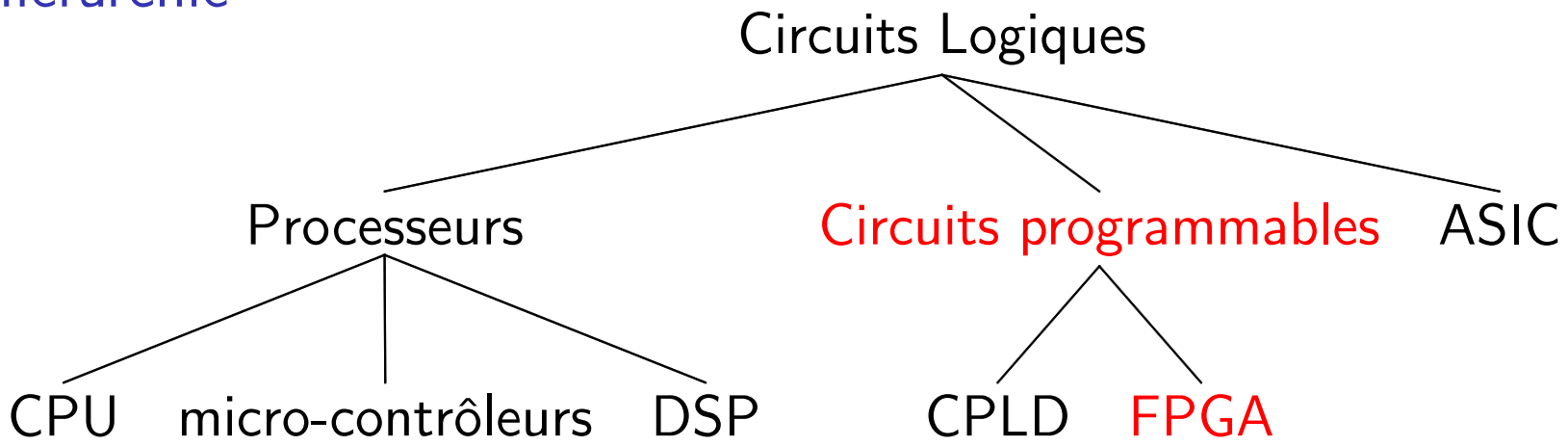
Introduction aux FPGA

Mickaël Dardaillon

M2RTS

16/11/2011

Hiérarchie



Définitions

- CPU : Computer Processing Unit
- DSP : Digital Signal Processor
- CPLD : Complex Programmable Logic Device
- FPGA : Field Programmable Gate Array
- ASIC : Application Specific Integrated Circuit

ASIC

- Développement long
- Coût de fabrication (en augmentation)
- Full custom : Performances maximales
- Fabrication grande série

FPGA

- Développement rapide
- Coût à l'unité (en diminution)
- Contraint par la technologie du FPGA
- Prototypage rapide

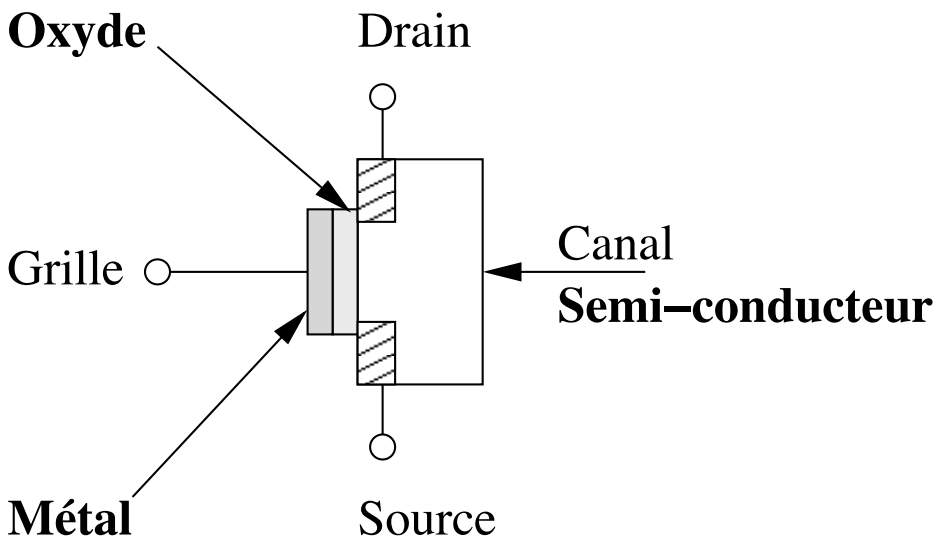
Sommaire

Sommaire

Le transistor MOS

MOS : Metal Oxide Semiconductor

- Composant électronique de base
- Porte logique ON/OFF

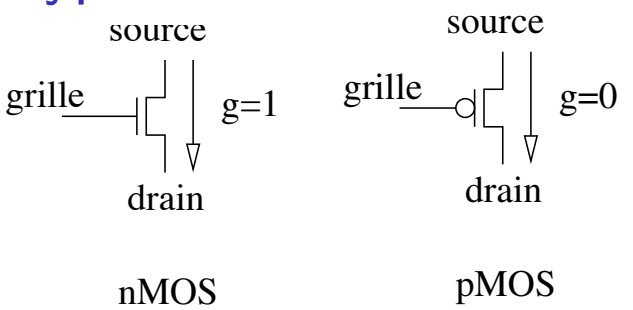


Technologie CMOS

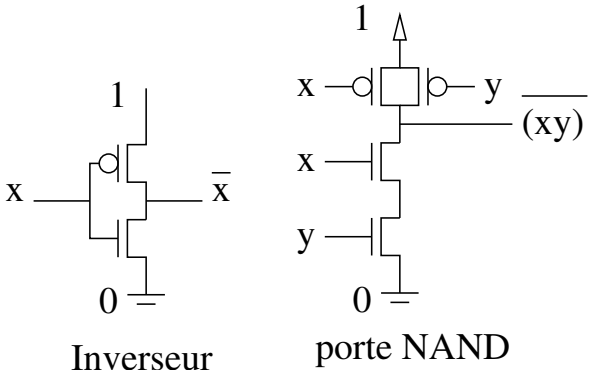
CMOS : Complementary MOS

- Niveaux logiques
  - 0 = 0V
  - 1 = 3V
- Deux types de MOS
  - nMOS : conducteur si la grille=1
  - pMOS : conducteur si la grille=0
- Puissance dissipée
  - $P \propto CV^2 f$
  - C : Capacité d'un MOS
  - V : Tension sur la grille
  - f : Fréquence de fonctionnement

Types de MOS

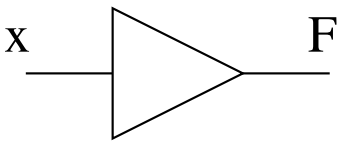


Exemples



Portes élémentaires

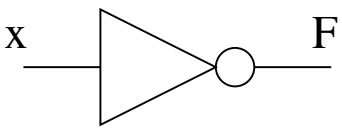
Amplificateur :



$F = x$

x	F
0	0
1	1

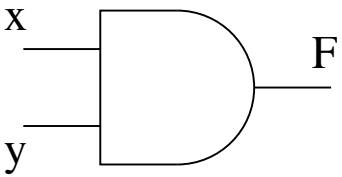
Inverseur :



$F = \overline{x}$

x	F
0	1
1	0

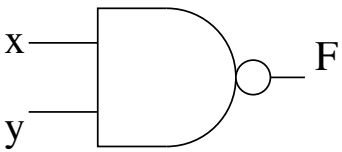
ET :



$F = x\ y$

x	y	F
0	0	0
0	1	0
1	0	0
1	1	1

NON ET :



$F = \overline{x\ y}$

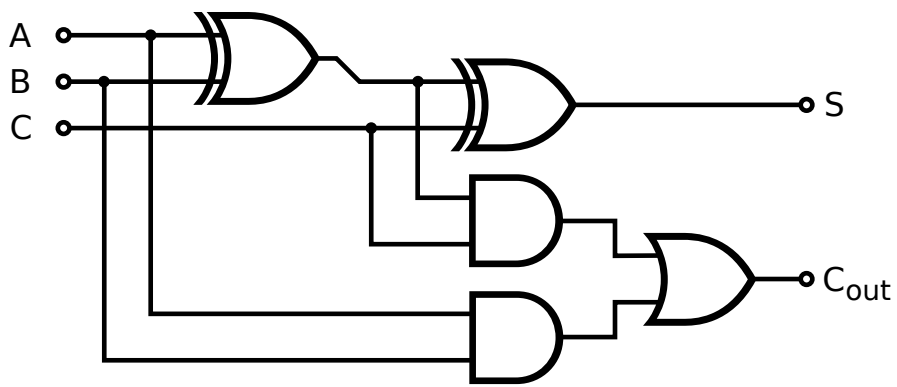
x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

Conception de circuit combinatoire

1. Description du problème : Addition entre deux bits a et b et une retenue c
2. Table de vérité :

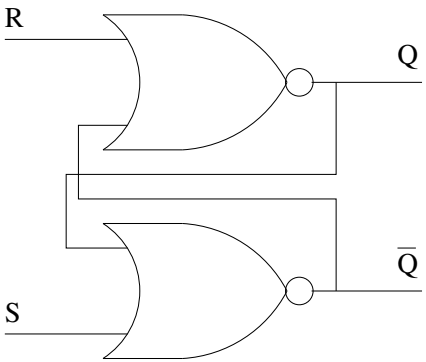
entrées			sorties	
a	b	c	S	c_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

3. Équations logiques :
   $S = \overline{a}\overline{b}c + \overline{a}b\overline{c} + a\overline{b}\overline{c} + abc$
   $c_{out} = \overline{a}bc + a\overline{b}c + ab\overline{c} + abc$
4. Équations simplifiées :
   $S = (a \oplus b) \oplus c$
   $c_{out} = ab + (a \oplus b)c$
5. Portes logiques :



Logique séquentielle

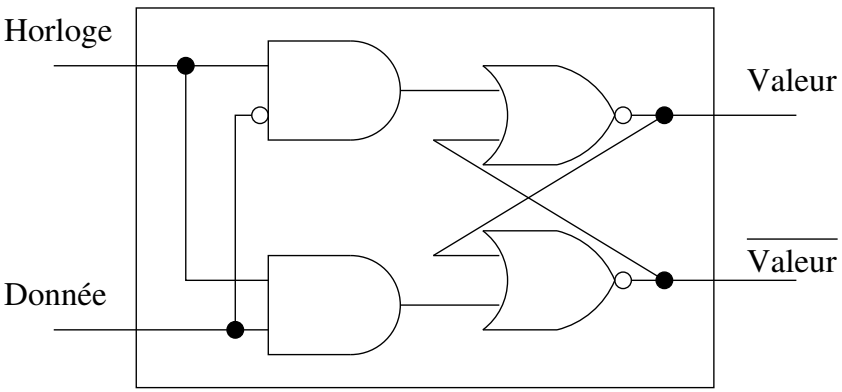
Bascule RS

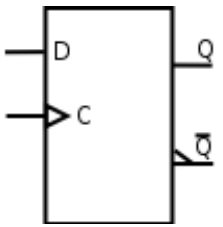


Bascule RS

S	R	Q	$\overline{Q}$
0	1	0	1
1	1	interdit	interdit
1	0	1	0
0	0	Q_{n-1}	$\overline{Q_{n-1}}$

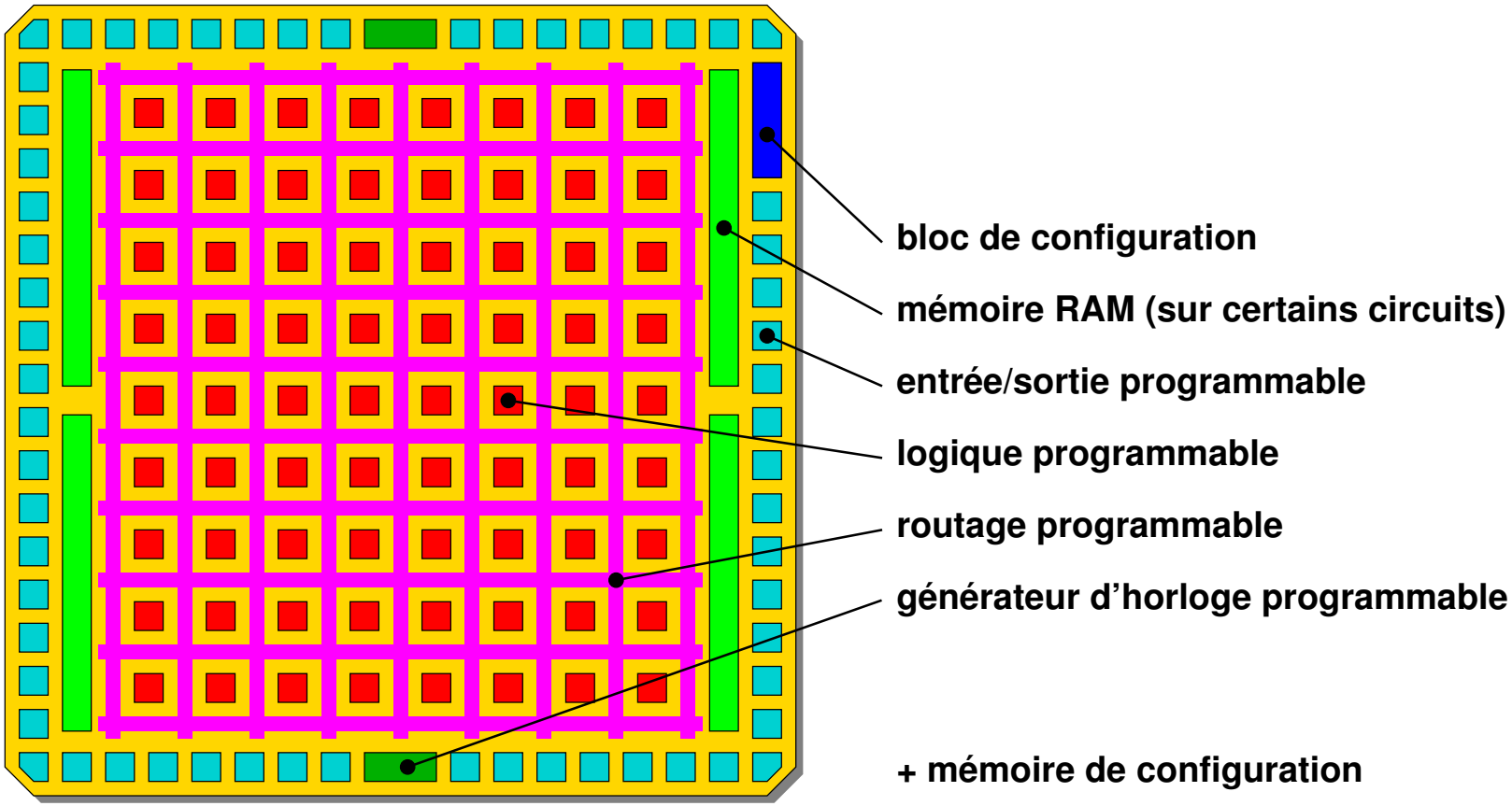
Bascule D : mémorise D lorsque l'horloge passe à '1'



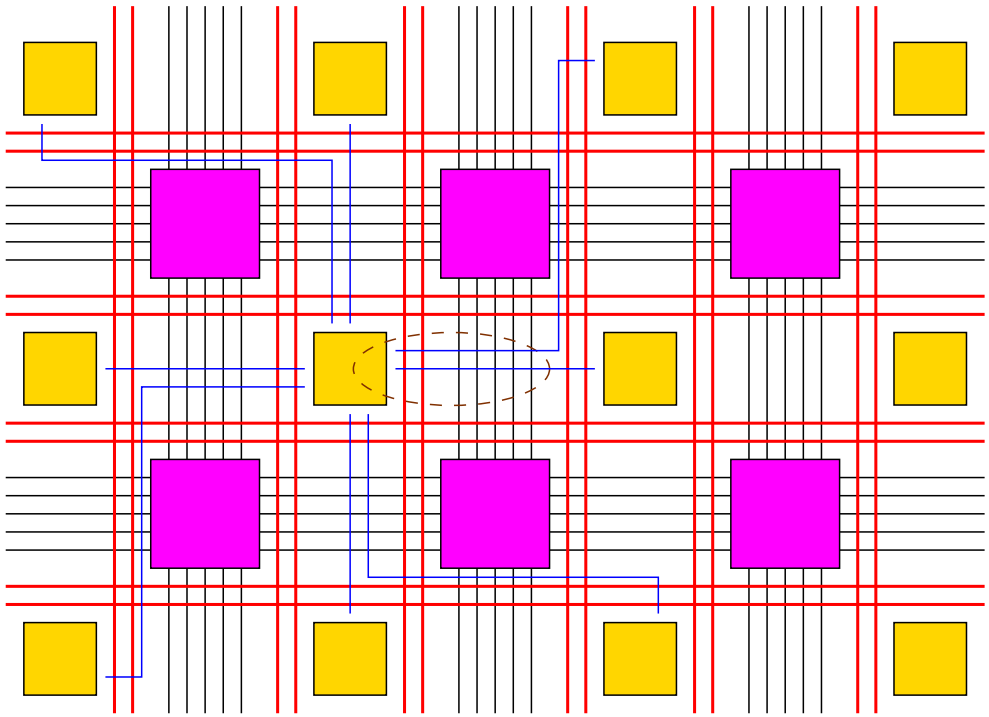


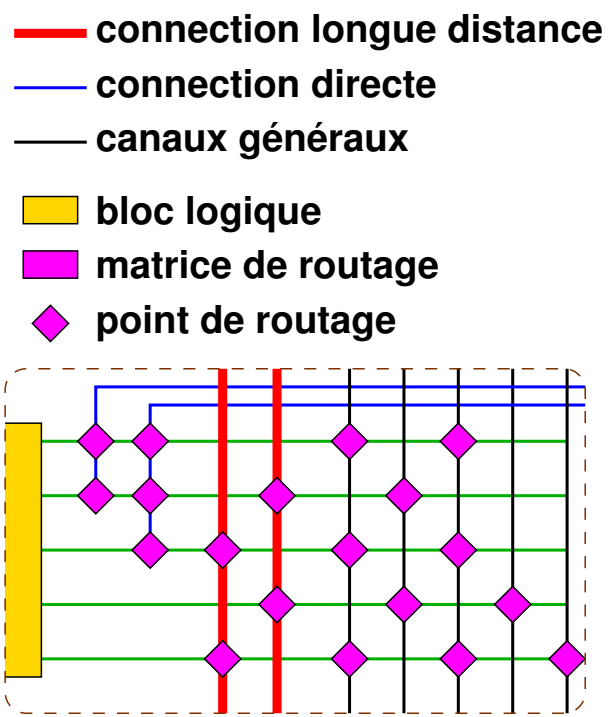
Sommaire

Ressources globales



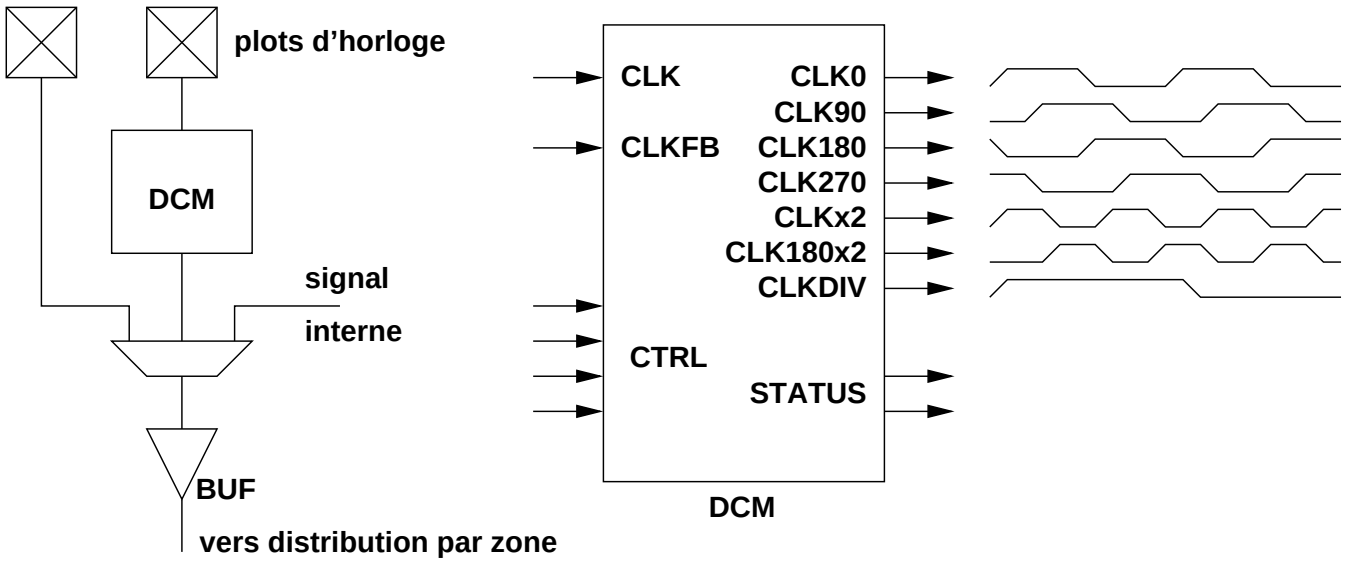
Routage des signaux





Génération d'horloge

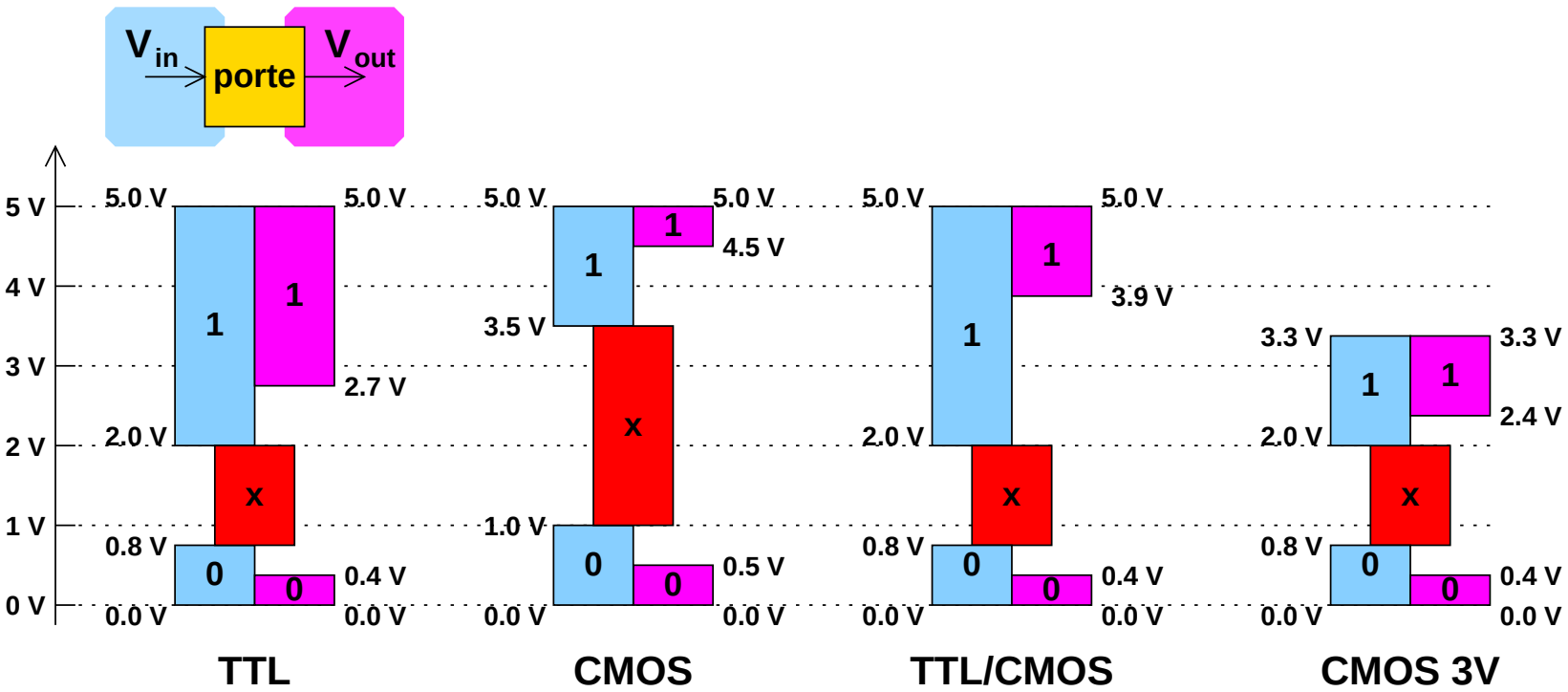
Les horloges sont distribuées dans l'ensemble du FPGA par des circuits de routage spécifique, aucun travail de la part du concepteur !



Entrées / Sorties

Il existe plusieurs niveaux de tensions d'entrée / sortie, qui sont à prendre en compte pour :

- La conception de la carte (compatibilité entre composants)
- Les contraintes de routage (technologie à utiliser sur I/O)



Composants logiques

Chaque constructeur définit sa propre cellule de base, qui varie selon les familles de FPGA.

Les cellules élémentaires :

- Portes logiques configurables : fonctions de base (AND, OR...)
- Registres : mémoire et synchronisme

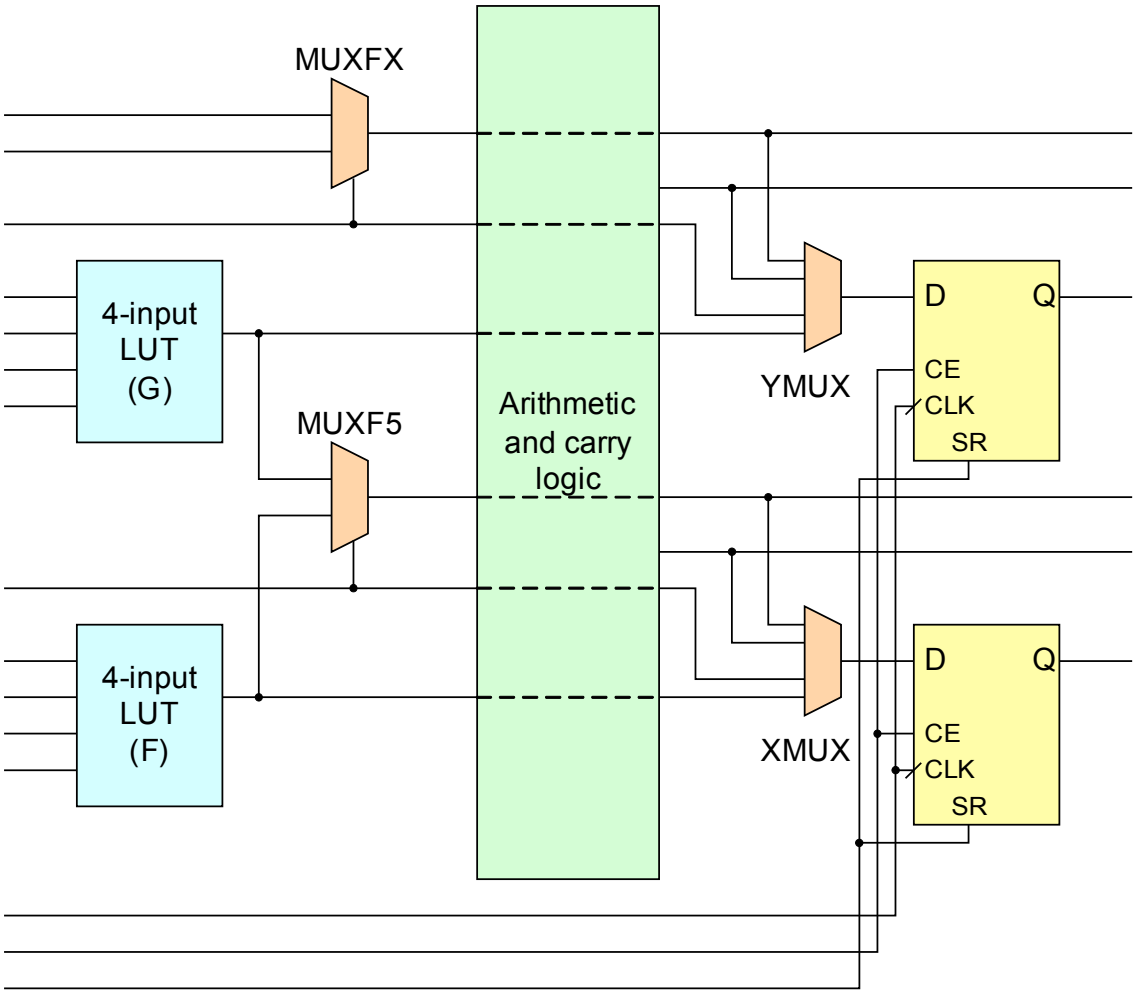
Les composants matériels (hardcore) :

- Logique complexe : additionneurs, multiplieurs...
- Mémoire RAM
- Processeurs (ARM, PowerPC...)

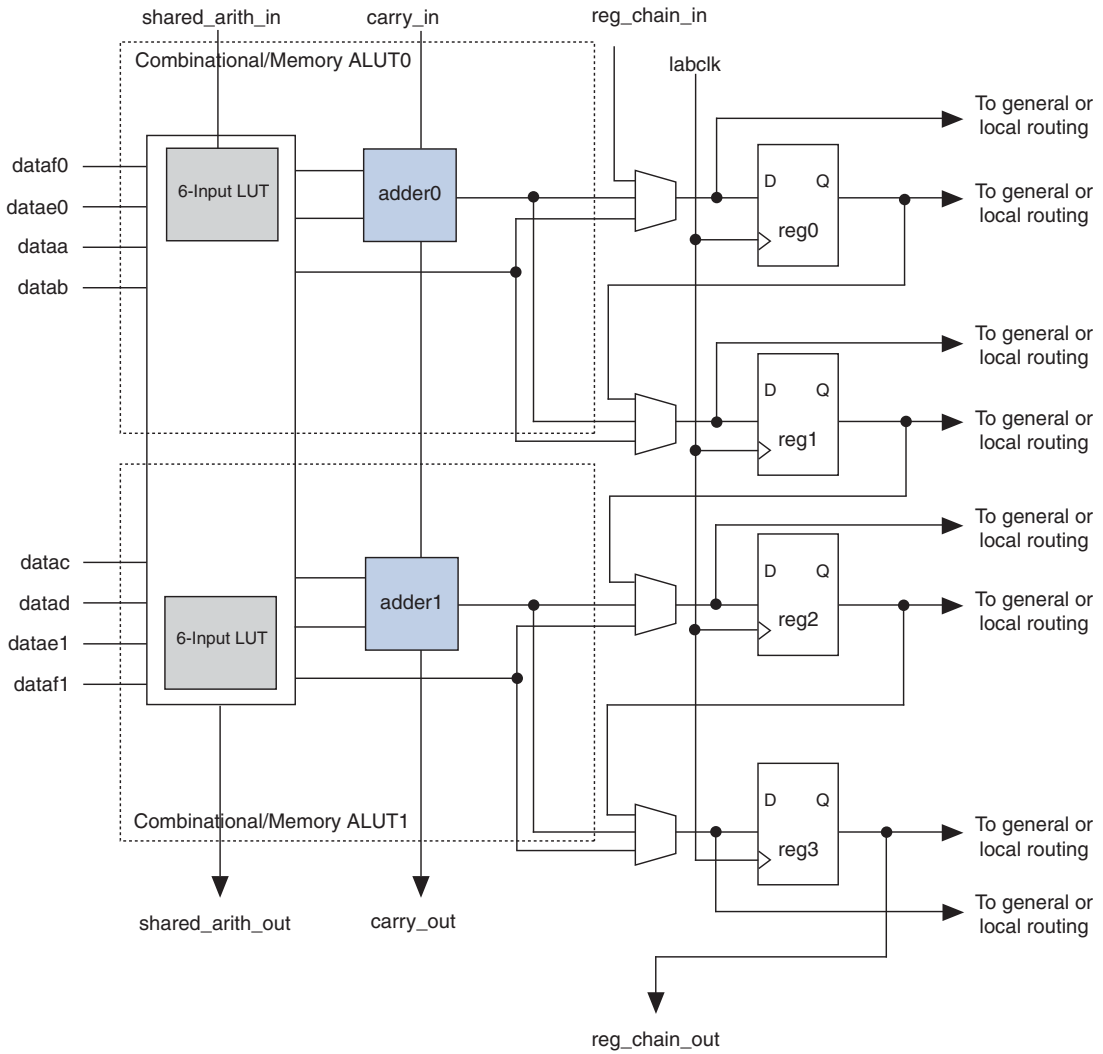
Les composants logiciels (softcore) :

- UART, USB, contrôleur d'écran...
- Processeurs (ARM, RISC...)
- FFT, JPEG, SHA-1...
- ...

Xilinx Virtex 4 : Slice

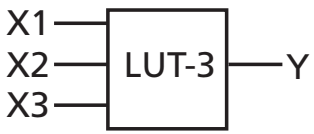


Altera Stratix 5 : ALM (Adaptative Logic Module)

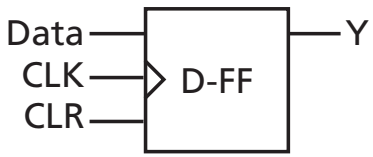


Actel Pro Asic 3 : VersaTile

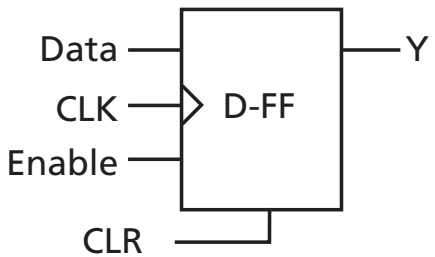
LUT-3 Equivalent



D-Flip-Flop with Clear or Set



Enable D-Flip-Flop with Clear or Set



Composants logiques

Chaque constructeur définit sa propre cellule de base, qui varie selon les familles de FPGA.

Les cellules élémentaires :

- Portes logiques configurables : fonctions de base (AND, OR...)
- Registres : mémoire et synchronisme

Les composants matériels (hardcore) :

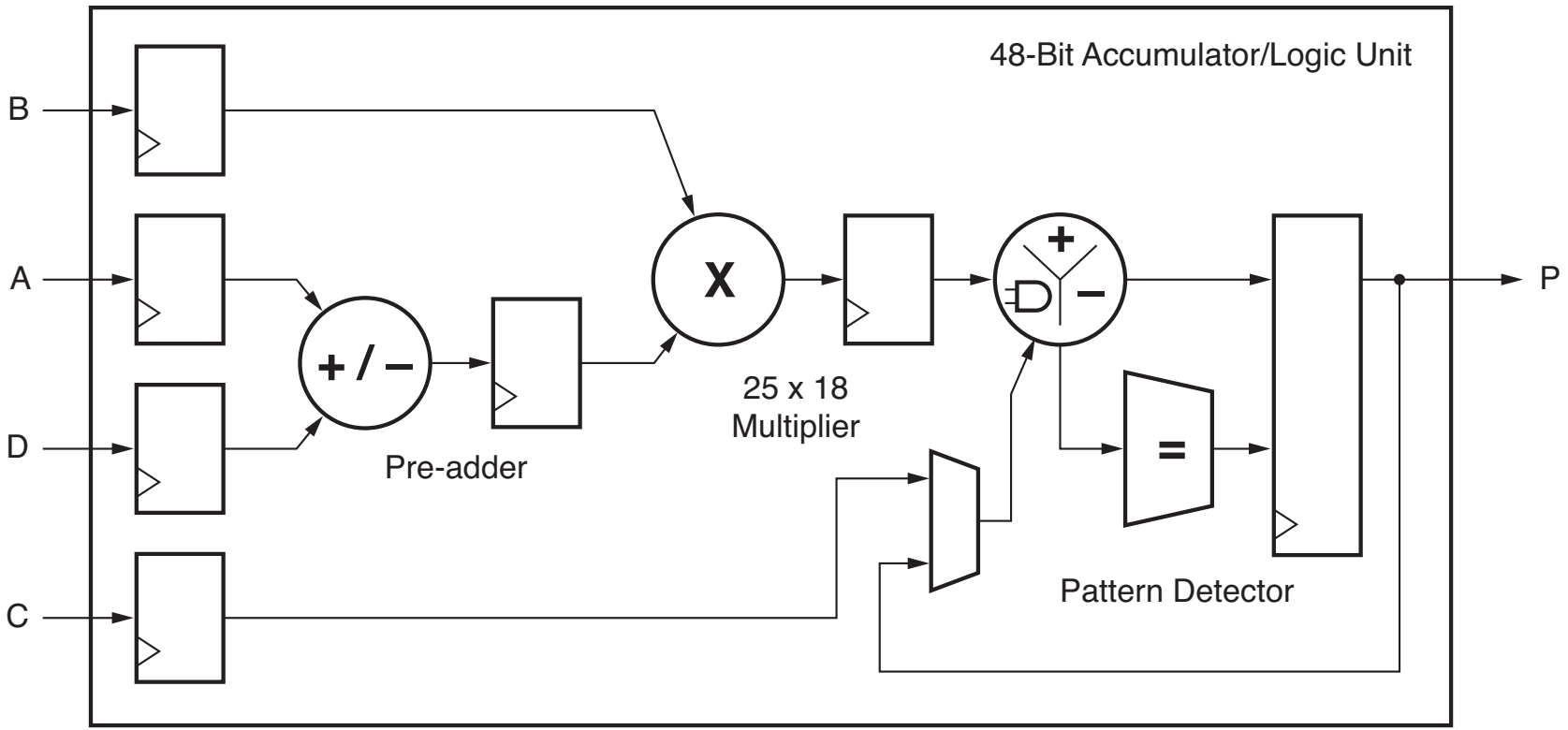
- Logique complexe : additionneurs, multiplieurs...
- Mémoire RAM
- Processeurs (ARM, PowerPC...)

Les composants logiciels (softcore) :

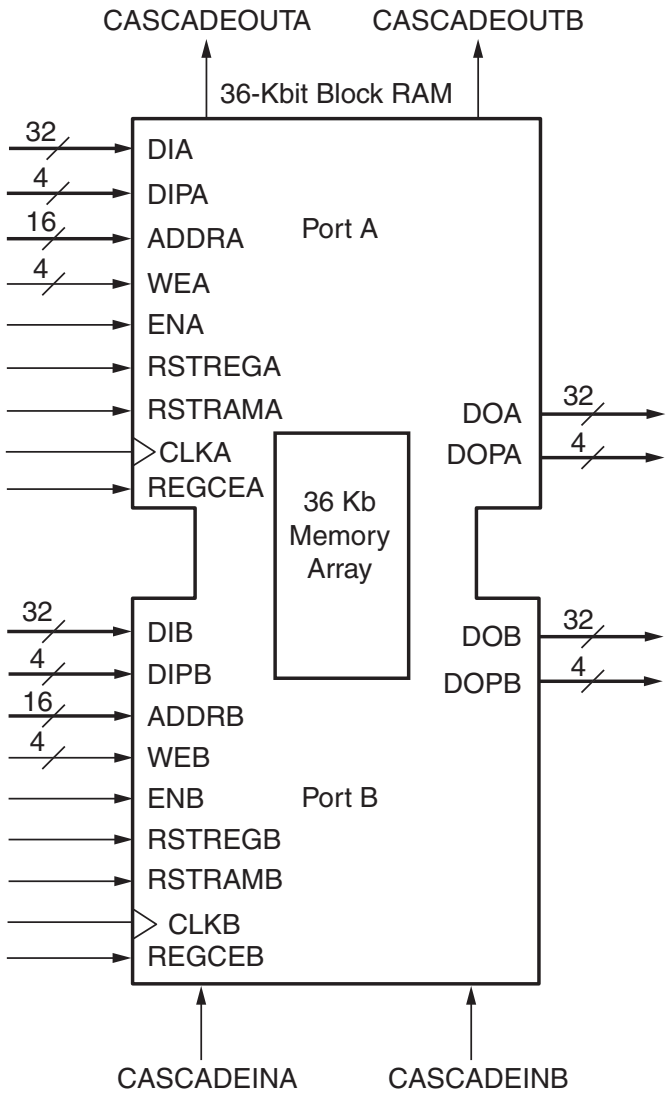
- UART, USB, contrôleur d'écran...
- Processeurs (ARM, RISC...)
- FFT, JPEG, SHA-1...
- ...

DSP Slice pour Virtex 4

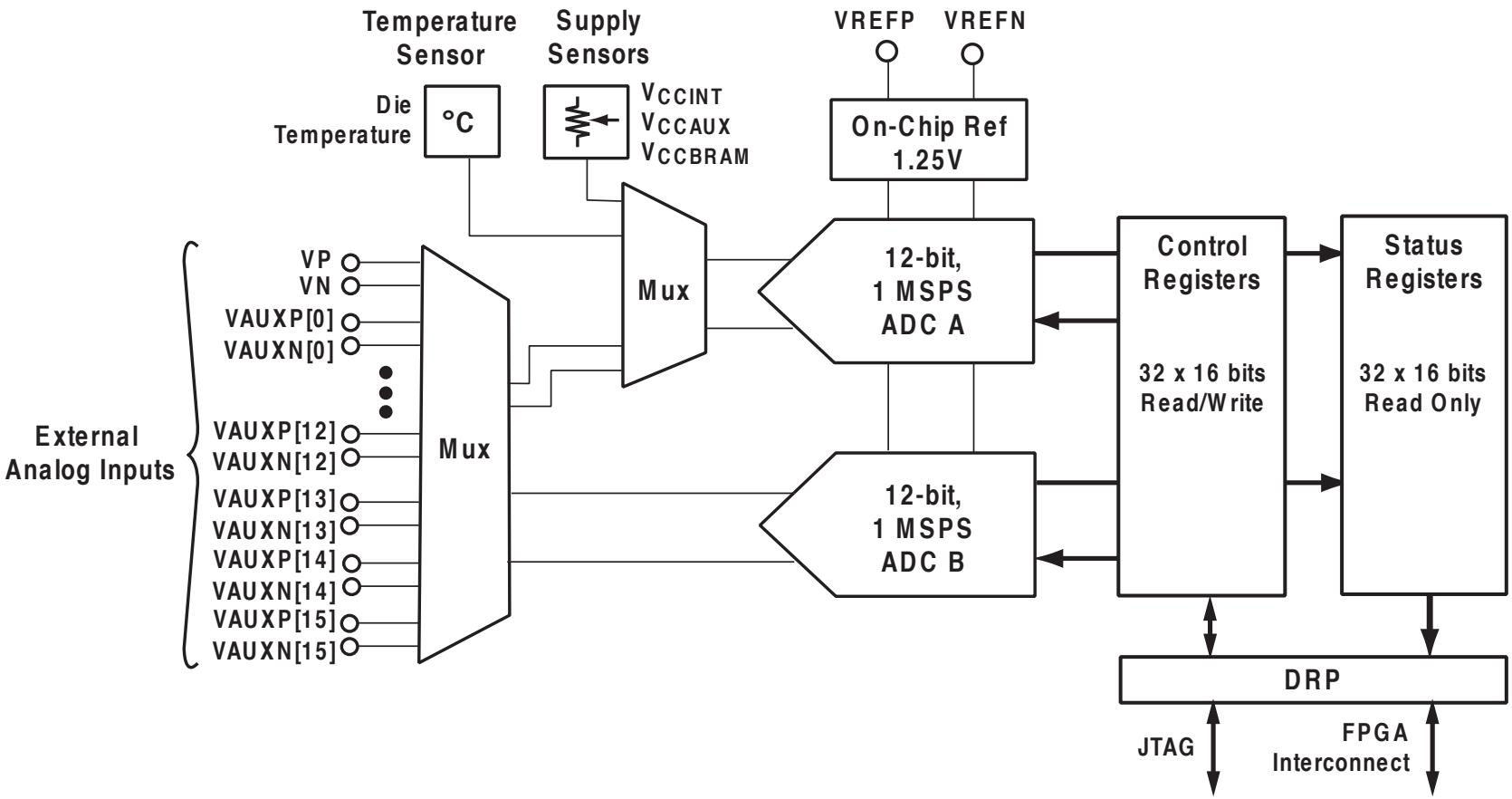
Entrées 18 bits avec multiplication / accumulation sur 32 échantillons



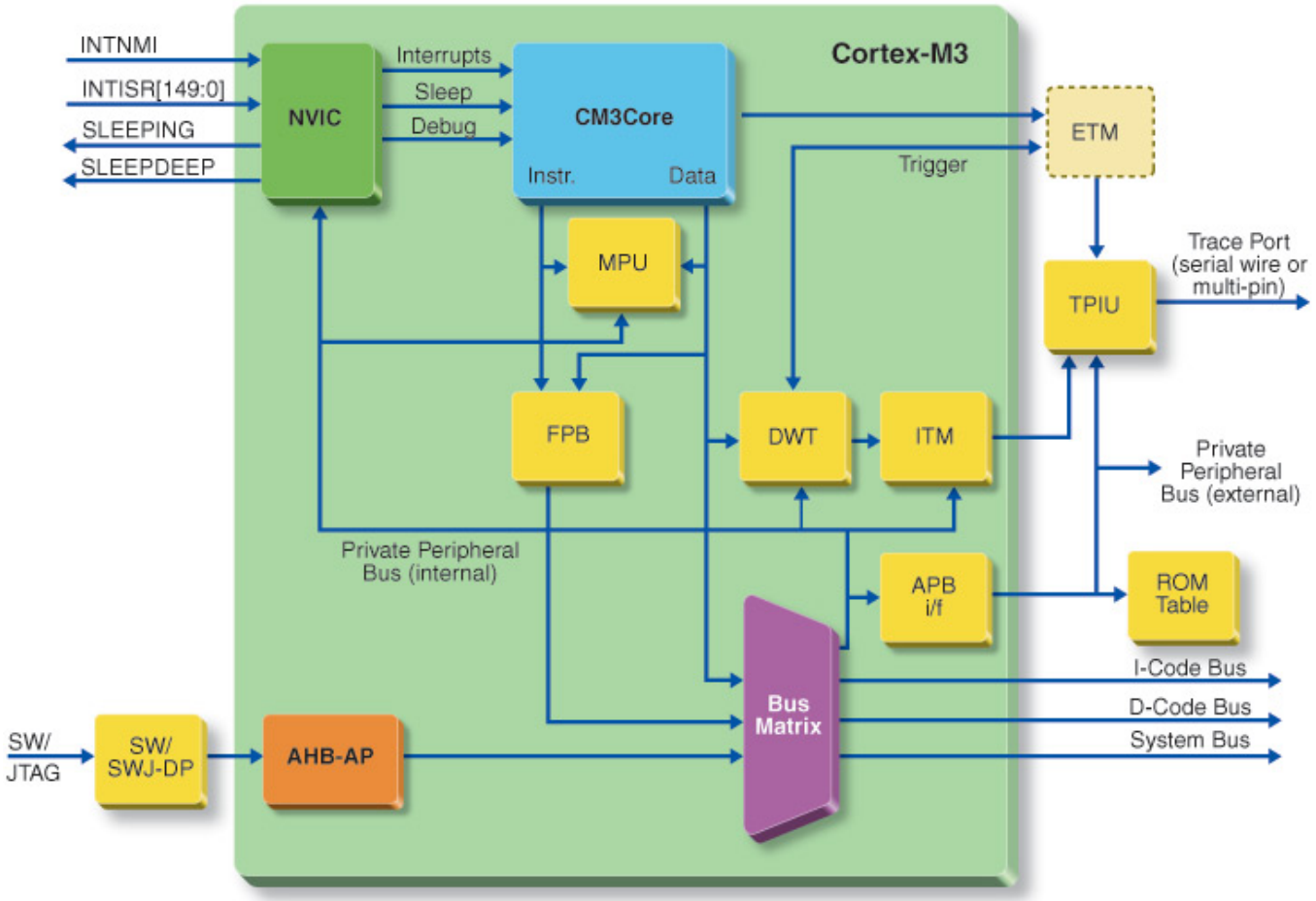
RAM double port



ADC


Temperature Sensor
Supply Sensors
VREFP
VREFN
Die Temperature
°C
VCCINT
VCCAUX
VCCBRAM
On-Chip Ref 1.25V
VP
VN
VAUXP[0]
VAUXN[0]
VAUXP[12]
VAUXN[12]
VAUXP[13]
VAUXN[13]
VAUXP[14]
VAUXN[14]
VAUXP[15]
VAUXN[15]
External Analog Inputs
Mux
Mux
12-bit, 1 MSPS ADC A
12-bit, 1 MSPS ADC B
Control Registers
32 x 16 bits Read/Write
Status Registers
32 x 16 bits Read Only
DRP
JTAG
FPGA Interconnect

Processeur ARM sur SmartFusion



Composants logiques

Chaque constructeur définit sa propre cellule de base, qui varie selon les familles de FPGA.

Les cellules élémentaires :

- Portes logiques configurables : fonctions de base (AND, OR...)
- Registres : mémoire et synchronisme

Les composants matériels (hardcore) :

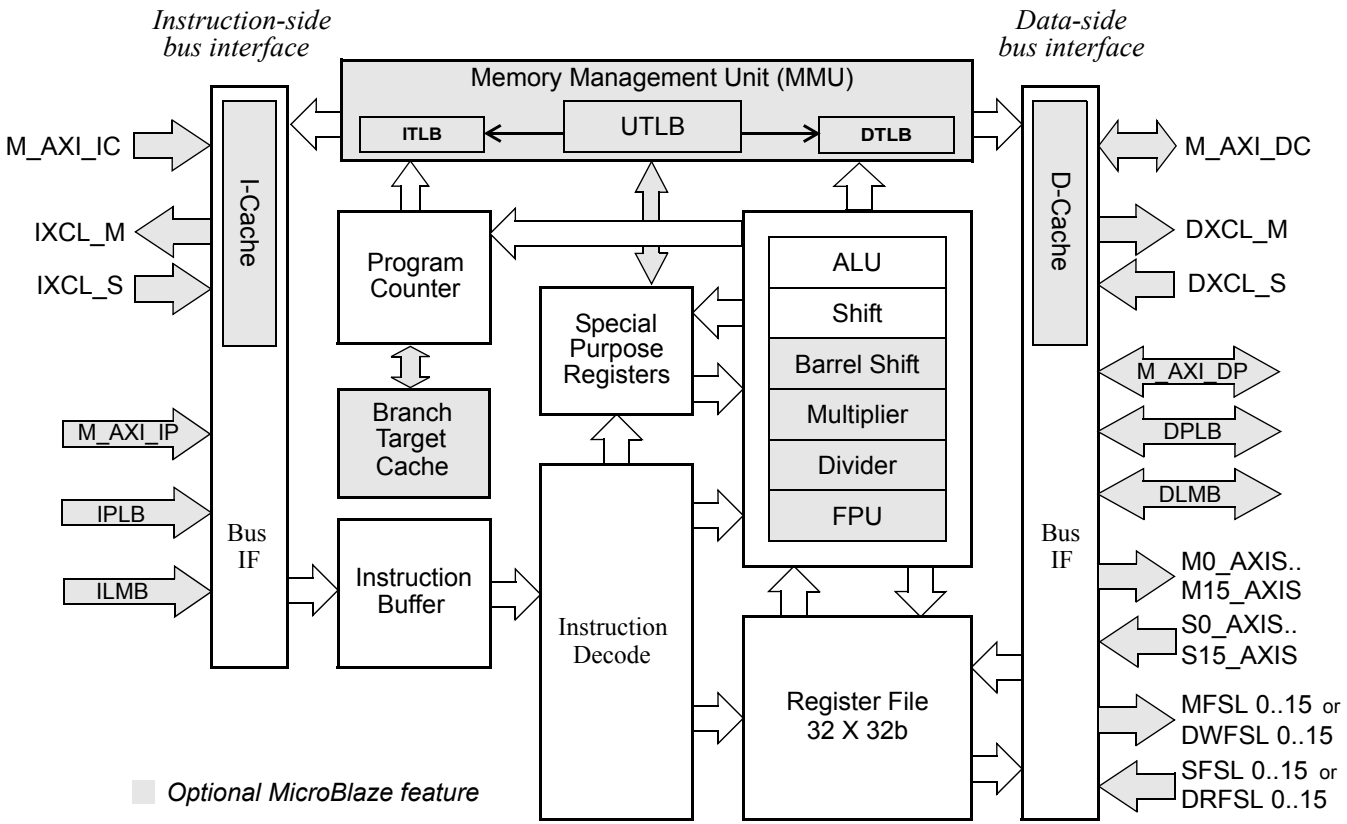
- Logique complexe : additionneurs, multiplieurs...
- Mémoire RAM
- Processeurs (ARM, PowerPC...)

Les composants logiciels (softcore) :

- UART, USB, contrôleur d'écran...
- Processeurs (ARM, RISC...)
- FFT, JPEG, SHA-1...
- ...

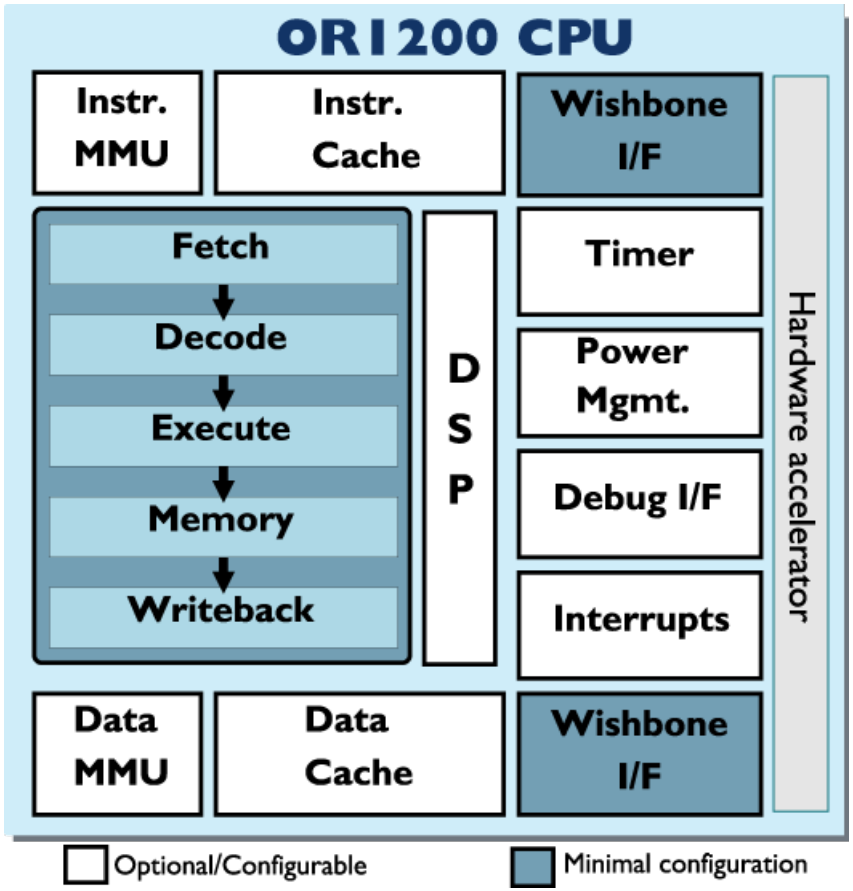
Xilinx MicroBlaze (propriétaire)

- 5.1K LUTs at 241MHz on Xilinx Virtex 6
- 2.4K LUTs at 131MHz on Xilinx Spartan 6



Processeur OpenRISC (open-source)

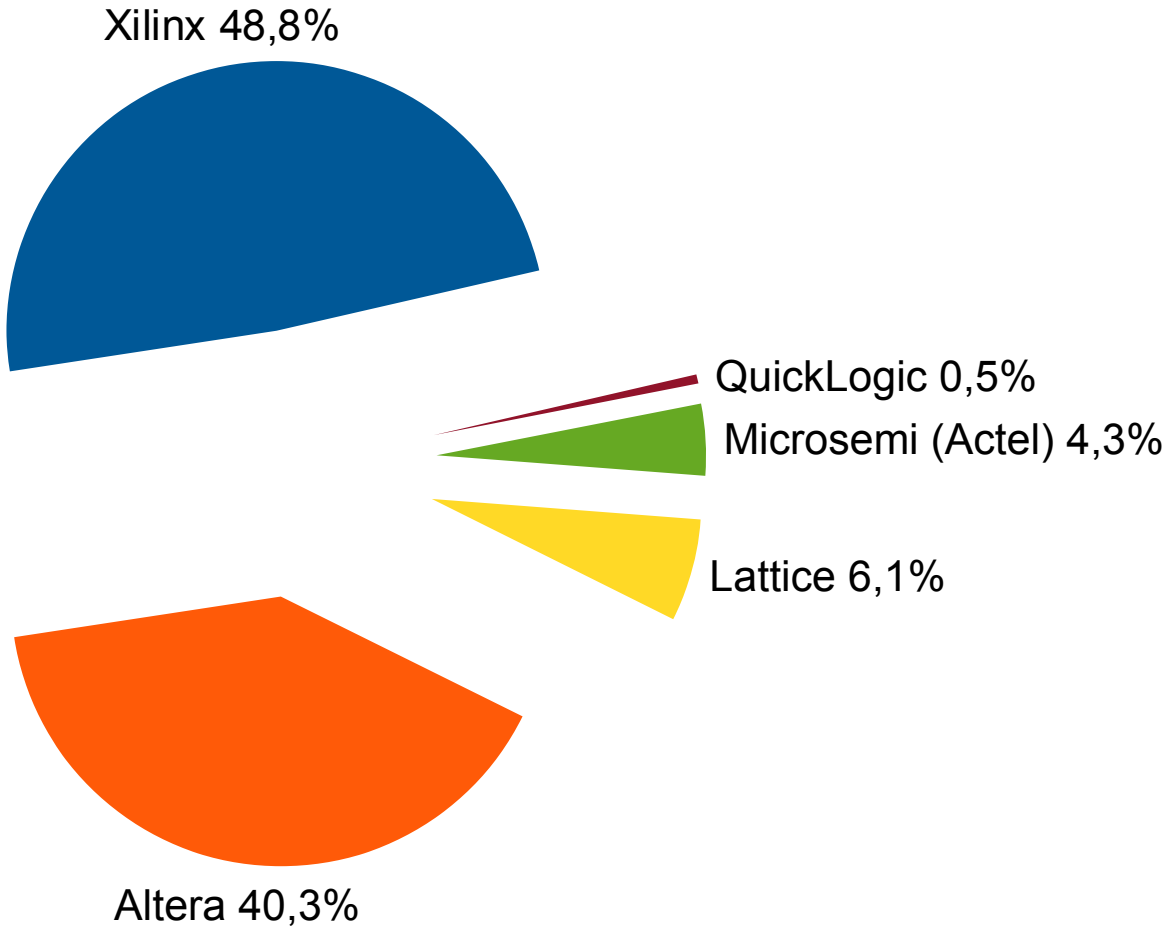
- 7K core cells, 4 block RAMs at 35MHz on Actel ProASIC3
- 2.4K LUTs, 1 block RAM at 125MHz on Xilinx Virtex 5



Sommaire

Les constructeurs

Part de marché en termes de CA en 2010



Technologie

SRAM

- Xilinx
- Altera
- Lattice

Flash

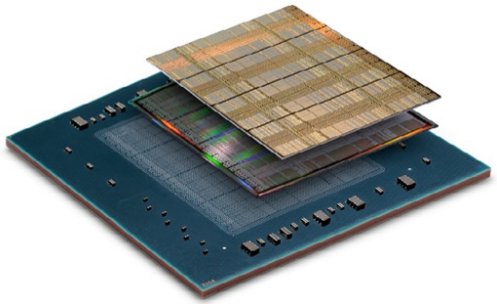
- Actel
- QuickLogic

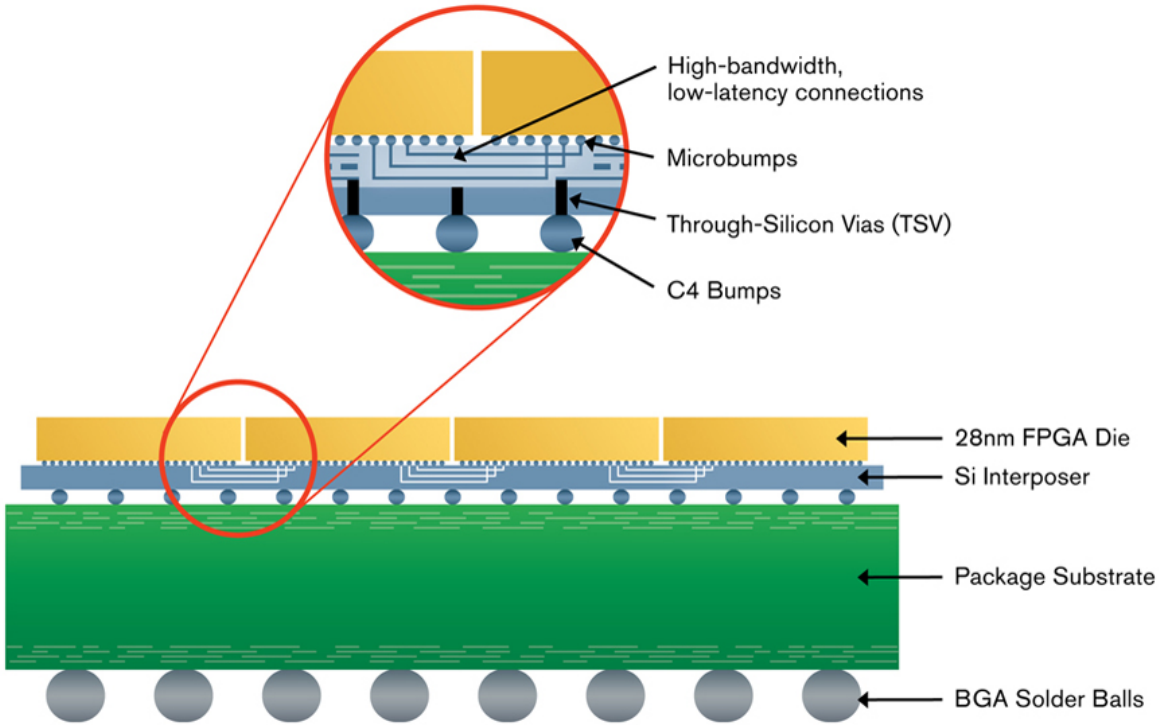
Familles

	Xilinx	Altera	Actel
Hautes performances	Virtex	Stratix	
Rentable	Kintex (ex Spartan)	Arria	Pro Asic 3
Économie d'énergie	Artix	Cyclone	Igloo

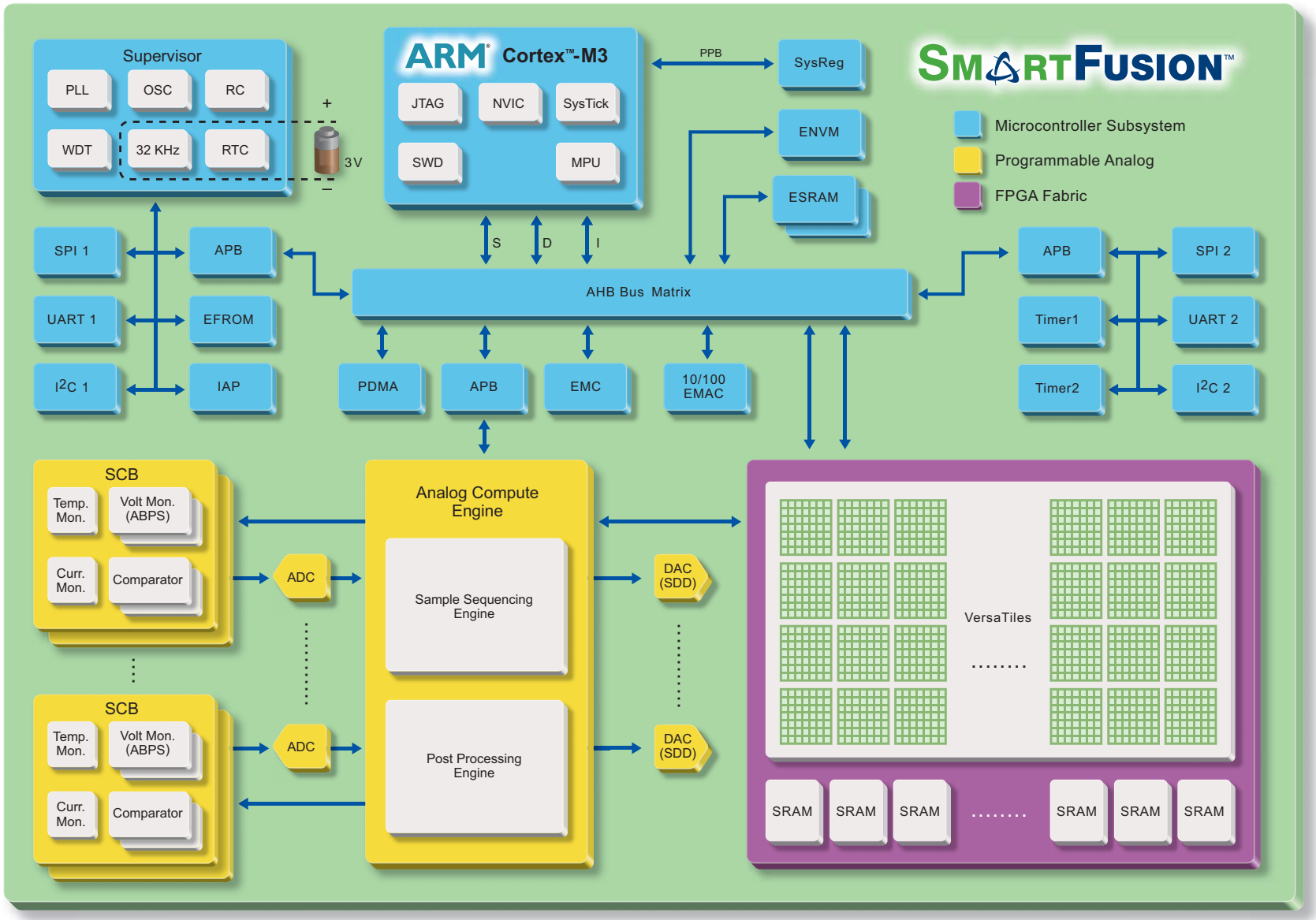
Xilinx Virtex 7

- 28nm Process
- 6,8 Milliards de transistors (intel core i7 : 995 Millions)





Actel SmartFusion



Sommaire

Flot de conception (1)

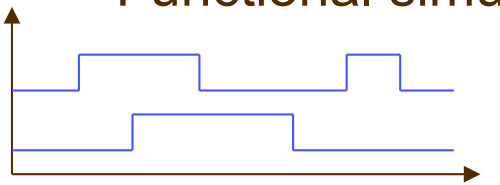
Design and implement a simple unit permitting to speed up encryption with RC5-similar cipher with fixed key set on 8031 microcontroller. Unlike in the experiment 5, this time your unit has to be able to perform an encryption algorithm by itself, executing 32 rounds…..

Specification (Lab Experiments)

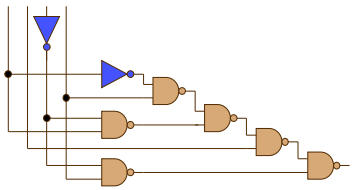
```
Library IEEE;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity RC5_core is
        port(
            clock, reset, encr_decr: in std_logic;
            data_input: in std_logic_vector(31 downto 0);
            data_output: out std_logic_vector(31 downto 0);
            out_full: in std_logic;
            key_input: in std_logic_vector(31 downto 0);
            key_read: out std_logic;
        );
end AES_core;
```

VHDL description (Your Source Files)

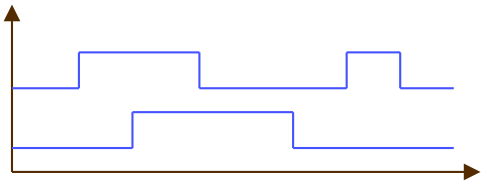
Functional simulation



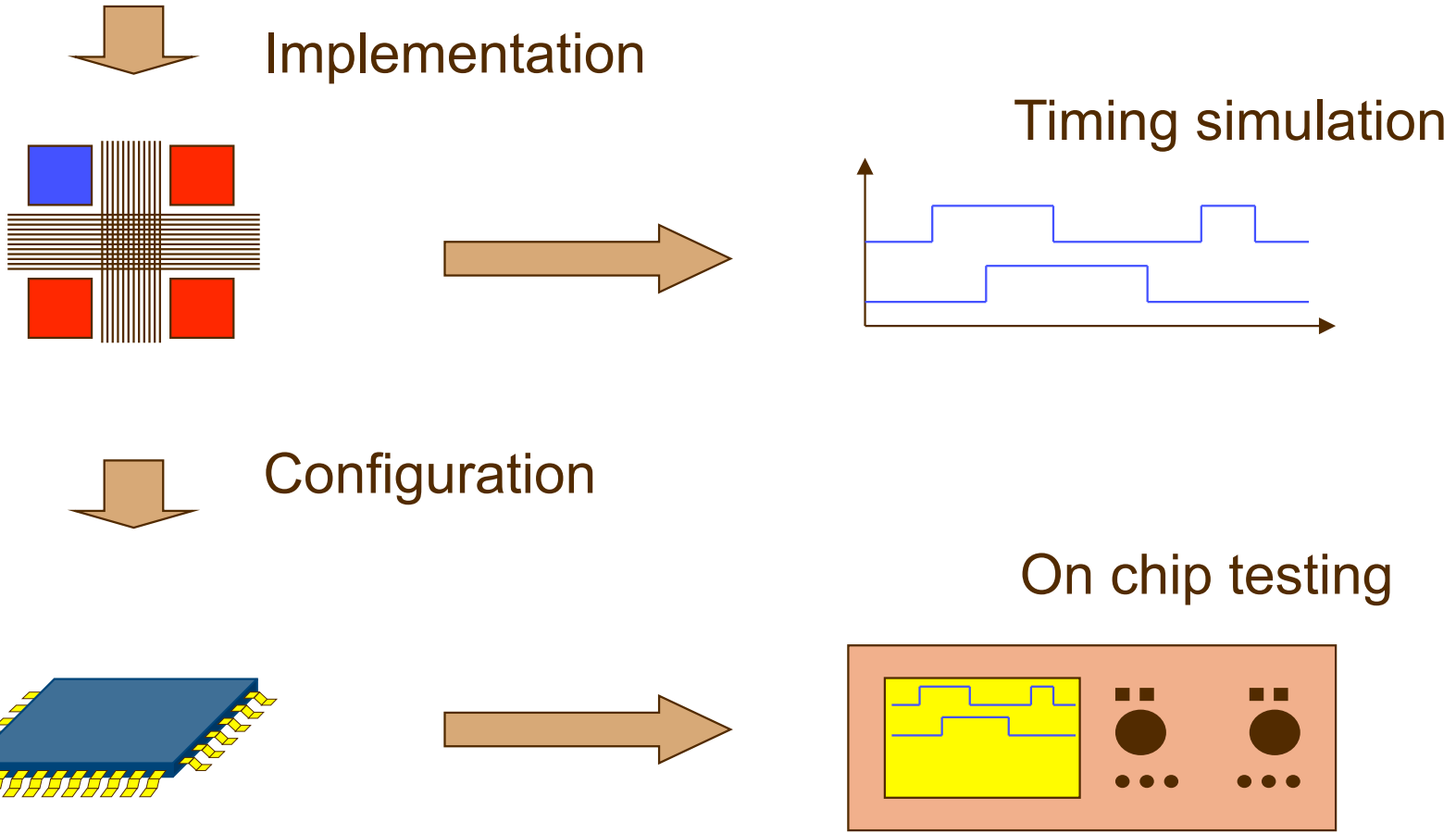
Synthesis



Post-synthesis simulation



Flot de conception (2)



Environnement de développement

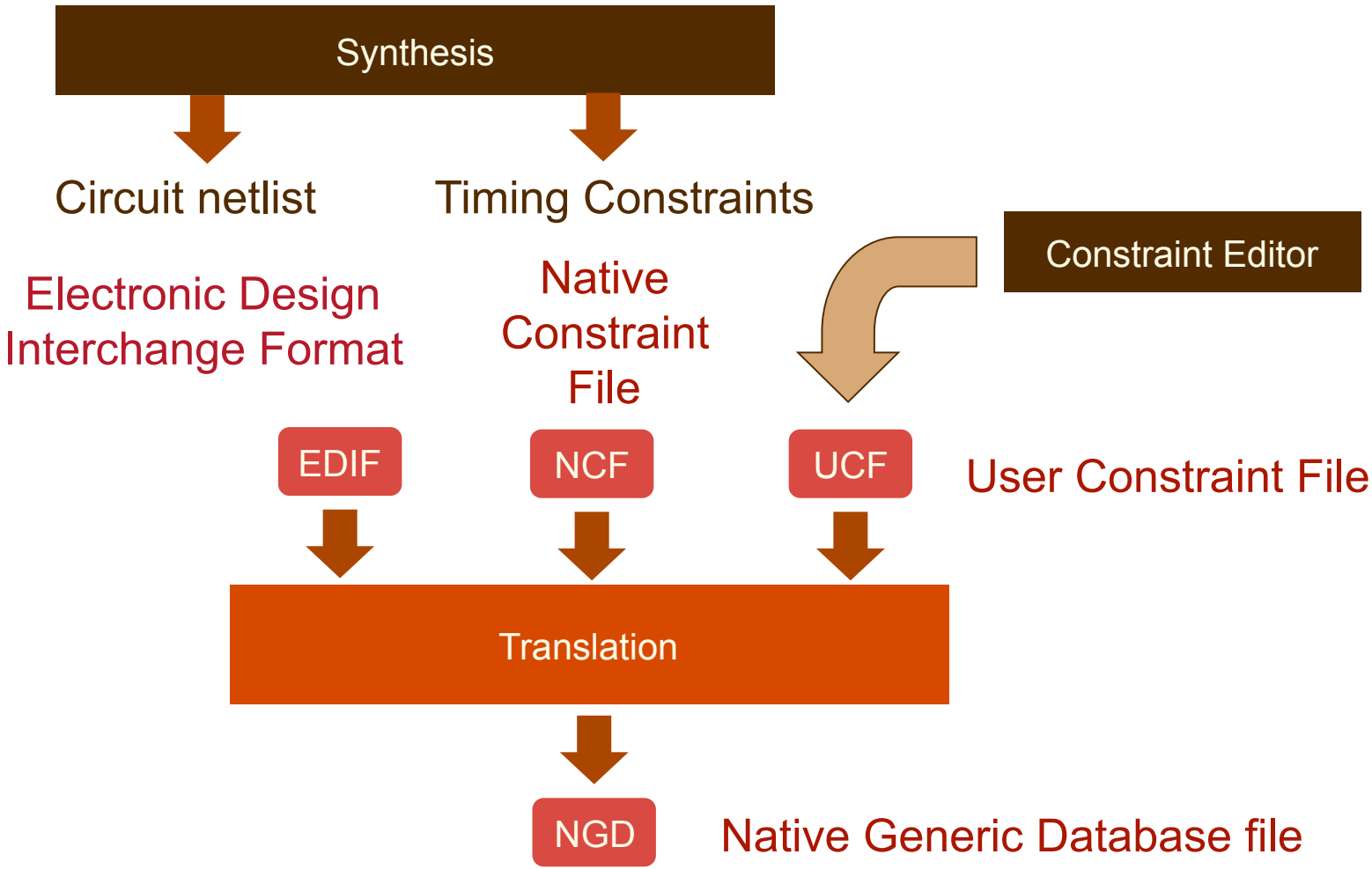
Synthèse

- Actel Libero
- Altera Quartus II
- Synopsys Synplify
- Xilinx ISE
- ...

Simulation

- ModelSim
- ...

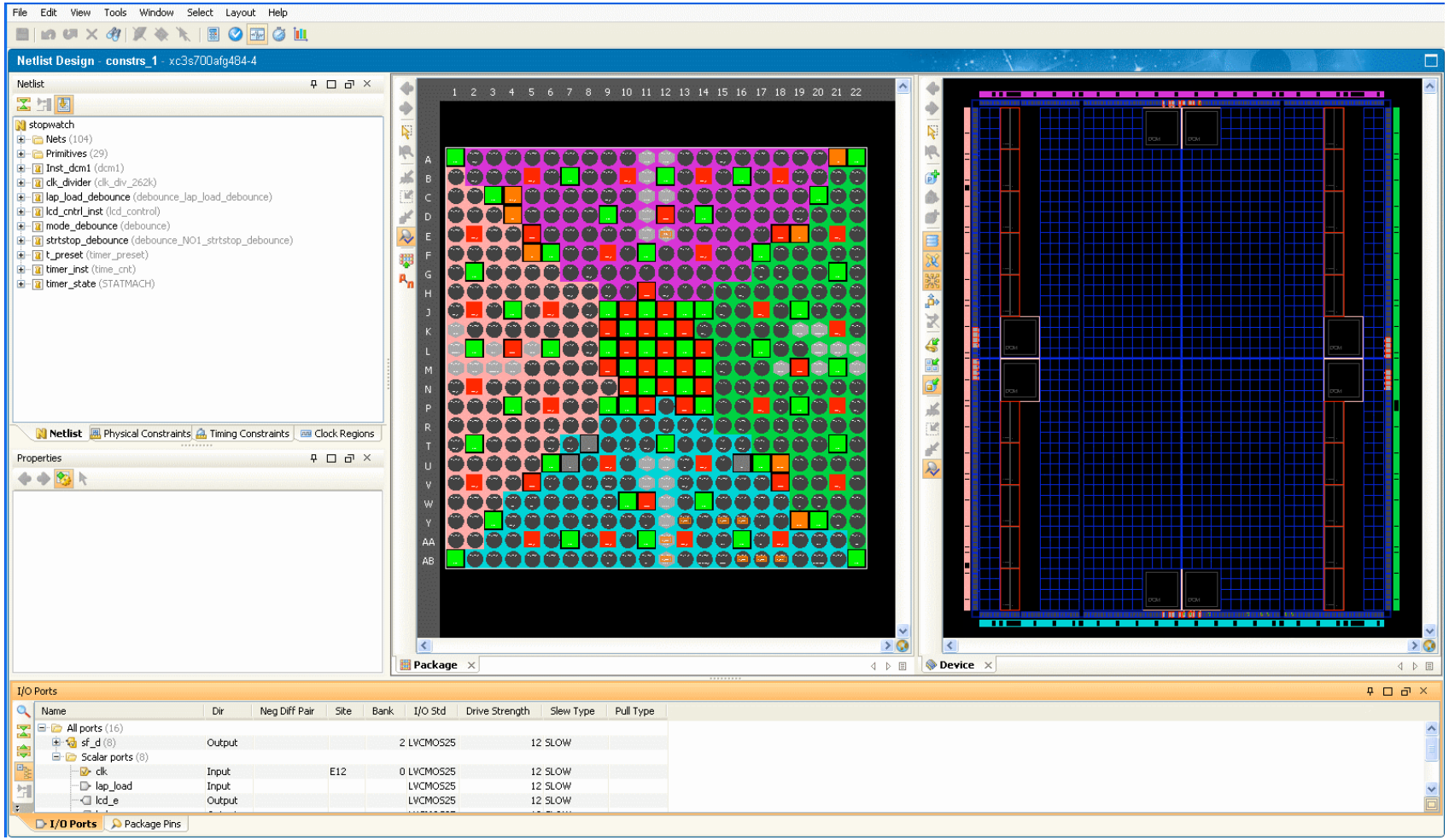
Traduction


Synthesis
Circuit netlist
Timing Constraints
Constraint Editor
Electronic Design Interchange Format
Native Constraint File
EDIF
NCF
UCF
User Constraint File
Translation
NGD
Native Generic Database file

Pin assignment file (UCF)

```
NET "clock" LOC = "P8";
NET "control<0>" LOC = "K4";
NET "control<1>" LOC = "K3";
NET "control<2>" LOC = "K2";
NET "reset" LOC = "E11";
NET "segments<0>" LOC = "R10";
NET "segments<1>" LOC = "P10";
NET "segments<2>" LOC = "M11";
NET "segments<3>" LOC = "M6";
NET "segments<4>" LOC = "N6";
NET "segments<5>" LOC = "T7";
NET "segments<6>" LOC = "R7";
```

Pin assignment graphique

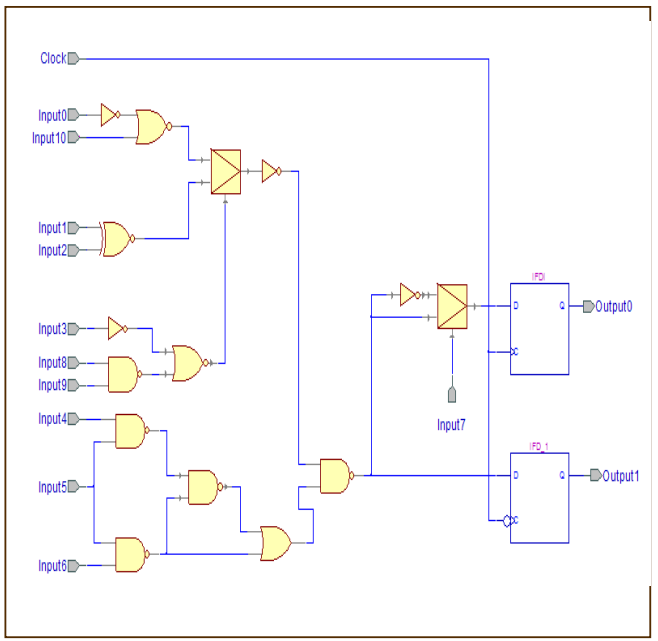


Synthèse Netlist

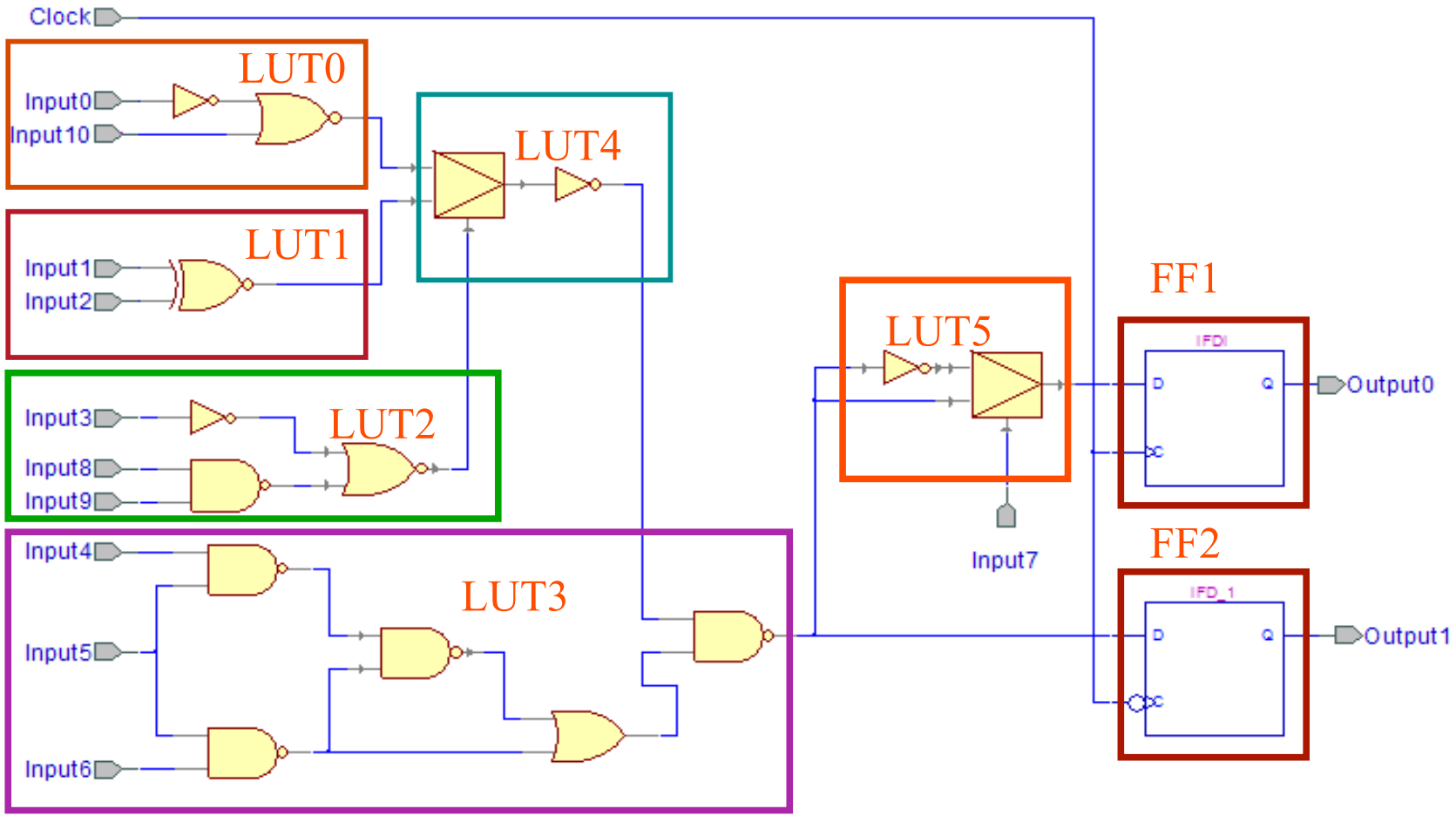
VHDL description

```
architecture MLU_DATAFLOW of MLU is

signal A1:STD_LOGIC;
signal B1:STD_LOGIC;
signal Y1:STD_LOGIC;
signal MUX_0, MUX_1, MUX_2, MUX_3: STD_LOGIC;

begin
                A1<=A when (NEG_A='0') else
                                not A;
                B1<=B when (NEG_B='0') else
                                not B;
                Y<=Y1 when (NEG_Y='0') else
                                not Y1;

                MUX_0<=A1 and B1;
                MUX_1<=A1 or B1;
                MUX_2<=A1 xor B1;
                MUX_3<=A1 xnor B1;

                with (L1 & L0) select
                                Y1<=MUX_0 when "00",
                                                MUX_1 when "01",
                                                MUX_2 when "10",
                                                MUX_3 when others;

end MLU_DATAFLOW;
```

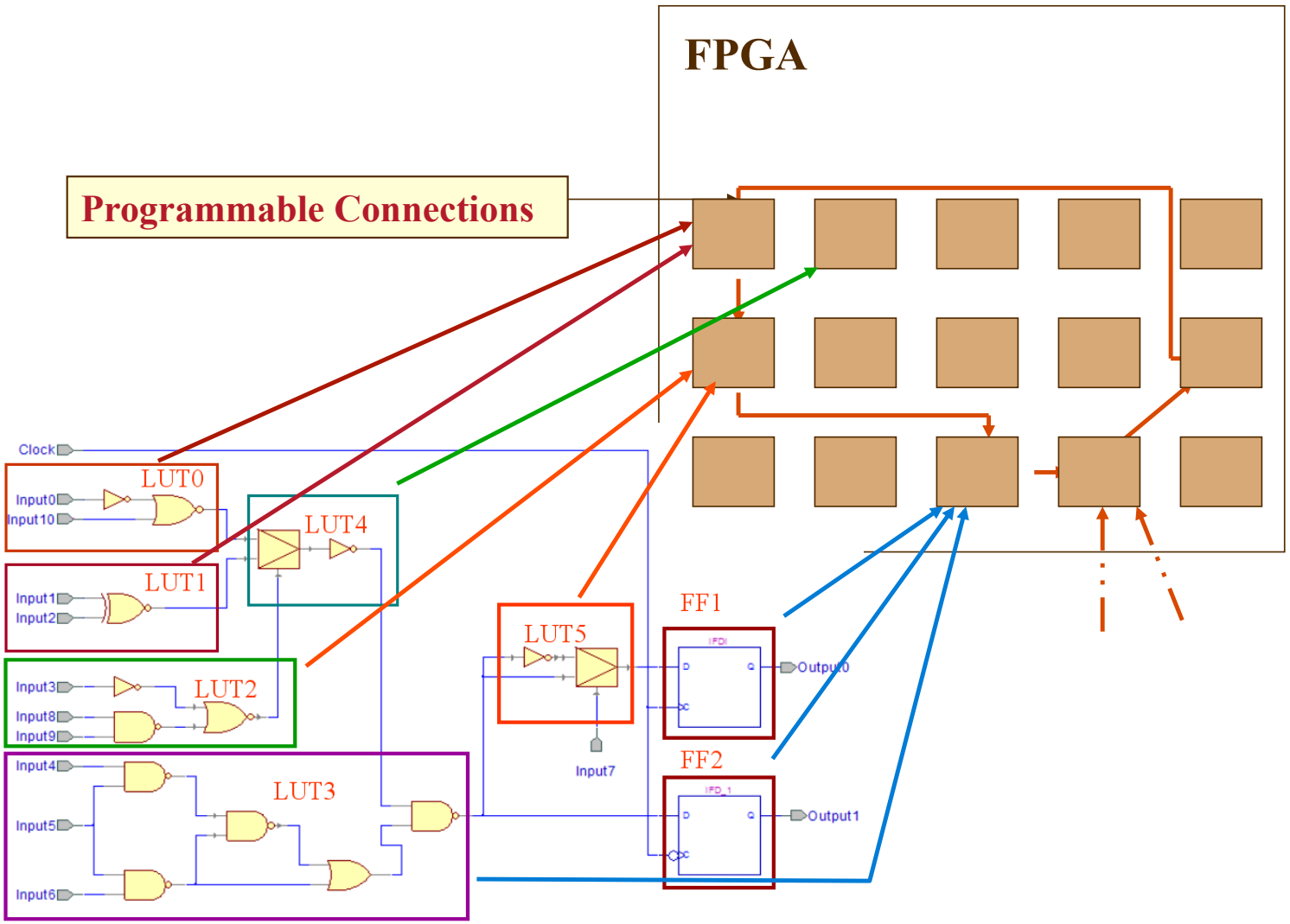
Circuit netlist



Mapping

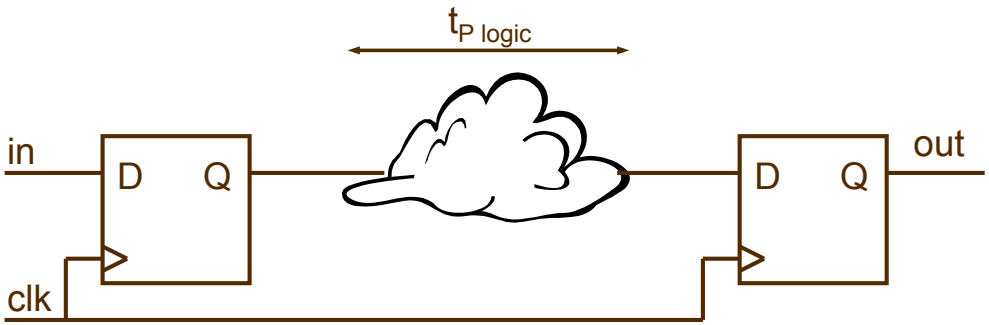

Clock
LUT0
Input0
Input10
LUT4
LUT1
Input1
Input2
LUT5
FF1
IFDI
D
Q
C
Output0
Input3
LUT2
Input8
Input9
Input7
FF2
IFD_1
Input4
LUT3
Input5
Input6
Output1

Placement/Routage



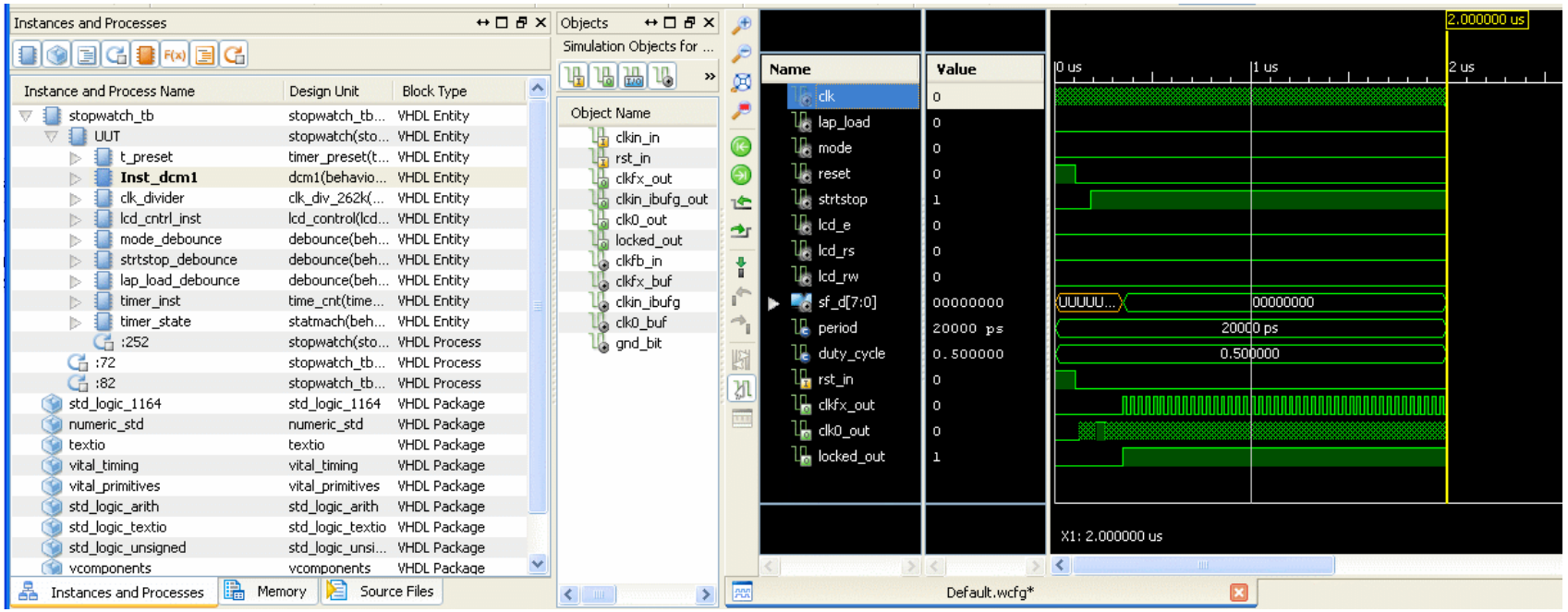
Analyse des temps de propagation

- Recherche du chemin critique (chemin le plus long)
- Détermine la fréquence maximale de fonctionnement
- Simulation post-routage basé sur ces données

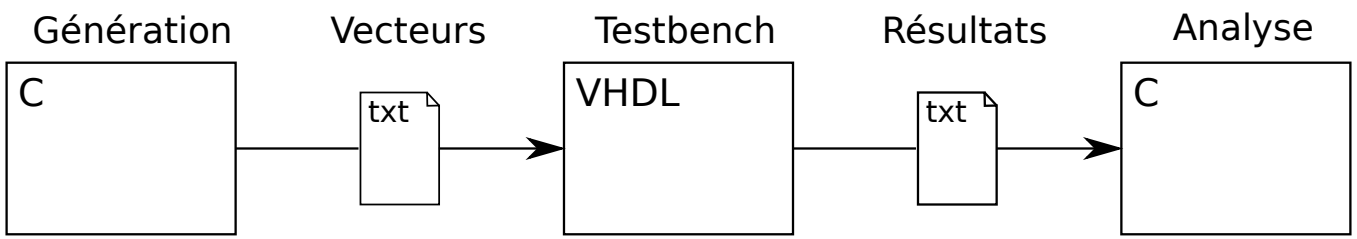


$$t_{Critical} = t_{P\ FF} + t_{P\ logic} + t_{S\ FF}$$

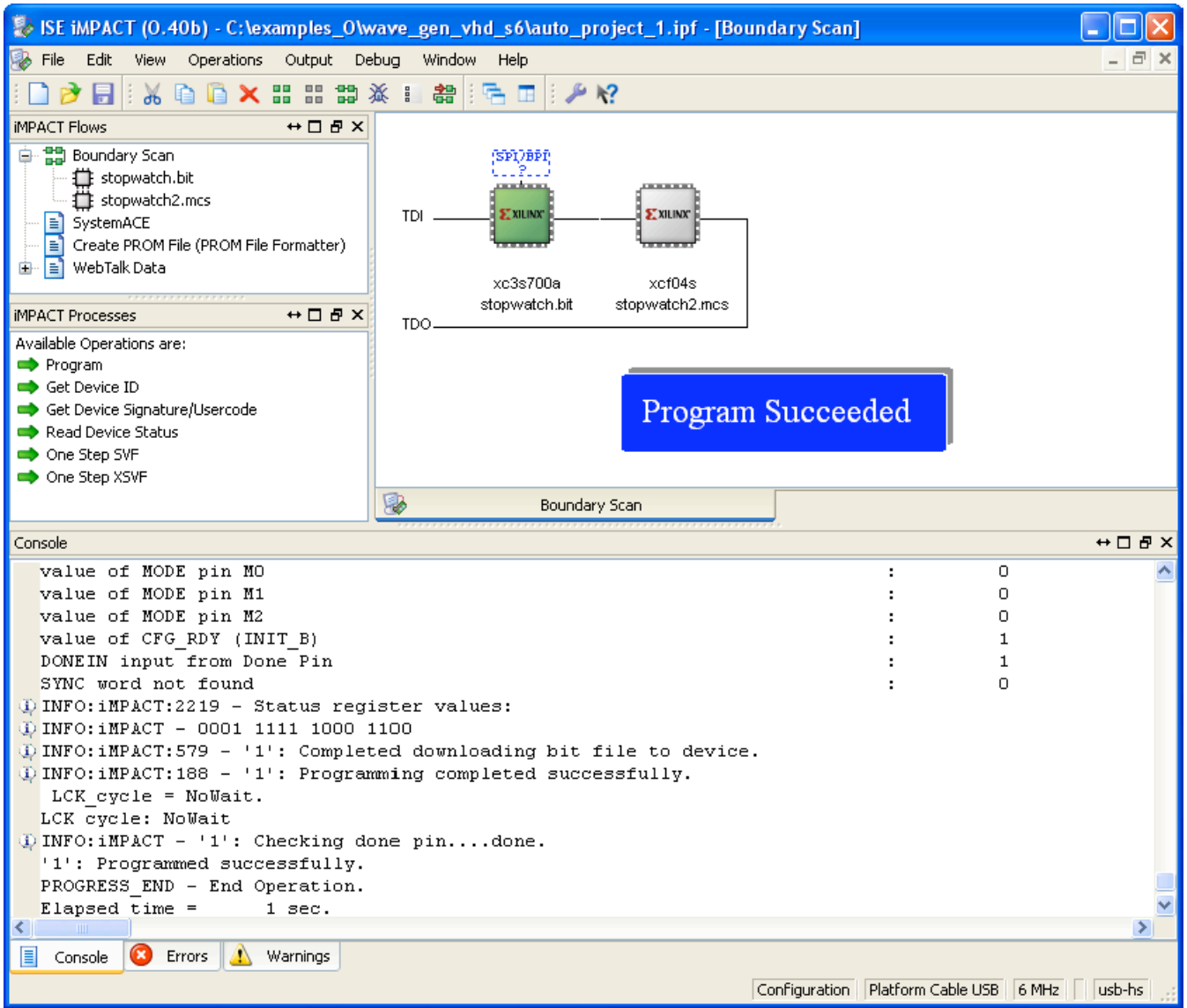
Simulation : ModelSim



Automatisation des tests



Programmation



Sommaire

Arithmétique sur FPGA

Projet Arenaire

- Équipe INRIA / Laboratoire LIP / ENS
- Développement de l'arithmétique sur ordinateur
- FloPoCo (Floating Point Cores)
  - Génération d'opérateurs de calculs pour FPGA
  - Opérateurs classiques ($+, -, \times, /$ et $\sqrt{x}$)
  - Opérateurs *exotiques* (multiplication par une constante...)
  - Virgule fixe, flottant, formats exotiques...

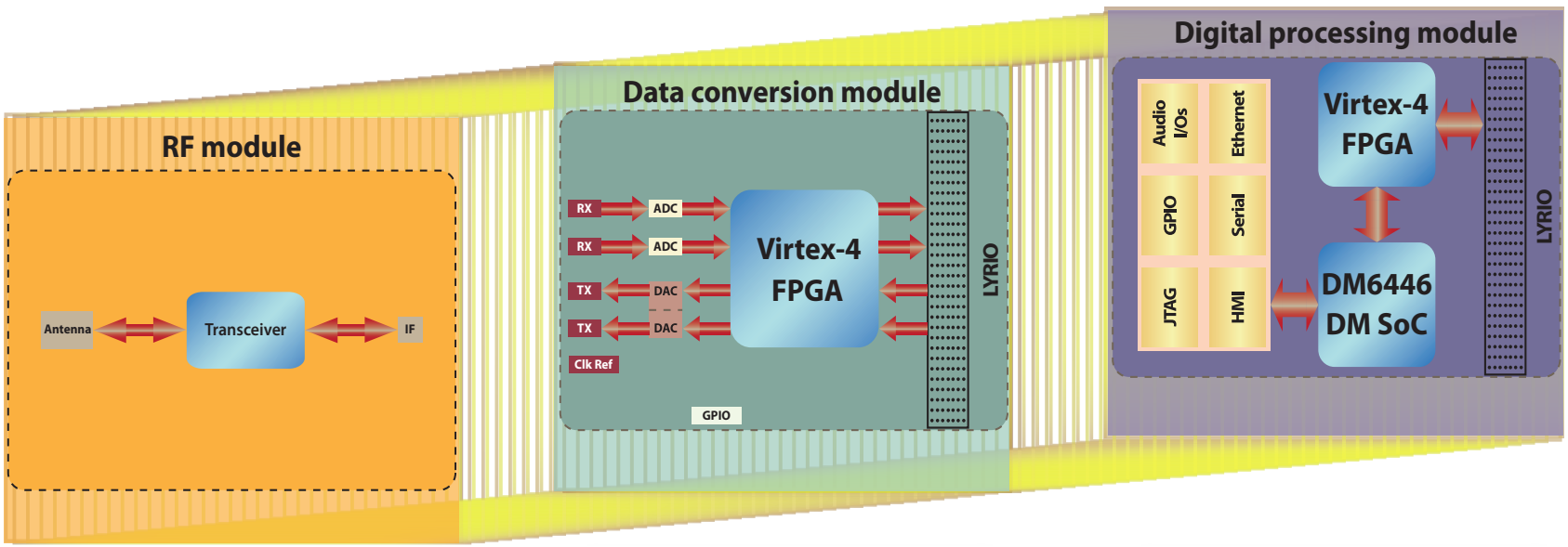
Cryptographie

- Laboratoire CITI / INSA
- Développement/expérimentations de nouvelles architectures pour la cryptographie

Radio Logicielle

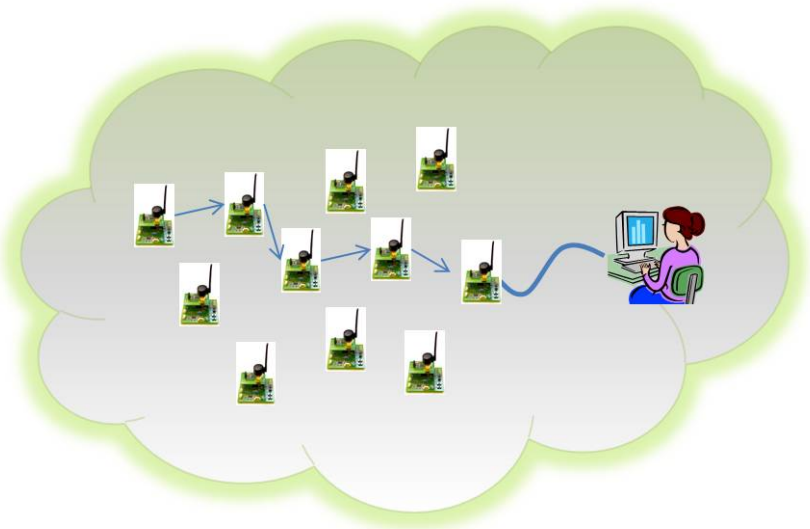
Plateforme matérielle

- Équipe Airelle / Laboratoire CITI / INSA
- Développement de composants radio en logiciel
- Plateforme d'expérimentation
- Lyrtech Small Form Factor

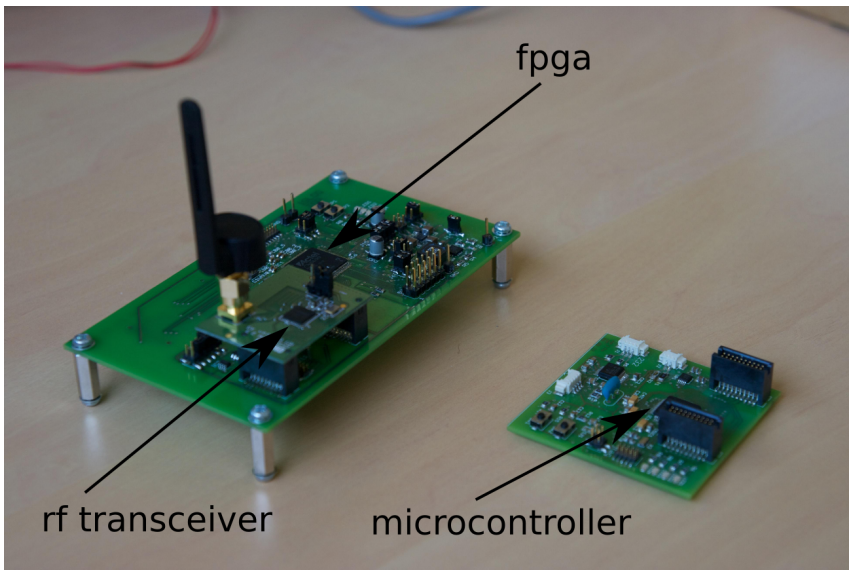


Exemple : Cryptographie sur FPGA (GPS)

Les réseaux de capteurs sans fil

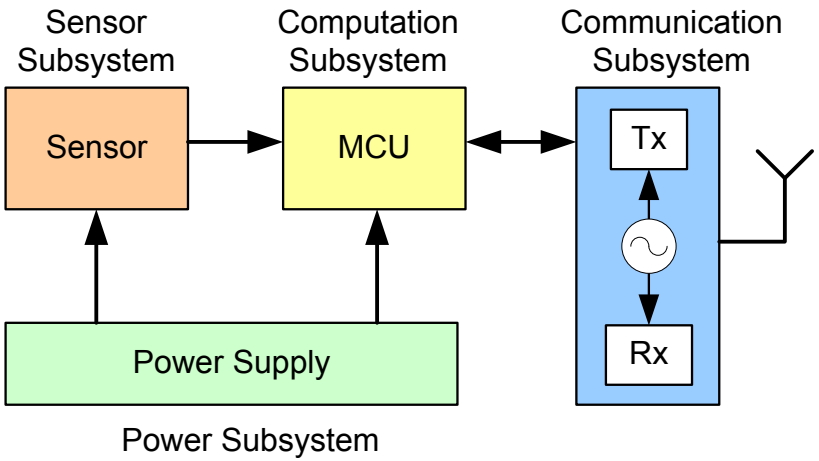


PowWow

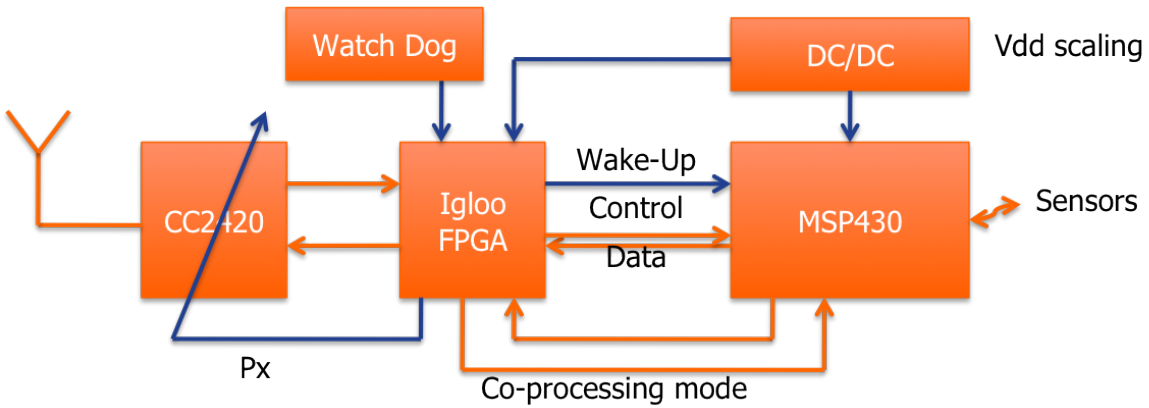


- Actel Igloo AGL250
- TI MSP430
- TI CC2420

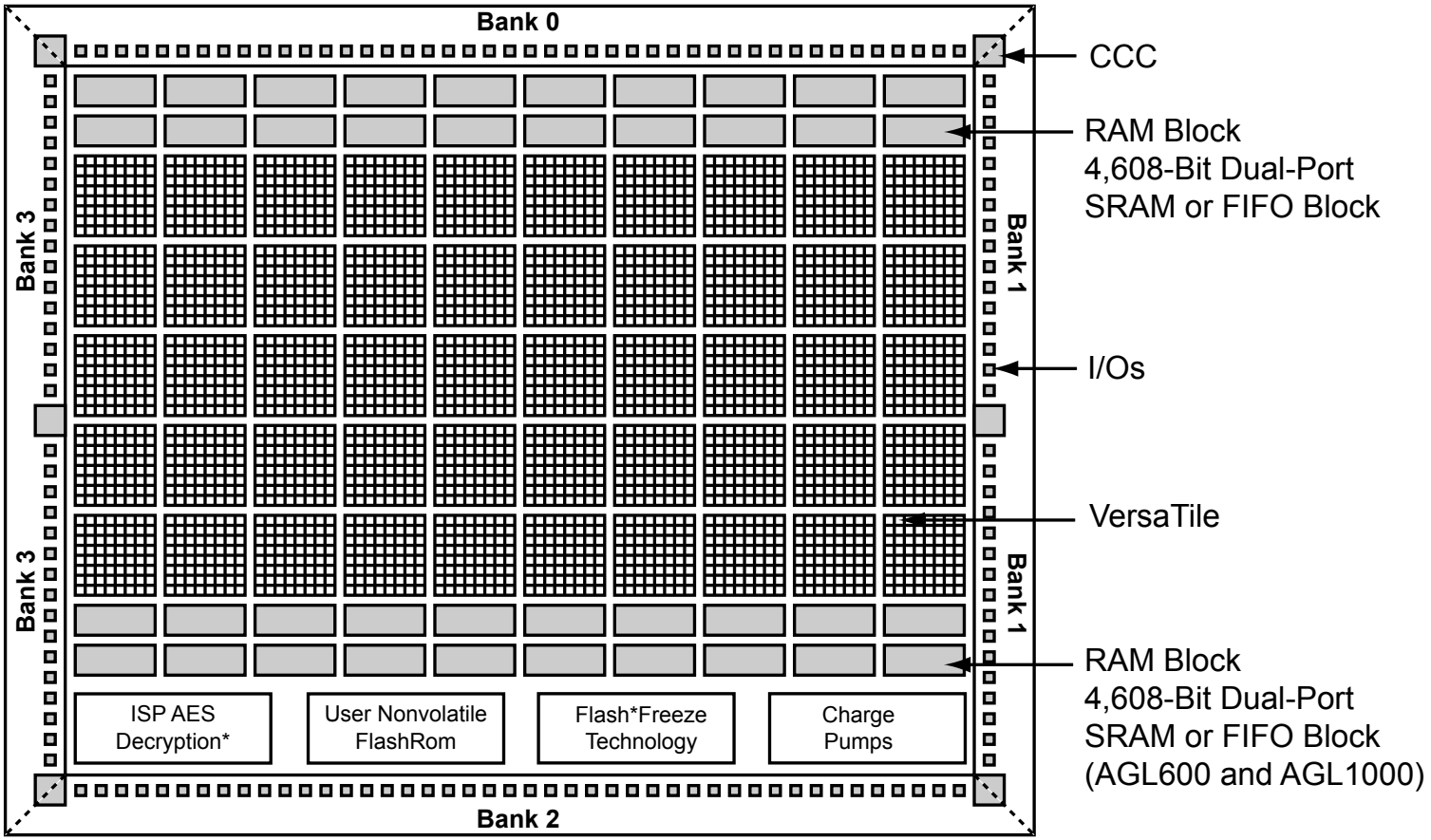
Architecture classique



Architecture PowWow



Actel Igloo AGL250



- 250 000 portes logiques $\simeq$ 6144 VersaTiles
- Horloge cadencée à 8 MHz

Implémentation de GPS

Réaliser une multiplication :

- Une clé connue de 128/256/512 bits
- une valeur de 32 bits

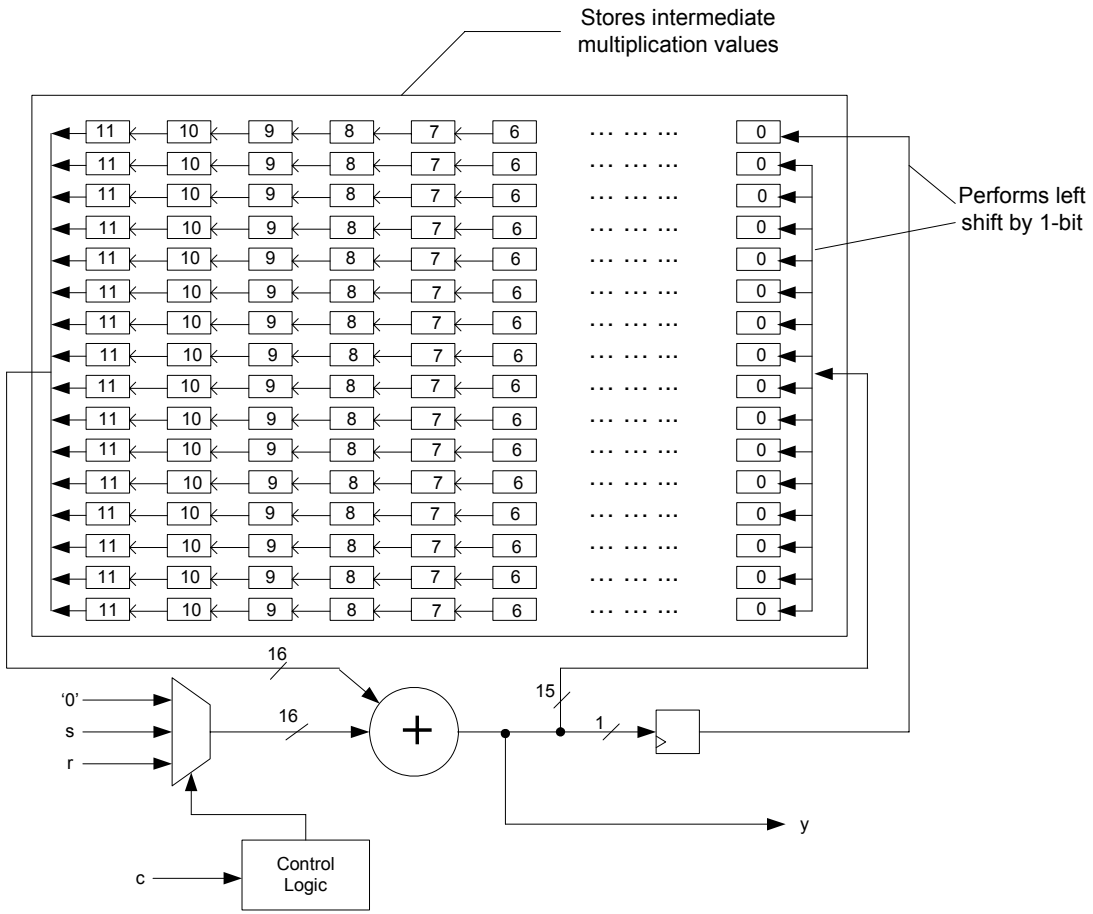
De manière efficace ?

Multiplication série : méthode *shift and add*

					1	0	0	1	0	1	0
	×							1	1	0	1
1		1	0	0	1	0	1	0			
1	+		1	0	0	1	0	1	0		
0	+			0	0	0	0	0	0	0	
1	+				1	0	0	1	0	1	0
		1	1	1	1	0	0	0	0	1	0

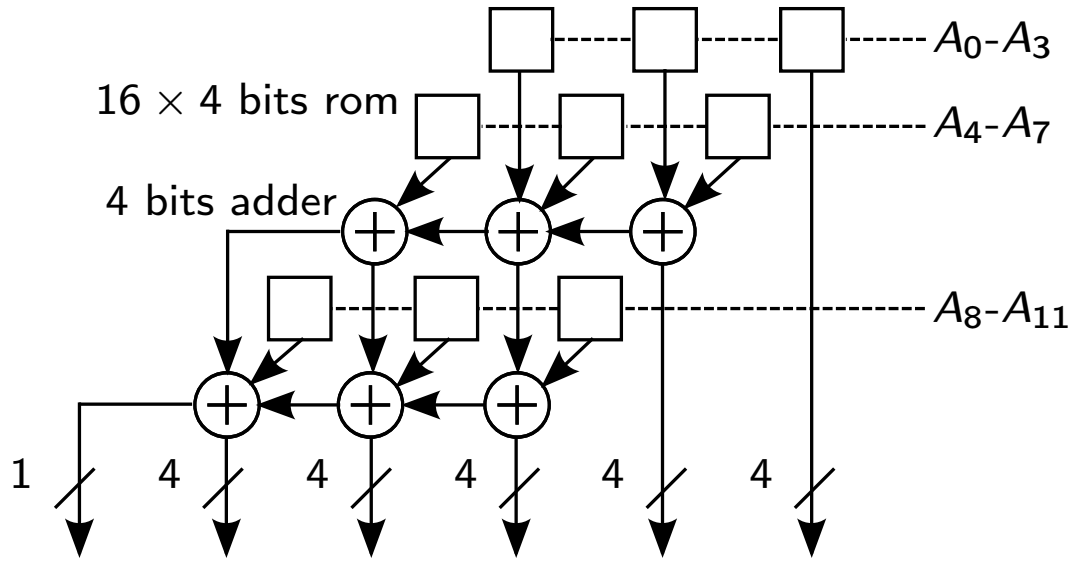
Multiplication par une constante : méthode KCM

Multiplication série : McLoone and Robshaw, 2007



- Architecture de référence en cryptographie embarquée

Multiplication par une constante : Architecture KCM



- Architecture générée par FloPoCo

Résultats

	secret size	128 bits	256 bits	512 bits
Actel Igloo VersaTiles	parallel impl.	10476	20755	44307
	serial impl.	1546	2253	3698
	parallel/serial ratio	6.8	9.2	12
Run cycles	parallel impl.	8	12	20
	serial impl.	339	603	1131
	serial/parallel ratio	42.4	50.3	56.55

- AGL250 ($20) : 6144 VersaTiles
- AGL1000 ($100) : 24576 VersaTiles

Résultats

	secret size	128 bits	256 bits	512 bits
Xilinx Virtex4 Slices	parallel impl.	2213	4358	7115
	serial impl.	493	784	1377
	parallel/serial ratio	4.5	5.5	5.2
Run cycles	parallel impl.	8	12	20
	serial impl.	339	603	1131
	serial/parallel ratio	42.4	50.3	56.55

- XC4VSX35 ($600) : 15360 Slices
- source prix : Digi-Key, 11/2011

Sommaire

Sources

- Introduction aux circuits FPGA, Arnaud Tisserand
- Introduction to FPGA Devices, Boards and tools (Slides from George Manson University)
- `http://www.fpgadeveloper.com/2011/07/list-and-comparison-of-fpga-companies.html`
- `http://www.actel.com`
- `http://www.xilinx.com`
- `http://www.altera.com`
- `http://www.lyrtech.com`
- `http://www.digikey.com`
- `http://powwow.gforge.inria.fr`
- `http://opencores.org`