# Course Name

## Title:

## Subtitle

F. Gebali

EOW 433

Office Phone: 250-721-6509

https://ece.engr.uvic.ca/~fayez/

## Outline

# Course Name

## Title:

## Subtitle

F. Gebali

EOW 433

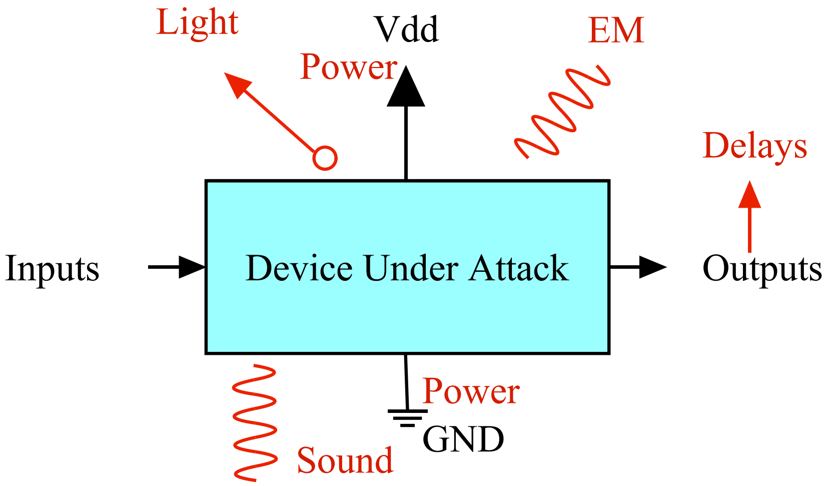
Office Phone: 250-721-6509

https://ece.engr.uvic.ca/~fayez/

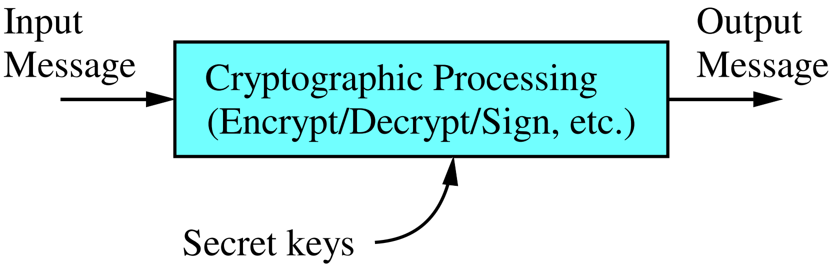
## Outline

## Introduction

## Side Channel Attack Channels



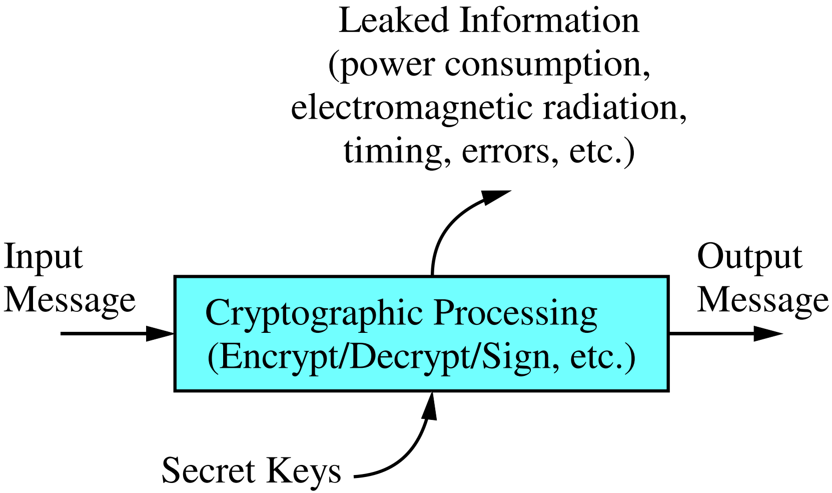
## Cryptographic Processing



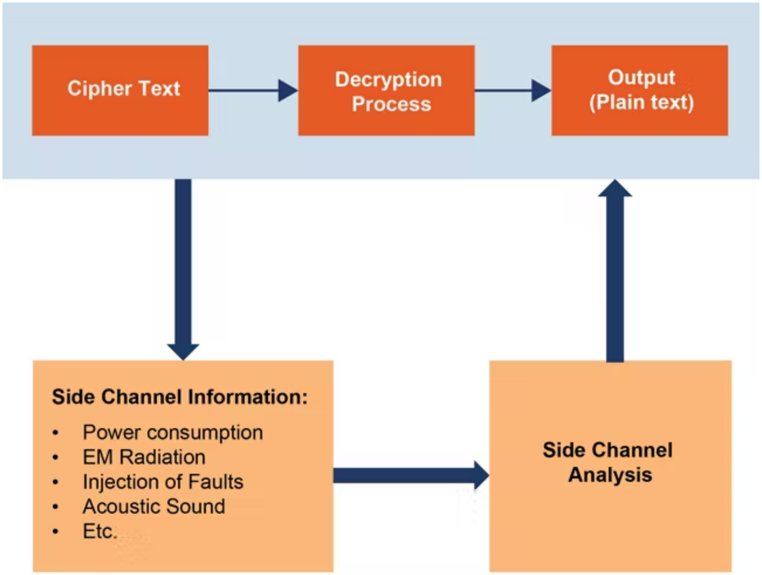
## Side-Channel Attacks (SCA)

1. Noninvasive and passive attack
2. Targets implementation of algorithm
3. Not interested in exploiting algorithm weaknesses
4. Use information obtained from physical implementation rather than crypto-analysis of the cipher
5. Exploit physical signals leaking from the hardware [1]:
    1. Power
    2. Electromagnetic radiation
    3. Delay

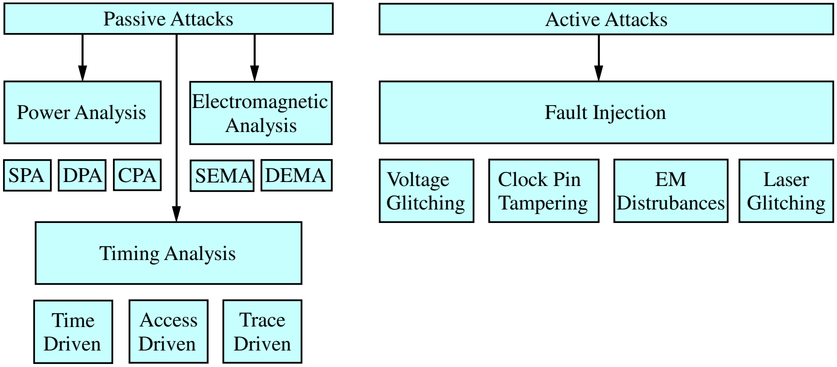
## Side Channel Attacks



## SCA During Decryption in IoT Devices



## Post-Silicon Attack Taxonomy [2] (does not include Trojans!)



## Power Analysis

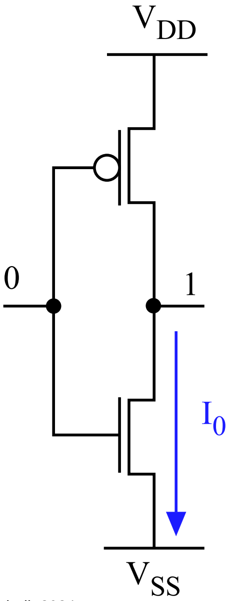
## Power Analysis Attacks

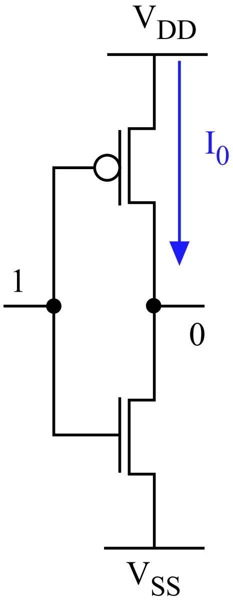
1. Analyze power consumption of device
2. Need to capture current from $V_{DD}$ of $V_{SS}$
3. Aims to reveal secret key when execution path depends on key bits
4. Succeeded in breaking AES secret key in few minutes

## Sources of Power Dissipation in ICs

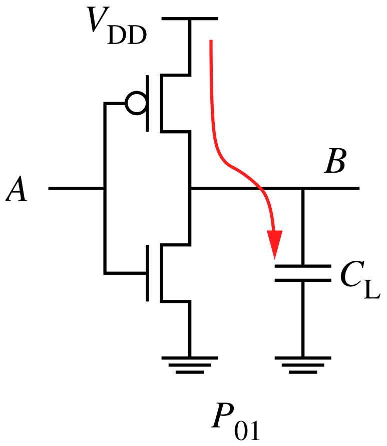
1. Static due to transistor & substrate leakage ($\propto V_{DD} I_0$)
2. Dynamic due to gate transitions & activities ($\propto CV_{DD}^2 f/2$)
3. Overlap/short-circuit due to using CMOS technology ($\propto V_{DD} If$)

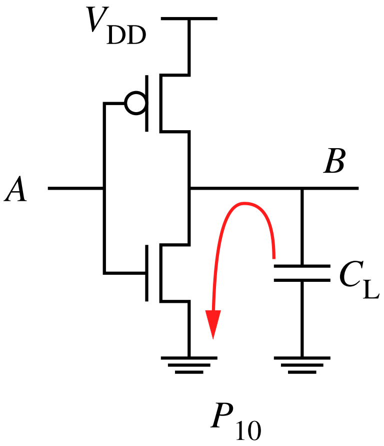
## Static Power Disspication: Leakage Current $I_0$



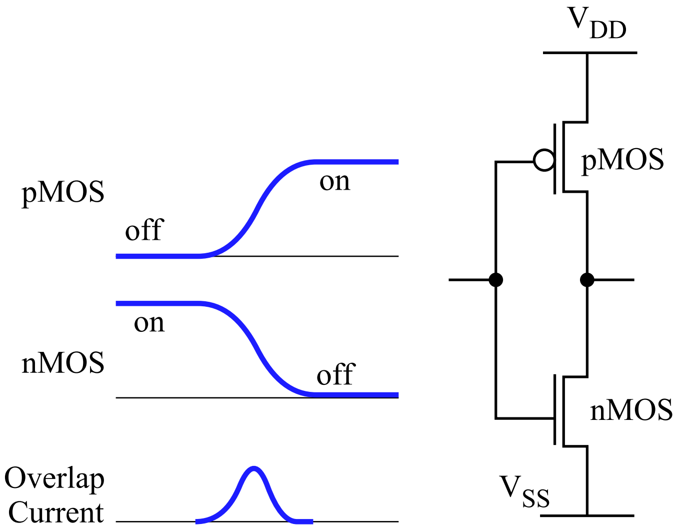


## Dynamic Power Dissipation: Parasitic Capacitor Charge/Discharge





## Overlap Power Dissipation: Slow Transition Time



## AND Gate Transitions

| a | b | q | Energy |
|---|---|---|---|
| $0 \to 0$ | $0 \to 0$ | $0 \to 0$ | $E_{0\to0}$ |
| $0 \to 0$ | $0 \to 1$ | $0 \to 0$ | $E_{0\to0}$ |
| $0 \to 0$ | $1 \to 0$ | $0 \to 0$ | $E_{0\to0}$ |
| $0 \to 0$ | $1 \to 1$ | $0 \to 0$ | $E_{0\to0}$ |
| $0 \to 1$ | $0 \to 0$ | $0 \to 0$ | $E_{0\to0}$ |
| $0 \to 1$ | $0 \to 1$ | $0 \to 1$ | $E_{0\to1}$ |
| $0 \to 1$ | $1 \to 0$ | $0 \to 0$ | $E_{0\to0}$ |
| $0 \to 1$ | $1 \to 1$ | $0 \to 1$ | $E_{0\to1}$ |
| $1 \to 0$ | $0 \to 0$ | $0 \to 0$ | $E_{0\to0}$ |
| $1 \to 0$ | $0 \to 1$ | $0 \to 0$ | $E_{0\to0}$ |
| $1 \to 0$ | $1 \to 0$ | $1 \to 0$ | $E_{1\to0}$ |
| $1 \to 0$ | $1 \to 1$ | $1 \to 0$ | $E_{1\to0}$ |
| $1 \to 1$ | $0 \to 0$ | $0 \to 0$ | $E_{0\to0}$ |
| $1 \to 1$ | $0 \to 1$ | $0 \to 1$ | $E_{0\to1}$ |
| $1 \to 1$ | $1 \to 0$ | $1 \to 0$ | $E_{1\to0}$ |
| $1 \to 1$ | $1 \to 1$ | $1 \to 1$ | $E_{1\to1}$ |

## Energy Statistics in AND Gate Transitions

1. Probability gate remains in 0 power state

$$p(E_{0\to 0}) = \frac{9}{16}$$

2. Probability gate remains in 1 power state

$$p(E_{1\to 1}) = \frac{1}{16}$$

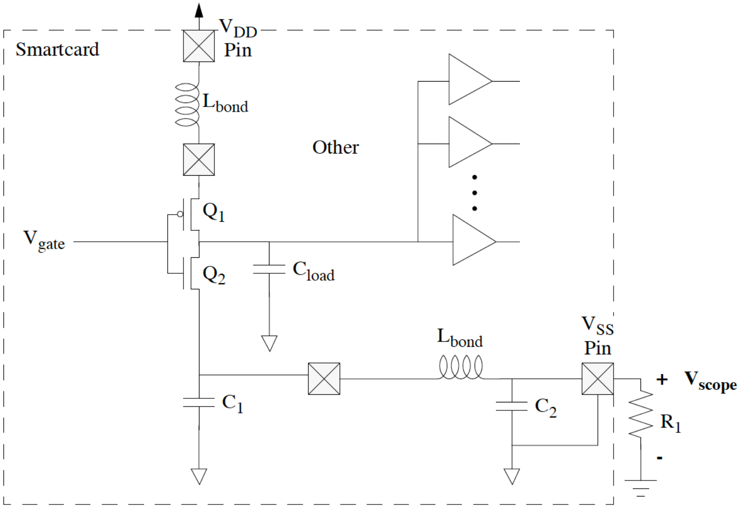
3. Probability of gate state changing from 0 to 1

$$p(E_{0\to 1}) = \frac{3}{16}$$

4. Probability of gate state changing from 1 to 0

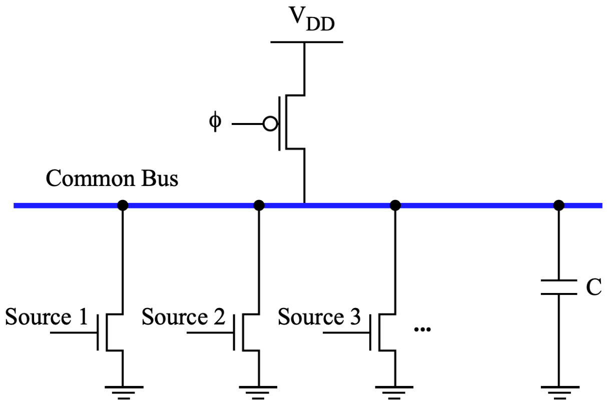
$$p(E_{1\to 0}) = \frac{3}{16}$$

## Dynamic Power Dissipation: SmartCard Power Consumption [3]

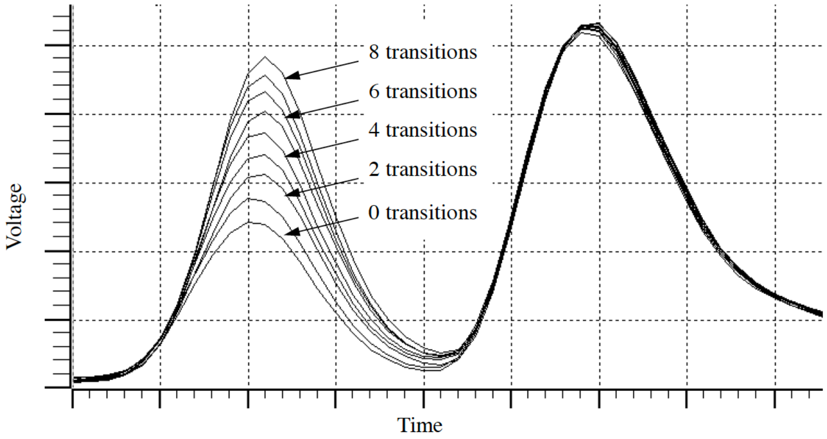


## Information Leakage from Data Bus

1. Hamming weight leakage in precharged bus: i.e. how many sources discharge bus to ’0’.
2. Transition count leakage: i.e. how many gates change state by data bus.



## Transition Count Power Leakage: 8-bit RAM to Register Transfer



## Power Analysis Types

## Types of Power Analysis

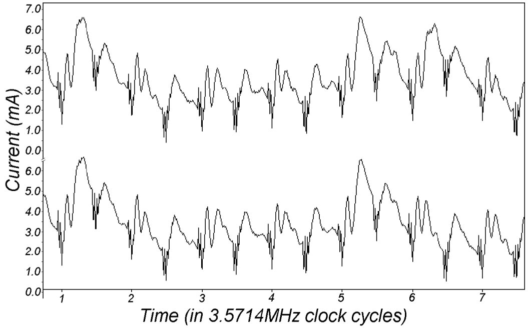
1. Simple power analysis (SPA)
2. Differential power analysis (DPA)
3. Correlation power analysis (CPA)

## SPA

## Simple power analysis (SPA) [4, 5]

1. Does not require DSP or statistical techniques
2. Visual analysis for patterns that identify key bits or functions
3. Used as first step before more sophisticated attacks
4. Allows to recognize instructions or groups of instructions
5. Infer hamming weight when loading on a bus
6. Simple to prevent

## SPA for 16-Round DES



1. Upper trace for 7 clock cycles when JMP is performed
2. Lower trace for 7 clock cycles when JMP is not performed
3. Divergence is in clock cycle 6

## Sources of Power Variations

1. Large-scale features identify the rounds or iterations
2. Small-scale features identify individual operations can be identified, e.g. multiplication vs. squaring. This can reveal if key bit is 0 or 1
3. Higher-magnification can even reveal data bits in reduction or multiplication or division operations since these could be done sequentially

## DPA

## Differential Power Analysis (DPA)

1. Attacker need not know hardware architecture of device
2. Requires a large number of traces
3. Requires two phases: (a) data collection and (b) data analysis
4. Attack uses statistical analysis and error correction techniques
5. Difficult to avoid

## Differential Power Analysis (DPA) Analysis [3]

1. Attacker gathers $N$ random plaintext input (PTI) messages and their ciphertext output (CTO) using DES key $Ki$
2. Attacker defines a partitioning function $D(CTO, CTO, K16)$
3. Low and high power traces are constructed:

$$\begin{aligned} \mathbb{S}_0 &= \{S(i,j) | D = 0\} \\ \mathbb{S}_1 &= \{S(i,j) | D = 1\} \end{aligned}$$

4. $D(\cdot,\cdot,\cdot)$ was chosen as

$$D(CTO_1, CTO_6, K16) = CTO_1 \oplus SBOX_1(CTO_{1:6} \oplus K16_{1:6})$$

## Differential Power Analysis (DPA) Analysis [3]

4. Average power is obtained

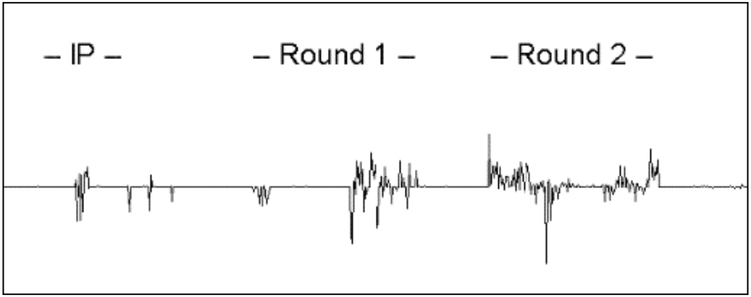
$$\begin{aligned} P_0(j) &= (1/|\mathbb{S}_0|) \sum_{S(i,j)\in\mathbb{S}_0} S(i,j), \\ P_1(j) &= (1/|\mathbb{S}_1|) \sum_{S(i,j)\in\mathbb{S}_1} S(i,j) \end{aligned}$$

5. A discrete time DPA signal trace is obtained

$$T(j) = P_0(j) - P_1(j)$$

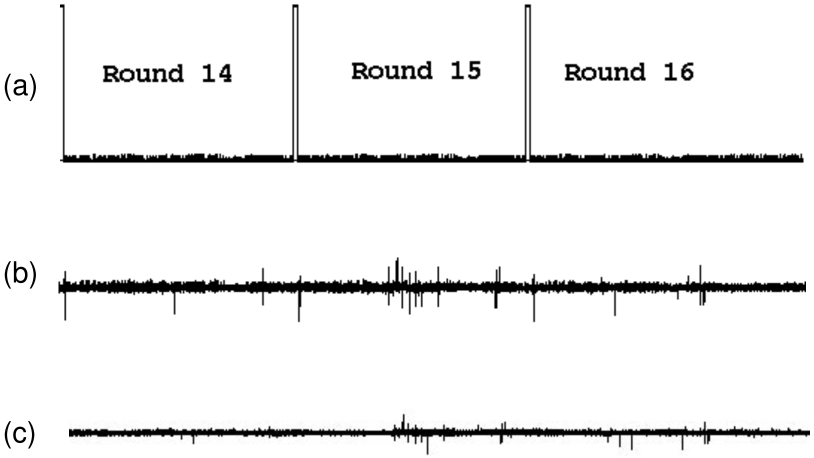
6. $T(j)$ is used to guess the secret key

## DPA Results for DES [6]



1. Function $D$ is chosen as bit $S(i, 5)$
2. This bit is placed in $R$ register and affects Round 1
3. This also affects Round 2

## DPA Results for DES [3]



(b) when key is correct and (c) when key is incorrect

## DPA Countermeasures

1. Filters to reduce power supply fluctuations
2. Physical shielding to reduce leakage
3. Gate/circuit design that reduces leakage
4. Algorithm designs to obfuscate operations
5. Introduce noise in leaked information
6. Use dataflow computing paradigm

## Timing Attack

## Encrypt/Decrypt Timing Variations Causes

1. Value of encryption key bit 0 or 1
2. Optimizations that depend on input
3. Bypass unnecessary operations
4. Branching and conditional statements
5. Cache hits
6. Processor instructions that require different times (multiplication, division)

## Timing Attack Countermeasures [7]

1. Ensure all operations take same time (not practical)
2. Use timer to delay output results (not practical)
3. Introduce random delays
4. Introduce dummy instructions/operations
5. Hide inputs to the modular exponentiation
6. Use blinding as suggested by Kocher [7]

## Other Timing Attack Countermeasures [9]

1. Use parallel implementations of the algorithm
2. Use dataflow processing
3. Randomized dataflow processing [8]

## Timing Attack: Attack on a Password Verification

1. Assume a password is 8-bytes long
2. Guessing the password would require $2^{64}$ trials
3. Password verification scans the input bytes sequentially (see pseudo code on next slide)

## Timing Attack: Insecure Password Verification Sequential Algorithm

```
1: function $flag = verify\_PWD(\widehat{P},\ P)$
2: for $i = 1:7$ do
3:     if $\widehat{p}_i \neq p_i$ then
4:         return $flag = `false'$
5:     end if
6: end for
7: return $`true'$
```

1. Attacker measures delay to get ‘*false*’
2. Can guess bits one bit at-a-time
3. Exit loop as soon as a mismatch is found

## Timing Attack: Attack Steps

1. Use all values of password first byte

$$\widehat{P} = (n, 0, 0, 0, 0, 0, 0, 0), \qquad 0 \leq n < 256$$

2. Measure execution time $\tau(n)$ for all values of $n$

3. Extract maximum execution time as indicative of a correct byte:

$$\tau(n_0) = \max_{0 \leq n < 256} \tau(n)$$

4. Value of byte 0 is $n_0$.

5. Repeat for all remaining bytes

## Timing Attack: Secure Password Verification Sequential Algorithm

```
1: function $flag = verify\_PWD(\widehat{P},\ P)$
2: $flag = $ 'true'
3: for $i = 1 : 7$ do
4:     if $\widehat{p}_i \neq p_i$ then
5:         $flag = $ 'false'
6:     end if
7: end for
8: return  $flag$
```

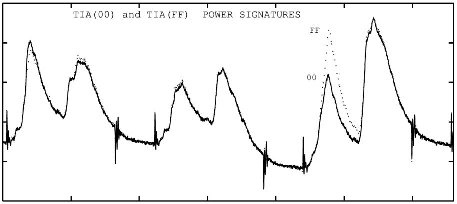
No early exit from loop. Constant time implementation.

## EM Attacks

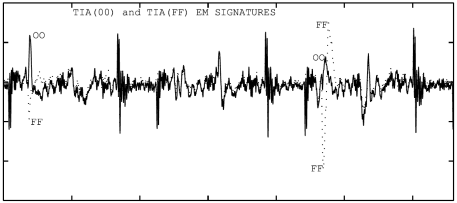
## Electromagnetic Attacks [10]

1. Requires chip-scale EM inductive probes ($\approx 100\mu m$)
2. ADC Sampling frequencies $> 1GHz$ are now possible
3. Sometimes, package surface can be eroded for probe to be closer to chip
4. Most activity is near CPU

## EM Attacks



Power consumption



EM signal

## EMA Countermeasures

1. Shield hardware
2. Redundant EM noise-generating modules
3. Extra dummy instructions or operations

## Fault Injection Attack (FIA)

## Fault Injection Attacks (FIA)

1. Fault injection attack is an active attack (not passive like SCA)
2. Faulty input is injected to produce erroneous outputs
3. Used in combination with SCA to reduce complexity of attack
4. There is invasive (destructive) and non-invasive FIA
5. Creates transient fault during an operation to reduce or disable security features & countermeasures

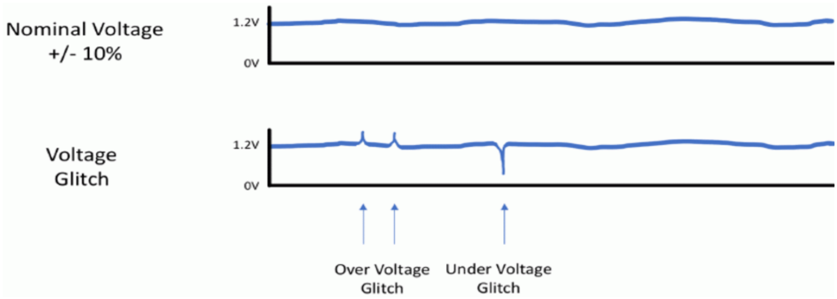
## Fault Injection Attack Approaches [11, 12]

1. Reduce supply voltage (transient faults)
2. Injection of power spikes or brownouts (voltage glitching)
3. Vary clock frequency (clock glitching)
4. Overheat the device (affect DRAM)
5. Shine intense light (laser or flash)
6. Strong EM pulses

## FIA Through Firmware Boot

1. Assume the firmware is on an external malicious flash
2. CPU attempts to authenticate bytes from the flash
3. Authentication is through hashing or CRC, etc.
4. Authenticated boot is indicated to CPU by a single “yes” pulse
5. Attacker might try to mimic the “yes” pulse through a glitch

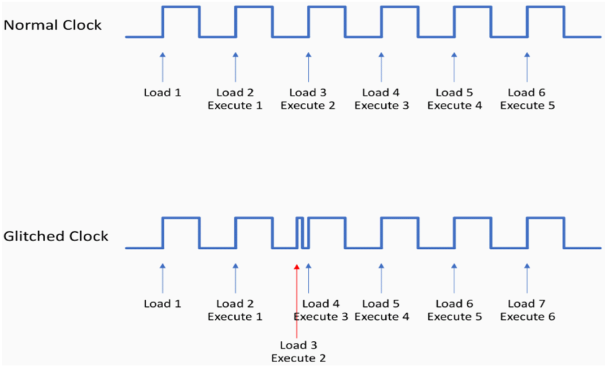
## FIA Voltage Glitching



## FIA Clock Glitching

1. Clock glitching attempts to tamper with external clock lines
2. Short pulse(s) introduced creating early rising clock edge(s)
3. The glitch is synchronized with instruction execution
4. During RAM read, data is loaded before it is stable on bus
5. During RAM fetch, instruction is not executed

## FIA Clock Glitching



1. Instruction #2 is not executed
2. Preempted by Instruction #3

## FIA Countermeasures: Rambus CryptoManager Root of Trust

1. Security co-processor
2. Full-programmable
3. FIPS 140-2 compliant
4. Offer layered security
5. Protect agains tampering, software and hardware attacks
6. Cryptographic accelerators: AES, 3DES, HMAC, SHA-2/SHA-3
7. Can be offered with DPA protection

[1] J. Kelsey, B. Schneier, D. Wagner, and C. Hall, “Side channel cryptanalysis of product ciphers,” in *Proceedings of ESORICS*, Sep. 1998, pp. 97–110.

[2] S. Bhunia and M. Tehranipoor, *Hardware Security*. Morgan Kaufmann, 2019.

[3] T. Messerges, E. Dabbish, and R. Sloan, “Investigation of power analysis attacks on smartcards,” in *Usenix Workshop on Smartcard Technology*, 1999.

[4] C. Clavier, D. Marion, and A. Wurcker, *Simple Power Analysis on AES Key Expansion Revisited*. Springer, 2014, pp. 279–297.

[5] J. C. Courrège, B. F., and M. Roussellet, *Smart Card Research and Advanced Application*. Springer, 2010, ch. Simple Power Analysis on Exponentiation Revisited, pp. 65–79.

[6] P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis: Leaking secrets,” in *CRYPTO’99", volume 1666 of LCNS*, 1999, pp. 388–397. Springer.

[7] P. Kocher, “Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems,” in *Advances in Cryptology- CRYPTO’96, Springer-Verlag*, 1996, pp. 104–113.

[8] A. Alzahrani and F. Gebali, “Multi-core dataflow design and implementation of secure hash algorithm-3,” *IEEE Access*, vol. 6, pp. 6092–6102, 2018.

[9] J.-F. Dhem, F. Koeune, P.-A. Leroux, P. Mestré, J.-J. Quisquater, and J.-L. Willems, “A practical implementation of the timing attack,” Université Catholique de Louvain, Tech. Rep., 1998.

[10] K. Gandolfi, C. Mourtel, and F. Olivier, *Cryptographic Hardware and Embedded Systems – CHES*. Springer,

2001, ch. Electromagnetic Analysis: Concrete Results, pp. 251–261.

[11] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache, “Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures,” *Proceedings of the IEEE*, vol. 100, no. 11, pp. 3056–3076, 2012.

[12] B. Stevens, “Fault injection attacks: A growing plague,” https://www.eeweb.com/profile/bstevens/articles/fault-injection-attacks-a-growing-plague, Mar. 2019.