UNIVERSITÉ MOULOUD MAMMERI TIZI-OUZOU

Faculté de Génie Électrique et D'informatique

Département d 'électronique

Mémoire de fin d'études

En vue de l'obtention du diplôme de MASTER II en Electronique

Option : Réseaux et Télécommunication

Thème

Implémentation d'une application de tracking sur FPGA.

Proposé et dirigé par :

Mr K .BENNAMANE

Réalisé par :

Hadj Said Djamal

Promotion 2012-2013

Remerciements

Je remercie le Bon Dieu de nous avoir donné la force, le courage et la volonté pour accomplir ce travail.

Je remercie mon promoteur Mr. BENNAMANE pour son aide et conseils tout au long du projet.

Je remercie également le président de jurys ainsi que les membres de jurys d'avoir accepter d'évaluer et de juger mon travail.

Mes remerciements s'adressent aussi pour la responsable du laboratoire projet Mme SLIMANI pour son soutien et sa confiance.

Dédicaces

Je dédie ce modeste travail

À Toute ma famille, mon père ma mère ainsi que mes frères.

À Tous mes ami(e)s de la promotion : Rafik, Mhenna, Rabah, Mahfoud, Sofiane, Hichem, Sidali, Malek, Mohand, Ami-said, Naima, Naima,Titi, Lidia.

À tous mes amis :
Fateh, Hakim, Farid, Lotfi.

À toutes les personnes qui me sont chères.

Djamal

Résume

Une chaîne d'acquisition permet d'extraire les infirmations qui caractérisent le milieu extérieur, dans notre cas la grandeur à extraire comme information est l'image pour nous renseigner sur l'extérieur ensuite faire le traitement nécessaire et agir selon la situation par un actionneur. Dans notre cas c'est un moteur pas à pas qui permet le suivi d'un objet selon son mouvement linéaire (un seul degré de liberté).

Cependant, le circuit numérique adéquat nécessite une architecture plus lourde et le traitement des informations doit se faire en parallèle (acquisition et actionnement) pour se rapprocher plus de la notion temps réel.

Pour cela notre choix s'est posé sur un circuit FPGA qui repend à nos exigences, qui permet une exécution parallèle des informations à une vitesse considérable.

Notre travail consiste a implémenter une application sur FPGA qui permet le traitement des données (images) reçue du capteur (CCD) pour détecter la position d'un objet dans l'image par rapport a une image référence (l'objet se situe au centre) et veiller à ce que cet'objet soit conduit et ramené au centre, en tournant le capteur fixé sur le moteur dans le sens et la position correspondant.

Cette tache a été réalisée en utilisant une carte DE2 d'Altéra, dans l'environnement de programmation Quartus et pour cela nous avons opté au plan suivant :

Les circuits programmables sont présentés au chapitre I, la carte de développement DE2 d'Altéra, et le support d'utilisation est présenté en chapitre II, l'application et les résultats sont présentés en chapitre III. Enfin nous terminons par une conclusion.

Ce travail noua a permet de découvrir le monde de la programmation des systèmes embarqués et le domaine du temps réel, j ai eu l'occasion de programmer expérimentalement un circuit logique programmable de type FPGA, sous l'environnement de développement QUARTUS d'ALTERA sous différents modes de programmation.

Mots clés :

FPGA, VHDL, Quartus, Implimentation, Tracking.

Sommaire

Chapitre II : Présentation de la carte DE2

Chapitre III : Application

Introduction générale

Introduction

Une chaîne d'acquisition permet d'extraire les infirmations qui caractérisent le milieu extérieur, dans notre cas la grandeur à extraire comme information est l'image pour nous renseigner sur l'extérieur ensuite faire le traitement nécessaire et agir selon la situation par un actionneur. Dans notre cas c'est un moteur pas à pas qui permet le suivi d'un objet selon son mouvement linéaire (un seul degré de liberté).

Cependant, le circuit numérique adéquat nécessite une architecture plus lourde et le traitement des informations doit se faire en parallèle (acquisition et actionnement) pour se rapprocher plus de la notion temps réel.

Pour cela notre choix s'est posé sur un circuit FPGA qui repend à nos exigences, qui permet une exécution parallèle des informations à une vitesse considérable.

Notre travail consiste a implémenter une application sur FPGA qui permet le traitement des données (images) reçue du capteur (CCD) pour détecter la position d'un objet dans l'image par rapport a une image référence (l'objet se situe au centre) et veiller à ce que cet'objet soit conduit et ramené au centre, en tournant le capteur fixé sur le moteur dans le sens et la position correspondant.

Cette tache a été réalisée en utilisant une carte DE2 d'Altéra, dans l'environnement de programmation Quartus et pour cela nous avons opté au plan suivant :

Les circuits programmables sont présentés au chapitre I, la carte de développement DE2 d'Altéra, et le support d'utilisation est présenté en chapitre II, l'application et les résultats sont présentés en chapitre III. Enfin nous terminons par une conclusion.

Chapitre I

I.1 Introduction

Il y a quelques années la réalisation d'un montage en électronique numérique impliquait l'utilisation d'un nombre important de circuits intégrés logiques. Ceci avait pour conséquences un prix de revient élevé, une mise en œuvre complexe et un circuit imprimé de taille importante.

Le développement des mémoires utilisées en informatique fut à l'origine des premiers circuits logiques programmables (PLD : programmable logic device). Ce type de produit peut intégrer dans un seul circuit plusieurs fonctions logiques programmables par l'utilisateur. Sa mise en œuvre se fait très facilement à l'aide d'un programmateur, d'un micro-ordinateur et d'un logiciel adapté.

I.2 Structure de base d'PLD

La plupart des PLDs suivent la structure suivante :

- Un bloc d'entrée qui permet de fournir au bloc combinatoire l etat de chaque entrée et de son complément.
- Un bloc combinatoire et logique qui comporte
  - Un ensemble d'opérateurs « ET » sur lesquels viennent se connecter les variables d'entrée et leurs compléments.
  - Un ensemble d'opérateurs « OU » sur lesquels les sorties des opérateurs « ET » sont connectées.
- Un bloc de sortie.
- Un bloc d'entrée-sortie.

Une éventuelle structure de sortie (Portes inverseuses, logique 3 états, registres...).

Les deux ensembles logiques forment chacun ce qu'on appelle une matrice. Les interconnexions de ces matrices doivent être programmables. C'est la raison pour laquelle elles sont assurées par des fusibles qui sont grillés lors de la programmation. Lorsqu'un PLD est vierge toutes les connexions sont assurées.

Un exemple de ce type de structure est présenté par la figure suivante.

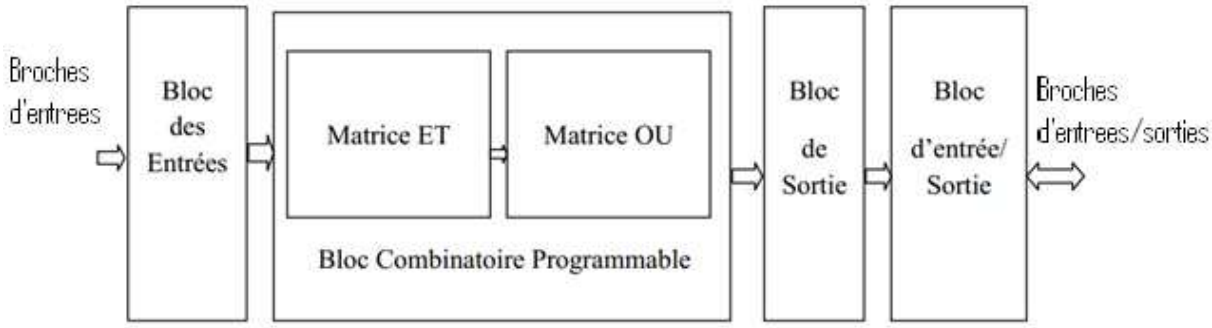


Figure I.1 Structure de base d'un PLD

I.3 Les différentes familles de PLD

Il existe plusieurs familles de PLD qui sont différenciées par leur structure interne. Le tableau suivant présente certaines de ces familles.

TYPE	Nombre de portes intégrées	Matrice ET	Matrice OU	Effaçable
PAL	10 à 100	Programmable	Fixe	Non
GAL	10 à 100	Programmable	Fixe	Electriquement
EPLD	100 à 3000	Programmable	Fixe	Par U-V
FPLA	2000 à 3000	Programmable	Programmable	Electriquement
FPGA	Plus de 50 000	Programmable	Programmable	Electriquement

Tableau 1 récapitulatif des différentes familles des PLDs

I.3.1 Les PALs (Programmable Array logique ou réseau logique programmable)

Les PALs, ce sont les premiers circuits intégrés programmables pour réaliser des fonctions logiques. Ce type de circuit a été développé par le constructeur AMD. Ils sont constitués de matrices ‹‹ET›› programmables et de matrices ‹‹OU›› fixes.

La programmation de ces PAL s'effectue par la destruction des fusibles, et une fois programmés on ne peut plus les effacer.

On distingue deux sous familles :

❖ Les PAL combinatoires (simple)

Ce type de PAL possède une architecture interne la plus simple car il possède uniquement des portes logiques, ils sont conçus pour la représentation des fonctions de logiques combinatoires.

❖ Les PAL séquentielles

Parmi les types de PAL séquentielle on distingue trois types qui sont :

- **Les PAL à registre (FPLS)**

Les PALs à registres ou FPLS (Field Programmable Logic Séquencer), c'est séquenceur sont constitués de portes logiques, de bascules et de registres.

- **Les PAL asynchrone à registre**

Ce type de PAL diffère du précédent de la manière de répartition et distribution du signal d'horloge vis-à-vis des bascules internes, en effet, dans ce type chaque bascule dispose de son propre signal d'horloge (combinaison de signaux avec des multiplicateurs), aussi il possède un multiplexeur qui permet la sélection du registre avec les entrées de sélection AP et AR qui sont aussi des reset asynchrones.

- **Les PAL versatiles (VPAL)**

Ces VPAL présentent une évolution des PAL vers les circuits logique programmable de haut niveau, en effet ils continuent à respecter les principes généraux des PALs mais utilisent une structure de cellule de sortie très proche des macro-cellules ou OLMC des GALs que nous verrons par la suite, la structure de ça sortie est dite versatile et configurable, le mode de fonctionnement est obtenu en utilisant deux multiplexeurs qui utilisent comme signal de sélection les points de programmation S0 et S1, la combinaison de S0 et S1 donne les différentes configurations de la sortie.

I.3.2 Les GAL (GENERIC ARRAY LOGIC)

Les PALs possèdent un inconvénient majeur qui réside dans l'impossibilité de la reprogrammation une deuxième fois, dans le but de remédier a ce problème on a penser à remplacer les fusibles irréversibles des PALs par des transistors MOS FET pouvant être regénérés, ces circuits peuvent donc être reprogrammés à volonté sans pour autant avoir une durée de vie restreinte et cela a donné naissance a ce qu'on appelle GAL, on a aussi remplacé la technologie bipolaire des PALs par la CMOS dans les GALs ce qui a réduit la consommation de l'énergie.

I.3.3 Les EPLDs (Erasable Programmable Logic Device)

Les EPLDs sont des circuits logiques programmables électriquement et effaçable, soit par ultraviolets ou électriquement, Ces circuits, comme les PAL et les GAL, font appel à la notion de macro-cellule qui permet, par programmation, de réaliser de nombreuses fonctions logiques de base. Ils sont caractérisés par leur densité d'intégration qui est nettement supérieure à celle offerte par les PAL et une vitesse de fonctionnement comparable à celle des PALs bipolaires. En plus de ces deux caractéristiques, les EPLD possèdent un atout majeur qui réside dans la possibilité d'effacement, ils ont été mis en œuvre par la firme ALTERA qui les a commercialisé pour la première fois en 1984.

I.3.4 Les CPLD (Complex Programmable Logic Device)

Les PAL, GAL, CPLD et EPLD, sont de conception plus ancienne, utilisent des « macro-cellules » logiques, composées d'un réseau combinatoire de portes ET et OU afin d'implémenter des équations logiques. Des bascules sont disponibles seulement dans les blocs d'entrée-sortie. Un composant contient de quelques dizaines à quelques centaines de macro-cellules.

Comme le routage est fixe, les temps de propagations sont bornés et permettent une fréquence de fonctionnement élevée et indépendante du design. Par contre, l'utilisation des ressources n'est pas optimale (tout terme non utilisé dans une équation logique équivaut à des portes perdues), avec des taux d'utilisation d'environ 25 %.

On distingue les CPLD des autres PLD car ils contiennent l'équivalent de plusieurs composants PLDs, reliés par une matrice d'interconnexion. Un certain nombre de macro-cellules de base sont regroupées pour former des blocs logiques. Grâce à leurs structures, ils peuvent atteindre des vitesses de fonctionnement élevées (plusieurs centaines de MHz).

Ces circuits ont une capacité en nombre de portes et en possibilités de configuration très supérieure à celle des précédents.

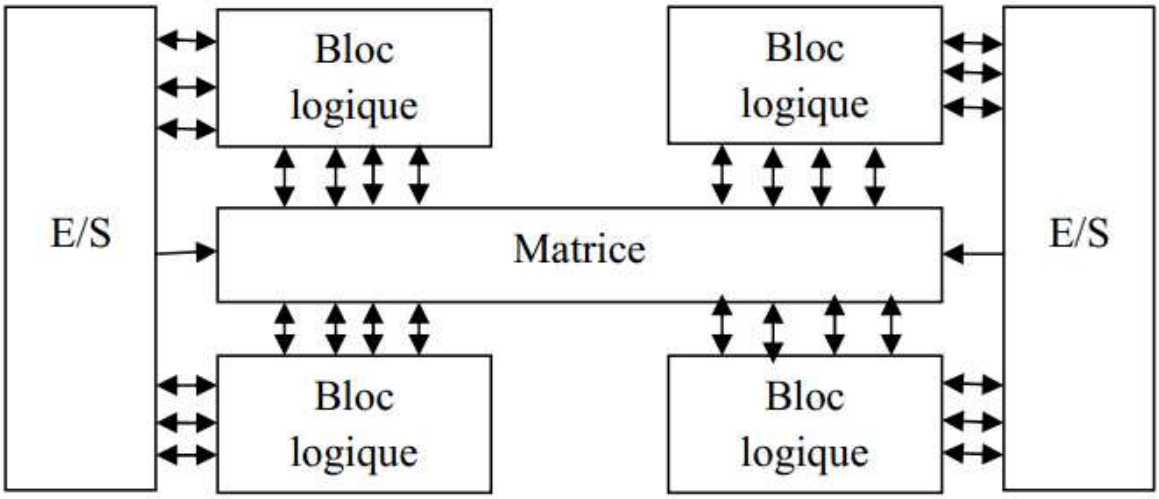


Figure I.2 Physionomie d'un CPLD

I.3.5 Les FPGA (Field Programmable Gate Array)

Un FPGA (réseau de cellules logiques programmables) est un circuit logique reprogrammable. Les FPGAs sont des circuits intégrés qui contiennent des blocs logiques configurables (programmables), ainsi que des interconnexions entre ces blocs configurables.

Les premiers circuits intégrés reconfigurables de type FPGA ont été commercialisés par la société XILINX en 1984 et commercialisé en 1985. Depuis d'autres fabriquants sont apparus sur ce marché comme ALTERA, ACTEL, Texas Instrument…. Au fur et à mesure que la complexité des FPGA s'est développée, jusqu'à concurrencer sérieusement les circuits spécifiques (les ASICs) pour de petits volumes d'integration. C'est pourquoi on envisage de plus en plus de développer l'utilisation des FPGA dans les diverses applications.

I.3.5.1 Généralité

Les FPGAs sont utilisés dans diverses applications de l'électronique numérique (télécommunications, aéronautique, …), ils sont également utilisés pour le prototypage des ASICs. Ils ont plusieurs avantages :

- Délai de mise sur le marché plus court, car ce sont des composants standards,
- Temps de développement plus court, car on réutilise des fonctions de base et la reconfigurabilité autorise une validation préalable moins stricte,

- Coût inférieur pour de petites séries (moins de 10 000 unités). Avec l'évolution technologique, cette quantité tend à augmenter, en effet, le prix d'une puce est proportionnel à sa surface, qui diminue avec la finesse de gravure.
- Il est parfois possible de transformer directement un FPGA en une version ASIC plus rapide, moins chère et consommant moins (car les matrices de routage sont remplacées par une couche de métallisation fixe).
- Plusieurs FPGA modernes possèdent la possibilité d'être reconfigurés.

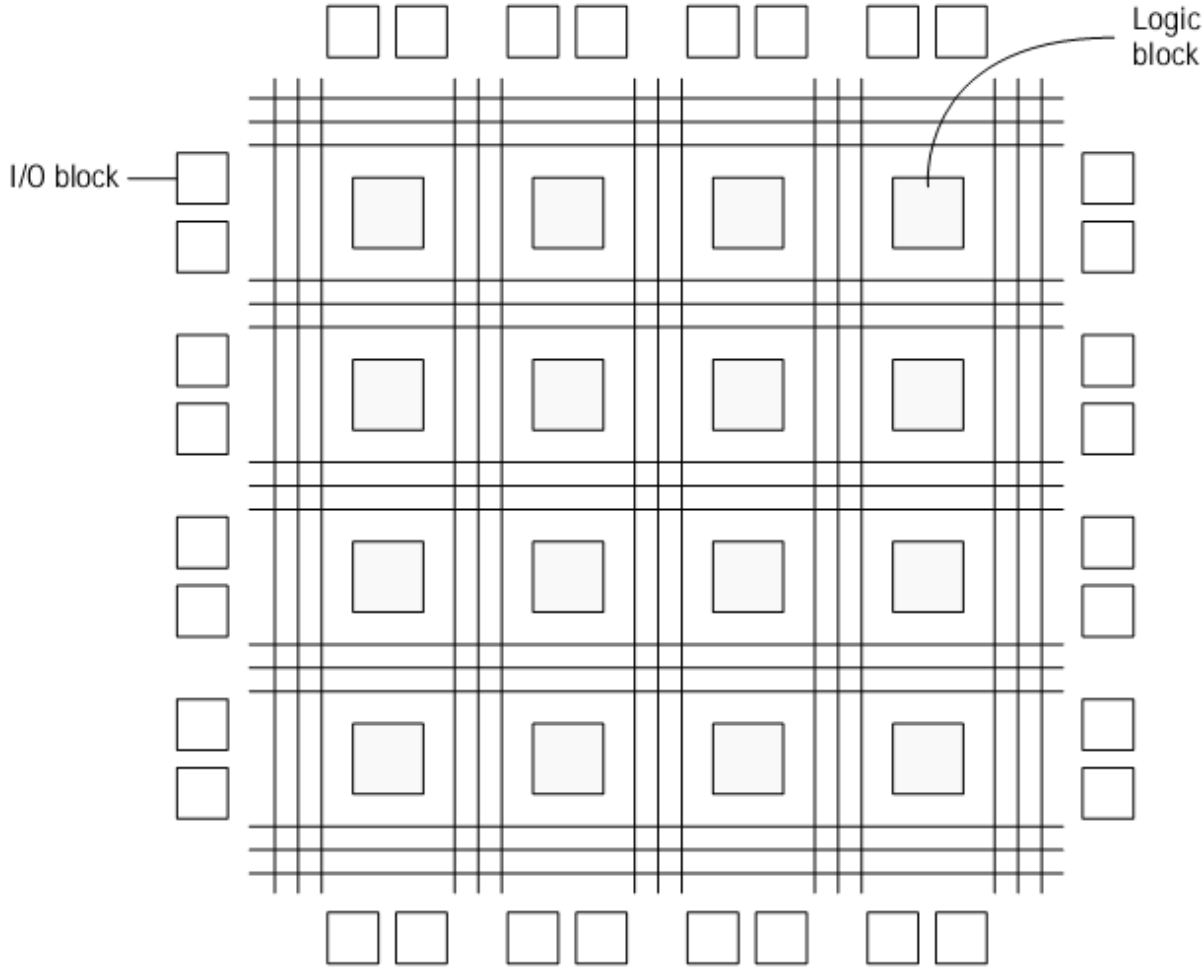


Figure I.3 Physionomie d'un FPGA

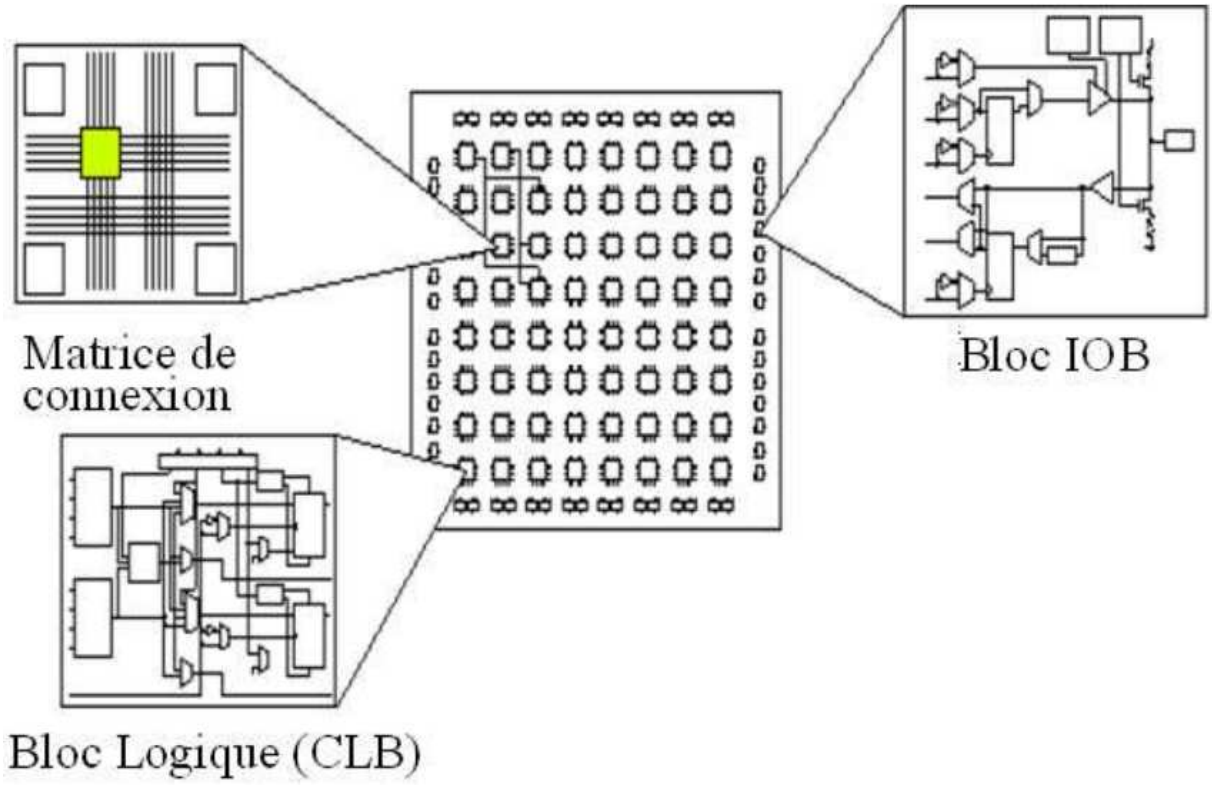


Figure I.4 Structure générale d'un FPGA

I.3.5.2 Les différents types d'FPGAs

Les FPGAs ont différentes architectures et on distingue deux types principaux de FPGA qui sont :

- Les FPGAs à anti fusibles proposé entre autre, par Texas Instrument et ACTEL.
- Les FPGAs de type SRAM, appelé LCA chez XILINX et FLEX chez ALTERA.

Ces deux technologies différent essentiellement dans la façon dont leurs interconnections sont réalisées

I.3.5.2.1 Les FPGAs de type anti fusible

Les réseaux programmables à anti-fusibles sont des composants configurables une seule fois.

Les anti-fusibles sont des composants généralement constitués de deux couches conductrices entourant un diélectrique. L'application d'une tension élevée fait claquer le diélectrique créant une liaison entre les deux couches conductrices du composant. Les anti-fusibles peuvent donc être considérés comme des circuits ouverts dans leur état initial, et des circuits

fermés une fois programmés. La Figure ci-dessous montre un anti-fusible utilisé pour configurer une connexion entre deux fils. L'avantage de cette technologie est sa faible taille, même si les transistors utilisés pour la configuration sont dimensionnés de manière à laisser passer de forts courants lors de la programmation et réaliser une bonne isolation entre les tensions élevées de configuration et les faibles tensions de fonctionnement, et sont de fait de taille imposante. Les autres points forts de ces composants sont leur haut niveau de confidentialité, due aux difficultés de relecture de l'état du fusible, et leur faible sensibilité aux radiations et autres perturbations environnementales, induisant une bonne fiabilité. Ces caractéristiques impliquent cependant un manque de flexibilité, ainsi que des fortes tensions de programmation.

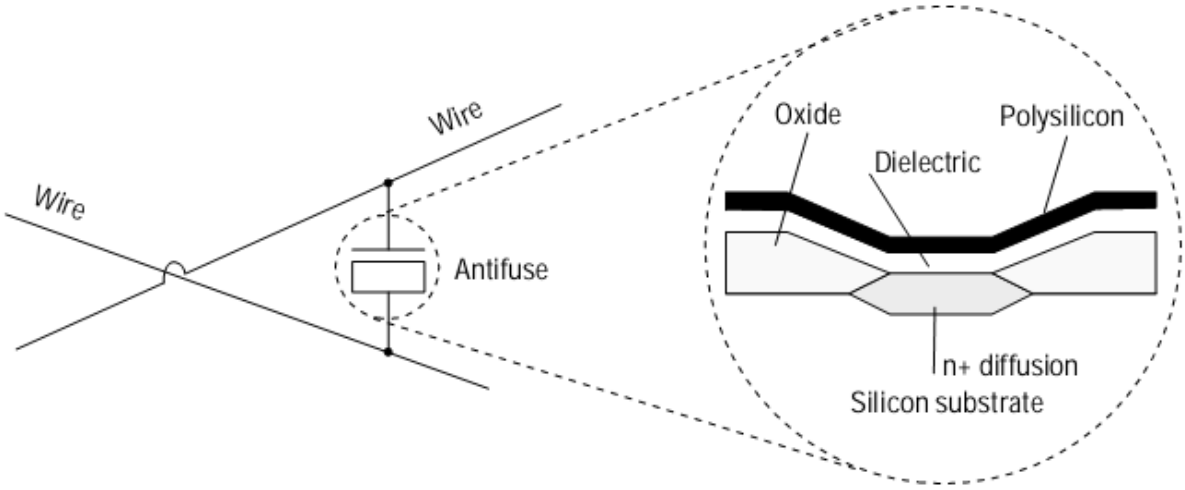


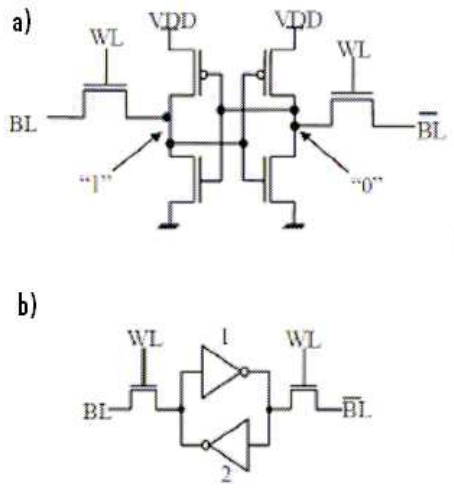
Figure I.5 Programmation par anti-fusible

I.3.5.2.2 Les FPGAs de type SRAM

La structure de base d'un FPGA de type SRAM est complexe, le point de connexion entre les différentes cellules est un ensemble de transistors MOS de commutation commandés par des cellules de mémoire vive (RAM).

Les mémoires SRAM sont également utilisées pour la conception de réseaux programmables du fait de leurs performances (leur rapidité d'accès notamment) et de leur faible coût. On trouve deux grandes architectures pour ces points mémoires, les 6T-SRAM et 5T-SRAM, respectivement à six et cinq transistors. La cellule à six transistors est constituée de deux inverseurs montés tête-bêche et deux transistors d'accès (Figure II.6). Elle est commandée par le bit « Word Line » (WL, Figure II.6.a) pour l'écriture et la lecture. On positionne des valeurs complémentaires sur BL et BL| tout en ayant WL à « 1 » pour écrire

une donnée BL. On laisse les lignes BL et BL| flottantes et on positionne WL à « 1 » pour lire la donnée contenue par le point mémoire.



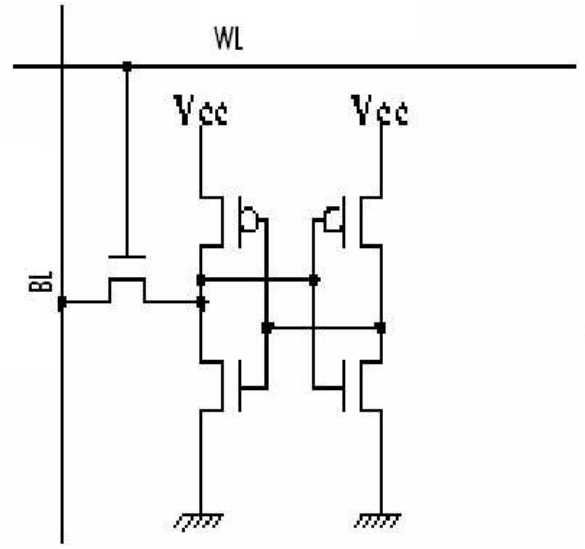


Figure I.6 Cellule SRAM à six transistors *Figure I.7 Cellule SRAM à cinq transistors*

La configuration est stockée dans une mémoire SRAM qui présente l'avantage d'être relativement compacte et rapide, bien qu'elle est aussi volatile et sensible aux radiations naturelles.

I.4 Etude de la structure générale d'un circuit FPGA

I.4.1 Les cellules logiques de base

On distingue deux principaux types de cellules de base qui sont :

- **Les cellules d'entrées-sorties**

Ils constituent l'interface entre les branches de sortie du circuit et les CLB. Ils sont présents sur toute la périphérie du circuit FPGA car ces cellules sont des intermédiaires par lesquelles les données transitent depuis les blocs logiques internes jusqu'aux ressources externes et vice versa. Chaque bloc IOB contrôle une broche du composant et peut être défini en entrée, en sortie, en entrée/sortie.

Le rôle principal des interfaces d'entrés/sorties est de transmettre et de recevoir des données du milieu extérieur, néanmoins l'interface d'entré/sorties peut être dotée d'options tels que des registres, impédances et buffers.

Chaque fabricant a sa propre appellation pour désigner l'interface d'entrés/sorties mais la fonction reste toujours la même.

- Chez XILINX, les interfaces d'entrés/sorties sont nommées IOB pour Input Output Blocks. L'IOB est constitué de registres, de diviseurs de tensions, des résistances de rappel pull up et autres ressources spécifiques.
- Altera, les nomme IOE Input Output Element. L'IOE remplie toujours son rôle d'interface d'entrés/sorties, elle dispose d'une résistance de rappel pull-up et un temporisateur du signal.

❖ Les macro-cellules

Ces cellules logique sont appelé aussi par :

-Soit CLB (configurable logique bloc), dénomination adoptée par XILINX.

-Soit LC (cellule logique), le nom choisi par CYPRESS.

-Soit LE (élément logique), l'appellation d'ALTERA.

Ces macro-cellules sont plus nombreuses. La macro-cellule est constituée d'une partie combinatoire et une partie séquentielle.

- La partie combinatoire sur la quelle sont réalisées les fonctions de complexité moyenne car les constructeurs ont proposé pour chacun une ou plusieurs solutions de synthèse dont les principaux sont :

➢ La synthèse de fonctions à 4 ou 5 variables avec des portes classiques ET, OU et NON (not).
➢ La synthèse de fonction utilisant un multiplieur.
➢ La synthèse de fonction combinatoire à l'aide de mémoire vive RAM.

Dans ce dernier cas, on dit aussi réalisation de fonction logique par LUT (look-up ou table d'observation).

- La partie séquentielle est dans le FPGA est représentée par des bascules, généralement de type D. il est rare de trouver des macro-cellules unique pourvues de la partie combinatoire.

I.4.2 Réseau d'interconnexion

Dans les FPGA à architecture symétrique on peut distinguer une composante d'interconnexion locale et une composante d'interconnexion globale. La composante locale a pour rôle d'interconnecter les éléments des blocs logiques en fonction de la configuration. La deuxième assure la circulation des données à l'échelle du FPGA entre les blocs logiques éloignés.

- Au niveau local, on trouve donc des interconnexions configurables au sein des blocs logiques dans et entre les cellules logiques élémentaires. Elles sont réalisées au moyen de multiplexeurs, de buffers trois états et de portes passantes (un simple NMOS ou bien une porte de transmission associant un NMOS et un PMOS en parallèle). On trouve également un groupe d'interconnexions vers les blocs logiques directement adjacents selon le même principe. Il s'agit d'interconnexions courtes présentant de faibles retards si on les compare aux interconnexions à longue distance.
- Au niveau global, on trouve un réseau d'interconnexions horizontales et verticales à l'échelle du circuit situé entre les blocs logiques montrés à la figure ci-dessous.

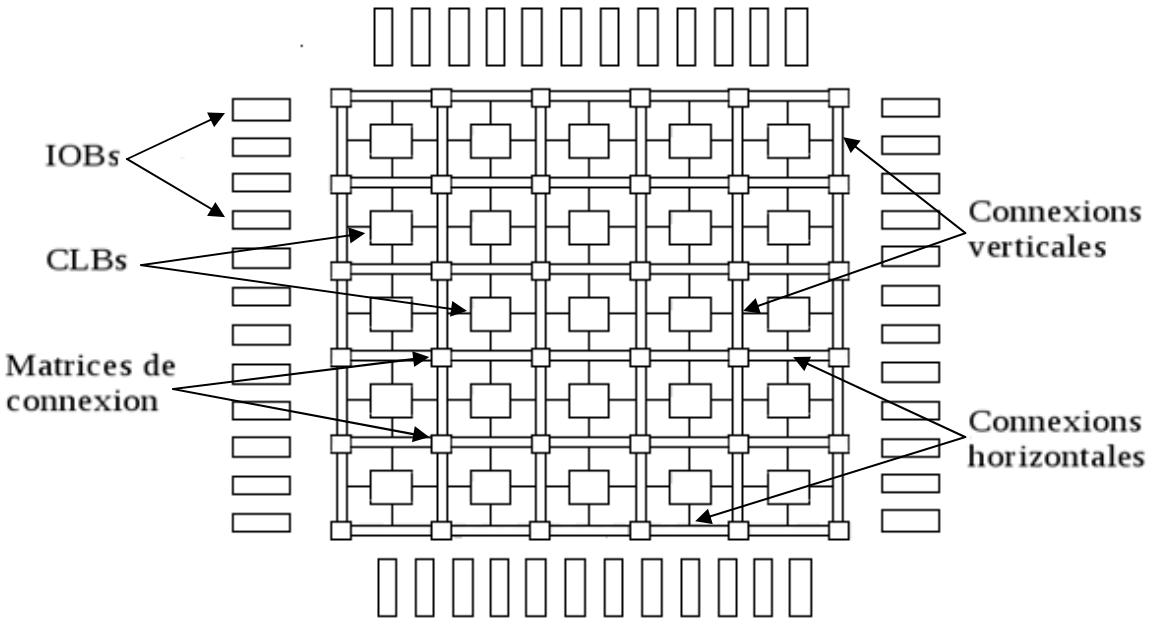


Figure I.8 Concept architectural de base des FPGA

- On distingue trois types d'interconnexions :

❖ Les interconnexions à usage général

Ce Sont des lignes verticales et horizontales qui encadrent chaque CLB, auxquelles peut être reliée par une matrice de commutation montées à la figure II.8, le rôle de cette dernière est de raccorder les segments de communication entre eux selon diverses configurations, assure aussi la commutation des signaux d'une voie à autre.

❖ Les interconnexions directes

Ces interconnexions permettent d'établir des chemins entre les CLB adjacents et les cellules entrées/sorties avec un maximum d'efficacité en terme de vitesse et d'occupation de circuit.

❖ Les longues lignes

Sont des longues lignes verticales et horizontales qui n'utilisent pas de matrice de commutation. Elles parcourent toutes la longueur et la largeur du circuit. Elles permettent aussi de transporté les signaux qui parcourt un long trajet. Elles égalisent les délais entre les signaux de façon à permettre un décalage minimum entre deux points distants de la ligne. Ces lignes conviennent pour transporter les signaux d'horloge.

- L'échange des données entre les blocs logiques et ces interconnexions globales se fait par l'intermédiaire de Cellules d'Interconnexions Locales (CIL). L'aiguillage des données au sein du quadrillage d'interconnexions globales est assuré par des Cellules d'Interconnexions Globales (CIG) placées à chacune de ces intersections selon le schéma synoptique de la figure suivante.

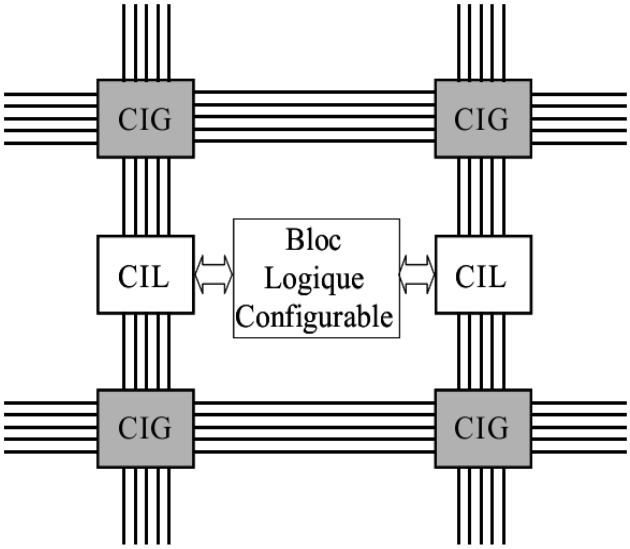


Figure I.9 Réseau d'interconnexions dans l'FPGA

I.4.3 Les éléments de mémorisation

Actuellement, Les FPGAs sont utilisés pour des applications plus complexes et plus importantes qui demandent un espace de stockage plus important à une vitesse élevée, d'où la nécessité d'intégrer des blocs de mémoire directement dans l'architecture des FPGAs est vite devenue capitale. De cette façon, les temps d'accès à la mémoire sont diminués puisqu'il n'est plus nécessaire de communiquer avec des éléments extérieurs au circuit.

I.4.4 Les éléments de contrôle des horloges

L'horloge est un élément essentiel pour le bon fonctionnement d'un système électronique.

Les circuits FPGA sont prévus pour recevoir une ou plusieurs horloges. Des entrées peuvent être spécialement réservées à ce type de signaux, ainsi que des ressources de routage spécialement adaptées au transport d'horloges sur de longues distances. Les circuits FPGA disposent des éléments d'asservissement des horloges (des PLL ou des DLL) afin d'avoir la même horloge dans tout le circuit (synchronisation des signaux). Ces éléments permettent de créer à partir d'une horloge d'autres horloges à des fréquences multiples de la fréquence de l'horloge incidente.

I.4.5 Les éléments de routage

Les éléments de routages sont les composants les plus importants dans les FPGA. En fait, ces éléments représentent la plus grosse partie du silicium consommée sur la puce du circuit. Ces ressources sont composées de segments (de longueurs différentes) qui permettent de relier entre eux les autres éléments via de s matrices de connexions. Le routage de ces ressources est un point critique du développement d'une application sur un FPGA. Ces éléments sont très importants puisqu'ils vont déterminer la vitesse et la densité logique du système. Par exemple, les matrices de routage sont physiquement réalisées grâce à des transistors de cellules SRAM, qui ont une résistance et une capacité, ce qui entraîne l'existence de constantes de temps.

I.5 Illustration avec la famille Cyclone II Altera

I.5.1 Caractéristiques principales de la famille Cyclone II

Leurs principales caractéristiques sont :

- Une gravure en 90 nm. C'est un paramètre important car plus la gravure est fine et plus l'élément de base qu'est le transistor est petit et voit sa consommation diminué, c'est-à-dire augmenter la densité d'intégration.

NB: Graver plus fin permet également de placer davantage de portes dans le FPGA à surfaces égales.

- Une architecture haute densité avec entre 4608 et 68416 LE (Eléments logiques).
- Ils intègrent jusqu'à 1.1 Mbits de RAM (M4K Blocks) avec :
- une largeur de bus de données configurable (x1, x2, x4, x8, x 16, x32 et x36),
- un mode double accès,
- une vitesse de fonctionnement pouvant atteindre 260 MHz.
- Ils comprennent une zone où sont gravés dès la fabrication jusqu'à 150 Multiplexeurs 18x18 (Embeded Multipliers).
- Ils possèdent des blocs d'entrées/sorties avancées (IOE : Input/Output Element) avec :
- des entrées/sorties différentielles (LVDS, LVPECL . . .),
- des entrées/sorties simples (3.3v, 2.5v, 1.8v et 1.5v LVCMOS, 3.3v, 2.5v et 1.8v LVTTL . . .),

- une compatibilité avec les signaux PCI et PCI Express.
- Il y a une circuiterie dédiée aux horloges (PLL) :
- 2 à 4 PLL pour multiplier ou diviser les fréquences d'horloge entrantes, les retarder ...
- jusqu'à 16 lignes d'horloges en interne
- une gestion jusqu'à la fréquence maximale de 402.5 MHz
- Un dispositif de configuration avec :
- Un mode rapide pour une configuration en moins de 100 ms
- Mode série ou JTAG possibles avec des mémoires de configuration série de bas coût.

I.5.2 Architecture interne globale

La figure ci-dessous représente un schéma synoptique de l architecture interne du Cyclone II

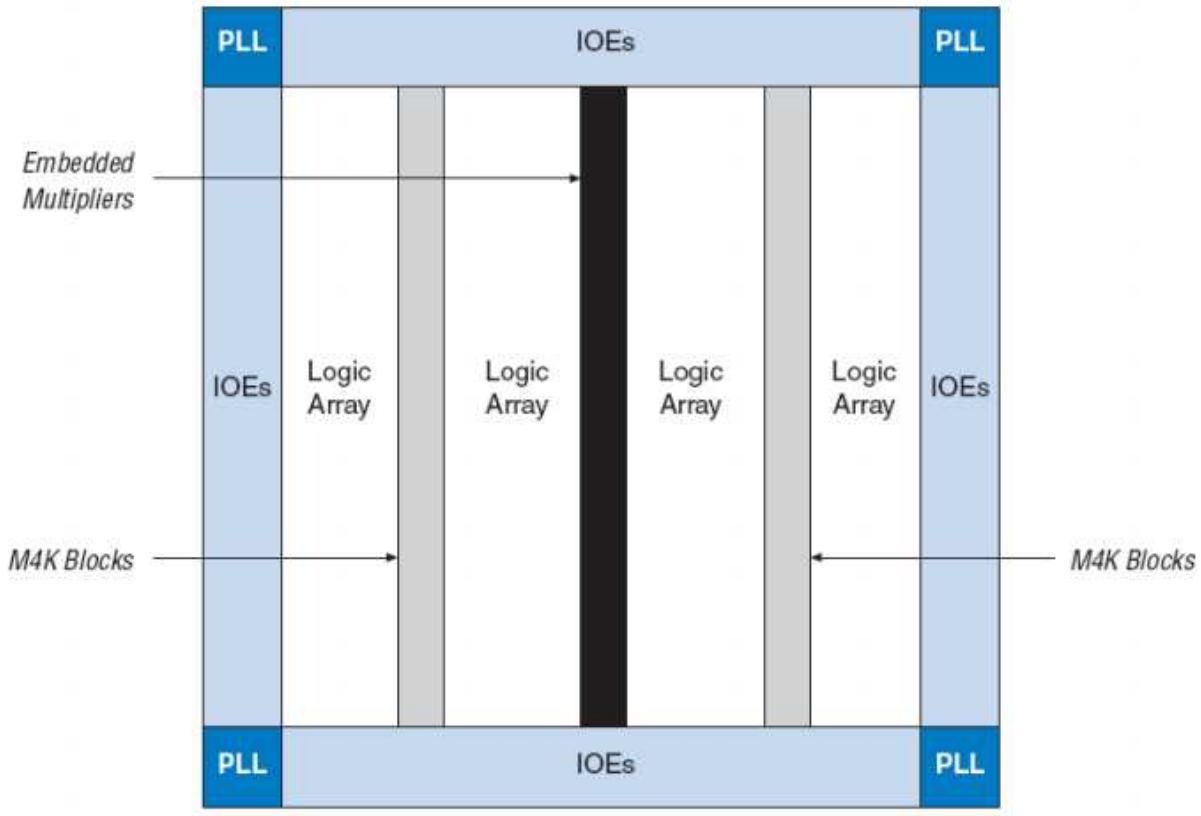


Figure I.10 Structure interne d'un Cyclone II

❖ Elément logique (LE)

C'est le plus petit élément de logique dans le Cyclone II. Sa structure apparaît sur la figure ci-dessous

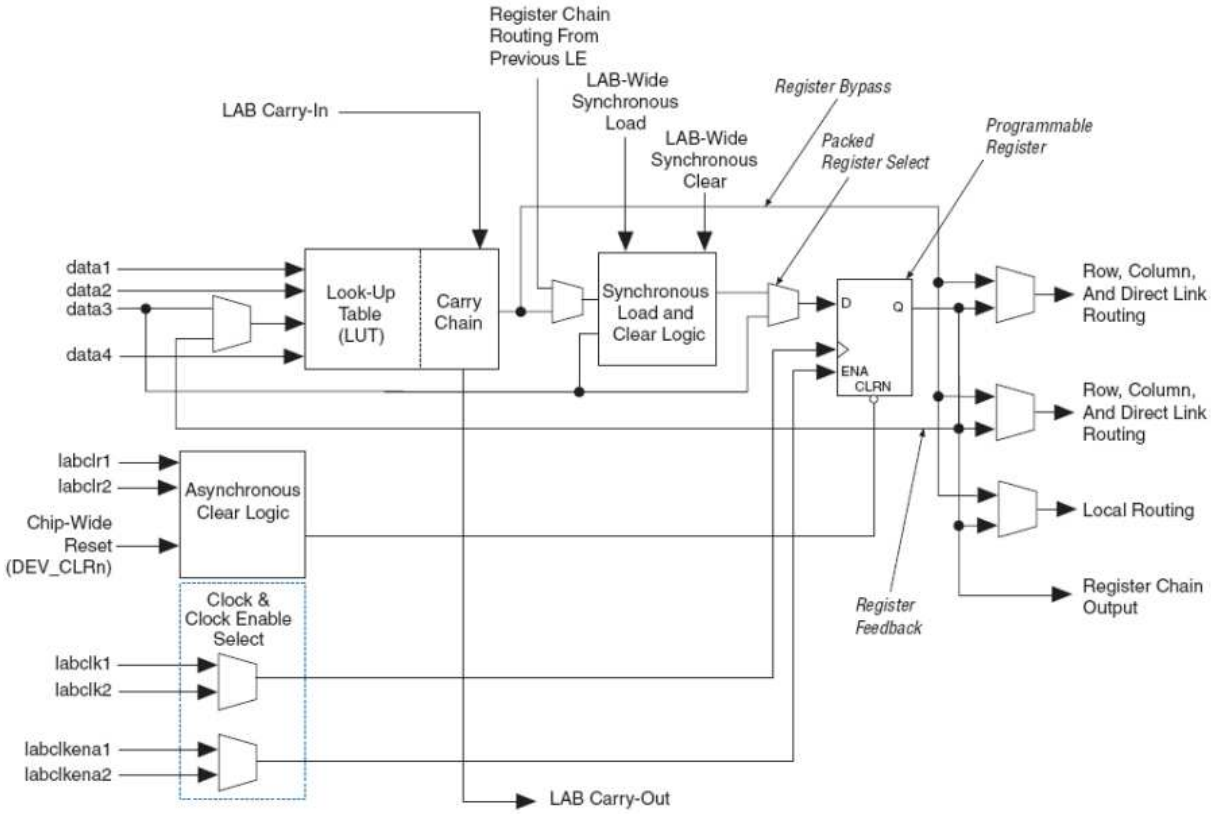


Figure I.11 Structure d'un Logic Element LE

Cette cellule reconfigurable contient divers éléments :

- Une LUT (Look up table) à 4 entrées permettant le calcul de n'importe quelle fonction de 4 entrées.
- Un registre de sortie programmable.

❖ Les blocs de réseaux logiques (LAB)

Un bloc de réseaux logiques contient :

- 16 Eléments Logiques (LE).
- Un réseau d'interconnexion local pour la communication entre LE du même LAB (voir figure II.12).
- Un accès direct aux éléments adjacents du LAB dans la structure du FPGA (voir figure II.12) tels que :
  - un autre LAB.
  - un bloc mémoire pour les LAB adjacents aux zones mémoires.
  - un signal d'horloge d'une PLL.

- un multiplexeur.
- un IOE (Elément d'entrée/sortie).
- un accès aux réseaux d'interconnexions lignes/colonnes pour atteindre n'importe quel point du composant.

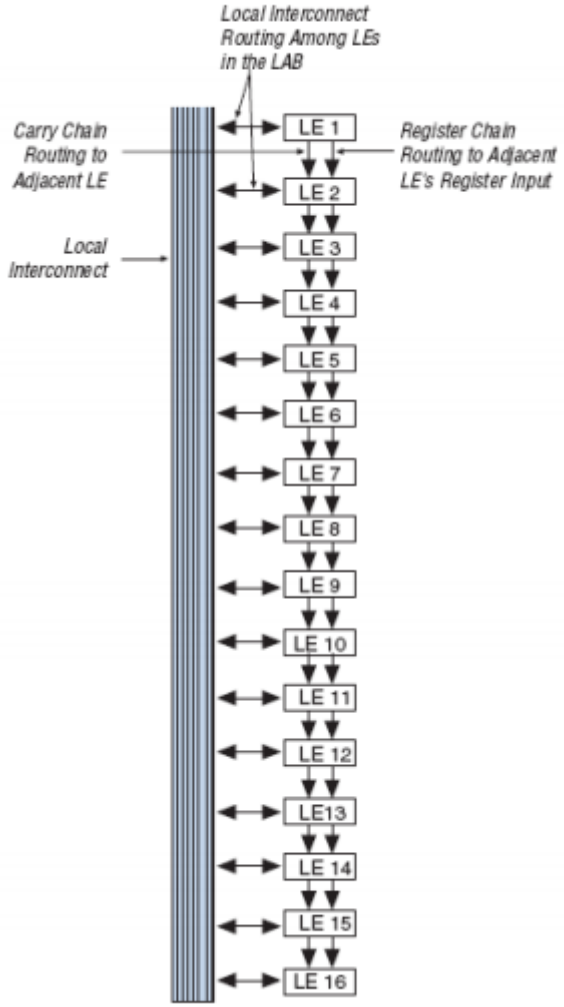


Figure I.12 Connexions chaînées entre LEs dans un LAB

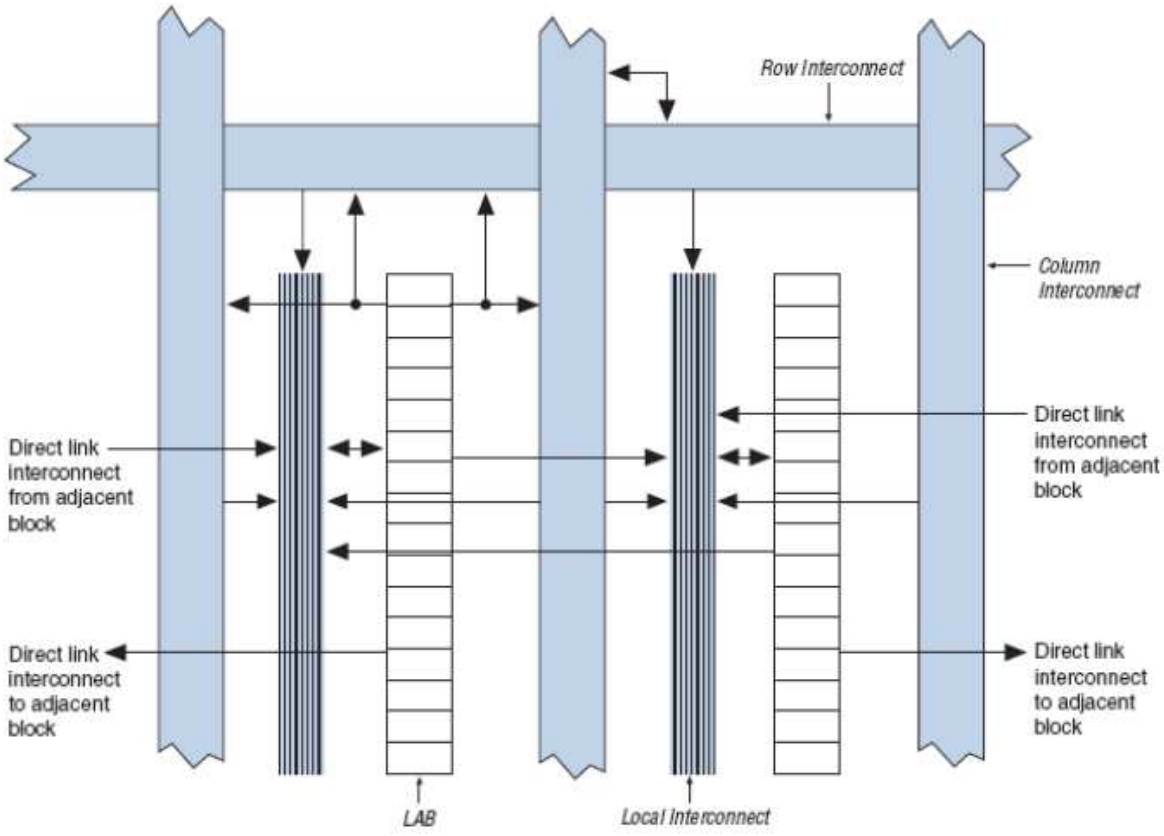


Figure I.13 Connexion directe au niveau d'un LAB

❖ Bloc d'entrée/sortie (IOE : Input Output Element)

Avec l'accroissement constant du nombre de standards d'entrées/sorties en électronique numérique, la conception d'un FPGA a progressivement nécessité de faire apparaître des blocs dédiés capables d'interfaçages s'adaptant à une grande diversité de situation.

La frange supérieure des composants de la famille Cyclone II dispose donc de nombreux blocs d'entrées/sorties (I/O Banks) répartis à la périphérie du composant.
La famille Cyclone II peut s'interfacer avec des circuits logiques :

- LVTTL et LVCMOS : Interfaçage avec des circuits logiques d'usage général, fonctionnant à des fréquences moins de 100MHz.
- SSTL : Standard mis en place pour la mémoire SDRAM DDR (Double Data Rate).
- HSTL : Signaux des mémoires QDR^2 SRAM (Quad Data Rate).
- LVDS (Low Voltage Differential Signaling) : Signaux différentiels (ils garantissent une plus grande immunité au bruit) pour des communications à fort débit (jusqu'à 805Mbps) et faible EMI (émissions électromagnétiques)

- LVPECL (Low Voltage Positive Emitter Coupled Logic) : Signaux différentiels à haute immunité au bruit utilisés en vidéo, télécom, distribution d'horloge.
- PCI et PCI Express Bus locaux des PC utilisés pour la connexion de car te d'extension (Vidéo . . .).

I.6 Conclusion

Dans ce chapitre nous avons présenté les différents circuits logiques programmables et conclure que la technologie FPGA s’inscrit au sommet de l’évolution des composants logiques programmables et ouvrent de grandes perspectives en terme de cout, rapidité et le contrôle en temps réel.

Chapitre II

II.1 Introduction

Les circuits programmables FPGA offrent et permettent a l'utilisateur l'avantage d'intégrer et d'implémenter des systèmes et fonctions beaucoup plus complexes sur un seul circuit (FPGA).

IL existe des constructeurs joignent des périphériques au circuit FPGA, tels que les ports de communications, des afficheurs, des ports d'extension,… et ils offrent des cartes (munies de FPGA+ périphériques) adéquates pour diverses applications, parmi ces carte celle que nous avons utilisé la DE2 de Altera.

II.2 Vue générale sur la carte DE2

La figure ci-dessous nous montre la disposition des composants de la carte DE2 et indique l emplacement des connecteurs et des composants.

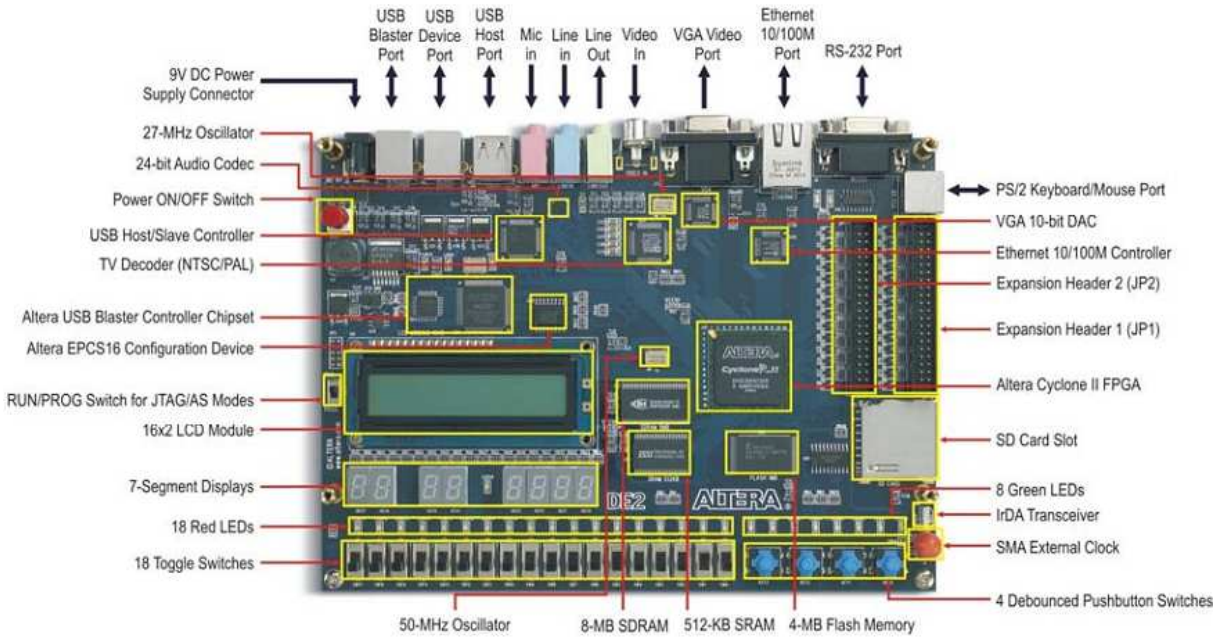


Figure II.1 Aperçu de la carte DE2

La carte DE2 a beaucoup de dispositifs permettent à l'utilisateur de faire de diverses implémentations de circuits, de simples à plusieurs projets multimédias.

Les différents constituants de la carte DE2 sont :

- Un circuit Altera Cyclone® II 2C35 FPGA device.
- Un circuit de configuration série Altera EPCS16.
- USB Blaster pour la programmation et l utilisation du contrôle API.
- Une mémoire SRAM de 512 koctets.
- Une mémoire SDRAM de 8 Moctets.
- Une mémoire Flash de 4 Moctets.
- SD Card socket
- 4 boutons poussoirs
- 18 interrupteurs (switches).
- 27 LEDs (18 rouges et 9 vertes).
- 2 oscillateurs (source d'horloge) de 50-MHz et 27-MHz.

En plus de ceux-ci traits du matériel, la carte DE2 est accompagnée d un support logiciel qui met en œuvre les standards et les interfaces d entrée/sortie et d un control panel pour faciliter l accès au différents composants de la carte ou périphériques. Et aussi le logiciel est accompagné de plusieurs exemples de démonstrations qui illustrent les capacités avancées de la carte DE2.

II.3 Schéma bloc de la carte DE2

La figure I.2 représente le synoptique de la carte et ses différents blocs constitutifs et leurs emplacements vis-à-vis de circuit FPGA.

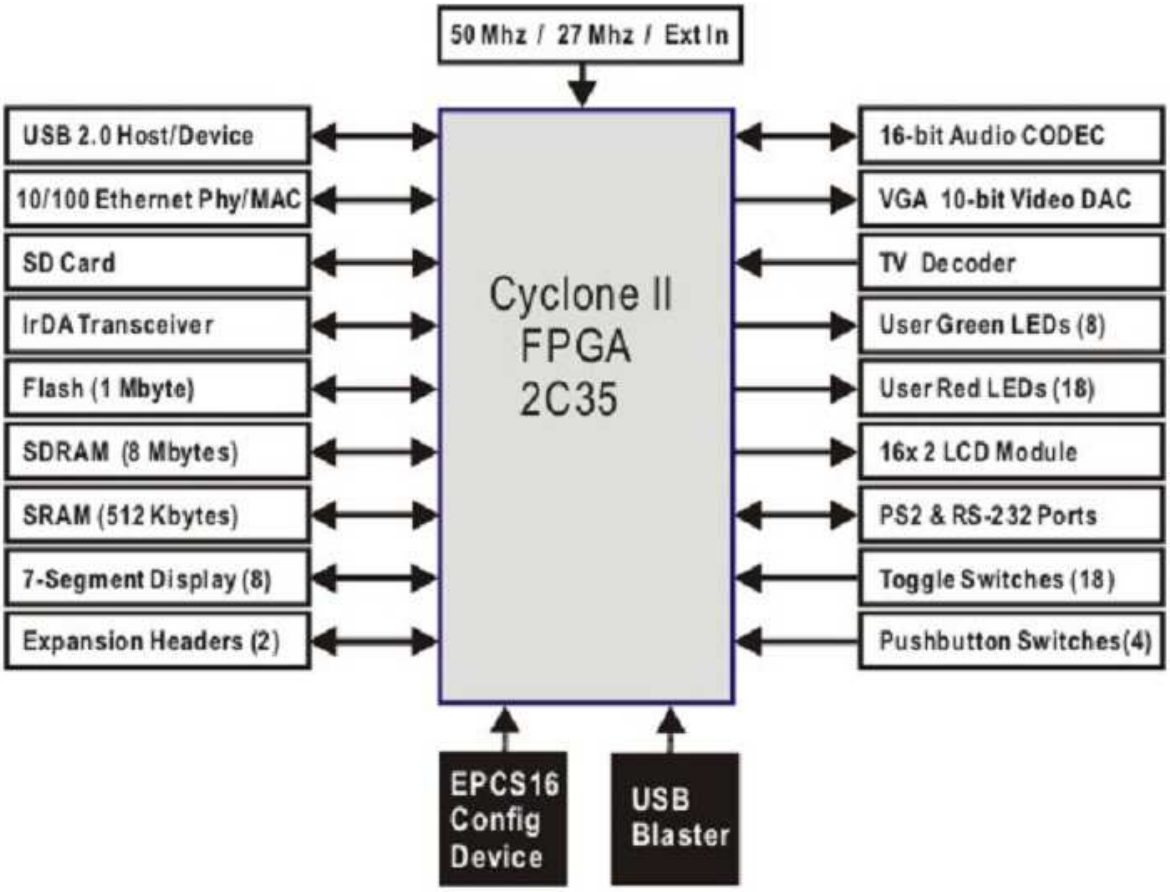


Figure II.2 Schéma bloc de la carte DE2

II.3.1 Les constituants de la carte DE2

❖ Cyclone II 2C35 FPGA

- 33,216 LEs (Logic Element)
- 105 blocs RAM M4K
- 483,840 total RAM bits
- 35 multiplicateurs
- 4 PLLs
- 475 pins I/O
- FineLine BGA 672-pin package

❖ Circuit de Configuration série et l'USB Blaster

- circuit de configuration série EPCS16 de Altera
- L USB Blaster pour la programmation et l'utilisation du contrôle API

❖ SRAM

- RAM statique de 512 Ko (256K x 16 bits)

❖ SDRAM

- de taille 8 Mo

❖ Mémoire Flash

- mémoire Flash de 4 Mo
- bus de données de 8 bits

❖ SD card socket

- Muni de mode SPI pour l'accès a la carte SD

❖ Boutons-poussoirs

- 4 boutons poussoirs
- Accompagnés de triggers de schmitt
- Normalement élevés et génère une impulsion vers le bas lorsque l'interrupteur est enfoncé

❖ Les Switchs

- 18 interrupteurs (Switchs) a utiliser comme entrees
- Un Switch est a l état 0 s il est en position vers le bas et a 1 s il vers le haut

❖ Clock inputs

- Oscillateur de 50-MHz
- Oscillateur de 27-MHz
- entrée SMA pour une horloge externe

❖ CODEC Audio

- Wolfson WM8731 24-bit sigma-delta audio CODEC
- composé de 3 prises jacks Line-level input, line-level output, and microphone input
- sa Fréquence d'échantillonnage de 8 à 96 KHz

❖ VGA output

- Utilise le ADV7123 de 240 MHz et un triple DAC vidéo (CNA) de 10-bit à grande vitesse
- Prend en charge jusqu'à une résolution de 1600 x 1200 à un taux de rafraîchissement de 100 Hz

❖ NTSC/PAL TV decoder circuit

- Utilise le ADV7181B Multi-format SDTV Video Decoder
- Intègre 3 ADCs (CANs) de 9-bits à 54-MHz
- Il a une horloge d entrée de 27-MHz
- Prend en charge des formats de sortie numériques (8-bit et 16-bit)

❖ 10/100 Ethernet controller

- Supporte les normes 100Base-T et 10Base-T en mode full-duplex
- Entièrement compatible avec la norme IEEE 802.3u

❖ USB Host/Slave controller

- Il est complètement conforme à la norme 2.0
- Prend en charge le transfert de données à pleine vitesse et à vitesse réduite
- Prend en charge à la fois l'hôte USB et le périphérique
- Prend en charge des E / S programmées (PIO) et l'accès direct à la mémoire (DMA)

❖ Ports serie

- Un port RS-232 plus un connecteur série DB-9
- Un port PS/2 plus 2 connecteurs pour connecter une souris ou un clavier PS2

❖ IrDA transceiver

- Contient un émetteur-récepteur infrarouge de 115,2 kb / s

❖ **Deux ports d'extension**

- 72 Cyclone II I/O pins, dont 8 pour l'alimentation et de masse, et 40-pins (broches) pour l'extension
- Les 40 pins acceptent le standard câble ruban de 40 pins utilisées pour les disques dur IDE

II.4 La tension d'alimentation

La carte est alimentée via un adaptateur 220V/9V, accompagnée d un bouton-poussoir ON / OFF sur la carte DE2.

II.5 Utilisation de la carte DE2

Ce titre donne des indications et instructions nécessaires pour l'utilisation de la carte DE2 et décrit chacun de ses périphériques d'E / S.

II.5.1 Configuration du FPGA Cyclone II

La procédure de ramener le circuit conçu à partir d'un ordinateur à la carte DE2 est décrite dans le tutoriel *Quartus II Introduction.* Ce tutoriel se trouve dans le dossier *DE2_tutorials* sur l' **DE2 CD-ROM Système** et également disponible sur le site d'Altera DE2, pour bien comprendre la procédure l'utilisateur devrais lire le tutoriel d'abord, et consulter le manuel d'utilisation. Nous montrons ci-dessous le principe de fonctionnement et d utilisation de la carte DE2.

Sur la carte on trouve série EEPROM qui enregistre les données de la configuration du Cyclone II FPGA. Ces dernières sont automatiquement chargées à partir de la mémoire EEPROM dans le FPGA. Avec l'utilisation du logiciel Quartus II, il est possible de reprogrammer le FPGA à tout moment, et il est également possible de modifier les données non volatiles (programme) qui sont stockées dans l'EEPROM.

Les deux types de programmation sont décrits ci-dessous :

- Programmation *JTAG:* Dans ce mode de programmation, nommé d'après les standards IEEE *Joint Test action Groupe*, le train de bits de configuration est chargé directement dans le cyclone II FPGA.

Le FPGA conservera cette configuration tant que la carte est alimentée, et la configuration est perdue des que l'alimentation est coupée.

- programmation *AS* : appelée programmation « Active Serial », Dans cette méthode, le flux de bits de configuration (programme) est chargé dans l'EEPROM « Altera EPCS16 ». Il permet le stockage non-volatile du train de bits, de sorte que l'information est conservée même lorsque l'alimentation de la carte DE2 est éteinte. Lorsque la carte est alimentée, les données de configuration dans le circuit EPCS16 sont automatiquement chargées dans le FPGA Cyclone II.

Les sections suivantes décrivent les étapes à suivre pour effectuer à la fois la programmation JTAG et AS. Pour les deux méthodes, la carte DE2 est connectée à un ordinateur via un câble USB. Avec cette connexion, la carte sera identifiée par l'ordinateur comme un périphérique « *USB Blaster* Altera ». Procédé ensuite a l'installation du pilote nécessaire de périphérique.

II.5.1.1 programmation du FPGA en mode JTAG

La figure ci-dessous illustre la configuration de la programmation JTAG.

Pour charger un flux de bits dans le Cyclone II FPGA, effectuez les étapes suivantes:

• Assurer que la carte DE2 est alimentée

• Branchez le câble USB fourni au port USB Blaster sur la carte

• configurer le circuit de programmation JTAG en définissant la RUN / PROG commuter (sur la partie gauche de le conseil d'administration) à l' RUN la position.

• Le FPGA peut maintenant être programmé à l'aide de Programmeur Quartus II, le fichier a sélectionner est d'extension «. *sof* »

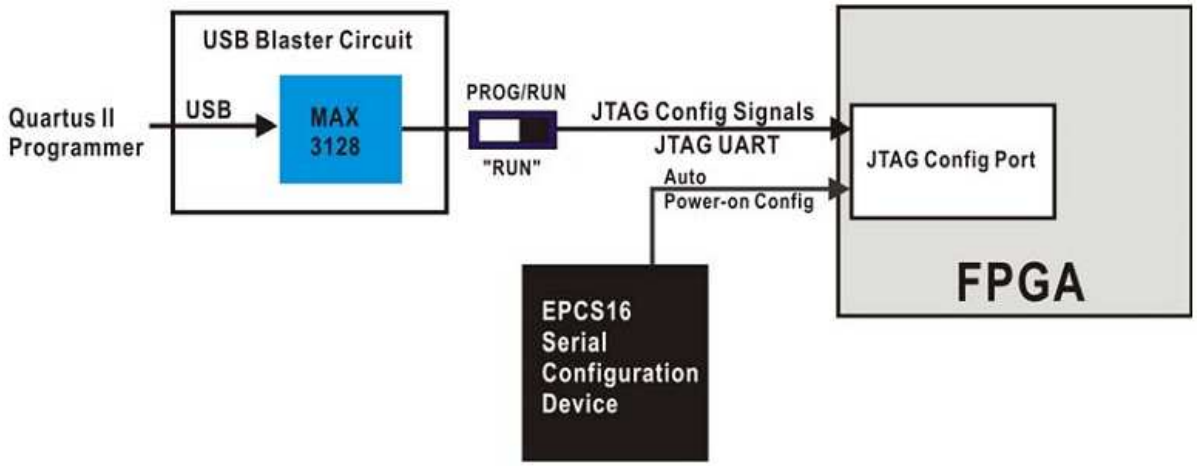


Figure II.3 Le schéma de la configuration JTAG

II.5.1.2 Programmation de l EEPROM EPCS16 en mode AS

La figure ci-dessous illustre la programmation AS mise en place. Pour charger le programme dans l'EEPROM série EPCS16, effectuez les étapes suivantes:

- Assurer que la carte DE2 est alimentée
- Branchez le câble USB fourni au port USB Blaster sur la carte
- configurer le circuit de programmation JTAG en définissant la RUN / PROG commuter (sur la partie gauche de le conseil d'administration) à l'PROG la position.
- Le circuit EPCS16 peut maintenant être programmé à l'aide de Programmeur QuartusII, sélectionnez un fichier de programmation muni de l'extension « . *pof* »
- Une fois l'opération de programmation est terminée, mettre le commutateur RUN / PROG à la position RUN, puis réinitialiser la carte sur le bouton ON/OFF. Cette action provoque la mise d'un nouveau programme dans l'EPCS16.

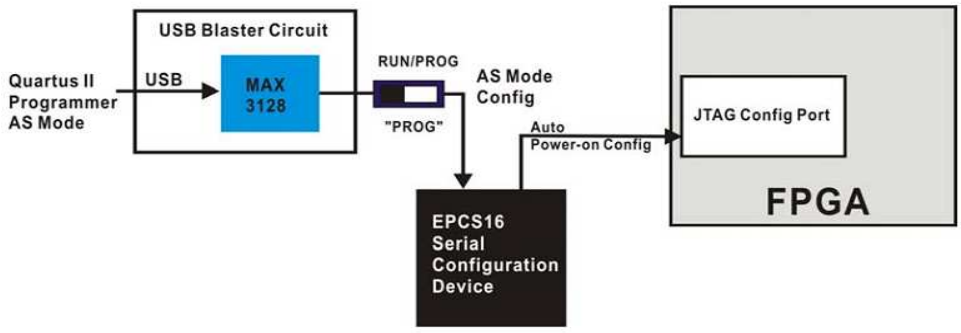


Figure II.4 Le schéma de la configuration AS

En plus de son utilisation pour la programmation, le port USB Blaster de la carte DE2 peut également être utilisé pour contrôler certaines fonctions de la carte à distance depuis un ordinateur.

II.5.2 Utilisation des boutons-poussoirs et des Switchs

La carte DE2 fournit quatre boutons-poussoirs. Chacun des boutons est anti-rebonds en utilisant un trigger de Schmitt, comme c'est indiqué à la figure I.5, les quatre sorties des triggers sont *KEY0, KEY1, KEY2, KEY3,* elles sont reliées directement aux pins de l'FPGA. Chaque bouton fournit un niveau logique haut de 3,3 volts quand il n'est pas enfoncé, et un niveau logique bas de 0 volt dans le cas contraire.

Étant donné que les boutons poussoirs sont anti-rebond, elles sont appropriées pour une utilisation comme horloge ou réinitialiser les entrées d'un circuit.

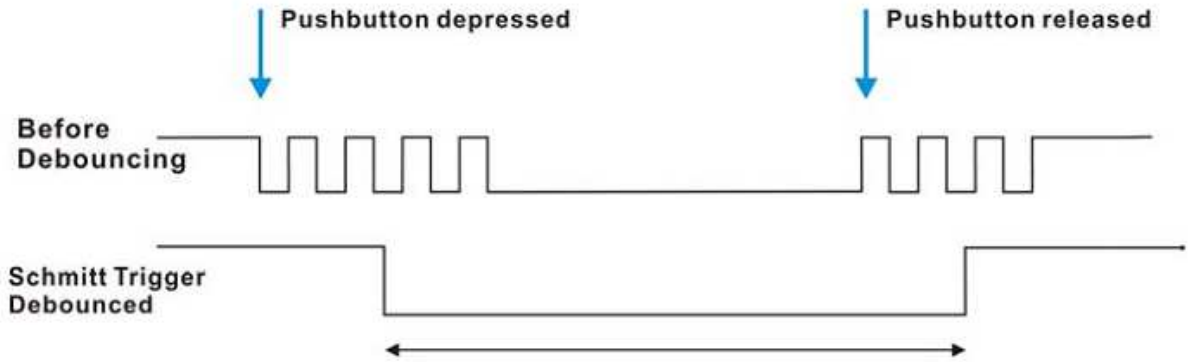


Figure II.5 Représentation du trigger anti rebondissement

La liste des noms des pins du l'FPGA qui sont connectés aux boutons-poussoirs est donnée dans le Tableau 2.1 dans l'annexe 2.

Il ya aussi 18 Switchs sur la carte DE2. Ils ne sont pas anti-rebonds, sont destinés à être utilisés comme des entrées de données d'un circuit. Chacun d'eux est connecté directement à une broche de l'FPGA Cyclone II. Lorsqu'un interrupteur est en position vers le bas, il fournit un niveau logique bas (0 volt) pour le FPGA, et lorsqu'il est dans la position haut, fournit un niveau logique haut (3,3 volts).

La figure ci-dessous montre les circuits électriques des boutons-poussoirs et des Switchs.

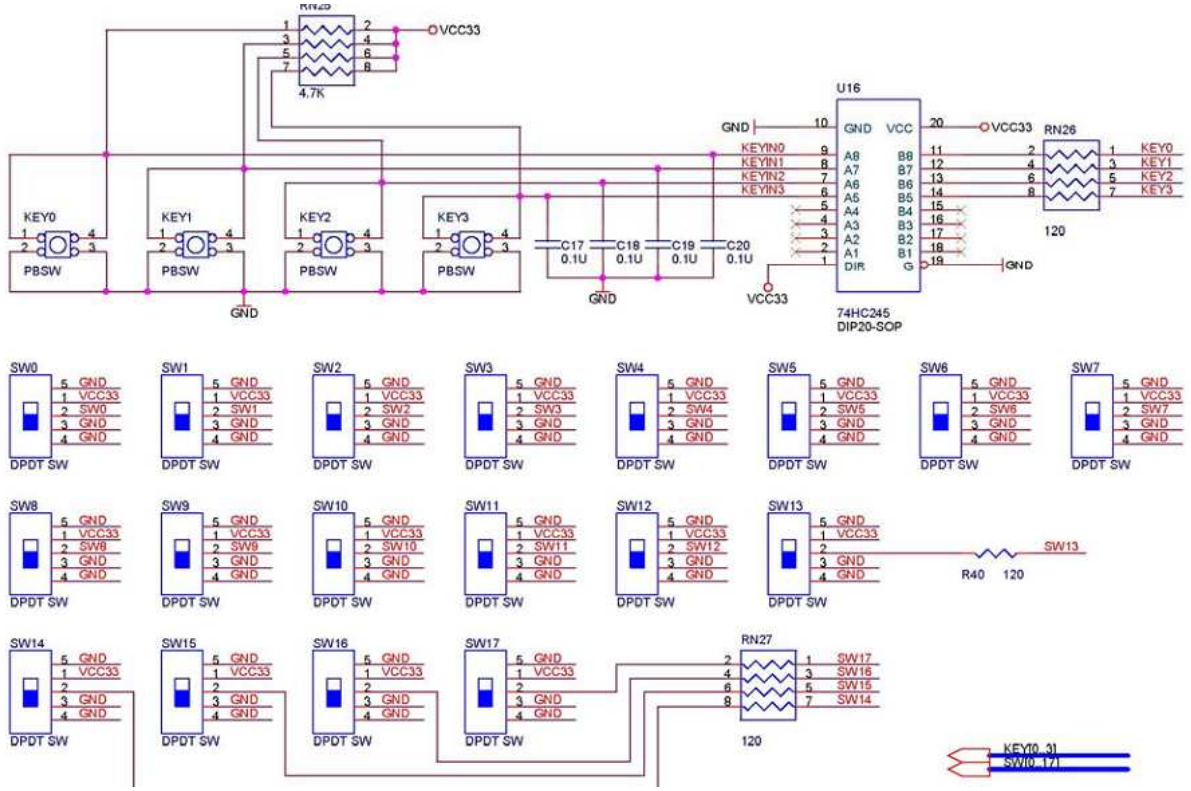


Figure II.6 schéma électrique des Switchs et des boutons poussoirs

La liste des noms des broches du l'FPGA qui sont connectés aux Switchs est donnée dans le Tableau 2.2 dans l'annexe 2.

II.5.3 Utilisation des LEDs

La carte dispose de 27 LEDs contrôlables par l'utilisateur, dont 18 rouges sont situés au-dessus de la 18 Switchs, et 8 vertes se trouvent au-dessus des boutons-poussoirs, la 9eme est verte située au milieu des afficheurs 7 segments. Chaque LED est connectée directement à une broche sur l'FPGA.

Les circuits électriques des LEDs sont donnes dans la figure suivante

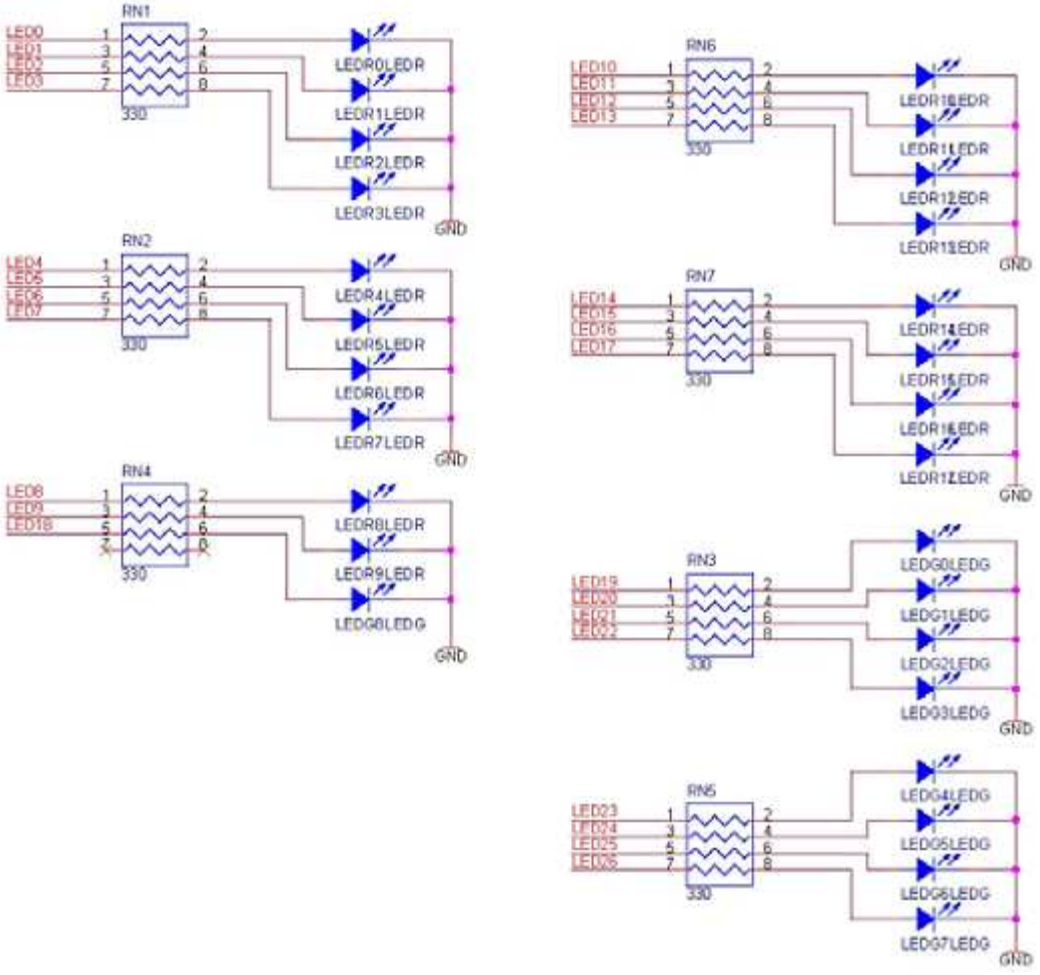


Figure II.7 schéma électrique des LEDs

La liste des noms des pins du l'FPGA qui sont connectés LEDs est donnée dans le Tableau 2.3 dans l'annexe 2.

II.5.4 Utilisation des afficheurs 7 segments

La carte DE2 a huit afficheurs 7 segments. Ces derniers sont disposés en deux paires et un groupe de quatre, avec l'intention d'afficher des numéros de différentes tailles.
L'application d'un niveau logique bas à un segment amène à éclairer, et l'application d'un niveau logique haut l éteint (les afficheurs sont a anodes communes). Chaque segment d'un afficheur est identifié par un index de 0 à 6.

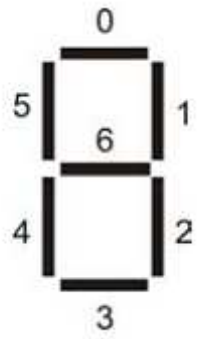


Figure II.8 Position et index de chaque segment d'un afficheur 7 segments

Et les schémas de la figure I.9 nous montre comment que les afficheurs 7 segments sont reliés aux broches de l'FPGA, et on voie bien que le point dans chaque afficheur n'est pas connecté donc n'est pas utilisable.

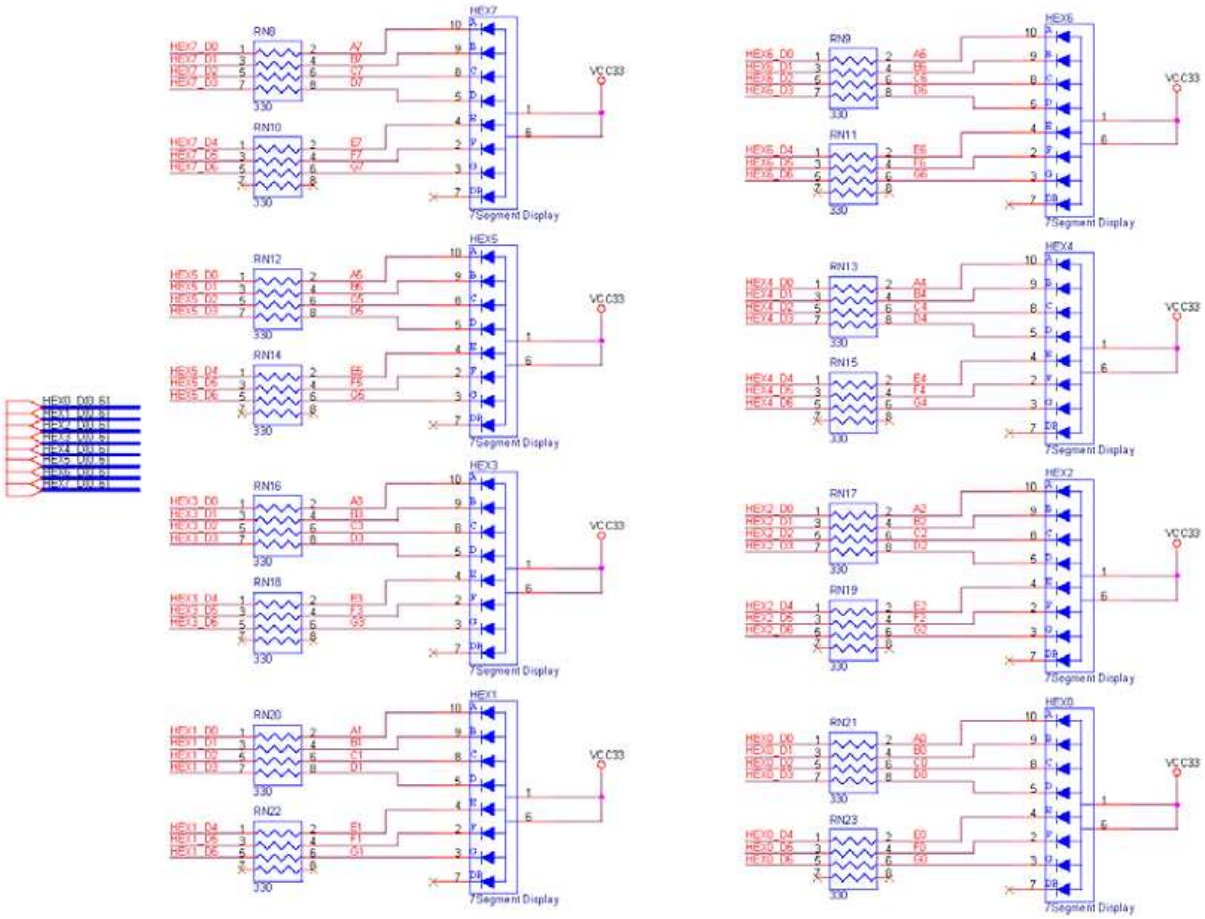


Figure II.9 Brochage des afficheurs 7 segments

NB: Le Tableau 2.4 dans l annexe 2 montre les affectations des broches du FPGA vers les afficheurs 7 segments.

II.5.5 Les horloges d'entrées

La carte DE2 comprend deux oscillateurs de 50 MHz et 27 MHz, et comprend également un connecteur SMA utilisable lors que la source d'horloge est externe.

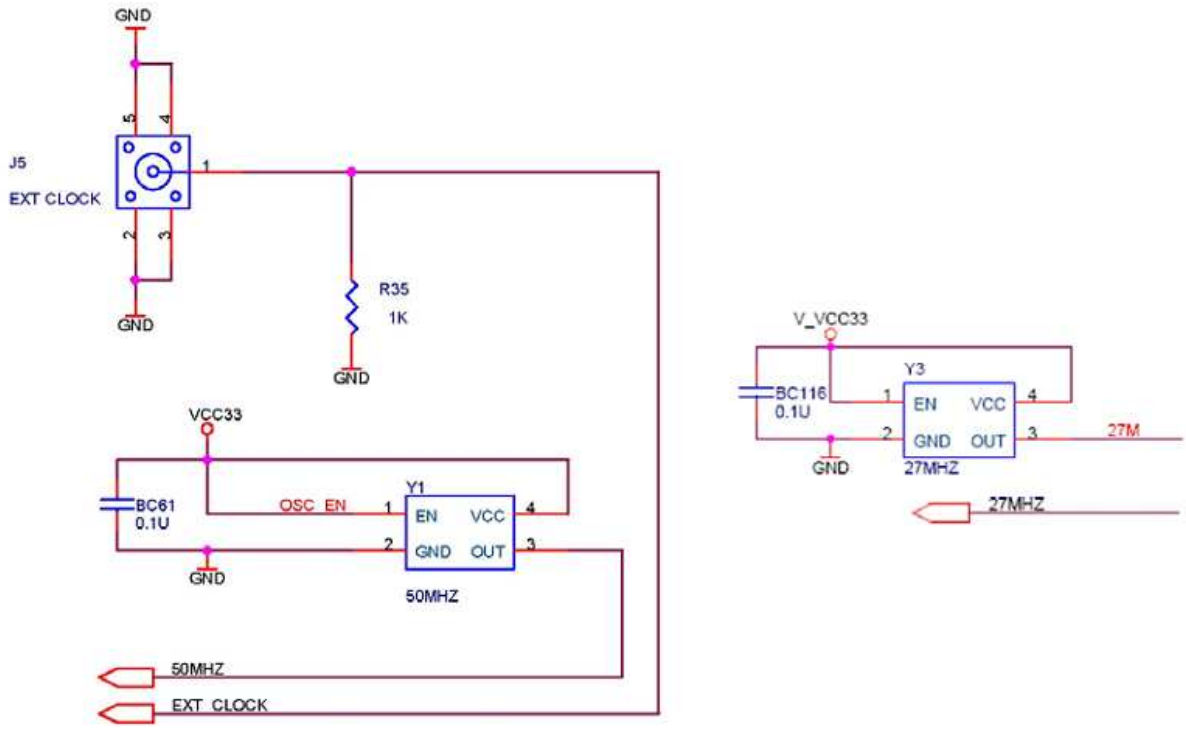


Figure II.10 Schéma du circuit d'horloge

II.5.6 Utilisation de l'afficheur LCD

Le module LCD est utilisé pour afficher du texte en envoyant des commandes appropriées au contrôleur d'affichage (HD44780). On trouve des informations détaillées sur le datasheet du circuit.

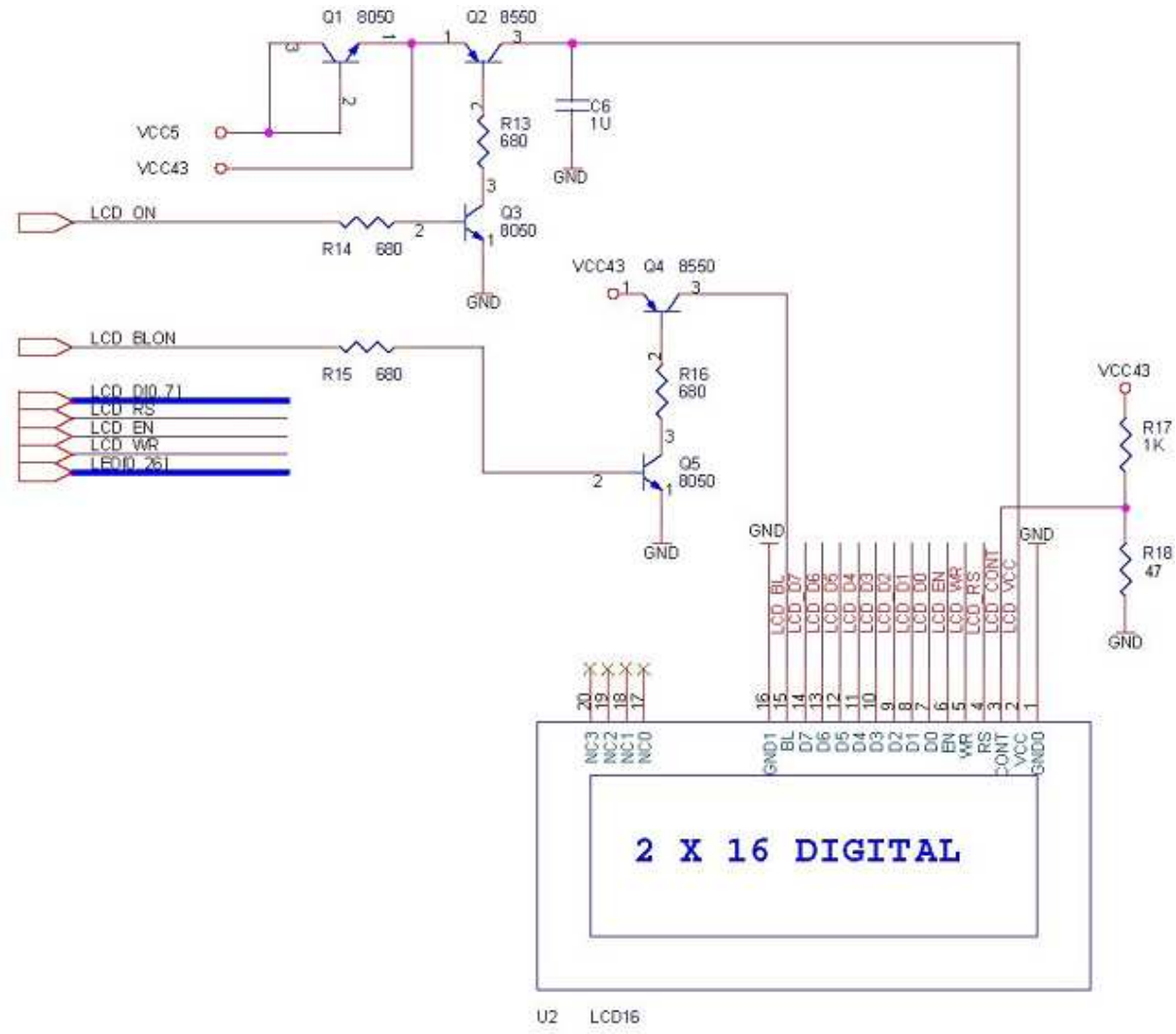


Figure II.11 Schéma de principe du module LCD

II.5.7 Utilisation des ports d'extension

La DE2 offre deux ports (connecteurs) d'extension de 40 broches. Chaque port se connecte directement à 36 pins de l' FPGA, et fournit également un DC 5 V (VCC5), VCC 3,3 (VCC33), et deux broches GND.

La figure ci-dessous montre les schémas associés. Chaque broche sur les ports d'extension est connectée à deux diodes et une résistance qui offrent une protection de haute et basse tensions. La figure montre Le circuit de protection pour seulement 4 broches sur chaque port, mais ce circuit accompagne 72 pins sur les 2 ports.

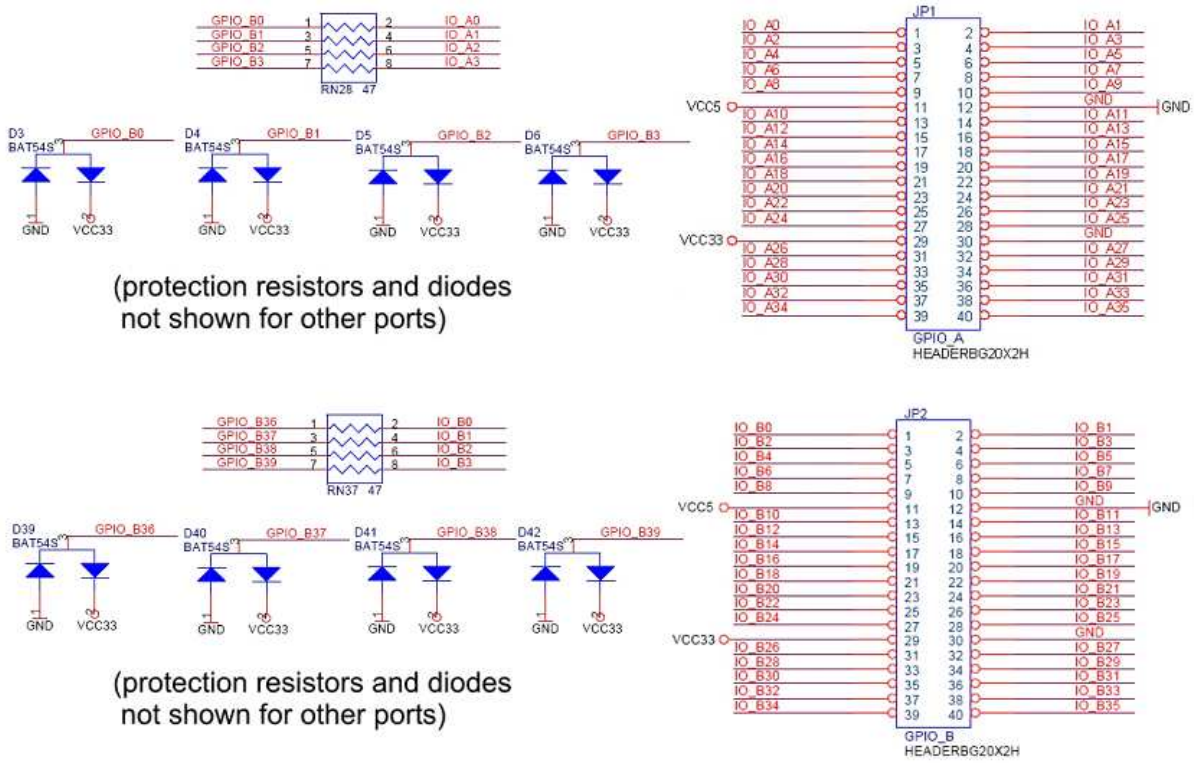
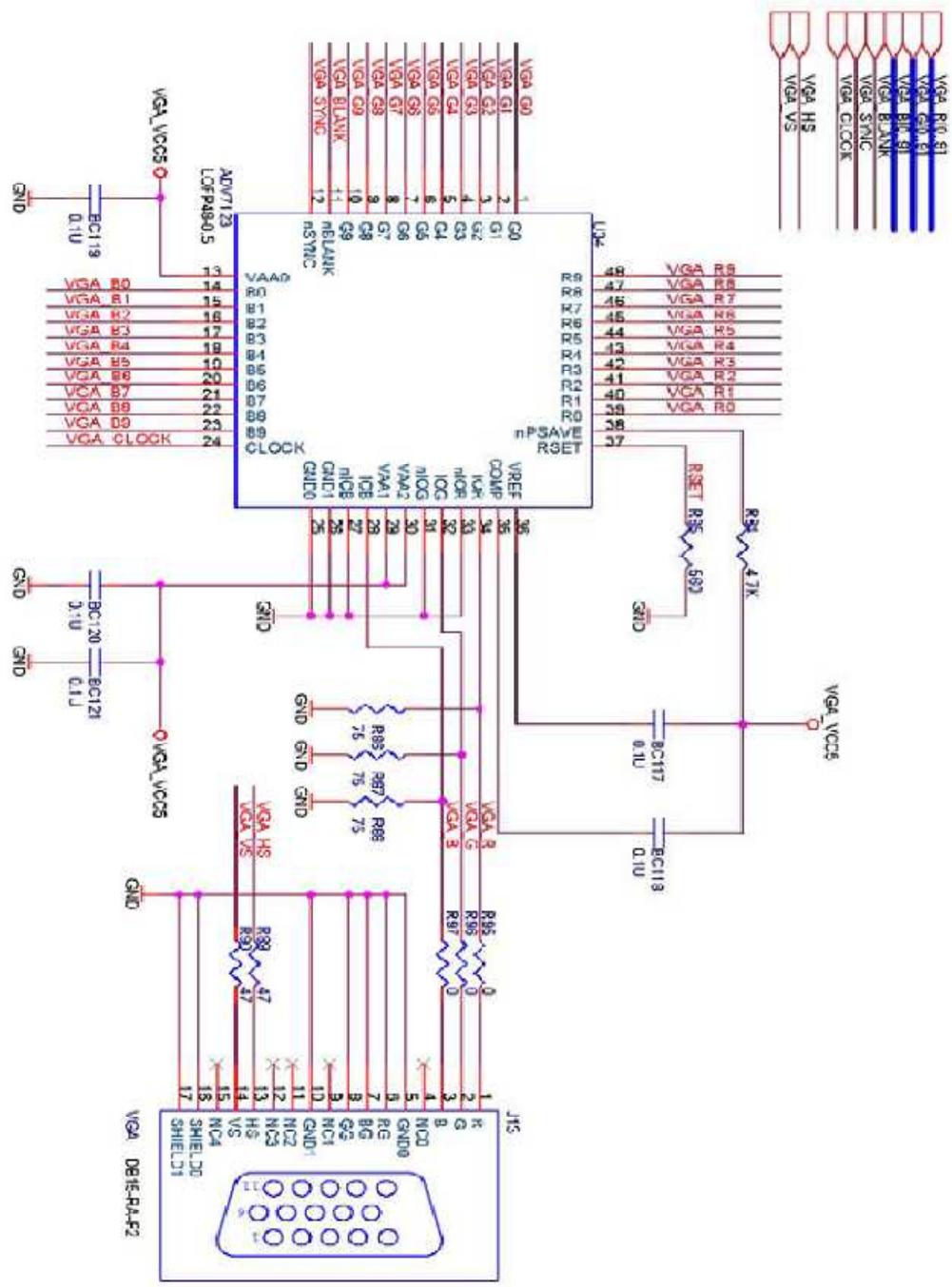


Figure II.12 schéma électrique des ports d'extensions

II.5.8 Utilisation du port VGA

La carte comprend un connecteur D-SUB 16 broches pour la sortie VGA. La synchronisation des signaux VGA se fait à partir du FPGA, et ADV7123 triple DAC 10 bits est utilisé pour convertir les signaux de données en analogique (rouge, vert, bleu). Il peut prendre en charge une résolution allant jusqu'à 1600 x 1200 pixels, à 100 MHz.



Figue II.13 schéma et circuit du port VGA

II.5.9 Utilisation du codec audio 24-bit

La carte DE2 possède (WM8731) Encodeur / décodeur audio de 24 bits haute qualité au port microphone, line-in et line-out, travaillant dans la plage 8 kHz à 96 kHz. Le WM8731 est commandé par une interface de bus série I2C, qui est le relie à l'FPGA.

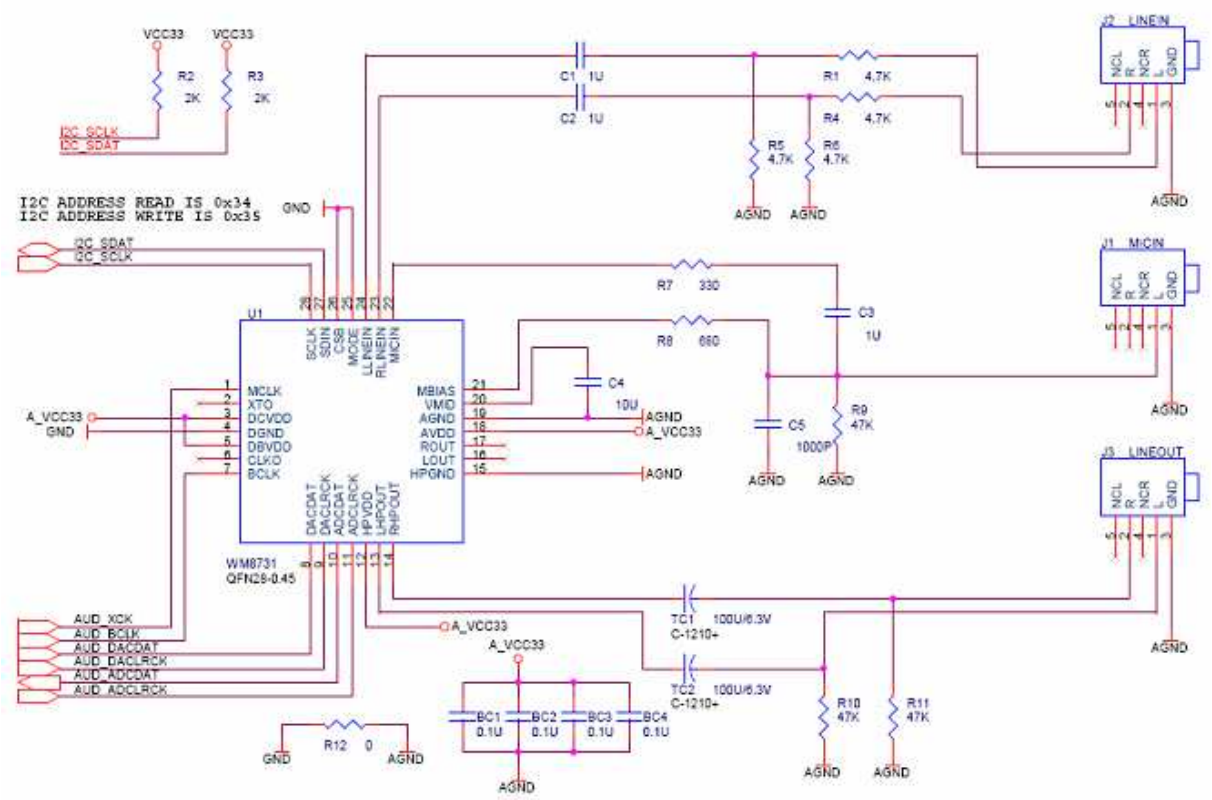


Figure II.14 schéma éclectique du codec audio

II.5.10 Le port série RS-232

La DE2 utilise le MAX232 et un connecteur SUB-D de 9 broches pour la communication RS-232.

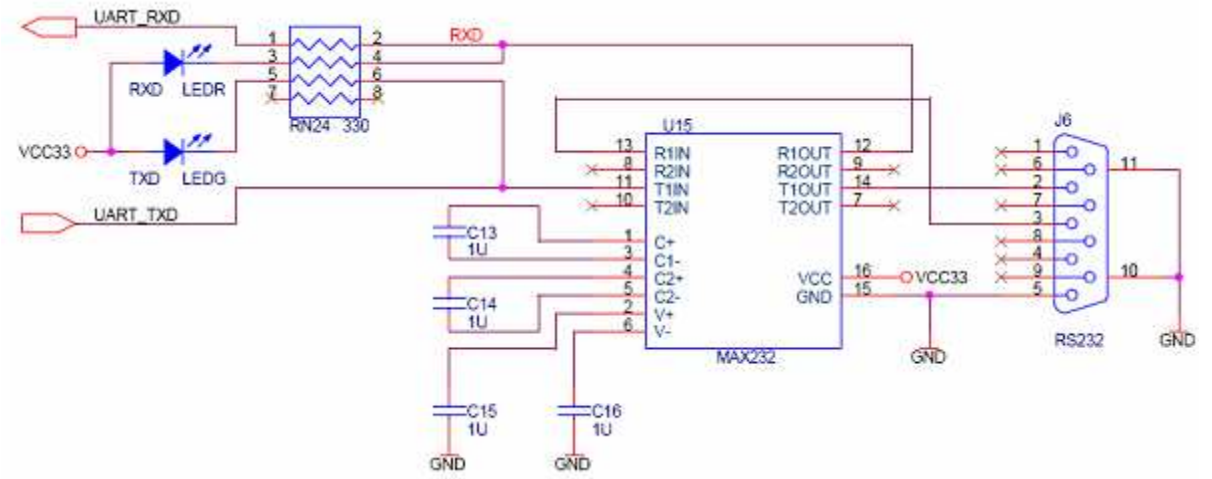


Figure II.15 schéma et circuit du MAX232 et RS-232

II.5.11 Le port série PS / 2

La carte comprend une interface standard PS / 2 et un connecteur pour clavier ou souris PS / 2.

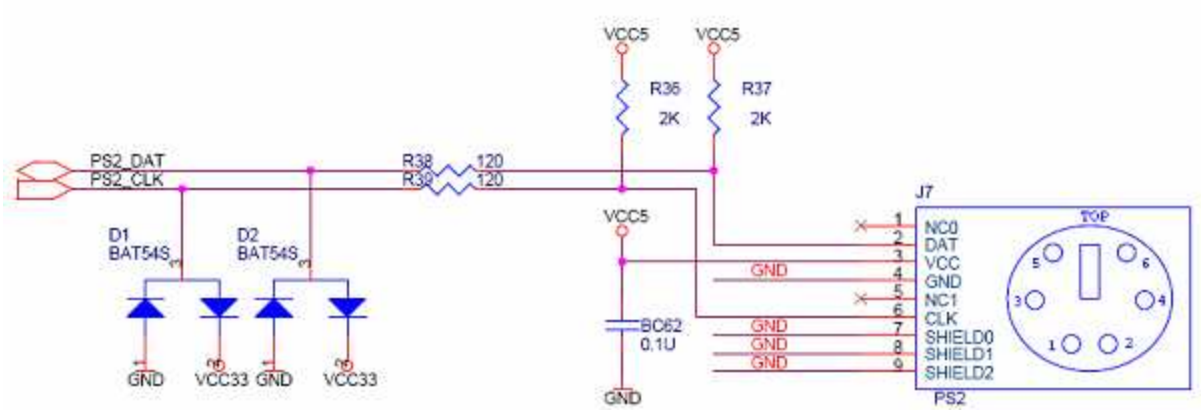


Figure II.16 schéma et circuit du PS/2

II.5.12 Contrôleur Fast Ethernet

La carte DE2 est dotée d'un contrôleur Fast Ethernet le Davicom. DM9000A, ce dernier comprend l'interface générale du processeur, une SRAM de 16 Ko, une unité de contrôle d'accès au support (MAC), et un émetteur-récepteur PHY 10/100M.

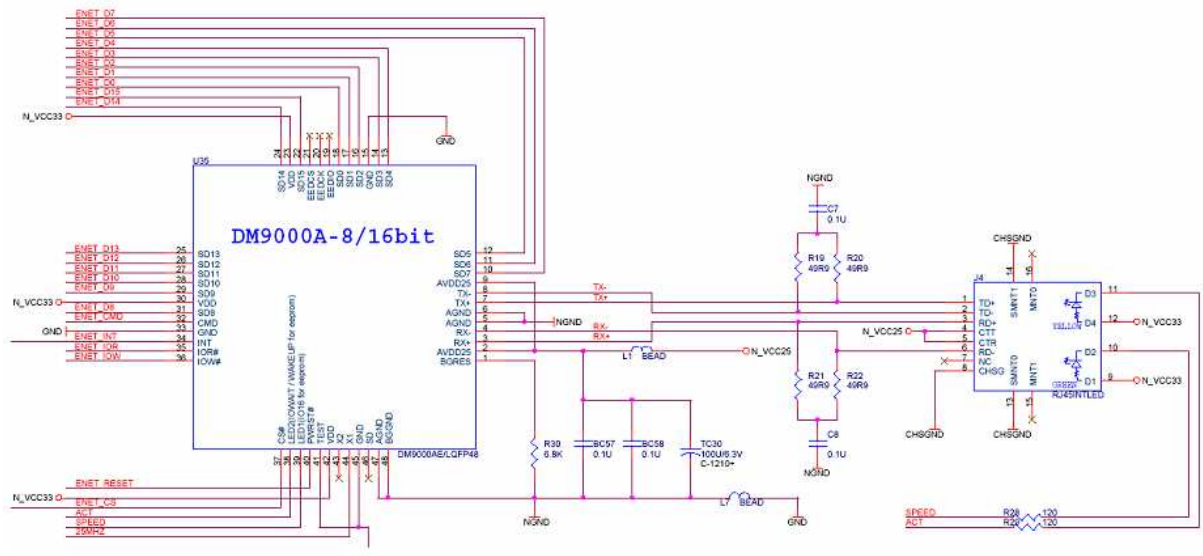


Figure II.17 schema du module Fast Ethernet

II.5.13 Le décodeur TV

La carte DE2 est équipée du circuit décodeur TV le ADV7181,c'est un décodeur vidéo intégré qui détecte et convertit un signal télévision en bande de base standard analogique (NTSC, PAL et SECAM) en 4:02:02 données vidéo en composantes compatibles avec 16-bit/8-bit (CCIR601/CCIR656). Le ADV7181 est compatible avec une large gamme d'appareils vidéo, comme les lecteurs de DVD, sources sur bande, sources de diffusion, et les caméras de surveillance.

Les registres du décodeur TV peuvent être programmés par un bus I2C, qui le relie à l'FPGA.

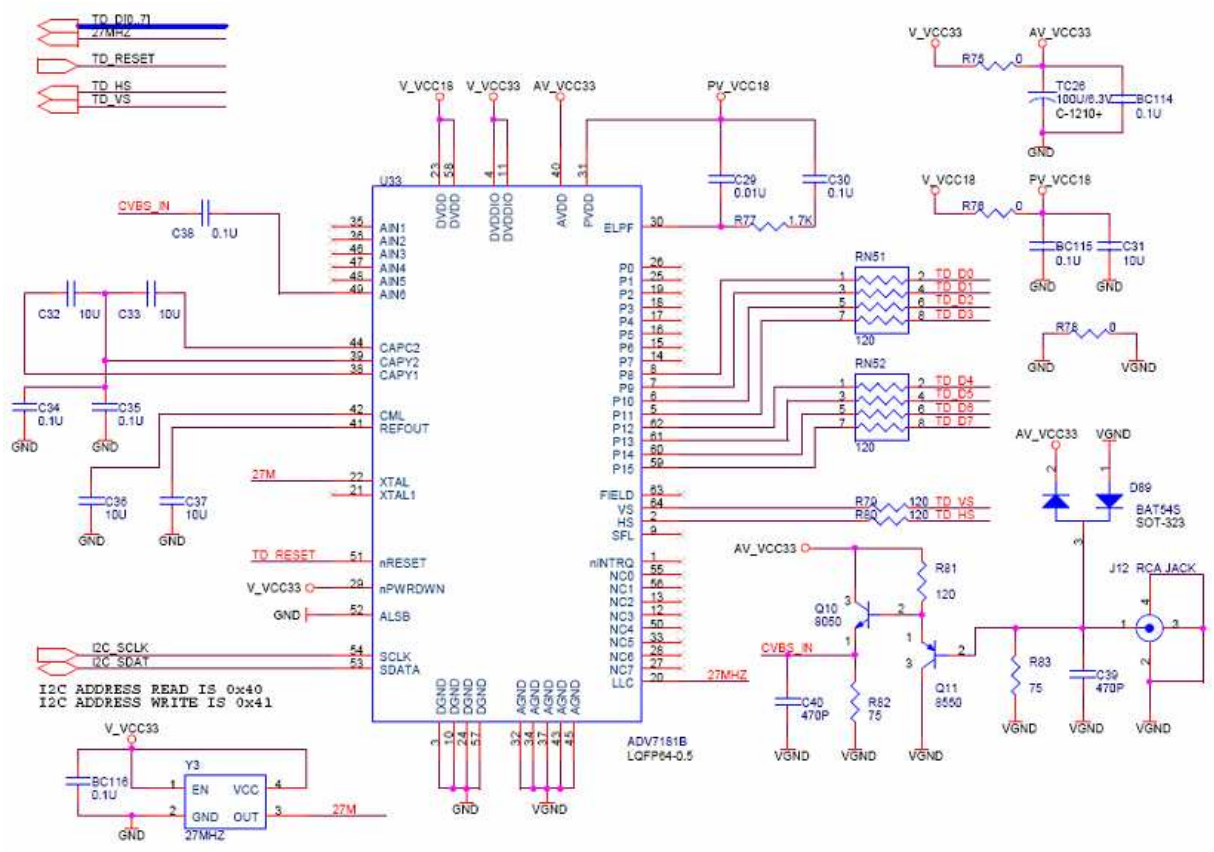


Figure II.18 schéma du décodeur TV

II.5.14 La mise en œuvre d'un encodeur TV

Bien que la carte DE2 ne comprend pas un module d'encodage TV, le ADV7123 (10-bit high-speed triple ADCs) peut être utilisé comme un encodeur TV de qualité professionnelle avec la partie de traitement numérique dans le FPGA.

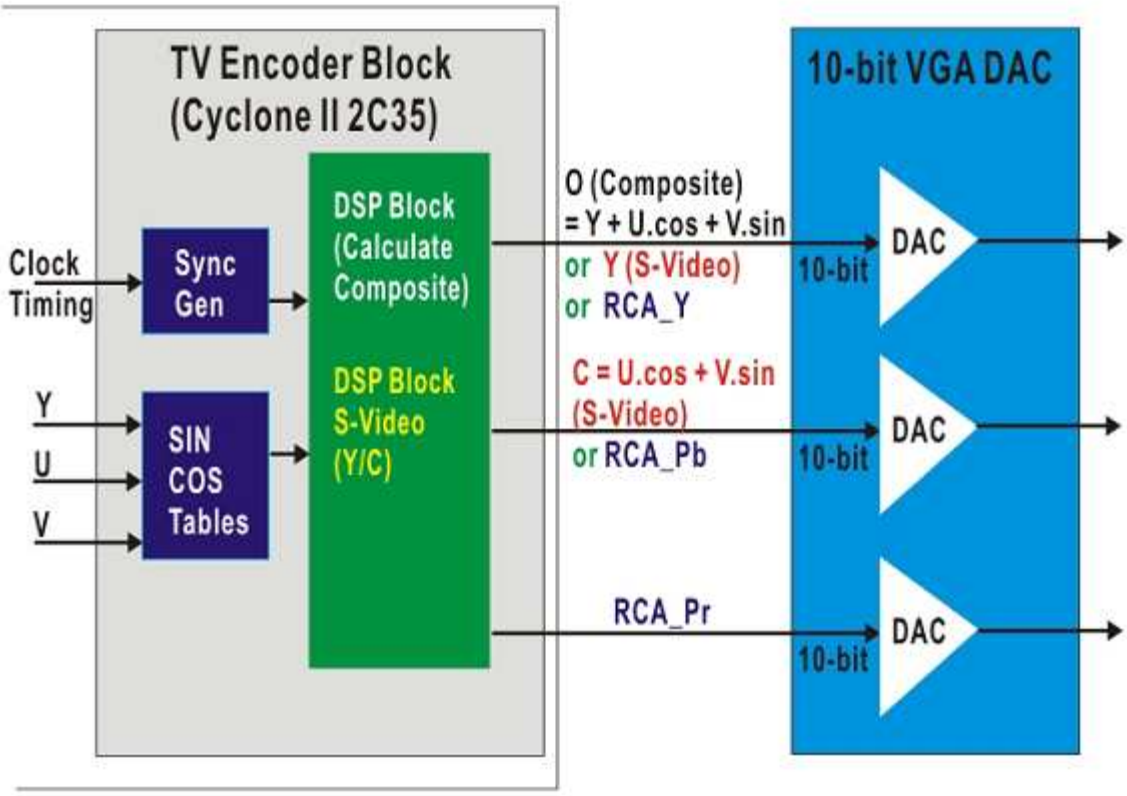


Figure II.19 L'encodeur TV qu'utilise le FPGA Cyclone II (ADV7123)

II.5.15 Utilisation de l'USB Host (pilote) et Périphérique

La carte DE2 met à disposition à la fois l'hôte USB et les circuits d'interfaces en utilisant le Philips ISP1362. Les contrôleurs hôte (maitre) et les contrôleurs USB sont conformes à l'Universal Serial Bus Specification Rev. 2.0, soutenant du transfert de données à pleine vitesse (12 Mbit / s) et basse vitesse (1,5 Mbit / s).

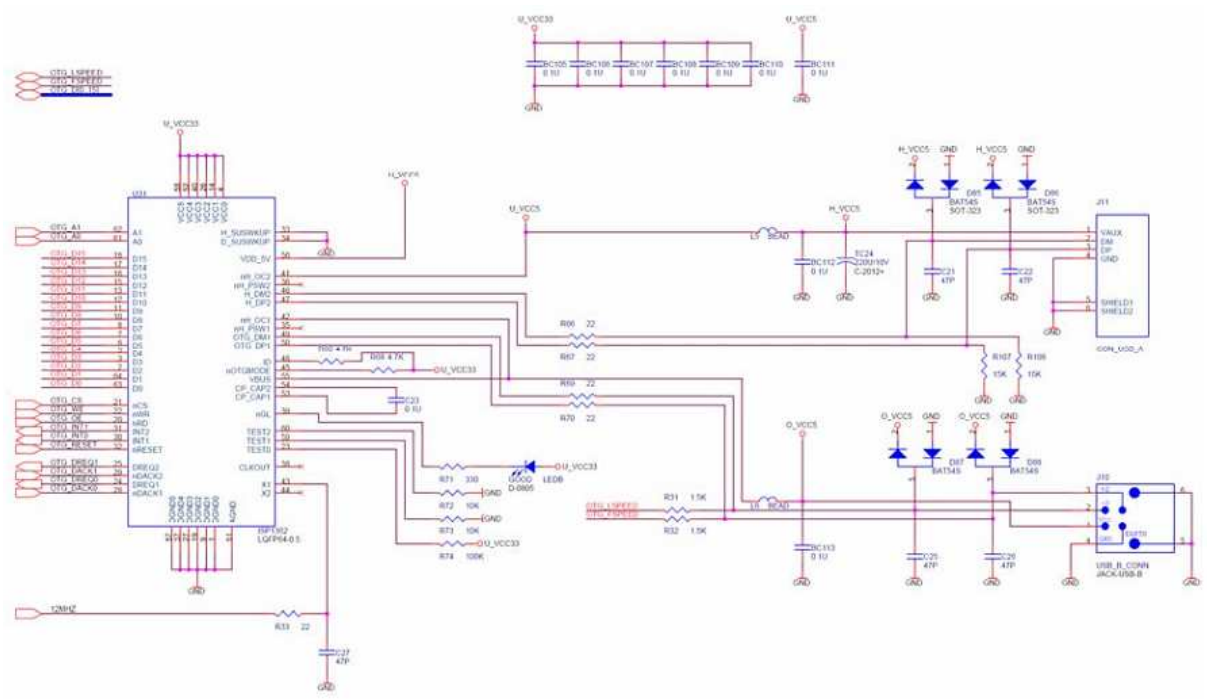


Figure II.20 schéma du host et périphérique USB

II.5.16 L'utilisation du IrDA

La carte DE2 fournit un moyen de communication sans fil simple en utilisant un émetteur-récepteur infrarouge l'**Agilent HSDL-3201**.

Notez que le taux de transmission avec une vitesse max est de 115.2 Kbit / s. La figure ci-dessous montre le schéma de la liaison de la communication IrDA.

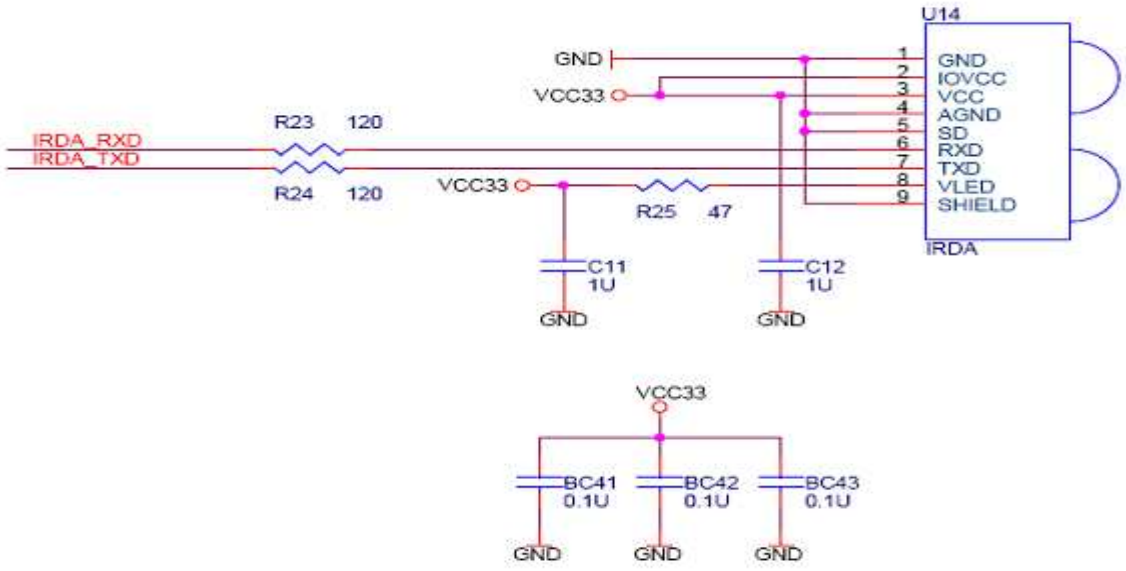


Figure II.21 Schéma du IrDA

II.5.17 Utilisation des RAM flash SDRAM / SRAM

La carte DE2 fournit une mémoire SDRAM de 8 Mo, SRAM de 512 Ko. Les figures suivantes montrent les schémas des puces mémoire.

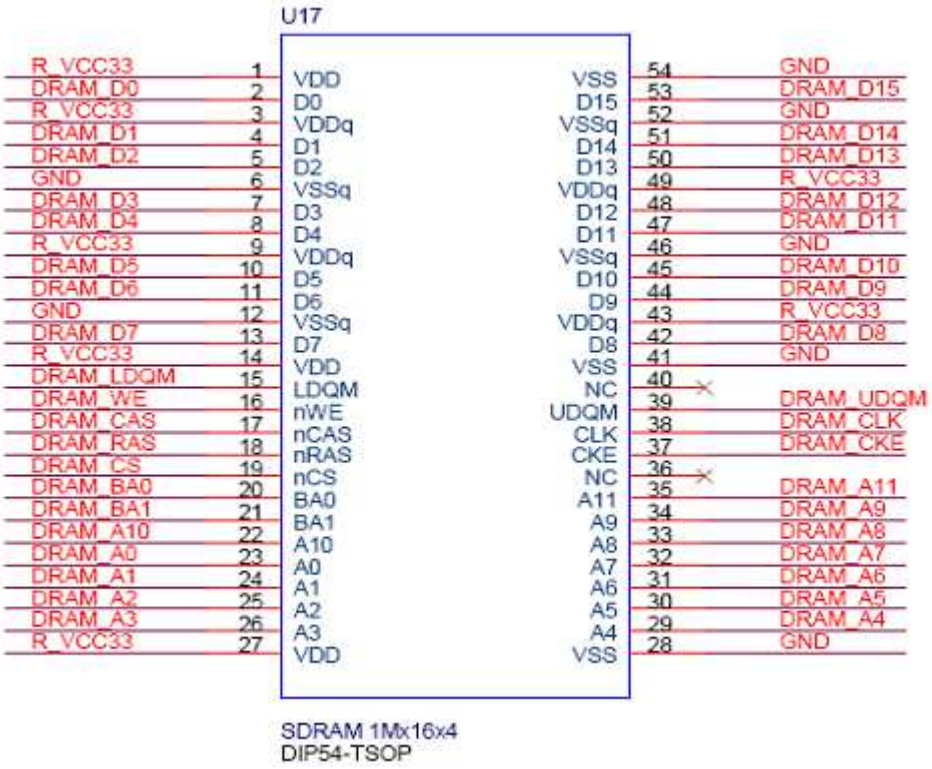


Figure II.22 Le bloc SDRAM

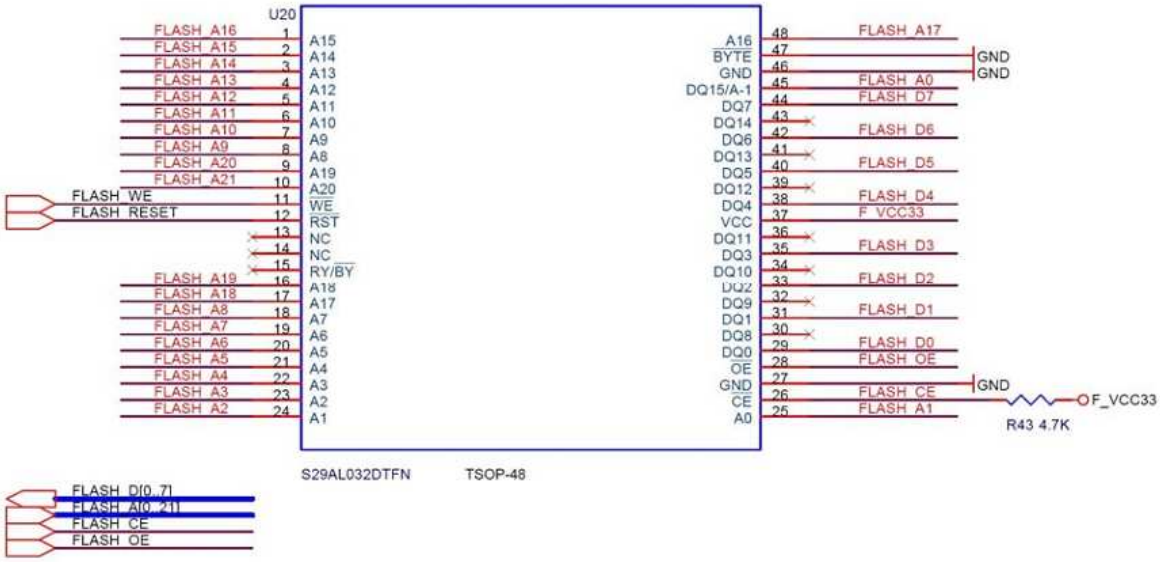


Figure II.23 Le bloc de la mémoire FLASH

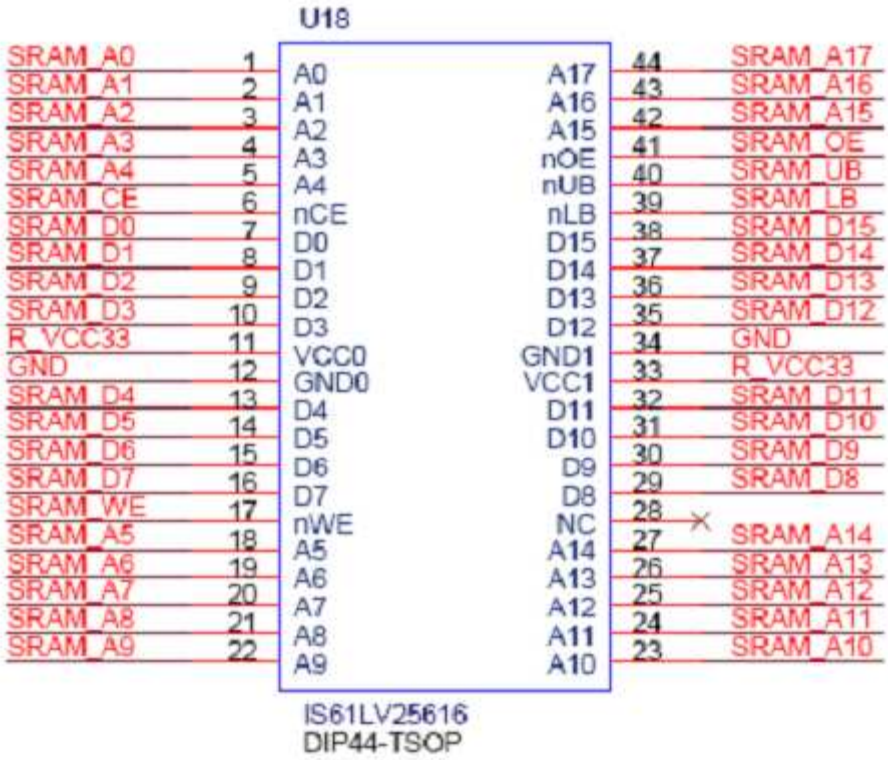


Figure II.24 Le bloc SRAM

II.6 Conclusion

Ce chapitre a été dédié à la présentation de la carte DE2 en expliquant les différents éléments constituant et ses diverses utilisations que nous exploiterons dans le chapitre suivant.

Chapitre III

III.1 Introduction

Pour poursuivre un objet en mouvement on dispose d'un moteur pas a pas sur lequel est fixé un capteur CCD; et une carte DE2 sur laquelle nous allons implémenter un programme de commande et de poursuite sous la plateforme de développement Quartus.

III.2 Capteur CCD ou CMOS

L'appareil photo ou le scanner sont dotés de capteurs traduisant des rayons lumineux colorés en signaux numériques. Ce signal est enregistré en valeurs mathématiques plus ou moins complexes donc plus ou moins "lourdes". Ce capteur est composé de millions de photosites, chaque photosite étant muni d'un filtre R, V ou B pour enregistrer un signal exclusivement R, V ou B, le photosite vert étant deux fois moins sensible sera doublé, il faut donc 1 photosite R, 1 B et 2V pour obtenir un pixel après traitement dans le logiciel de l'appareil. Pour des raisons pratiques de fabrication les photosites actuels sont de forme carrée, cela devrait changer dans le futur, on voit apparaître des photosites hexagonaux qui devraient réduire l'effet d'escalier (pixellisation) sur les agrandissements et permettre ainsi une augmentation de la qualité.

III.3 Généralités sur l'image

III.3.1 Le Pixel

Un pixel a une couleur exprimée (codée) en langage binaire mathématique (uniquement des 0 et des 1). L'œil humain est capable de discerner aux environs de 220 nuances dans une couleur. On comprend donc tout de suite qu'il faut cette quantité minimum de nuances pour avoir une qualité correcte, appelée parfois qualité photo. Ce seront donc toujours des multiples de 2 en progression géométrique qui donneront les valeurs de travail 2, 2^2, 2^4, 2^8, 2^{12}, 2^{16} .....

C'est 2^8 soit 256 qui se rapproche le plus des 220 niveaux requis pour une bonne séparation des nuances, on comprend donc tout de suite pourquoi on retrouve la majorité des APN réglés sur 256 niveaux de nuances par couleur.

III.3.2 L'image

III.3.2.1Définition

Une image est avant tout un signal 2D (x,y), cette image représente une réalité 3D(x,y ,z). D'un point de vue mathématique une image est une matrice de nombres représentant un

signal une image contient plusieurs informations sémantiques il faut interpréter le contenu au-delà de la valeur des nombres.

Une image matricielle est formée d'un tableau de points ou pixels.

Plus la densité des points est élevée, plus le nombre d'informations est grand et plus la résolution de l'image est élevée.

Corrélativement la place occupée en mémoire et la durée de traitement sera d'autant plus grandes. Les images vues sur un écran de télévision ou une photographie sont des images matricielles. On obtient également des images matricielles à l'aide d'un appareil photo numérique, d'une caméra vidéo numérique ou d'un scanner.

III.3.2.2 Types d'images

Images en niveaux de gris I(x,y) Î [0..255]

Images binaires I(x,y) Î {0 , 1}

Images couleurs IR(x,y) IG(x,y) IB(x,y)

III.3.2.3 Représentation des images

Une image est une matrice de dimension $M * N$, et chaque élément à une valeur entière dans l'intervalle [$Lmin$, $Lmax$]. Le nombre de « bits » requis pour représenter les niveaux de gris dans l'intervalle « L » est « K » tels que: $L = 2^k$, Le nombre de bits pour entreposer une image est donc : $b = M * N * K$.

III.3.2.4 L'image RGB (RVB)

L'image est obtenue par superposition de trois rayonnements lumineux, le rouge (R), le vert (V) et le bleu (B). Une image RVB est composée de la somme de trois rayonnements lumineux rouge, vert, et bleu dont les faisceaux sont superposés. A l'intensité maximale ils produisent un rai de lumière blanche, et à l'extinction une zone aussi noire que l'éclairage ambiant le permet (c'est la raison pour laquelle vous avez installé le moniteur de votre ordinateur dans une pièce où règne, autant que faire se peut une certaine pénombre, et qu'un rayon de soleil qui frappe en plein un écran le rend presque illisible).

Si vous mettez le nez sur l'écran de votre moniteur (ou mieux encore sur celui de votre téléviseur, dont le pas est plus grossier) vous distinguerez les trois sources RVB en forme de nid d'abeille, ou de grille, qui donnent l'illusion d'un fond blanc.

En fait, croyant voir un point blanc, vous percevez simultanément trois lumières rouge, vert et bleu. La gamme des couleurs reproductibles par ce mode, quoique conditionnée par la qualité du matériel employé, est très étendue, et reproduit bien les couleurs saturées.
En contrepartie, elle convient mal à la restitution des nuances délicates des lumières intenses et des tons pastel.

L'image est codée en *8 bits par couche* (2^8 = 256 niveaux) soit 8 pour le R, 8 pour le V, 8 pour le B au total *24 bits en RVB.*
Remarque : Il existe aussi d'autres codages de la couleur que RVB…

Figure III.1 montre les 3 couleurs constitutives de limage

III.4 La plateforme de développement Quartus

III.4.1 Présentation

Les fabricants de circuit fpga tels Xilinx ou Altera offrent des environnements de développement qui constituent une excellente base de travail. De sa part, Altera propose l'outil de conception Quartus II.

III.4.1.2 L'environnement Quartus II

L'environnement de Quartus II est le suivant :

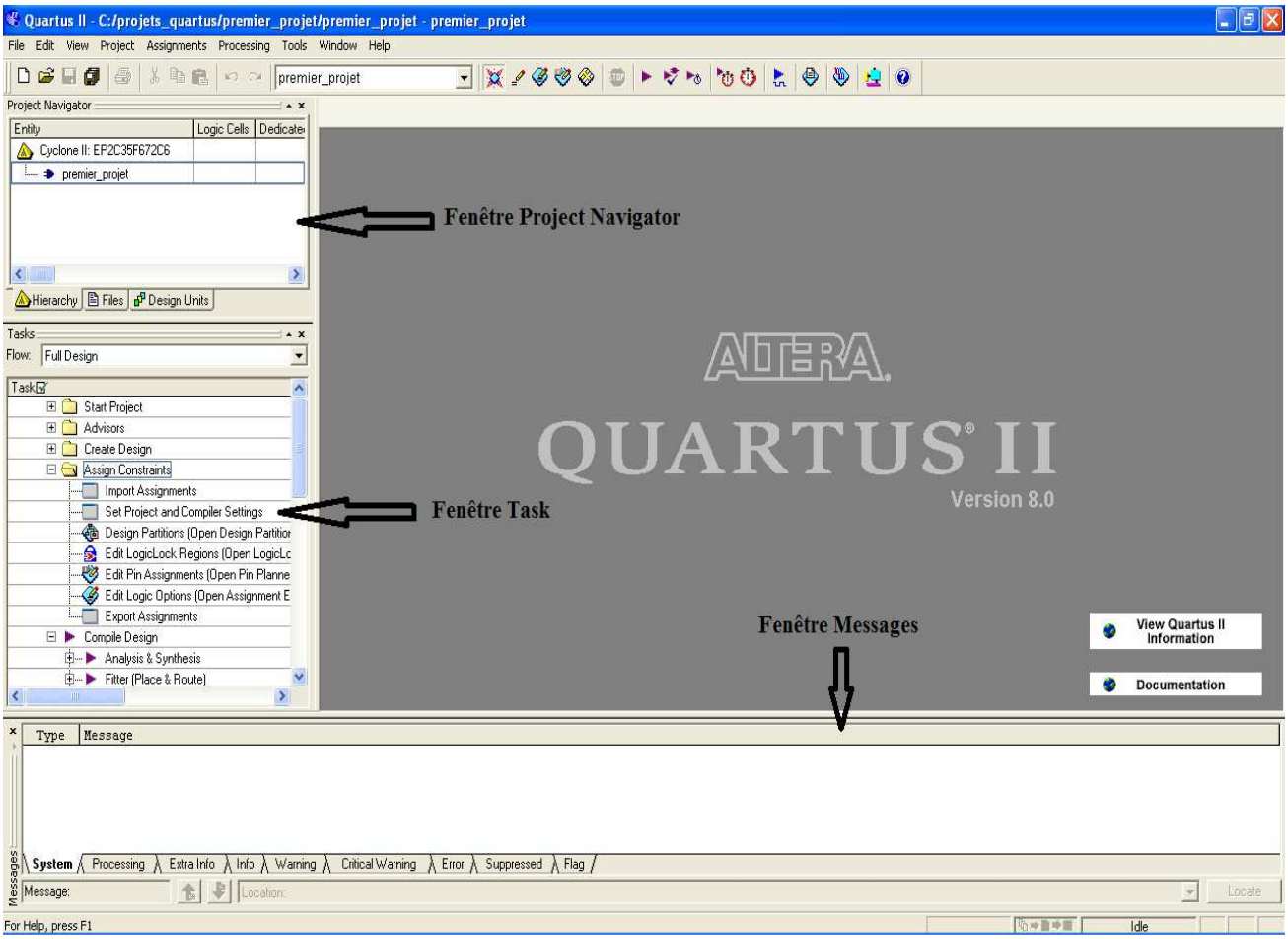


Figure III.2 l'environnement de développement Quartus II

L'environnement se découpe en 4 parties, une pour l'éditeur de code qui apparait après avoir créé un nouveau projet, 3 sont visibles après le lancement de Quartus.

- **Une fenêtre Project Navigator :** qui contient toutes les informations relatives au projet dont les détails sur la hiérarchie du projet, les fichiers le constituant, les différentes unités définies (entités, architecture, schémas, machines d'états ...).
- **La fenêtre Tasks :** présente les différentes tâches du Design Flow avec la possibilité d'accéder à l'ensemble des processus de traitement de Quartus et les comptes rendus associés.
- **La fenêtre Messages :** permet d'être informé en continu des informations, avertissements et erreurs apparaissant lors de l'exécution des diverses tâches.

III.4.1.3 Développement sous Quartus II

❖ Création d'un projet sous Quartus

Après avoir démarrer Quartus , on crée un nouveau projet **: *File -> New Project...***

- La première boite de dialogue qui apparait permet de fixer le nom du projet et son lieu de stockage :

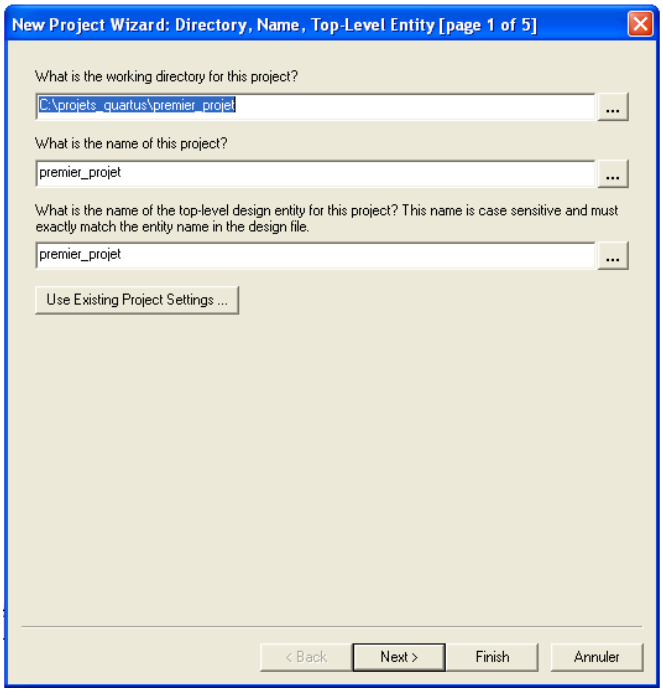


Figure III.3 création de projet sous Quartus

- Dans la seconde, il est possible d'ajouter des fichiers déjà existants au projet. Ne rien ajouter et cliquer sur le bouton Suivant.
- La troisième fenêtre de dialogue permet de choisir le FPGA ou CPLD ciblé.
- Une fois choisi, on en termine avec l'assistant de création de projet en cliquant sur Finish.

La réalisation sous Quartus amène à l'écriture de 3 sources VHDL et d'un schéma hiérarchique.

- **Ajout d'une source VHDL**

Pour ajouter une nouvelle source VHDL **: *File -> New***... et on choisit une source de type VHDL (il est intéressant de remarquer l'étendue des possibilités en matière de source).

- **Sauvegarde d'un projet**

Après avoir taper le code, la sauvegarde se fait comme suite *:* ***File -> Save As***...puis entrer un nom avec l'extension **.vhd**).

- **Inclure un modèle**

Il est possible d'inclure des modèles d'un simple clic dans le code source en cours d'édition. Cet outil (Template) est accessible via le menu ***Edit -> Insert Template...*** ou via l'outil dans la palette d'outils de l'éditeur.

❖ Vérification de la syntaxe du code

Quartus propose un outil de contrôle d'erreur pour limiter la présence de coquilles et d'erreurs d'inattention syntaxique dans le code. On le fait comme suite : ***Menu Processing -> Analyse current file*** ou l'outil dans la palette d'outils de l'éditeur).

❖ Création d'un stimulus de simulation

Quand le code VHDL est correct en matière de syntaxe, il est ensuite nécessaire de le tester en lui appliquant différents vecteurs de test, la solution pour réaliser ce test avec Quartus II est l'éditeur graphique de vecteurs de test (Vector Waveform Editor).

✧ Pour créer un fichier de test graphique, atteindre le menu et faire ***File -> New...*** et choisir ***Vector Waveform File***. utiliser la commande ***Save As...*** pour enregistrer le fichier de test.

✧ Pour rendre les signaux d'entrées et de sorties accessibles à l'édition dans l'éditeur graphique, il faut réaliser une compilation de Design : ***menu Processing -> Start Compilation.***

✧ Ajouter des signaux d'entrée :

- Pour créer des stimuli sur les entrées, il faut ajouter les signaux d'entrées dans la colonne ***Name*** par un double clic dans cette même colonne (ou la commande ***Insert -> Insert Node or Bus... du menuEdit***). Poursuivre par un clic sur le bouton ***Node***

Finder... de la boite de dialogue qui apparaît. Dans le second dialogue, sélectionner ***Pins :all*** dans la boite de liste ***Filter*** et lancer l'opération de filtrage par un clic sur le bouton ***Lis***t.

- Ajouter les signaux d'entrée dans la liste des signaux sélectionnés (***Selected Nodes***) en utilisant le bouton > . Puis valider par ***OK*** deux fois.

✧ Donner un stimulus à un signal :

- Sélectionner le signal dans la colonne ***Name***.
- Création d'un signal de type horloge pour le stimulus du signal sélectionné: ***Menu Edit -> Value -> Clock...***(ou cliquer sur l'outil  de création d'horloge). On remplit le dialogue d'horloge pour créer une horloge. Pour changer la durée de la simulation, utiliser la commande du menu ***Edit -> End Time...***

❖ Simulation du projet

Il convient préalablement de définir les paramètres de simulation en accédant à la boite de dialogue correspondante (menu ***Assigments -> Settings puis Category Simulator Settings*** ou également par la fenêtre de tâches, section ***Verify Design -> Simulate Design -> Quartus II simulator -> Edit settings***).

✧ Choisir une simulation fonctionnelle (***Functional Simulation***) pour le mode de simulation (***Simulation mode***).

✧ Ajouter le fichier test.vwf pour les paramètres d'entrées de la simulation (***Simulation input***).

✧ Valider par ***OK***.

✧ Dans la fenêtre des tâches, lancer la simulation par un double clic sur ***Quartus II Simulator*** (***Functional***).

✧ Pour une simulation temporelle, choisir *Timing* pour le mode de simulation.

III.5 Description du moteur pas-a-pas

III.5.1 Généralités :

Un moteur pas à pas est un moteur dont la partie mobile se déplace d'une quantité élémentaire appelée PAS chaque fois que le moteur reçoit une impulsion de courant convenable.

À chaque pas, on obtient un déplacement angulaire. On définit alors le nombre de pas par tours :

- pour les moteurs les plus courants, on trouve : 12 24 36 48 60 100 200 400 pas par tours
- le pas angulaire correspondant étant respectivement : 30° 15° 10° 7.5° 6° 3.6° 1.8° 0.9°

D'autre part, ces remarquables possibilités d'adaptation lui permettent également de pouvoir travailler selon le cas à vitesse continue ou variable.

III.5.2 Commande d'un moteur pas à pas

Pour commander un moteur pas-à-pas il est nécessaire de passer par une chaine de commande constituée de différents étages : **Unité de pilotage, Production des signaux de commande du moteur pas-à-pas, Etage amplificateur.** Ces différents étages sont illustrés par la figure suivante

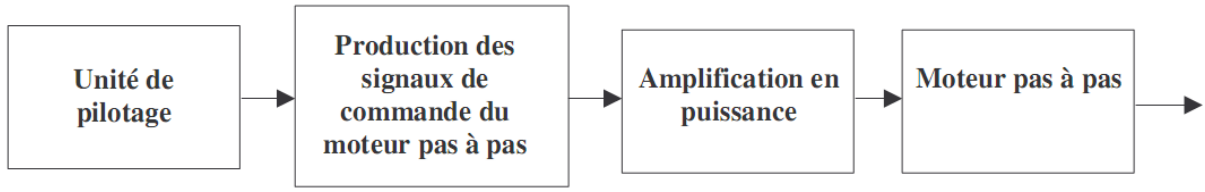


Figure III.4 Les différents étages de commande d'un moteur pas-à-pas

Unité de pilotage : elle permet de commander :

- le sens de rotation,
- l'angle de rotation et donc la vitesse qui en résulte.

Production des signaux de commande du moteur : elle permet de produire les signaux de commande des moteurs pas à pas. Ces signaux dépendent du choix du mode de fonctionnement du moteur pas à pas.

Amplification en puissance : cette fonction permet d'amplifier en puissance les signaux de commande du moteur afin de pouvoir commander le moteur pas à pas. Cette fonction dépend des caractéristiques du moteur pas à pas.

Moteur pas à pas : le moteur pas à pas permet de convertir les signaux de commande du moteur pas à pas en déplacement linéaire ou angulaire.

III.5.1.1 Les différents modes de commande des moteurs pas à pas

Les modes de fonctionnement des moteurs pas à pas sont obtenus par la fonction : « Production des signaux de commande du moteur ».

Alors que l'alimentation (unipolaire ou bipolaire) sera fixée par la fonction «Amplification en puissance ».

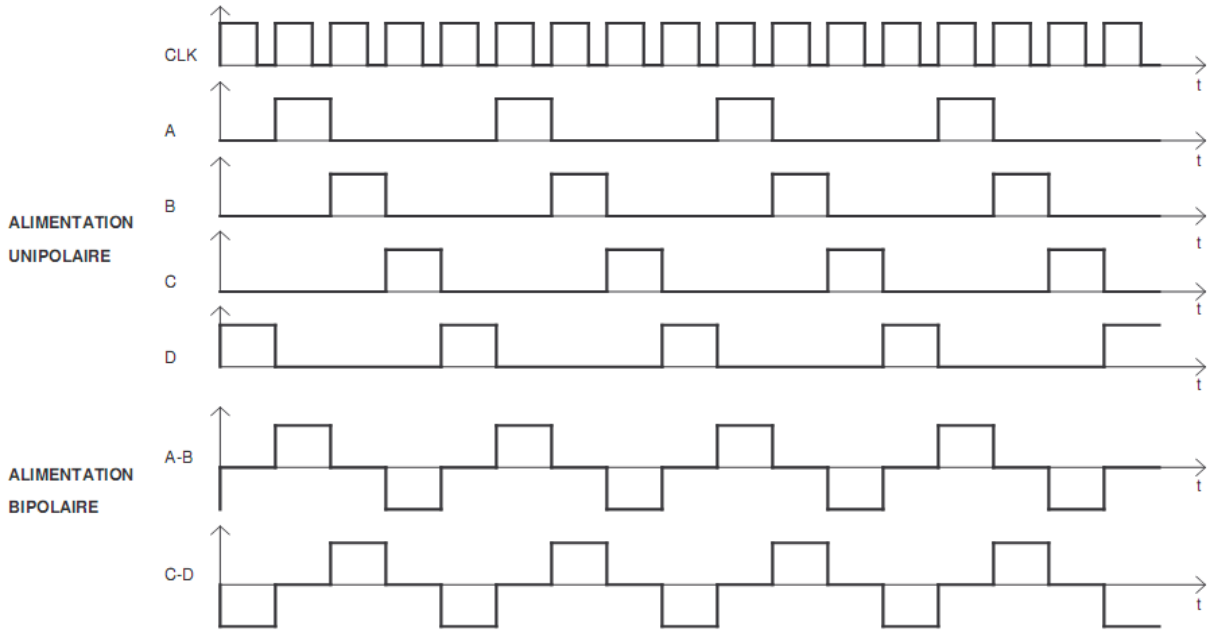


Figure III.5 Mode de commande une phase "on" pas entier : one phase on drive mode

Les phases du moteur pas à pas sont alimentées l'une après l'autre, à chaque impulsion de l'horloge l'angle de rotation est équivalent à un pas.

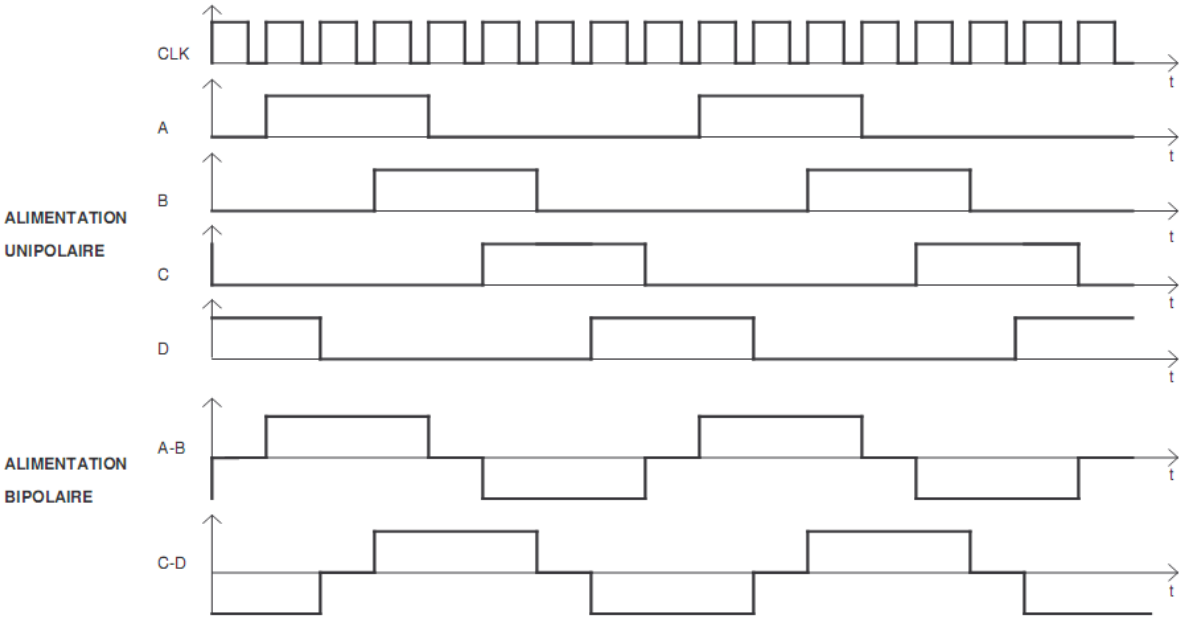


Figure III.6 Mode de commande demi-pas : half step mode

Les bobines sont alimentées alternativement une seule, et ensuite les deux. À chaque impulsion d'horloge l'angle de rotation est équivalent à un demi-pas, ceci permet de gagner en précision avec un même moteur.

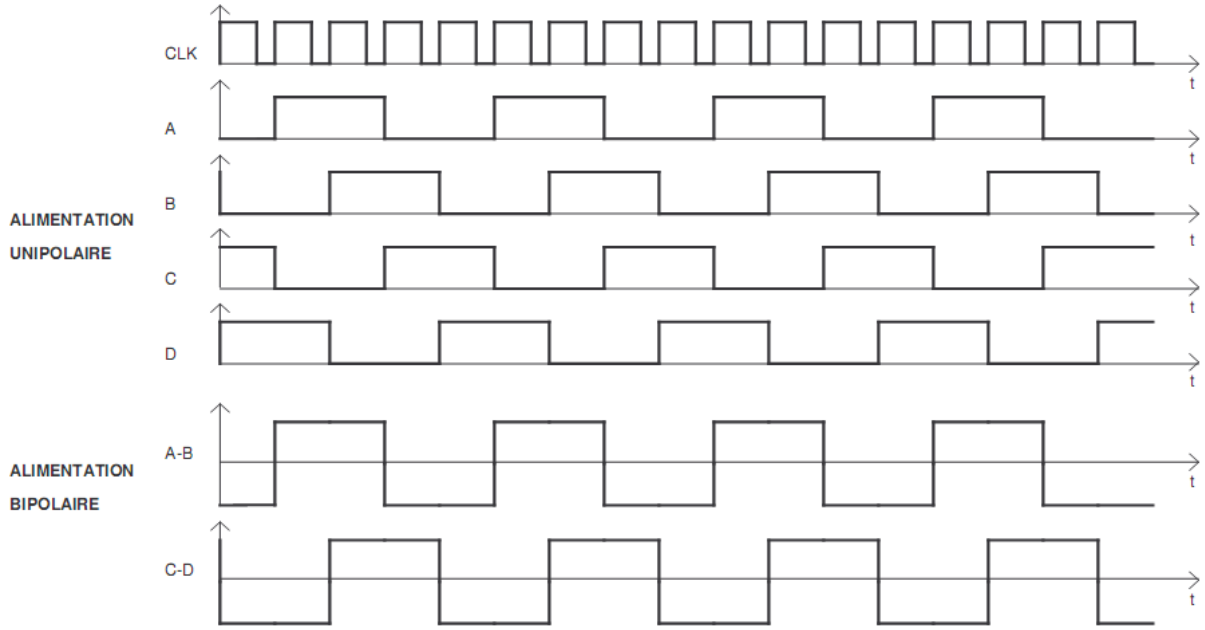


Figure III.7 Mode de commande 2 phases "on" pas entier : two-phase-on drive mode

Le mode de commande ci-dessus est en pas entier, la seule différence est que l'on alimente à chaque fois deux phases à la fois ceci permet d'avoir un couple plus élevé.

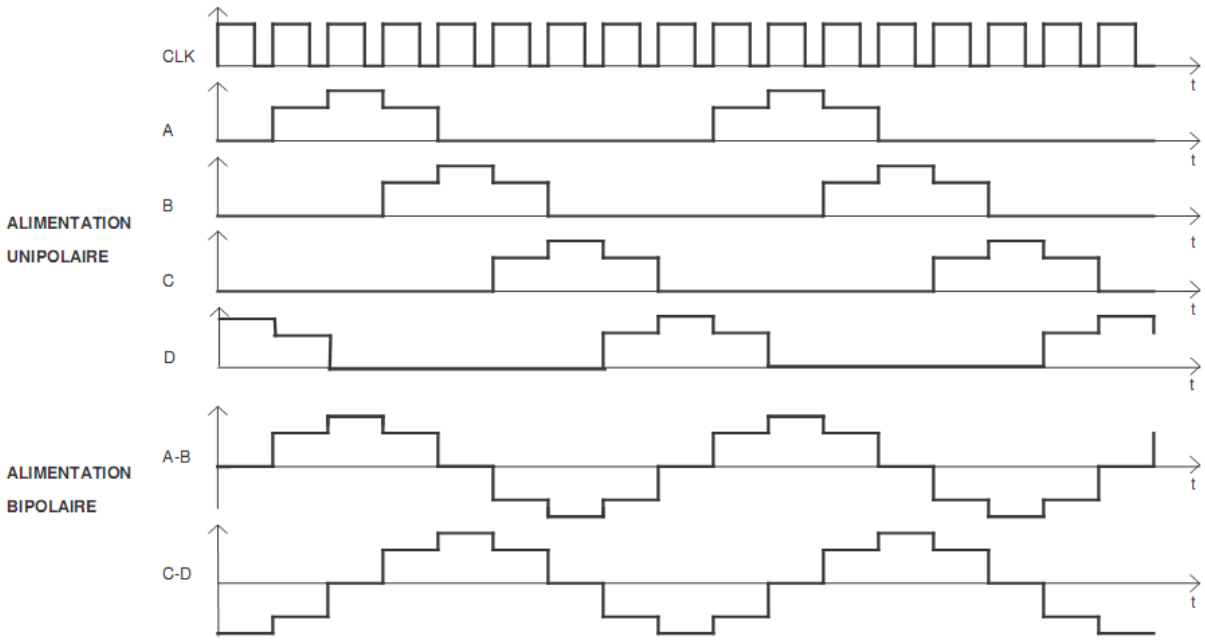


*Figure III.8 Mode de commande demi-pas ** (couple constant) : half step mode*

Ce mode de commande permet d'obtenir un fonctionnement en demi pas à couple constant (la somme des tensions appliquées sur les bobines est toujours constante, contrairement au mode demi pas classique où une bobine est alimentée puis deux …).

Remarque dans application on utilise une commande pour un moteur pas à pas.

III.6 La partie programmation

III.6.1 Principe et raisonnement

On a vu précédemment que l'image est une matrice de pixels, chaque pixel est représenté sur 8bits pour trois couleurs, rouge, vert, bleu, donc trois matrices. On va exploiter cette représentation de l'image pour le suivi d un objet se mouvant dans une ligne droite (gauche, droite) (admet un seul degré de liberté).

❖ La partie acquisition

Lors de l'acquisition de vidéos (succession d'images), on utilise un capteur CCD avec un arrière plan blanc, donc l'image sans l'objet est blanche ce qui implique que les trois matrices qui représentent l'image référence (blanche) sont des matrices de « 1 ».

❖ Le prétraitement

Les images dans lesquelles figure l'objet sont différentes de l'image référence dans les pixels représentant l'objet.

❖ La comparaison

Pour localiser l'objet dans l'image, on effectue une comparaison entre les trois matrices acquises (R, V, B) la où se figure l'objet avec l'image référence (entièrement blanche) pour déterminer seulement la partie couverte par l'objet.

Pour cela on effectue l'opération XOR entre les trois matrices et la matrice référence, donc on aura trois matrices, chacune d'elle indique les zones occupées par le rouge, le vert, et le bleu contenus dans l'objet, ensuite on regroupe les trois matrices en une seule en utilisant l'operateur logique OU en elles, et cela va garder toujours le fond en « 00000000 » (zone hors objet).

❖ Détermination de position

Le but de programme est de veiller à mettre l'objet au centre de l'image, au début on doit centrer l'objet dans l'image pour déterminer ses dimensions et mettre les indices pour s'en référencier lors de la comparaison avec les images entrantes (envoyées par le capteur).

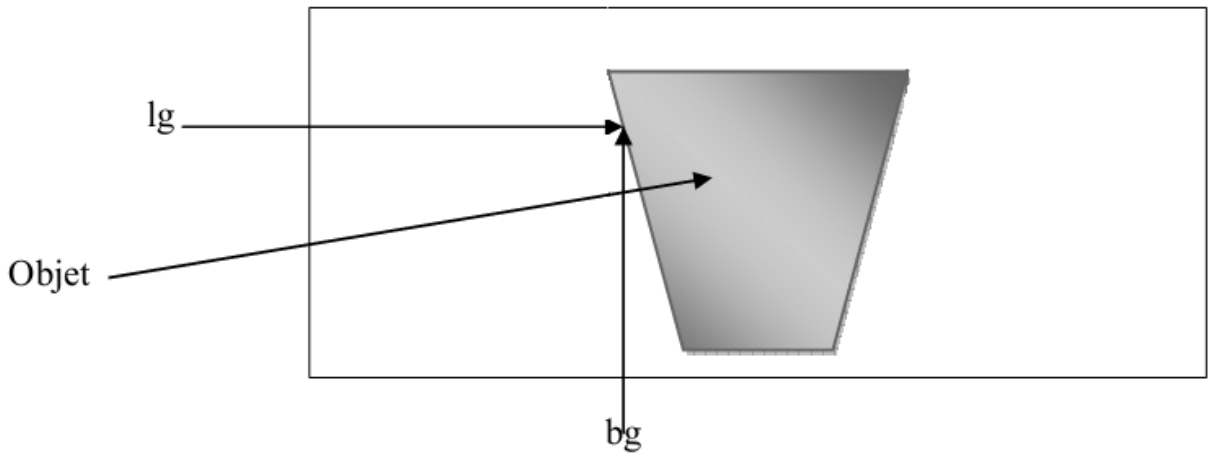


Figure III.9 image montrant l'objet au milieu

Dans notre cas, nous avons utilisé un seul indice « bg » (borne gauche), pour une ligne bien déterminée « lg » on fait un balayage de haut en bas, de gauche à droite.

Dans notre cas le mouvement du capteur est commandé par un seul moteur, ce qui signifie qu'il admet un seul degré de liberté horizontal (gauche, droite).

A la première rencontre du premier pixel différent de « 00000000 » (contour de l'objet), la position de ce pixel sera comparée à l'indice « bg » (borne gauche).

- S'il est inferieur (l < bg), l'objet est à gauche par rapport au centre, ce qui induira l'enclenchement du moteur, il tournera vers la gauche jusqu'à l=bg, ce qui signifie que l'objet est centre.

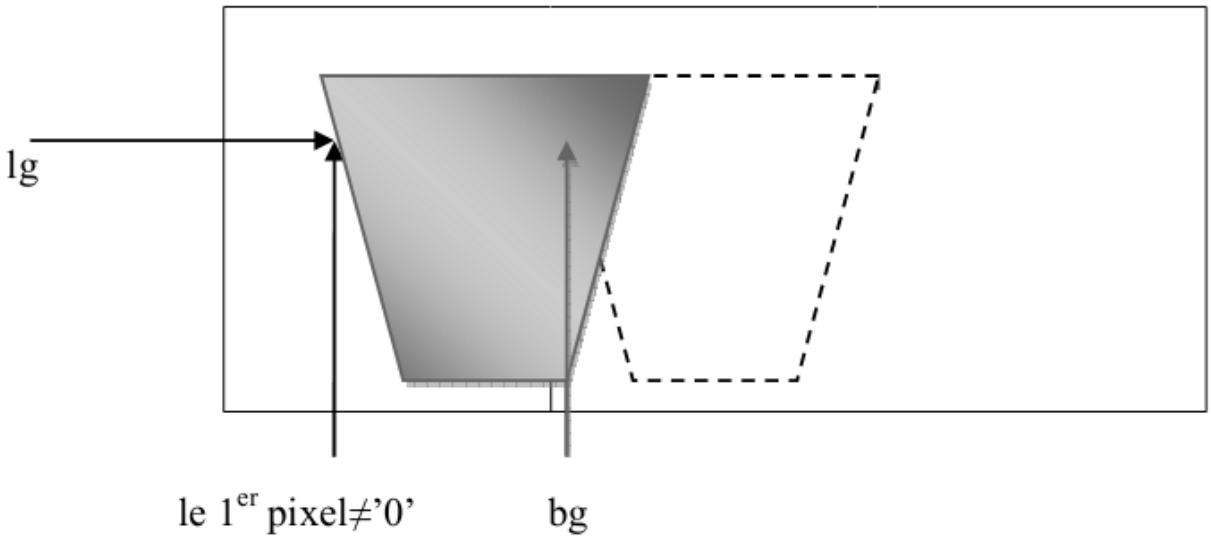


Figure III.10 image montrant l'objet à gauche

- S'il est supérieur (l > bg), le moteur va orienter le capteur vers la droite jusqu'à l=bg.

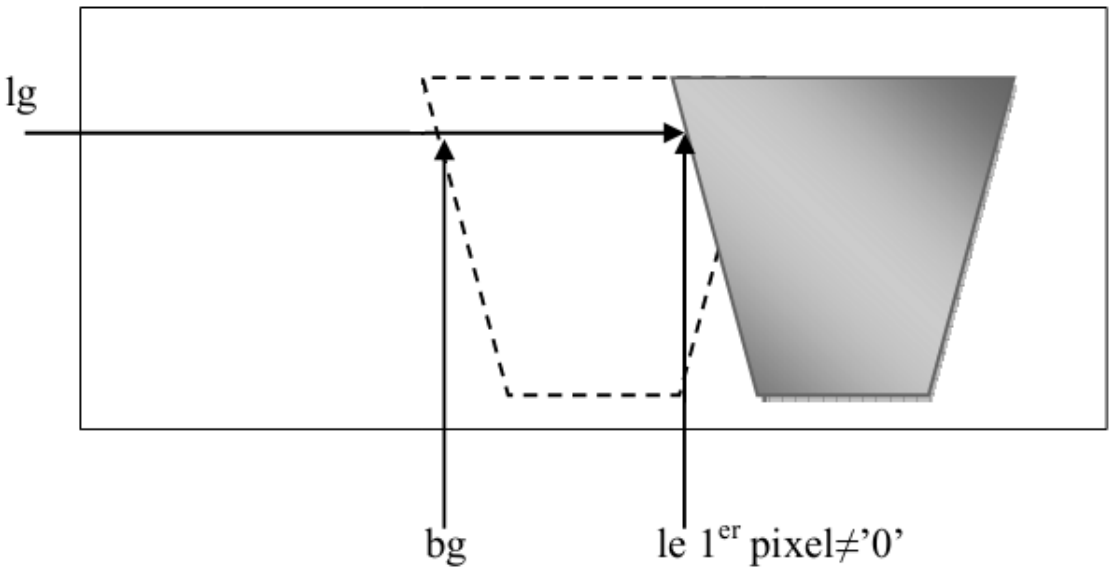


Figure III.11 image montrant l'objet à droite

III.6.2 Mise en application

- Les traitements effectués sur l'image et la détermination de la positon de l objet par rapport au centre jusqu'a l'enclenchement des moteurs sont donnes dans l'organigramme suivant:

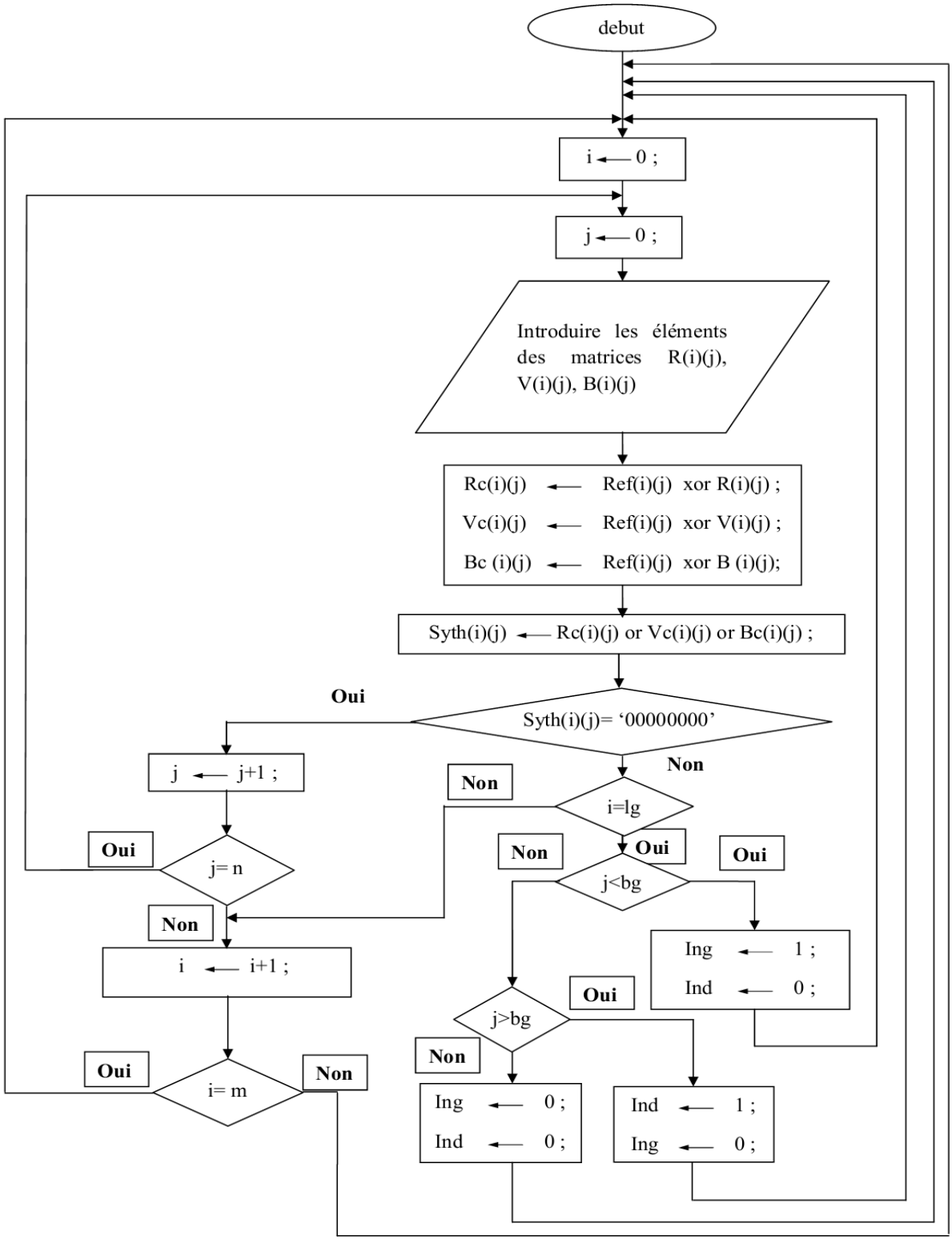


Figure III.12 Organigramme de positionnement et de commande

- R, V, B les les matrices qui constituent l'image acquise de taille m*n
- Ref : c est la matrice de l'image référence qu'on a pris sans l'objet (entièrement blanche)
- Synth : La matrice de synthèse des traitements effectués sur l'image
- lg : la position ou la ligne ou se situe l'indice qu'on a mis dans la matrice dans le cas ou l'objet situé au centre.
- bg : La position (la colonne), ou se situe l'indice, dans le cas ou l'objet est centré
- Ing, ind : Variables de sortie et d'entrée de la machine a états pour l'enclenchement du moteur (gauche, droite respectivement)

❖ Le séquencèrent du moteur est représenté par la machine à état suivante :

telle que l'entrée est a 2 bit est « ing ind »utilises comme sortie dans l'organigramme.

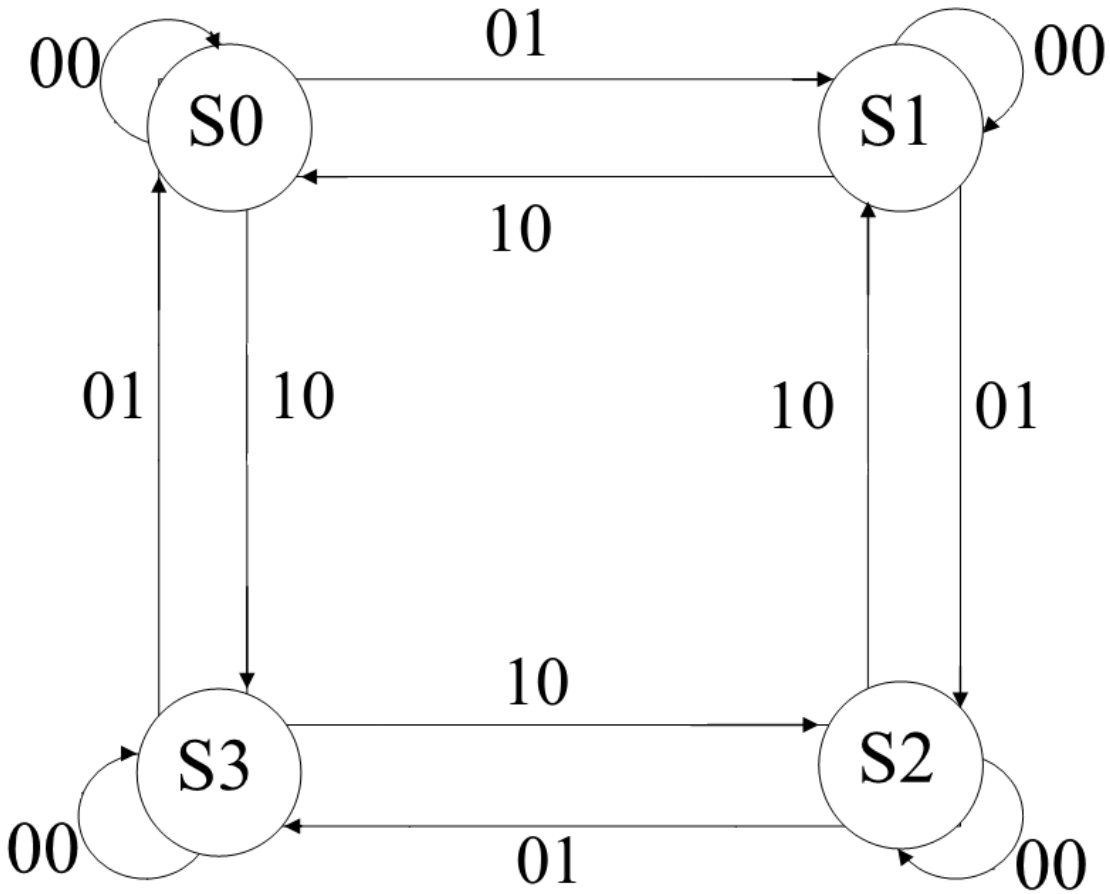


Figure III.13Machine a état pour générer les signaux de commande du moteur

Le moteur admet 4 combinaisons de commande telle qu'on le voit sur la figure III.13

- S0 représente la sortie « 1100 »
- S1 représente la sortie « 0110 »
- S2 représente la sortie « 0011 »
- S3 représente la sortie « 1001 »

III.6.3 Résultats de simulation

- **Dans le cas ou l'objet dans le coté gauche**

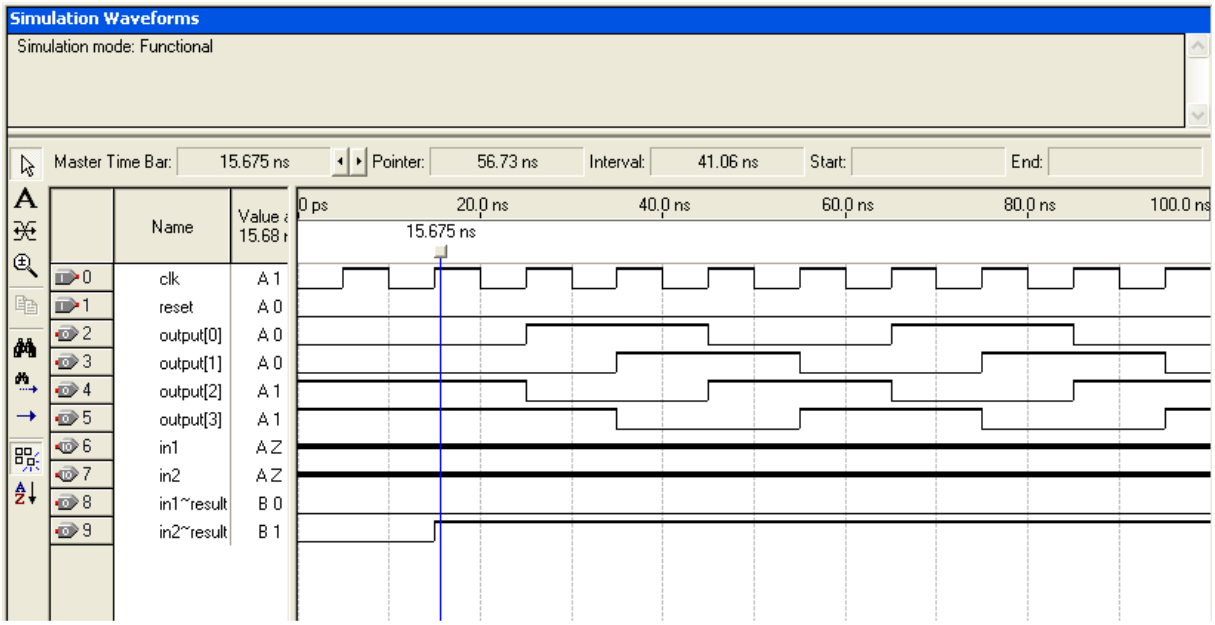


Figure III.14 Signaux de commande du moteur dans le cas ou l'objet est a gauche de l'image

- **Dans le cas ou l'objet dans le coté droit**

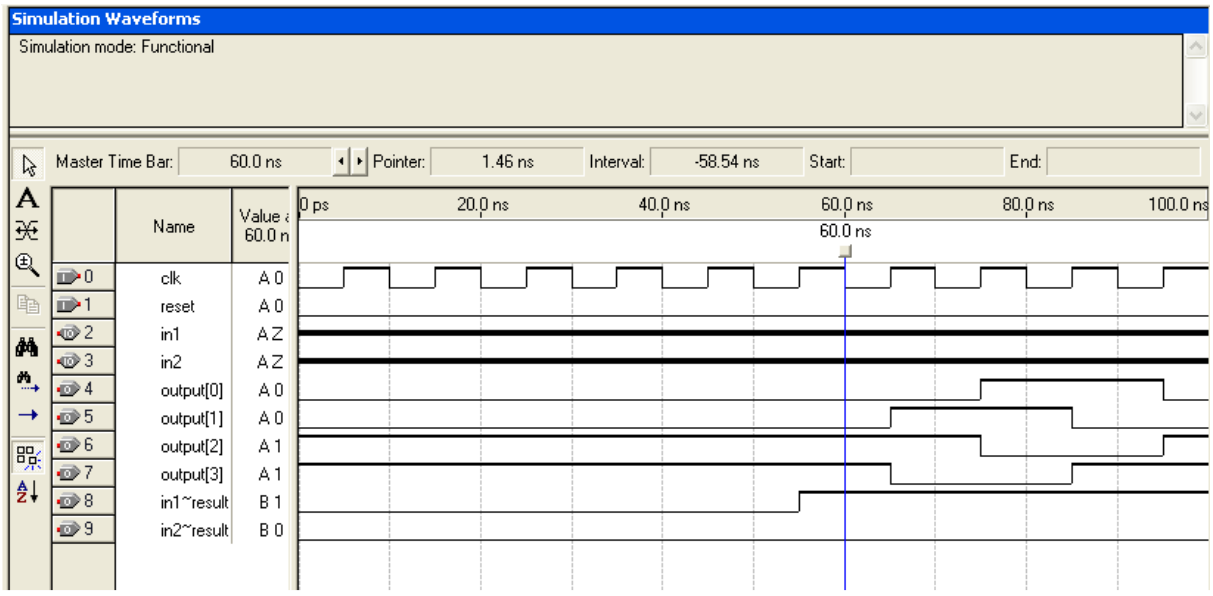


Figure III.15 signaux de commande du moteur dans le cas ou l'objet est a droite de l'image

❖ **Dans le cas ou l'objet est au centre**

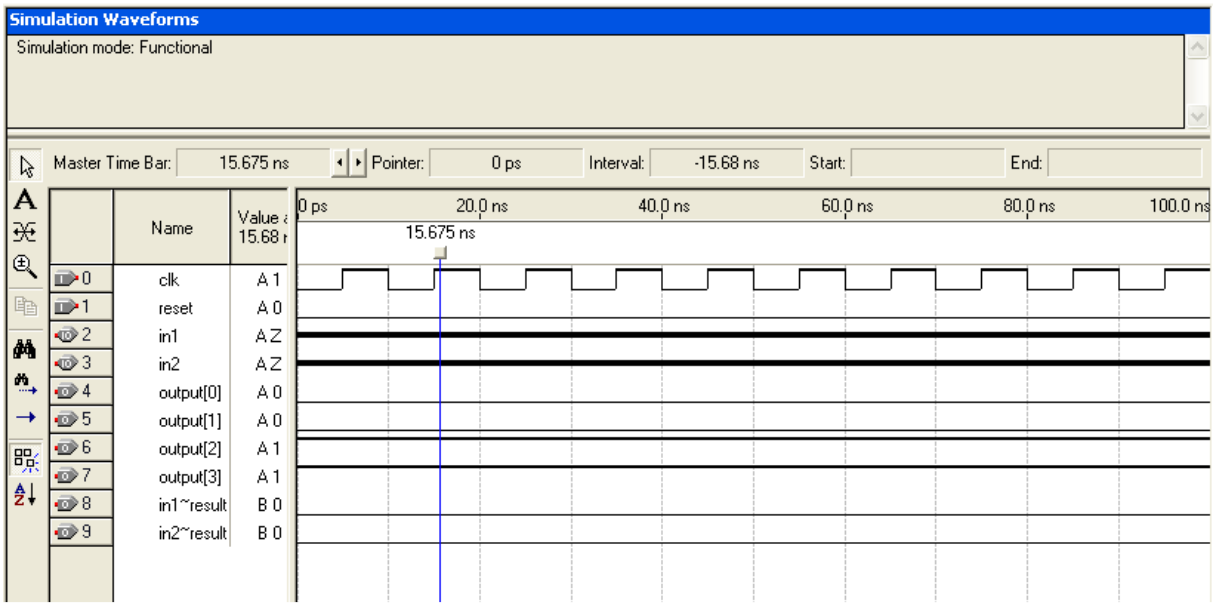


Figure III.16 Signaux de commande du moteur dans le cas ou l'objet est au centre de l'image

III.7 Conclusion

Dans chapitre nous avons adopté une approche expérimentale pour implémenter notre solution de suivi d'objet par une commande de moteur pas à pas.

Les algorithmes utilisés ont été validés par simulation sous le logiciel Quartus, le choix des fonctions logiques binaire permet d atteindre nos objectifs en un minimum de temps de traitement.

Conclusion générale

Conclusion

Le travail qui m'a été confié nous a permis de découvrir le monde de la programmation des systèmes embarqués et le domaine du temps réel, j ai eu l'occasion de programmer expérimentalement un circuit logique programmable de type FPGA, sous l'environnement de développement QUARTUS d'ALTERA sous différents modes de programmation.

Durant la préparation de ce mémoire, on s'est heurté à la difficulté de l'acquisition vidéo et le traitement d'image ainsi que la commande. En dépit de ces difficultés, nous espérons avoir été a la hauteur du travail qui nous a été proposé.

Notre travail peut être amélioré et peut constituer des perspectives très intéressantes.

Enfin, nous espérons que notre travail puisse servir de support pour les promotions à venir.

Bibliographie

Bibliographie

- ✓ D.Pellerin, D.Taylor, VHDL Made Easy!, edition PTR PRENTICE HALL,1997.

- ✓ S.HAUCK, A.DEHON, RECONFIGURABLE COMPUTING THE THEORY AND PRACTICE OF FPGA-BASED COMPUTATION, edition ELSEVIER, 2008.

- ✓ S.Kilts, Advanced FPGA DesignArchitecture, Implementation, and Optimization, edition John Wiley & Sons, Inc, 2007.

- ✓ O.PATRICE, Moteur pas-a-pas et PC, edition DUNOD, 2004.

- ✓ LINGRAND DIANE, Introduction au traitement d'image, edition Vuibert, 2008.

- ✓ S.DJAMA, A.MANSEUR, memoire de fin d'étude MASTER, Etude et implementation d'un modulateur FSK sur un circuit FPGA, department d'electronique, 2011.

Sites :

http://www.altera.com

Annexe

Annexe :

Le brochage de chaque périphérique et les assignements des pins dans le cicuit FPGA sont représentés dans les tableaux suivants :

Signal Name	FPGA Pin No.	Description
KEY0	PIN_G26	Pushbutton[0]
KEY1	PIN_N23	Pushbutton[1]
KEY2	PIN_P23	Pushbutton[2]
KEY3	PIN_W26	Pushbutton[3]

Tableau 1 Liste des pins connectées aux boutons-poussoirs

Signal Name	FPGA Pin No.	Description
SW0	PIN_N25	Toggle Switch[0]
SW1	PIN_N26	Toggle Switch[1]
SW2	PIN_P25	Toggle Switch[2]
SW3	PIN_AE14	Toggle Switch[3]
SW4	PIN_AF14	Toggle Switch[4]
SW5	PIN_AD13	Toggle Switch[5]
SW6	PIN_AC13	Toggle Switch[6]
SW7	PIN_C13	Toggle Switch[7]
SW8	PIN_B13	Toggle Switch[8]
SW9	PIN_A13	Toggle Switch[9]
SW10	PIN_N1	Toggle Switch[10]
SW11	PIN_P1	Toggle Switch[11]
SW12	PIN_P2	Toggle Switch[12]
SW13	PIN_T7	Toggle Switch[13]
SW14	PIN_U3	Toggle Switch[14]
SW15	PIN_U4	Toggle Switch[15]
SW16	PIN_V1	Toggle Switch[16]
SW17	PIN_V2	Toggle Switch[17]

Tableau 2 Liste des pins connectées aux Switchs

Signal Name	FPGA Pin No.	Description
LEDR0	PIN_AE23	LED Red[0]
LEDR1	PIN_AF23	LED Red[1]
LEDR2	PIN_AB21	LED Red[2]
LEDR3	PIN_AC22	LED Red[3]
LEDR4	PIN_AD22	LED Red[4]
LEDR5	PIN_AD23	LED Red[5]
LEDR6	PIN_AD21	LED Red[6]
LEDR7	PIN_AC21	LED Red[7]
LEDR8	PIN_AA14	LED Red[8]
LEDR9	PIN_Y13	LED Red[9]
LEDR10	PIN_AA13	LED Red[10]
LEDR11	PIN_AC14	LED Red[11]
LEDR12	PIN_AD15	LED Red[12]
LEDR13	PIN_AE15	LED Red[13]
LEDR14	PIN_AF13	LED Red[14]
LEDR15	PIN_AE13	LED Red[15]
LEDR16	PIN_AE12	LED Red[16]
LEDR17	PIN_AD12	LED Red[17]
LEDG0	PIN_AE22	LED Green[0]
LEDG1	PIN_AF22	LED Green[1]
LEDG2	PIN_W19	LED Green[2]
LEDG3	PIN_V18	LED Green[3]
LEDG4	PIN_U18	LED Green[4]
LEDG5	PIN_U17	LED Green[5]
LEDG6	PIN_AA20	LED Green[6]
LEDG7	PIN_Y18	LED Green[7]
LEDG8	PIN_Y12	LED Green[8]

Tableau 3 Liste des pins connectées aux LEDs

Signal Name	FPGA Pin No.	Description
HEX0 0	PIN_AF10	Seven Segment Digit 0[0]
HEX0 1	PIN_AB12	Seven Segment Digit 0[1]
HEX0 2	PIN_AC12	Seven Segment Digit 0[2]
HEX0 3	PIN_AD11	Seven Segment Digit 0[3]
HEX0 4	PIN_AE11	Seven Segment Digit 0[4]
HEX0 5	PIN_V14	Seven Segment Digit 0[5]
HEX0 6	PIN_V13	Seven Segment Digit 0[6]
HEX1 0	PIN_V20	Seven Segment Digit 1[0]
HEX1 1	PIN_V21	Seven Segment Digit 1[1]
HEX1 2	PIN_W21	Seven Segment Digit 1[2]
HEX1 3	PIN_Y22	Seven Segment Digit 1[3]
HEX1 4	PIN_AA24	Seven Segment Digit 1[4]
HEX1 5	PIN_AA23	Seven Segment Digit 1[5]
HEX1 6	PIN_AB24	Seven Segment Digit 1[6]
HEX2 0	PIN_AB23	Seven Segment Digit 2[0]
HEX2 1	PIN_V22	Seven Segment Digit 2[1]
HEX2 2	PIN_AC25	Seven Segment Digit 2[2]
HEX2 3	PIN_AC26	Seven Segment Digit 2[3]
HEX2 4	PIN_AB26	Seven Segment Digit 2[4]
HEX2 5	PIN_AB25	Seven Segment Digit 2[5]
HEX2 6	PIN_Y24	Seven Segment Digit 2[6]
HEX3 0	PIN_Y23	Seven Segment Digit 3[0]
HEX3 1	PIN_AA25	Seven Segment Digit 3[1]
HEX3 2	PIN_AA26	Seven Segment Digit 3[2]
HEX3 3	PIN_Y26	Seven Segment Digit 3[3]
HEX3 4	PIN_Y25	Seven Segment Digit 3[4]
HEX3 5	PIN_U22	Seven Segment Digit 3[5]
HEX3 6	PIN_W24	Seven Segment Digit 3[6]
HEX4 0	PIN_U9	Seven Segment Digit 4[0]
HEX4 1	PIN_U1	Seven Segment Digit 4[1]
HEX4 2	PIN_U2	Seven Segment Digit 4[2]
HEX4 3	PIN_T4	Seven Segment Digit 4[3]
HEX4 4	PIN_R7	Seven Segment Digit 4[4]
HEX4 5	PIN_R6	Seven Segment Digit 4[5]
HEX4 6	PIN_T3	Seven Segment Digit 4[6]
HEX5 0	PIN_T2	Seven Segment Digit 5[0]

HEX5 1	PIN_P6	Seven Segment Digit 5[1]
HEX5 2	PIN_P7	Seven Segment Digit 5[2]
HEX5 3	PIN_T9	Seven Segment Digit 5[3]
HEX5 4	PIN_R5	Seven Segment Digit 5[4]
HEX5 5	PIN_R4	Seven Segment Digit 5[5]
HEX5 6	PIN_R3	Seven Segment Digit 5[6]
HEX6 0	PIN_R2	Seven Segment Digit 6[0]
HEX6 1	PIN_P4	Seven Segment Digit 6[1]
HEX6 2	PIN_P3	Seven Segment Digit 6[2]
HEX6 3	PIN_M2	Seven Segment Digit 6[3]
HEX6 4	PIN_M3	Seven Segment Digit 6[4]
HEX6 5	PIN_M5	Seven Segment Digit 6[5]
HEX6 6	PIN_M4	Seven Segment Digit 6[6]
HEX7 0	PIN_L3	Seven Segment Digit 7[0]
HEX7 1	PIN_L2	Seven Segment Digit 7[1]
HEX7 2	PIN_L9	Seven Segment Digit 7[2]
HEX7 3	PIN_L6	Seven Segment Digit 7[3]
HEX7 4	PIN_L7	Seven Segment Digit 7[4]
HEX7 5	PIN_P9	Seven Segment Digit 7[5]
HEX7 6	PIN_N9	Seven Segment Digit 7[6]

Tableau 4 Liste des pins connectées aux afficheurs 7 segments

Signal Name	FPGA Pin No.	Description
CLOCK_27	PIN_D13	27 MHz clock input
CLOCK_50	PIN_N2	50 MHz clock input
EXT_CLOCK	PIN_P26	External (SMA) clock input

Tableau 5 Liste des pins connectées aux horloges d'entrée

Signal Name	FPGA Pin No.	Description
LCD_DATA[0]	PIN_J1	LCD Data[0]
LCD_DATA[1]	PIN_J2	LCD Data[1]
LCD_DATA[2]	PIN_H1	LCD Data[2]
LCD_DATA[3]	PIN_H2	LCD Data[3]
LCD_DATA[4]	PIN_J4	LCD Data[4]
LCD_DATA[5]	PIN_J3	LCD Data[5]
LCD_DATA[6]	PIN_H4	LCD Data[6]
LCD_DATA[7]	PIN_H3	LCD Data[7]
LCD_RW	PIN_K4	LCD Read/Write Select, 0 = Write, 1 = Read
LCD_EN	PIN_K3	LCD Enable
LCD_RS	PIN_K1	LCD Command/Data Select, 0 = Command, 1 = Data
LCD_ON	PIN_L4	LCD Power ON/OFF
LCD_BLON	PIN_K2	LCD Back Light ON/OFF

Tableau 6 Liste des pins connectées a l'afficheur LCD

Signal Name	FPGA Pin No.	Description
GPIO_0[0]	PIN_D25	GPIO Connection 0[0]
GPIO_0[1]	PIN_J22	GPIO Connection 0[1]
GPIO_0[2]	PIN_E26	GPIO Connection 0[2]
GPIO_0[3]	PIN_E25	GPIO Connection 0[3]
GPIO_0[4]	PIN_F24	GPIO Connection 0[4]
GPIO_0[5]	PIN_F23	GPIO Connection 0[5]
GPIO_0[6]	PIN_J21	GPIO Connection 0[6]
GPIO_0[7]	PIN_J20	GPIO Connection 0[7]

GPIO_0[8]	PIN_F25	GPIO Connection 0[8]
GPIO_0[9]	PIN_F26	GPIO Connection 0[9]
GPIO_0[10]	PIN_N18	GPIO Connection 0[10]
GPIO_0[11]	PIN_P18	GPIO Connection 0[11]
GPIO_0[12]	PIN_G23	GPIO Connection 0[12]
GPIO_0[13]	PIN_G24	GPIO Connection 0[13]
GPIO_0[14]	PIN_K22	GPIO Connection 0[14]
GPIO_0[15]	PIN_G25	GPIO Connection 0[15]
GPIO_0[16]	PIN_H23	GPIO Connection 0[16]
GPIO_0[17]	PIN_H24	GPIO Connection 0[17]
GPIO_0[18]	PIN_J23	GPIO Connection 0[18]
GPIO_0[19]	PIN_J24	GPIO Connection 0[19]
GPIO_0[20]	PIN_H25	GPIO Connection 0[20]
GPIO_0[21]	PIN_H26	GPIO Connection 0[21]
GPIO_0[22]	PIN_H19	GPIO Connection 0[22]
GPIO_0[23]	PIN_K18	GPIO Connection 0[23]
GPIO_0[24]	PIN_K19	GPIO Connection 0[24]
GPIO_0[25]	PIN_K21	GPIO Connection 0[25]
GPIO_0[26]	PIN_K23	GPIO Connection 0[26]
GPIO_0[27]	PIN_K24	GPIO Connection 0[27]
GPIO_0[28]	PIN_L21	GPIO Connection 0[28]
GPIO_0[29]	PIN_L20	GPIO Connection 0[29]
GPIO_0[30]	PIN_J25	GPIO Connection 0[30]
GPIO_0[31]	PIN_J26	GPIO Connection 0[31]
GPIO_0[32]	PIN_L23	GPIO Connection 0[32]
GPIO_0[33]	PIN_L24	GPIO Connection 0[33]
GPIO_0[34]	PIN_L25	GPIO Connection 0[34]
GPIO_0[35]	PIN_L19	GPIO Connection 0[35]
GPIO_1[0]	PIN_K25	GPIO Connection 1[0]
GPIO_1[1]	PIN_K26	GPIO Connection 1[1]
GPIO_1[2]	PIN_M22	GPIO Connection 1[2]
GPIO_1[3]	PIN_M23	GPIO Connection 1[3]
GPIO_1[4]	PIN_M19	GPIO Connection 1[4]
GPIO_1[5]	PIN_M20	GPIO Connection 1[5]
GPIO_1[6]	PIN_N20	GPIO Connection 1[6]
GPIO_1[7]	PIN_M21	GPIO Connection 1[7]
GPIO_1[8]	PIN_M24	GPIO Connection 1[8]
GPIO_1[9]	PIN_M25	GPIO Connection 1[9]

GPIO_1[10]	PIN_N24	GPIO Connection 1[10]
GPIO_1[11]	PIN_P24	GPIO Connection 1[11]
GPIO_1[12]	PIN_R25	GPIO Connection 1[12]
GPIO_1[13]	PIN_R24	GPIO Connection 1[13]
GPIO_1[14]	PIN_R20	GPIO Connection 1[14]
GPIO_1[15]	PIN_T22	GPIO Connection 1[15]
GPIO_1[16]	PIN_T23	GPIO Connection 1[16]
GPIO_1[17]	PIN_T24	GPIO Connection 1[17]
GPIO_1[18]	PIN_T25	GPIO Connection 1[18]
GPIO_1[19]	PIN_T18	GPIO Connection 1[19]
GPIO_1[20]	PIN_T21	GPIO Connection 1[20]
GPIO_1[21]	PIN_T20	GPIO Connection 1[21]
GPIO_1[22]	PIN_U26	GPIO Connection 1[22]
GPIO_1[23]	PIN_U25	GPIO Connection 1[23]
GPIO_1[24]	PIN_U23	GPIO Connection 1[24]
GPIO_1[25]	PIN_U24	GPIO Connection 1[25]
GPIO_1[26]	PIN_R19	GPIO Connection 1[26]
GPIO_1[27]	PIN_T19	GPIO Connection 1[27]
GPIO_1[28]	PIN_U20	GPIO Connection 1[28]
GPIO_1[29]	PIN_U21	GPIO Connection 1[29]
GPIO_1[30]	PIN_V26	GPIO Connection 1[30]
GPIO_1[31]	PIN_V25	GPIO Connection 1[31]
GPIO_1[32]	PIN_V24	GPIO Connection 1[32]
GPIO_1[33]	PIN_V23	GPIO Connection 1[33]
GPIO_1[34]	PIN_W25	GPIO Connection 1[34]
GPIO_1[35]	PIN_W23	GPIO Connection 1[35]

Tableau 7 Liste des pins connectées aux ports d'extension

Signal Name	FPGA Pin No.	Description
VGA_R[0]	PIN_C8	VGA Red[0]
VGA_R[1]	PIN_F10	VGA Red[1]
VGA_R[2]	PIN_G10	VGA Red[2]
VGA_R[3]	PIN_D9	VGA Red[3]
VGA_R[4]	PIN_C9	VGA Red[4]
VGA_R[5]	PIN_A8	VGA Red[5]
VGA_R[6]	PIN_H11	VGA Red[6]
VGA_R[7]	PIN_H12	VGA Red[7]
VGA_R[8]	PIN_F11	VGA Red[8]
VGA_R[9]	PIN_E10	VGA Red[9]
VGA_G[0]	PIN_B9	VGA Green[0]
VGA_G[1]	PIN_A9	VGA Green[1]
VGA_G[2]	PIN_C10	VGA Green[2]
VGA_G[3]	PIN_D10	VGA Green[3]
VGA_G[4]	PIN_B10	VGA Green[4]
VGA_G[5]	PIN_A10	VGA Green[5]
VGA_G[6]	PIN_G11	VGA Green[6]
VGA_G[7]	PIN_D11	VGA Green[7]
VGA_G[8]	PIN_E12	VGA Green[8]
VGA_G[9]	PIN_D12	VGA Green[9]
VGA_B[0]	PIN_J13	VGA Blue[0]
VGA_B[1]	PIN_J14	VGA Blue[1]
VGA_B[2]	PIN_F12	VGA Blue[2]
VGA_B[3]	PIN_G12	VGA Blue[3]
VGA_B[4]	PIN_J10	VGA Blue[4]
VGA_B[5]	PIN_J11	VGA Blue[5]
VGA_B[6]	PIN_C11	VGA Blue[6]
VGA_B[7]	PIN_B11	VGA Blue[7]
VGA_B[8]	PIN_C12	VGA Blue[8]
VGA_B[9]	PIN_B12	VGA Blue[9]
VGA_CLK	PIN_B8	VGA Clock
VGA_BLANK	PIN_D6	VGA BLANK
VGA_HS	PIN_A7	VGA H_SYNC
VGA_VS	PIN_D8	VGA V_SYNC
VGA_SYNC	PIN_B7	VGA SYNC

Tableau 8 Liste des pins connectées a l'ADV7123

Signal Name	FPGA Pin No.	Description
AUD_ADCLRCK	PIN_C5	Audio CODEC ADC LR Clock
AUD_ADCDAT	PIN_B5	Audio CODEC ADC Data
AUD_DACLRCK	PIN_C6	Audio CODEC DAC LR Clock
AUD_DACDAT	PIN_A4	Audio CODEC DAC Data
AUD_XCK	PIN_A5	Audio CODEC Chip Clock
AUD_BCLK	PIN_B4	Audio CODEC Bit-Stream Clock
I2C_SCLK	PIN_A6	I2C Data
I2C_SDAT	PIN_B6	I2C Clock

Tableau 9 Liste des pins connectées au codec audio

Signal Name	FPGA Pin No.	Description
UART_RXD	PIN_C25	UART Receiver
UART_TXD	PIN_B25	UART Transmitter

Tableau 10 Liste des pins connectées au port série RS-232

Signal Name	FPGA Pin No.	Description
PS2_CLK	PIN_D26	PS2 Data
PS2_DAT	PIN_C24	PS2 Clock

Tableau 11 Liste des pins connectées au port série PS / 2

Signal Name	FPGA Pin No.	Description
ENET_DATA[0]	PIN_D17	DM9000A DATA[0]
ENET_DATA[1]	PIN_C17	DM9000A DATA[1]
ENET_DATA[2]	PIN_B18	DM9000A DATA[2]
ENET_DATA[3]	PIN_A18	DM9000A DATA[3]
ENET_DATA[4]	PIN_B17	DM9000A DATA[4]
ENET_DATA[5]	PIN_A17	DM9000A DATA[5]
ENET_DATA[6]	PIN_B16	DM9000A DATA[6]
ENET_DATA[7]	PIN_B15	DM9000A DATA[7]
ENET_DATA[8]	PIN_B20	DM9000A DATA[8]
ENET_DATA[9]	PIN_A20	DM9000A DATA[9]
ENET_DATA[10]	PIN_C19	DM9000A DATA[10]
ENET_DATA[11]	PIN_D19	DM9000A DATA[11]
ENET_DATA[12]	PIN_B19	DM9000A DATA[12]
ENET_DATA[13]	PIN_A19	DM9000A DATA[13]
ENET_DATA[14]	PIN_E18	DM9000A DATA[14]
ENET_DATA[15]	PIN_D18	DM9000A DATA[15]
ENET_CLK	PIN_B24	DM9000A Clock 25 MHz
ENET_CMD	PIN_A21	DM9000A Command/Data Select, 0 = Command, 1 = Data
ENET_CS_N	PIN_A23	DM9000A Chip Select
ENET_INT	PIN_B21	DM9000A Interrupt
ENET_RD_N	PIN_A22	DM9000A Read
ENET_WR_N	PIN_B22	DM9000A Write
ENET_RST_N	PIN_B23	DM9000A Reset

Tableau 12 Liste des pins connectées au Contrôleur Fast Ethernet

Signal Name	FPGA Pin No.	Description
TD_DATA[0]	PIN_J9	TV Decoder Data[0]
TD_DATA[1]	PIN_E8	TV Decoder Data[1]
TD_DATA[2]	PIN_H8	TV Decoder Data[2]
TD_DATA[3]	PIN_H10	TV Decoder Data[3]
TD_DATA[4]	PIN_G9	TV Decoder Data[4]
TD_DATA[5]	PIN_F9	TV Decoder Data[5]
TD_DATA[6]	PIN_D7	TV Decoder Data[6]
TD_DATA[7]	PIN_C7	TV Decoder Data[7]
TD_HS	PIN_D5	TV Decoder H_SYNC
TD_VS	PIN_K9	TV Decoder V_SYNC
TD_RESET	PIN_C4	TV Decoder Reset
I2C_SCLK	PIN_A6	I2C Data
I2C_SDAT	PIN_B6	I2C Clock

Tableau 13 Liste des pins connectées au décodeur TV

Signal Name	FPGA Pin No.	Description
OTG_ADDR[0]	PIN_K7	ISP1362 Address[0]
OTG_ADDR[1]	PIN_F2	ISP1362 Address[1]
OTG_DATA[0]	PIN_F4	ISP1362 Data[0]
OTG_DATA[1]	PIN_D2	ISP1362 Data[1]
OTG_DATA[2]	PIN_D1	ISP1362 Data[2]
OTG_DATA[3]	PIN_F7	ISP1362 Data[3]
OTG_DATA[4]	PIN_J5	ISP1362 Data[4]
OTG_DATA[5]	PIN_J8	ISP1362 Data[5]
OTG_DATA[6]	PIN_J7	ISP1362 Data[6]
OTG_DATA[7]	PIN_H6	ISP1362 Data[7]
OTG_DATA[8]	PIN_E2	ISP1362 Data[8]
OTG_DATA[9]	PIN_E1	ISP1362 Data[9]
OTG_DATA[10]	PIN_K6	ISP1362 Data[10]
OTG_DATA[11]	PIN_K5	ISP1362 Data[11]
OTG_DATA[12]	PIN_G4	ISP1362 Data[12]
OTG_DATA[13]	PIN_G3	ISP1362 Data[13]
OTG_DATA[14]	PIN_J6	ISP1362 Data[14]
OTG_DATA[15]	PIN_K8	ISP1362 Data[15]
OTG_CS_N	PIN_F1	ISP1362 Chip Select

OTG_RD_N	PIN_G2	ISP1362 Read
OTG_WR_N	PIN_G1	ISP1362 Write
OTG_RST_N	PIN_G5	ISP1362 Reset
OTG_INT0	PIN_B3	ISP1362 Interrupt 0
OTG_INT1	PIN_C3	ISP1362 Interrupt 1
OTG_DACK0_N	PIN_C2	ISP1362 DMA Acknowledge 0
OTG_DACK1_N	PIN_B2	ISP1362 DMA Acknowledge 1
OTG_DREQ0	PIN_F6	ISP1362 DMA Request 0
OTG_DREQ1	PIN_E5	ISP1362 DMA Request 1
OTG_FSPEED	PIN_F3	USB Full Speed, 0 = Enable, Z = Disable
OTG_LSPEED	PIN_G6	USB Low Speed, 0 = Enable, Z = Disable

Tableau 14 Liste des pins connectées au port USB

Signal Name	FPGA Pin No.	Description
IRDA_TXD	PIN_AE24	IRDA Transmitter
IRDA_RXD	PIN_AE25	IRDA Receiver

Tableau 15 Liste des pins connectées a l' IrDA

Signal Name	FPGA Pin No.	Description
DRAM_ADDR[0]	PIN_T6	SDRAM Address[0]
DRAM_ADDR[1]	PIN_V4	SDRAM Address[1]
DRAM_ADDR[2]	PIN_V3	SDRAM Address[2]
DRAM_ADDR[3]	PIN_W2	SDRAM Address[3]
DRAM_ADDR[4]	PIN_W1	SDRAM Address[4]
DRAM_ADDR[5]	PIN_U6	SDRAM Address[5]
DRAM_ADDR[6]	PIN_U7	SDRAM Address[6]
DRAM_ADDR[7]	PIN_U5	SDRAM Address[7]
DRAM_ADDR[8]	PIN_W4	SDRAM Address[8]
DRAM_ADDR[9]	PIN_W3	SDRAM Address[9]
DRAM_ADDR[10]	PIN_Y1	SDRAM Address[10]
DRAM_ADDR[11]	PIN_V5	SDRAM Address[11]
DRAM_DQ[0]	PIN_V6	SDRAM Data[0]
DRAM_DQ[1]	PIN_AA2	SDRAM Data[1]
DRAM_DQ[2]	PIN_AA1	SDRAM Data[2]
DRAM_DQ[3]	PIN_Y3	SDRAM Data[3]
DRAM_DQ[4]	PIN_Y4	SDRAM Data[4]
DRAM_DQ[5]	PIN_R8	SDRAM Data[5]
DRAM_DQ[6]	PIN_T8	SDRAM Data[6]
DRAM_DQ[7]	PIN_V7	SDRAM Data[7]
DRAM_DQ[8]	PIN_W6	SDRAM Data[8]
DRAM_DQ[9]	PIN_AB2	SDRAM Data[9]
DRAM_DQ[10]	PIN_AB1	SDRAM Data[10]
DRAM_DQ[11]	PIN_AA4	SDRAM Data[11]
DRAM_DQ[12]	PIN_AA3	SDRAM Data[12]
DRAM_DQ[13]	PIN_AC2	SDRAM Data[13]
DRAM_DQ[14]	PIN_AC1	SDRAM Data[14]
DRAM_DQ[15]	PIN_AA5	SDRAM Data[15]

DRAM_BA_0	PIN_AE2	SDRAM Bank Address[0]
DRAM_BA_1	PIN_AE3	SDRAM Bank Address[1]
DRAM_LDQM	PIN_AD2	SDRAM Low-byte Data Mask
DRAM_UDQM	PIN_Y5	SDRAM High-byte Data Mask
DRAM_RAS_N	PIN_AB4	SDRAM Row Address Strobe
DRAM_CAS_N	PIN_AB3	SDRAM Column Address Strobe
DRAM_CKE	PIN_AA6	SDRAM Clock Enable
DRAM_CLK	PIN_AA7	SDRAM Clock
DRAM_WE_N	PIN_AD3	SDRAM Write Enable
DRAM_CS_N	PIN_AC3	SDRAM Chip Select

Tableau 16 Liste des pins connectées a la SDRAM

Signal Name	FPGA Pin No.	Description
SRAM_ADDR[0]	PIN_AE4	SRAM Address[0]
SRAM_ADDR[1]	PIN_AF4	SRAM Address[1]
SRAM_ADDR[2]	PIN_AC5	SRAM Address[2]
SRAM_ADDR[3]	PIN_AC6	SRAM Address[3]
SRAM_ADDR[4]	PIN_AD4	SRAM Address[4]
SRAM_ADDR[5]	PIN_AD5	SRAM Address[5]
SRAM_ADDR[6]	PIN_AE5	SRAM Address[6]
SRAM_ADDR[7]	PIN_AF5	SRAM Address[7]
SRAM_ADDR[8]	PIN_AD6	SRAM Address[8]
SRAM_ADDR[9]	PIN_AD7	SRAM Address[9]
SRAM_ADDR[10]	PIN_V10	SRAM Address[10]
SRAM_ADDR[11]	PIN_V9	SRAM Address[11]
SRAM_ADDR[12]	PIN_AC7	SRAM Address[12]
SRAM_ADDR[13]	PIN_W8	SRAM Address[13]
SRAM_ADDR[14]	PIN_W10	SRAM Address[14]
SRAM_ADDR[15]	PIN_Y10	SRAM Address[15]

SRAM_ADDR[16]	PIN_AB8	SRAM Address[16]
SRAM_ADDR[17]	PIN_AC8	SRAM Address[17]
SRAM_DQ[0]	PIN_AD8	SRAM Data[0]
SRAM_DQ[1]	PIN_AE6	SRAM Data[1]
SRAM_DQ[2]	PIN_AF6	SRAM Data[2]
SRAM_DQ[3]	PIN_AA9	SRAM Data[3]
SRAM_DQ[4]	PIN_AA10	SRAM Data[4]
SRAM_DQ[5]	PIN_AB10	SRAM Data[5]
SRAM_DQ[6]	PIN_AA11	SRAM Data[6]
SRAM_DQ[7]	PIN_Y11	SRAM Data[7]
SRAM_DQ[8]	PIN_AE7	SRAM Data[8]
SRAM_DQ[9]	PIN_AF7	SRAM Data[9]
SRAM_DQ[10]	PIN_AE8	SRAM Data[10]
SRAM_DQ[11]	PIN_AF8	SRAM Data[11]
SRAM_DQ[12]	PIN_W11	SRAM Data[12]
SRAM_DQ[13]	PIN_W12	SRAM Data[13]
SRAM_DQ[14]	PIN_AC9	SRAM Data[14]
SRAM_DQ[15]	PIN_AC10	SRAM Data[15]
SRAM_WE_N	PIN_AE10	SRAM Write Enable
SRAM_OE_N	PIN_AD10	SRAM Output Enable
SRAM_UB_N	PIN_AF9	SRAM High-byte Data Mask
SRAM_LB_N	PIN_AE9	SRAM Low-byte Data Mask
SRAM_CE_N	PIN_AC11	SRAM Chip Enable

Tableau 17 Liste des pins connectées a la SRAM

Signal Name	FPGA Pin No.	Description
FL_ADDR[0]	PIN_AC18	FLASH Address[0]
FL_ADDR[1]	PIN_AB18	FLASH Address[1]
FL_ADDR[2]	PIN_AE19	FLASH Address[2]
FL_ADDR[3]	PIN_AF19	FLASH Address[3]
FL_ADDR[4]	PIN_AE18	FLASH Address[4]
FL_ADDR[5]	PIN_AF18	FLASH Address[5]
FL_ADDR[6]	PIN_Y16	FLASH Address[6]
FL_ADDR[7]	PIN_AA16	FLASH Address[7]
FL_ADDR[8]	PIN_AD17	FLASH Address[8]
FL_ADDR[9]	PIN_AC17	FLASH Address[9]
FL_ADDR[10]	PIN_AE17	FLASH Address[10]
FL_ADDR[11]	PIN_AF17	FLASH Address[11]
FL_ADDR[12]	PIN_W16	FLASH Address[12]
FL_ADDR[13]	PIN_W15	FLASH Address[13]
FL_ADDR[14]	PIN_AC16	FLASH Address[14]
FL_ADDR[15]	PIN_AD16	FLASH Address[15]
FL_ADDR[16]	PIN_AE16	FLASH Address[16]
FL_ADDR[17]	PIN_AC15	FLASH Address[17]
FL_ADDR[18]	PIN_AB15	FLASH Address[18]
FL_ADDR[19]	PIN_AA15	FLASH Address[19]
FL_ADDR[20]	PIN_Y15	FLASH Address[20]
FL_ADDR[21]	PIN_Y14	FLASH Address[21]
FL_DQ[0]	PIN_AD19	FLASH Data[0]
FL_DQ[1]	PIN_AC19	FLASH Data[1]
FL_DQ[2]	PIN_AF20	FLASH Data[2]
FL_DQ[3]	PIN_AE20	FLASH Data[3]
FL_DQ[4]	PIN_AB20	FLASH Data[4]
FL_DQ[5]	PIN_AC20	FLASH Data[5]
FL_DQ[6]	PIN_AF21	FLASH Data[6]
FL_DQ[7]	PIN_AE21	FLASH Data[7]
FL_CE_N	PIN_V17	FLASH Chip Enable
FL_OE_N	PIN_W17	FLASH Output Enable
FL_RST_N	PIN_AA18	FLASH Reset
FL_WE_N	PIN_AA17	FLASH Write Enable

Tableau 18 Liste des pins connectées aux