Les Circuits Logiques Programmables FPGA

Ce document est réalisé dans le cadre d'un cours d'introduction à la conception numériques avancées en utilisant les circuits FPGA, pour les étudiants de Master en Télécommunication.

Sommaire

Chapitre 1. Les Réseaux Logiques Programmables : PLD (3 Semaines)

- Introduction
- Structure des réseaux logiques combinatoires
- Classification des réseaux logiques combinatoires

Chapitre 2. Les technologies des éléments programmables (3 Semaines)

Chapitre 3. Architecture des FPGA (3 Semaines)

- Présentation des CP (Circuits programmables type PLA, CPLD)
- Structure des FPGA & ASICs
- Architecture générale
- Blocs logiques programmables
- Terminologies
- Blocs de mémoire intégrée
- Exemples de constructeurs Altera et Xilinx
- Applications

Chapitre 4. Programmation VHDL (3 Semaines)

- Introduction
- Outils de programmation : Altera Quartus II, Modelsim, Xilinx ISE
- Structure d'un programme
- Structure d'une description VHDL simple
- Entité
- Les différentes descriptions d'une architecture (de type flot de données, comportemental ou procédural, structurel et architecture de test)
- Process
- Les structures de contrôle en VHDL
- Instructions séquentielles et concurrentes
- Les paquetages et les bibliothèques

Chapitre 5. Applications : Implémentation de quelques circuits logiques dans les circuits FPGA (3 Semaines)

- Multiplexeur
- Compteur
- Comparateur
- Registre à décalage
- Filtre simple

Mode d'évaluation :

Contrôle continu: 40% ; Examen: 60%.

Références bibliographiques :

1. Volnei A. Pedroni, "Circuit Design with VHDL", MIT press, 2004
2. Jacques Weber , Sébastien Moutault, Maurice Meaudre, "Le langage VHDL : du langage au circuit, du circuit au langage", DUNOD, 2007
3. Christian Tavernier, "Circuits logiques programmables", DUNOD 1992

Chapitre I Les Réseaux Logiques Programmables : PLD

I.1 Introduction

Avant les circuits logiques programmables

Tout circuit numérique était construit à partir de composants logiques standards dont les fonctions étaient bien définies par le constructeur dans les data sheets. La complexité et le nombre de circuits standards nécessaires à une application rendaient la mise au point d'une carte électronique longue et coûteuse et la moindre erreur de conception entraînait une refonte du circuit imprimé.

Remplacer les composants discrets par un circuit programmable

Vers la fin des années 70 apparaissent des composants logiques programmables qui permettent de remplacer plusieurs composants standards et ceci dans le but de diminuer le coût de production, d'augmenter l'intégration et de protéger le design. Malgré le degré d'intégration assez faible (quelques centaines de portes) de ces premiers composants, ils sont bien accueillis en raison de l'utilisation de composants homogènes et de leur souplesse d'utilisation. Au milieu des années 80 viennent alors les FPGAs (Field Programmable Gate Array), nouvelle technologie de composants programmables alliant souplesse d'utilisation (car programmables par l'utilisateur) et très grande densité d'intégration (quelques centaines de milliers de portes logiques).

Les circuits ASICs (Application Specific Integrated Circuits)

Dans la littérature, le terme ASIC est employé pour décrire l'ensemble des circuits spécifiques à une application. Or, dans le langage courant des concepteurs, le terme ASIC est presque toujours utilisé pour décrire les circuits réalisés chez un fondeur. On désigne, par le terme générique PLD (Programmable logic Device), l'ensemble des circuits programmables par l'utilisateur.

On peut distinguer deux familles d'ASICs :

- Les circuits spécifiques : ce sont des produits non préfabriqués chez le fondeur. Ce sont des composants coûteux que seules la très haute densité (millions de portes) et un très grand volume de production peuvent justifier.
- Réseaux pré-diffusés Gate Arrays : pas programmable par l'utilisateur mais le dernier masque chez le fondeur est sur mesure.
- Circuits pré-caractérisés Cell-Based ASIC : le fondeur propose une bibliothèque d'éléments que l'on peut assembler pour former un circuit spécifique. Tous les masques sont alors sur mesure.

- Circuits Full Custom ASIC : les dopages des différentes zones, la largeur des caissons, la géométrie des transistors, … Tout est décidé par le concepteur. Le fondeur n'est plus alors qu'un exécuteur.
- Circuits semi-spécifiques : réseaux programmables par l'utilisateur : ce sont des produits déjà fabriqués chez le fondeur qui nécessitent une personnalisation (interconnexions) pour réaliser la fonction désirée. Ces circuits peuvent être obtenus à faibles coûts.

Dans ce cours on se propose l'étude des composants semi-spécifiques que sont les réseaux programmables, notamment les FPGAs, et aux méthodes de conception d'une application en utilisant un langage de description hardware incontournable : le VHDL (Very high speed integrated circuits Hardware Description Language).

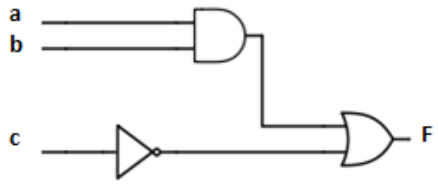
I.2 Structure des réseaux logiques programmables combinatoires

Toute fonction logique peut être écrite sous forme de somme de produit (minterms) ou produit de somme (maxterms). L'écriture suivante donne une fonction logique à trois variables sous forme de somme de produit : F(a,b,c) = a.b+ /c

Une implémentation matérielle directe impliquerait l'utilisation de :

- Une porte logique ET (AND)
- Une porte logique OU (OR)
- Une porte logique inverseuse NON (NOT°)

Le circuit devrait ressembler à ceci :

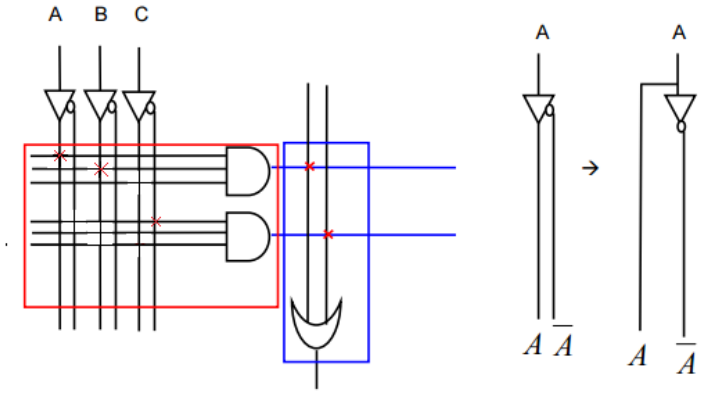


Si nous choisissons d'implémenter un circuit logique en utilisant l'écriture somme de produit, se traduisant par une conception en AND-OR-NOT, on aura besoins de :

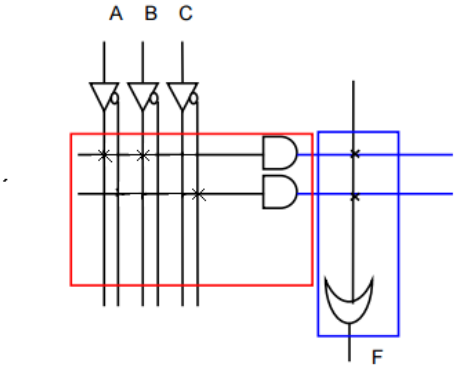
- Un ensemble de portes logiques AND
- Un ensemble de portes logiques OR
- Un ensemble de portes logiques NOT

Afin d'optimiser notre conception, on pourrait organiser l'ensemble des portes AND en un réseau, les portes OR dans un autre.

Le réseau de porte OR devrait être celui qui délivre les sorties, tandis que le réseau des AND se trouve au niveau des entrées. Afin de permettre l'implémentation d'une entrée complémentée on introduit au niveau de chaque entrée une porte inverseuse.

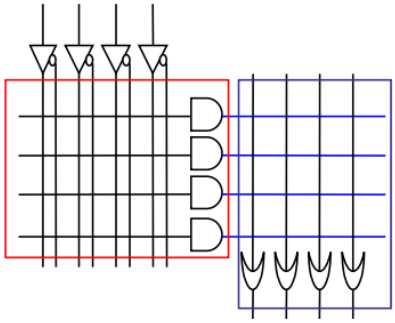


La figure suivante montre une représentation simplifiée du circuit précédent



Dans la figure, le X indique qu'il y a une connexion entre les deux lignes d'interconnexion. Il suffit de réaliser un réseau ayant plus de portes AND et OR, et donc ayant plus d'entrée-sortie. Ceci implique la possibilité d'implanter un nombre plus grand de fonction sur le même circuit. C'est l'architecture de base des premiers dispositifs logiques programmables combinatoires.

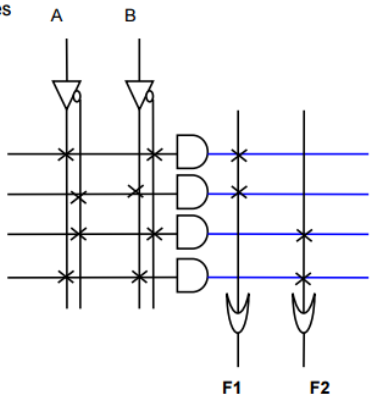
Schéma général d'un réseau logique programmable



Exemple

Réaliser les fonctions suivantes

$$f1 = A.\overline{B} + \overline{A}.B$$
$$f2 = \overline{A}.\overline{B} + A.B$$



En conclusion :

• Un réseau logique programmable (circuit logique programmable) est un circuit qui peut être configurer par l'utilisateur pour avoir une ou plusieurs fonctions logiques.

• Un circuit programmable est constitué d'un ensemble d'opérateurs ET et OU organisés sous forme de deux matrices.

• La matrice des ET est un ensemble de portes AND qui permet de relier les différentes variables d'entrées.

• La matrice des OU est un ensemble de portes OR qui permet de relier les différents termes AND.

• Une matrice peut être programmable (paramétrable) ou figée (préconfigurée).

• La programmation consiste à faire bruler (sauter) les fusibles des termes (ou des variables) qu'on ne veut pas utiliser -> laisser les fusibles utiles.

• La programmation se fait une seule fois : une fois les fusibles brulés on peut pas les réparer.

• La programmation est réalisée grâce à un dispositif spécial.

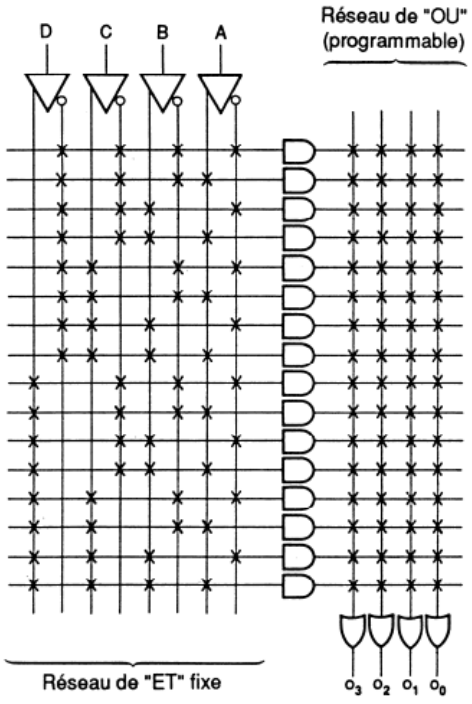
I.3 Classification des réseaux logiques programmables combinatoires

Les circuits logiques programmables peuvent être classifiés selon plusieurs critères. Selon la programmabilité des deux matrices AND et OR on peut distinguer :

- Matrice ET figée et OU programmable -> PROM (Programmable Read-Only Memory)
- Matrice ET programmable et OU figée -> PAL (Programmable Array Logic)
- Matrice ET programmable et OU programmable -> PLA (Programmable Array Logic)

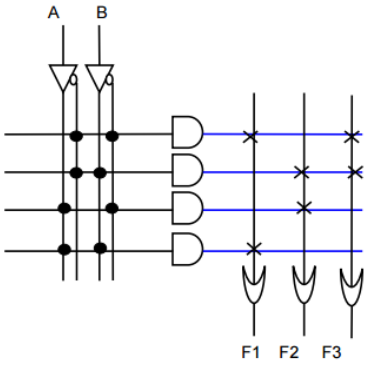
a) PROM

La PROM est une mémoire morte programmable. On entend par mémoire morte qu'il s'agit là de mémoire non-volatile, ne causant aucune perte de données si le circuit n'est plus alimenté.



Les PROM : exemple

$$f1 = \overline{A}.\overline{B} + A.B$$
$$f2 = A.\overline{B} + \overline{A}.B$$
$$f2 = \overline{A}.\overline{B} + \overline{A}.B$$



Les premiers circuits programmables apparus sur le marché sont les PROM bipolaires à fusibles. Cette mémoire est l'association d'un réseau de ET fixes, réalisant le décodage d'adresse, et d'un réseau de OU programmables, réalisant le plan mémoire proprement dit. On peut facilement comprendre que, outre le stockage de données qui est sa fonction première, cette mémoire puisse être utilisée en tant que circuit logique.

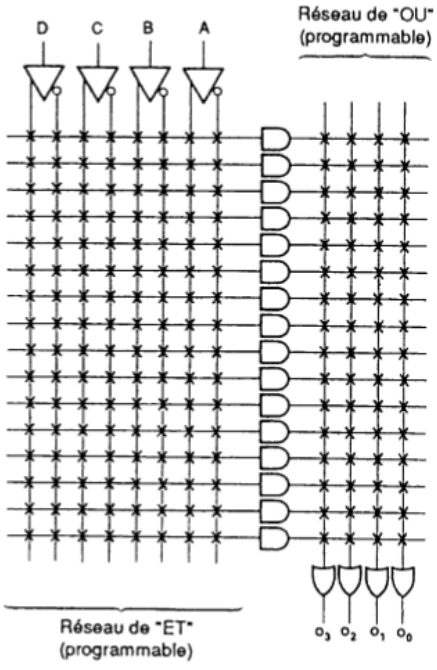
Chaque sortie Oi peut réaliser une fonction OU de 16 termes produits de certaines combinaisons des 4 variables A, B, C et D. Avec les PROM, les fonctions logiques programmées sont spécifiées par les tables de vérités. Il suffit de mettre les variables d'entrées sur les adresses et de récupérer la fonction logique

sur le bit de donnée correspondant. Le temps de propagation est indépendant de la fonction implantée (c'est le temps d'accès de la mémoire).

B. Les PLA

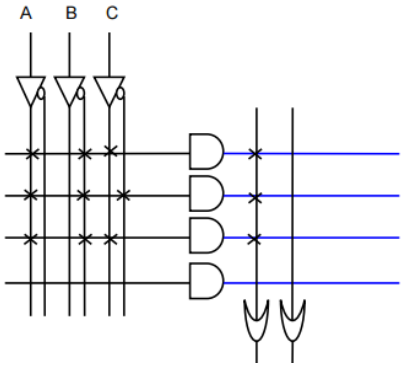
La structure des PLA est une évolution des PROM bipolaires. Elle est constituée d'un réseau de ET programmables et d'un réseau de OU programmables.

Chaque sortie Oi peut réaliser une fonction OU de 16 termes produits des 4 variables A, B, C et D. Avec cette structure, on peut implémenter n'importe quelle fonction logique combinatoire. Ces circuits sont évidemment très souples d'emploi, mais ils sont plus difficiles à utiliser que les PROM. Statistiquement, il s'avère inutile d'avoir autant de possibilité de programmation, d'autant que les fusibles prennent beaucoup de place sur le silicium. Ce type de circuit n'a pas réussi à pénétrer le marché des circuits programmables. La demande s'est plutôt orientée vers les circuits PAL.



Exemple

Réaliser la fonction suivante en utilisant un PLA $f(A,B,C)=ABC+A\bar{B}\bar{C}+A\bar{B}C$



C. Les PAL

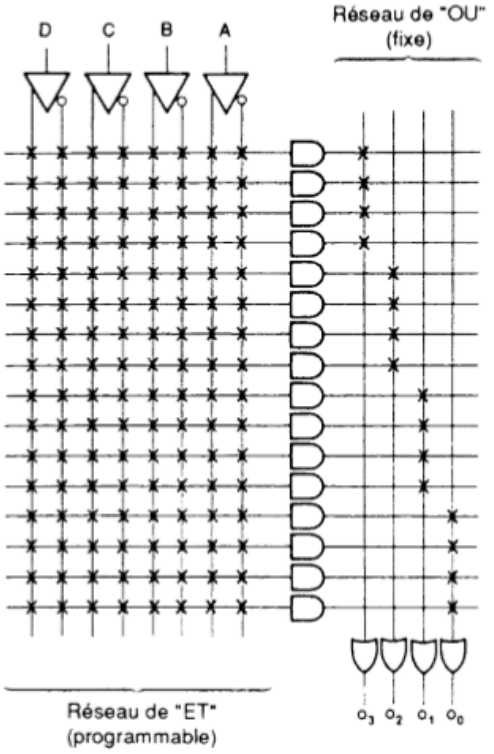
Contrairement aux PLA, les PAL (Programmable Array Logic) imposent un réseau de OU fixes et un réseau de ET programmables. La technologie employée est la même que pour les PLA. La figure qui suit représente la structure logique d'un PAL où chaque sortie intègre 4 termes produits de 4 variables.

L'architecture du PAL a été conçue à partir d'observations indiquant qu'une grande partie des fonctions logiques ne requiert que quelques termes produits par sortie. L'avantage de cette architecture est l'augmentation de la vitesse par rapport aux PLA. En effet, comme le nombre de connexions

programmables est diminué, la longueur des lignes d'interconnexion est réduite. Le temps de propagation entre une entrée et une sortie est par conséquent plus faible.

En revanche, il arrive qu'une fonction logique ne puisse être implantée, car une sortie particulière n'a pas assez de termes produits. Prendre un boîtier plus gros, peut être préjudiciable en termes de prix et de rapidité, le temps de propagation étant proportionnel à la longueur des lignes d'interconnexion du réseau de ET et donc au nombre d'entrées. Pour remédier à cette limitation, il a fallu modifier les entrées/sorties du circuit. Le PAL possède toujours des entrées simples sur le réseau de ET programmables, mais aussi des broches spéciales (voir figure ci-dessous) qui peuvent être programmées :

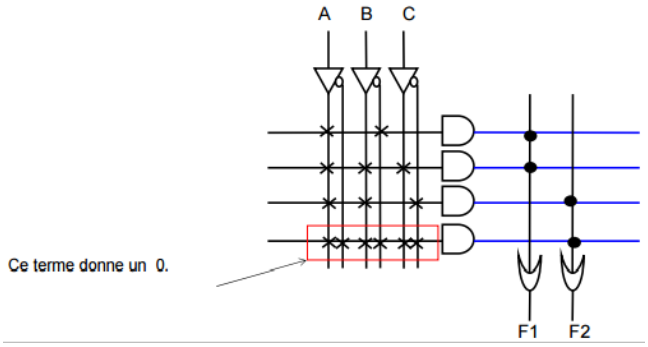
- en entrée simple en faisant passer le buffer de sortie trois états en haute impédance,
- en sortie réinjectée sur le réseau de ET. Cela permet d'augmenter le nombre de termes produits disponibles sur les autres sorties.

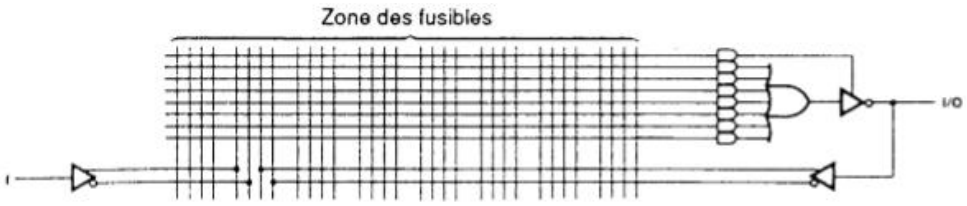


Les PAL : exemple

$$f1(a,b,c) = a.\overline{b} + a.b.c$$

$$f2(a,b,c) = a.b.\overline{c} + 0$$





Chapitre II : Les technologies des éléments programmables

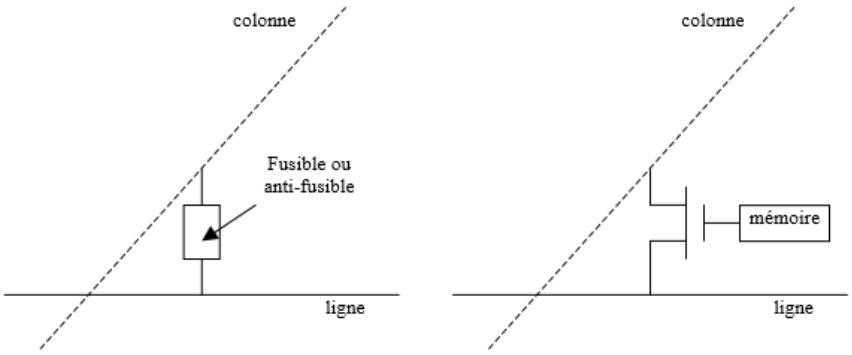
Les cellules standards implantées dans les circuits "Semi-custom" vont de la simple porte jusqu'à une structure complexe utilisant un grand nombre de transistors. Il existe plusieurs méthodes servant à interconnecter ces cellules :

- par masque (fondeur),
- par fusible,
- par anti-fusible,
- par cellule mémoire : EPROM, EEPROM, flash EPROM et SRAM.

Dans la méthode dite « interconnexion par masque », le fondeur réalise les interconnexions par métallisation en créant les derniers masques de fabrication (2 masques par couches de métallisation). Cette méthode n'est utilisée que pour les circuits prédiffusés.

Les autres méthodes sont utilisées dans les PLD. Dans ces circuits, les fils de liaison existent déjà (organisée en lignes et en colonnes), mais ils ne sont reliés ni entre eux, ni avec les éléments logiques du circuit. Il faut donc arriver à créer une interconnexion entre deux fils.

Deux possibilités existent : les interconnexions directes ou les interconnexions par cellule mémoire.



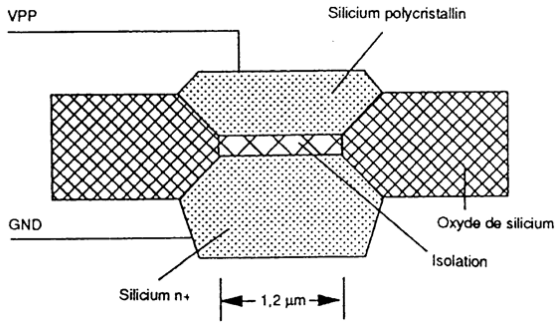
a) Interconnexion directe

a).1 Interconnexion par fusible

C'est la technique des PROM bipolaires à fusibles (Programmable Read Only Memory). On insère, entre chaque intersection, une diode en série avec un fusible. Pour supprimer la connexion entre deux lignes, il suffit d'appliquer une tension élevée pour claquer le fusible. Le boîtier n'est donc programmable qu'une seule fois par l'utilisateur. Cette méthode n'est plus utilisée aujourd'hui.

a).2 Interconnexion par anti-fusible

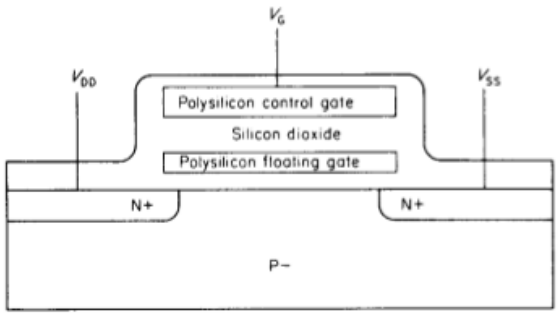
Avec cette technique, c'est l'opération inverse qui est réalisée. On ne coupe pas une liaison, mais on l'établit. L'anti-fusible isole deux lignes métalliques placées sur deux niveaux différents grâce à une fine couche d'oxyde de silicium. Si on applique une impulsion élevée (≈21V) calibrée en temps (moins de 5 ms), la couche d'oxyde est trouée et les deux lignes se retrouvent en contact. La résistance entre les deux lignes passe alors de 100 MΩ à 100Ω. Comme pour la technique du fusible, le boîtier n'est programmable qu'une seule fois par l'utilisateur. Cette méthode est peu utilisée (à part par ACTEL).



b) Interconnexion par cellule mémoire

b).1 La cellule EPROM

Chaque cellule EPROM (Erasable Programmable Read Only Memory) est constituée d'un transistor FAMOS (Floating gate Avalanche injection MOS, Intel 1971) qui est programmable électriquement et effaçable aux rayons ultraviolets. La figure suivante montre que le transistor FAMOS possède deux grilles.



La grille supérieure est utilisée pour la sélection et la grille inférieure entre la grille de sélection et le substrat est dite flottante car elle n'est reliée à rien. Elle est entièrement isolée par l'oxyde de silicium (SiO2). Par application d'une tension positive élevée sur la grille de sélection, on communique aux électrons dans le canal une énergie suffisante qui leur permet de passer au travers de cet isolant. Ces charges s'accumulent sur la grille isolée où elles se trouvent piégées. La cellule mémoire est alors programmée. Pour l'effacement, on expose la puce aux rayons ultra-violets. Les photons communiquent leur énergie aux électrons et leur font franchir le diélectrique en sens inverse. La grille flottante du transistor perd alors sa charge et la cellule redevient vierge. Pour cette technique, les boîtiers doivent posséder une fenêtre en quartz pour laisser passer les U.V. Il existe une variante de cette technologie qui

n'est programmable qu'une seule fois par l'utilisateur : l'OTP (One Time Programming). Pour des raisons de coûts du boîtier, la fenêtre en quartz servant à laisser passer les UV est supprimée. La technologie EPROM n'est plus utilisée aujourd'hui.

b).2 La cellule EEPROM

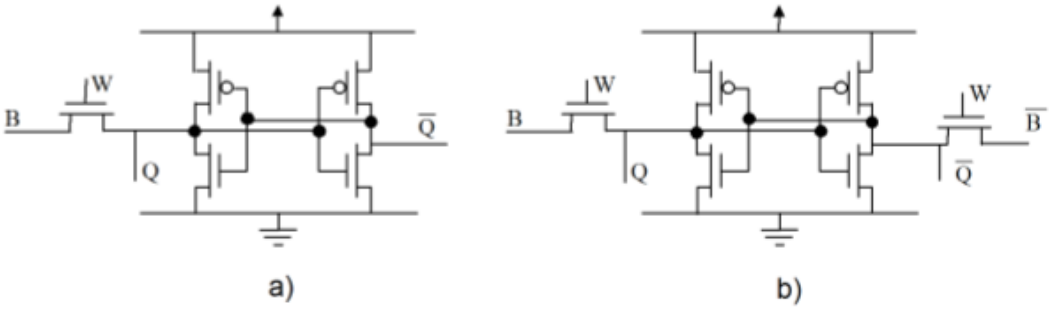
La cellule EEPROM (Electrically Erasable Programmable Read Only Memory) est similaire à la cellule EPROM, mais une deuxième grille recouvre la première grille flottante. La phase de programmation reste identique. En revanche, le boîtier est effacé électriquement en appliquant une tension suffisante sur la deuxième grille. Les électrons piégés dans la première grille sont déchargés par effet tunnel. Les dimensions des cellules EEPROM étant beaucoup plus élevées que celles des cellules EPROM, cette méthode n'est plus utilisée aujourd'hui.

b).3 La cellule flash

Comme pour la cellule EEPROM, la cellule flash se programme électriquement par injection d'électrons et s'efface électriquement par effet tunnel. En revanche, la dimension de la cellule est beaucoup plus réduite. C'est une technologie de mémoire morte reprogrammable relativement récente (1985) qui connaît un fort développement. Cette solution non-volatile serait idéale si les PLD basés sur de la mémoire Flash n'avaient pas deux générations de retard sur les PLD basés sur de la mémoire SRAM (pour des raisons de procédé de fabrication).

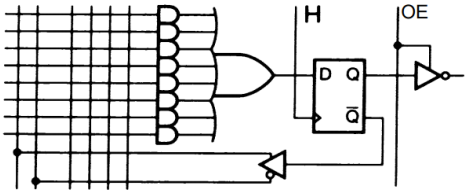
b).4 La cellule SRAM

La cellule SRAM (Static Random Access Memory) consiste en deux inverseurs CMOS connectés en boucle pour former un bistable. L'état de cette cellule peut être modifié par un signal électrique externe (ligne B). La cellule RAM est une structure de stockage volatile. La figure suivante représente la cellule d'une SRAM à 5 transistors (a) et une cellule à 6 transistors (b). Malgré son coût, C'est la méthode utilisée dans les FPGA les plus performants à ce jour.

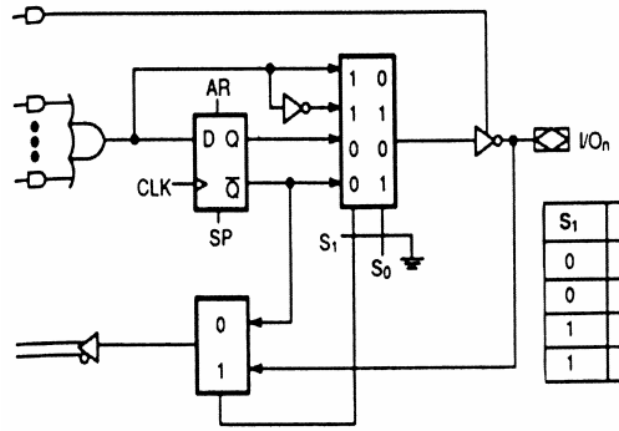


Chapitre III : Architecture des FPGA

Les structures présentées jusqu'à maintenant ne font intervenir que de la logique combinatoire. Les architectures des PAL ont évolué vers les PAL à registres. Dans ces PAL, la sortie du réseau de fusibles aboutit sur l'entrée d'une bascule D. La sortie Q peut aller vers une sortie, la sortie Qb étant réinjectée sur le réseau via un inverseur/non inverseur.



Le **PAL versatile** (polyvalent), dont le membre le plus connu est le 22V10, présente une évolution des PAL vers les circuits logiques programmables de plus haut niveau. En effet, ils continuent de respecter le principe de fonctionnement énoncé précédemment, mais ils utilisent une structure de cellule de sortie qui s'apparente à un EPLD. D'après la figure suivante, on remarque que la cellule de sortie dispose d'une bascule D pré-positionnable associée à deux multiplexeurs programmables. Les connexions S0 et S1 sont réalisées grâce à des fusibles internes.



S_1	S_0	Configuration des sorties
0	0	Registre, actif bas
0	1	Registre, actif haut
1	0	Combinatoire, actif bas
1	1	Combinatoire, actif haut

Cette sortie peut adopter plusieurs configurations (d'où le terme polyvalent), le 22V10 pouvant donc être utilisé à la place de tous les PAL bipolaires classiques :

- Sortie combinatoire active au niveau bas,
- Sortie combinatoire active au niveau haut,
- Sortie registre active au niveau bas,
- Sortie registre active au niveau haut.

Les premiers PAL pouvaient être assez facilement programmés à la main. Toutefois, la réalisation de fonctions complexes est devenue rapidement inextricable. Des logiciels de développement sont donc apparus afin de faciliter ce travail. Il en existe de nombreux, les plus connus étant PALASM (société AMD) et ABEL (société DataIO). Au-delà d'un certain niveau de complexité, l'utilisation de leur simulateur intégré permet une mise au point rapide de la fonction à réaliser.

Tous les PAL disposent d'un fusible ou bit de sécurité. Ce fusible, une fois claqué, interdit la relecture d'un composant déjà programmé. En effet, il arrive que des entreprises indélicates soient tentées de copier les PAL développés par leurs concurrents.

Un des inconvénients des circuits bipolaires à fusibles, est qu'ils ne peuvent pas être testés à la sortie de l'usine. Pour tester leur fonctionnement, il faudrait en effet claquer les fusibles, ce qui interdirait toute programmation ultérieure. A l'origine, les premiers PAL étaient bipolaires puisqu'ils utilisaient la même technologie que les PROM bipolaires à fusibles. Il existe maintenant des PAL en technologie CMOS (appelés GAL (Generic Array Logic) par certains fabricants), programmables et effaçables électriquement, utilisant la même technologie que les mémoires EEPROM. Comme ils sont en technologie CMOS, ils consomment beaucoup moins, en statique, que les PAL bipolaires de complexité équivalente

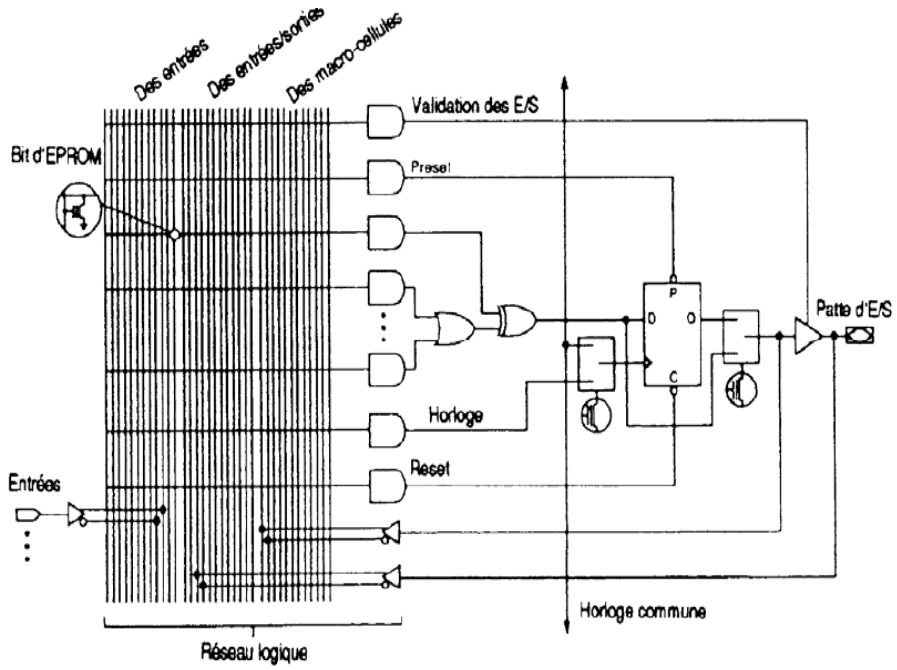
Les EPLD

Les EPLD (Erasable Programmable logic Device) sont des circuits programmables électriquement et effaçables, soit par exposition aux UV pour les plus anciens, soit électriquement. Ces circuits, développés en premier par la firme ALTERA, sont arrivés sur le marché en 1985. Les EPLD sont une évolution importante des PAL CMOS. Ils sont basés sur le même principe pour la réalisation des fonctions logiques de base. Les procédés physiques d'intégration permis par les EPLD sont nettement plus importants que ceux autorisés par les PAL CMOS. En effet, les plus gros EPLD actuellement commercialisés intègrent jusqu'à 24000 portes logiques dont 12000 sont réellement accessibles à l'utilisateur. On peut ainsi loger dans un seul boîtier, l'équivalent d'un schéma logique utilisant jusqu'à 50 à 100 PAL classiques.

Comme les PAL CMOS, les EPLD font appel à la notion de macro-cellule qui permet, par programmation, de réaliser de nombreuses fonctions logiques combinatoires ou séquentielles. Le schéma type de la macro-cellule de base d'un EPLD est présenté ci-dessous. On remarque que le réseau logique est composé de 3 sous-ensembles :

- Le réseau des signaux d'entrées provenant des broches d'entrées du circuit,
- Le réseau des signaux des broches d'entrées/sorties du circuit,

- Le réseau des signaux provenant des autres macro-cellules



Outre la logique combinatoire, la macro-cellule possède une bascule configurable (bascule D, T, RS ou JK). Cette bascule peut être désactivée par programmation d'un multiplexeur. Le signal d'horloge peut être commun à toutes les macro-cellules, ou bien provenir d'une autre macro-cellule via le réseau logique.

Quel que soit la famille d'EPLD, la fonctionnalité de la macro-cellule ne change guère. En revanche, plus la taille des circuits augmentent, plus les possibilités d'interconnexions et le nombre de macro-cellules augmentent.

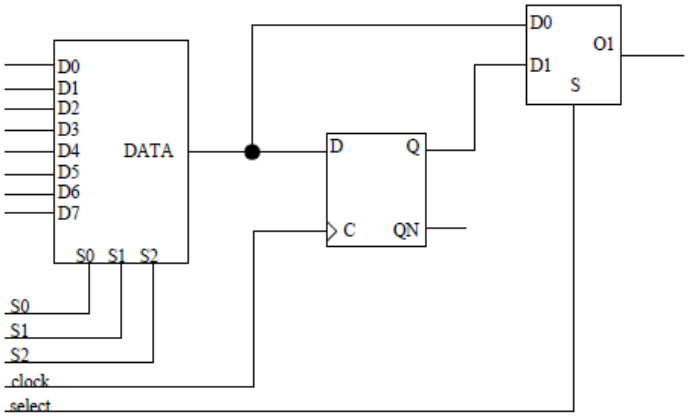
Il existe plusieurs types d'EPLD en technologie CMOS :

- Les circuits programmables électriquement et non effaçables. Ce sont les EPLD de type OTP (One Time Programmable).
- Les circuits programmables électriquement et effaçables aux UV.
- Les circuits programmables électriquement et effaçables électriquement dans un programmateur.
- Les circuits programmables électriquement et effaçables électriquement sur la carte (ISP : In Situ Programmable), utilisant une tension unique de 5 V.

Les plus rapides des EPLD ont des temps de propagation (entrée vers sortie sans registre) de l'ordre de 12 ns. En revanche, comme ils sont en technologie CMOS, leur consommation croît avec l'augmentation de la fréquence de fonctionnement. Le taux d'utilisation des ressources d'un EPLD dépasse rarement 80 %. Avec les EPLD, il est possible de prédire la fréquence de travail maximale d'une fonction logique, avant son implémentation. On rencontre parfois le terme CPLD (Complex Programmable Logic Device). Ce terme est généralement utilisé pour désigner des EPLD ayant un fort taux d'intégration.

Les FPGA

Lancé sur le marché en 1984 par la firme XILINX, le FPGA (Field Programmable Logic Device) est un circuit prédiffusé programmable. Le concept du FPGA est basé sur l'utilisation d'un multiplexeur comme élément combinatoire de la cellule de base. La figure suivante représente la cellule type de base d'un FPGA. Elle comprend un multiplexeur 8 vers 1 permettant de réaliser n'importe quelle fonction logique combinatoire de 4 variables (appelé LUT : Look Up Table ou encore générateur de fonction). La bascule D permet la réalisation de fonctions logiques séquentielles. La configuration du multiplexeur 2 vers 1 de sortie autorise la sélection des deux types de fonction.



Les cellules de base d'un FPGA sont disposées en rangées et en colonnes. Des lignes d'interconnexions programmables traversent le circuit, horizontalement et verticalement, entre les diverses cellules. Ces lignes d'interconnexions permettent de relier les cellules entre elles, et avec les plots d'entrées/sorties. Les connexions programmables sur ces lignes sont réalisées par des transistors MOS dont l'état est contrôlé par des cellules mémoires SRAM. Ainsi, toute la configuration d'un FPGA est contenue dans des cellules SRAM.

Contrairement aux EPLD, on ne peut pas prédire la fréquence de travail maximale d'une fonction logique, avant son implémentation. En effet, cela dépend fortement du résultat de l'étape de placement-routage.

Tous les FPGA sont fabriqués en technologie CMOS, les plus gros d'entre eux intègrent jusqu'à 1000000 portes logiques utilisables. Il faut noter que la surface de silicium d'un FPGA est utilisée au 2/3 pour les interconnexions et au 1/3 pour les fonctions logiques. Le taux d'utilisation global des ressources ne dépasse pas 80 %.

Par rapport aux prédiffusés classiques, les interconnexions programmables introduisent des délais plus grands que la métallisation. Par contre, les cellules logiques fonctionnent à la même vitesse. Pour minimiser les délais de propagation dans un FPGA, il faut donc réduire le nombre de cellules logiques utilisées pour réaliser une fonction. Par conséquent, les cellules logiques d'un FPGA sont plus complexes que celles d'un prédiffusé.

Les FPGA à anti-fusibles

Commercialisés à partir de 1990, ce FPGA, programmable une seule fois, est basé sur la technologie des interconnexions à anti-fusibles. Sa structure s'apparente à celle d'un prédiffusé mer-de-portes, c'est-à-dire qu'il dispose de cellules élémentaires organisées en rangées et en colonnes. Les lignes d'interconnexions programmables traversent le circuit, horizontalement et verticalement, entre les diverses cellules. La technologie à anti-fusibles permet de réduire considérablement la surface prise par les interconnexions programmables, par

rapport aux interconnexions à base de SRAM. La cellule élémentaire diffère d'un fabricant à un autre, mais elle est généralement composée de quelques portes logiques. Le nombre de ces cellules est généralement très important. Alors que le FPGA SRAM est utilisé pour des prototypes ou des petites séries, le FPGA à anti-fusibles est destiné pour des plus grandes séries, en raison de son coût de fabrication moins élevé. Il est généralement conçu avec des outils de synthèse de type VHDL.

Comparaison entre les FPGA et les autres circuits spécifiques

La comparaison et donc le choix entre les différentes technologies est une étape délicate car elle conditionne la conception mais aussi toute l'évolution du produit à concevoir. De plus, elle détermine le coût de la réalisation et donc la rentabilité économique du produit. Généralement, les quantités à produire imposent leurs conditions de rentabilité, dans le domaine du grand public par exemple. Par contre, dans le matériel professionnel, toutes les options sont ouvertes. Il faut établir un rapport coût / souplesse d'utilisation le plus souvent avec des données partielles (pour les quantités à produire par exemple). Nous allons nous contenter dans ce paragraphe de comparer ce qui est comparable (PLD / ASIC, EPLD / FPGA) et de donner une méthode de calcul des coûts des familles ASIC et PLD.

Comparaison entre les PLD et les ASIC.

Un premier choix doit être fait entre les ASIC et les PLD. Les avantages des PLD par rapport aux ASIC sont les suivants :

- Ils sont entièrement programmables par l'utilisateur,
- Ils sont généralement reprogrammables dans l'application, ce qui facilite la mise au point et garantit la possibilité d'évolution,
- Les délais de conception sont réduits, il n'y a pas de passage chez le fondeur.

En revanche, les inconvénients des PLD par rapport aux ASIC sont les suivants :

- Ils sont moins performants en termes de vitesse de fonctionnement (d'un facteur 2 à 3),
- Le taux d'intégration est moins élevé (d'un facteur 10 environ),
- Les ressources d'interconnexion utilisent en général les 2/3 de la surface de silicium.

De plus, le coût de l'ASIC est beaucoup plus faible que le coût du PLD (quoique les choses évoluent très rapidement dans ce domaine, notamment dans la compétition entre FPGA et prédiffusés). Au-delà d'une certaine quantité, l'ASIC est forcément plus rentable que le PLD. Toute la question est donc de savoir quelle est cette quantité ?

6.2 Comparaison entre les FPGA et les EPLD

Si un PLD est choisi, il faut savoir si on doit utiliser un EPLD ou un FPGA. Les avantages des FPGA par rapport aux EPLD sont les suivants :

- Le taux d'utilisation des ressources peut atteindre 80 %, ce qui est meilleur qu'un EPLD,
- Ils consomment moins à fonctionnalité identique (< 10 mA par 1000 portes),
- Les fonctions réalisables sont plus complexes.

Les inconvénients des FPGA par rapport aux EPLD sont les suivants :

- Les EPLD sont plus performants pour certaines fonctions arithmétiques rapides,
- Les fréquences de fonctionnement sont variables suivant la méthode de placement routage retenue. Les EPLD ont des fréquences de travail "prédictibles".

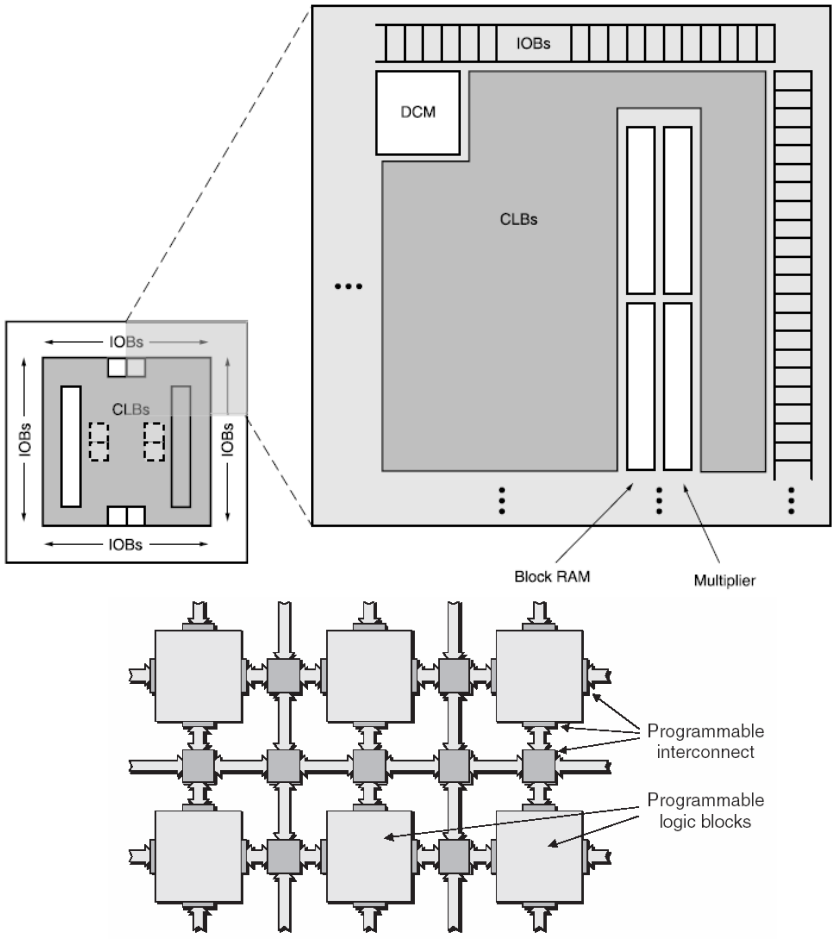
En fait, le domaine d'utilisation des FPGA est celui des prédiffusés, par exemple les fonctions logiques ou arithmétiques complexes ou le traitement du signal. Le domaine d'utilisation des EPLD est plutôt celui des PAL, par exemple les machines d'état complexes. Il est à noter qu'un marché important des PAL et des EPLD est la correction des erreurs de conception dans les ASIC afin d'éviter un aller-retour coûteux chez le fondeur.

2. Un exemple de FPGA : la famille Spartan-3

Ce paragraphe détaille la structure interne des FPGA de la famille Spartan-3 fabriqués par la société XILINX.

2.1 Caractéristiques générales

La figure suivante représente la structure simplifiée de la famille Spartan-3. On reconnaît là, une structure de type prédiffusé.



La famille Spartan-3 est une famille de FPGA CMOS SRAM faible coût basée sur la famille Virtex-II (FPGA complexité élevée). La liste suivante résume ses caractéristiques :

- Matrice de blocs logiques programmables ou CLB (Configurable Logic Block),
- Blocs d'entrée/sortie programmables ou IOB (Input Ouput Block)

- Réseau de distribution d'horloge avec une faible dispersion via les DCM,
- Des blocs RAM 18 kbits,
- Des multiplieurs 18 bits x 18 bits,
- De nombreuses ressources de routage.
  La suite du cours sera consacrée à L'FPGA Spartan 3^E (XC3S500)

2.2 Blocs d'entrée-sortie (IOB)

2.2.1 Généralités

Des blocs d'entrée-sortie (IOB) configurables sont répartis sur toute la périphérie du boîtier.

Chaque IOB assure l'interface entre une broche d'entrée/sortie du boîtier et la logique interne.

Un bloc d'E/S (IOB) assure une interface programmable, unidirectionnelle ou bidirectionnelle entre les pins du boitier et la logique interne de l'FPGA. La figure suivante représente un schéma simplifié du bloc IOB.

Il existe trois chemins principaux pour le signal : sortie, entrée et trois états. Chaque chemin possède ses propres éléments de stockages qui peuvent fonctionner comme registres ou comme latches.

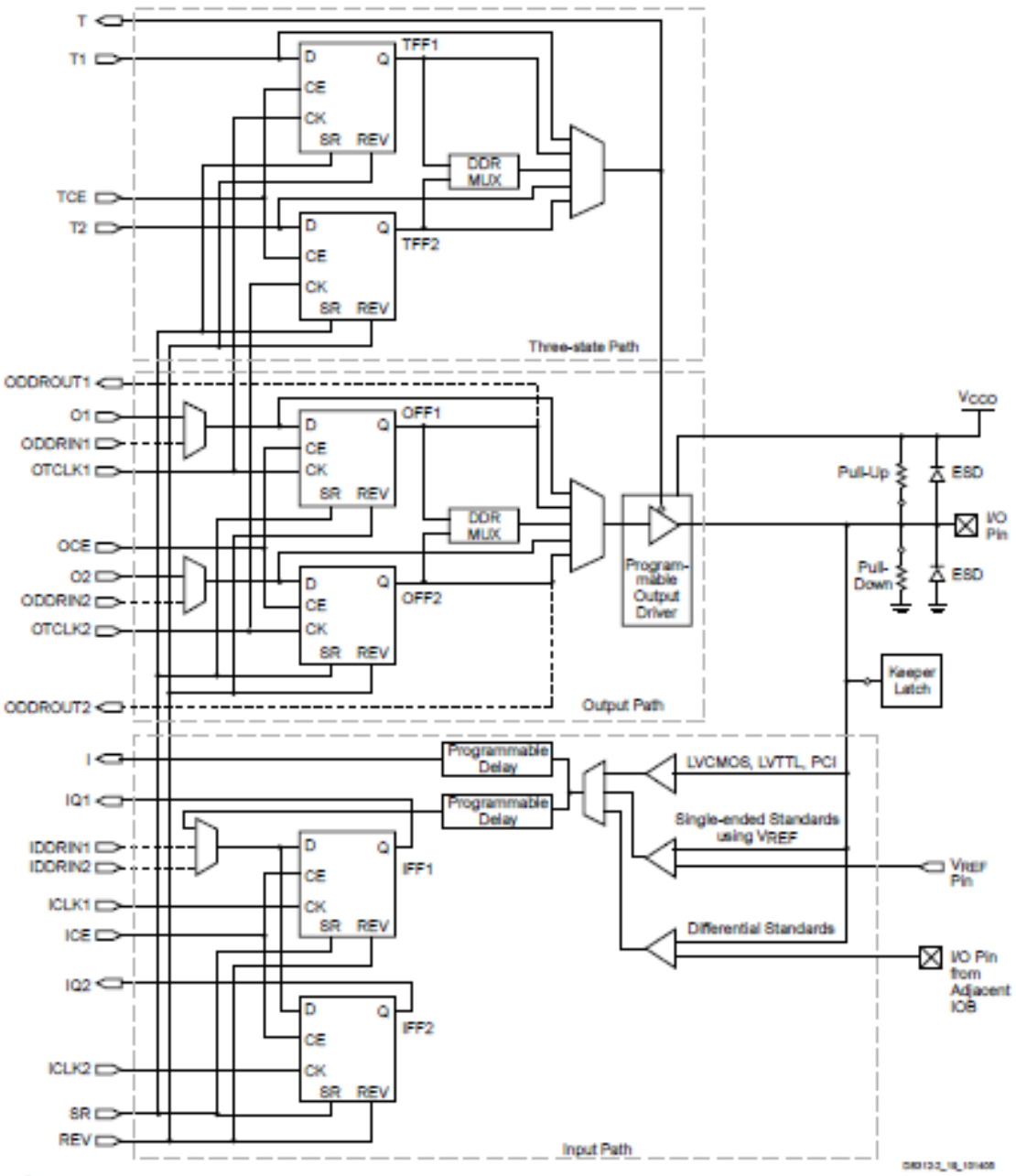
2.2.2 Caractéristiques d'entrée

Le signal sur la broche d'entrée (I/O pin) est amené vers les CLB soit directement via le signal I, soit à travers une paire de bascules D (ou de latch) via IQ1 et IQ2. Les caractéristiques de l'entrée de l'IOB sont les suivantes :

- Diodes de protection ESD,
- Résistance de "pull-up" ou "pull-down",
- Contrôle de l'impédance d'entrée (DCI),
- 23 standards d'entrée (différentiels ou non),
- Horloge indépendante de la sortie,
- Un délai de quelques ns peut être inséré dans le chemin de la donnée d'entrée pour compenser le retard de l'horloge,
- Support du Double Data Rate pour écrire dans les SDRAM DDR.

2.2.3 Caractéristiques de sortie

Le signal de sortie peut être optionnellement inversé à l'intérieur de l'IOB et sortir directement sur la broche ou bien être mis en mémoire par une paire de bascules D actives sur un front. Les caractéristiques de la sortie d'un IOB sont les suivantes :



Notes:

1. All IOB control and output path signals have an inverting polarity option within the IOB.
2. Signals shown with dashed lines connect to the adjacent IOB in a differential pair only, not to the FPGA fabric.

Figure 5: **Simplified IOB Diagram**

- Buffer 3 états piloté par une paire de bascules D,
- Sortie collecteur ouvert,
- 23 standards de sortie (différentiels ou non),
- Contrôle de "slew-rate" (rapide ou lent),
- Contrôle de l'impédance de sortie (DCI),
- Support du Double Data Rate pour lire dans les SDRAM DDR.
- sortance de 24 mA max

2.3 Bloc logique configurable (CLB)

2.3.1 Généralités

Le CLB est l'élément fonctionnel de base de ce FPGA. Sa programmation permet à l'utilisateur de réaliser des fonctions logiques combinatoires ou séquentielles.

- Chaque CLB contient quatre SLICEs, chaque SLICE contient deux (LUTs) pour implémenter de la logique ainsi que 2 éléments dédiés au stockage qui peuvent être utilisés comme bascules ou latches.
- LUT: peut être utilisée comme mémoire 16x1 (RAM16) ou registre à décalage 16-bit (SRL16),
- MUXs et logique de rentenu (carry logic): simplifient les grands circuits et fonctions logiques.
- Une logique à usage générale est automatiquement “mapped” aux ressources des CLB, qui sont identiques

La densité dépend du modèle.

Table 9: **Spartan-3E CLB Resources**

Device	CLB Rows	CLB Columns	CLB Total(1)	Slices	LUTs / Flip-Flops	Equivalent Logic Cells	RAM16 / SRL16	Distributed RAM Bits
XC3S100E	22	16	240	960	1920	2160	960	15360
XC3S250E	34	26	612	2448	4896	5508	2448	39168
XC3S500E	46	34	1164	4656	9312	10476	4656	74496
XC3S1200E	60	46	2168	8672	17344	19512	8672	138752
XC3S1600E	76	58	3688	14752	29504	33192	14752	236032

Notes:

1. The number of CLBs is less than the multiple of the rows and columns because the block RAM/multiplier blocks and the DCMs are embedded in the array (see Figure 1 in Module 1).

La figure ci-dessous représente un schéma simplifié d’un CLB :

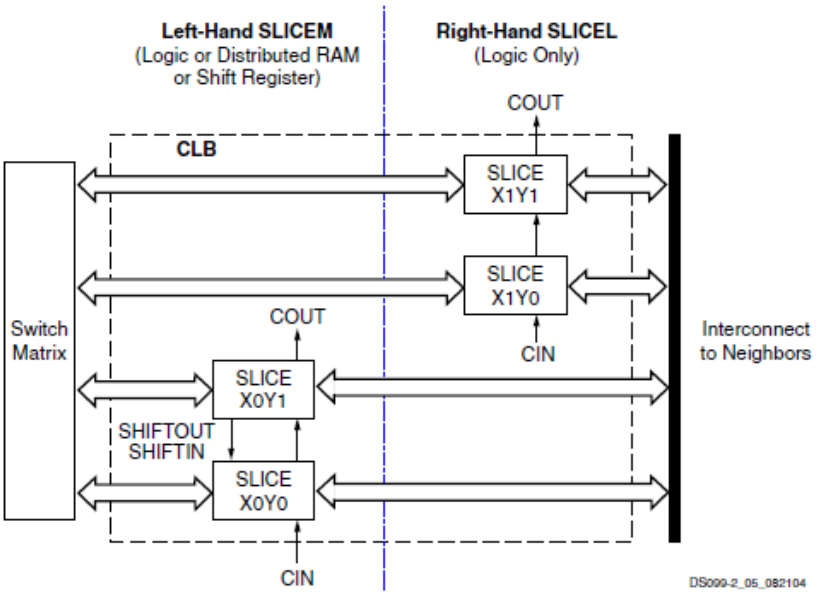


Figure 17: **Arrangement of Slices within the CLB**

A l'intérieur d'un CLB, les 4 SLICEs sont regroupés deux-à-deux : SLICEM (côté gauche) et SLICEL (côté droit).

Le logiciel de développement de Xilinx désigne les locations des SLICEs selon leurs cordonnées en partant du bas gauche : X désigne la colonne, Y désigne la ligne

- SLICEM toujours paire
- SLICEL toujours impaire

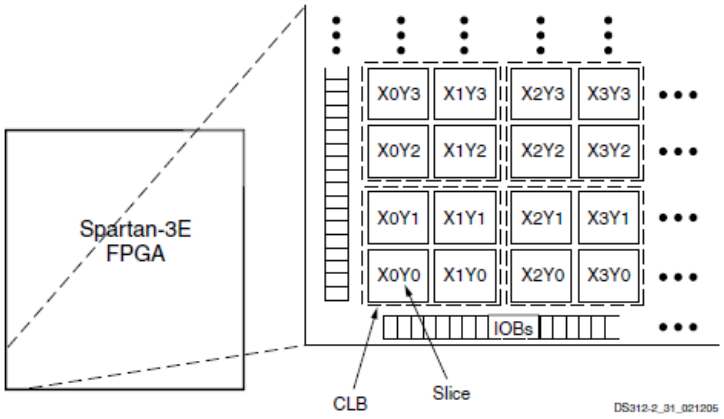


Figure 15: **CLB Locations**

Les SLICEs contiennent quelques éléments en commun et d'autres spécifiques. Chacune contient deux générateurs de fonction LUT et deux éléments de stockage avec une logique additionnelle.

SLICEM et SLICEL contiennent en commun :

- Deux générateurs de fonctions LUT à 4 entrées (F G)
- Deux éléments de stockage
- Deux grands MUXs, (F5MUX FiMUX)

- logique de retenu et arithmétique

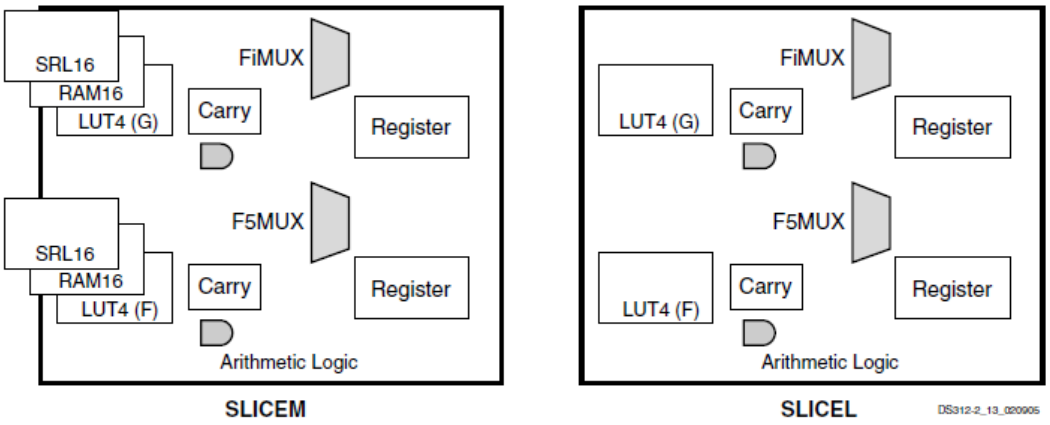


Figure 18: **Resources in a Slice**

La tranche SLICEM possède en plus :

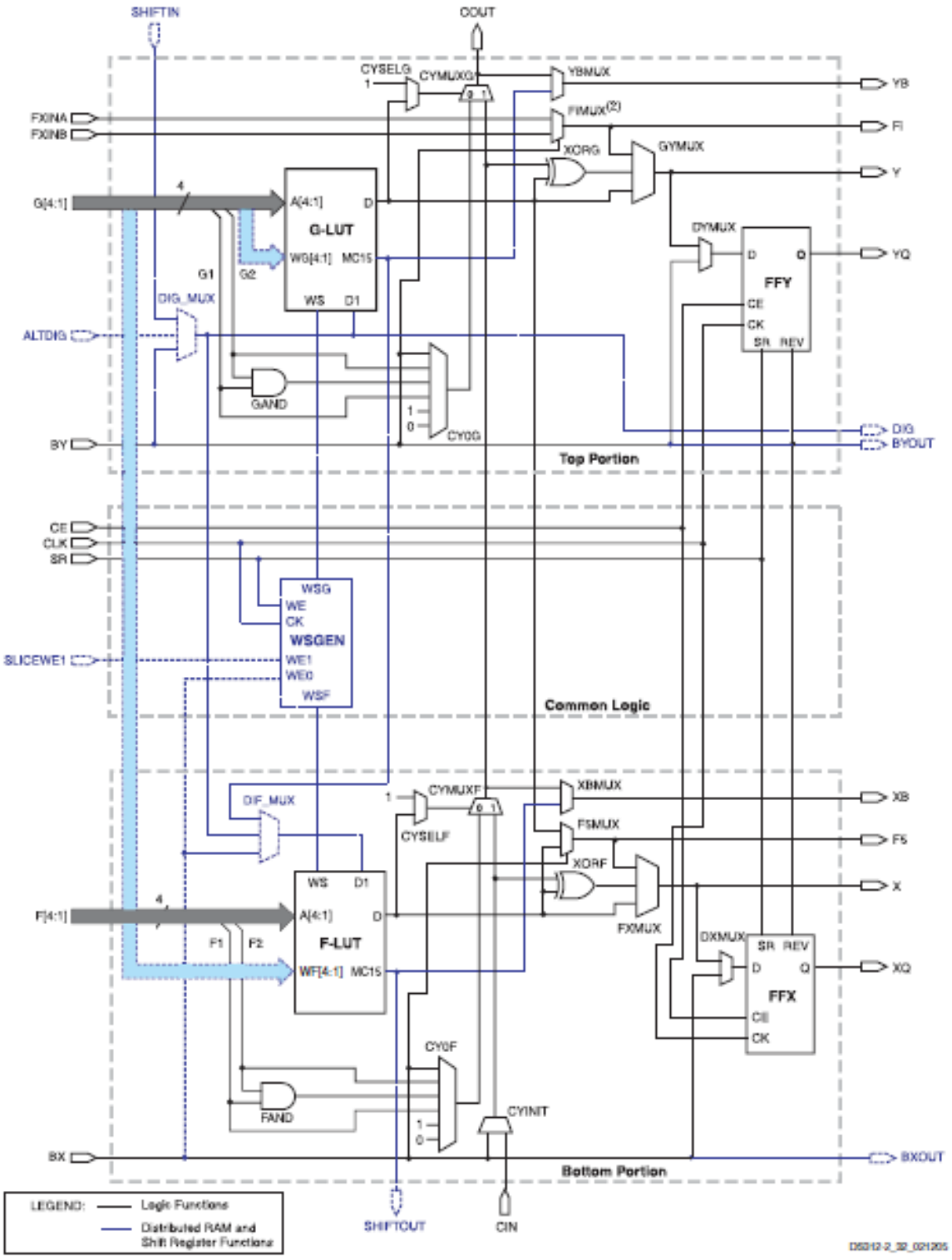
- Deux blocs de RAM distribuée RAM16
- Deux registres à décalage SRL16
- La tranche SLICEM supporte les fonctions logiques et les mémoires

La tranche SLICEL ne supporte que la logique

- => Moitié des LUT supportent logique et mémoire, l'autre moitié logique seulement, les deux types s'alternent à travers les colonnes du réseau

Une SLICEM en détail:

- Les entrées de contrôle d'horloge sont partagées entre les deux moitiés (supérieure et inférieure)
- La chaîne de retenu (carry) entre par CIN et traverse 5 MUX: CYINT, CY0F et CYMUXF dans la portion inférieure et CY0G, CYMUXG dans la portion haute
- La logique arithmétique dédiée contient des portes XOR (XORF et XORG) ainsi que les portes AND (FAND et GAND)


SHIFTIN
COUT
CYSELG
CYMUXG
YBMUX
YB
FXINA
FXINB
FIMUX(2)
FI
XORG
GYMUX
Y
G[4:1]
A[4:1]
D
G-LUT
WG[4:1]
MC15
WS
D1
G1
G2
DYMUX
YQ
FFY
CE
CK
SR
REV
DIG_MUX
ALTDIG
GAND
CY0G
DIG
BYOUT
BY
Top Portion
CE
CLK
SR
WSG
WE
CK
WSGEN
WE1
WE0
WSF
SLICEWE1
Common Logic
DIF_MUX
CYMUXF
XBMUX
XB
CYSELF
FSMUX
F5
XORF
X
F[4:1]
F1
F2
F-LUT
WF[4:1]
FXMUX
DXMUX
FFX
XQ
CY0F
FAND
CYINIT
BX
BXOUT
Bottom Portion
LEGEND:
Logic Functions
Distributed RAM and Shift Register Functions
SHIFTOUT
CIN

Notes:

Le générateur de fonctions LUT

La LUT est un générateur de fonction à base de RAM et est la ressource principale pour implémenter des fonctions logiques.

- Dans une SLICEM, les LUT peuvent être configurées comme RAM distribuée ou comme registre à décalage 16 bits
- Chacune des LUT F et G peut implémenter toute fonction booléenne à 4 variables.
- Pour plus de variables, il est possible de cascader des LUT en utilisant les grand MUX: F5MUX sélectionne entre les LUT dans une tranche, FiM

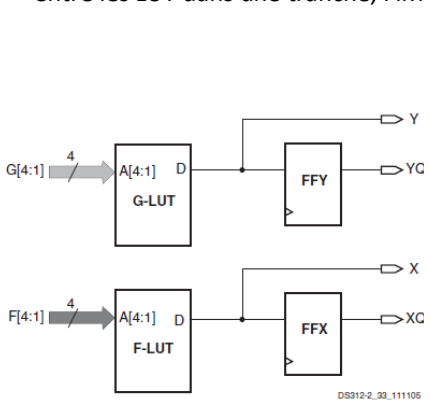


Figure 19: **LUT Resources in a Slice**

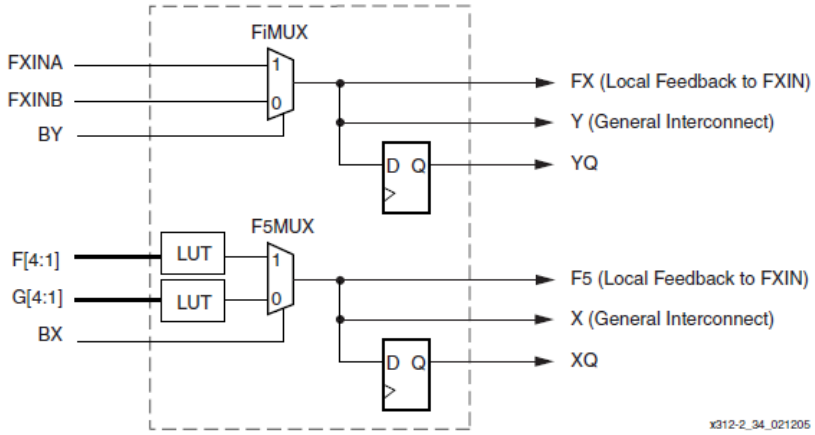
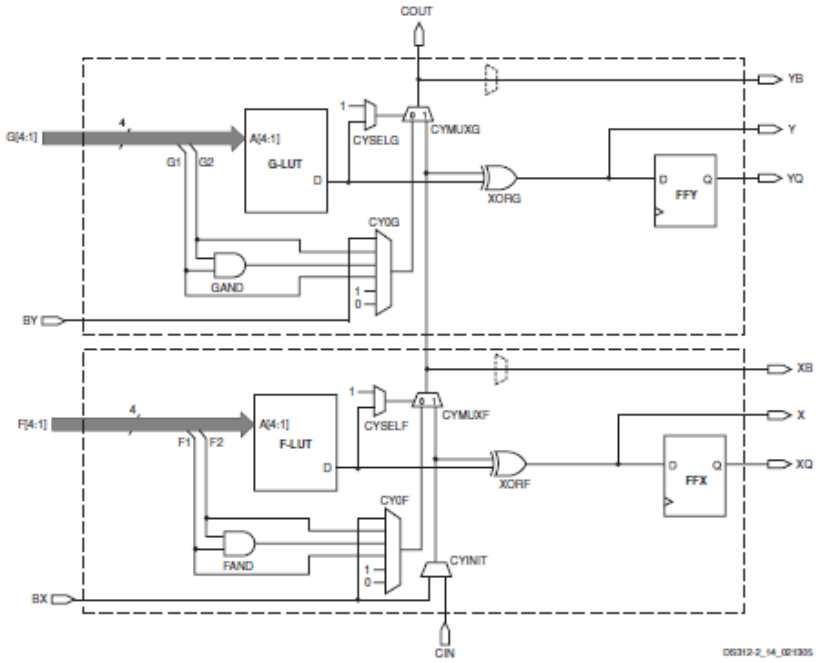


Figure 20: **Dedicated Multiplexers in Spartan-3E CLB**

Logique arithmétique et de retenu

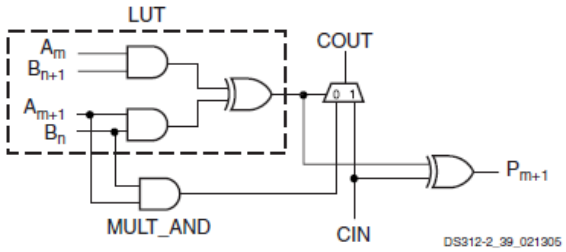
Chaque slice contient une logique arithmétique dédiée pour générer rapidement une retenue (carry). Cette logique dédiée accélère grandement toutes les opérations arithmétiques telles que l'addition, la soustraction, l'accumulation, la comparaison... Elle accélère aussi la vitesse de fonctionnement des compteurs. Chaque slice peut être configuré comme un additionneur 2 bits avec retenue qui peut être étendu à n'importe quelle taille avec d'autres CLB. La sortie retenue (COUT) est passée au CLB se trouvant au-dessus. La retenue se propage en utilisant une interconnexion directe.



La chaine de retenu connecte directement les CLB voisins à travers CIN et COUT. Il est possible toutefois de l'initialiser n'importe où par l'entrée BX (BY). La logique arithmétique dédiée comporte les portes XORF et XORG ainsi que les portes GAND et FAND.

L'usage principal de la logique de retenu est la génération d'une somme partielle dans la LUT à travers la porte XOR, qui génère ou propage une sortie retenu COUT via CYMUXF/G et complète la somme par XORF/G et CIN. Cette structure permet deux bits dans chaque tranche.

La porte F/GAND est utilisée pour une multiplication partielle et peut être instanciée en utilisant le composant MULT_AND



Le composant MULT_AND est utile pour les petits multiplicateurs. Pour les plus grands, il est préférable d'utiliser les blocs de multiplicateurs dédiés 18x18.

Les éléments de stockages

L'élément de stockage, qui peut être programmé en tant que bascule D ou en tant que latch, assure la synchronisation des données avec l'horloge, entre autres utilisations. FFY possède un MUX fixe à l'entrée D, choisissant soit la sortie combinatoire soit le signal de contournement BY. FFX sélecte entre X et BX.

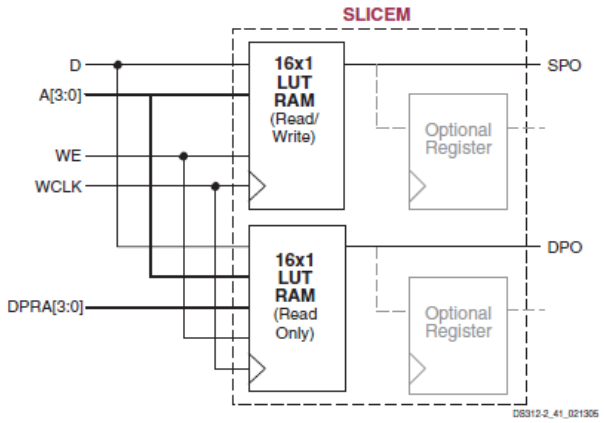
Table 16: **FD Flip-Flop Functionality with Synchronous Reset, Set, and Clock Enable**

Inputs					Outputs
R	S	CE	D	C	Q
1	X	X	X	↑	0
0	1	X	X	↑	1
0	0	0	X	X	No Change
0	0	1	1	↑	1
0	0	1	0	↑	0

Les éléments de stockages sont initialisés à la mise sous tension, durant la configuration, par le signal global GSR, et par les entrées individuelles du CLB SR ou REV.

La RAM distribuée

Les LUT dans SLICEM peuvent être programmés en RAM distribuée. Ce type de mémoire permet des quantités modérées de tomponnage de données dans le chemin logique. Une LUT SLICEM stocke 16 bits (RAM16). Les quatre entrées F[4 :1] et G[4 :1] deviennent lignes d'adresses. Il est possible de combiner plusieurs LUTS SLICEM pour stocker plus de données, soit 16x4, 32x2 ou 64x1 dans un seul CLB.



Les cinquièmes et sixièmes lignes nécessaires pour adresser les configurations 32 et 64 bits sont implémentées via les entrées BY et BX.

L'opération d'écriture est toujours synchronisée avec l'horloge du SLICEM (WCLK) et est autorisée par l'entrée SR qui fait office d'entrée write enable (WE) active-haut.

L'opération d'écriture est asynchrone, donc, durant l'écriture, la sortie reflète initialement la donnée ancienne dans l'adresse en cours.

Une option dual-port combine deux LUT de manière que l'accès à la mémoire est possible à partir de deux lignes de données séparées. La même donnée est écrite sur les deux mémoires 16x1 mais ayant des lignes d'adresses d'écriture et des sorties différentes.

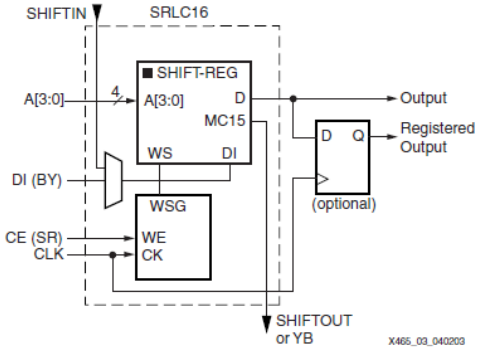
Cette fonction est implémentée en cascadant les deux lignes d'adresse G-LUT, qui sont utilisées pour l'écriture et la lecture, avec les lignes d'adressage d'écriture du F-LUT. L'entrée des données D1 du G-LUT est cascadée avec D1 du F-LUT. Un CLB assure alors une mémoire dual-port 16x1.

Toute opération d'écriture sur l'entrée de données D, et toute opération de lecture sur la sortie SPO peuvent être faites simultanément ou indépendamment d'une lecture de la deuxième sortie de lecture seule DPO.

La RAM distribuée est utile pour de petites mémoires. Des besoins plus grands peuvent être satisfaits en utilisant les blocs mémoires dédiés 18Kbit.

Les registres à décalage

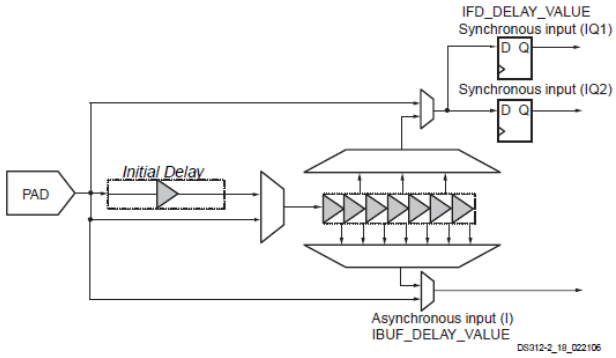
Il est possible de programmer chaque SLICEM LUT comme registre à décalage 16-bits. Utilisée comme telle, chaque LUT peut retarder les données de 1 à 16 cycles d'horloge sans avoir recours aux bascules dédiées. Ces retards programmables peuvent être utilisés pour balancer le timing des données pipeline.



Les quatre LUTs peuvent être cascadées afin de réaliser des retards allant jusqu'à 64 cycles d'horloge. Il est aussi possible de produire de plus grands registres à décalage en cascadant plusieurs CLBs.

Les fonctions retards d'entrée

Chaque bloc IOB possède un bloc de retard programmable, capable de retarder le signal d'entrée de 0 à 4000ps. Dans la figure, le signal est d'abord retardé de 2000ps et est alors appliqué à une ligne de retard 8tap, chaque tap de 250ps.



Les retards sont définis une fois pour toute lors de la configuration- ils ne sont pas modifiables lors du fonctionnement.

Les fonctions de l'élément de stockage

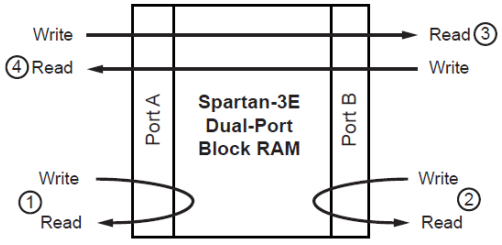
Trois paires de bascules existent dans chaque IOB, une paire pour chaque chemin. Elles peuvent être configurées en bascule D ou en latch.

La bascule dans le chemin de sortie ou celui trois états, peut être utilisée avec un MUX pour produire une transmission en double taux DDR.

Les blocs mémoires dédiés

Le XC3S500E contient en total 20 blocs mémoires groupés en deux colonnes et 368640 locations adressables. Immédiatement devant chaque bloc mémoire se trouve un multiplicateur 18x18 embarqué. Les 16 premiers bits de l'entrée de donnée A (du bloc mémoire) sont partagés avec les 16 bits supérieurs du bus multiplicande, et de même pour le port et multiplicande B.

Les blocs mémoire ont une structure dual-port. Les deux ports de données appelés A et B permettent un accès indépendant au bloc mémoire commun, dont la capacité atteigne 18432. Chaque port possède ses propres lignes de données, de contrôle et d'horloge pour des écritures et lectures synchrones.



Les quatre chemins de données sont :

1. Ecrire et lire à travers le port A
2. Ecrire et lire à travers le port B

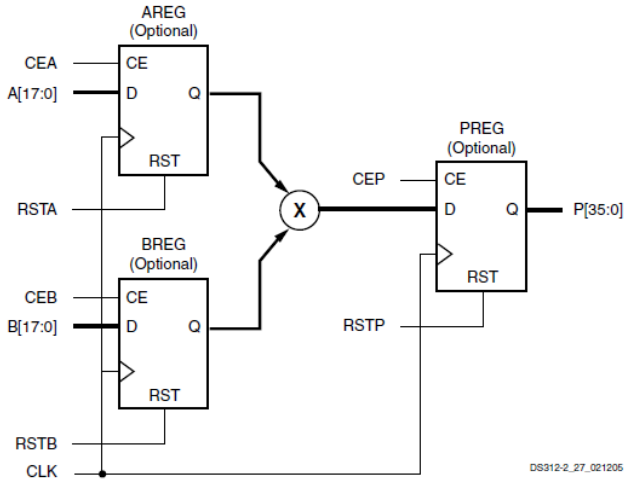
3. Transfert de donnée du port A vers le port B
4. Transfert de donnée du port B vers le port A

Le choix de la primitive dans le logicielle de CAO permet de déterminer s'il s'agit d'un bloc dual-port ou port unique. Un nom de la forme RAM16_S[WA]_S[WB] rappel une mémoire dual port dont la largeurs des ports A et B sont respectivement définis par WA et WB. RAM1_S[W] par contre rappel une mémoire simple dont la largeur du port est W.

Il est possible de configurer des largeurs différentes pour les lignes d'entrée et de sortie.

Les multiplicateurs dédiés

Les blocs de multiplicateurs réalisent la multiplication en complément à deux, mais est capable aussi de réaliser des opérations plus simples telles que le stockage ou le décalage.



Les registres AREG, BREG et PREG qui possèdent une horloge commune mais des clock enable et reset synchrone indépendants, sont idéaux pour stocker des échantillons de données ou des coefficients. Lorsqu'ils sont utilisés pour le pipelining, les registres boostent le clock rate, ce qui est bénéfique pour les applications de haute performance.

Les gestionnaires d'horloges (DCM)

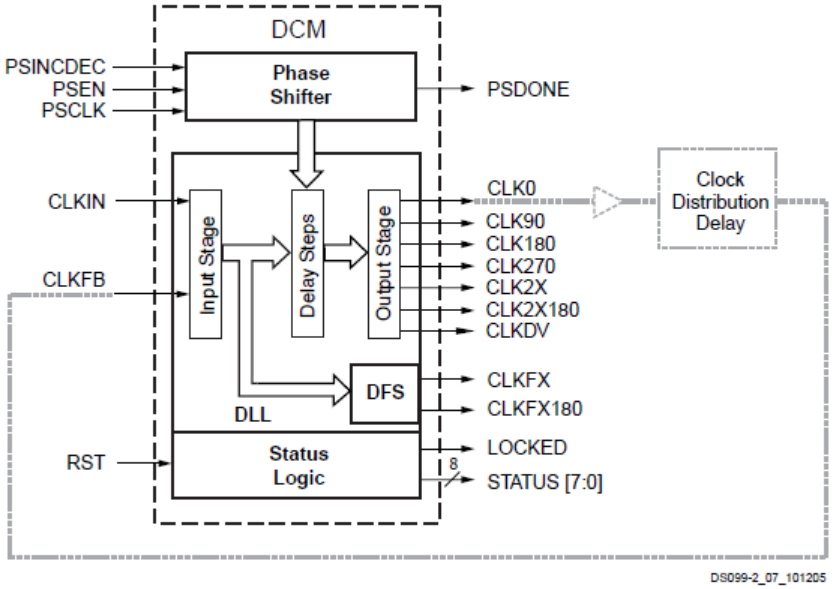
Les DCM assurent un contrôle flexible et complet de la fréquence, le déphasage et le skew de l'horloge. Pour ce faire, les DCM utilisent une DLL, un système de contrôle entièrement digital qui utilise le feedback afin de préserver les caractéristiques du signal horloge, malgré les variations de température et tension. Le XC3S500E possède quatre DCM.

Les trois fonctions principales du DCM sont :

Elimination du clock skew : le clock skew décrit à quel point le signal horloge peut dévier d'un alignement phase-zéro, dans des conditions normales. Il a lieu lorsque de légères différences dans les retards font que le signal arrive aux différents points du chip en des moments différents. Ceci a comme résultat l'augmentation des besoins sur les hold time, clock-to-out time, chose indésirable pour les applications de haute fréquence.

Synthèse de fréquence : muni d'un signal d'entrée, le DCM génère un large gabarit de fréquences de sortie. Ceci ce fait par multiplication/division de la fréquence d'entrée par l'un de plusieurs facteurs.

Décalage de phase : DCM permet le décalage de toute ses sorties par rapport à l'entrée.


DCM
PSINCDEC
PSEN
PSCLK
Phase Shifter
PSDONE
CLKIN
CLKFB
Input Stage
Delay Steps
Output Stage
CLK0
CLK90
CLK180
CLK270
CLK2X
CLK2X180
CLKDV
Clock Distribution Delay
DFS
CLKFX
CLKFX180
DLL
RST
Status Logic
LOCKED
8
STATUS [7:0]
DS099-2_07_101205