riscure

driving your security forward

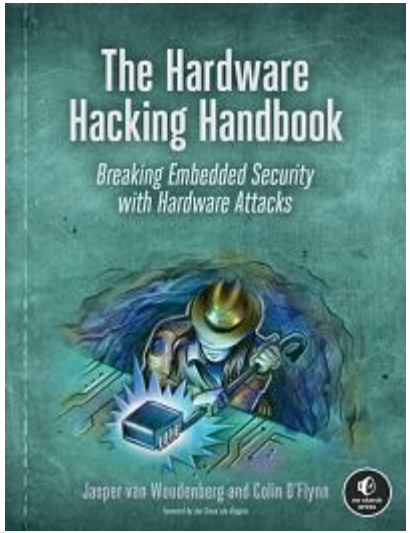
EMBEDDED SECURITY IN A 5G WORLD

JASPER VAN WOUDENBERG

ABOUT JASPER

CTO Riscure North America
Author “Hardware Hacking Handbook”

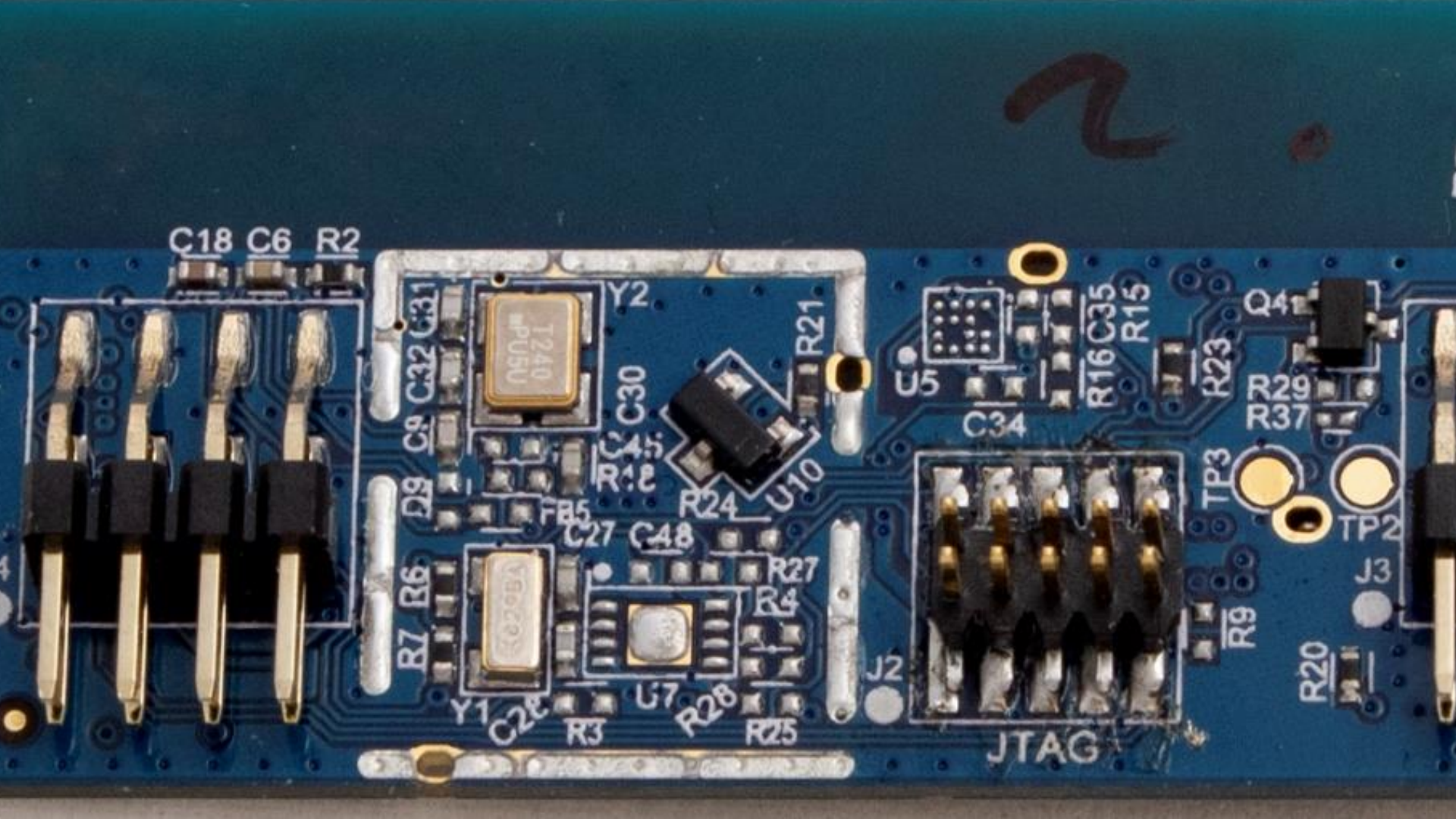
- Software security since y2k
- Riscure: SCA/FI/Hardware security since 2005
- Author since 2021
- Grey hair since ~~2015~~ becoming a father

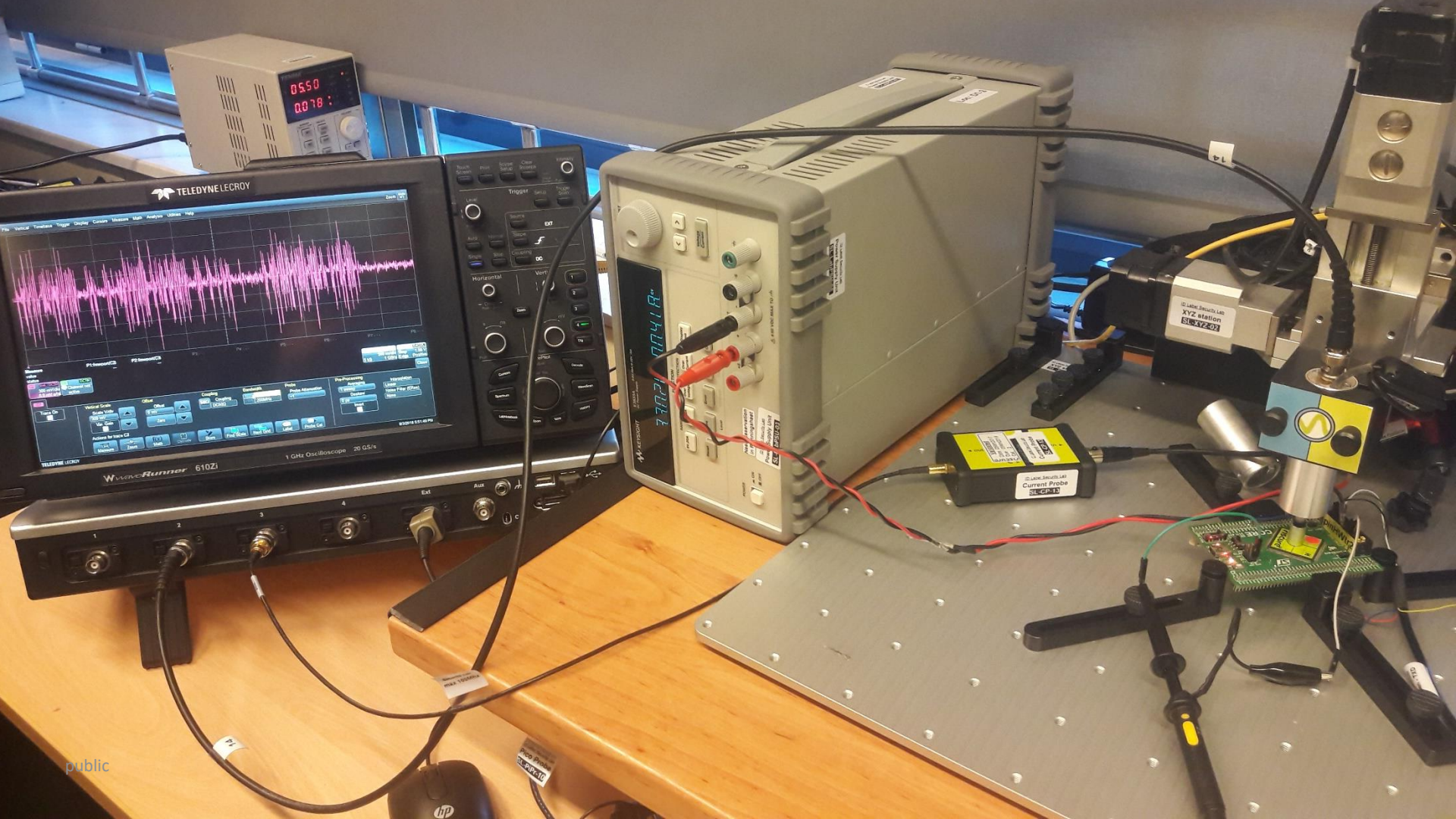


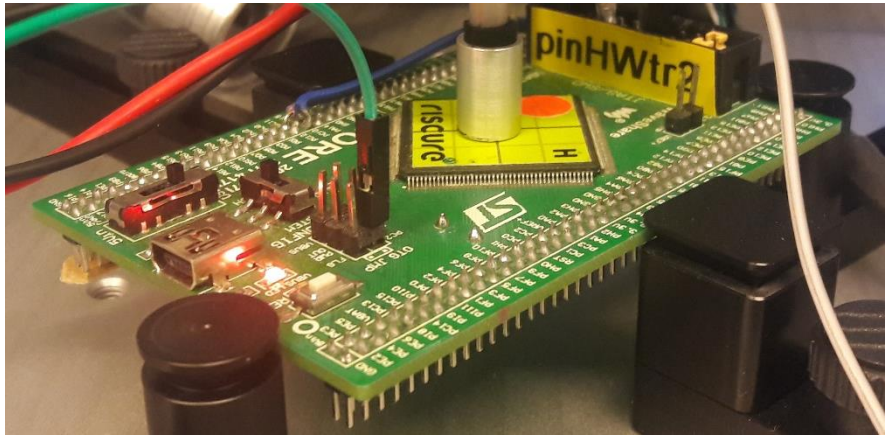
https://www.hardwarehacking.io/

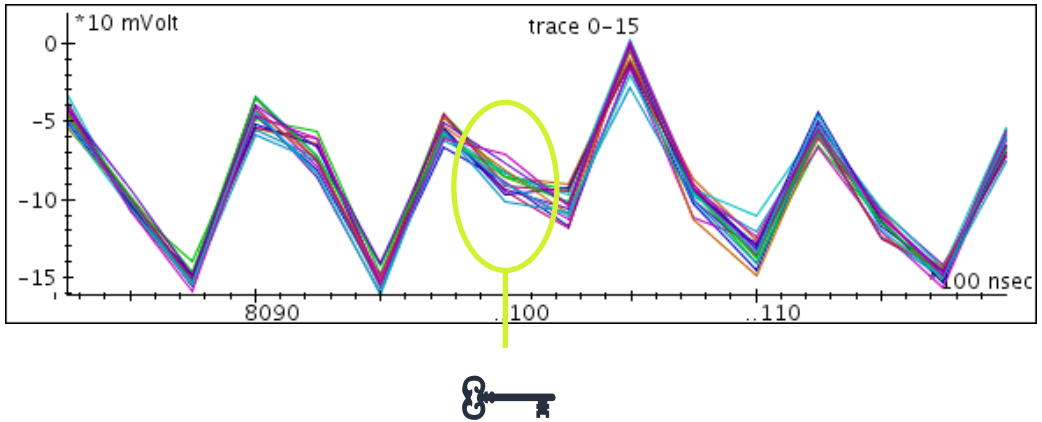
https://linkedin.com/jaspervw
https://twitter.com/@jzvw

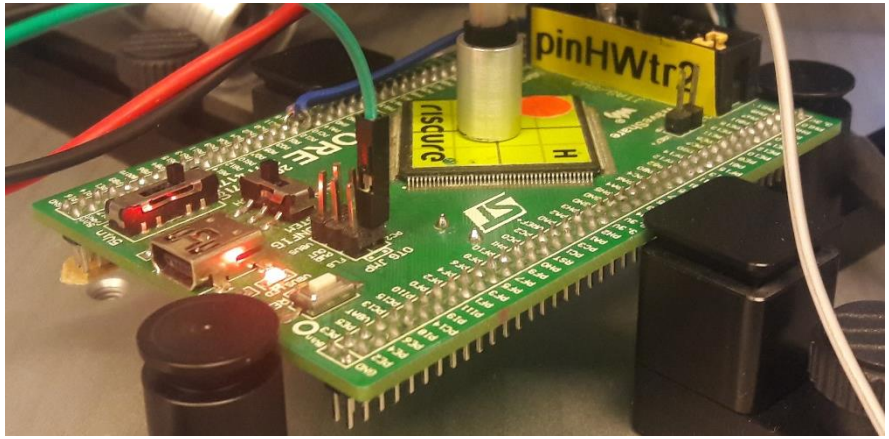
TYPES OF HARDWARE ATTACKS


C18
C6
R2
Y2
C31
C32
C9
C30
R21
C45
R24
U10
U5
C34
C35
R15
R16
R23
Q41
R29
R37
TP3
TP2
J3
R9
R20
D9
FB5
C27
C48
R27
R4
R6
R7
Y1
C26
R3
U7
R28
R25
J2
JTAG


TELEDYNE LECROY
WaveRunner 610Zi
1 GHz Oscilloscope 20 GS/s
Current Probe
SL-CP-13
XYZ station
SL-XYZ-20

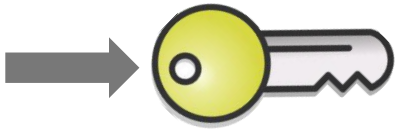

pinHWtr

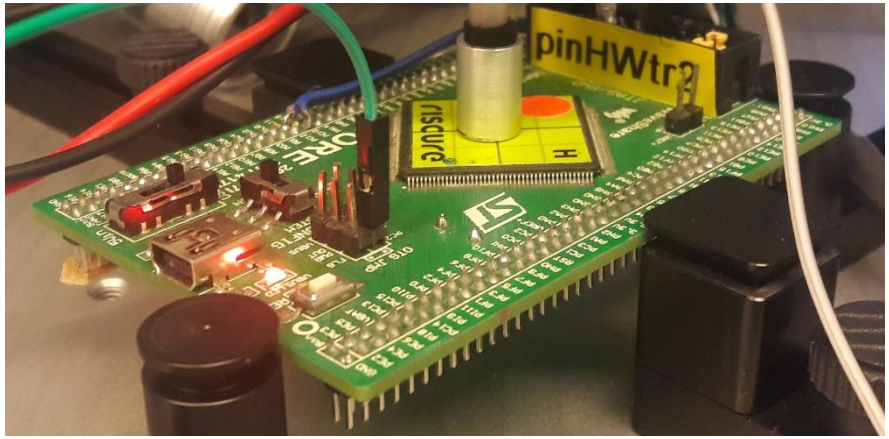

*10 mVolt
trace 0-15
0
-5
-10
-15
8090
..100
..110
100 nsec

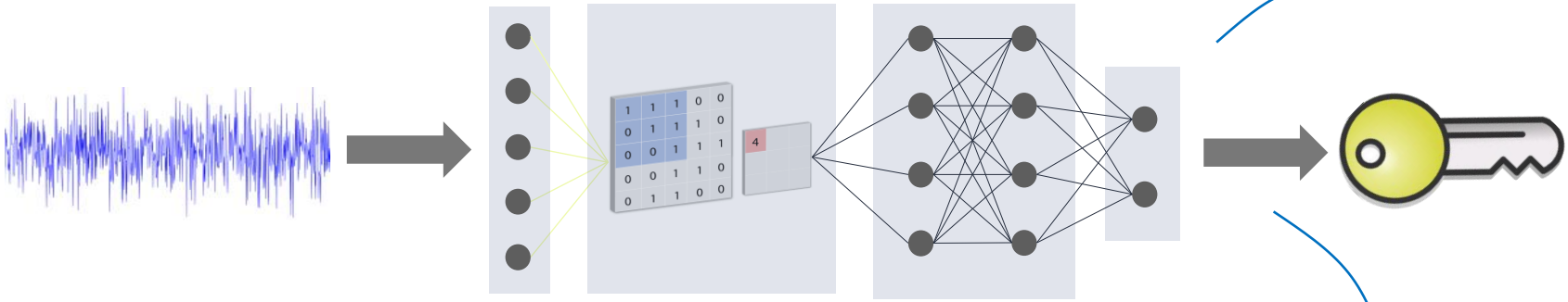


Signal processing

Leakage modeling




pinHWtr2


Leakage info
Metrics

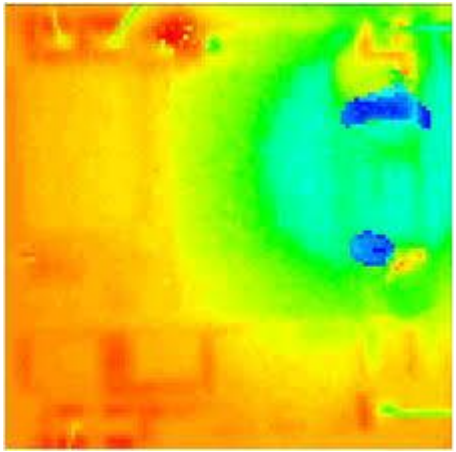
Advantages Of The Reset Glitch Chip

Budget Priced $199 installed with a 500Gig Hard drive

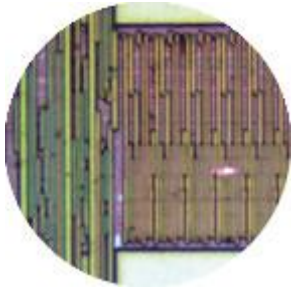
Custom Dashboards MAME Arcade Games Homebrew Software Emulators

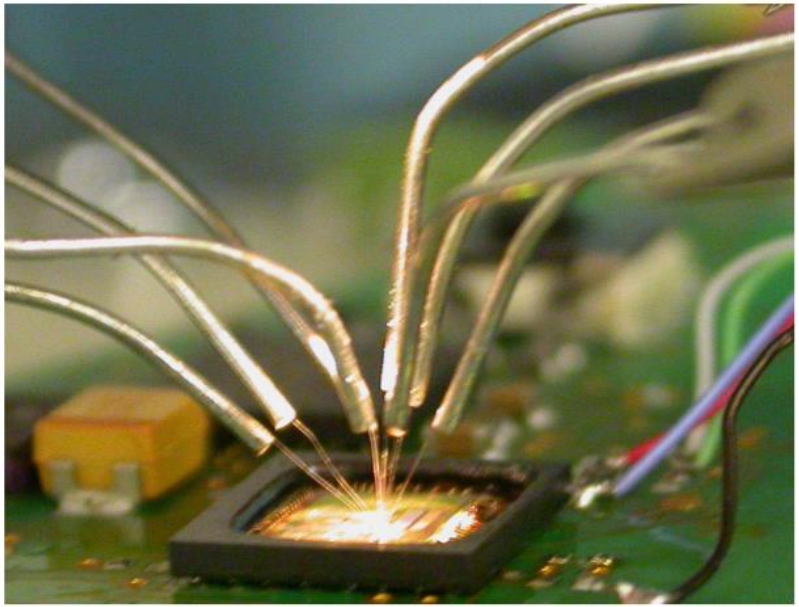
Can easy be Installed by a 10 year old Child

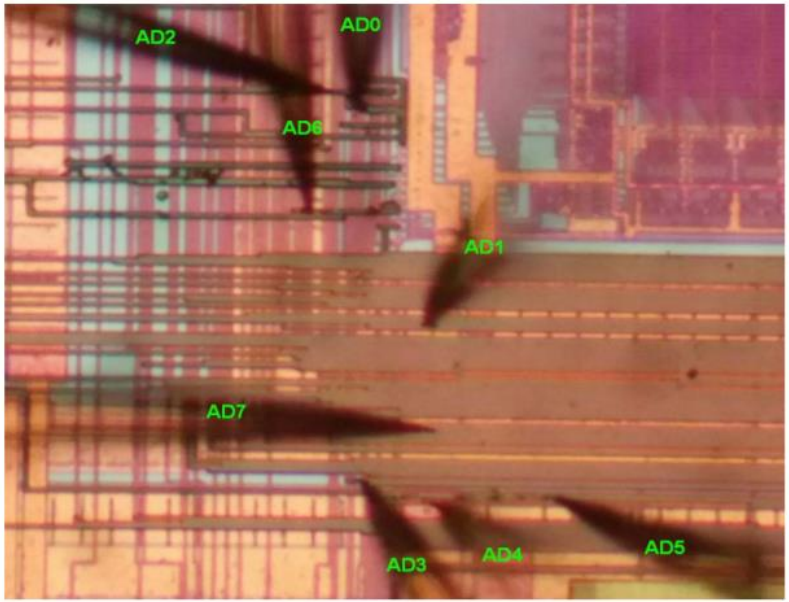
http://www.ausgamestore.com/xbox-360-slim/matrix-glitcher-slim.html




riscure
riscure
VC Glitcher







https://www.cl.cam.ac.uk/~sps32/ahsa2017_prob.pdf

HOW MUCH SECURITY DO WE NEED?

WHY DO WE NEED SECURITY AT ALL?

- Brand risk
- Monetary loss

Internal incentives

- Health & safety
- Certification & regulatory
- National security

External incentives

WHAT DOES 5G MEAN FOR CHIP SECURITY?

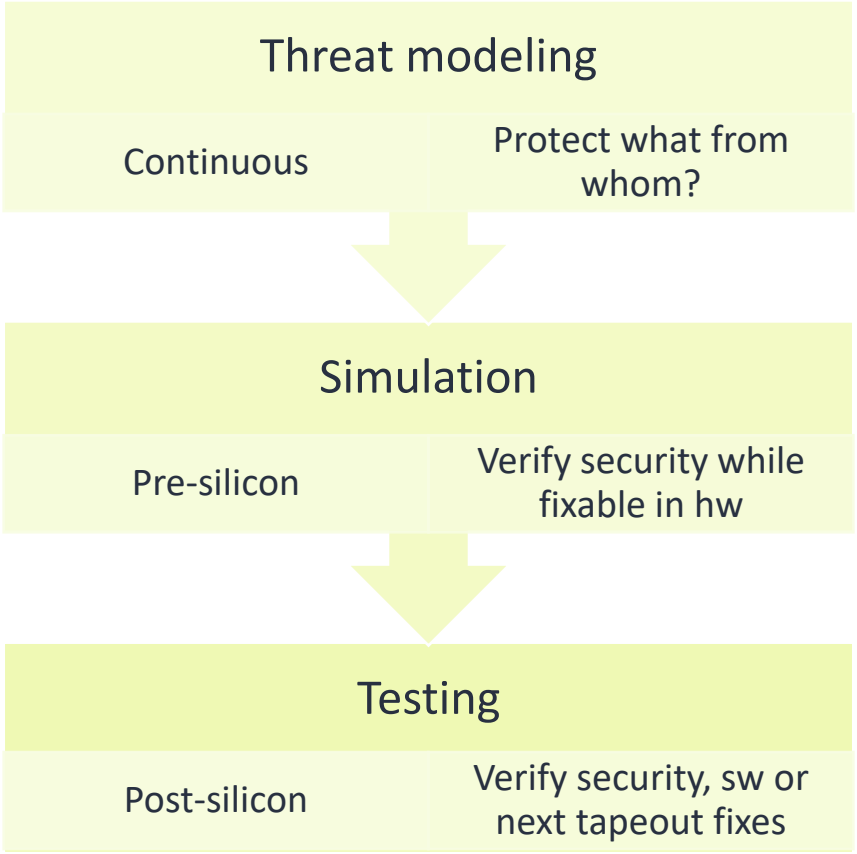
Attackers:

- Garage-level hobbyist
- Industry competition
- Hacktivism
- Criminal enterprise
- Nation states

Rating security:

https://www.sogis.eu/documents/cc/domains/sc/JIL-Application-of-Attack-Potential-to-Smartcards-v3-1.pdf

HARDWARE SECURITY PROCESS



HARDWARE SECURITY PROCESS

Threat modeling

Continuous

Protect what from whom?

Simulation

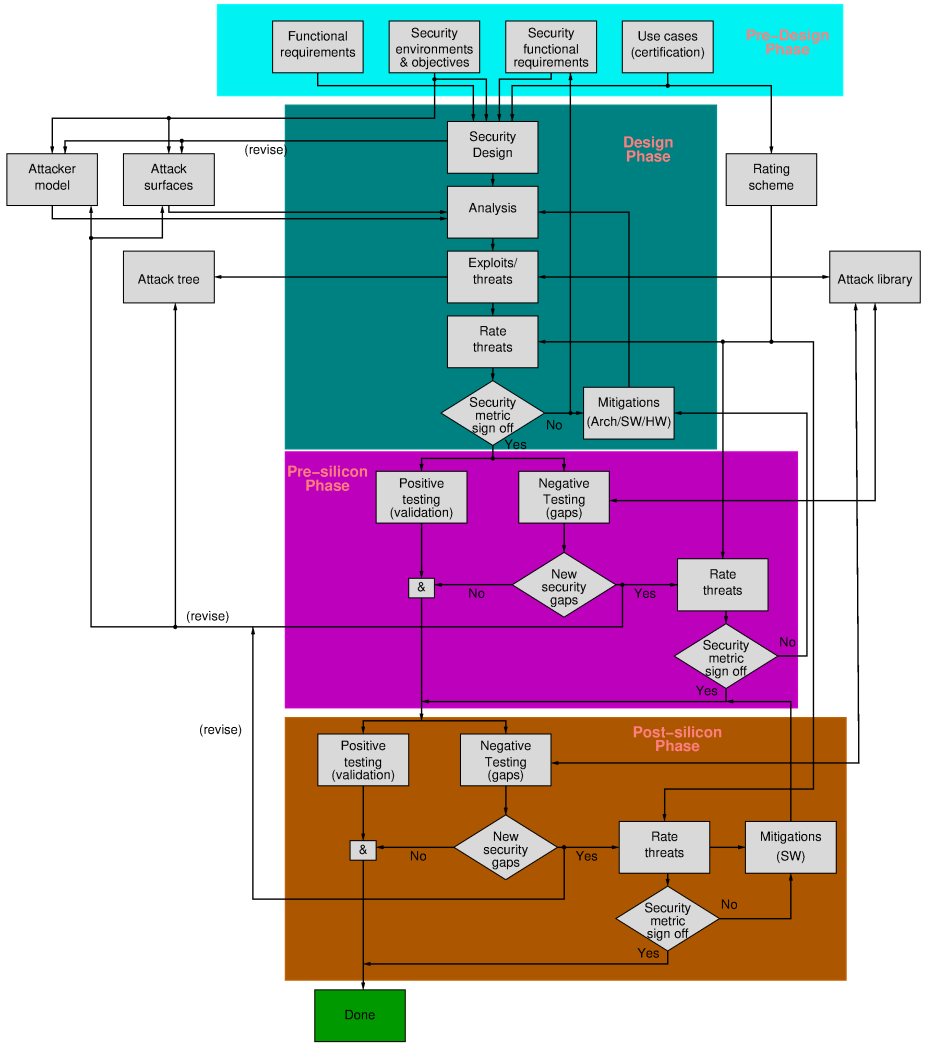
Pre-silicon

Verify security while fixable in hw

Testing

Post-silicon

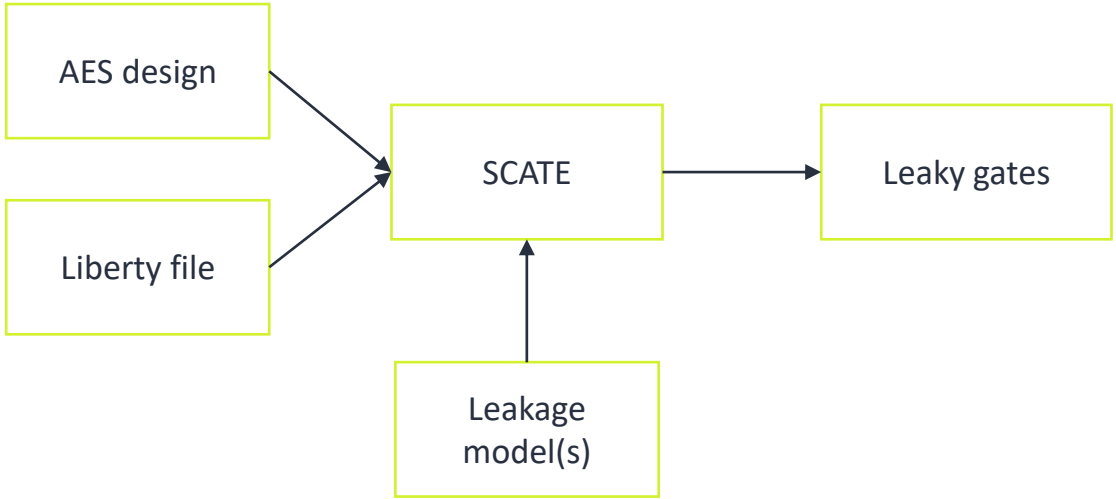
Verify security, sw or next tapeout fixes



PRE-SILICON CASE STUDY:

MASKED AES DESIGN

SCA SIMULATION: SCATE



Patented leaky gate identification

POWER TRACE GENERATION RUNTIME SUMMARY

AES Core	RTL Gate Count	Synthesis Gate Count	P&R Gate Count	RTL Trace Gen. Time	Synthesis Trace Gen. Time	P&R Trace Gen. Time
aes_256_serial_wddl	20991	19038	20991	1:20:21	3:54:20	4:03:43
aes_256_serial	6808	5849	7136	0:38:23	1:52:03	2:22:33
picoaes	10663	8765	9833	0:29:38	1:48:00	2:44:35
aes_256	278800	275412	297010	1:59:06	6:34:21	12:55:57
aes_fast_mix_sub	31321	25455	27627	1:18:15	2:32:13	3:02:21
aes_sbox_scramble2	32514	28928	31192	3:20:11	3:44:57	4:35:44
femtoaes	4274	3442	4177	1:01:53	1:50:10	2:05:47
aes_128_serial	5303	4335	5101	1:02:58	1:39:51	1:55:03

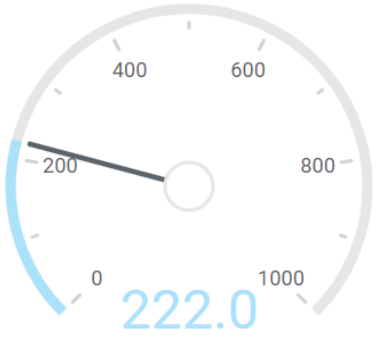
- HDL simulation of 256 traces – 12x parallel
- Dominant runtimes: place and route, power analysis
- Time per trace improves slightly with increased power trace count

SCATE caes_sky130_nokeep_nokeep_hierarchy

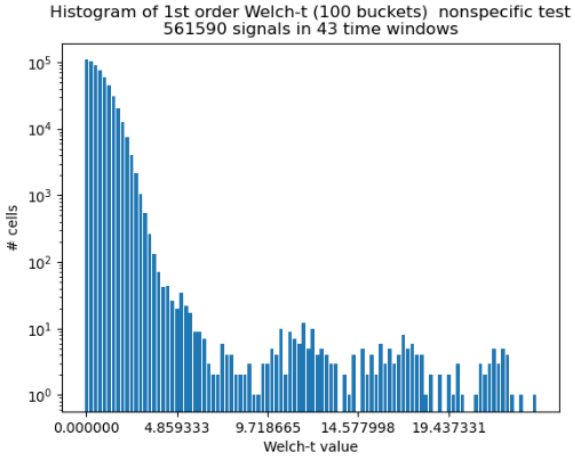
Project

Overview | Leaky signals | Leaky elements | Source | VCD | Trace

Leaky signals



Max. leakage



Overview | Leaky signals | Leaky elements | Source | VCD | Trace

Select all

	signal1	rank_sum	sum
	filter data...		
☐	test_aes_128.uut._10691_._1_	1	68.82745845829957
☐	test_aes_128.uut._03304_	2	68.82745845829957
☐	test_aes_128.uut._10690_.Y	3	68.82745845829957
☐	test_aes_128.uut._10691_.B	4	68.82745845829957
☐	test_aes_128.uut._10690_._4_	5	68.82745845829957
☐	test_aes_128.uut._11132_._4_	6	68.27002063177406
☐	test_aes_128.uut._03746_	7	68.27002063177406
☐	test_aes_128.uut._11132_.Y	8	68.27002063177406
☐	test_aes_128.uut._11133_.B	9	68.27002063177406
☐	test_aes_128.uut._11133_._1_	10	68.27002063177406

« ‹ 1 / 23 › »

SCATE caes_sky130_nokeep_keep_hierarchy Project

Overview | Leaky signals | Leaky elements | Source | VCD | Trace

Select all

	signal1	rank_sum	sum
	filter data...		
☑	test_aes_128.uut.round.sbox[0].sbe.inv.dinv._122_._1_	1	71.20231574599406
☑	test_aes_128.uut.round.sbox[0].sbe.inv.dinv._043_	2	71.20231574599406
☑	test_aes_128.uut.round.sbox[0].sbe.inv.dinv._121_.Y	3	71.20231574599406
☑	test_aes_128.uut.round.sbox[0].sbe.inv.dinv._122_.B	4	71.20231574599406
☑	test_aes_128.uut.round.sbox[0].sbe.inv.dinv._121_._4_	5	71.20231574599406
☑	test_aes_128.uut.round.sbox[0].sbe.inv._179_	6	70.70803634553815
☑	test_aes_128.uut.round.sbox[0].sbe.inv._497_.Y	7	70.70803634553815
☑	test_aes_128.uut.round.sbox[0].sbe.inv._498_.B	8	70.70803634553815
☑	test_aes_128.uut.round.sbox[0].sbe.inv._497_._4_	9	70.70803634553815
☑	test_aes_128.uut.round.sbox[0].sbe.inv._498_._1_	10	70.70803634553815

« ‹ 1 / 34 › »

Overview | Leaky signals | Leaky elements | Source | VCD | Trace

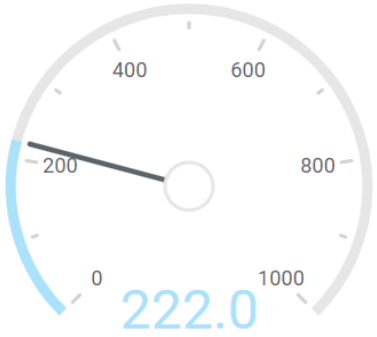
```
53 endmodule
54
55 /* inverse in GF(2^4)/GF(2^2), using normal basis [alpha^8, alpha^2] */
56 (*keep_hierarchy *) module GF_INV_4 ( A, M, N, Q );
57     input  [3:0] A;
58     input  [3:0] M;    /* input mask */
59     input  [3:0] N;    /* output mask */
60     output [3:0] Q;
61     (*nokeep*) wire   [1:0] a, b, m, n, c, e, d, p, q, an, mb, mn, m2, dn, em, qm;
62     (*nokeep*) wire   [1:0] csa, csb, cst, qsa, qsb, dm, psa, psb, cm, pm; /* partial sums */
63     (*nokeep*) wire   [2:0] af, bf, mf, nf, ef, df;  /* factors w/ bit sums */
64
65     assign a = A[3:2];
66     assign b = A[1:0];
67     assign m = M[3:2];
68     assign n = M[1:0];
69     assign m2 = N[3:2];  // mask for 2-bit sums, indep of M
70     FAC_2 afac(a, af);
71     FAC_2 bfac(b, bf);
72     FAC_2 mfac(m, mf);
73     FAC_2 nfac(n, nf);
74     GF_MULS_2 anmul(af, nf, an);
75     GF_MULS_2 mbmul(mf, bf, mb);
76     GF_MULS_2 mnmul(mf, nf, mn);
77 // YO! NEED TO DO SUMMATION BELOW IN SEQUENTIAL ORDER FOR SECURITY !!!!
78 /* optimize section below using NOR gates */
79     assign cst = {   /* note: ~| syntax for NOR won't compile */
80       ~(a[1] | b[1]) ^ (~(af[2] & bf[2])) ,
81       ~(af[2] | bf[2]) ^ (~(a[0] & b[0])) }
```

SCATE caes_sky130_nokeep_nokeep_hierarchy Project

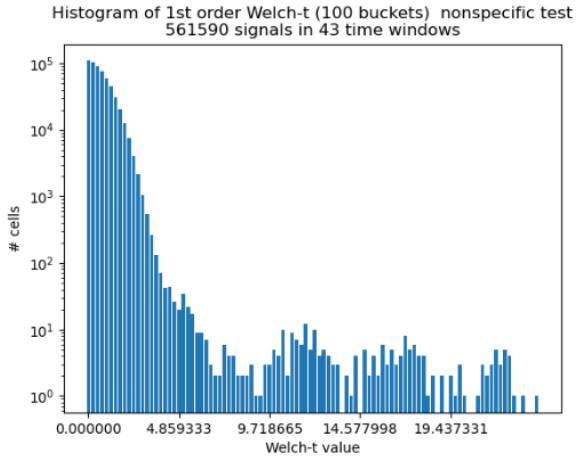
Masking – unconstrained synthesis

Overview | Leaky signals | Leaky elements | Source | VCD | Trace

Leaky signals



Max. leakage



$x \oplus m_1 \rightarrow x \oplus m_2$

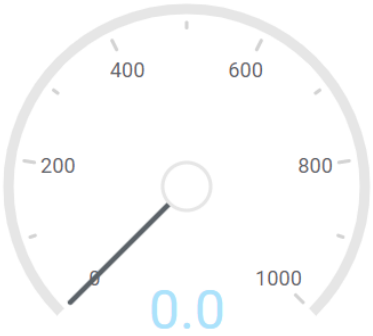
$((x \oplus m_1) \oplus m_1) \oplus m_2 \rightarrow x \oplus m_2$

SCATE caes_sky130_keep_keep_hierarchy Project

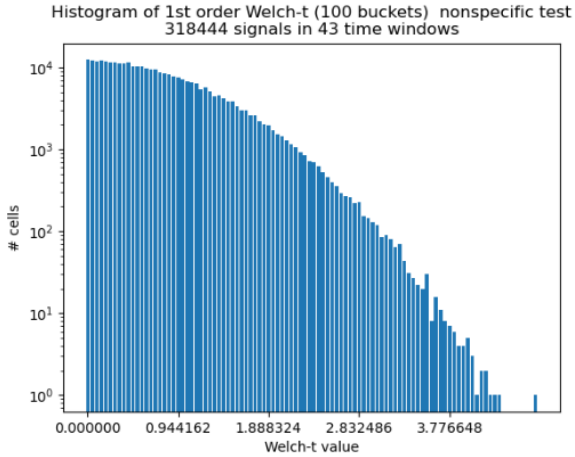
Masking – constrained synthesis

Overview | Leaky signals | Leaky elements | Source | VCD | Trace

Leaky signals



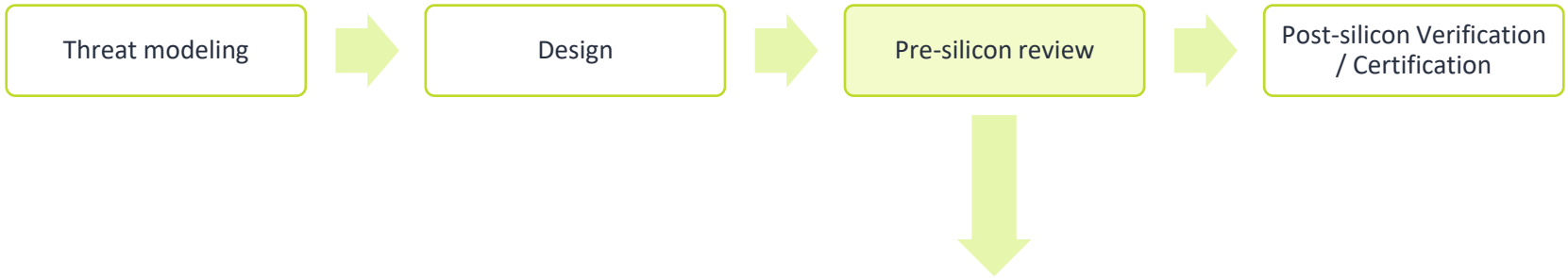
Max. leakage



$x \oplus m_1 \rightarrow x \oplus m_2$

$((x \oplus m_1) \oplus m_2) \oplus m_1 \rightarrow x \oplus m_2$

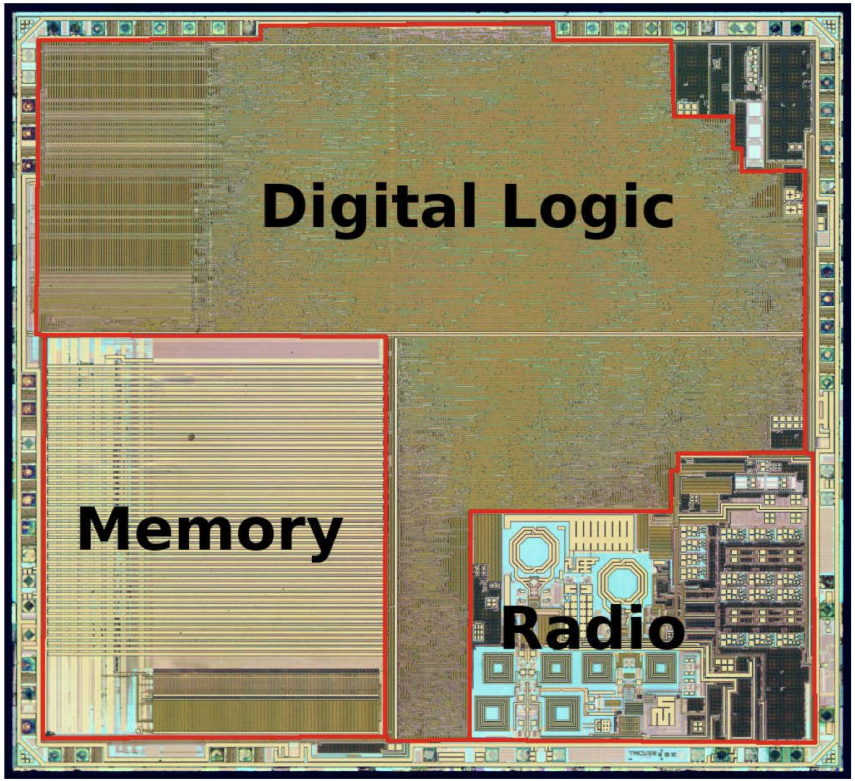
CASE STUDY: MASKED AES DESIGN



- Implementation of masking requires only basic understanding of security
- Leakage introduced by synthesis caught!
- Smaller risk of leakage post-silicon

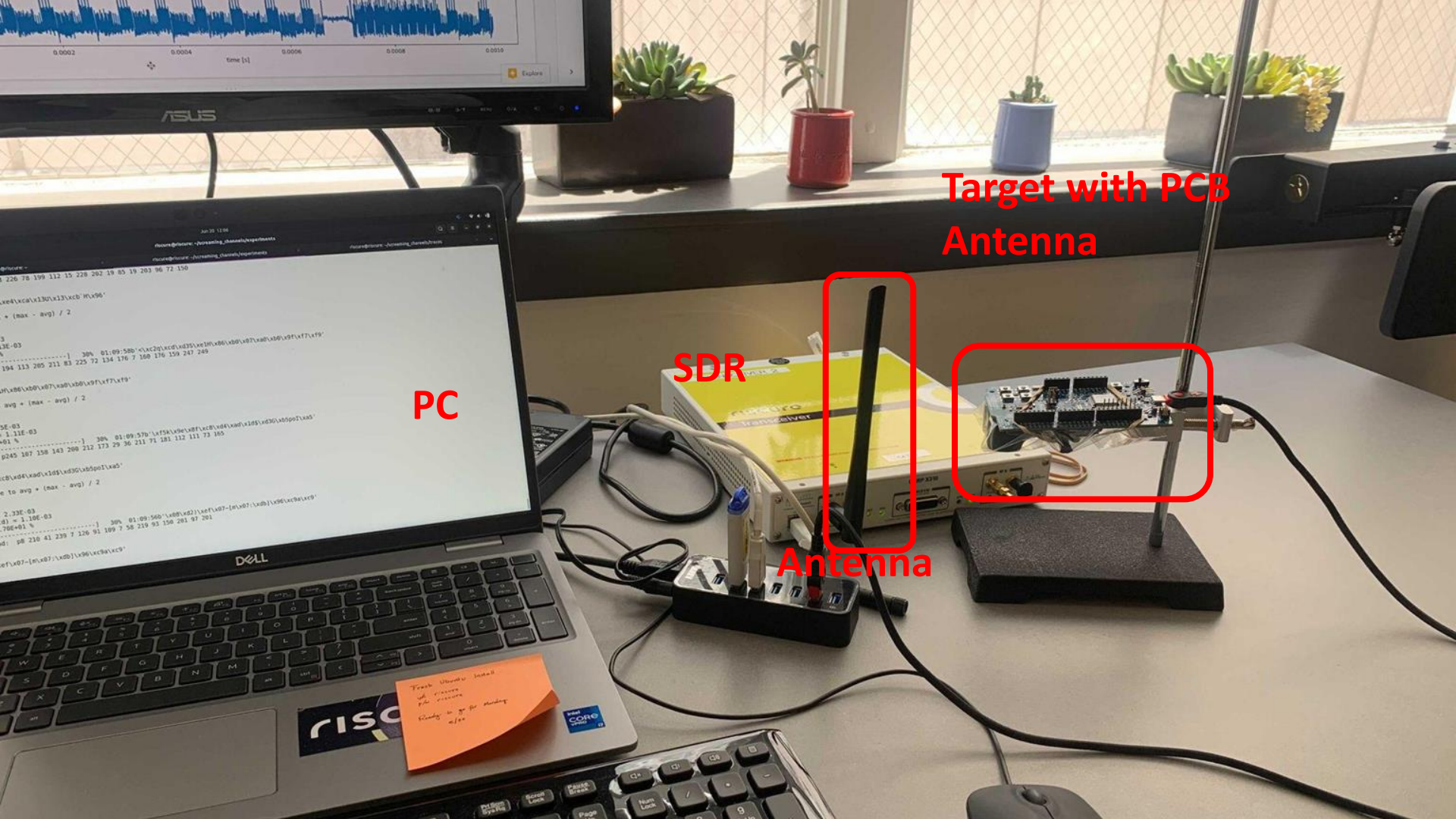
POST-SILICON CASE STUDY:

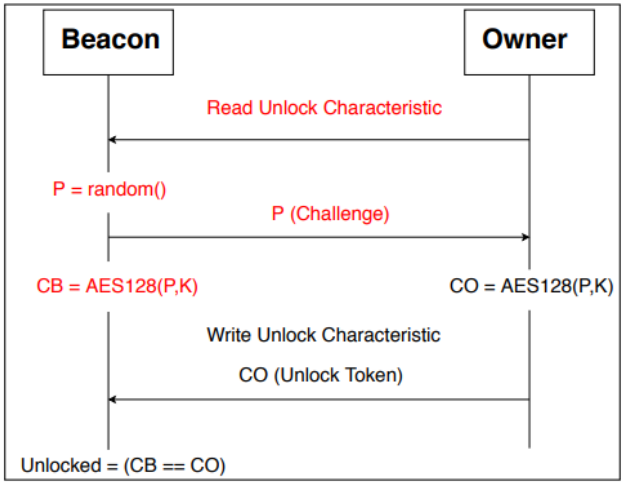
SCREAMING CHANNELS



Die of nRF51822

Screaming Channels: When Electromagnetic Side Channels Meet Radio Transceivers, Camurati et al, 2018
https://dl.acm.org/doi/10.1145/3243734.3243802


Target with PCB
Antenna
SDR
PC
Antenna

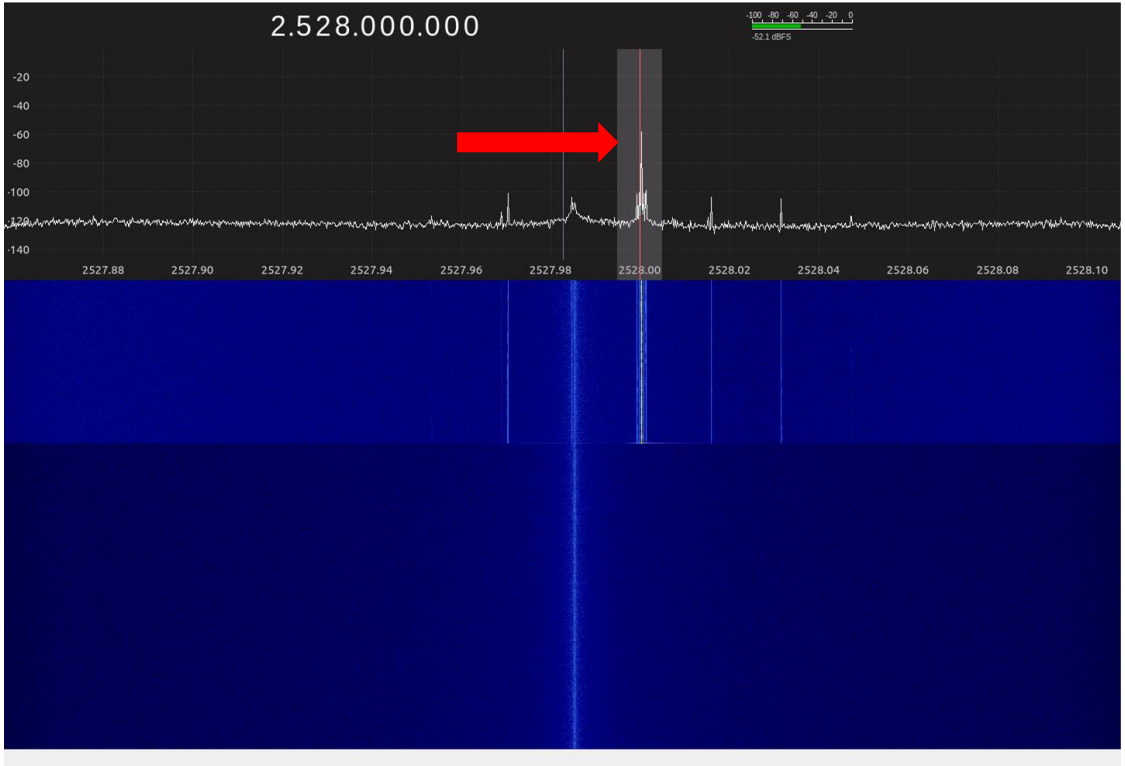


Unlock scheme used by
Google Eddystone Beacons

Understanding Screaming Channels: From a Detailed Analysis to Improved Attacks, Camurati et al, 2020
https://tches.iacr.org/index.php/TCHES/article/view/8594

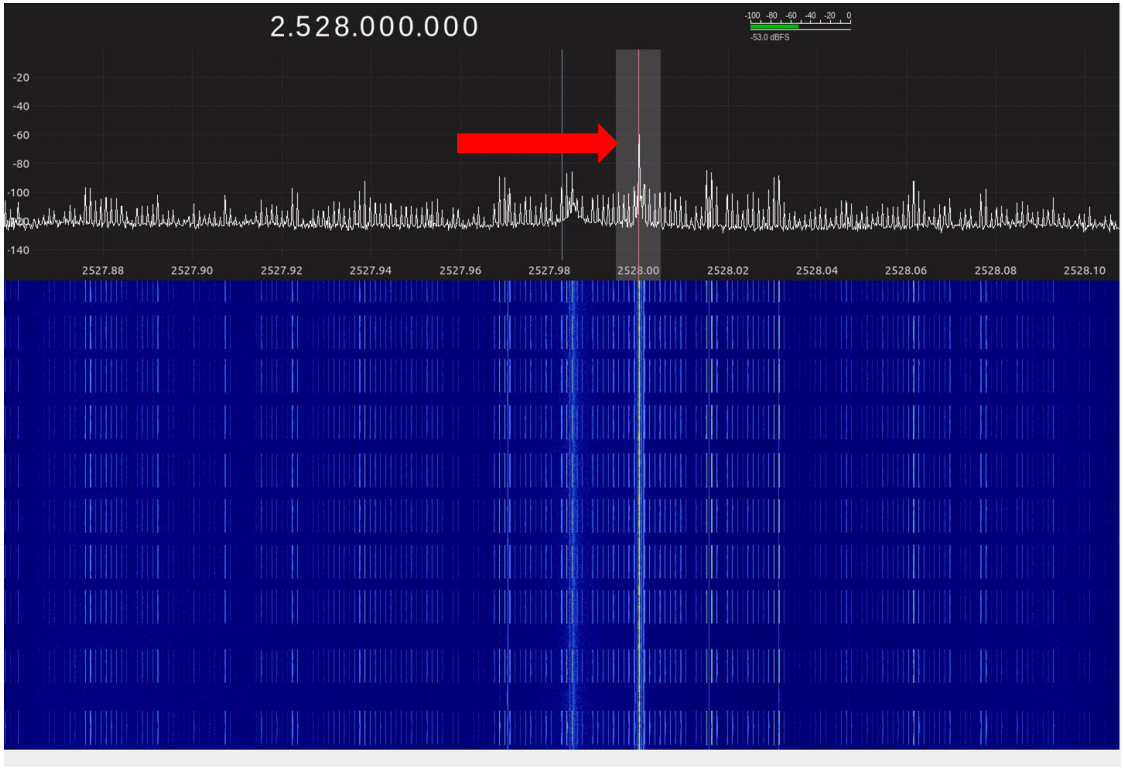
- $f_{target} = f_{channel} \pm \lambda * f_{board}$
- $f_{board} = 64MHz$
- $f_{channel} = 2.4GHz$
- $\lambda = 2$

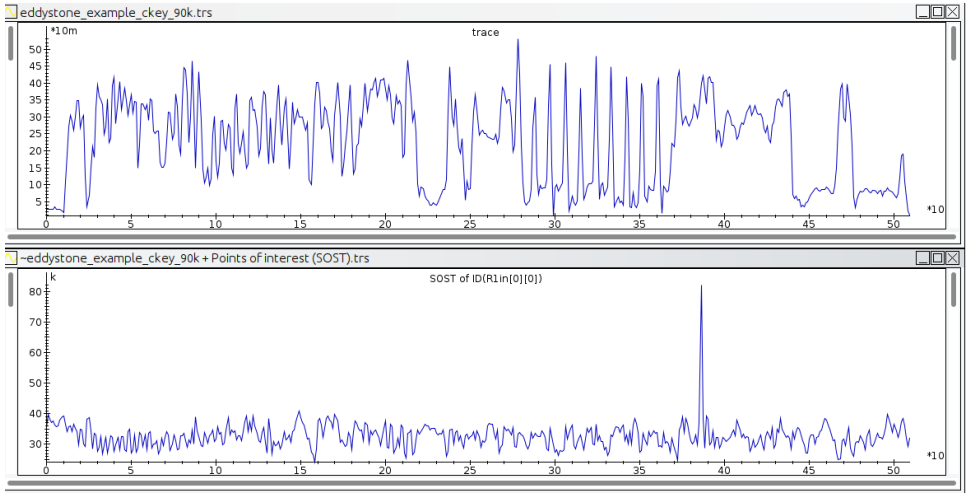
- Encryptions off



- $f_{target} = f_{channel} \pm \lambda * f_{board}$
- $f_{board} = 64MHz$
- $f_{channel} = 2.4GHz$
- $\lambda = 2$

- Encryptions on





Screaming Channels Traces template attack:

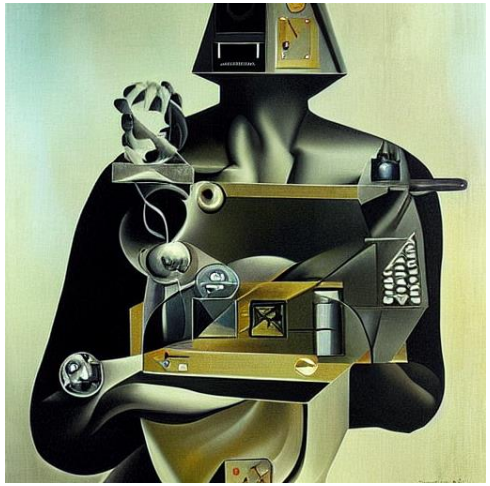
a. 70,000x1 for training with a fixed random key.
b. 33,000x1 for attacking with a fixed random key.

Understanding Screaming Channels: From a Detailed Analysis to Improved Attacks, Camurati et al, 2020
https://tches.iacr.org/index.php/TCHES/article/view/8594

CONCLUSION

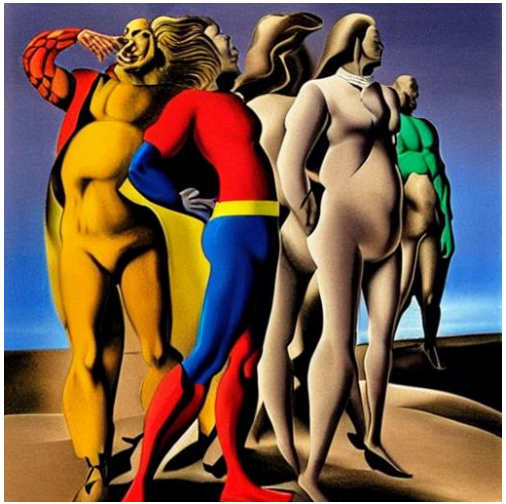
CONCLUSION

- Scaling of number of devices through 5g requires **scaling security**
- Good security is not an accident, but an engineering discipline
  - Threat model
  - Validate at every stage!
- Security is only enabled through **recognition of actual risks**
  - Brand, financial, health/safety, certification, nat’l security
- **Chip security** requires **countermeasures** against **hardware and software** attacks
- **Scale** can only be achieved through **automation**



ACKNOWLEDGMENTS

- Screaming Channels – Zhenyuan (Charlotte) Liu, Cees-Bart Breunesse
  - And the authors of the academic papers
- Panasayya Yalla – Threat modeling / secure design slides
- SCATE acknowledgements:
- Peter Grossman et al, Intrinsix (a CEVA company)
- Patrick Shaumont et al, Worchester Polytechnic Institute
- Shreyas Sen et al, Purdue University
- Cees Breunesse, Nicole Fern, Raj Velegalati, Riscure



Riscure B.V.
Frontier Building, Delftechpark 49
2628 XJ Delft
The Netherlands
Phone: +31 15 251 40 90
www.riscure.com

Riscure North America
550 Kearny St., Suite 330
San Francisco, CA 94108 USA
Phone: +1 650 646 99 79
inforequest@riscure.com

Riscure China
Room 2030-31, No. 989, Changle Road,
Shanghai 200031
China
Phone: +86 21 5117 5435
inforcn@riscure.com

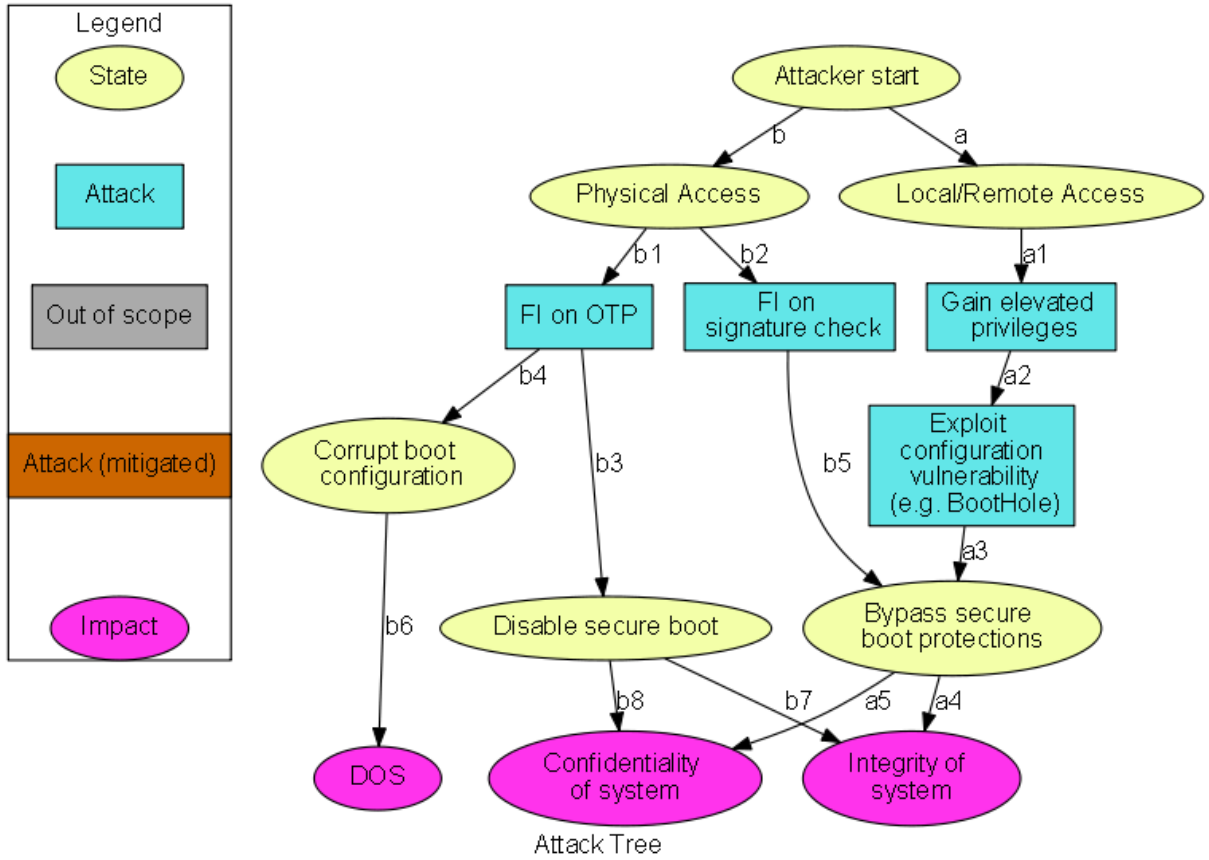
riscure

driving your security forward

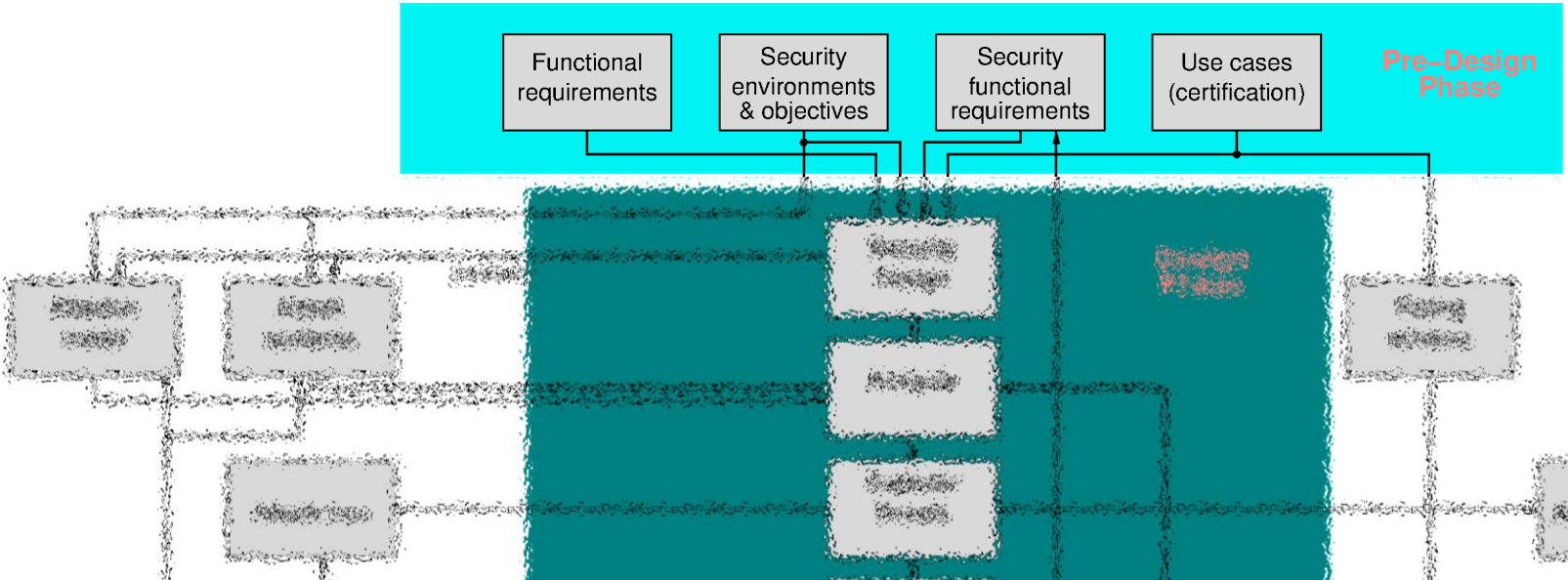
BACKUP SLIDES

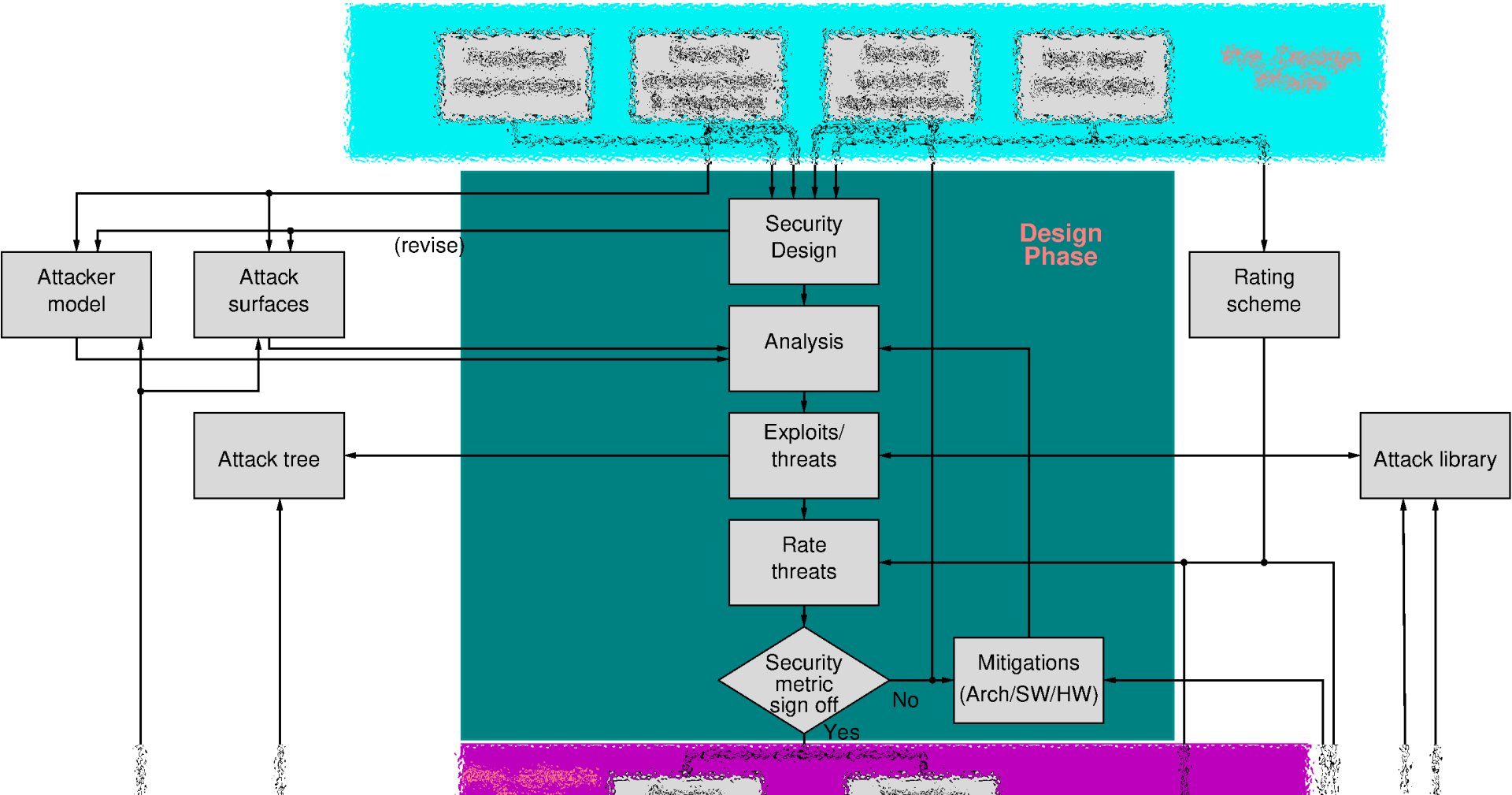
ATTACK TREES

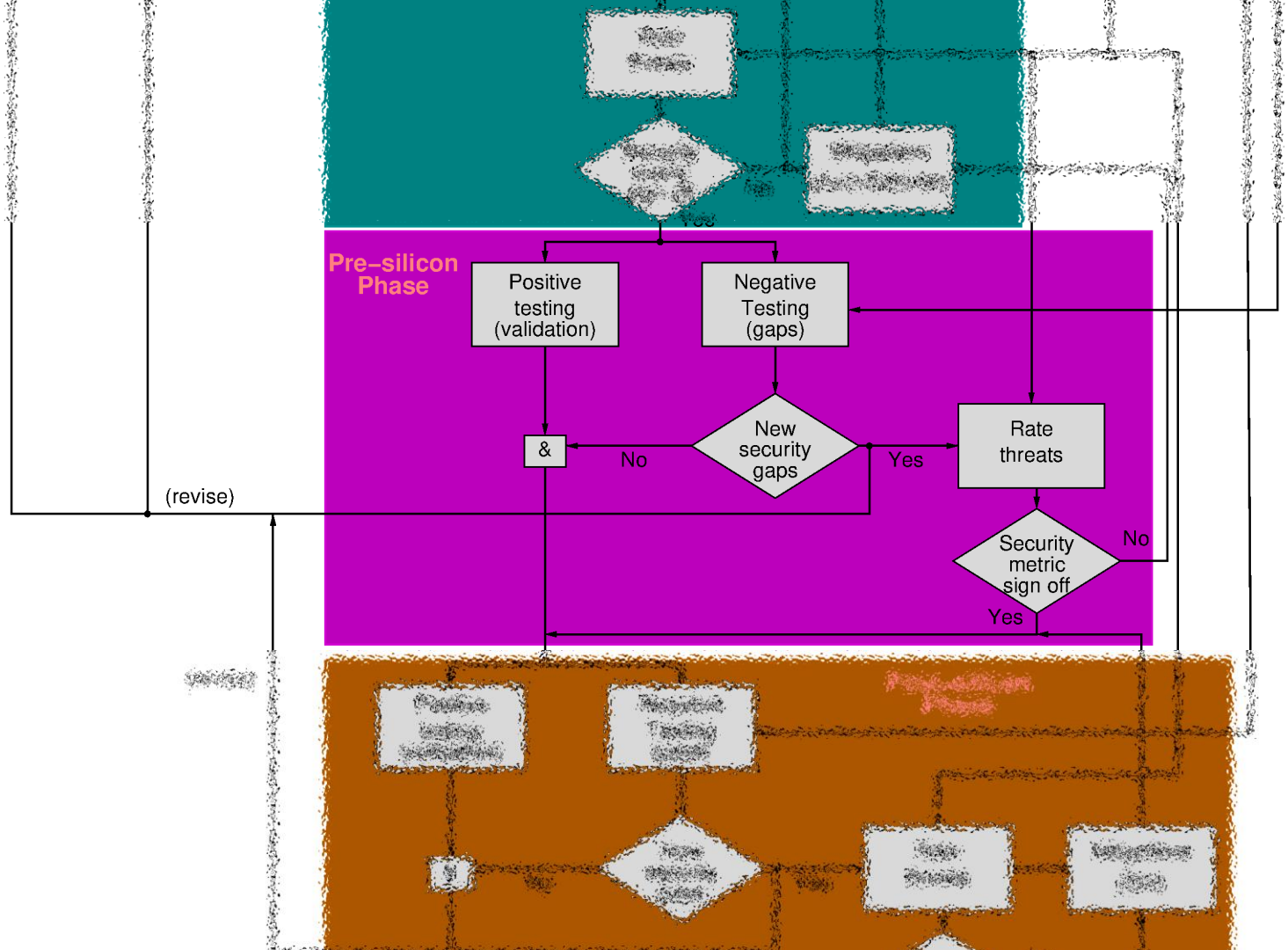
- Introduced by Bruce Schneier et al.
- Conceptual attack diagrams
- Different ways to reach an asset

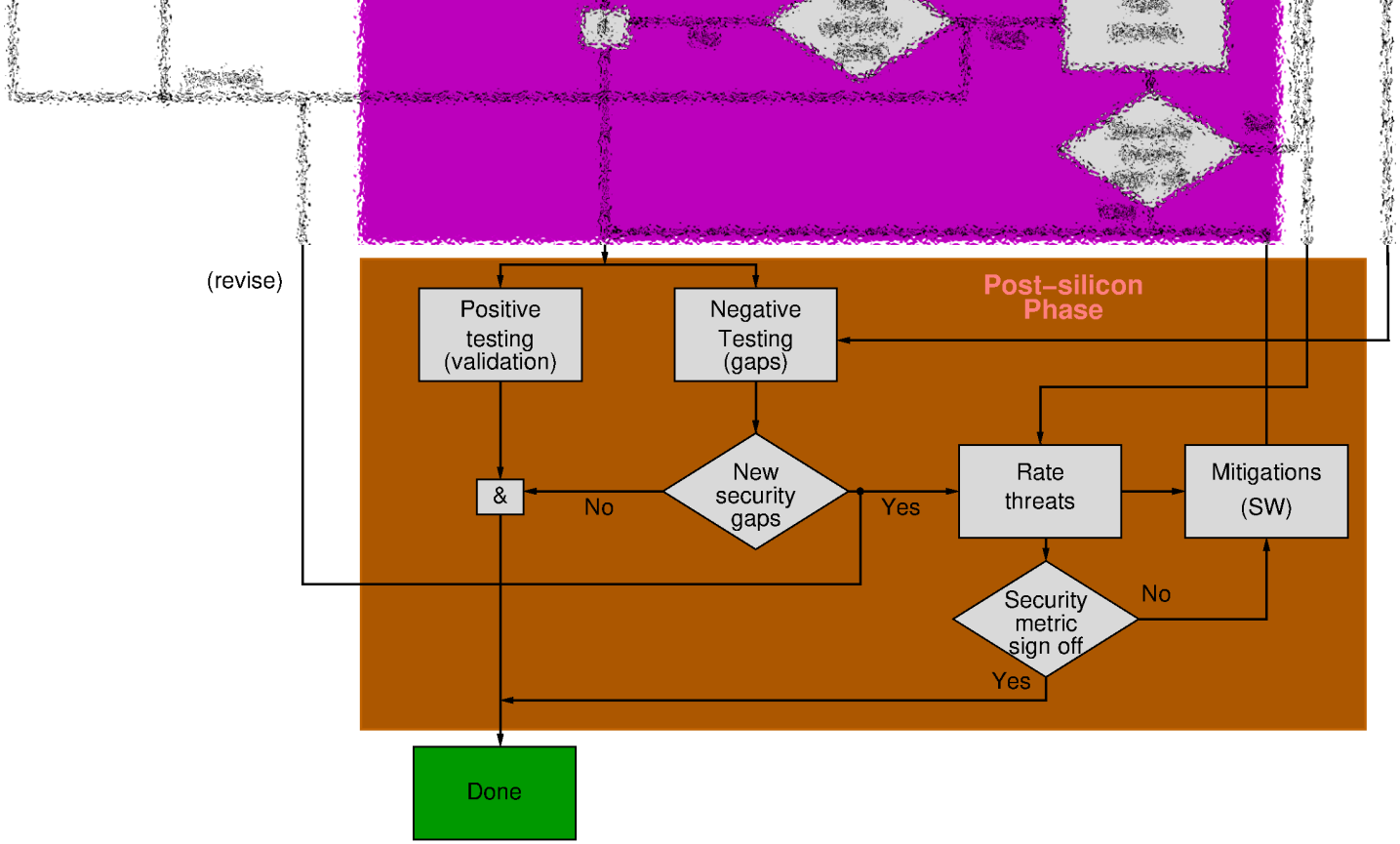


THREAT MODELING DETAILS


Functional requirements
Security environments & objectives
Security functional requirements
Use cases (certification)
Pre-Design Phase


Security Design
Design Phase
(revise)
Attacker model
Attack surfaces
Rating scheme
Analysis
Attack tree
Exploits/ threats
Attack library
Rate threats
Security metric sign off
No
Mitigations (Arch/SW/HW)
Yes


Pre-silicon Phase
Positive testing (validation)
Negative Testing (gaps)
&
No
New security gaps
Yes
Rate threats
Security metric sign off
No
Yes
(revise)


(revise)
Post-silicon Phase
Positive testing (validation)
Negative Testing (gaps)
&
No
New security gaps
Yes
Rate threats
Mitigations (SW)
Security metric sign off
No
Yes
Done

PRE-SILICON CASE STUDY:

RISC-V ROM HARDENING

CPU CASE STUDY: PICORV32

Identify sensitive RTL registers

How can we flip a branch condition?

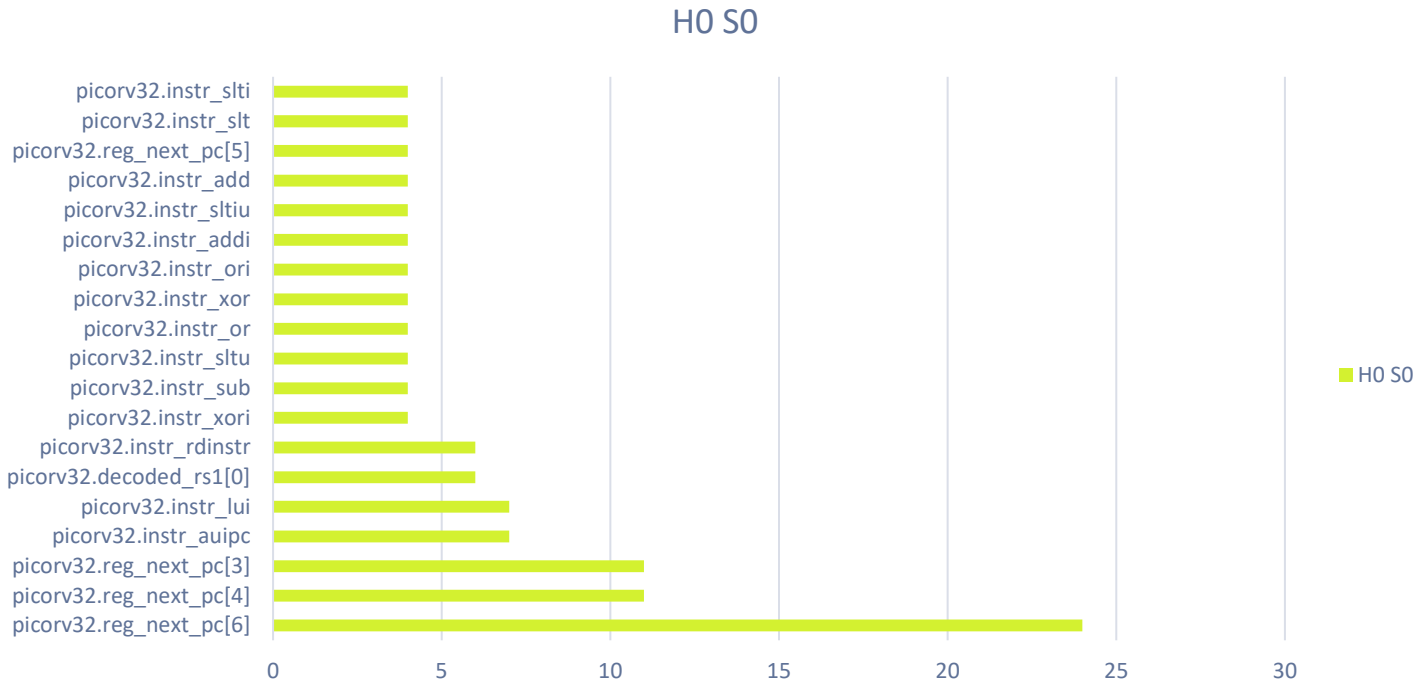
- Use picorv32 core
- Brute-force all single flips for all clocks and all registers/wires

https://github.com/YosysHQ/picorv32

```c
void secureboot(void) {
    set_test_status(TEST_START);
    debug("branch test\n");

    set_test_status(TEST_TRIGGER_UP);
    char success = *(volatile unsigned char *)(UART0_BASE_ADDR
    if(success) {
        set_test_status(TEST_TRIGGER_DOWN);
        debug("success\n");
        set_test_status(TEST_FI_SUCCESS);
    } else {
        set_test_status(TEST_TRIGGER_DOWN);
        debug("failure\n");
        set_test_status(TEST_FI_FAIL);
    }
}
```

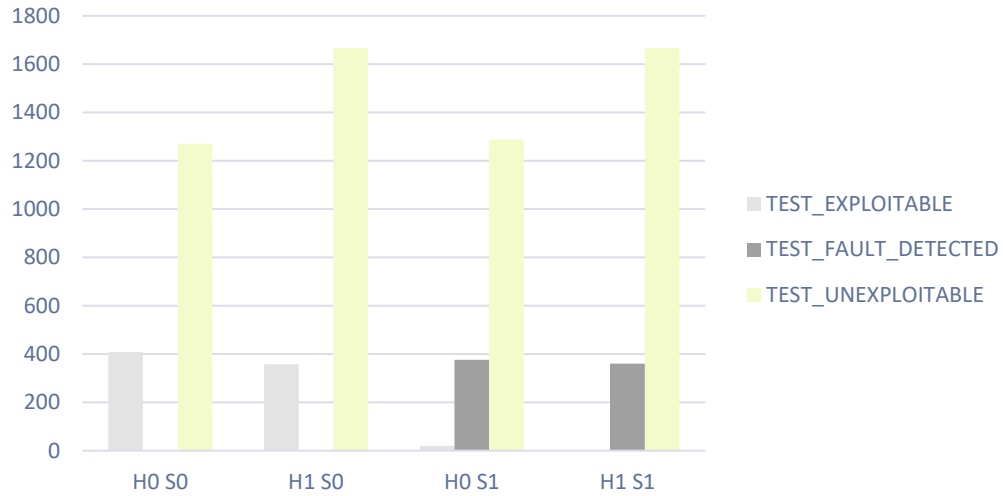
TOP SIGNALS SENSITIVE TO FAULTS



- Reg_next_pc particularly vulnerable
- Long tail not shown here (408 signals total)

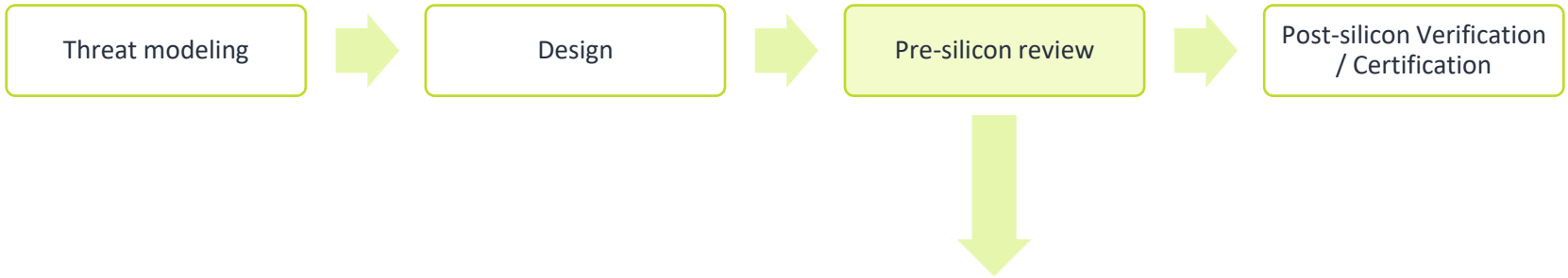
CPU CASE STUDY: PICORV32

	Vanilla	Harden RTL	Harden sw	Harden RTL + sw
Total	1386400	1540800	1386400	1540800
Exploitable faults	408	358	20	2
Exploitable signals	268	264	4	2
Detections	0	0	376	361



(this experiment is waaay to limited to conclude anything general about hardware vs software countermeasures)

CPU CASE STUDY: PICORV32



- Reducing FI was as trivial as adding redundancy
- Re-running tool allows validating various countermeasure options