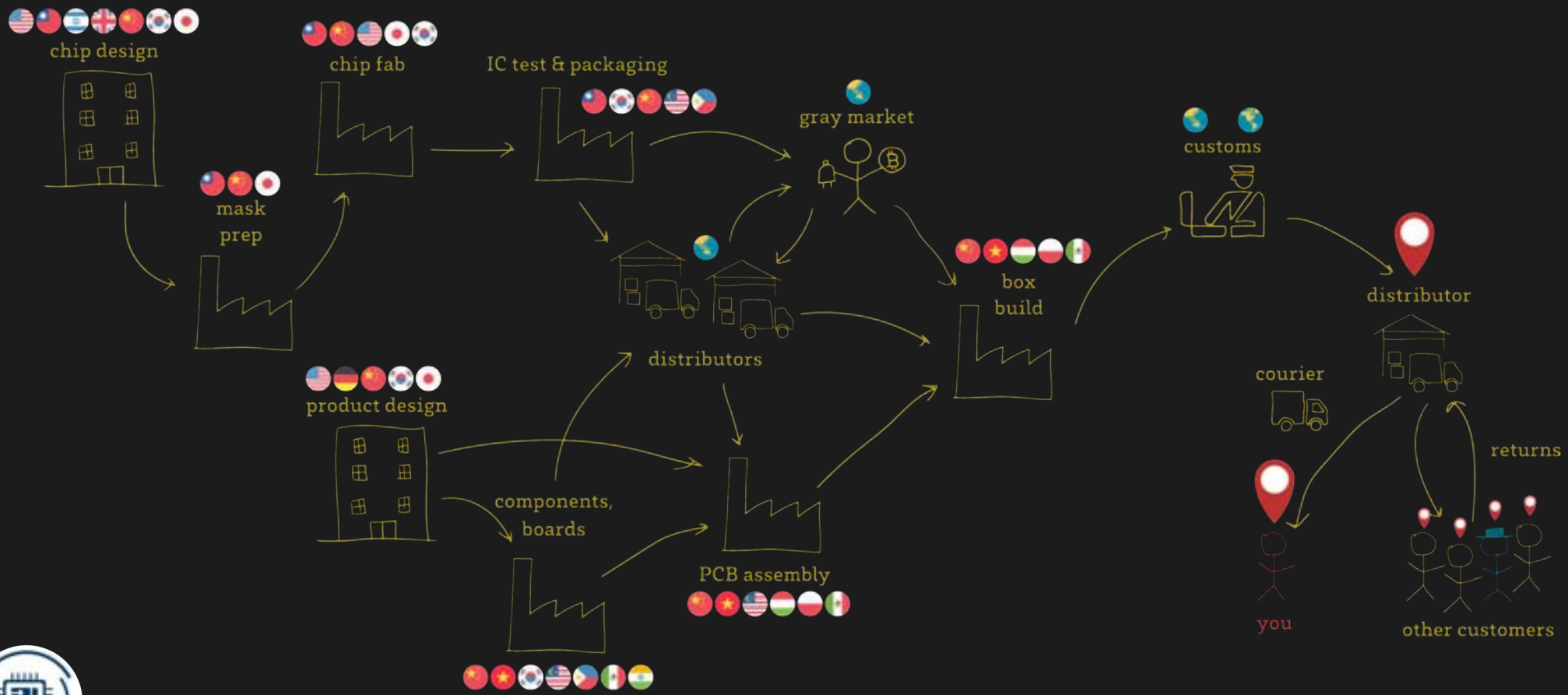
Improving Trust in Supply Chains

Translating Research into Everyday-Use Techniques

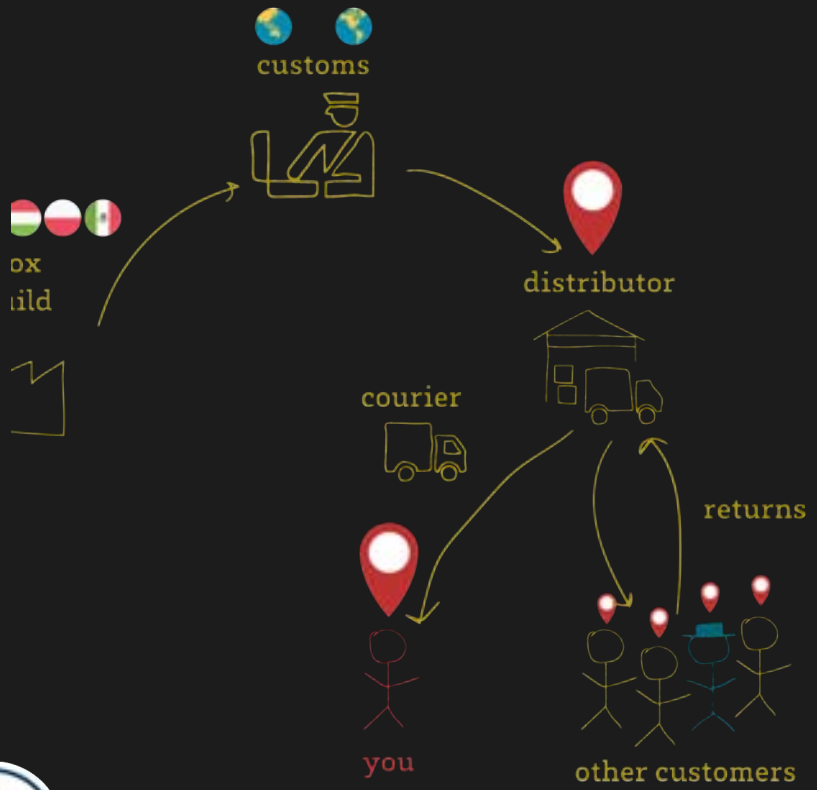
bunnie | masto: @bunnie@treehouse.systems | bsky: @bunnie.org

HARRIS 2025

Trust Issues: Concerns about the Supply Chain



Trust Threat May be Both Dynamic and Local



- Localized:
  - Attacker is not constrained to modify 100% of material
- Dynamic:
  - Attacker is not constrained to operate consistently
- Upshot:
  - Sample-based testing is ineffective
  - Trusted supplier alone is ineffective
    - End-to-end traceability
    - Point of use verification

Reflections from A Parallel Problem

- Nuclear fallout detection is surprisingly similar to the supply chain trust problem
  - Question at hand:
    - Is this area safe to enter?
    - Is this food safe to eat?

Relevant Experience: Safecast Geiger Counter

- In 2011, I designed and helped to produce an open-source reference geiger counter in response to the crisis

Characteristics of Fallout Detection

- Dynamic
  - Position of fallout changes daily with rainfall and wind

- Localized
  - Fallout accumulates in small pools around a dwelling
  - ^{90}Sr emits β radiation, detectable only at ~1m range

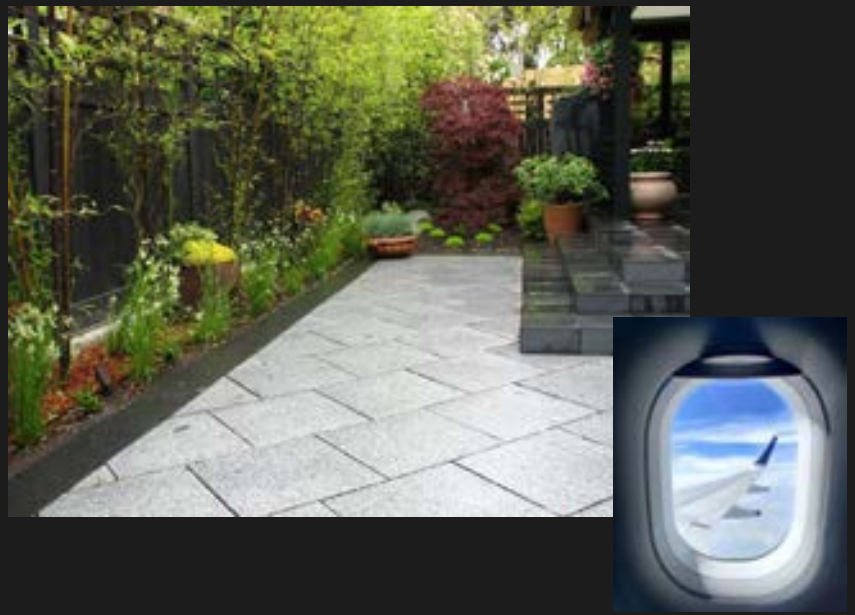
Other Parallels

- Governments & corporations have an incentive to make things seem safer than they are

- Gold-standard testing is destructive

Other Parallels

- A lack of baseline data complicates analysis & policymaking



- Takes a crisis for the public to care; overwhelmed experts, knee-jerk policy responses

Key Finding #1: Translating Research is Hard

- Meter reads: "35 cpm"
- People ask: "Am I safe?"
- Physicist's response:
  - "It depends"
  - Starts on 20-minute lecture on nuclear physics
  - This approach was ultimately not fruitful

Solution: "Traffic Light" Dashboards

- Everyday people don't have the time to acquire nuance
  - "zero, one, or many" rule of cognitive load
    - Green – don't worry
    - Yellow – ask for help
    - Red – worry
  - Perfect safety is as impossible as perfect measurements

Key Finding #2: Reducing Barriers Takes Effort

- Technology: Low-cost, consumer-ready metrology gear
- Citizens: Volunteers to maintain & gather
- Regulators: An indifferent or permissive power structure

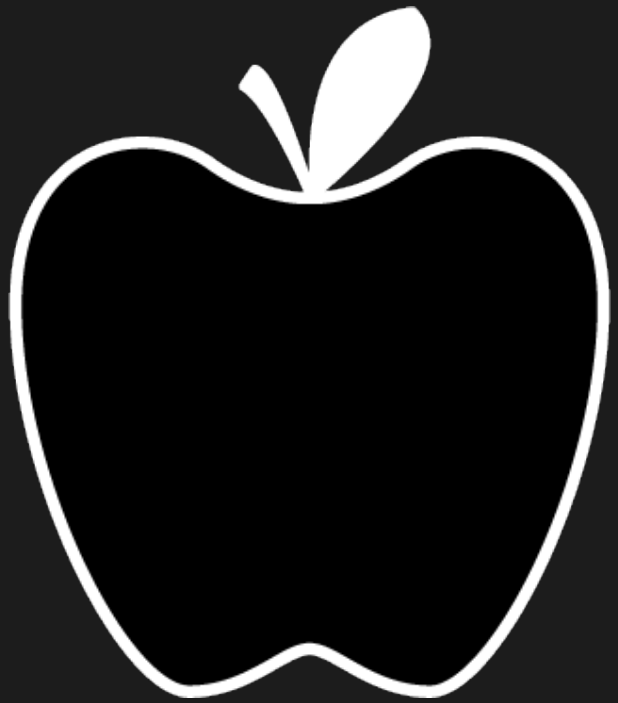
Mapping These Experiences to Problems in Supply Chains & Trust

- #1 Simplify the Discussion
  - Reduce a nuanced, multi-dimensional discussion into a single linear scale
- Proposal: levels graded by cost to detect an attack
  - Use concrete examples to ground the levels
- #2 Reduce cost of detection
  - Reduce gap between "state of art" and "state of practice"
  - Share data so we have baselines
- Proposal: reduce the cost of metrology, make the tools open source

Simplifying the Discussion: Creating a Categorization System By Analogy

Can I trust this chip?

Is this safe to eat?



Limitations of the Analogy

- Stakes:
  - A modified chip in a server could impact millions of users
- Remedies:
  - Chips are made in billion-dollar fabs

- Stakes:
  - A poisoned fruit might make the person who ate it sick
- Remedies:
  - Fruit grows on trees

However, both require global supply chains...

...and we verify our chips about as much as we verify our fruit.

Four-Level Classification System

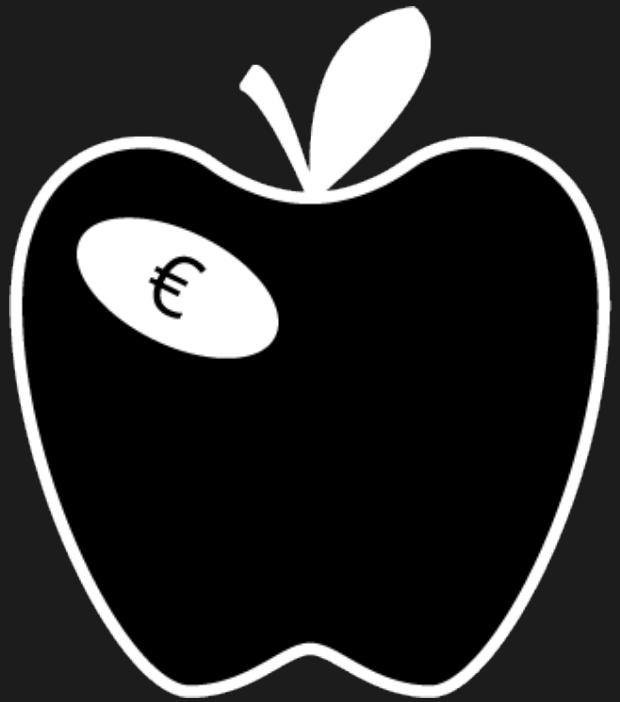
Level 3: Detected only with $1mm+ tools and/or requires new techniques

Level 2: Detected with $10k-$100k tools

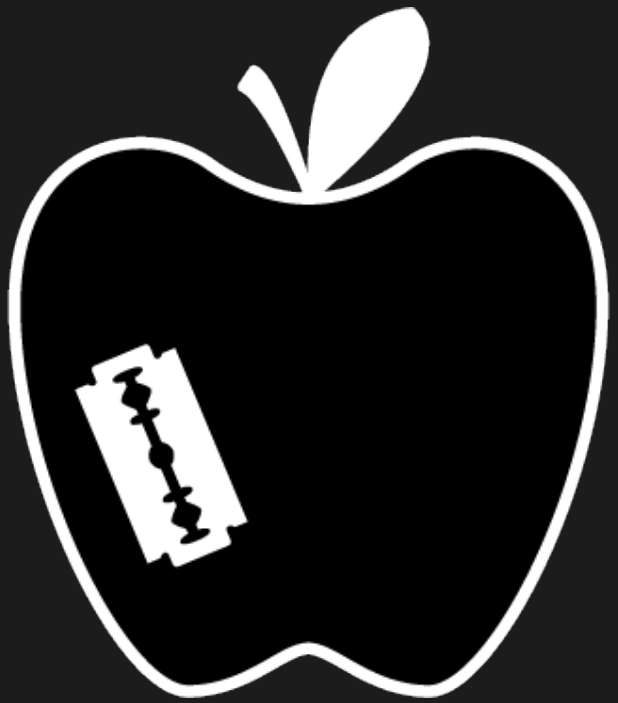
Level 1: Detected with $1k-$10k tools

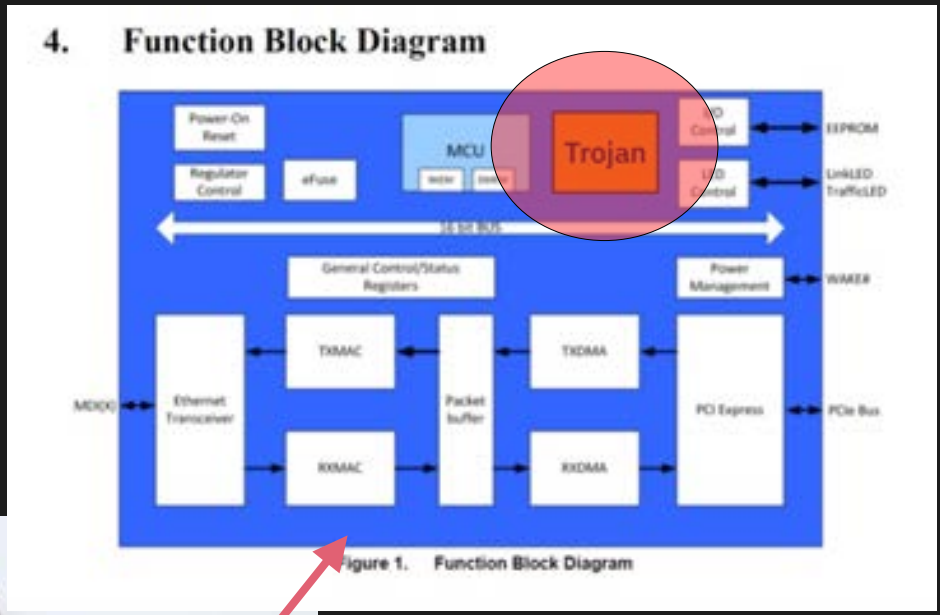
Level 0: Detected with <$1k tools

Level 0: Detectable at Home (Point of Use)
Exemplar: Misrepresentation of Goods

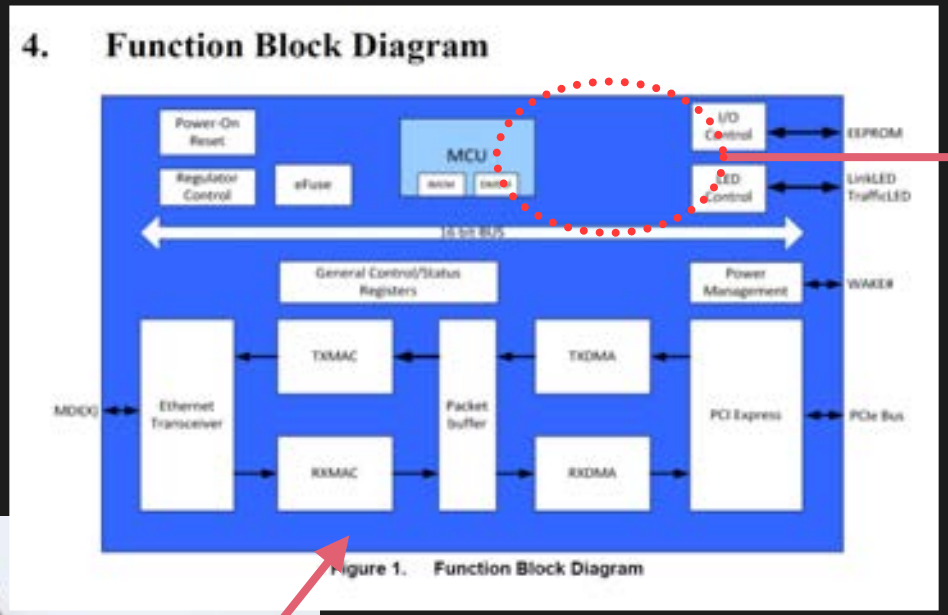


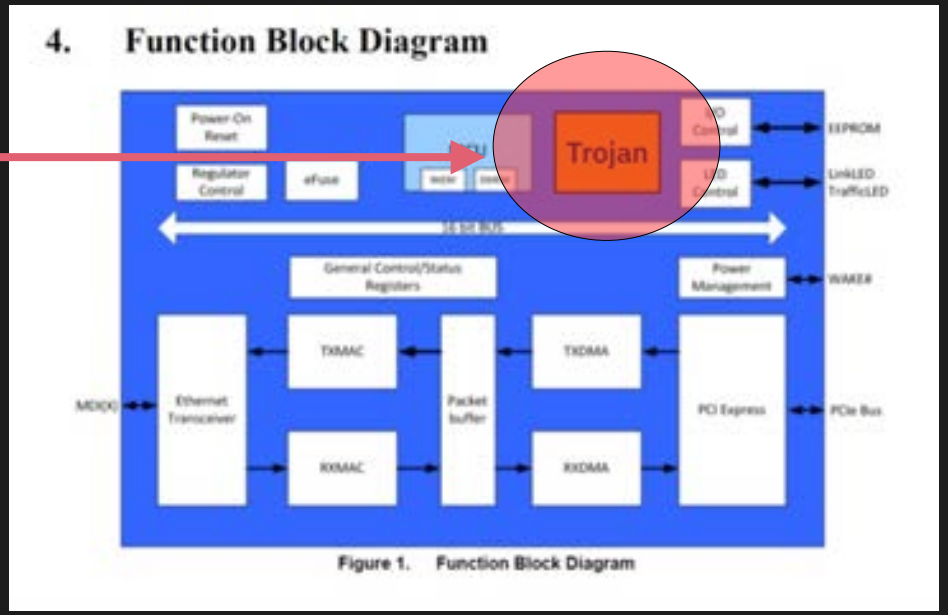
Level 1: Easily Detected With $1k–$10k Tools "Block-Level Modifications"





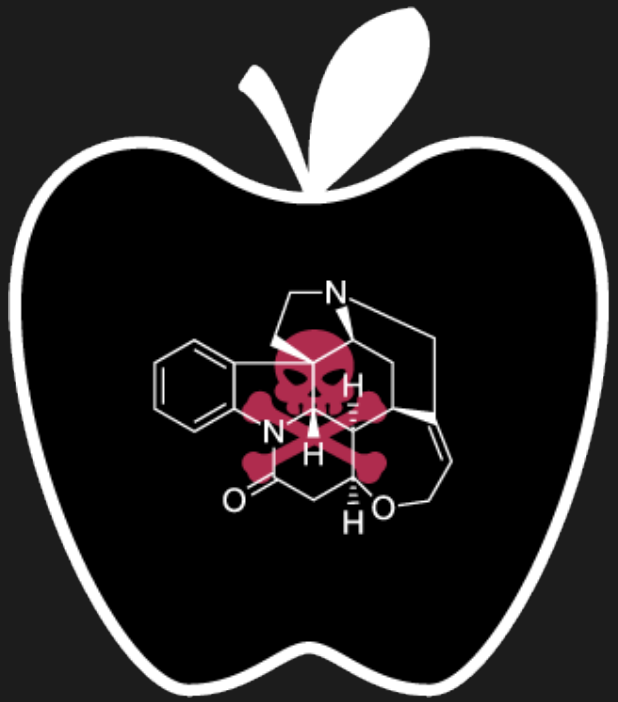
Examplar: Modified NIC Chip

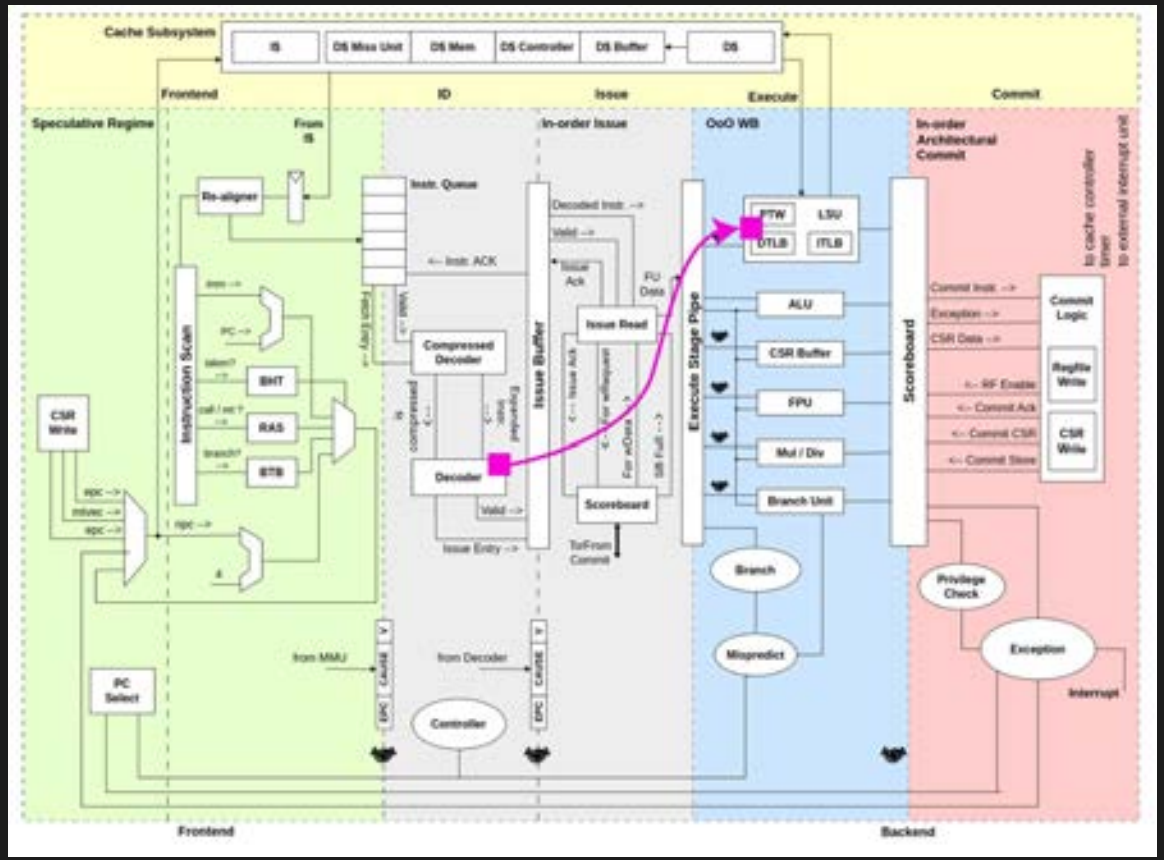




- NIC blocks available now as F/OSS or low-cost IP
- Uses older process (~65nm)
- Estimate <$300k up-front cost to mount attack
- Unit cost is possibly even profitable

Level 2: Detected With $10k–$100k tools
Sub-block RTL-Level Modifications

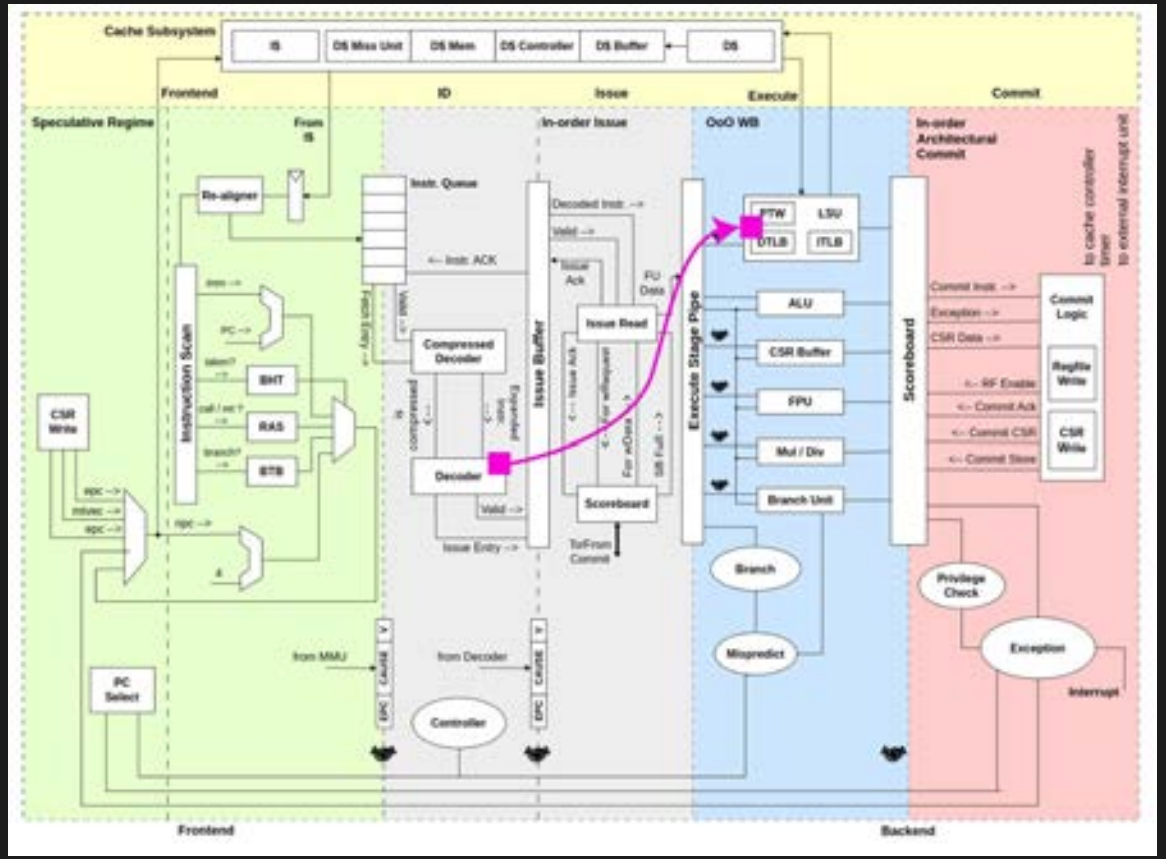




https://github.com/openhwgroup/cva6?tab=readme-ov-file

Key Assumptions

- Assumption: there are two versions of the chip in the supply chain, one with the modification, and one without
  - "Bad by design" is a different question
  - https://ghostwriteattack.com/riscvuzz.pdf



https://github.com/openhwgroup/cva6?tab=readme-ov-file

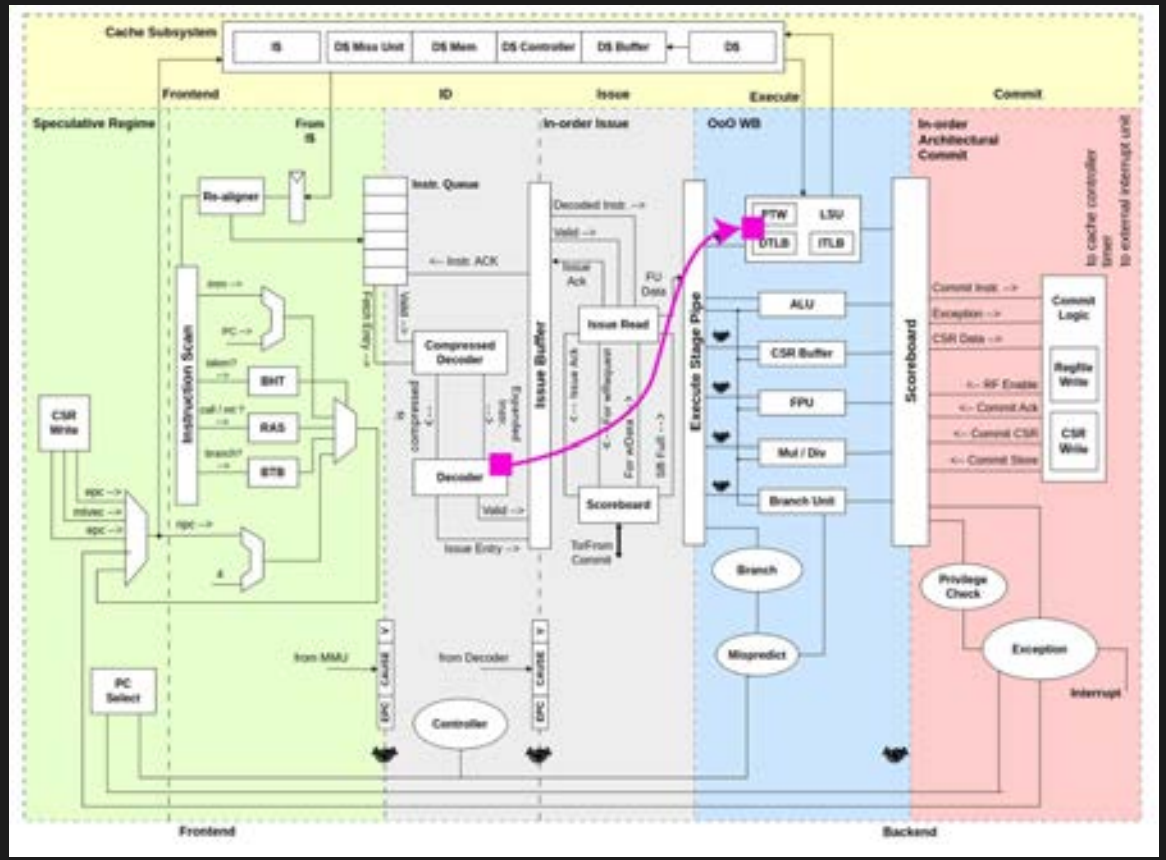
Exemplar: Modifying a CPU Pipeline

- Observation:
  - ra (x1) on RISC-V is the link register
  - Compiled code only uses it in limited contexts, e.g.: "jalr, ra target"
- Create a memory protection bypass with trigger using this primitive

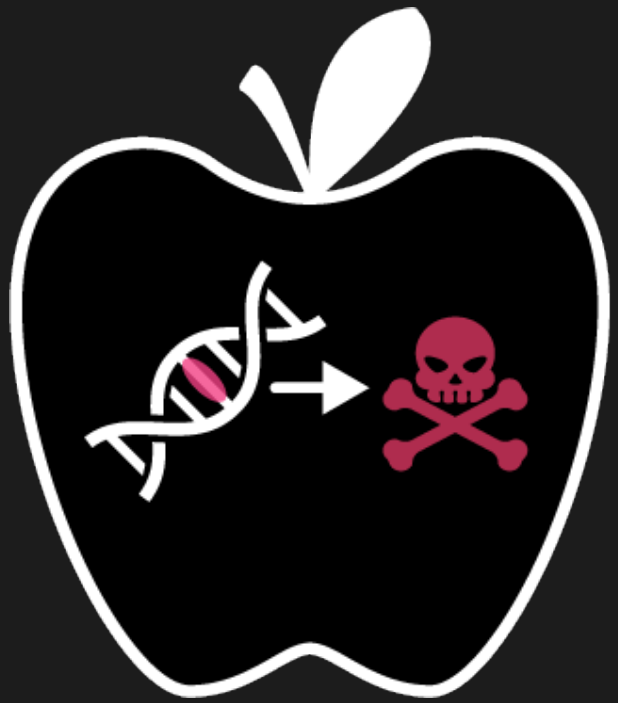
```
ffd0381e <xous_kernel::arch::riscv::current_pid>:
ffd0381e:   1141                    addi    sp,sp,-16
ffd03820:   c606                    sw  ra,12(sp)
ffd03822:   0000f097            auipc   ra,0xf
ffd03826:   db4080e7            jalr    -588(ra) # ffd125d6 <__read_satp>
ffd0382a:   8159                    srli    a0,a0,0x16
ffd0382c:   0ff57593            zext.b  a1,a0
ffd03830:   c581                    beqz    a1,ffd03838 <xous_kernel::arch::riscv::curre
ffd03832:   40b2                    lw  ra,12(sp)
ffd03834:   0141                    addi    sp,sp,16
ffd03836:   8082                    ret
ffd03838:   ffd15537            lui a0,0xffd15
ffd0383c:   a2450513            addi    a0,a0,-1500 # ffd14a24 <_ebss+0xfff91d64>
ffd03840:   0000d097            auipc   ra,0xd
ffd03844:   01a080e7            jalr    26(ra) # ffd1085a <core::option::unwrap_failed>
```

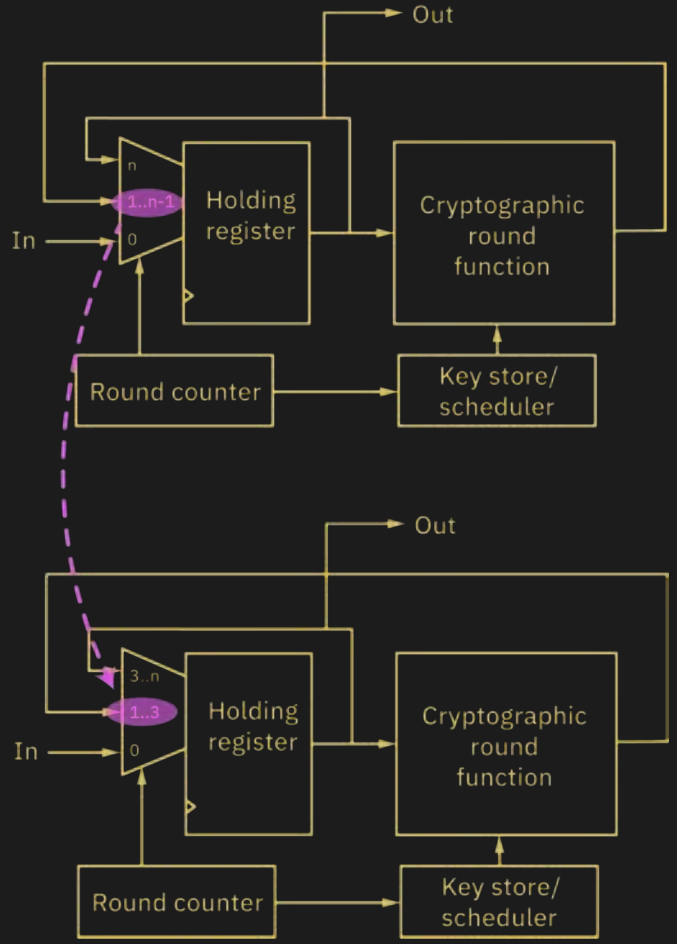
Exemplar: Modifying a CPU Pipeline

- Hypothetical Trojan:
  - Decoding a "load" using ra as the address base...
  - ...causes ra contents to be treated as if a physical address
  - Thus bypassing virtual memory protection
  - Optional:
    - Use unlock "knock" sequence to frustrate discovery by fuzzing
    - i.e. sequence is armed by a preceding "dummy" instruction like "addi x0, x0, 0x666"
  - Requires O(10)-O(100) logic cells to implement

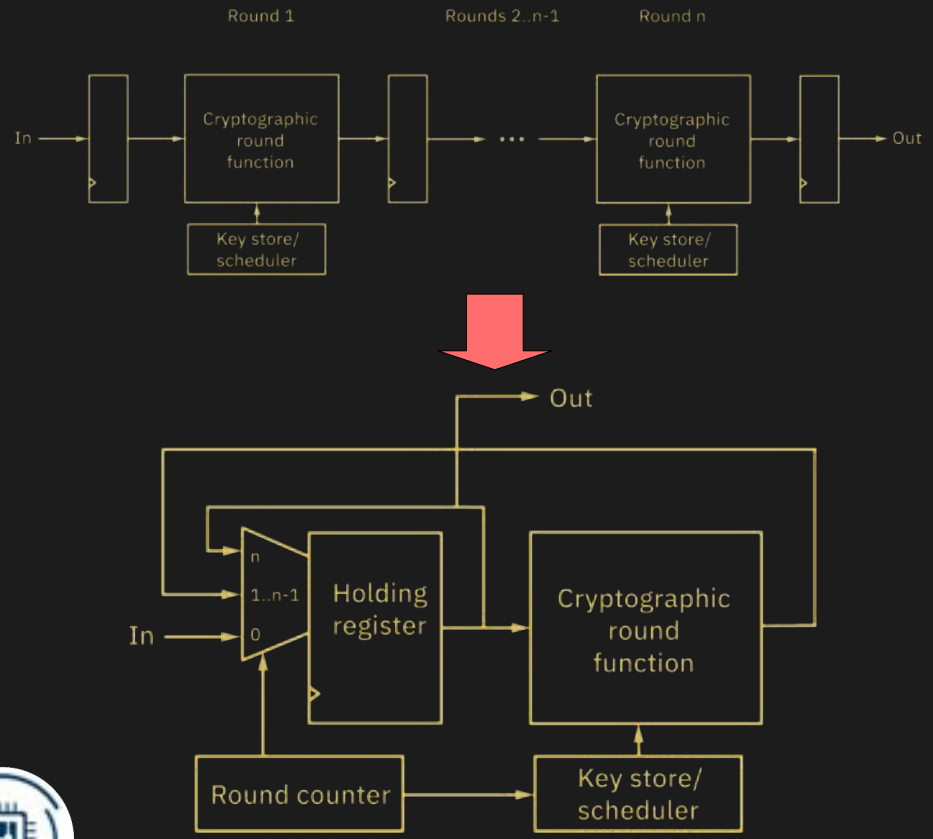


Level 3: Requires $1mm+ Tools/Novel Techniques
Exemplar: Tailored Mask Edits



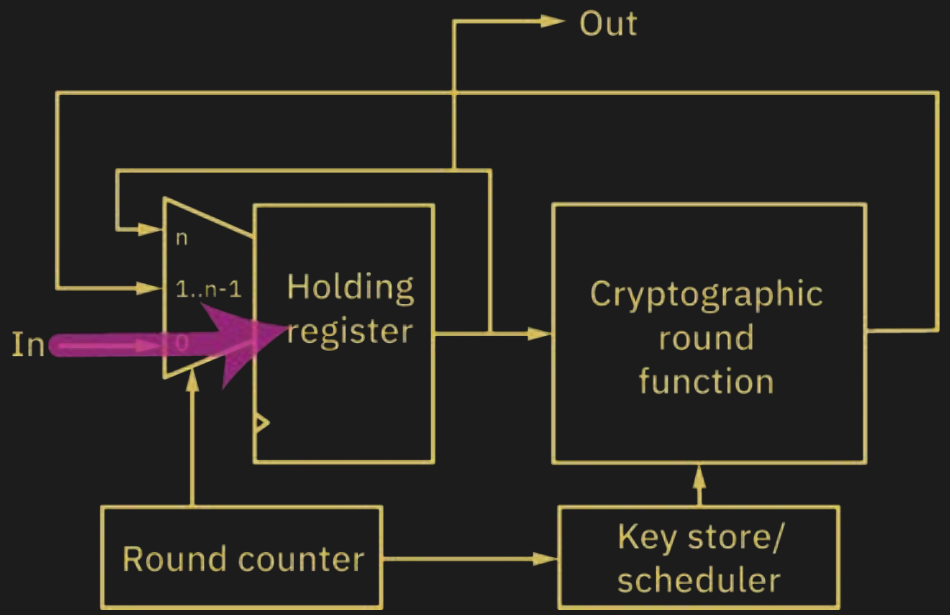


Exemplar: Reduced Round Cryptography Using a Small Mask Edit



- Some ciphers use repeated round of computation for security
  - Instead of implementing N copies of the hardware...
  - ...a single round is implemented in a loop

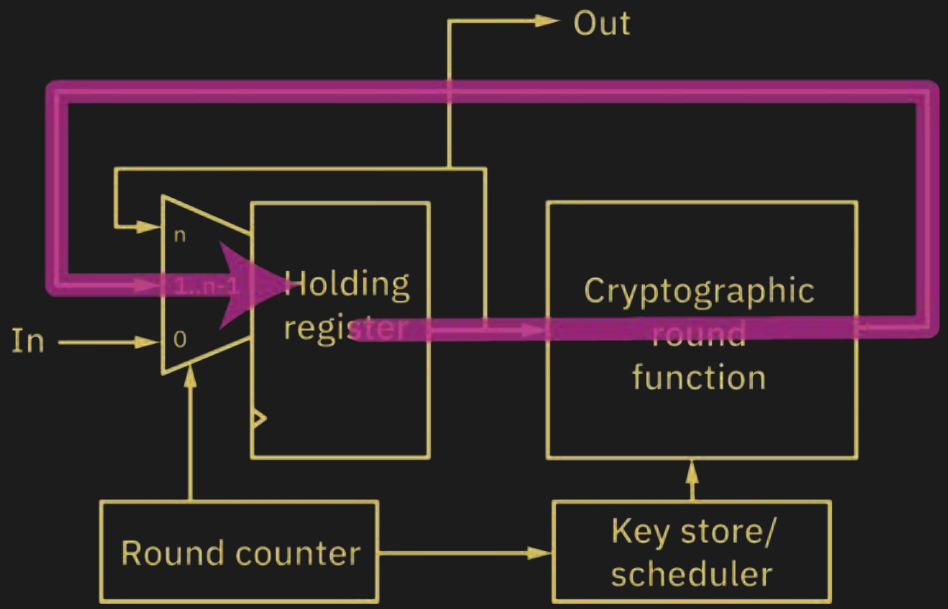
Background: Multi-Round Cipher



- Round "0"
  - Load in fresh data

```
0000 - load
0001 - round
0010 - round
0011 - round
0100 - round
0101 - round
0110 - round
0111 - round
1000 - round
1001 - round
1010 - round
1011 - round
1100 - round
1101 - round
1110 - round
1111 - hold
```

Background: Multi-Round Cipher

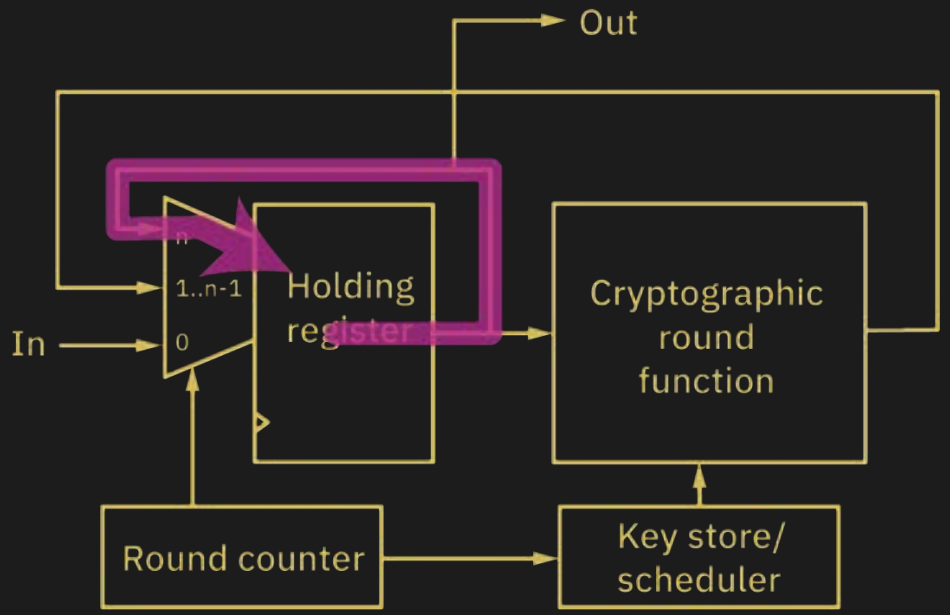


- Rounds "1..(n-1)"
  - Repeatedly apply the round function to the data

```
0000 - load
0001 - round
0010 - round
0011 - round
0100 - round
0101 - round
0110 - round
0111 - round
1000 - round
1001 - round
1010 - round
1011 - round
1100 - round
1101 - round
1110 - round
1111 - hold
```

e.g. 14 rounds for AES-256

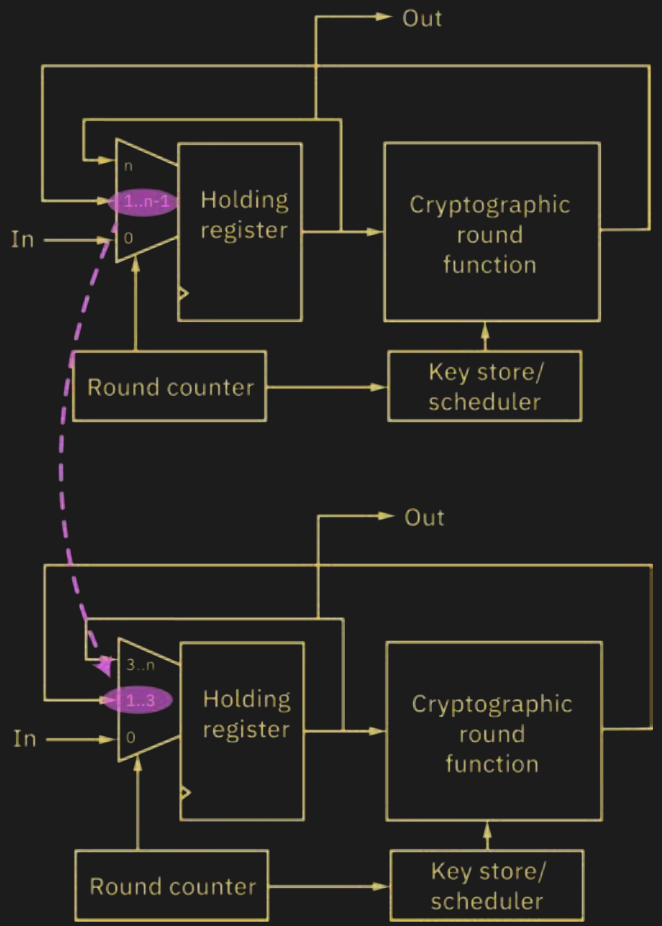
Background: Multi-Round Cipher



- Round "n"
  - Hold the result for read-out

```
0000 - load
0001 - round
0010 - round
0011 - round
0100 - round
0101 - round
0110 - round
0111 - round
1000 - round
1001 - round
1010 - round
1011 - round
1100 - round
1101 - round
1110 - round
1111 - hold
```

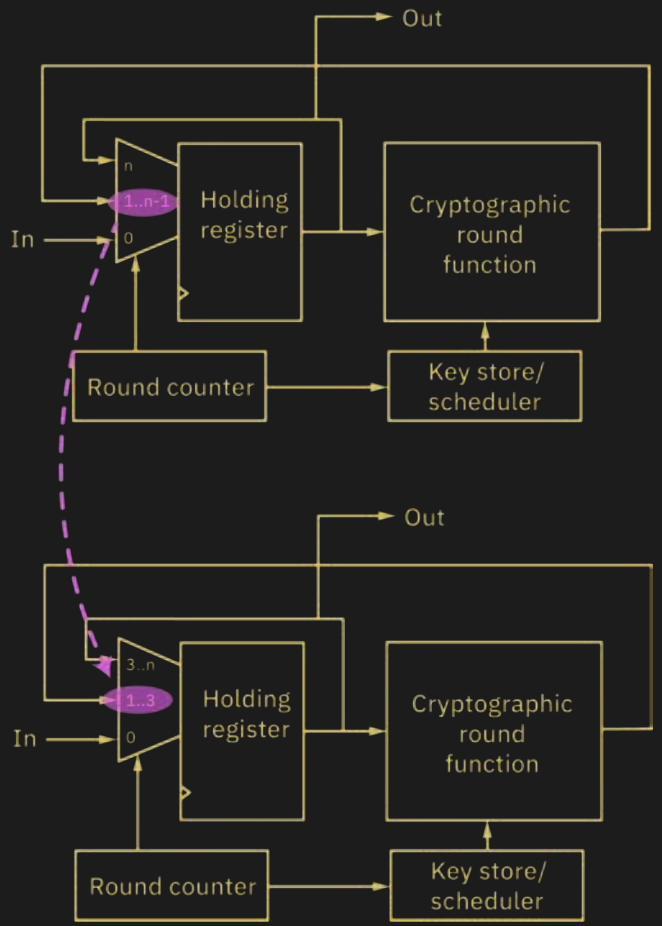
The Attack



- What if you tied the upper bits of the "holding register" selection input together?

```
0000 - load
0001 - round
0010 - round
0011 - round
0100 - round
0101 - round
0110 - round
0111 - round
1000 - round
1001 - round
1010 - round
1011 - round
1100 - round
1101 - round
1110 - round
1111 - hold
```

The Attack

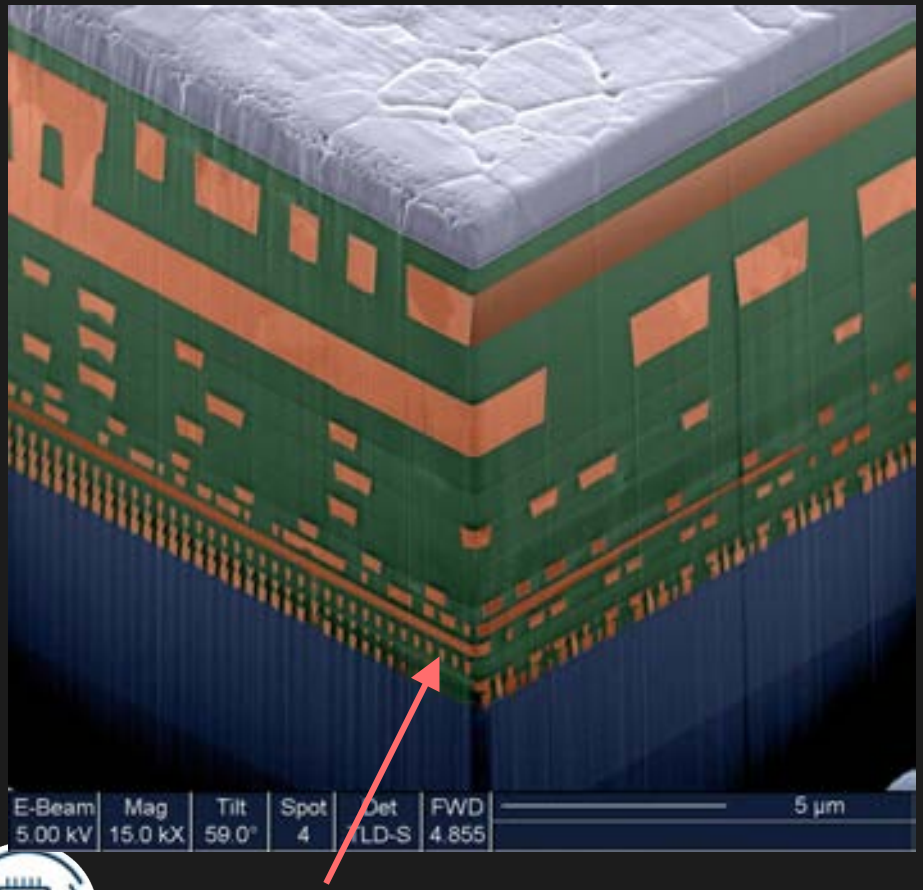


- What if you tied the upper bits of the "holding register" selection input together?

```
0000 - load
0001 - round
1110 - round
1111 - hold
0000 - load
0001 - round
1110 - round
1111 - hold
0000 - load
0001 - round
1110 - round
1111 - hold
0000 - load
0001 - round
1110 - round
1111 - hold
```

- Only 2 rounds matter!
  - But! Timing side channel and power side channel looks "as if" the full rounds happened

The Attack



- Observations:
  - Symmetric reduction of rounds -> decryption/encryption works "fine"
  - Sidechannels same or very similar
  - Reduced-round variants still have reasonable bulk statistics
  - If secret key is truly kept secret inside the chip...
    - ...Detection requires cryptanalysis of ciphertext
- Implementation is subtle:
  - Maybe just a via-only change!

Part 1 Summary: Classification System

Level 3: Detected only with $1mm+ tools and/or requires new techniques

Level 2: Detected with $10k-$100k tools

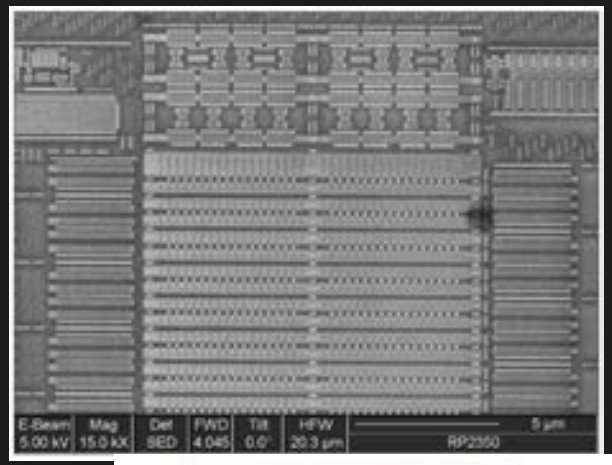
Level 1: Detected with $1k-$10k tools

Level 0: Detected with <$1k tools

- Current state of practice:
  - Level 3: maybe destructive analysis required???
  - Level 2: academic papers
  - Level 1: practiced by targeted industries
  - Level 0: routinely practiced

Part 2, Supply Chain Verification: Improving State of Art vs State of Practice

- State of Art



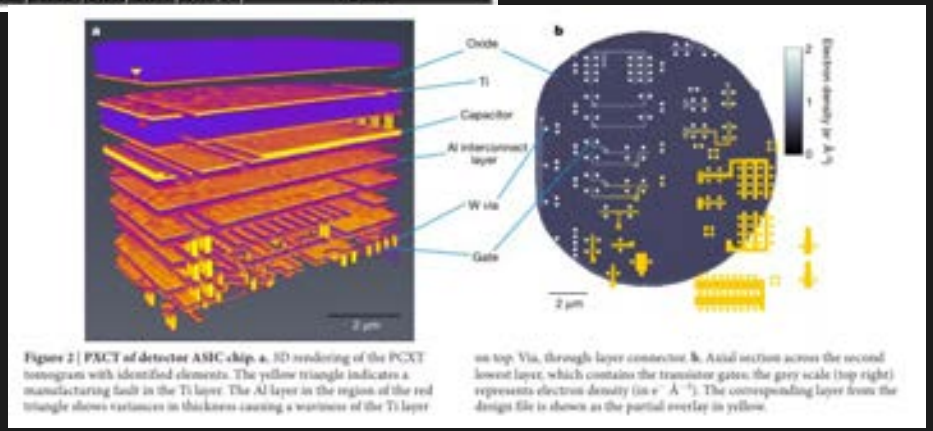
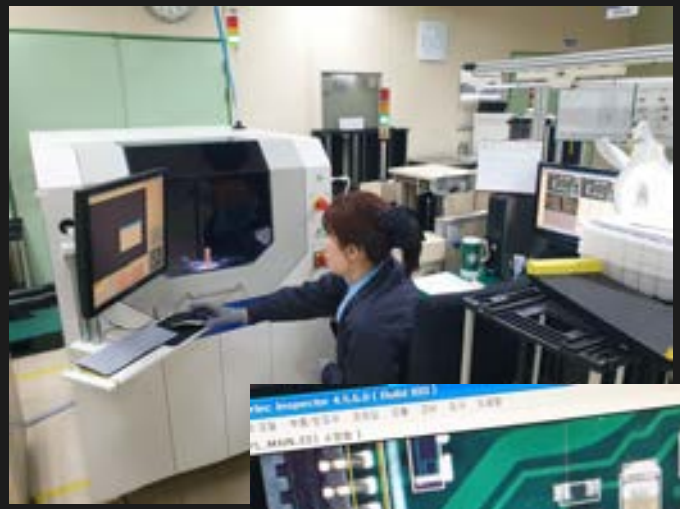
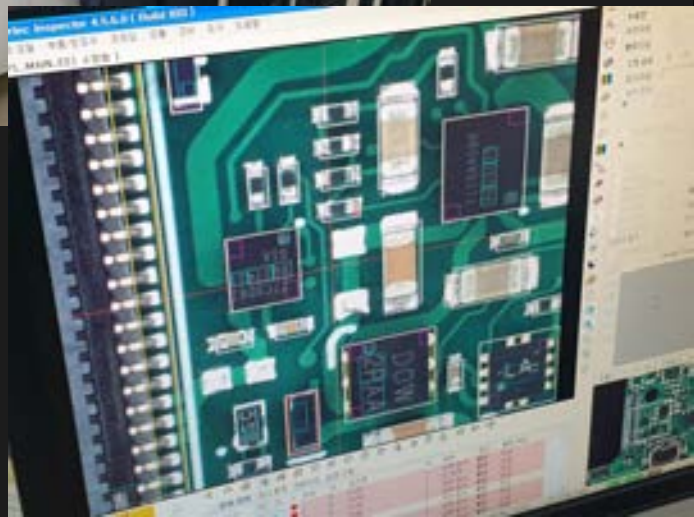


Figure 2 | PXCT of detector ASIC chip. a. 3D rendering of the PCXT tomogram with identified elements. The yellow triangle indicates a manufacturing fault in the Ti layer. The Al layer in the region of the red triangle shows variances in thickness causing a waviness of the Ti layer on top. Via, through layer connector. **b.** Axial section across the second lowest layer, which contains the transistor gates; the grey scale (top right) represents electron density (in e⁻ Å⁻³). The corresponding layer from the design file is shown as the partial overlay in yellow.

- State of Practice





In Practice, Nobody is Checking

Nobody is checking

A few people are checking

- The general public does not check chips beyond Level 0
  - Public companies that do check also do not disclose problems
  - Disclosing supply chain issues is bad for business
- Threat actors have broad latitude to operate without consequence

The Importance of Research Translation

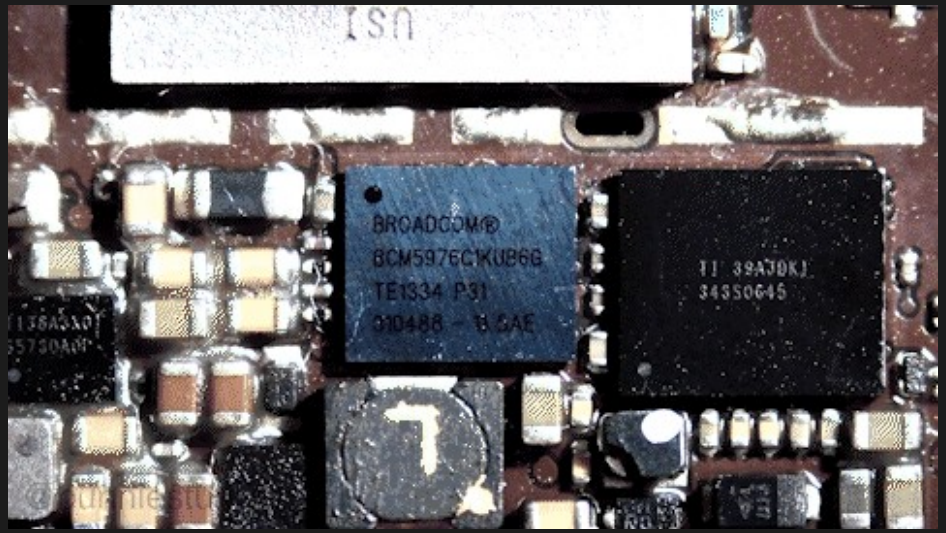
Academics & agencies

Targeted Industries

Point of Use (at-home)

- Reducing deployment costs makes more attacks detectable
  - *Improves* trust in hardware for everyday people

Improving State of Practice: Translating Backside Silicon Imaging From Research To Practice

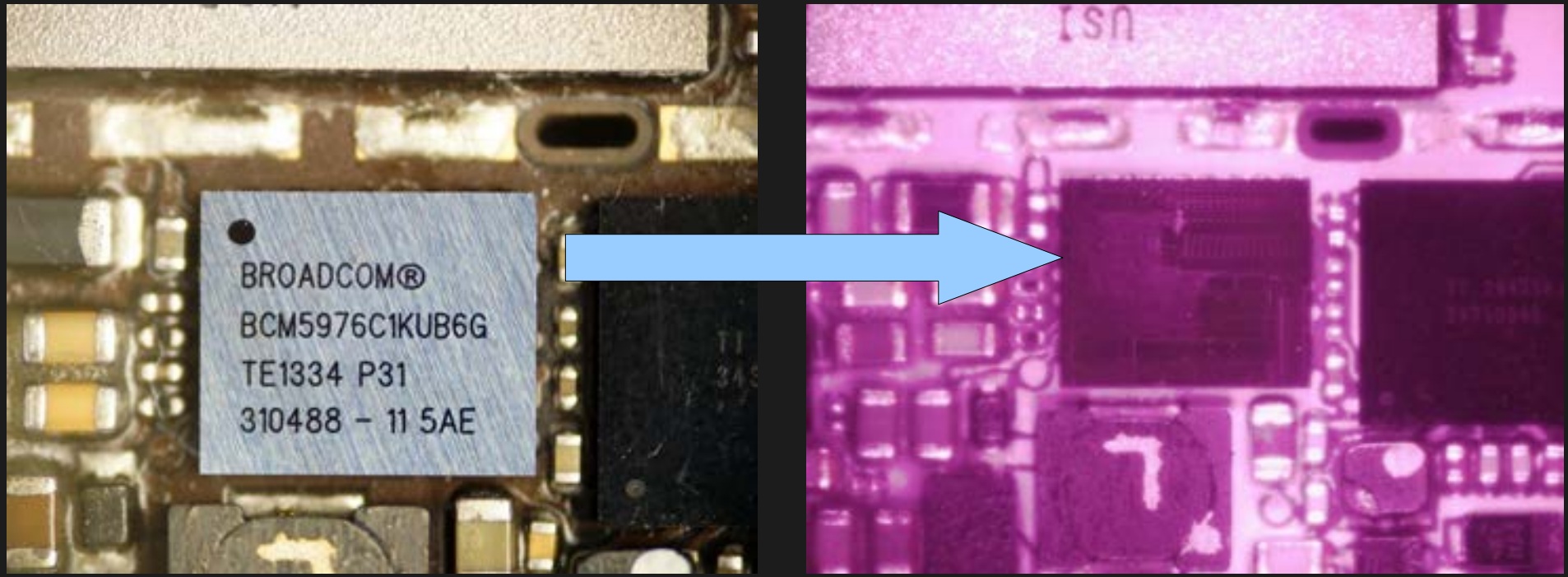


- Infra-Red, *in situ* (IRIS) Verification of Silicon
  - A method for inspecting certain types of chips
  - After they are attached to a circuit board
  - Without damage

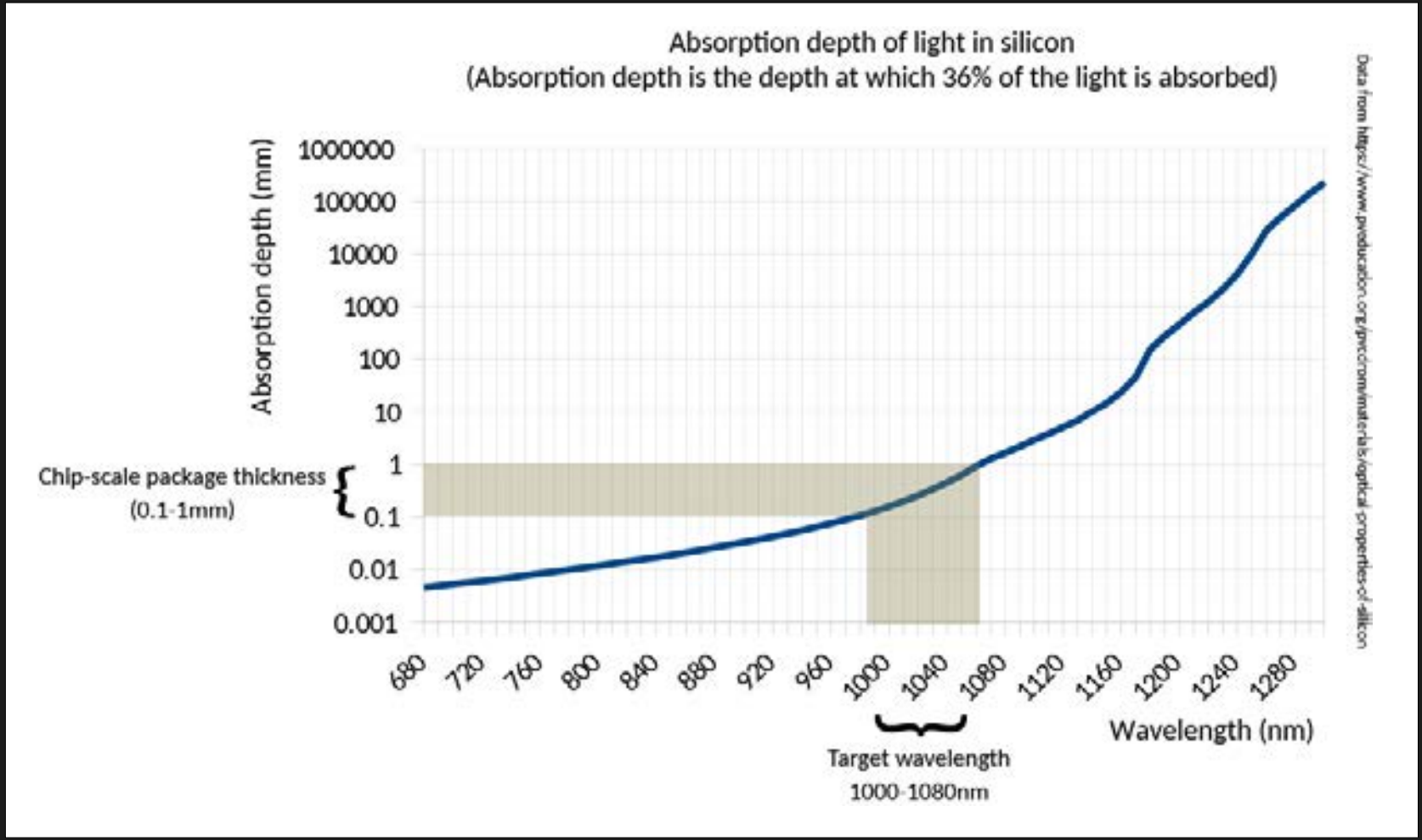
What Type of Chips?

- Short answer: "The shiny ones"
  - WLCSP or FCBGA types of packages
  - Exposed silicon back with no film or paint applied
  - Ideally polished and/or thinned
  - P- (lightly) doped substrate
    - TSMC-like foundry
    - P+ doped substrate (Intel) scatters light, requires lasers
      - Lasers ~$100, LEDs ~$0.10
- Does not work for chips in plastic packages
  - Manufacturer must "design for inspectability"

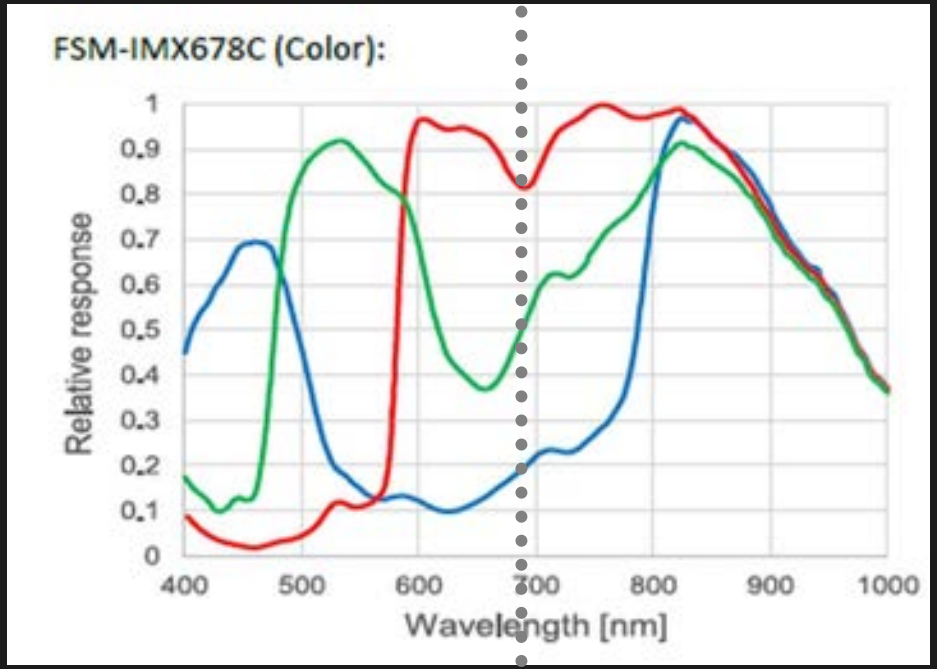
Review:
Silicon is Transparent to Infrared Light



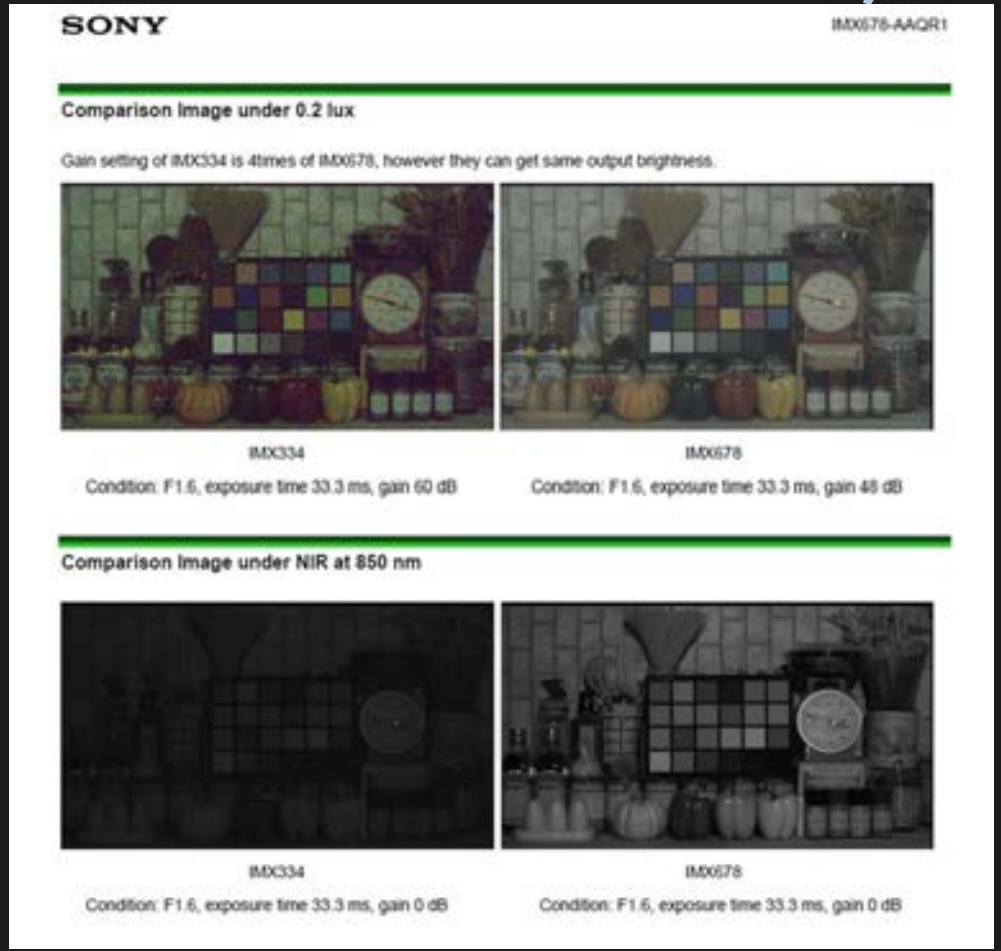
Silicon is Transparent to Infrared Light



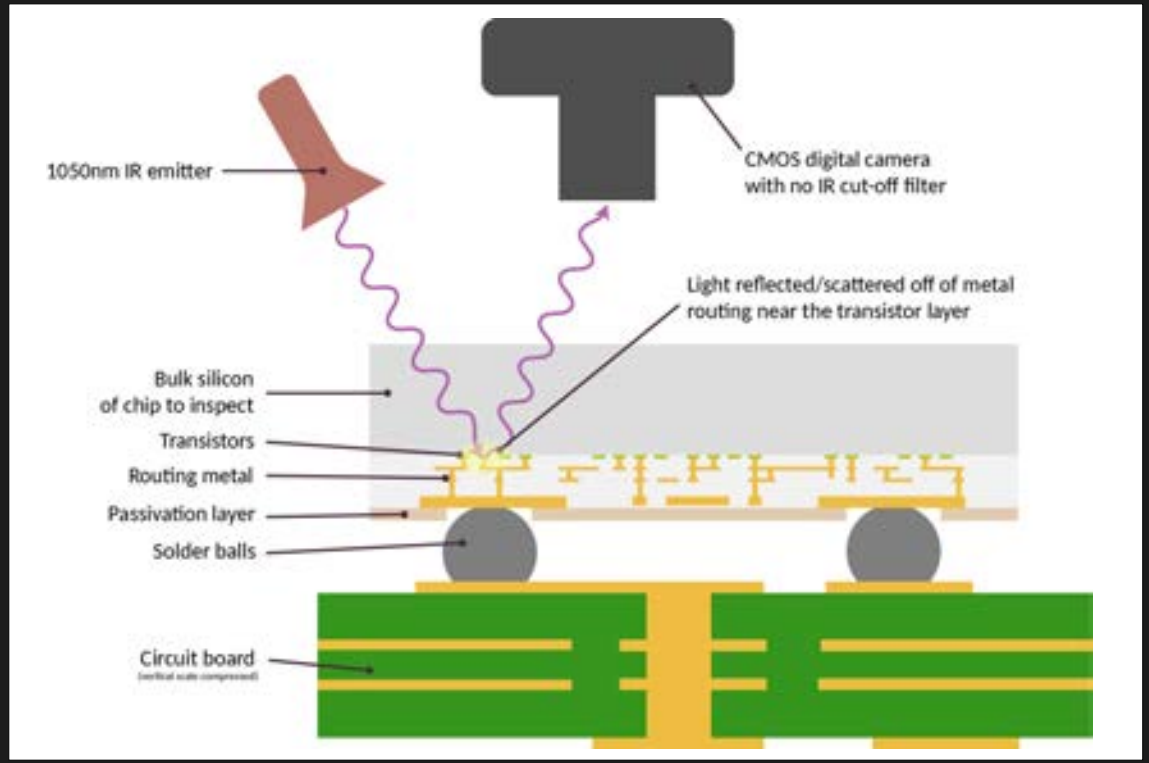
Some Commodity CMOS Sensors are IR-Enhanced (e.g.: Sony Starvis2 → Surveillance Market, ~$10)



visible infrared



Putting it All Together: IRIS



- Inspection of chips from the back side
- After they have been assembled into a product

Prior Work

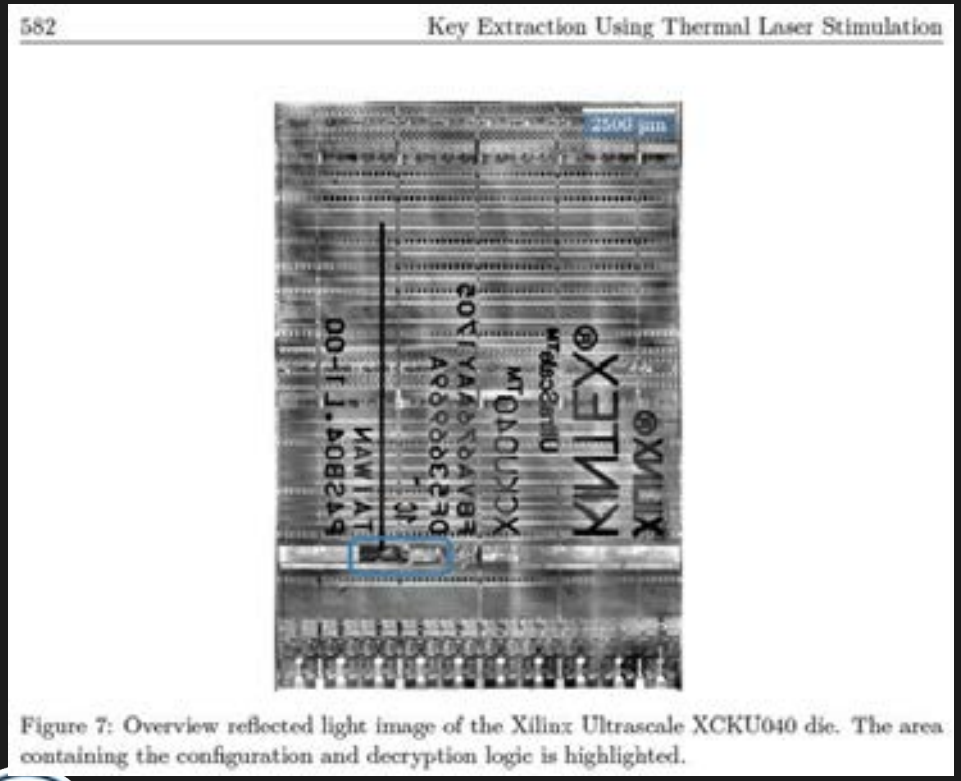
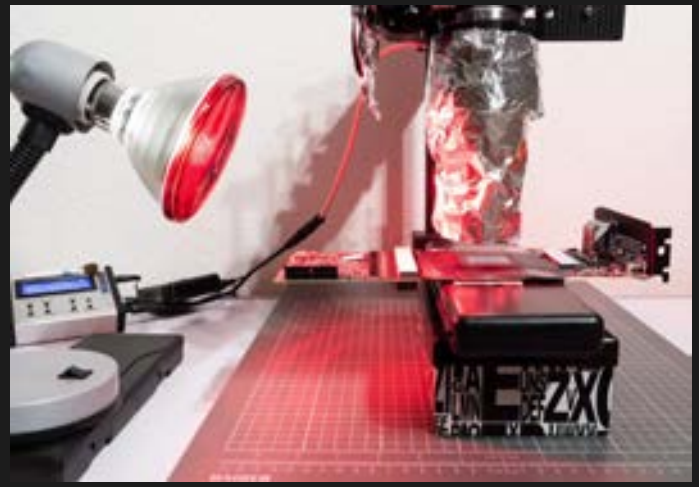

582 Key Extraction Using Thermal Laser Stimulation

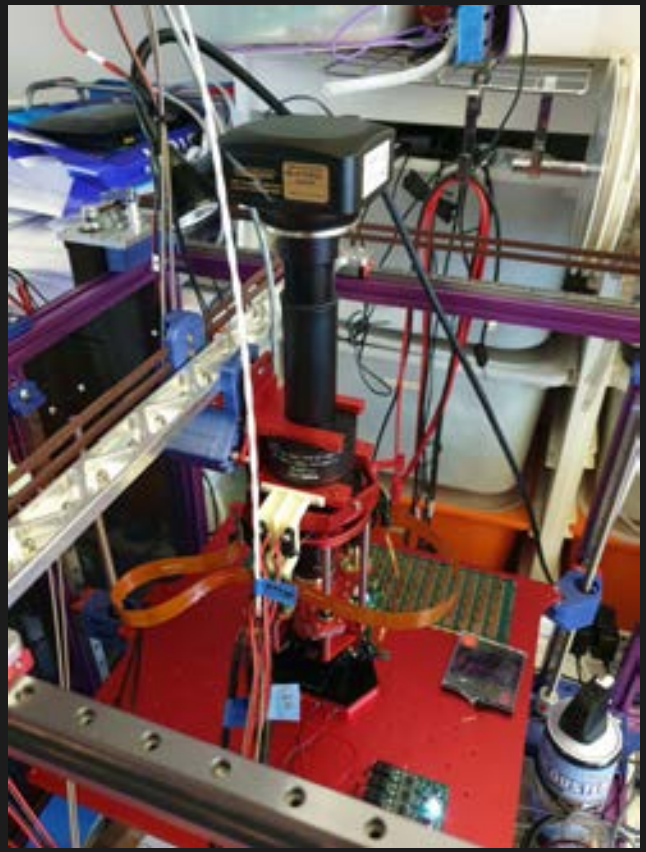
Figure 7: Overview reflected light image of the Xilinx Ultrascale XCKU040 die. The area containing the configuration and decryption logic is highlighted.

- IR backside imaging is a well-established lab technique
- Fritzchens Fritz flickr feed
  - Backside IR imaging with CMOS camera



IRIS Implementations

<EUR300
fully manual adjustments

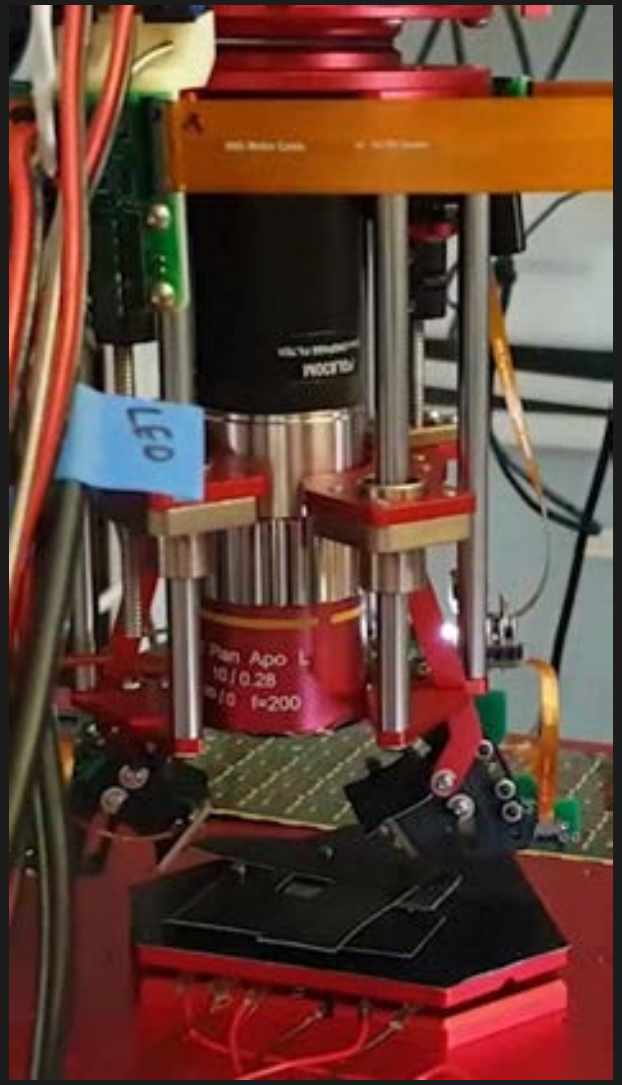


~EUR5000, fully automatic adjustments

Manual Adjustment

- Possible to generate high quality images
- Fussy to set up
- Repeatability issues

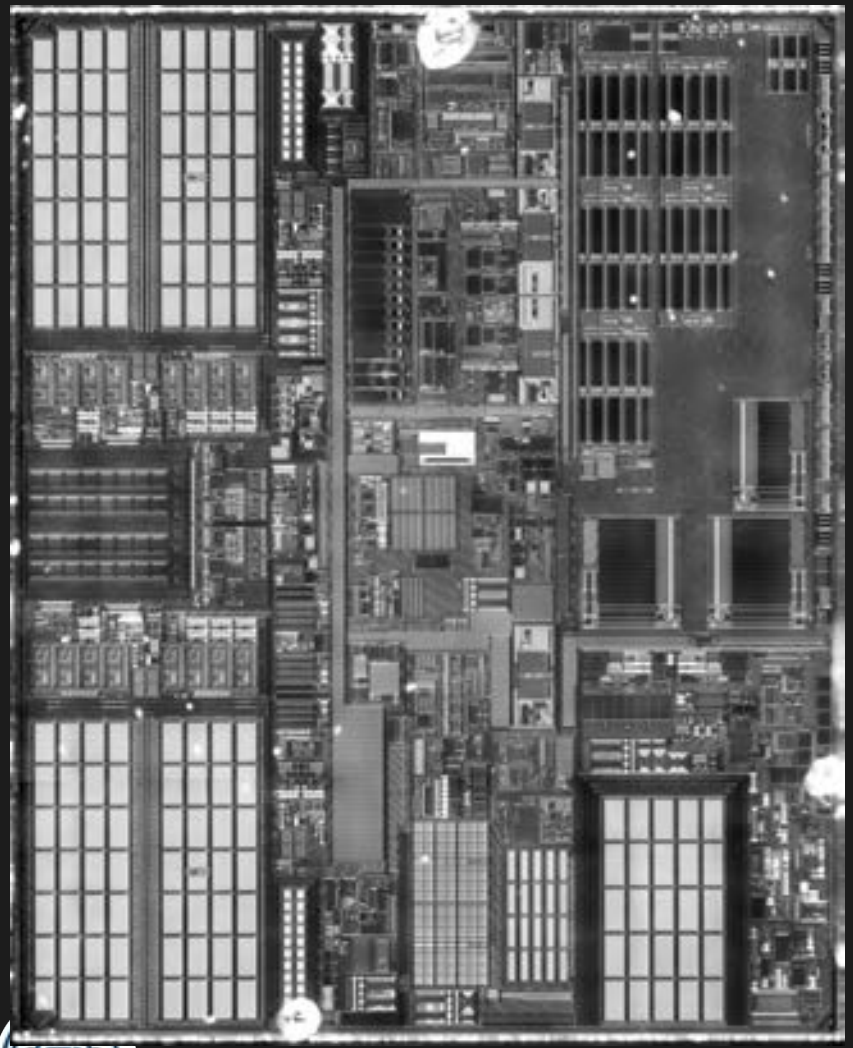
- Useful for end-user verification setups
  - Lower cost
  - More effort, but used rarely - only when new chips are acquired



Automated Adjustment

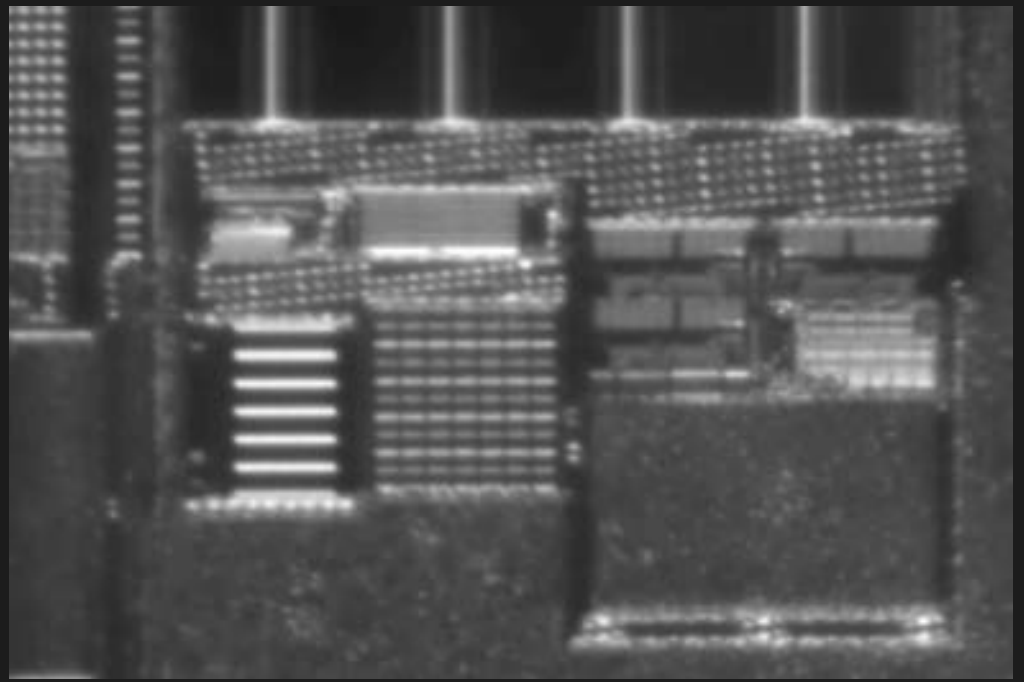
- <10 micron precision repeatability
- Fully automated X/Y/Z positioning
- Fully automated light positioning
- Good repeatability
- Useful for
  - Generating reference images
    - Higher quality images used as comparison point for end users
  - Higher throughput screening
  - Higher confidence measurements

Chip Features vs. Angle of Incident Light

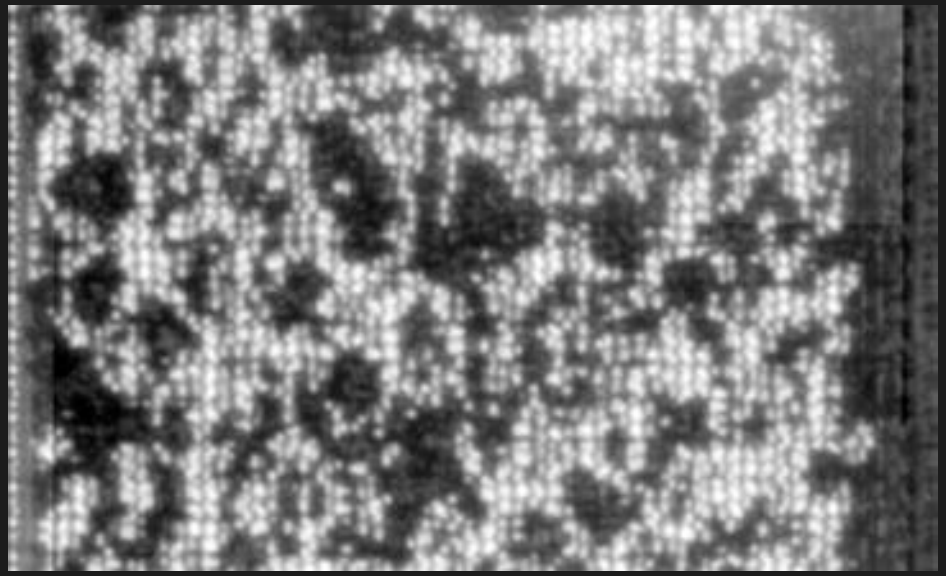


Sharing Data

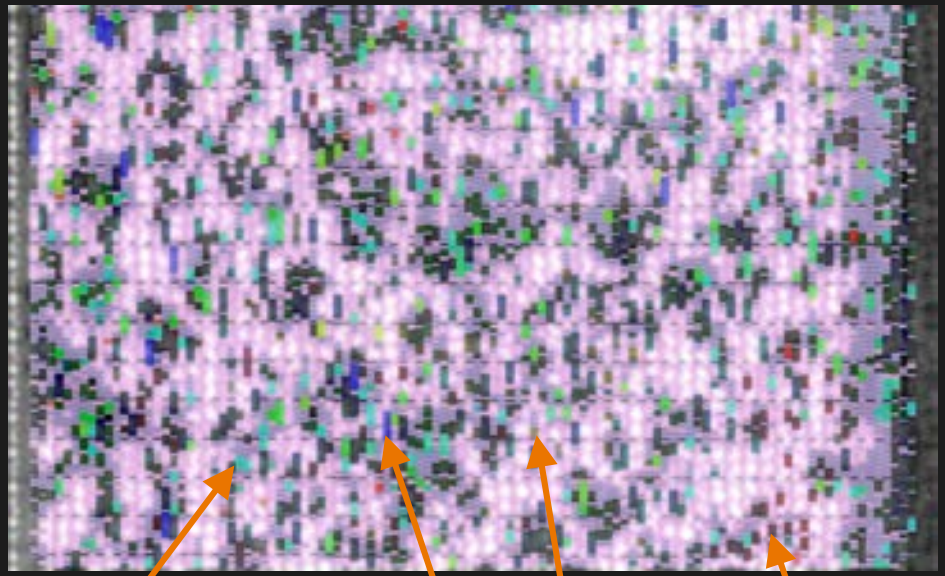
- https://siliconpr0n.org/archive/doku.php?id=tag:collection_bunnie&do=showtag&tag=collection_bunnie



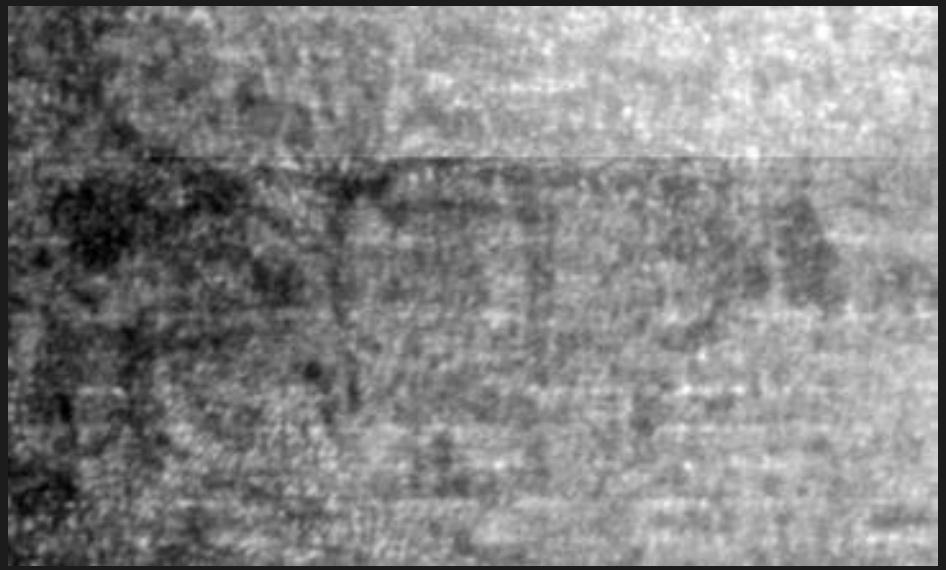
IRIS Examples: Seeing Standard Cells

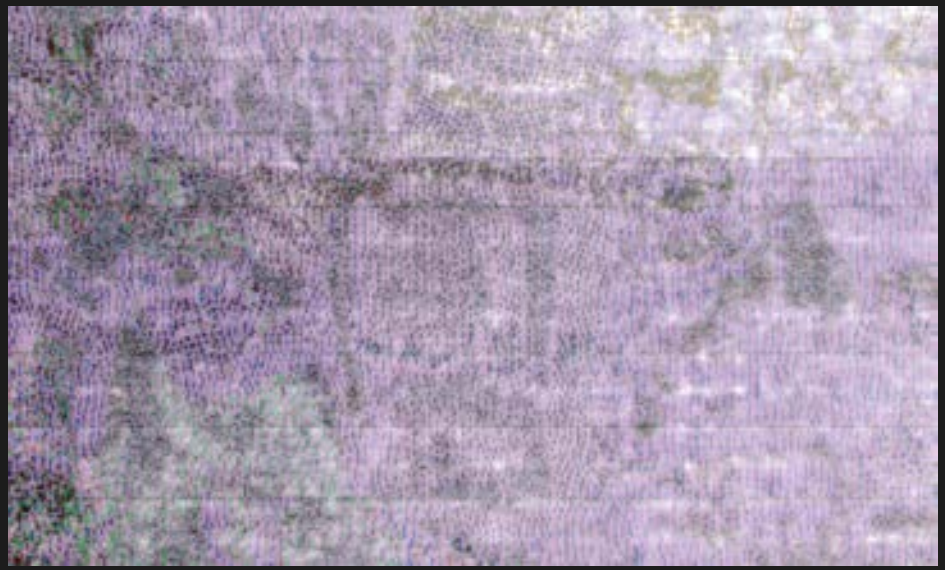


SKY130 process



More Standard Cells

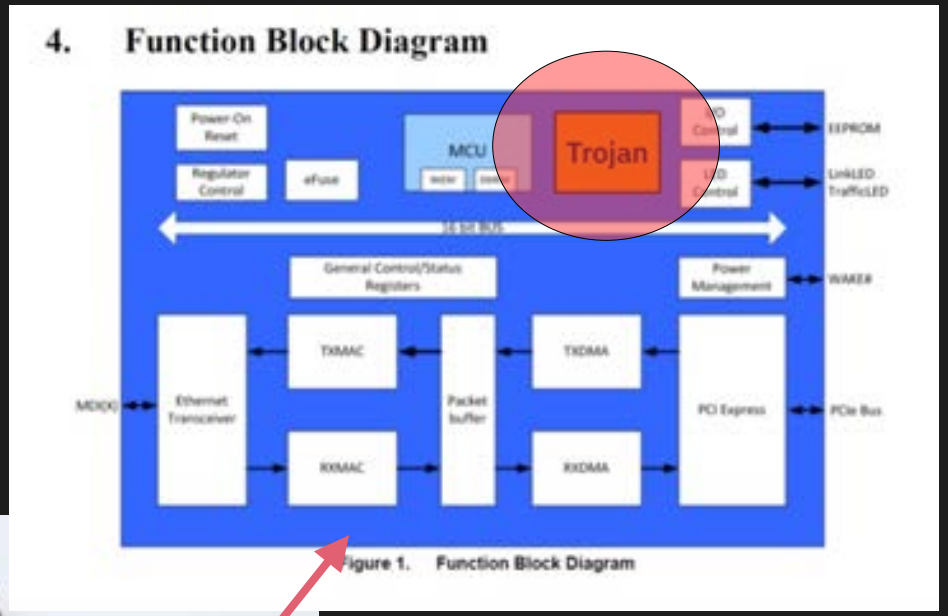




TSMC 22nm process, same scale as SKY130 on previous slide

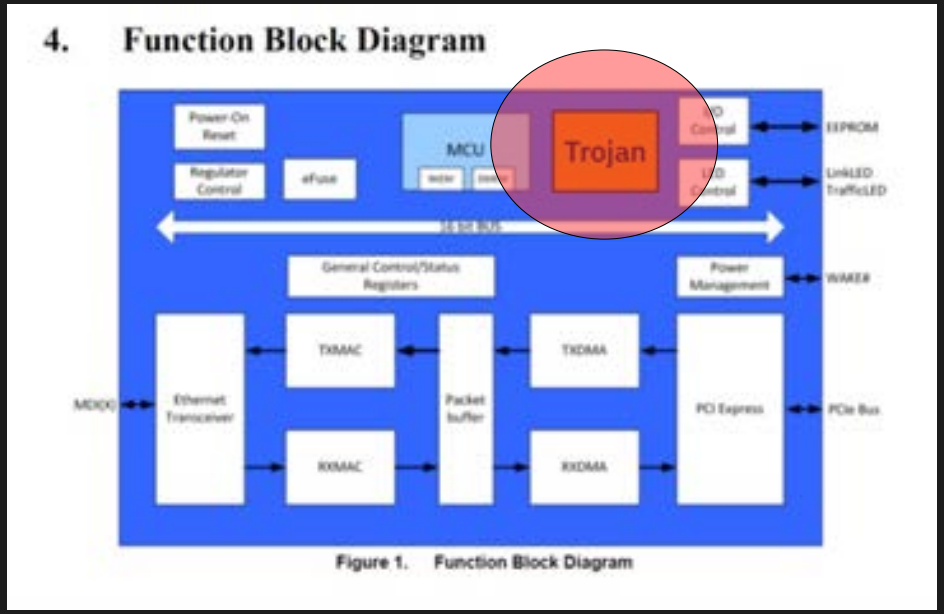
So, What Does IRIS Get Us?

Level 1: Block-Level Modification



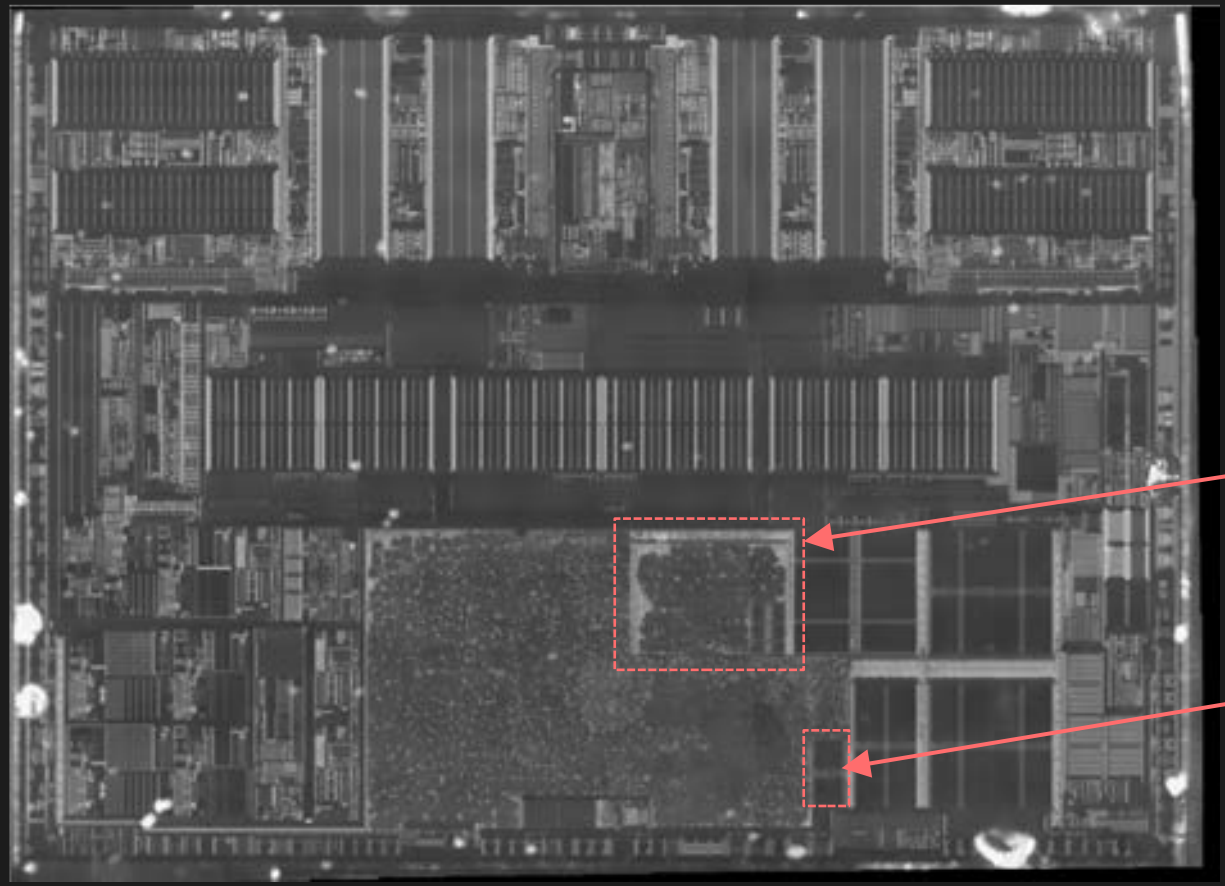
- If chip in WLCSP package:
  - Easy to "diff out" block-level modifications
  - Would need reference images, possibly crowd-sourced

Grounding a Hypothetical Trojan



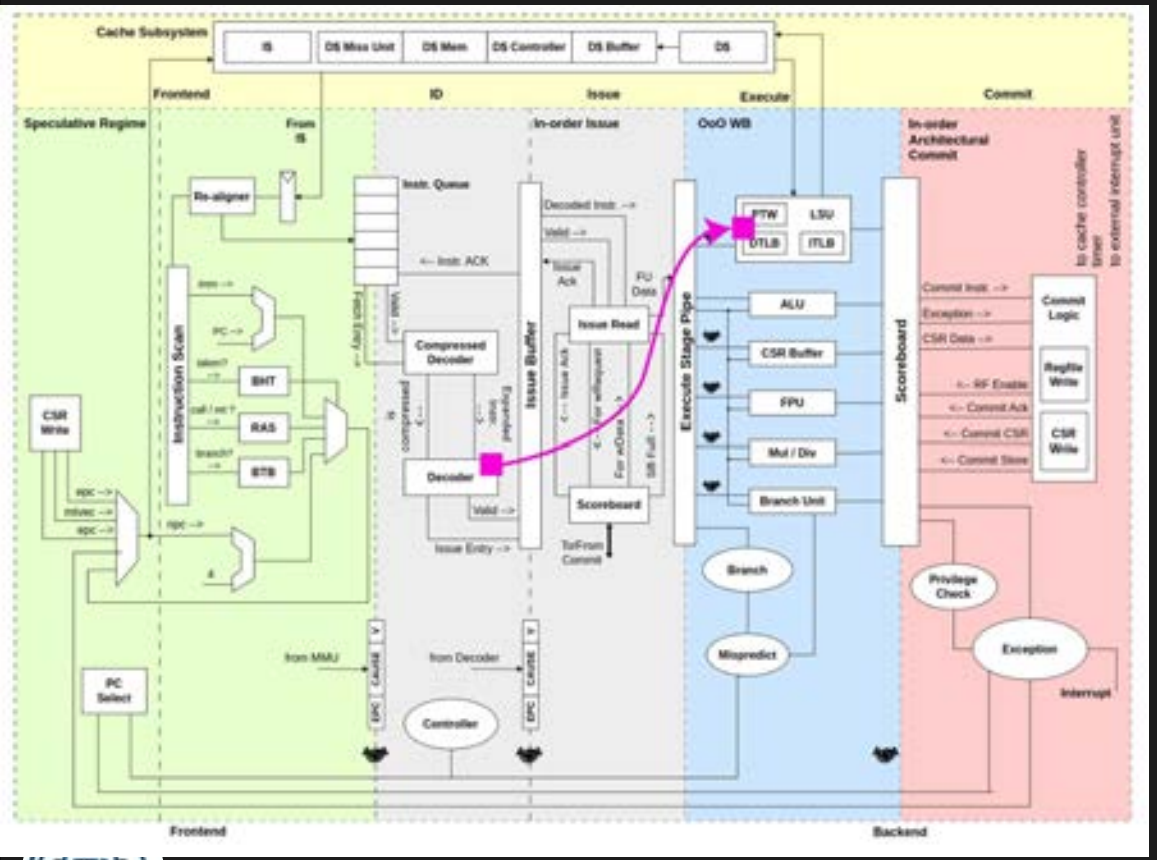
- Hypothetical "Trojan":
  - Records ~few kiB of network traffic
  - Has a trigger
    - Say, respond to ICMP secret knock to exfiltrate data

Example of Block Sizes



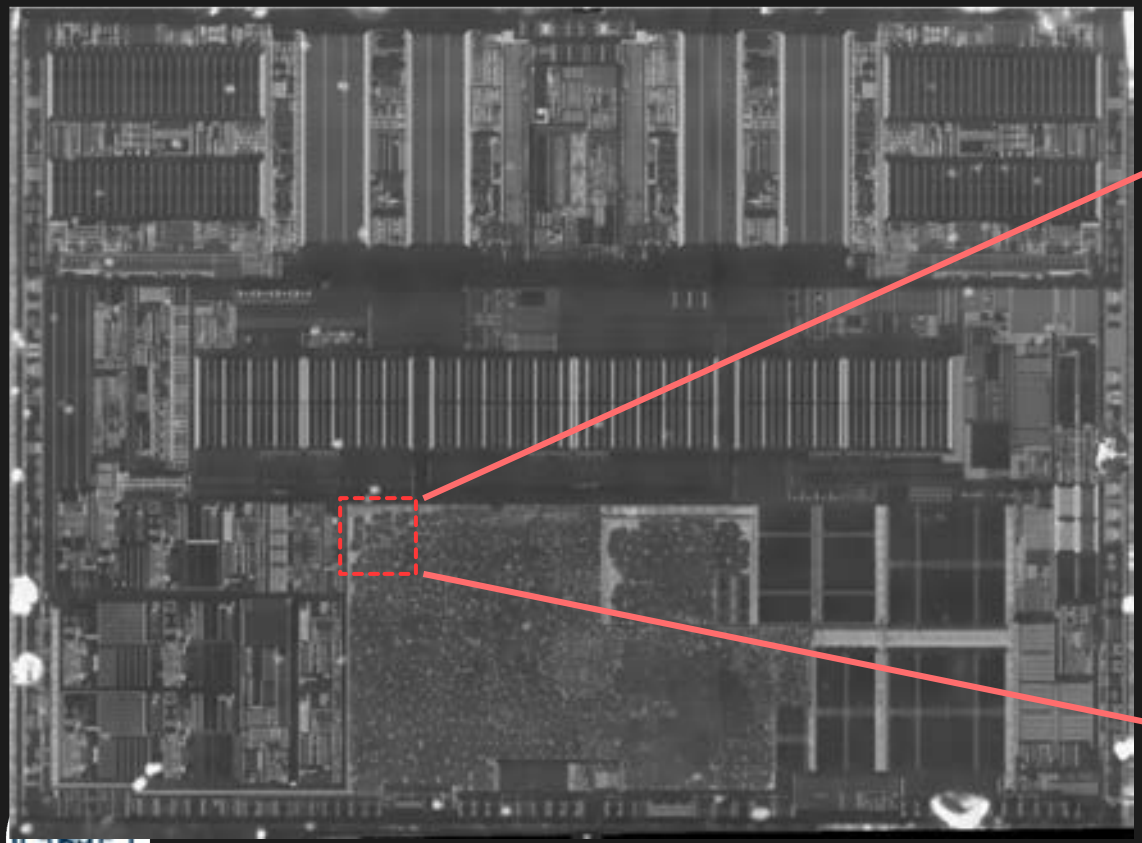
Estimated @ 65nm node

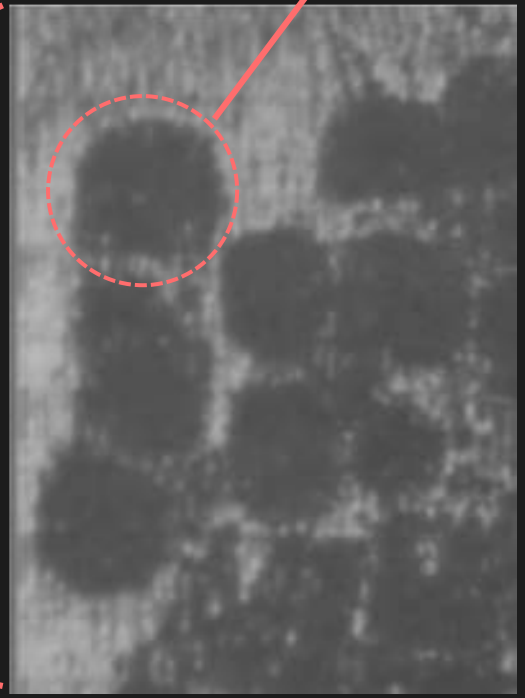
Level 2: Small RTL Modifications



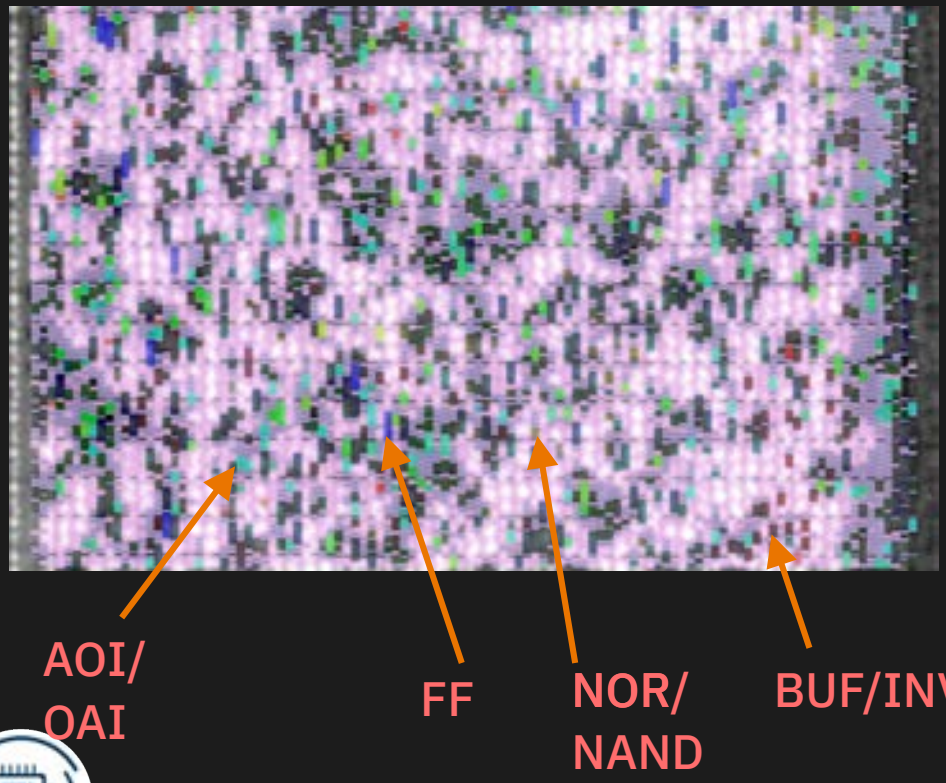
- "Probably detectable"
  - Naive RTL insertion would have place/route deviations
  - Recall from earlier discussion:
    - O(10)-O(100) cells added

Example of Place & Route Logic Patterns



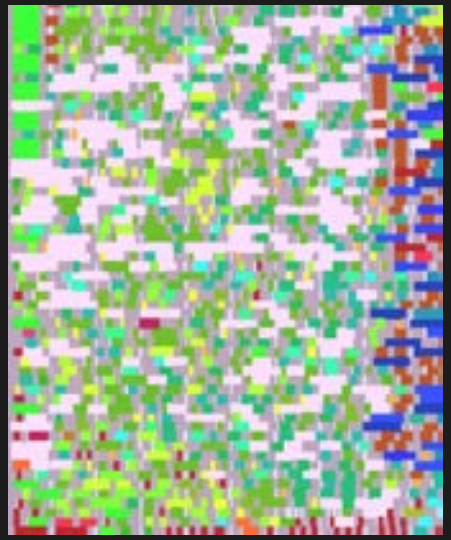


Limitations of Comparing IRIS Images



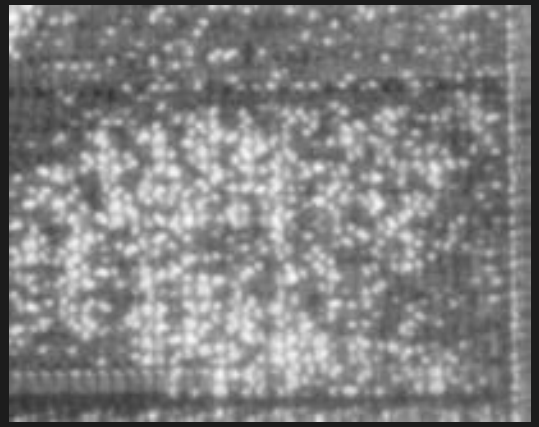
- Logic gates show up as fuzzy blobs "by type of gate"
  - In reality we can only know "how many gates"
  - "Exactly what gates" may be spoofable
- An omnipotent adversary could "lock down" place/route paths to maintain net shape, logic cell types
  - Would leave some trace, e.g. reduced timing margin, power consumption changes

Related Work in Progress: Automated Gate Count Census

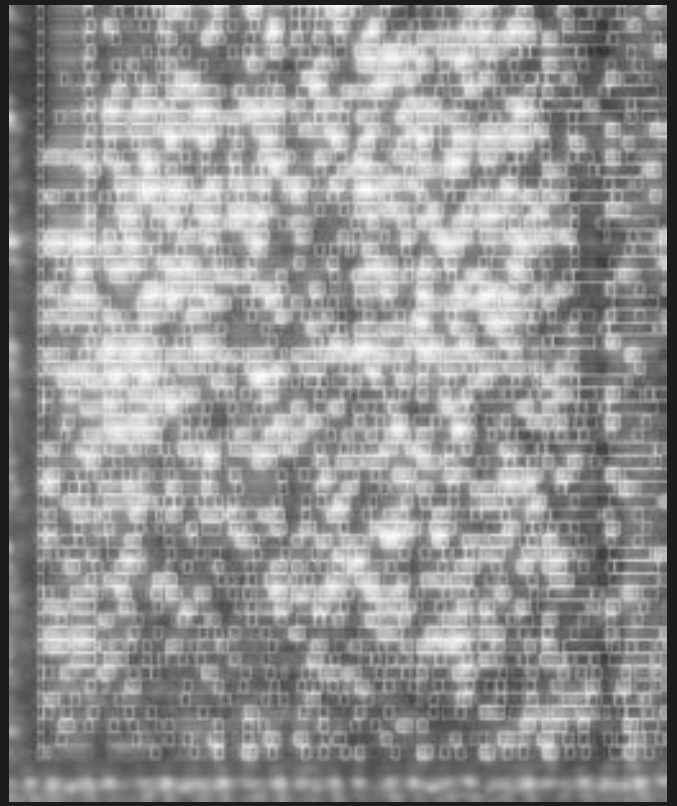


Design data
(standard cell map)

+

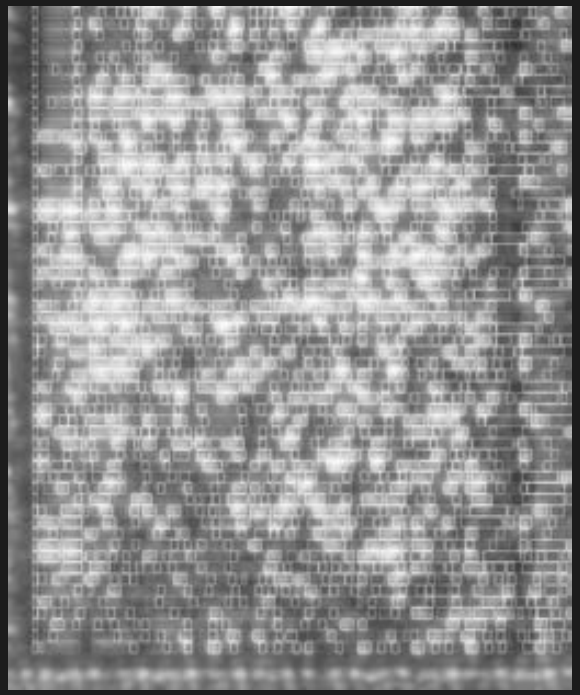


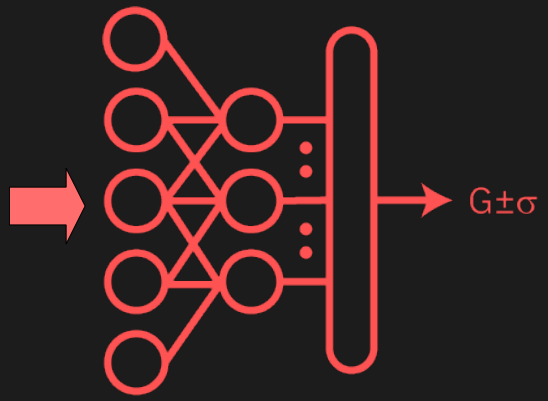
Imaging data
(arbitrary rotation &
translation)



Aligned cell-to-image map

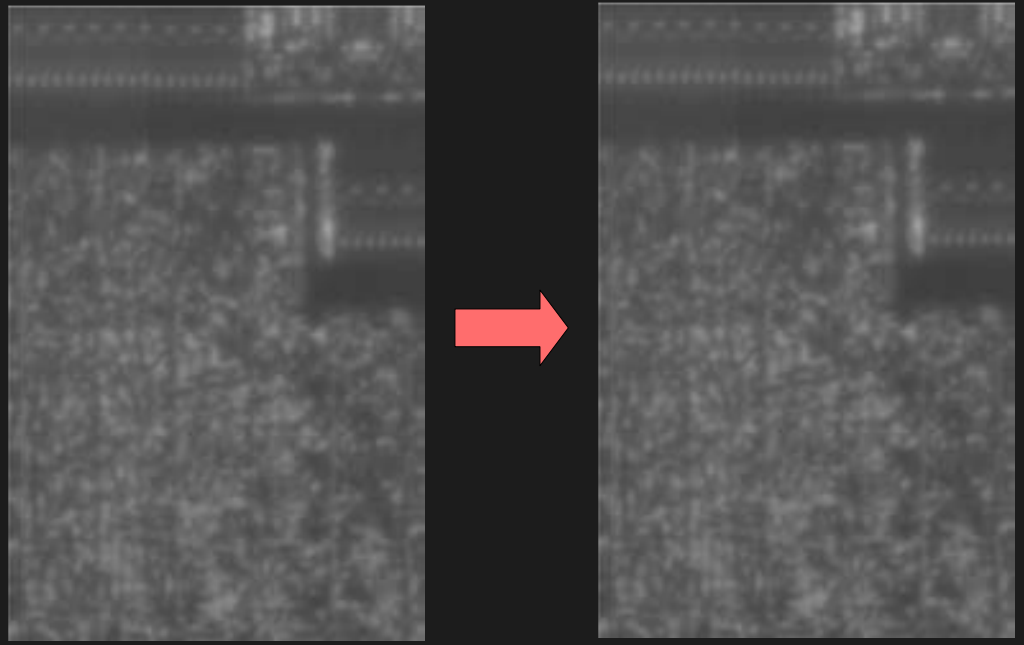
Quantifying Gate Counts





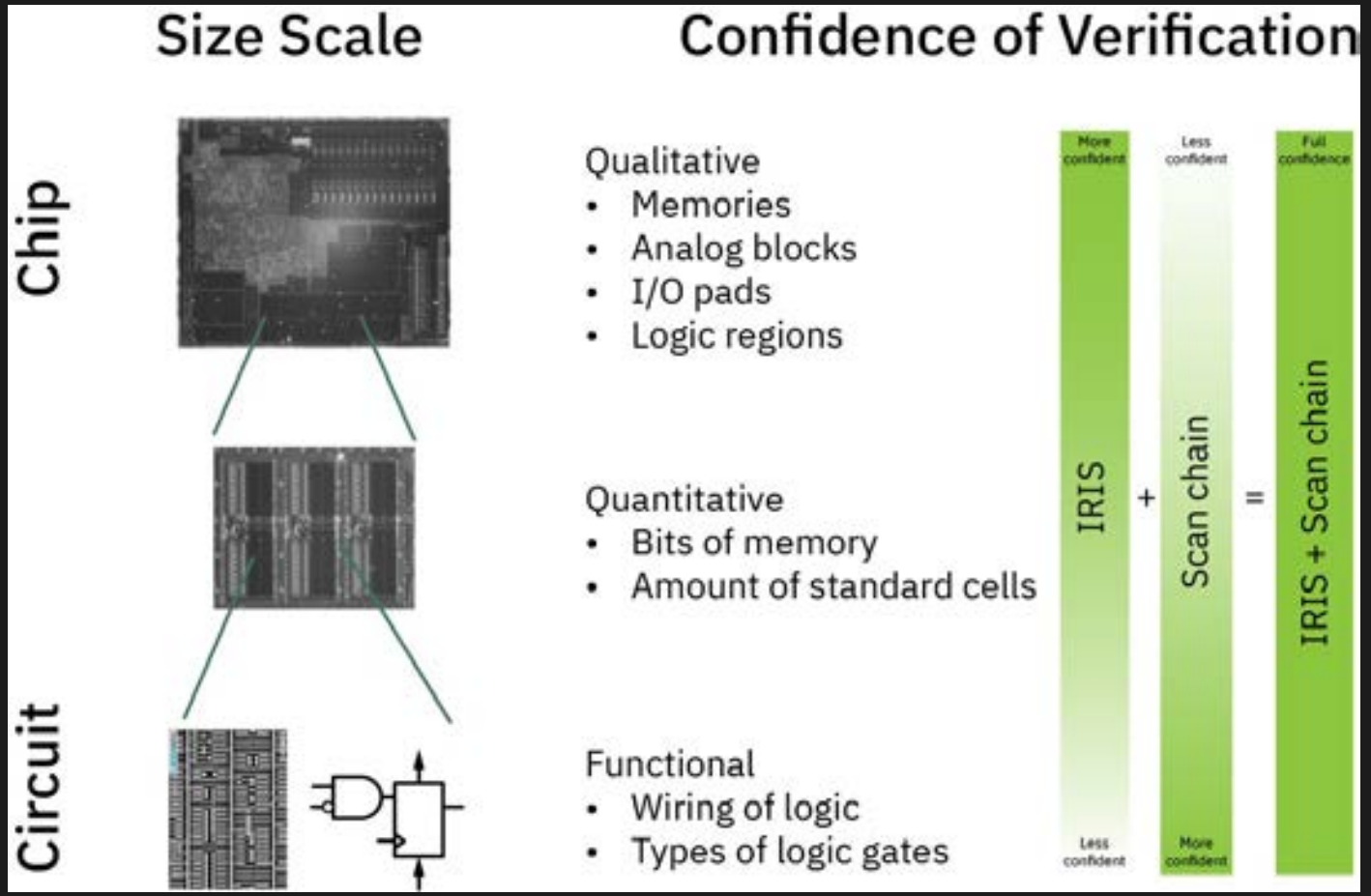
- Trying to train a CNN classifier to estimate gate count
  - "G" plus/minus an uncertainty of "sigma"
    - Uncertainty due to noise, dirt, scratches, process variations...
  - Bonus if it can classify types of logic cells

Level 3: Targeted Mask Modifications



- No difference in images, by attacker's intention
  - Modifications solely on mid-level metal layers
  - No extra logic gates, but functionality is changed
    - "Spare cells" possibly used for malicious purposes

Next Steps: Hybrid Verification?



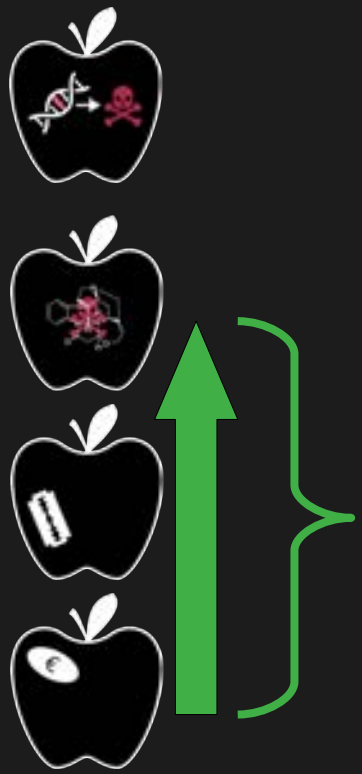
Even If We Can't Get to 100% Confidence: IRIS is Better than Just Trusting The Label

~100MiB image of chip

>>

~64 bytes of text labeling

Recap: Improving Trust in Supply Chains



Still some things we can't catch, but...

...IRIS could raise the bar

- #1 Simplify the Discussion
  - Categorize attacks by level
  - Graded by cost to detect an attack
- #2 Reduce cost of detection
  - Reduce gap between "state of art" and "state of practice"
  - Share data so we have baselines

Thank You!

@bunnie@treehouse.systems
@bunniestudios.bsky.social

INFRARED IN SITU

With thanks to:

nlnet FOUNDATION

NGI ZERO ENTRUST

Github sponsors:

https://bunnie.org/iris