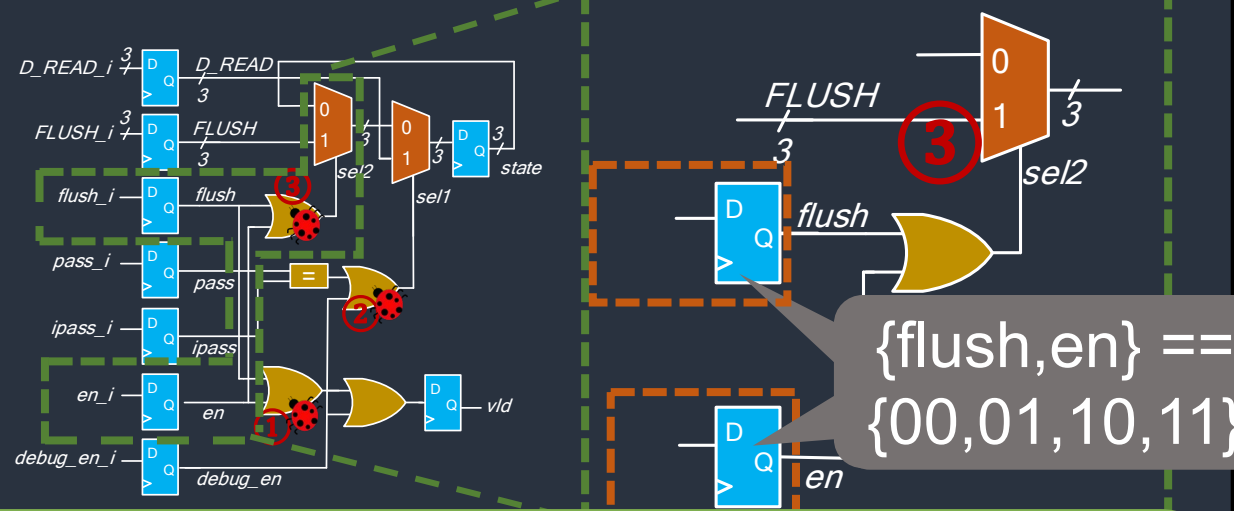
RFUZZ [2]



- Novelty: first hardware fuzzer

- Doesn't detect MUX ②
- Doesn't cover activity in combinational logic and flip-flops
- Computationally expensive
- Does not scale to large designs

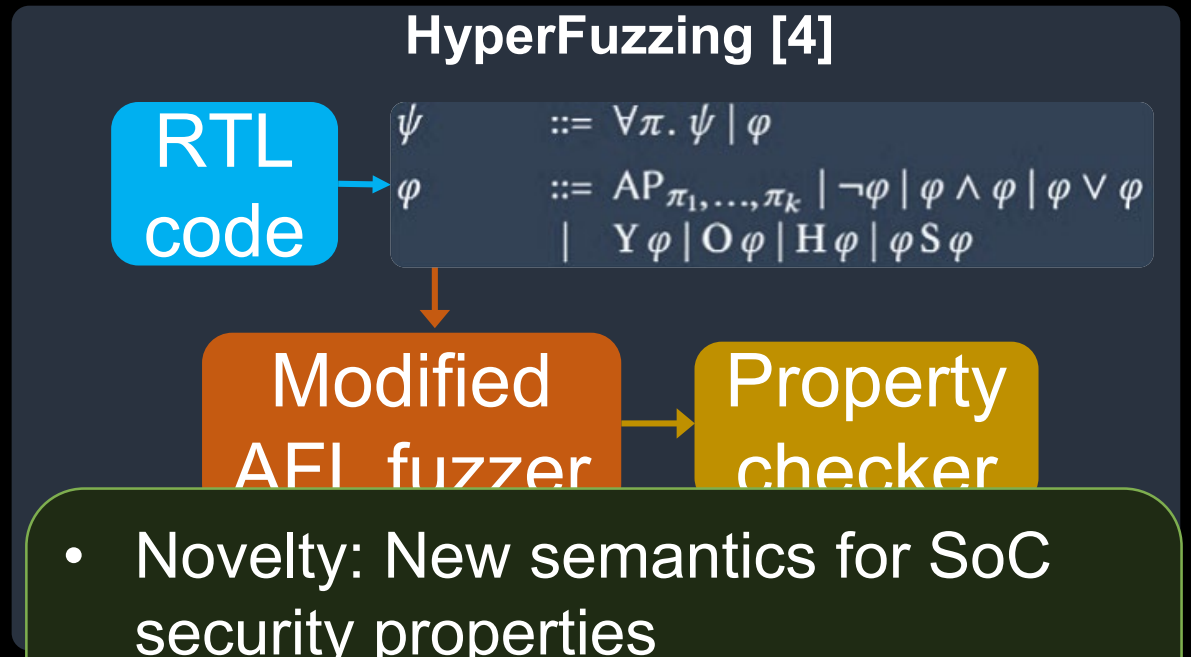
DifuzzRTL [3]



- Novelty: Resolved scalability issue of

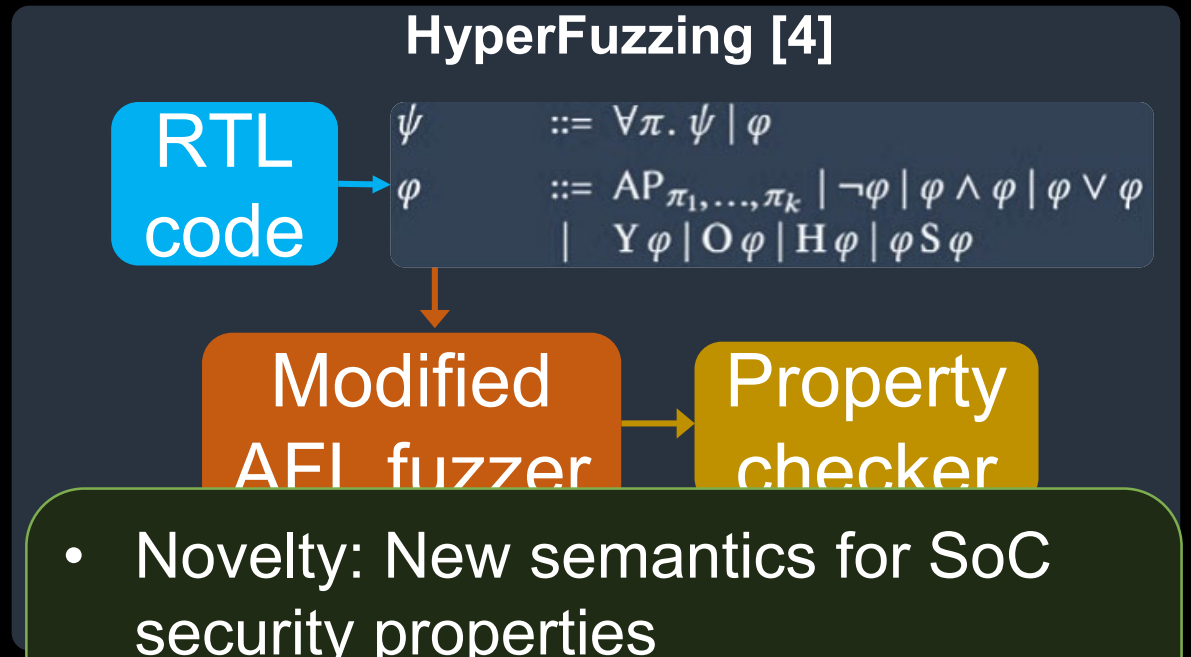
- Doesn't detect MUX ②
- Doesn't cover activity in combinational logic and flip-flops
- Bug comparison at end of program

Existing Hardware Fuzzers



- Novelty: New semantics for SoC security properties
- Fuzzer accelerates property checking

Existing Hardware Fuzzers

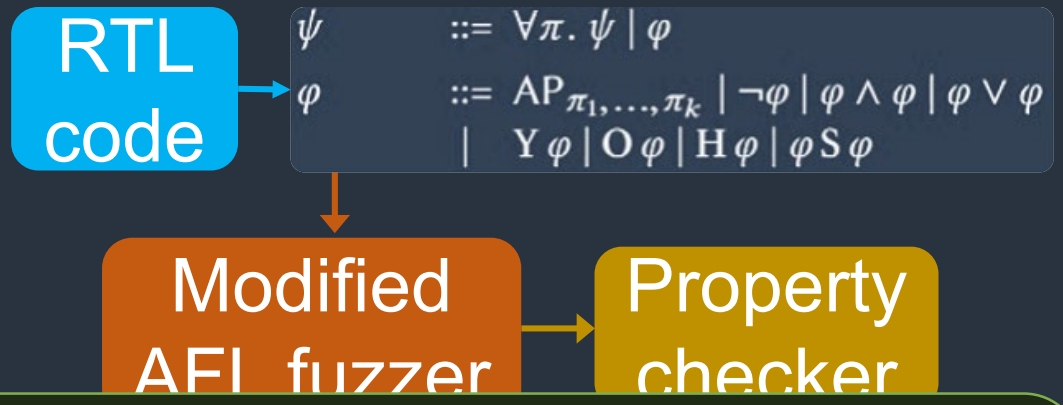


- Novelty: New semantics for SoC security properties
- Fuzzer accelerates property

- Not applicable to general hardware like FSMs or combinational logic
- Need to write security properties
- Supports Verilator simulator only

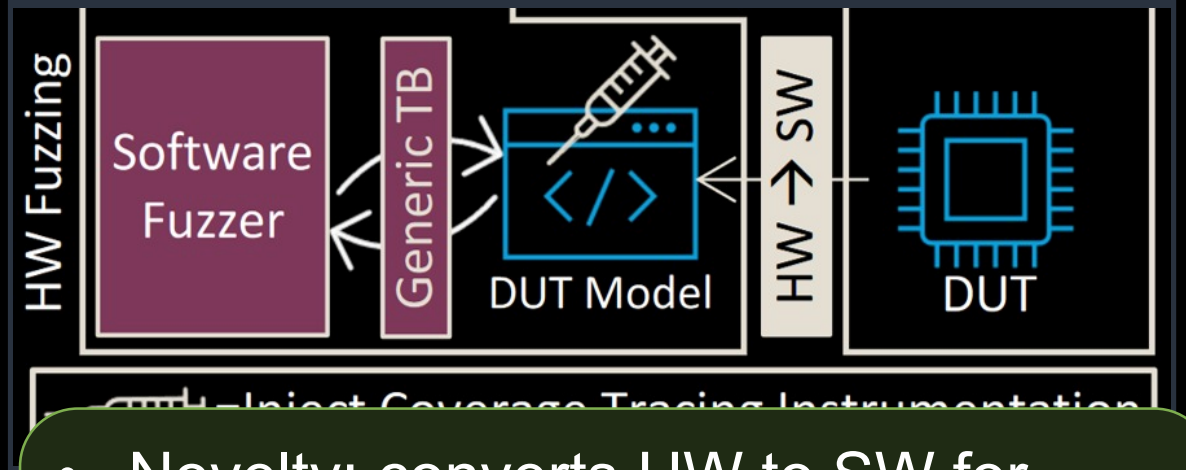
Existing Hardware Fuzzers

HyperFuzzing [4]



- Novelty: New semantics for SoC security properties
- Fuzzer accelerates property
- Not applicable to general hardware like FSMs or combinational logic
- Need to write security properties
- Supports Verilator simulator only

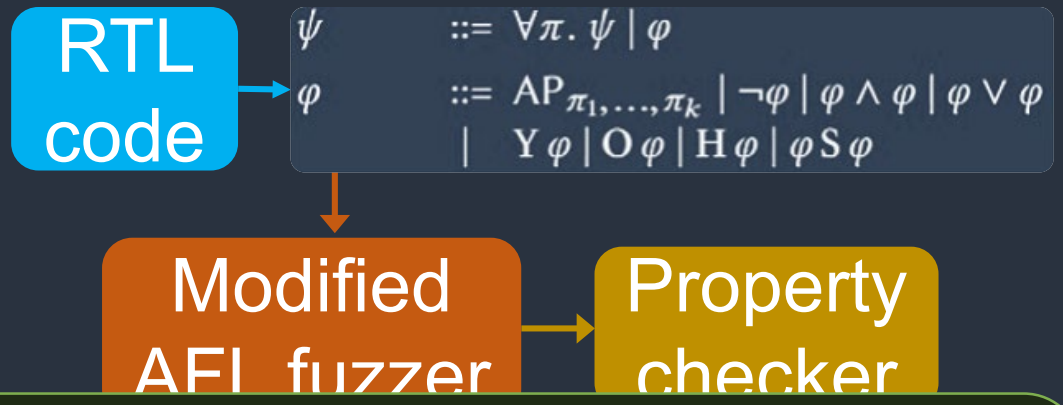
Trippel et al. [5]



- Novelty: converts HW to SW for fuzzing
- Existing software fuzzers can be integrated to fuzz hardware

Existing Hardware Fuzzers

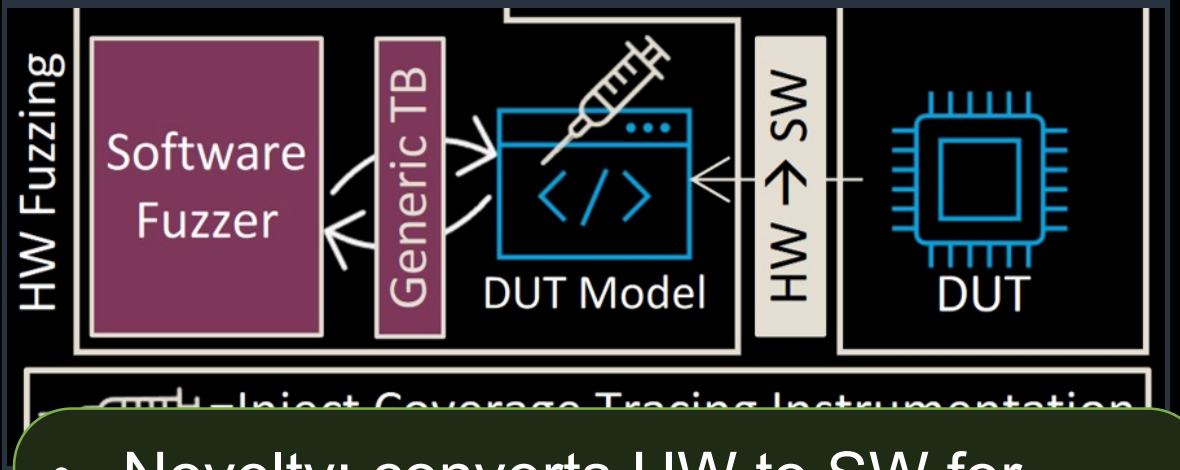
HyperFuzzing [4]



- Novelty: New semantics for SoC security properties
- Fuzzer accelerates property

- Not applicable to general hardware like FSMs or combinational logic
- Need to write security properties
- Supports Verilator simulator only

Trippel et al. [5]



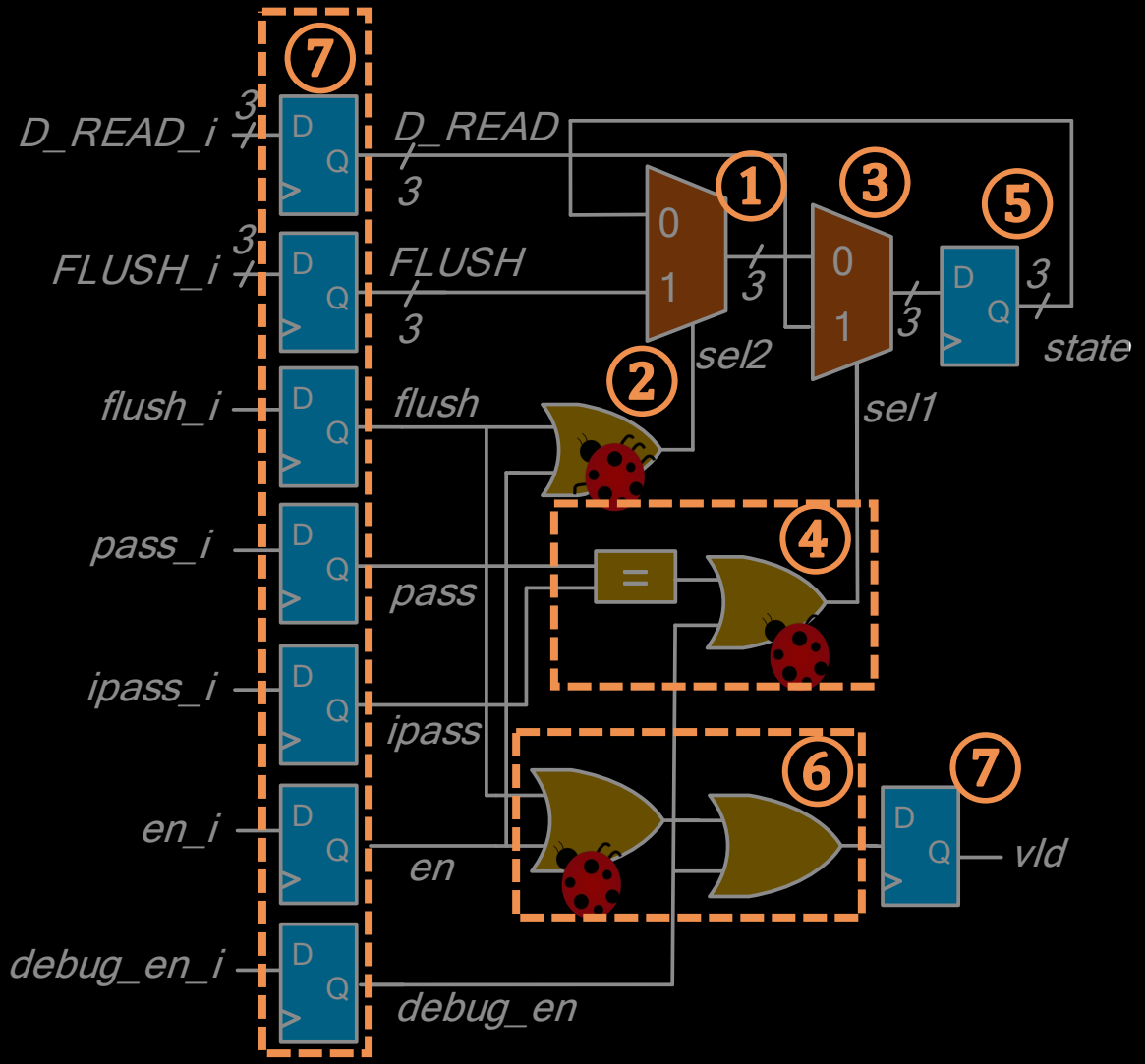
- Novelty: converts HW to SW for fuzzing
- Existing software fuzzers can be

- Supports Verilator like simulator only
- Does not support all Verilog constructs like latches, floating wires

Summary of Existing Techniques

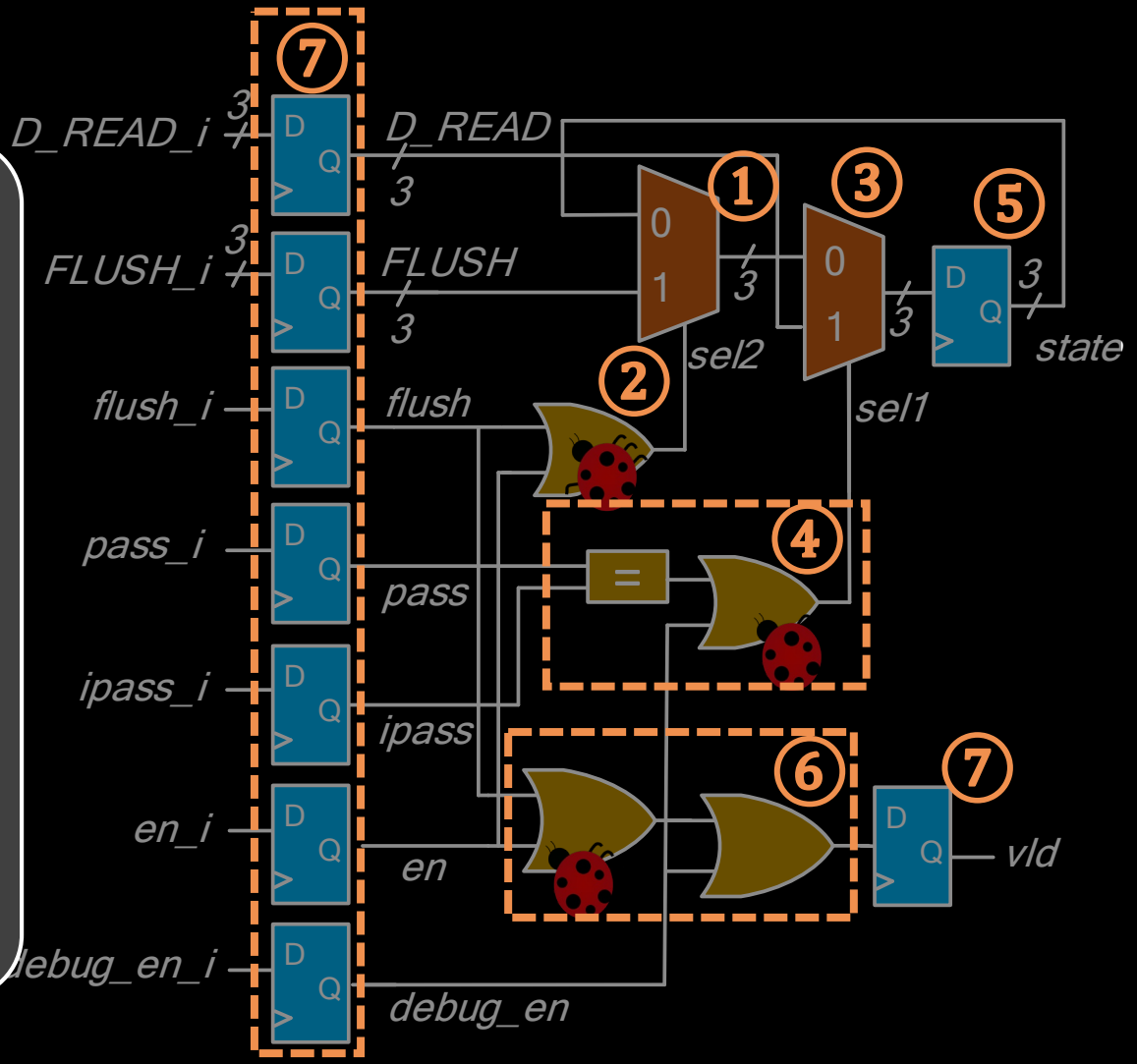
Technique	Hardware Components Covered	Scalability (Largest Design's LOC)	Applicability	Simulator	# Bugs
RFUZZ [2]	Select signals of some MUXs	5-stage Sodor core (4,088)	Any RTL	Any	0
DifuzzRTL [3]	Registers driving select signals of some MUXs	Boom (12,956 (Scala))	Processors	Any	16
HyperFuzzing [4]	Inserted properties	SHA crypto engine (1,196)	SoCs	Verilator	0
Trippel et al. [5]	SW FSM, line & edge HW toggle, functional	KMAC (4,585)	Any RTL	.v/.sv to C (Verilator)	5

Coverage Metrics of *TheHuzz*

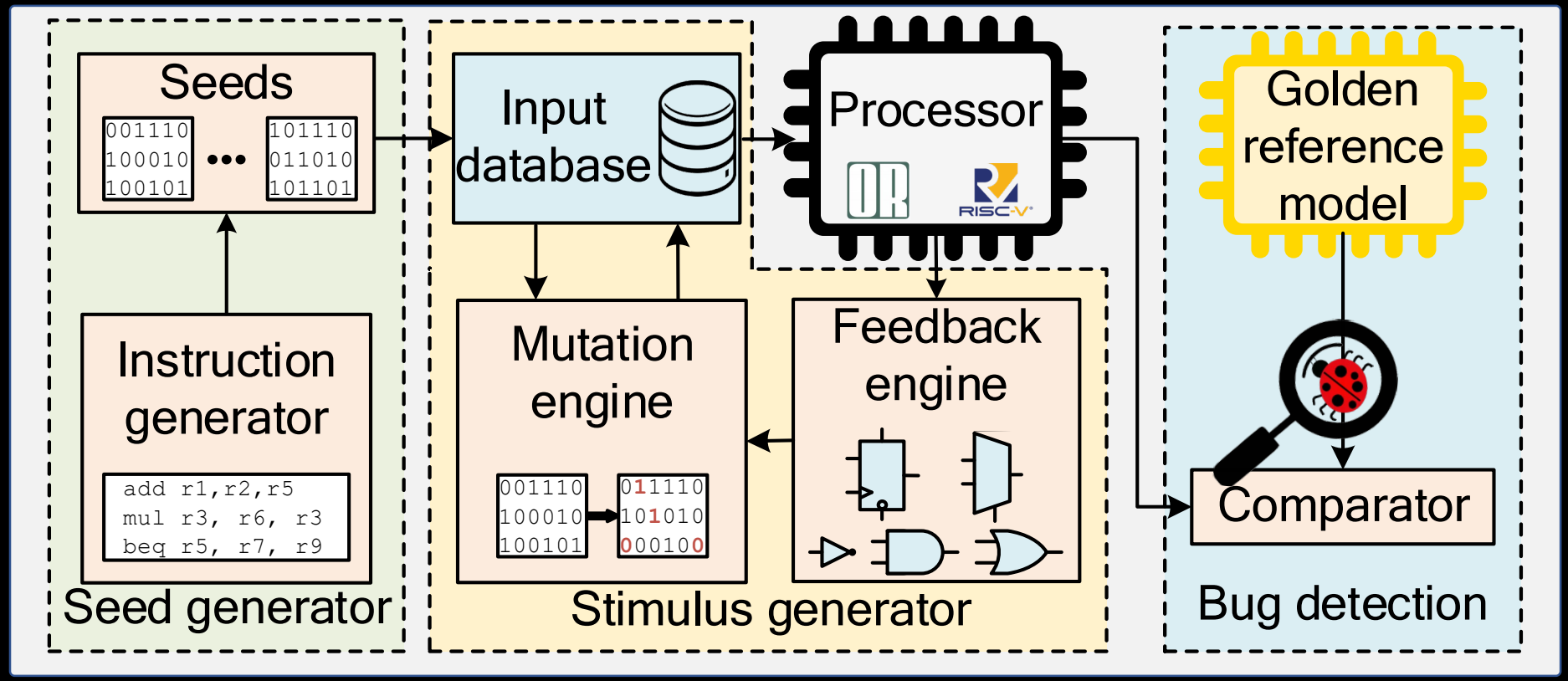


Coverage Metrics of *TheHuzz*

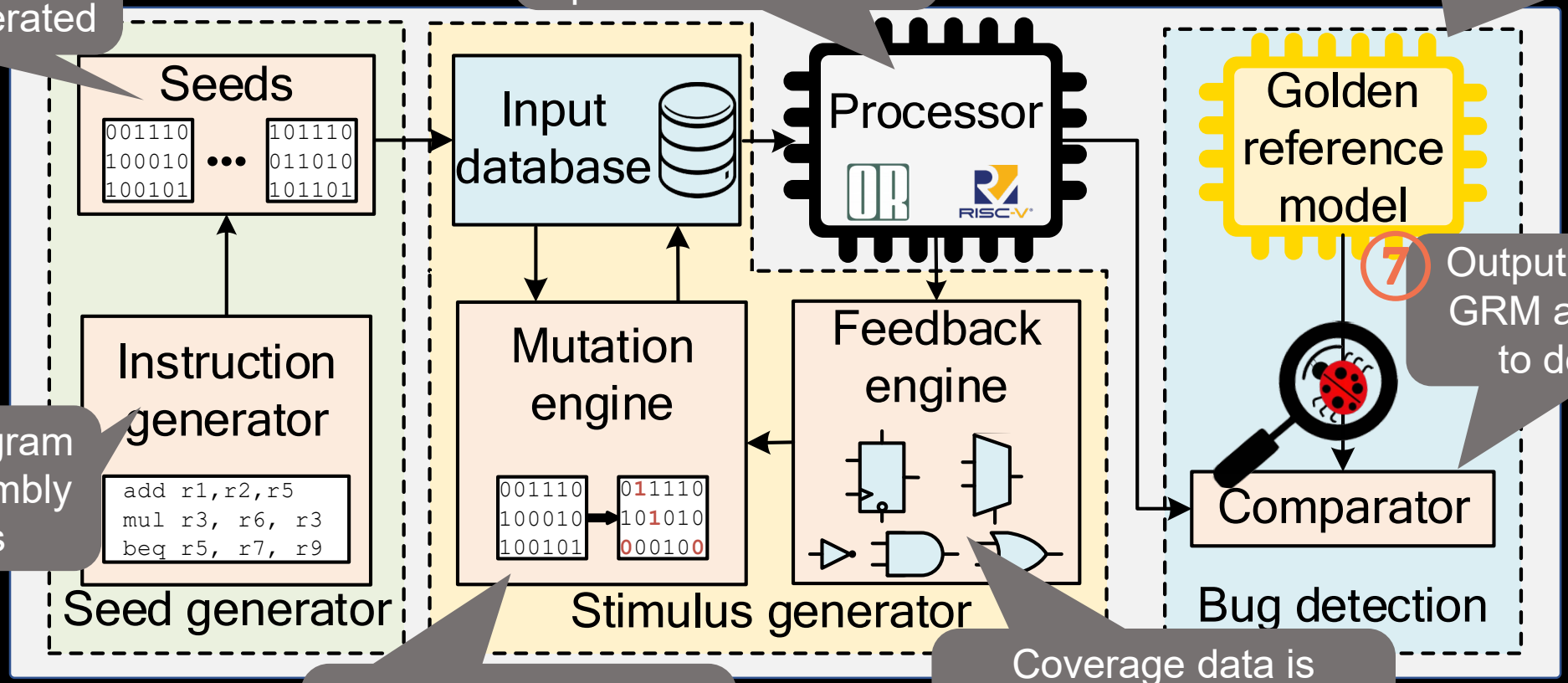
- **Statement**: All statements in RTL code
- **Branch**: Control signals *(sel1,sel3* of ③,①)
- **Toggle**: 0→1/1→0 transitions of flip-flops ⑦
- **FSM**: States & state transitions of FSM ⑤
- **Condition**: Control path combinational logic (*AND* gate in ②)
- **Expression**: Data path combinational logic (gates in ④ & ⑥)



Design of *TheHuzz*



Design of *TheHuzz*



1 Initial set of inputs (seeds) are randomly generated

2 Inputs are program files with assembly instructions

3 Target processor is simulated using inputs from database

4 A GRM is also simulated with the same input

5 Coverage data is analyzed to determine if new logic is explored

6 Current inputs are mutated to generate new inputs

7 Output of target and GRM are compared to detect bugs

Bugs Detected

***TheHuzz* detected 11 bugs including 8 new bugs, 5 CVEs**

Processor	Bug Description	CVE/CWE	Location	Coverage
Ariane (cva6)[6] RISC-V [8] 2.07 $\times 10^4$ LOC	Incorrect implementation of logic to detect the FENCE.I instruction.	CWE-440	Decoder	Branch
	Failure to detect cache coherency violation	CWE-1202	Cache controller	FSM
mor1kx [7] OpenRISC [9] 2.21 $\times 10^4$ LOC	Read/write access check not implemented for privileged reg.	CVE-2021 -41614	Register file	Condition
	Incomplete implementation of EEAR register write logic	CVE-2021 -41613	Register file	Condition
or1200 [7] OpenRISC [9] 3.16 $\times 10^4$ LOC	Incomplete update logic of overflow bit for MSB/MAC instrs.	CVE-2021 -40506	ALU	Toggle

Bugs Detected

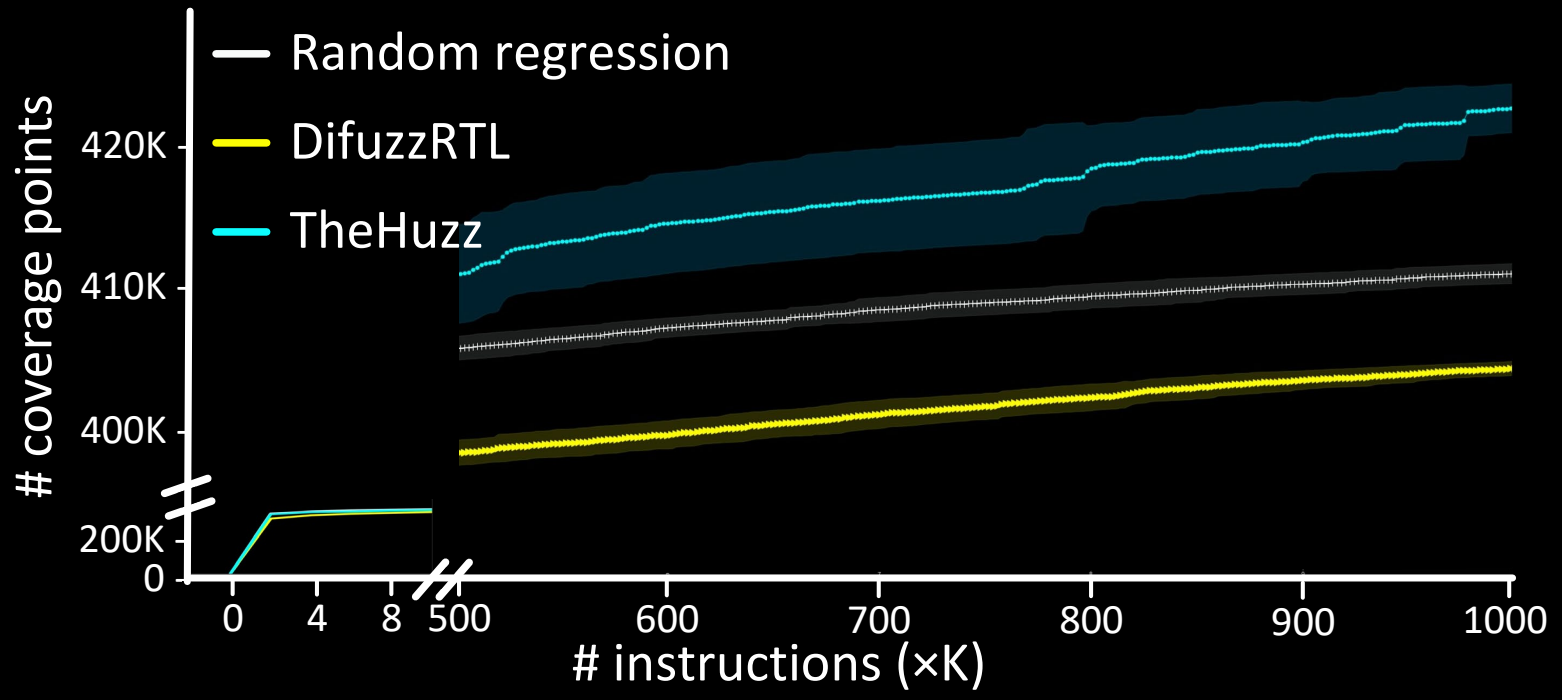
***TheHuzz* detected 11 bugs including 8 new bugs, 5 CVEs**

Processor	Bug Description	CVE/CWE	Location	Coverage
Ariane (cva6)[6] RISC-V [8] 2.07 ×10^4 LOC	Incorrect implementation of logic to detect the FENCE.I instruction.			ch
	Failure to detect cache coherency violation			
mor1kx [7] OpenRISC [9] 2.21 ×10^4 LOC	Read/write access check not implemented for privileged reg.			ition
	Incomplete implementation of EEAR register write logic			ition
or1200 [7] OpenRISC [9] 3.16 ×10^4 LOC	Incomplete update logic of overflow bit for MSB/MAC instrs.	CVE-2021 -40506	ALU	Toggle

Exploit 1:
Arbitrary Code Execution on Ariane

Exploit 2:
Privilege Escalation on mor1kx

Coverage Results



- **Rocket core:** RISC-V, 32-bit, 5-stage pipelined, 6.65 x 10^4 coverage points
- **1.98x** and **3.33x** the speed of random regression & DifuzzRTL

Conclusion

- Our hardware fuzzer, *TheHuzz* is
  - **Compatible**: Chisel/.v/.vhdl, any commercial hardware simulator
  - **Automated**: Design agnostic
  - **Practical**: Simple to run (50+ students trained)
  - **Efficient**: Detected 11 bugs, higher coverage than existing techniques
- We demonstrated the security impact of bugs through two exploits
- Future work
  - Extend TheHuzz to support FPGA emulation
  - Fuzzing non-processor designs
  - Fuzzing parametric properties of hardware
  - Fuzzing to detect side-channel vulnerabilities

Thank you!

rahulkande@tamu.edu
https://seth.engr.tamu.edu/