Bricks and Tools for Secure Hardware Implementations

Francesco Regazzoni

Why Electronic Design Automation?

"Surely the purpose of science is to ease human hardship"

Galileo, Bertolt Brecht

- Handle the complexity
- Time to market
- Design optimization

From G. De Micheli, *Synthesis and Optimization of Digital Circuits*, McGraw-Hill Higher Education, 1994.

Why Electronic Design Automation for security?

- Security is very often considered at later stages of design
- Cost and Time to Market
- Possible Security pitfalls

EXTRA CONSTRAINT

Use as much as possible "standard" EDA commodities!

Outline

- Logic Synthesis (Secure)
- Design Flow for secure ISE
- Quick note on Software

Simplified Hardware Design Flow (ASIC)

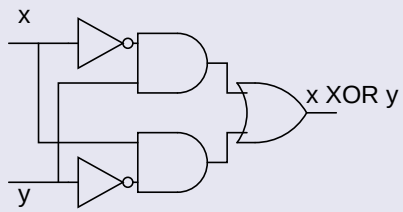
Algorithm Design

C, Matlab, VHDL

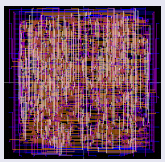
RTL (Architecture) Design

Synthesizable HDL

Gate



Layout



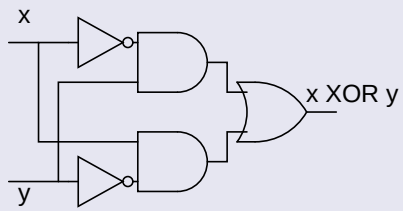
Let's focus on Synthesis

RTL (Architecture) Design

Synthesizable HDL

Logic Synthesis

Gate Level



A bit of history

- Few algorithms and tools existed in the 70's
- First prototype synthesis tools in the early 80's
- First logic synthesis companies in the late 80's

Design Automation Conference (DAC) turned 51 years last week: happy birthday!

Definitions

Logic Synthesis

is the manipulation of logic specifications to create logic models as an interconnection of logic primitives

Logic Synthesis

determines the **gate level** structure of a circuit

From G. De Micheli, *Synthesis and Optimization of Digital Circuits*, McGraw-Hill Higher Education, 1994.

Logic Synthesis Input and Output

INPUT:

- HDL Description
- Technological Library (area, timing, power)
- Synthetic Library (multipliers...)
- Constraints

OUTPUT:

- Gate Level Netlist
- Estimation of area, timing, power (!)
- Timing constraints

Typical Logic Synthesis Steps

one

State Minimization

two

State Encoding

three

Combinatorial Logic Minimization

four

Technology Mapping

Is it sufficient for Security?

Paul Kocher, Joshua Jaffe, and Benjamin Jun, "**Differential Power Analysis**", in Proceedings of *Advances in Cryptology-CRYPTO'99*, Santa Barbara, California, USA, August 15-19, 1999. (Cited by 4128)

Approach One

INPUT:

- HDL Description
- Technological Library (area, timing, power)
- Synthetic Library (multipliers...)
- Constraints

OUTPUT:

- DPA resistant Gate Level Netlist
- Estimation of area, timing, power (!)
- Timing constraints

Approach Two

INPUT:

- HDL Description
- Technological Library (area, timing, power)
- Synthetic Library (multipliers...)
- Constraints (limit the gates)

OUTPUT:

- Gate Level Netlist

"Cell Substitution":

- Replace cells
- Reload in the tool for correct area and timing constraints

K. Tiri and I. Verbauwhede, *A digital design flow for secure integrated circuits*, IEEE TCAD, 2006

Careful!

As a example of design for security, we have focused on synthesis, and we have detailed two possible approaches for synthesis of DPA resistant circuits

However

- Synthesis is only **one** step of the whole design flow
- Security should be considered in **every steps** of the of the design flow
- Doing DPA resistant synthesis alone is not sufficient!

Outline

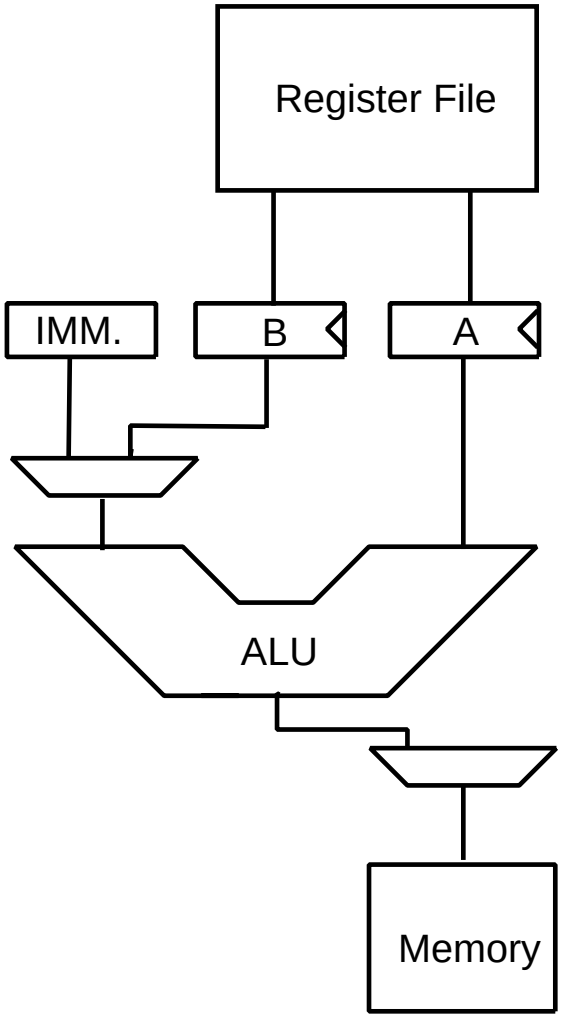
- Logic Synthesis (Secure)
- Design Flow for secure ISE
- Quick note on Software

Protect PRESENT with secure hardware

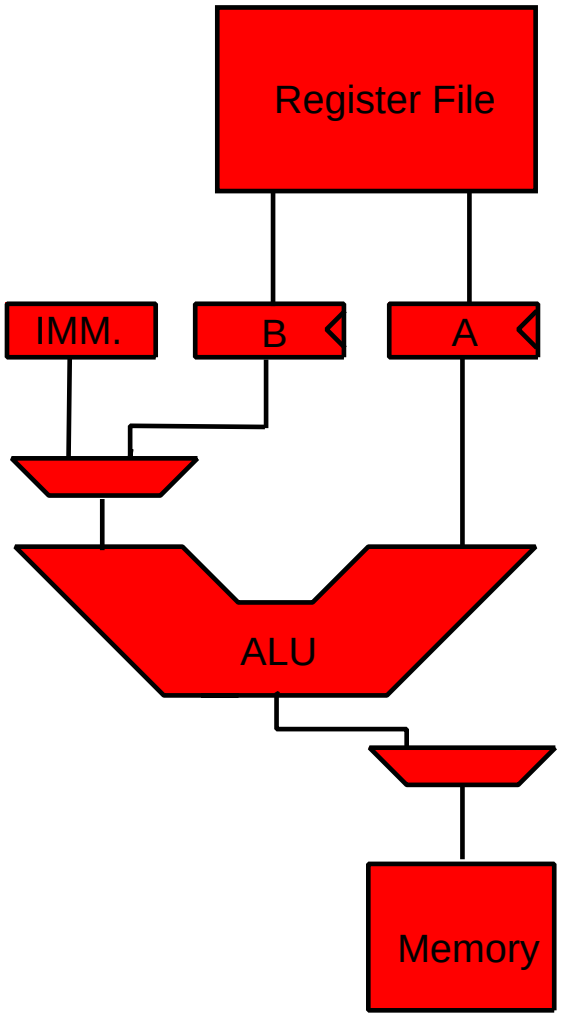
- Lightweight block cipher
- **4 bit** S-box
- *addRoundKey*, *sBoxLayer*

```
// Calculate S-box (plaintext XOR key)
int PRESENT(int plaintext, int key) {
1 int result = 0; // initialize the result
2 plaintext = plaintext ^key; // perform the xor with the key
3 result = S[plaintext]; // perform the S-box
4 return result; }; // return the result
```

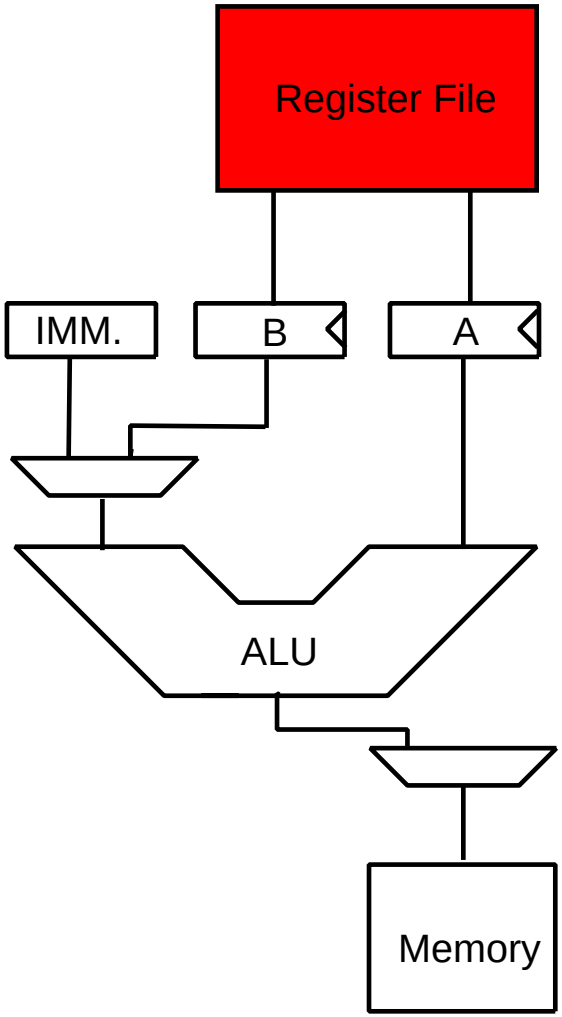
What can I do?



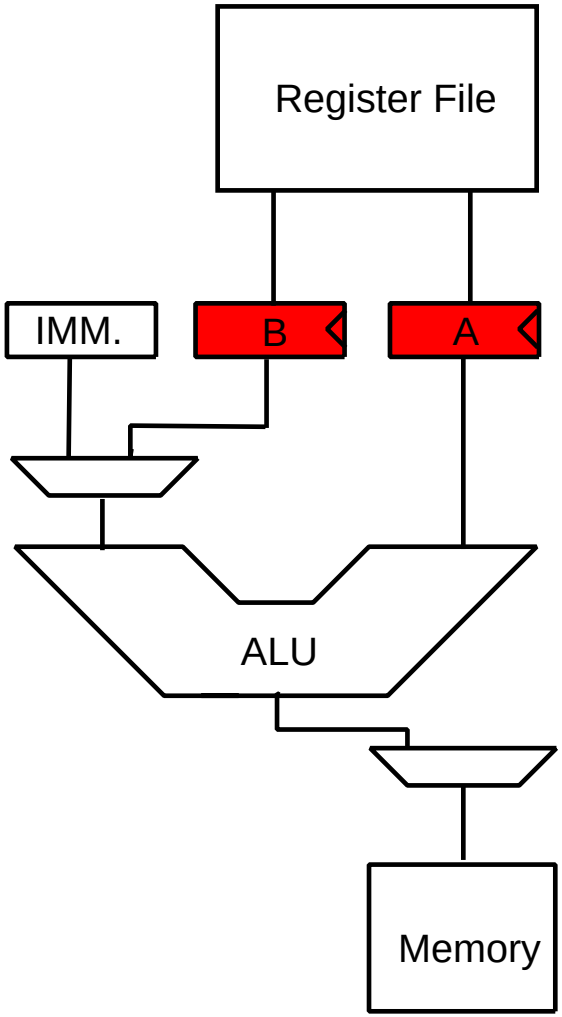
What can I do?



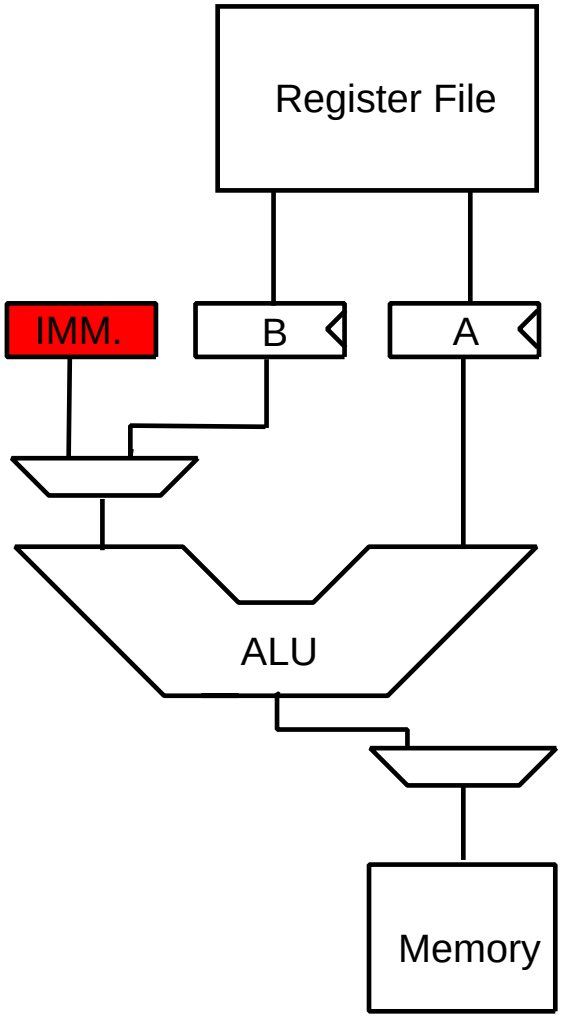
What can I do?



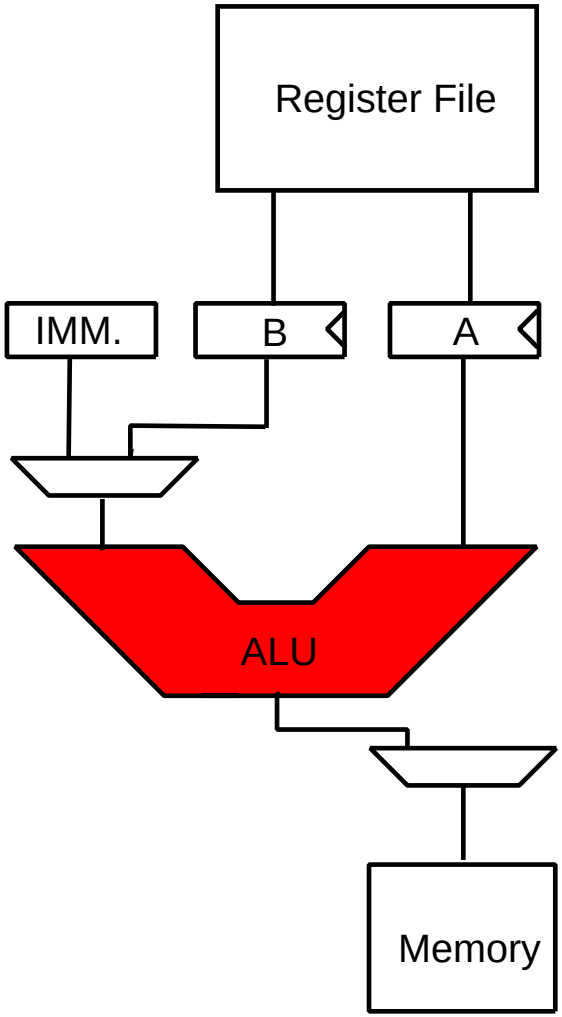
What can I do?



What can I do?



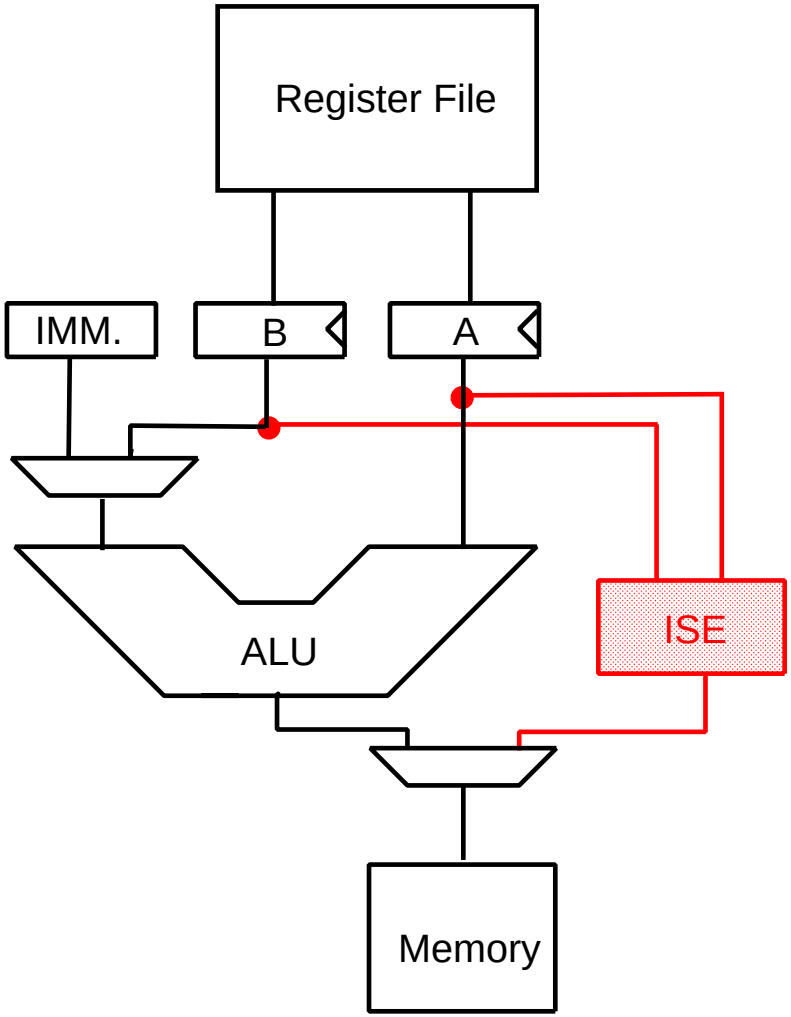
What can I do?



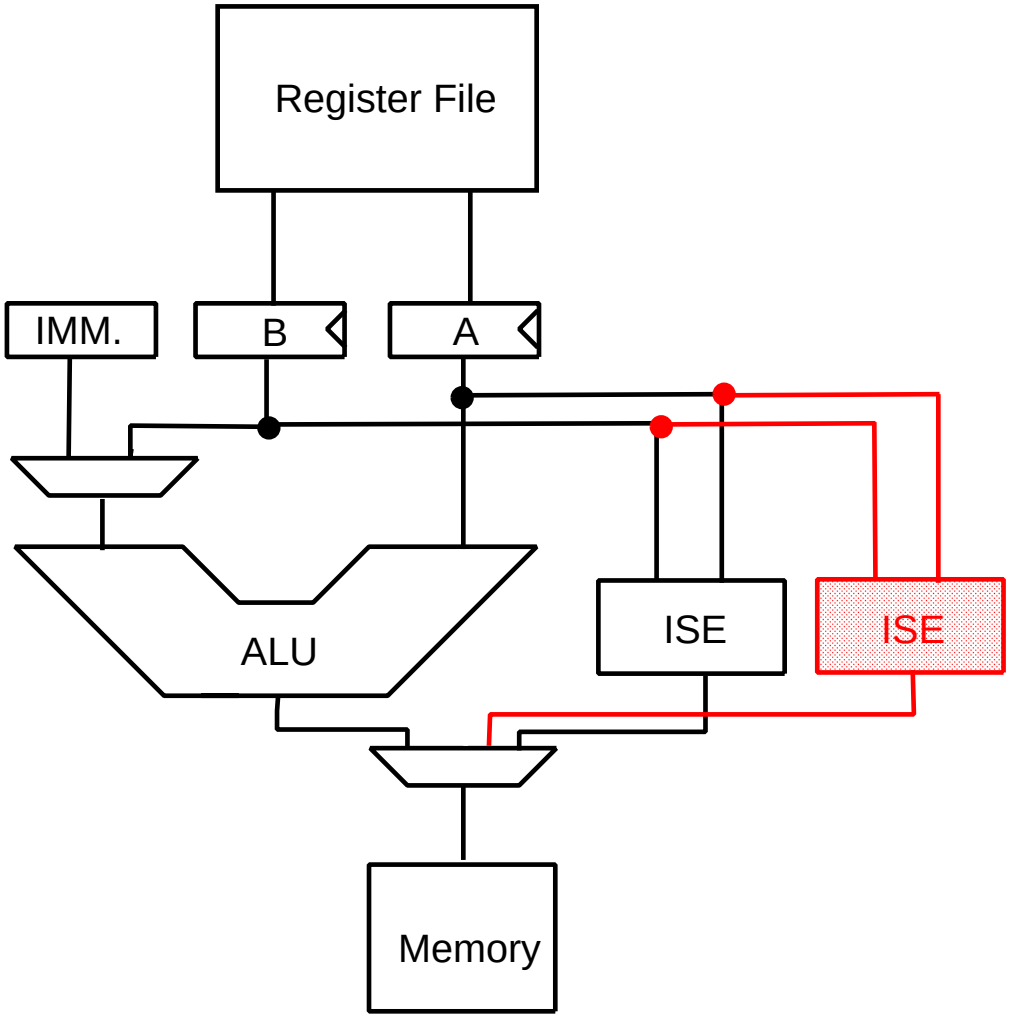
What can I do?

Something easier?

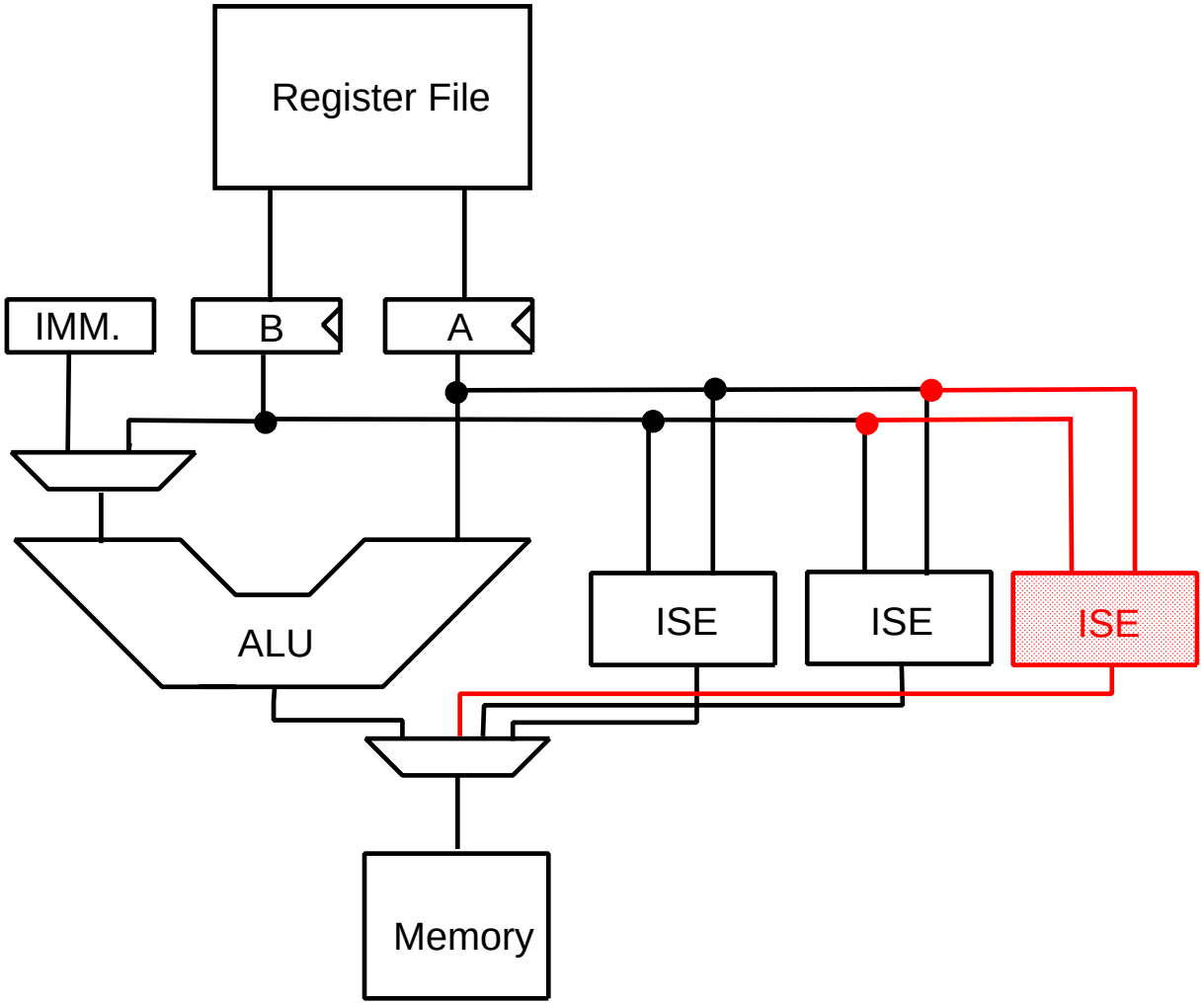
Protected / Non Protected Co-Design!



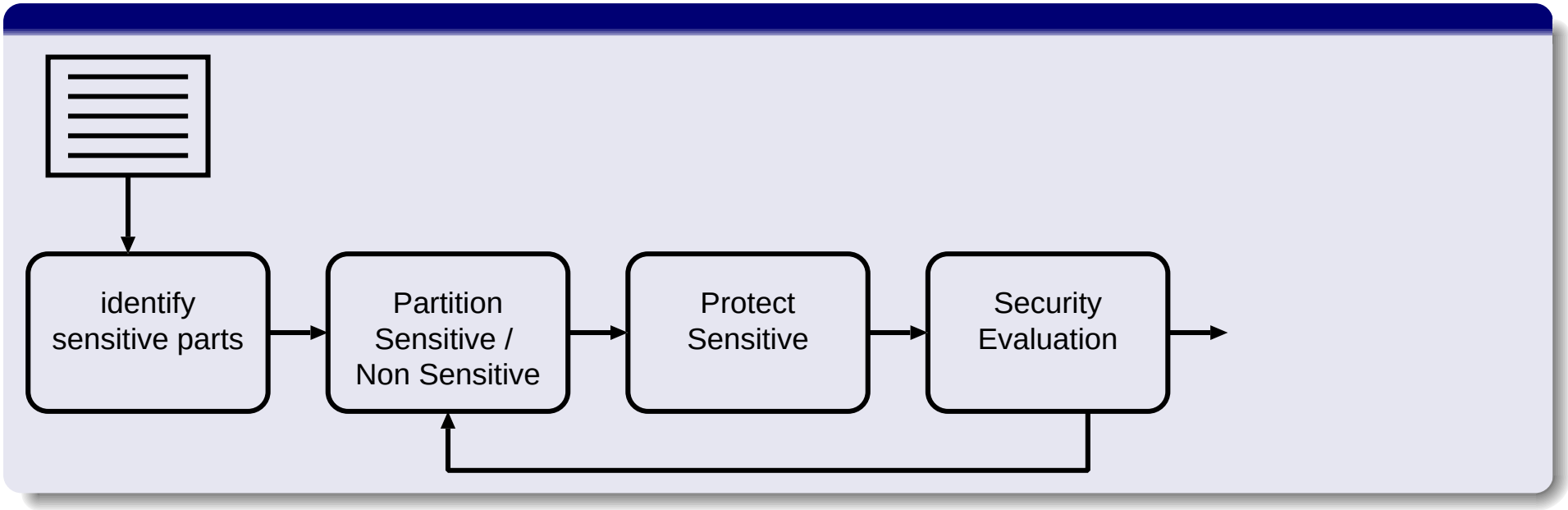
Protected / Non Protected Co-Design!



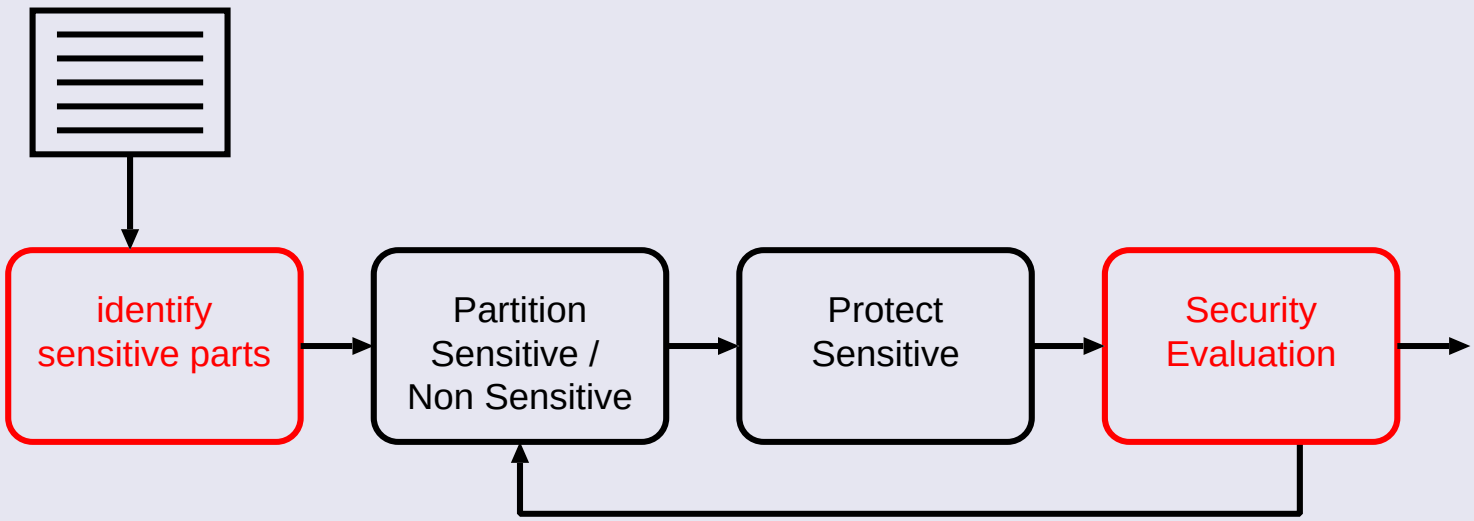
Protected / Non Protected Co-Design!



Automatic design of DPA resistant ISE

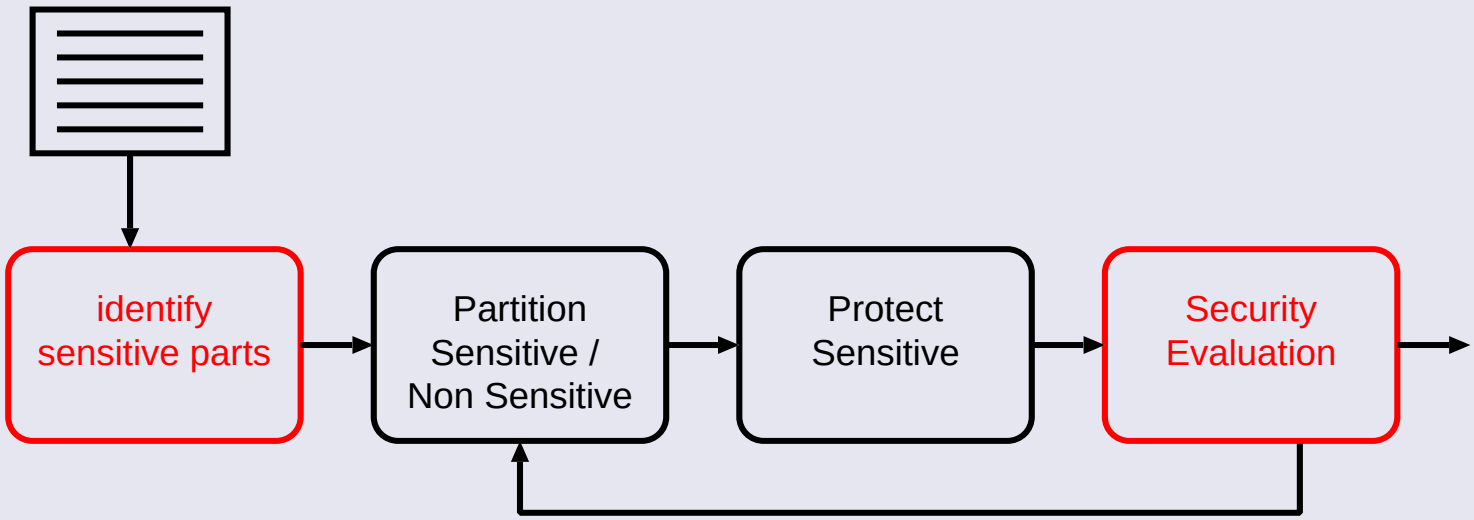


Needed "Basic Blocks"



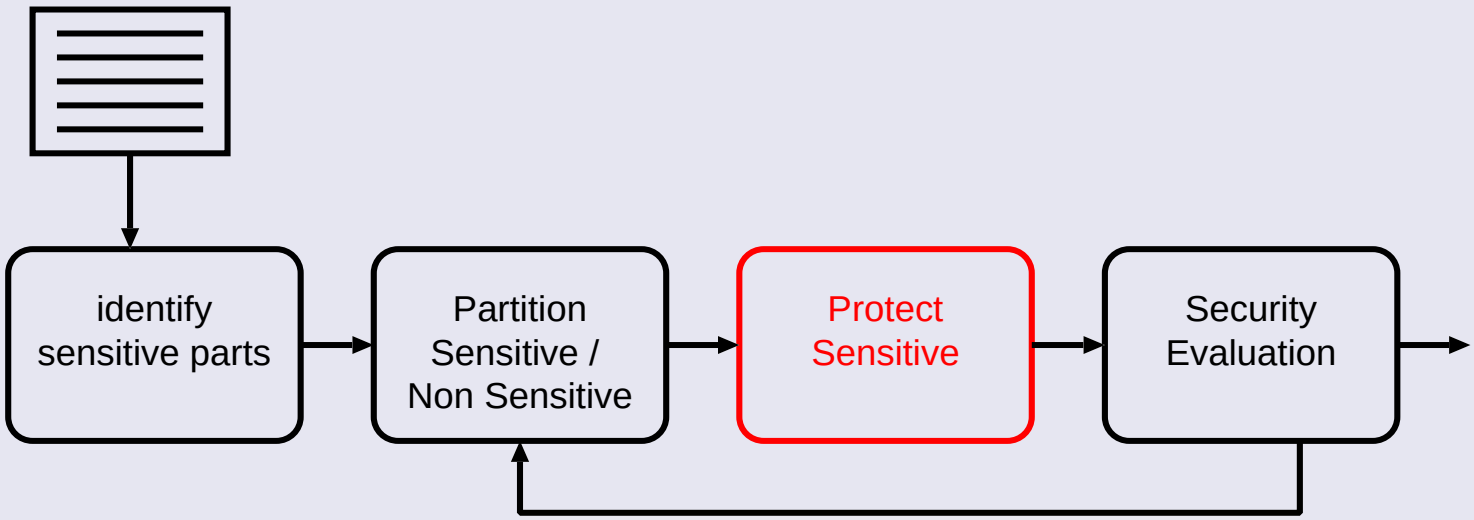
- Generate useful power traces?

Needed "Basic Blocks"



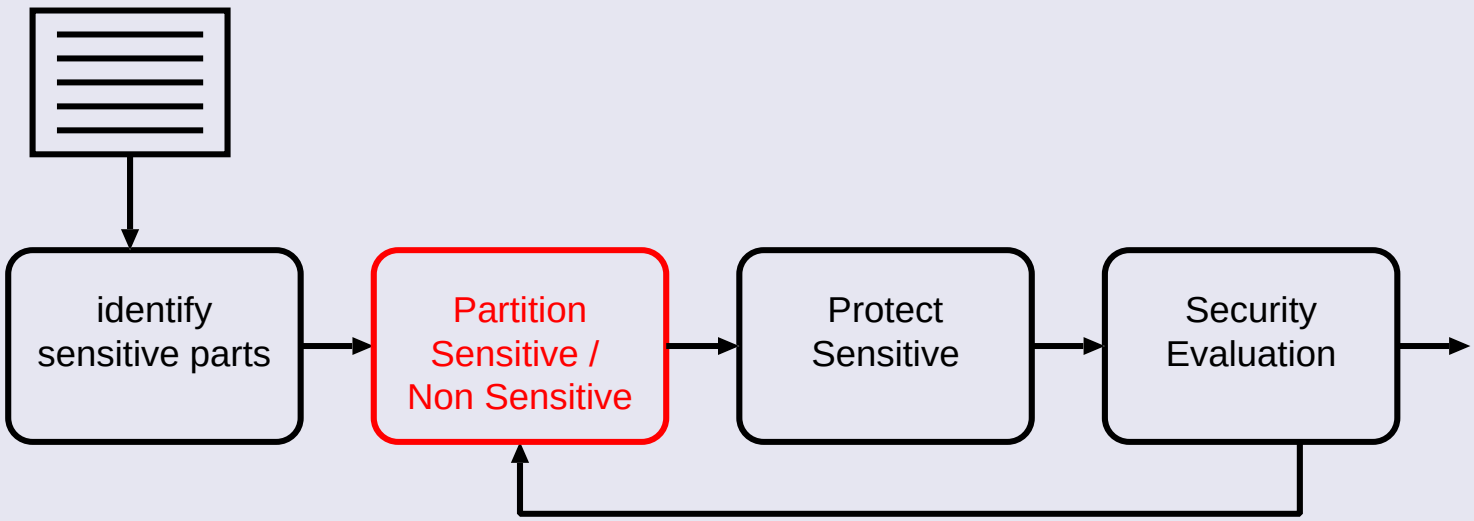
- Generate useful power traces?
- Measure the DPA resistance?

Needed “Basic Blocks”



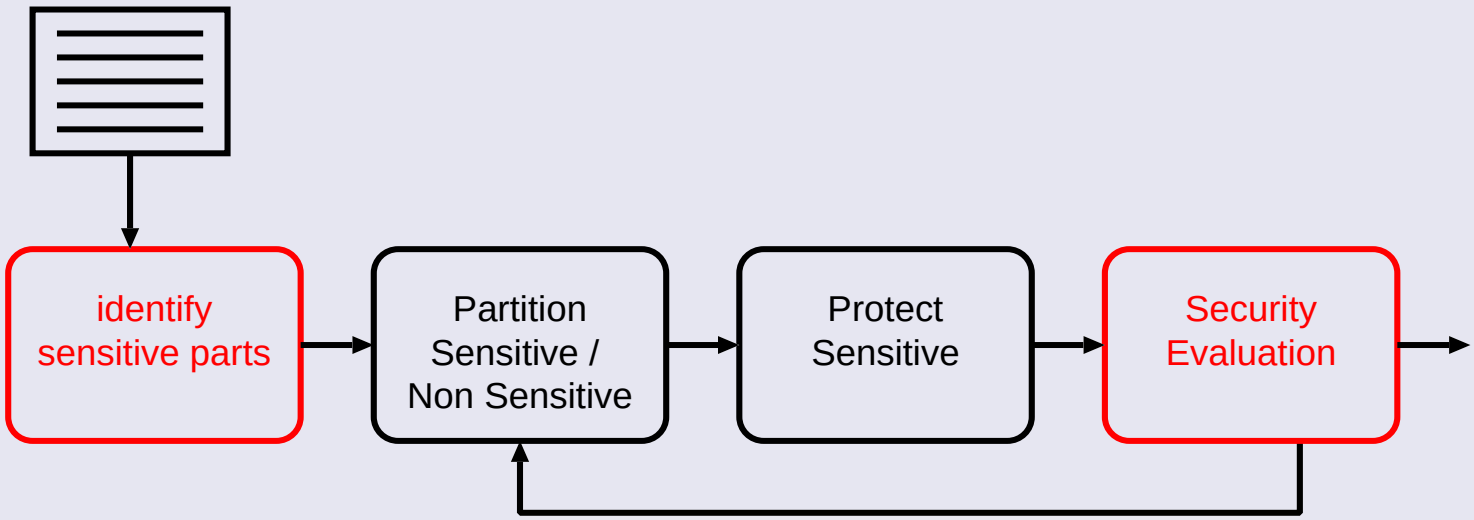
- Generate useful power traces?
- Measure the DPA resistance?
- Countermeasure and its design flow?

Needed "Basic Blocks"



- Generate useful power traces?
- Measure the DPA resistance?
- Countermeasure and its design flow?
- Partition the algorithm?

Needed "Basic Blocks"



- Generate useful power traces?
- Measure the DPA resistance?
- Countermeasure and its design flow?
- Partition the algorithm?

Fast Simulation SPICE level

Simulate Complex Design at SPICE level (whole processor)

Fast Simulation SPICE level

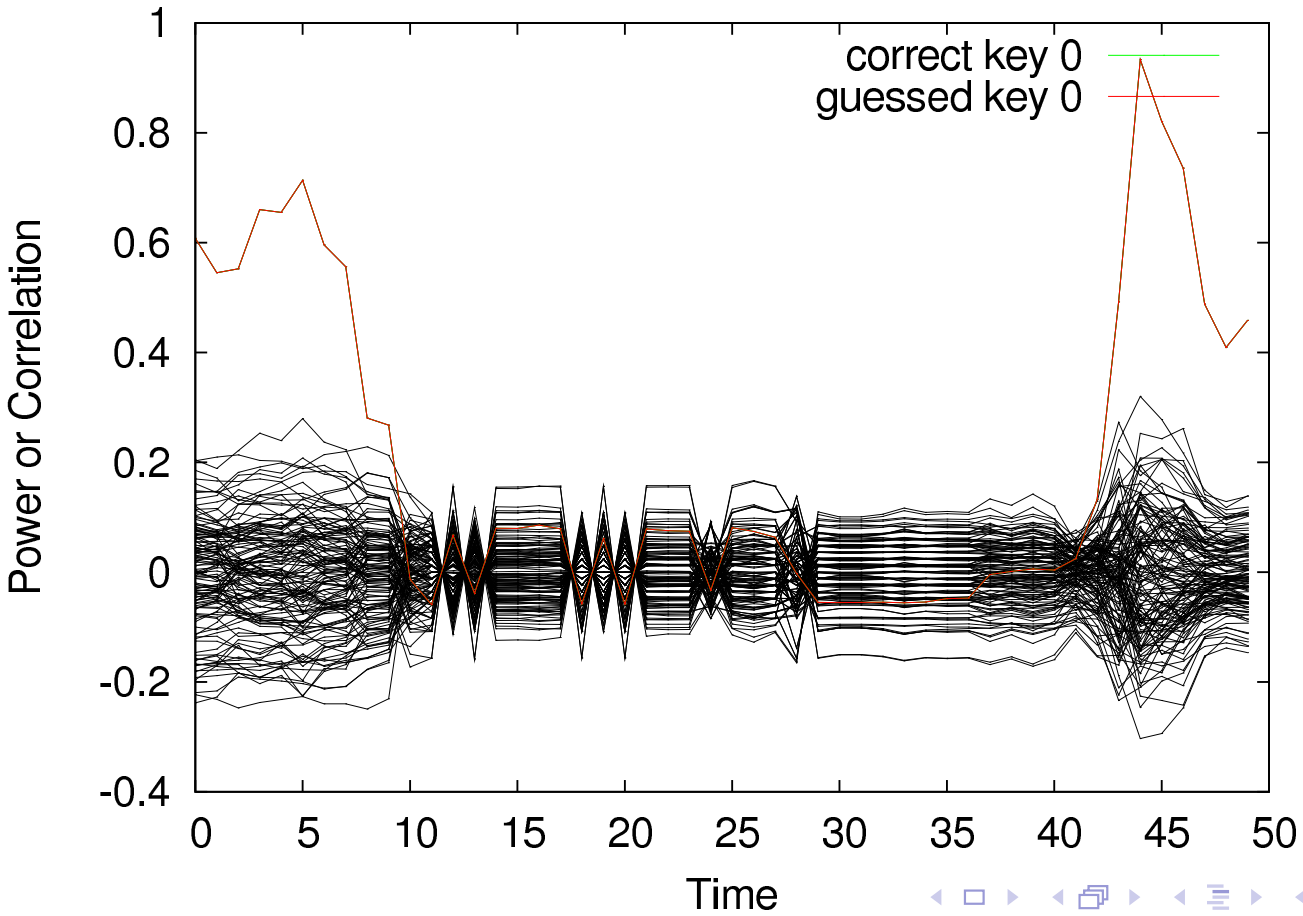
Simulate Complex Design at SPICE level (whole processor)

Simulated about 400 traces: approximately 20 hours!

Fast Simulation SPICE level

Simulate Complex Design at SPICE level (whole processor)

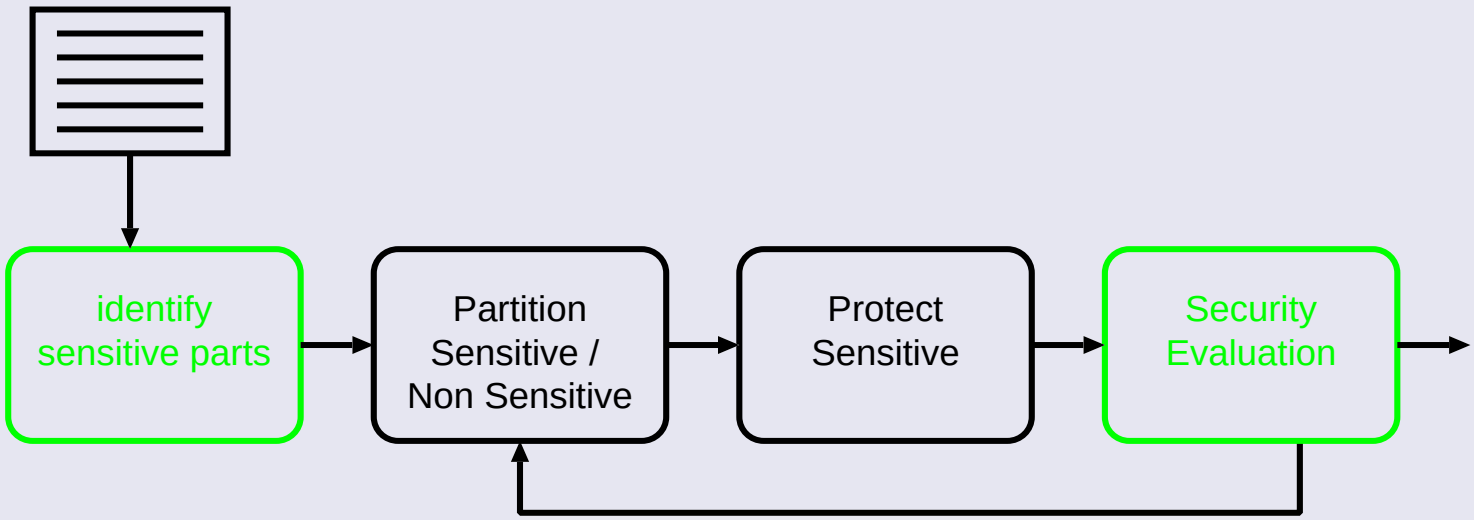
Simulated about 400 traces: approximately 20 hours!



Careful!

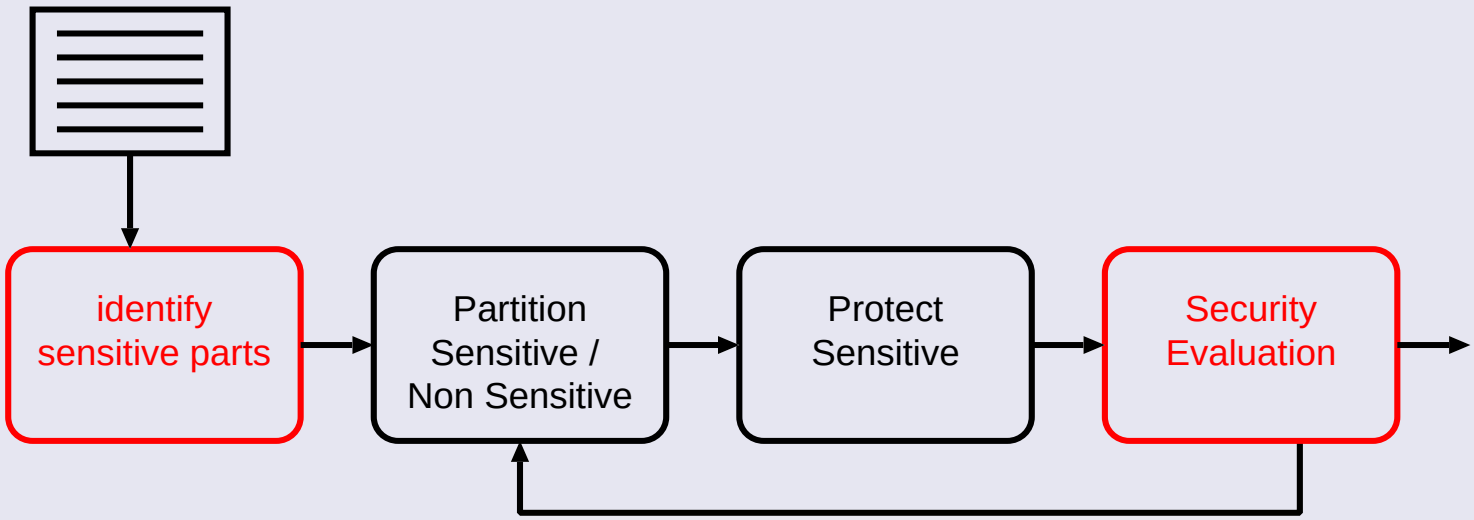
- Results obtained in simulations are often very different from the ones obtained from the real silicon
- Check and evaluate if and to which extent simulations results are matching the real measures

Needed "Basic Blocks"



- Generate useful power traces? ✓
- Measure the DPA resistance?
- Countermeasure and its design flow?
- Partition the algorithm?

Needed "Basic Blocks"



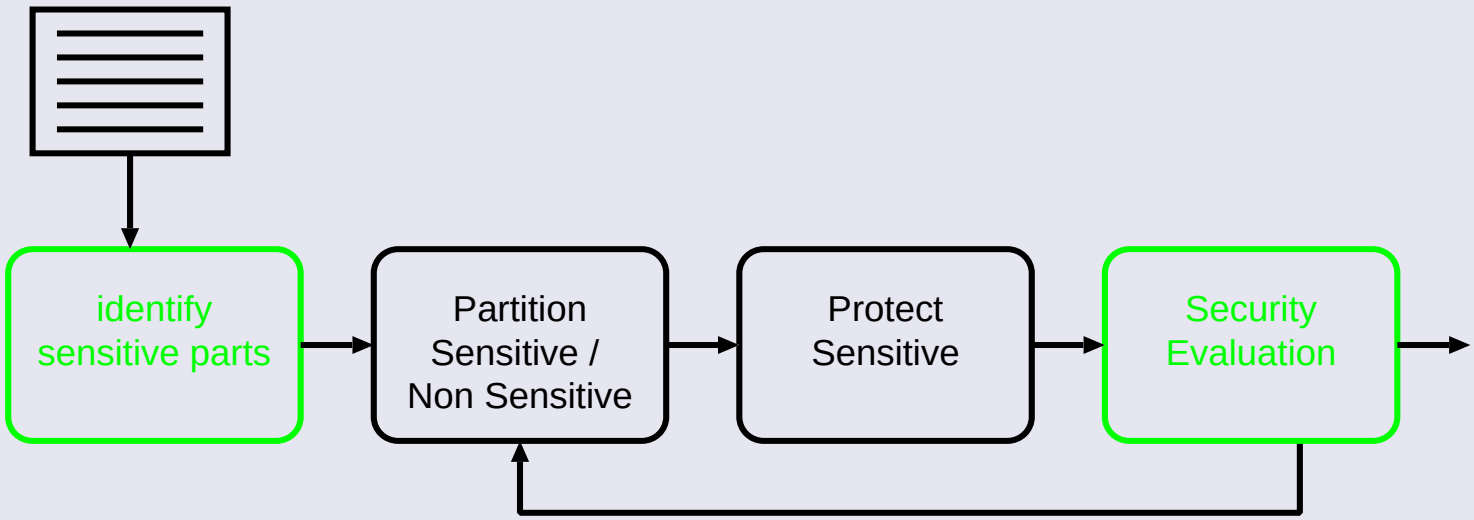
- Generate useful power traces? ✓
- Measure the DPA resistance?
- Countermeasure and its design flow?
- Partition the algorithm?

Information Theory Metric

$$\mathrm{H}[K|L] = -\sum_{k} \Pr[k] \cdot \sum_{x} \Pr[x] \int \Pr[l|k,x] \cdot \log_2 \Pr[k|l,x] \; dl.$$

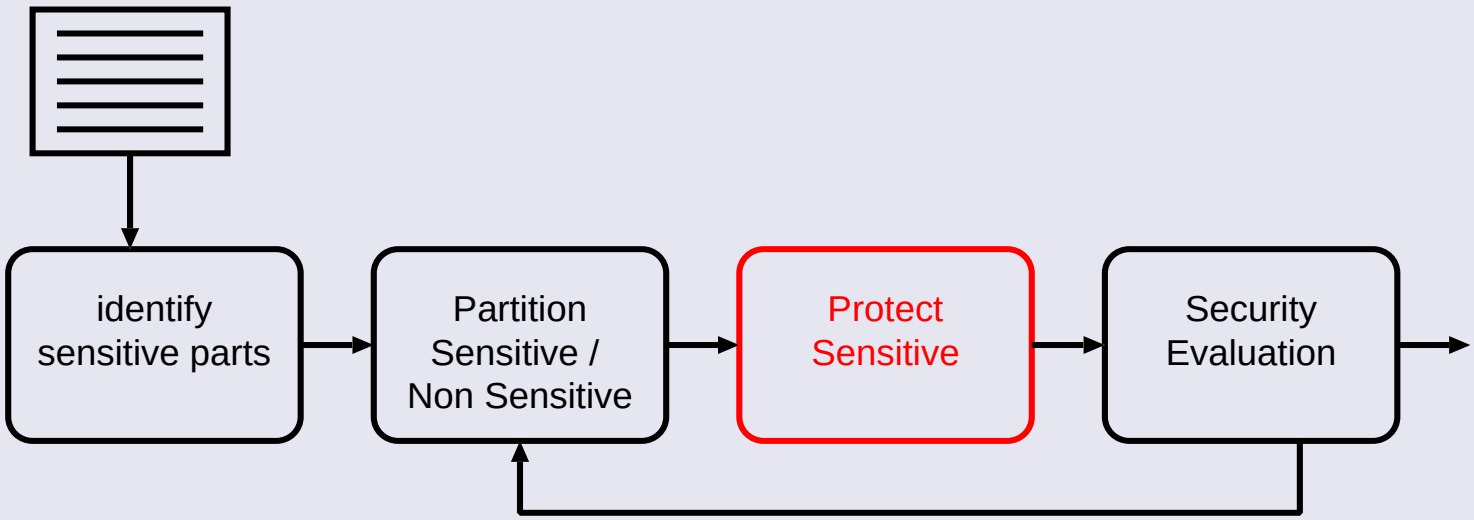
- Add white noise
- Reduce the dimension using compression
- Compute the mutual information

Needed "Basic Blocks"



- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow?
- Partition the algorithm?

Needed "Basic Blocks"

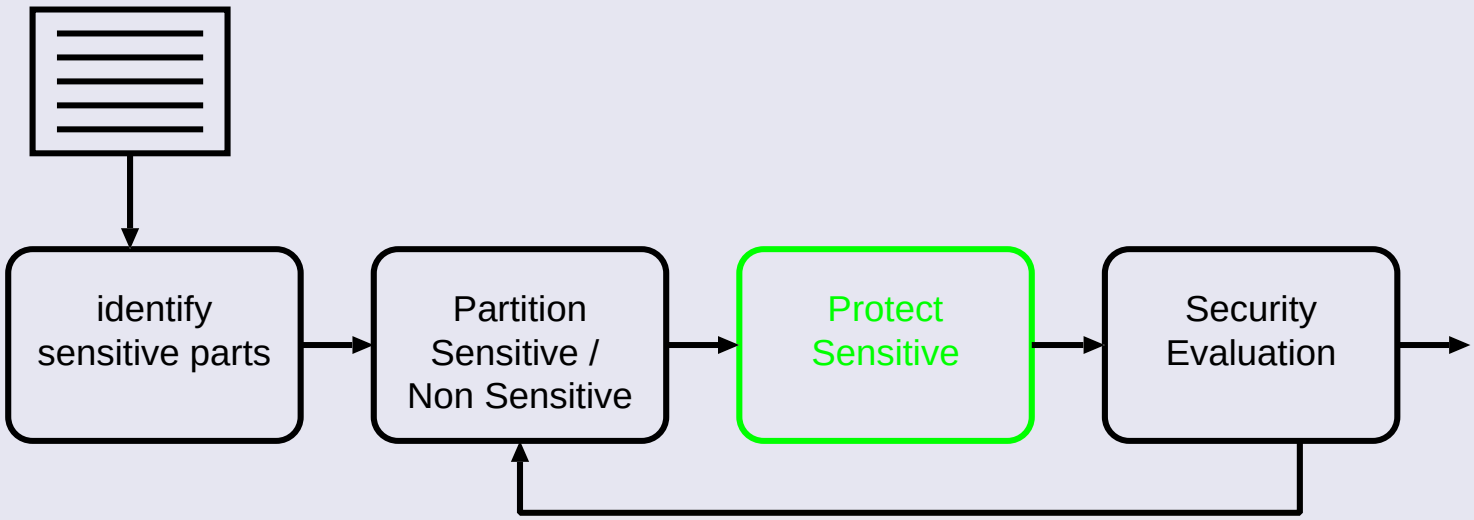


- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow?
- Partition the algorithm?

Protected Logic styles

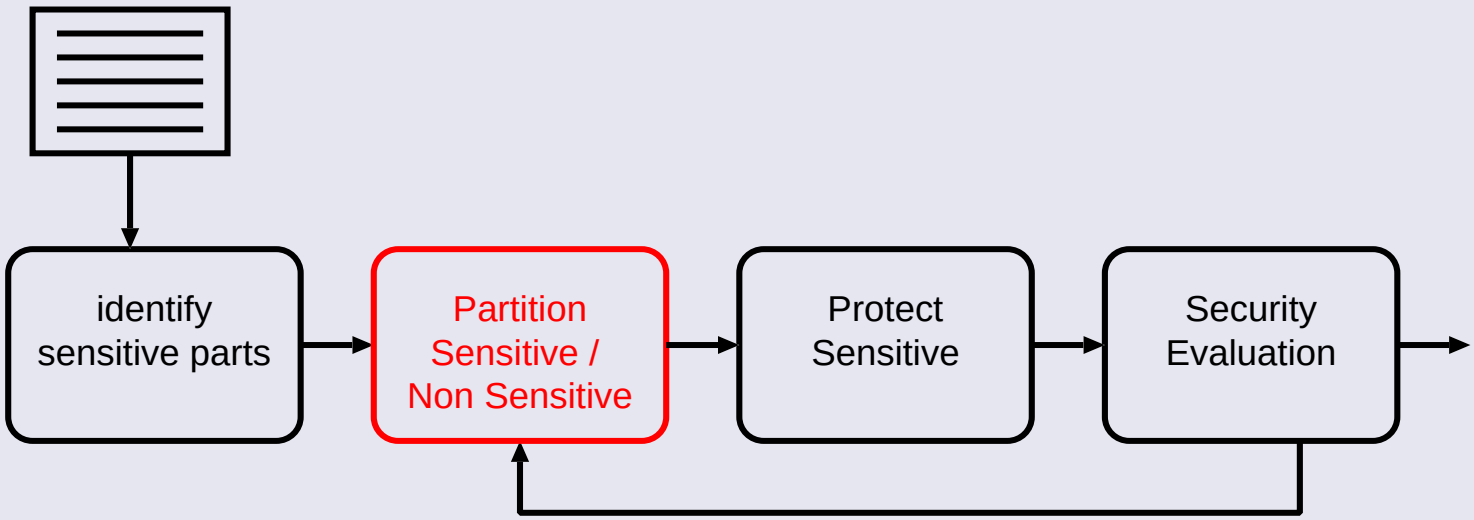
- **WDDL**
- **iMDPL**
- **MCML**
- **...**

Needed "Basic Blocks"



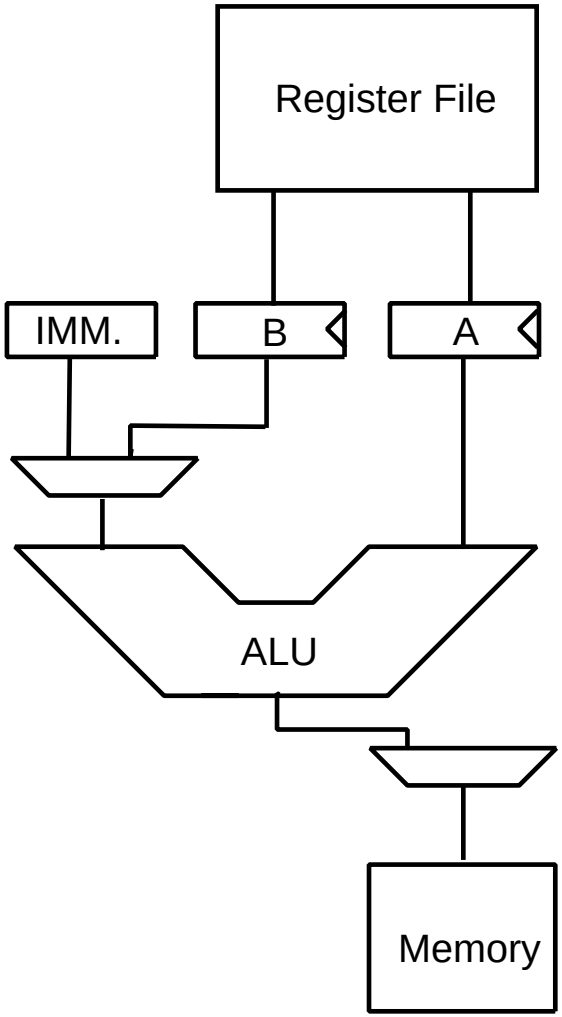
- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow? ✓
- Partition the algorithm?

Needed "Basic Blocks"

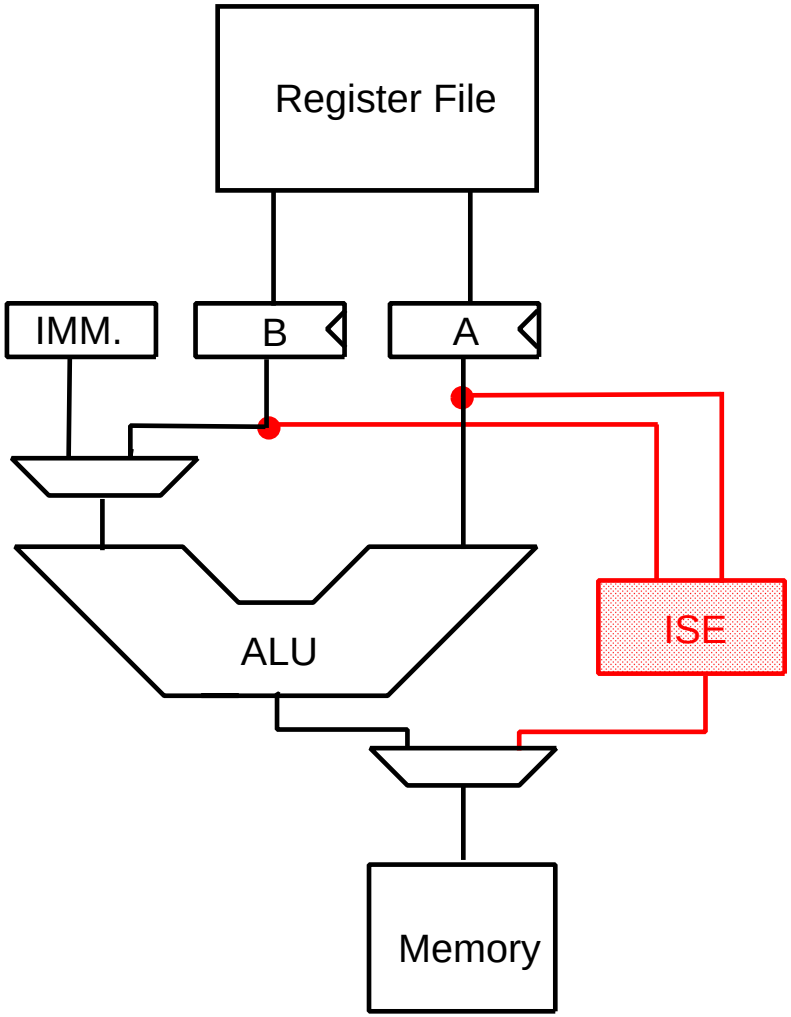


- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow? ✓
- Partition the algorithm?

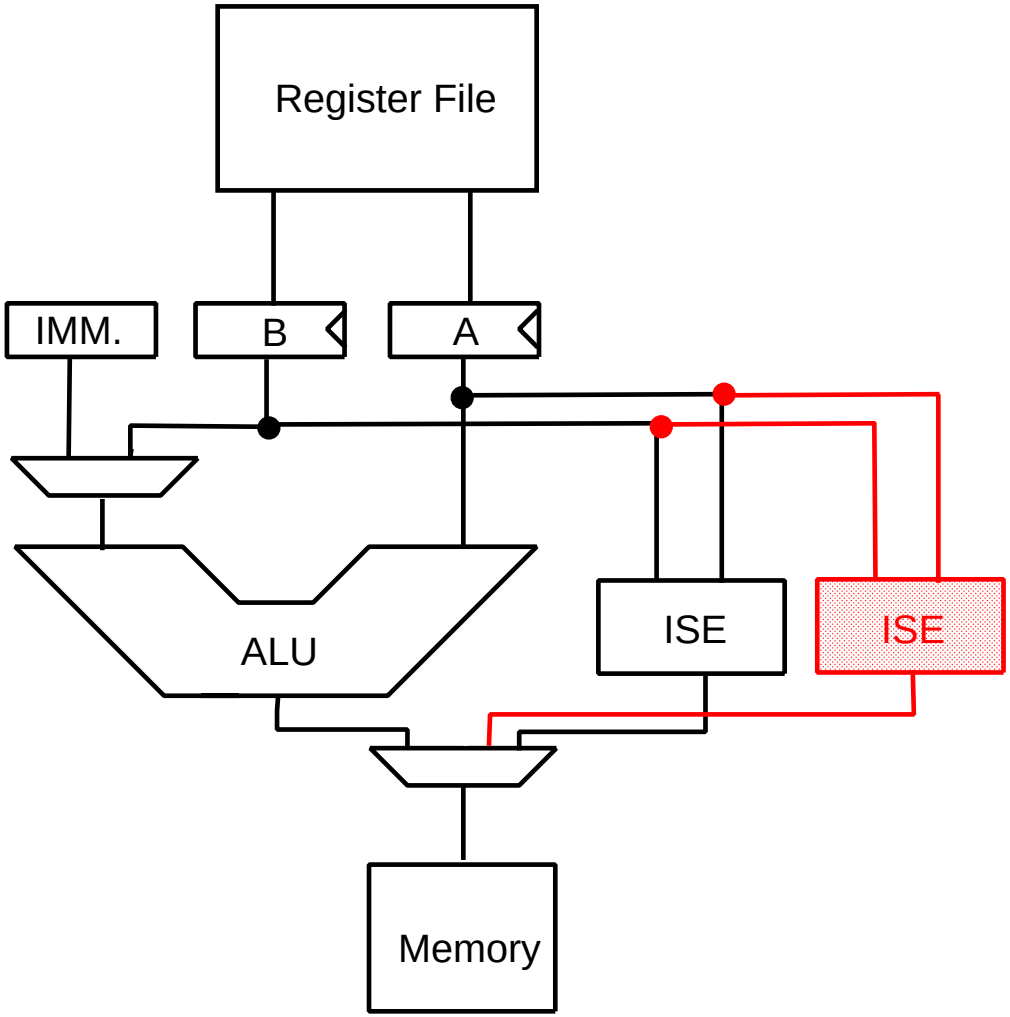
Algorithm partitioning tool



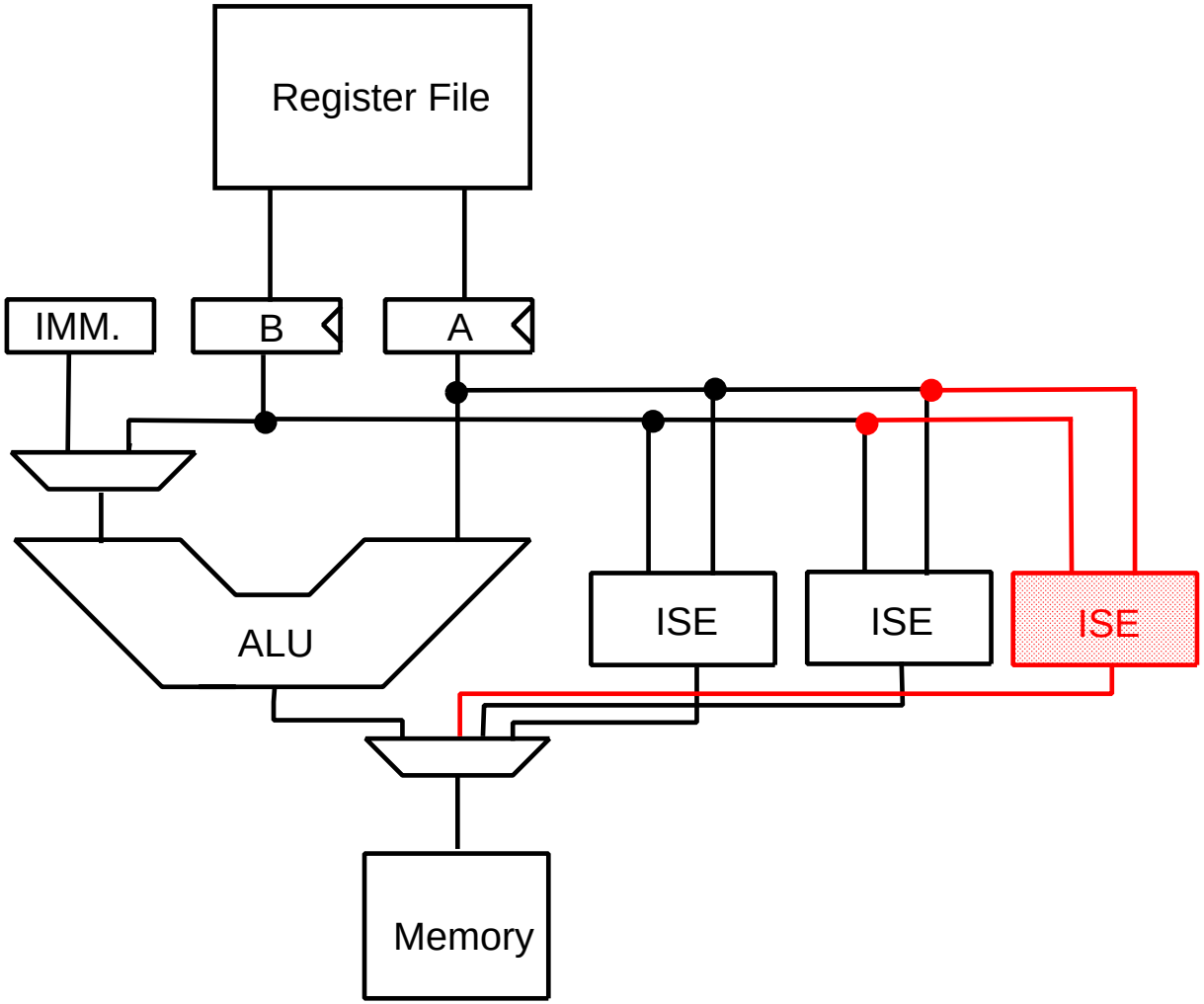
Algorithm partitioning tool



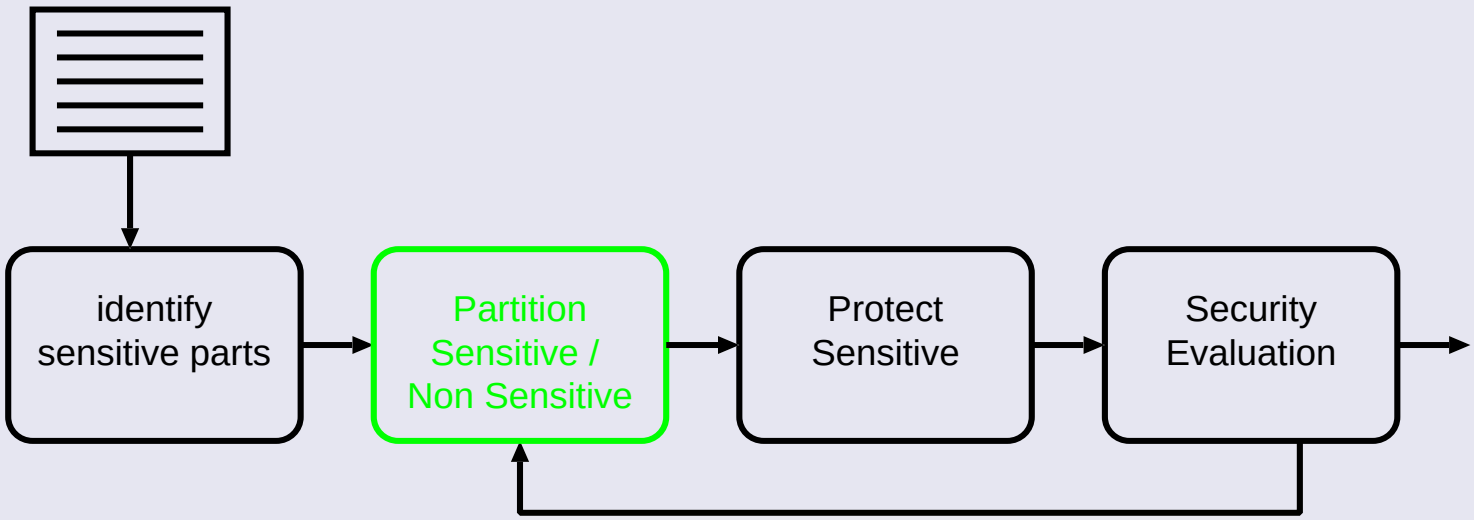
Algorithm partitioning tool



Algorithm partitioning tool

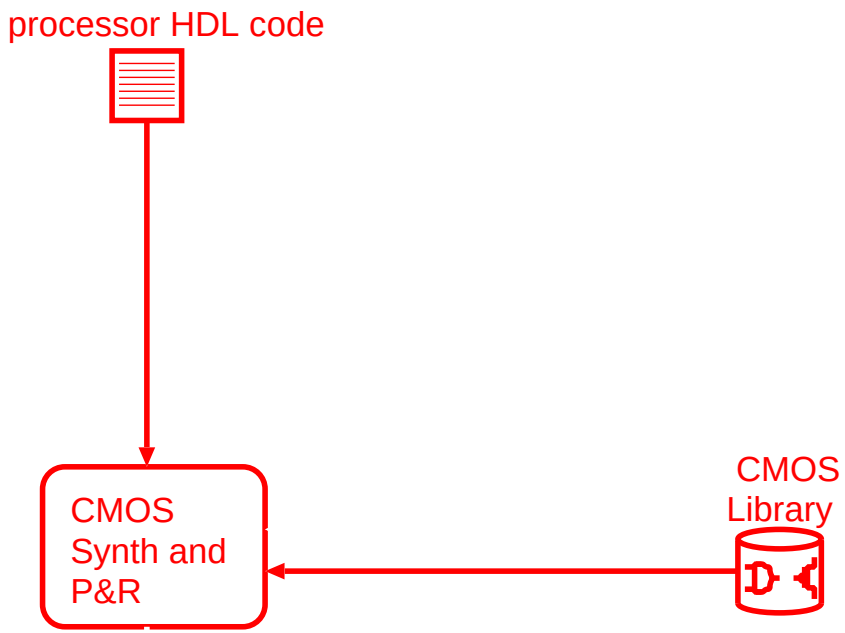


Needed "Basic Blocks"

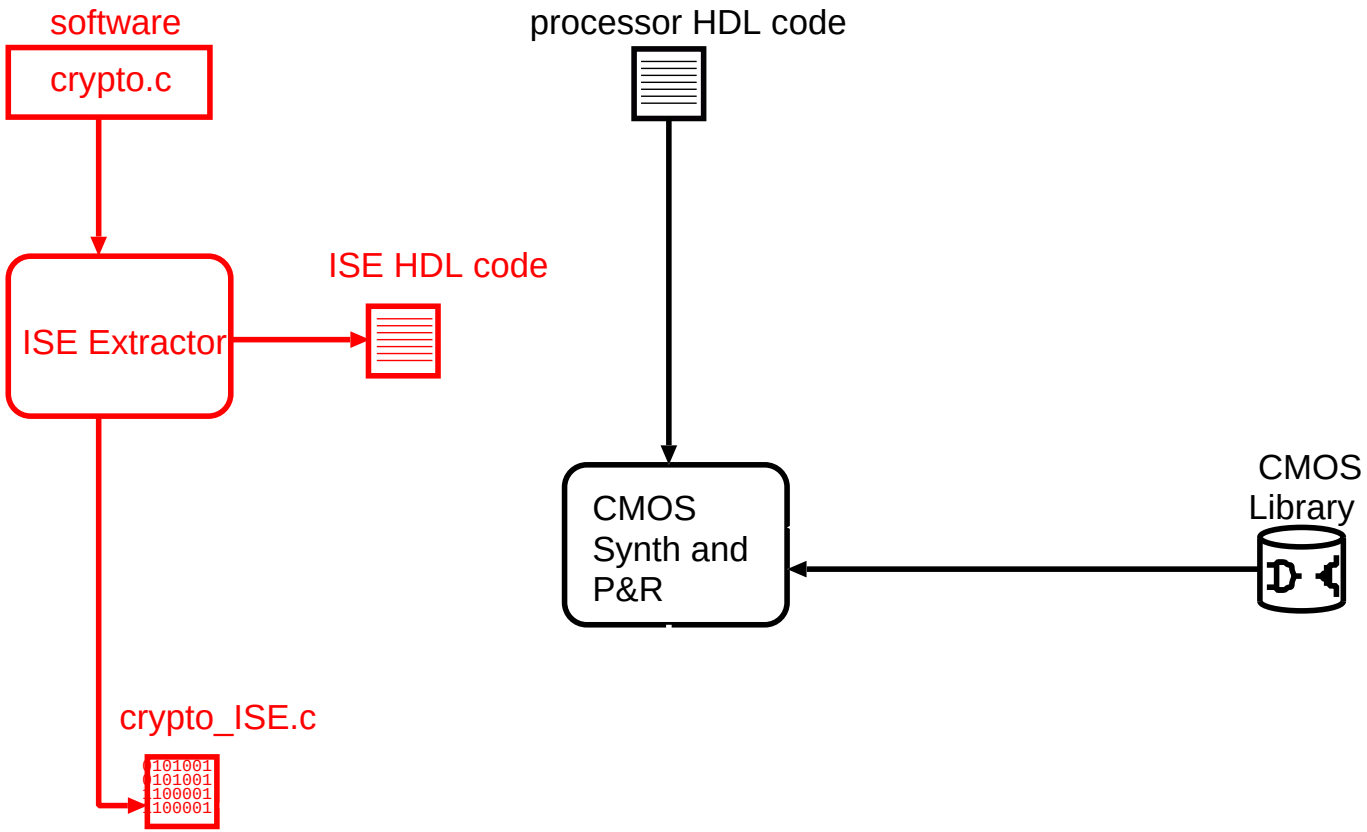


- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow? ✓
- Partition the algorithm? ✓

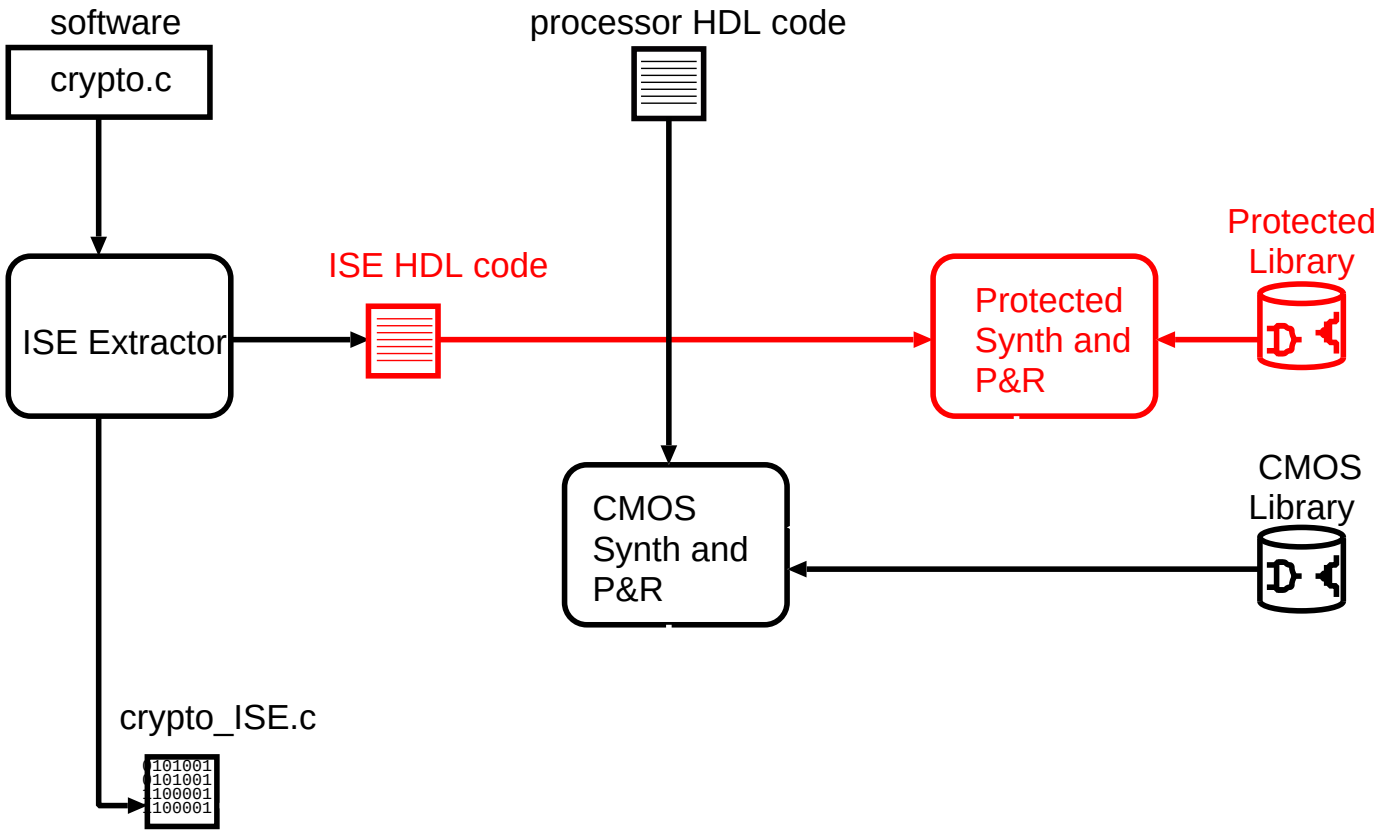
The CMOS Design Flow



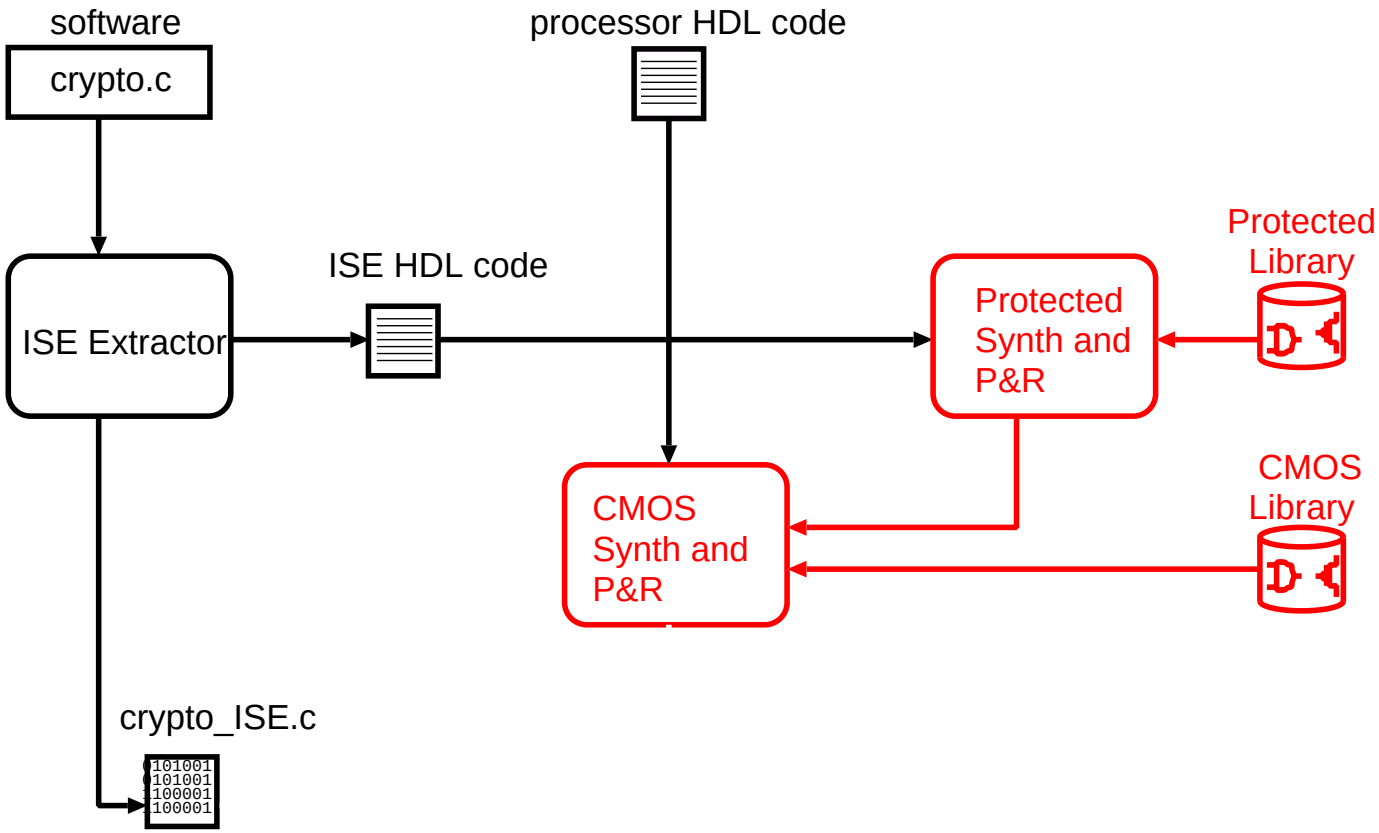
The Processor Customization



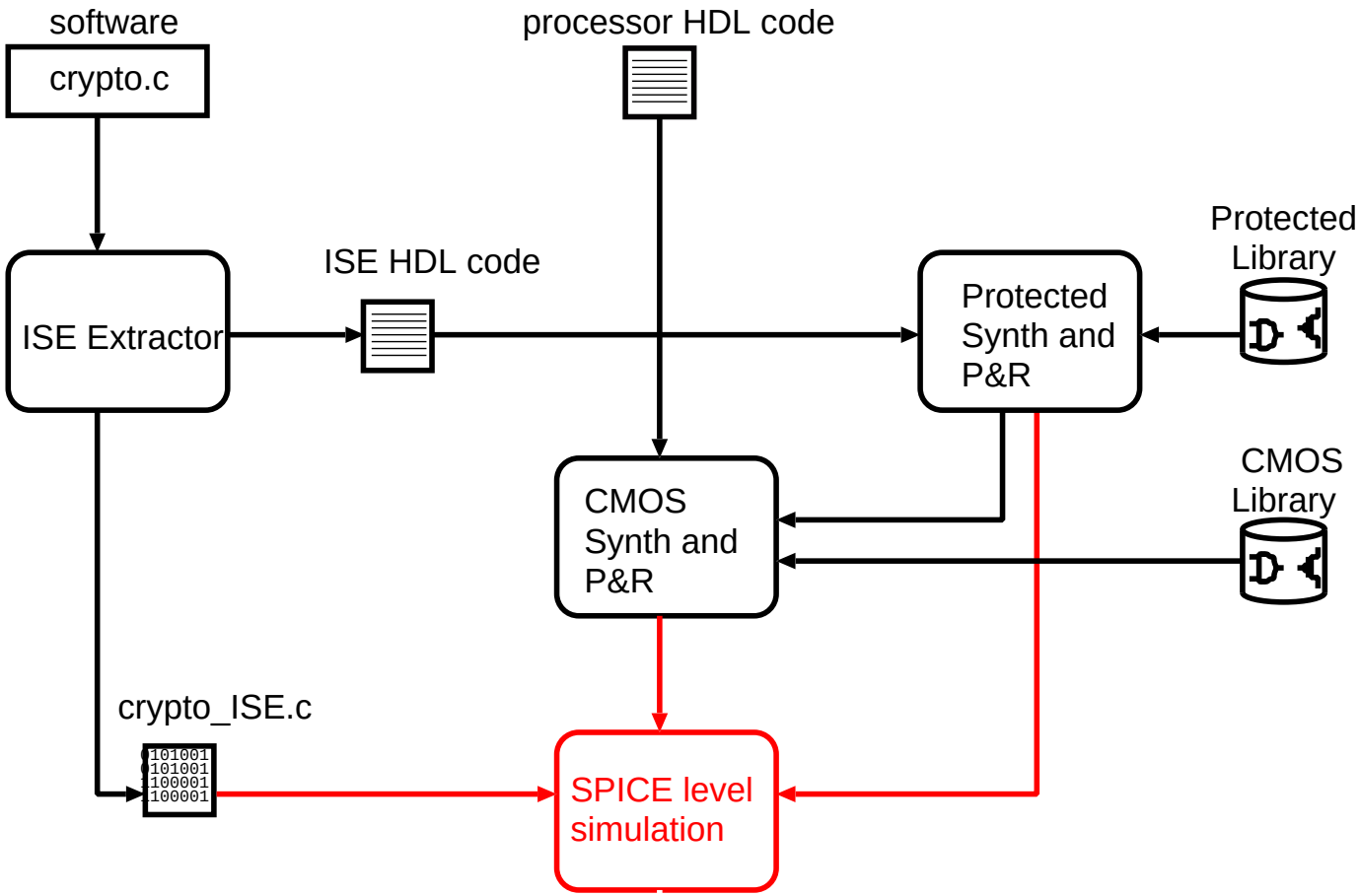
The Protected Design Flow



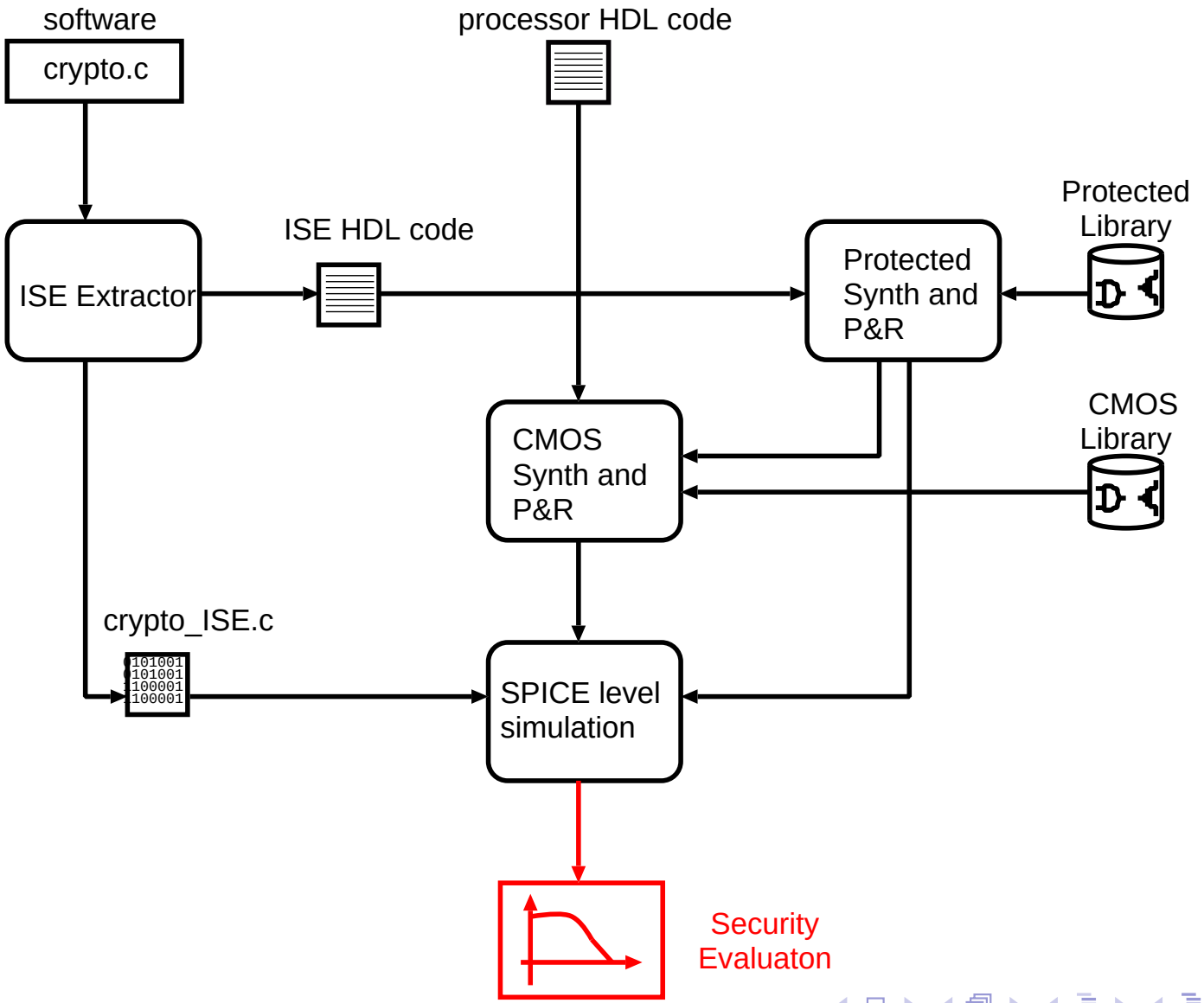
The Hybrid Design Flow



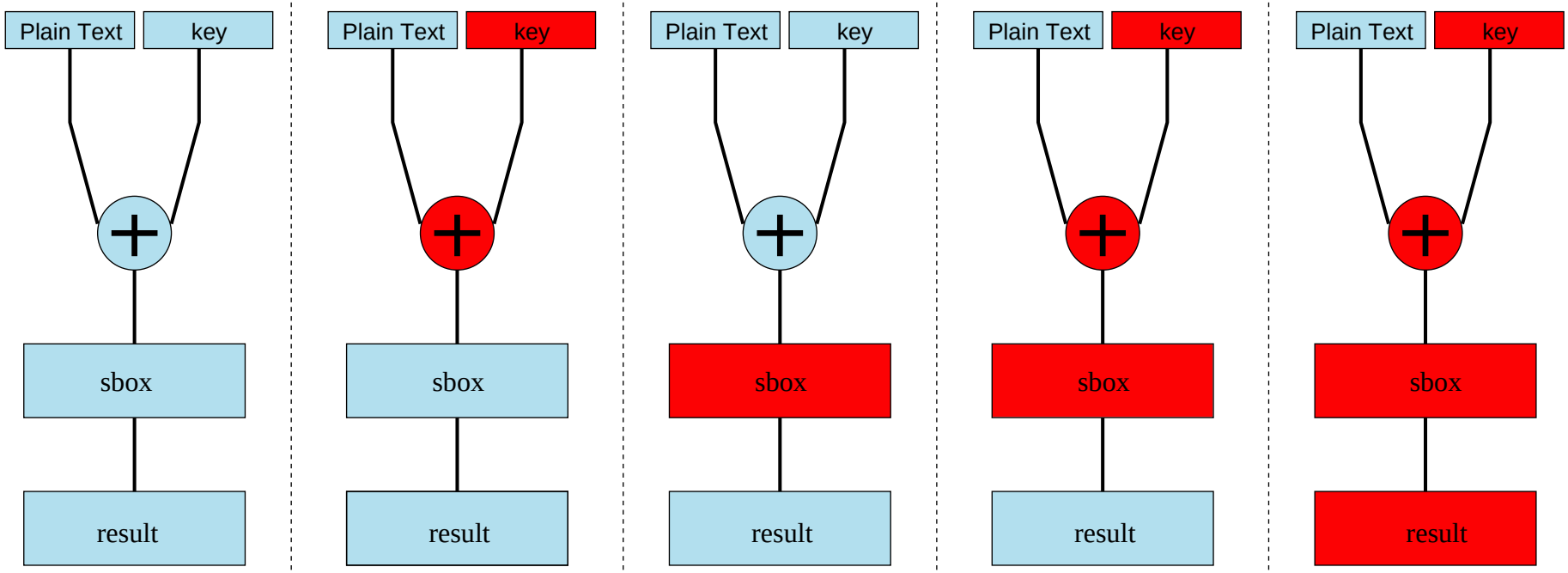
The Simulation Environment

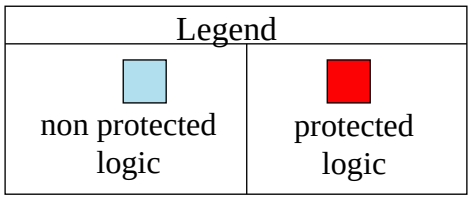


The Design Evaluation



Partitioning of the PRESENT algorithm S-box



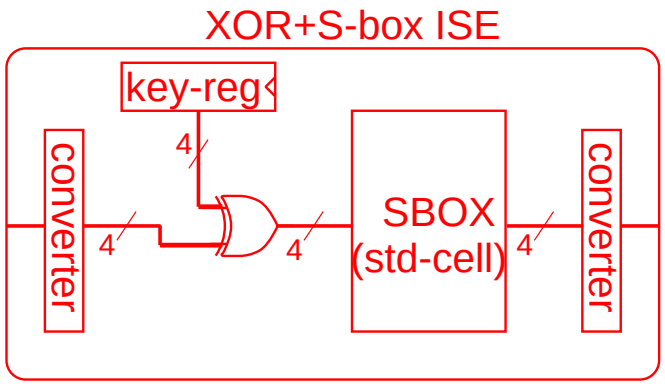


Example of ISE and its Source Code

```
// Calculate S-box (plaintext XOR key)
int PRESENT(int plaintext, int key) {
1 int result = 0; // initialize the result
2 plaintext = plaintext ^key; // perform the xor with the key
3 result = S[plaintext]; // perform the S-box
4 return result; }; // return the result
```

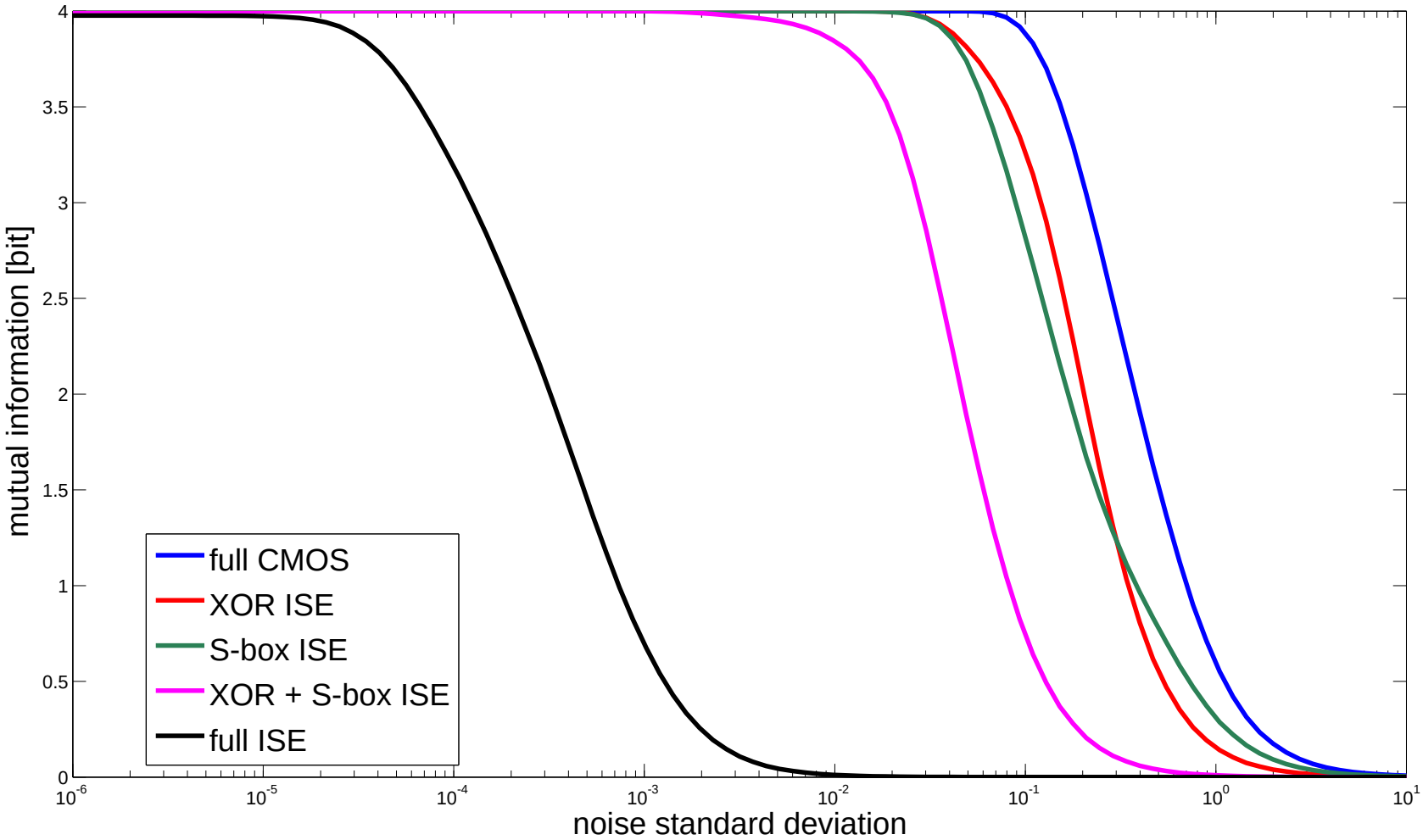
Example of ISE and its Source Code

```
// Calculate S-box (plaintext XOR key)
int PRESENT(int plaintext, int key) {
1 int result = 0; // initialize the result
2 plaintext = plaintext ^key; // perform the xor with the key
3 result = S[plaintext]; // perform the S-box
4 return result; }; // return the result
```

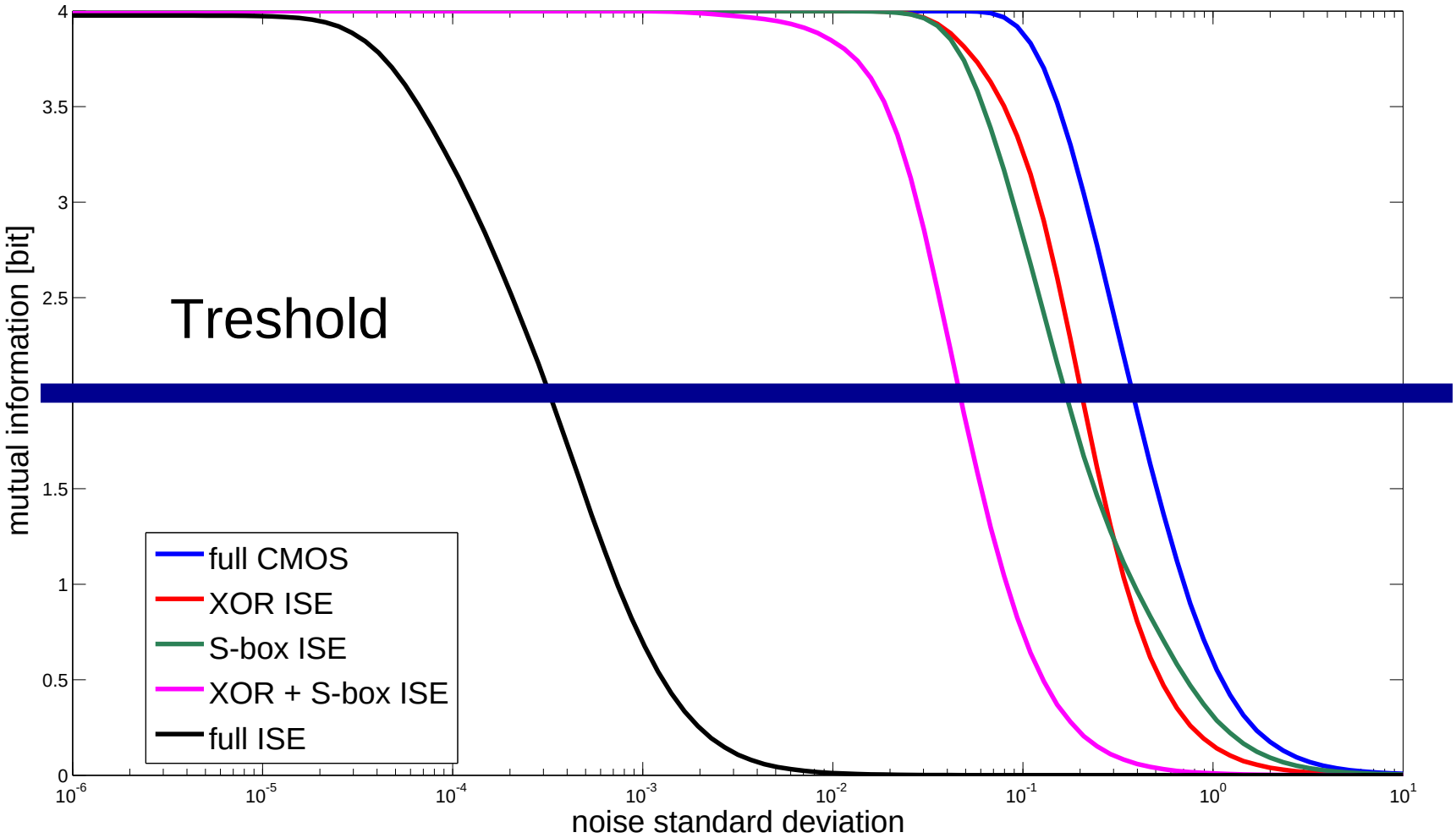



```
// Calculate S-box (plaintext XOR key)
int PRESENT_XOR+S-box-ISE(int plaintex) {
1 int result = 0; // initialize the result
  // instantiate the new instruction s-box(pt ^key)
2 Instr_1(plaintex, result);
3 return result; }; // return the result
```

Security Evaluation



Security Evaluation



Total Time for experiments

PC Features:

- CPU: Intel(R) Core(TM)2 Quad CPU Q6700
- GHz 2.6
- Memory: 4 GB

Example program 470 clock cycles (boot+cipher)

SPICE Level Simulation (Synopsys Nanosim resolution: 1ps):

- Total simulated time 4700ns
- Total simulation time more or less 20 minutes
- 2.8s per clock cycle (full processor simulation core+ISE)

Security Evaluation

- 4 hours per partitioning

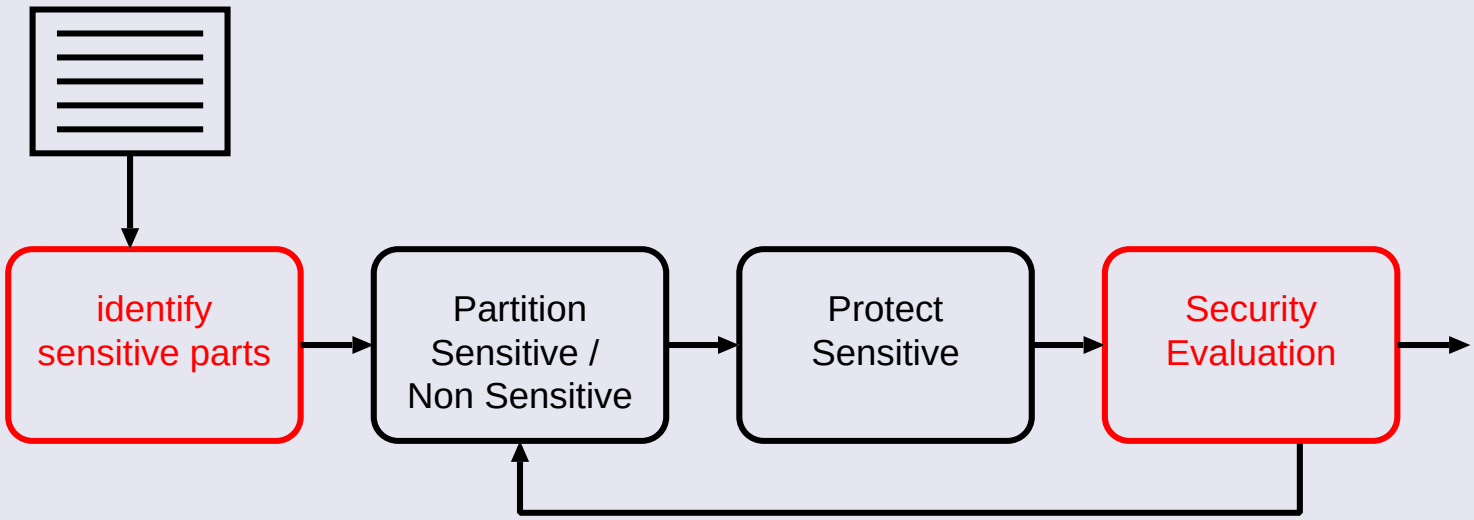
Full case study

- Worst case: 15 days on a single PC
- Parallelizable! Actual experiment: 2 days on 8 PCs

Outline

- Logic Synthesis (Secure)
- Design Flow for secure ISE
- Quick note on Software

Needed "Basic Blocks"

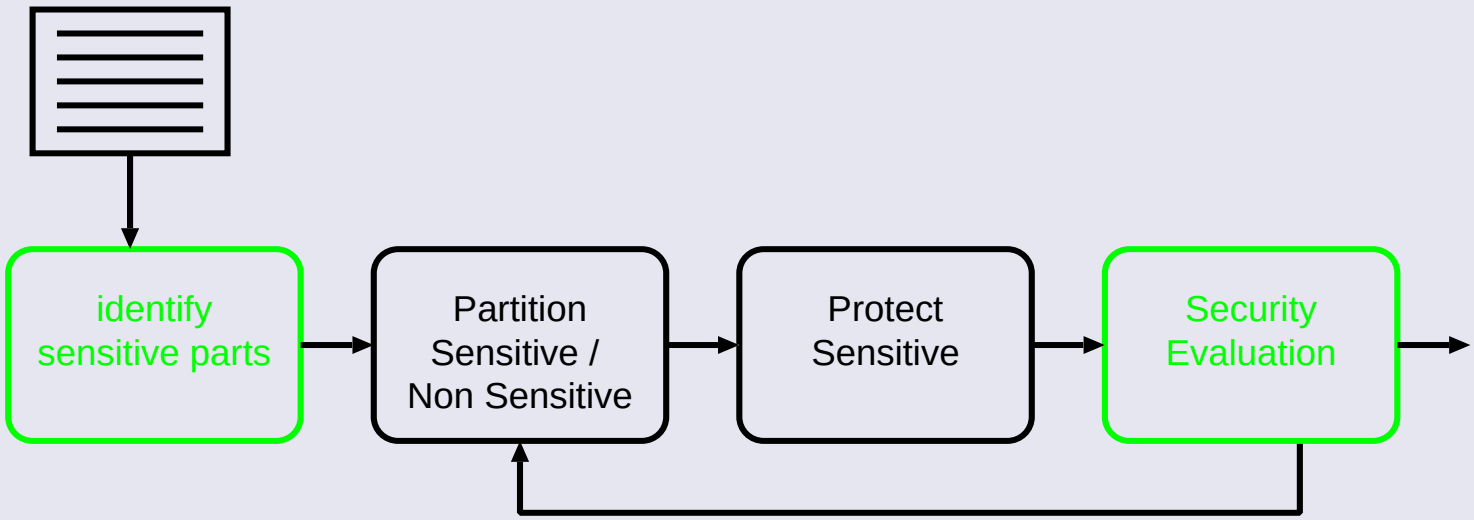


- Generate useful power traces?
- Measure the DPA resistance?
- Countermeasure and its design flow?
- Partition the algorithm?

Real Measures on Microcontroller

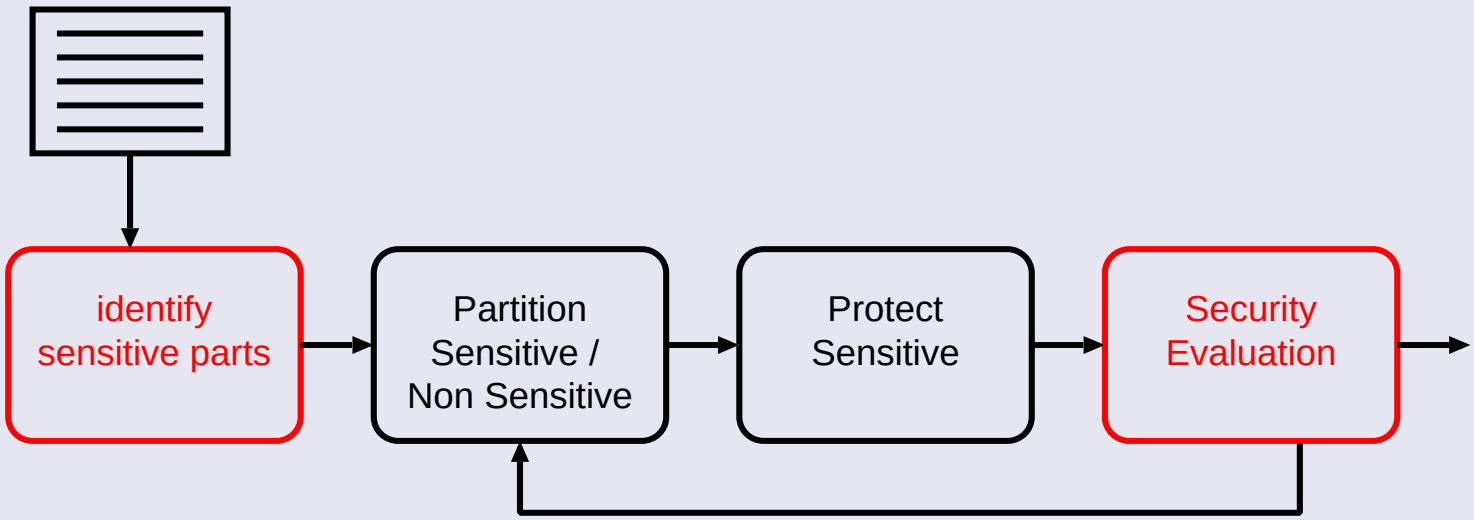
- No need to simulate or emulate
- Power traces are obtained directly by measuring with an oscilloscope the software running on the microcontroller

Needed "Basic Blocks"



- Generate useful power traces? ✓
- Measure the DPA resistance?
- Countermeasure and its design flow?
- Partition the algorithm?

Needed "Basic Blocks"



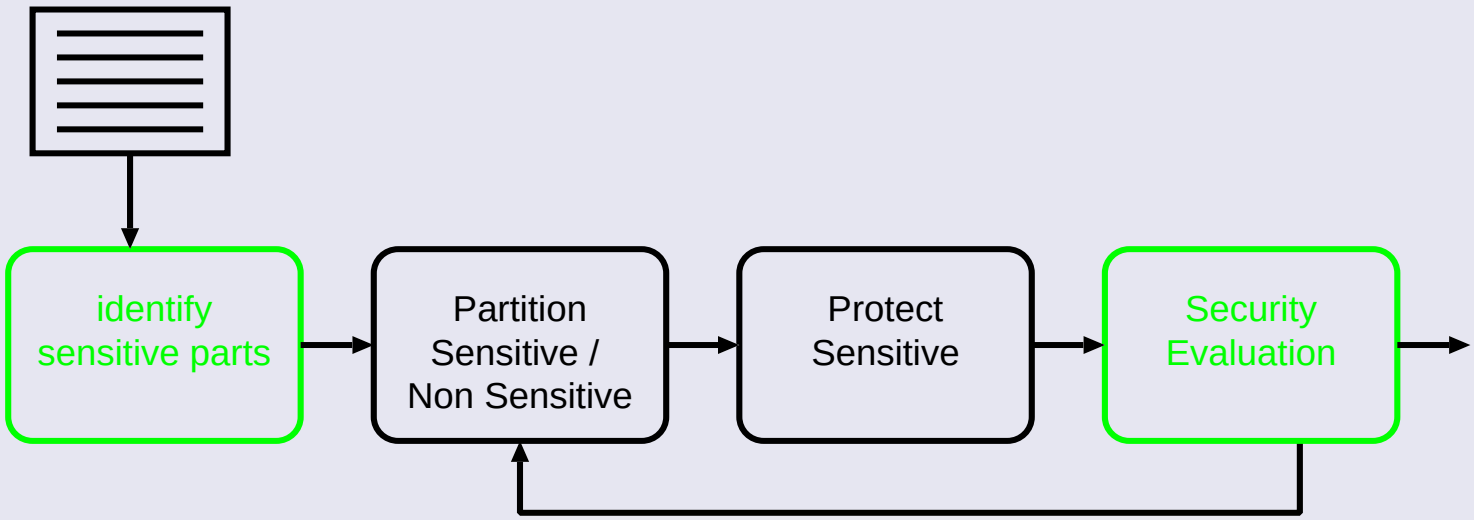
- Generate useful power traces? ✓
- Measure the DPA resistance?
- Countermeasure and its design flow?
- Partition the algorithm?

Metric

Same as before....

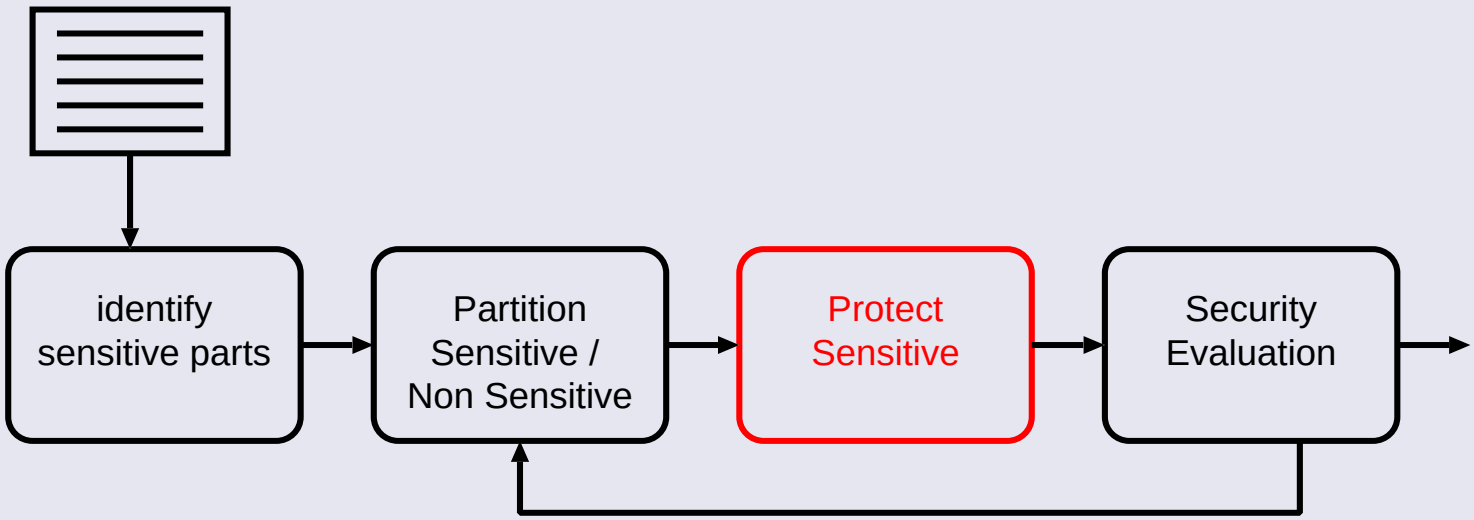
Applied instruction by instruction!

Needed "Basic Blocks"



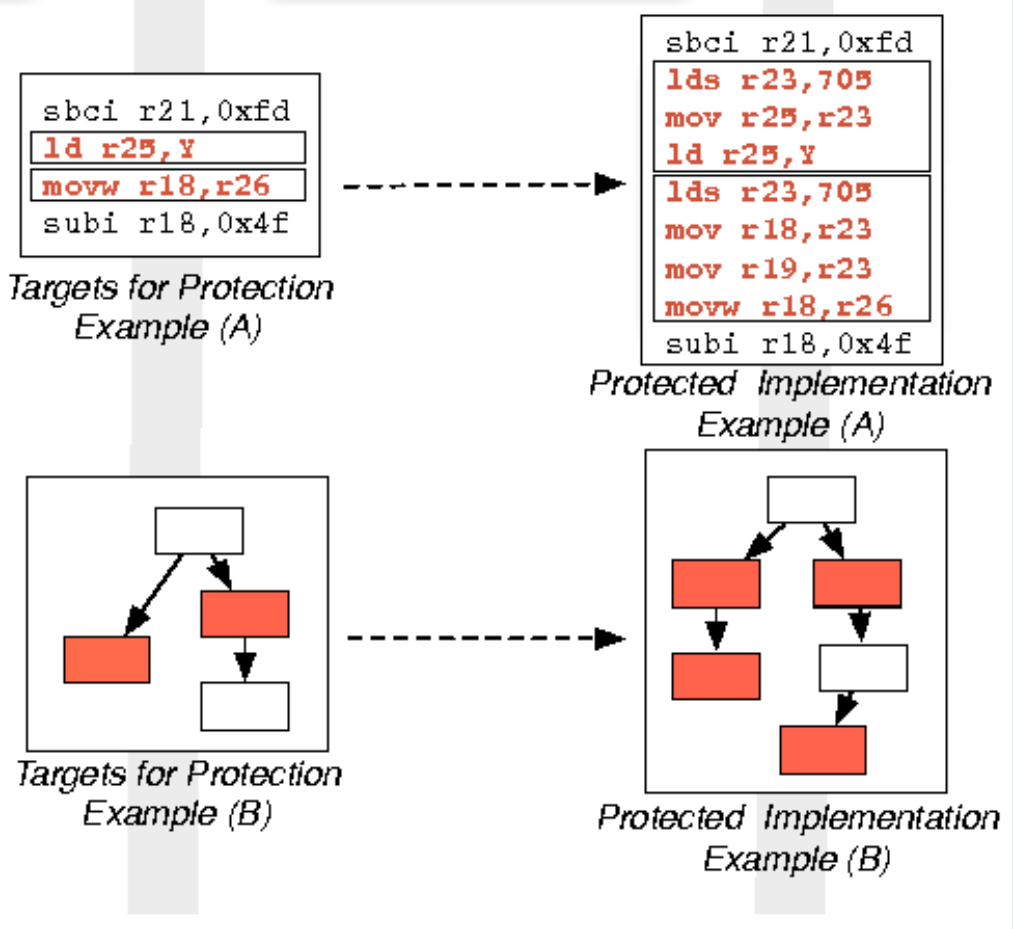
- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow?
- Partition the algorithm?

Needed "Basic Blocks"

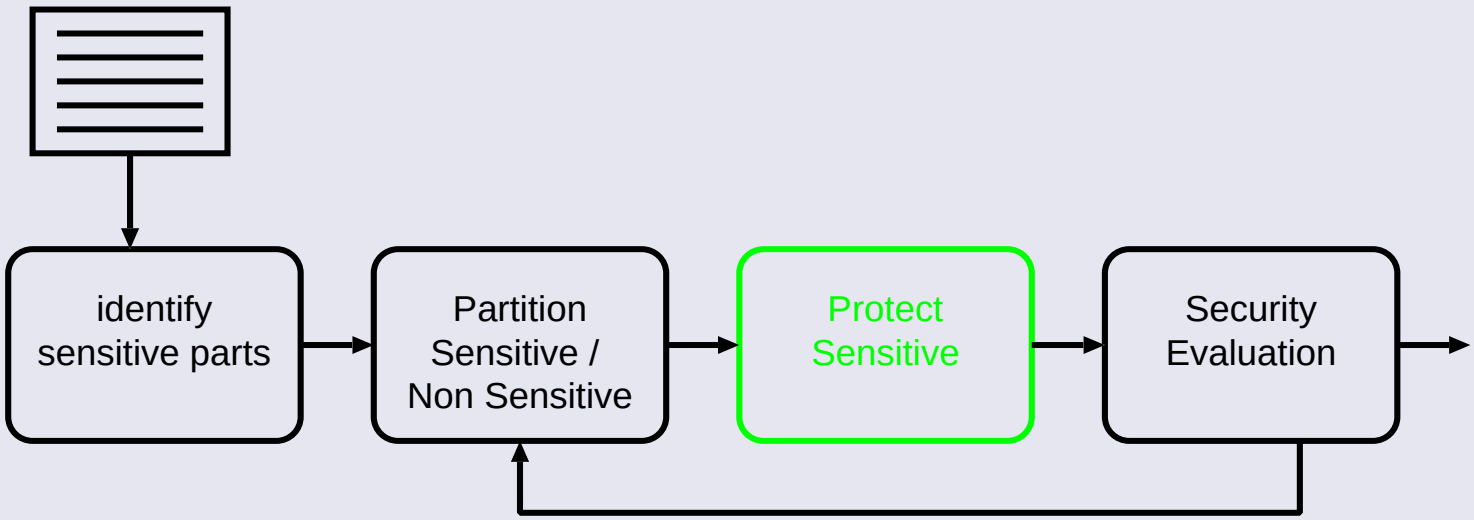


- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow?
- Partition the algorithm?

Code Transformation

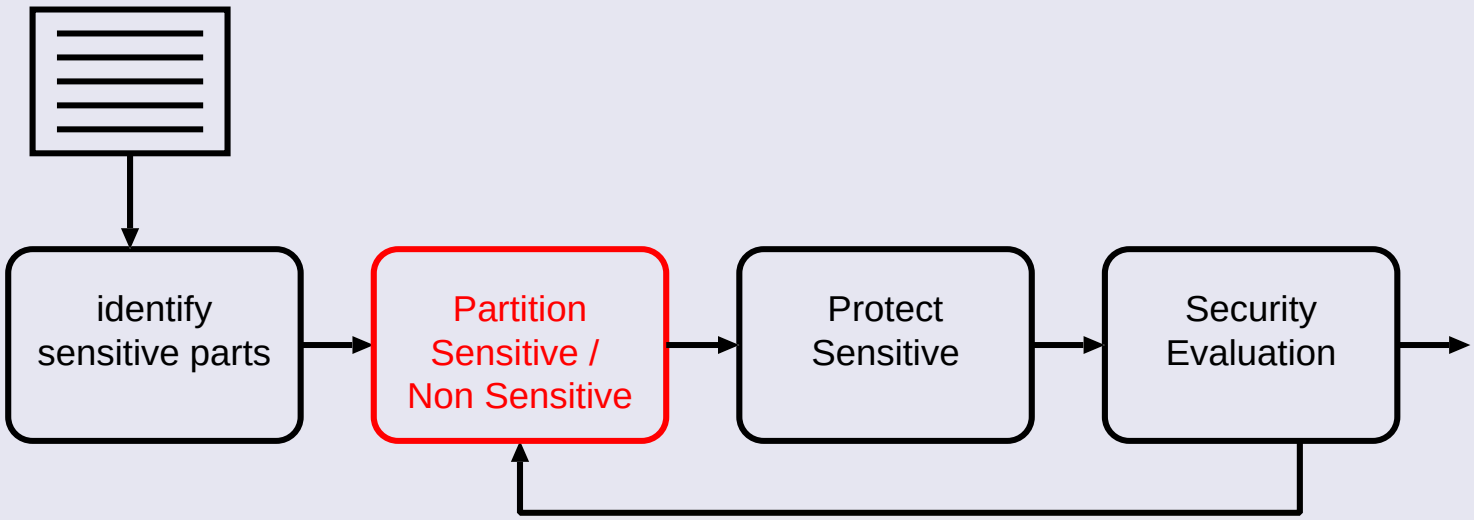


Needed "Basic Blocks"



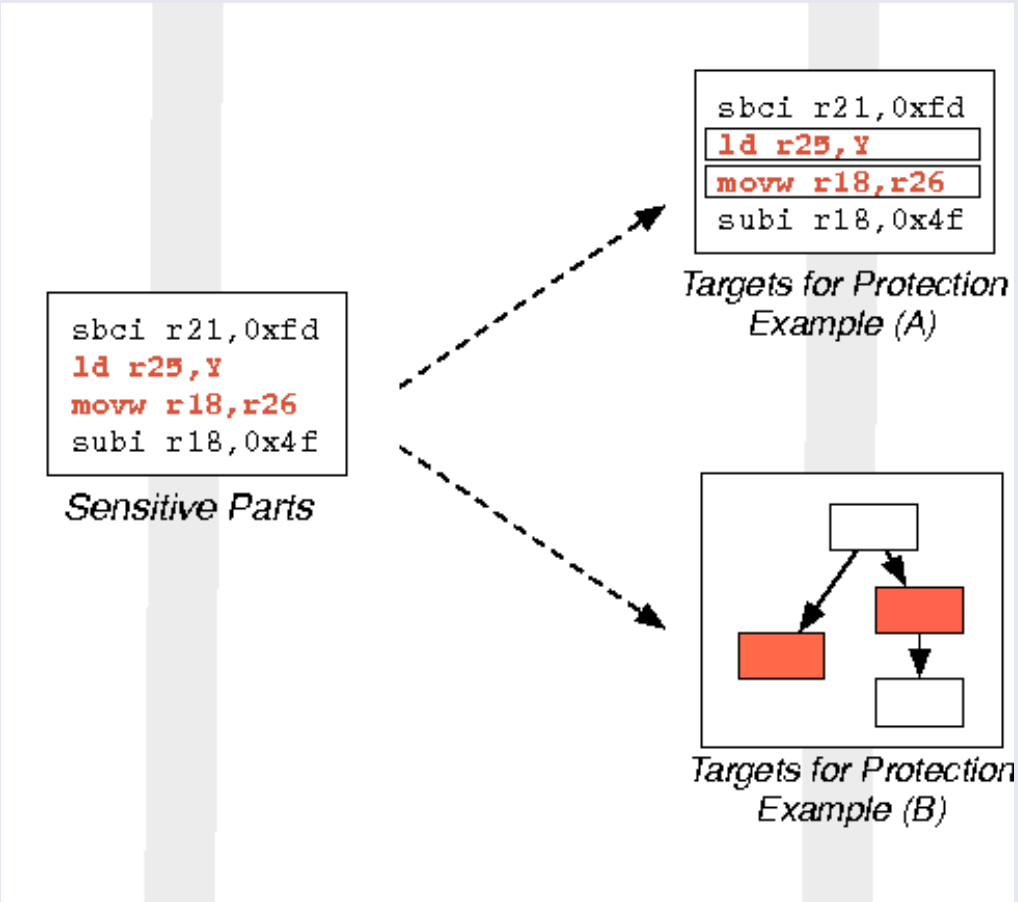
- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow? ✓
- Partition the algorithm?

Needed "Basic Blocks"

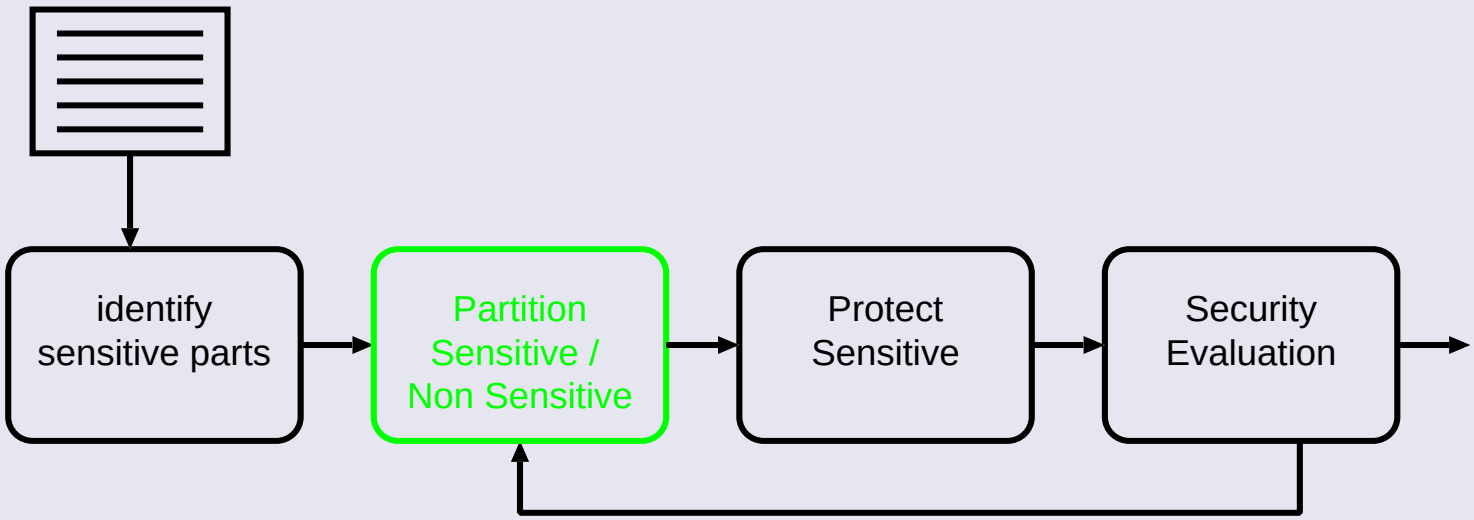


- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow? ✓
- Partition the algorithm?

Transformation Target Identification

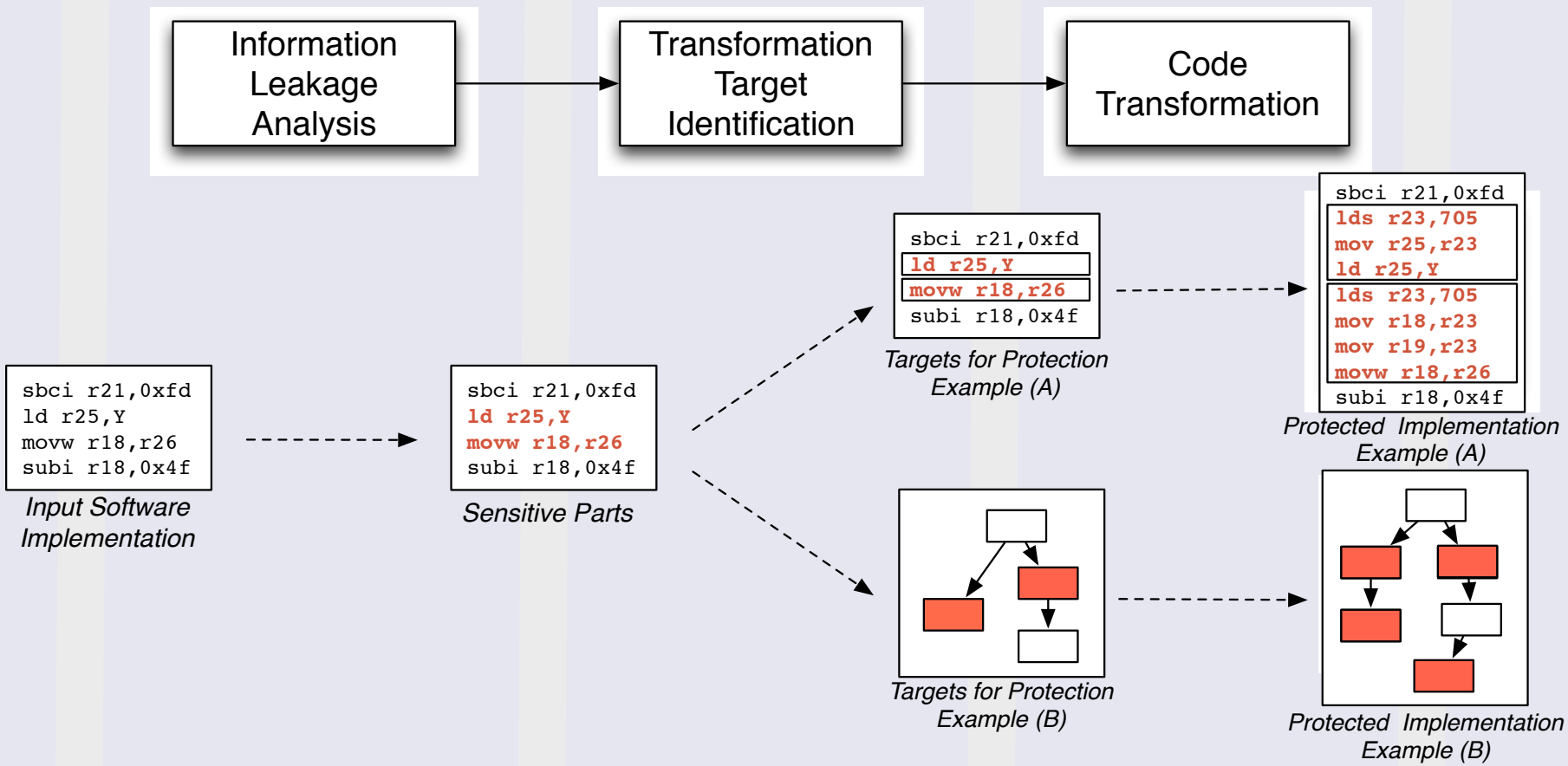


Needed "Basic Blocks"

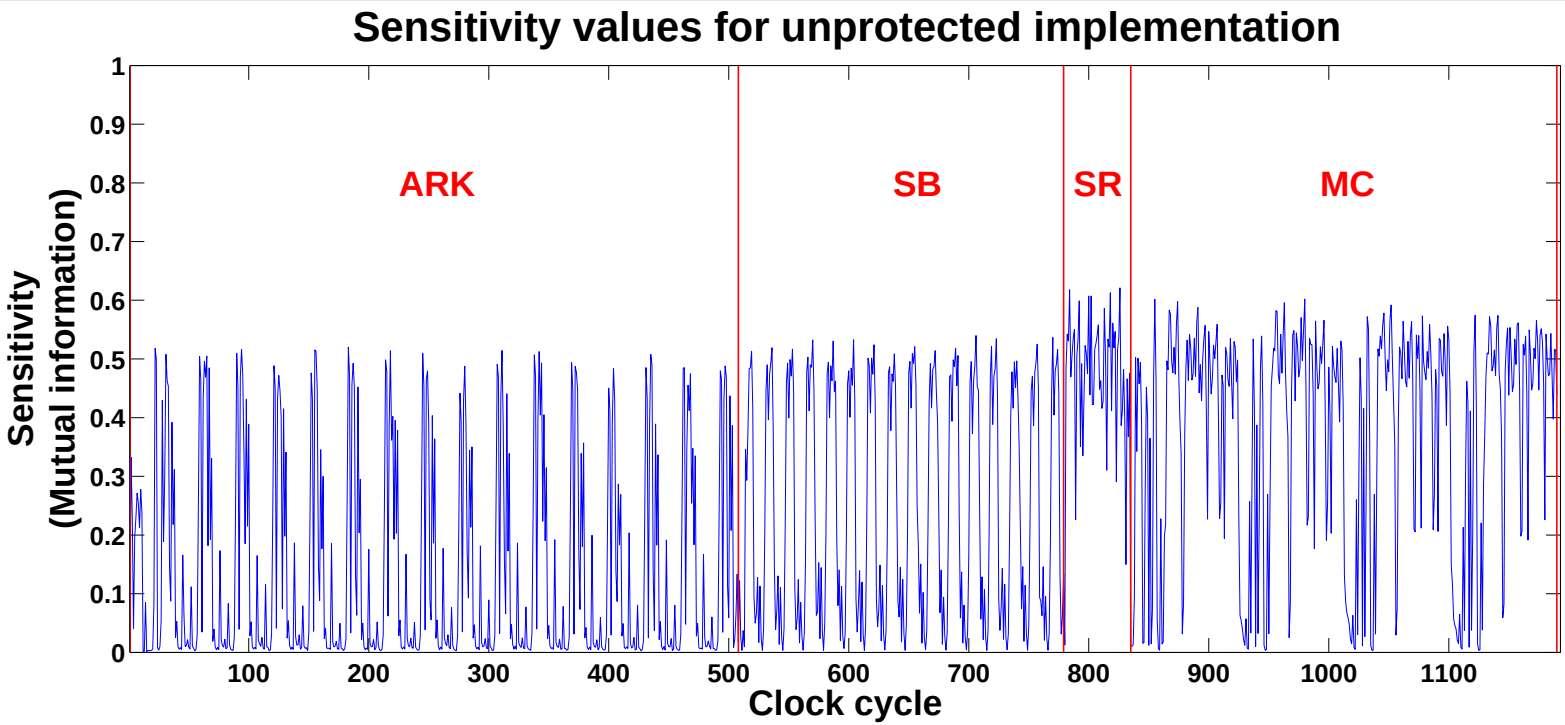


- Generate useful power traces? ✓
- Measure the DPA resistance? ✓
- Countermeasure and its design flow? ✓
- Partition the algorithm? ✓

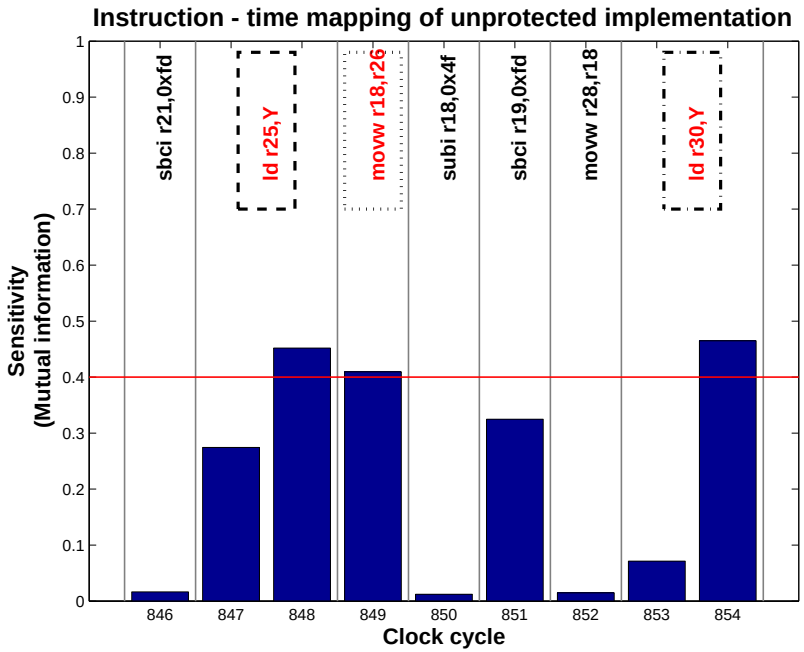
Overall Software Flow



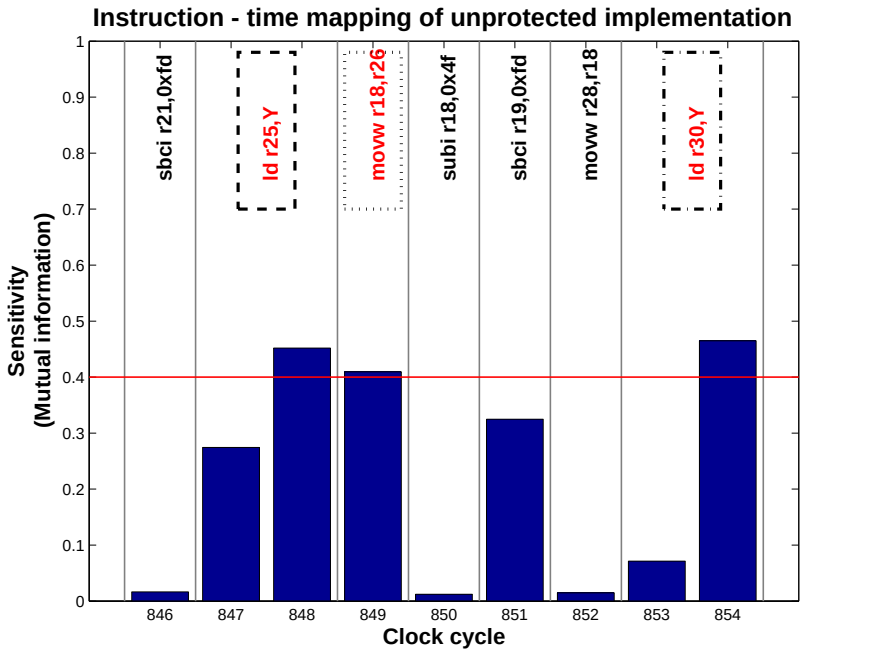
Information Leakage Analysis

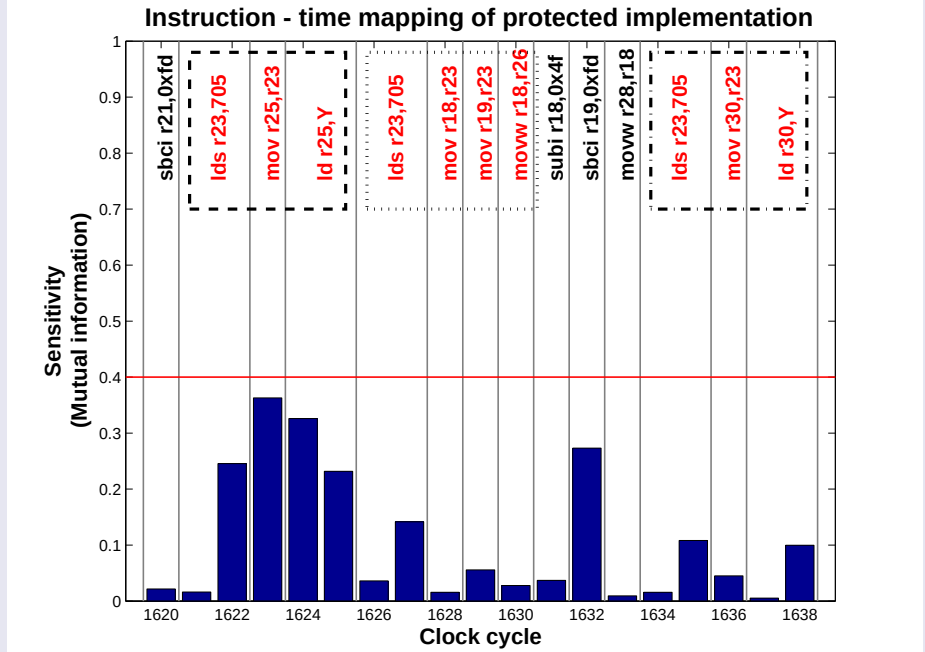


Example on Software

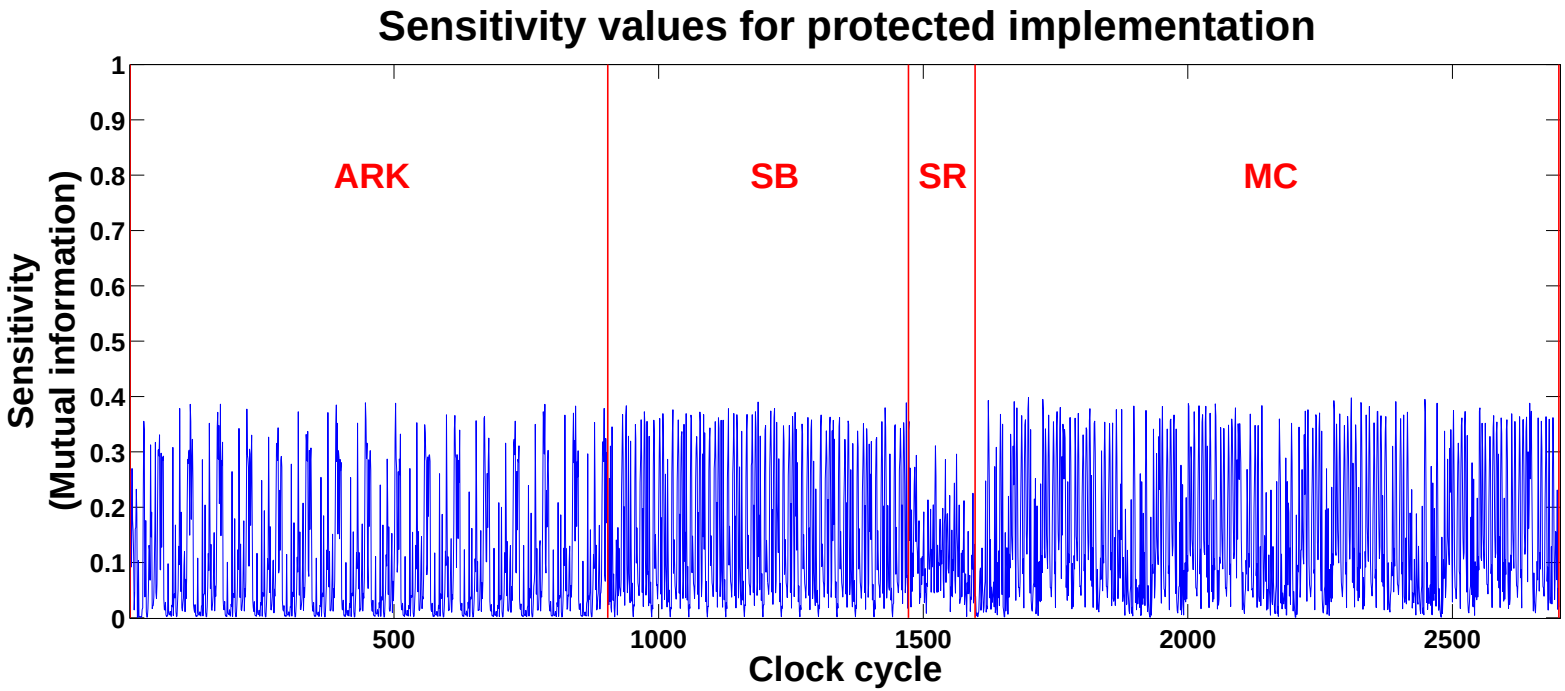


Example on Software





Security Evaluation



Conclusions and Tips

- Initial steps for power analysis are promising
- This is just the beginning...

PS: **Never** re-invent the wheel!

Acknowledgments

- Paolo Ienne, Alessandro Cevrero, Yusuf Leblebici, Stéphane Badel, Johann Großschädl, Ali Galip Bayrak, Axel Poschmann, Zeynep Toprak, Marco Macchetti, Laura Pozzi, Christof Paar, Frank Gurkaynak, François-Xavier Standaert, Theo Kluter, Philip Brisk, Michael Schwander, Thomas Eisenbarth

Questions?

"There is beauty in what we do in EDA!"

Alberto Sangiovanni-Vincentelli, EDA Café - 2009

Thank you for your attention!

mail: regazzoni@alari.ch