



- CAHIER DE RECETTE -

INTERFACE VIDEO VGA

Formation AJC FPGA - Eve CHAR



03 JUILLET 2023
EVE CHAR

Table des matières

I. CRITERES D'ACCEPTATION DES ESSAIS	2
II. RESULTATS DE TESTS - PHASE 1 : CONTROLEUR VGA SANS FILTRE DE GAUSS	2
1. RESULTAT DE TEST DE LA GENERATION DE L'HORLOGE VGA « PIXEL CLOCK »	2
2. RESULTAT DE TEST DE LA SYNCHRONISATION HORIZONTALE (H-SYNC)	3
3. RESULTAT DE TEST DE LA SYNCHRONISATION VERTICALE (V-SYNC)	7
4. RESULTAT DE TEST DE L'ENSEMBLE HSYNC ET VSYNC : TIMING GENERAL (TIMING_GEN)	10
5. RESULTAT DE TEST DE LA POSITION DU PIXEL (H_CNT) ET SON RESET	12
6. RESULTAT DE TEST DE LA POSITION DU PIXEL SUR UNE TRAME (V_CNT) ET SON RESET	13
7. RESULTAT DE TEST DE LA ZONE D'AFFICHAGE DE L'ECRAN (VIDEO_ON)	13
8. RESULTAT DE TEST GLOBAL DE LA PHASE 1 « CONTROLEUR_VGA » SANS FILTRE	15
III. RESULTATS DE TESTS - PHASE 2 : CONTROLEUR VGA AVEC FILTRE DE GAUSS.....	16
1. RESULTAT DE TEST DES ENTREES / SORTIES DU FIFO1 (LECTURE - ECRITURE).....	16
2. RESULTAT DE TEST DES ENTREES / SORTIES DU FIFO2 (LECTURE- ECRITURE)	17
3. RESULTAT DE TESTS DES ENTREES / SORTIES DU CONV-FILTER-GAUSS (CALCUL)	18
4. RESULTAT DE LA SYNCHRONISATION DU SYSTEME APRES L'AJOUT DU FILTRE	20
5. RESULTAT DE TEST DU DEGRADE DE GRIS DANS LES BORDS APRES L'AJOUT DU FILTRE	22
6. RESULTAT DE TEST GLOBAL DE LA PHASE 2 « CONTROLEUR_VGA » AVEC FILTRE	23
IV. MOYEN DE TEST 2 :TEST SUR CIBLE -SET UP -DEBUG - ILA	24
V. MOYEN DE TEST 3 : TEST SUR CIBLE - OSCILLOSCOPE.....	29
VI. ANNEXE :	33
1. ANNEXE 1 : PHASE 1 : PLACEMENT / ROUTAGE	33
2. ANNEXE 2 : PHASE 2 : PLACEMENT / ROUTAGE	34
3. ANNEXE 3 : CALCUL DES RESULTATS OBTENUS DU CONV-FILTER-GAUSS:	35

I. Critères d'acceptation des essais

Pour chaque essai, l'activation et l'acceptation est basée sur les critères suivants :

- ✓ Le test est activable :
 - 1) les données en entrée sont prêtes,
 - 2) les outils, fonctions, systèmes nécessaires sont disponibles,
 - 3) la procédure d'essai peut être déroulée,
 - 4) les essais précédents dont dépend cet essai se sont déroulés avec succès.
- ✓ Les sorties de l'essai sont conformes aux sorties spécifiées.

Le test est refusé si une des conditions ci-dessus n'est pas respectée.

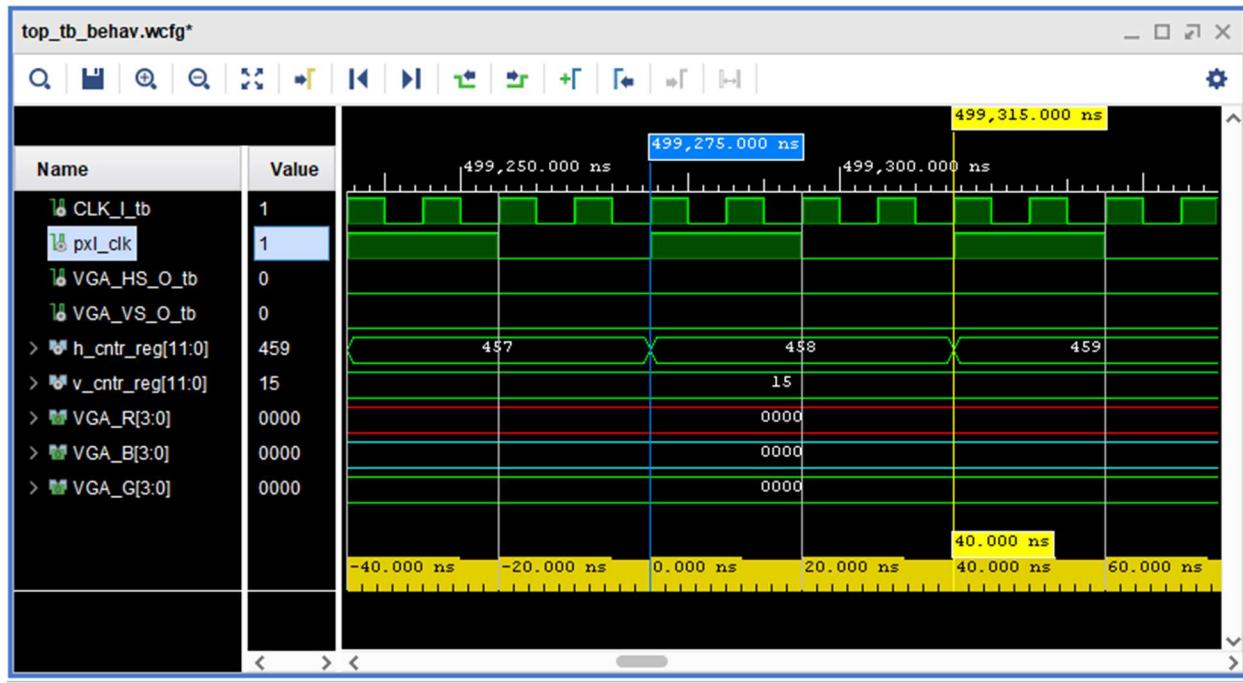
II. Résultats de tests - phase 1 : contrôleur VGA sans filtre de GAUSS

1. Résultat de test de la génération de l'horloge VGA « pixel clock »

Résultats du test :

Test N°1 clock_gen_01 : Vérification de la génération de « pixel_clock »			
<u>Résumé:</u> Tests permettant de s'assurer que la fréquence générée est celle attendue.			
<u>Quoi :</u> la fréquence d'horloge d'affichage des pixels VGA est de (25,2 MHz) : cela signifie qu'un pixel a une durée de 1/25200000 s).			
<u>comment/pré-requis :</u> Réaliser un testbench de la fonction pixel-clock avec en entrée une clock de 100MHz et une sortie pixel_clock			
<u>N° d'étape</u>	<u>Actions de pas :</u>	<u>Résultats attendus :</u>	<u>Mesure -résultats</u>
1	Lancer la simulation	Vérifier qu'il y a 1/25200000 s entre 2 fronts montants de la clock de sortie pixel_clock => Pixel time = 1/ Pixel clock = 0,04 µs	=> Pixel time = 1/ Pixel clock = 0,04 µs Résultat = ok
<u>Cahier d'exigences</u>	EXIGENCE_clock_gen_01		✓ Pass

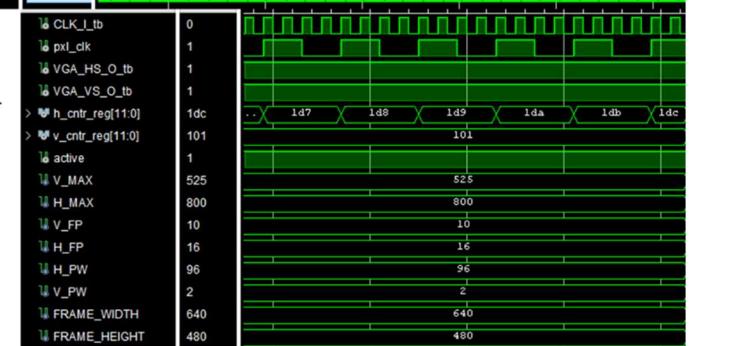
Démonstration :



Génération de la clock 148.5 MHz pxl_clk -- ***1920x1080 @60Hz***



Génération de la clock 25 MHz-- ***640x480@60Hz*** (800 525 zone active)



2. Résultat de test de la synchronisation horizontale (H-sync)

Résultats du test :

Test N°2 fonction *horizontal_timing_gen_03* : Vérification de la synchronisation horizontale

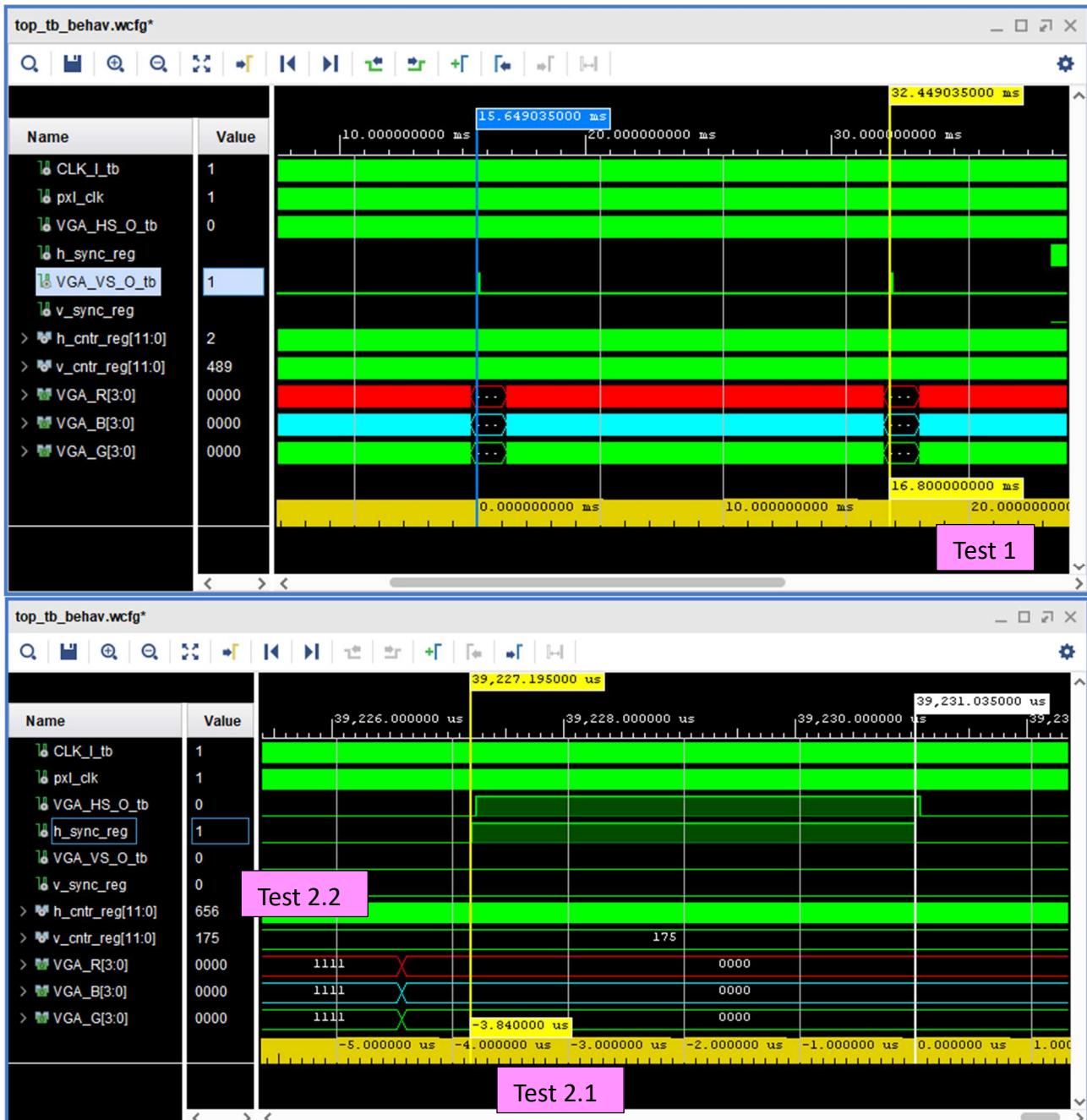
Résumé : Tests permettant de s'assurer que la fonction *horizontal_timing_gen_03* : génère la synchronisation horizontale

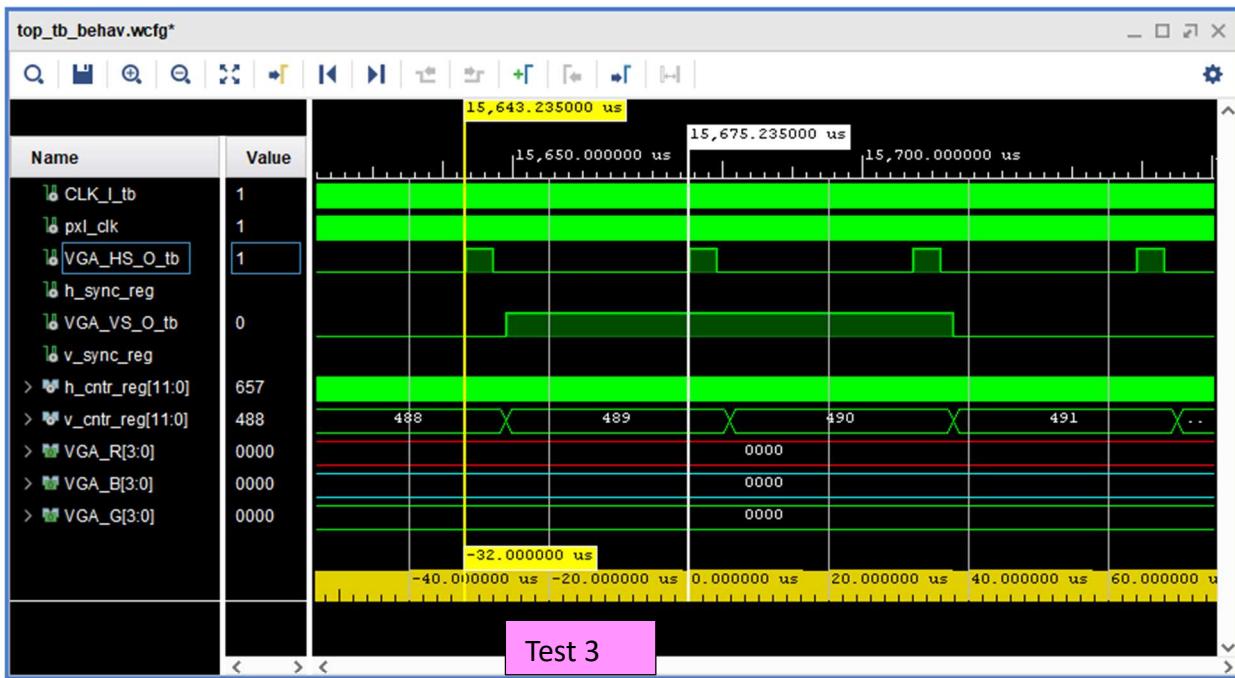
Quoi : valider un rafraîchissement de l'écran (image entière) toutes les 60 Hz

Comment/pré-requis : Réaliser un test-bench de la fonction *horizontal_timing_gen_03* avec :

*en entrée : *pixel_clock* et *RESET*

<p>*en sortie : signaux logiques de synchronisation horizontale (h sync)</p> <p>- Lancer la simulation de la fonction horizontal_timing_gen_03 : rajouter les signaux synchronisation horizontale (h sync) et le compteur h_cnt</p>			
N° d'étape	<u>Actions de pas:</u>	<u>Résultats attendus:</u>	<u>Résultat obtenu</u>
1	Vérifier le rafraîchissement de l'écran (image entier) toutes les 60 Hz : Mesurer le temps entre deux images consécutives quand les compteurs (h_cnt) et (v_cnt) sont à zéro	<p>Le temps entre deux (h_cnt) à 0 et (v_cnt) à 0 et le prochain h_cnt = 0 = v_cnt doit être = 1/60 s (60Hz)</p> <p>Equivaut à $800 \times 525 \times 0.04 \mu\text{s} = 16.8 \text{ ms}$</p>	Ecart entre 2 pulse de v_sync = 16.8 ms OK
2.1	Vérifier la durée H_PW : Durée de l'impulsion de synchronisation	Le temps doit être = 96 pixels x 0,04 μs = 3,84 μs	H_PW = 96 pixels x 0,04 μs = 3,84 μs OK
2.2	Vérifier la plage de signal H_PW :	Vérifier pour chaque front montant de clock pixel et dans le cas où RST est égal à '0' : si h_sync = '1' si la position pixel ($HD + H_{FP} + H_{PW} > H_{POS} \geq (HD + H_{FP})$) $656 \geq H_{POS} \geq 751$	656 et 751 OK
3	Vérifier la durée de la ligne horizontale	Vérifier si j'ai pour chaque ligne horizontale = $H_{PW} + H_{BP} + h_{sync} = 32 \mu\text{s}$ $HD + H_{FP} = 128 + 16 + 640 + 16 = 800 \times 0,04 \mu\text{s} = 32 \mu\text{s}$	Ecart entre 2 pulse de h_sync = 32μs OK
4	Vérifier le reset de la synchronisation horizontale	Vérifier pour chaque front montant de clock pixel et dans le cas où RST est à '1' : si h_sync = '0'	OK
5	- Test différent : changer la durée de l'impulsion de synchronisation (SP) à 128 au lieu de 96 et changer la durée (BP) qui précède l'affichage à 16 pixel au lieu de 48	On doit retrouver : $H_{PW} = 128 \text{ pixels} \times 0,04 \mu\text{s} = 5,12 \mu\text{s}$ et $H_{BP} = 16 \text{ pixels} \times 0,04 \mu\text{s} = 0,64 \mu\text{s}$	Non effectué
<u>Cahier d'exigences</u>	<u>EXIGENCE_horizontal_timing_gen_03</u>		✓ Pass

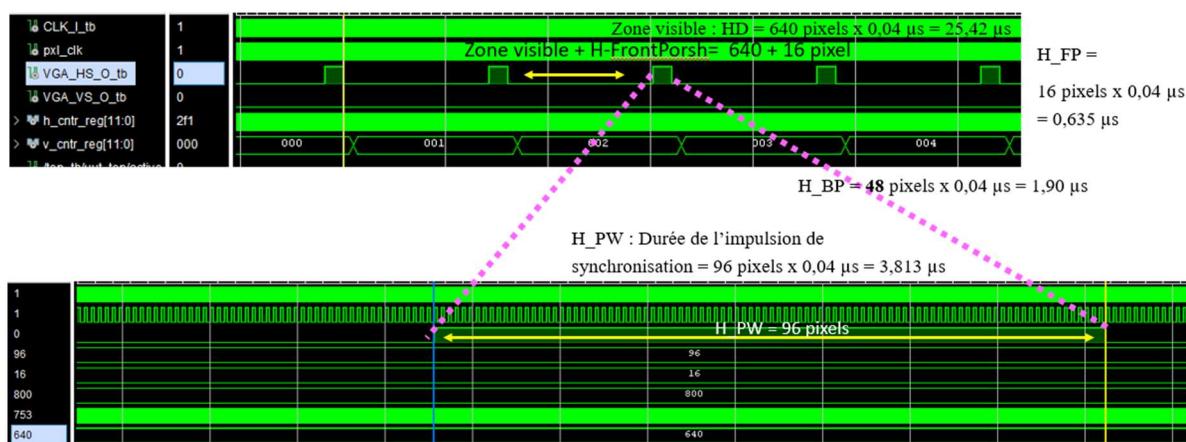
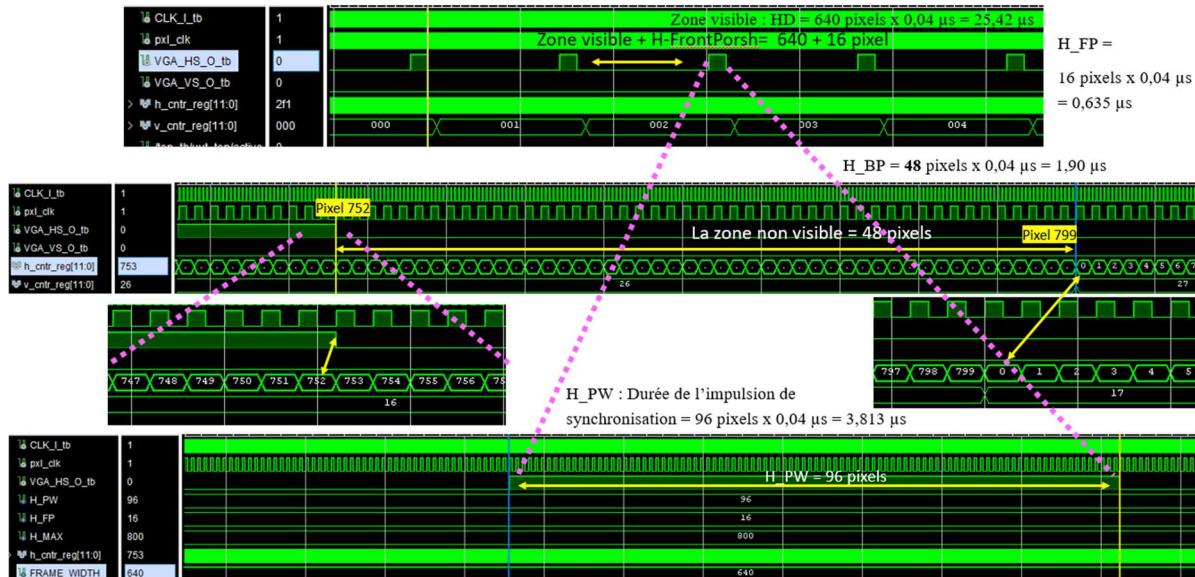
Démonstration :



***640x480@60Hz

***(800 h- 525 v: zone active)

La synchronisation horizontale = 32 us avec 800 pixel (ligne complète)

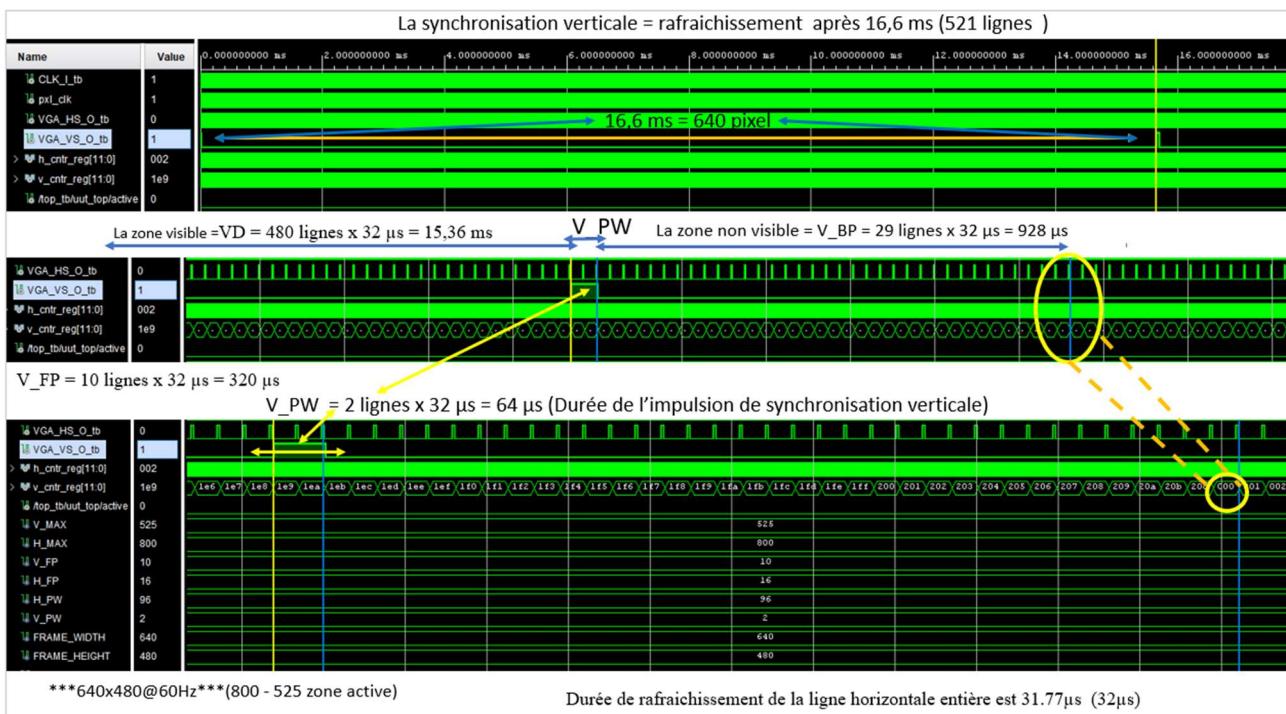
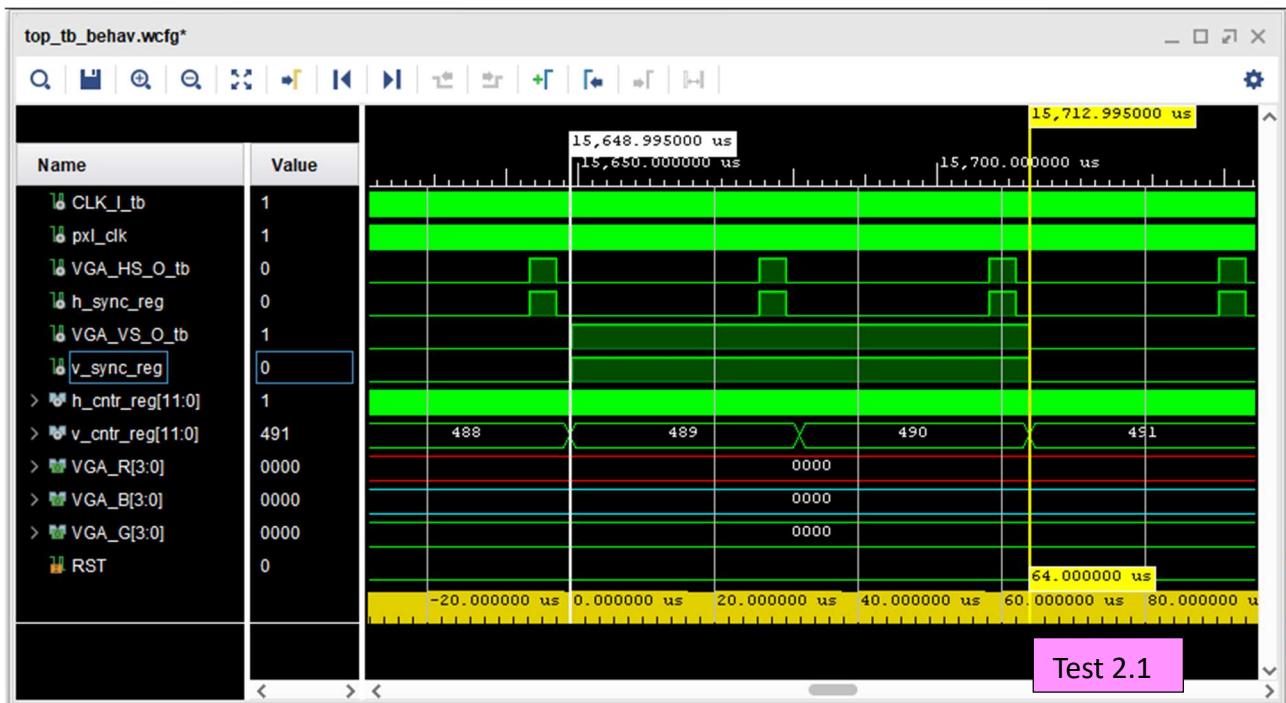


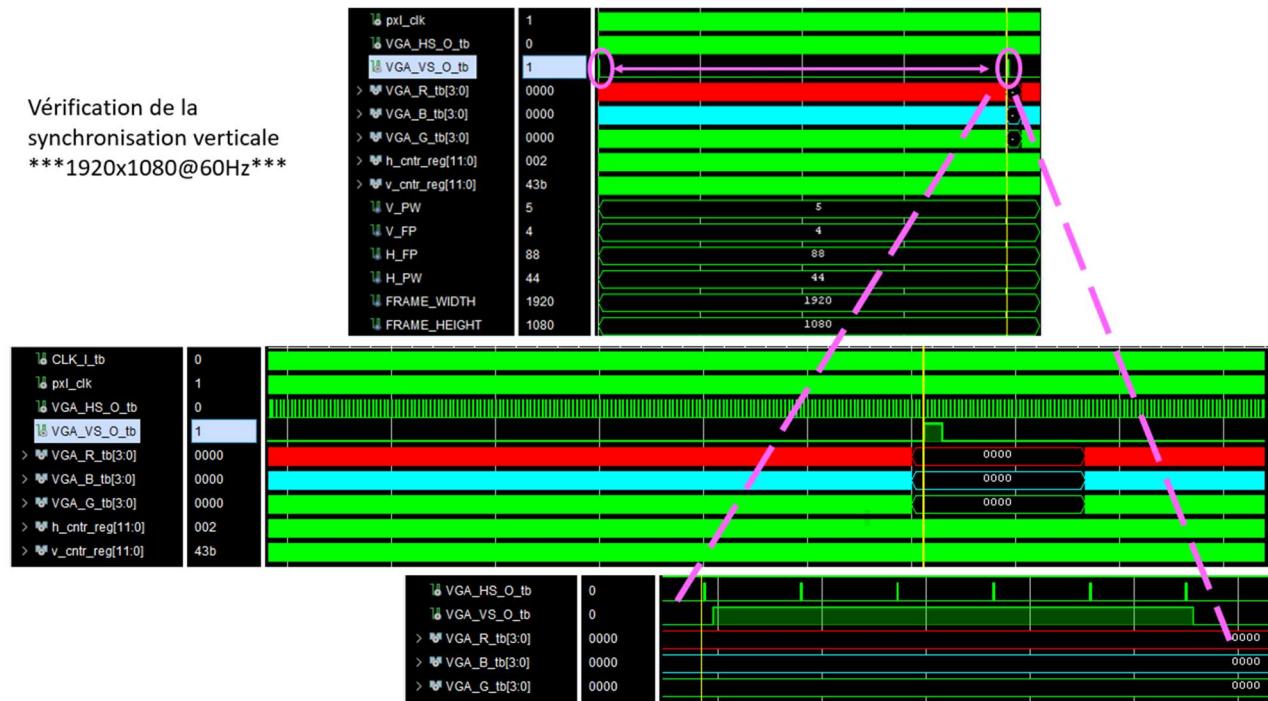
3. Résultat de test de la synchronisation verticale (v-sync)

Résultats du test :

Test N°3 fonction : vertical_timing_gen_04 : Vérification de la synchronisation verticale			
<p><u>Résumé:</u> Tests permettant de s'assurer que la fonction vertical_timing_gen_04 : génère la synchronisation vertical</p> <p>Quoi : valider un rafraîchissement vertical : rafraîchissement d'une ligne entière à 31.46875 kHz</p> <p>comment/pré-requis : Réaliser un test-bench de la fonction vertical_timing_gen_04 avec :</p> <ul style="list-style-type: none"> *en entrée : pixel_clock et RESET *en sortie : signaux logiques de synchronisation verticale (vsync) - Lancer la simulation de la fonction vertical_timing_gen_04 : rajouter les signaux synchronisation vertical (vsync) et le compteur v_cnt 			
N° d'étape	<u>Actions de pas :</u>	<u>Résultats attendus :</u>	<u>Résultat obtenu</u>
1	Vérifier le rafraîchissement vertical : donc rafraîchissement d'une ligne complète à 31.46875 kHz quand les compteurs sont à zéro	Le temps entre deux (h_cnt) à 800 consécutifs doit être 32µs (h_cnt = fin de la ligne et v_cnt = 2 par exemple)	Temps de synchro verticale = 32µs OK
2.1	Vérifier la durée V_PW : Durée de l'impulsion de synchronisation verticale	Le temps doit être = 2 lignes x 32 µs = 64 µs (Durée de l'impulsion de synchronisation verticale)	V_PW = 64 µs
2.2	Vérifier la plage de signal V_PW :	Vérifier pour chaque front montant de clock pixel et dans le cas où RST est égal à '0' : si V_sync = '1' quand la position pixel (HD + V_FP + V_PW) > V_POS >= (VD + V_FP) Soit 490 >= V_POS >= 491	489 et 490 Ok
3	Vérifier la durée d'une trame verticale	Vérifier si affichage vertical = V_PW + V_BP + VD+ V_FP = 521 lignes x 32µs = 16,67ms = 1/60Hz	16.8 ms Ok
4	Vérifier le reset de la synchronisation verticale	Vérifier pour chaque front montant de clock pixel et dans le cas où RST est à '1' : si v_sync = '0'	Ok
<hr/>			
<u>Type d'exécution:</u>	Test_bench		
<u>Cahier d'exigences</u>	EXIGENCE_vertical_timing_gen_04		✓ Pass

Démonstration :



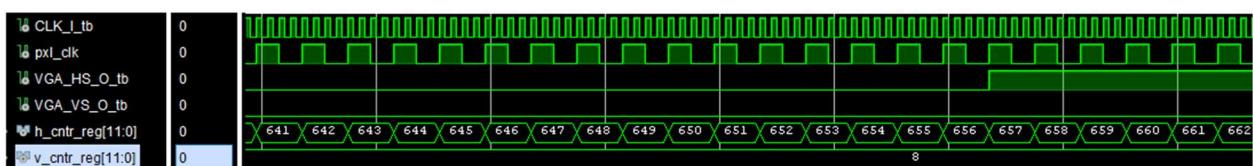
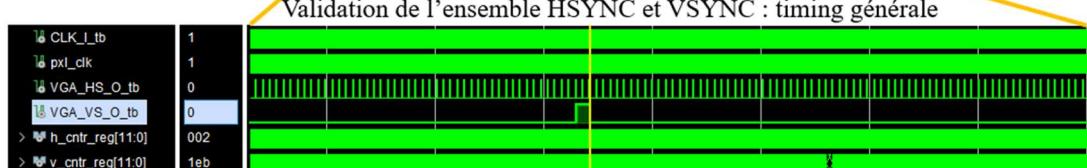


4. Résultat de test de l'ensemble HSYNC et VSYNC : timing général (timing_gen)

Résultats du test :

Test N°4 fonction : timing_gen_02 : Vérification de la génération des signaux			
<p><u>Résumé :</u> Tests permettant de s'assurer que le composant contrôleur_VGA génère bien les signaux logiques de synchronisation horizontale et verticale (h_sync et v_sync)</p> <p><u>comment/pré-requis :</u> Réaliser un test Bench de la fonction timing_gen_02 avec :</p> <ul style="list-style-type: none"> *en entrée : pixel_clock et RST *en sortie : signaux logiques de synchronisation horizontale et verticale (h_sync et v_sync) et signal couleurs - Lancer la simulation de la fonction timing_gen_02 : rajouter les signaux synchronisation horizontale (h_sync) et verticale (v_sync) ainsi que les compteurs h_cnt et v_cnt 			
<u>N° d'étape</u>	<u>Actions de pas:</u>	<u>Résultats attendus:</u>	<u>Résultat obtenu</u>
1	Sur front montant de la synchronisation horizontale (h_sync), observer h_cnt.	Vérifier (position pixel) pour un front montant de h_cnt donné => doit être 656	h_cnt = 656
2	Sur front descendant de la synchronisation horizontale (h_sync), observer h_cnt.	Vérifier (position pixel) pour un front descendant de h_cnt donné => doit être 751	h_cnt = 751
3	Sur front montant de la synchronisation verticale (v_sync) observer h_cnt.	Vérifier (position pixel) pour un front montant de v_cnt donné => doit être 490	v_cnt = 489
4	Sur front descendant de la synchronisation verticale (v_sync) observer h_cnt.	Vérifier (position pixel) pour un front descendant de v_cnt donné => doit être 491	v_cnt = 490
<u>Cahier d'exigences</u>	<u>EXIGENCE_timing_gen_02</u>		✓ Pass

Démonstration :



5. Résultat de test de la position du pixel (h_cnt) et son reset

Résultats du test :

Test N°5 fonction : compteur_H_POS_05 : Vérification de la position pixel sur la ligne horizontale

Résumé : Tests permettant de s'assurer que la fonction `compteur_H_POS` compte correctement la position pixel sur la ligne horizontale

Comment/pré-requis : Réaliser un test-bench de la fonction `compteur_H_POS` avec :

*en entrée : `pixel_clock` et `RESET`

*en sortie : signaux logiques de synchronisation (`hsync`)

N° d'étape	Actions de pas :	Résultats attendus :	Résultat obtenu
1	Vérifier la position d'un pixel sur la ligne horizontale	Regarder un pixel sur ligne avec une couleur donnée : <code>h_cnt = 355</code> et couleur rouge On doit voir une colonne horizontale rouge d'épaisseur 1 pixel	

2	Vérifier le reset du compteur horizontal	Vérifier pour chaque front montant de clock pixel et dans le cas où RST est à '1' : si le compteur H se réinitialise = '0'	
<u>Cahier d'exigences</u>	<u>EXIGENCE_compteur_H_POS_05</u>	✓ Non effectué	

Démonstration :**6. Résultat de test de la position du pixel sur une trame (v_cnt) et son reset****Résultats du test :**

Test N°6 fonction : compteur_V_POS_06 : Vérification de la position pixel sur la ligne Verticale			
<u>Résumé :</u> Tests permettant de s'assurer que la fonction compteur_V_POS_06 compte correctement la position pixel sur la ligne horizontale			
<u>Comment/pré-requis :</u> Réaliser un test-bench de la fonction compteur_V_POS_06 avec :			
	*en entrée : pixel_clock et RESET		
	*en sortie : signaux logiques de synchronisation (v_sync)		
N° d'étape	Actions de pas :	Résultats attendus :	Résultat obtenu
1	Vérifier la position d'un pixel sur la ligne Verticale	Regarder un pixel sur ligne avec une couleur donnée : exemple v_cnt = 500 et couleur rouge On doit voir une ligne horizontale rouge d'épaisseur 1 pixel	
2	Vérifier le reset du compteur Verticale	Vérifier pour chaque front montant de clock pixel et dans le cas où RST est à '1' : si le compteur V se réinitialise à '0'	
<u>Cahier d'exigences</u>	<u>EXIGENCE_compteur_V_POS_06</u>	✓ Non effectué	

Démonstration :**7. Résultat de test de la zone d'affichage de l'écran (Video_On)****Résultats du test :**

Test N°7 fonction : display_area_07 : Vérification de la zone visible de l'écran		

Résumé : Tests permettant de s'assurer que la fonction **display_area_07** affiche la bonne zone visible

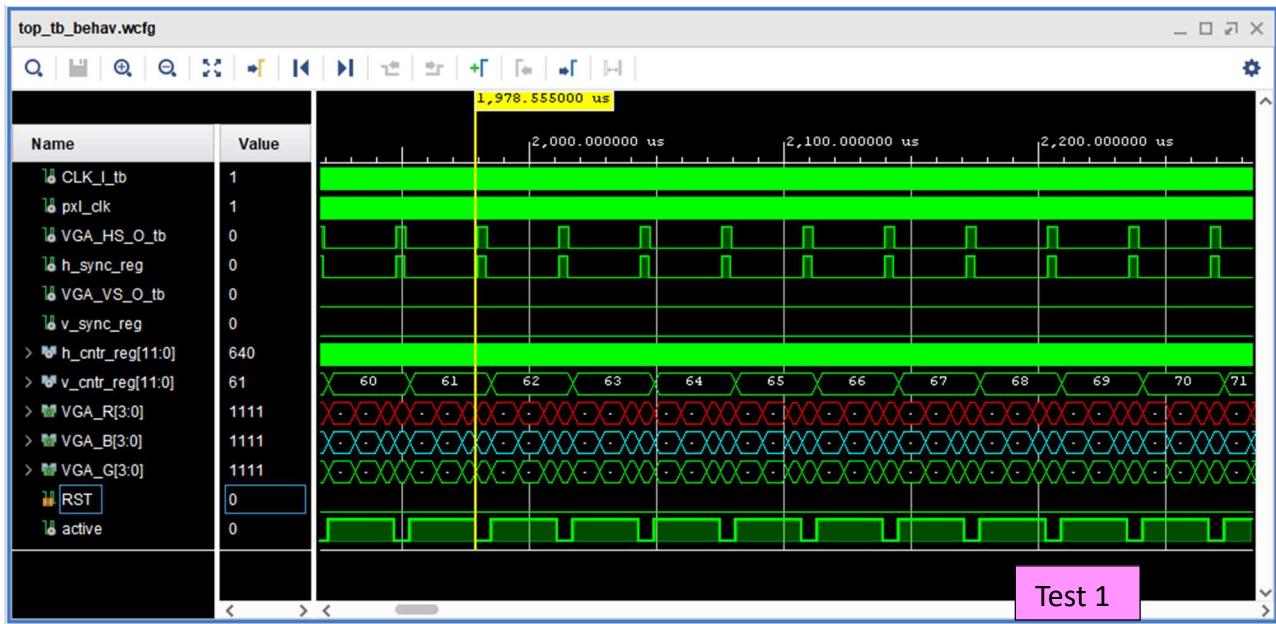
Comment/pré-requis : Réaliser un test-bench de la fonction **display_area_07** avec :

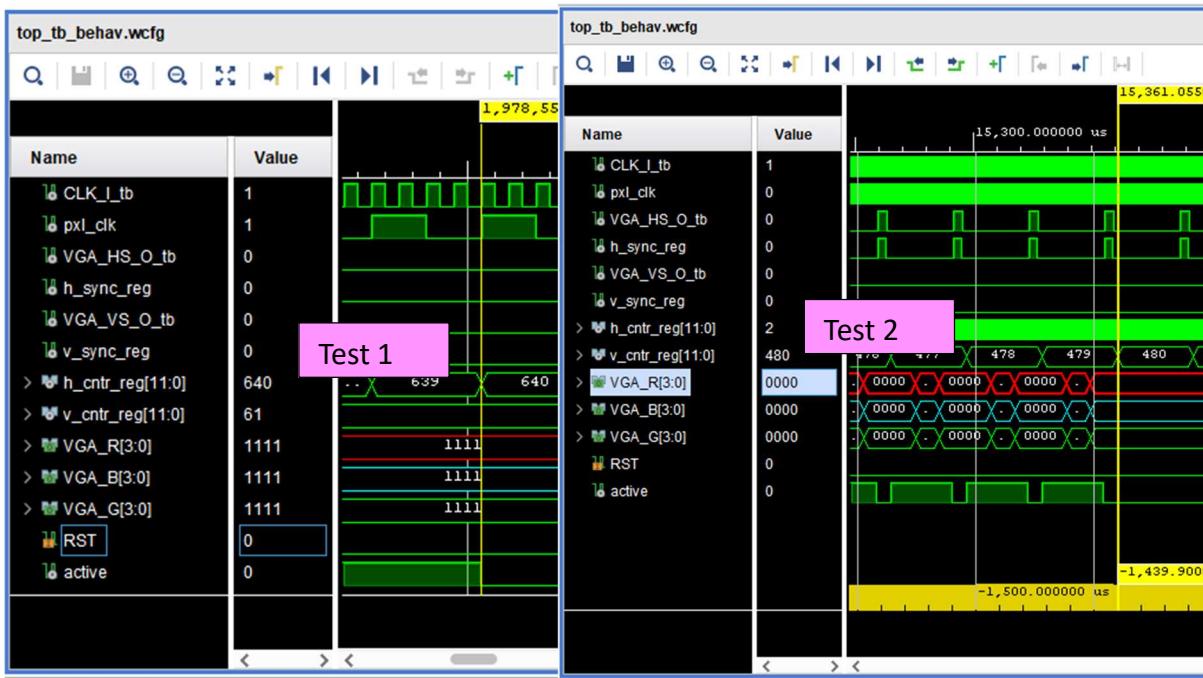
*en entrée : **pixel_clock** et **RST**

*en sortie : signaux logiques de synchronisation (**h_sync** et **v_sync** avec leurs compteurs respectives)

N° d'étape	Actions de pas:	Résultats attendus:	Résultat obtenu
1	Vérifier la largeur de la zone visible	Regarder le nombre de pixels visibles sur une ligne de l'écran = on doit trouver $h_cnt = 640$	$h_cnt = 640$ OK
2	Vérifier la hauteur de la zone visible	Regarder le nombre de pixels visible en hauteur sur l'écran = on doit trouver $v_cnt = 480$	$v_cnt = 480$ OK
<u>Cahier d'exigences</u>		EXIGENCE_ <i>display_area_07</i>	✓ PASS

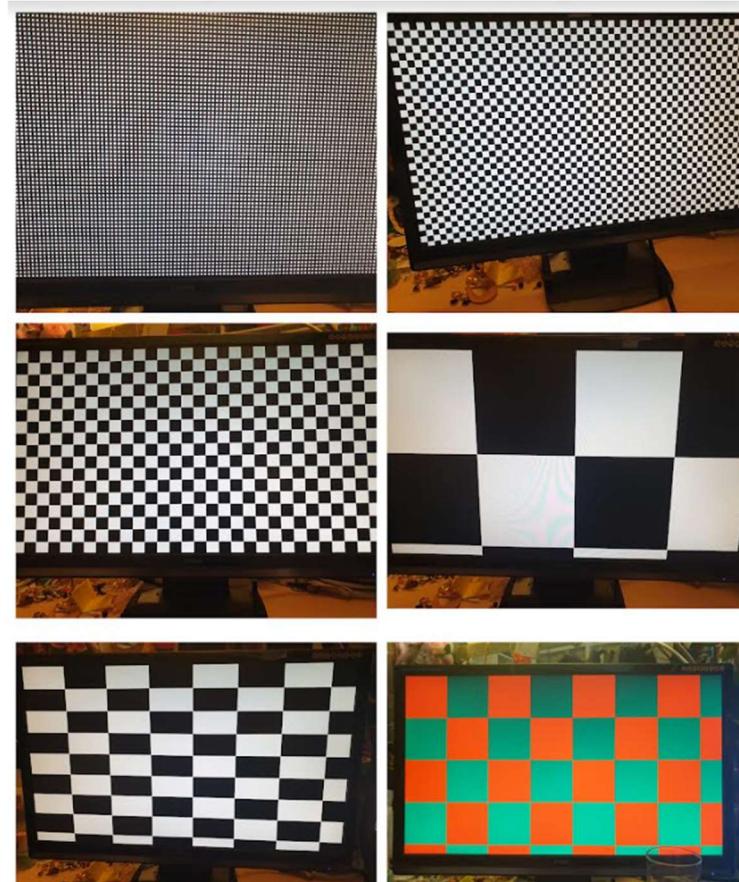
Démonstration :





8. Résultat de test global de la phase 1 « contrôleur_VGA » sans filtre

Démonstration :

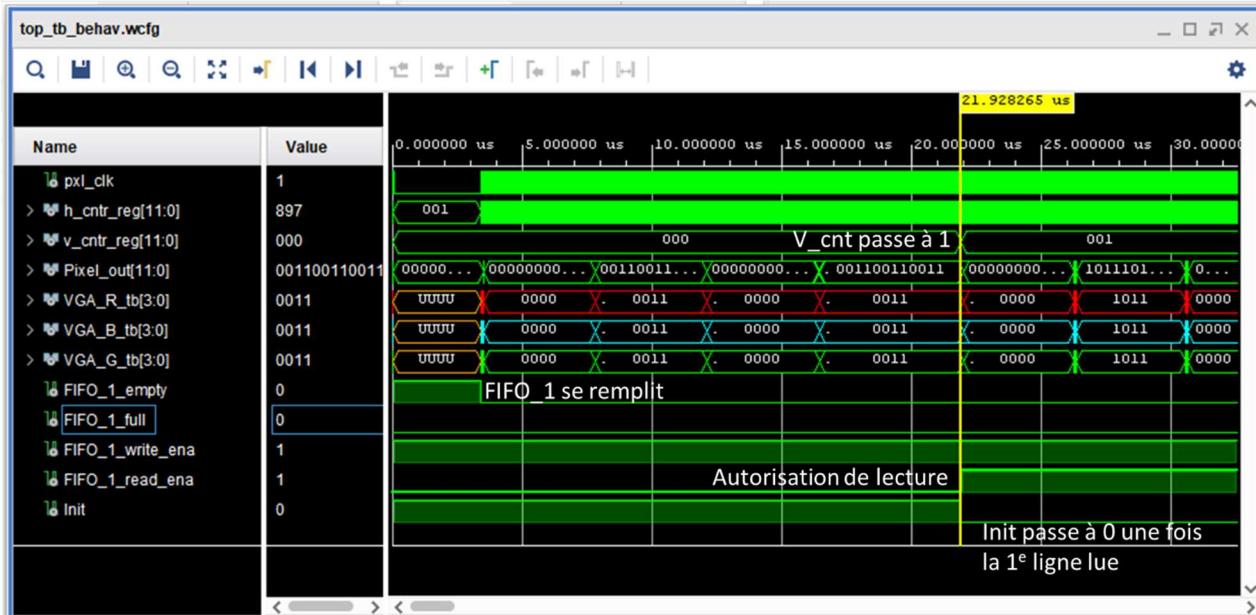


III. Résultats de tests - phase 2 : contrôleur VGA avec filtre de GAUSS

1. Résultat de test des entrées / sorties du fifo1 (lecture - écriture)

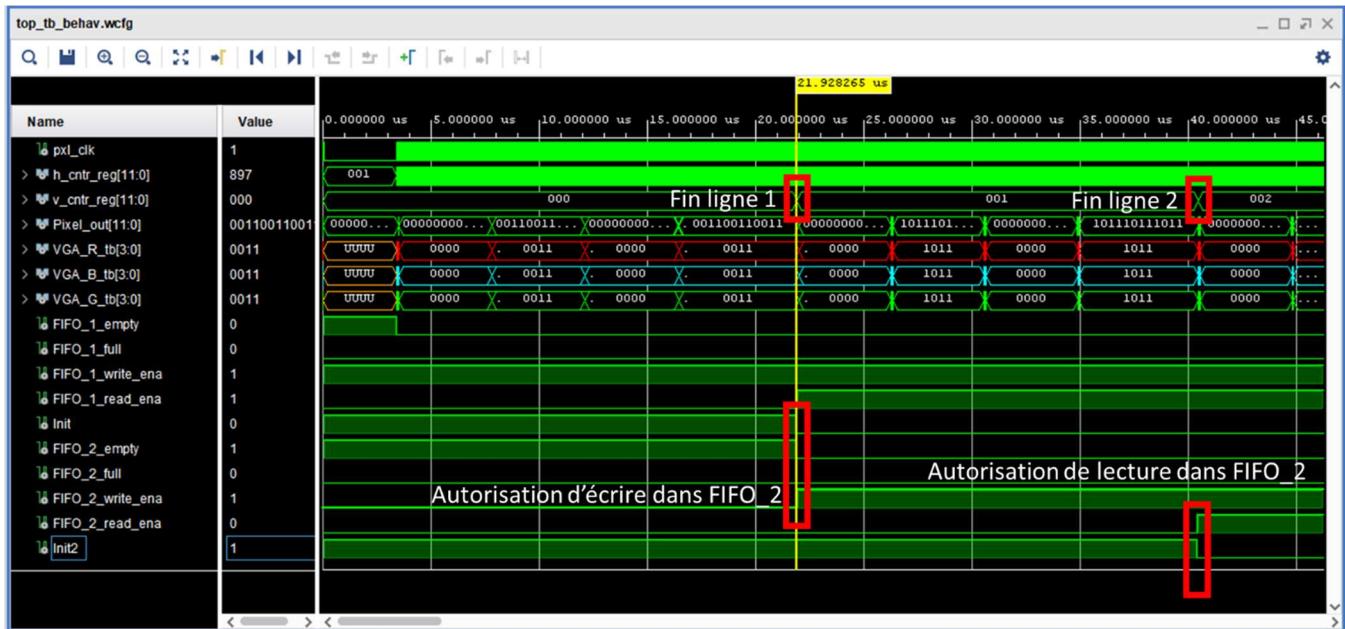
Résultats du test :

Test N°9 test des entrées / sorties du fifo1 (lecture- écriture)			
<p><u>Résumé</u> : Tests permettant de s'assurer que la fonction FIR_Filter_08 remplit son rôle de filtre de matrice 3x3</p> <p><u>Comment/pré-requis</u> : Réaliser un test-bench de la fonction FIR_Filter_08 avec :</p> <ul style="list-style-type: none"> *en entrée : signal sortie de pattern_gen (12 bits) et autorisation d'écriture dans le FIFO1 *en sortie : signal data RGB : couleur sur (12 bits) 			
<u>N° d'étape</u>	<u>Actions de pas :</u>	<u>Résultats attendus :</u>	<u>Résultat obtenu</u>
1	Vérifier la valeur du pixel_L3 (l'écriture dans la FIFO)	Regarder quand la 1e ligne est en train d'être lue si (FIFO_1_write_ena) est active	OK
2	Vérifier les conditions de lecture de la fifo : On lit dans FIFO_1 si : - FIFO_1 n'est pas vide - et si la première ligne est entièrement lue (Init doit être à 0)	Vérifier quel le signal full est toujours à zéro et que la lecture est autorisée quand Init passe à zéro	Quand v_cnt va passer à 1, FIFO_1_write_ena passe à 1 OK
3	Vérifier le flux de l'autorisation de lecture de la FIFO1	Regarder quand la première ligne est entièrement lue : si (FIFO_1_read_ena) est active (h_cnt = 800)	OK
<u>Cahier d'exigences</u>	<u>EXIGENCE_display_area_07</u>		✓ PASS

Démonstration :**2. Résultat de test des entrées / sorties du fifo2 (lecture- écriture)****Résultats du test :**

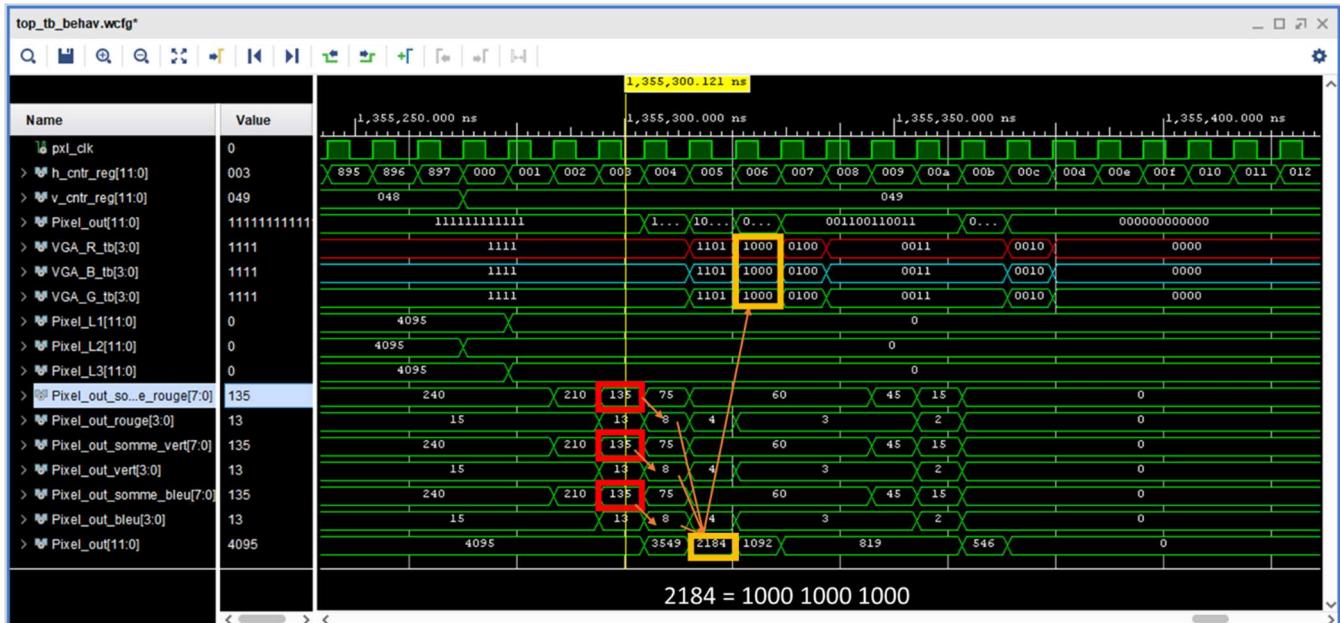
Test N°10 test des entrées / sorties du fifo2 (lecture- écriture)			
N° d'étape	Actions de pas :	Résultats attendus :	Résultat obtenu
1	Vérifier les conditions d'écriture dans la FIFO_2	Regarder quand (FIFO_2_write_ena) est active : la 1e ligne est entièrement lue (Init passe à 0) et la ligne 2 en train d'être lue (par FIFO_1)	OK On commence à écrire dans FIFO_2 quand Init = 0
2	Vérifier les conditions de remplissage de la fifo2 : On écrit dans FIFO2 si : - FIFO2 n'est pas rempli - et si la première est entièrement lue	Vérifier quel le signal full est toujours à zéro Regarder si l'autorisation d'écriture passe à 1 pour v_cnt = 0 et h_cnt = 800	OK On commence à lire dans FIFO_2 quand Init2 passe à 0
3	Vérifier le flux de l'autorisation de lecture de la FIFO2	FIFO_2_read_ena est active quand la deuxième ligne est entièrement lue Regarder si l'autorisation de lecture passe à 1 pour v_cnt = 1 et h_cnt = 800	OK

<u>Cahier d'exigences</u>	<u>EXIGENCE_FIR_Filter_08</u>	✓ PASS
---------------------------	-------------------------------	---------------

Démonstration :**3. Résultat de tests des entrées / sorties du Conv-filter-Gauss (calcul)****Résultats du test :**

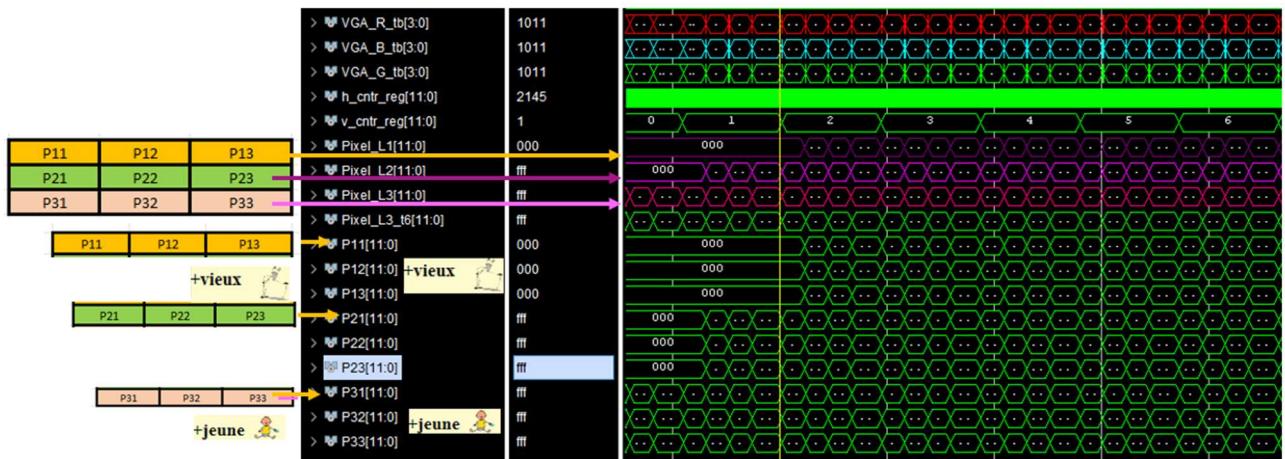
Test N°11 test des entrées / sorties du Conv-filter-Gauss (calcul)			
<u>Résumé :</u> Tests permettant de s'assurer que la fonction FIR_Filter_08 remplit son rôle de filtre de matrice 3x3			
<u>Comment/pré-requis :</u> Réaliser un test-bench de la fonction FIR_Filter_08 avec :			
*en entrée : signal sortie de pattern_gen (12 bits) et autorisation d'écriture dans le FIFO1			
N° d'étape	Actions de pas :	Résultats attendus :	Résultat obtenu
1	Vérifier la première opération de calcul : multiplication : Application du masque KERNEL	Regarder le décalage de bit à droite en fonction des éléments du masque ($x2$ = décalage 1 bit à droite) ($x4$ = décalage 2 bits à droite)	OK
2	Vérifier la somme	Vérifier la somme	OK
3	Vérifier la 3ème opération de calcul : division	Regarder le décalage de bit à gauche : on divise par le poids du masque (16) ($/16 = /2^4$ donc on aura un décalage 4 bits à gauche)	OK

<u>Cahier d'exigences</u>	<i>EXIGENCE_FIR_Filter_08</i>	✓ PASS

Démonstration :

	rouge 4 bits	bleu 4 bits	vert 4 bits	Masque Gauss	rouge * gaus	bleu * gauss	vert * gauss
P11	15	15	15	1	15	15	15
P12	15	15	15	2	30	30	30
P13	0	0	0	1	0	0	0
P21	15	15	15	2	30	30	30
P22	0	0	0	4	0	0	0
P23	0	0	0	2	0	0	0
P31	15	15	15	1	15	15	15
P32	15	15	15	2	30	30	30
P33	15	15	15	1	15	15	15
				somme	135	135	135
				division par 16	8,4375	8,4375	8,4375
				conversion en 12 bits			
							2184

Voir annexe les autres tableaux de validation des résultats des autres valeurs calculés par la fonction Filtre (240 et 210 et 60).

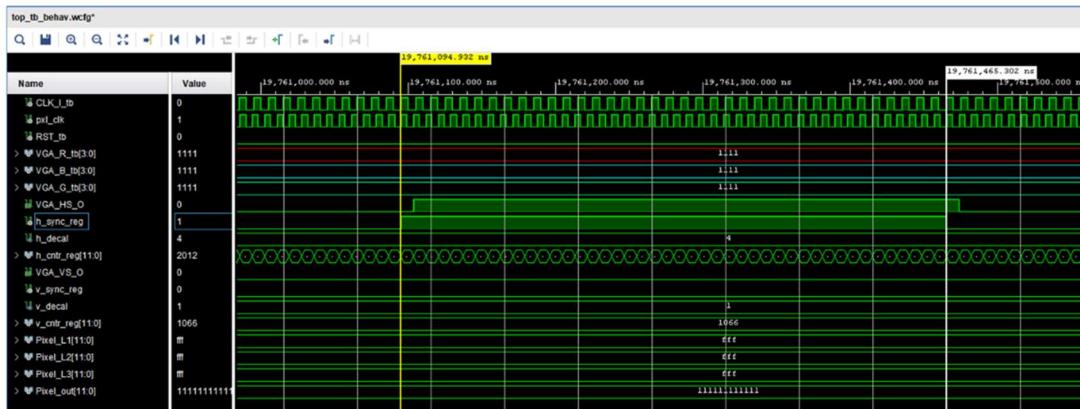


4. Résultat de la synchronisation du système après l'ajout du filtre

Résultats du test :

Test N°12 de la synchronisation du système après l'ajout du filtre			
<p><u>Résumé</u> : Tests permettant de s'assurer que la fonction FIR_Filter_08 rempli son rôle de filtre de matrice 3x3</p> <p><u>Comment/pré-requis</u> : Réaliser un test-bench de la fonction FIR_Filter_08 avec :</p> <ul style="list-style-type: none"> *en entrée : signal sortie de pattern_gen (12 bits) et autorisation d'écriture dans le FIFO1 *en sortie : signal data_RGB : couleur sur (12 bits) 			
<u>N° d'étape</u>	<u>Actions de pas:</u>	<u>Résultats attendus:</u>	<u>Résultat obtenu</u>
1	Vérifier les coordonnées h_sync et v_sync	Regarder les coordonnées h_sync et v_sync	OK
2	Regarder le nombre de cycle d'horloge de décalage	Regarder les signaux h_decal et v_decal sont ajoutés aux calculs de h_sync et v_sync	OK
<u>Cahier d'exigences</u>		<u>EXIGENCE_display_area_07, timing_gen_02</u>	✓ PASS

Démonstration :

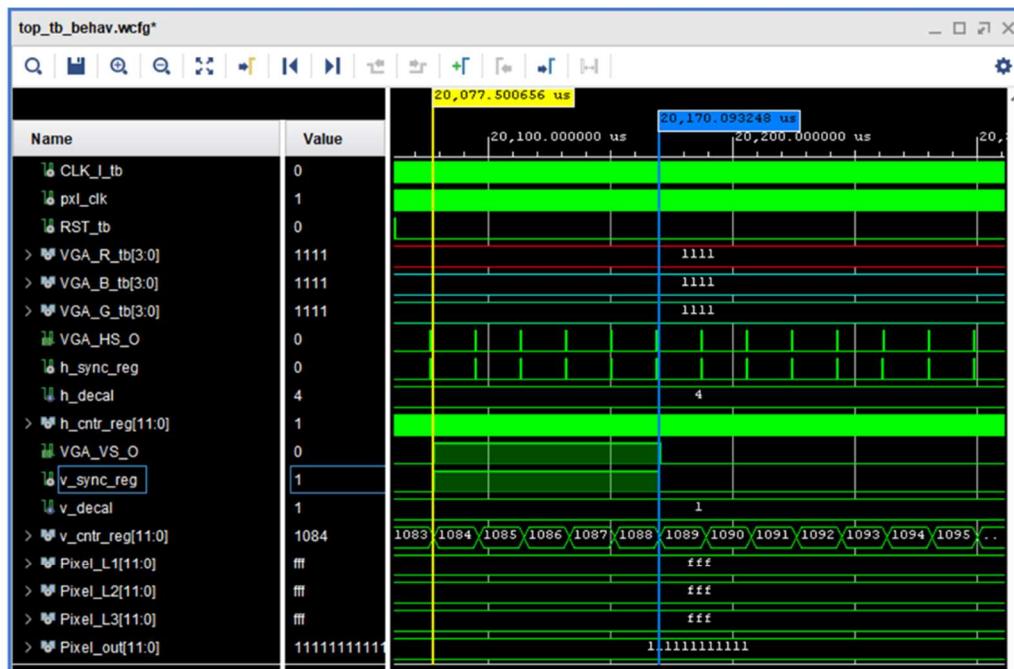


Décalage de h-sync

H_sync_reg vaut 1 quand h_cntr_reg vaut 2012 jusqu'à 2055 (à 2056, repasse à 0)

H_decal = 4 donc H_Sync_Reg vaut 1 de 2008 à 2051

H_FP + FRAME_WIDTH -1 = 2007, pulse width = 44 (donc 2007 étant la 1^e colonne jusqu'à 2050 la 44^e colonne)



Décalage de v-sync

V_sync_reg vaut 1 quand v_cntr_reg vaut 1084 jusqu'à 1088

V_decal = 1 donc V_Sync_Reg vaut 1 de 1083 à 1087

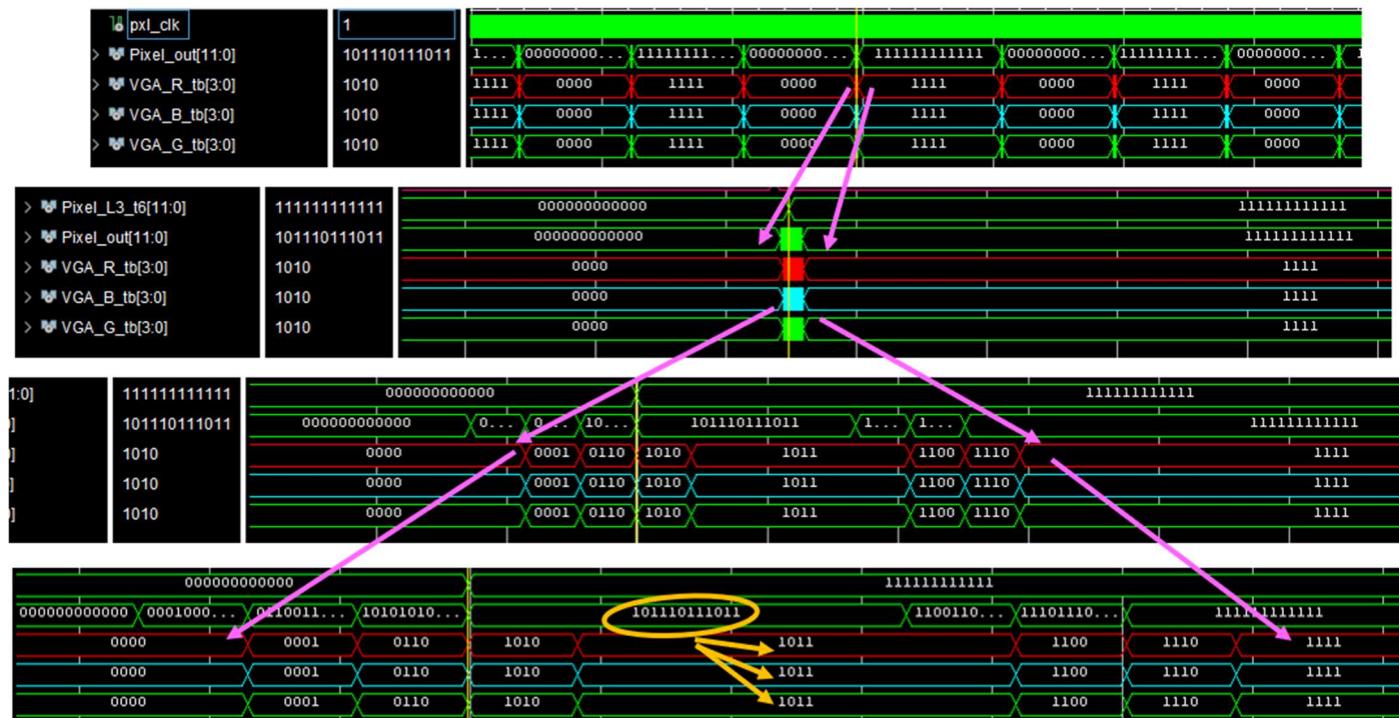
V_FP + FRAME_HEIGHT -1 = 1083, pulse width = 5 (donc 1083 étant la 1^e ligne jusqu'à 1087 la 5^e ligne)

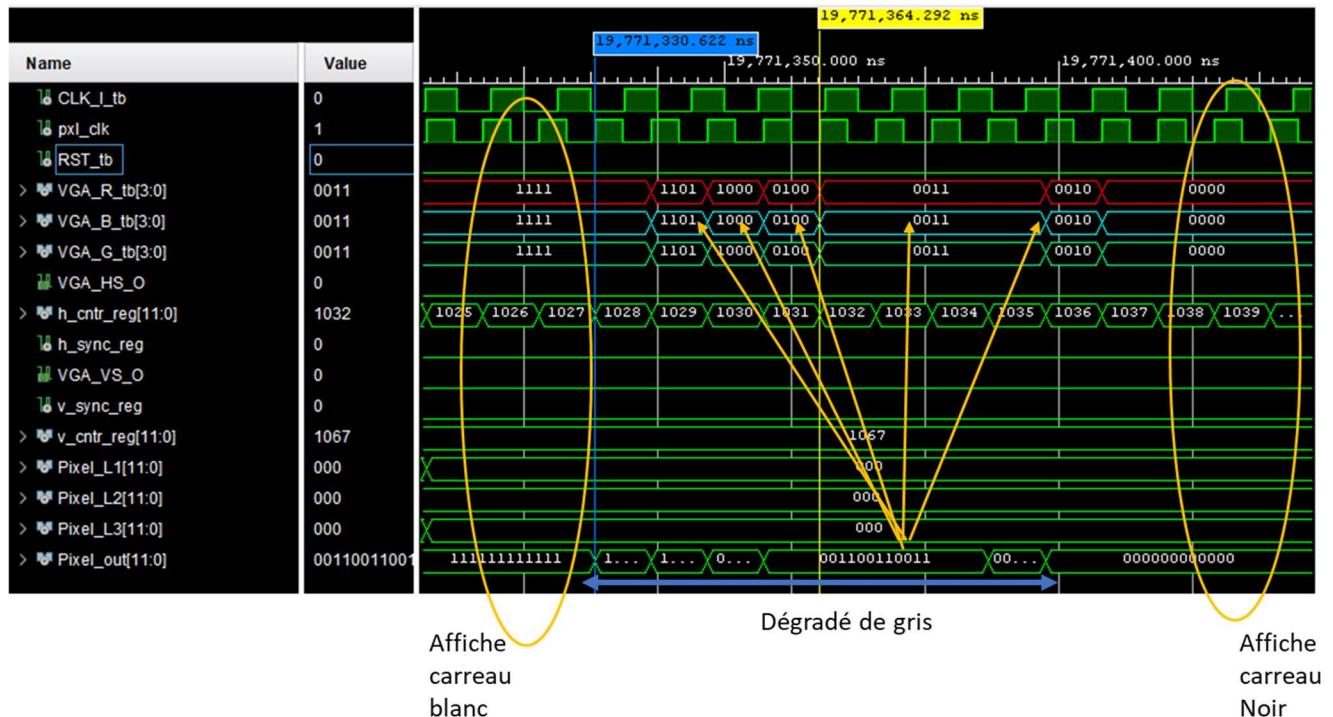
5. Résultat de test du dégradé de gris dans les bords après l'ajout du filtre

Résultats du test :

Test N°13 de la dégradation de gris dans les bords après l'ajout du filtre			
<u>Résumé:</u> Tests permettant de s'assurer que la palette de notre image en niveaux de gris est constituée par les 16 couleurs comprises entre 0,0 et 15,15,15 qui y sont sauvegardées donc <u>16 nuances de gris</u>			
N° d'étape	Actions de pas:	Résultats attendus:	Résultat obtenu
1	Vérifier le nombre de dégradation de la couleur gris	Regarder le niveau de dégradé de gris entre un carreau blanc et noir (fff et 000 en hexa)	OK
2	Regarder le nombre de cycle d'horloge de décalage	Regarder les signaux h_decal et v_decal sont ajoutés aux calculs de h_sync et v_sync	OK
<u>Cahier d'exigences</u>	<u>EXIGENCE_display_area_07</u>		✓ PASS

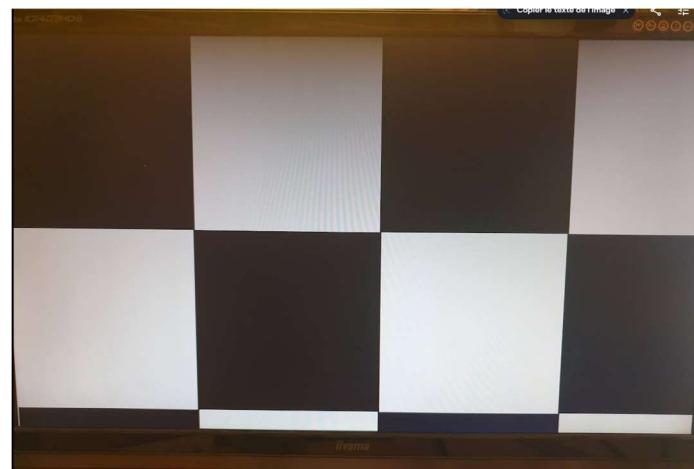
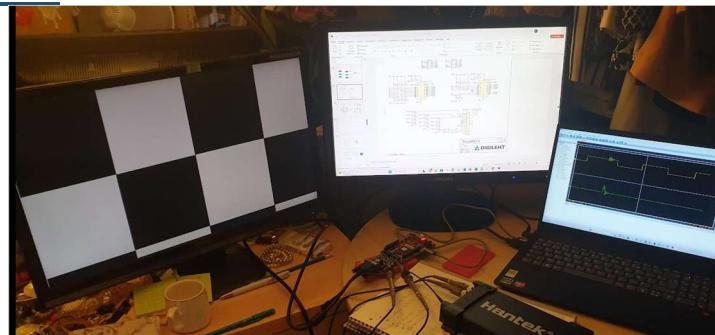
Démonstration :





6. Résultat de test global de la phase 2 « contrôleur_VGA » avec filtre

Observations sur écran :



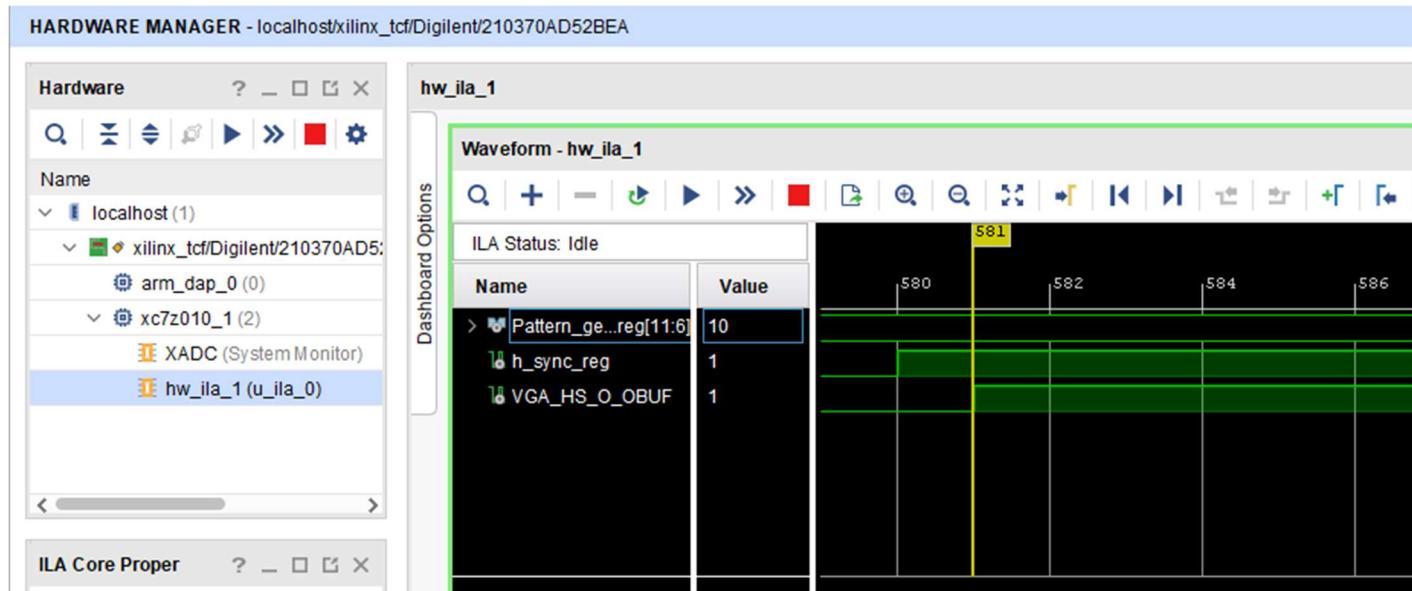
On note des parties en bleu ciel dans les transitions liées à l'application du filtre. Ceci a été corrigé en séparant l'application du filtre sur chaque composante de couleur pour éviter les erreurs d'arrondi.

IV. Moyen de test 2 :test sur cible -set up -debug - ILA

Résultats du test :

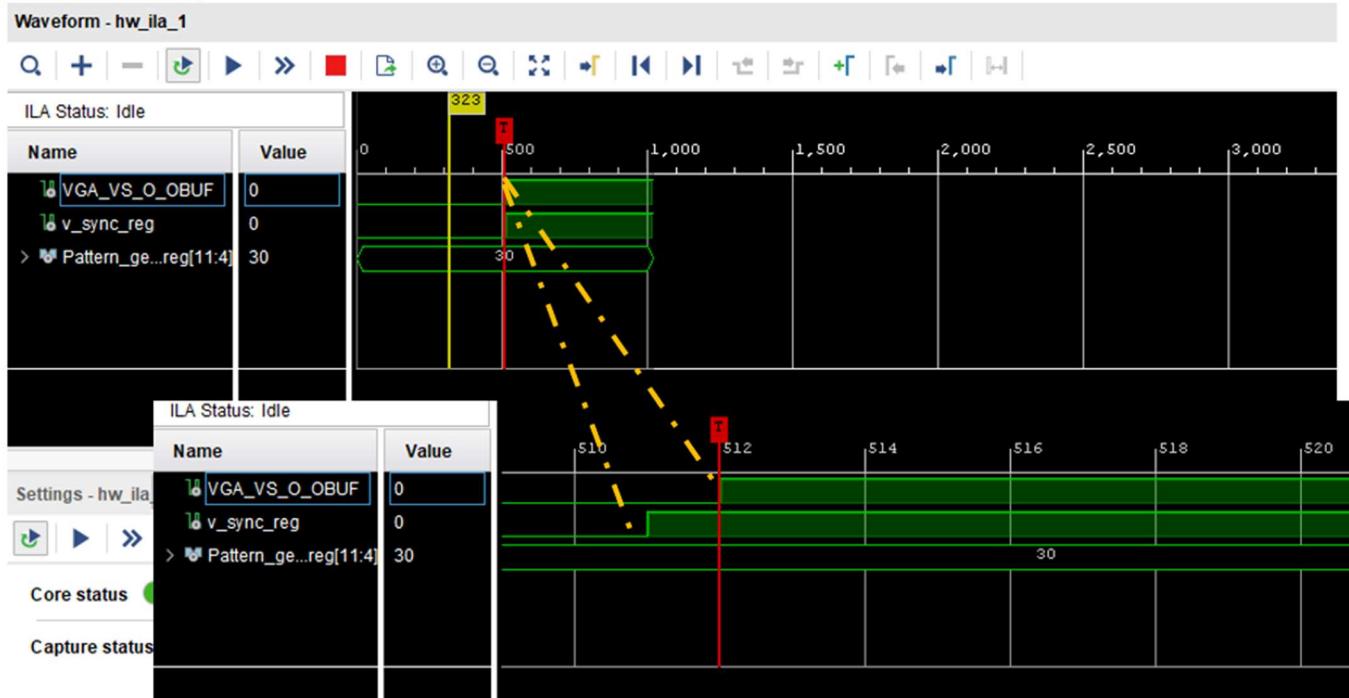
Test N°15 sur cible horizontal_timing_gen_03 : Vérification de la synchronisation horizontale			
<p><u>Résumé</u> : Tests permettant de s'assurer que la fonction <code>horizontal_timing_gen_03</code> génère bien les signaux logiques de synchronisation horizontale</p> <p><u>Quoi</u> : validation sur cible de signal de synchronisation horizontale (<code>hsync</code>)</p> <p><u>Comment/pré-requis</u> : Réaliser un test ILA de la fonction <code>horizontal_timing_gen_03</code></p>			
<u>N° d'étape</u>	<u>Actions de pas :</u>	<u>Résultats attendus :</u>	<u>Résultat obtenu</u>
1	Lancer la simulation de la synchronisation horizontale (<code>hsync</code>)	Mettre un trigger sur un front montant de la sortie de VGA-HS-OBUF → Observer le décalage de registre <code>h_cnt-reg</code> pour assurer la synchronisation	OK
2	Vérifier le rafraîchissement de l'écran (image entier) à 60 Hz	Regarder le temps entre deux triggers placés sur <code>h_cnt=0</code> et <code>v_cnt=0</code> (Le temps doit être = 1/60 s)	Non effectué
<u>Cahier d'exigences</u>	<u>EXIGENCE_horizontal_timing_gen_03</u>		✓ PASS

Démonstration :



Résultats du test :

Test N°16 test fonctionnel sur cible vertical_timing_gen_04 : Vérification de la synchronisation verticale			
<u>Résumé :</u> Tests permettant de s'assurer que la fonction vertical_timing_gen_04 génère bien les signaux logiques de synchronisation verticale			
<u>Quoi :</u> validation sur cible de signal de synchronisation horizontale (vsync)			
<u>Comment/pré-requis :</u> Réaliser un test ILA de la fonction vertical_timing_gen_04			
N° d'étape	<u>Actions de pas :</u>	<u>Résultats attendus :</u>	<u>Résultat obtenu</u>
1	Lancer la simulation de la synchronisation verticale (vsync)	Mettre un trigger sur un front montant de la sortie de VGA-VS-OBUF → Observer le décalage de registre h_cnt-reg pour assurer la synchronisation	OK
2	Vérifier le rafraîchissement vertical à 31.46875 kHz	Regarder le temps entre deux triggers placés sur h_cnt=800 et v_cnt=2 (Le temps doit être = 1/31.46875 kHz)	Non effectué
<u>Cahier d'exigences</u>	<u>EXIGENCE_vertical_timing_gen_04</u>		✓ PASS

Démonstration :

Résultats du test :

Test N°17 Vérification de la génération des signaux, sortie de calcul et sorties des FIFOs

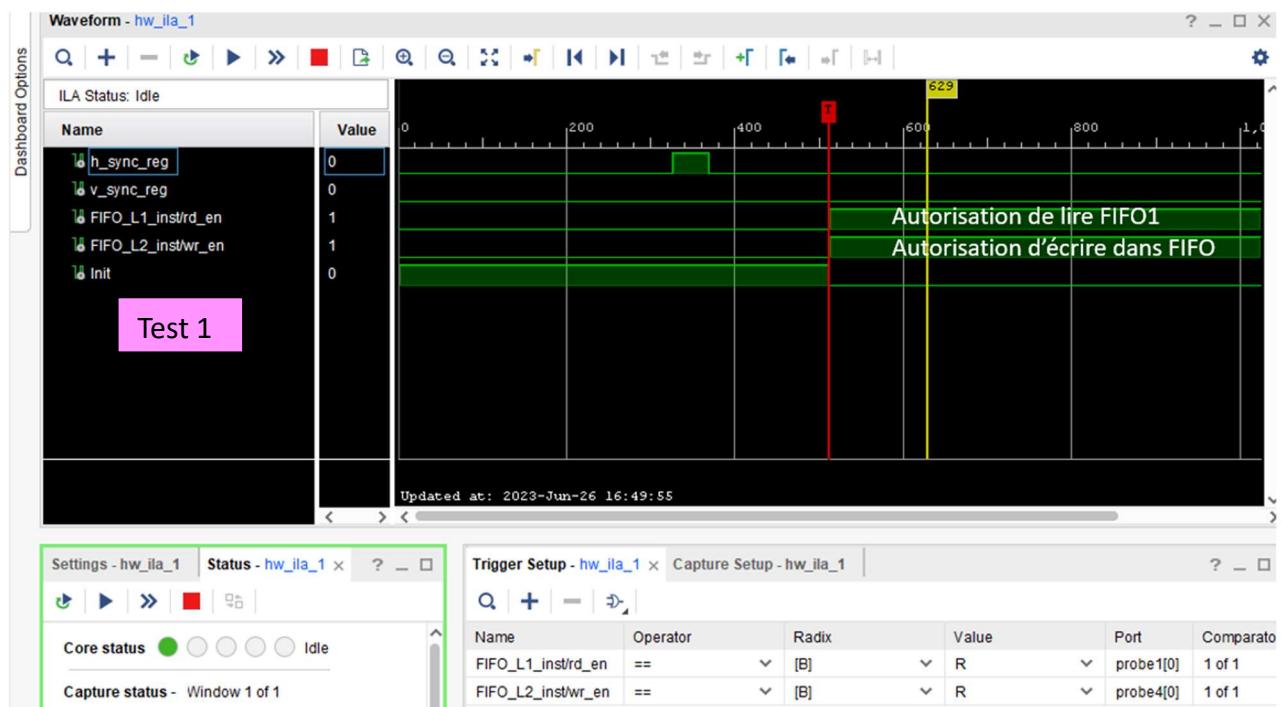
Résumé : Tests permettant de s'assurer que le composant contrôleur_VGA génère bien les signaux logiques de synchronisation horizontale et verticale (hsync et vsync) et le bon calcul après ajout de filtre.

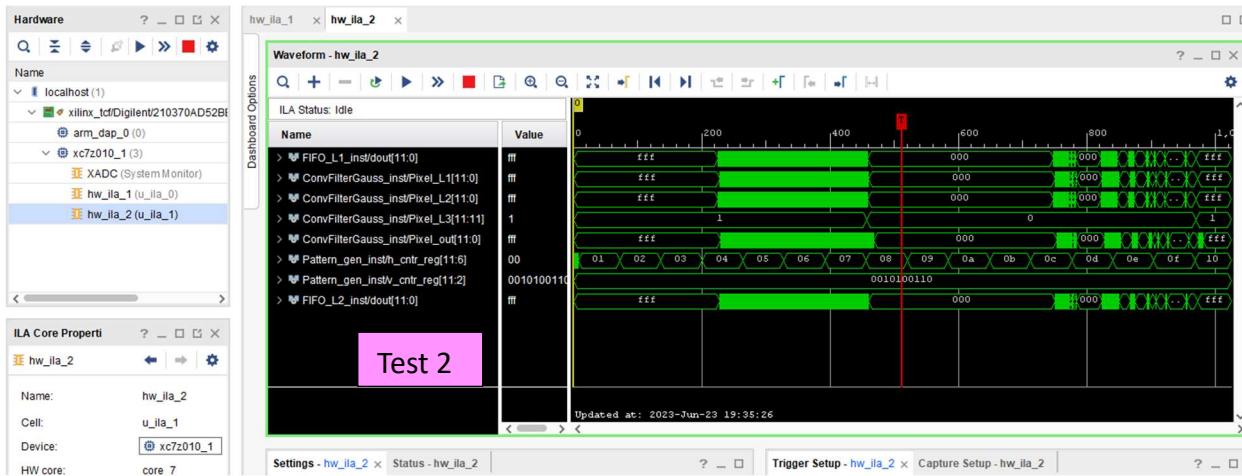
Quoi : validation sur cible des signaux logiques de synchronisation horizontale et verticale (hsync et vsync)

Comment/pré-requis : Réaliser un test ILA de la fonction timing_gen_02, FIR_Filter_08,

N° d'étape	Actions de pas :	Résultats attendus :	Résultat obtenu
1	Vérifier la synchronisation horizontale (hsync), avec la sortie de calcul du filtre et les sorties des Fifo	Mettre un trigger sur un front montant : FIFO1 read-enable et FIFO2 write-enable en regardant le h-sync	Ok
2	Vérifier la sortie du filtre (calcul)	Mettre un trigger sur les sorties des PIXEL L1 PIXEL L2 et PIXEL L3	Ok
<u>Cahier d'exigences</u>	<u>EXIGENCE_timing_gen_02, FIR_Filter_08</u>		✓ PASS

Démonstration :





Résultats du test :

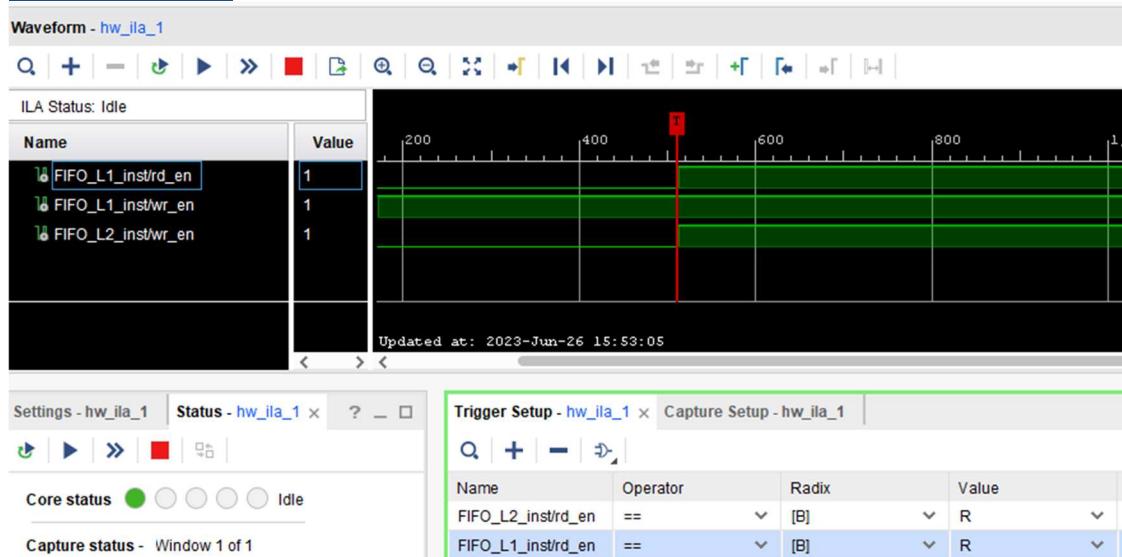
Test N°18 ILA test des entrées / sorties du Conv-filter-Gauss (calcul)

Résumé : Tests permettant de s'assurer que la fonction FIR_Filter_08 remplit son rôle de filtre de matrice 3x3

Comment/pré-requis : Paramétriser l'ILA avec les signaux à observer et à trigger.

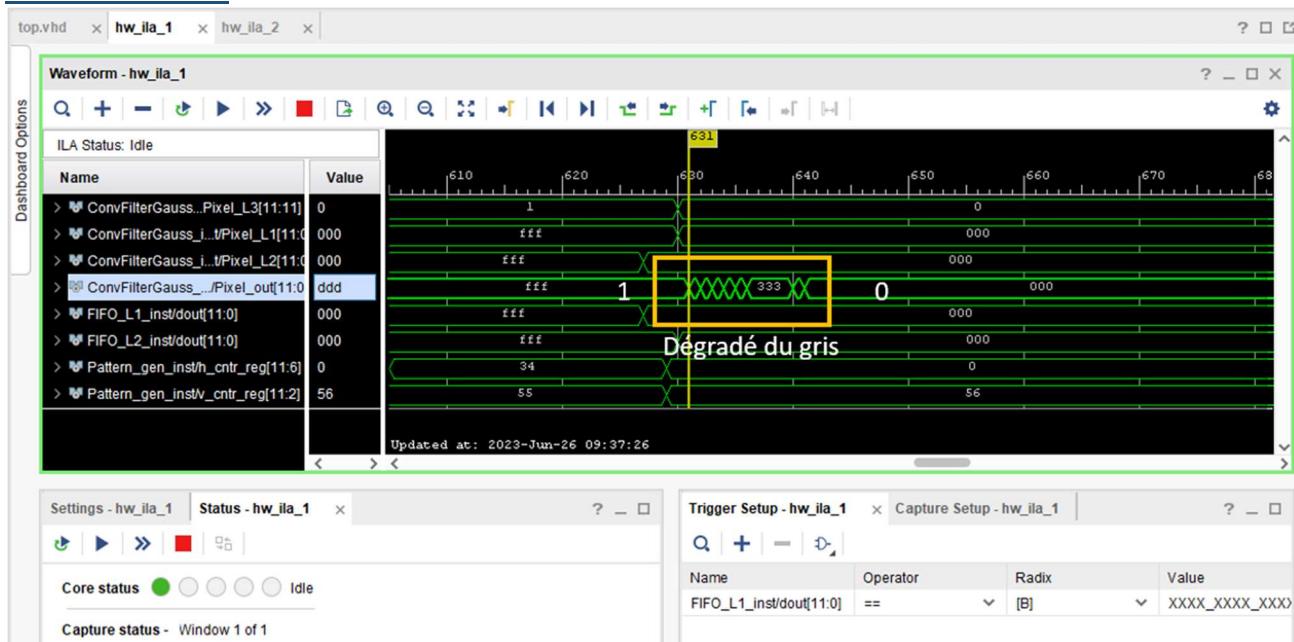
N° d'étape	Actions de pas :	Résultats attendus :	Résultat obtenu
1	Vérifier les conditions d'écriture dans la FIFO_1 et FIFO_2	Regarder quand (FIFO_2_write_ena) est active : la 1e ligne est entièrement lue (Init passe à 0) et la ligne 2 en train d'être lue (par FIFO_1)	ok
2	Vérifier le flux de l'autorisation de lecture de la FIFO_1 et FIFO_2	Regarder (FIFO_2_read_ena) est active : quand la deuxième ligne est entièrement lue	ok
<u>Cahier d'exigences</u>		EXIGENCE_FIR_Filter_08	
		✓ pass	

Démonstration :

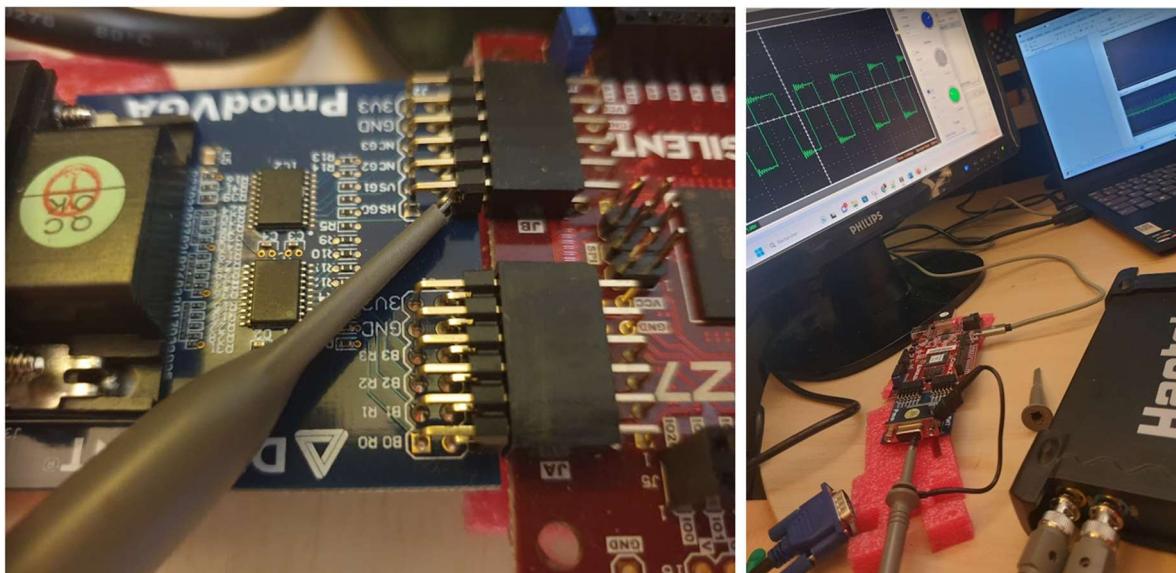
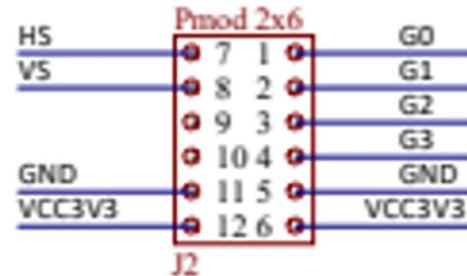
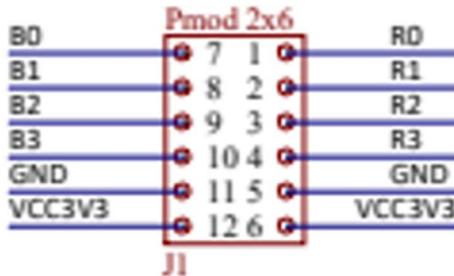


Résultats du test :**Test N°19** *ILA test dégradée du gris sur ILA*Résumé : Tests permettant de voir les niveaux de gris suite à l'application du filtre de matrice 3x3Comment/pré-requis : Paramétriser l'ILA avec les signaux à observer et à trigger.

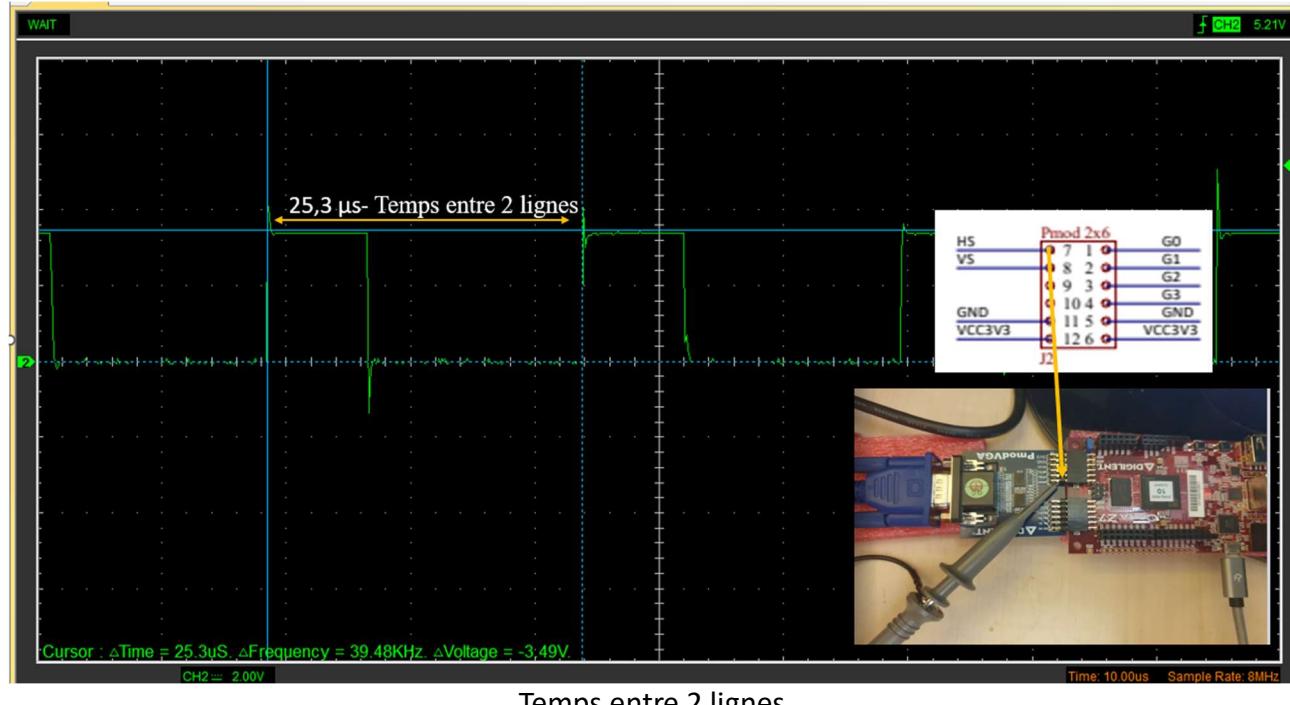
<u>N° d'étape</u>	<u>Actions de pas :</u>	<u>Résultats attendus :</u>	<u>Résultat obtenu</u>
1	Vérifier les niveaux de degradés du gris	Regarder combien de Niveau de gris	OK
<u>Cahier d'exigences</u>	<u>EXIGENCE_FIR_Filter_08</u>		✓ pass

Démonstration :

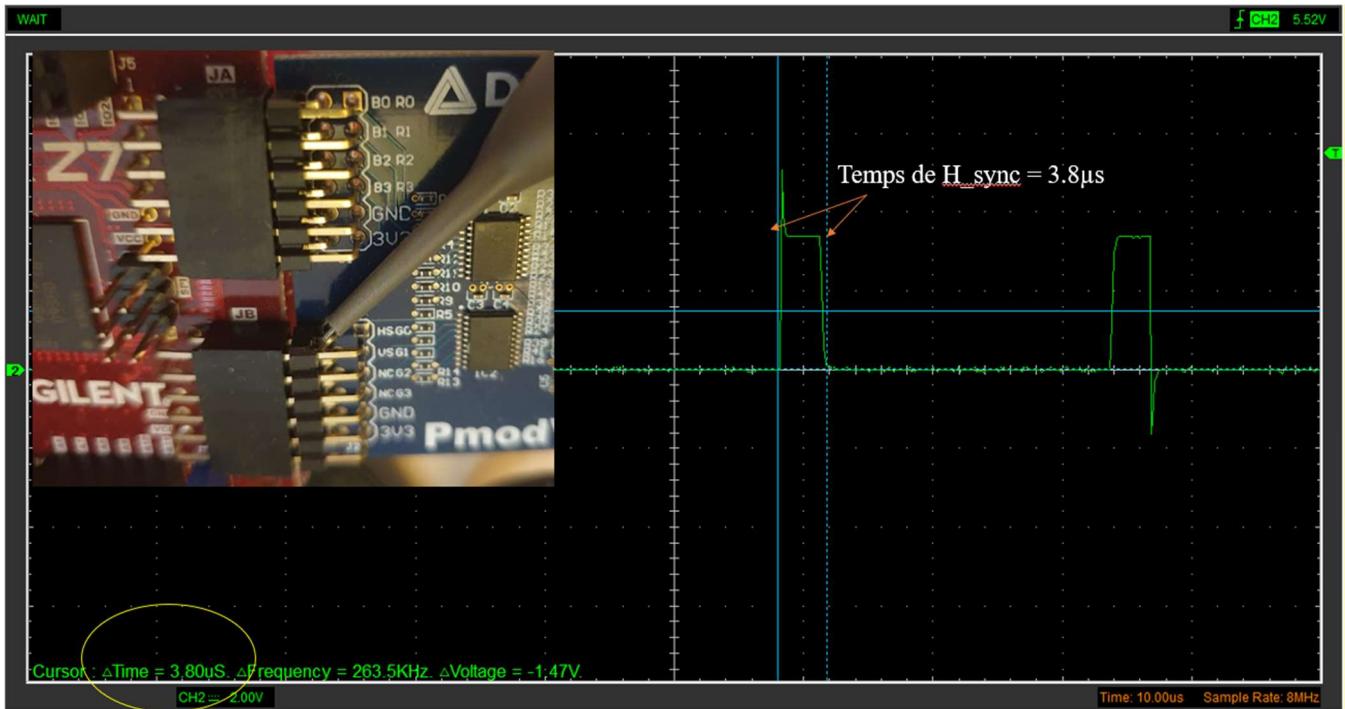
V. MOYEN DE TEST 3 : test sur cible - oscilloscope



Test N°20 Mesure du Temps entre 2 lignes (h_sync dans PIN 7 du pmodVGA)

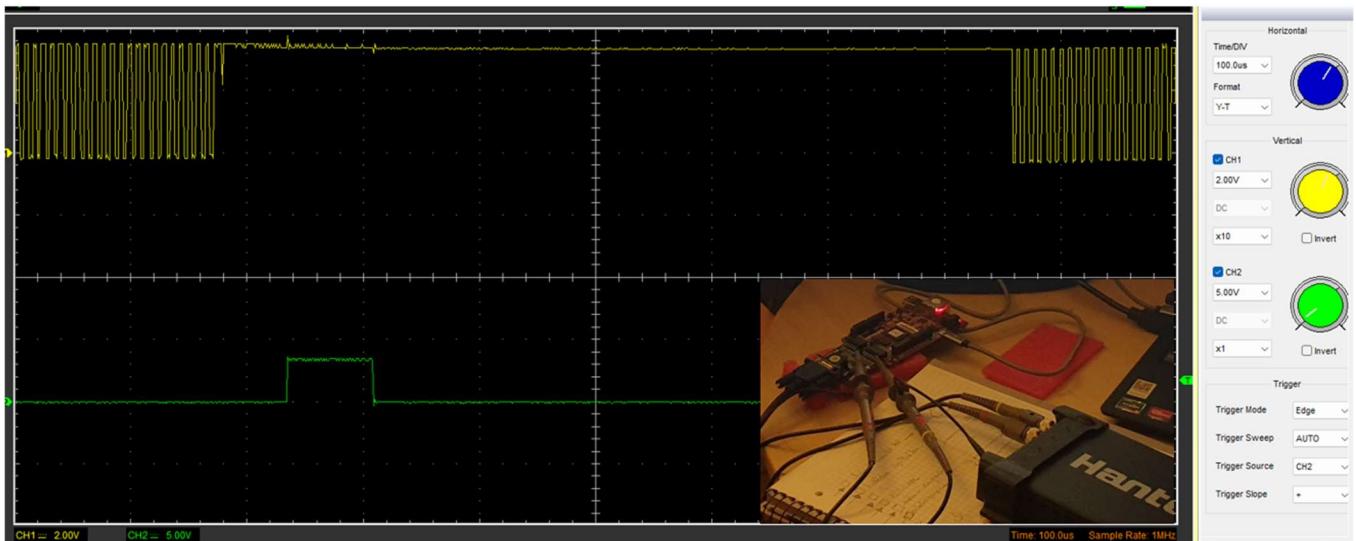


Test N°21 Mesure du Temps de H_sync = 3.8μs (h_sync dans PIN 7 du pmodVGA)

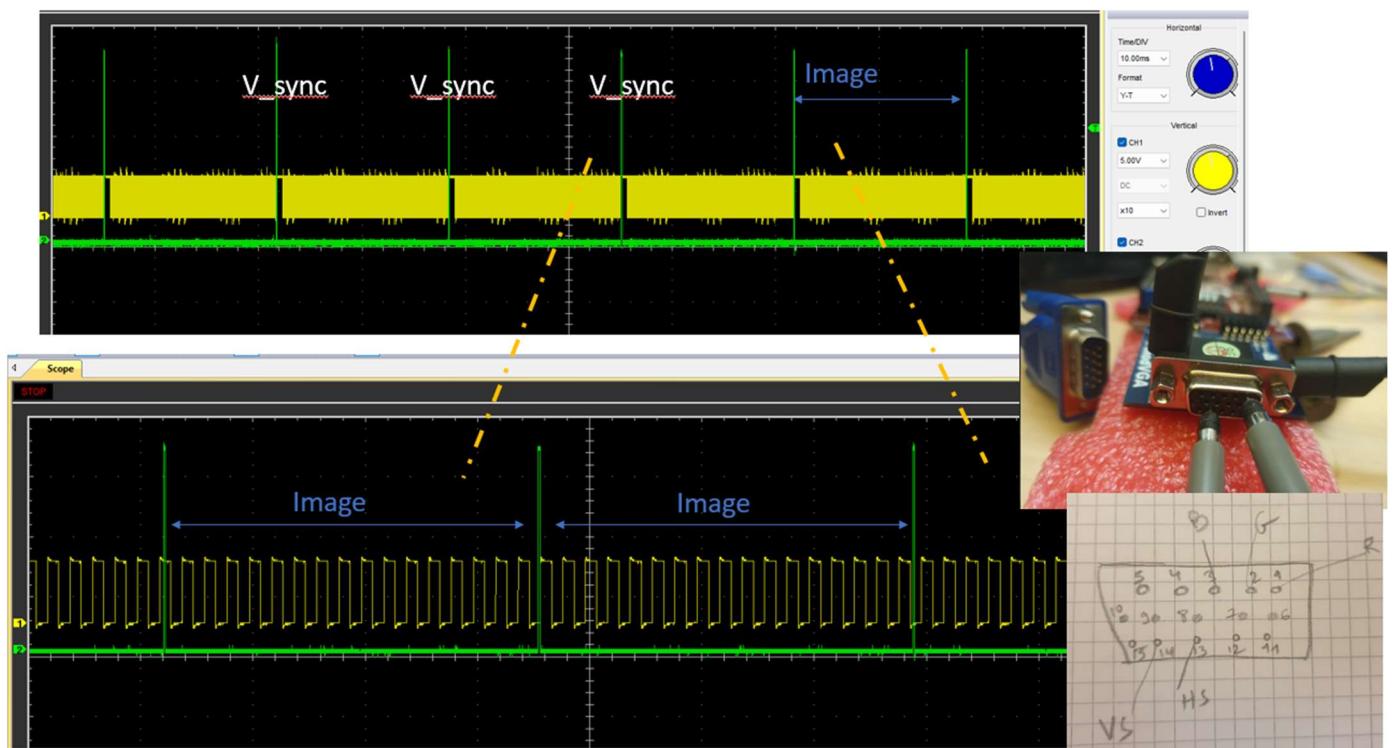


Temps de H_sync = 3.8μs

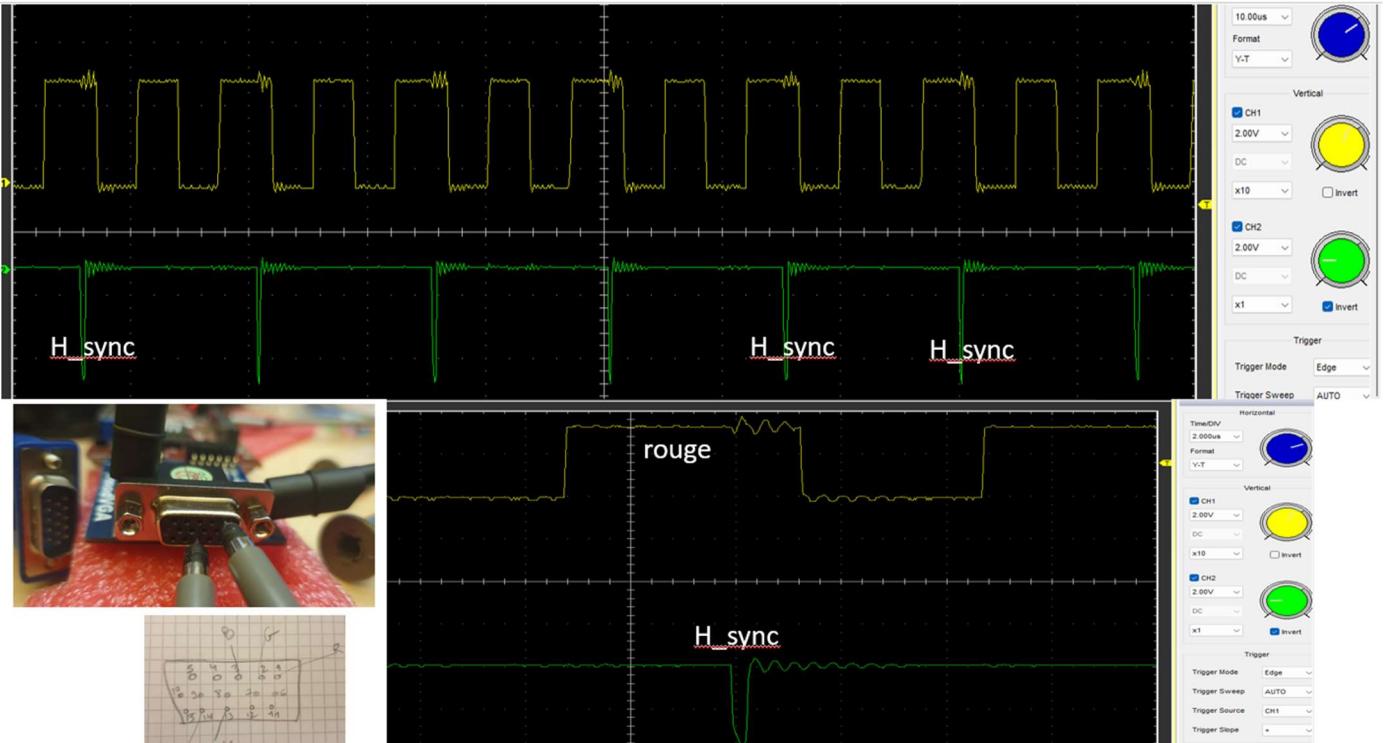
Test N°22 Mesure pulse H_PW (v_sync dans PIN 14 du VGA et PIN1du rouge)



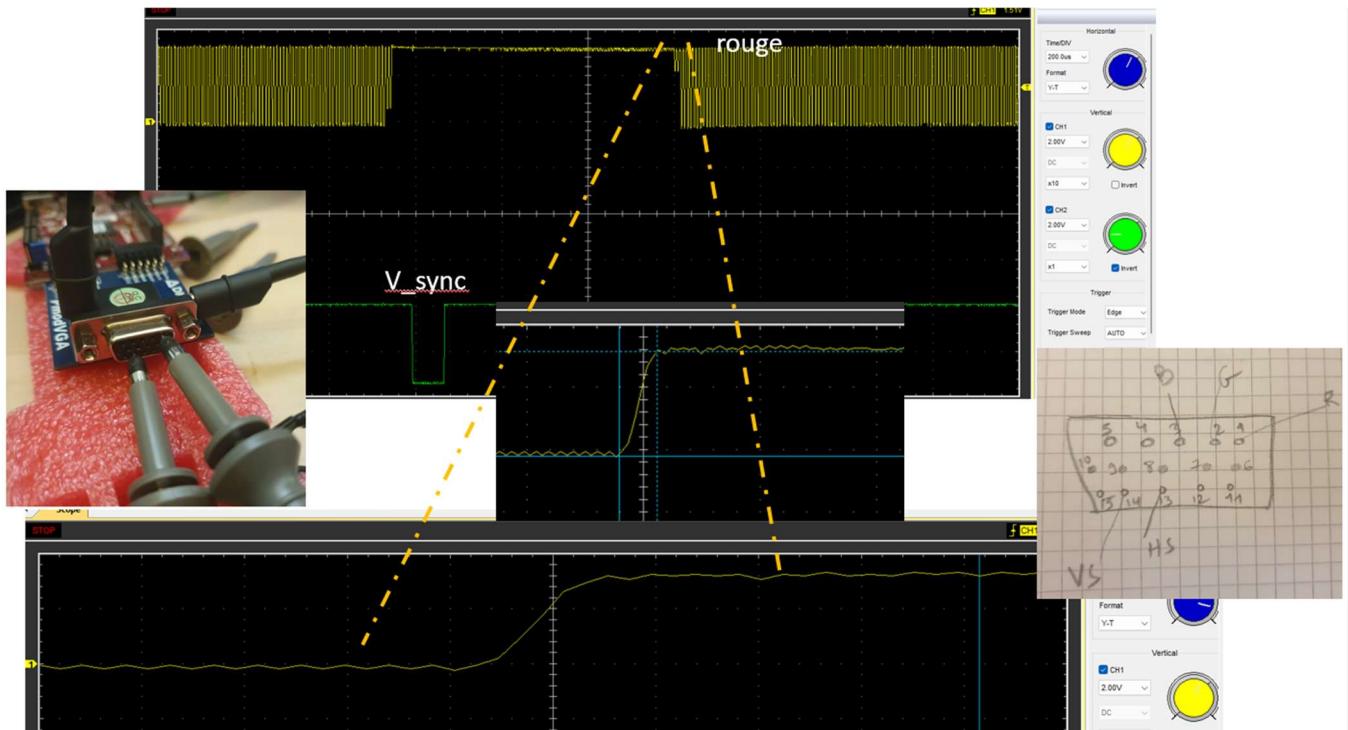
Test N°23 Mesure de V-SYNC (v_sync dans PIN 14 du VGA et PIN1 du rouge)



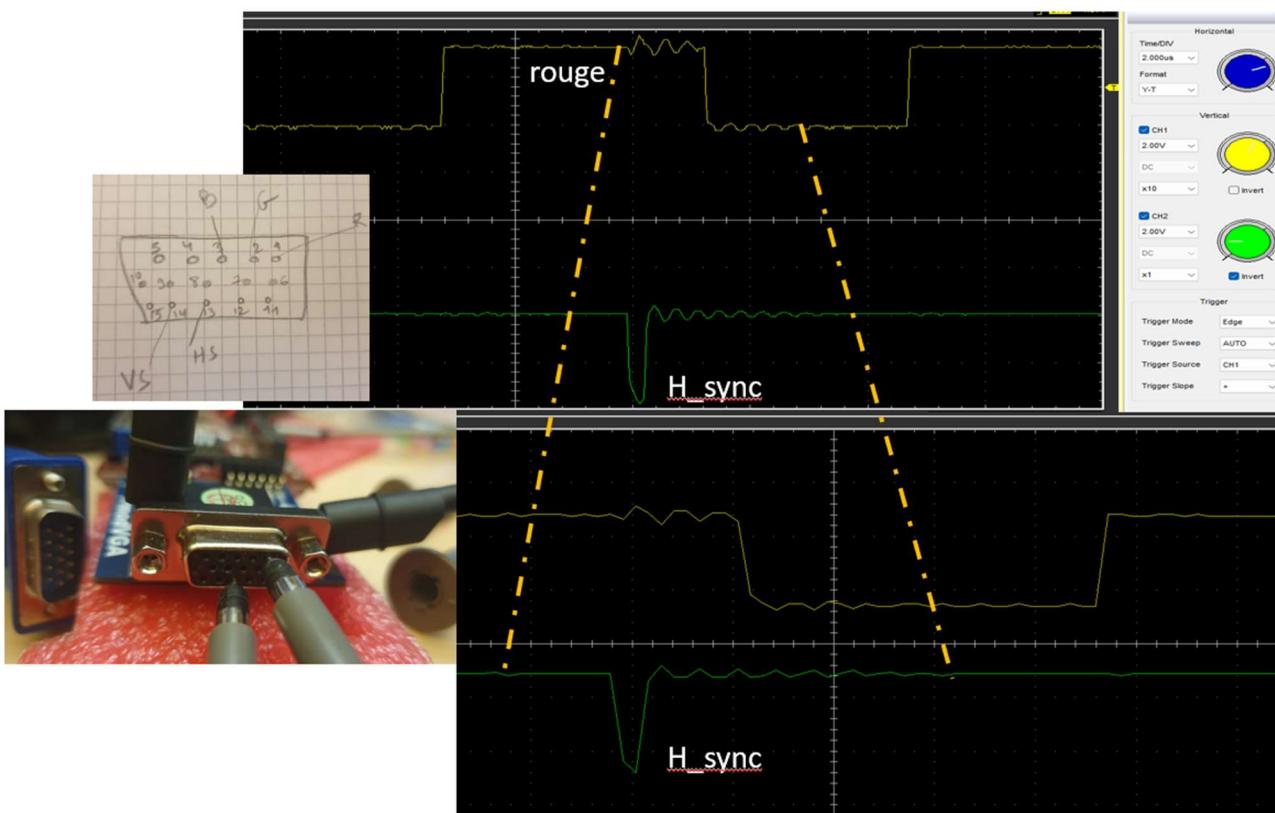
Test N°24 Mesure de V-SYNC (h_sync dans PIN 13 du VGA et PIN1 du rouge)



Test N°25 Mesure de dégradé du gris avec V-SYNC (v_sync dans PIN 14 du VGA et PIN1 du rouge)

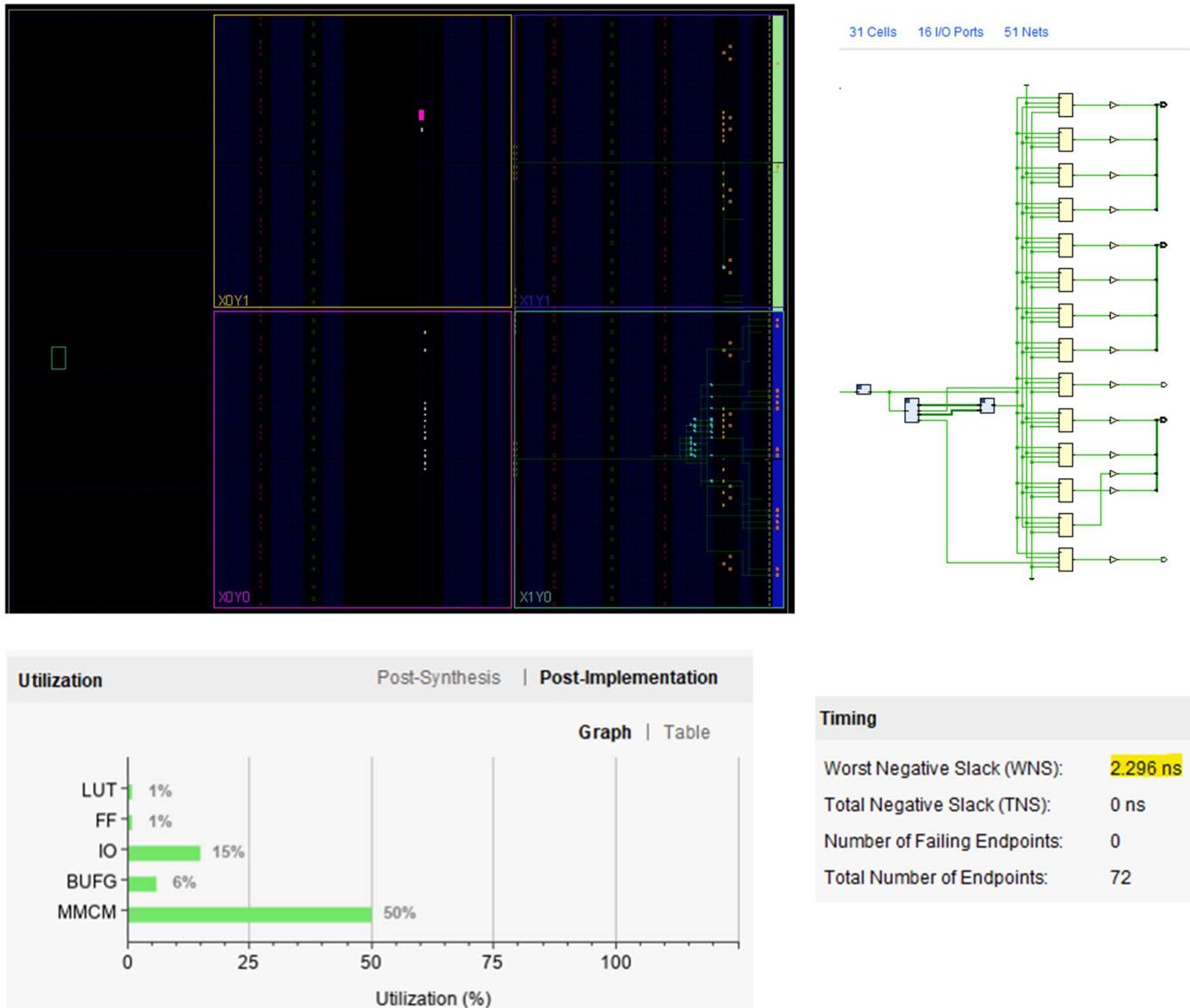


Test N°26 Mesure de dégradé du gris avec H-SYNC (h_sync dans PIN 13 du VGA et PIN1 du rouge)



VI. Annexe :

1. Annexe 1 : Phase 1 : Placement / routage



2. Annexe 2 : Phase 2 : Placement / routage

IMPLEMENTED DE SIGN - xc7z010clg400-1

Hardware Device Properties

Select an object to see properties

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	-0.307 ns	Worst Hold Slack (WHS):	0.050 ns	Worst Pulse Width Slack (WPWS):	2,000 ns
Total Negative Slack (TNS):	-1,034 ns	Total Hold Slack (THS):	0,000 ns	Total Pulse Width Negative Slack (TPWNS):	0,000 ns
Number of Failing Endpoints:	11	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	1222	Total Number of Endpoints:	1222	Total Number of Endpoints:	336

Timing constraints are not met.

3. Annexe 3 : calcul des Résultats obtenus du Conv-filter-Gauss:

	rouge 4 bits	bleu 4 bits	vert 4 bits	Masque Gauss	rouge * gaus	bleu * gauss	vert * gauss
P11	15	15	15	1	15	15	15
P12	15	15	15	2	30	30	30
P13	15	15	15	1	15	15	15
P21	15	15	15	2	30	30	30
P22	15	15	15	4	60	60	60
P23	15	15	15	2	30	30	30
P31	15	15	15	1	15	15	15
P32	15	15	15	2	30	30	30
P33	15	15	15	1	15	15	15
				somme	240	240	240
				division par 16	15	15	15
				conversion en 12 bits			4095
}							
	rouge 4 bits	bleu 4 bits	vert 4 bits	Masque Gauss	rouge * gaus	bleu * gauss	vert * gauss
P11	15	15	15	1	15	15	15
P12	15	15	15	2	30	30	30
P13	0	0	0	1	0	0	0
P21	15	15	15	2	30	30	30
P22	15	15	15	4	60	60	60
P23	15	15	15	2	30	30	30
P31	15	15	15	1	15	15	15
P32	15	15	15	2	30	30	30
P33	0	0	0	1	0	0	0
				somme	210	210	210
				division par 16	13,125	13,125	13,125
				conversion en 12 bits			3549
}							
	rouge 4 bits	bleu 4 bits	vert 4 bits	Masque Gauss	rouge * gaus	bleu * gauss	vert * gauss
P11	15	15	15	1	15	15	15
P12	15	15	15	2	30	30	30
P13	0	0	0	1	0	0	0
P21	15	15	15	2	30	30	30
P22	0	0	0	4	0	0	0
P23	0	0	0	2	0	0	0
P31	15	15	15	1	15	15	15
P32	15	15	15	2	30	30	30
P33	15	15	15	1	15	15	15
				somme	135	135	135
				division par 16	8,4375	8,4375	8,4375
				conversion en 12 bits			2184

	rouge 4 bits	bleu 4 bits	vert 4 bits	Masque Gauss	rouge * gaus	bleu * gauss	vert * gauss
P11	15	15	15	1	15	15	15
P12	0	0	0	2	0	0	0
P13	0	0	0	1	0	0	0
P21	0	0	0	2	0	0	0
P22	0	0	0	4	0	0	0
P23	0	0	0	2	0	0	0
P31	15	15	15	1	15	15	15
P32	15	15	15	2	30	30	30
P33	15	15	15	1	15	15	15
				somme	75	75	75
				division par 16	4,6875	4,6875	4,6875
				conversion en 12 bits			1092
	rouge 4 bits	bleu 4 bits	vert 4 bits	Masque Gauss	rouge * gaus	bleu * gauss	vert * gauss
P11	15	15	15	1	15	15	15
P12	0	0	0	2	0	0	0
P13	0	0	0	1	0	0	0
P21	0	0	0	2	0	0	0
P22	0	0	0	4	0	0	0
P23	0	0	0	2	0	0	0
P31	15	15	15	1	15	15	15
P32	15	15	15	2	30	30	30
P33	0	0	0	1	0	0	0
				somme	60	60	60
				division par 16	3,75	3,75	3,75
				conversion en 12 bits			819