Protection of Arithmetic Circuits against Physical Attacks

Arnaud Tisserand

CNRS, Lab-STICC

LIP Lyon, 2018.11.09

Summary

- Introduction
- Physical Attacks
- Arithmetic Circuits
- Protections
- Conclusion and References

Introduction

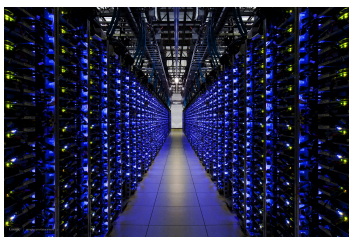
Applications with Security Requirements

- medical devices
- home automation
- digital administration
- e-commerce
- transports
- communications: cell. phones, Internet, industrial networks...
- IOT
- WSN
- embedded systems
- cloud computing
- RFID tags
- smart { grids | cities | buildings | ... }
- ...

Security and Embedded Systems

Integrated circuits perform security tasks, somewhere in the system...

Cases where a close access is difficult:





Cases where a close access can be possible:

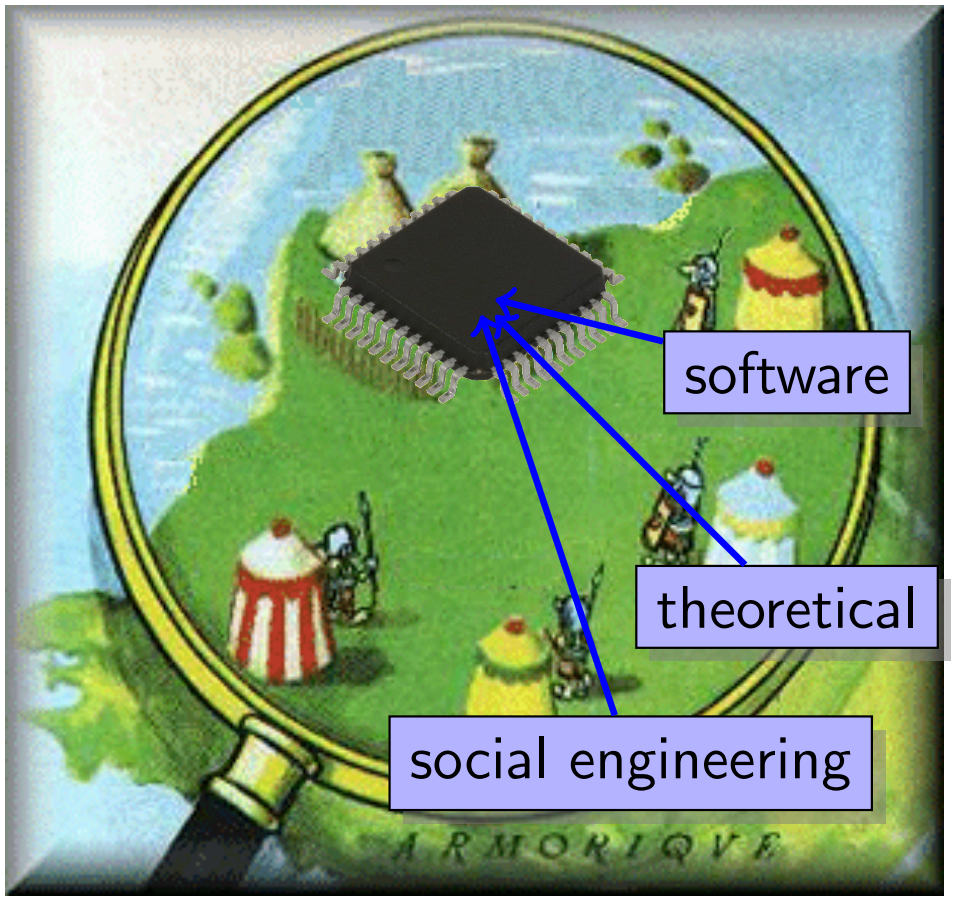



Physical Attacks

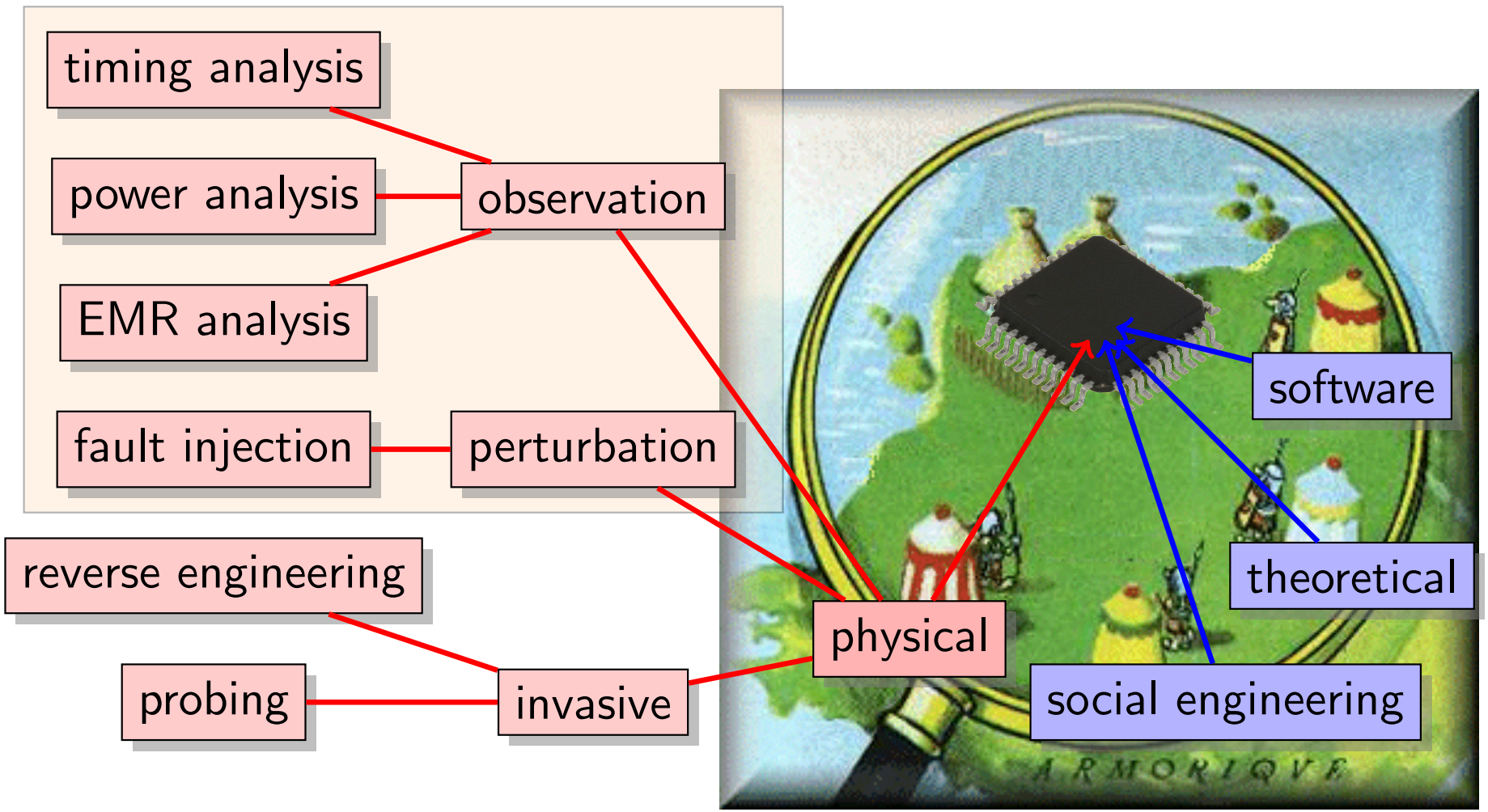
Attacks

Attacks

Attacks



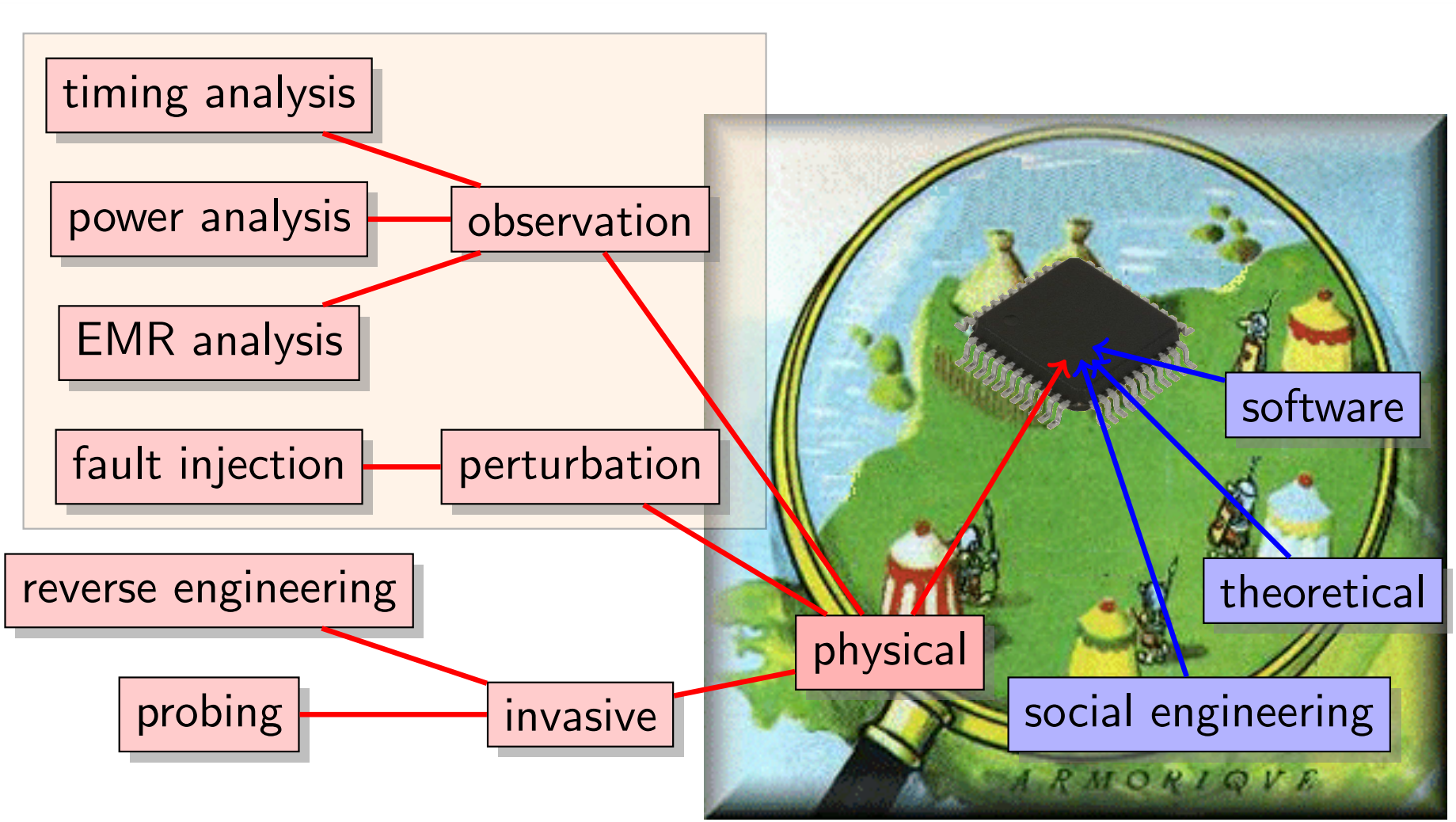
Attacks



EMR = Electromagnetic radiation

Attacks

Types of attacks (non-exhaustive):



EMR = Electromagnetic radiation

Observation Attacks

Question: what can/should be measured?

Answer: everything that can "enter" and/or "get out" in/from the device

- computation time
- power consumption
- electromagnetic radiation
- temperature
- sound
- number of cache misses
- number and type of error messages
- ...

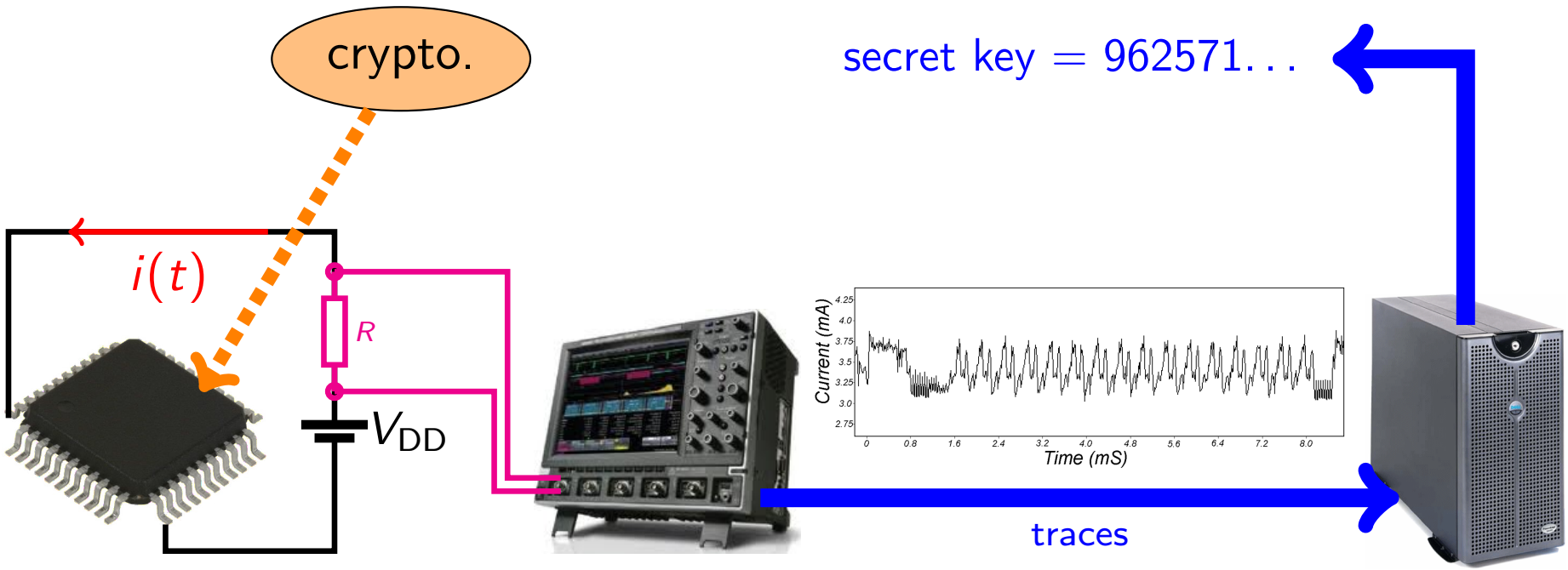
The measured parameters may provide informations on:

- global behavior (temperature, power, sound...)
- local behavior (microprobe, # cache misses...)

Power Consumption Analysis

General principle:

1. measure the current $i(t)$ in the cryptosystem
2. use those measurements to "deduce" secret informations



Differences & External Signature

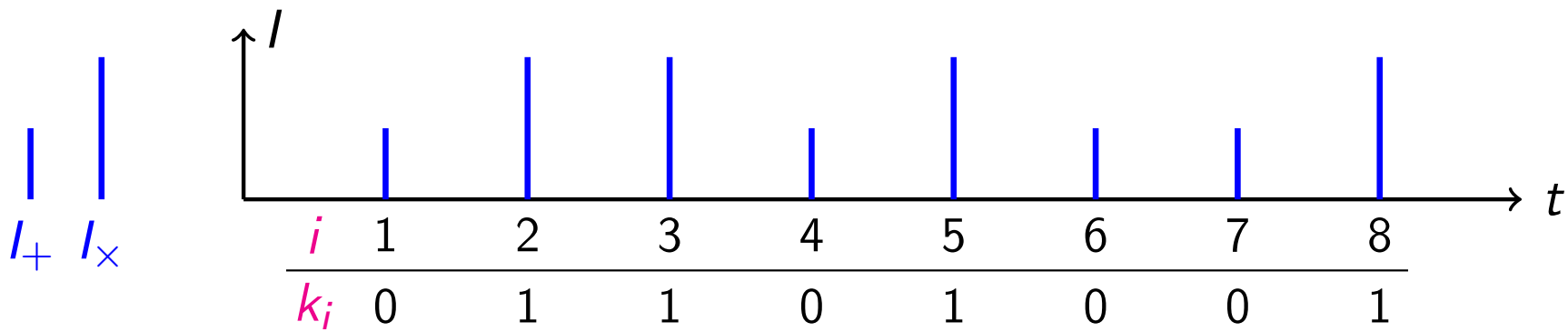
An algorithm :

$r = 0$
for i from 1 to n do
    if $k_i = 0$ then
        $r = r + a$
    else
        $r = r \times b$

Differences & External Signature

An algorithm has a current signature :

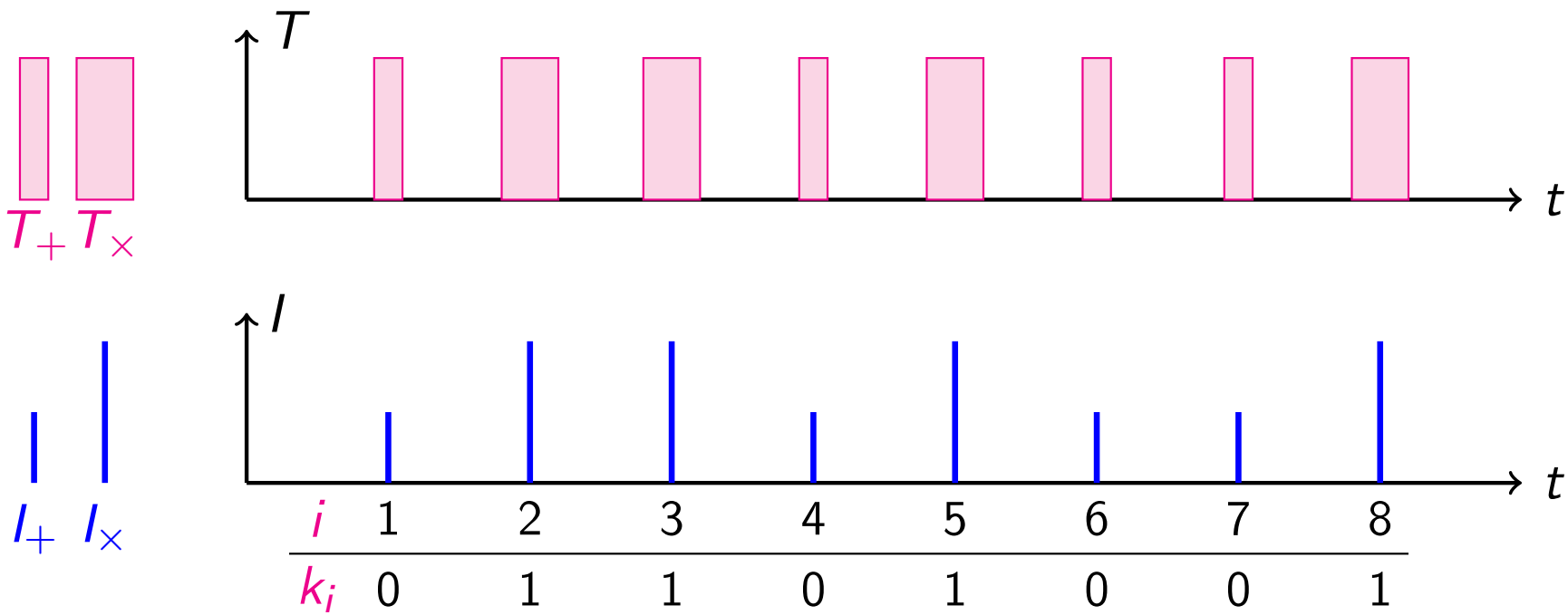
```
r = 0
for i from 1 to n do
    if k_i = 0 then
        r = r + a
    else
        r = r × b
```



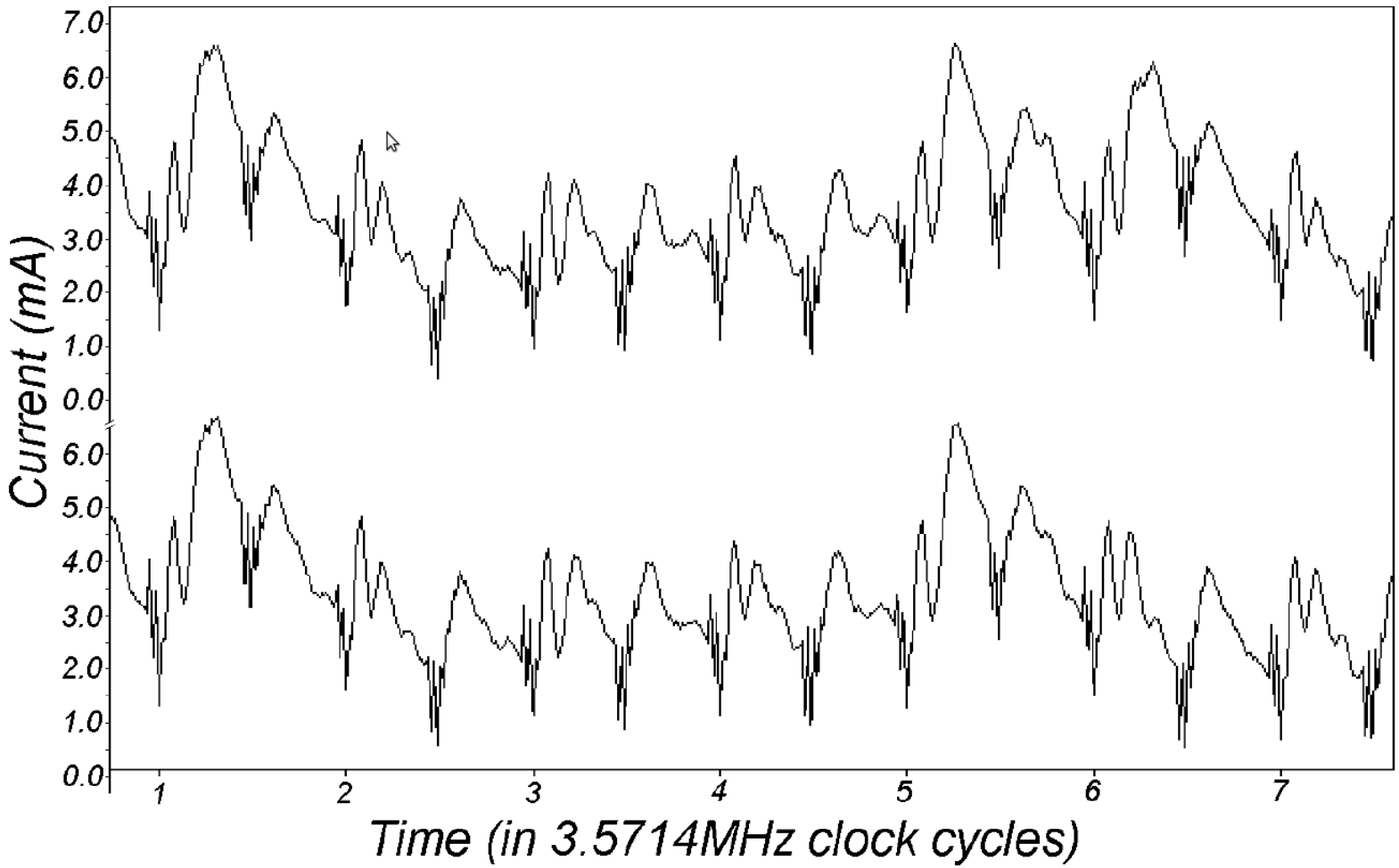
Differences & External Signature

An algorithm has a current signature and a time signature:

```
r = 0
for i from 1 to n do
    if k_i = 0 then
        r = r + a
    else
        r = r × b
```

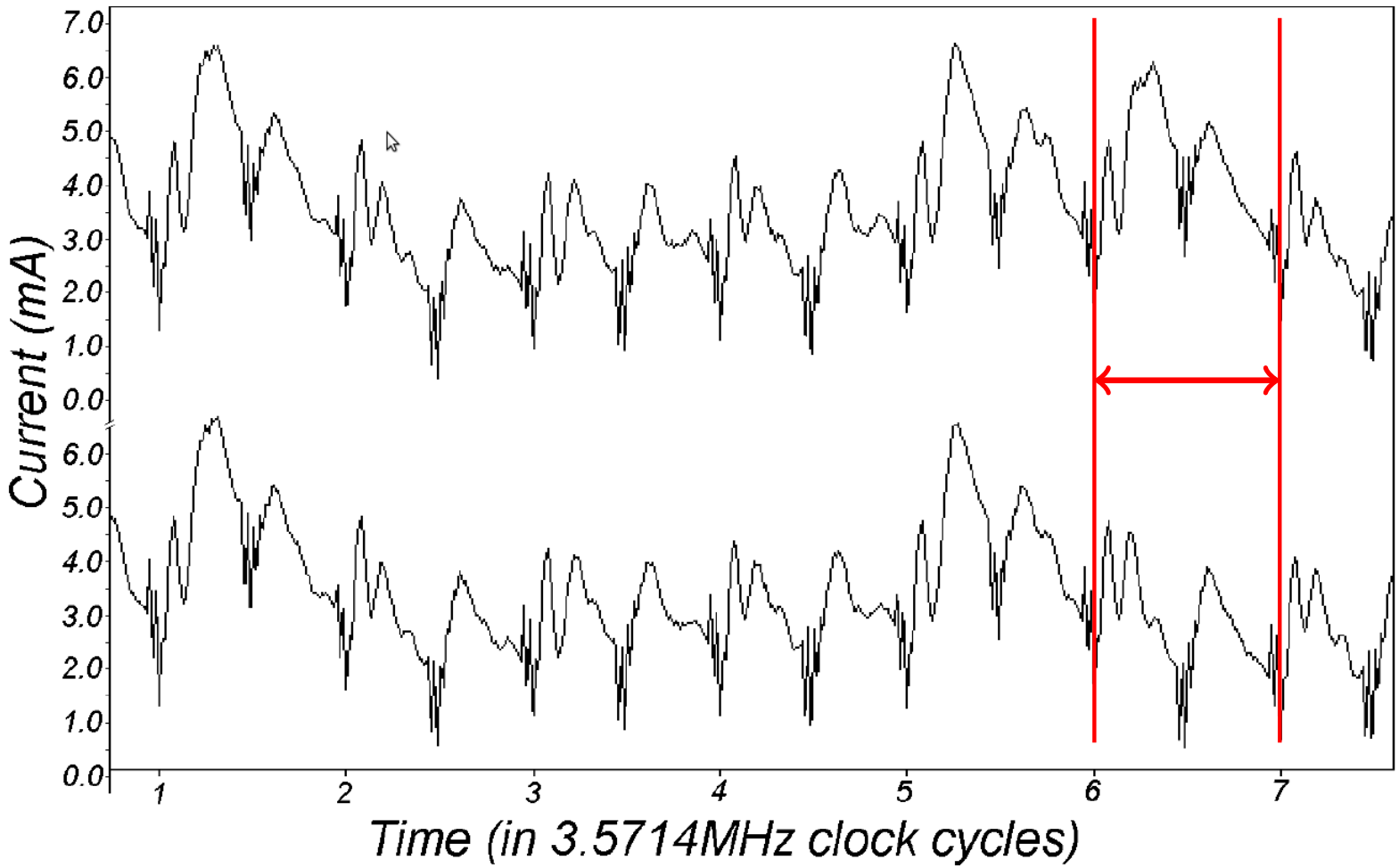


Observation Attacks



Source: [9]

Observation Attacks



Source: [9]

Perturbation or Fault Injection Attacks

Typical techniques:

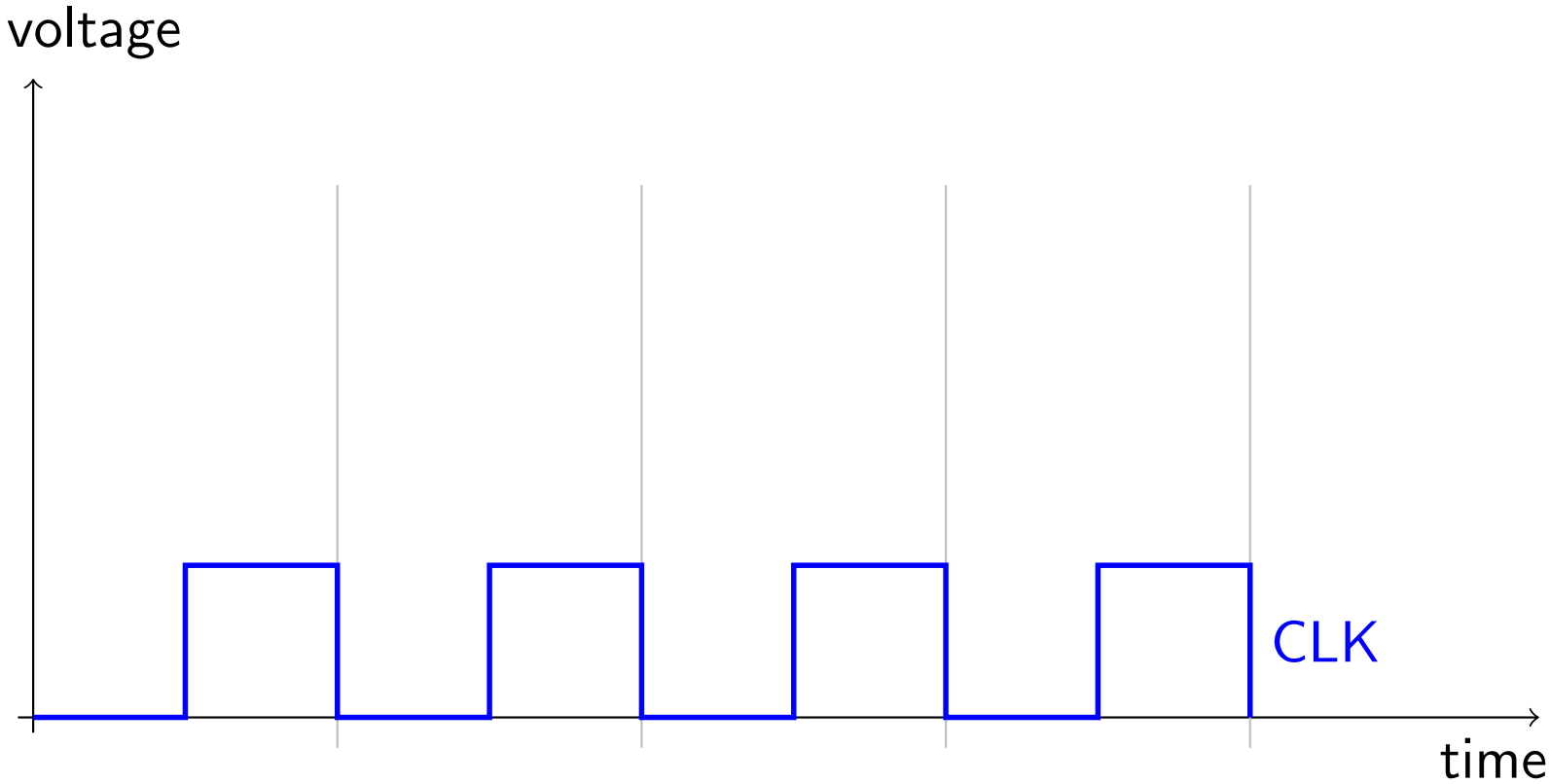
- perturbation in the power supply voltage
- perturbation of the clock signal
- temperature (over/under-heating the chip)
- radiation or electromagnetic (EM) disturbances
- exposing the chip to intense lights or beams
- etc

Accuracy:

- time: part of clock cycle, clock cycle, code block (instruction sequence)
- space: gate, block, unit, core, chip, package
- value: set to a specific value, bit flip, stuck-at 0 or 1, random modification

Perturbation on the External Clock

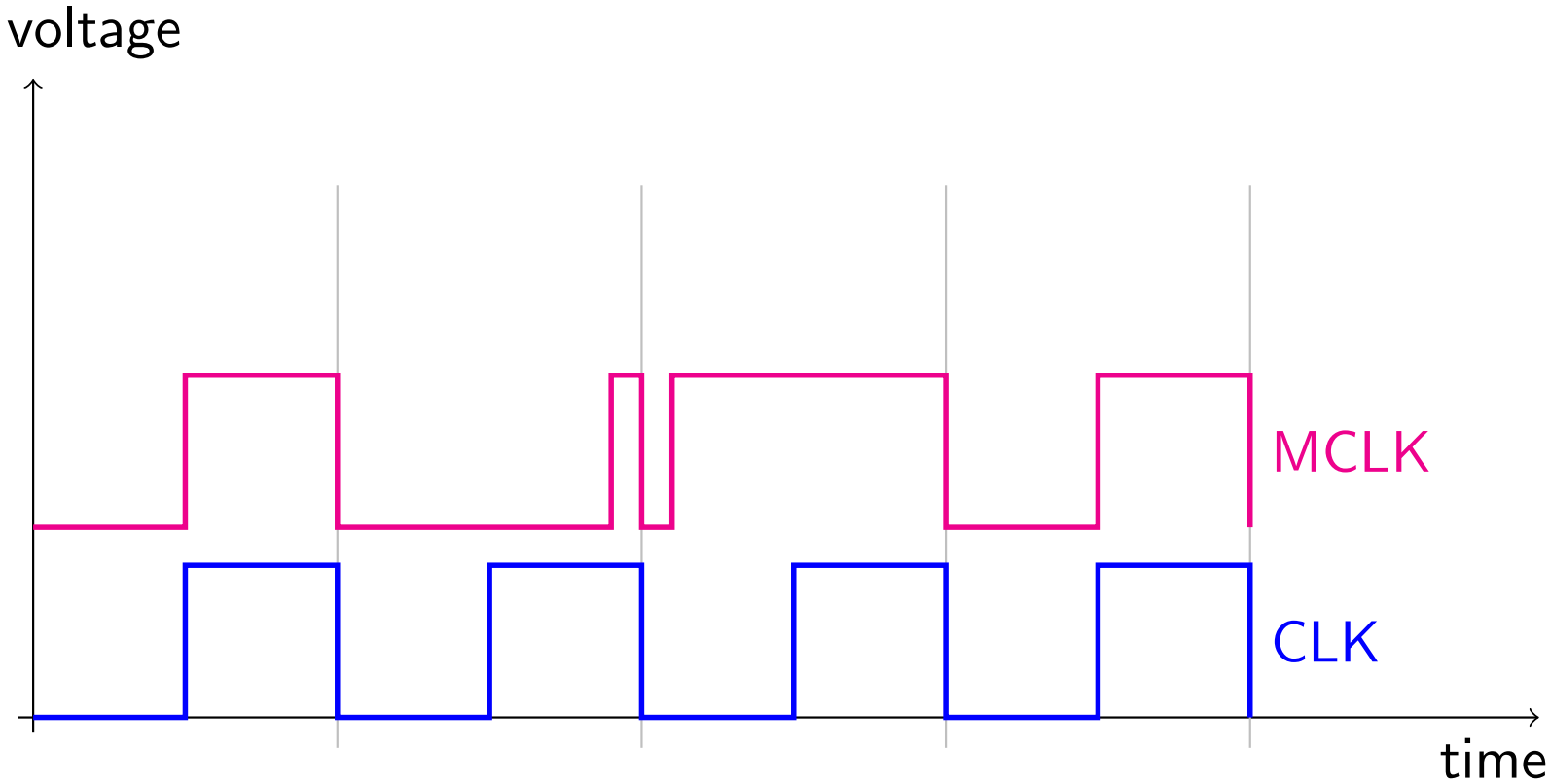
Principle:



- Normal clock (at a given frequency, duty cycle $\approx 50\%$)

Perturbation on the External Clock

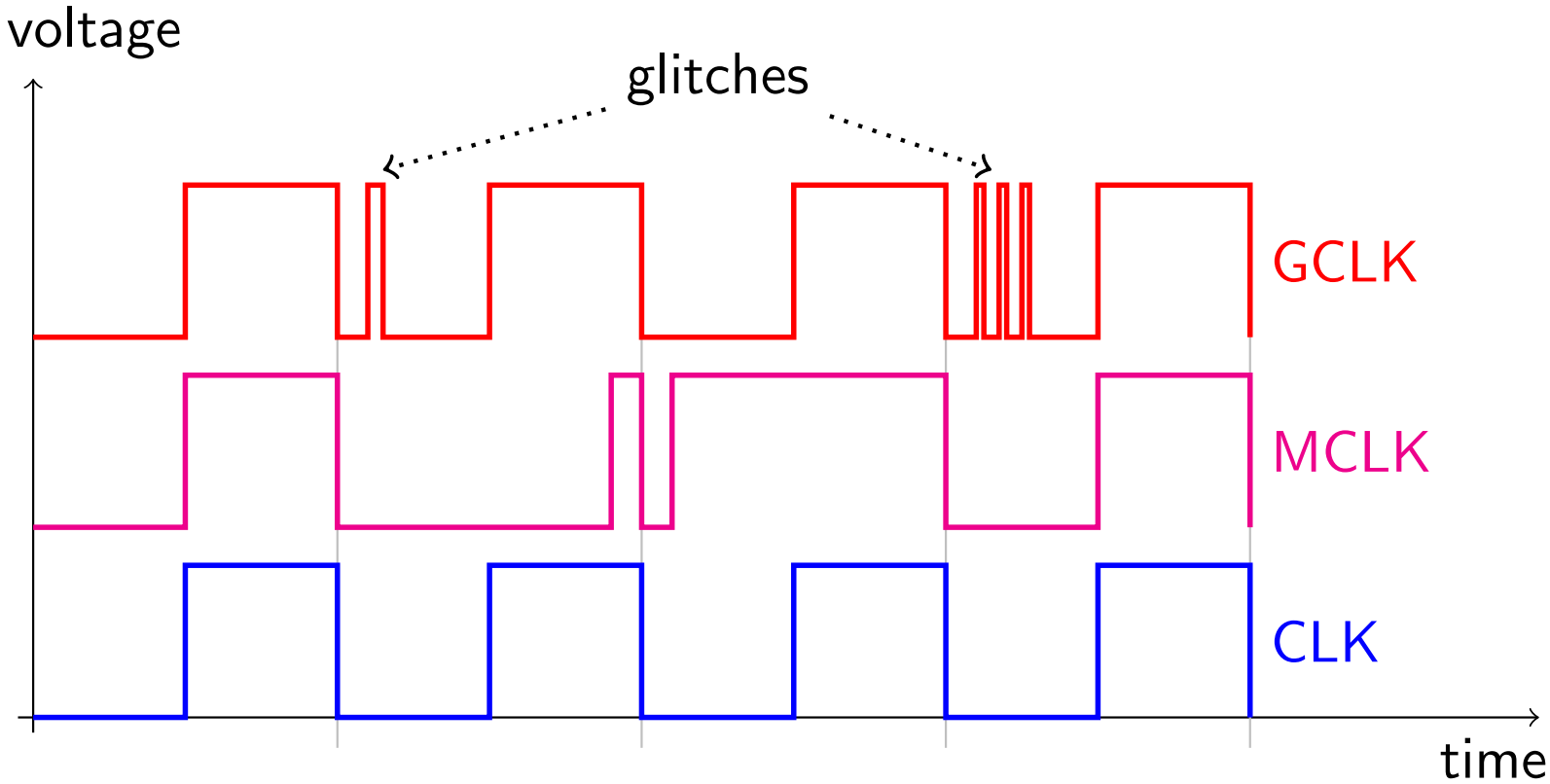
Principle:



- Normal clock (at a given frequency, duty cycle $\approx 50\%$)
- Clock with a modified duty cycle

Perturbation on the External Clock

Principle:



- Normal clock (at a given frequency, duty cycle $\approx 50\%$)
- Clock with a modified duty cycle
- Glitched clock
- Etc.

Clock Glitch Attack Example

Source: paper [1] presented at FDTC 2011 conference

Setup: AVR ATMega 163 microcontroller @ 1MHz

mode	glitch period	cycle	instruction	opcode (bin)
normal	-	i	`NOP`	0000 0000 0000 0000
normal	-	$i+1$	`EOR R15,R5`	0010 0100 1111 0101

Clock Glitch Attack Example

Source: paper [1] presented at FDTC 2011 conference

Setup: AVR ATMega 163 microcontroller @ 1MHz

mode	glitch period	cycle	instruction	opcode (bin)
normal	-	i	`NOP`	0000 0000 0000 0000
normal	-	$i+1$	`EOR R15,R5`	0010 0100 1111 0101
glitch	59 ns	$i+1$	`NOP`	0000 0000 0000 0000

Clock Glitch Attack Example

Source: paper [1] presented at FDTC 2011 conference

Setup: AVR ATMega 163 microcontroller @ 1MHz

mode	glitch period	cycle	instruction	opcode (bin)
normal	-	i	`NOP`	0000 0000 0000 0000
normal	-	$i+1$	`EOR R15,R5`	0010 0100 1111 0101
glitch	59 ns	$i+1$	`NOP`	0000 0000 0000 0000

mode	glitch period	cycle	instruction	opcode (bin)
normal	-	i	`NOP`	0000 0000 0000 0000
normal	-	$i+1$	`SER R18`	1110 1111 0010 1111

Clock Glitch Attack Example

Source: paper [1] presented at FDTC 2011 conference

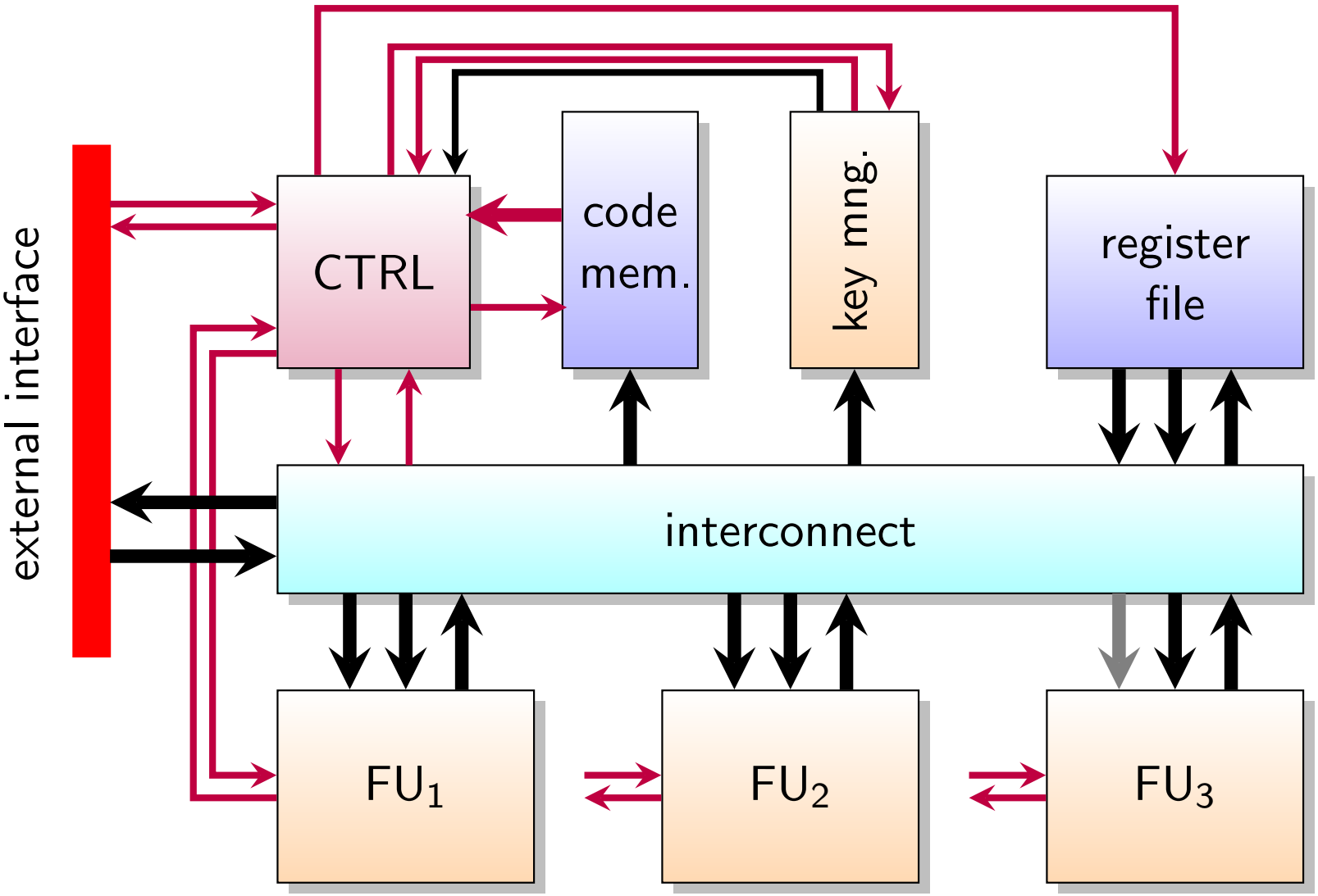
Setup: AVR ATMega 163 microcontroller @ 1MHz

mode	glitch period	cycle	instruction	opcode (bin)
normal	-	i	`NOP`	0000 0000 0000 0000
normal	-	$i+1$	`EOR R15,R5`	0010 0100 1111 0101
glitch	59 ns	$i+1$	`NOP`	0000 0000 0000 0000

mode	glitch period	cycle	instruction	opcode (bin)
normal	-	i	`NOP`	0000 0000 0000 0000
normal	-	$i+1$	`SER R18`	1110 1111 0010 1111
glitch	61 ns	$i+1$	`LDI R18,0xEF`	1110 1110 0010 1111
glitch	60 ns	$i+1$	`SBC R12,R15`	0000 1000 0010 1111
glitch	59 ns	$i+1$	`NOP`	0000 0000 0000 0000

Arithmetic Circuits

Example of Crypto-Processor Architecture



Functional Units: $\pm$, $\times$, $\div$ in finite fields $\mathbb{F}_p$ or $\mathbb{F}_{2^m}$ with $20 - 8000$ bits elements and (small) vectors/matrices

Protections

Protections

Principles for preventing attacks:

- embed additional protection blocks
- modify the original circuit into a secured version
- application levels: circuit, architecture, algorithm, protocol...

Protections

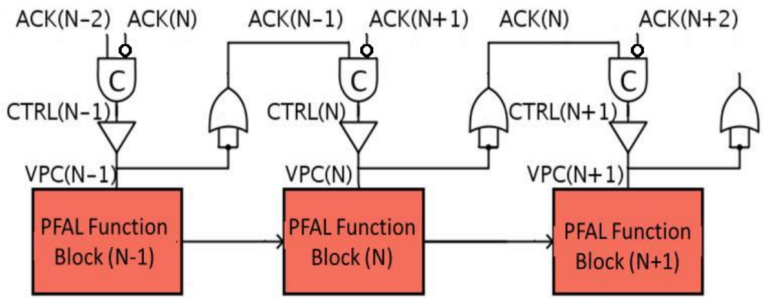
Principles for preventing attacks:

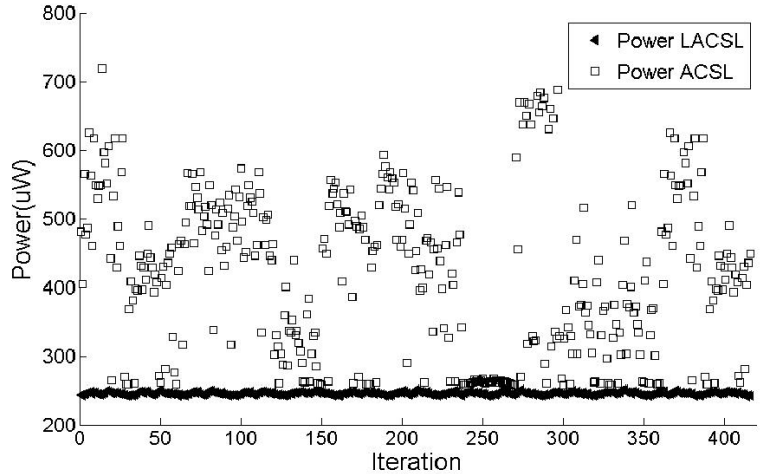
- embed additional protection blocks
- modify the original circuit into a secured version
- application levels: circuit, architecture, algorithm, protocol...

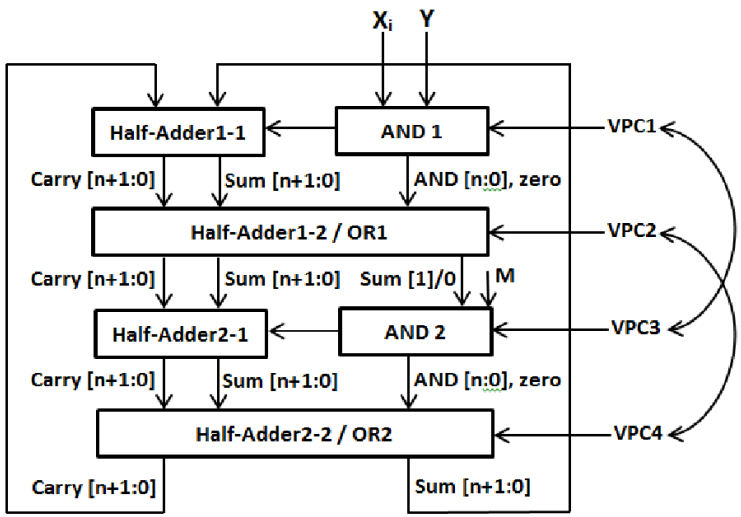
Countermeasures:

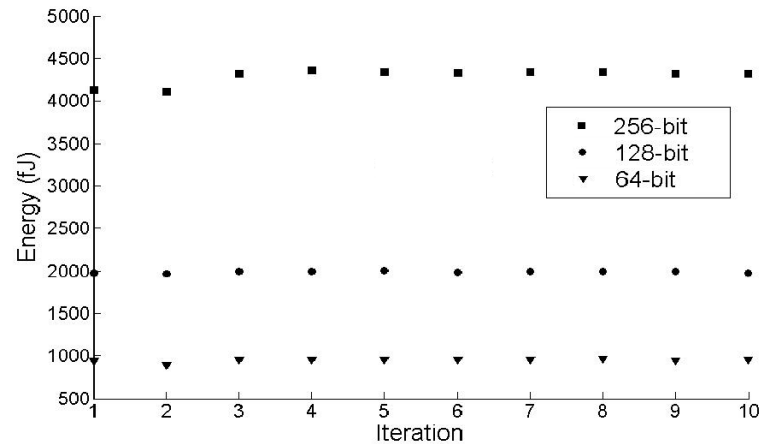
- electrical shielding
- detectors, estimators, decoupling
- use uniform computation durations and power consumption
- use detection/correction codes (for fault injection attacks)
- provide a random behavior (algorithms, representation, operations...)
- add noise (e.g. masking, useless instructions/computations)
- circuit reconfiguration (algorithms, block location, representation of values...)
- ...

Circuit-Level Protections for Arithmetic Operators





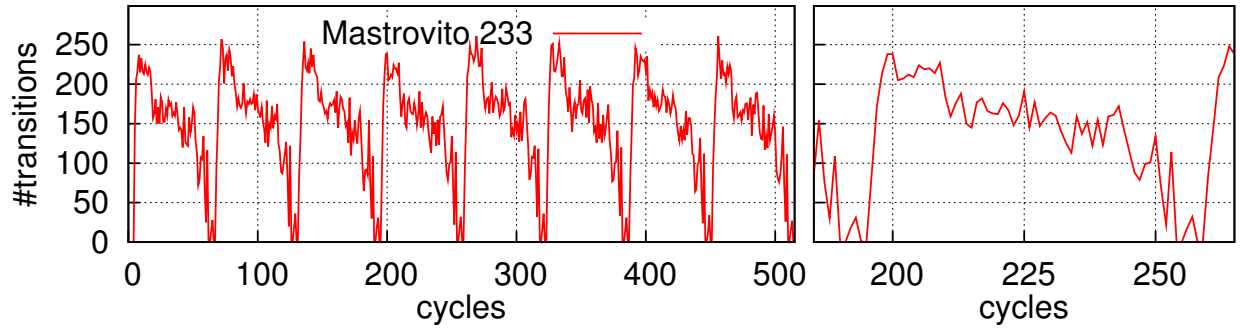




References: [6] and [7]

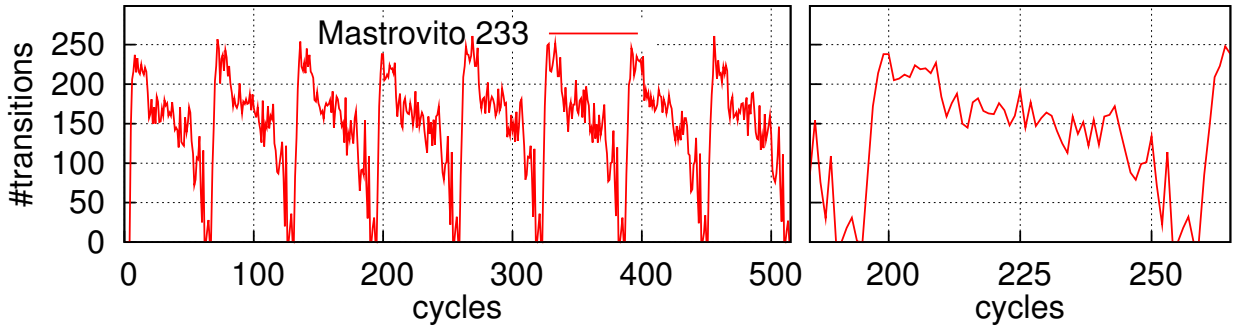
Protection of Arithmetic Operators

Unprotected



Protection of Arithmetic Operators

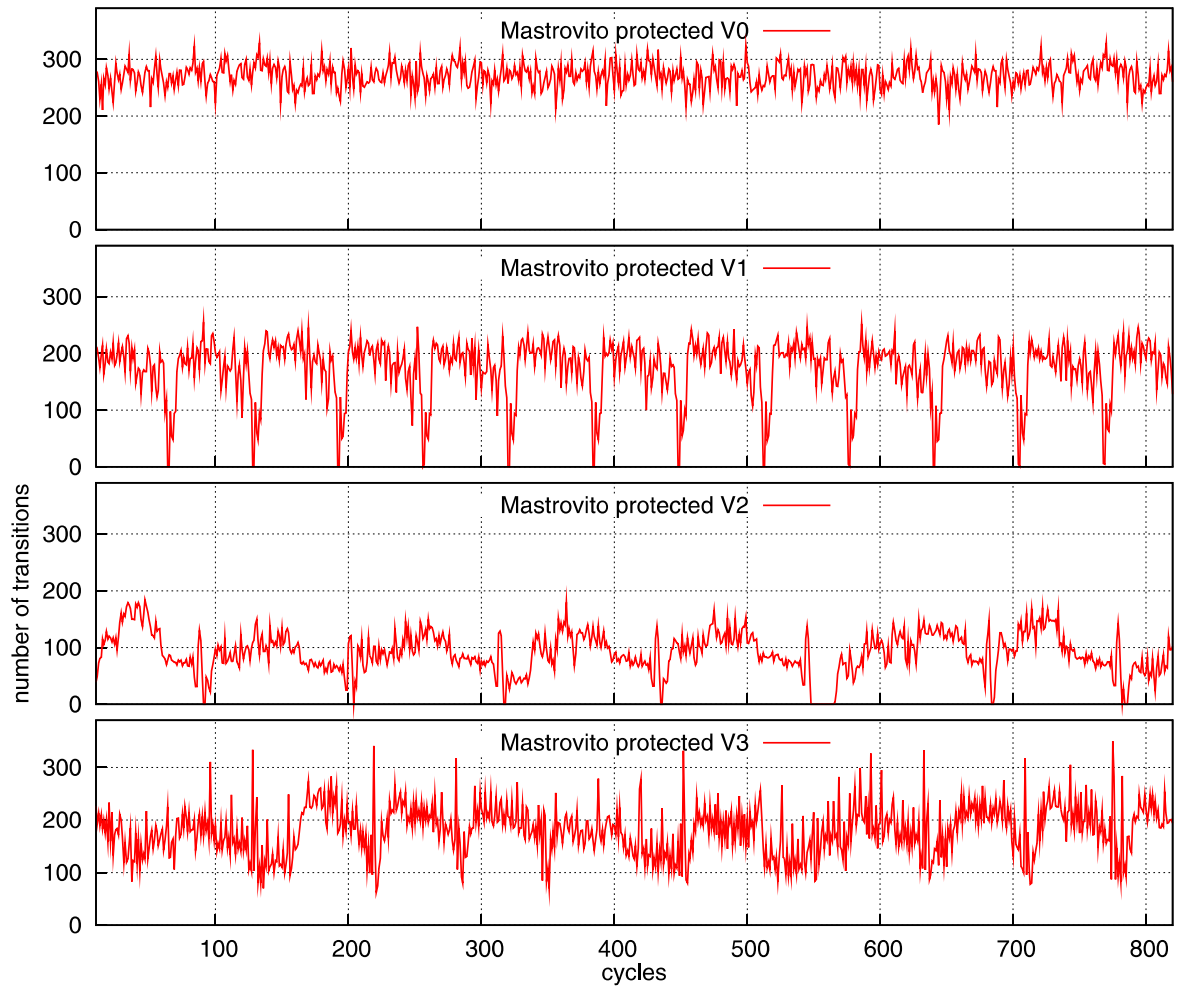
Unprotected



Protected

Overhead:
Area/time $< 10\,\%$

References:
PhD D. Pamula [10]
Articles: [12], [11]



Exotic Representations of Numbers

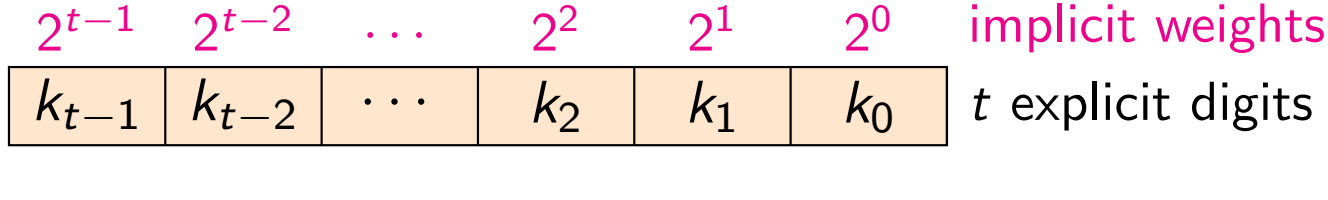
Standard radix-2 representation:

$$k = \sum_{i=0}^{t-1} k_i 2^i = \boxed{k_{t-1}}\boxed{k_{t-2}}\boxed{\cdots}\boxed{k_2}\boxed{k_1}\boxed{k_0}$$ t explicit digits

Exotic Representations of Numbers

Standard radix-2 representation:

$$k = \sum_{i=0}^{t-1} k_i 2^i =$$

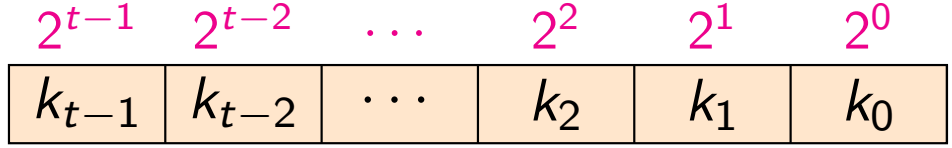


Digits: $k_i \in \{0, 1\}$

Exotic Representations of Numbers

Standard radix-2 representation:

$$k = \sum_{i=0}^{t-1} k_i 2^i =$$



Digits: $k_i \in \{0, 1\}$

Double-Base Number System (DBNS):

$$k = \sum_{j=0}^{n-1} k_j 2^{a_j} 3^{b_j} =$$

Exotic Representations of Numbers

Standard radix-2 representation:

$$k = \sum_{i=0}^{t-1} k_i 2^i =$$

2^{t-1}	2^{t-2}	$\cdots$	2^2	2^1	2^0	implicit weights
k_{t-1}	k_{t-2}	$\cdots$	k_2	k_1	k_0	t explicit digits

Digits: $k_i \in \{0, 1\}$

Double-Base Number System (DBNS):

$$k = \sum_{j=0}^{n-1} k_j 2^{a_j} 3^{b_j} =$$

k_{n-1}	$\cdots$	k_1	k_0	n $(2,3)-$terms
a_{n-1}	$\cdots$	a_1	a_0	explicit "digits"
b_{n-1}	$\cdots$	b_1	b_0	explicit ranks/weights

$a_j, b_j \in \mathbb{N}$, $k_j \in \{1\}$ or $k_j \in \{-1, 1\}$, size $n \approx \log t$

Exotic Representations of Numbers

Standard radix-2 representation:

$$k = \sum_{i=0}^{t-1} k_i 2^i = \begin{array}{|c|c|c|c|c|c|}\hline k_{t-1} & k_{t-2} & \cdots & k_2 & k_1 & k_0 \\ \hline \end{array}$$

2^{t-1} 2^{t-2} $\cdots$ 2^2 2^1 2^0 implicit weights

t explicit digits

Digits: $k_i \in \{0, 1\}$

Double-Base Number System (DBNS):

$$k = \sum_{j=0}^{n-1} k_j 2^{a_j} 3^{b_j} = \begin{array}{|c|c|c|c|}\hline k_{n-1} & \cdots & k_1 & k_0 \\ a_{n-1} & \cdots & a_1 & a_0 \\ b_{n-1} & \cdots & b_1 & b_0 \\ \hline \end{array}$$

n $(2,3)-$terms

explicit "digits"

explicit ranks/weights

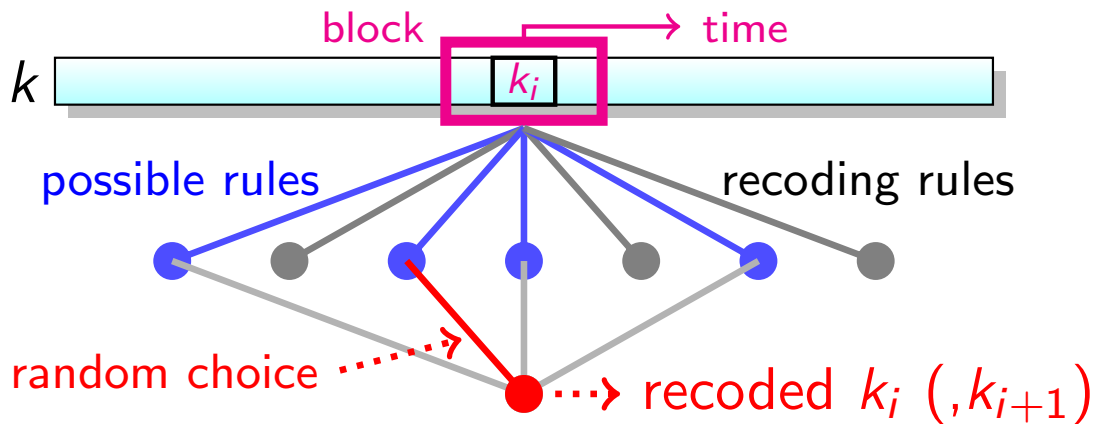
$a_j, b_j \in \mathbb{N}$, $k_j \in \{1\}$ or $k_j \in \{-1, 1\}$, size $n \approx \log t$

DBNS is a very redundant and sparse representation: $1701 = (11010100101)_2$

$$\begin{array}{rcrcrcl} 1701 & = & 243 + 1458 & = & 2^0 3^5 + 2^1 3^6 & = & (1,0,5),(1,1,6) \\ & = & 1728 - 27 & = & 2^6 3^3 - 2^0 3^3 & = & (1,6,3),(-1,0,3) \\ & = & 729 + 972 & = & 2^0 3^6 + 2^2 3^5 & = & (1,0,6),(1,2,5) \\ & \cdots & & & & & \end{array}$$

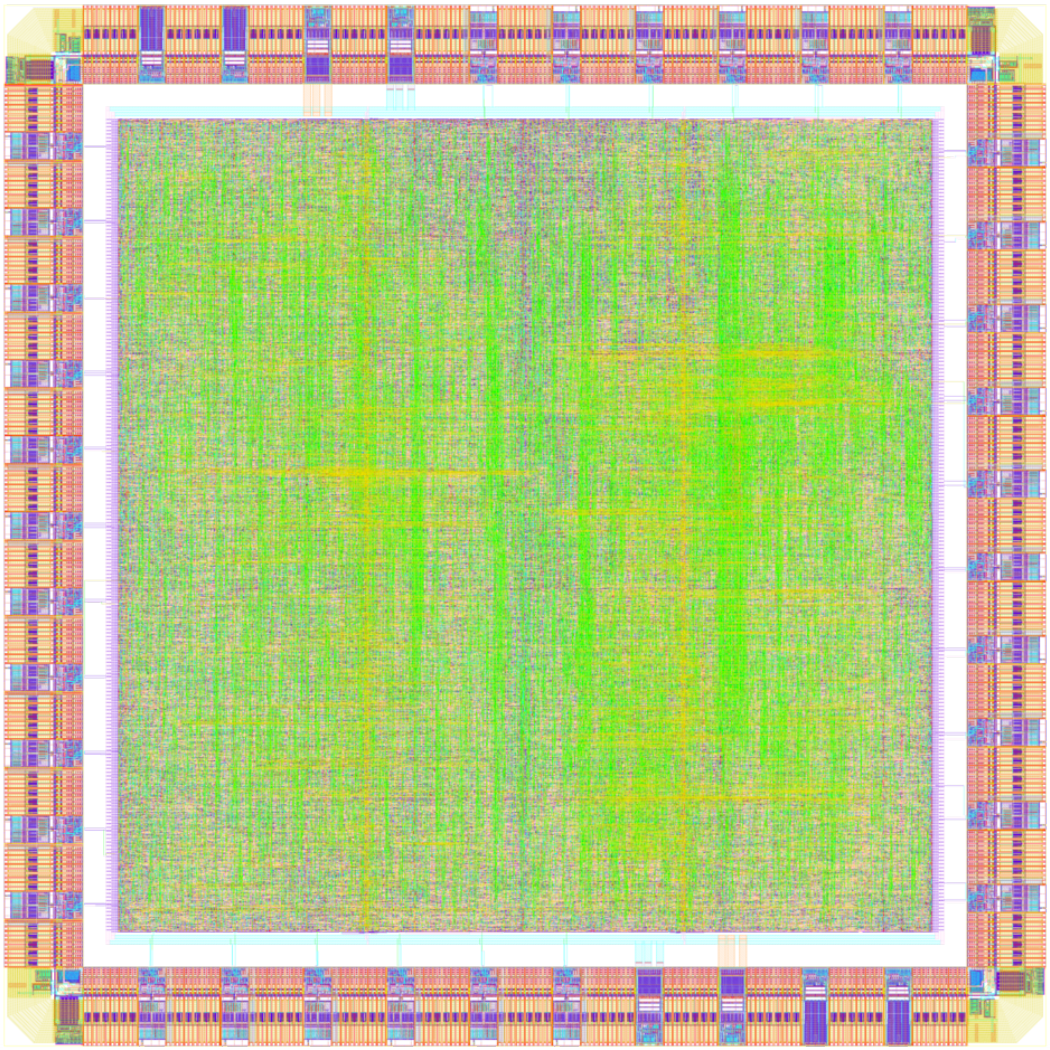
Randomized DBNS Recoding

- On-the-fly random recoding of secret values (e.g. scalars in ECC)
- In a limited window, randomly select one of recoding (if possible):
  - $1 + 2 \leftrightarrows 3$
  - $1 + 3 \leftrightarrows 2^2$
  - $1 + 2^3 \leftrightarrows 3^2$
- DBNS is redundant $\Rightarrow$ security $\nearrow$
- DBNS is sparse $\Rightarrow$ 20–30 % speed $\nearrow$
- Reference: [4] for DBNS, [5] for MBNS

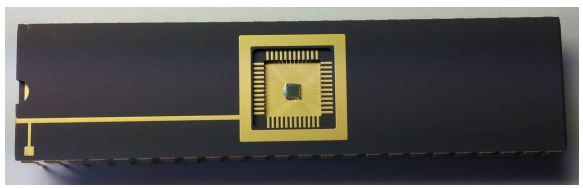


Prototyping in Real Circuits

Processor for Elliptic Curve Cryptography designed in the PAVOIS ANR project (2012–2016)



$\mathbb{F}_p$ 256 bits (gen.)
65 nm CMOS
1.5 mm^2



Conclusion and References

"Conclusion"

- Physical attacks are serious threats
- Attacks are more and more efficient (many variants)
- Security analysis is mandatory at all levels (specification, algorithm, operation, implementation, test, life cycle)
- Security = trade-off between performances, robustness and cost
- Security = *func*(secret value, attacker capabilities)
- Security = computer science + microelectronics + mathematics

References I

[1] J. Balasch, B. Gierlichs, and I. Verbauwhede.
An in-depth and black-box characterization of the effects of clock glitches on 8-bit MCUs.
In *Proc. 8th International Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pages 105–114, Nara, Japan, September 2011. IEEE.

[2] K. Bigou and A. Tisserand.
Binary-ternary plus-minus modular inversion in RNS.
IEEE Transactions on Computers, 65(11):3495–3501, November 2016.

[3] K. Bigou and A. Tisserand.
Hybrid position-residues number system.
In J. Hormigo, S. Oberman, and N. Revol, editors, *Proc. 23rd Symposium on Computer Arithmetic (ARITH)*, pages 126–133, Santa Clara, CA, U.S.A, July 2016. IEEE Computer Society.

[4] T. Chabrier, D. Pamula, and A. Tisserand.
Hardware implementation of DBNS recoding for ECC processor.
In *Proc. 44rd Asilomar Conference on Signals, Systems and Computers*, pages 1129–1133, Pacific Grove, California, U.S.A., November 2010. IEEE.

[5] T. Chabrier and A. Tisserand.
On-the-fly multi-base recoding for ECC scalar multiplication without pre-computations.
In A. Nannarelli, P.-M. Seidel, and P. T. P. Tang, editors, *Proc. 21st Symposium on Computer Arithmetic (ARITH)*, pages 219–228, Austin, TX, U.S.A, April 2013. IEEE Computer Society.

[6] J. Chen, A. Tisserand, E. M. Popovici, and S. Cotofana.
Robust sub-powered asynchronous logic.
In J. Becker and M. R. Adrover, editors, *Proc. 24th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, pages 1–7, Palma de Mallorca, Spain, September 2014. IEEE.

[7] J. Chen, A. Tisserand, E. M. Popovici, and S. Cotofana.
Asynchronous charge sharing power consistent Montgomery multiplier.
In J. Sparso and E Yahya, editors, *Proc. 21st IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, pages 132–138, Mountain View, California, USA, May 2015.

References II

[8] G. Gallin and A. Tisserand.
Hyper-threaded multiplier for HECC.
In *Proc. 51st Asilomar Conference on Signals, Systems and Computers*, pages 447–451, Pacific Grove, CA, USA, October 2017. IEEE.

[9] P. C. Kocher, J. Jaffe, and B. Jun.
Differential power analysis.
In *Proc. Advances in Cryptology (CRYPTO)*, volume 1666 of *LNCS*, pages 388–397. Springer, August 1999.

[10] D. Pamula.
Arithmetic Operators on GF(2^m) for Cryptographic Applications: Performance - Power Consumption - Security Tradeoffs.
Phd thesis, University of Rennes 1 and Silesian University of Technology, December 2012.

[11] D. Pamula and A. Tisserand.
GF(2^m) finite-field multipliers with reduced activity variations.
In *4th International Workshop on the Arithmetic of Finite Fields*, volume 7369 of *LNCS*, pages 152–167, Bochum, Germany, July 2012. Springer.

[12] D. Pamula and A. Tisserand.
Fast and secure finite field multipliers.
In *Proc. 18th Euromicro Conference on Digital System Design (DSD)*, pages 653–660, Madeira, Portugal, August 2015.

[13] A. Tisserand.
Hardware accelerators for ECC and HECC.
In *19th Workshop on Elliptic Curve Cryptography (ECC)*, Bordeaux, France, September 2015.
Invited talk.

The end, questions ?

Contact:

- mailto:arnaud.tisserand@univ-ubs.fr
- http://www-labsticc.univ-ubs.fr/~tisseran
- CNRS, Lab-STICC Laboratory
  University South Brittany (UBS),
  Centre de recherche C. Huygens, rue St Maudé, BP 92116,
  56321 Lorient cedex, France

Thank you