Houssein MARIAM
Thomas ANGENAULT

Sujet de projet : réalisation d'une IP de traitement d'image sur cible Zynq7020 et affichage VGA, étape intermédiaire.

I- Introduction

La norme VGA (Video Graphics Array) est une norme d'affichage vidéo couramment utilisée dans les ordinateurs et les écrans. Elle définit les spécifications pour la résolution, la fréquence de rafraîchissement et les signaux nécessaires pour afficher des images sur un moniteur VGA. La norme VGA prend en charge une résolution maximale de 640x480 pixels et une palette de couleurs de 16 couleurs. Elle utilise des signaux analogiques pour transmettre les informations d'image, y compris les signaux de synchronisation horizontale (HSYNC) et verticale (VSYNC) pour synchroniser l'affichage, ainsi que les signaux de couleur (rouge, vert et bleu) pour déterminer les niveaux de luminosité des pixels à l'écran. La norme VGA a été largement adoptée et est devenue une référence dans l'industrie pour les connexions d'affichage analogiques.

Le signal de synchronisation horizontale (HSYNC) indique le début et la fin de chaque ligne d'images sur l'écran, assurant une transition fluide entre les lignes. Il permet de maintenir une organisation cohérente de l'affichage. D'autre part, le signal de synchronisation verticale (VSYNC) indique le début et la fin de chaque image complète, marquant ainsi le passage d'une image à la suivante. Cela garantit une transition fluide entre les images et la stabilité verticale de l'affichage.

II- Norme VGA

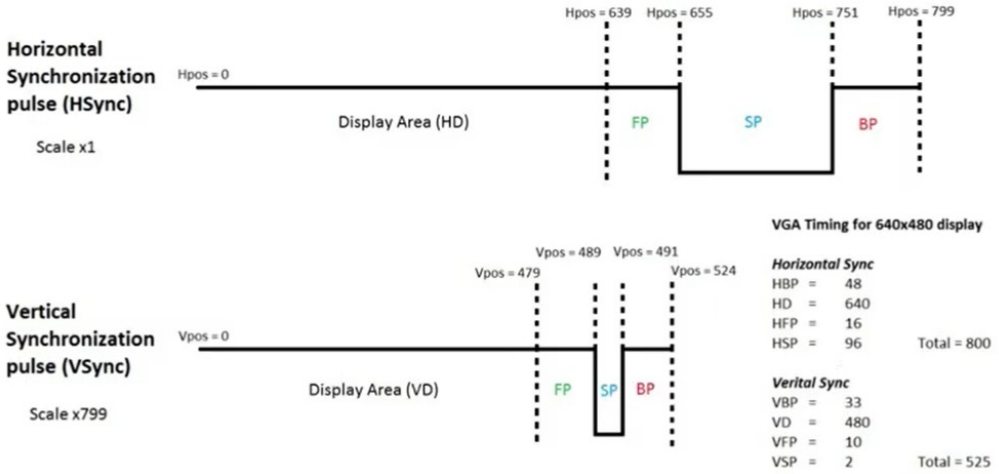
Avant de nous plonger dans les détails de la norme VGA, il est important de souligner que chaque image est essentiellement une matrice de pixels. Chaque pixel est constitué de trois composantes de couleur : rouge, vert et bleu. Ces pixels sont ensuite transmis à un écran où ils sont affichés selon un ordre spécifique, qui est expliqué dans le paragraphe 1.1.

La norme VGA implémente cinq signaux distincts pour son fonctionnement. Tout d'abord, il y a trois signaux dédiés à la représentation des couleurs (rouge, vert, bleu) et deux signaux de synchronisation, hsync (synchronisation horizontale) et vsync (synchronisation verticale).

1.1 Signaux de synchronisation

Les deux signaux de synchronisation sont essentiels pour assurer une synchronisation précise entre l'image et l'écran. En agissant comme des émetteurs et des récepteurs, ils permettent une coordination harmonieuse entre le générateur VGA (FPGA dans notre cas) et le moniteur (l'écran), garantissant ainsi un affichage correct des images et une expérience visuelle cohérente. Le signal hsync est utilisé pour synchroniser chaque ligne de l'image, tandis que le signal vsync assure la synchronisation de l'image dans son ensemble.

La figure suivante offre un aperçu synthétique des principales périodes liées aux signaux de synchronisation horizontale et verticale :



Lorsqu'une impulsion du signal HSYNC survient (SP), cela marque la fin d'une ligne et prépare le balayage de la ligne suivante. De même, lorsqu'une impulsion du signal VSYNC se produit, cela indique la fin du balayage de toute la zone et prépare un nouveau balayage à partir du coin supérieur gauche. Ces impulsions sont régies par la norme qui spécifie également leur durée, qui peut également être exprimée en nombre de pixels balayés de manière équivalente.

En raison de considérations historiques, il existe un délai entre l'envoi du dernier pixel d'une ligne et le début de la ligne suivante. C'est pendant cette période que l'impulsion de synchronisation horizontale est générée sur le signal HSYNCH. De même, pour la synchronisation Vsync, il y a un délai entre l'envoi du dernier pixel d'une image et le début de l'envoi du premier pixel de la nouvelle image, à partir du coin supérieur gauche. C'est pourquoi on distingue les pixels visibles (zone d'affichage ou « Display Area » en anglais) des pixels invisibles (zone FP +SP +BP).

Les informations nécessaires sont incorporées dans l'image elle-même. Étant donné que chaque pixel prend un certain temps pour être transmis, nous introduisons des pixels fictifs dans l'image active pour tenir compte du délai de transition entre deux lignes ou deux images. Par conséquent, pour une image active de 640 x 480 pixels, en termes de synchronisation, nous envoyons une image de 800 x 521 pixels.

1.2 Dimensions et timing

Pour la norme VGA, nous dimensionnerons l'image à afficher à 640x480 pixels; et l'image complète transmise sera de 800x521 pixels. Avec une fréquence de rafraîchissement de l'image par l'écran de 60 Hz, soit $\frac{1}{60} = 16.7\,ms$ pour afficher $800 \times 521 = 416800$ pixels. Le temps pour afficher un pixel sera donc de $\frac{0{,}0167}{416800} = 40.10^{-9}\,s = 40\,ns$, soit une fréquence de 25 MHz.

Nous retrouvons les valeurs suivantes, définies par la norme VGA :

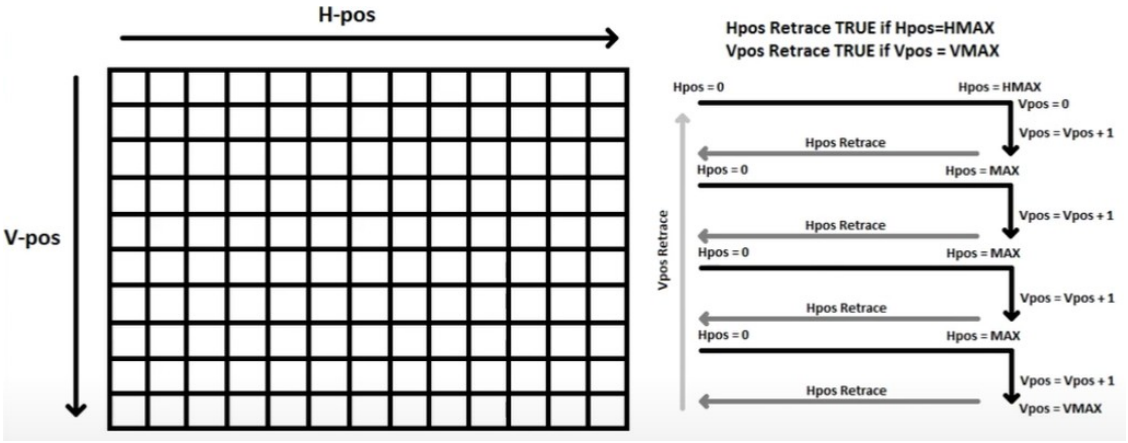
Paramètre		Synchronisation verticale		Synchronisation horizontale	
Symbole	Description	Durée	Lignes	Durée	Colonnes
T_S	Synch pulse	16,7ms	521	32us	800
T_{Disp}	Display time	15,36ms	480	25,6us	640

T_{SP}	Sync Pulse	64us	2	3,84us	96
T_{FP}	Front Porch	320us	10	640ns	16
T_{BP}	Back Porch	928us	29	1,92us	48

1.3 Envoie de l'image

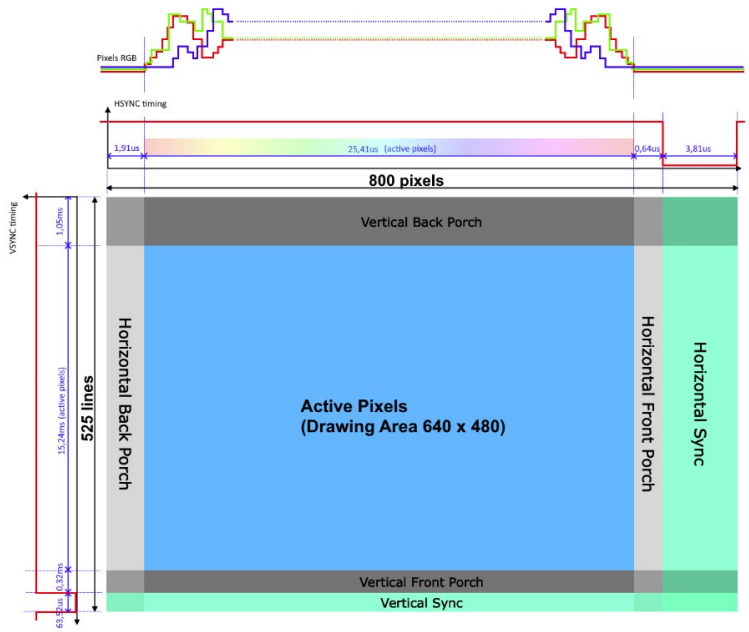
La transmission d'une image se réalise en transmettant les pixel un par un. Chaque pixel est composé de 3 couleurs (rouge, vert et bleu). Chacune de ces couleurs est codée sur 4 bits, permettant une gamme de valeurs allant de 1111 (représentant l'intensité maximale) à 0000 (représentant l'intensité minimale).

Le processus de transmission des données de l'image commence en envoyant le premier pixel situé en haut à gauche de l'image et se termine avec le dernier pixel en bas à droite. Ce processus suit un balayage ligne par ligne, comme illustré dans le schéma ci-dessous :



Avec H-pos représente la position horizontale du pixel, V-pos représente la position verticale.

Soit le schéma suivant qui nous résume le signal de synchronisation en parallèle avec l'image envoyée par dans le VGA :



Pour rappel, seul la zone « Display Area » sera affichée sur l'écran du VGA, les autres zones n'ont donc pas besoin d'obtenir des informations sur la couleur.

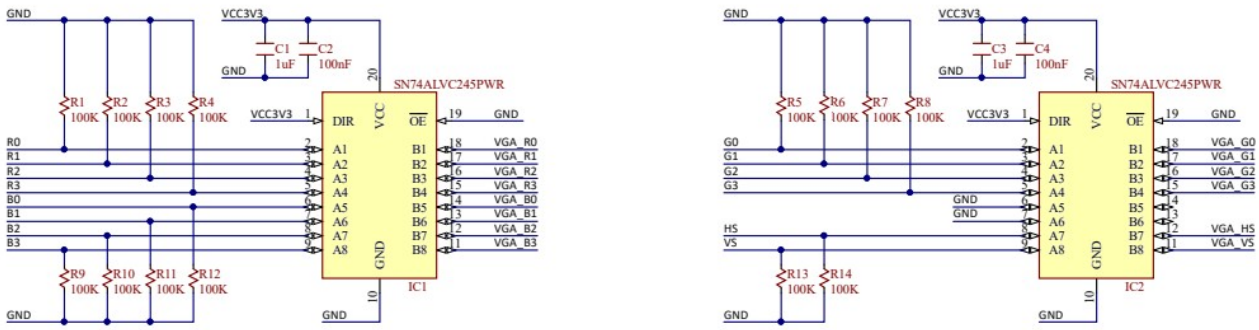
III- Connecteur Pmod VGA 410-345 :

Le connecteur Pmod VGA de Digilent est spécialement conçu pour faciliter la connectivité avec des périphériques d'affichage VGA (Video Graphics Array). Son rôle est de transmettre les signaux nécessaires à l'affichage des images sur un écran VGA. Il permet la transmission des signaux de synchronisation horizontale (HSYNC) et verticale (VSYNC), ainsi que des signaux de couleur (R, G, B) du générateur VGA vers l'écran. Grâce à ce connecteur, nous avons pu connecter notre carte FPGA à un moniteur VGA standard, permettant ainsi l'affichage d'images et de contenu visuel sur l'écran. La figure ci-dessous représente le connecteur Pmod VGA utilisé dans ce TP.

Le Pmod VGA utilise 14 broches d'entrée pour créer un port de sortie VGA analogique. Cela correspond à une profondeur de couleur de 12 bits et à deux signaux de synchronisation standard : la synchronisation horizontale (HS) et la synchronisation verticale (VS).

Il y a sur cette carte 2 émetteur-récepteur asynchrones, qui permettent de normaliser la tension numérique à transmettre au VGA à 3,3V.

La conversion numérique-analogique est réalisée à l'aide d'une simple échelle de résistances R-2R. Cette échelle fonctionne en conjonction avec la résistance de terminaison de 75 ohms de l'affichage VGA pour créer 16 niveaux de signal analogique pour les signaux VGA rouge, bleu et vert. Ce circuit produit des signaux de couleur vidéo qui progressent par incréments égaux entre 0V (complètement éteint) et 3.1V (complètement allumé). Avec 4 bits pour le rouge, le bleu et le vert, il est possible d'afficher 4096 (16x16x16) couleurs différentes, une pour chaque motif unique de 12 bits.



émetteur-récepteur asynchrone

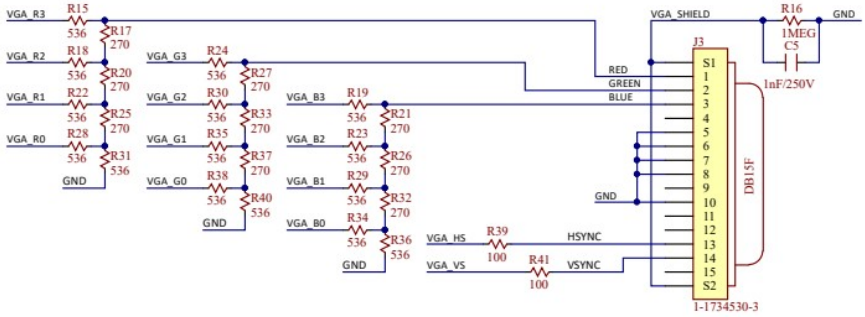


Schéma électrique convertisseur numérique-analogique

Le tableau suivant présente de manière détaillée les différentes broches du connecteur, les noms des signaux associés à chaque broche, ainsi qu'une description complète de chaque signal. Ce tableau constitue une ressource précieuse pour comprendre la fonction et l'utilisation de chaque broche du connecteur. La première colonne indique les numéros de pin correspondants à chaque broche, ce qui permet une identification facile. La deuxième colonne fournit les noms des signaux, ce qui facilite encore davantage l'identification de chaque signal spécifique. Enfin, la troisième colonne contient une description détaillée de chaque signal, expliquant son rôle et sa fonction dans le contexte de l'interface VGA. Ce tableau est une référence essentielle pour ceux qui souhaitent utiliser le connecteur de manière appropriée et comprendre les spécificités de chaque signal.

Pin	Signal	Description
1	R0	Red 0
2	R1	Red 1
3	R2	Red 2
4	R3	Red 3
5	GND	Power Supply Ground
6	VCC3V3	Positive Power Supply
7	G0	Green 0
8	G1	Green 1
9	G2	Green 2
10	G3	Green 3
11	GND	Power Supply Ground
12	VCC3V3	Positive Power Supply

Table 1. Pmod header J1.

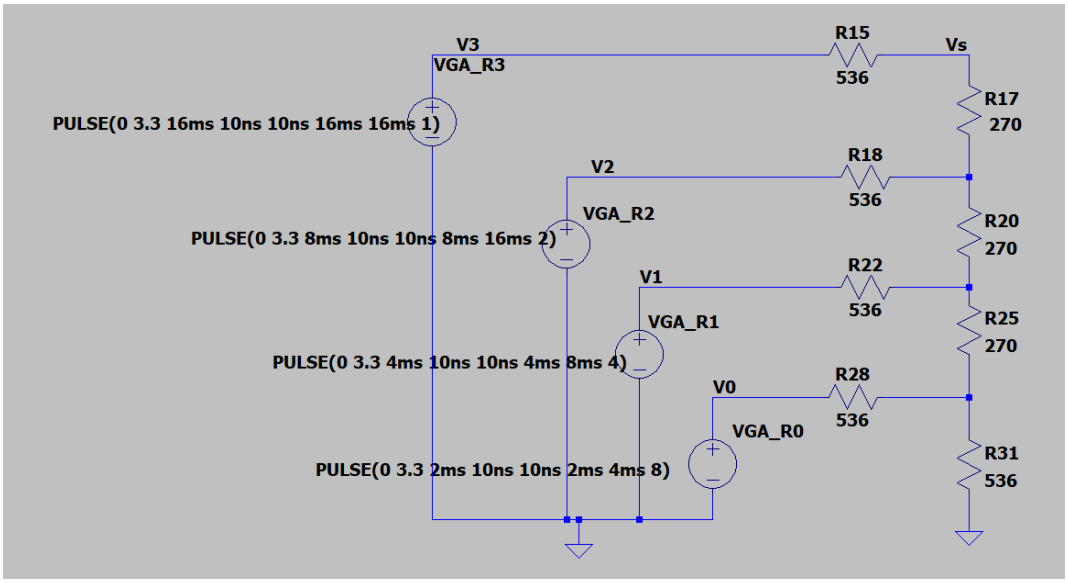
Pin	Signal	Description
1	B0	Blue 0
2	B1	Blue 1
3	B2	Blue 2
4	B3	Blue 3
5	GND	Power Supply Ground
6	VCC3V3	Positive Supply Ground
7	HS	Horizontal Sync
8	VS	Vertical Sync
9	NC	Not Connected
10	NC	Not Connected
11	GND	Power Supply Ground
12	VCC3V3	Positive Power Supply

Table 1. Pmod header J2.

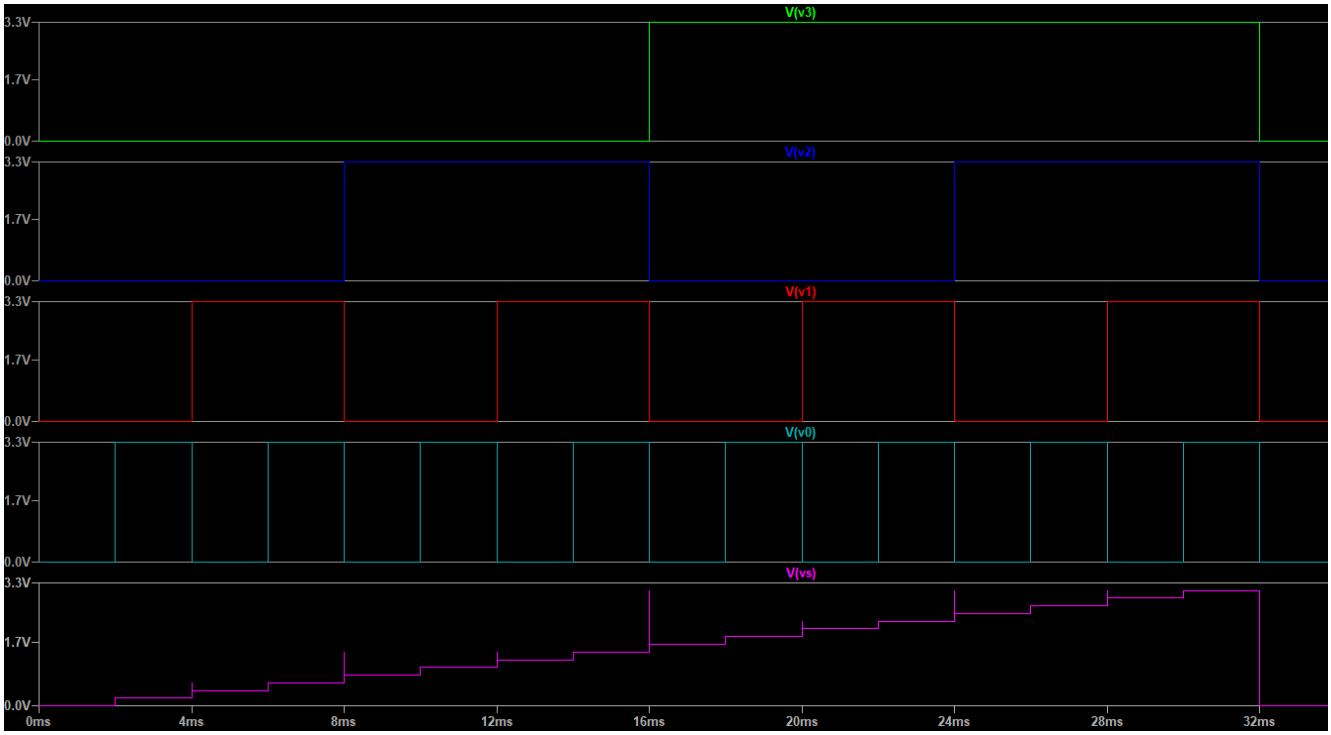
Le Pmod VGA joue un rôle crucial dans la conversion des données numériques en sortie du FPGA en signaux analogiques. La conversion Numérique/Analogique est détaillée dans la documentation du Pmod VGA et est

effectuée grâce à un schéma électrique spécifique pour chaque couleur. Chaque composante de couleur, représentée par 4 bits, est convertie en une tension analogique à l'aide d'un diviseur de tension.

Afin de pouvoir visualiser la conversion, nous avons reproduit le schéma électronique dans le logiciel LTspice représenté sur la figure suivante :



Nous avons procédé à une capture d'écran du schéma mentionné, ainsi qu'a une capture d'écran illustrant les courbes de simulation des quatre tensions d'entrée numériques (correspondant aux quatre bits) et la tension analogique de sortie.



Les courbes V3, V2, V1 et V0 correspondent aux 4 bits représentant l'intensité des couleurs, allant du bit de poids le plus élevé (V3) au bit de poids le plus faible (V0). La tension analogique résultante après conversion est représentée par la courbe Vs.

En outre, nous avons effectué une vérification et constaté que pour une intensité maximale d'une couleur représentée par la valeur numérique 1111, la tension analogique convertie atteint 3.1 volts.

IV- Architecture à fenêtre glissante et calcul de produit de convolution 2D

La procédure de l’ architecture à fenêtre glissante (ou "sliding window architecture" en anglais) est une technique utilisée dans le traitement d'image et d'autres domaines pour effectuer des opérations locales sur une région de l'image à la fois. Elle permet d'analyser ou de traiter progressivement une image en déplaçant une fenêtre de taille fixe sur l'ensemble de l'image.

Le fonctionnement de la procédure "sliding window architecture" est le suivant :

1. Définition de la taille de la fenêtre : une taille fixe est déterminée pour la fenêtre, qui peut être un carré, un rectangle ou toute autre forme spécifiée.

2. Positionnement initial de la fenêtre : la fenêtre est positionnée sur une région initiale de l'image.

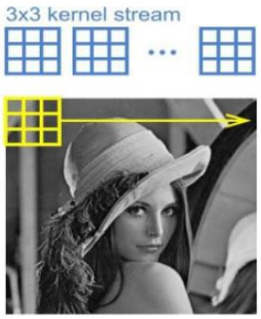
3. Traitement de la région couverte par la fenêtre : les opérations spécifiques sont appliquées à la région couverte par la fenêtre. Cela peut inclure des opérations telles que la détection de contours, la segmentation, le filtrage, l'extraction de caractéristiques, etc.

4. Déplacement de la fenêtre : la fenêtre est déplacée vers une nouvelle position adjacente, généralement en effectuant un décalage d'un ou plusieurs pixels selon une direction prédéfinie (horizontale, verticale ou diagonale).

5. Répétition des étapes 3 et 4 : les étapes de traitement et de déplacement de la fenêtre sont répétées jusqu'à ce que la fenêtre ait parcouru l'ensemble de l'image.

En utilisant la procédure "sliding window architecture", il est possible d'effectuer des opérations locales sur chaque région de l'image, ce qui permet d'extraire des informations spécifiques à des emplacements précis de l'image. Cela peut être utile dans des tâches telles que la détection d'objets, la reconnaissance de motifs, le suivi d'objets, etc.

Il convient de noter que la procédure "sliding window architecture" peut être adaptée en fonction des besoins spécifiques de l'application. Par exemple, la taille de la fenêtre peut varier, des techniques de traitement spécifiques peuvent être appliquées à chaque région, et des stratégies de recouvrement peuvent être utilisées pour éviter les lacunes entre les régions adjacentes. Un exemple est représenté ci-dessous avec une fenêtre carrée de taille fixe de 3x3.

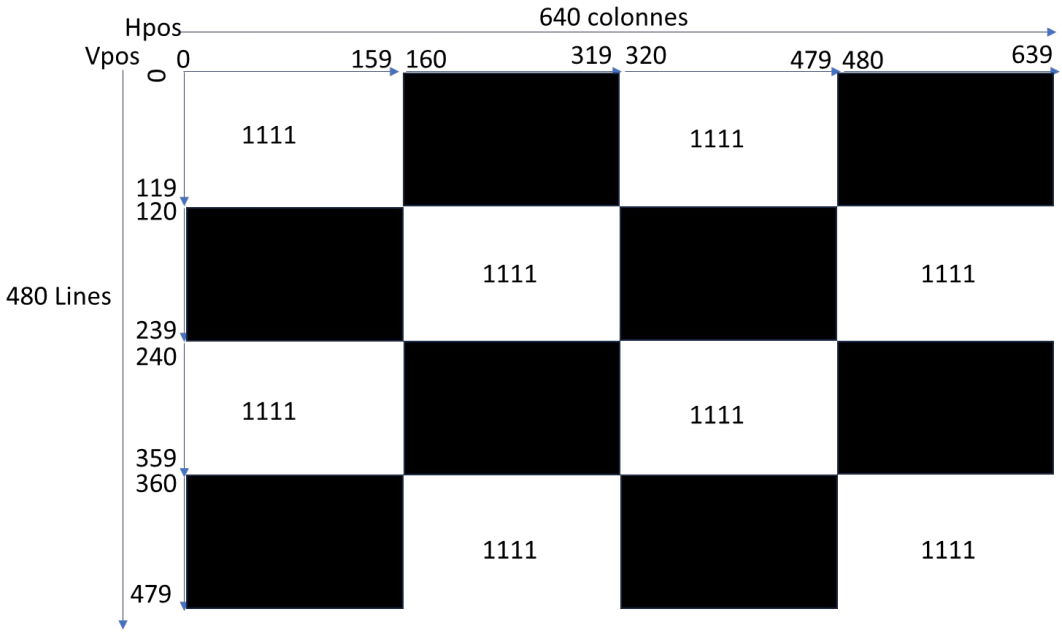


Dans la suite de notre démarche, nous envisageons d'utiliser une fenêtre de filtre gaussien de taille 3x3x ainsi que des buffers. Cette approche nous permettra d'appliquer une transformation adéquate sur les données de l'image. Le filtre gaussien contribuera à lisser les contours et réduire le bruit indésirable, tandis que les buffers nous aideront à stocker temporairement les valeurs des pixels afin de faciliter les opérations ultérieures.

Le cœur de produit de convolution pour un filtre gaussien est une matrice carrée qui représente les coefficients du filtre. Chaque coefficient indique le poids relatif attribué à un pixel voisin lors du calcul de la nouvelle valeur du pixel central. Pour un filtre gaussien 3x3, le cœur de produit de convolution ressemble à ceci :

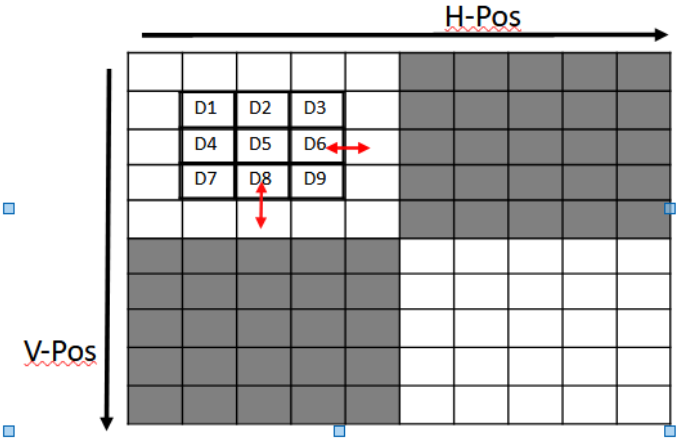
1	2	1
2	4	2
1	2	1

Dans le cadre de ce rapport, nous allons appliquer le filtre mentionné sur une image de dimensions 640x480 pixels. Cette image est constituée de carreaux noirs représentant une intensité de couleur de 0000 (ou 0 en décimal) et de carreaux blancs représentant une intensité de couleur de 1111 (ou 15 en décimal) telle que présentée ci-dessous :



Afin d'illustrer le processus, nous allons examiner le calcul de la convolution pour certains pixels spécifiques, tels que ceux situés aux transitions entre les intensités de couleur 1111 et 0000, ou vice versa. Cela nous permettra de mieux comprendre l'effet du filtre gaussien sur ces zones de transition.

Le résultat de la convolution sera attribué au pixel central de la fenêtre de convolution (D5). Les pixels voisins (D1 à D9) seront utilisés pour calculer la valeur du pixel central (voir représentation ci dessous).



Calcul du produit de convolution dans le cas où :

1. **La fenêtre de convolution est positionnée sur un carreau entièrement blanc, c'est-à-dire :**

   D1=D2=D3=D4=D5=D6=D7=D8=D9 = 15.

   - Résultat de convolution

     $$Output_{data} = \frac{15 \times 16}{16} = 1111_{B(2)} = 15_{B(10)}$$

2. **La fenêtre de convolution est positionnée sur un carreau entièrement noir, c'est-à-dire :**

   D1=D2=D3=D4=D5=D6=D7=D8=D9 = 0.

   - Résultat de convolution

     $$\text{Output\_data } \frac{0 \times 16}{16} = 0000_{B(2)} = 0_{B(10)}$$

3. **La fenêtre de convolution est positionnée entre un carreau entièrement blanc et un carreau entièrement noir.** Dans ce cas-là on a deux cas :

   Cas 1: décalage d'une colonne vers la droite ou d'une ligne vers le bas

   D1=D2=D4=D5=D7=D8 = 15 et D3=D6 = D9 =0 ou D1=D2=D3=D4=D5=D6 = 15 et D7=D8 = D9 =0

   - Résultat de convolution

     $$\text{Output\_data } = \frac{0 \times (1+2+1) + 15 \times (2+4+2+1+2+1)}{16} = 11_{B(10)} = 1011_{B(2)}$$

   Cas 2: décalage de deux colonnes vers la droite ou de deux lignes vers le bas

   D1=D4=D7 = 15 et D2=D3=D5=D6=D8=D9 =0 ou D1=D2=D3 = 15 et D4=D5=D5=D6=D7=D8 =D9= 0

   - Résultat de convolution

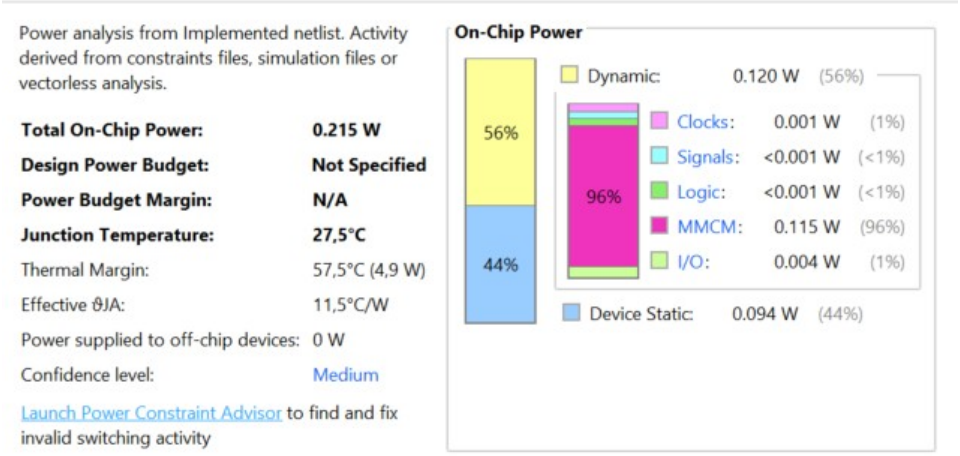
     $$\text{Output\_data } = \frac{0 \times (1+2+1+2+4+2) + 15 \times (+1+2+1)}{16} = 3_{B(10)} = 0011_{B(2)}$$

Si la fenêtre de convolution est située dans un coin, entre plusieurs carreaux blancs et noirs, nous appliquons le même principe de calcul que décrit précédemment.

V- Synthèse et ressources utilisés

Une analyse de puissance a été réalisée à partir de la netlist implémentée, en prenant en compte les informations issues des fichiers de contraintes, de fichiers de simulation. Le rapport de consommation de puissance résultant affiche la puissance totale consommée, qui s'élève à 0.215W. Les détails sur la consommation des différentes régions de la puce FPGA sont présentés dans la figure ci-dessous.

Il est noté que la consommation de puissance par la PLL est de 0.115W. ces informations sont essentielles pour évaluer et optimiser l'efficacité énergétique de la conception FPGA.



VI- Ressources utilisés

La synthèse de Vivado s'accompagne de rapports, qui nous permettent de connaître les ressources du FPGA utilisées.
Nous pouvons donc constater que nous avons utilisé 8,32 % des composantes du FPGA : 4,59 % de LUT (Look Up Table/tableau de correspondance) et 3,73 % de registres.

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	808	0	0	17600	4.59
LUT as Logic	808	0	0	17600	4.59
LUT as Memory	0	0	0	6000	0.00
Slice Registers	1313	0	0	35200	3.73
Register as Flip Flop	1313	0	0	35200	3.73
Register as Latch	0	0	0	35200	0.00
F7 Muxes	0	0	0	8800	0.00
F8 Muxes	0	0	0	4400	0.00

Si nous regardons plus en détail, nous avons :

- 1312 registres à reset asynchrone (FDCE)
- 2 buffer d'entrée (IBUF) pour le reset et le bouton qui nous permet d'activer ou non le filtre gaussien
- 16 buffers de sorties, pour le VGA. (vsync, hsync, et 12 pour la couleur), et pour les 2 LED qui indique si le filtre est actif ou non.

```
Report Cell Usage:
+------+----------------+------+
|      |Cell            |Count |
+------+----------------+------+
|1     |clk_wiz_0_bbox  |     1|
|2     |CARRY4          |    44|
|3     |LUT1            |    16|
|4     |LUT2            |  1471|
|5     |LUT3            |    18|
|6     |LUT4            |     5|
|7     |LUT5            |     7|
|8     |LUT6            |    14|
|9     |FDCE            |  1312|
|10    |FDRE            |     1|
|11    |IBUF            |     2|
|12    |OBUF            |    16|
+------+----------------+------+
```

VII- Chemin critique et métastabilité

Nous vérifions dans le rapport de timing le chemin critique du composant, pour pouvoir vérifier la latence du système. Nous constatons qu'il est de 27,357 ns, entre v_centre_reg[10] et line_buffers_1_reg[540]. Ce temps est inférieurs au 40 ns utilisés par pixel, donc nous n'auront pas de problème de synchronisation au sein du FPGA.

```
Max Delay Paths
--------------------------------------------------------------------------------------
Slack (MET) :             27.357ns  (required time - arrival time)
  Source:                 VTC_com_Phase2/v_cntr_reg_reg[10]/C
                            (rising edge-triggered cell FDCE clocked by clk_out1_clk_wiz_0  {rise@0.000ns fall@20.000ns period=40.000ns})
  Destination:            ImageFilter_com/line_buffers_1_reg[540][2]/D
                            (rising edge-triggered cell FDCE clocked by clk_out1_clk_wiz_0  {rise@0.000ns fall@20.000ns period=40.000ns})
```

Nous pouvons aussi vérifier qu'il n'y a pas de métastabilité autour du signal d'horloge. Cette information nous est donné par TNS et THS : s'il est à zéro, nous n'aurons pas d'état transitoire des signaux. C'est confirmé par sur la capture ci-dessous du rapport de timing.

```
------------------------------------------------------------------------------------------------
| Design Timing Summary
| ---------------------
------------------------------------------------------------------------------------------------

    WNS(ns)      TNS(ns)  TNS Failing Endpoints  TNS Total Endpoints      WHS(ns)      THS(ns)  THS Failing Endpoints  THS Total Endpoints     WPWS(ns)     TPWS(ns)
    -------      -------  ---------------------  -------------------      -------      -------  ---------------------  -------------------     --------     --------
      6.165        0.000                      0                 2596        0.110        0.000                      0                 2596        2.000        0.000
```