Projet licence IUP

Traitement d'image à base de FPGA

Traitement d'image à base de FPGA

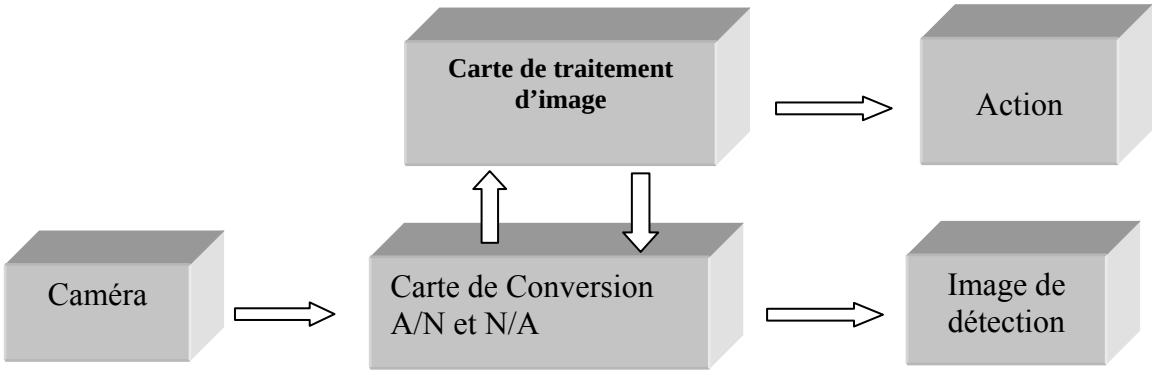
BOUVRY Colin
BONIFACE Laurent

Année 2003-2004

SOMMAIRE

1 OBJECTIFS ET CAHIER DES CHARGES

Le but de cette carte est d'effectuer un traitement de l'image en temps réel. Ce traitement sera réalisé par un composant configurable sur carte, un FPGA. Les possibilités de traitement sont nombreuses et dépendent des ressources fixées autour du FPGA (mémoire par exemple). On concevra une carte permettant de mémoriser une image, une mémorisation d'image, voire une détection de mouvement.
Le temps alloué pour la réalisation est de 60h, soit 30 séances.



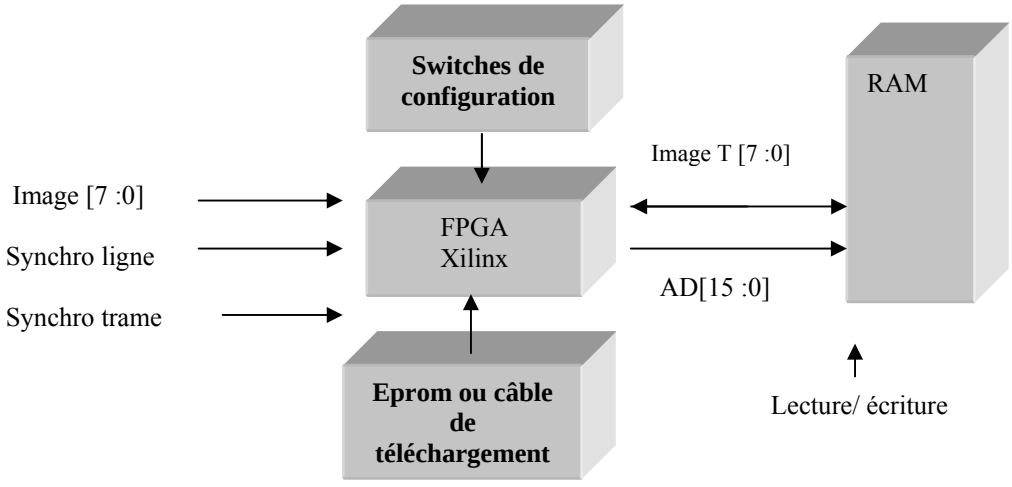
Ce projet comporta plusieurs étapes.

La première sera l'étude et la vérification d'une carte de conversion analogique numérique et numérique analogique du signal vidéo provenant d'une caméra.

La carte ayant déjà été utilisée, on l'étudiera à l'aide des documentations des composants et on refera un plan. On testera son fonctionnement à l'aide d'une nappe « court circuit » entre l'entrée et la sortie vidéo.

Durée : 4 séances. (Analyse du fonctionnement, test)

La deuxième étape sera la réflexion concernant la conception d'une carte de traitement d'image à base de FPGA, dont le schéma de principe est donné ci-dessous.



Le composant FPGA est configurable sur carte à volonté. On pourra y mettre à l'aide d'un outil de développement spécifique l'équivalent de composants discrets comme des compteurs, des bascules, des additionneurs, etc. Cette configuration sera chargée via le port parallèle du PC.

Dans un premier temps, on concevra ce que pourra être le contenu du composant programmable. On ajustera le schéma de la carte complète en fonction de nos choix.

Dans un deuxième temps, on dessinera un schéma d'implantation de la carte complète, a l'aide de la documentation des différents composants.

Durée : 4 séances.

Dans un troisième temps ; on étudiera le schéma d'une carte à base de FPGA du commerce. L'étude de cette carte sera complétée par le test de quelques éléments de base situés sur cette carte (afficheur 7 segments, mémoire interne du FPGA, entrées/ sorties de la carte, mémoire externe).

Durée : 4 à 5 séances.

Dans un quatrième temps, on tentera d'interfacer la carte de conversion analogique numérique et la carte à base de FPGA, afin de traiter le signal vidéo. On testera quelques traitements simples de type inversion de l'image, flou, gradient.

Durée : 2 à 3 séances.

Dans un cinquième temps, on réalisera l'incrustation de texte ou de petites images dans le signal vidéo, en utilisant la mémoire interne du FPGA.

Durée : 2 à 3 séances.

Dans un dernier temps, on mémorisera une image dans la mémoire externe (SDRAM) pour réaliser par exemple une détection de mouvement.

Durée : reste du projet (environ 12 séances)

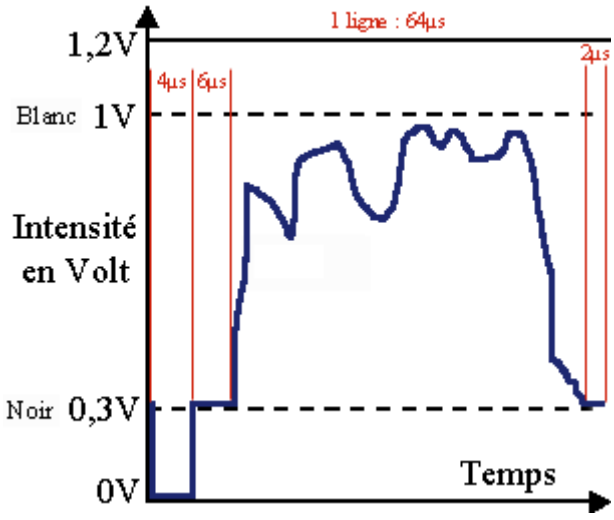
PRINCIPE DE FONCTIONNEMENT DE LA TELEVISION

1.1 Qu'est ce que le support vidéo?

Le support vidéo est un courant électrique dont les variations de tentions sont proportionnelles aux variations de l'éclairement d'une image. C'est pour cela que le signal est dit analogique. Une image très éclairée donne une tension forte alors qu'une image peu éclairée donne une faible tension.

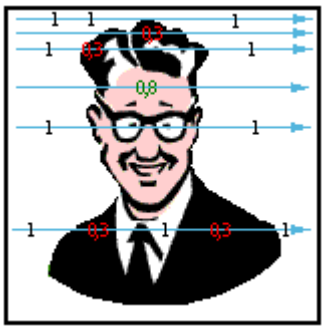
Une norme internationale veut que :

- 0,3 volt = Niveau de noir.
- 1 volt = Niveau de blanc.



Visualisation d'une ligne sur un oscilloscope.

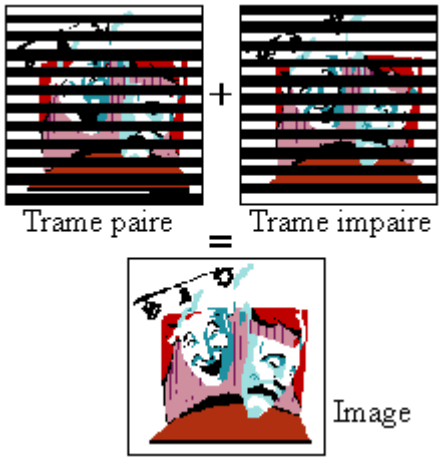
Toutes les images doivent être entre ces deux tensions sinon elles sont déclarées comme "non broadcast", les diffuseurs refusent de l'acheter. C'est un impératif technique d'avoir des images toujours entre 0,3 et 1 volt.



L'intensité électrique est différente en fonction de la luminance.

Un balayage est égal à 312,5 lignes, c'est une trame. A la fin d'une trame le faisceau d'électrons s'éteint et il y a un top de suppression trame. (Le faisceau s'éteint et remonte.)

1.2 Le balayage électronique

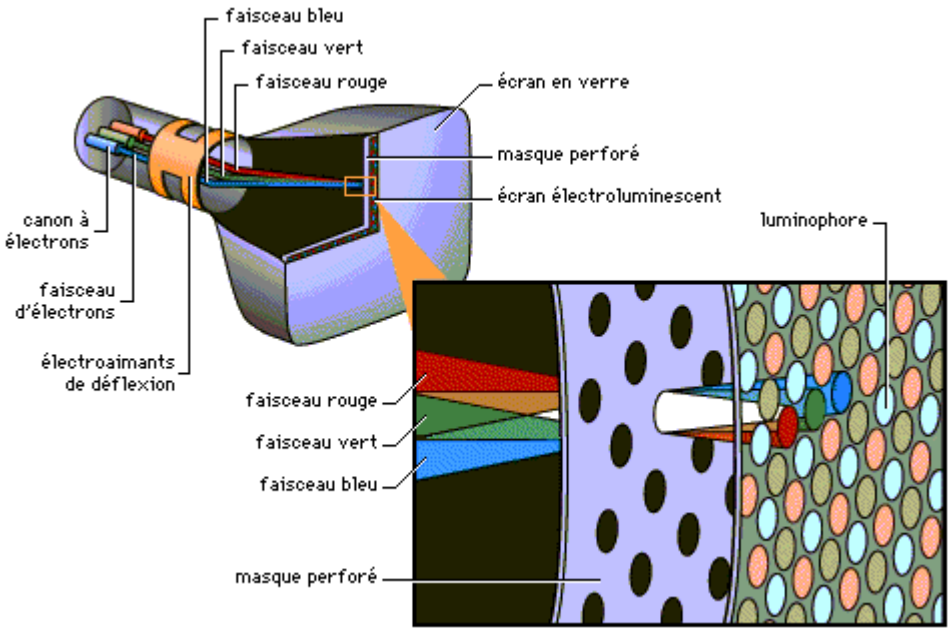


Trame paire et trame impaire : balayage entrelacé.

Une image est égale à deux trames : une paire et une impaire. Au total on compte 625 lignes. C'est un balayage entrelacer (différent du balayage progressif). Le balayage entrelacer permet d'avoir une image beaucoup plus stable, elle ne vibre pas.

1.3 La transduction courrant - lumière

Le téléviseur ou moniteur repasse le courant électrique en image.



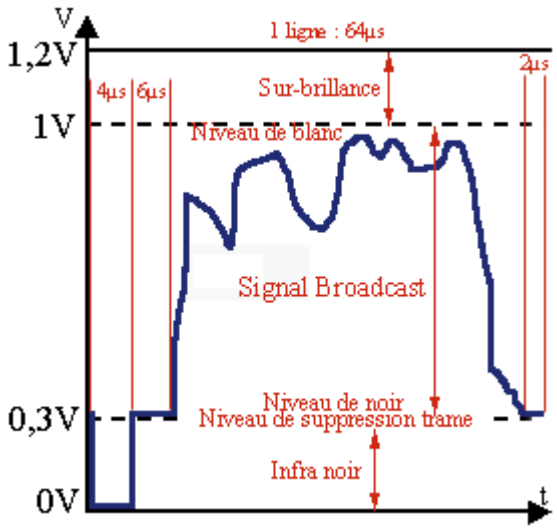
Le tube trichrome.

Un faisceau d'électrons bombarde la vitre du moniteur, les électrons aimants permettent d'orienter le faisceau d'électrons. Sans image, il y a de la neige, la neige est un bombardement d'électrons non orienté. Un poste de télévision couleur contient un tube image trichrome muni de trois canons à électrons, un pour chaque

couleur primaire (bleu, vert, ou rouge). Des électroaimants de déflexion projettent les trois faisceaux d'électrons sur le fond du tube, recouvert d'une couche de matière électroluminescente. Cette couche se compose de luminophores, petits grains qui émettent de la lumière lorsqu'ils sont bombardés par des électrons. Grâce à un masque perforé, chaque faisceau d'électrons vient frapper les luminophores correspondant à leur couleur. La réunion de ces taches de couleur forme l'image observée sur un écran de télévision.

Le signal "vidéo in" qui entre dans le téléviseur pilote le faisceau d'électrons. Une forte tension (1 volt) est égale à un fort bombardement de l'écran qui provoque un fort éclairement donc un point blanc. Une faible tension (0,3 volt) provoque un point noir. L'éclairement de la vitre est proportionnel à la tension du courant. On a le même phénomène qu'avec la caméra mais dans l'autre sens.

Le temps de la suppression de trame est en moyenne de 40 lignes. Le signal visible n'est donc pas de 625 lignes mais de 570 ou 580 lignes.



On ne devrait voir sur l'écran qu'un point plus ou moins brillant sur l'image en mouvement. Heureusement on voit l'image grâce à deux défauts :

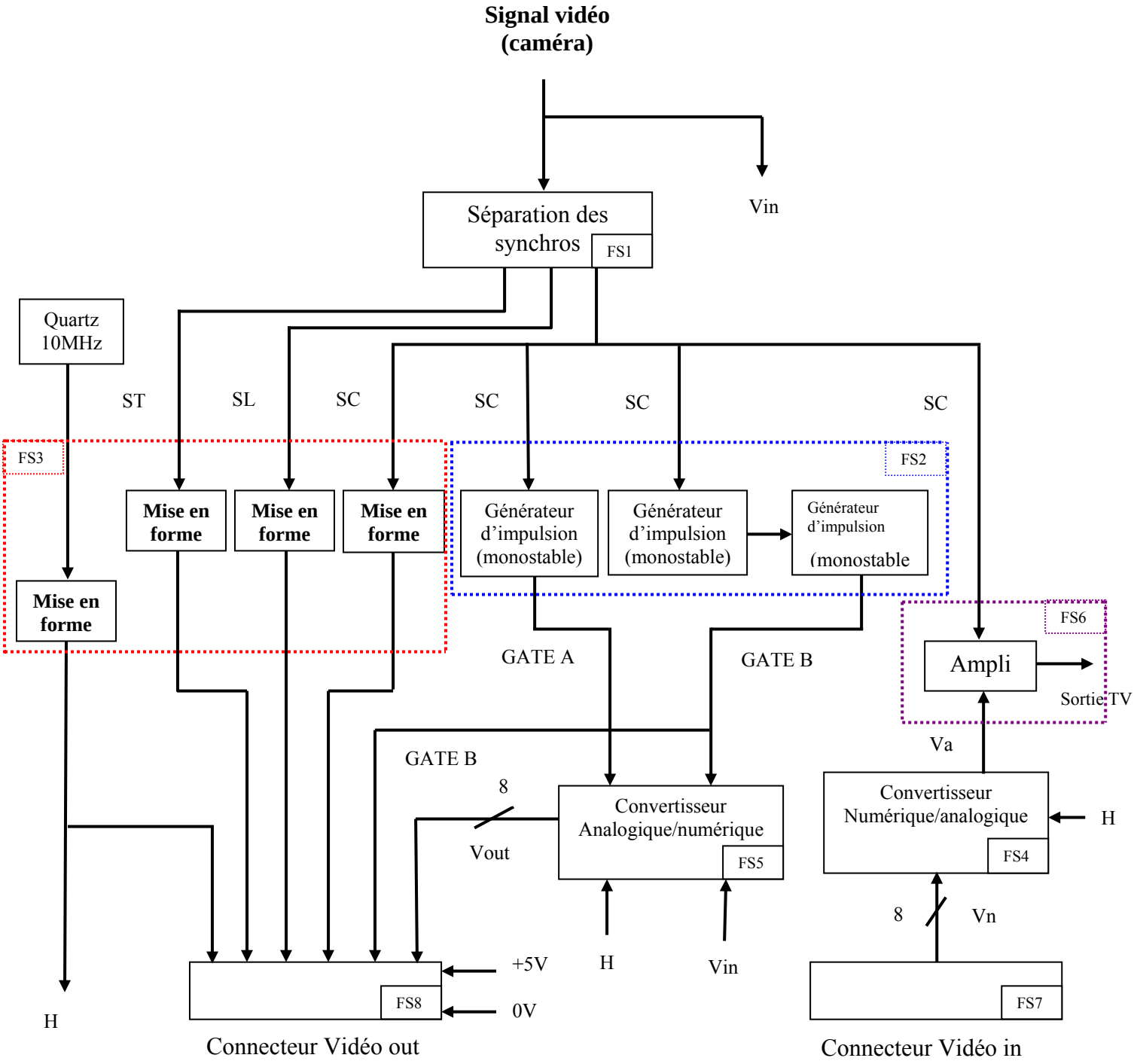
- La rémanence. La propriété du verre avec lequel on fabrique l'écran de la télévision fait que le point lumineux ne s'éteint pas automatiquement.
- La persistance rétinienne est une propriété de l'œil qui garde sur la rétine l'impression lumineuse après extinction de la source. C'est la base du procédé cinéma.

1.4 La synchronisation caméra - moniteur

Lorsque la caméra est en train de balayer la ligne 1, il faut que le moniteur balaye la ligne 1 également. Toutes les caméras et les téléviseurs doivent balayer la même ligne en même temps. Il faut un système de synchronisation, le signal vidéo va asservir le moniteur, il oblige le moniteur a envoyer sur l'écran la même ligne que sur la caméra.

2 CARTE DE CONVERSION A/N et N/A VIDEO

2.1 Etude fonctionnelle



2.2 Description des différentes fonctions du schéma

FS1 séparation des signaux de synchronisation :

Cette fonction sert à séparer différents signaux à partir du signal vidéo. En effet, trois signaux de synchronisation sont nécessaires pour la carte de traitement d'image. Cette fonction est réalisée avec le composant LM1881N.

Explications des signaux entrées / sorties :

Entrée :

-V_{in} : Signal vidéo provenant d'une caméra

Sortie :

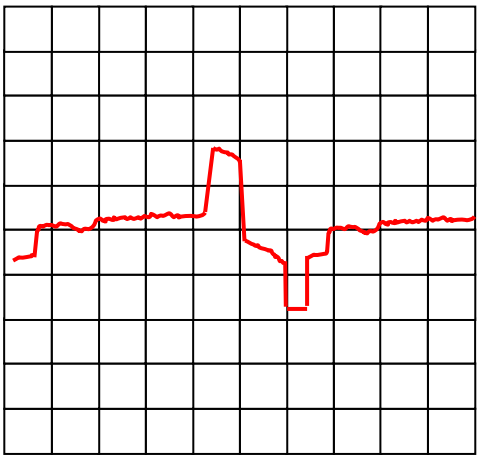
- ST : "Synchro Trame" Signal de synchronisation indiquant la fin ou le début d'une trame paire ou impaire quand le balayage est entrelacé.

-SL : "Synchro Ligne" Signal de synchronisation indiquant la fin d'une ligne et aussi le retour de ligne.

-SC : "Synchro Composite" Signal de synchronisation indiquant le début d'une ligne.

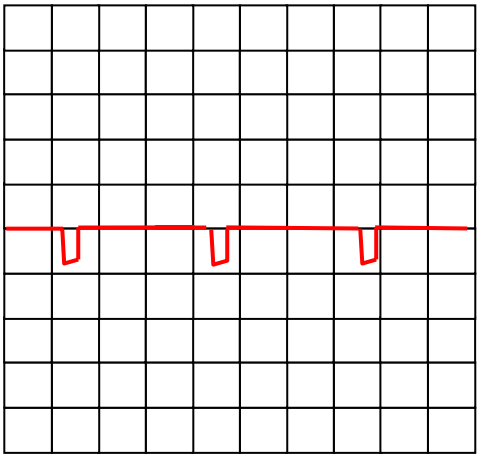
Graphe sortie caméra

500mV/ div
10µs/ div



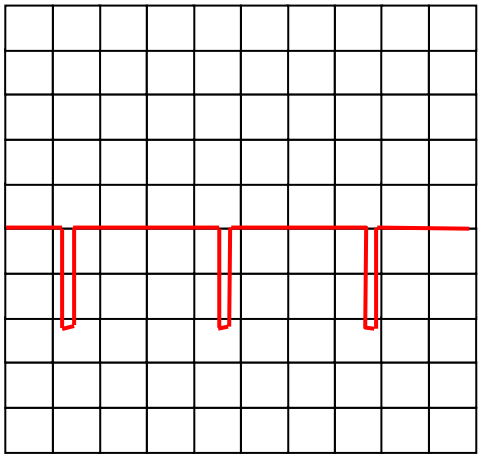
Graphe sortie TV sans caméra

200mV/ div
20µs/ div



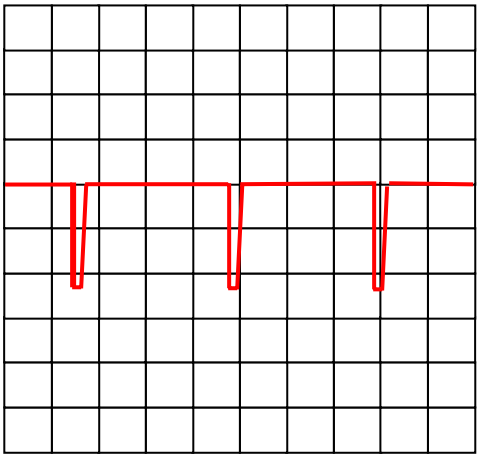
SYNCHRO LIGNE
SL

2V/ div
20µs/ div



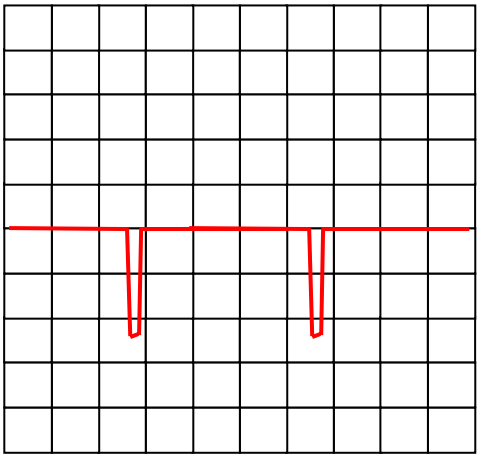
BURST
SC

2V/ div
20µs/ div



SYNCHRO TRAME
ST

2V/ div
5ms/ div



Générateurs d'impulsions :

Cette fonction est réalisée à partir du composant 74123 qui sont des monostables avec RAZ. Elle sert à mettre en forme des signaux nécessaire au bon fonctionnement de la conversion analogique numérique. Ces signaux permettent de synchroniser la conversion dans un mode de fonctionnement du CAN.
Ces signaux doivent avoir une durée à l'état haut bien précise et respecter une synchronisation avec le signal vidéo d'où l'utilisation de monostables pour réaliser cette fonction.

Explications des différents signaux entrée/sortie :

Entrée :

-SC : "synchro Composite" Signal de synchronisation composite

Sortie :

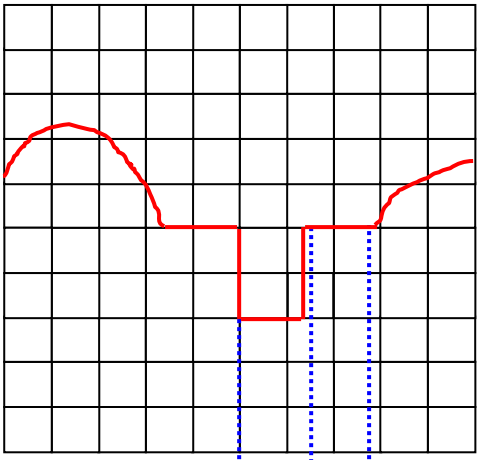
-Gate A : Signal indiquant le début d'une ligne pour le CAN (FS4). Ce signal doit être à l'état haut pendant un temps plus petit que le top synchro composite. Ce signal est réalisé avec un monostable permettant de régler sa durée à l'état haut nécessaire pour que le CAN fonctionne correctement. (Voir chronogrammes ci-dessous)

-Gate B : Signal indiquant le départ de la conversion au CAN (FS4). Ce signal doit être à l'état haut entre le top synchro composite et le départ du premier pixel. Par conséquent, deux monostables sont nécessaires pour réaliser un décalage dans le temps à partir du top synchro composite et pour régler la durée de l'état haut du signal. (Voir chronogrammes ci-dessous)

- 1Q : C'est la sortie du premier monostable qui est relier au deuxième monostable. Ainsi le signal gate B passe à l'état haut quand le signal 1Q passe à l'état bas. On choisit la sortie 1Q et non la sortie 1Q car le monostable est actif sur front montant. (Voir chronogrammes ci-dessous)

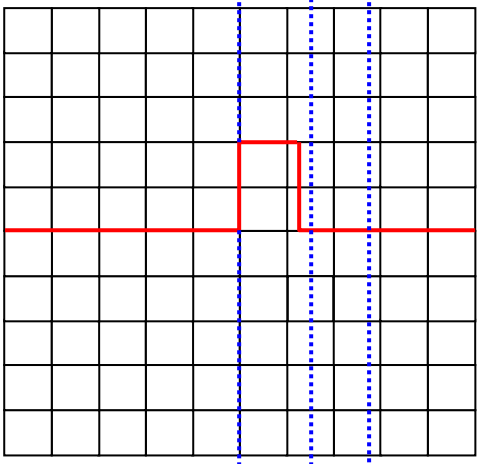
Graphe sortie Caméra

500mV/ div
2µs/ div



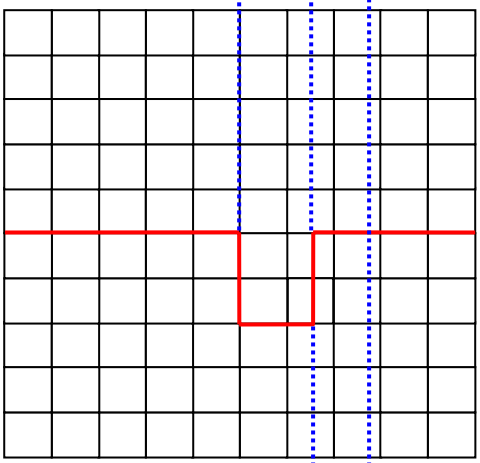
Graphe GATE A

2V/ div
2µs/ div



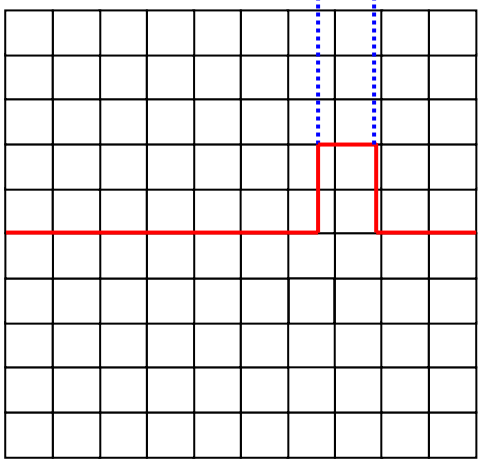
Graphe <u>1Q</u>

2V/ div
2µs/ div



Graphe GATE B

2V/ div
2µs/ div



FS3 Structures de mises en forme :

Cette fonction est réalisée à partir d'un 7808 qui est un circuit contenant 4 portes ET.
Les portes AND permettent d'obtenir une sortance plus importante et une remise en forme au niveau TTL.
Les entrées des portes AND sont reliées pour obtenir des portes OUI.

Rappels des différents signaux entrée/sortie :

Entrée :
- SL, ST, SC et H qui est l'horloge d'échantillonnage

Sortie :
-Les mêmes signaux remis à niveau TTL

FS4 Convertisseur N/A Vidéo :

Cette fonction est réalisée à partir d'un TDA 8702 qui est un convertisseur Numérique Analogique vidéo.
Il convertit lui aussi avec une fréquence d'échantillonnage H de 10MHz.

Rappels des différents signaux entrée/sortie :

Entrée :
-Vn : Signal vidéo numérique provenant du connecteur Vidéo IN
- H

Sortie :
-Va : Signal vidéo analogique issu de la conversion N/A du TDA 8702

FS5 convertisseur A/N Vidéo :

Cette fonction est réalisée à partir d'un TDA 8708A qui est un convertisseur analogique numérique vidéo.
Il convertit à chaque nouveau pixel, c'est à dire à une fréquence d'échantillonnage bien spécifique H qui est de 10MHz.
La conversion nous permet de transposer notre signal vidéo en une succession d'octets. On les stockera ensuite dans une mémoire. On convertit chaque nouveau pixel en un octet qui correspond à la luminance (contraste).

Rappels des différents signaux entrée/sortie :

Entrée :
- Gate A, Gate B, H et V_{in}

Sortie :
- Vout : Signal sur un octet
  Cet échantillon correspond à un extrait du signal V_{in} à un instant t.

FS6 Amplification :

Cette fonction réalise l'ajout de Signal de Synchronisation Composite au Signal Va et par la suite, ce signal est amplifié et adaptée.

FS7 Connecteur Vidéo In :

Ce connecteur nous permet de réaliser l'interface entre différent système donnant un signal vidéo numérique pouvant être traité par notre système et la carte de conversion.
L'information principale présentent sur ce connecteur est le signal vidéo numérique Vn.

Schéma de connectique :

NC	NC	Vn7	Vn6	Vn5	Vn4	Vn3	Vn2	Vn1	Vn0
GND	GND	GND	GND	GND	GND	GND	GND	GND	GND

Figure : Connecteur Vidéo In

FS8 Connecteur Vidéo Out :

Ce connecteur nous permet de réaliser l'interface entre la carte de conversion et différent système pouvant traiter les informations présente sur cette interface.

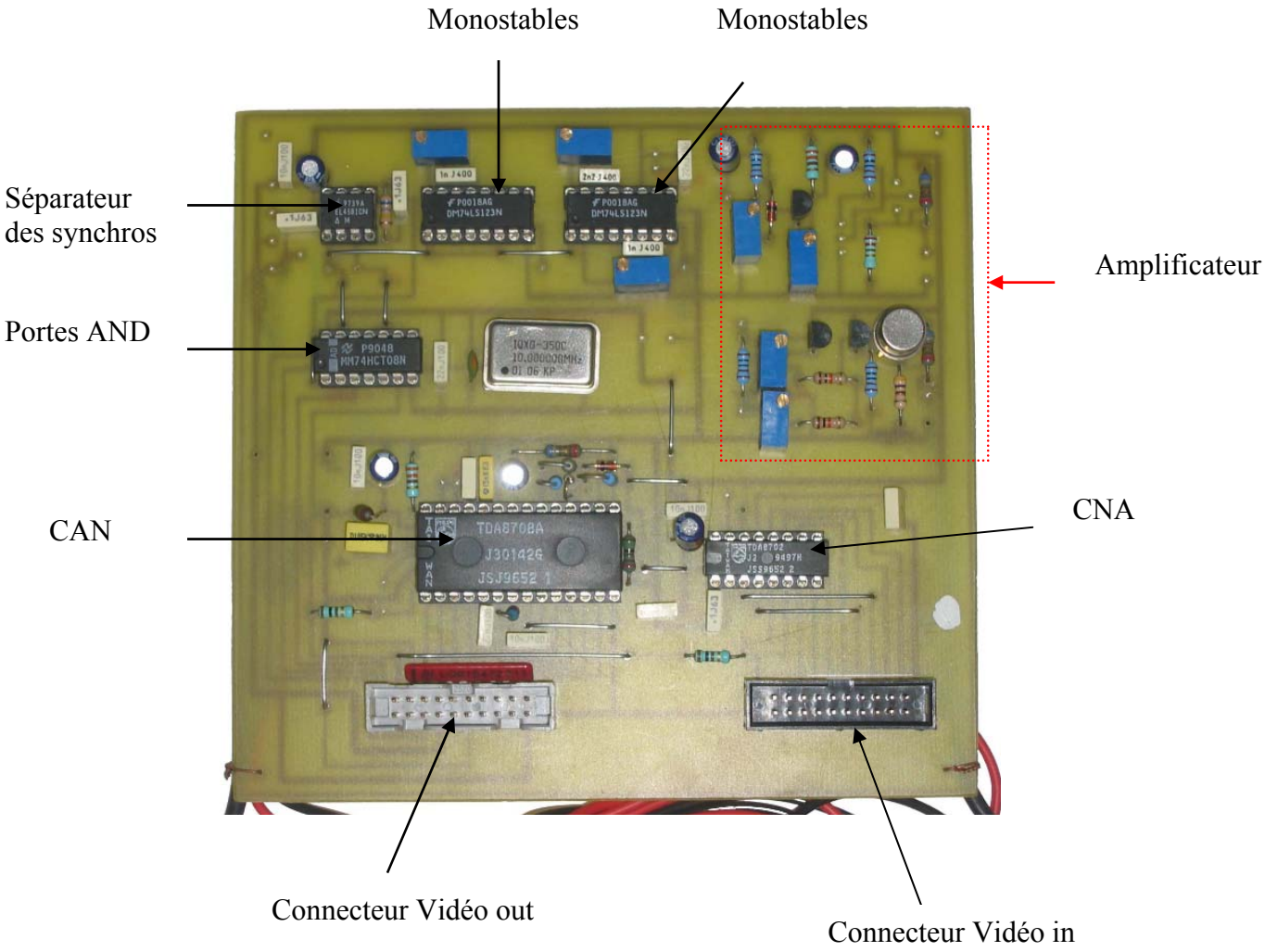
Informations présentes sur le connecteur :

Signaux de référence temporelle : H, ST, SL, SC, GateB
Signal vidéo numérique :V_{out}
Alimentation :+5v/GND

Schéma de connectique :

H	GateB	Vout7	Vout6	Vout5	Vout4	Vout3	Vout2	Vout1	Vout0
SC	ST	SL	GND	GND	+5V	GND	GND	GND	GND

Figure : Connecteur Vidéo Out


Monostables
Monostables
Séparateur
des synchros
Amplificateur
Portes AND
CAN
CNA
Connecteur Vidéo out
Connecteur Vidéo in

3 CARTE FPGA

3.1 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 oeuvre complexe et un circuit imprimé de taille importante. D'où l'utilisation croissante de circuits logiques programmables comme le FPGA (Field Programmable Gate Array). Ce type de produit peut intégrer dans un seul circuit plusieurs fonctions logiques programmables par l'utilisateur. Sa mise en oeuvre se fait très facilement à l'aide d'un programmateur, d'un micro-ordinateur et d'un logiciel adapté.
Pour le traitement d'image en temps réel, il est indispensable de disposer d'un composant aussi avantageux que le FPGA.
Il existe deux grandes marques sur le marché du FPGA : Altera et Xilinx. Pour le projet, un FPGA Xilinx et son logiciel a été choisi.

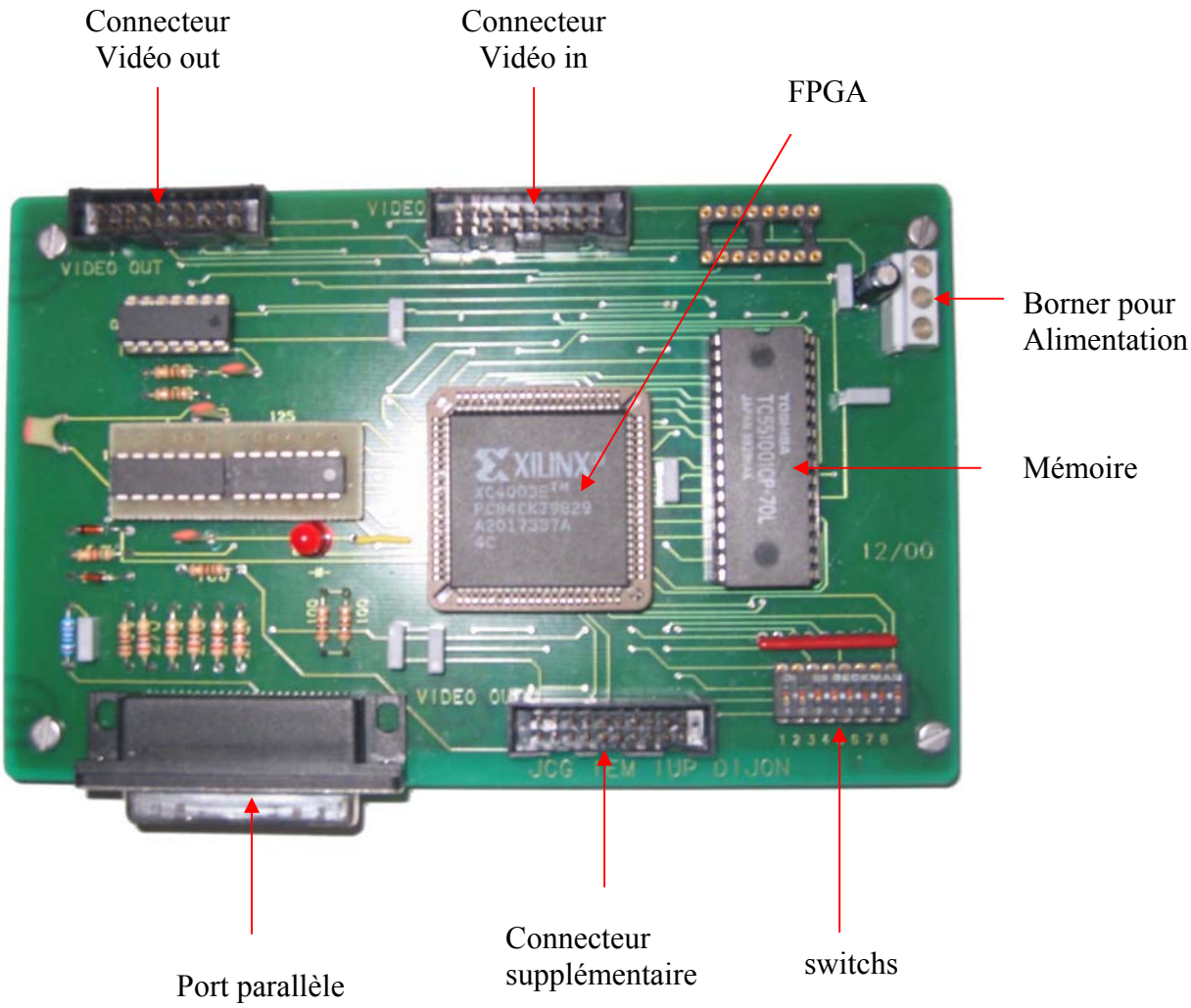
3.2 Description de la carte utilisant le FPGA

La carte FPGA comporte plusieurs parties (voir ANNEXE 2) :

- Un connecteur Vidéo out permettant de connecter un signal sur 8 bits sortant du FPGA vers le CNA (Convertisseur Numérique Analogique) de la carte de conversion A/N et N/A vidéo. Cela peut permettre de visualiser le résultat de notre traitement d'image sur l'écran LCD.
- Un connecteur Vidéo in permettant de connecter un signal sur 8 bits et 3 signaux de contrôle (GATE B, HPIX, ST) sortant de la carte de conversion. Le signal sur 8 bits est une entrée vidéo venant du CAN (Convertisseur Analogique Numérique), c'est à dire l'information numérique d'une image filmée par la caméra.
  Pour le traitement d'image, il est indispensable de connaître des informations sur la vidéo d'où le besoin des signaux GATE B, HPIX et ST (voir partie de la carte conversion A/N et N/A vidéo).
- Un connecteur supplémentaire pour éventuellement communiquer avec une autre carte électronique (n'est pas utilisé pour ce projet).
- Un connecteur parallèle permettant la communication entre la carte FPGA et le logiciel Xilinx.
- Un connecteur pour alimenter la carte FPGA en continu (masse et VCC).
- Un composant FPGA de Xilinx de référence XC4003, permettant de traiter et gérer des informations en temps réels. C'est le « cerveau » de l'ensemble des cartes. Il possède 84 pattes. Voir ANNEXE 1.
- La mémoire SRAM de référence TC551001CP qui permet d'enregistrer toutes sortes d'informations comme par exemple des images. Cette mémoire de 128 Ko est contrôlée par le FPGA. Elle possède un bus d'adresse sur 17 bits, un bus de données sur 8 bits et des bits de contrôle WE (Write Enable) et OE (Output Enable).
  WE est au niveau bas pour le mode écriture. Dans ce cas les données (8 bits) sont recopiées suivant l'adresse 17 bits. OE est une entrée de commande pour mettre les sorties du bus de données à haute impédance. Par défaut, OE est à la bonne valeur.

WE est au niveau haut pour le mode lecture. La mémoire fait parvenir au FPGA les données qu'il a stockée.
On remarque que la capacité de la mémoire est de 2^17 = 131072 octets soit 128 Ko. Une image de 256*256 pixels enregistrés prend une place de 64 Ko. Il est possible alors de mémoriser deux images.

- Un réseau de 8 switchs permettant d'intervenir manuellement sur le déroulement d'un test.



3.3 Test de la carte FPGA

3.3.1 Introduction

Dans un premier temps, il est utile de tester la carte FPGA avec un système simple utilisant que cette carte afin de s'initier au logiciel.
On apprendra à construire un schéma et à le compiler dans la carte FPGA.

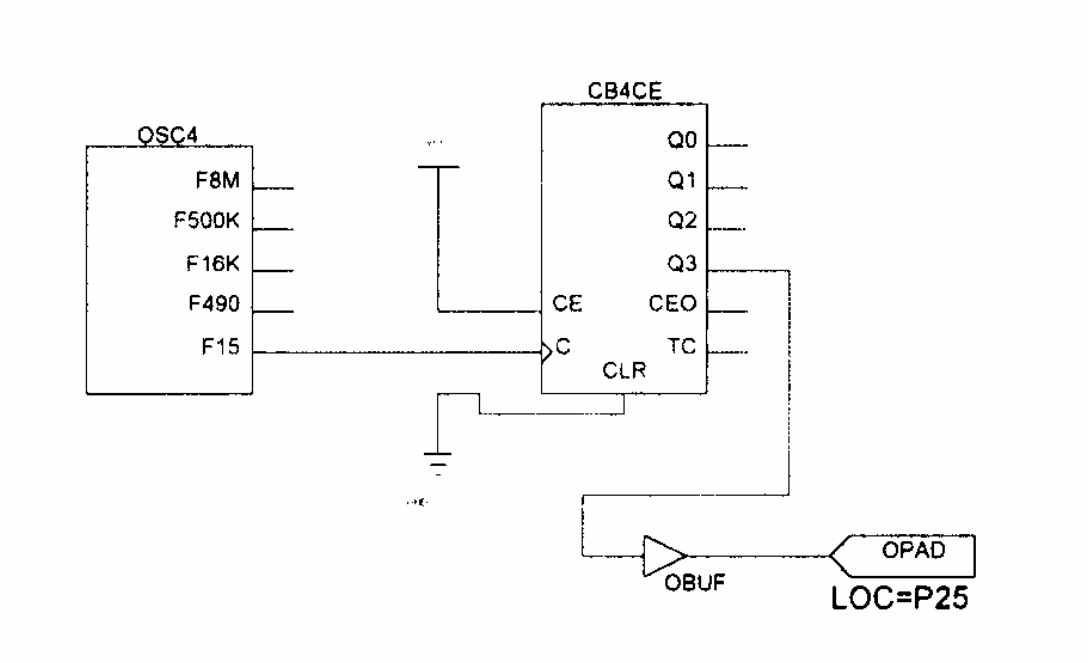


Figure 1 : schéma de clignotement d'une LED

3.3.2 Explication

Ce schéma permet de faire clignoter une LED avec un intervalle d'environ une demi seconde entre chaque changement d'état.
Pour réaliser ce schéma, on utilise les composants suivants : oscillateur, compteur, buffer de sortie et une étiquette pour affecter la sortie.
Tout d'abord, on relie l'horloge OSC4 avec la fréquence de 15 Hz au compteur CBACE sur C. L'entrée d'horloge C est actif sur font montant. On met l'entrée CE (Chip Enable) à VCC qui permet d'autoriser le fonctionnement du compteur et CLR à la masse qui est une entrée actif à l'état haut permettant de remettre à zéro le compteur. On prendra la sortie Q3 pour faire un diviseur par 16 afin qu'on est le temps d'observer le clignotement.
Cela fait une fréquence de 15/16 soit 0.94 Hz soit un changement d'état de la LED toutes les 0.53 secondes. OBUF est un buffer de sortie permettant de faire le lien entre l'intérieur et l'extérieur. Tout signal entrant dans un FPGA doit passer en premier dans un ibuff et tout signal sortant doit passer en dernier dans un obuff. L'étiquette OPAD permet d'affecter la sortie P25 du FPGA à l'endroit où se trouve la LED.

3.4 Test de vidéo en directe

3.4.1 Introduction

Il est maintenant nécessaire d'utiliser le signal vidéo venant de la carte de conversion et de le transmettre à un écran LCD.
Dans un premier temps, on visualisera l'image de la camera sur l'écran LCD et par la suite on visualisera l'image en négatif (voir schémas ci-dessous).

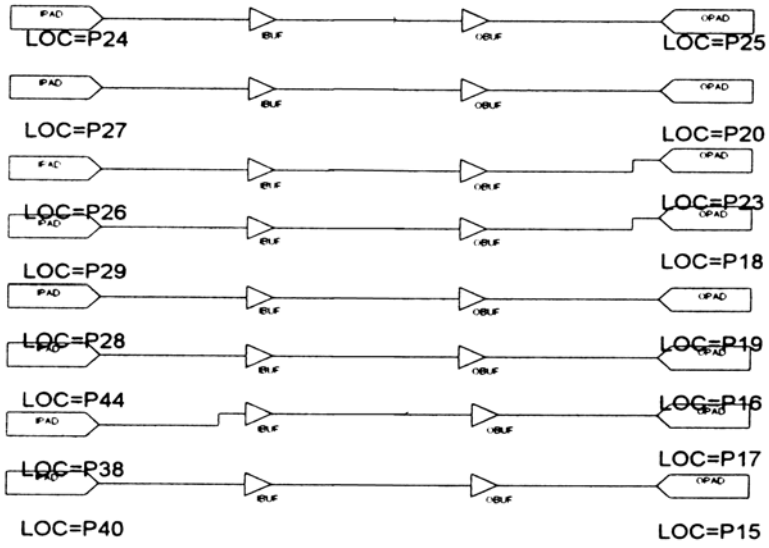


Figure 2 : schéma liaison directe

3.4.2 Explication

Il suffit de tirer des files entre les entrées et les sorties pour réaliser une liaison directe en sachant qu'il ne faut pas oublier les buffers d'entrées et de sortie, et l'affectation des étiquettes aux bonnes pattes du FPGA.

3.5 Test de vidéo en négatif

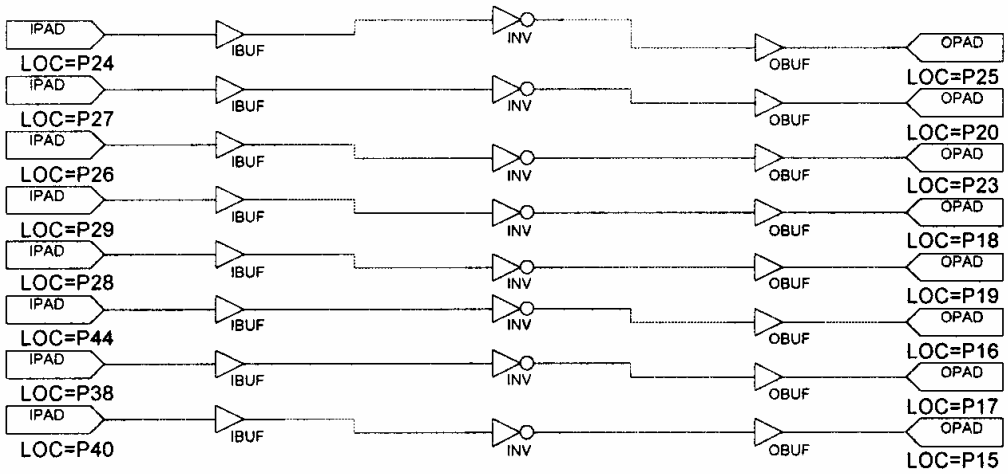


Figure 3 : schéma liaison négative

3.5.1 Explication

Ce schéma permet d'inverser le signal vidéo de 8 bits provenant de la conversion analogique/numérique (carte conversion) à la conversion numérique/analogique (carte conversion). Les niveaux de gris sont ainsi inversés. On voit sur l'écran du noir à la place du blanc et du blanc à la place du noir. Pour réaliser cette fonction, on utilise des étiquettes d'entrée et de sortie, des buffers d'entrée et de sortie, et des inverseurs.

3.6 Mémorisation dans la RAM

3.6.1 Introduction

Dans cette partie, il s'agit de tester la mémoire. On va donc mémoriser une image dans la SRAM TC551001CP. On distingue deux parties : Gestion des données et gestion des adresses

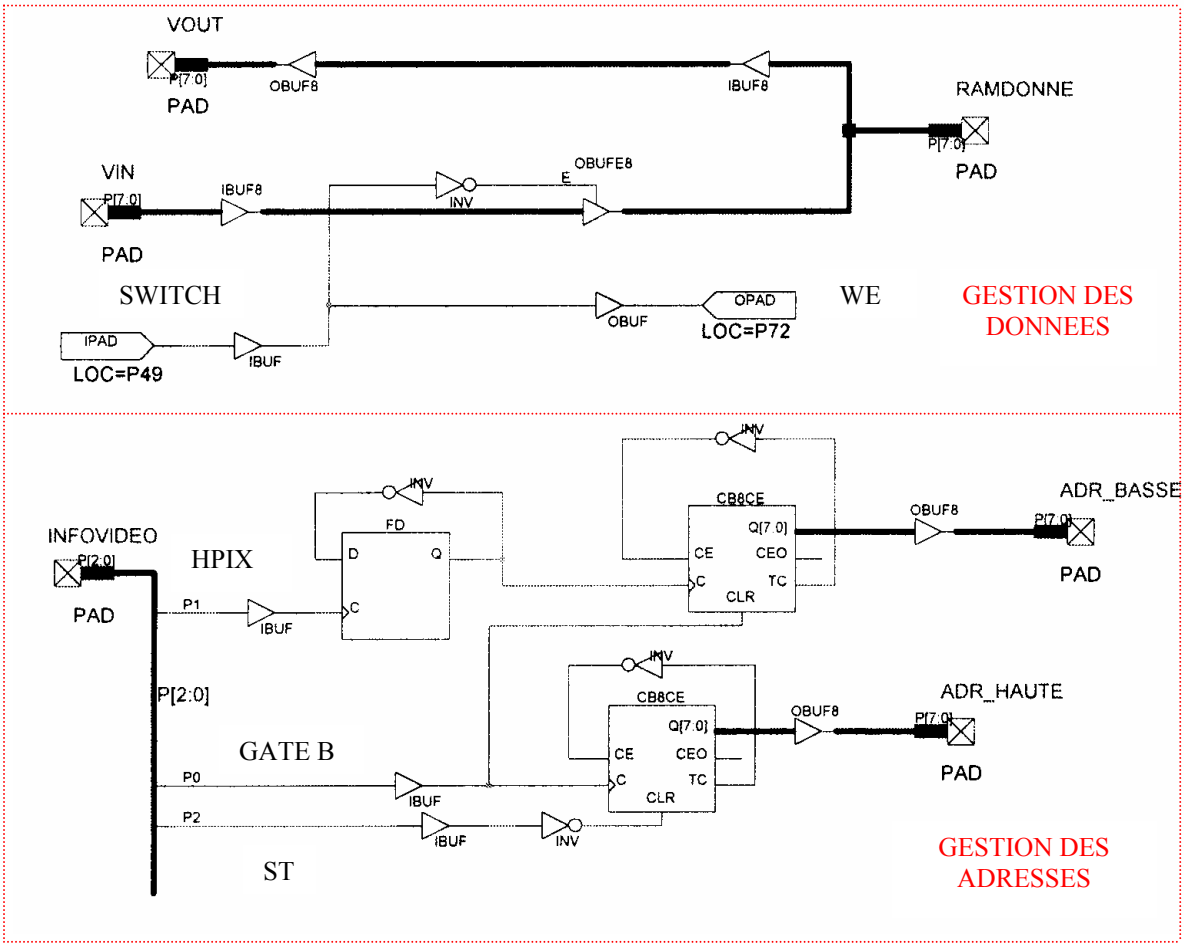


Figure 4 : schéma de mémorisation

3.6.2 Explication

Gestion des adresses :

Tout d'abord, la caméra génère un signal entrelacé sur 512 colonnes et 625 lignes. On travaillera seulement sur une image de 256 sur 256 pixels. Le but de cette partie est de générer les bonnes adresses pour la mémorisation et la lecture de l'image dans la RAM afin de profiter pleinement de la capacité du matériel. Pour réaliser cela, on prendra une colonne sur deux pour profiter de l'ensemble de l'image et les 256 premières lignes. Dans une image entrelacée les lignes qu'on reçoit sont paires ou impaires donc pour cette raison, il n'y a pas besoin de sauter une ligne sur deux (voir principe de fonctionnement).

On a besoin de sauter une colonne sur deux, on va donc diviser HPIX par 2 avec une bascule D et un inverseur. Sur chaque front montant de HPIX, Q recopie l'état de D. D est l'inverse de Q donc Q et D changent d'état. On obtient ainsi un diviseur par deux.

HPIX/2 est relié sur l'entrée d'horloge d'un compteur 8 bits. L'adresse basse va donc s'incrémenter une colonne sur deux. On a vu que 256 colonnes suffisaient, pour cela le compteur doit s'arrêter à 256 colonnes. On utilise ainsi la sortie TC envoyant un état haut quand le compteur est en débordement. TC est inversé et ainsi on met l'entrée CE (Chip Enable) à la masse permettant de bloquer le fonctionnement du compteur. Le compteur se remet à compter à la fin d'une ligne s'est à dire quand GATEB est à l'état haut. Un état bas sur CLR du compteur initialise ses sorties.

Il faut aussi incrémenter l'adresse haute quand on change de ligne. GATEB est relié à l'horloge d'un compteur de 8 bits. De même que le compteur vu précédemment, ce compteur s'arrête au bout de 256 lignes en attendant une nouvelle trame ST. ST est la synchronisation trame nous renseignant sur la présence du début d'une trame sur un état bas. On inversera donc ce signal pour réamorcer le compteur par son entrée CLR.

On obtient par le schéma gestion des adresses 256 fois 256 adresses possibles s'est à dire autant d'adresse possible qu'il a de pixel.

Gestion des données :

Le SWITCH permet d'actionner manuellement le mode lecture ou écriture de la mémoire externe TC551001CP. Lorsque le SWITCH est à l'état bas, WE (write enable) est aussi à l'état bas ce qui entraîne une autorisation d'écriture. L'inverseur provoque un état haut sur la commande du buffer de sortie OBUFE8. Celui-ci devient passant lors d'un état haut. L'entrée vidéo VIN peut alors circuler jusqu'à la RAM. Et ainsi, on enregistre une image.

Pour lire l'image mémorisée, il suffit de pousser le SWITCH à l'état haut pour mettre WE à l'état haut. La mémoire est ainsi en mode lecture. Le buffer de sortie commandé se trouve bloqué et VIN ne passe plus. Par conséquence, les données venant de la RAM (RAMDONNEE) se trouvent en liaison avec la sortie vidéo VOUT. On visualise ainsi l'image sur l'écran LCD.

Remarque : L'étiquette RAMDONNEE est déclarée comme entrée et sortie.

Conclusion

Ce système permet simplement de tester la mémoire. Il a l'inconvénient de mémoriser en permanence l'image. Lorsque qu'on pousse le SWITCH, la mémoire n'a pas fini d'enregistrer la trame entière. On a donc 2 parties de trames différentes.

3.7 Détection de mouvement

3.7.1 Comparaison entre une image enregistrée et l'image actuelle

Introduction

Ce schéma permet de comparer une image enregistrée dans la mémoire et l'image courante en temps réel dans le but de réaliser une détection de mouvement.
On retrouve dans ce schéma la partie gestion des adresses identique au schéma précédent (mémorisation dans la RAM).

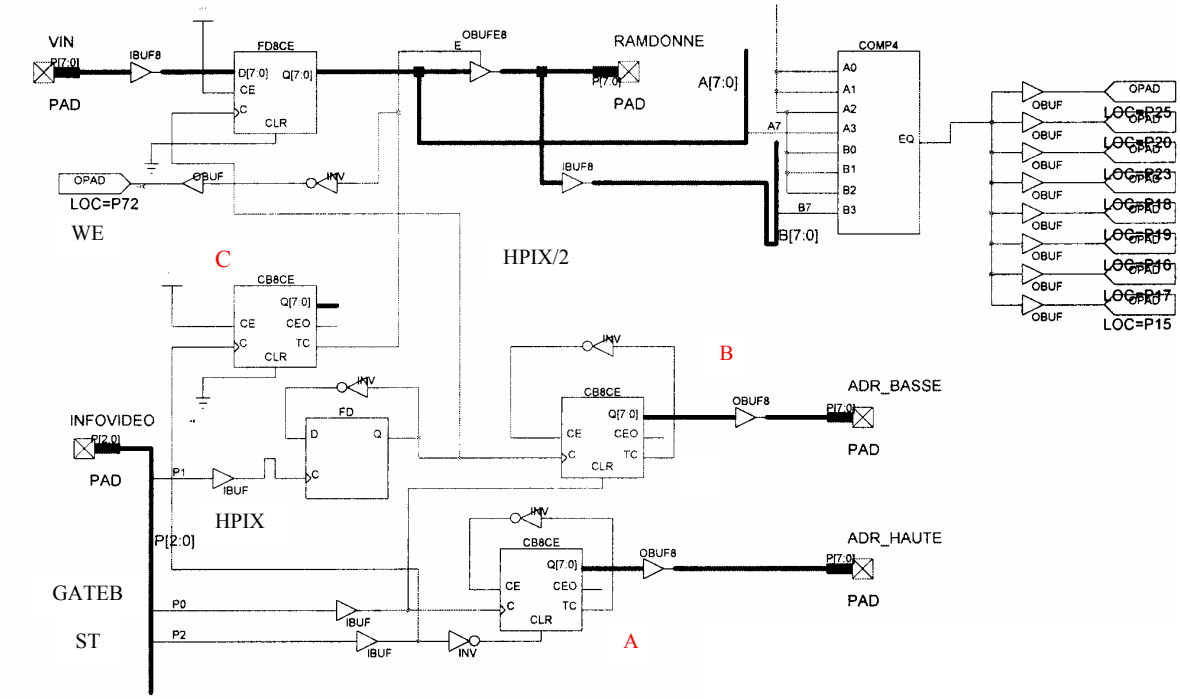


Figure 5 : schéma de comparaison

Explication

Le signal vidéo (VIN) de 8 bits passe par la bascule D (FD8CE). Cette bascule permet d'être synchronisée sur HPIX/2 afin qu'on est en sortie de celle-ci une information vidéo sautant une colonne sur deux.

Le compteur C (CB8CE) sert à compter les trames. Son entrée d'horloge C est relié à ST (synchronisation trame), CE à VCC pour valider le compteur et CLR à la masse car on ne l'utilise pas. Ce compteur 8 bits va servir à déclencher la mémorisation d'une image toutes les 256 trames, TC est connecté au buffer de sortie commandé OBUFE8. TC restera à l'état haut pendant une longueur de trame.
On aura ainsi en sortie une connexion entre l'image actuelle et la RAM (RAMDONNEE).

TC est aussi connecté au WE (actif à l'état bas) de la RAM par l'intermédiaire d'un inverseur afin de déclencher l'écriture de la mémoire.

Le comparateur COMP4 permet de comparer sur 4 bits l'image actuelle avec l'image enregistrée dans la mémoire. Il est possible de comparer sur autant de bit que l'on veut. Cela provoque un effet différent sur la comparaison. Plus on prend en compte les bits de poids faible des données vidéo, plus on prend en compte les faibles variations de niveau de gris comme les défauts de la caméra ou de scintillement. Sur le schéma, seul le bit de poids fort est comparé afin de travailler que sur les plus grosses variations de niveau de gris.
Si les deux bits de poids fort des deux images sont égaux alors la sortie EQ est à l'état haut. Et inversement si les deux bits sont différents. On connecte les 8 bits de la sortie vidéo sur le bit de sortie du comparateur. On visualise ainsi sur l'écran LCD une image sur deux niveaux de gris représentant les variations d'une image à un temps retard à une image en temps réelle.

Remarque :

Selon le rendu que l'on veut faire, on peut changer le temps de déclenchement de l'écriture dans la mémoire. Soit en utilisant un autre compteur ou soit en se connectant sur une sortie Q.

3.7.2 Soustraction avec valeur absolue

Introduction

Le schéma précédent (schéma avec comparaison) présentait des défauts pour une détection de mouvement. Il est possible d'améliorer celui-ci par une soustraction d'une image mémorisée à une image en temps réelle avec valeur absolue. S'il y a une différence entre deux images, on détecte ainsi un mouvement.

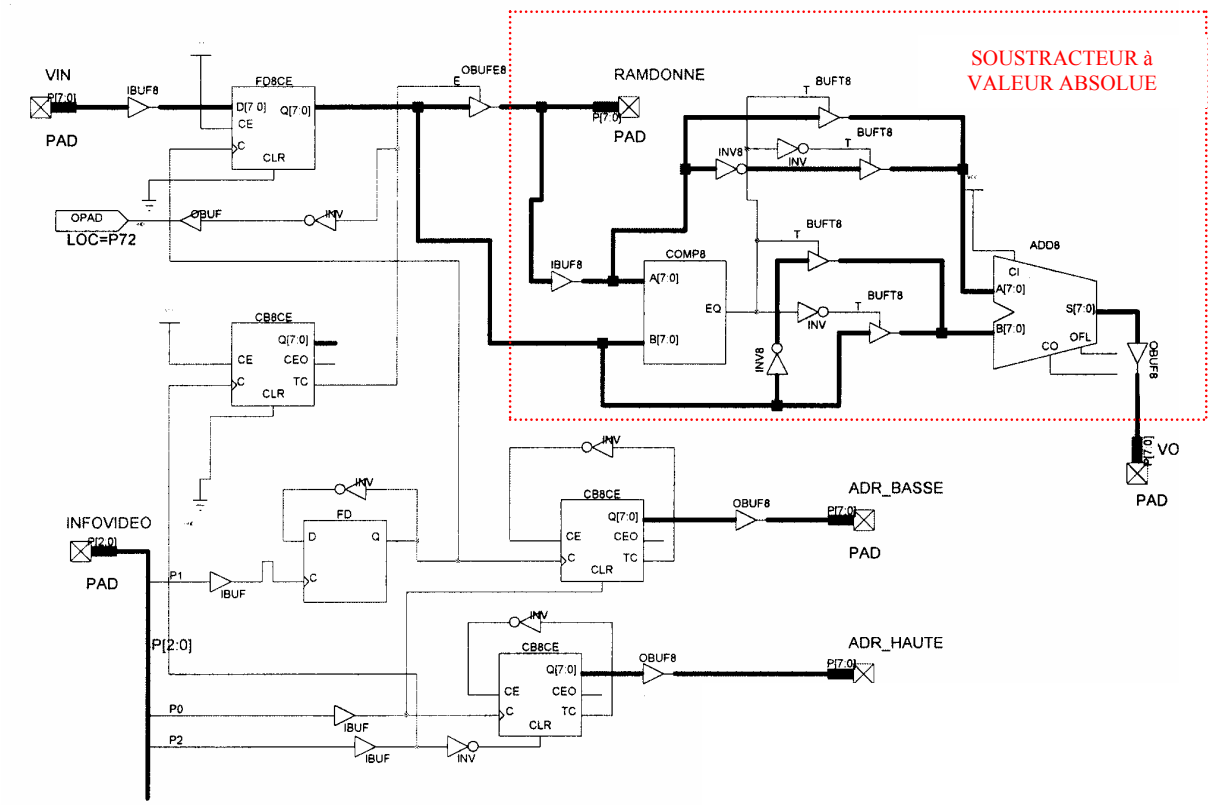


Figure 6 : schéma de détection de mouvement

Explication

L'idée du soustracteur à valeur absolue est de soustraire une image mémorisée A à une image actuelle B. Lorsque le résultat de la différence entre un pixel A et un pixel B est différent de 0 alors un mouvement est détecté.

Pour que le résultat de la soustraction soit toujours positif, il faut détecter deux cas.
Cas où A > B et cas où A < B. D'où l'utilisation d'un comparateur de deux bus de 8 bits (COMP8).
Pour réaliser la fonction de soustraction avec valeur absolue, il faut soustraire la valeur la plus grande à la valeur la plus petite.

Si A > B alors S = A – B
Si A < B alors S = B – A
Un soustracteur est alors utilisé.

Autrement dit, il est possible de faire une petite variante comme sur le schéma ci-dessus.
Si A > B alors S = A + complément à 2 de B = A + $\underline{B}$ + 1
Si A < B alors S = B + complément à 2 de A = B + $\underline{A}$ + 1
Un additionneur est ici utilisé.

Déroulement de l'opération :

Si A > B, la sortie EQ du comparateur est à l'état haut. Ce qui laisse passer A et $\underline{B}$ sur les entrées de l'additionneur par l'intermédiaire des buffers 8 bits commandés.
Par la suite, une addition est faite entre A et $\underline{B}$ avec la condition initiale CI = VCC. Ce qui ajoute 1 à l'addition.
Même raisonnement si A < B.

On peut ainsi visualiser une détection de mouvement sur l'écran LCD par la sortie VO.

Remarque

On peut changer le seuil à partir duquel une détection de mouvement se produit en changeant simplement le nombre de bits des signaux vidéo pris en compte avant la partie soustracteur avec valeur absolue. En travaillant sur les bits de poids fort, que les plus grosses variations de niveau de gris sont soustraites, et inversement. La finesse est donc réglable.

4 CARTE FPGA SPARTAN IIE

4.1 Introduction

En raison du besoin de puissance, on est passé sur une autre carte FPGA. Les FPGA est un marché en expansion, ils ne cessent d'évoluer et cela s'accompagne par un abaissement des prix. Par la suite du projet, on peut avoir besoin d'enregistrer plus d'images (2 au maximum dans l'ancienne carte), d'utiliser plus de place dans le FPGA et même utiliser plus de port pour communiquer avec l'extérieure.

4.2 Description de la carte SPARTAN IIE

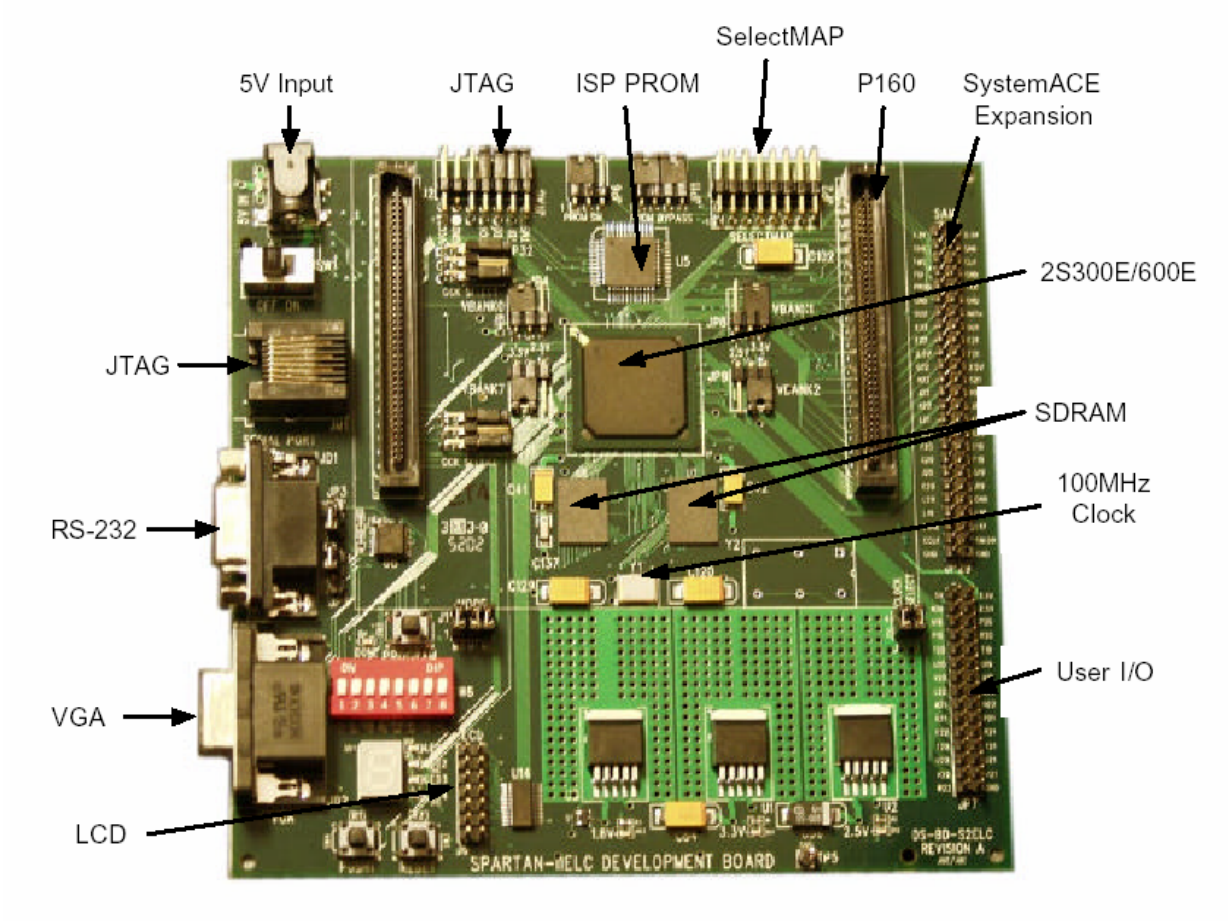


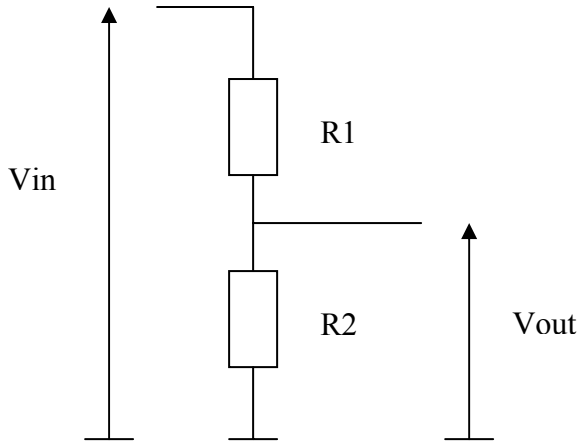
Figure 7 : carte spartan IIE

La carte comporte plusieurs parties :

- Un port JTAG pour communiquer avec le logiciel Xilinx.
- Un port RS232 pour la communication en série avec d'autres sources externes
- Un port VGA et LCD pour connecter des écrans.
- Une entrée d'alimentation en 5V.
- Un circuit intégré ISP PROM
- Un circuit FPGA 2S300E/600^{E} qui possède 300 000/600 000 portes et 82 ports I/O.
- Un connecteur double P160 pour accueillir un support de connexion de files.
- Deux mémoires SDRAM de 16 Mo
- Une source d'horloge de 100 MHz
- Un système ACE expansion
- Des connecteurs I/O
- Un port select MAP
- Des switchs et deux boutons poussoirs pour intervenir manuellement
- Des LEDs et un afficheur 7 segments
- Trois régulateurs de tension. On a le choix des niveaux de tension !

4.3 Problématique

La carte FPGA n'est pas compatible au niveau de la tension avec la carte conversion A/N et N/A. Elle n'est pas compatible dans le sens de la carte conversion vers la carte FPGA car la carte de conversion sort en 5 V et la carte FPGA ne doit pas recevoir une tension supérieure à 3.3 V sous peine de griller le composant. Dans l'autre sens, il n'y a pas de problème car le CNA comprend un état haut à partir de 2,5 V. Il faut réaliser une adaptation en tension sur 11 bits (8 bits de données + 3 bits de synchro). Pour cela on utilise des résistances montées en diviseur de tension comme sur le schéma ci-dessous.



Vin est égal au maximum à 5 V
On choisit R2 = 10 kΩ et R1 = 5.2 kΩ pour avoir Vout au maximum égal à 3,3 V.

En pratique :

Pour l'adaptation de la tension de HPIX, on a un signal trop faible en dessous du seuil de détection d'un état haut à cause du slow rate trop faible. Cela est dû peut-être à un effet de capacité en parallèle avec les résistances. On peut résoudre ce problème avec un montage simple par transistor.

4.4 Test de vidéo en direct et en négatif

Il est utile de tester la carte SPARTAN II. On fait comme sur la première carte FPGA. Dans un premier temps, on visualisera l'image de la camera sur l'écran LCD et par la suite on visualisera l'image en négatif.

Figure 8 : schéma direct

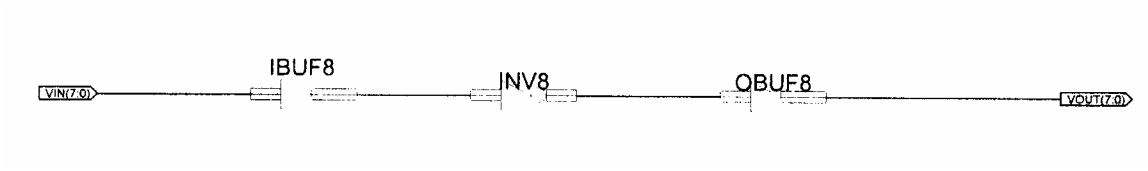


Figure 9 : schéma négatif

4.5 Détection de mouvement avec un comparateur

4.5.1 Introduction

Il existe deux façons de mémoriser, l'une dans la SRAM externe et l'autre dans la RAM interne. La mémoire interne est très insuffisante pour enregistrer une image en 256*256 pixels.
Pour commencer, les données seront mémorisées d'abord dans la RAM interne au FPGA. Comme sur la première carte FPGA, on fait une détection de mouvement avec un comparateur. Le schéma est différent pour la mémorisation dans la mémoire interne.

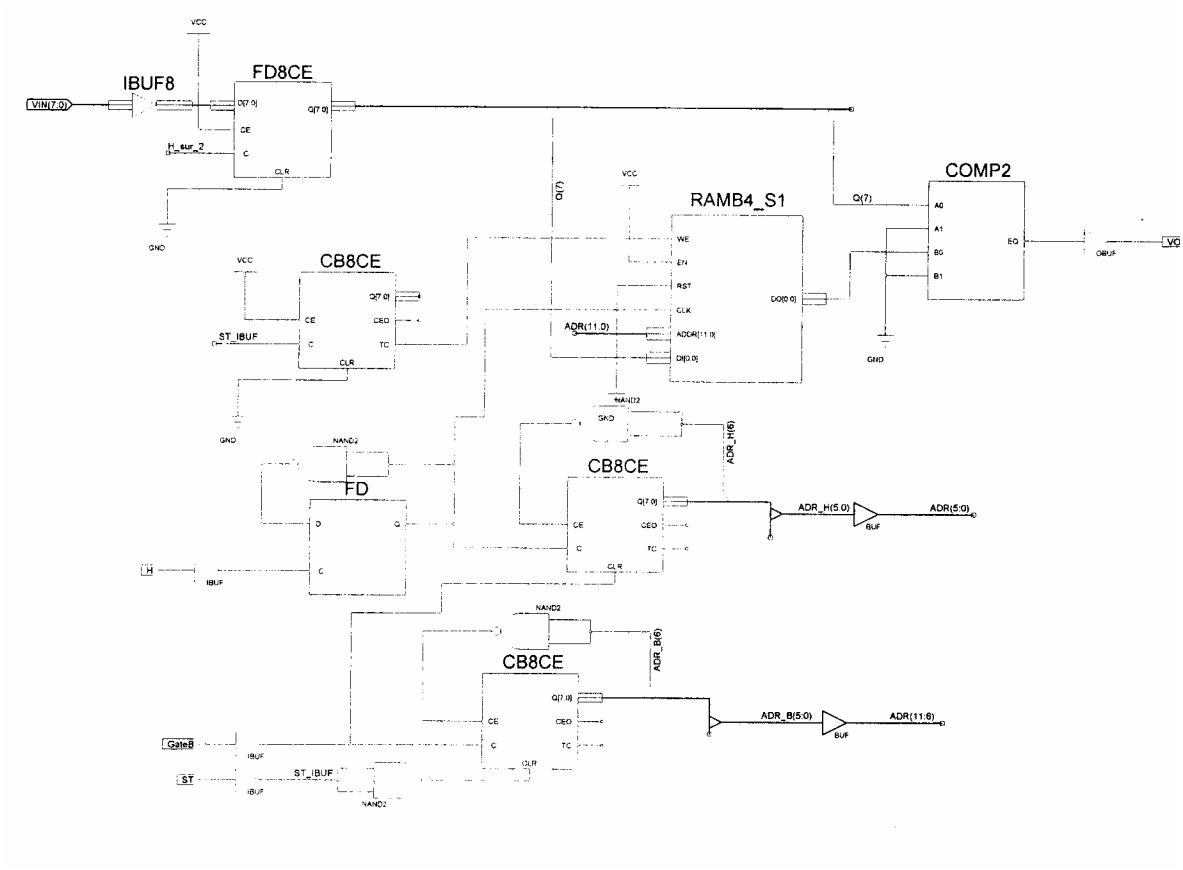


Figure 10 : détection de mouvement avec comparateur

4.5.2 Explication

La mémoire RAMB4_S1 a été choisie pour ses 12 bits d'adresse et son bit de donnée.
Le WE de la RAMB4_S1 est relié à un compteur de trame. Le CLK à HPIX/2 pour se synchroniser, lors de la lecture dans la RAM toutes les deux colonnes et lors de l'écriture. EN (Enable) est une entrée mise à l'état haut pour valider le circuit. RST (Reset) est mis à la masse car on n'utilise pas pour l'instant la remise à 0.
On compare une image mémorisée (1 bit par pixel) à une image (1 bit par pixel) courante.

4.6 Test de la RAM interne en VHDL

Comme on n'arrivait pas à mémoriser en schématique (à cause du problème sur HPIX), on a essayé de mémorisation en langage VHDL afin de savoir si cela venait du composant RAMB4_S1.
Un petit programme VHDL a été réalisé (voir ANNEXE 3). Ce programme réalise une mémorisation des états de 7 switchs en actionnant le dernier switch. Ensuite lors de la commutation de ce dernier, la mémoire passe en mode lecture et affiche sur les 7 segments de l'afficheur l'état des positions des switchs enregistrées. Cela a permis de vérifier le bon fonctionnement de l'enregistrement dans la mémoire interne.

5 CONCLUSION

Ce projet, nous a permis de faire une initiation au traitement d'image en noir et blanc et d'utiliser deux cartes FPGA. C'est une application complète et concrète dans le domaine de l'image. Ce projet à été très enrichissant : d'un côté nous avons découvert les composants programmables et leur programmation (FPGA de chez Xilinx), de l'autre, nous avons développer notre capacité à construire un projet et à réaliser ses différentes étapes.

Nous avons le regret de ne pas avoir eu un véritable cours sur l'image en début d'année, et d'avoir changer de carte FPGA en plein milieu du projet. Ce changement nous a fait prendre du retard et donc nous n'avons pas pu répondre entièrement au cahier des charges. Nous n'avons pas exploité la puissance de la carte SPARTAN II.
Par conséquent, on a l'avantage d'avoir étudier deux cartes FPGA différentes, ce qui a développer notre capacité d'adaptation à une autre carte FPGA.

ANNEXE 1

Les FPGA:

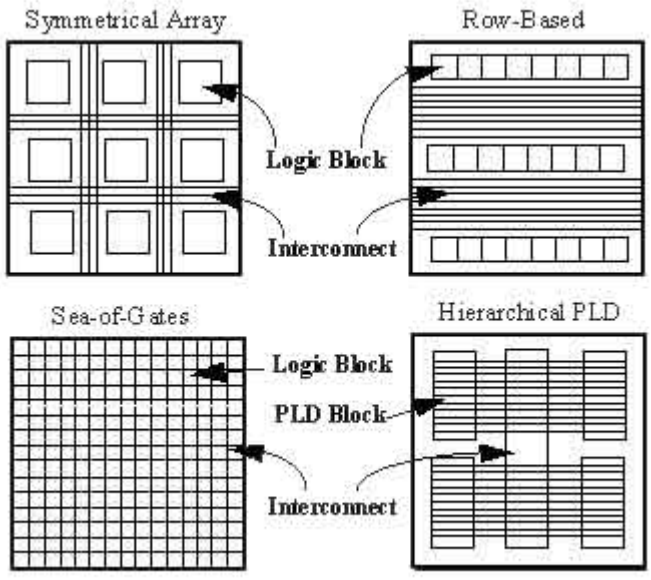
Tous les mots en rouge dans le texte sont des liens vers le glossaire.

Présentation rapide: Les FPGA (Field Programmable Gate Arrays ou "réseaux logiques programmables") sont des composants **VLSI** entièrement reconfigurables ce qui permet de les reprogrammer à volonté afin d'accélérer notablement certaines phases de calculs.
L'avantage de ce genre de circuit est sa grande souplesse qui permet de les réutiliser à volonté dans des algorithmes différents en un temps très court.
Le progrès de ces technologies permet de faire des composants toujours plus rapides et à plus haute intégration, ce qui permet de programmer des applications importantes.

Introduction: Les circuits FPGA sont constitués d'une matrice de blocs logiques programmables entourés de blocs d'entrée sortie programmable.
L'ensemble est relié par un réseau d'interconnexions programmable.
Les FPGA sont bien distincts des autres familles de circuits programmables tout en offrant le plus haut niveau d'intégration logique.

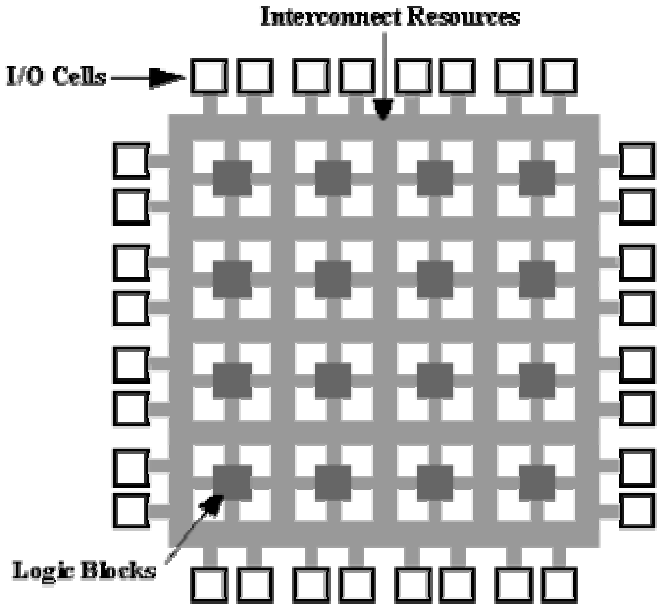
Il y a 4 principales catégories disponible commercialement:

- Tableau symétrique.
- En colonne.
- Mers de portes.
- Les PLD hiérarchique.



Les différentes classes de FPGA.

Voici la structure interne d'un FPGA de type matrice symétrique. Il s'agit de l'architecture que l'on retrouve dans les FPGA de la série XC4000 de chez Xilinx.



Structure interne d'un FPGA

L'utilisateur peut programmer la fonction réalisée par chaque cellule (appelée CLB par Xilinx: Configurable Logic Block):

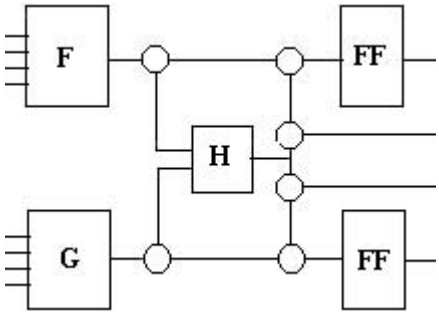
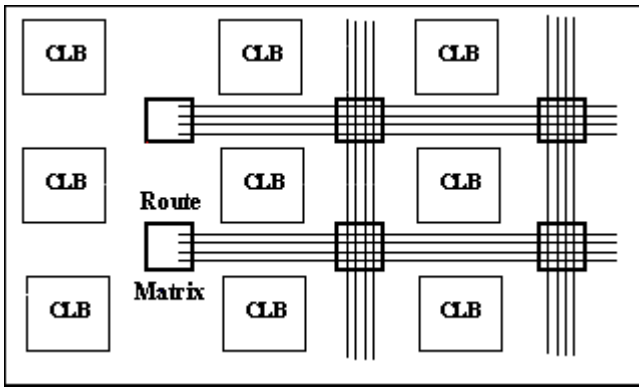


Schéma bloc d'une cellule

On programme aussi les interconnexions entre les cellules:



Les interconnexions entre les cellule d'un FPGA

Anisi que les entrées et sorties du circuit. L'avantage des FPGA est de pouvoir être configuré sur place, sans envoi du circuit chez le fabricant, ce qui permet de les utiliser quelques minutes après leur conceptions. Les FPGA les plus récents sont configurables en une centaine de millisecondes. Les FPGA sont utilisés pour un développement rapide et bon marché des **ASIC.**

Inventés par la société Xilinx,le FPGA, dans la famille des **ASICs**, se situe entre les **réseaux logiques programmables** et les **prédiffusés**. C'est donc un composant standard combinant la densité et les performances d'un prédiffusé avec la souplesse due à la reprogrammation des PLD. Cette configuration évite le passage chez le fondeur et tous les inconvénients qui en découlent.

Glossaire:

ASIC: Application-Specific Integrated Circuits. Les ASIC fournissent précisement la fonctionnalité nécessaire pour une tâche spécifique. Les ASIC sont conçus pour une tâche bien précise et cela leur permet d'être plus petits, moins chers, plus rapide et de consommer moins de puissance qu'un processeur programmable (tel un Intel© Pentium IV).
Une puce graphique faite sur mesure pour PC, par exemple, peut tracer des lignes ou des images 10 à 100 fois plus rapidement qu'un processeur central à usage générique.

Design asynchrone: S'oppose au design synchrone. Prohibé par les règles de l'art de la conception des ASIC et des FPGA. Ce manque de synchronisme signifie que la sortie de certains bistables ne change pas sur un flanc montant de l'horloge mais à n'importe quel moment. Dès lors, en fonction du routage, les délais entre cellules étant différents, le fonctionnement du circuit risque de varier lui aussi! Ceci peut être très gênant si des précautions particulières ne sont pas prises. De plus, la compréhension du schéma en est compliquée. Les symptômes d'un circuit asynchrone sont les Gated Clocks, les resets et sets asynchrones, l'horloge mère comme entrée des portes logiques.

Les circuits semi-personnalisés: Les semi-personnalisés sont des réseaux prédéfinis de transistors ou de fonctions logiques qui nécessitent une personnalisation de l'utilisateur pour réaliser la fonction désirée. Cette famille comprend :

- les réseaux logiques programmables,
- les réseaux prédiffusés.

Les réseaux logiques programmables: Ce composant ne nécessite aucune étape technologique supplémentaire pour être personnalisé. Nous y trouvons les PAL/PLD, ce sont des circuits standards programmables par l'utilisateur grâce à différents outils de développement. La programmation consiste à établir des connexions en imposant un courant supérieur aux courants de fonctionnement normaux (claquage de fusibles ou de jonctions). Les circuits logiques programmables incluent un grand nombre de solutions, toutes basées sur des variantes de l'architecture des portes ET OU.

Nous y trouvons :

- PAL (Programmable Array Logic) matrice ET programmable, matrice OU figée),
- PLA (Programmable Logic Array) matrice ET ou matrice OU programmable,
- EPLD (Erasable PLD) effaçables par rayons ultraviolet, ils peuvent être reprogrammer,
- EEPLD (Electrically Erasable PLD) programmables et effaçables

électriquement, ils peuvent être reprogrammés sur site. Les limites de l'architecture du PLD résident dans le nombre de bascules, le nombre de signaux d'entrées/sorties, la rigidité du plan logique ET OU et des interconnexions. Précisons que ces composants très souples d'emploi sont limités à des fonctions numériques et adaptés à des productions de petites séries et ne présentent aucune garantie quant à la confidentialité.

Fonction combinatoire: sa sortie ne dépend que des états présents de ses entrées et non des ses états passés.

Fonction séquentielle: ses sorties dépendent à la fois des états présents et des états passés de ses entrées.

Les prédiffusés: Les réseaux prédiffusés sont des circuits partiellement préfabriqués. L'ensemble des éléments (transistors, diodes, résistances, capacités, etc.) est déjà implanté sur le circuit suivant une certaine topologie, mais les éléments ne sont pas connectés entre eux (sauf au niveau diffusion). La réalisation des connexions dans le but de définir la fonction souhaitée est la tâche du concepteur, pour cela il dispose de bibliothèques de macrocellules et d'outils logiciels d'aide à la conception. A partir de cette liste d'interconnexions (netlist) le fondeur n'aura que quelques étapes technologiques à effectuer pour achever le circuit, c'est à dire le dépôt d'une ou plusieurs couches de métallisation.

Cette technique est intéressante sur le plan de la conception et de la fabrication, par contre elle présente l'inconvénient de ne pas permettre une optimisation en terme de densité de composants puisque les éléments de base sont préimplantés et pas forcément utilisés et que leur positionnement a priori n'est pas forcément optimal pour le but recherché.

Les circuits personnalisés: Ce sont des circuits non préfabriqués. Pour chaque application on optimise le circuit intégré, ce qui conduit à la création de son propre composant. Cette famille comprend :

- les circuits à la demande
- les circuits précaractérisés

Les circuits à la demande: Les solutions circuit à la demande (qu'on appelle full custom en anglais) présentent l'avantage d'autoriser une meilleure optimisation duplacement puisque celui-ci n'est pas prédéfini. On dispose d'une bibliothèque de modèles mathématiques de comportement et via un "compilateur de silicium" logiciel très sophistiqué on peut concevoir toute l'architecture du circuit en faire une validation logicielle (simulation logique) puis dans une avant dernière étape en déduire le dessin des divers masques de fabrication.

Toutes les opérations, de la conception à la fabrication, sont effectuées de façon spécifique adaptées aux exigences de l'utilisation. L'ensemble des critères techniques est au choix du concepteur, que ce soit la taille du composant, le nombre de broches, le placement du moindre transistor. C'est l'ASIC le plus optimisé car aucune contrainte ne lui est imposée. Le placement des blocs

fonctionnels et le routage des interconnexions, même si ces opérations sont assistées par ordinateur, sont effectuées avec beaucoup plus d'interventions manuelles pour atteindre l'optimisation au niveau de chaque transistor. Cependant, les phases de mise au point sont longues et onéreuses, il va de soi que la rentabilisation des investissements de développement nécessite un fort volume de production.

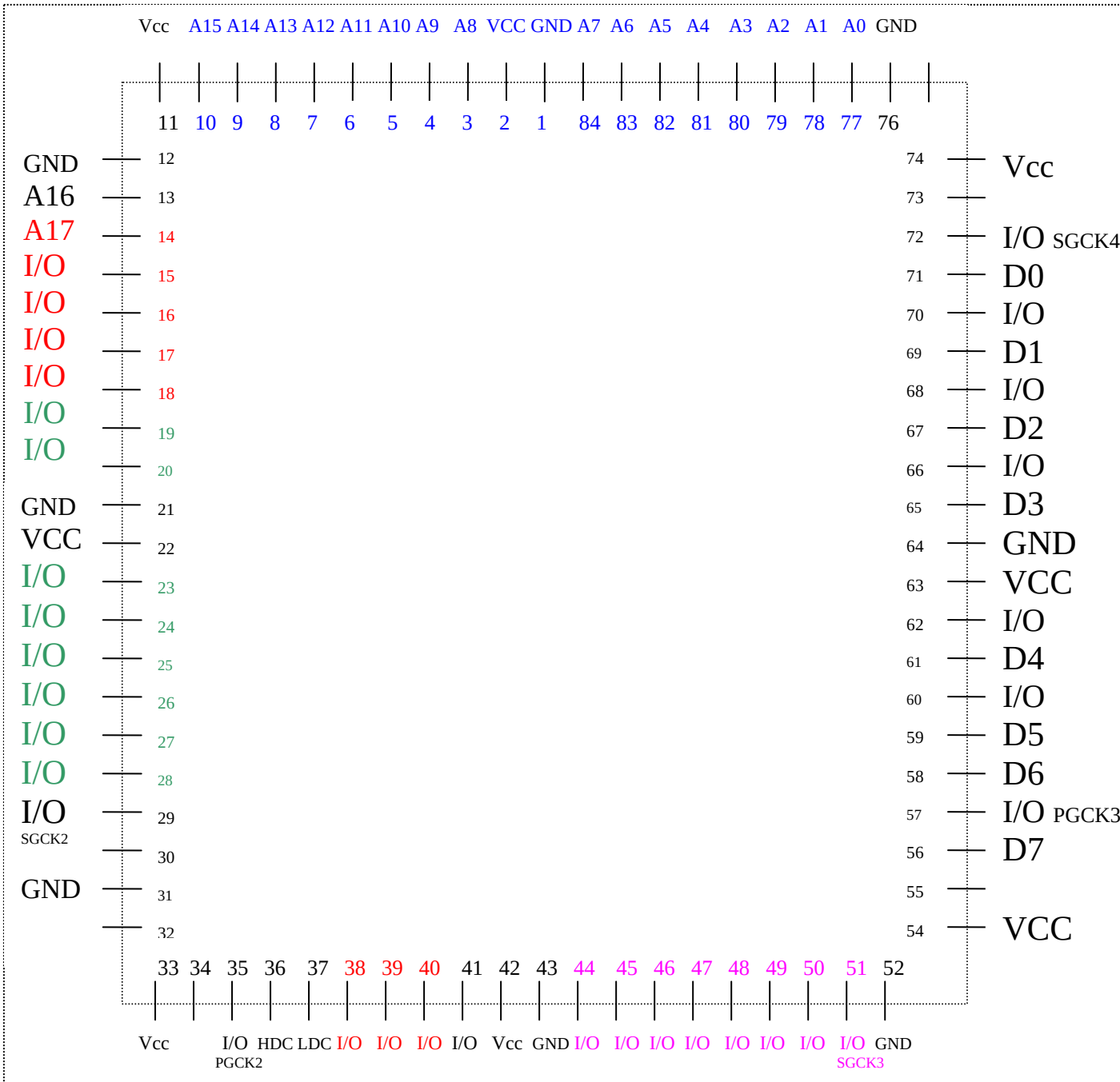
Les circuits précaractérisés: Pour la réalisation de circuits précaractérisés on dispose d'une bibliothèque de circuits élémentaires que le fondeur sait fabriquer et dont il peut garantir les caractéristiques et on les associe pour réaliser le circuit à la demande. Ici encore on utilisera en fabrication des masques personnalisés pour chacune des couches diffusées et des métallisations.

Le concept est très semblable de celui des circuits à la demande. La seule différence réside dans la réalisation du schéma puisque l'on accède à une bibliothèque de cellules prédéfinies générant de très nombreuses fonctions élémentaires ou élaborées. Cette dernière constitue un véritable catalogue dans lequel le concepteur se sert pour constituer son schéma. Il existe trois types de cellules :

- les cellules standards (standard cells) correspondent à la logique classique,
- les mégacellules (megacells) peuvent être des blocs du type microprocesseur, périphérique,
- les cellules compilées (compilable cells) dont les blocs RAM ou ROM. Il est nécessaire de personnaliser complètement la diffusion, et par conséquent de créer tous les masques. Cependant un avantage évident en découle : alors qu'il est impossible avec les prédiffusés d'utiliser à 100% le réseau de cellules ou portes, ce qui se traduit par une perte de silicium, les précaractérisés permettent d'exploiter complètement la surface du circuit.

VLSI: Very Large Scale Integration: circuits intégrés à très haut niveau d'intégration. Typiquement, on estime qu'un circuit intégré VLSI comprend entre 10.000 et 99.999 portes. Néanmoins, on utilise aussi cette appelation pour les circuits intégrés comprenant plus de portes.

ANNEXE 2



Adresse vers RAM
Switches
Vidéo Data Out
Vidéo In

On utilise 62 broches pour l’ensemble :

-RAM : on a besoin de 27 broches :
(A0 ….. A16) adresse vers RAM
8 bits de données I/O vers RAM
WE
OE ;

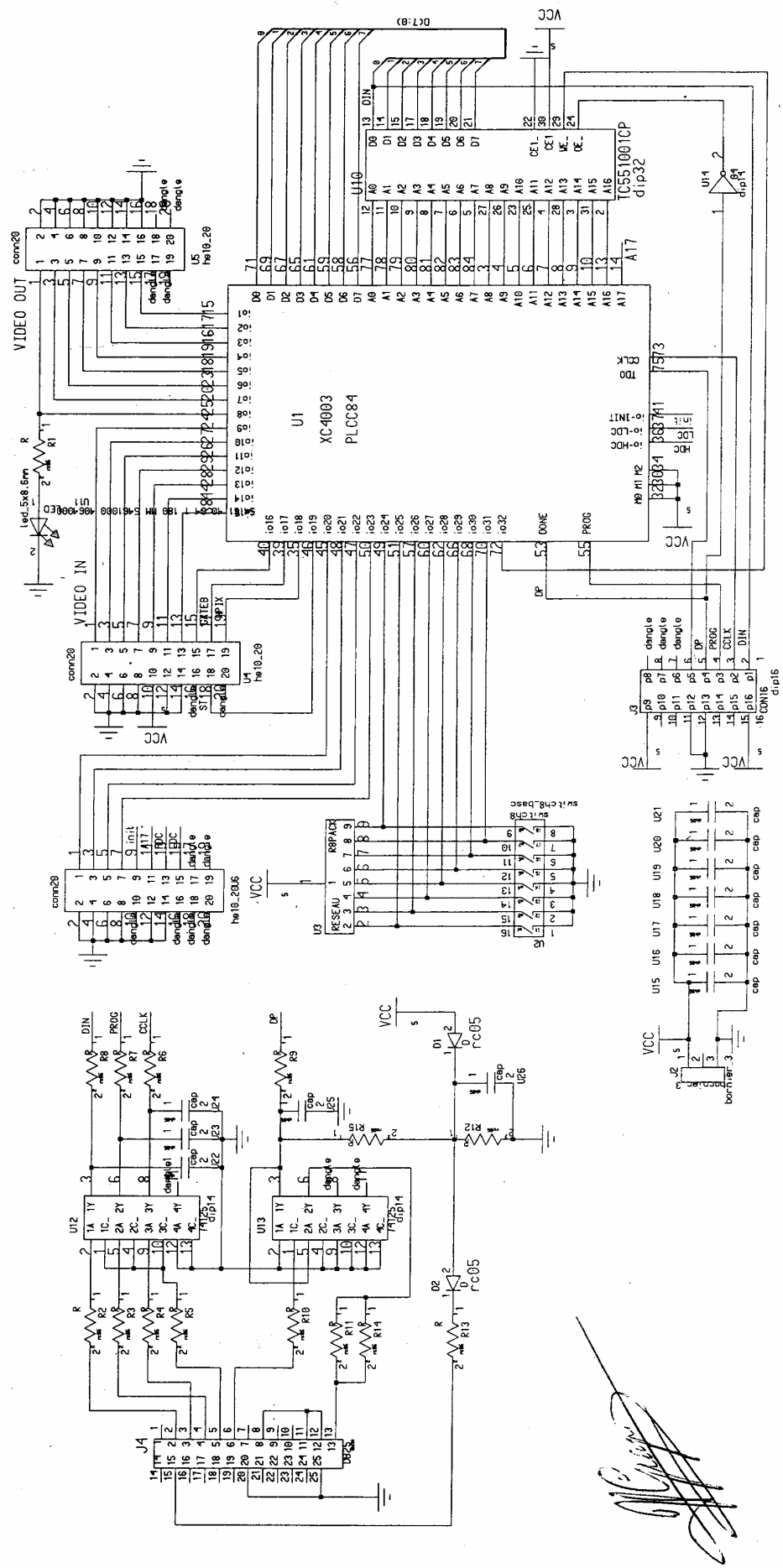
-DATA IN : on a besoin de 11 broches :
8 bits pour Vidéo IN
Synchro Ligne
Synchro Trame
Horloge Pixel ;

-Vidéo OUT : on a besoin de 8 broches :
8 bits pour Vidéo OUT ;

-8 broches :
une sortie sur 8 bits pour la commande d’un moteur (un bit pour une LED par exemple) ;

-8 broches :
8 switches pour le paramétrage du traitement.

Schéma structurel de la première carte FPGA :



ANNEXE 3

Carte FPGA SPARTAN II

Entrée :

Pin FPGA	Expansion slot	Connecteur J6	
A9	RIOA 5	8	D0
A10	RIOA 7	10	D1
D8	RIOA 9	12	D2
D9	RIOA 11	14	D3
D10	RIOA 13	16	D4
D11	RIOA 15	18	D5
F11	RIOA 17	20	D6
F12	RIOA 19	22	D7

Sortie :

Pin FPGA	Expansion slot	Connecteur J3	
A7	RIOA 1	5	D0
B8	RIOA 2	7	D1
A6	LIOB 9	9	D2
A5	LIOB 11	11	D3
A4	LIOB 13	13	D4
A3	LIOB 15	15	D5
D3	LIOB 17	17	D6
F4	LIOB 19	19	D7
G4	LIOB 21	21	3Y (Synchro Trame)
H4	LIOB 23	23	2Q (GATE B)
J4	LIOB 25	25	2Y (Quartz)

Test de la RAM interne en VHDL

RAM.vhd

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

--  Uncomment the following lines to use the declarations that are
--  provided for instantiating Xilinx primitive components.
--library UNISIM;
--use UNISIM.VComponents.all;

entity ram is
   Port ( video_IN : in std_logic_vector(7 downto 0);
          video_OUT : out std_logic_vector(7 downto 0);
          ST : in std_logic;
          SL : in std_logic;
          Q : in std_logic;

                  switch : in std_logic_vector(6 downto 0);
                  switch_RW,CLK_CAN : in std_logic;
                  segments : out std_logic_vector(6 downto 0));
end ram;

architecture architecture_ram of ram is

signal tempO, tempI : std_logic_vector(7 downto 0);


-- Component Declaration for RAMB4_Sn
-- Should be placed after architecture statement but before begin keyword


component RAMB4_S8
port (DO : out STD_LOGIC_VECTOR (7 downto 0);
          ADDR : in STD_LOGIC_VECTOR (8 downto 0);
          CLK : in STD_ULOGIC;
          DI : in STD_LOGIC_VECTOR (7 downto 0);
          EN : in STD_ULOGIC;
          RST : in STD_ULOGIC;
          WE : in STD_ULOGIC);
end component;

begin
-- Component Attribute Specification for RAMB4_Sn
-- Should be placed after architecture declaration but before the begin keyword
-- Put attributes, if necessary
-- Component Instantiation for RAMB4_Sn
```

```
-- Should be placed in architecture after the begin keyword
RAMB4_S1_INSTANCE_NAME : RAMB4_S8
-- synthesis translate_off
-- synopsys translate_on
port map (DO => tempO,
        ADDR => "000000001",
        CLK => CLK_CAN,
        DI => tempI,
        EN => '1',
        RST => '0',
        WE => switch_RW);

segments <= tempO(6 downto 0);
tempI(6 downto 0) <= switch(6 downto 0);
tempI(7) <= '0';

end architecture_ram;
```

location.ucf

```
--Le fichier ucf permet d'associer des noms de signaux à des ports du FPGA

#NET ST LOC = G4;
#NET SL LOC = H4;
#NET Q LOC = J4;

#NET video_OUT<0> LOC = A9;
#NET video_OUT<1> LOC = A10;
#NET video_OUT<2> LOC = D8;
#NET video_OUT<3> LOC = D9;
#NET video_OUT<4> LOC = D10;
#NET video_OUT<5> LOC = D11;
#NET video_OUT<6> LOC = F11;
#NET video_OUT<7> LOC = F12;

#NET video_IN<0> LOC = A7;
#NET video_IN<1> LOC = B8;
#NET video_IN<2> LOC = A6;
#NET video_IN<3> LOC = A5;
#NET video_IN<4> LOC = A4;
#NET video_IN<5> LOC = A3;
#NET video_IN<6> LOC = D3;
#NET video_IN<7> LOC = F4;


NET switch<0> LOC = H2;
NET switch<0> pullup;
NET switch<1> LOC = H1;
NET switch<1> pullup;
```

```
NET switch<2> LOC = J2;
NET switch<2> pullup;
NET switch<3> LOC = J1;
NET switch<3> pullup;
NET switch<4> LOC = K2;
NET switch<4> pullup;
NET switch<5> LOC = K1;
NET switch<5> pullup;
NET switch<6> LOC = L2;
NET switch<6> pullup;

NET switch_RW LOC = L1;
NET switch_RW pullup;


NET segments<0> LOC = V3;
NET segments<1> LOC = V4;
NET segments<2> LOC = W3;
NET segments<3> LOC = T4;
NET segments<4> LOC = T3;
NET segments<5> LOC = U3;
NET segments<6> LOC = U4;

NET CLK_CAN LOC = AA12;
```