A DPA robust S-BOX implementation on a secure asynchronous FPGA

Taha Beyrouthy - Laurent Fesquet

taha.beyrouthy@imag.fr - Laurent.fesquet @imag.fr

Cryptarchi'09

Outline

- Motivation
- Security shortcomings of FPGAs
- Asynchronous Logic principles
- Asynchronous FPGA
- Tech mapping strategy
- Test result – SBOX Implementation
- Conclusion

Motivation- Why FPGAs?

- Algorithm agility :
  - *The majority of modern security protocols, such as SSL, are algorithm independent and allow multiple encryption algorithm*
- Architecture Efficient:
  - *In certain cases a hardware architecture can be much more efficient if it is designed for a specific set of parameters*

Motivation- Why FPGAs?

- Resource Efficiency:
  - *The same FPGA could be used for multiple through run-time reconfiguration*
- Throughput:
  - *General-purpose CPUs are not optimized for fast execution of cryptographic algorithms*
- Cost Efficiency:
  - Cost of development/time-to-market
  - Unit Price

Security shortcomings of FPGAs

- Objectives of an attacker:
  - *To learn a confidential cryptographic key*
  - *The one-to-one copy or cloning*
  - *Reverse-engineer , specially in cases like proprietary cryptographic algorithms.*

Different type of attacks:

- Black box Attack:
  - *Not a real threat due to the complexity.*

Security shortcomings of FPGAs

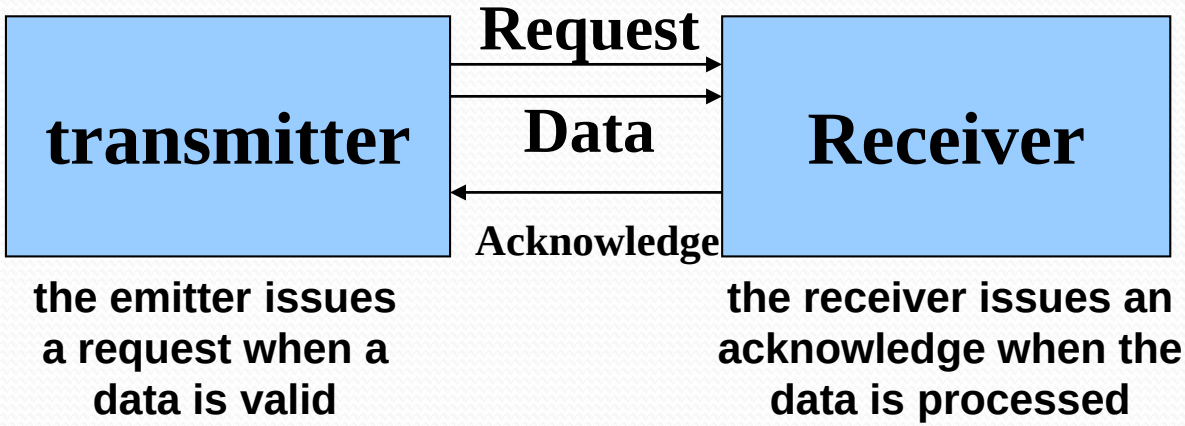
- Cloning or reverse-engineer:
  - *The attacker has to be in possession of the bitstream in order to get the design of the proprietary algorithm or the secret key*
- Physical attack
  - *Aim to investigate the chip design in order to get information about the cryptographic algorithm, by probing some points inside the chip.*

Security shortcomings of FPGAs

- Side channel Attack:
  - *Any cryptographic system might provide a side channel that leaks unwanted information: power consumption, timing behavior, electromagnetic radiation, temperature,…*

Asynchronous Logic Principles (1/5)

- Communication protocol : *Hand-Shake Protocol*



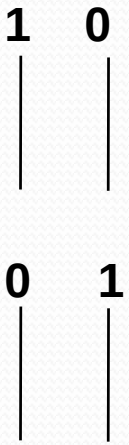
  - No global clock = no global timing assumption
  - Hazard free is required
  - 2 types of hand-shake protocol 2-phase & 4-phase

Asynchronous Logic Principles (2/5)

- Data encoding:
  - *1-of-n encoding → balanced hamming weight*
    - *Ex: 1-of 2 = dual rail = 2 wires to encode one bit*

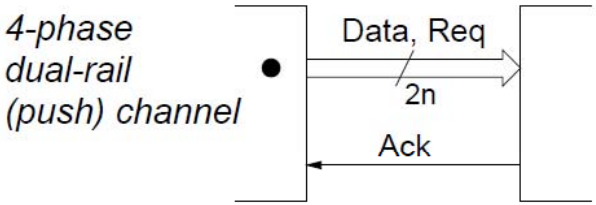
logic 1 → 2 wires

logic 0 → 2 wires

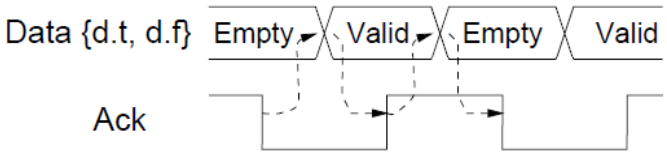


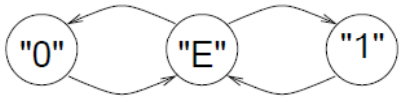
Asynchronous Logic Principles (3/5)

- 4-phase protocol principles:



	W1	W2
Empty (« E »)	0	0
Valid « 0 »	0	1
Valid « 1 »	1	0
Not used	1	1

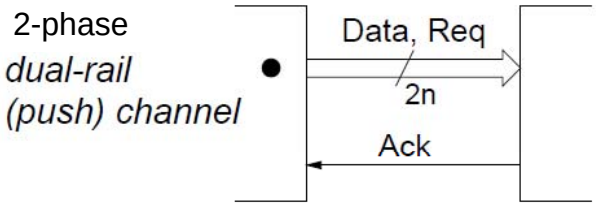


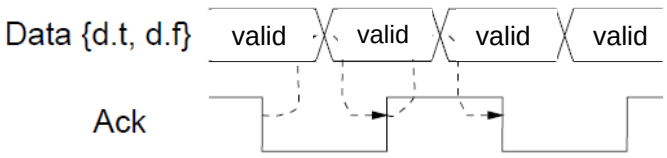


(1) The Sender issues a valid codeword,
(2) The Receiver absorbs the codeword and sets acknowledge high,
(3) The Sender responds by issuing the empty codeword, and
(4) The receiver acknowledges this by taking acknowledge low. At this point the sender may initiate the next communication cycle

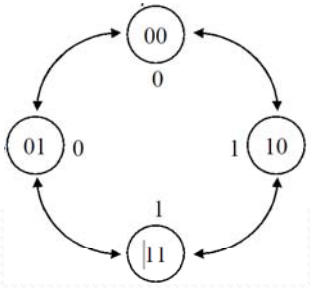
Asynchronous Logic Principles (4/5)

- 2-phase protocol principles:





	w1	w0
Valid « 0 »	0	0
Valid « 1 »	0	1
Valid « 0 »	1	0
Valid « 1 »	1	1



The information is encoded as transitions (events)

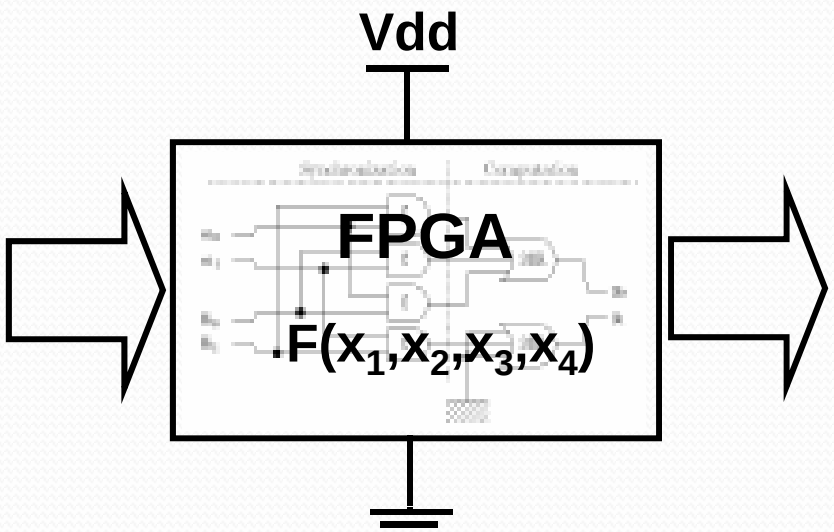
(1) The Sender issues a valid codeword,
(2) The Receiver absorbs the codeword and sends an acknowledge signal,
At this point the sender may initiate the next communication cycle

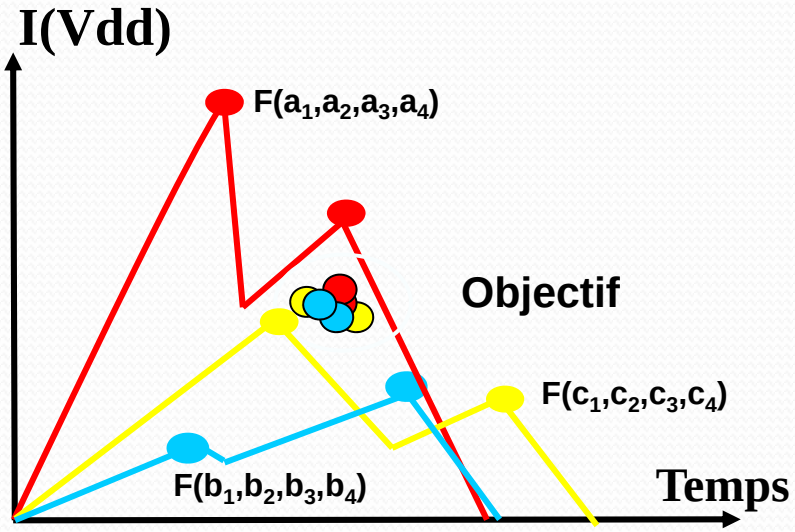
Asynchronous Logic Principles (5/5)

- **.. for security applications :**
  - Balanced data encoding (1-of-n)
  - Smooth and low power consumption
  - Low electromagnetic emission.

Asynchronous FPGA (1/4)

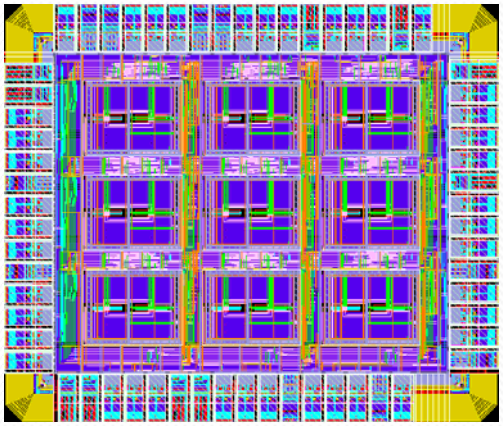
- Secure FPGA :
  - Security constraint to be robust against DPA :
    - Data independent power consumption

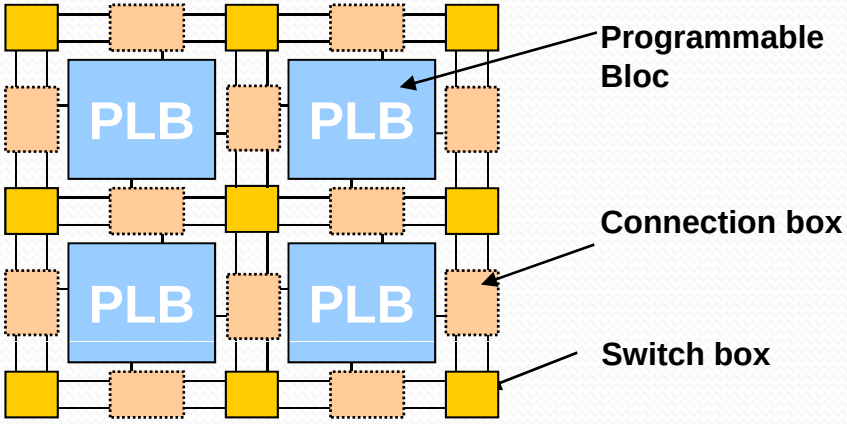




Asynchronous FPGA (2/4)

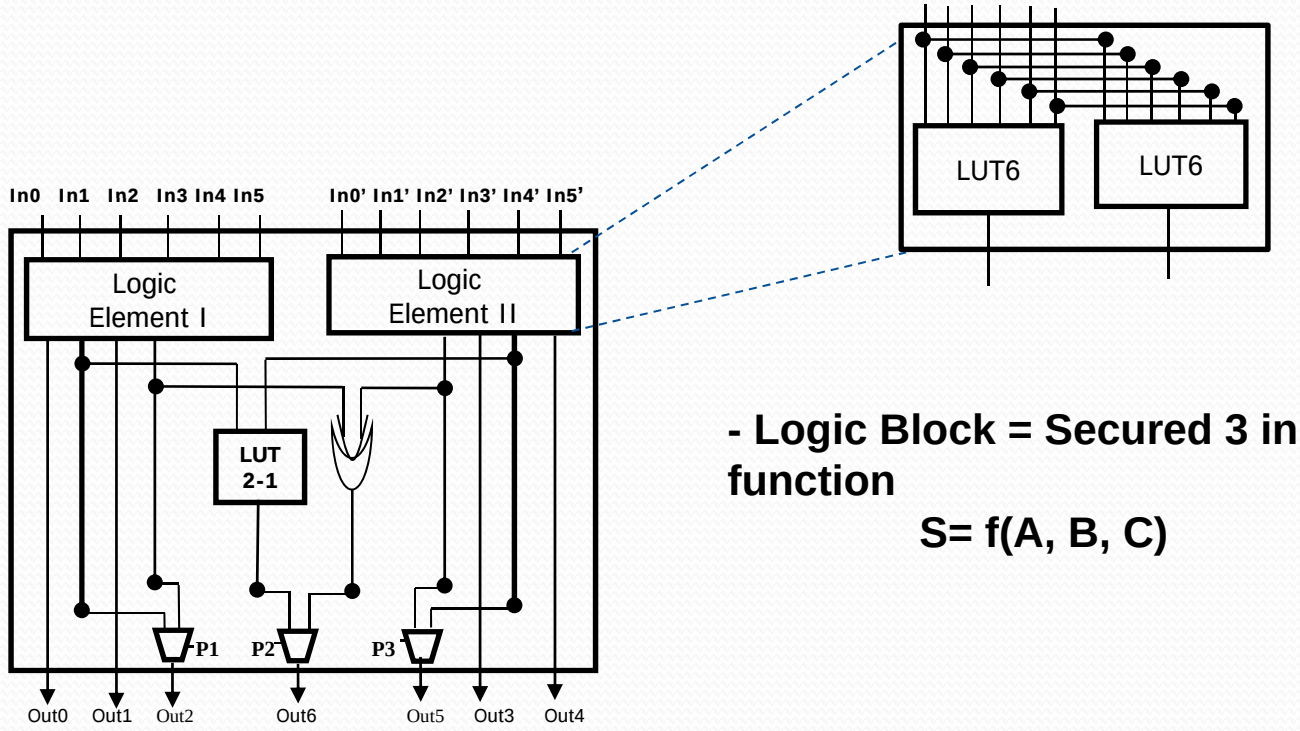
- Island style, designed to be :
  - Electrically balanced (Balanced input capacitances)
  - Logically balanced (Same logical depth for all inputs)





Asynchronous FPGA (4/4)

- Programmable Logic Bloc:



- Logic Block = Secured 3 input dual rail function

S= f(A, B, C)

Asynchronous FPGA (3/4)

- Programmable Bloc :
  - Full custom layout
  - Electrically balanced (symmetric architecture)
  - Support 2-phase and 4-phase protocols
  - Support multi-rail encoding

Tech-mapping strategy (1/3)

- The goal of the technology mapping is :
  - to balance the architecture of the whole circuit
  - to reduce area of the circuit

Tech-mapping strategy (2/3)

The strategy:

With « n » = number of inputs :

If n ≤ 6 then → *use one LE to implement the function*

If n ≥ 7 then → *Run the "method_for_more_than_7"*

Tech-mapping strategy (3/3)

Method_for_more_than_7 :

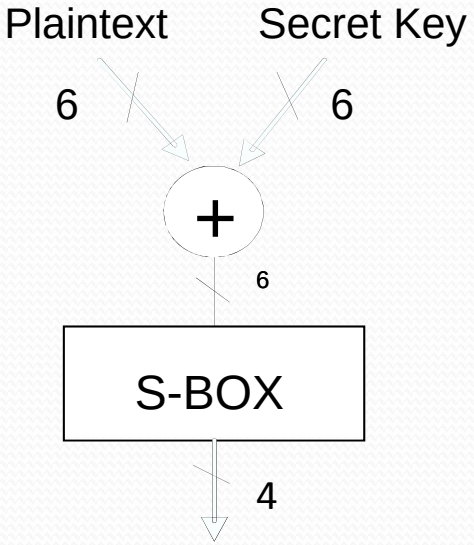
- Separate communication protocol from data computation

  - Step 1 : compute outputs without considering the communication protocol.
  - Step 2 : add the communication protocol to the circuit
  - Step 3 : countermeasures are added to balance - logically and electrically - the circuit.

Experimental results

- Mapping a sensitive DES* sub-module, in dual rail 4-phase and 2-phase protocol:

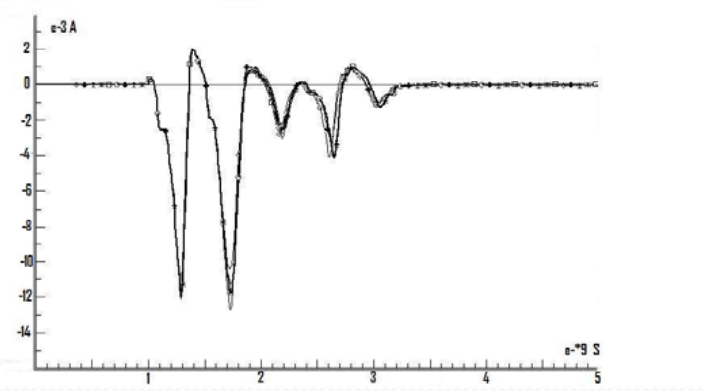
	2-phase implementation	4-phase implementation
Nb of PLB - SBOX	7	6
Nb of cycle / Data exchange	2	4



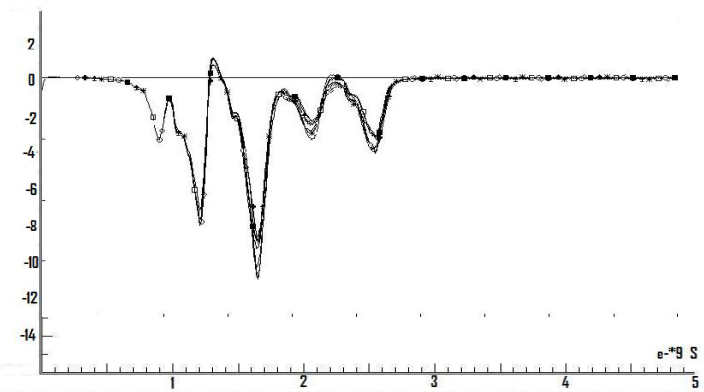
* *DES:* Data Encryption standard

Experimental results

- **Current profiles for 2-phase and 4-phase protocol**



4-phase implementation

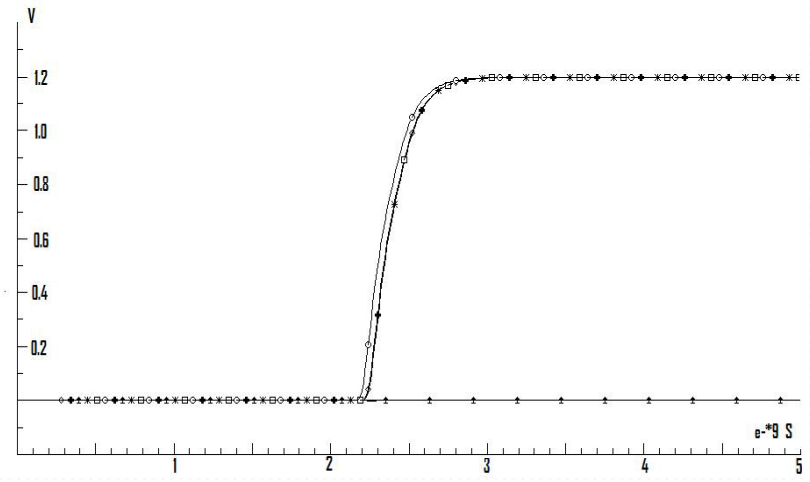


2-phase implementation

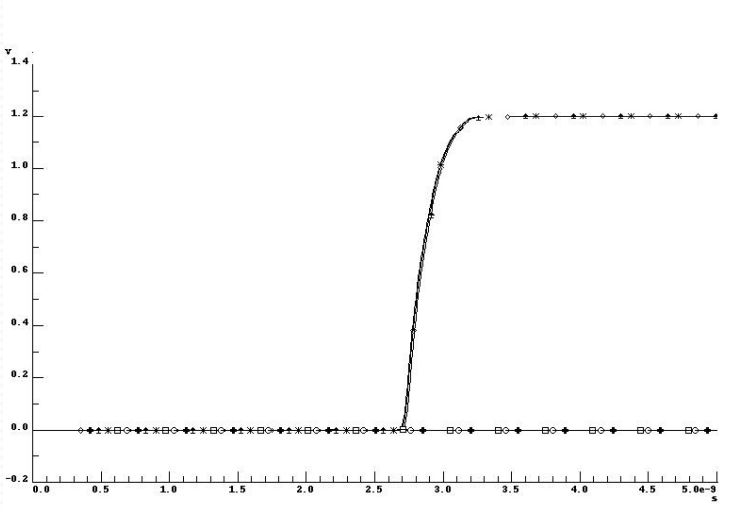
The Power consumption is data independent

Experimental results

- **Outputs for 2-phase and 4-phase protocol**



4-phase implementation

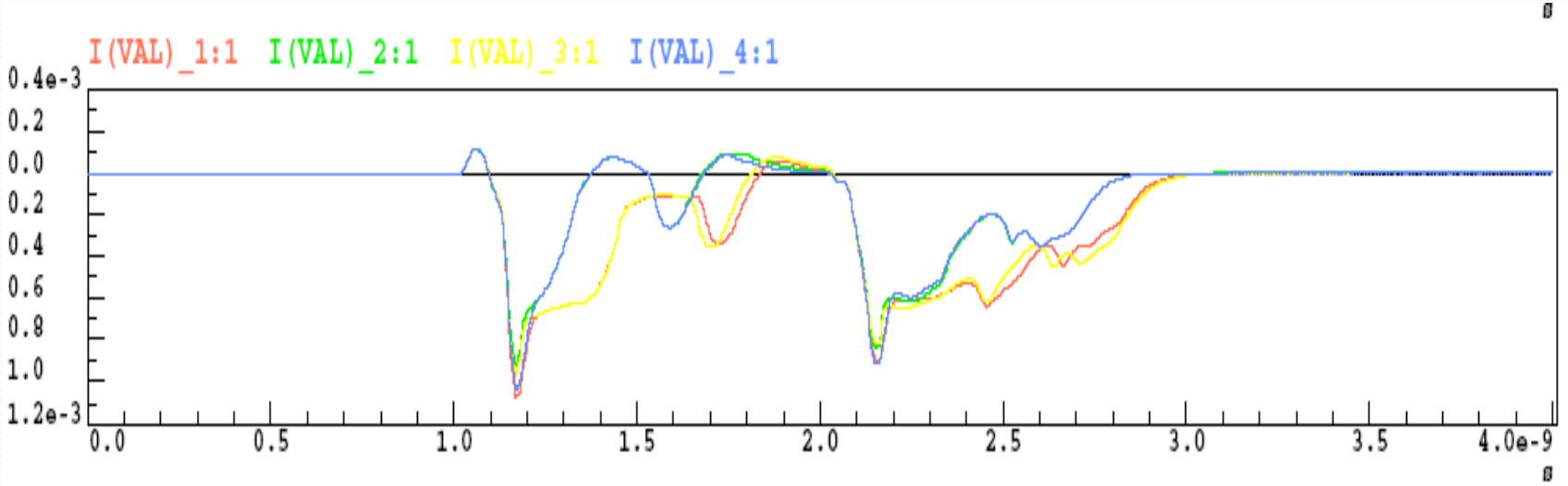


2-phase implementation

The same running time

Experimental results

- Consumption of the same sub-module implemented without countermeasures.



Conclusion

- Flexible FPGA
  - Many countermeasures
  - 2-phase and 4 phase protocol
- Robust against power-based attacks
  - Asynchronous intrinsic robustness
  - 1-of-n encoding
  - Electrically balanced (Balanced input capacitances)
  - Logically balanced (Same logical depth for all inputs)
- 2-phase and 4-phase SBOX implementation:
  - Almost the same PLB cost
  - The communication in 2 phase is twice faster than the 4-phase