Circuits Intégrés Programmables FPGA

Chapitre 1

Olivier Romain
Professeur des Universités
Olivier.romain@gmail.com
http://olivieromain.free.fr

IUT CERGY-PONTOISE

Plan du chapitre 1

1. Pourquoi la logique programmable ?
2. Principes de base des PLD
3. Technologies des PLD
4. PAO
5. Altera Quartus
6. Présentation de la carte DE2-ALTERA

Références

- Conception des ASICS – P. Naish et P. Bishop – Masson
- Logique programmable – L. Dutrieux et D. Demigny – Eyrolles
- Circuits logiques programmables – C. Tavernier – Dunod
- Systèmes numériques – Floyd – 9ème édition – Reynald
- www.altera.com
- www.xilinx.com
- www.atmel.com

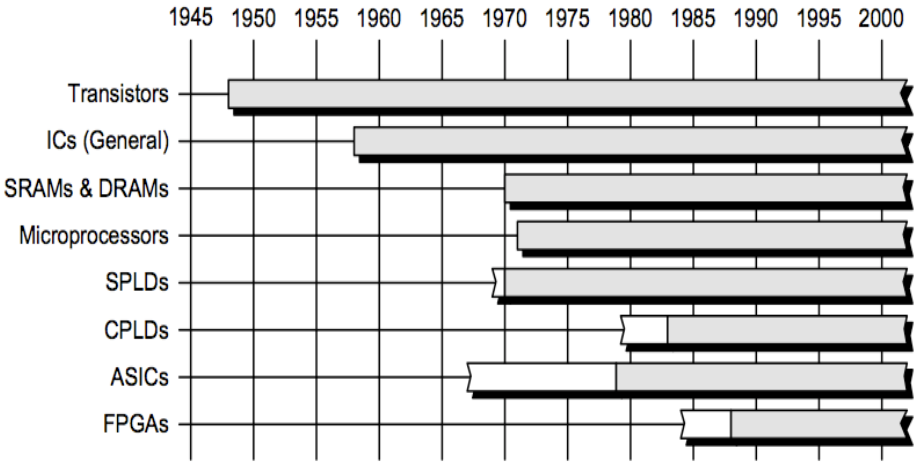
Vocabulaire

ASIC *Application Specific Integrated Circuit* - Circuit intégré conçu à la demande
CPLD *Complex Programmable Logic Device* - Réseau logique programmable complexe
DRAM *Dynamic Random Access Memory* - Mémoire dynamique à accès aléatoire
DSP *Digital Signal Processor* - Processeur orienté vers le traitement du signal
EDIF *Electronic Design Interchange Format*
EEPLD *Electrically Erasable Programmable Logic Device* - PLD effaçable électriquement
FPGA *Field Programmable Gate Array* - Réseau de portes programmables
GAL *Generic Array Logic* - PAL générique
ISP *In-System (In Situ) Programmable* - Composant programmable sur carte
JEDEC *Joint Electronic Device Engineering Council* - Organisme de normalisation
JTAG *Joint Test Action Group* - Bus de test des composants
LCA *Logic Cell Array (Xilinx)* - Réseau de cellules logiques
LUT *Look-Up Table*
MAX *Multiple Array Matrix* - Megapals d'Altera
NOVRAM ou NVRAM *Non Volatile Random Access Memory* - RAM non volatile
OTP *One Time Programmable* - Programmable une seule fois
PAL *Programmable Array Logic* - Réseau logique programmable
PGA *Programmable Gate Array* - Réseau de portes programmable
PLA *Programmable Logic Array* - Réseau logique programmable
PLD *Programmable Logic Device* - Dispositif logique programmable, EPLD : Erasable PLD : PLD Effacable
ROM *Read Only Memory,* Mémoire à lecture seule*,* **PROM** *: Programmable* **ROM,**. **EPROM:** *Erasable :* Effacable, **EEPROM :** Electricaly **EPROM** : Mémoire à lecture seule, électriquement effaçable.
RAM *Random Access Memory* - Mémoire à accès aléatoire
SDF *Standard Delay File*
SOG *Sea-of-Gates* - Mer de portes : réseau actif logique prédiffusé
SRAM *Static Random Access Memory* - Mémoire statique à accès aléatoire
TTL *Transistor Transistor Logic* - Logique transistor-transistor
VHDL *VHSIC Hardware Description Language* - Langage de description matérielle VHSIC

1. Pourquoi les circuits logiques programmables ?

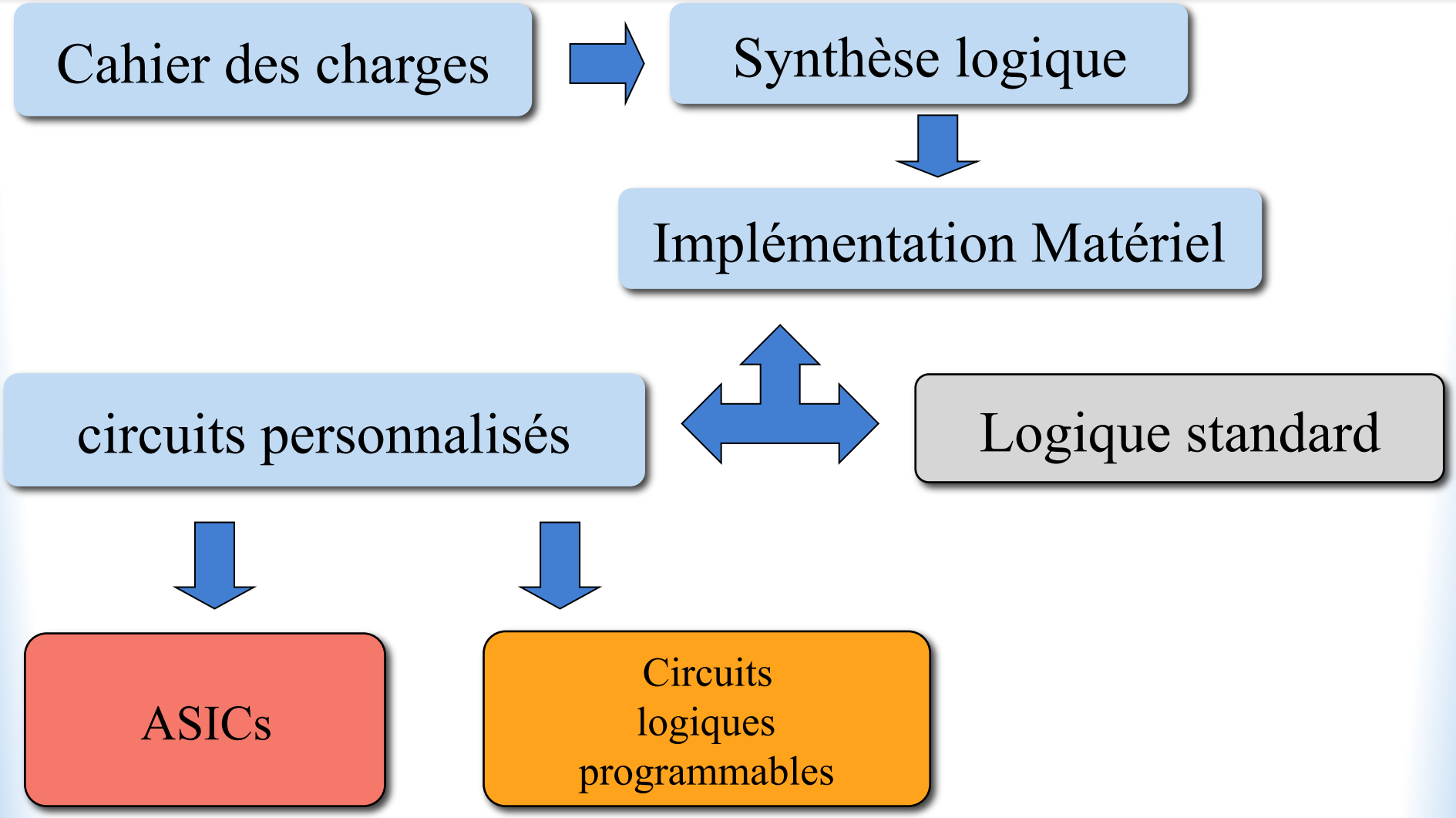
Evolution historique

- Les premiers composants programmables remontent aux début des années 70, commercialisés sous l'appellation HAL (Hard Array Logic) puis PAL (Programmable Array Logic) par la société MMI (maintenant AMD).



- L'évolution des fonctionnalités des systèmes électroniques conditionnés par l'augmentation des procédés d'intégration
  - SSI et MSI devenu obsolète
  - Avènement des PAL, GAL, PLD, ASIC et des FPGA

Quelle implémentation choisir?



Pourquoi des circuits logiques programmables ?

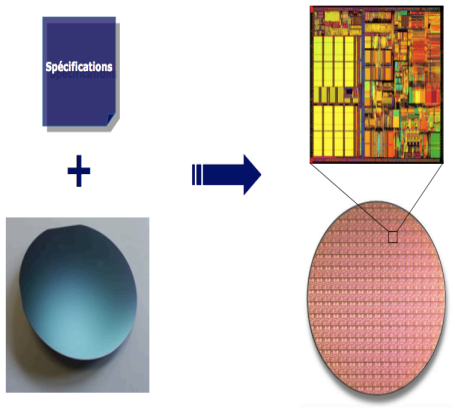
ASICs	Circuits logiques programmables	Logique standard
• ASIC (Application Specific Integrated Circuit) • La + grande intégration • Coût et temps importants • Meilleurs performances en consommation et fréquence de fonctionnement	• Grande flexibilité • Conception réutilisable • Parallélisme massif • Temps de développement court	• Peu de flexibilité • Faible coût • Disponibilité immédiate • Fréquence de fonctionnement faible

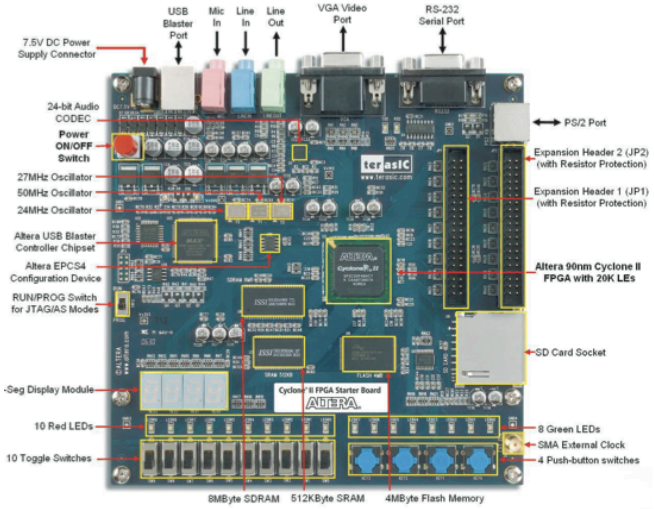
Pourquoi des circuits logiques programmables ?

ASICs

Circuits logiques programmables

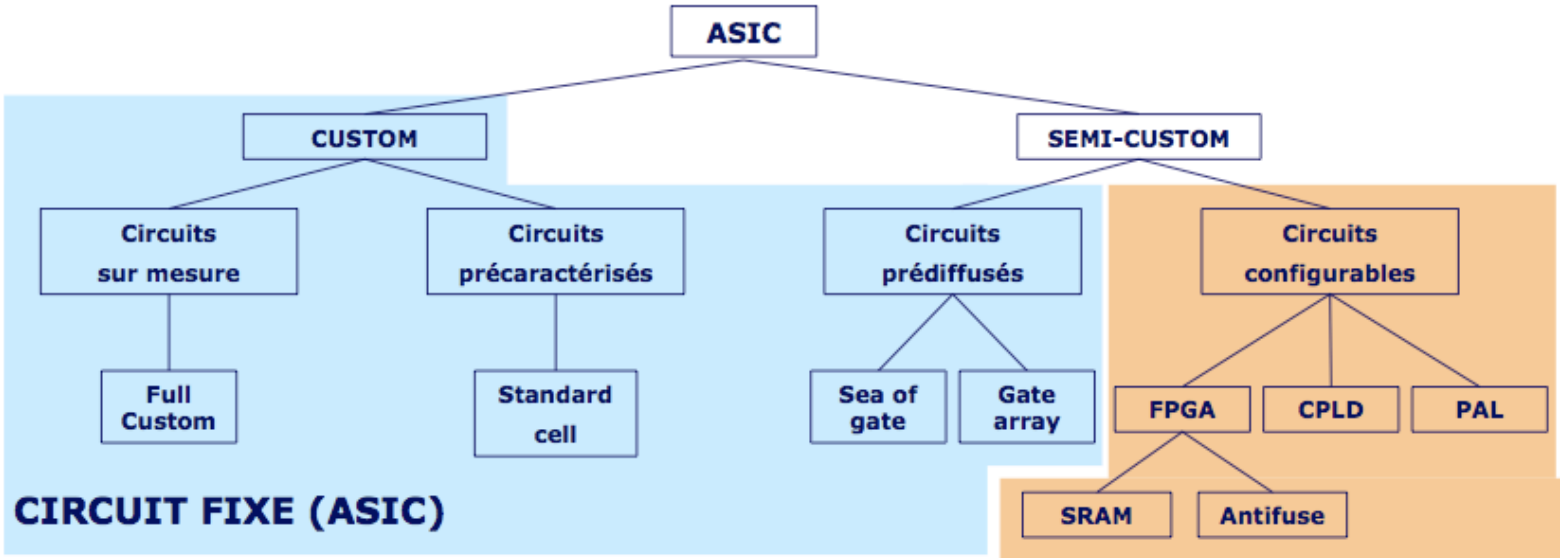
Logique standard



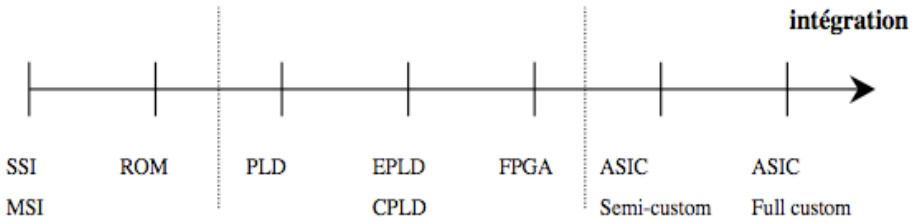


- Full-custom
- Pré-caractérisé
- Pré-diffusé

Taxonomie des cibles HW

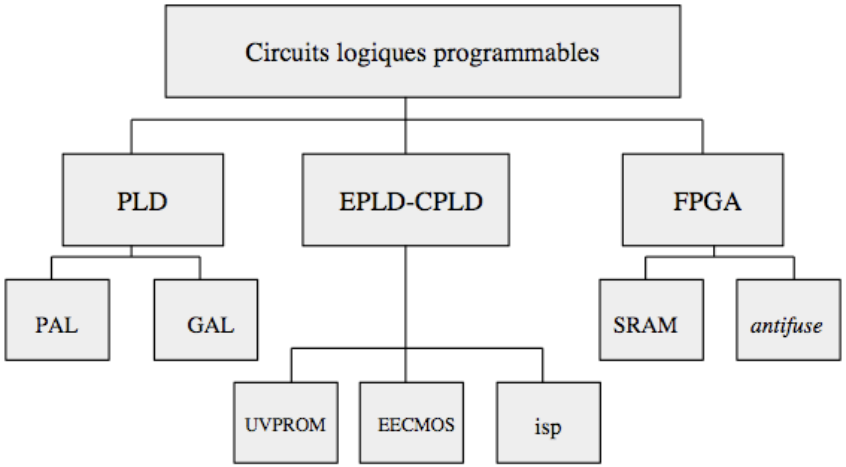


ASIC : Application Specific Integrated Circuit
FPGA : Field Programmable Gate Array
CPLD : Complex Programmable Logic De
PAL : Programmable Array Logic
GAL : Generic Array Logic = PAL
SRAM : Static Random Acess Memory



Taxonomie des PLD

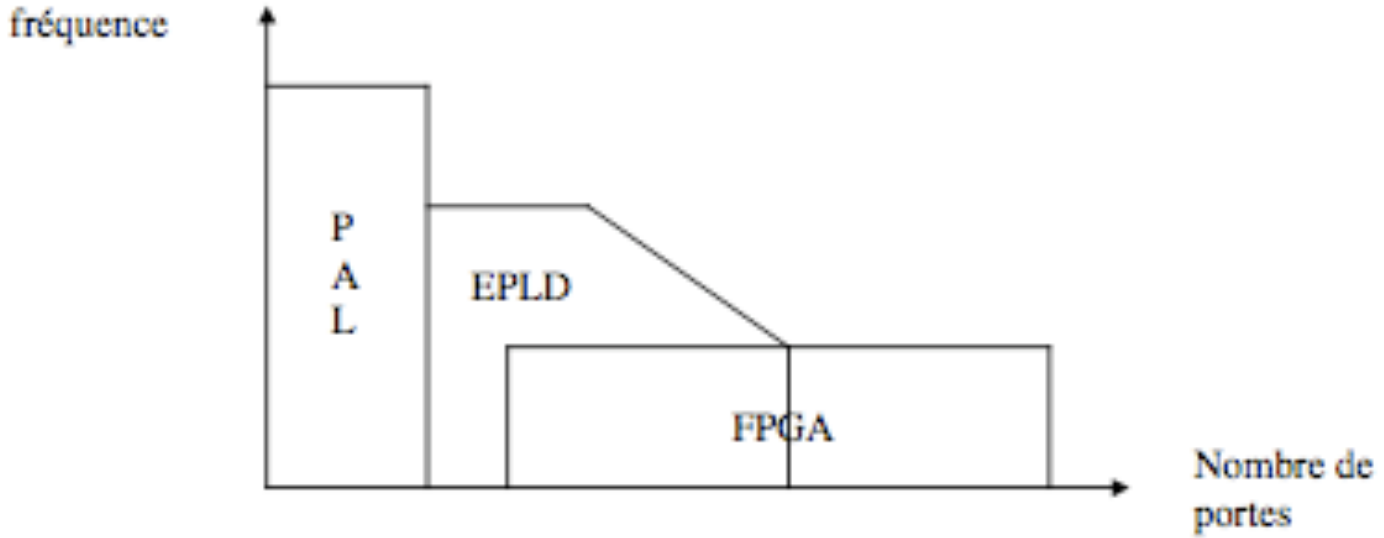
- **PAL**
  - PLD
  - PAL bipolaire (OTP)
  - PAL à fusibles (effaçables)
- **CPLD** (*Erasable PLD* ou *Complex PLD)*
  - GAL, Generic Array Logic
    - Marque déposée par Lattice
    - Programmables et effaçables électriquement
  - EPLD (PAL – CMOS) programmable électriquement et effaçable aux UV
  - EEPLD programmable et effeçable électriquement



- **FPGA** (*Field Programmable Gate Array)*
  - Réseau de Blocs logiques et de bus d' interconnexion entièrement programmable
    - SRAM
    - Antifuse (OTP)

Taxonomy des PLD

- Classification / performances

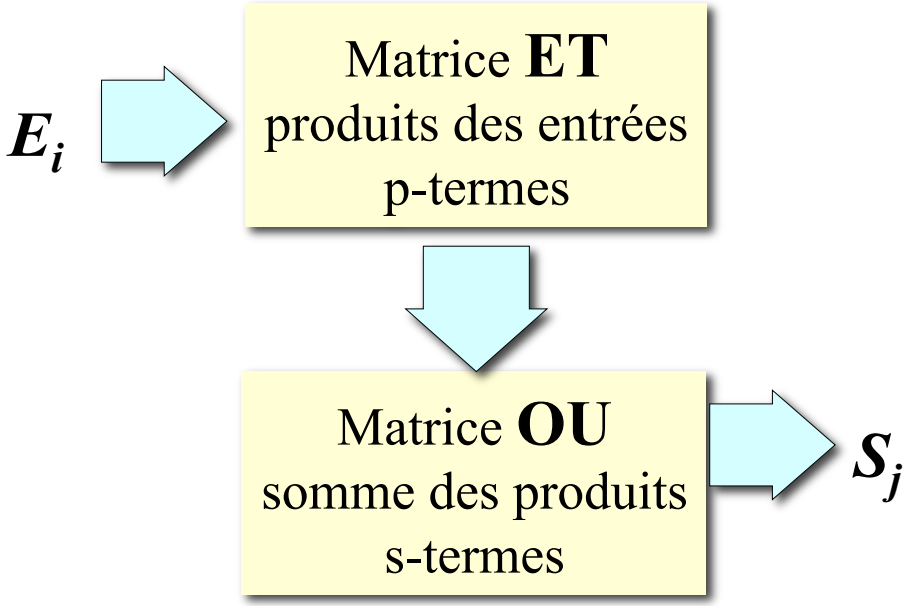


2. Principes de base des PLD

Principe de codage

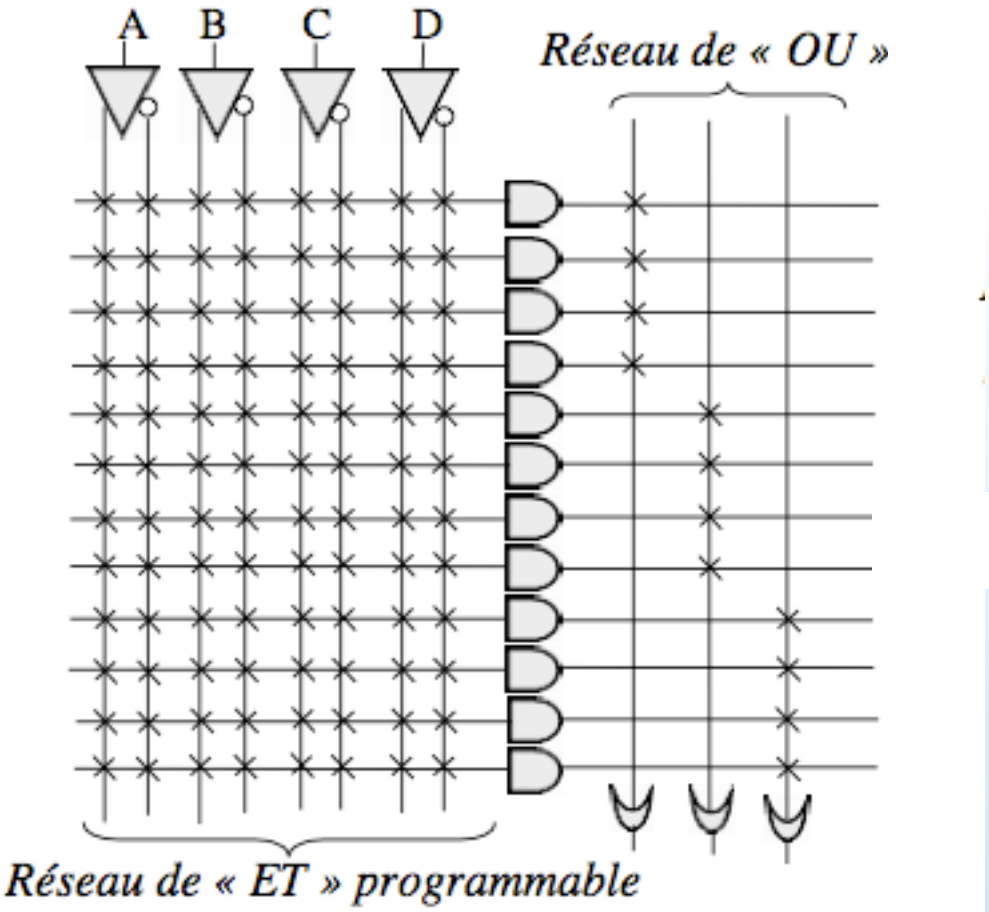
- Toute fonction combinatoire peut être mise sous la première forme canonique, somme de produit
- Les circuits programmables exploitent ce principe

$$S_j = E_3.E_1.E_0 + \overline{E_2} + E_0.E_2.E_3$$



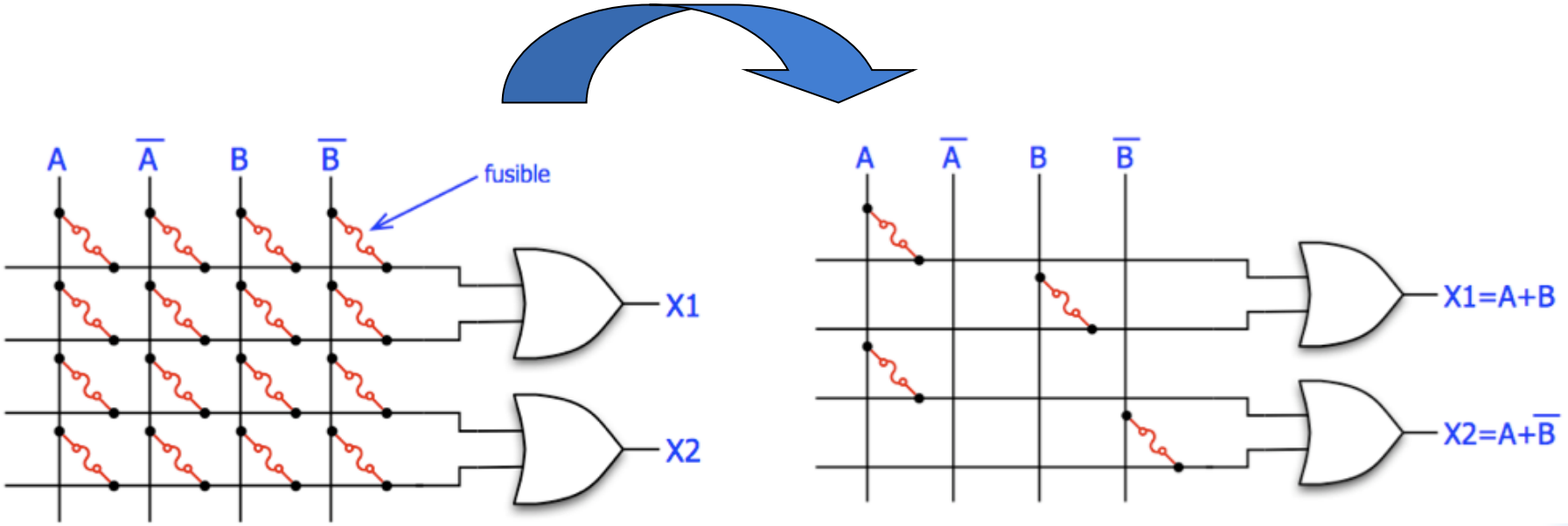
Principe de codage

- Matrice ET (p-terme)
  - Table d'interconnexion programmable
  - Différentes technologies existent concernant la réalisation de l'interconnexion
- Matrice OU (s-terme)
  - Table d'interconnexion fixe ou programmable



Principe de codage

- Le circuit est personnalisé par création/destruction de connections sur la structure prédéfinie.



Principe de codage

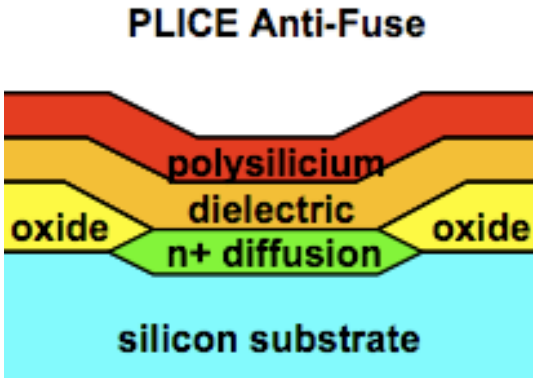
Plusieurs technologies pour réaliser le point de connexion

- **Fusibles**

Intact Programmé

  - 1ère technologie employée
  - Fusible > lien métallique
  - Un courant de forte intensité fait fondre le fusible
  - Opération irréversible
    - On Time Programmable : OTP

- **Antifusibles**

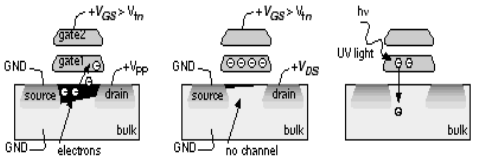


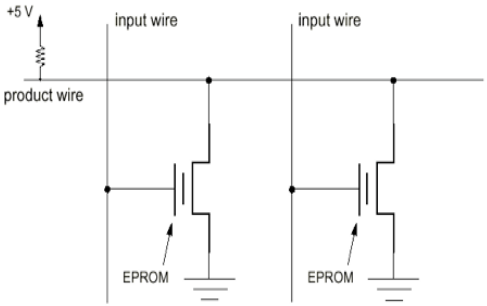
  - Crée une connexion au lieu de la détruire
    - Circuit ouvert avant programmation

Principe de codage

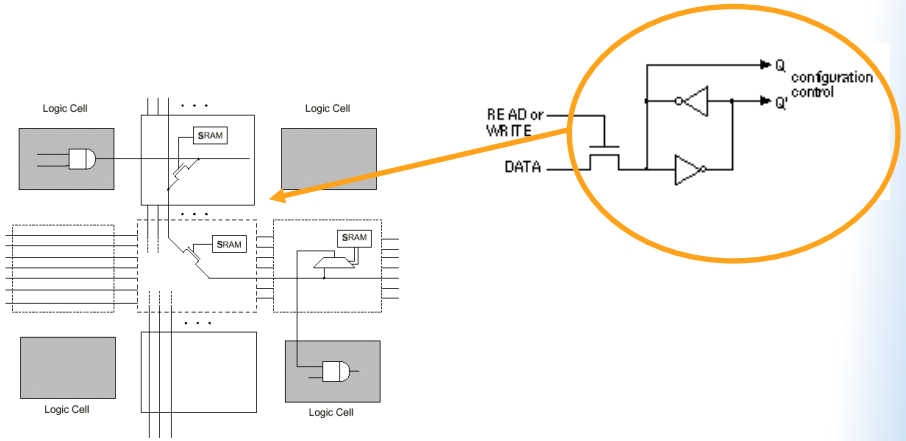
- **EPROM**
  - *Erasable Programmable Read-Only Memory*
  - Transistors MOS à grilles flottantes
    - Effet tunnel de Fowler-Nordheim
    - Effaçable / UV

- **FLASH**
  - Même principe mais effaçable électriquement



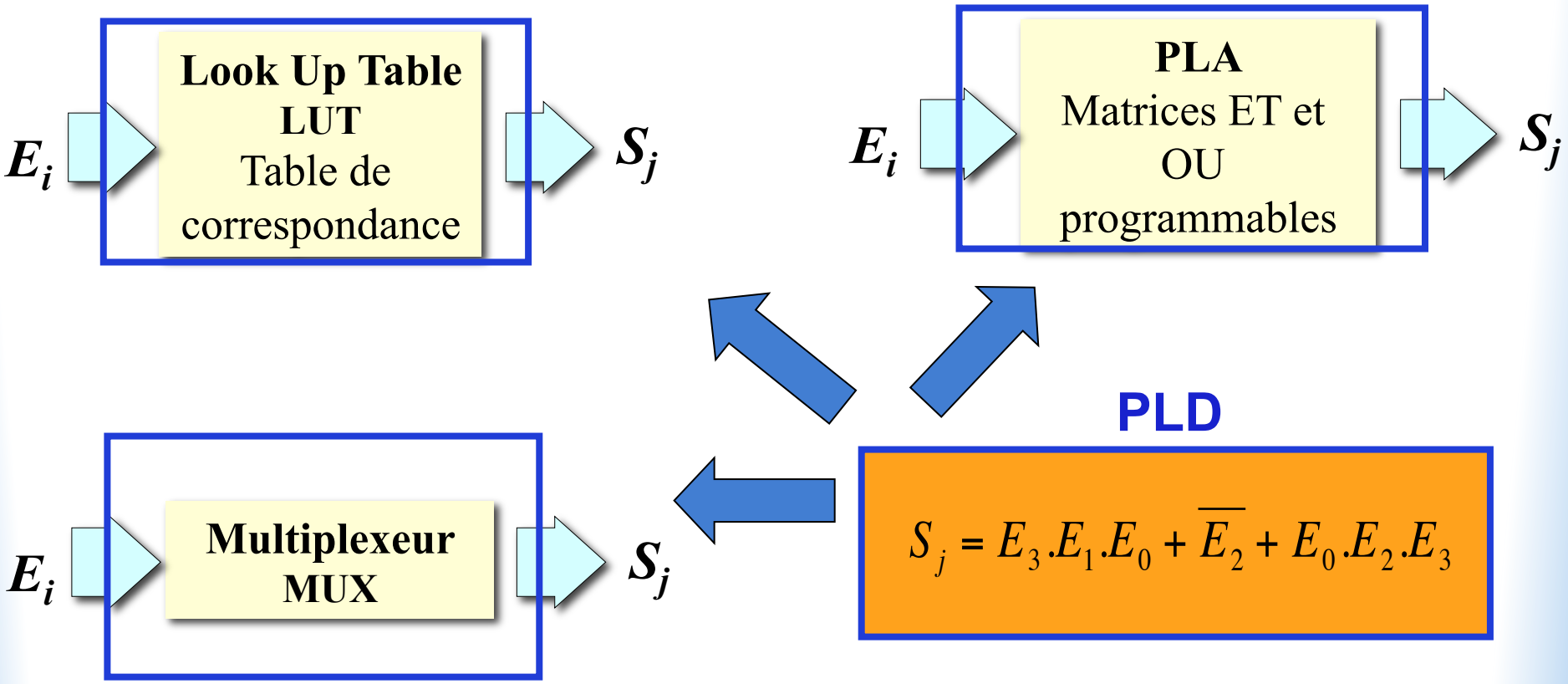


- **SRAM**
  - Chaque cellule de mémoire statique est utilisée pour commander un transistor MOS
  - Technologie volatile qui nécessite une programmation du circuit avant chaque utilisation



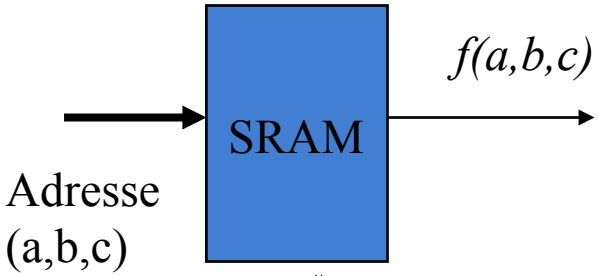
Principe de base

- Les PLD utilisent d'autre principe de codage que les PLA pour réaliser les fonctions logiques



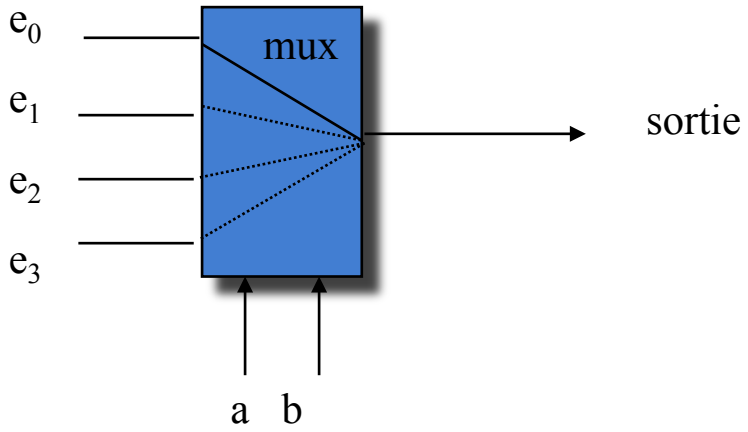
Principe de codage

- LUT
  - Une LUT est une mémoire ou est enregistrée la table de vérité de la fonction logique



a	b	c	$f(a,b,c)$
0	0	0	0
0	0	1	0
⋮	⋮	⋮	⋮
1	1	0	1
1	1	1	1

- MUX
  - Un multiplexeur est un circuit de type aiguillage



$$sortie = \bar{a}\bar{b}e_0 + \bar{a}be_1 + a\bar{b}e_2 + abe_3$$

Exemple

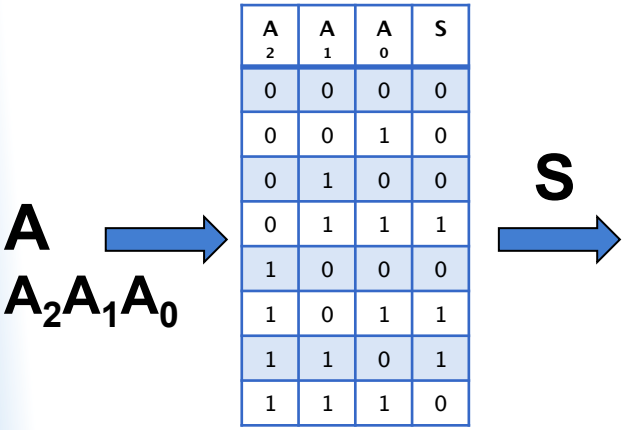
- On souhaite réaliser la fonction combinatoire de détection de parité d'un mot A de trois bits ($A_2A_1A_0$). Si A comporte un nombre de bits pair, S=1

A_2	A_1	A_0	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$$S = \overline{A_2}.A_1.A_0 + A_2.(A_1 \oplus A_0)$$

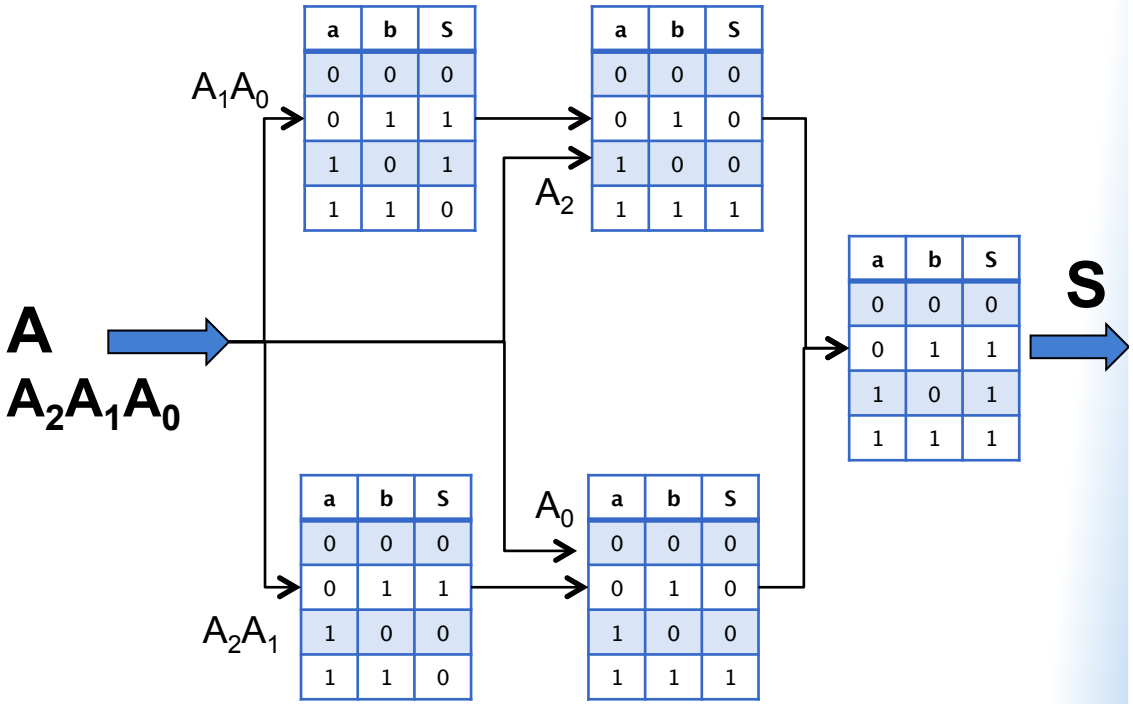
Exemple

- 1xLUT8



A_2	A_1	A_0	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

- 5xLUT4



a	b	S
0	0	0
0	1	1
1	0	1
1	1	0

a	b	S
0	0	0
0	1	0
1	0	0
1	1	1

a	b	S
0	0	0
0	1	1
1	0	0
1	1	0

a	b	S
0	0	0
0	1	0
1	0	0
1	1	1

a	b	S
0	0	0
0	1	1
1	0	1
1	1	1

Résumé

- Un circuit programmable contient :
  - des éléments logiques (portes, circuits, …)
  - des connections entre les portes logiques

- Les différentes familles de circuits programmables utilisent :
  - Un des trois codages possibles
  - Plusieurs codages à la fois

	PLA	LUT	MUX
PLD	X		
CPLD	X	X	
FPGA	X	X	X

3. Technologies des PLD

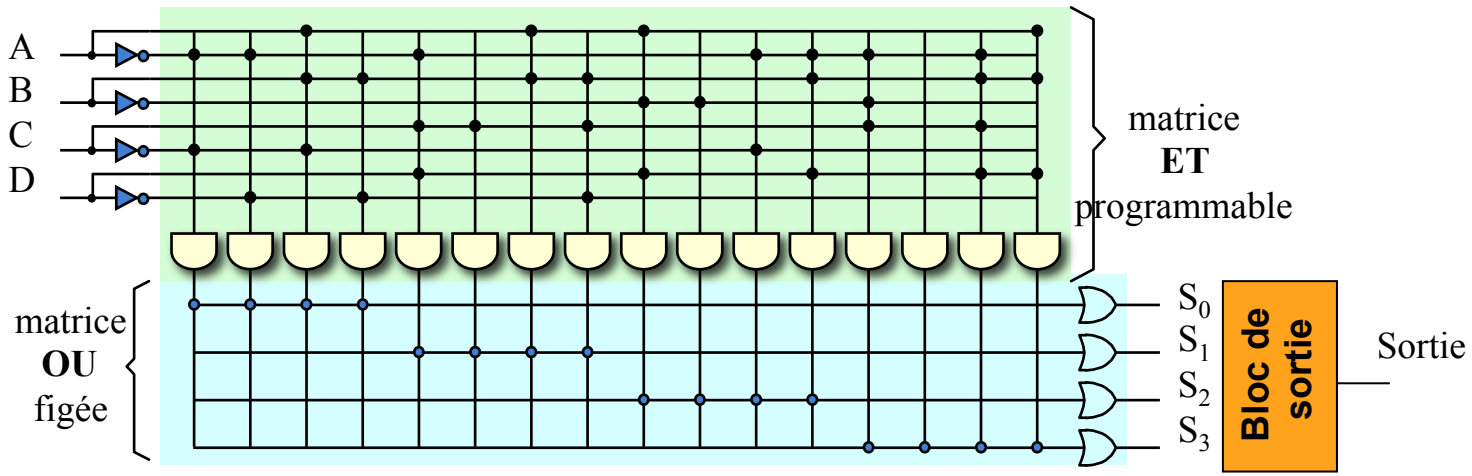
a. PAL

b. CPLD

c. FPGA

Technologie des PLD / PAL

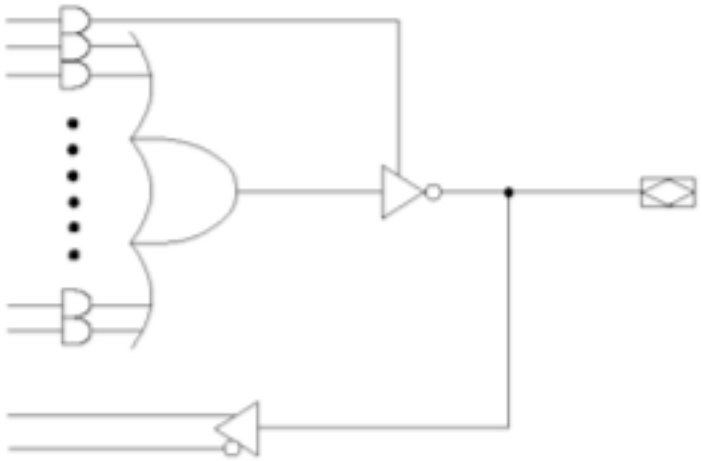
- Point d'interconnexion de la matrice sont soit des transistors bipolaires soit des MOS.



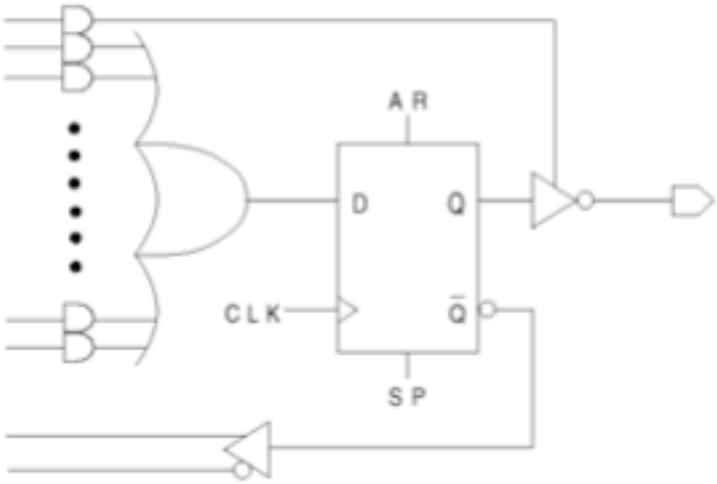
- En fonction du bloc de sortie, plusieurs sous-familles
  - PAL combinatoire
  - PAL à registre
  - PAL versatile

Technologie des PAL

- PALs Combinatoires
  - Le plus simple
  - OU en sortie

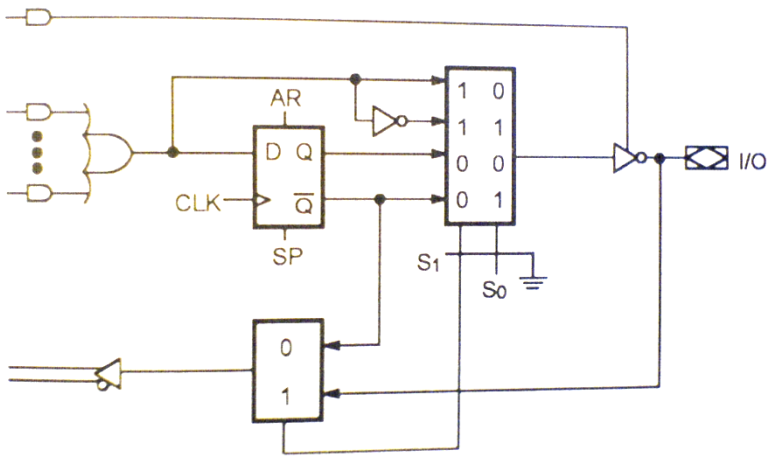


- PALs à Registres
  - Sortie mémorisée
    - Bascule D
  - Horloge commune à toutes les bascules D en sorties



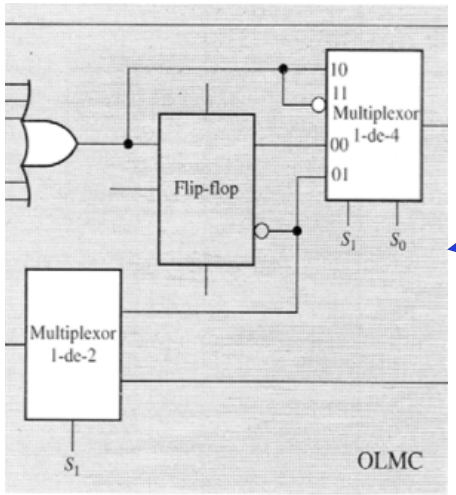
Technologie des PAL

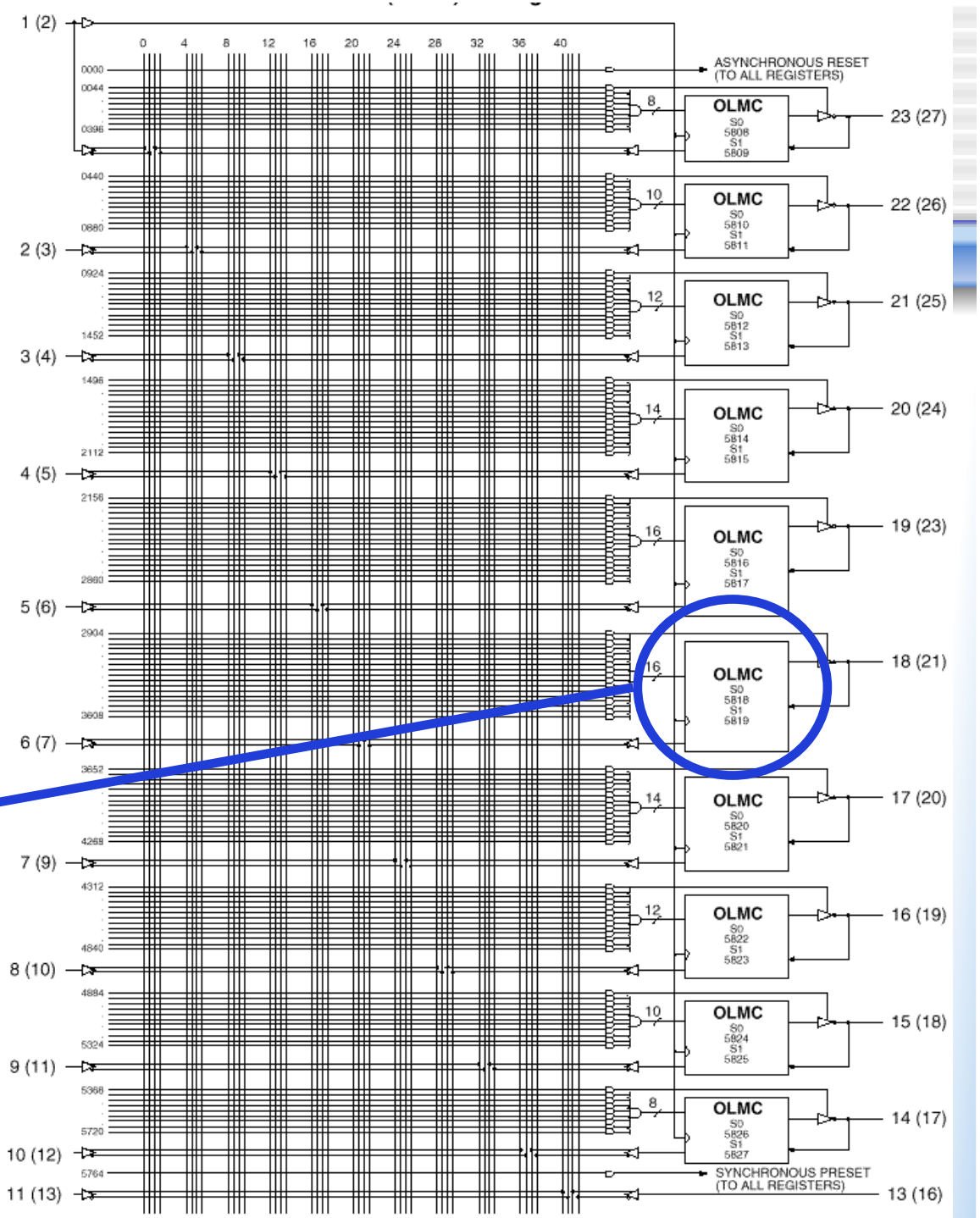
- PAL Versatile
  - Le + complet des PALs
  - Bloc de sortie permet 4 modes de fonctionnement
  - Deux retours possibles de la sortie dans le réseau
    - Mise en cascade des sorties pour réaliser la fonction logique désirée



S1	S0	Type Sortie	Retour
0	0	Registre complémentée	Registre
0	1	Registre	Registre
1	0	Combinatoire, complémentée	I/O
1	1	Combinatoire	I/O

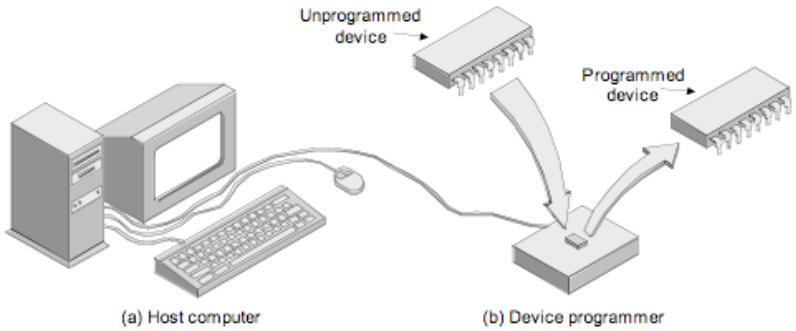
PAL22V10


10
11
Multiplexor
1-de-4
00
01
Flip-flop
S_1
S_0
Multiplexor
1-de-2
S_1
OLMC


1 (2)
2 (3)
3 (4)
4 (5)
5 (6)
6 (7)
7 (9)
8 (10)
9 (11)
10 (12)
11 (13)
0 4 8 12 16 20 24 28 32 36 40
ASYNCHRONOUS RESET
(TO ALL REGISTERS)
SYNCHRONOUS PRESET
(TO ALL REGISTERS)
OLMC
S0
S1
23 (27)
22 (26)
21 (25)
20 (24)
19 (23)
18 (21)
17 (20)
16 (19)
15 (18)
14 (17)
13 (16)

Technologie des PAL

- Programmation des PAL
  - Saisie sous un logiciel de CAO (Abel, Warp, Palasm, etc …)
    - Représentation schématique
    - Equations
    - VHDL, etc…
  - Compilation pour la génération du fichier JEDEC
    - JEDEC (Joint Electron Device Engineering Council)
    - Table des fusibles du circuit
  - Insertion du circuit dans le programmateur



Technologie des PAL

- Programmation des PAL

Fichier de programmation sous PALASM

```
CHIP  per_gal  PALCE16V8

;---------------------------------- PIN Declarations --------------
PIN  1          a15                             COMBINATORIAL ; INPUT
PIN  2          a14                             COMBINATORIAL ; INPUT
PIN  3          a13                             COMBINATORIAL ; INPUT
PIN  4          r_w                             COMBINATORIAL ; INPUT
PIN  6          a10                             COMBINATORIAL ; INPUT
PIN  7          a9                              COMBINATORIAL ; INPUT
PIN  8          a8                              COMBINATORIAL ; INPUT
PIN  9          E                               COMBINATORIAL ; INPUT
PIN  10         GND                                           ;
PIN  12         /rramck                         COMBINATORIAL ; OUTPUT
PIN  13         /ram                            COMBINATORIAL ; OUTPUT
PIN  14         /eprom                          COMBINATORIAL ; OUTPUT
PIN  15         /wramck                         COMBINATORIAL ; OUTPUT
PIN  16         /csclav                         COMBINATORIAL ; OUTPUT
PIN  17         /time                           COMBINATORIAL ; OUTPUT
PIN  18         /basc                           COMBINATORIAL ; OUTPUT
PIN  19         /csaff                          COMBINATORIAL ; OUTPUT
PIN  20         VCC                                           ;


;---------------------------------- Boolean Equation Segment ------
EQUATIONS
csaff=/a15*/a14*/a13*/a10*/a9*a8*E
basc=/a15*/a14*/a13*/a10*a9*/a8*E
time=/a15*/a14*/a13*/a10*a9*a8*E
csclav=/a15*/a14*/a13*/a10*/a9*/a8*E
ram=(/a15*/a14*a13*E)+(/a15*a14*/a13*E)+(/a15*a14*a13*E)
rramck=(/a15*/a14*a13*E*/r_w)+(/a15*a14*/a13*E*/r_w)+(/a15*a14*a13*E*/r_w)
wramck=(/a15*/a14*a13*E*r_w)+(/a15*a14*/a13*E*r_w)+(/a15*a14*a13*E*r_w)
eprom=(a15*/a14*/a13*E)+(a15*/a14*a13*E)+(a15*a14*/a13*E)+(a15*a14*a13*E)
;------------------------------------------------------------------
```

compilation

Fichier JEDEC

```
L1600 00000000000000000000000000000000*
L1632 00000000000000000000000000000000*
L1664 00000000000000000000000000000000*
L1696 00000000000000000000000000000000*
L1728 00000000000000000000000000000000*
L1760 00000000000000000000000000000000*
L1792 01101111101111111111111111110111*
L1824 11100111101111111111111111110111*
L1856 00000000000000000000000000000000*
      00000000000000000000000000000000*
      00000000000000000000000000000000*
      00000000000000000000000000000000*
      00000000000000000000000000000000*
L2016 00000000000000000000000000000000*
L2048 00000000000000000000000000000000*
L2080 00000000000000000000000000000000*
L2112 00000000000000001111111111111111*
L2144 11111111111111111111111111111111*
L2176 1111111111111110*
X0*
V0001 000XXXXX1NXHHHHLHHHN*
V0002 100XXXXX1NXHHLHHHHHN*
V0003 101XXXXX1NXHHLHHHHHN*
V0004 110XXXXX1NXHHLHHHHHN*
V0005 111XXXXX1NXHHLHHHHHN*
V0006 111XXXXX0NXHHHHHHHHN*
V0007 000XX0001NXHHHHLHHHN*
V0008 000XX0011NXHHHHHHHLN*
V0009 000XX0101NXHHHHHHHLHN*
V0010 000XX0111NXHHHHHLHHN*
V0011 0001X0111NXHHHHHLHHN*
V0012 0000X0111NXHHHHHLHHN*
```

1 : fusible grillé

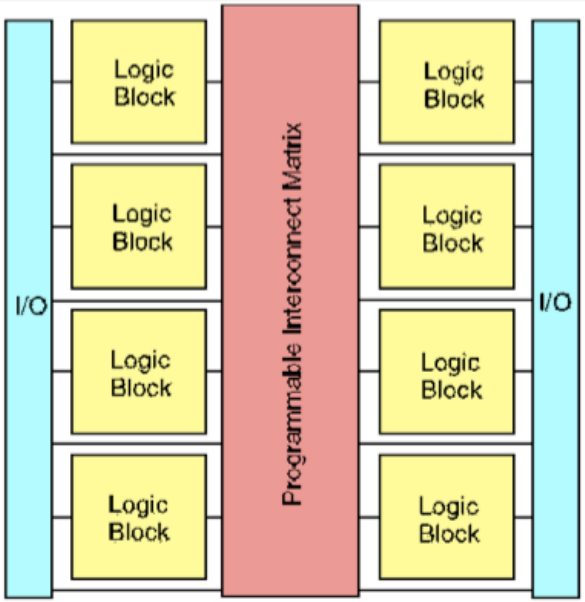
0 : fusible intacte

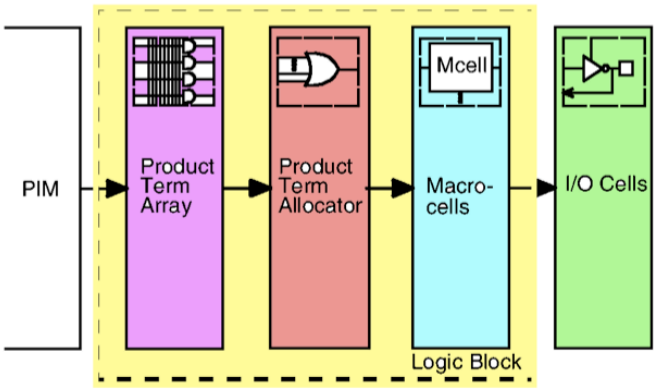
b. CPLD

Complex Programmable Logic Device

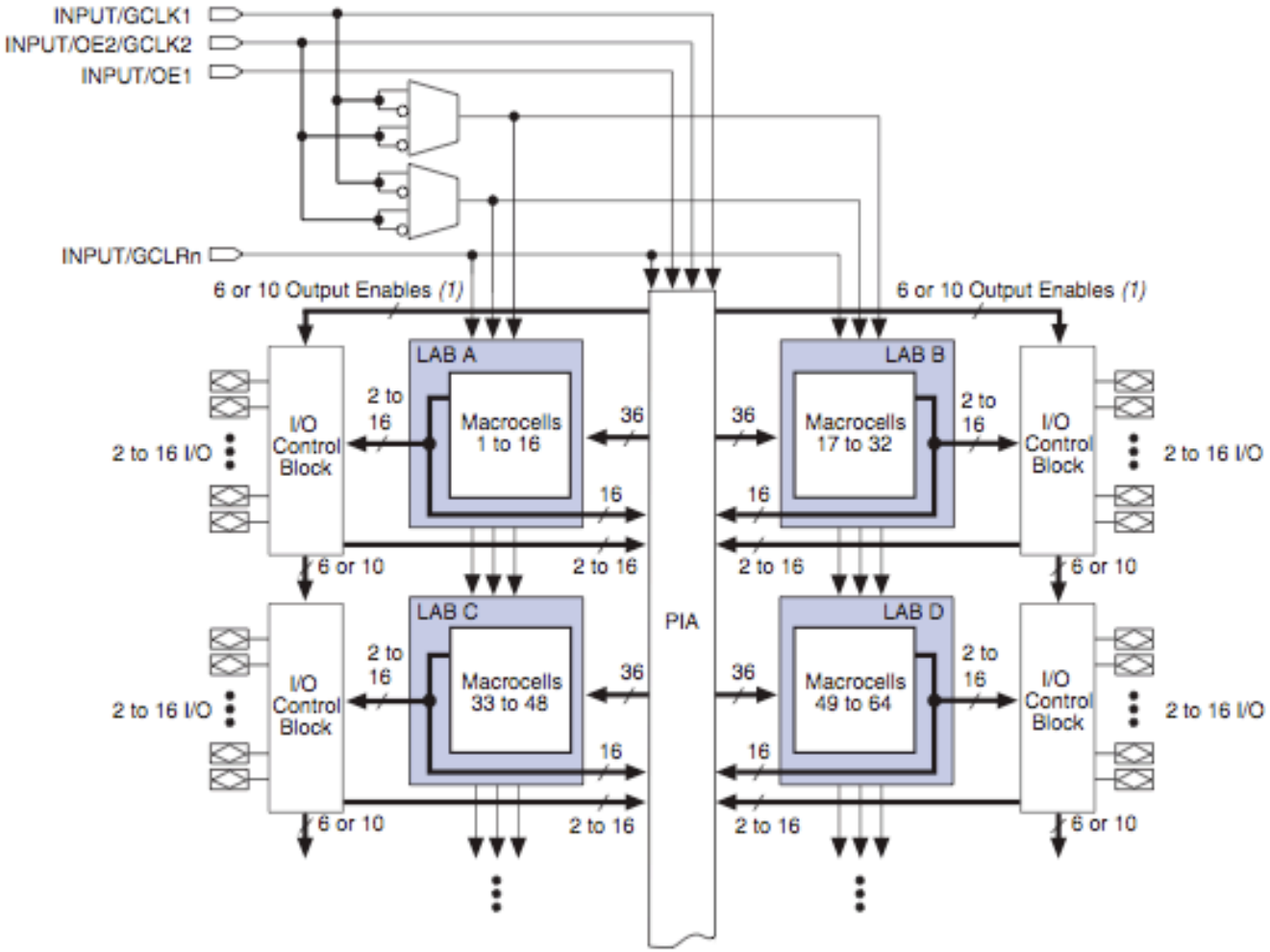
Technologies des CPLD

- Architecture des CPLD
  - Ensemble de fonctions de type PAL connectées via une matrice centrale programmable
  - Macrocellules
    - fonctions logiques de base regroupées en blocs logiques
      - EL (Altera)
      - CLB (Xilinx)





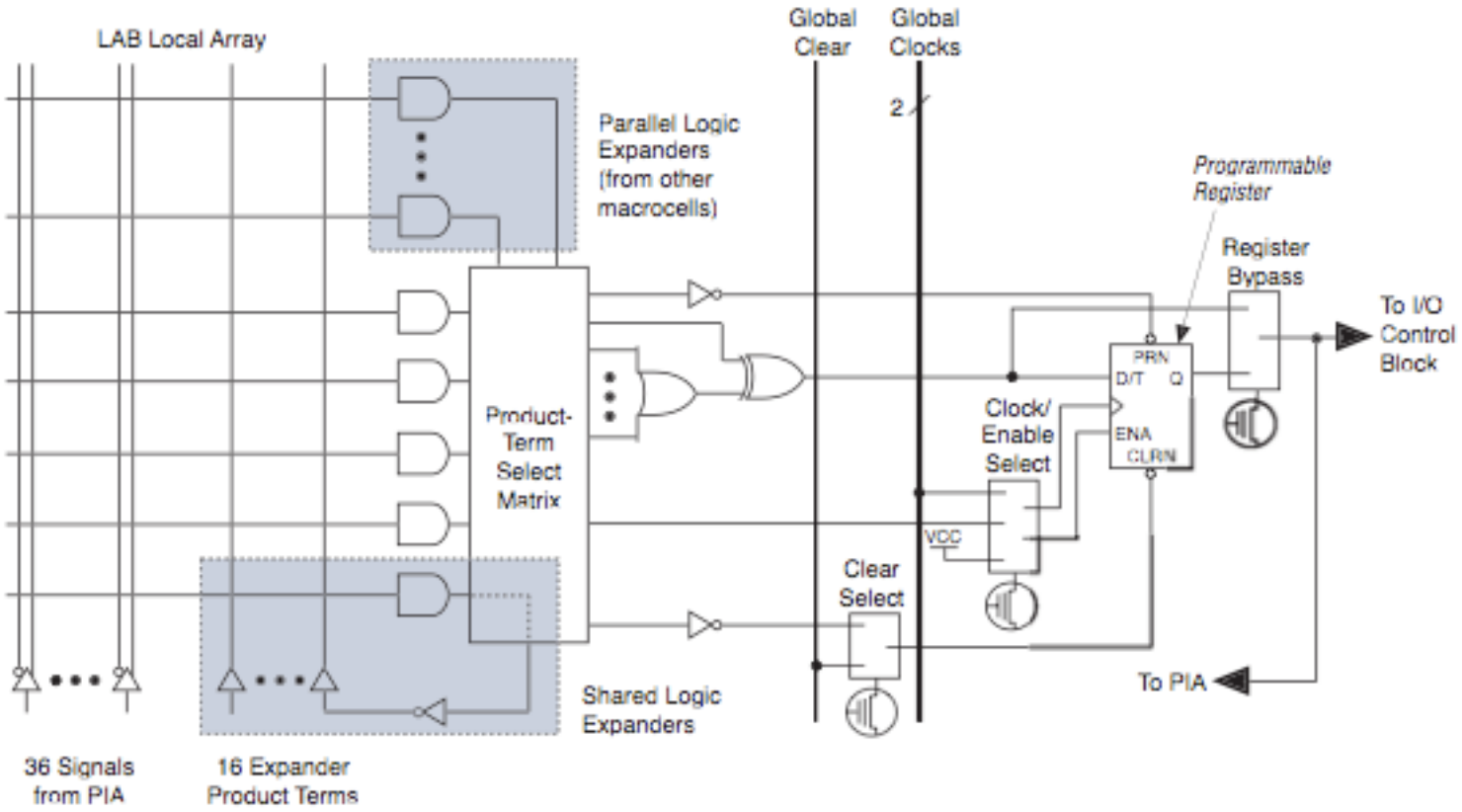
Altera Max3000



PIA : Programmable Interconnect Array

Altera Max3000

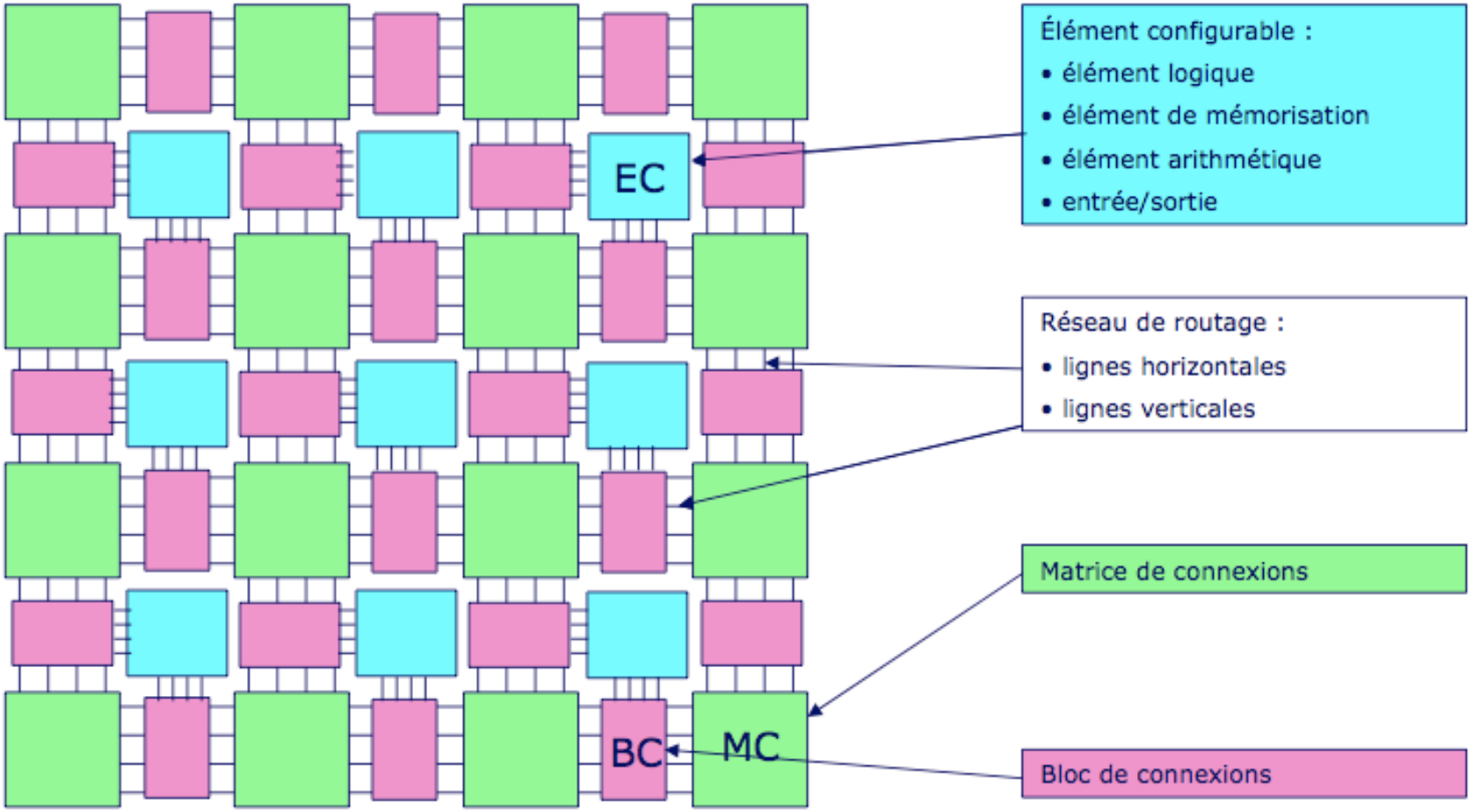
- Détails d'une macrocellule



c. FPGA

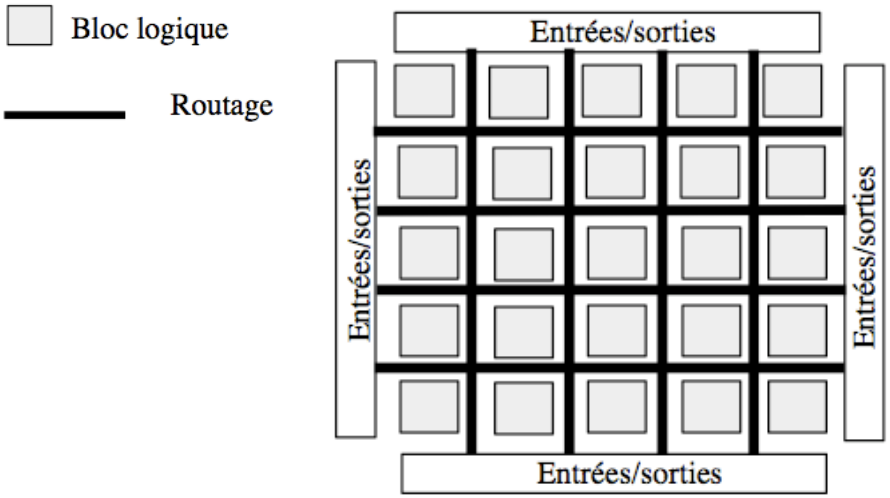
Field Programmable Gate Array

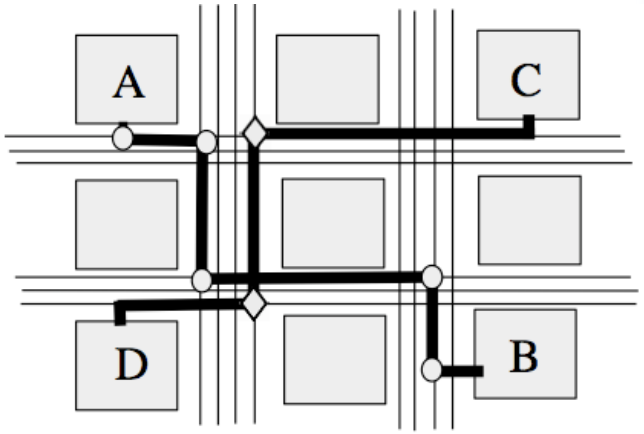
Architecture



Architecture

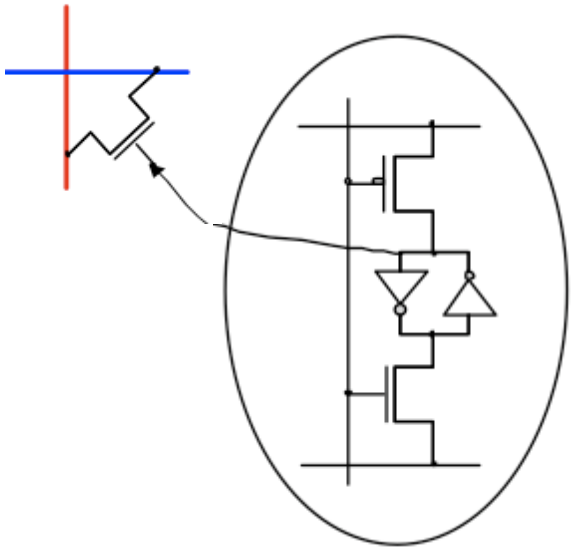
- Plusieurs chemins de routages possibles entres les blocs
- Variations des performances
- Dépend des performances des outils de synthèse et de placement/routage

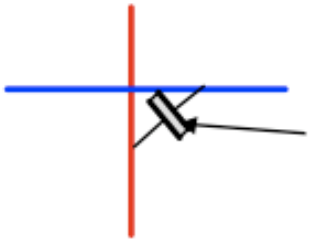




Technologie de configuration

- L'implémentation d'un système numérique dans un FPGA nécessite de :
  - Configurer les éléments fonctionnels
  - Configurer les interconnections (ligne de routage)
- 3 technologies de configuration => 3 familles technologiques de FPGA
  - Technologie antifuse
  - Technologie Flash EPROM
  - Technologie SRAM





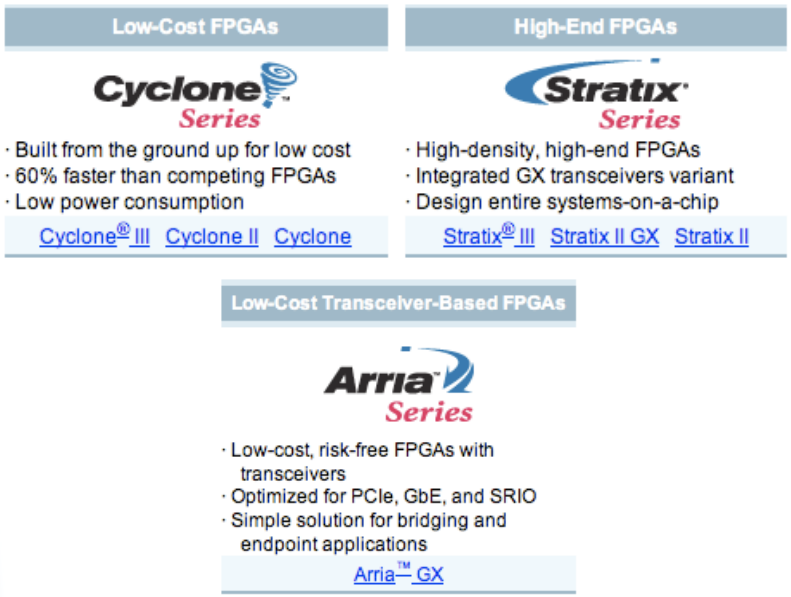
interconnexions par antifusible

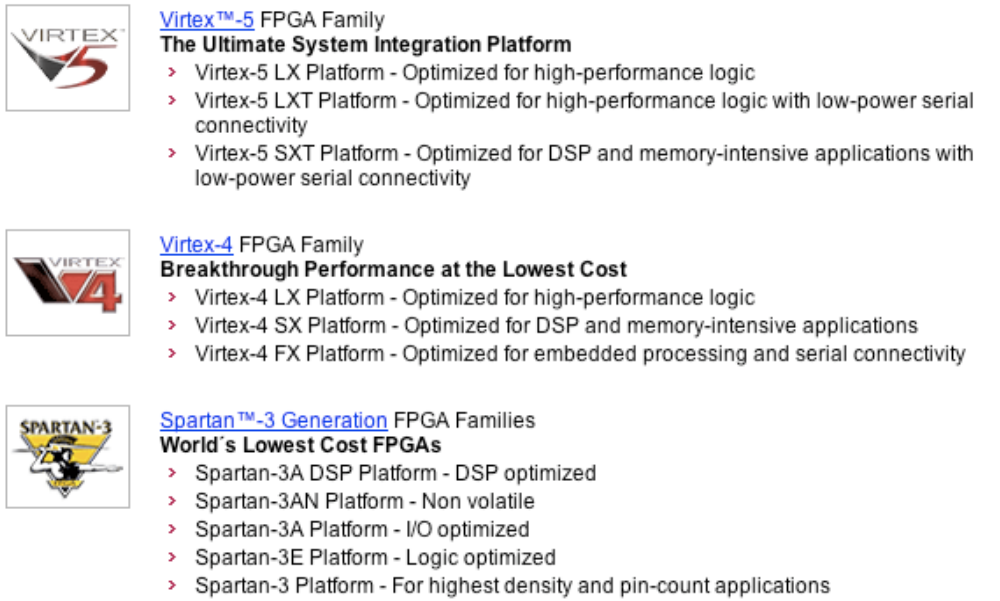
Technologie de configuration

Technologie	Antifusible	Flash	SRAM
Densité	Grande (1 antifusible / cellules)	Moyenne (2 transistors / cellules)	Faible (6 transistors / cellules)
Consommation (Watt)	Faible	Moyenne	Moyenne à grande
Rapidité (fréquence)	Faible	Moyenne	Moyenne à grande
Reconfiguration	NON	Oui (In Situ ou Offline)	Oui (In Situ)
Temps de programmation	--	Lente (3 X plus qu'avec SRAM)	Rapide
Configuration	Permanente (irréverssible)	Non Volatile (réverssible)	Volatile (sans alimentation)
Prototypage ?	NON	Oui	OUI (Très bon)
Sécurité de la conception (IP)	Très bonne	Très bonne	Moyenne à grande
Sensibilité aux radiations	Très faible	Moyenne	Grande
Fabricants	Actel, QuickLogic	Actel, Cypress, Lattice	Xilinx, Altera, Atmel, Lattice

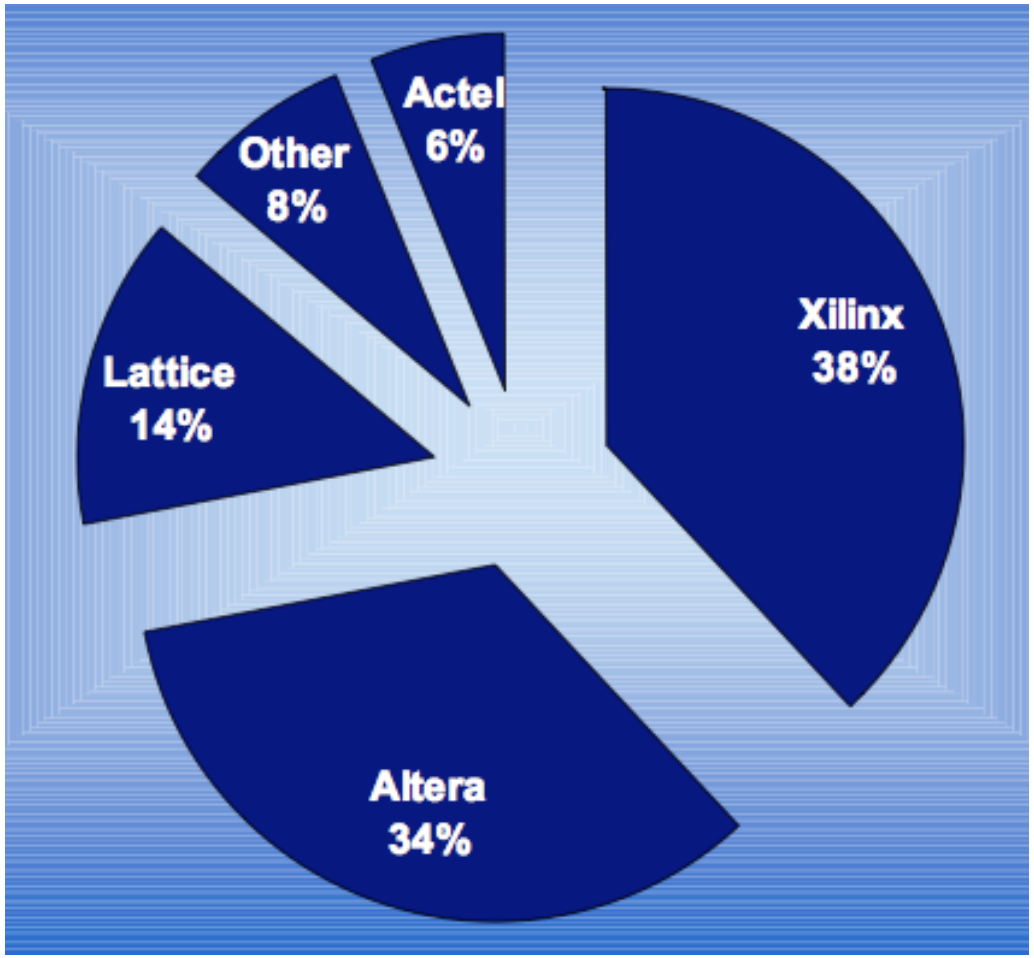
Fabricants de FPGAs

- Les fabricants de FPGA sont les mêmes que les SPLD et CPLD
  - Essentiellement 2 grands constructeurs ALTERA et Xilinx
  - Altera
    - 3 familles : Cyclone, Stratix et Arria
  - Xilinx
    - 2 familles : Spartan et Virtex



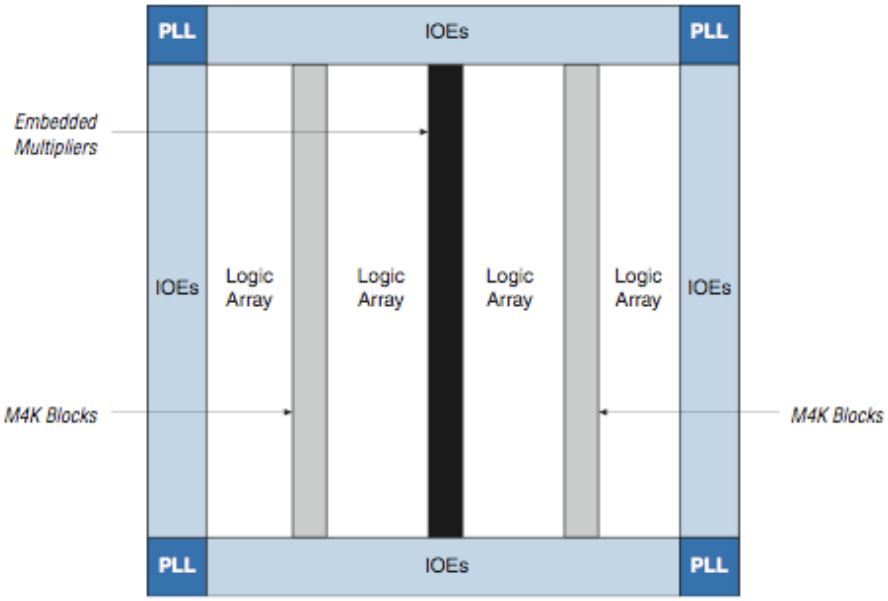


Fabricant de FPGAs



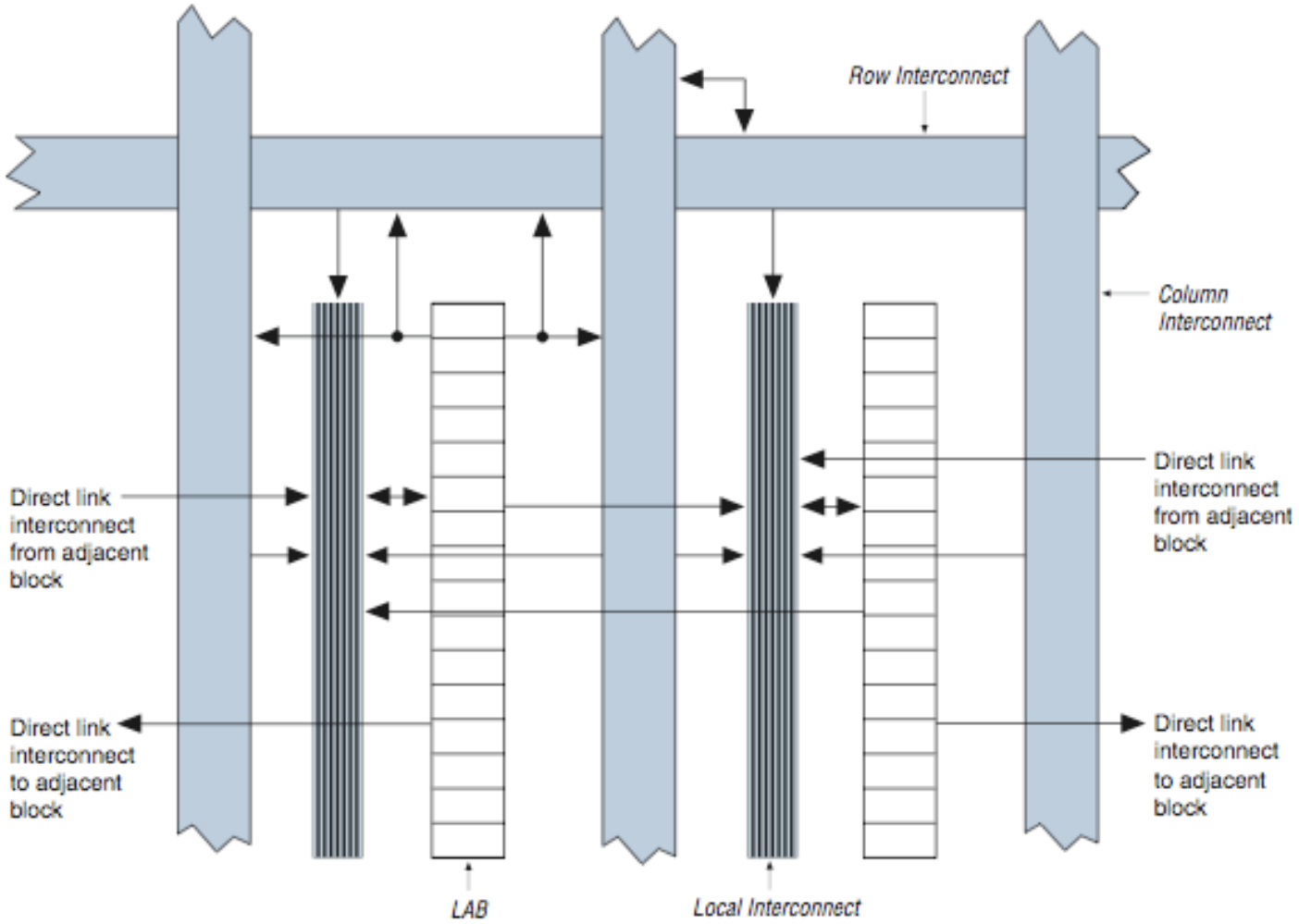
Altera Cyclone II

- Architecture
  - Structure matricielle de
    - Bloc logique (LAB)
    - Bloc mémoire (M4K)
    - Multiplieur
  - LAB
    - composé de 16 éléments logiques (LE)
    - De 4608 à 68416 LE / cyclone2
  - 4 PLL pour la distribution des signaux d'horloge
  - M4K blocs
    - Mémoire double port
    - 36bits / 260 MHz



  - Multiplieur
    - 8x8 bits
    - 19x19 bits
    - 250MHz

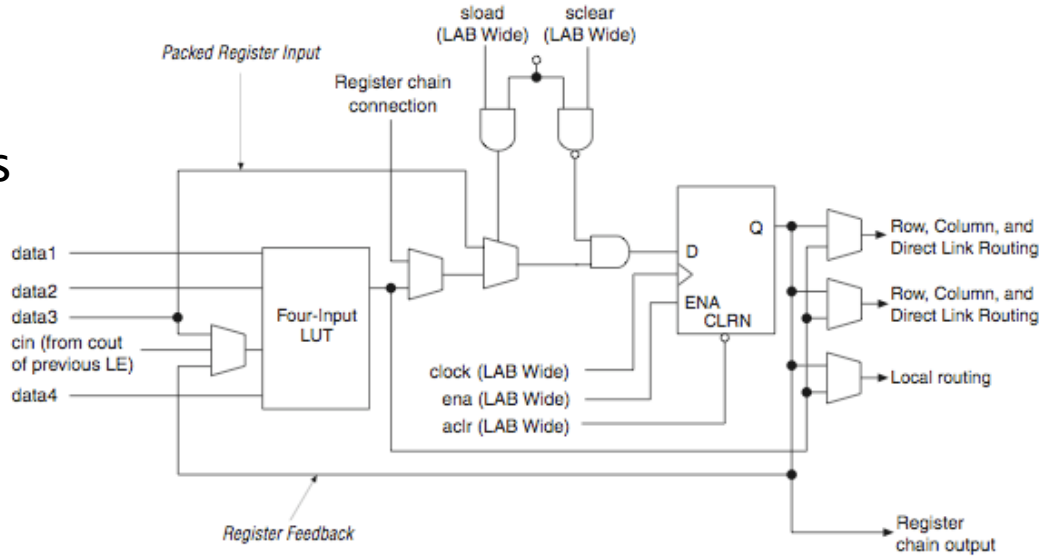
Altera Cyclone II



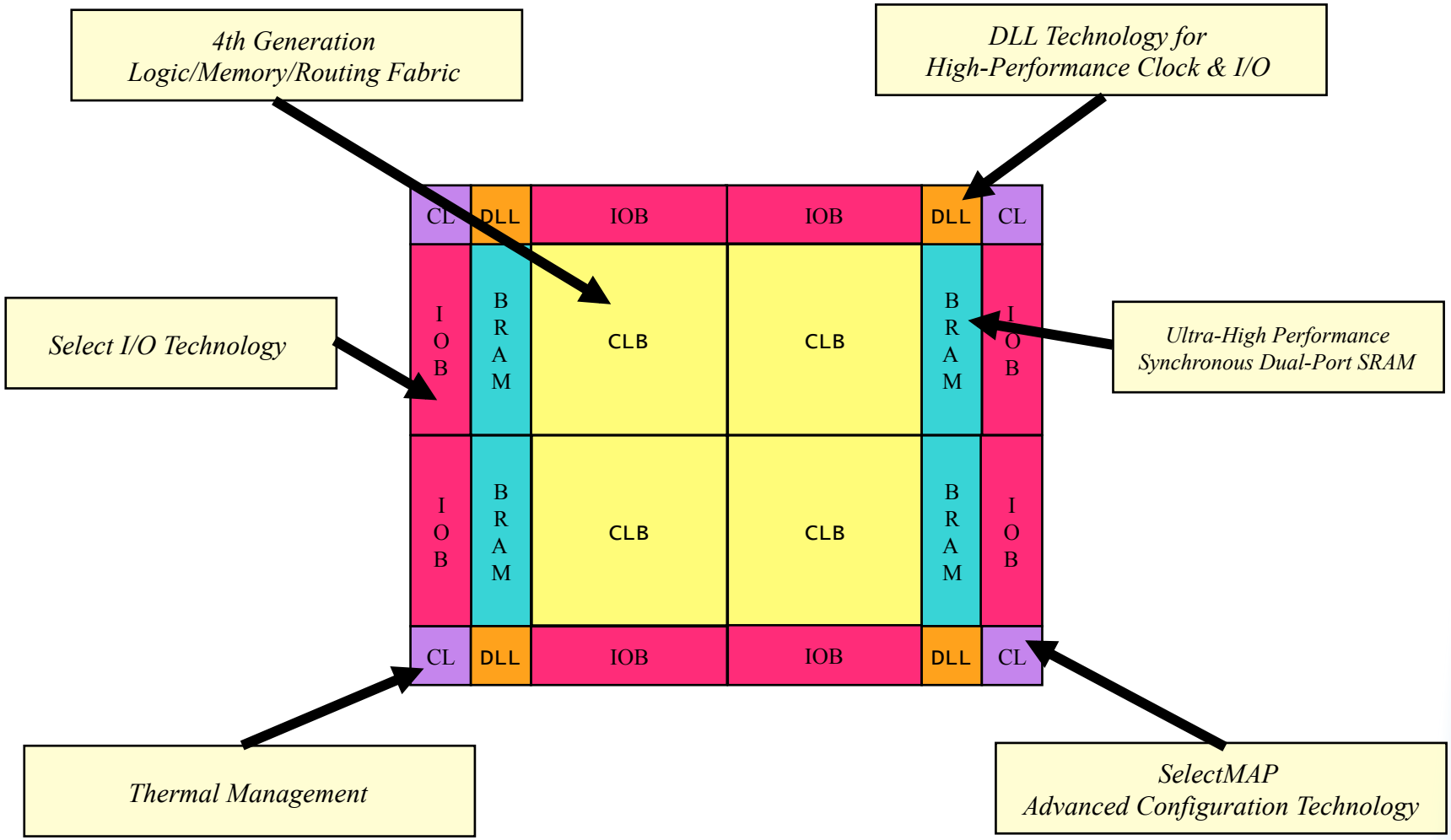
Altera Cyclone II

- Logic Element

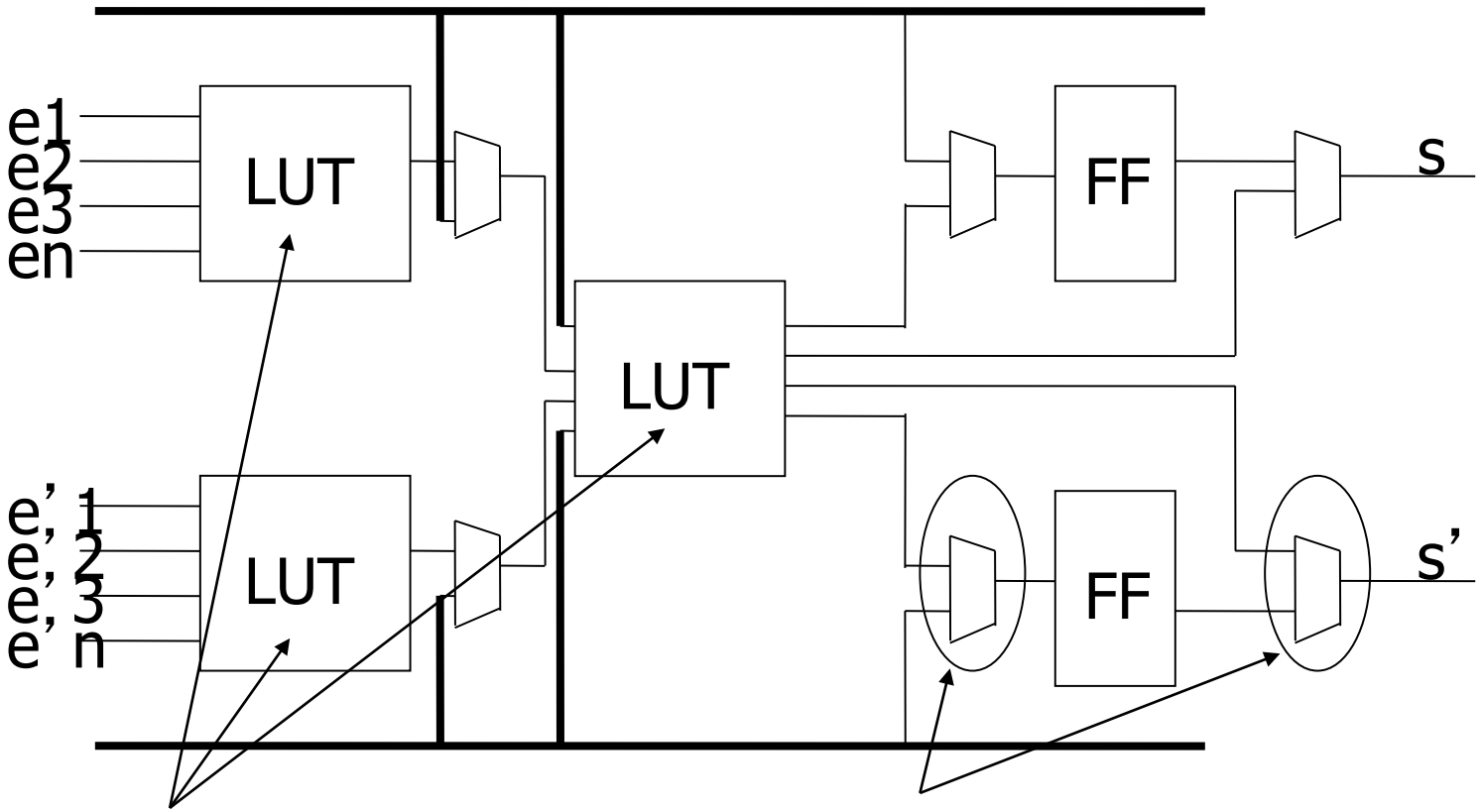
- Représente la plus petite entité logique
  - LUT : fonction à 4 variables
  - Registre programmable
  - Interface d'accès aux bus colonne et ligne
  - Signaux d'interconnexion directe aux LE voisins
    - Propagation de signaux de retenue par exemple
  - 2 modes de fonctionnement / compilateur



XILINX : Famille VIRTEX

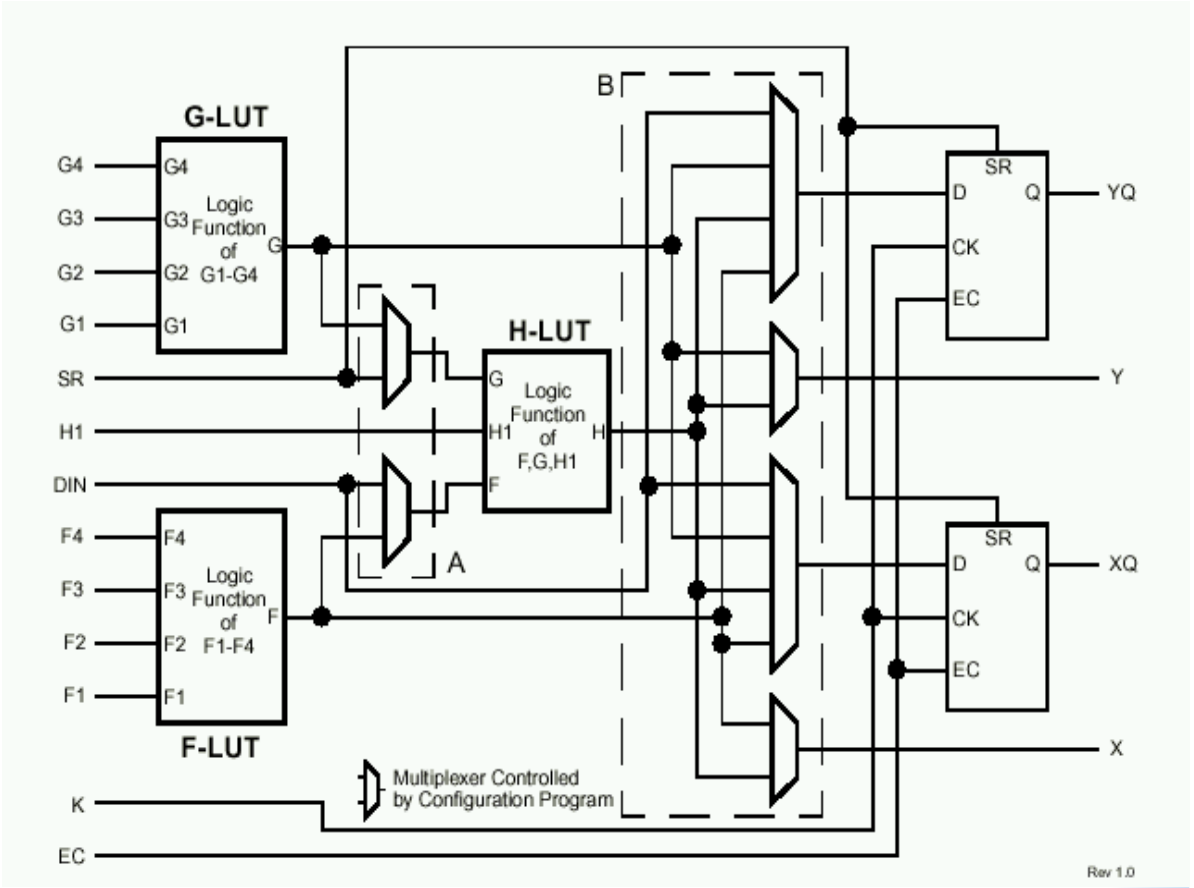


Architecture d'un CLB



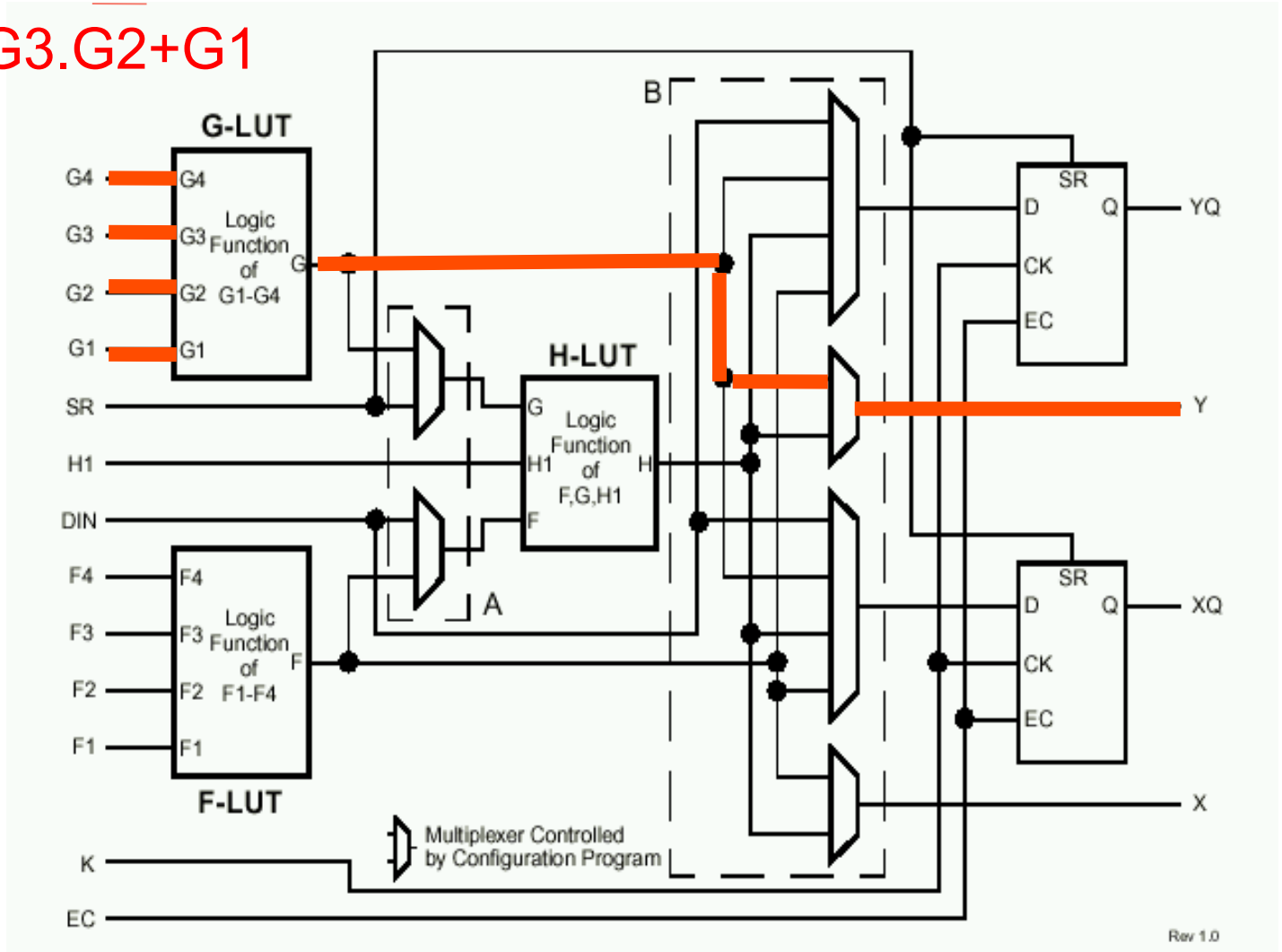
Exemple de CLB

- Chaque LUT peut réaliser une fonction combinatoire
- Des fonctions combinatoires et séquentielles peuvent être réalisées par combinaisons des ressources



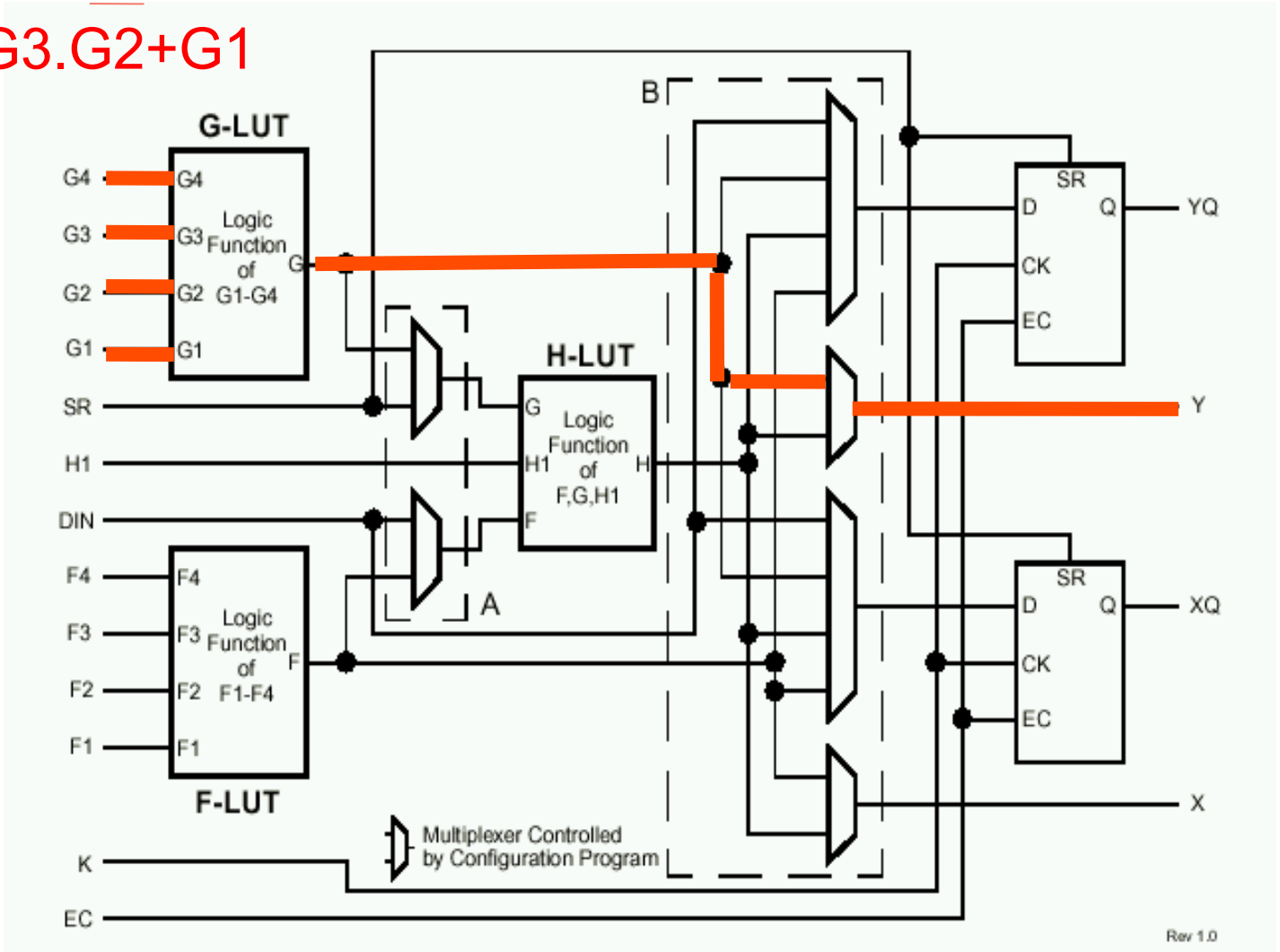
Fonction combinatoire réalisée avec un CLB

$y=G4+\overline{G3}.\overline{G2}+G1$



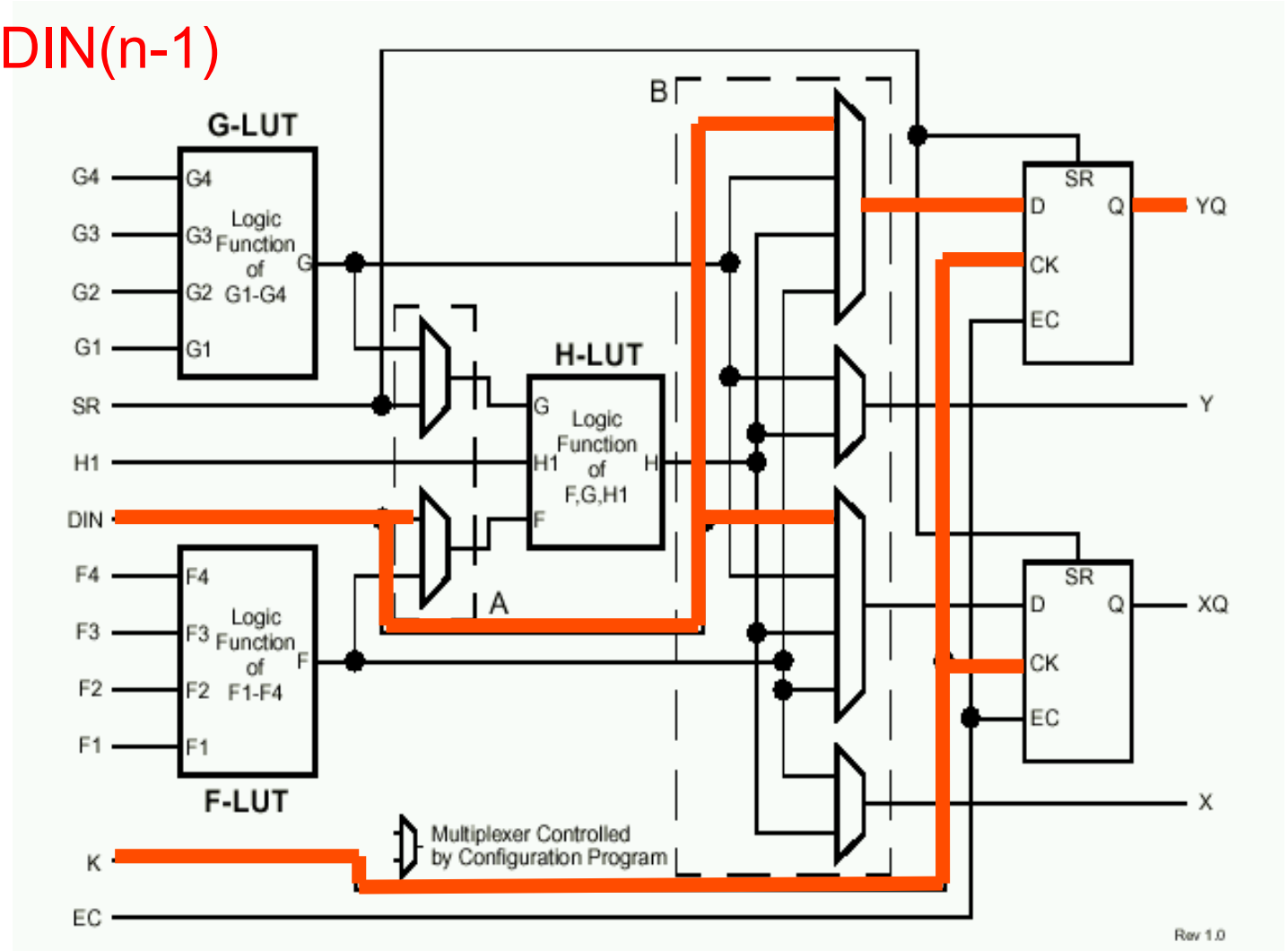
Fonction combinatoire réalisée avec un CLB

$y=\overline{G4}+\overline{G3}.\overline{G2}+G1$



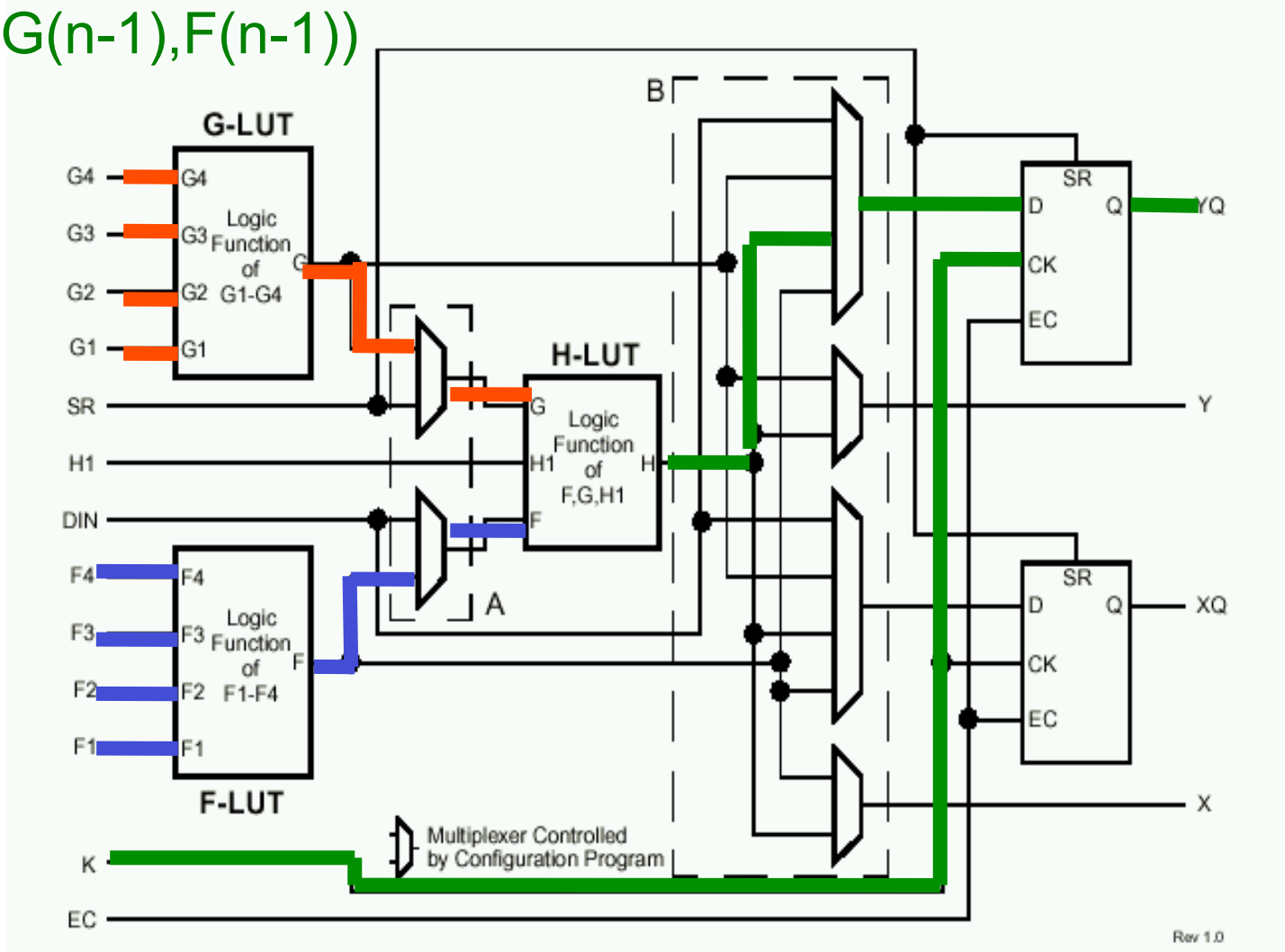
Fonction séquentielle réalisée avec un CLB

YQ(n)=DIN(n-1)

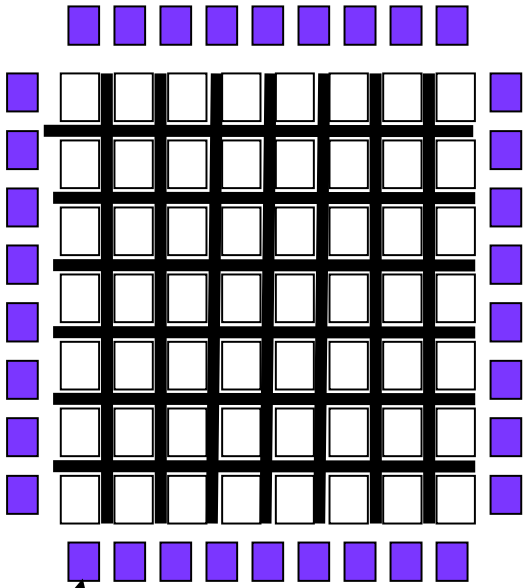


Fonction complexe réalisée avec un CLB

YQ(n)=H(G(n-1),F(n-1))



Structure des entrées/sorties IO



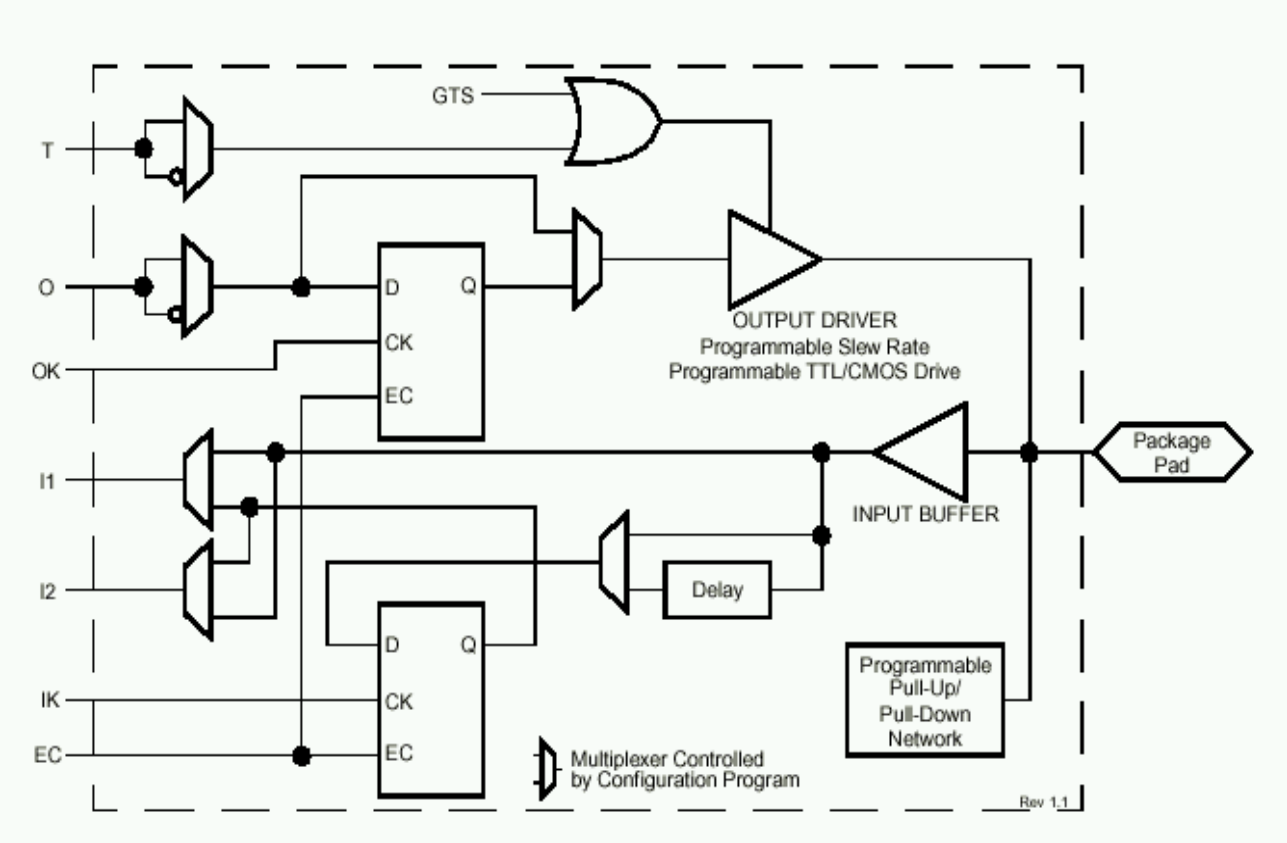
IO (Ports d'E/S)
input/output blocks

Ports dentrée/sortie configurables :

- Seuils d'entrée TTL ou CMOS
- Buffer de sortie programmable en haute impédance
- Entrées et sorties directes ou mémorisées
- Inverseur programmable

IO d'un FPGA Xilinx

- Détails d'un IOB (Input Output Block)

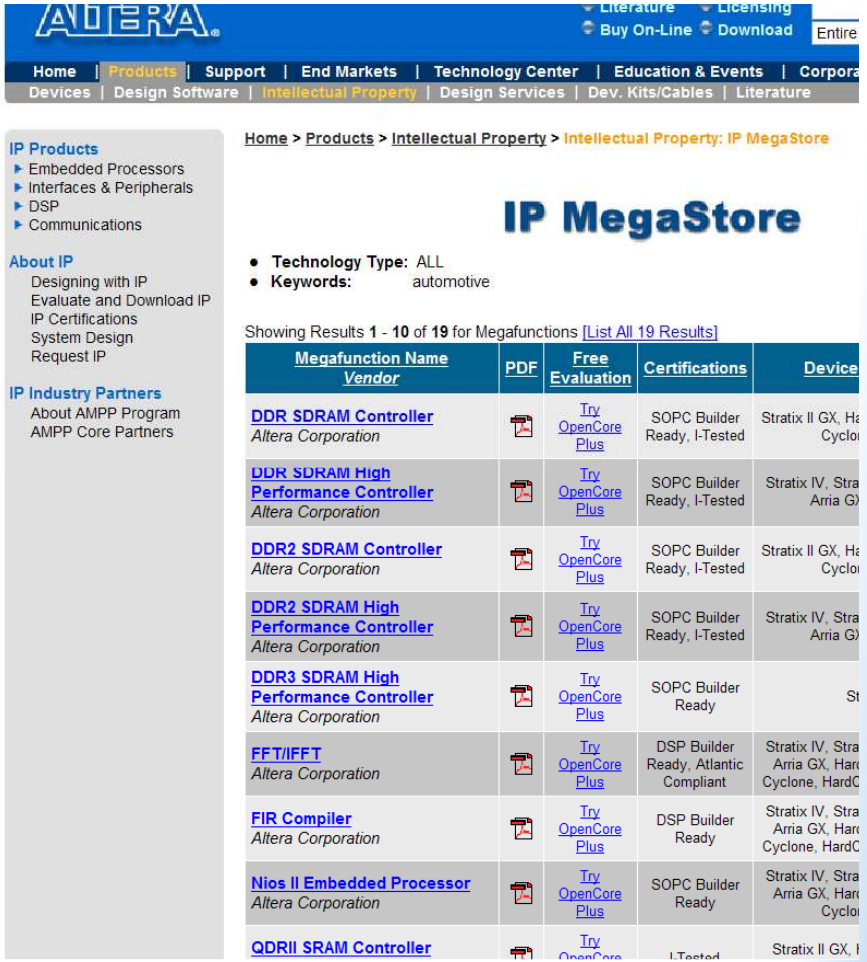


Autres ressources

- Les FPGAs de dernière génération possèdent suivant les constructeurs, des ressources matérielles spécifiques implantées directement sur le circuit comme :
  - Bloc de calcul :
    - DSP block (multiplieur 18x18 bits)
  - Mémoires
    - RAM, ROM, DPRAM, etc.
  - Micro-processeur
    - ARM, Nios II
  - Périphérique de communication : RapidIO, …
  - Etc..

Autres ressources

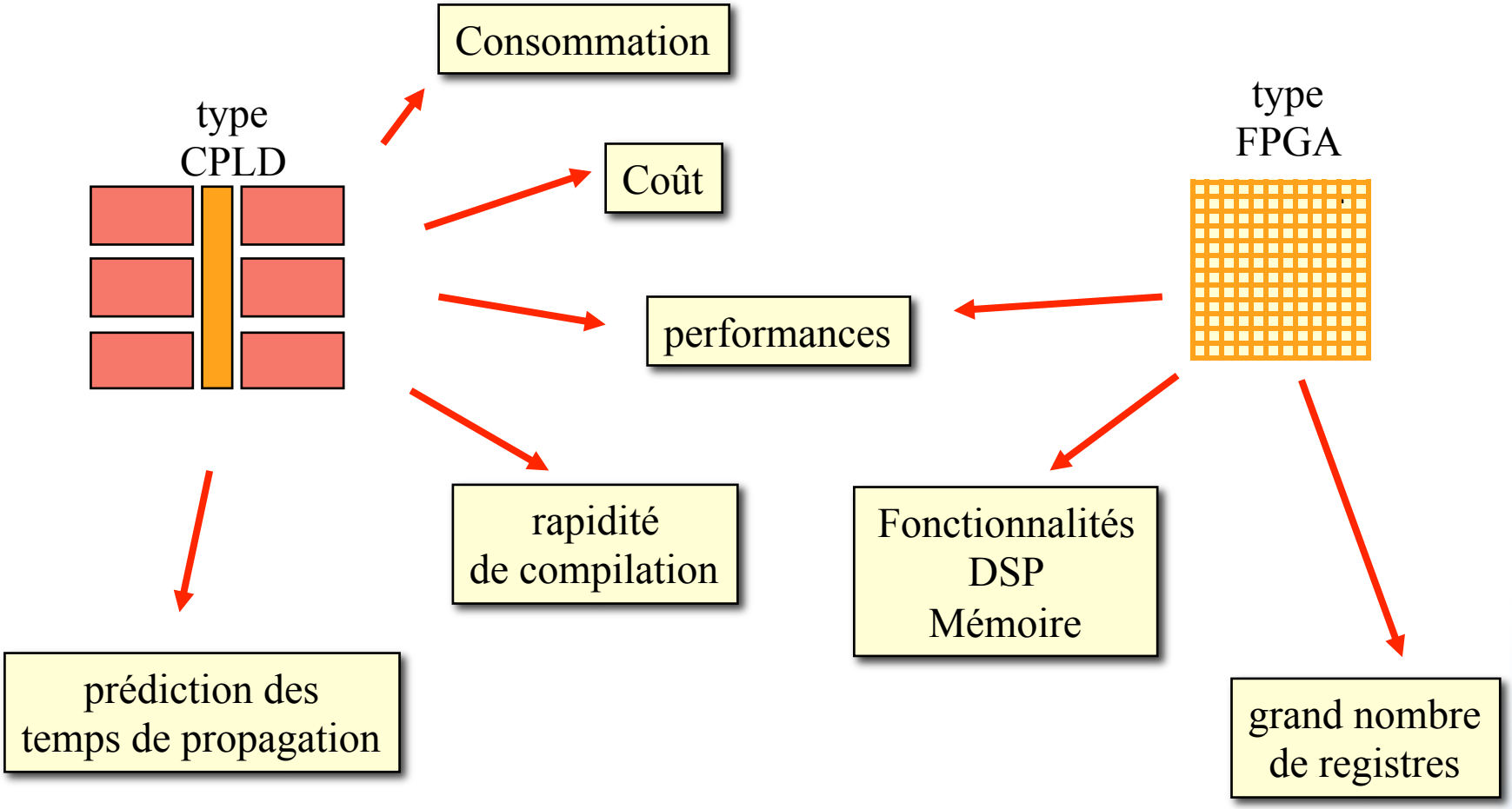
- Chaque constructeur propose des bibliothèques de composants appelés aussi IP (Intellectual Property)



Résumé

- PAL :
  - Architecture de type PLA avec matrice de ET programmable et OU fixe : quelques portes
  - NB I/O : environ une vingtaine
  - Fréquence max : 250 MHz
- GAL :
  - Même architecture que les PALs mais effaçable
- CPLD:
  - Plusieurs GAL autour d'un interconnect programmable
  - 10k portes
  - NB I/O : + d'une centaine
  - Fréquence max : 200 MHz
- FPGA
  - Bus d'interconnexion matriciel
  - Matrice de GAL
  - Multiplieur câblé
  - Mémoire
  - +1 million de portes
  - + 500 I/O
  - Fréquence max : + 300MHz

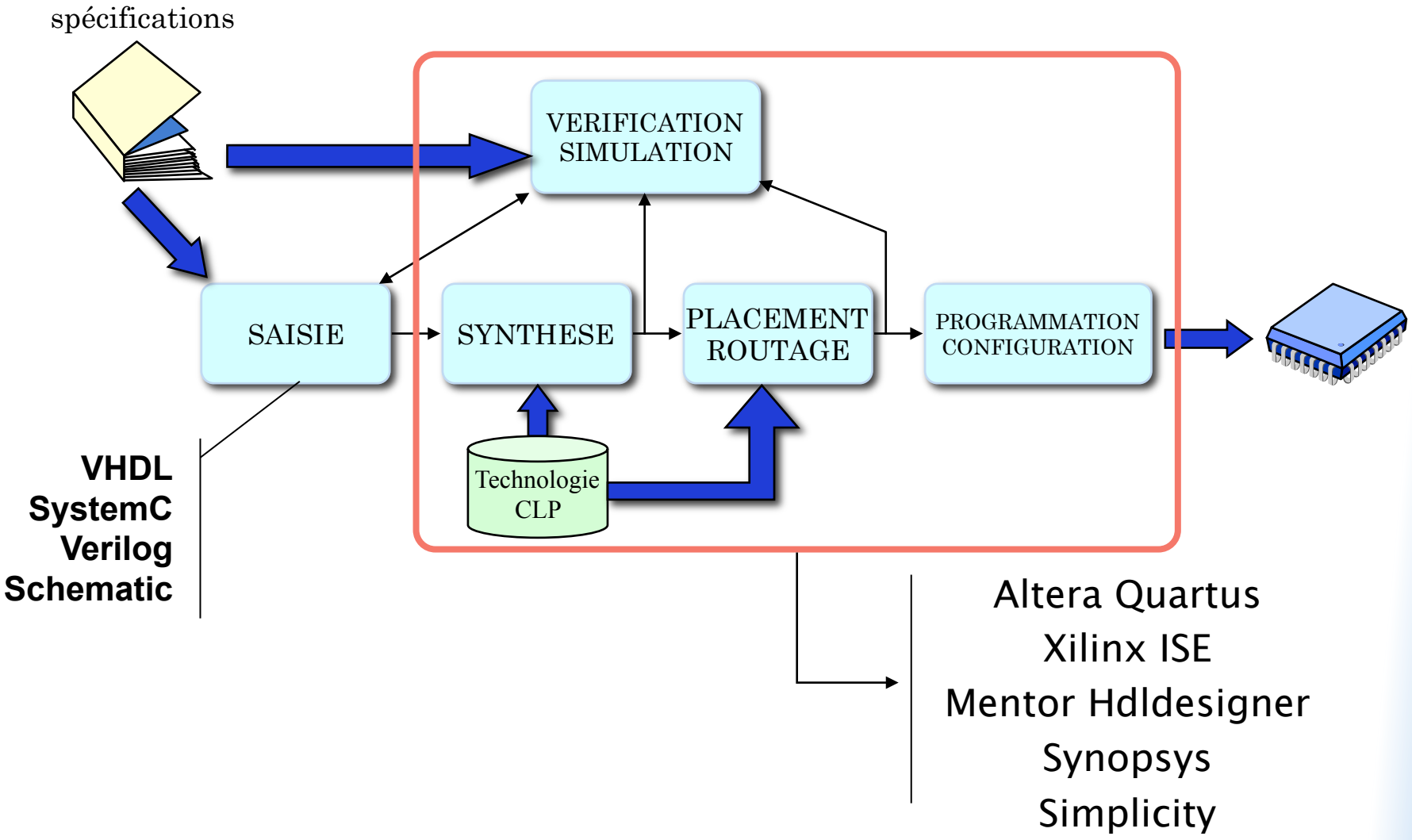
Résumé


Consommation
type
CPLD
Coût
performances
type
FPGA
rapidité
de compilation
Fonctionnalités
DSP
Mémoire
prédiction des
temps de propagation
grand nombre
de registres

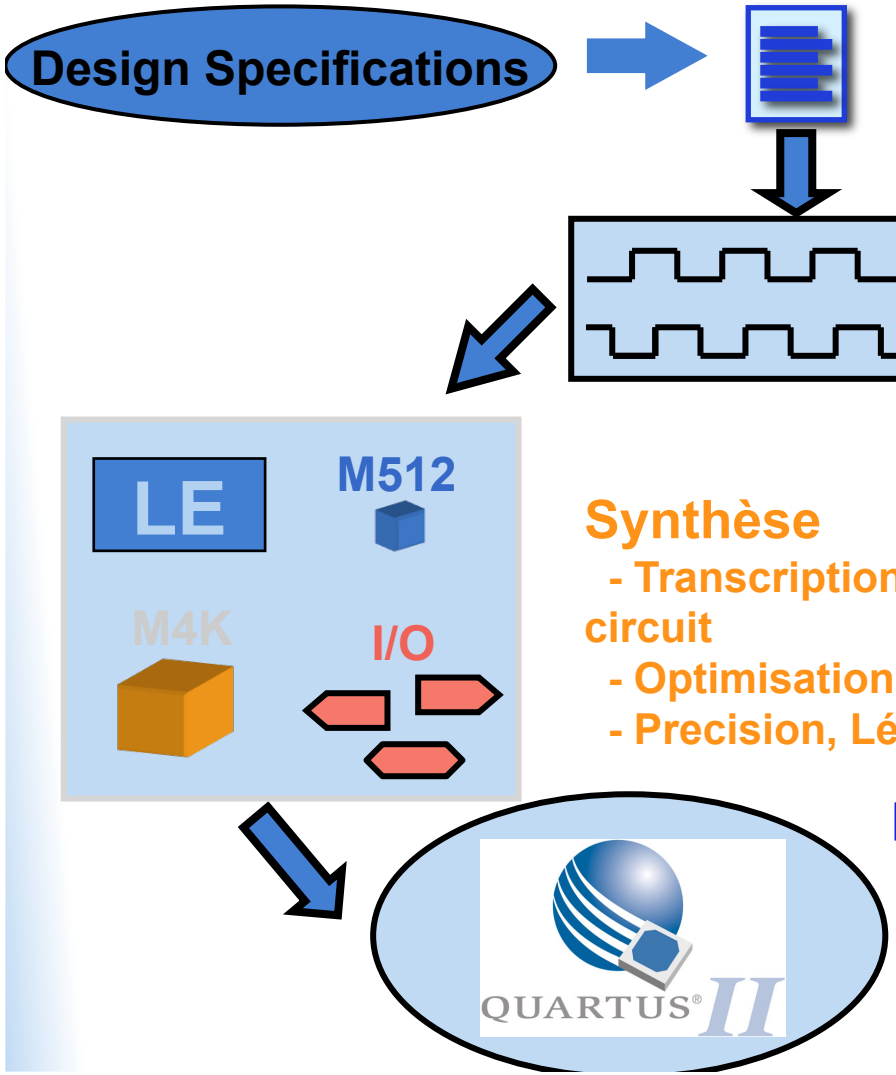
4. PAO

Programmation Assisté par Ordinateur

Flot de conception



Flot de conception



Design Entry/RTL Coding

- Description du Design comportemental ou structurel

Simulation

- Simulation fonctionnelle (Modelsim, Quartus)
- Verification Logique du Modèle & Data Flow

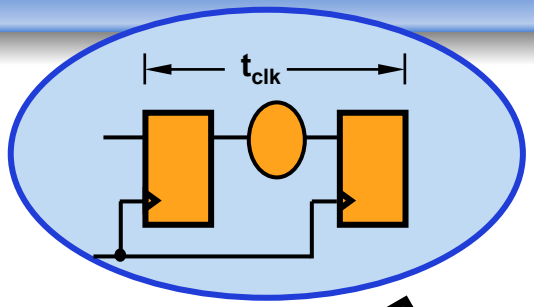
Synthèse

- Transcription du Design en primitives spécifiques à un circuit
- Optimisation des Performances / Contraintes
- Precision, Léonardo Spectrum ou Synplify sous Quartus II

Placement & Routage

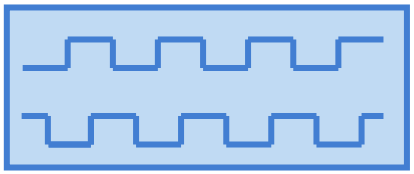
- Map Primitives to Specific Locations Inside Target Technology with Reference to Area & Performance Constraints
- Specify Routing Resources to Be Used

Flot de conception



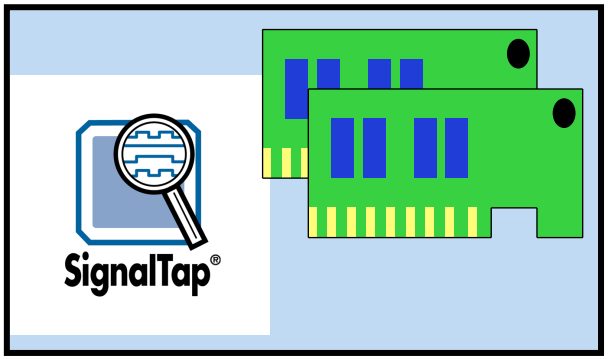
Analyse du temps : simulation post synthèse

- Verifications des Performances / Specifications
- Analyse du timing (static)



Simulation au niveau porte

- Simulation temportelle
- Verification du Design / circuit



PC Board Simulation & Tests

- Simulate Board Design
- Program & Test Device on Board
- Use SignalTap II for Debugging

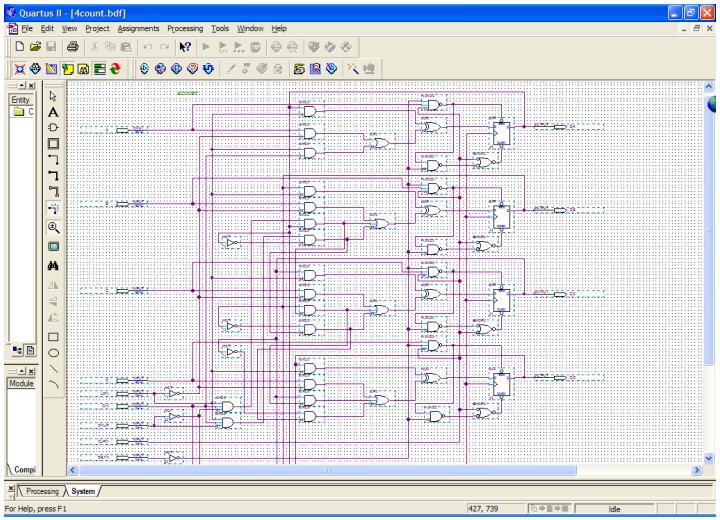
Flot de conception

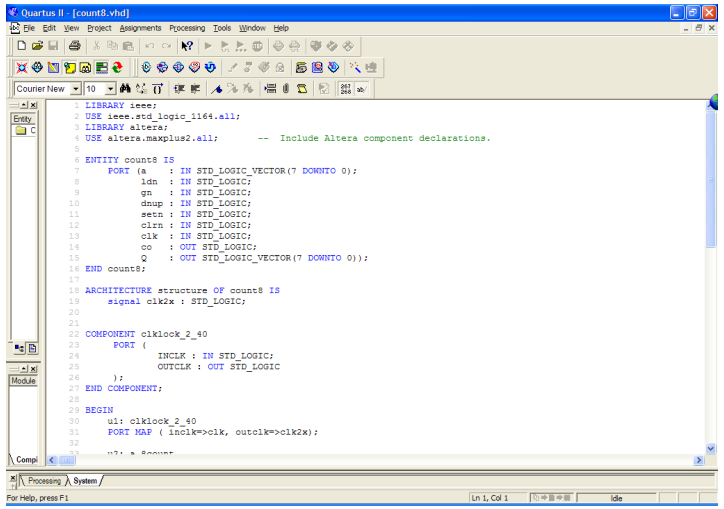
- Flot descendant « Top-Down »

- Composé de 6 grandes étapes :
  1. Codage
  2. Simulation fonctionnelle
  3. Synthèse
  4. Placement routage
  5. Simulation post-routage (niveau porte)
  6. Programmation et test

1. codage

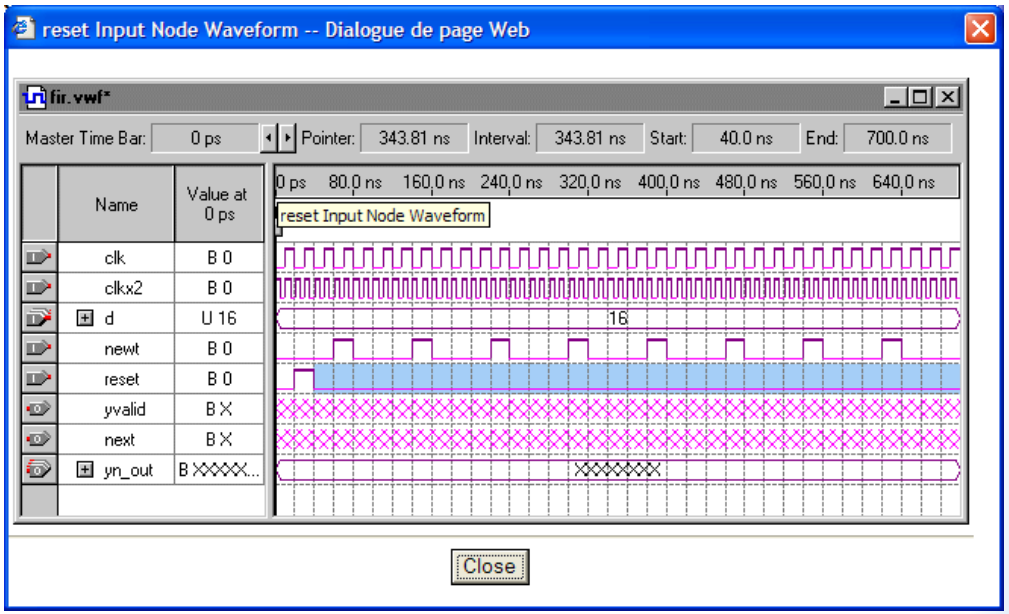
- Description de la fonction numérique à réaliser
- 2 possibilités
  - Codage schématique
    - Bibliothèque de fonction constructeurs (peu portable)
  - Codage par langage de programmation
    - High Description Language





2. Simulation fonctionnelle

- Vérification du comportement de la description
  - Génération de vecteur de test en entrée
  - Observation des sorties
  - TestBench
- Remarque : les paramètres dynamiques des blocs sont ignorés

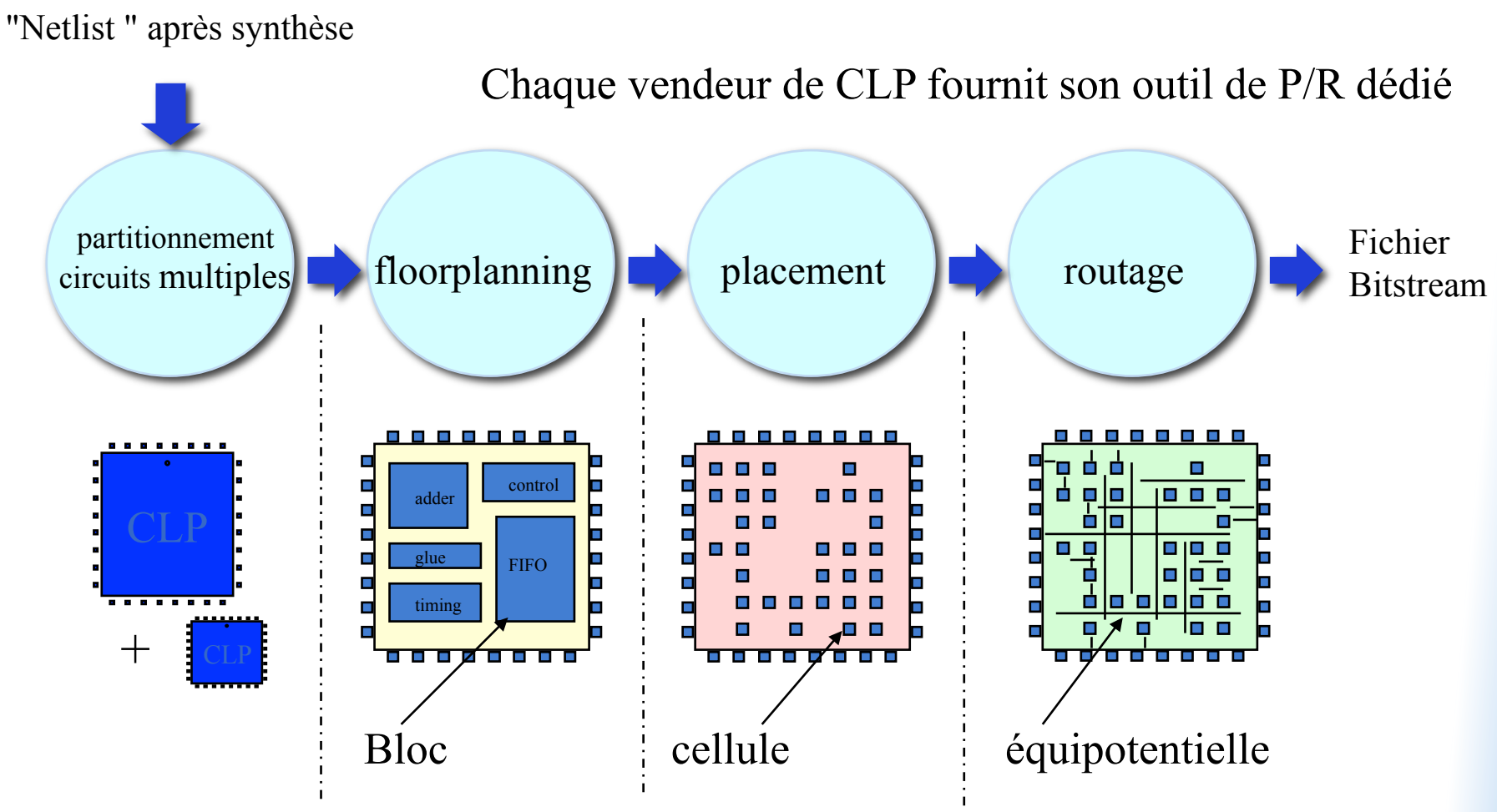


3. Synthèse

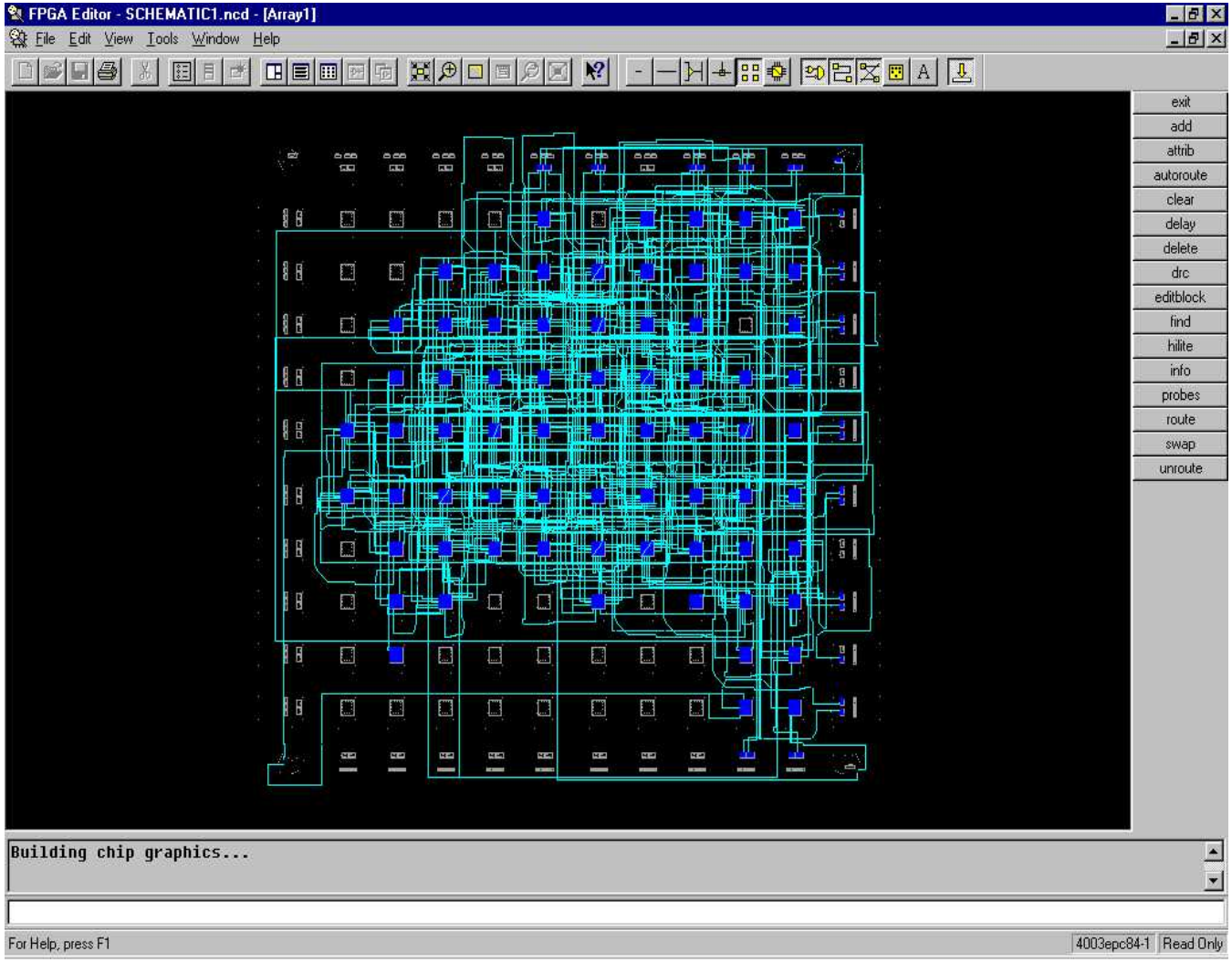
- Transcription de la description schématique ou textuelle en primitives disponibles dans le circuit
- Optimisation paramètres dynamiques de la description en fonction des contraintes imposées (surface, consommation, fréquences, …)
- Génération d'une Netlist
  - Format EDIF

```
(cell XORCY (cellType GENERIC)
   (view PRIM (viewType NETLIST)
    (interface
     (port LI (direction INPUT))
     (port CI (direction INPUT))
     (port O (direction OUTPUT))
    )
   )
)
```

4. Placement routage



Exemple de PR



5. Simulation post-routage

- Pourquoi refaire une seconde simulation ?
  - Après placement routage dans le circuit, les paramètres dynamiques peuvent avoir changés avec les temps de propagation entre chaque bloc

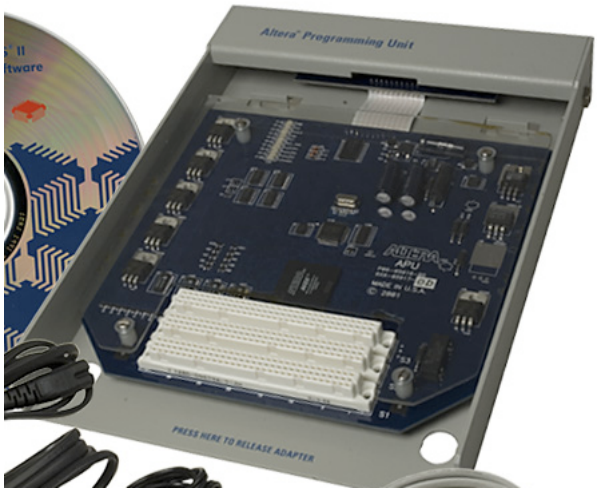
- Simulation post-routage
  - Test Bench identique à la simulation fonctionnelle
  - Vérification du comportement
    - Le comportement dans du système dans le circuit sera identique aux résultats obtenus par simulation

6. Programmation et test

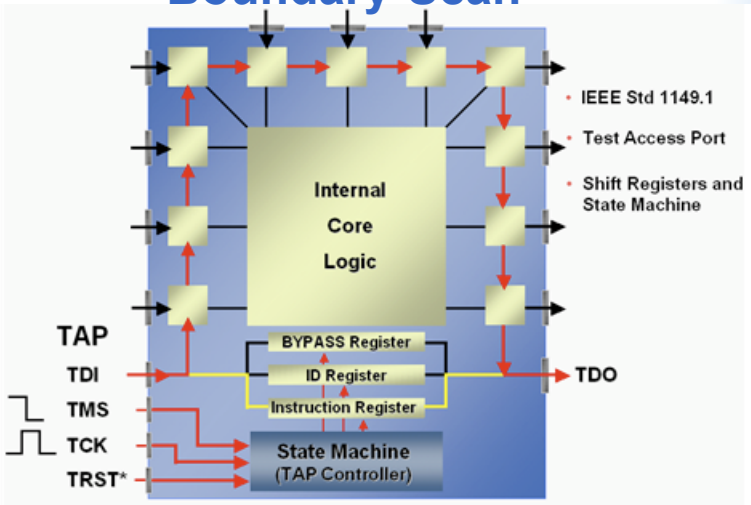
- Programmation du circuit
  - Génération d'un bitstream
    - Bitstream = suite de bits de configuration donnant l'état des fusibles, des entrées sorties, etc.
  - Utilisation d' un programmateur
  - Utilisation du port JTAG FPGA

- Test
  - Reproduction du test bench
  - Utilisation d'un analyseur logique

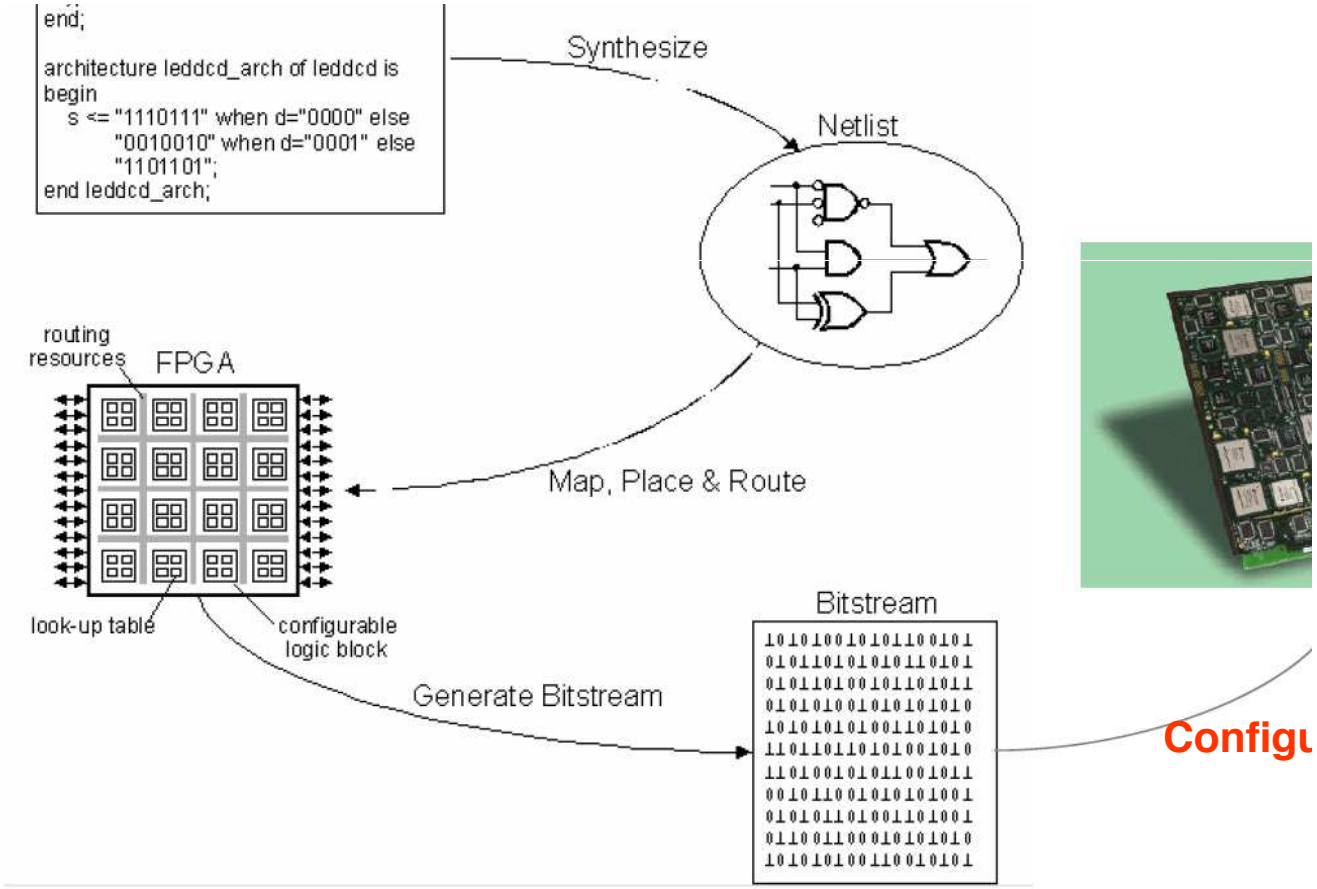
Programmateur Altera



Boundary-Scan



Résumé : Flot de conception



5. Altera Quartus

Quartus II

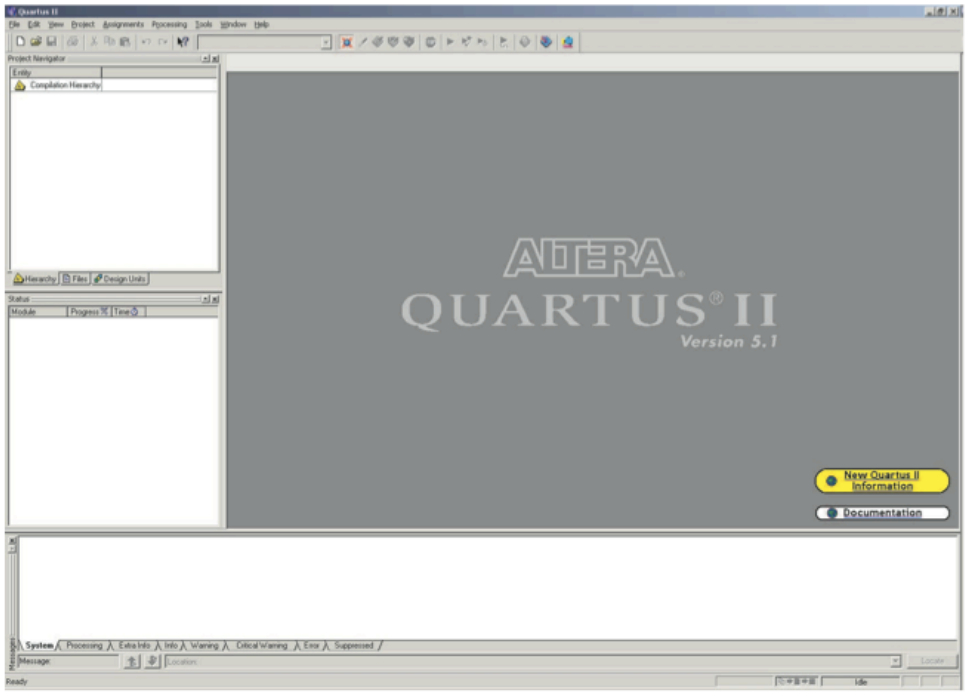
- Logiciel de CAO développé par ALTERA pour la programmation des circuits SPLD, CPLD et FPGA

- Ensemble d'applications
  - Saisie du code
  - Simulation fonctionnelle
  - Simulation post-synthèse
  - Synthèse
  - Placement Routage et programmation

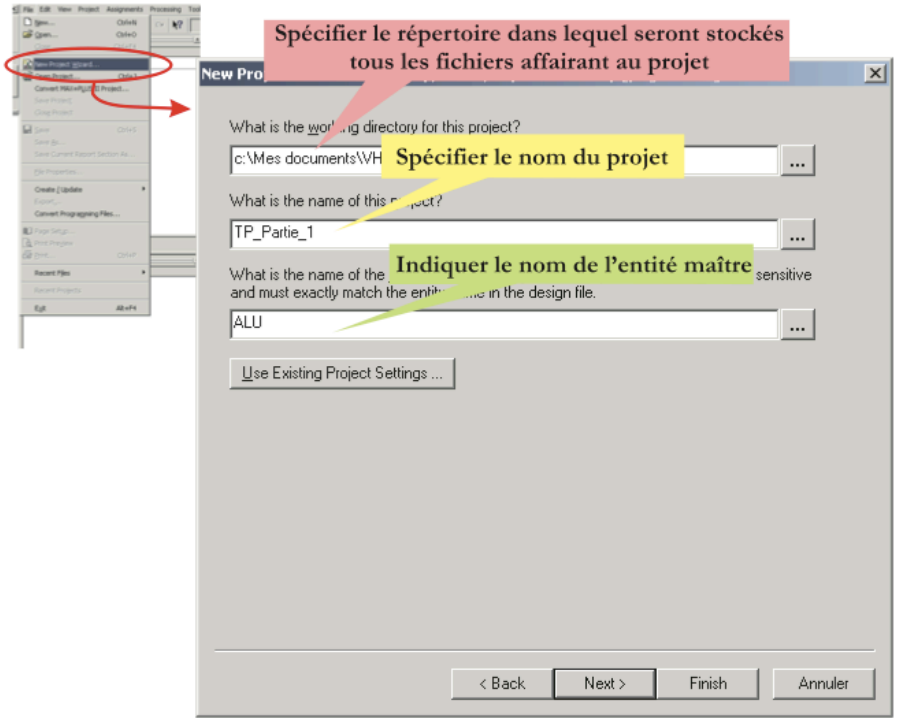
- Compatible avec d'autre outil EDA
  - Modelsim
  - FPGA compiler
  - Precision
  - Etc…

Quartus II

- Fenêtre d'accueil

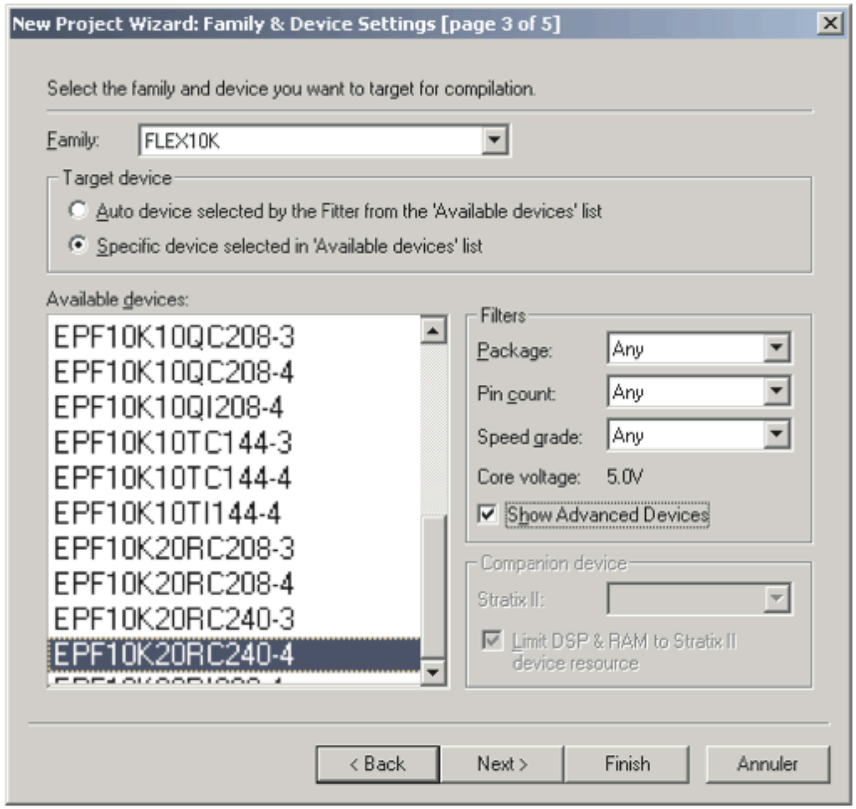


- Spécifications du projet

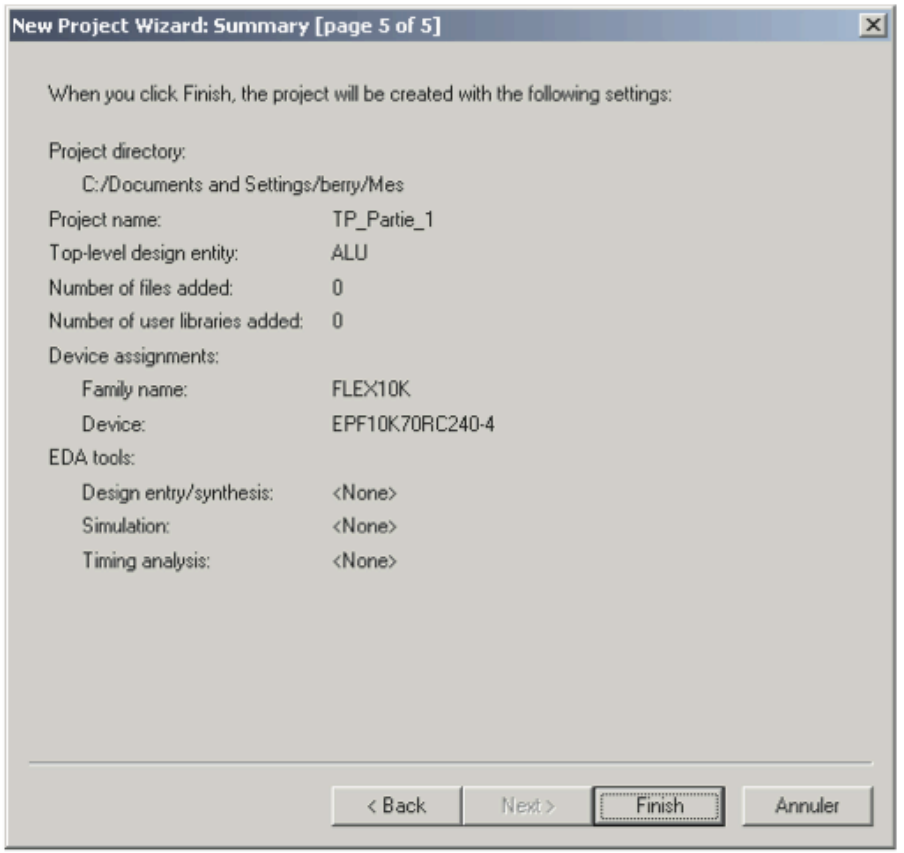


Quartus II

- Spécifications du circuit

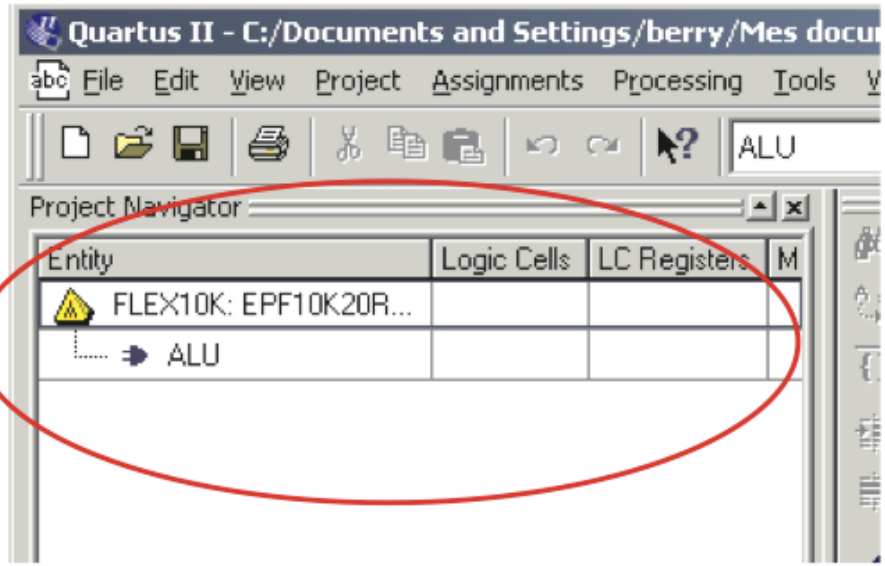


- Résumé du projet

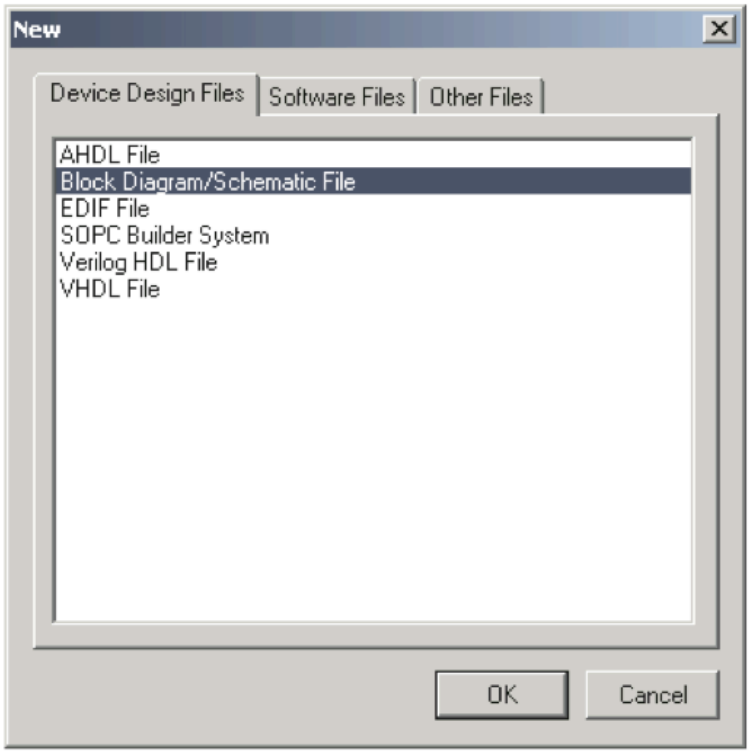


Quartus II

- Projet navigateur

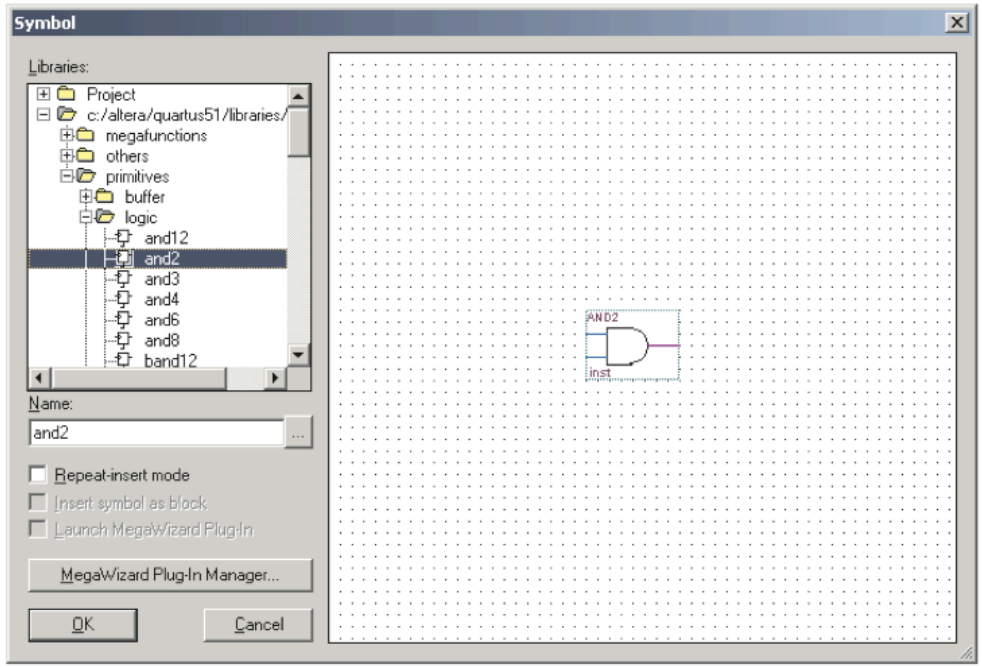


- Ajout de fichiers de descriptions

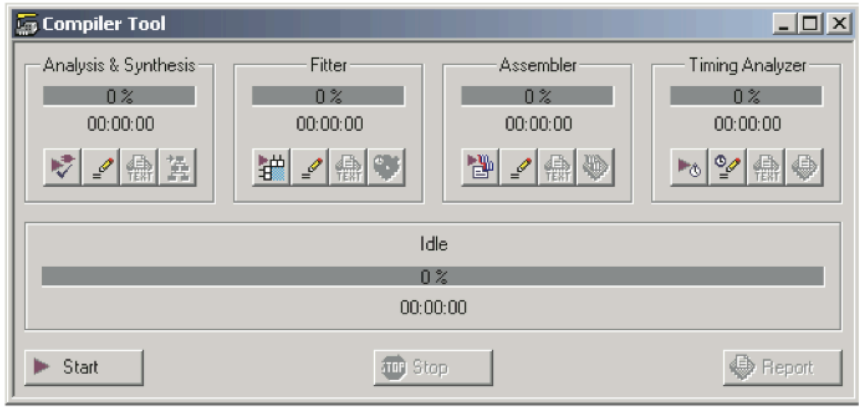


Quartus II

- Exemple de portes de la bibliothèque



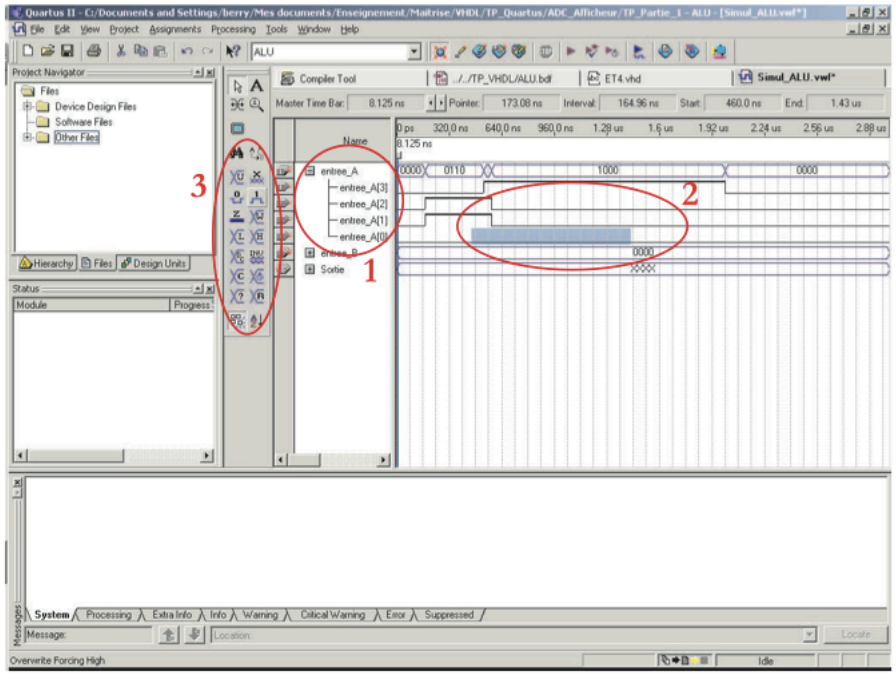
- Outils de compilation

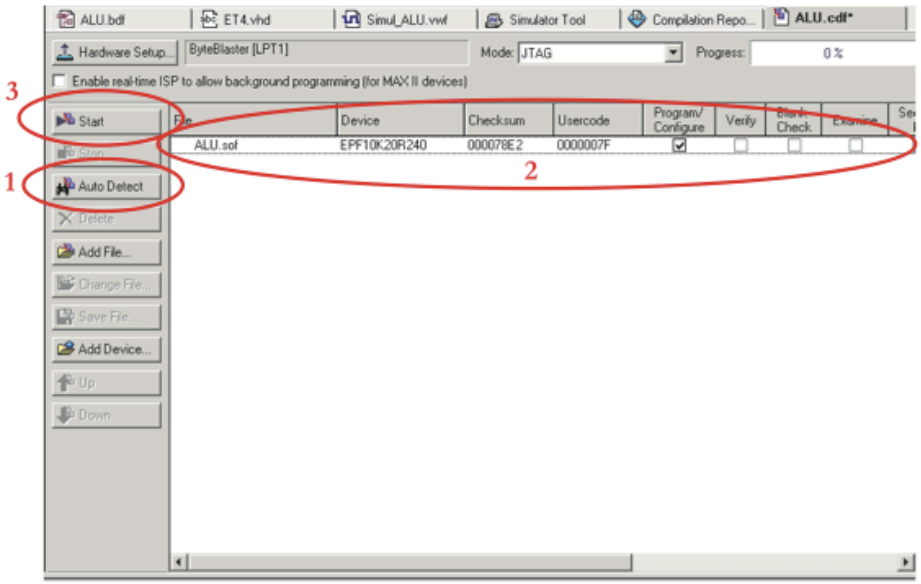


Quartus II

- Simulateur
  1. Entrées/sorties
  2. Chronogrammes
  3. Programmation stimuli

- Programmateur
  1. Détection du circuit à programmer
  2. Résumé des caractéristiques
  3. Démarrage du programmateur



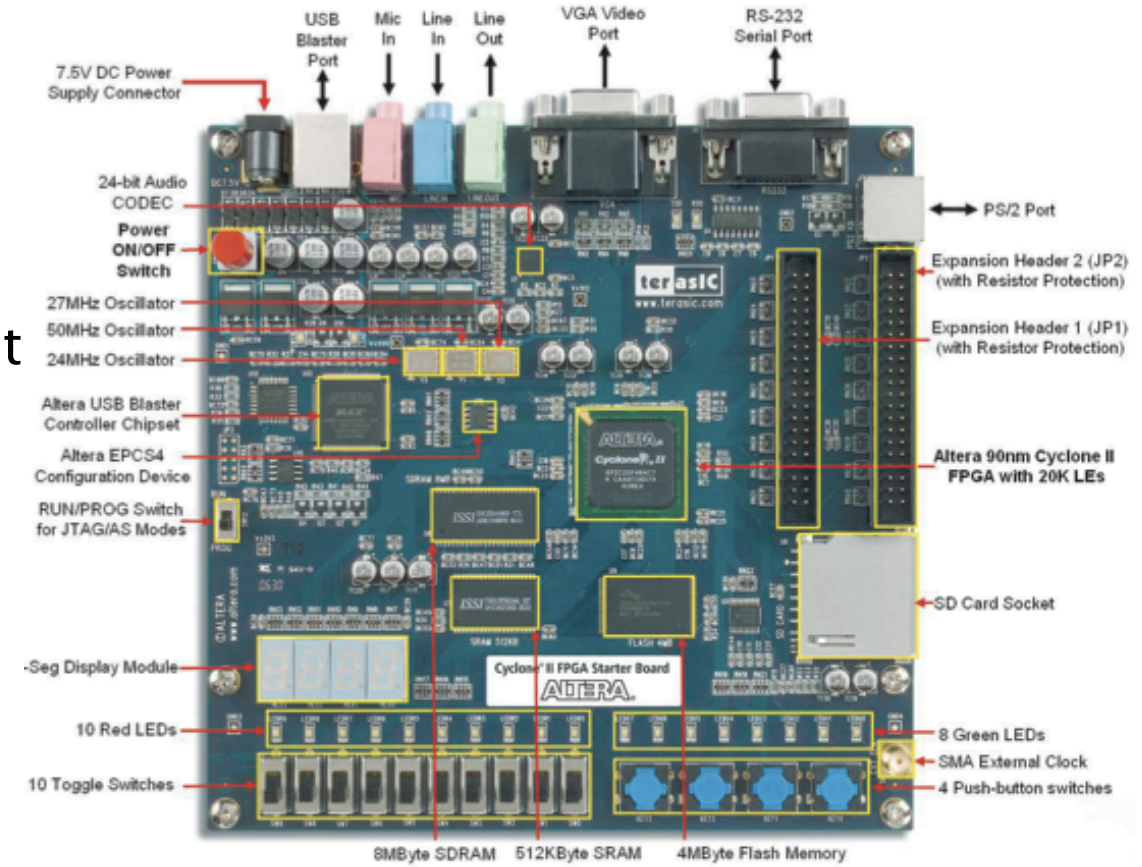


6. Starter Kit Cyclone II

Caractéristiques

- CycloneII EP2C20F484C7N

- Mémoire
  - 8 Mo de SDRAM
  - 512k SRAM
  - 4 Mo de Flash
- 4 afficheurs 7 segment
- 10 boutons
- 4 pousse boutons
- Codec Audio 24 bits
- Codec Vidéo VGA
- 40 I/O programmable
- PS/2



Conclusion

- Circuits programmables caractérisés
  - Architectures
  - Flot de conception
    - Outils de description et de programmation
- Plate-forme Starter Board d'ALTERA
  - Utilisée pour les TDs et les TPs
- Comment décrire le comportement d'un système électronique en vue de l'implémenter sur un circuit programmable ?
  - Langage de description matériel (chapitre 2)