Oops..! I Glitched it Again! How to Multi-Glitch the Glitching-Protections on ARM TrustZone-M

Marvin Saß, Richard Mitev, Ahmad-Reza Sadeghi

System Security Lab

Technical University of Darmstadt, Germany

System Security Lab

TECHNISCHE UNIVERSITÄT DARMSTADT

Motivation - Countermeasures

DCFG_CC_SOCU: Credential constraints:

DCFG_CC_SOCU is a bit mask that specifies debug access rights. It is derived from combination of PFR words CMPA.CC_SOCU_DFLT, CMPA.CC_SoCU_PIN, CFPA.CC_SOCU_DFLT_NS, CFPA.CC_SoCU_PIN_NS:

- Lower half-words of these PFR words define the functionality.

48.10.6.24 IDXBLK_L_DP register

This register is duplicate of IDXBLK_L register and provides protection against malicious

48.10.6.21 Index blocking duplicate register (IDX8 - IDX15)

This register is duplicate of IDXBLK_H register and provides protection against malicious

47.4.74 Master secure level anti-pole register

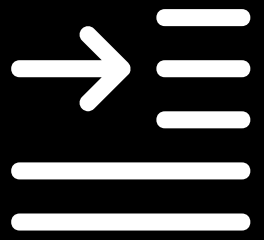
This register is inverse of MASTER_SEC_LEVEL register above. Secondary register with

47.4.77 Secure control duplicate register

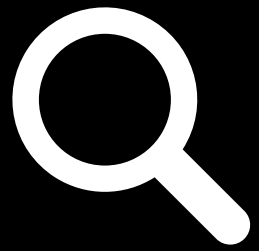
This register is duplicate of **MISC_CTRL_REG**. A secondary register with duplicate programing is implemented to provide better protection against malicious hacking attacks such as glitch attack.

LPC55S6x/LPC55S2x/LPC552x User manual

Consequence



Modify Register

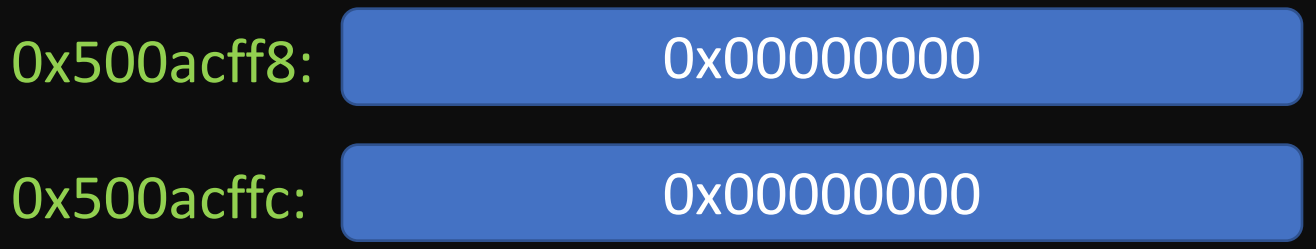


Direct Memory Access

Read Secret Keys

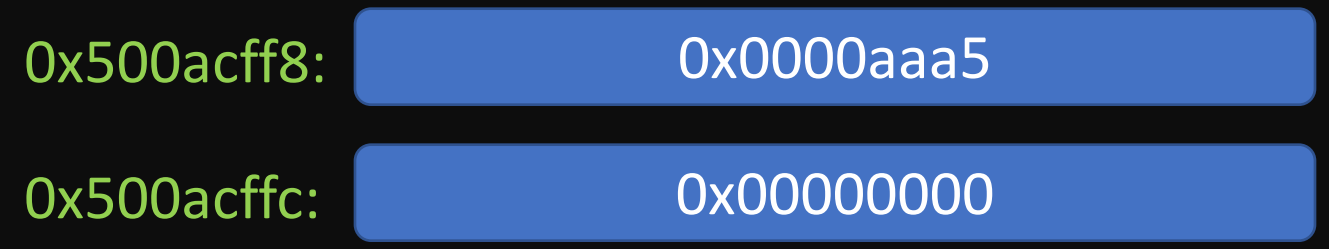
Access TrustZone

Duplicated Registers



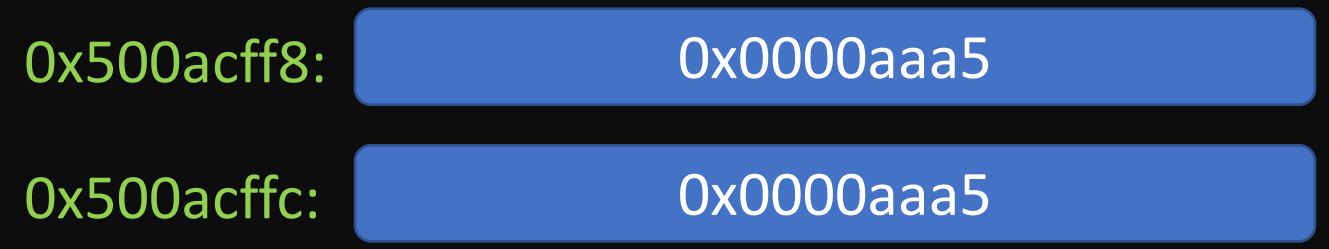

```
ldr     r1, =0x500acff8
movw    r2, #0xaaa5
str     r2, [r1]
…
ldr     r1, =0x500acffc
movw    r2, #0xaaa5
str     r2, [r1]
```

Duplicated Registers




```
ldr     r1, =0x500acff8
movw    r2, #0xaaa5
str     r2, [r1]
...
ldr     r1, =0x500acffc
movw    r2, #0xaaa5
str     r2, [r1]
```

Duplicated Registers




```
ldr     r1, =0x500acff8
movw    r2, #0xaaa5
str     r2, [r1]
…
ldr     r1, =0x500acffc
movw    r2, #0xaaa5
str     r2, [r1]
```

Duplicated Registers

0x500acff8: 0x00000000

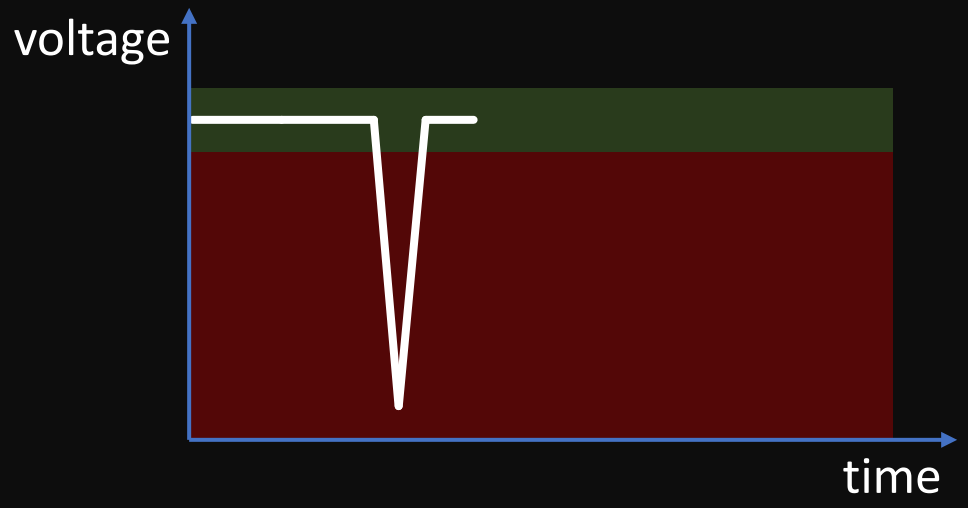
0x500acffc: 0x00000000


```
ldr     r1, =0x500acff8
movw    r2, #0xaaa5
str     r2, [r1]
...
ldr     r1, =0x500acffc
movw    r2, #0xaaa5
str     r2, [r1]
```

Duplicated Registers

0x500acff8: 0x00000000

0x500acffc: 0x00000000

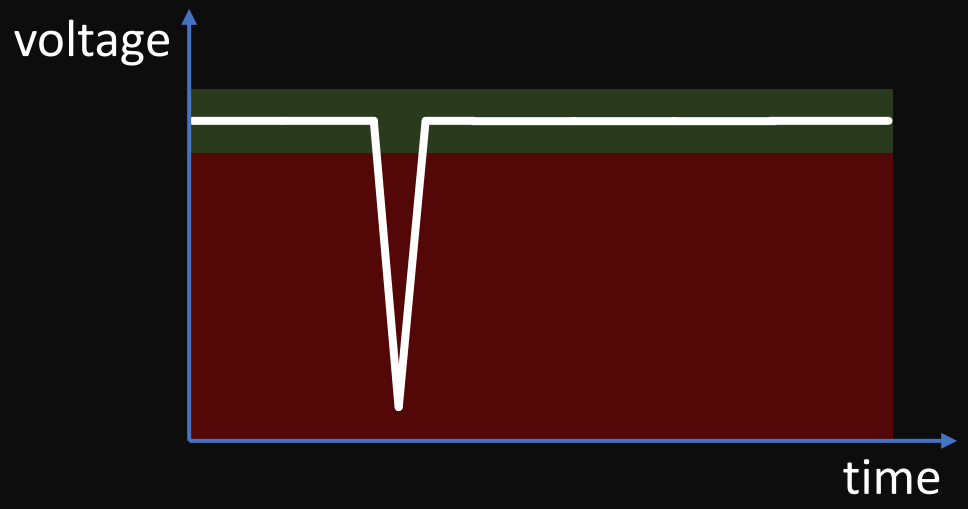



```
ldr     r1, =0x500acff8
movw    r2, #0xaaa5
str     r2, [r1]
...
ldr     r1, =0x500acffc
movw    r2, #0xaaa5
str     r2, [r1]
```

Duplicated Registers

0x500acff8: 0x00000000

0x500acffc: 0x0000aaa5



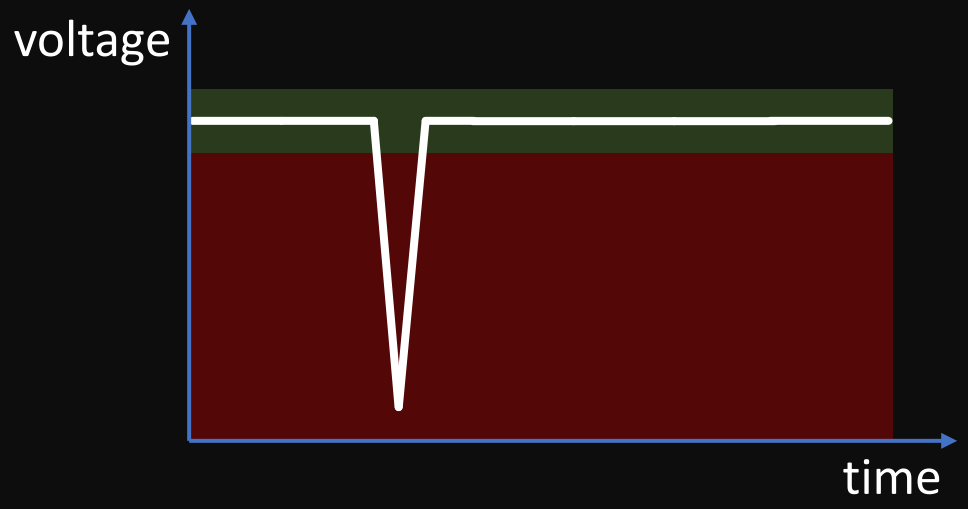

```
ldr     r1, =0x500acff8
movw    r2, #0xaaa5
str     r2, [r1]
...
ldr     r1, =0x500acffc
movw    r2, #0xaaa5
str     r2, [r1]
```

Duplicated Registers

0x500acff8: 0x00000000

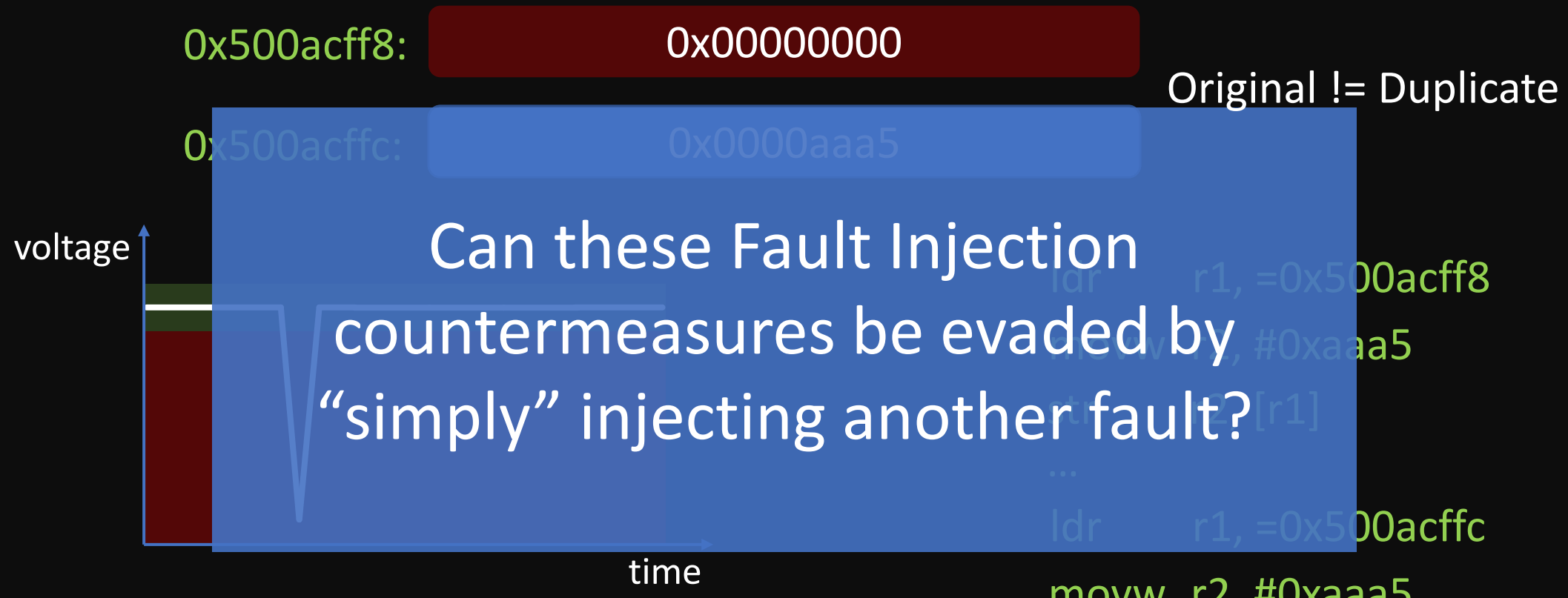
0x500acffc: 0x0000aaa5

Original != Duplicate



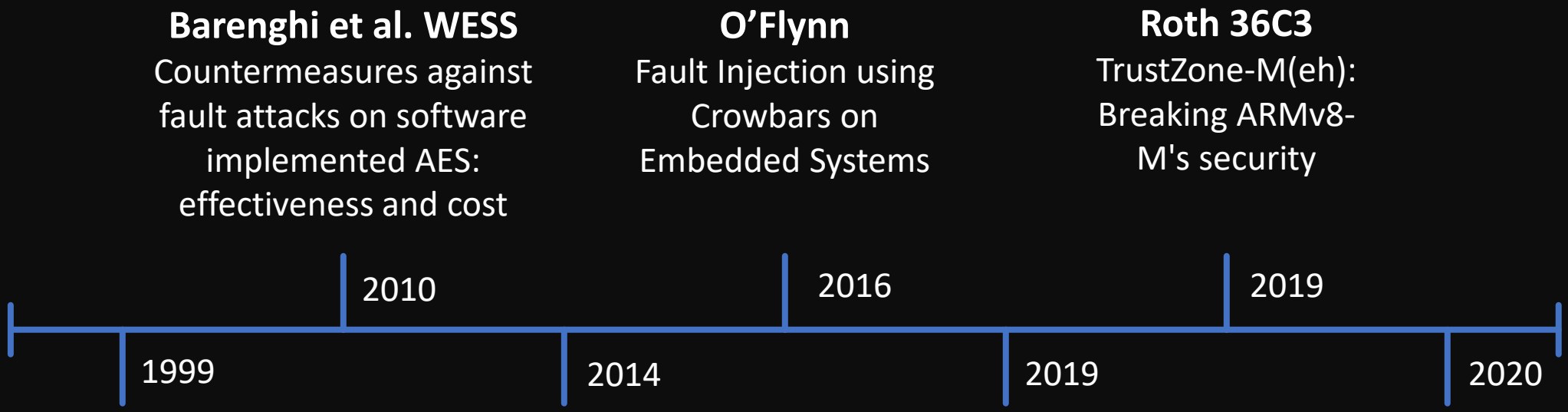

```
ldr     r1, =0x500acff8
movw    r2, #0xaaa5
str     r2, [r1]
…
ldr     r1, =0x500acffc
movw    r2, #0xaaa5
str     r2, [r1]
```

Duplicated Registers

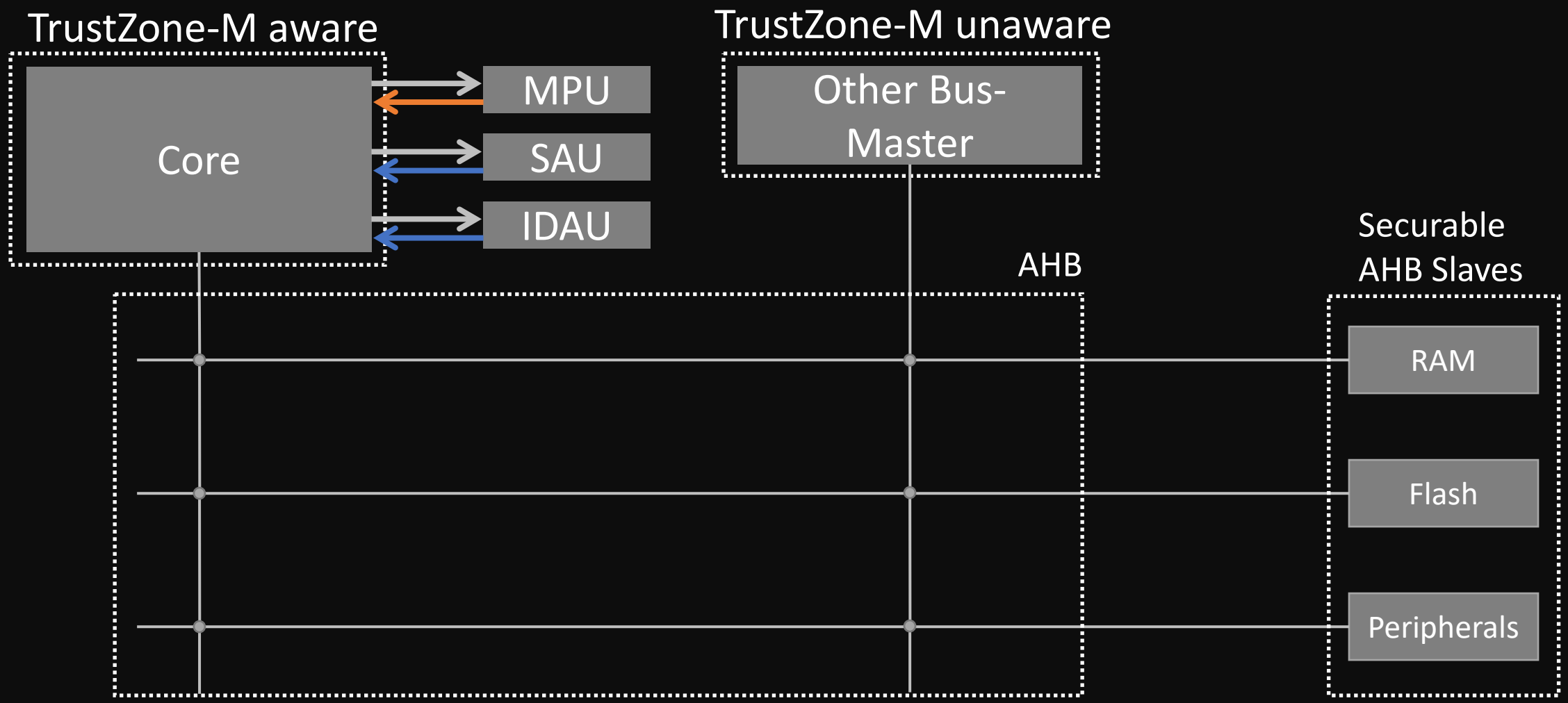


How secure are “glitch protected embedded processors” against multiple fault injection?

Related Work

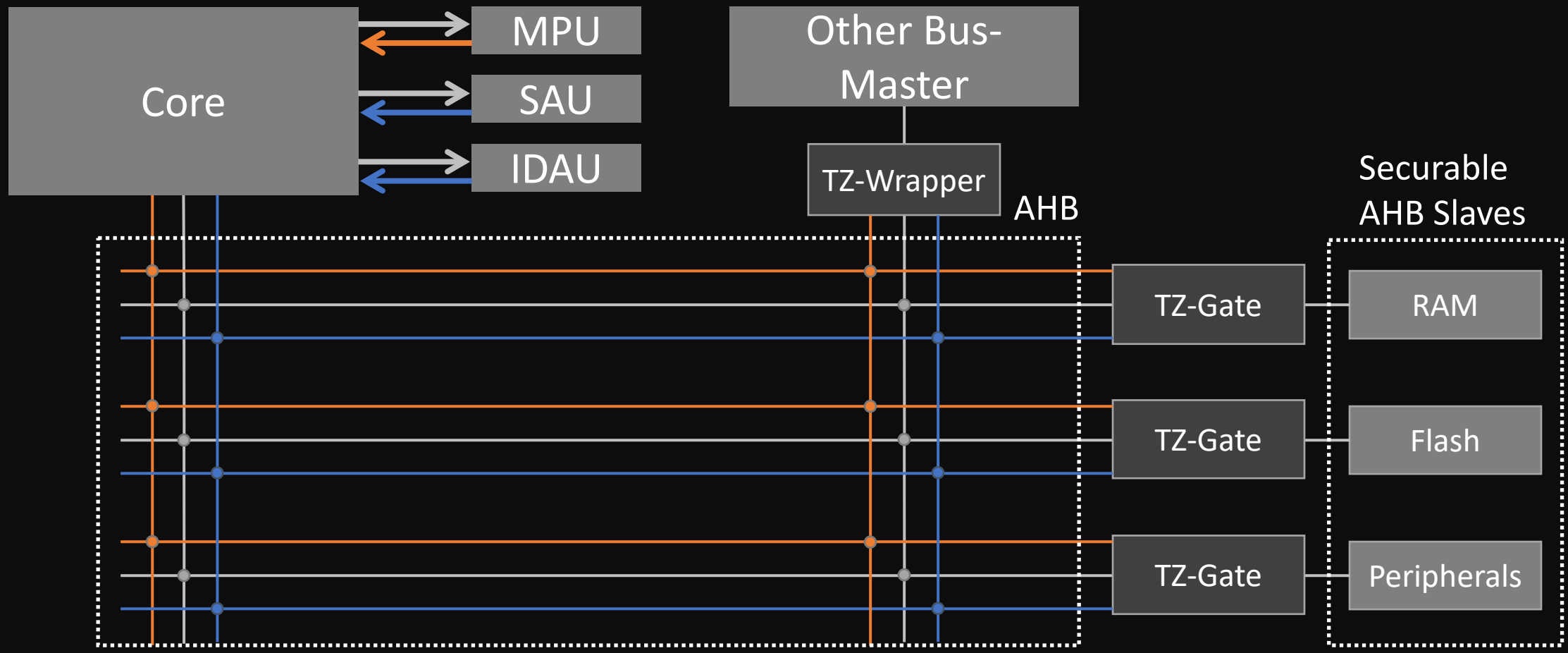


TrustZone-M Architecture

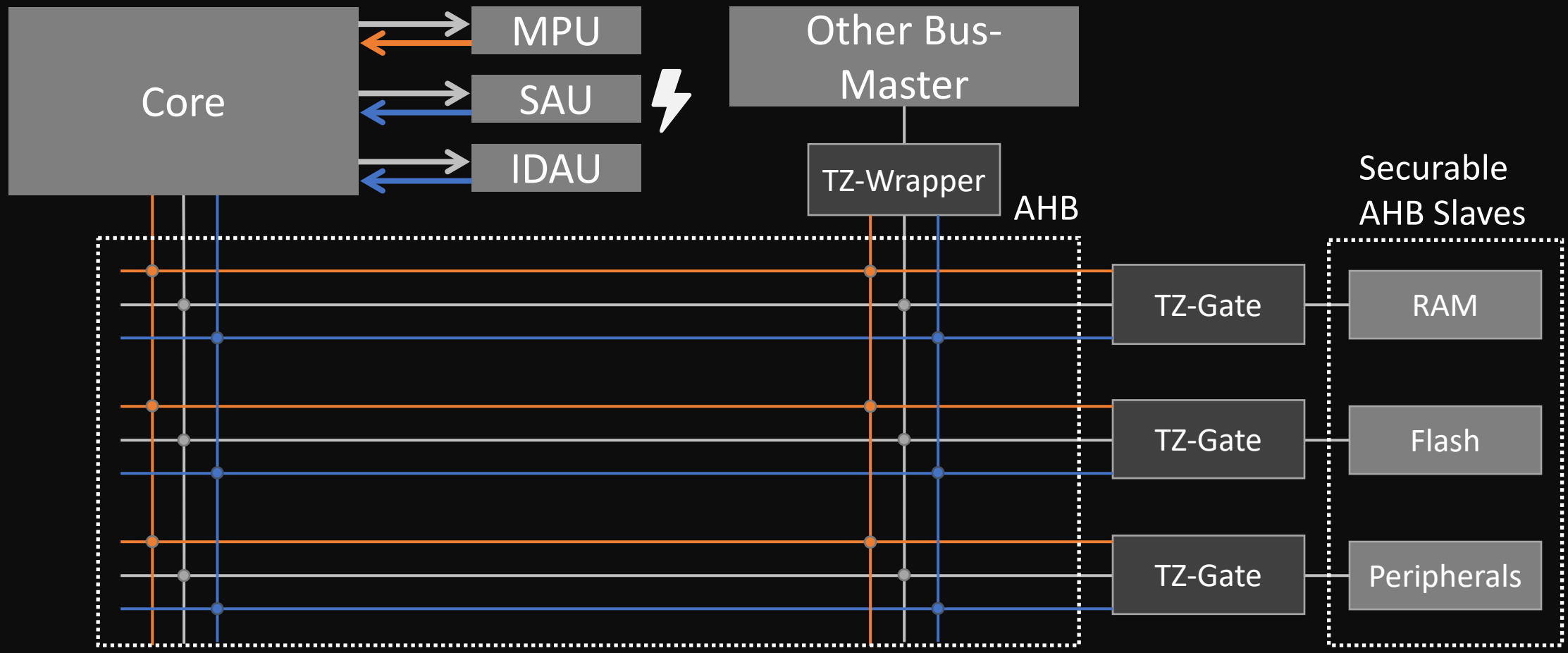


Address Privilege Level Security State

TrustZone-M Architecture



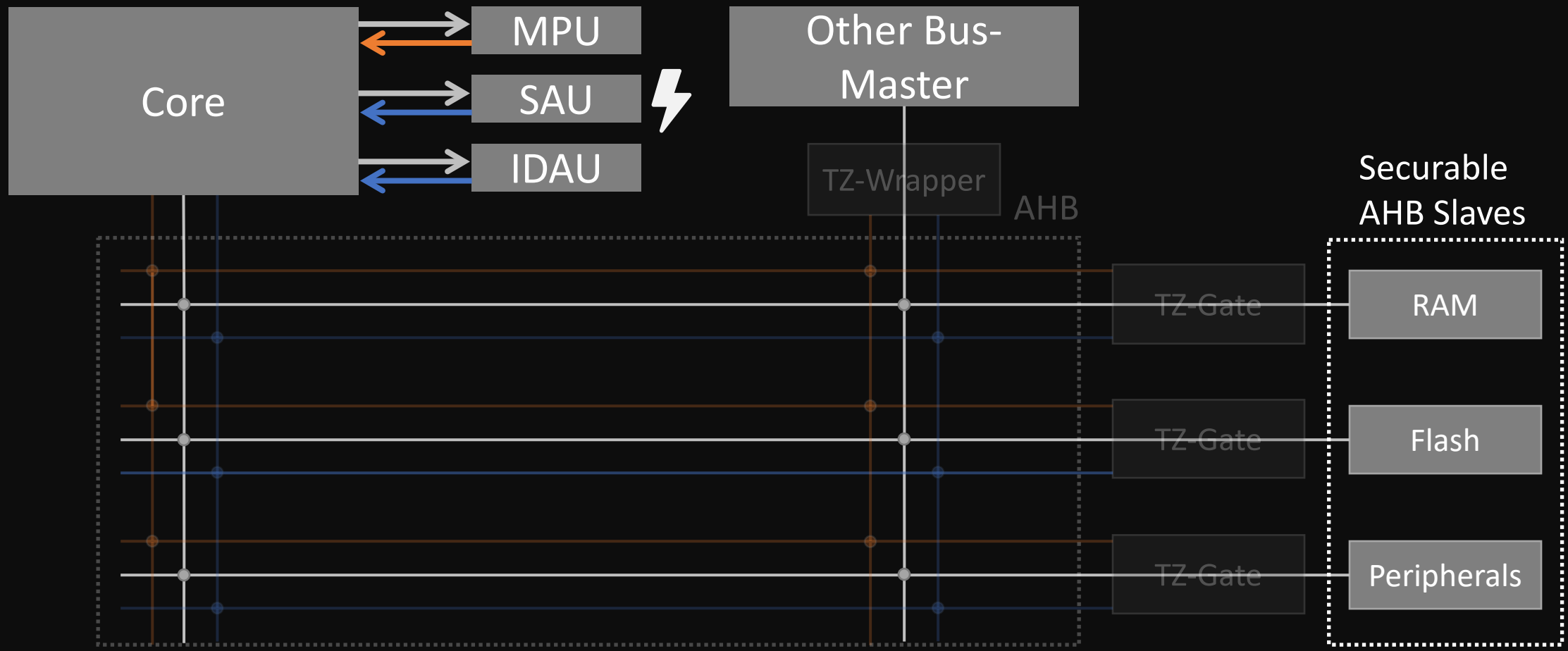
Most Related Work



Address Privilege Level Security State

(e.g., STM32L5, M2351, SAML11)

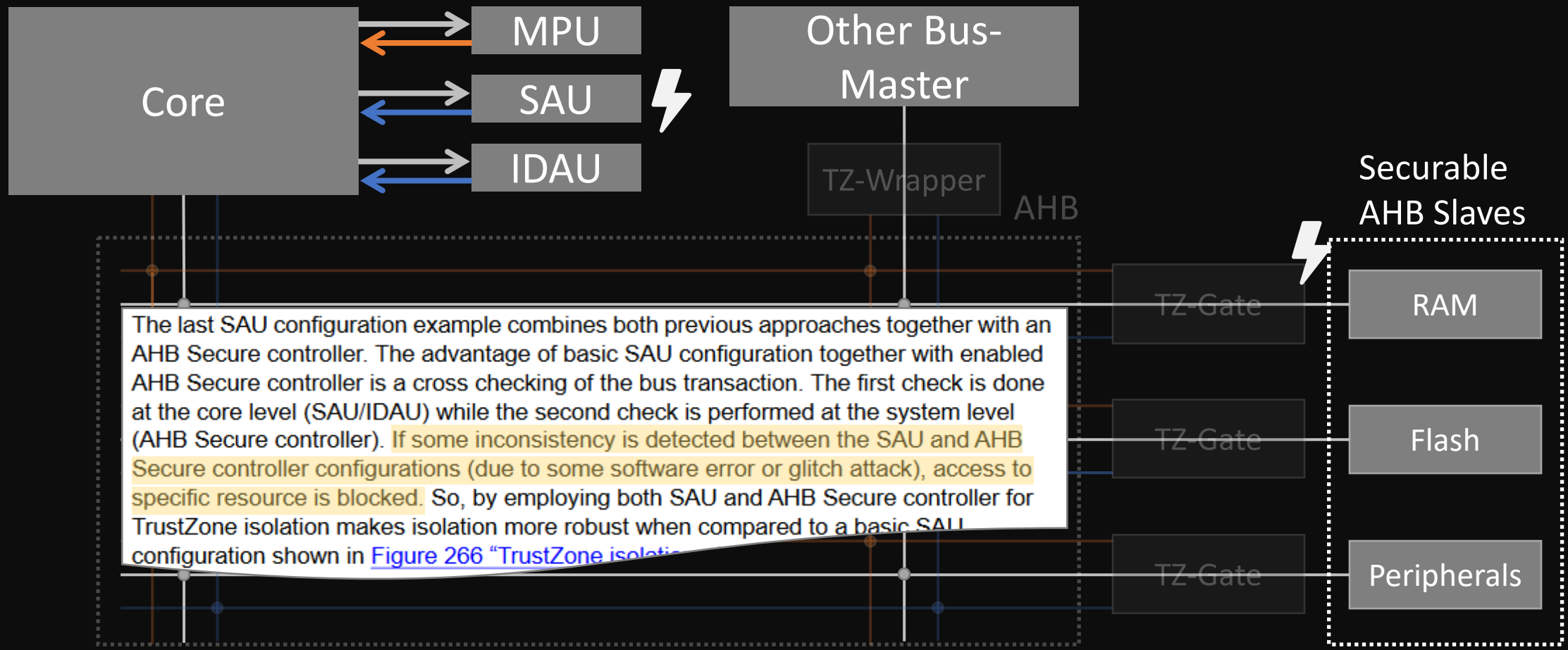
Most Related Work



Address Privilege Level Security State

(e.g., STM32L5, M2351, SAML11)

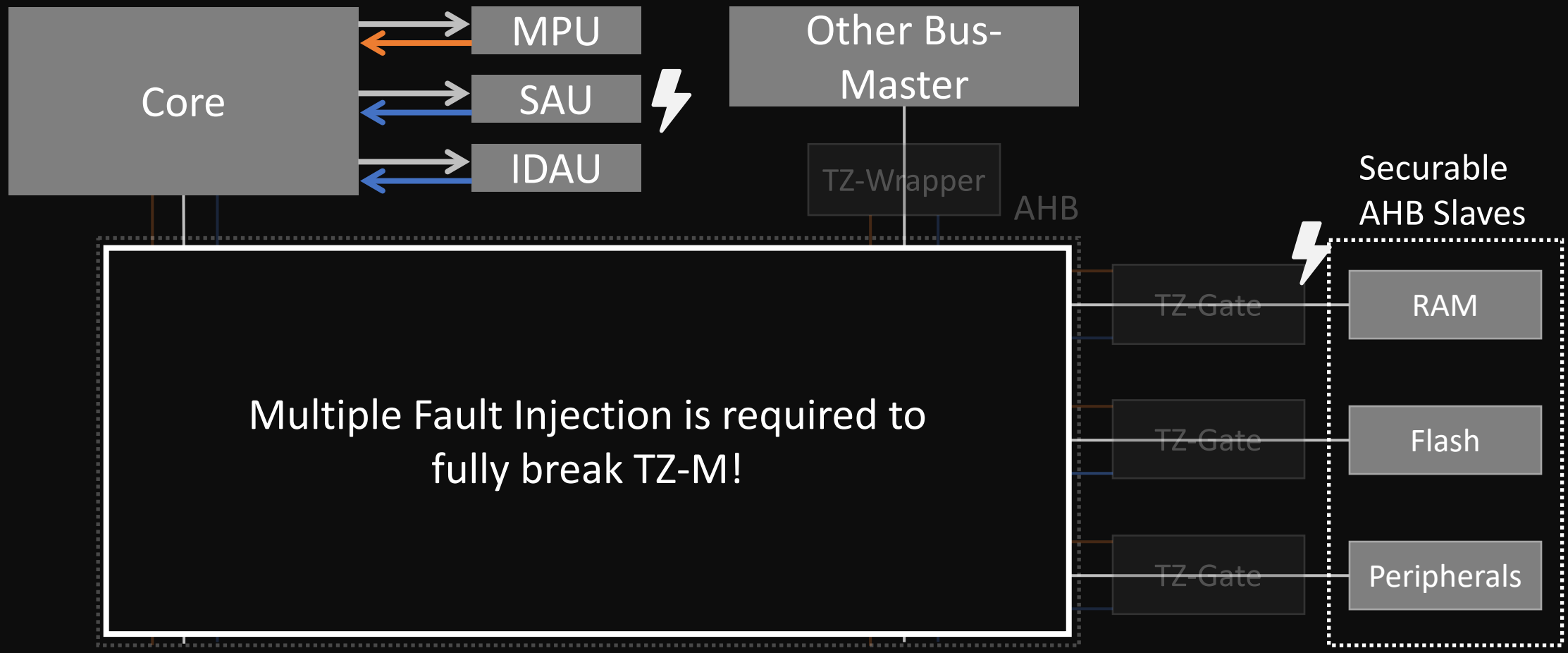
Most Related Work



Address Privilege Level Security State

(e.g., STM32L5, M2351, SAML11)

Most Related Work

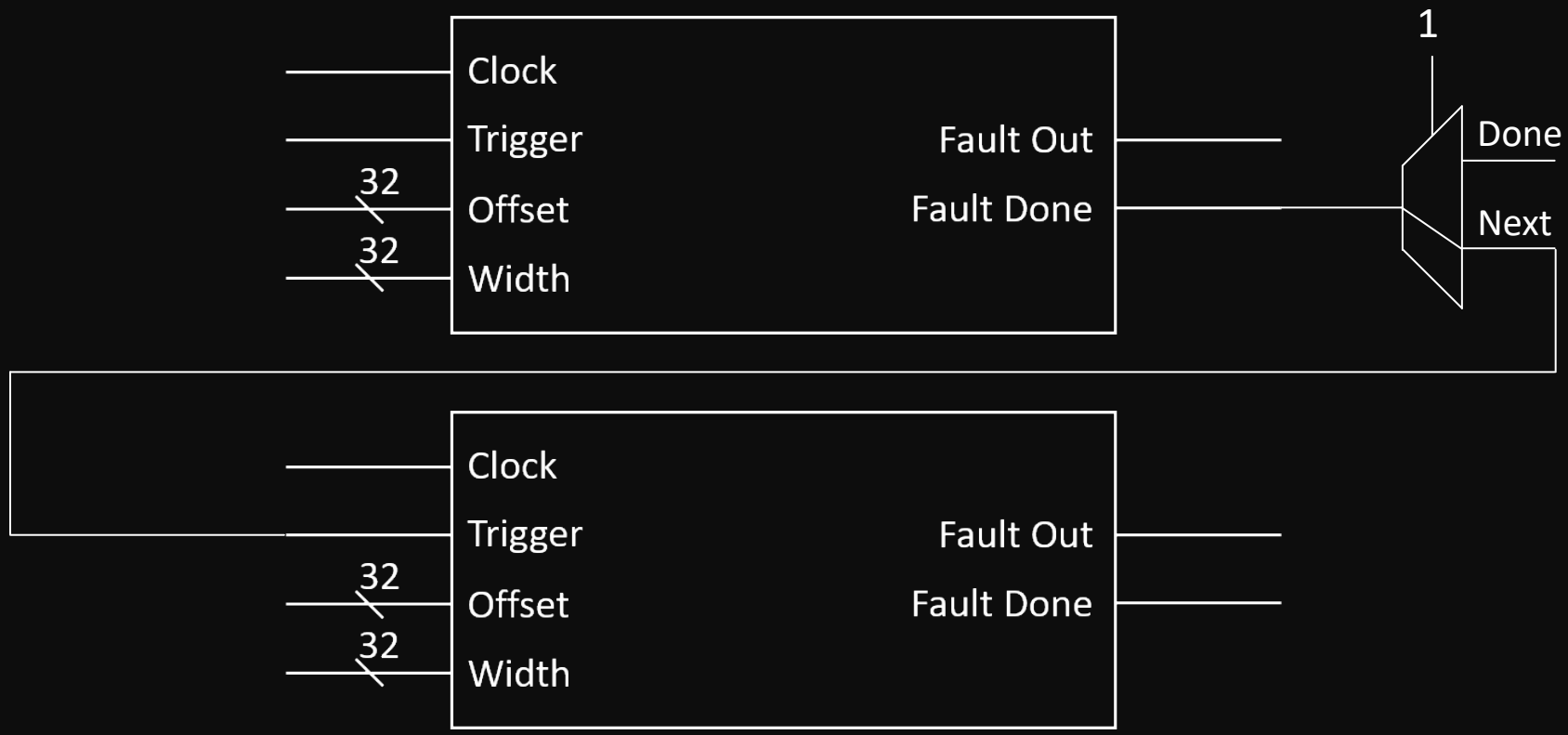


Address Privilege Level Security State

(e.g., STM32L5, M2351, SAML11)

Multiple Voltage Fault Hardware

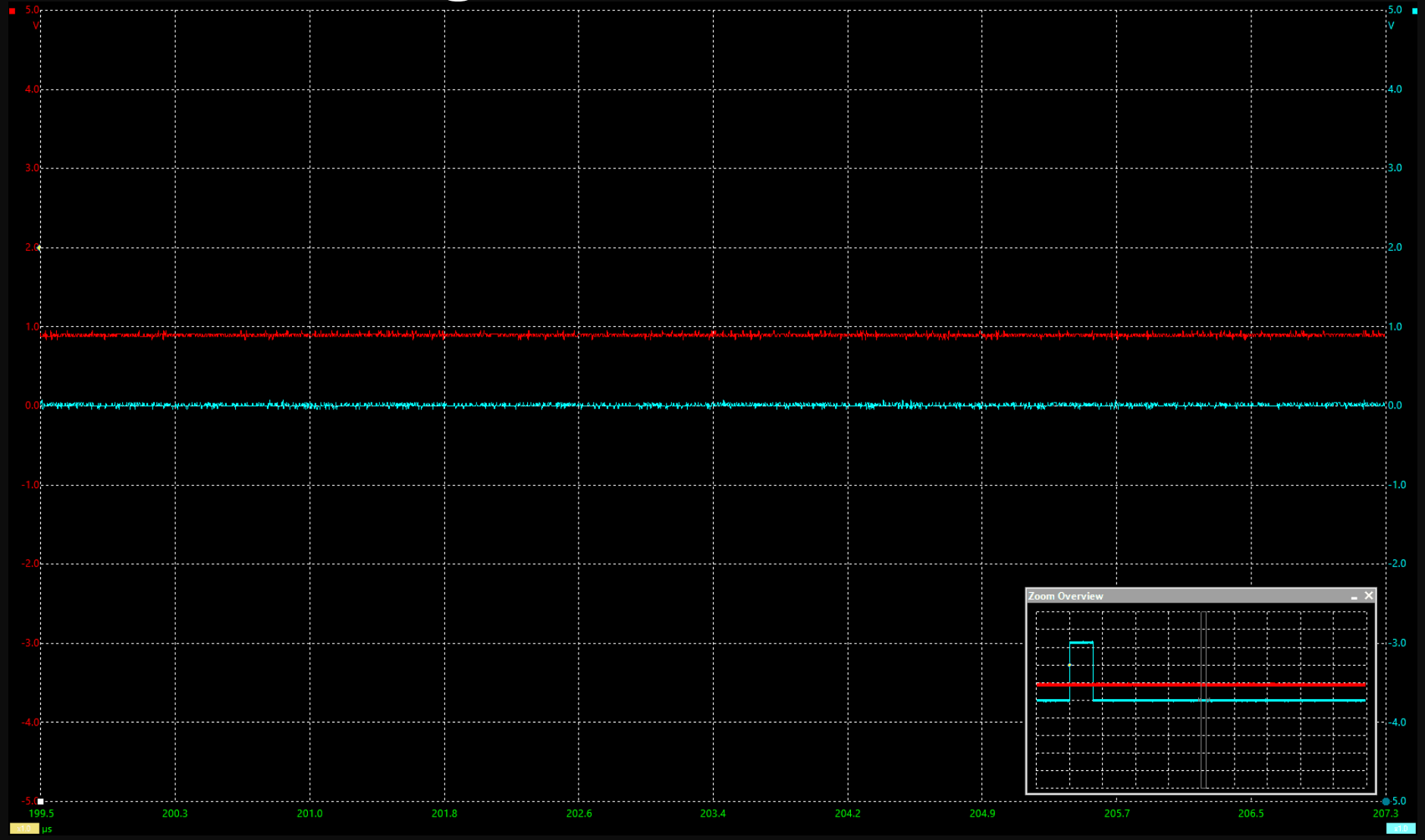
MVFI Engine



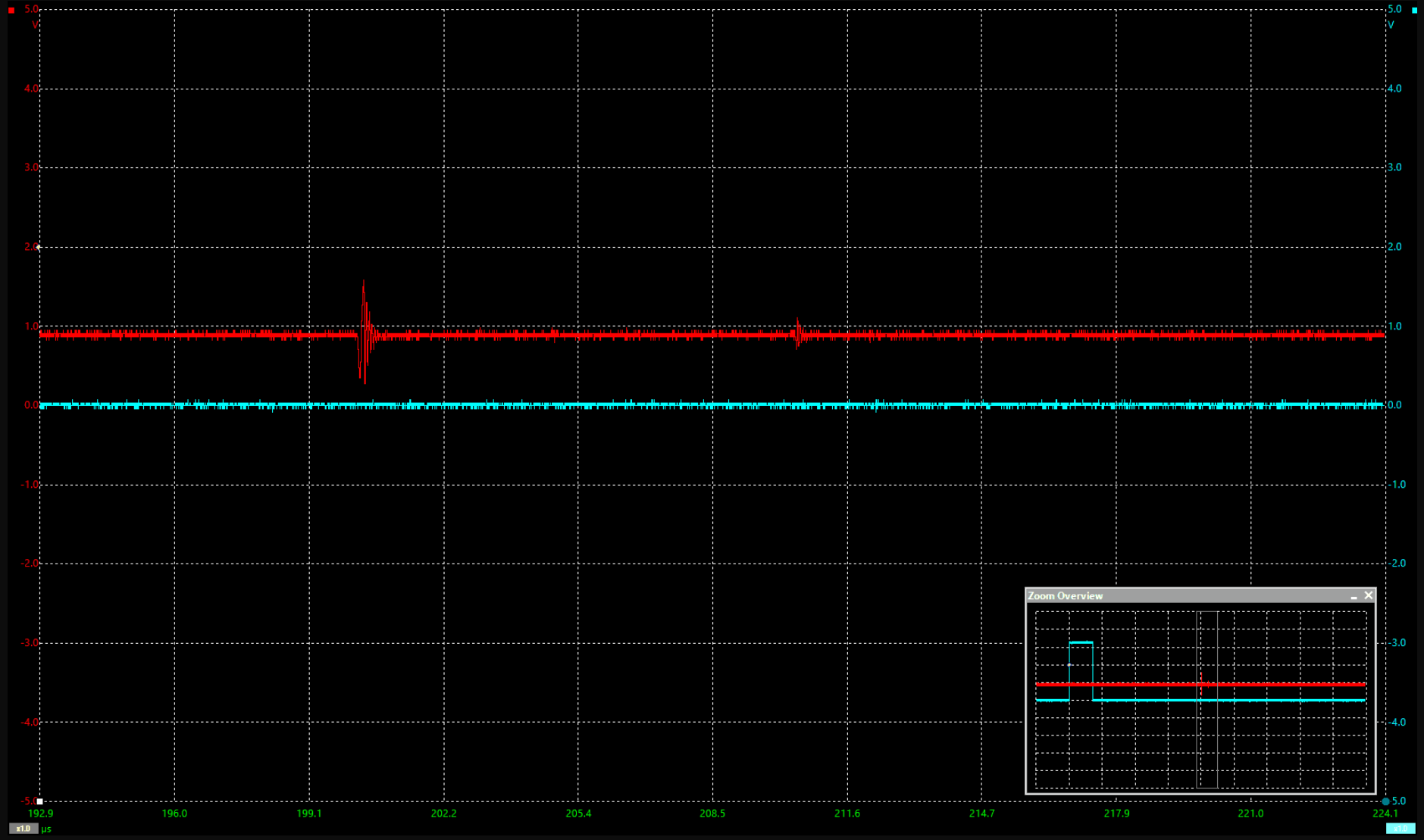
Due to the multiplexers, we are able to change the number of injected faults even after hardware synthesis

MVFI Parameter Search

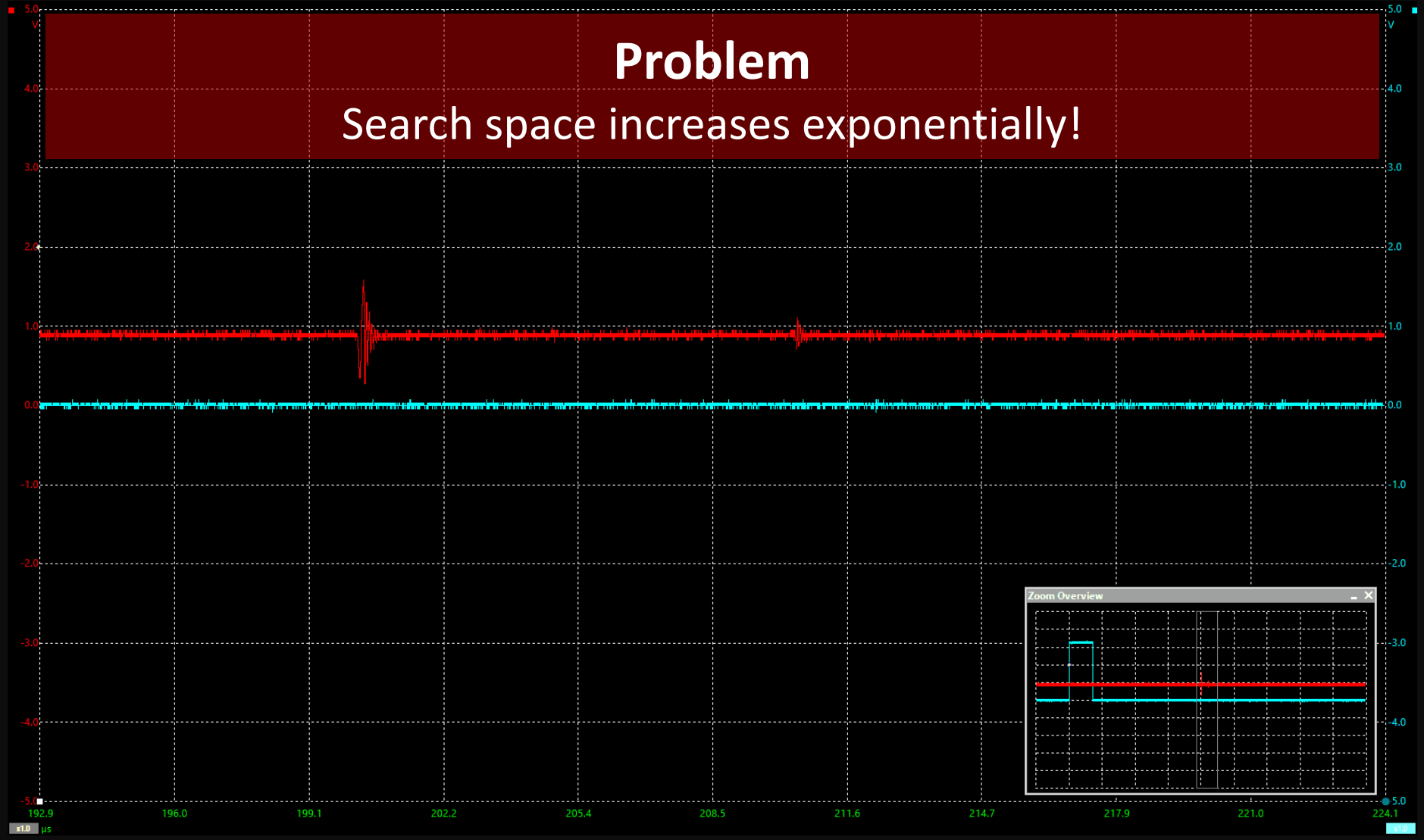
Brute-Force Single-Fault Parameter Search



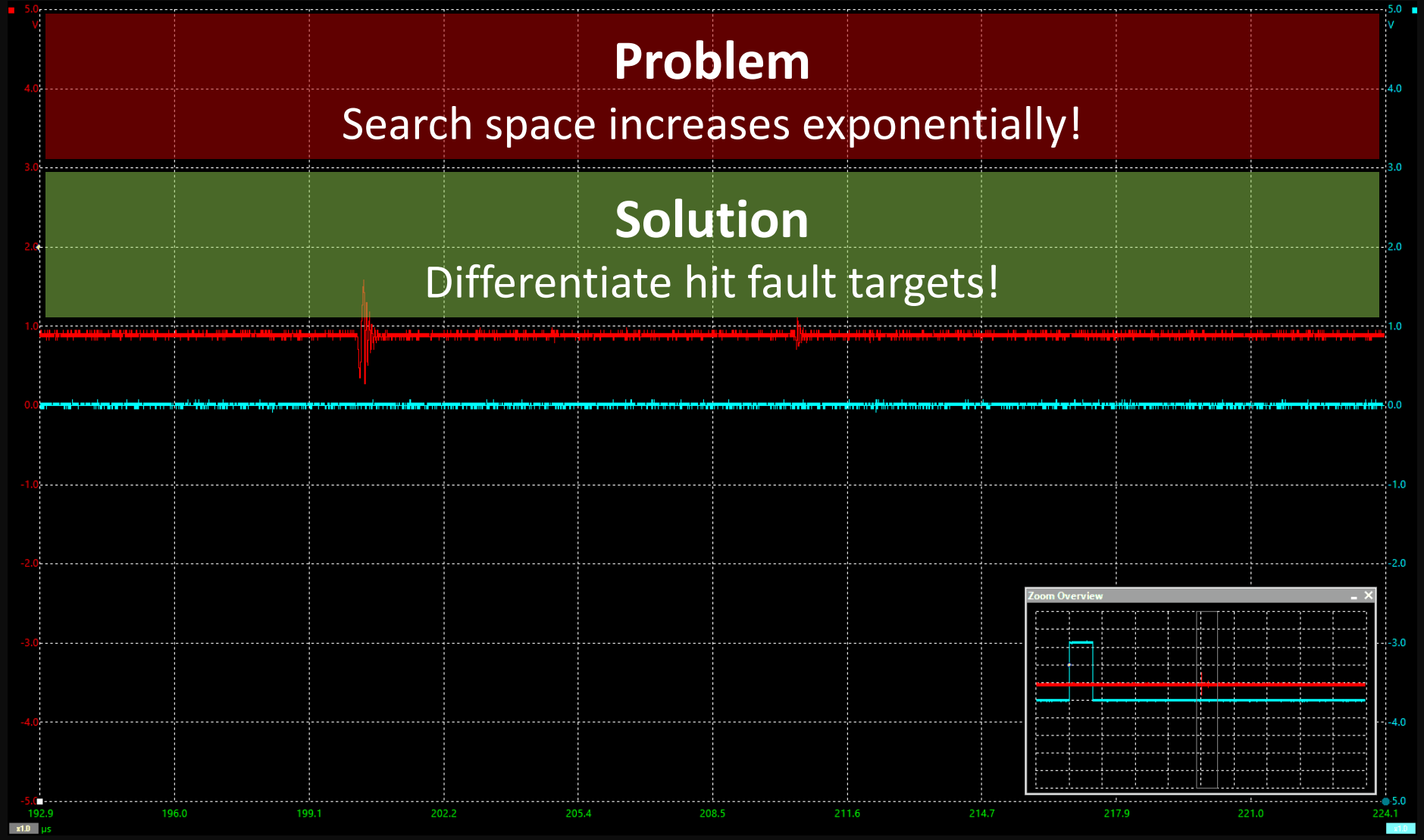
Brute-Force Double-Fault Parameter Search



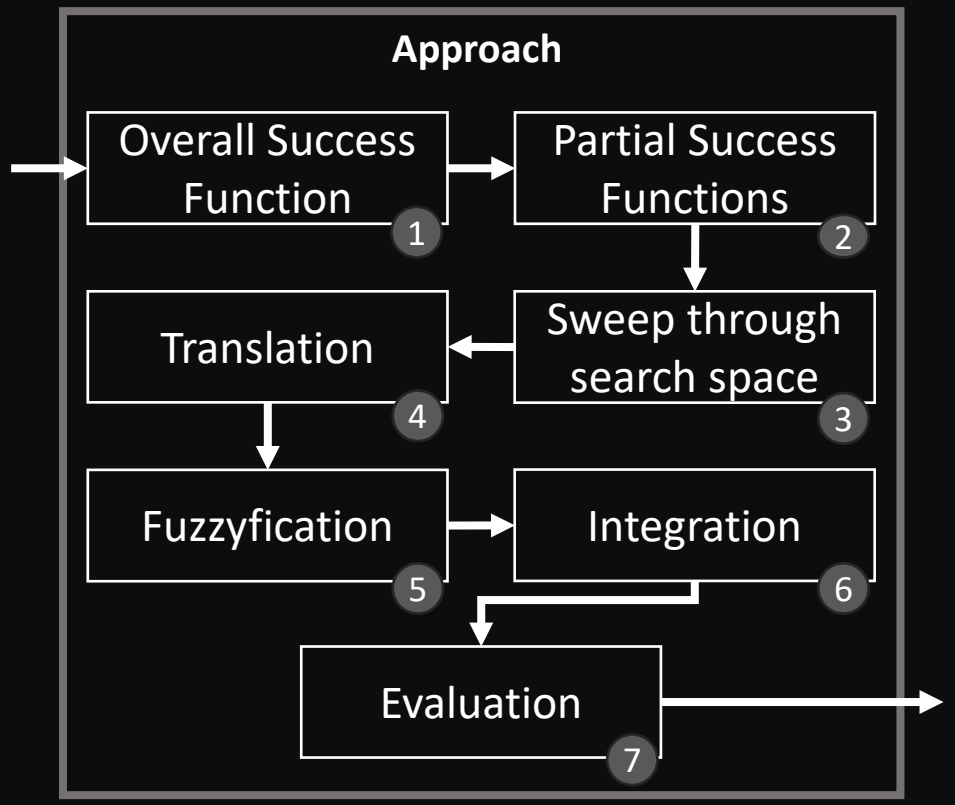
Brute-Force Double-Fault Parameter Search



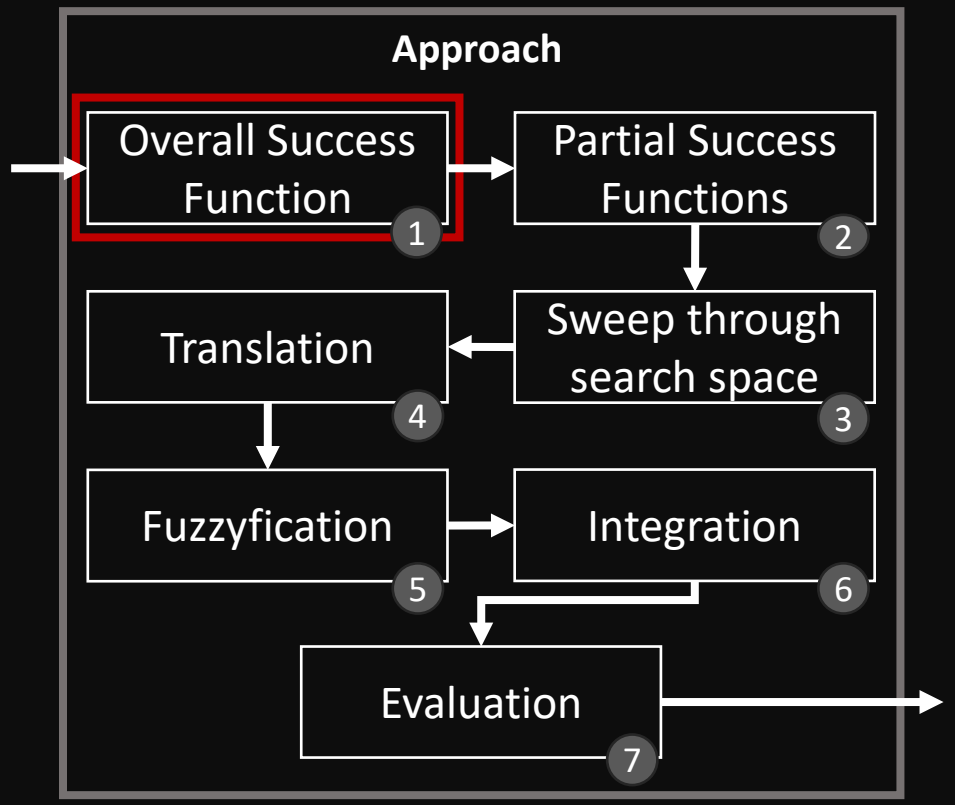
Brute-Force Double-Fault Parameter Search



Our Approach

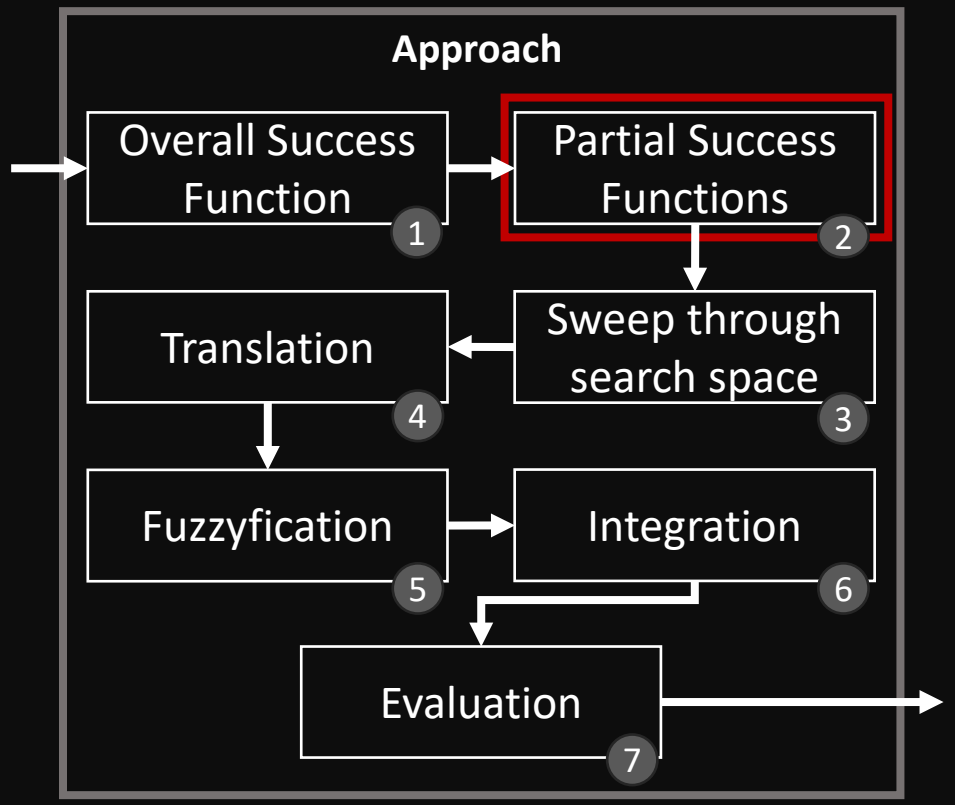


Our Approach



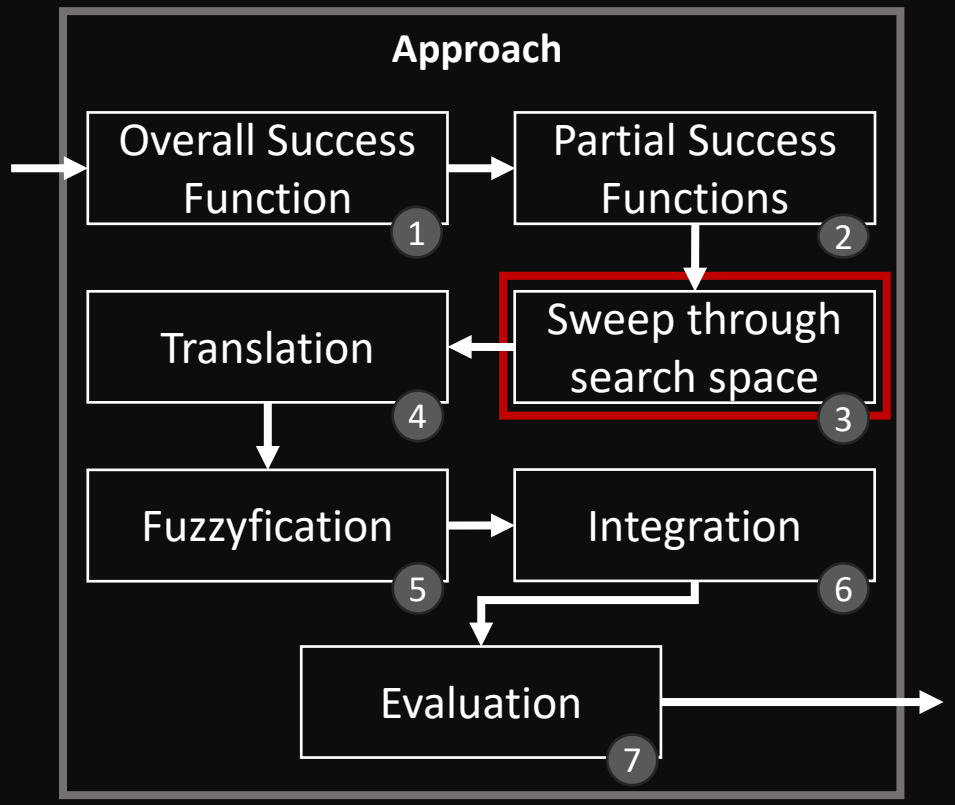
OSF determines if all fault targets have been hit (Overall Success)

Our Approach



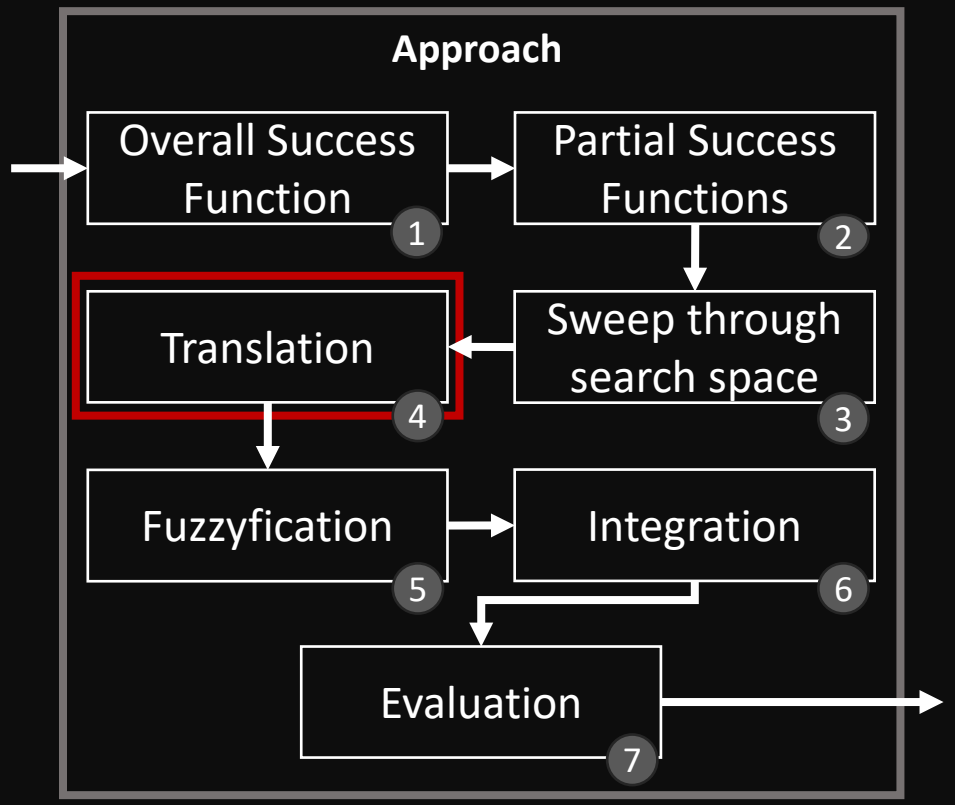
PSFs determine if a specific fault target has been hit (Partial Success)

Our Approach



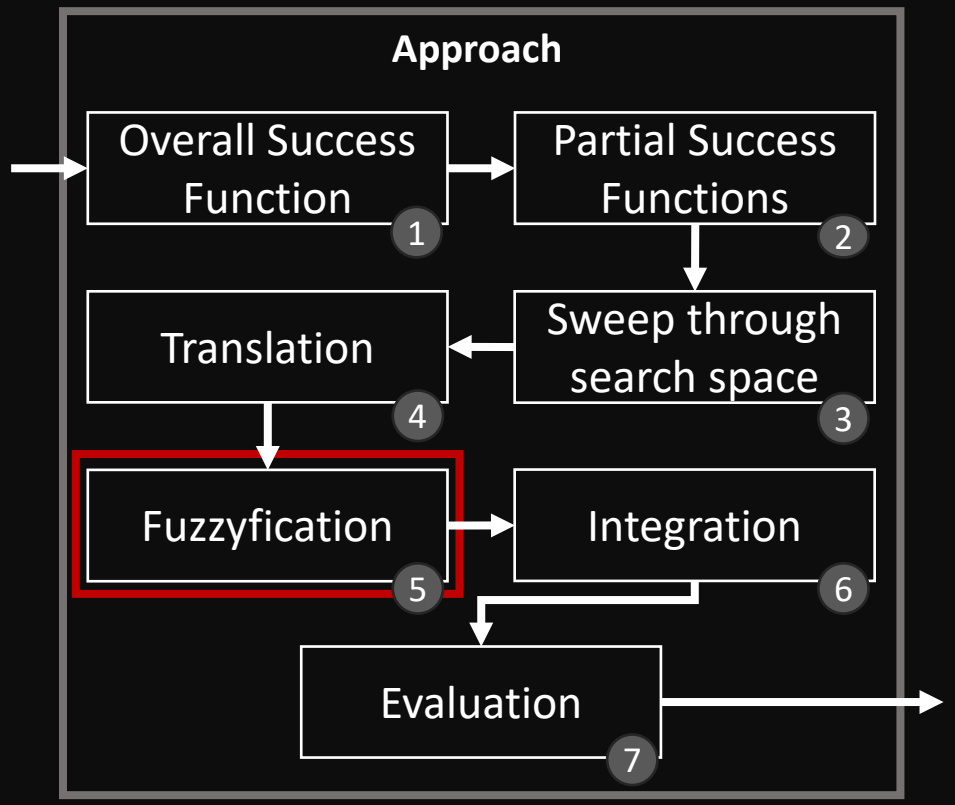
Use a single voltage fault to sweep through search space, record partial successes

Our Approach



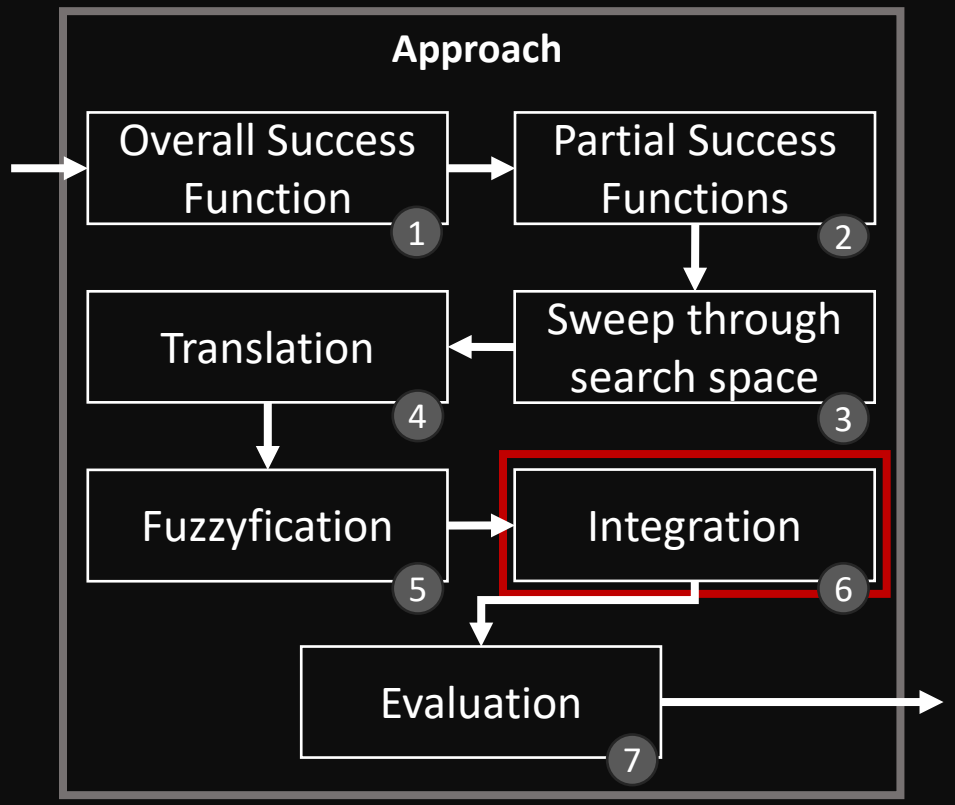
Translate absolute parameters into relative ones

Our Approach



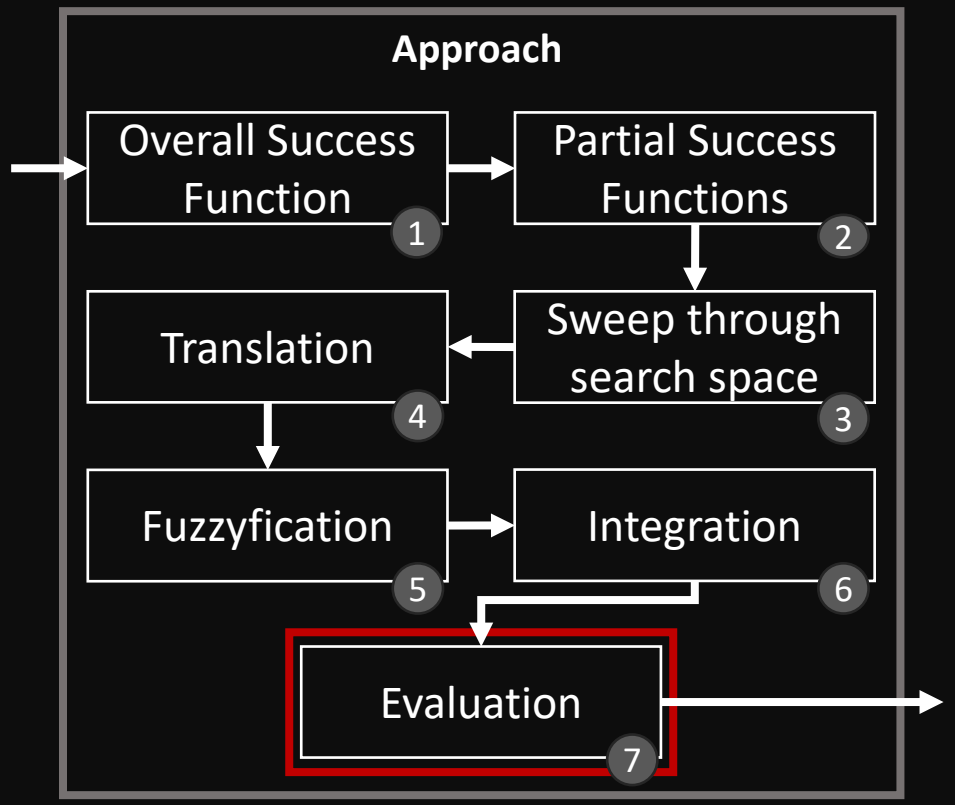
Fuzzify parameter ranges to counter uncertainty

Our Approach



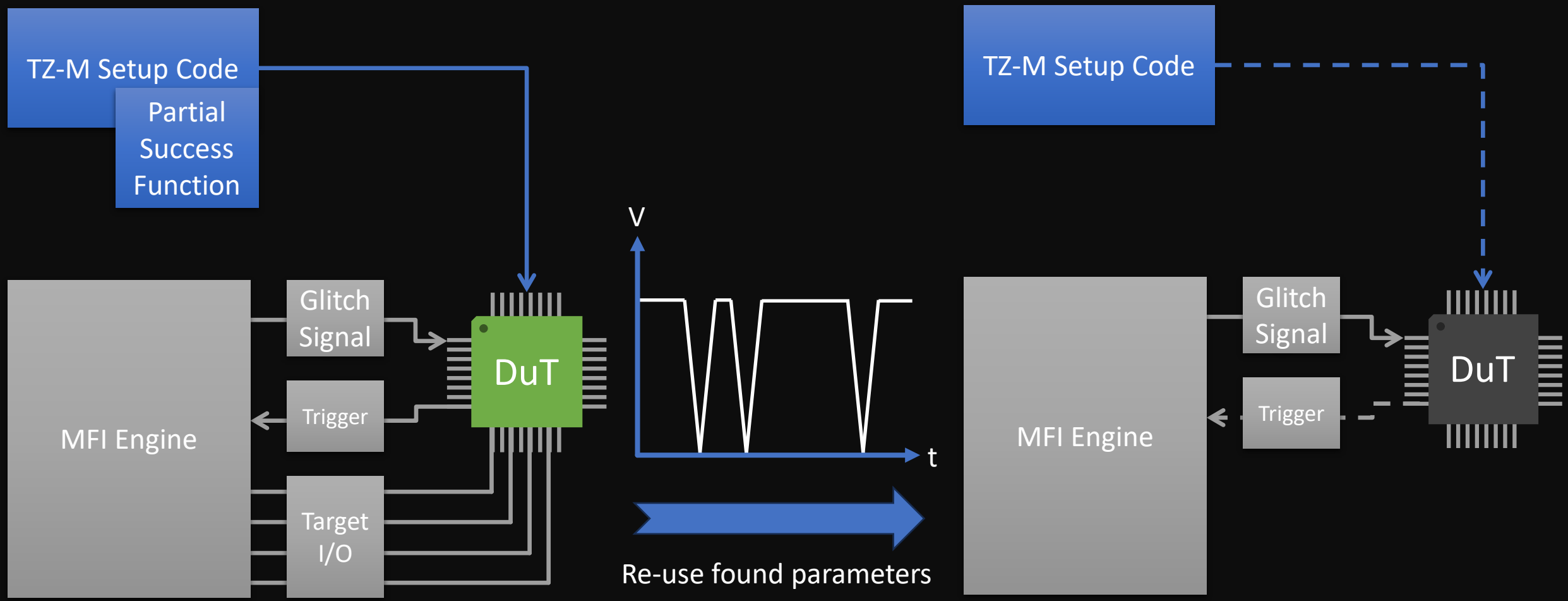
Combined brute-force search on small ranges generated by fuzzification

Our Approach



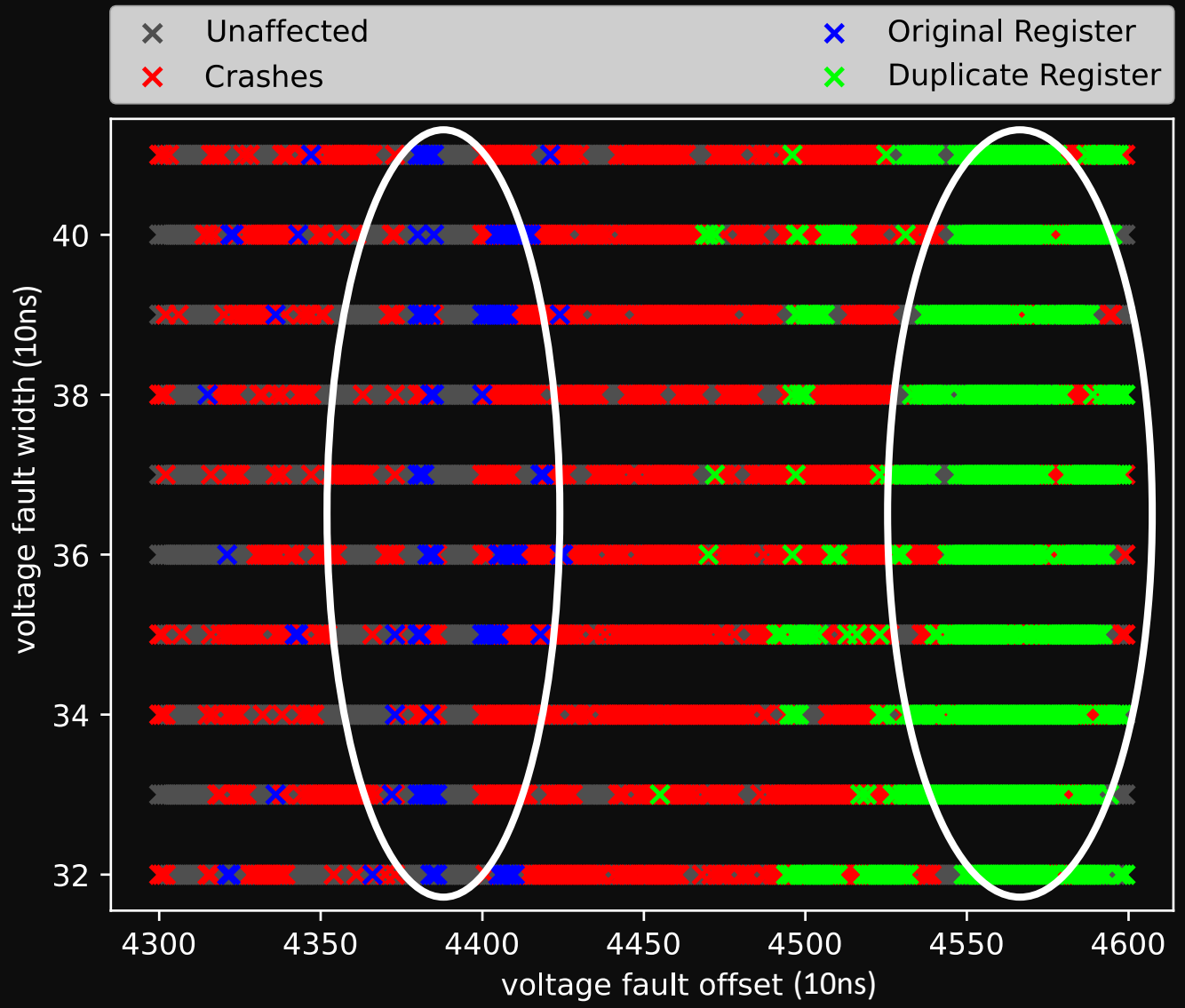
Evaluate findings from integration, check for repeatability

System Overview

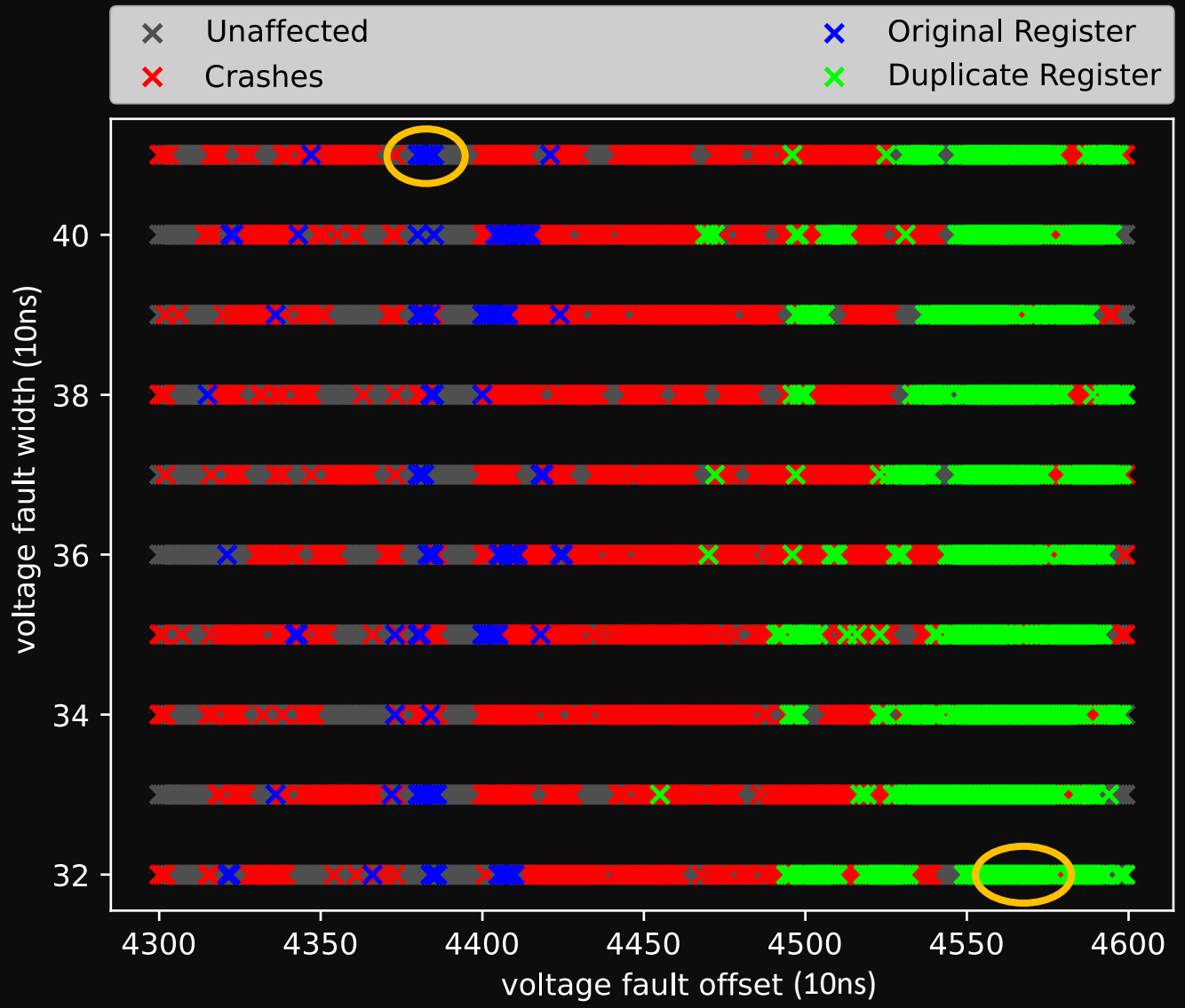


Our attack: Disabling TZ-M

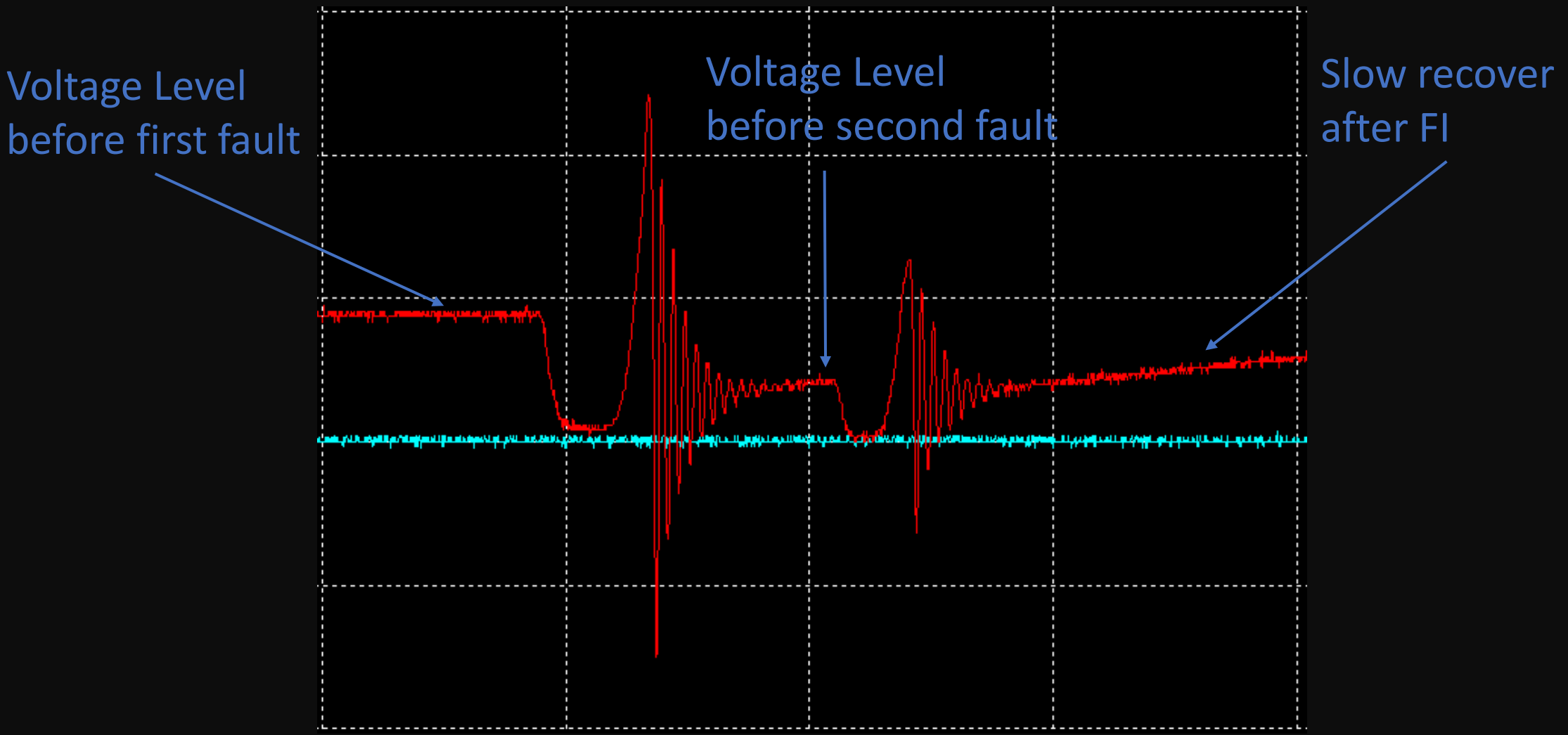
Finding Parameters



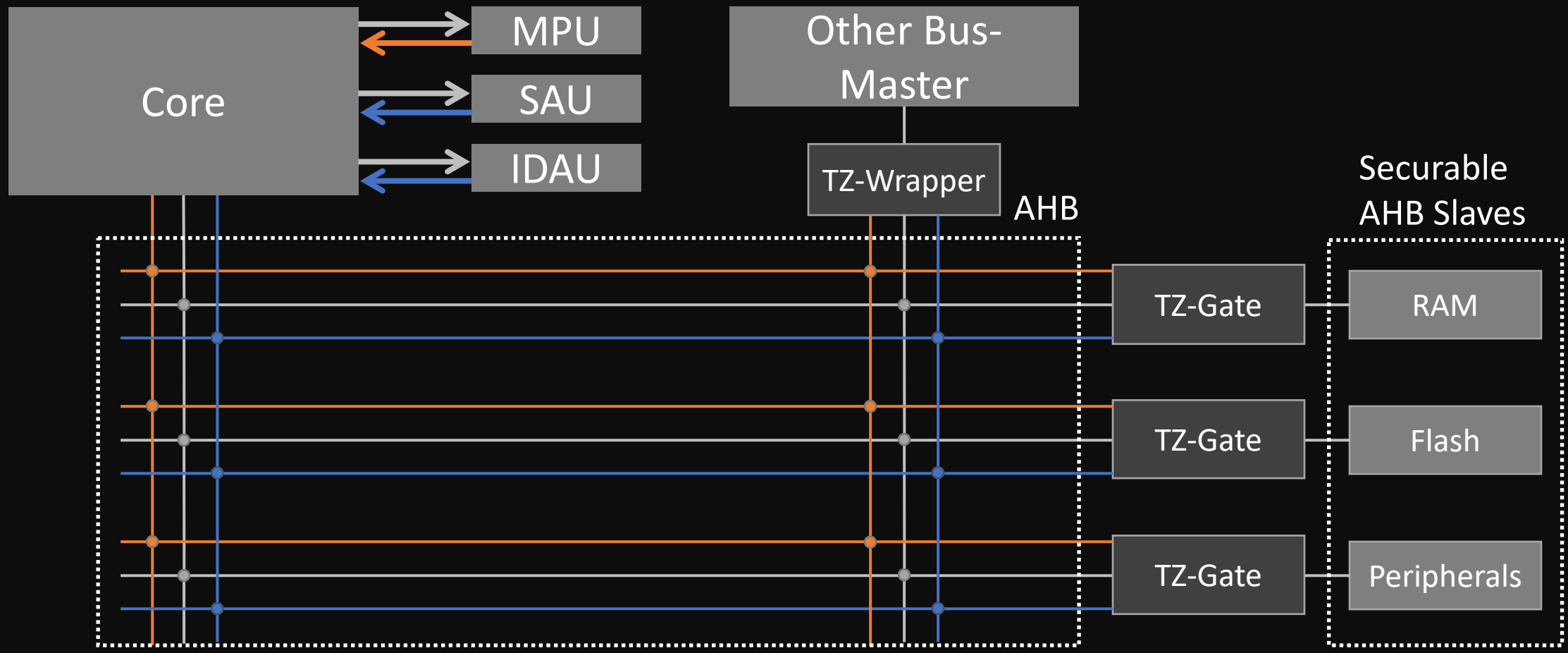
Finding Parameters



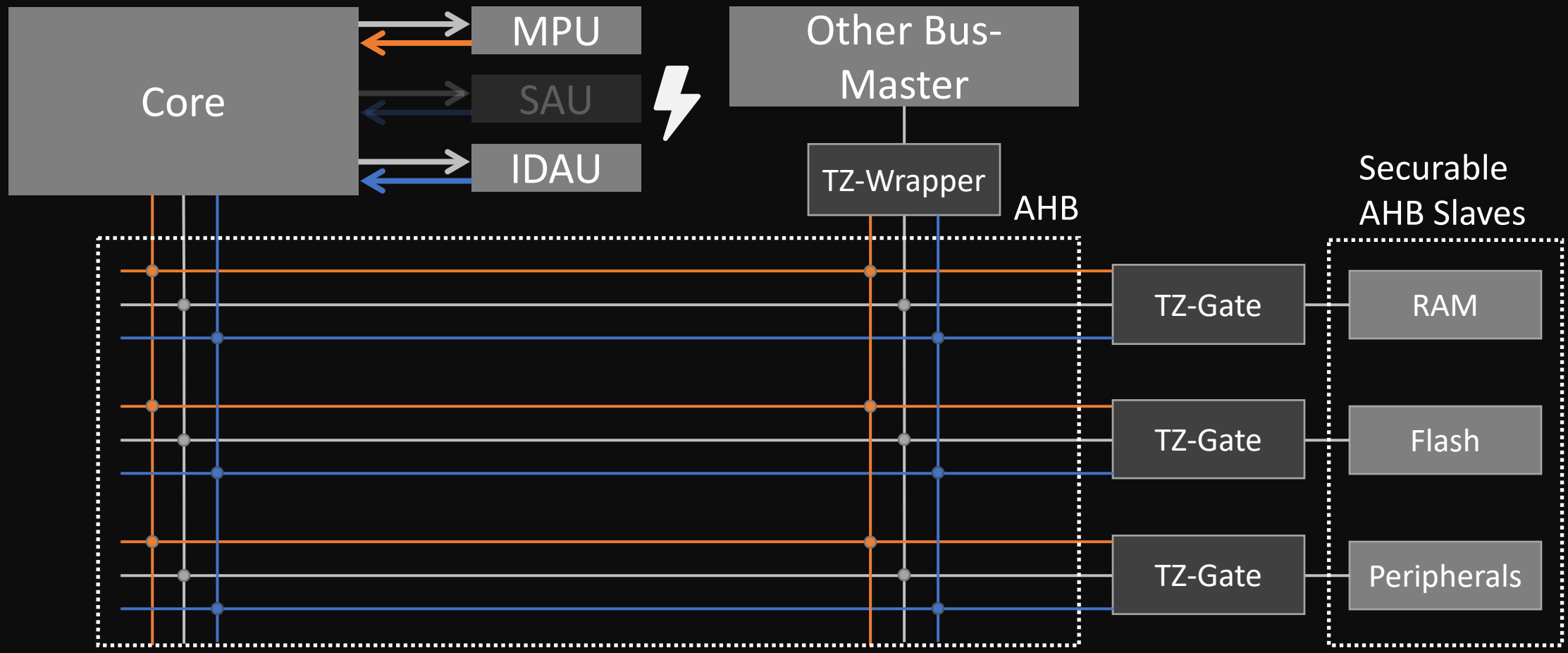
Finding Parameters



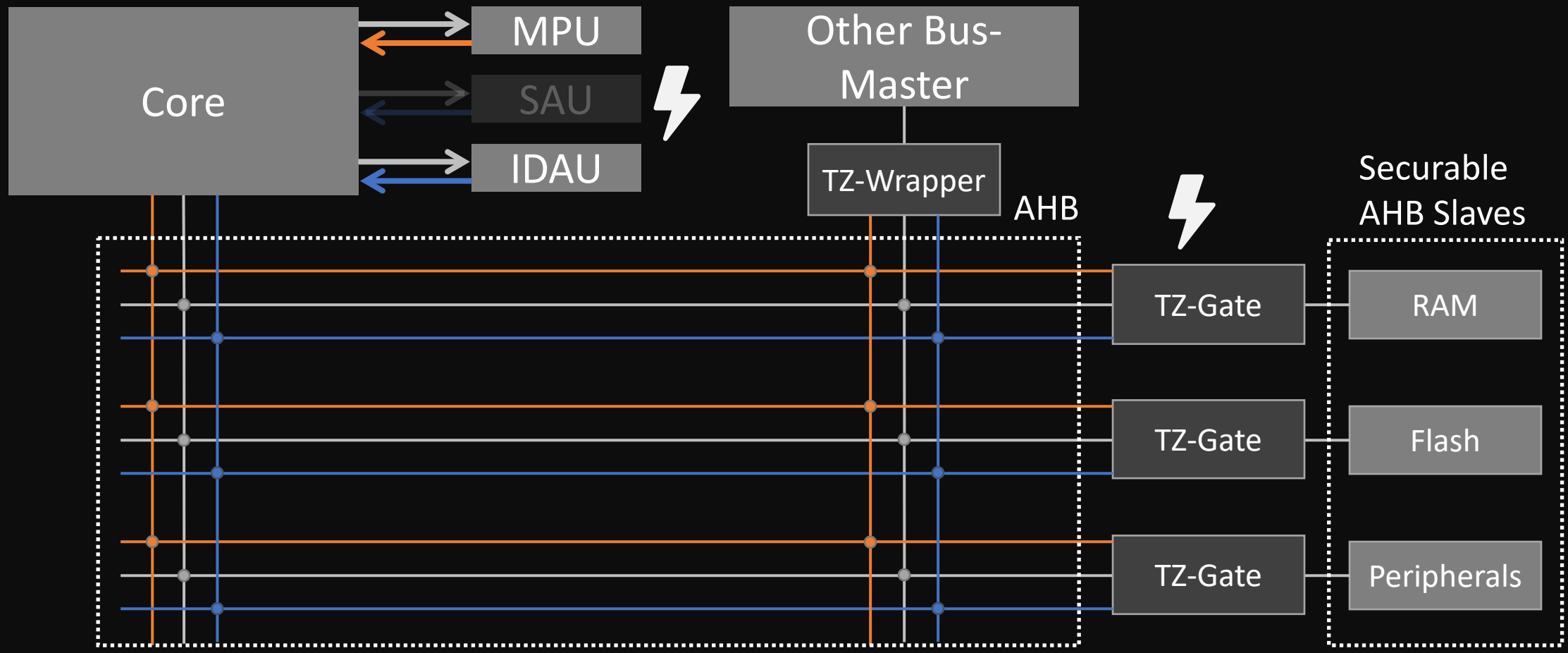
Attacking NXPs LPC55SXX & RT6XX



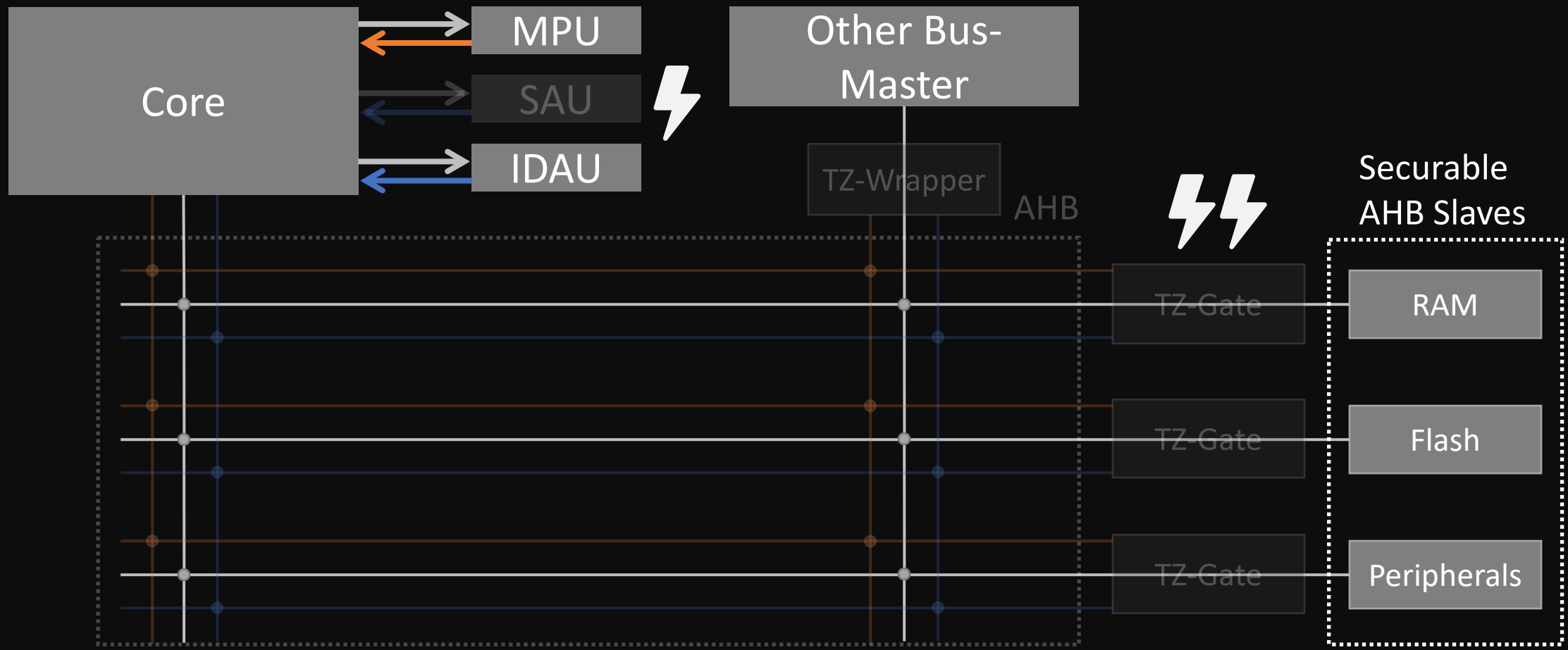
Attacking NXPs LPC55SXX & RT6XX



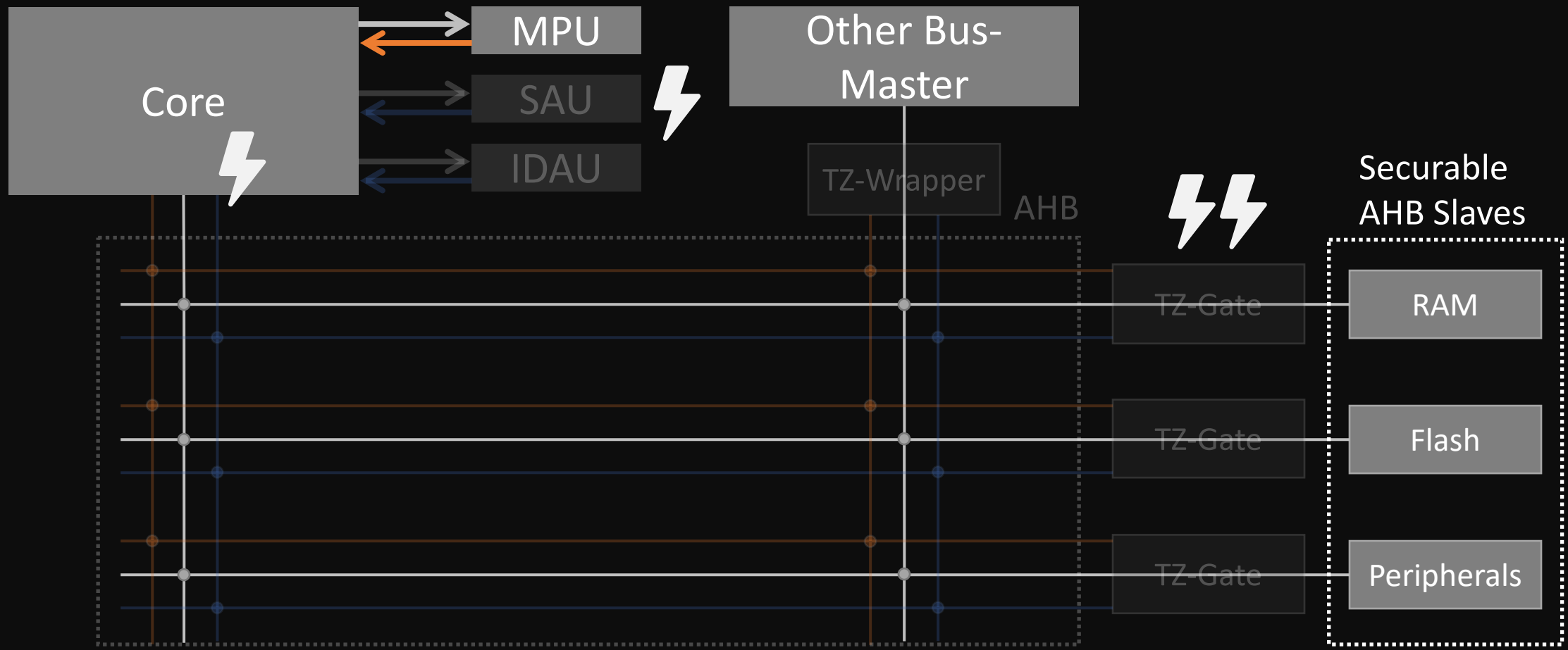
Attacking NXPs LPC55SXX & RT6XX



Attacking NXPs LPC55SXX & RT6XX

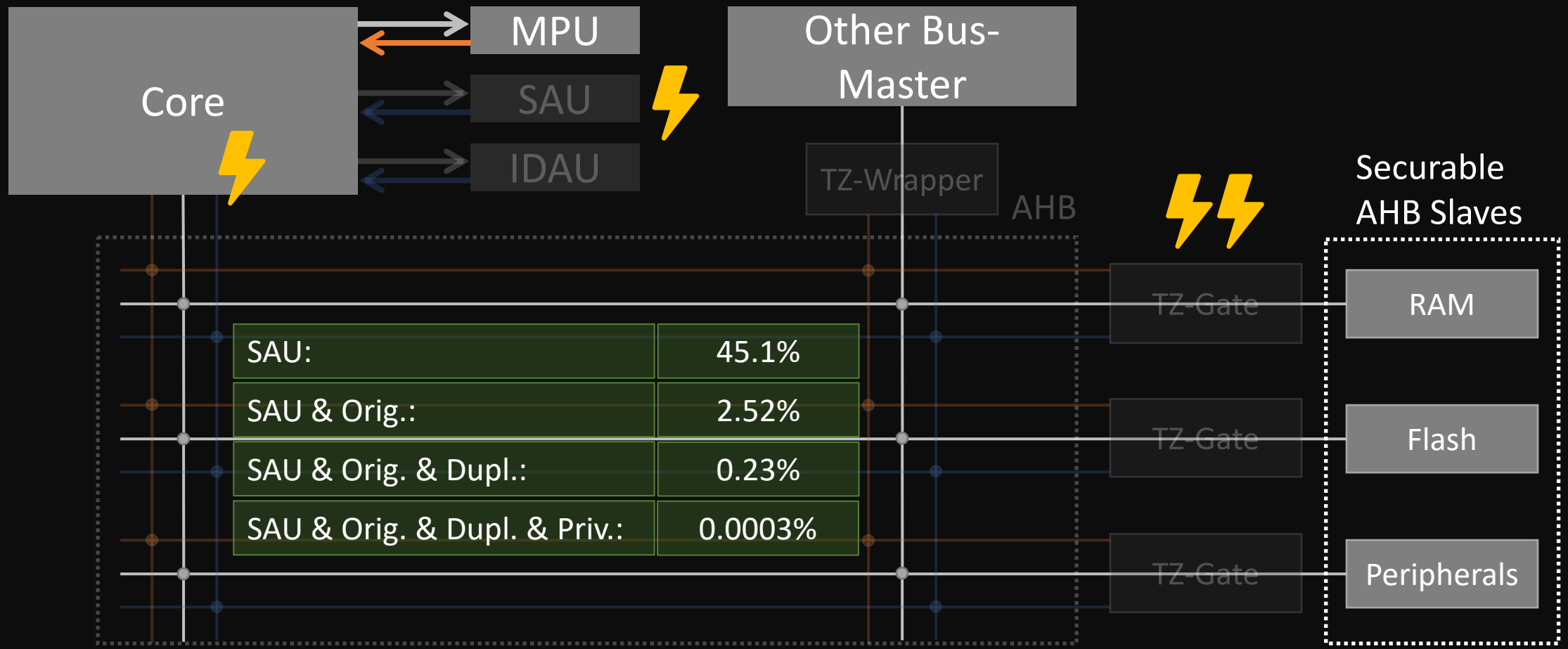


Attacking NXPs LPC55SXX & RT6XX



Address Privilege Level Security State

Attacking NXPs LPC55SXX & RT6XX



Thanks!

System
Security
Lab

TECHNISCHE
UNIVERSITÄT
DARMSTADT