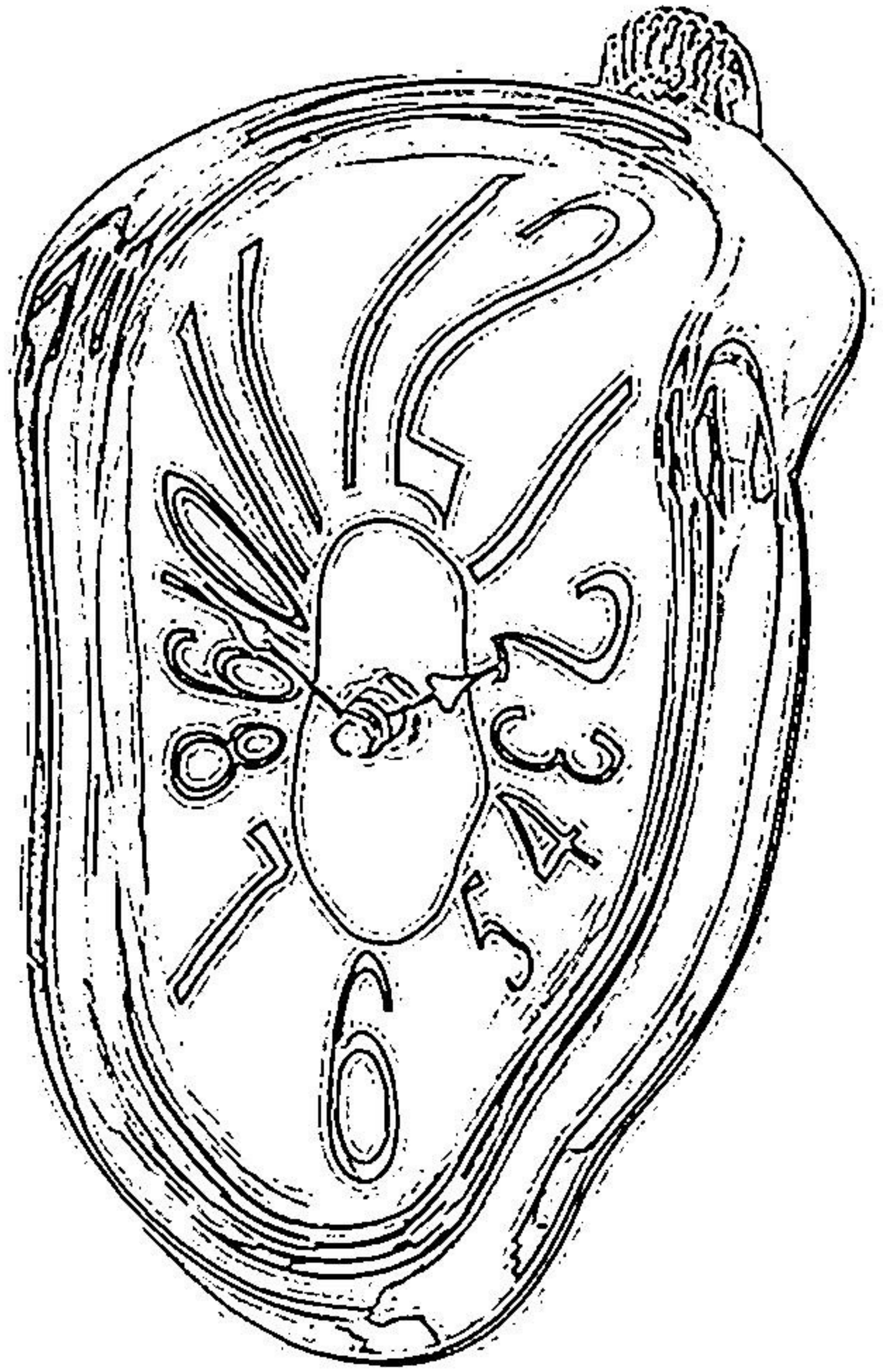
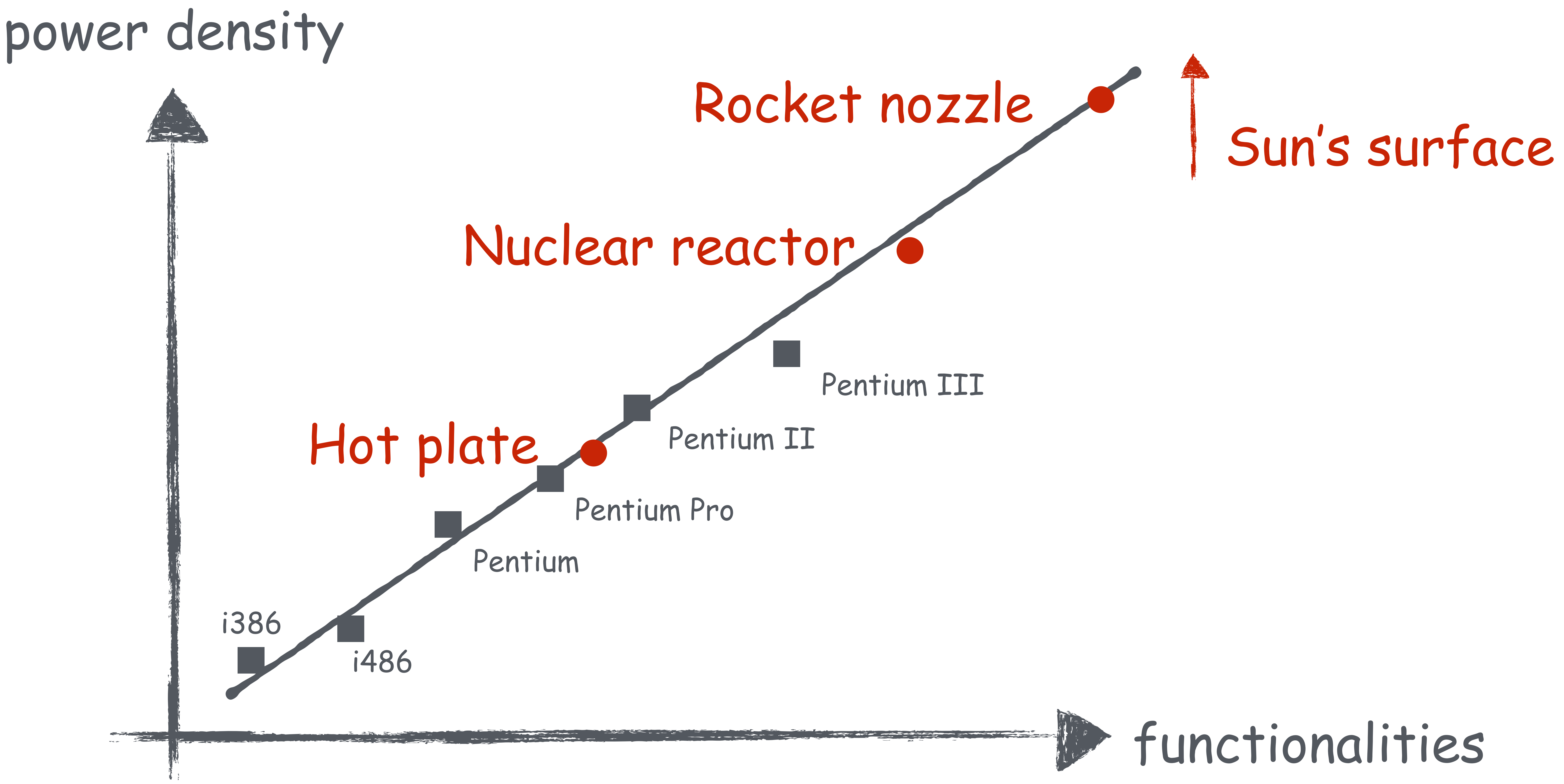


CLKSCREW

Exposing the Perils of Security-Oblivious Energy Management

Adrian Tang, Simha Sethumadhavan, Salvatore Stolfo

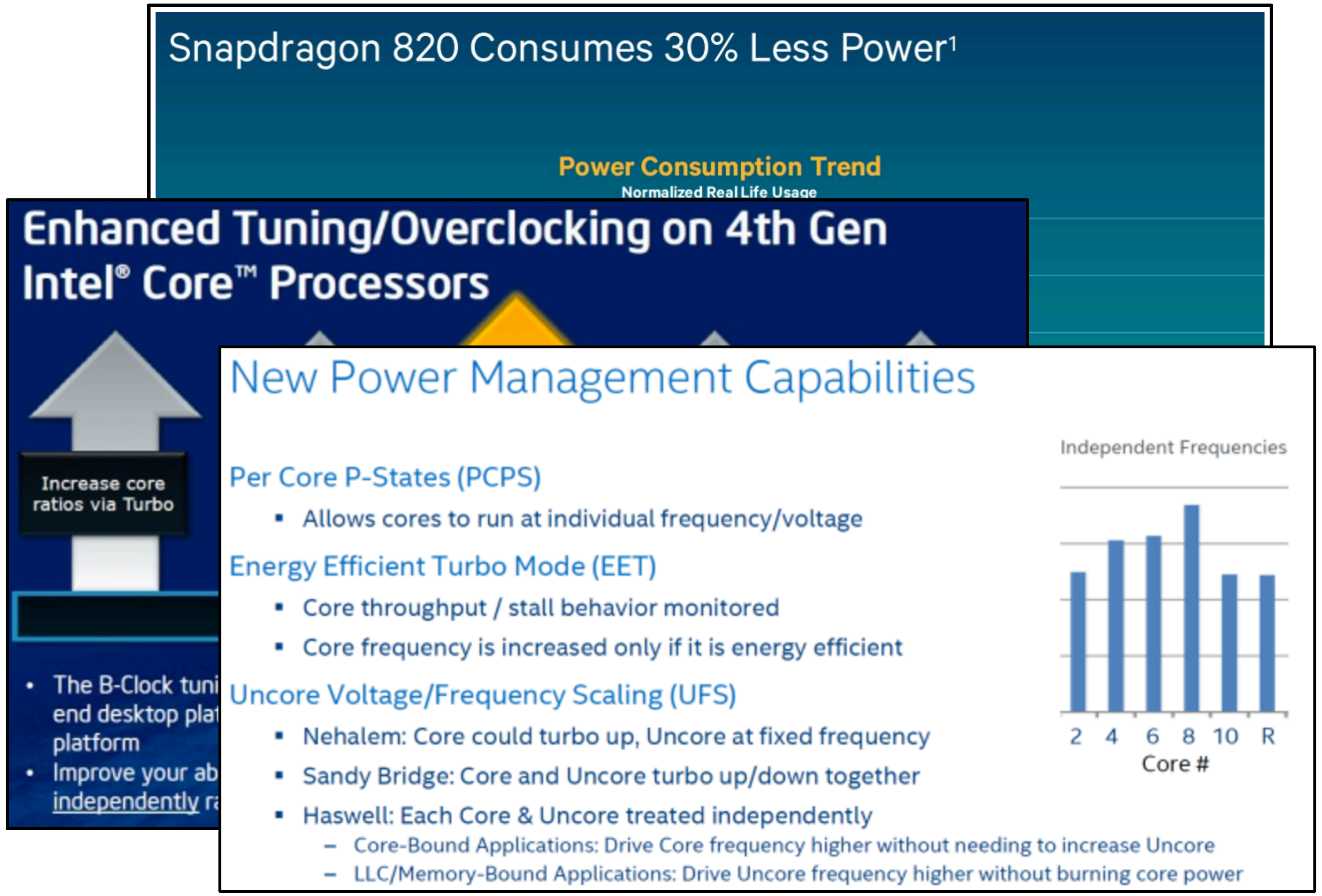
Today's systems cannot exist without **Energy Management**



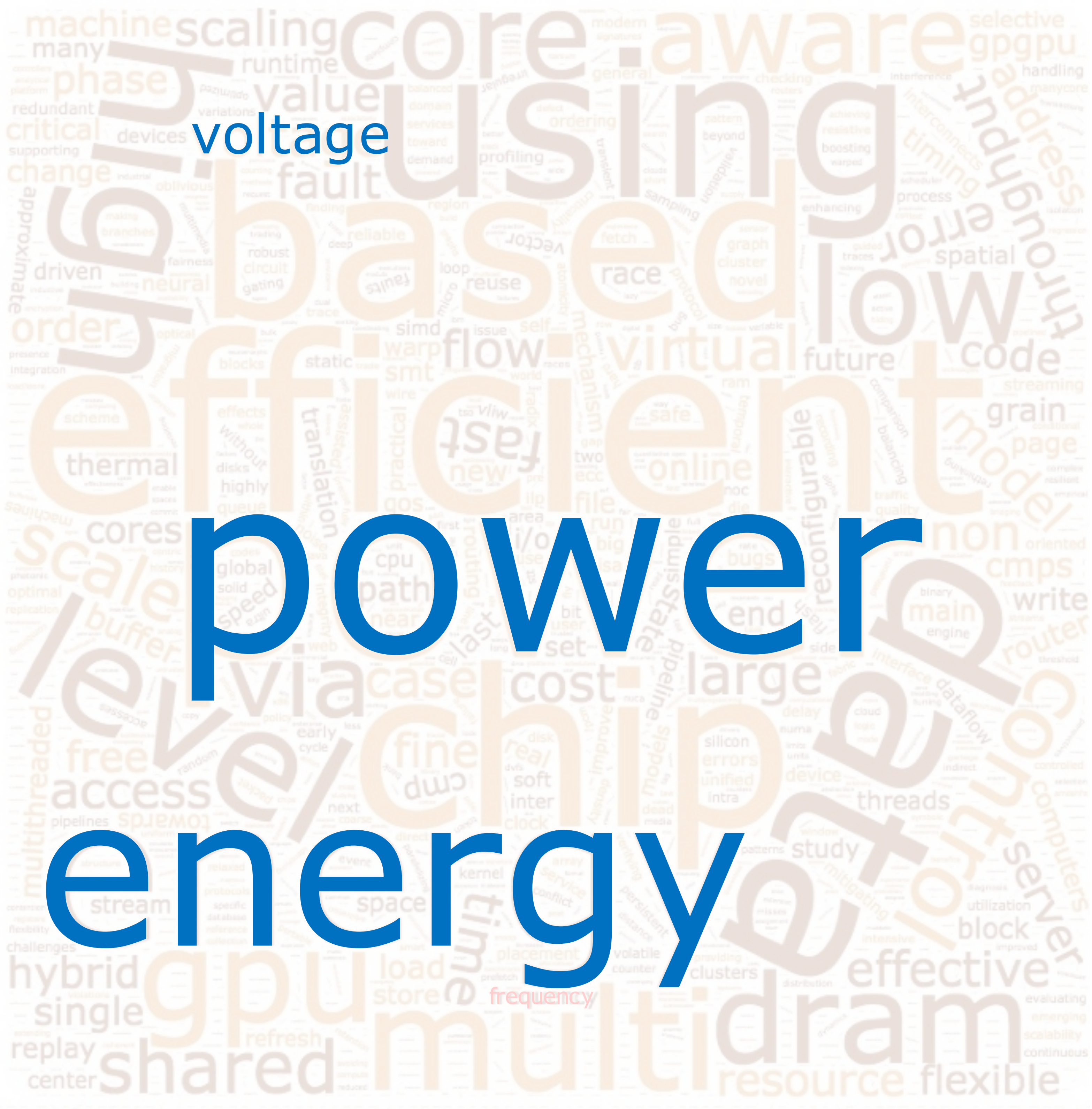
Source: Adapted from S. Borkar (Intel)

Today's systems cannot exist without **Energy Management**

Industry



Academia



Source: Word-cloud from ISCA, ASPLOS, MICRO, HPCA (2000 - 2016)

Pervasive

Essential

Today's systems cannot exist without

Energy Management

Complicated

stay secure with

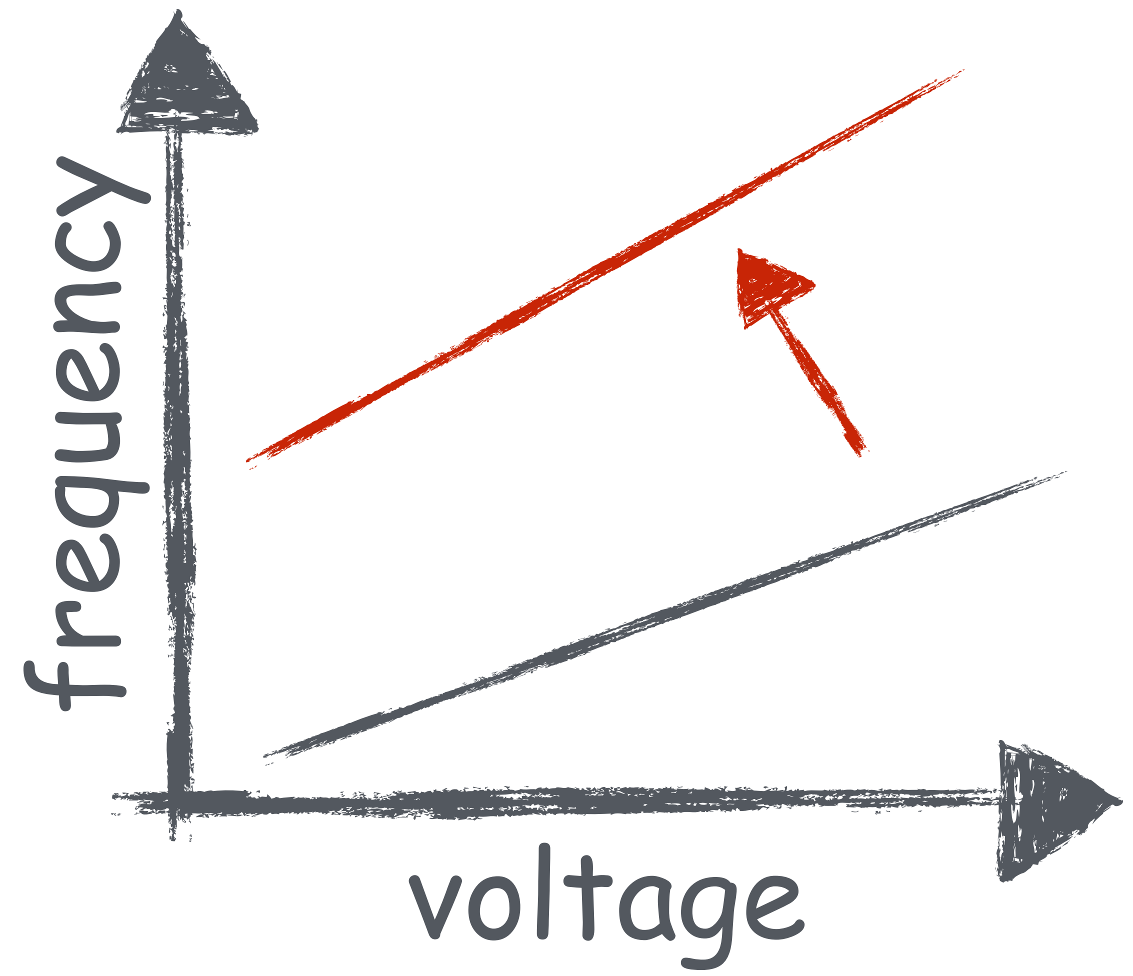
Today's systems cannot ~~exist without~~

Energy Management

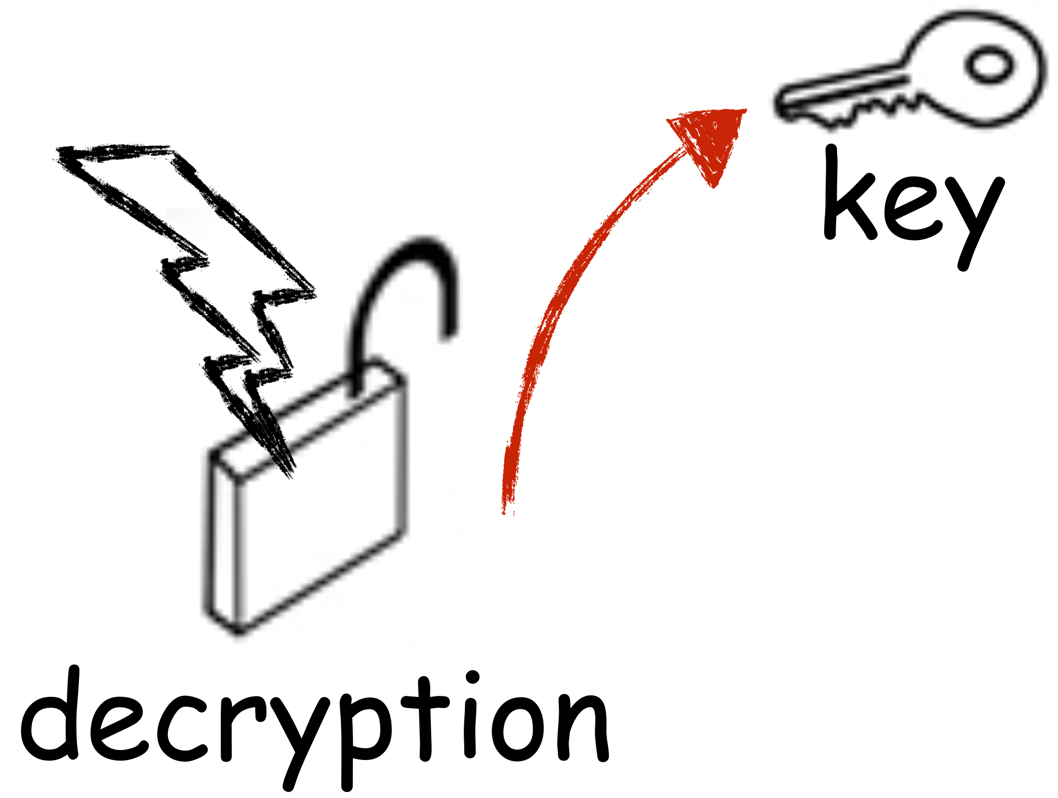
Exploiting software interfaces to **Energy Management**

Software-based attacker

Stretch operational limits



Induce faults



Exploiting software interfaces to **Energy Management**

Software-based attacker

Stretch

Traditional fault attacks

~~Need physical proximity~~

~~Need separate equipment~~

~~Soldering, crocodile clips, wire, etc~~

Induce faults

CLKSCREW: Exposing the perils of security-oblivious **Energy Management**

New attack vector that exploits energy management

Practical attack on trusted computing on ARM devices

Impacts hundreds of millions of deployed devices

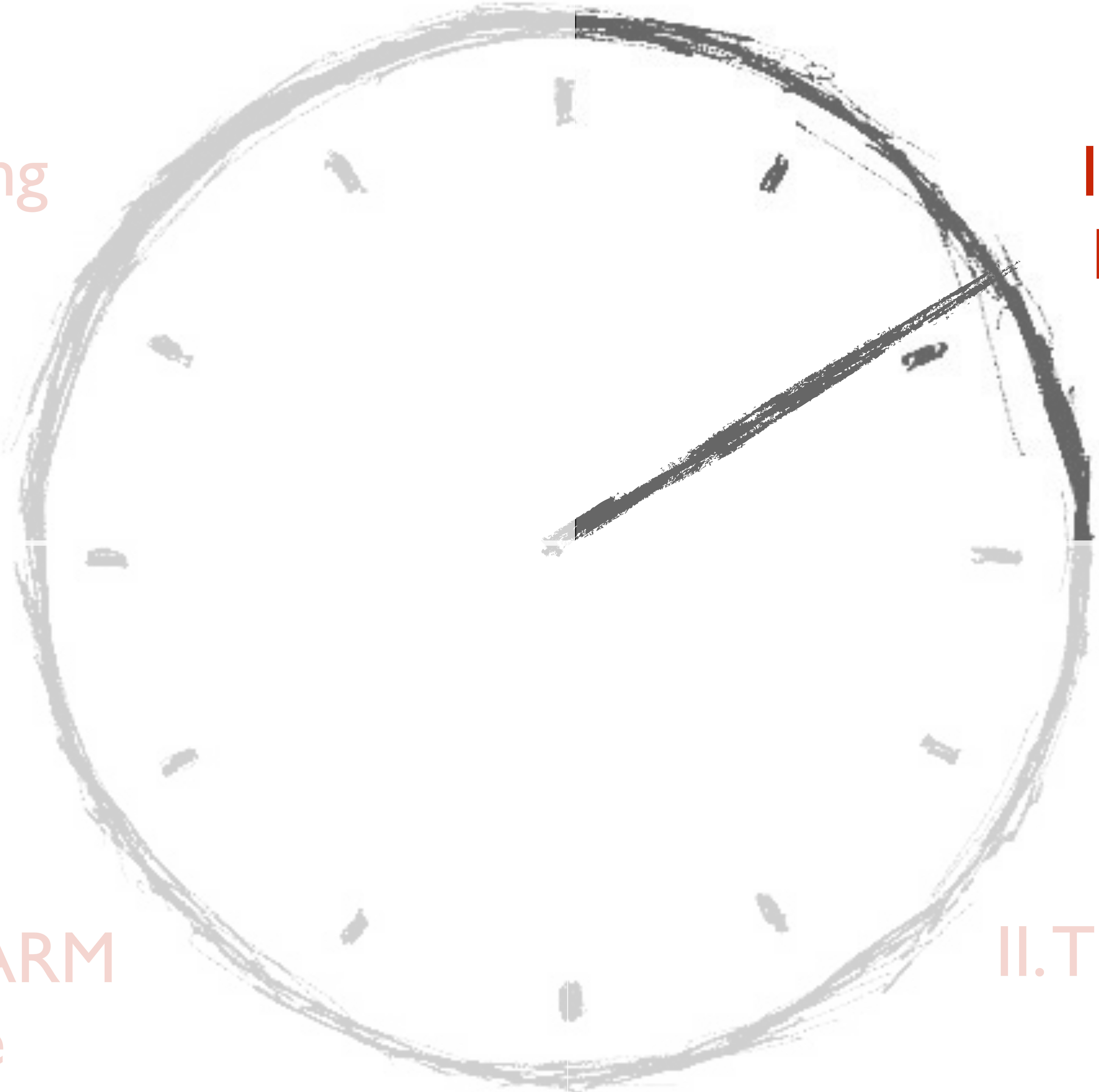
Lessons for future energy management designs to be security-conscious

I. DVFS and Regulators

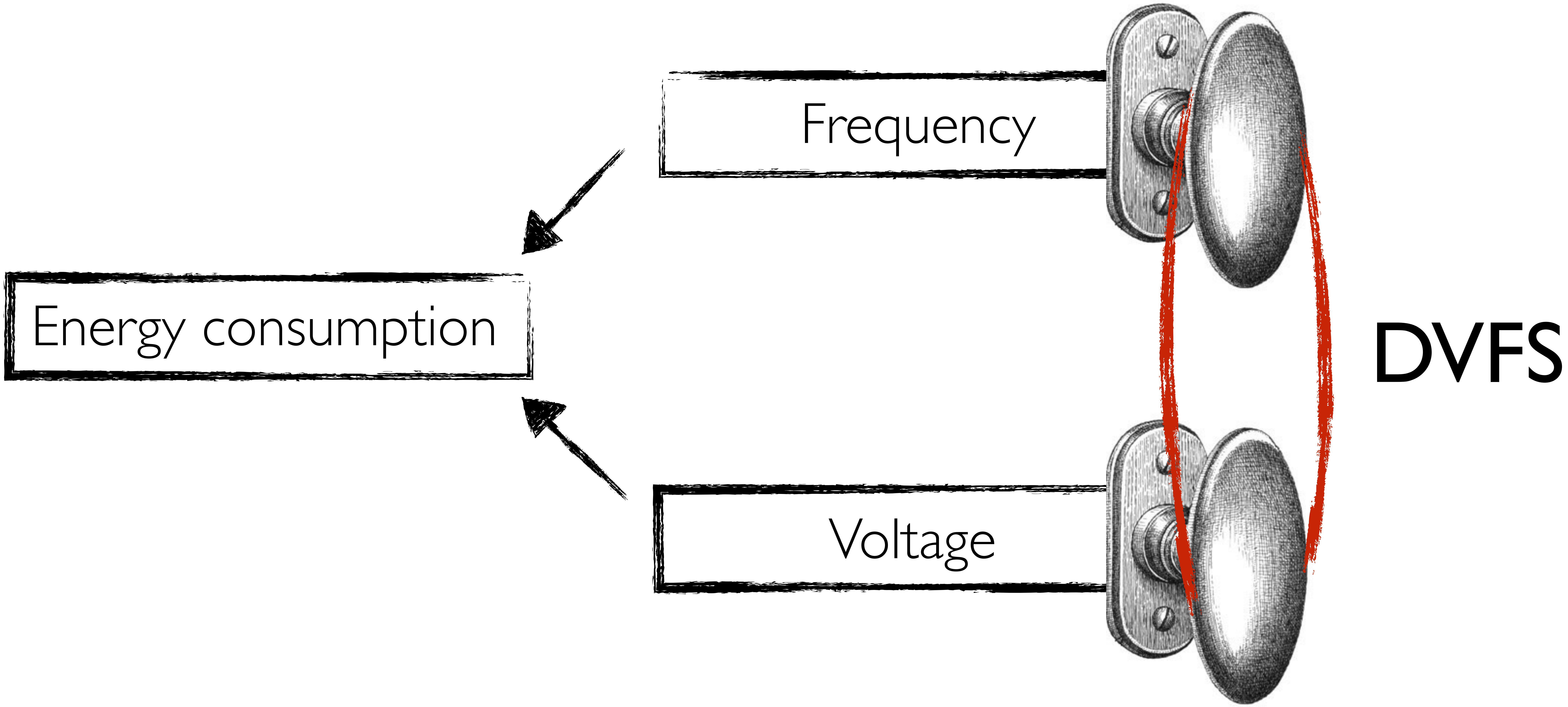
II. The CLKscrew Attack

III. Attacking ARM Trustzone

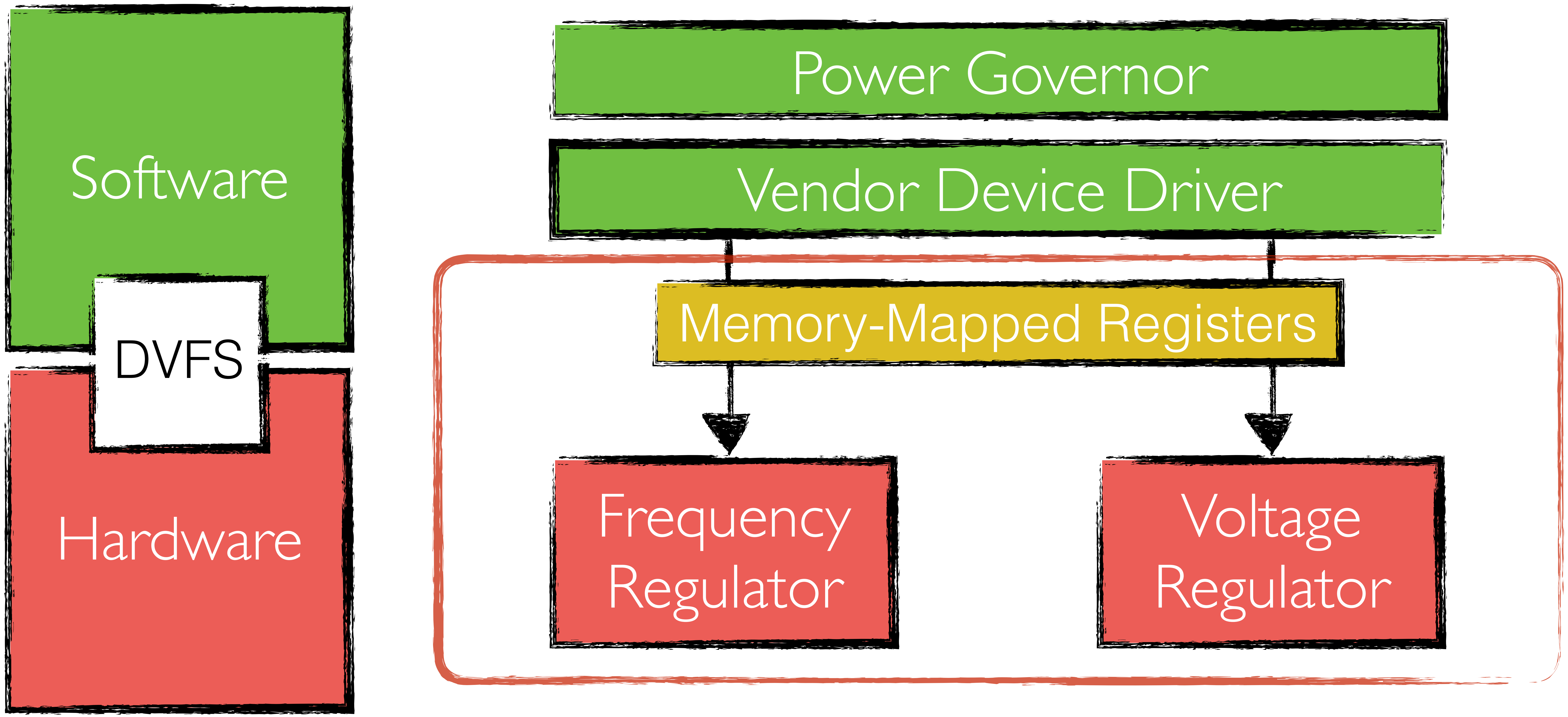
IV. Concluding Remarks



Dynamic Voltage and Frequency Scaling (DVFS)

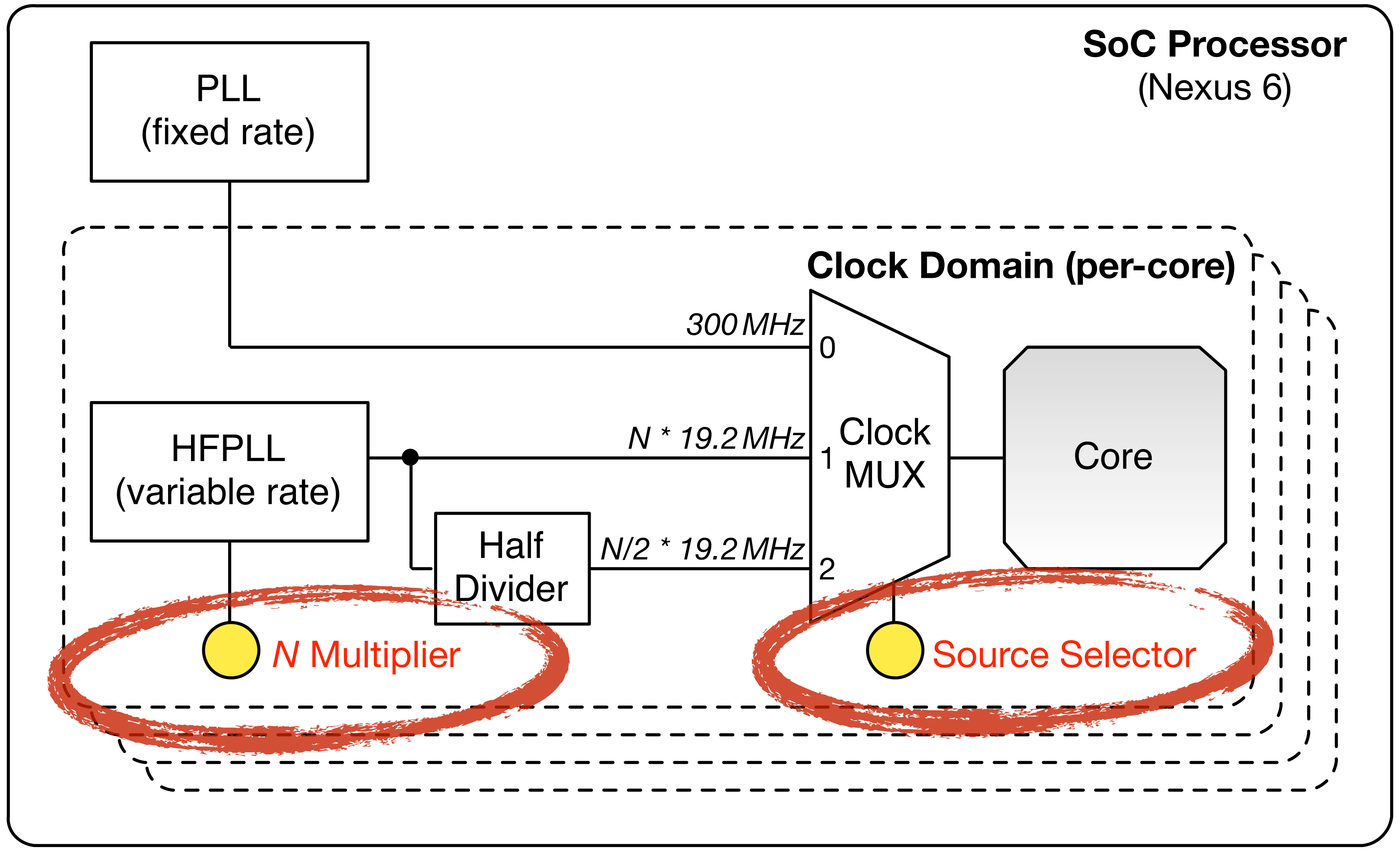


Hardware & Software Support for DVFS

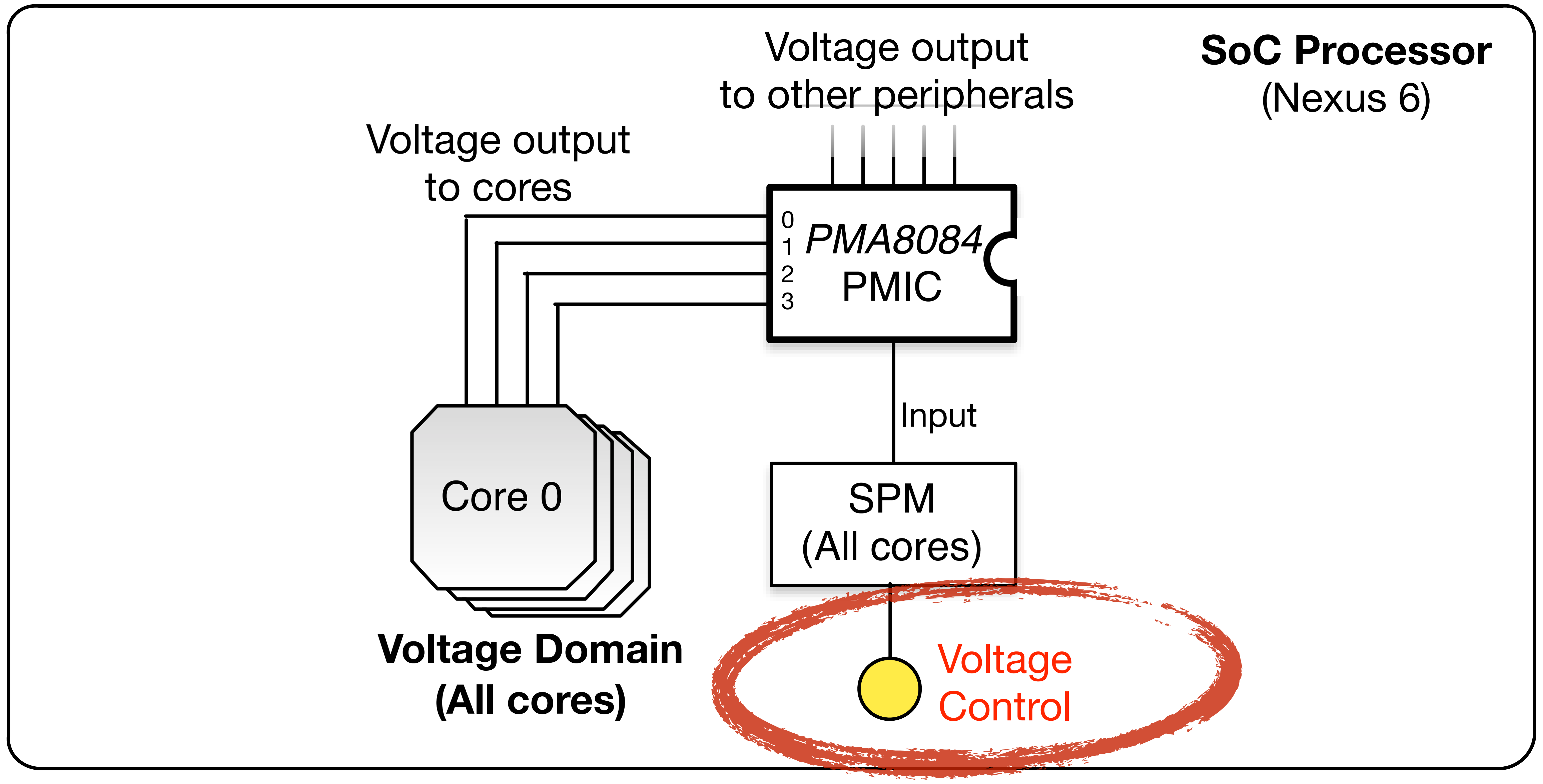


Hardware Regulators and Software Interfaces

Frequency regulators



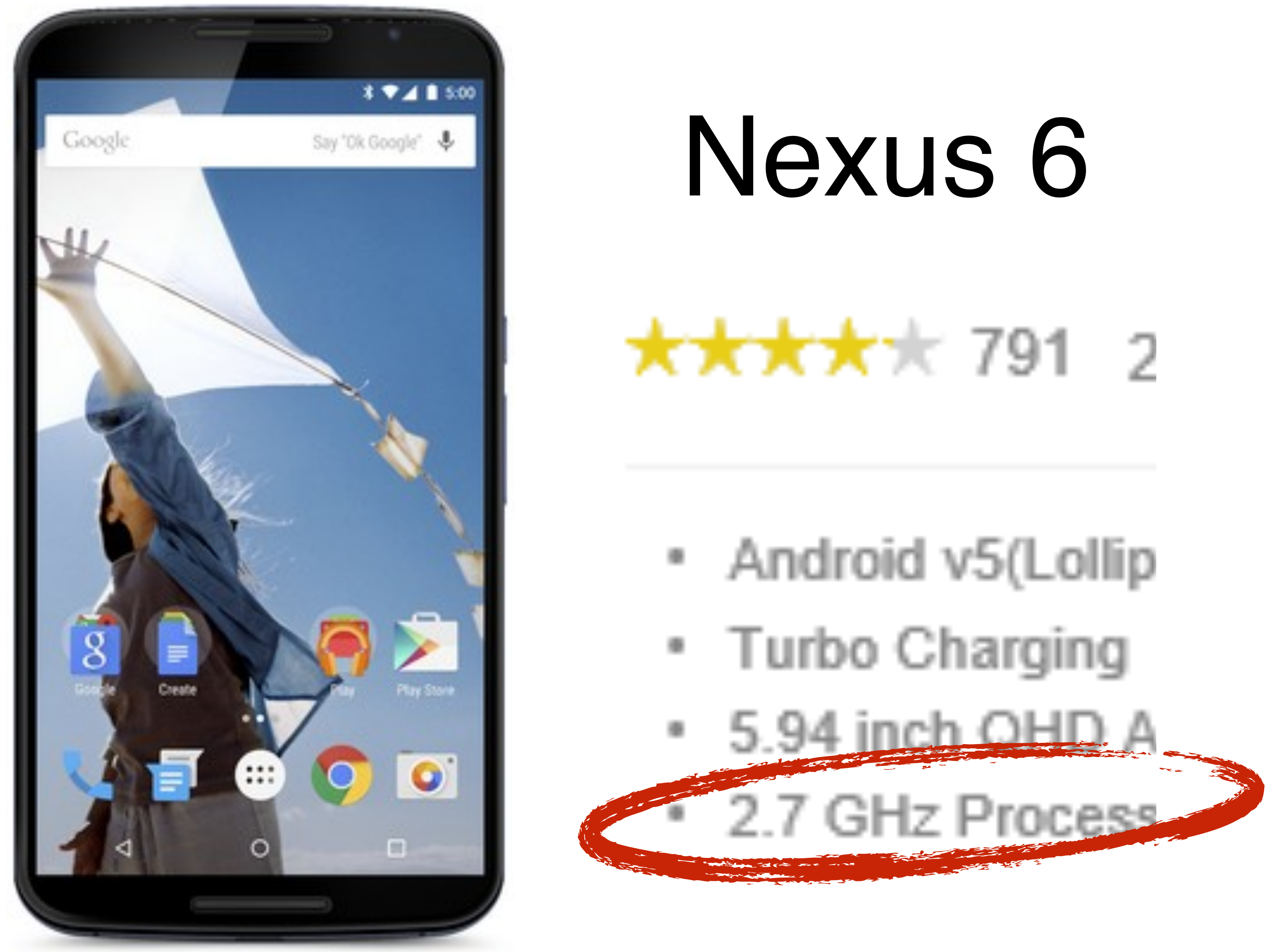
Voltage regulators

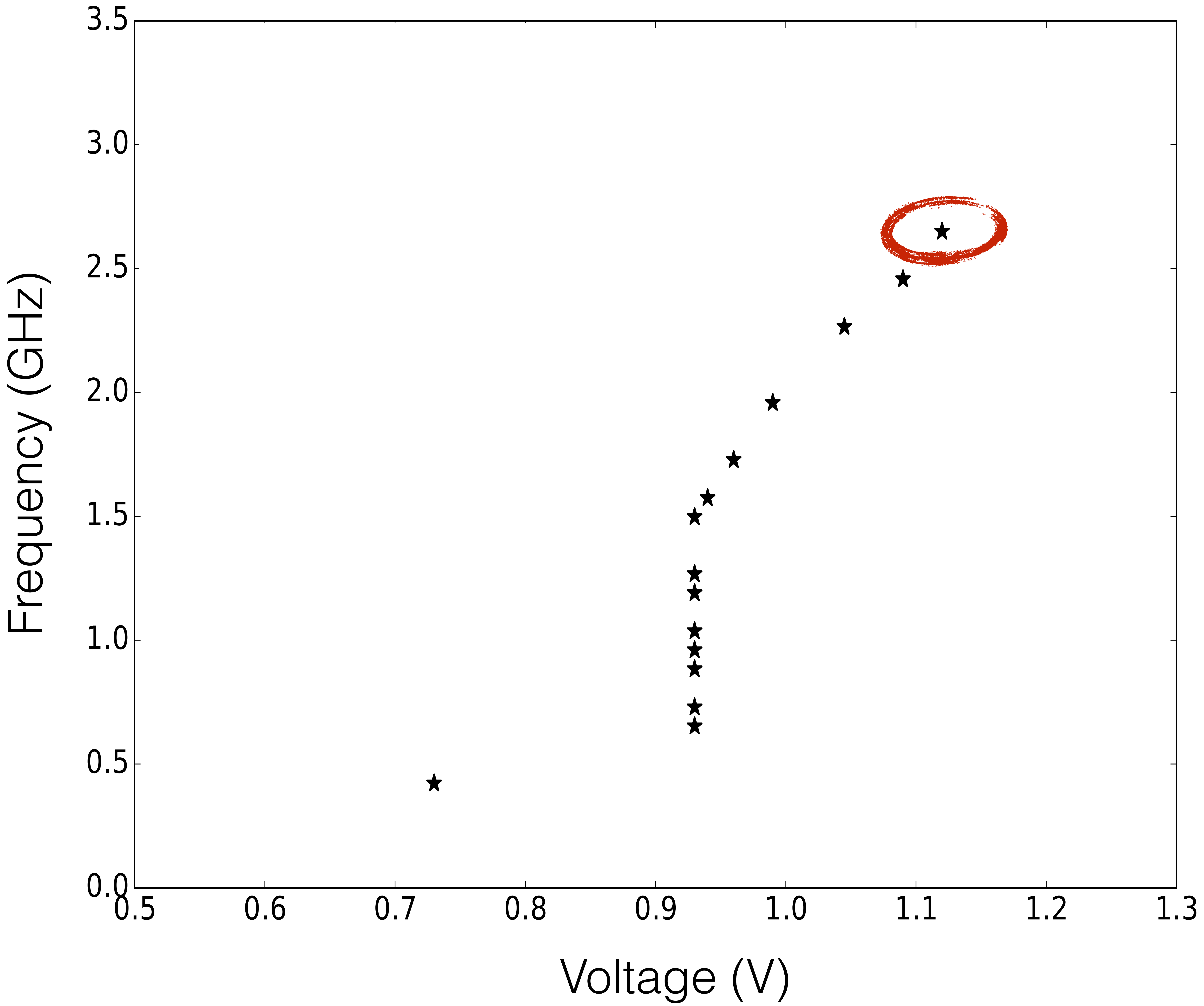


Operating frequency and voltage can be configured
via memory-mapped registers from software

Do hardware regulators impose limits
to frequency/voltage changes?

Frequency / Voltage Operating Point Pairs (OPPs)





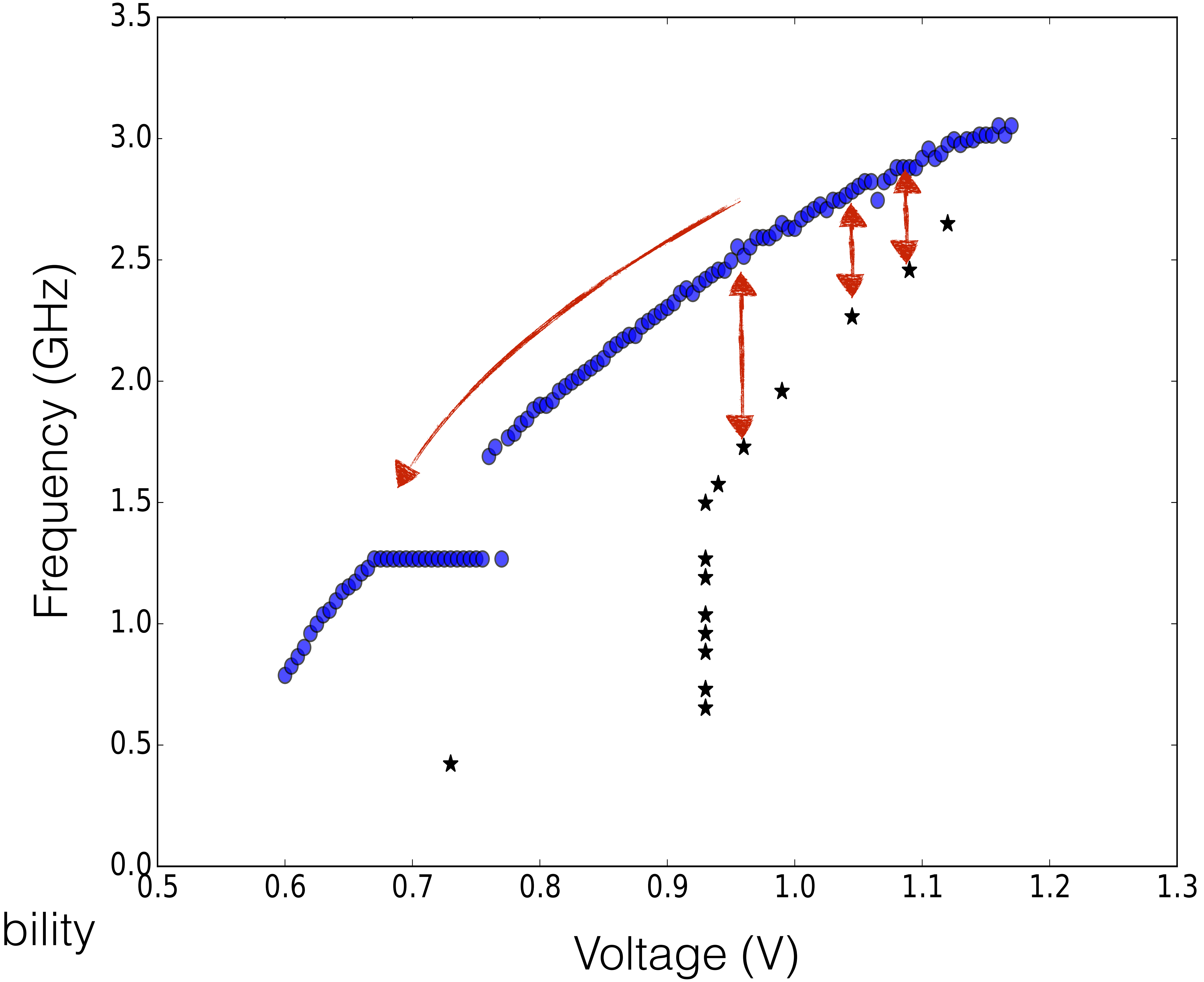
Legend:

★★★ Vendor-recommended

Frequency / Voltage Operating Point Pairs (OPPs)

No safeguard hardware limits

Lower voltage ⟶
Lower minimum required frequency to induce instability

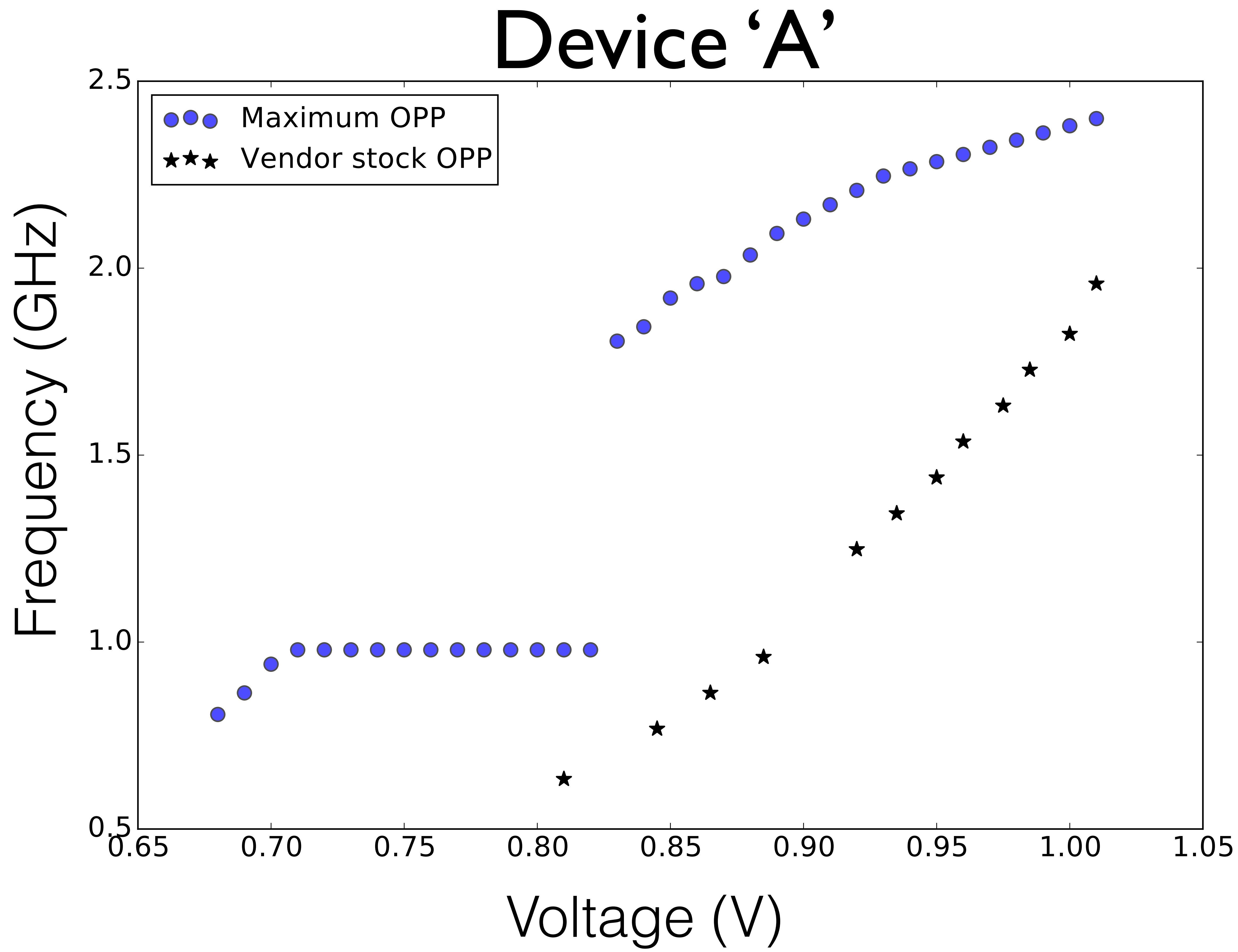


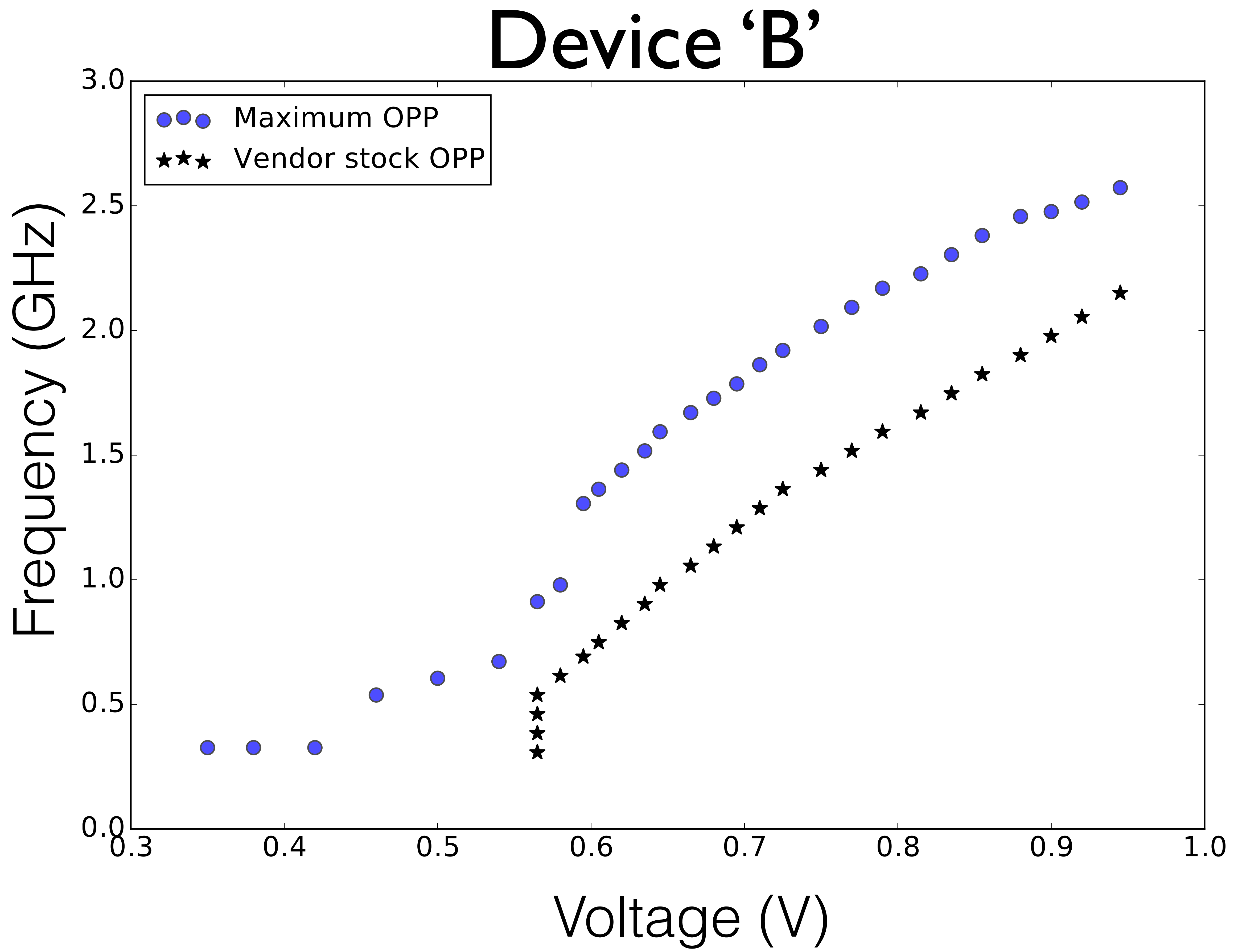
Legend:

★★★ Vendor-recommended

●●● Max OPP reached before instability

Frequency / Voltage Operating Point Pairs (OPPs)

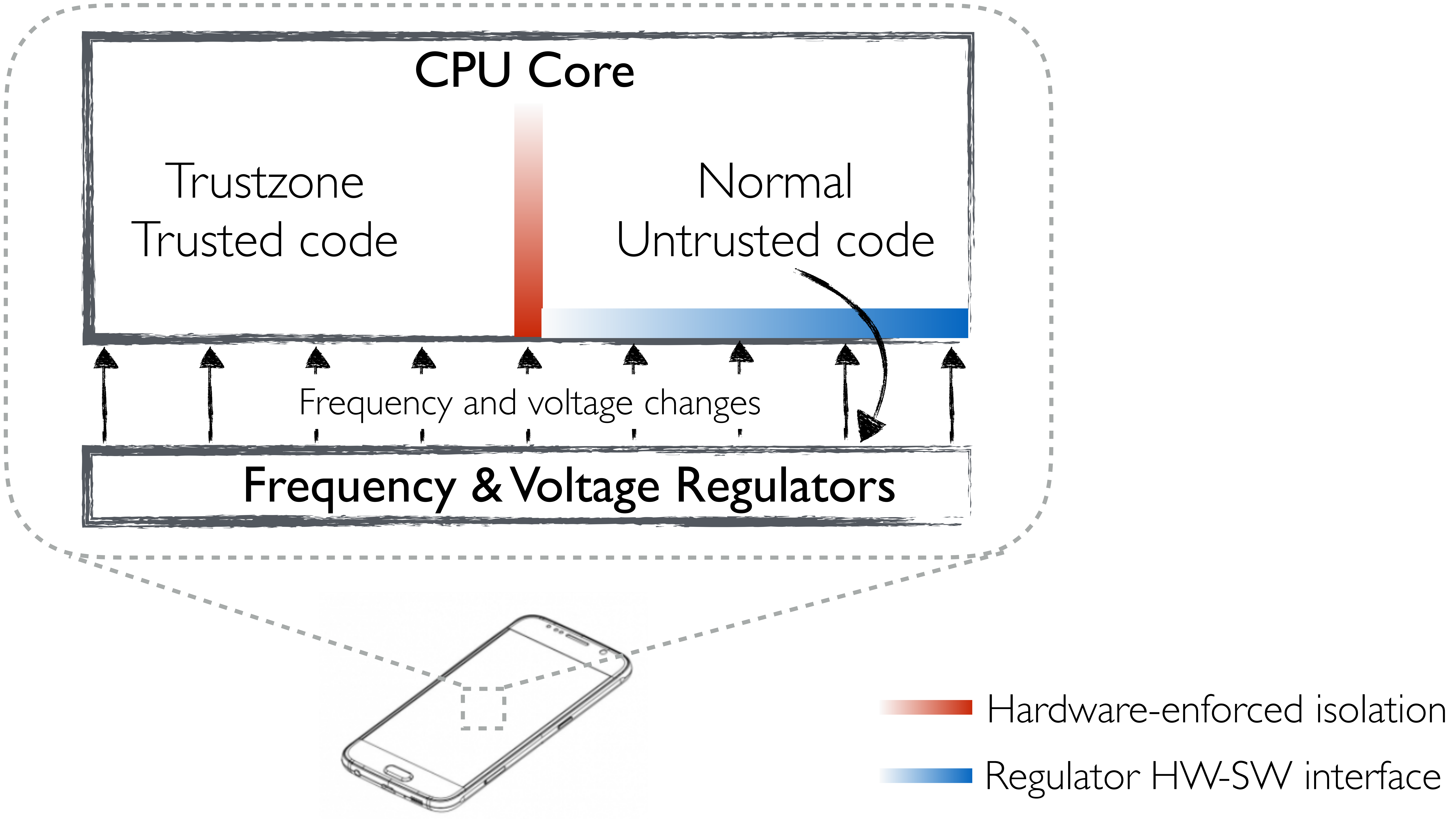




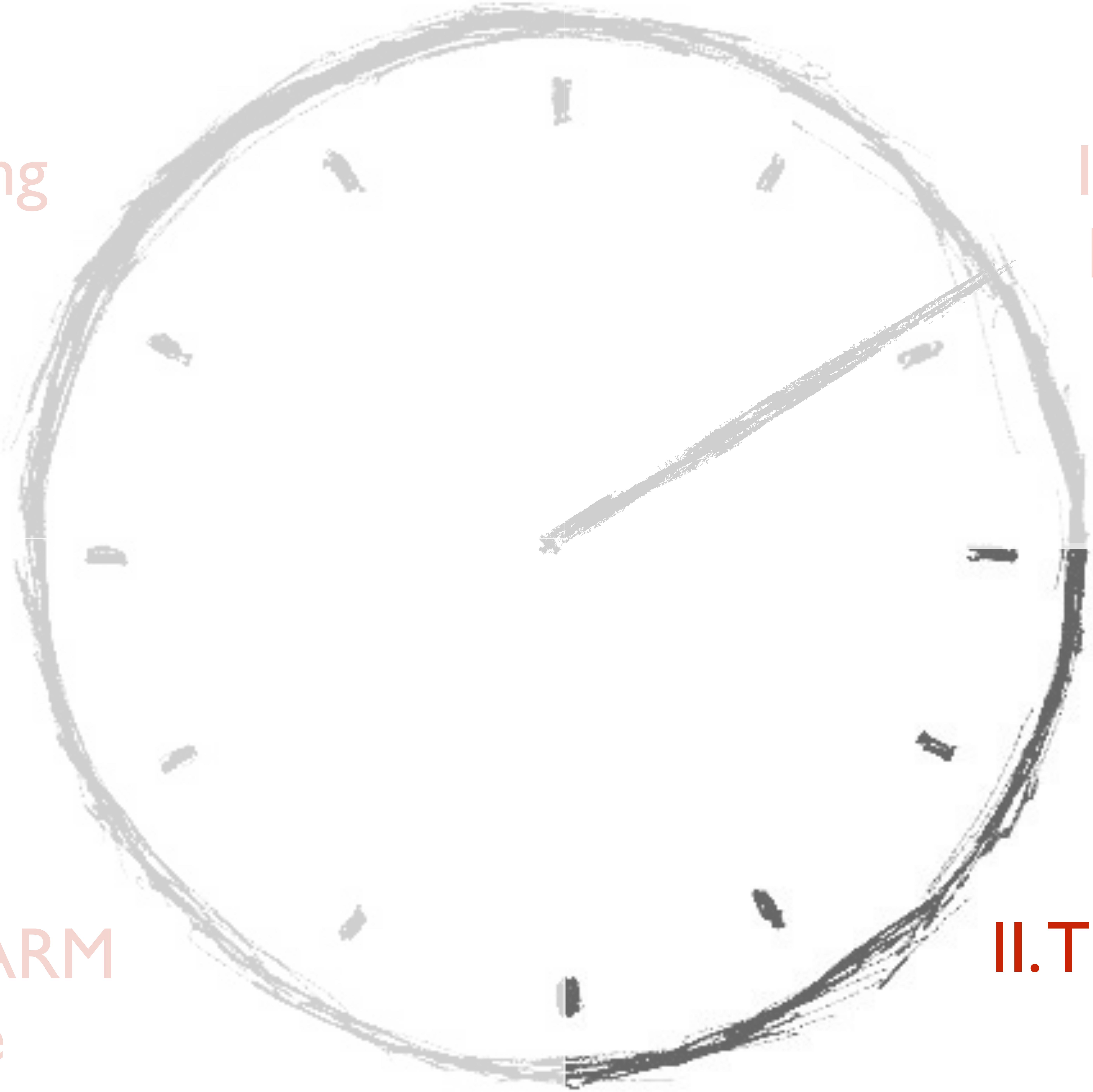
Does DVFS operate across security boundaries?

Trusted Execution Environments (TEE)

Is DVFS Trustzone-Aware? No!



IV. Concluding Remarks



I. DVFS and Regulators

III. Attacking ARM Trustzone

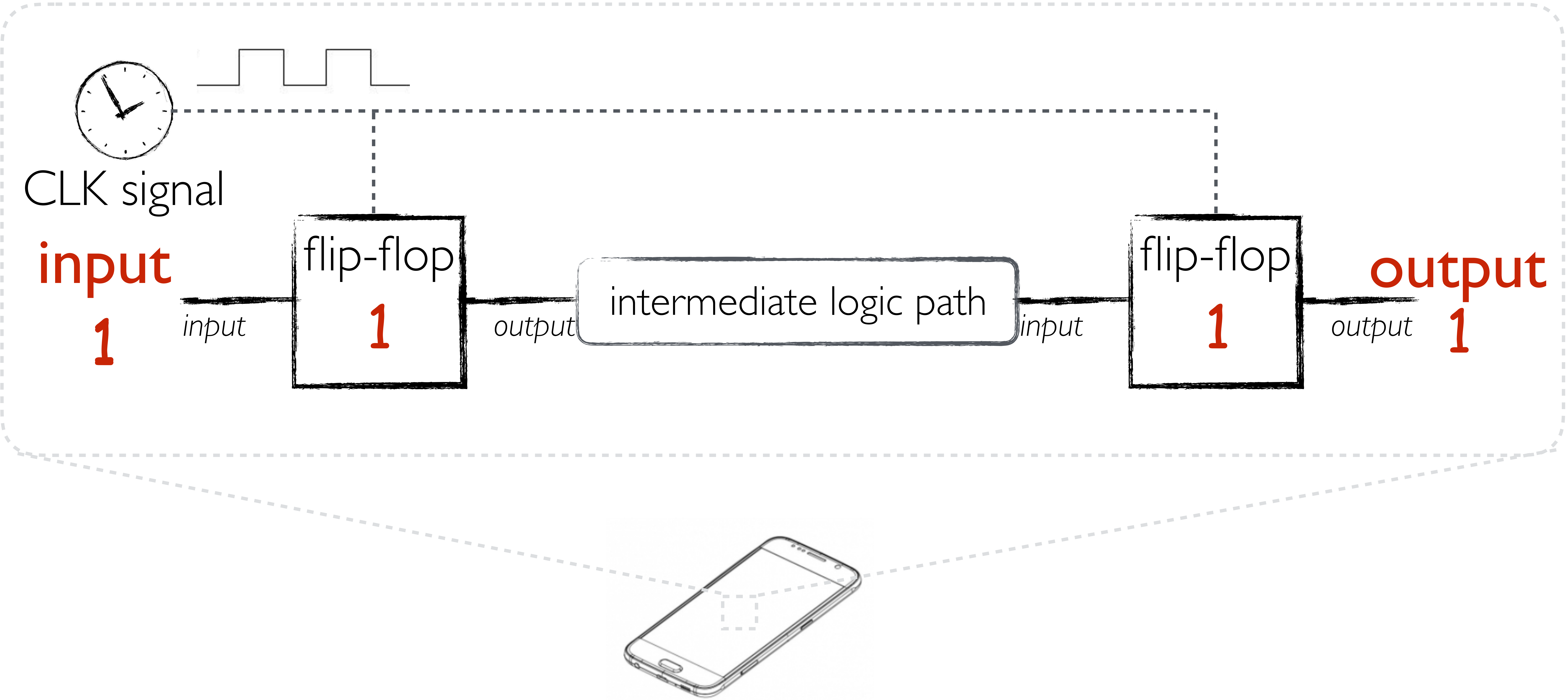
II. The CLKSCREW Attack

Can we attack Trustzone code execution using software-only control of the regulators?

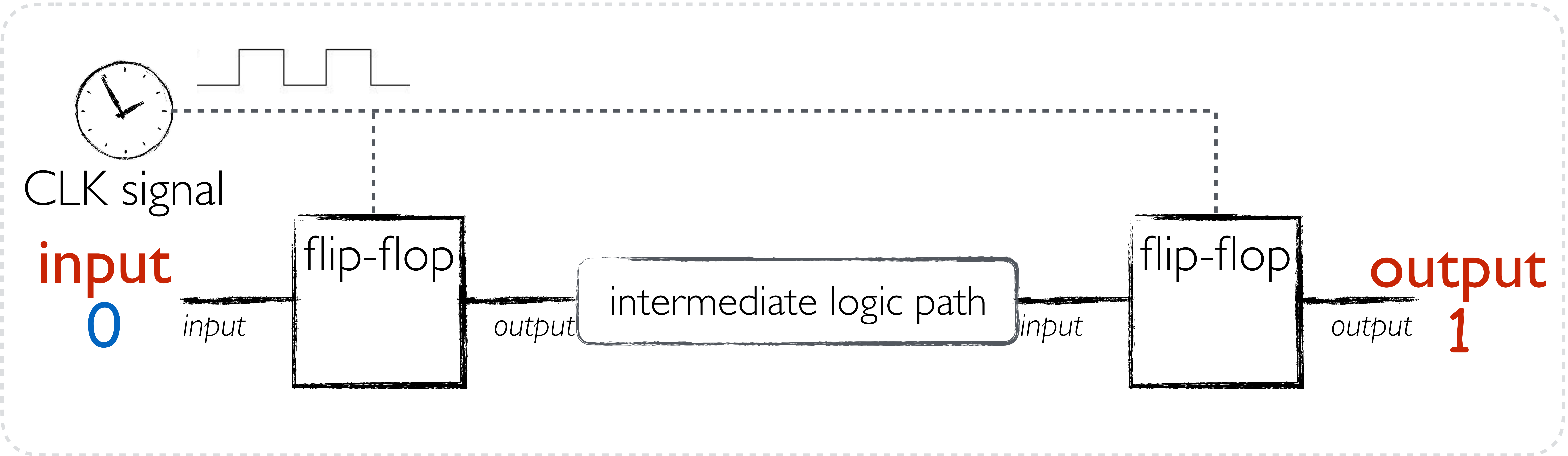
Induce timing faults

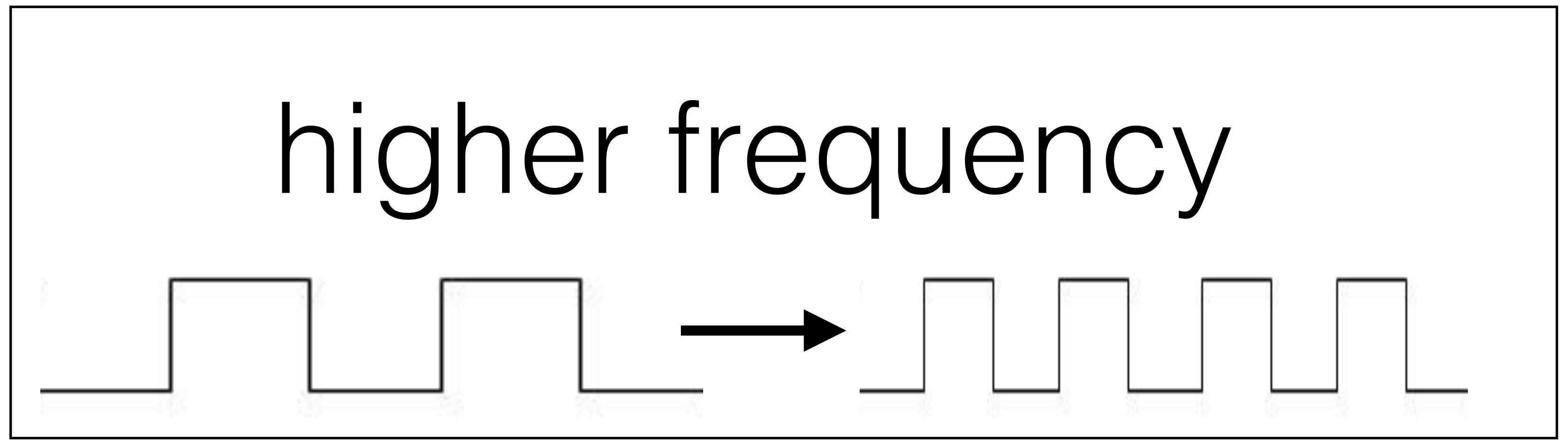
confidentiality
integrity
~~availability~~

How do faults occur (due to over-raising frequency)?



How do faults occur (due to over-raising frequency)?





$\Longrightarrow$ less time for data to propagate $\Longrightarrow$ timing violation

'0' $\longrightarrow$ '1'

How do faults occur (due to over-raising frequency)?

Expected: … `a777511b` …

Faulty output: … `a7775151` …

CLKSCREW Challenges & Solutions

#1: Regulator operating limits

#2: Self-containment within same device

#3: Noisy complex OS environment

#4: Precise timing

#5: Fine-grained timing resolution

CLKSCREW Challenges & Solutions

#1: Regulator operating limits

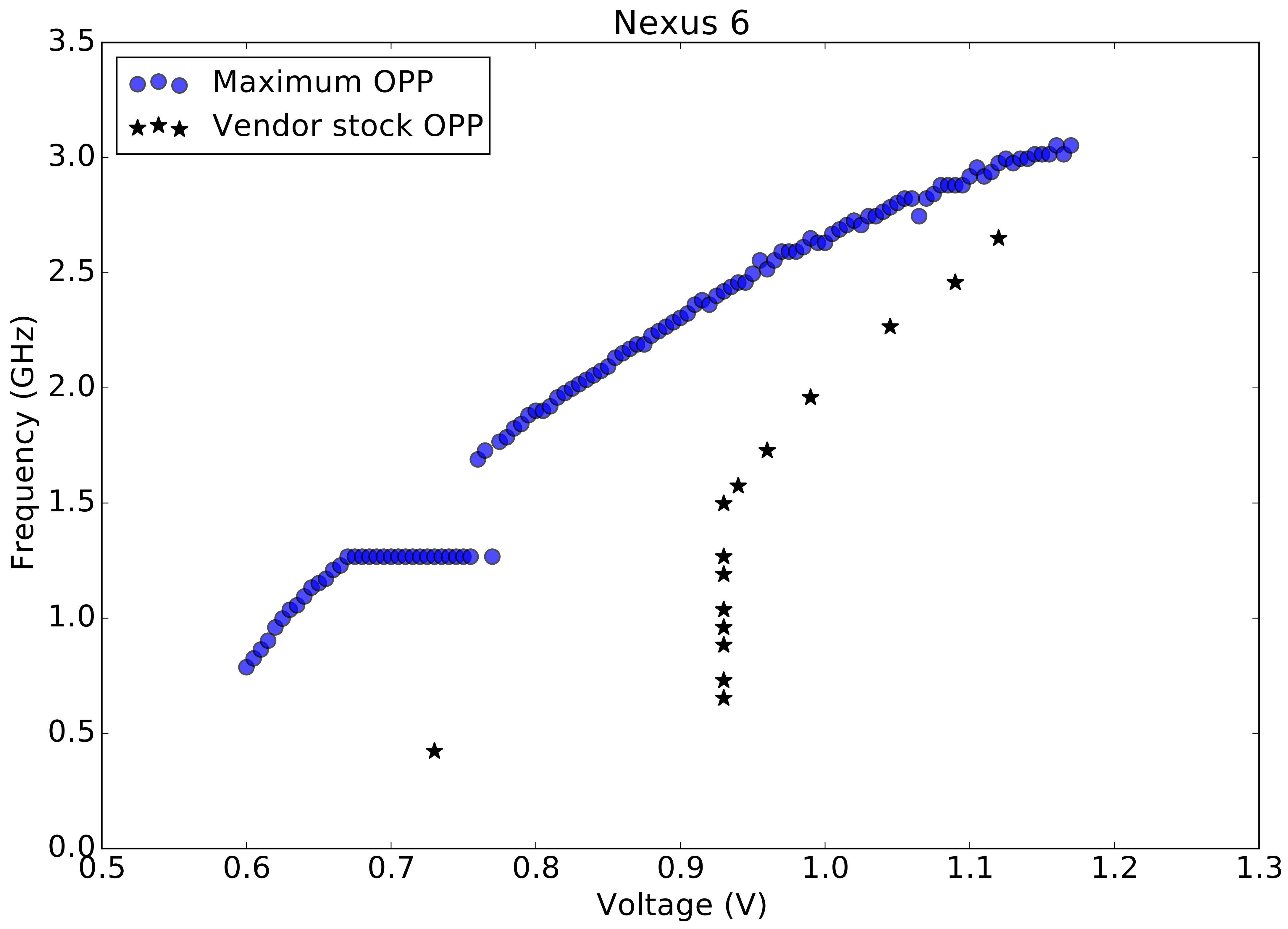
#2: Self-containment within same device

#3: Noisy complex OS environment

#4: Precise timing

#5: Fine-grained timing resolution

Addressed earlier in DVFS regulators



CLKSCREW Challenges & Solutions

#1: Regulator operating limits

#2: Self-containment within same device

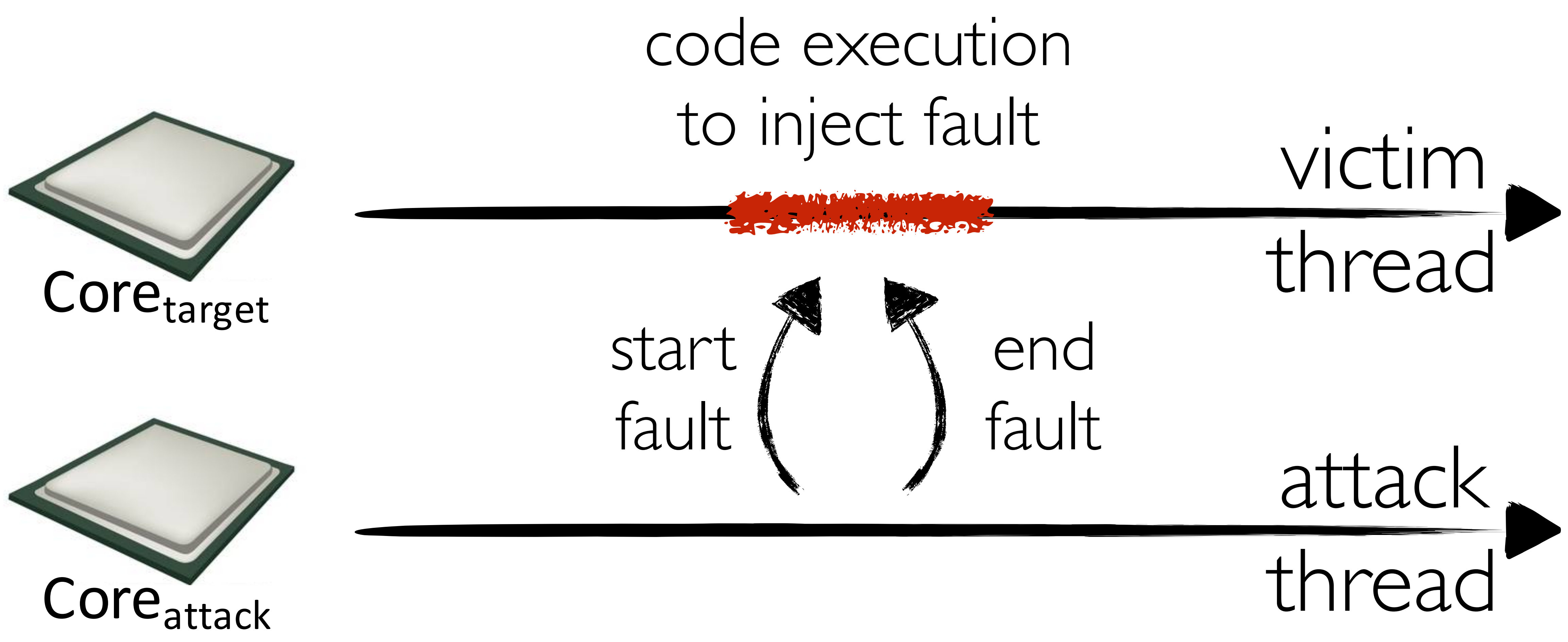
Cores have different frequency regulators

Core pinning

#3: Noisy complex OS environment

#4: Precise timing

#5: Fine-grained timing resolution



CLKSCREW Challenges & Solutions

#1: Regulator operating limits

#2: Self-containment within same device

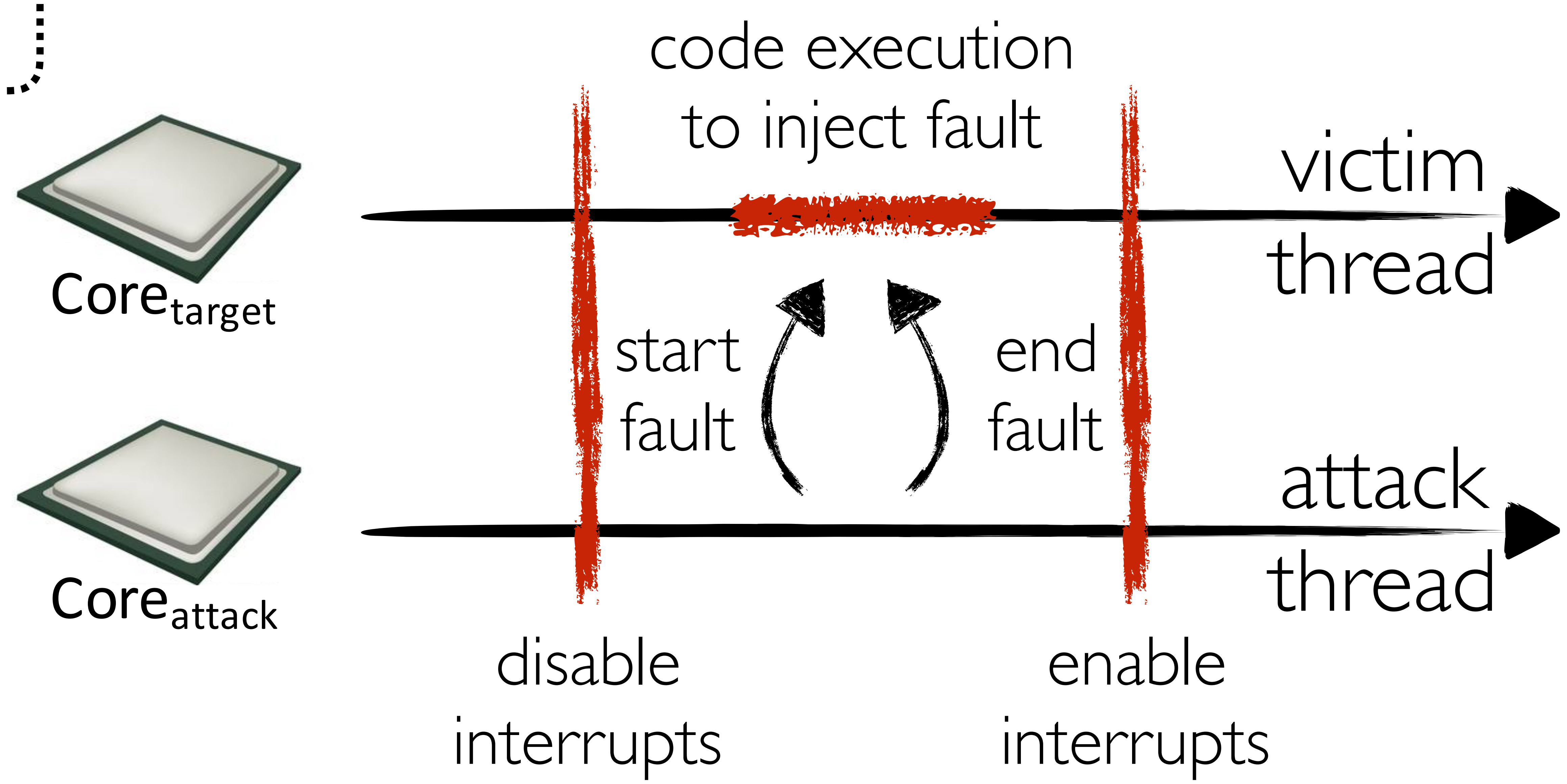
#3: Noisy complex OS environment

#4: Precise timing

#5: Fine-grained timing resolution

Core pinning

Disable interrupts during attack

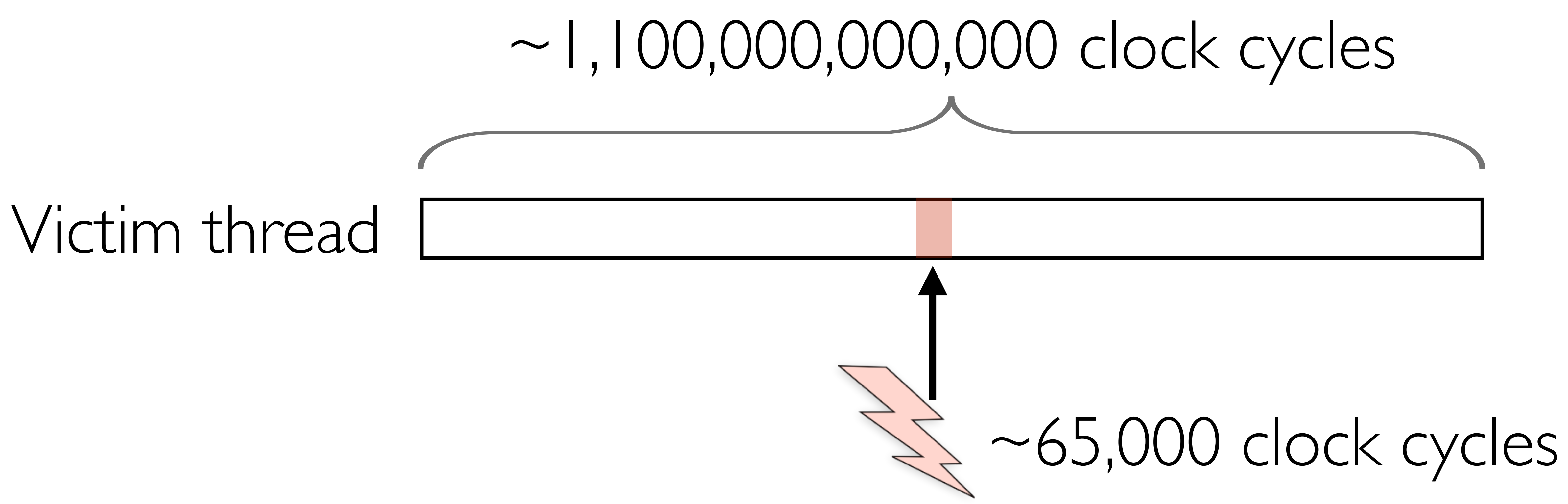


CLKSCREW Challenges & Solutions

#1: Regulator operating limits

#2: Self-containment within same device

#3: Noisy complex OS environment




```
asm volatile("1:  subs %0, %0, #1 \n"
             "    bhi 1b \n"::"r" (loops));
```

#4: Precise timing

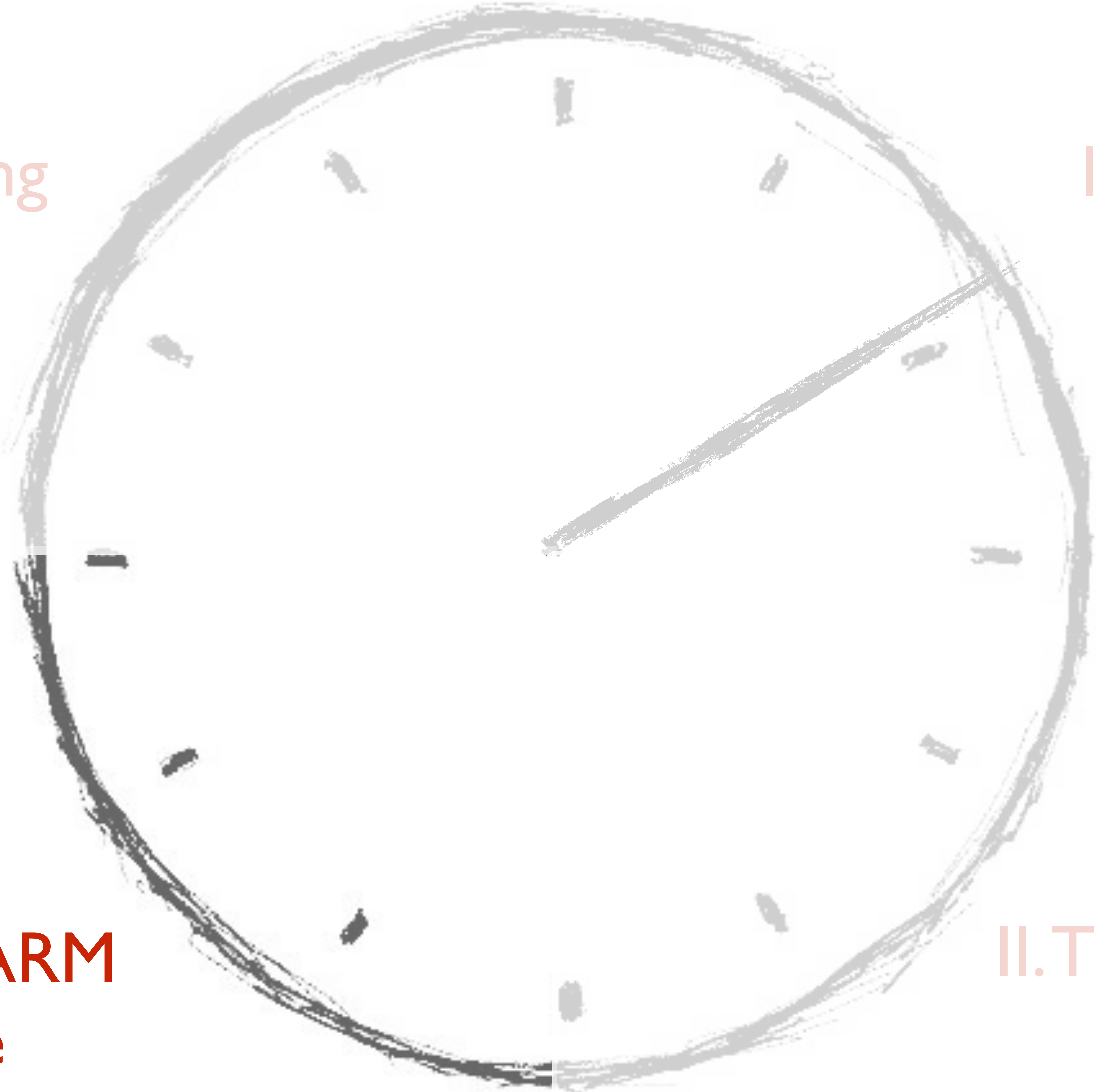
High-precision timing loops in attack architecture

#5: Fine-grained timing resolution

Cache-based execution timing profiling

IV. Concluding Remarks

I. DVFS and Regulators



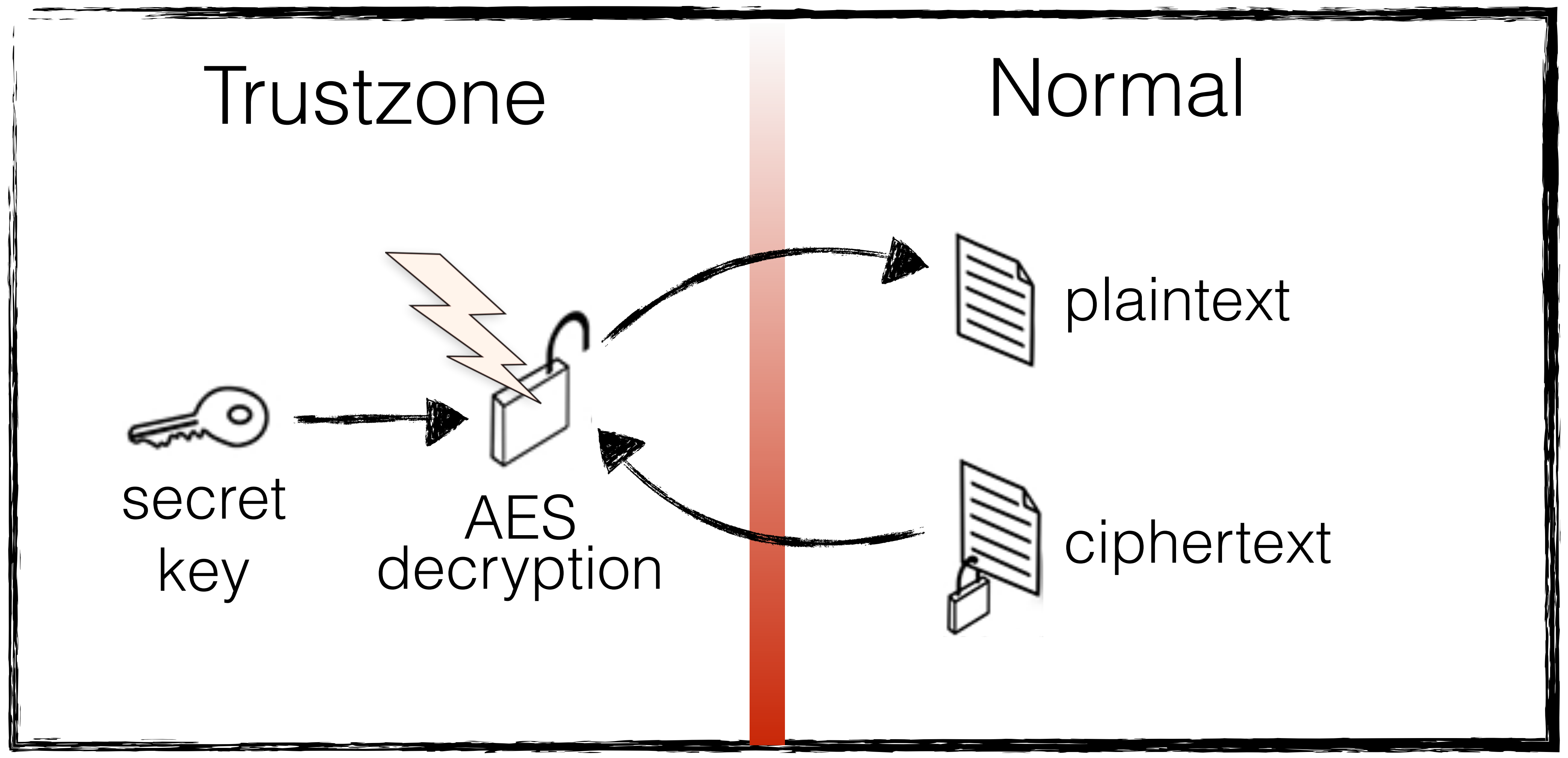
III. Attacking ARM Trustzone

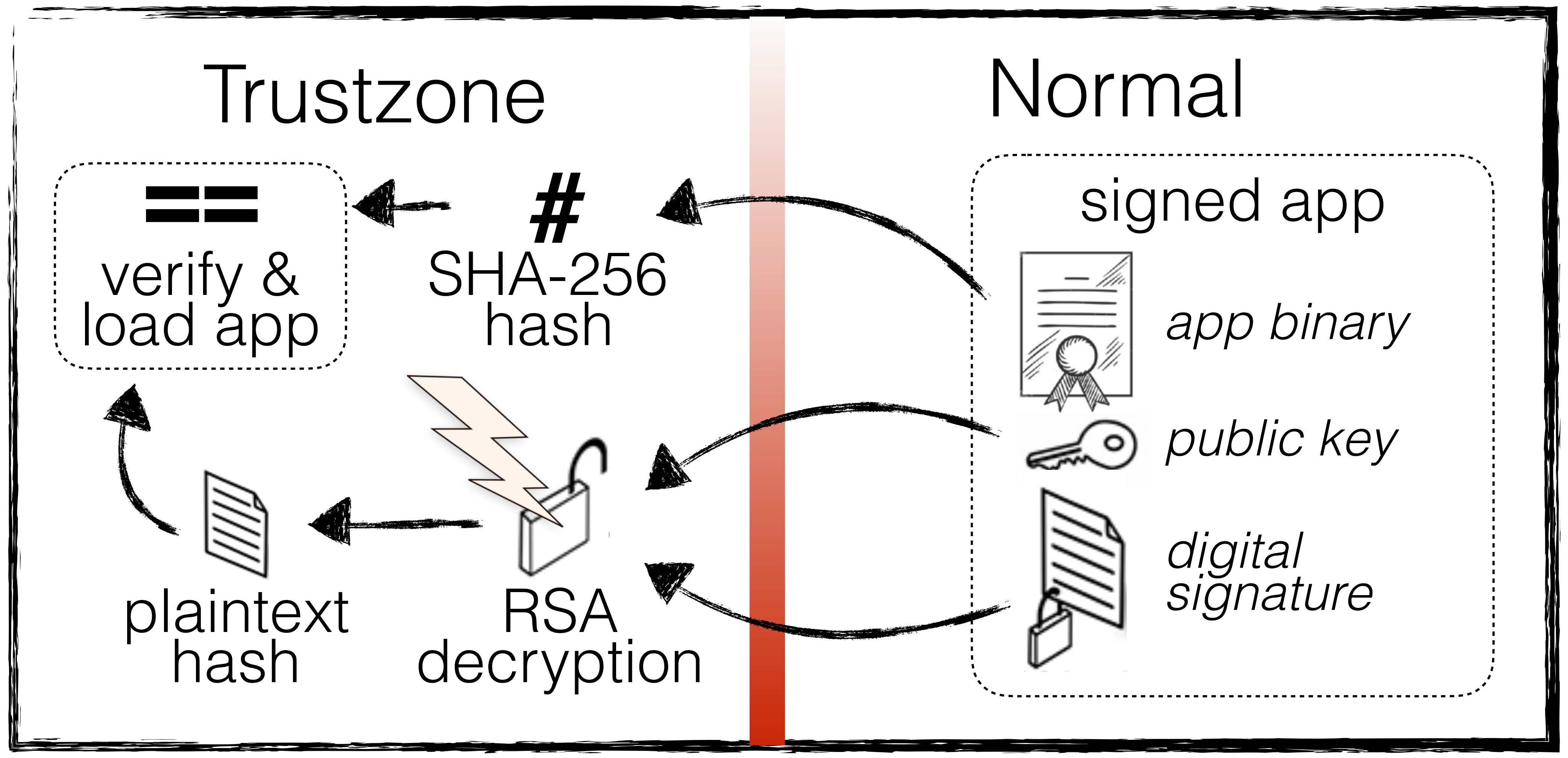
II. The CLKSCREW Attack

Subverting Trustzone Isolation with CLKSCREW

I. **Confidentiality Attack**
infer secret AES key stored within Trustzone

II. **Integrity Attack**
load self-signed app into Trustzone





(More details in the paper…)

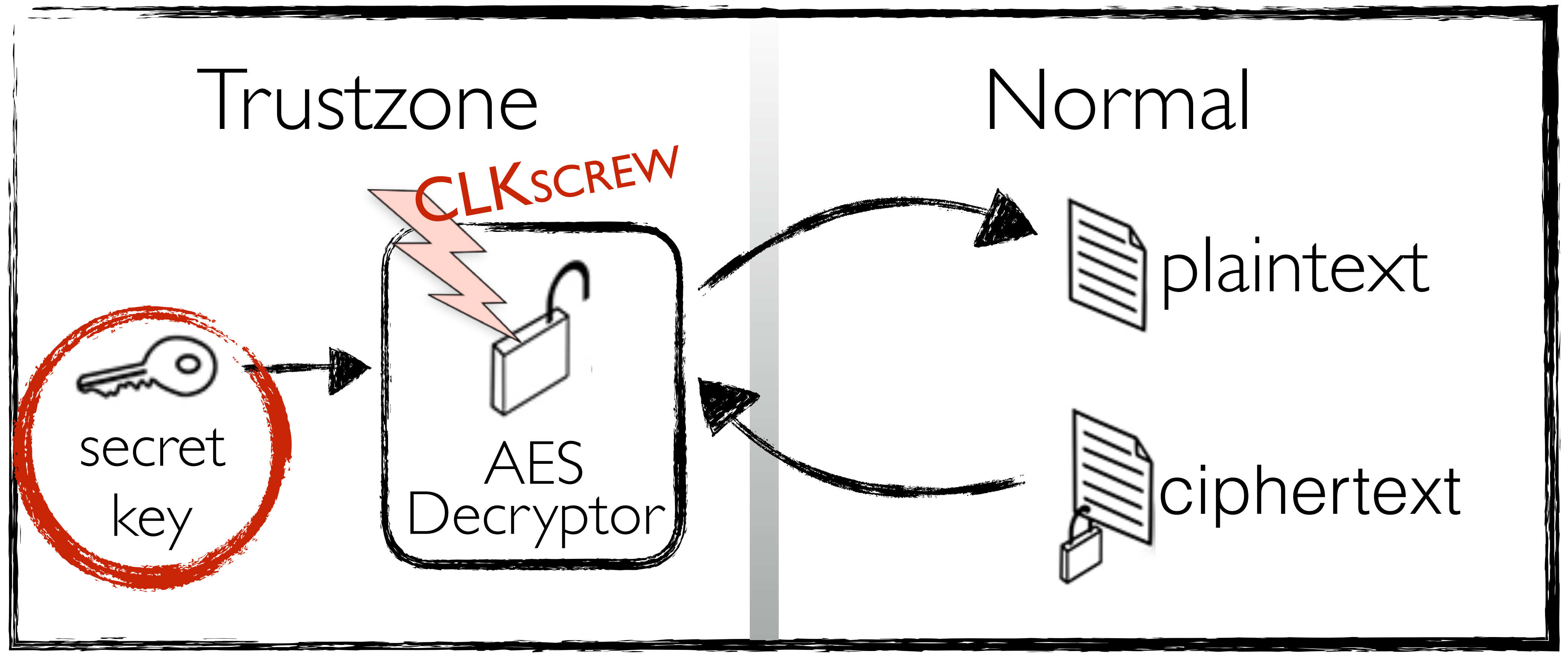
Key Inference Attack: Threat Model

Victim app: AES decryption app executing in Trustzone

Attacker's goal: Get secret AES key from outside Trustzone

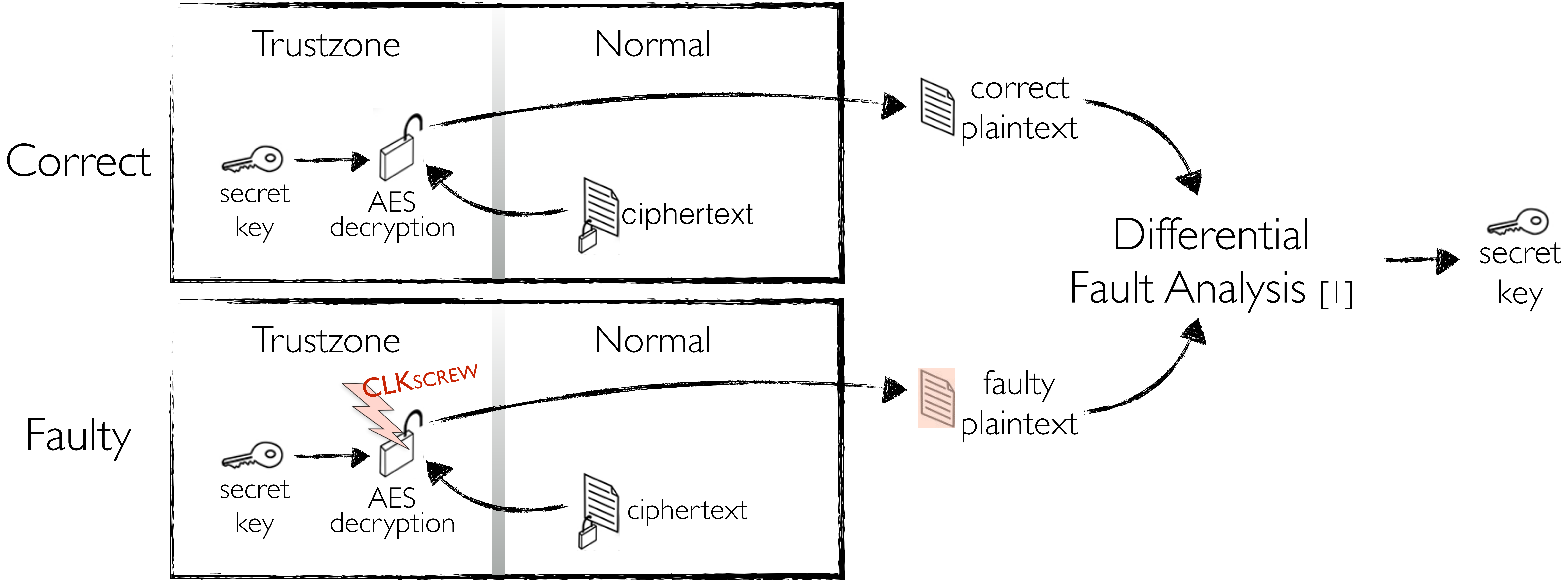
Attacker's capabilities: 1) Can repeatedly invoke the decryption app

2) Has software access to hardware regulators



Key Inference Attack: Summary

Idea: Induce a fault during the AES decryption
Infer key from a pair of correct and faulty plaintext



[1] *Tunstall et al. Differential Fault Analysis of the Advanced Encryption Standard using a Single Fault. In IFIP International Workshop on Information Security Theory and Practices (2011).*

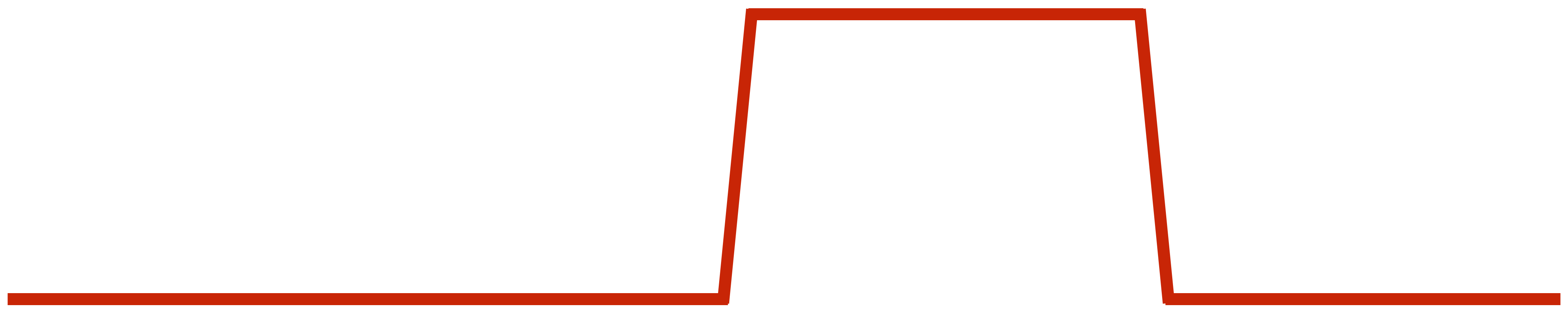
Key Inference Attack: CLKSCREW Parameters

Base voltage: 1.055V

Fault injection duration:
680 no-op loops (~39 µsec)

High frequency: 3.69GHz

Low frequency: 2.61GHz



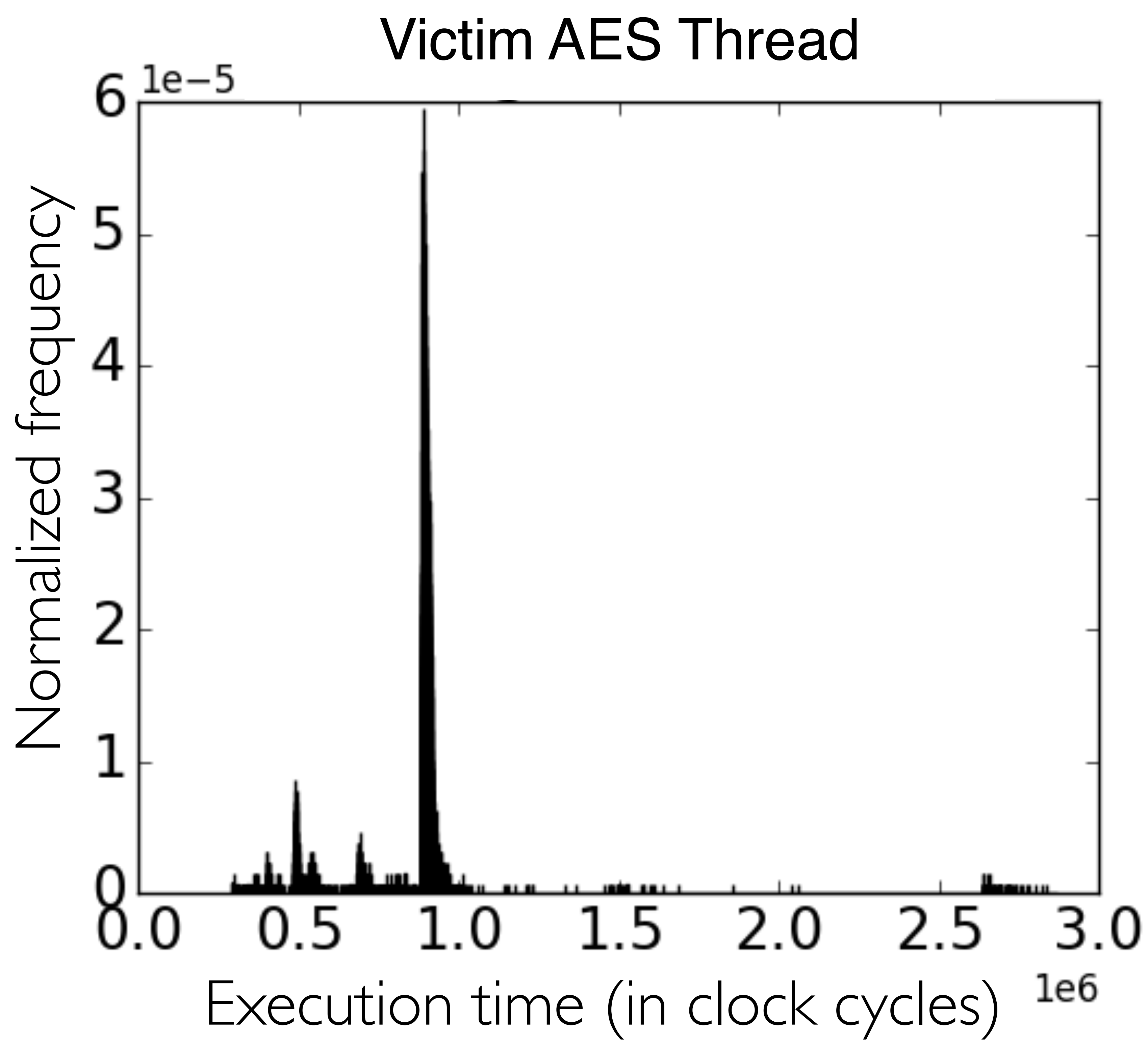
Differential Fault Analysis needs CLKSCREW to deliver a one-byte fault to the 7th AES round

Key Inference Attack: Timing Profiling

Execution timing of Trustzone code can be profiled with hardware cycle counters that are accessible outside of Trustzone

Key Inference Attack: Timing Profiling

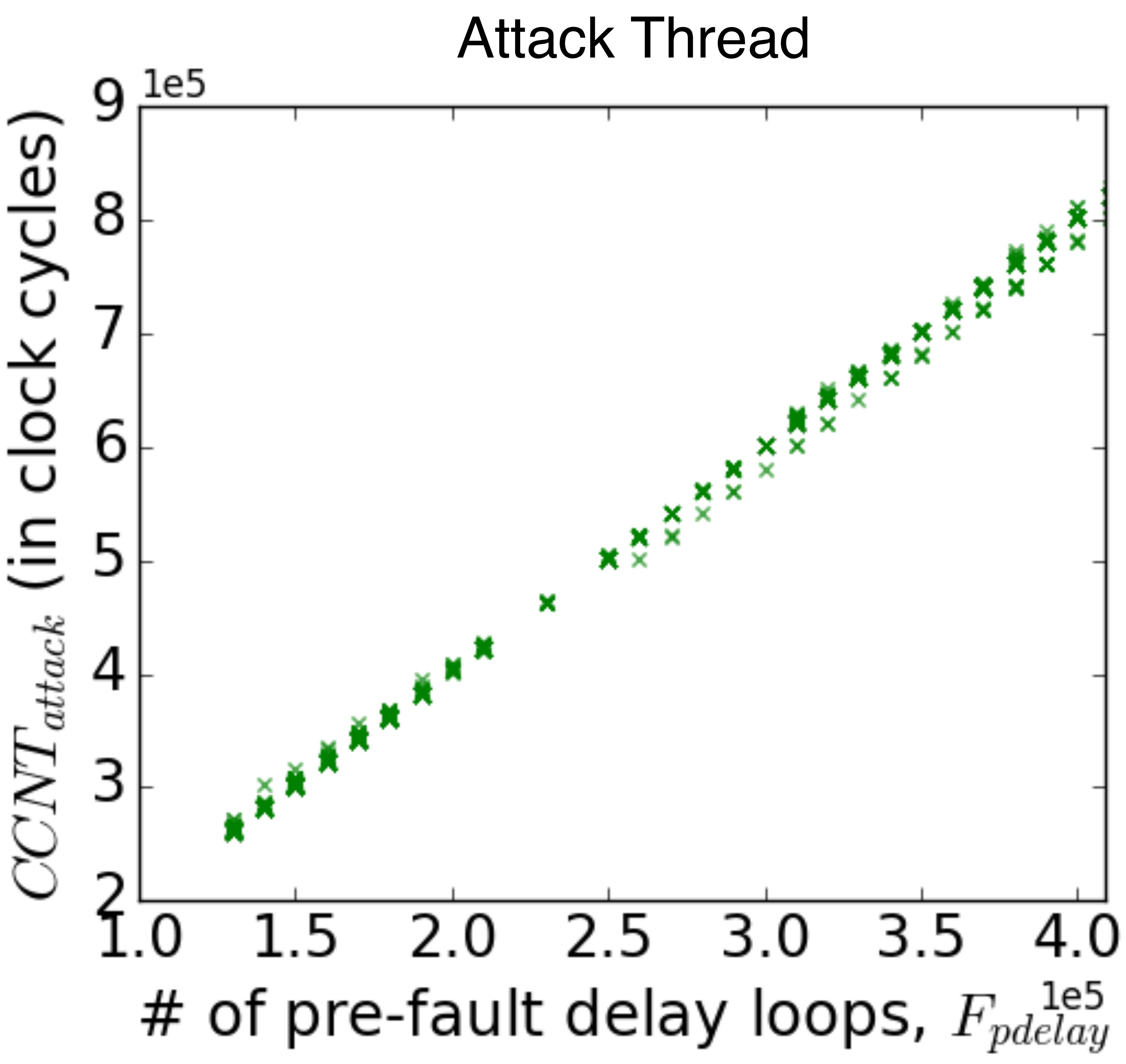
How varied is the execution timing of the victim decryption app?



Not too much variability in terms of execution time

Key Inference Attack: Timing Profiling

Can we effectively control the timing of the fault delivery with no-op loops?



Number of no-op loops is a good proxy to control timing of fault delivery

Key Inference Attack: Fault Model

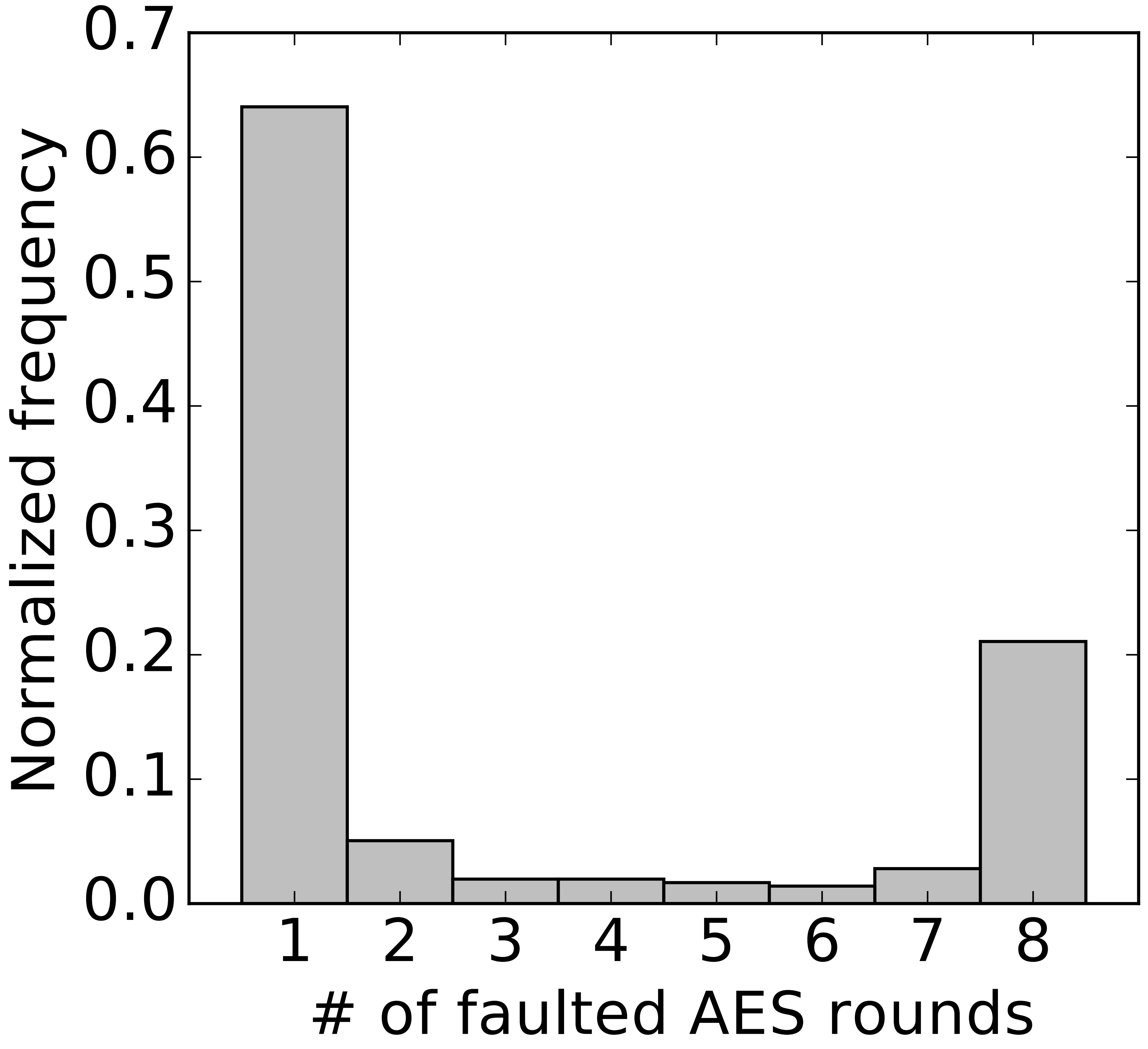
Our fault model requires our attack to inject fault

Exactly one AES round at the 7th round

Corruption of exactly one byte

Key Inference Attack: Fault Model

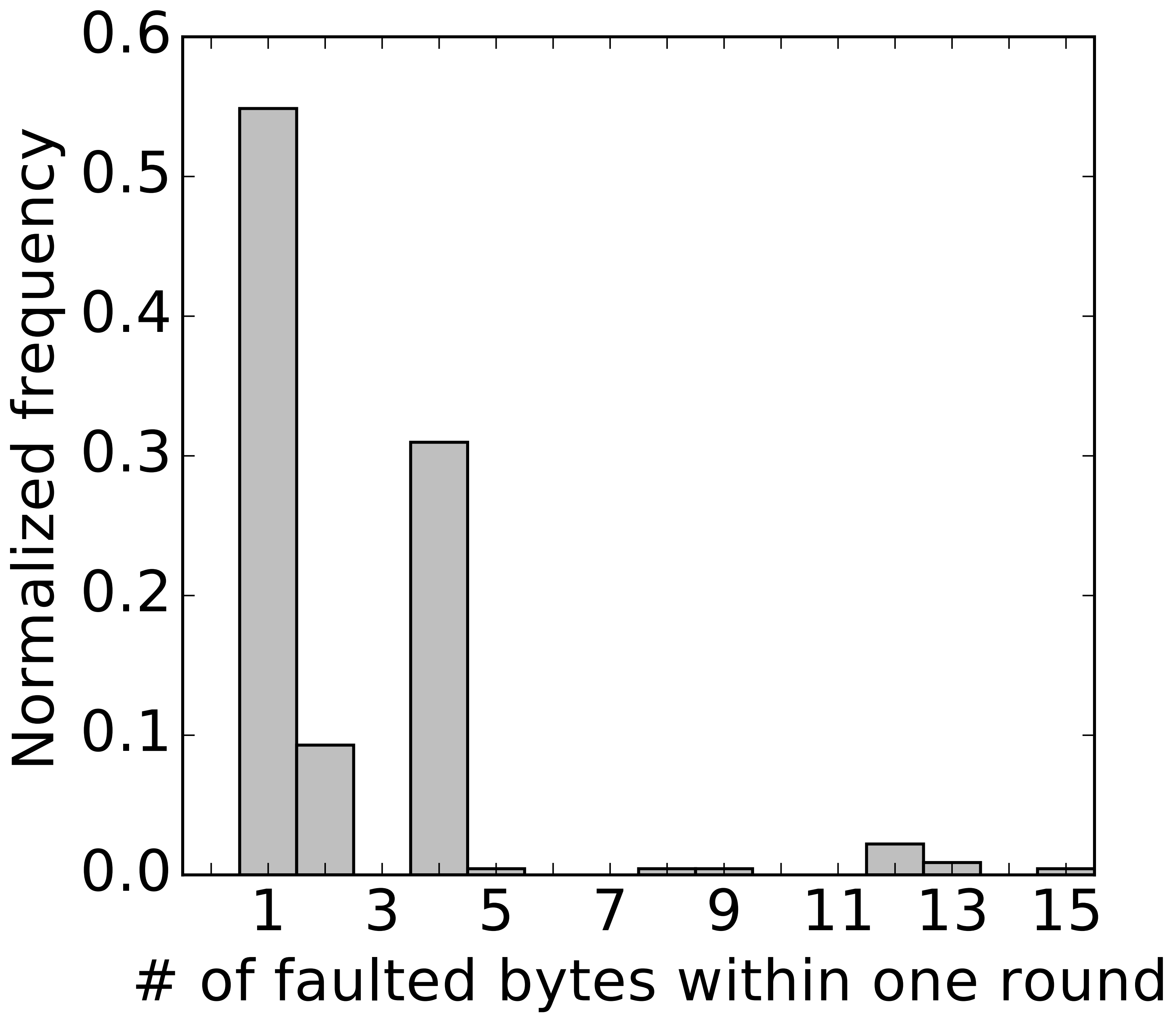
Precision: How likely can we inject fault in exactly one AES round?



More than 60% of the resulting faults are precise enough to corrupt exactly one AES round

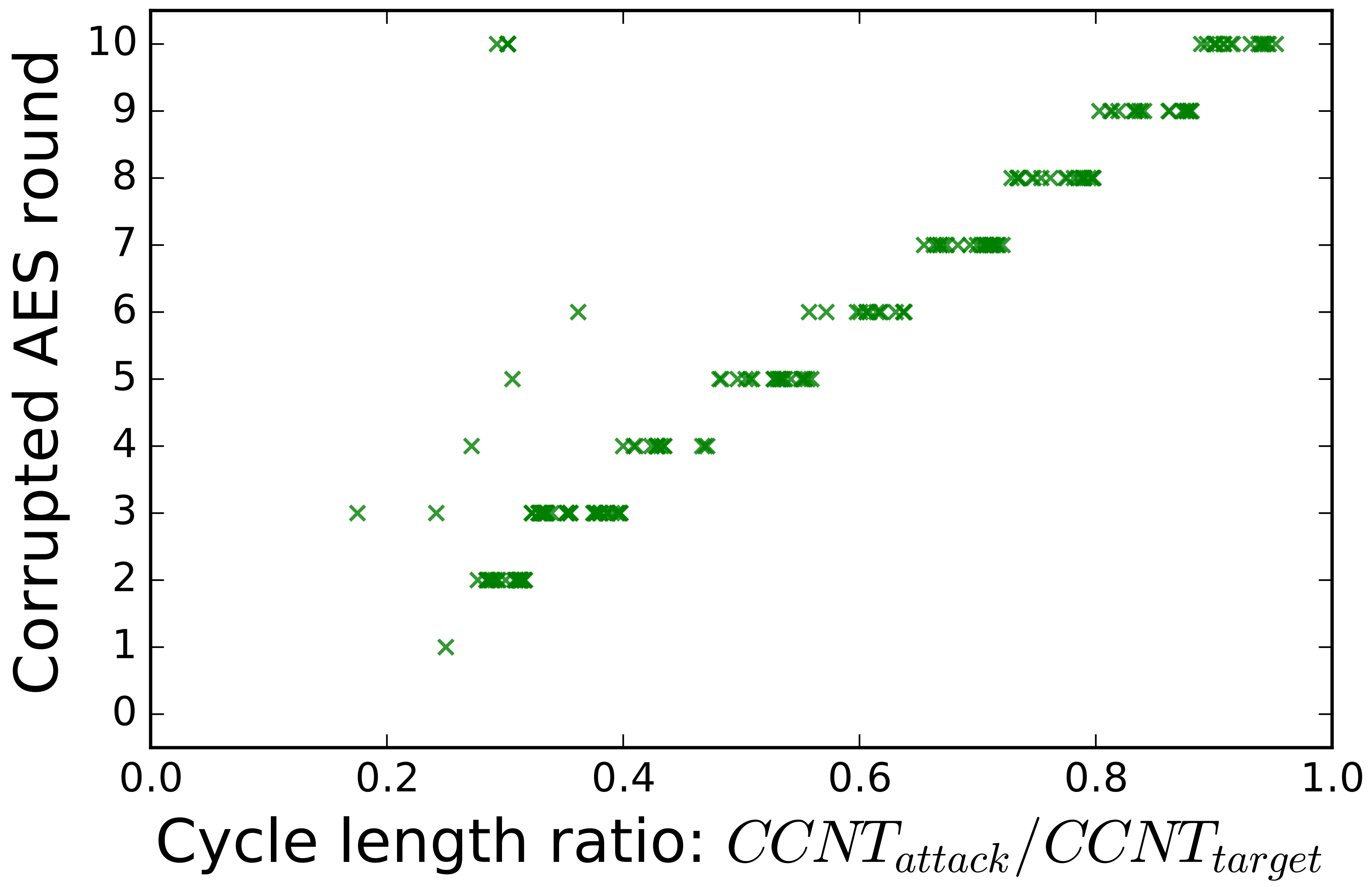
Key Inference Attack: Fault Model

Transience: How likely can we corrupt exactly one byte?



Out of the above faults that affect one AES round, more than half are transient enough to corrupt exactly one byte

Key Inference Attack: Results



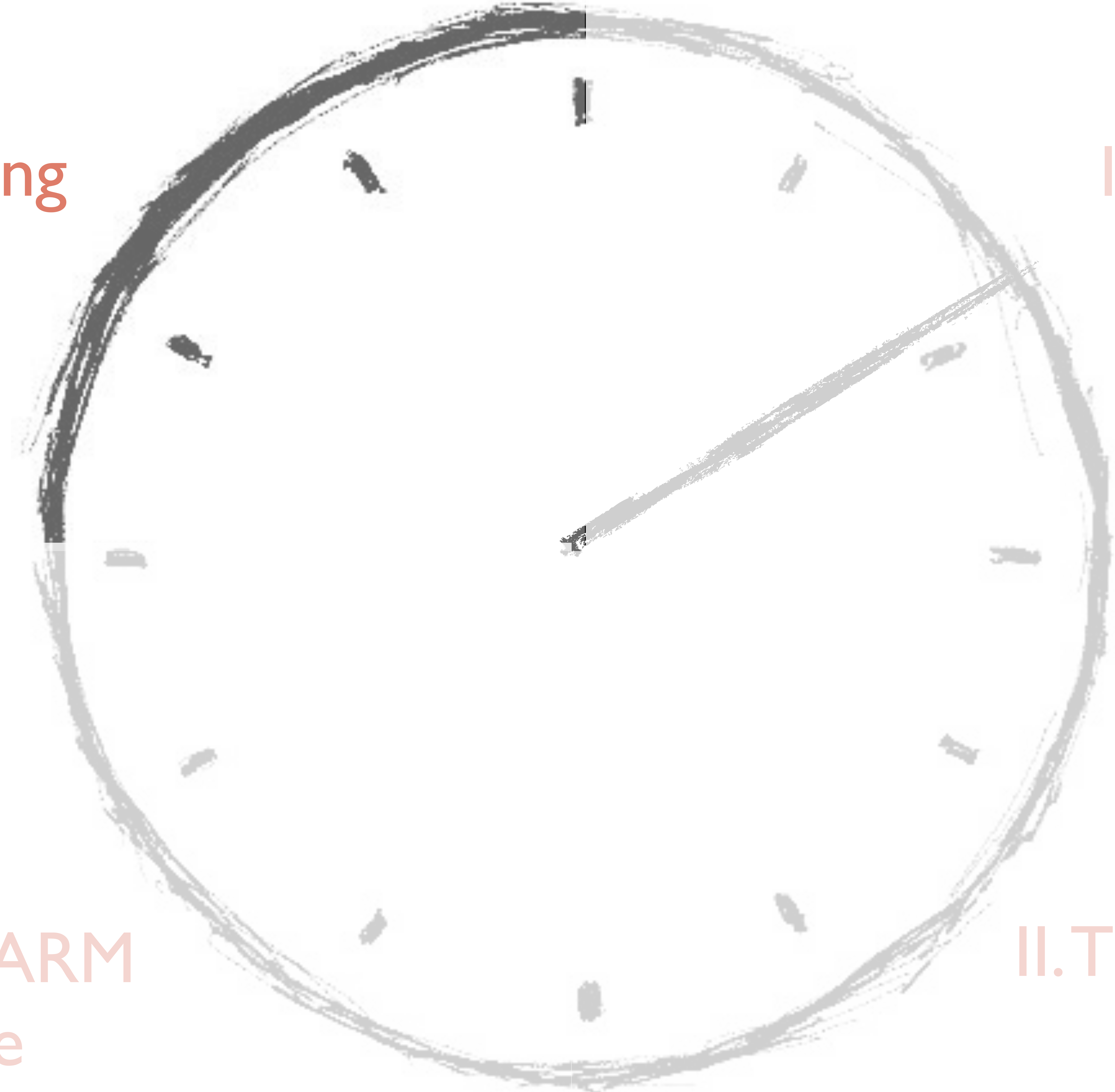
Controlling F_{pdelay} allows us to precisely time the delivery of the fault to the targeted AES round

Statistics:

- ~20 faulting attempts to induce one-byte fault to desired AES round.
- ~12 min on a 2.7GHz quad core CPU to generate 3650 key hypotheses

IV. Concluding Remarks

I. DVFS and Regulators

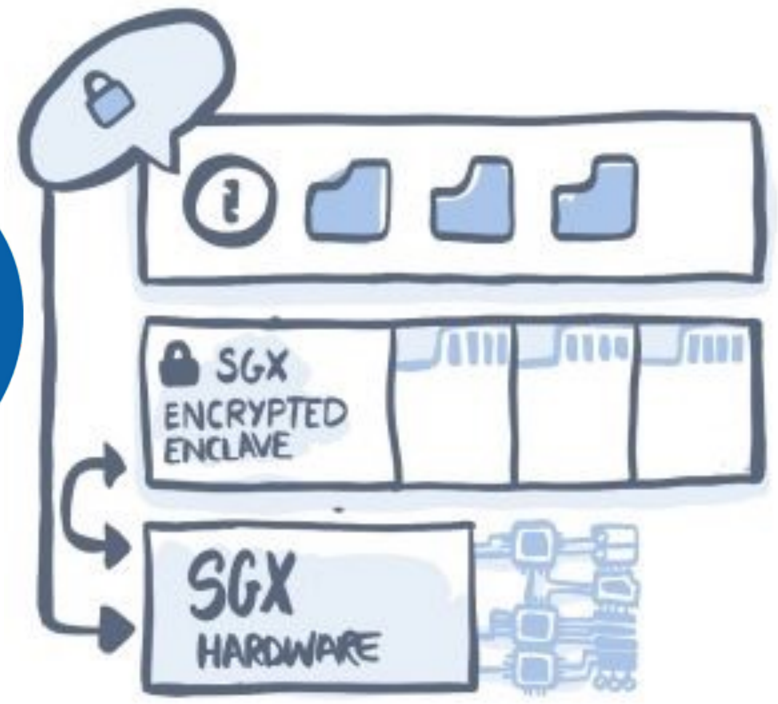


III. Attacking ARM Trustzone

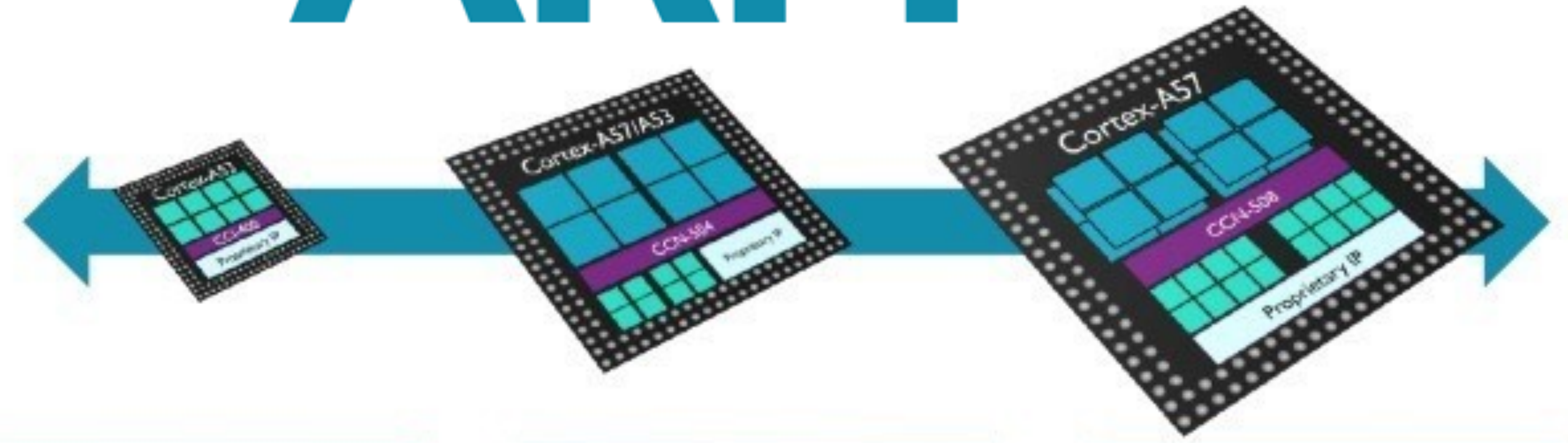
II. The CLKSCREW Attack

Attack Applicability to Other Platforms

Energy management mechanisms in the industry is trending towards finer-grained and increasingly heterogeneous designs



ARM v8



Cloud computing providers

Possible Defenses

Hardware-Level

- Operating limits in hardware
- Separate cross-boundary regulators
- Microarchitectural Redundancy

Software-Level

- Randomization
- Code execution redundancy

CLKSCREW: Exposing the perils of security-oblivious **Energy Management**

New attack surface via energy management software interfaces

Not a hardware or software bug
Fundamental design flaw in energy management mechanisms

Future energy management designs must take security into consideration

Adrian Tang - @0x0atang

Simha Sethumadhavan, Salvatore Stolfo