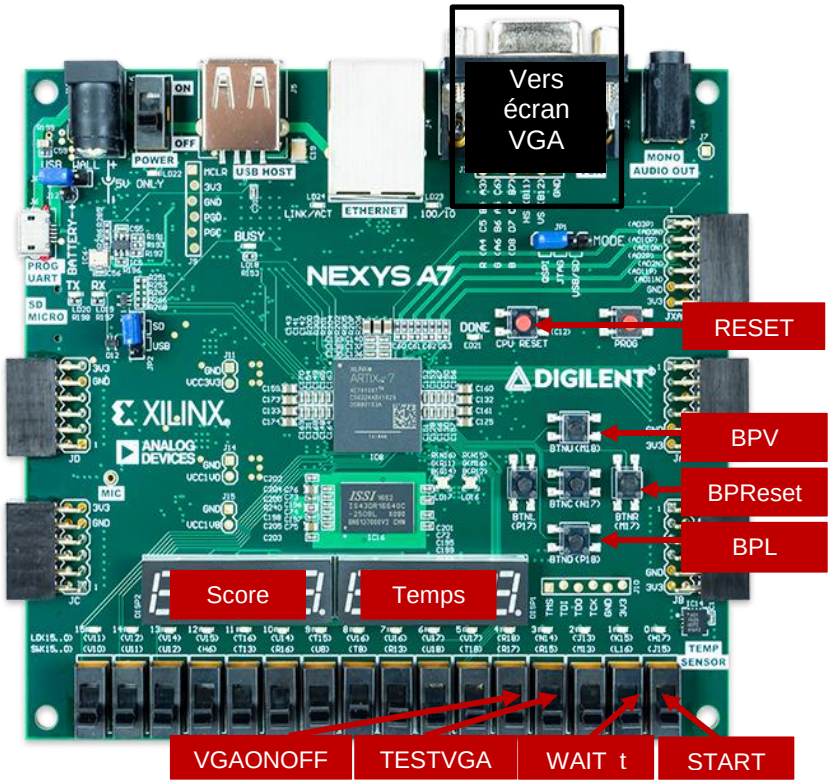
ELN2 : PROJET SCORING 2.0

Affichage de la durée et du score d'un match de football à l'aide d'un FPGA



Phase 3 : Gestion d'un écran VGA

La **phase 3** du projet **Scoring 2.0** est consacrée à l'affichage sur écran **LCD** compatible de la norme **VGA** à partir du sous-bloc **vgaDislay**. Ce sous-bloc fait appel à deux machines d'états générant un signal de synchronisation horizontale de l'écran, **HSYNCH** et un signal de synchronisation verticale **VSYNCH**. Ces machines d'états sont associées à deux compteurs complexes faisant l'objet de cette dernière phase du projet.

1 Rappels sur l'architecture de chronoscore

1.1 TOP Module

La vue de niveau « TOP » du système **chronoscore** est donnée à la figure 1.

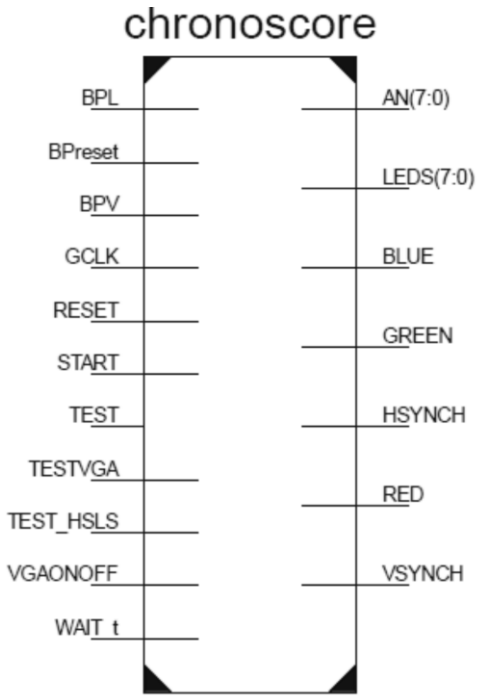


*Figure 1 : Vue TOP de l'entité **chronoscore***

1.2 Entrées et sorties du système

Le tableau 1, précise les signaux d'entrée de **chronoscore**, leur direction, leur localisation sur la carte **NEXYS A7** (voir « Présentation générale du projet » figure 2) et le port d'entrée sur le **FPGA**.

Signal	**Direction du signal**	**Elément de la carte NEXYS A7**	**Port du FPGA** *(fichier .ucf)*	**Description**
GCLK	Input	Quartz	E3	Oscillateur **Epson SG-8002JF** générant une fréquence de **100 MHz**
START	Input	SW0	J15	Démarrage du chronomètre
WAIT_t	Input	SW1	L16	Mise en pause du chronomètre
RESET	Input	BTN CPU RESET	C12	Remise à 0 du chronomètre
BPL	Input	BTNU	P18	Incrémentation du score pour l'équipe locale
BPV	Input	BTND	M18	Incrémentation du score pour l'équipe des « visiteurs »
BPreset	Input	BTNR	M17	Remise à zéro du score
VGA ONOFF	Input	SW4	R17	Activation de l'écran VGA
TEST VGA	Input	SW3	R15	Activation des images de test de l'écran VGA
TEST_ HSLS	Input	SW2	M13	Réservé
TEST	Input	BTNC	N17	Réservé

*Tableau 1 : Description des signaux d'entrée de **chronoscore***

Le tableau 2, précise les signaux de sortie de **chronoscore**, leur direction, l'élément concerné sur la carte **NEXYS A7** (voir « Présentation générale du projet » figure 2) et le port d'entrée sur le **FPGA**.

Signal	Direction du signal	Elément de la carte NEXYS A7	Port du FPGA *(fichier .ucf)*	Description
AN(0)	Output	Afficheurs 7-segments (anodes)	J17	AN[7 :0] : signaux de commande des anodes des afficheurs
AN(1)	Output		J18	
AN(2)	Output		T9	
AN(3)	Output		J14	
AN(4)	Output		P14	
AN(5)	Output		T14	
AN(6)	Output		K2	
AN(7)	Output		U13	
LEDS(0)	Output	Afficheurs 7-segments (cathodes)	T10	LEDS[6 :0] : signaux de commande des segments « a » à « g » LEDS[7] : signal de commande du point
LEDS(1)	Output		R10	
LEDS(2)	Output		K16	
LEDS(3)	Output		K13	
LEDS(4)	Output		P15	
LEDS(5)	Output		T11	
LEDS(6)	Output		L8	
LEDS(7)	Output		H15	
HSYNCH	Output	Port VGA	B11	Synchronisation horizontale
VSYNCH	Output		B12	Synchronisation verticale
RED	Output		A4	Contrôle des pixels de couleur « rouge »
BLUE	Output		B6	Contrôle des pixels de couleur « bleu »
GREEN	Output		D8	Contrôle des pixels de couleur « vert »

*Tableau 2 : Description des signaux de sortie de **chronoscore***

1.3 Décomposition de **chronoscore** en sous-blocs

Le système global **chronoscore** a été décomposé en cinq sous-blocs représentés figure 2.

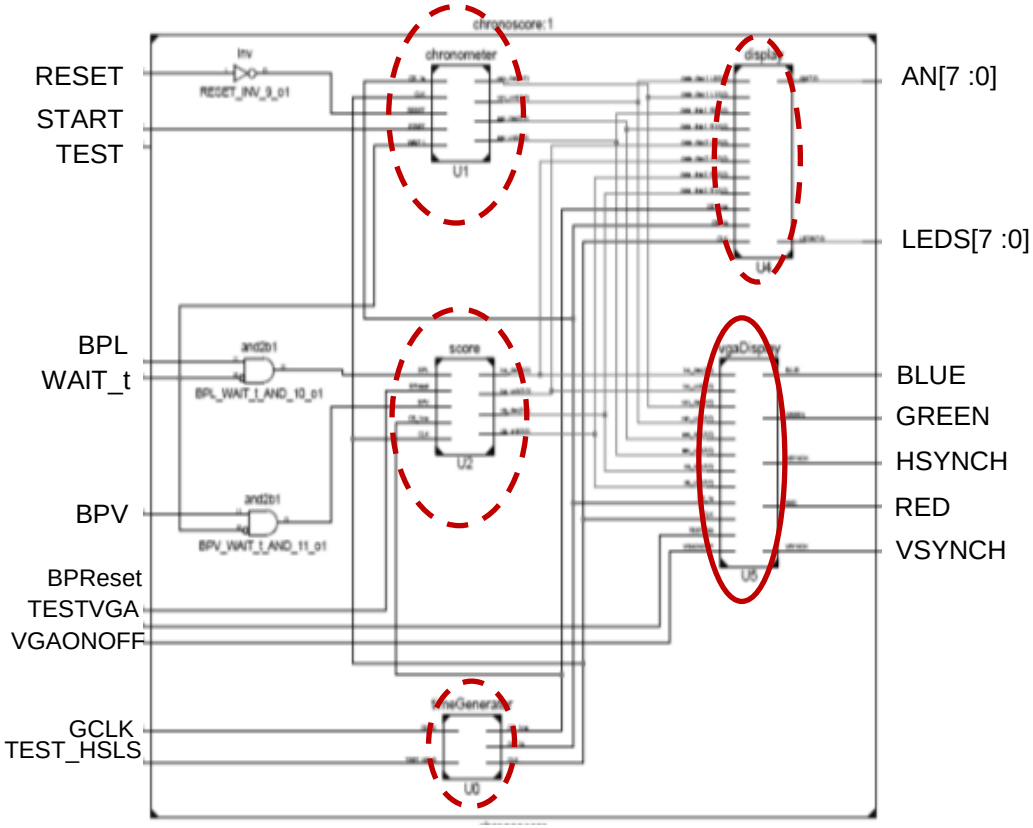


*Figure 2 : Architecture interne de l'entité **chronoscore***

La phase 3 du projet met en œuvre les sous-blocs **timeGenerator, display**, **chronometer, score** et **vgaDisplay.**

1.4 Sous-blocs **timeGenerator** (rappel)

Le synoptique détaillé de **timeGenerator** est rappelé figure 3 (pour la description des signaux CE_1ms et CE_1s, se reporter au document « Présentation générale du projet »).

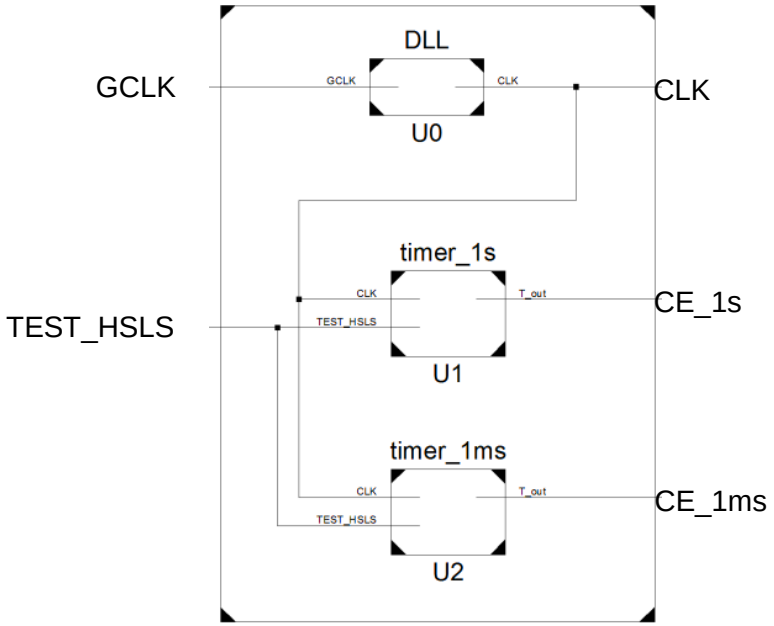


*Figure 3 : Architecture du sous-bloc **timeGenerator***

Pour simplifier la mise en œuvre du projet, une version exécutable de **timeGenerator** (timeGenerator.ngc) est **fournie** sur **CPe-Campus**.

1.5 Sous-bloc display (rappel)

Le synoptique détaillé de **display** est rappelé figure 4 (une version pleine page est donnée en annexe 1)**.**

Les quatre afficheurs de droite de la carte NEXYS A7 (disp1 : afficheurs 0 à 3) doivent indiquer le temps écoulé en minutes et secondes. Le point de l'afficheur 2 clignote au rythme de la seconde.

Les quatre afficheurs de gauche (disp2 : afficheurs 4 à 7) doivent indiquer le score de l'équipe locale et celui de l'équipe des visiteurs.

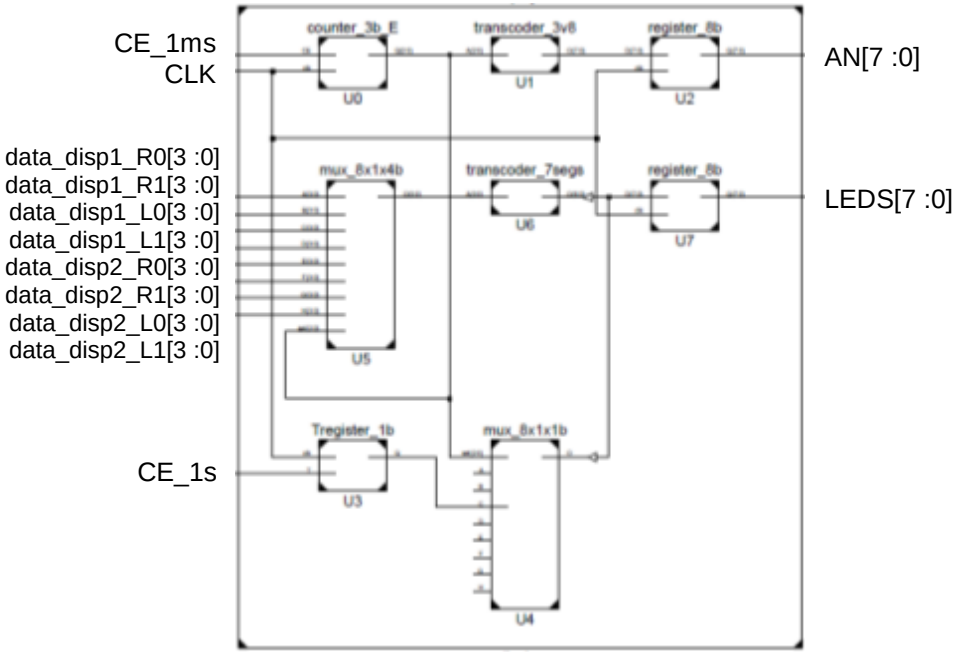


*Figure 4 : Architecture du sous-bloc **display***

Une version exécutable de l'entité **display** (display.ngc) est **fournie** sur **CPe-Campus**.

Une version exécutable de l'entité **chronometer** (chronometer.ngc) est **fournie** sur **CPe-Campus**.

1.7 Sous-bloc **chronometer** (rappel)

Le sous-bloc **chronometer** fournit aux sous-blocs **display** et **vgaDisplay** les données temporelles (minutes et secondes) à afficher.

Le synoptique détaillé de **chronometer** est donné figure 5 (une version pleine page est donnée en annexe 2).

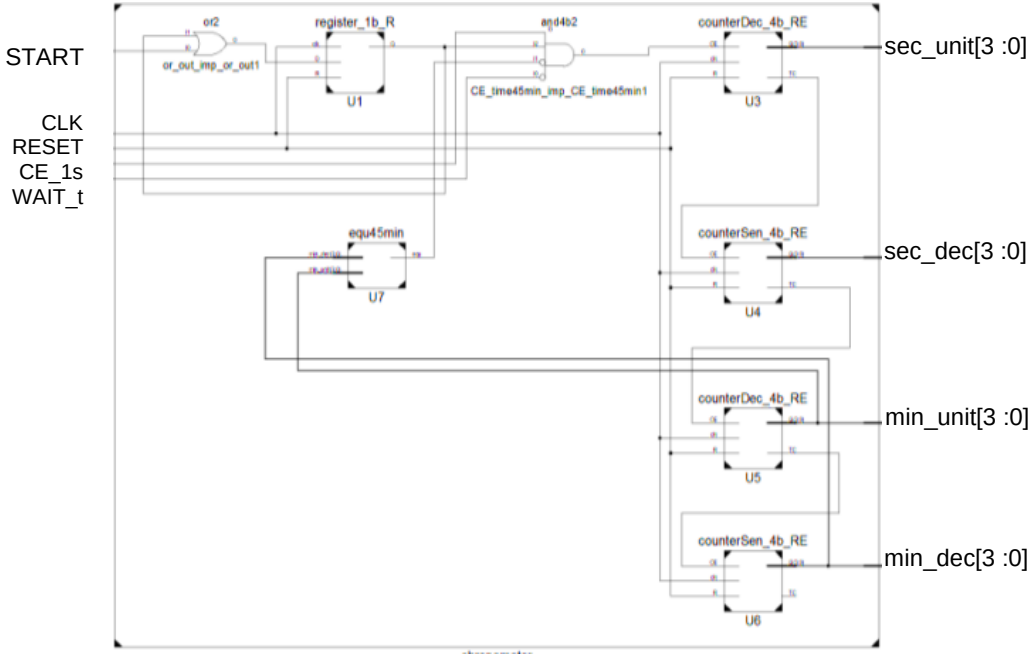


*Figure 5 : Architecture de l'entité **chronometer***

1.8 Sous-bloc **score** (rappel)

Le sous-bloc **score** fournit aux sous-blocs **multiplexData** et **vgaDisplay** les scores des équipes (unités et dizaines) à afficher.

Le synoptique détaillé de **score** est donné figure 6 (une version pleine page est donnée en annexe 3).

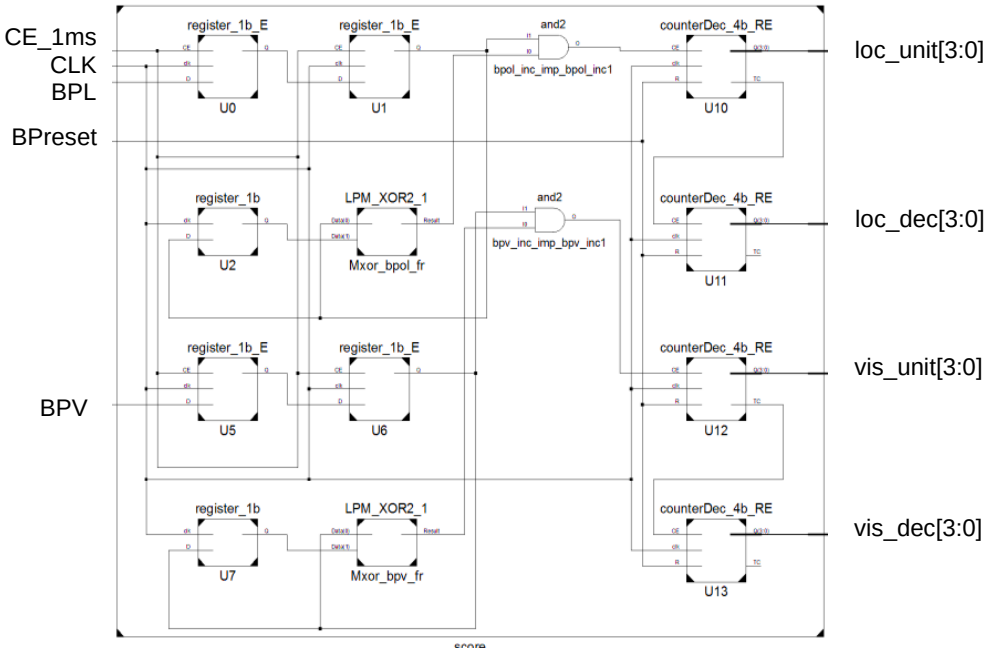


*Figure 6 : Architecture de l'entité **score***

Une version exécutable de l'entité **chronometer** (chronometer.ngc) est **fournie** sur **CPe-Campus**.

1.9 Sous-bloc **vgaDisplay**

Le sous-bloc **vgaDisplay** génère deux signaux de synchronisation **HSYNCH** et **VSYNCH** ainsi que l'affichage d'images auto-générées. Le synoptique complet de **vgaDisplay** est donné figure 7 (une version pleine page est donnée en annexe 4).

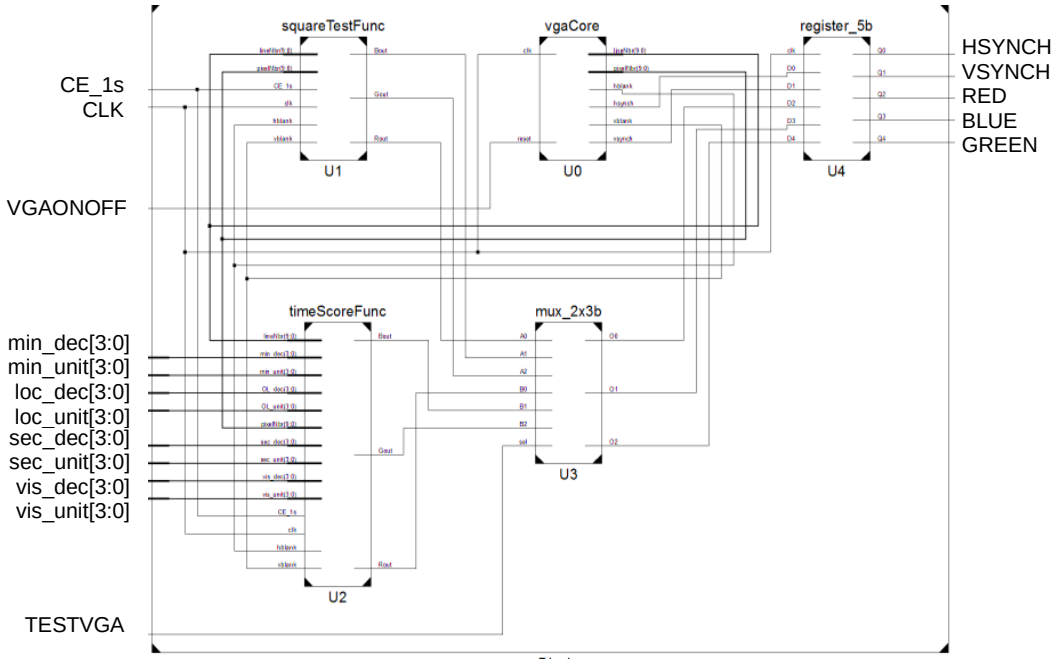
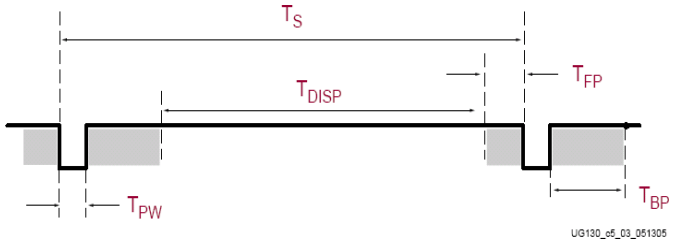


*Figure 7 : Architecture de l'entité **vgaDisplay***

Le sous bloc **vgaDisplay** comporte cinq entités :

- **vgaCore :** génère les signaux de synchronisation horizontaux et verticaux et identifie le numéro de **pixel** et le numéro de **ligne** en cours de traitement pour chacune des quatre étapes des signaux de synchronisation horizontale (ligne) et verticale (page) de la norme **VGA**,
  Ces quatre étapes sont :
  - **Active Video** (Display time),
  - **Front Porch**,
  - **Synch Pulse** (Pulse width),
  - **Back Porch**.



Symbol	Parameter
T_S	Sync pulse time
T_{DISP}	Display time
T_{PW}	Pulse width
T_{FP}	Front porch
T_{BP}	Back porch

- **squareTestFunc :** génère des images de test,
- **timeScoreFunc :** génère une image contenant le score des deux équipes ainsi que le temps écoulé,
- **mux_2x3b :** sélectionne soit l'affichage des images de test, soit le temps et les scores,
- **register_5b :** resynchronise les signaux internes afin de générer des signaux de sortie synchrones.

Les descriptions **VHDL** de **vgaDisplay**, **squareTestFunc**, **timeScoreFunc**, **mux_2x3b**, **register_5b** seront **fournies** sur **CPe-Campus**.

La description **VHDL** de **vgaCore** repose sur la modélisation de **deux compteurs** et de deux **FSM** (machines d'états). Le synoptique complet de **vgaCore** est donné figure 8 (une version pleine page est donnée en annexe 5).

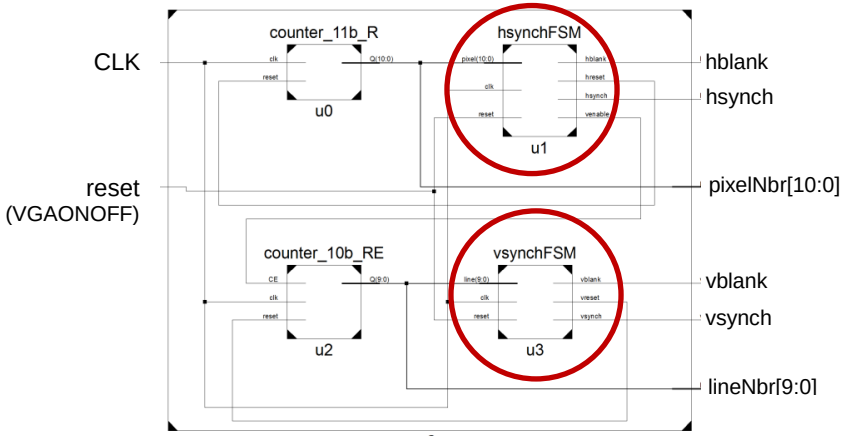


*Figure 8 : Architecture de l'entité **vgaCore***

L'entité **vgaCore** comporte quatre fonctions :

- **counter_11b_R** : compteur **11 bits** qui s'incrémente à chaque front montant de l'horloge **CLK**, il sera remis à zéro par le signal **reset synchrone**, son bus de sortie indique le **numéro du pixel** en cours de traitement, **pixelNbr**.
- **counter_10b_RE** : compteur **10 bits** qui s'incrémente à chaque front montant de l'horloge **CLK** lorsque le signal **CE synchrone** est à l'état **HAUT**, il sera remis à zéro par le signal **reset synchrone**, son bus de sortie indique le **numéro de la ligne** en cours de traitement**, lineNbr.**
- **hsynchFSM** : machine d'état qui contrôle les arrêts de comptage de **counter_11b_R** et génère les signaux **hsynch** et **hblank**.
- **vsynchFSM :** machine d'état qui contrôle les arrêts de comptage de **counter_10b_RE** et génère les signaux **vsynch** et **vblank**.

Un **état HAUT** sur l'entrée **reset** (**signal actif à l'état HAUT**) de **vgaCore**, a pour effet de démarrer le processus d'avancement des machines d'états **hsynchFSM** et **vsynchFSM**. Cette entrée est directement reliée à l'entrée **VGAONOFF** du système.

Les compteurs sont **mis à zéro** à la fin de chaque zone des signaux de synchronisation de la norme **VGA** (**Front Porch**, **Sync Pulse**, **Back Porch** et **Active Video)**. Cette réinitialisation, **synchrone et active à l'état HAUT,** est obtenue à l'aide des signaux **hreset** et **vreset** émis par chacune des **machines d'états**.

Les signaux **hsynch**, **hblank**, **vsynch** et **vblank** générés par **vgaCore** sont définis par rapport aux étapes des signaux de synchronisation de la norme **VGA** :

- **hsynch** (signal de synchronisation **horizontale**) :
  vaut '0' pendant la phase **Sync Pulse** et '1' lors des trois autres phases.
- **hblank** (signal indiquant au système qu'il est à l'état **Active Video** lorsque le faisceau balaye les pixels) :
  vaut '0' pendant la phase **Active Video** et '1' lors des trois autres phases.
- **vsynch** (signal de synchronisation **verticale**) :
  vaut '0' pendant la phase **Sync Pulse** et '1' lors des trois autres phases.
- **vblank** (signal indiquant au système qu'il est à l'état **Active Video** lorsque le faisceau balaye les lignes) :
  vaut '0' pendant la phase **Active Video** et '1' lors des trois autres phases.

Le signal **pixelNbr** (signal issu du compteur **counter_11b_R**) indique le numéro du **pixel** en cours de traitement pour la ligne balayée.
Le signal **lineNbr** (signal issu du compteur **counter_10b_RE**) indique le numéro de la **ligne** en cours de traitement pour la page balayée.

2 Phase 3 du projet – Gestion d'un écran VGA

Les entités concernées par la troisième partie du projet, nommée **chronoscore_phase3**, sont indiquées dans l'arbre figure 9 :

chronoscore *(disponible sur CPe-Campus)*
timeGenerator *(fichier timeGenerator.ngc disponible sur CPe-Campus)*
chronometer *(fichier chronometer.ngc disponible sur CPe-Campus)*
score *(fichier score.ngc disponible sur CPe-Campus)*
display *(fichier display.ngc disponible sur CPe-Campus)*
vgaDisplay *(disponible sur CPe-Campus)*
├ squareTestFunc *(disponible sur CPe-Campus)*
├ timeScoreFunc *(disponible sur CPe-Campus)*
├ mux_2x3b *(disponible sur CPe-Campus)*
├ register_5b *(disponible sur CPe-Campus)*
└ **vgaCore**
  ├ hsynchFSM *(disponible sur CPe-Campus)*
  ├ counter_11b_R
  ├ vsynchFSM *(disponible sur CPe-Campus)*
  └ counter_10b_RE

*Figure 9 : Entités du projet **chronoscore_phase3***

Le sous-bloc **vgaDisplay** comporte quatre fonctions plus le sous-bloc **vgaCore** :

- squareTestFunc
- timeScoreFunc
- mux_2x3b
- register_5b
- **vgaCore**

Le sous-bloc **vgaCore** comporte quatre fonctions :

- **counter_11b_R**
- hsynchFSM
- **counter_10b_RE**
- vsynchFSM

Lors de cette phase 3 du projet, seule l'entité **vgaCore** et les compteurs **counter_11b_R** et **counter_10b_RE** sont à réaliser.

2.1 Spécifications de la fonction **counter_11b_R**

La fonction **counter_11b_R** est un compteur 11 bits **actif sur front montant** du signal de l'horloge **clk**. Le bus de sortie **Q** du compteur indique le numéro du **pixel** en cours de traitement.
Le compteur dispose d'une entrée de remise à zéro **reset**, **synchrone** et **active à l'état HAUT**.

Les entrées de l'entité **VHDL** seront définies comme des signaux : **reset**, **clk**.
La sortie de l'entité **VHDL** sera définie comme un vecteur de dimension 11: **Q**[10 :0].
Un signal interne **Q_int**[10 :0] sera nécessaire pour décrire l'état du compteur.

Rappel : Un compteur peut être défini de manière simple comme un **additionneur**. Pour pouvoir faire des additions, il faut ajouter à la l'entité la bibliothèque **IEEE.NUMERIC_STD.ALL.**

2.2 Spécifications de la sous-fonction **counter_10b_RE**

La fonction **counter_10b_RE** est un compteur 10 bits **actif sur front montant** de l'horloge **clk**. Le bus de sortie **Q** du compteur indique le numéro de **ligne** en cours de traitement.

Le compteur dispose d'une entrée de remise à zéro **reset**, **synchrone** et **active à l'état HAUT** et d'une entrée de validation **CE** (Clock Enable), **synchrone** et **active à l'état HAUT**.

Les entrées de l'entité **VHDL** seront définies comme des signaux :

reset, **CE**, **clk**.

La sortie de l'entité **VHDL** sera définie comme un vecteur de dimension 10 : **Q**[9 :0].

Un signal interne **Q_int**[9 :0] sera nécessaire pour connaître l'état du compteur.

Rappel : Un compteur peut être défini de manière simple comme un **additionneur**. Pour pouvoir faire des additions, il faut ajouter à la l'entité la bibliothèque **IEEE.NUMERIC_STD.ALL.**

3 Réalisation du projet chronoscore_phase3

Le projet **chronoscore_phase3** met en œuvre **chronoscore** en intégrant le sous-bloc **vgaDisplay**. Le développement s'effectue en équipe.

3.1 Création du projet

Démarrer **ISE version 14.6** (64_bit) via le menu **Démarrer** de Windows et créer le projet **chronoscore_phase3** ayant les caractéristiques suivantes :

Name	**chronoscore_phase3**
Location	**(…)\ELN2\Groupe_X\Equipe_N**
Working Directory	**(…)\ELN2\Groupe_X\Equipe_N**
Top_Level source type	**HDL**

(…) : CPE_users\TPelec_3ETI (Répertoire de travail sous **Windows**).

Les répertoires **Groupe_X** et **Poste_N** devront être créés au préalable :

Groupe_X : X vaut de A à D en fonction du groupe de TP

Equipe_N : N numéro d'équipe

Rappel ! Le système crée automatiquement un répertoire associé au nom du projet. Il travaille uniquement dans ce répertoire. **Il est important que tous les fichiers sources y soient placés.**

Evaluation Development Board	None Specified
Product Category	All
Family	**Artix7**
Device	**XC7A100T**
Package	**CSG324**
Speed	**-1**
Top-Level Source Type	**HDL**
Synthesis Tool	**XST (VHDL/Verilog)**
Simulator	**ISim (VHDL/Verilog)**
Preferred Language	**VHDL**
Property Specification in Project File	Store non-default values only
Manual Compile Orders	☐
VHDL Source Analysis Standard	**VHDL-200X**
Enable Massage Filtering	☐

3.2 Réalisation des sous-fonctions **counter_11b_R** et **counter_10b_RE**

Pour chacune des sous-fonctions **:**

1. Créer l'entité* VHDL correspondante (Projet → New Source).

Définir ses signaux d'entrées* et sorties*.

(*) : Respecter les noms imposés dans les spécifications.

2. Ecrire le code VHDL de son architecture.
   Préciser dans la zone d'entête de chaque fichier source les noms des auteurs (par ordre alphabétique) :
   -- Engineer: **NOM1_NOM2** (ajouter **NOM3** si trinôme).
3. Faire la synthèse.
4. Simuler en **Behavioral**.

Pour chaque entité jugée valide (résultats corrects après simulations) :
- imprimer et **valoriser** (c'est-à-dire commenter) les documents suivants :
  le fichier source,
  la vue **Technologique**,
  les chronogrammes des simulations **Behavorial**.
- compléter le tableau(1) de caractérisation de **vgaCore** (fourni sur CPe-Campus) en précisant les informations(2) suivantes :
  le nombre de SLICES,
  le nombre de SLICES LUTS,
  le nombre de SLICES REGISTERS ou IOB FLIP-FLOP,
  le nombre d'IOBS (entrées / sorties).

(1) Préciser le groupe de TP et l'équipe dans l'entête du fichier.

(2) Les informations à rassembler dans le tableau sont disponibles dans le document **Place and Route Report** une fois la phase d'implémentation réussie. Si la page **HTML** ne se met pas à jour, on peut les trouver dans le fichier d'extension « **.par »** créé par l'outil **ISE** et disponible dans le répertoire du projet.

3.3 Etude des machines d'états **hsynchFSM** et **vsynchFSM**

L'étude de chaque machine d'états doit mettre en œuvre son compteur associé.
Pour cela, il est donc nécessaire de créer les entités **Test_hsynchFSM** (machine d'états **hsynchFSM)** et **Test_vsynchFSM** (machine d'états **vsynchFSM)**.

3.3.1 Entité **Test_hsynchFSM**

Les entrées de l'entité **VHDL** sont définies comme des signaux : **reset**, **CLK**.
Les sorties de l'entité **VHDL** sont définies comme des signaux et un vecteur de dimension 11 : **hsynch**, **hblank**, **venable**, **pixelNbr**[10 :0].
L'entité comporte également les signaux internes suivants : **clk_int**, **hreset_int**, **pixel_int**[10 :0].

L'architecture de l'entité **Test_hsynchFSM** est donnée figure 10.

1. Créer l'entité* VHDL correspondante (Projet → New Source).
2. Définir ses signaux d'entrées* et sorties*.
   (*) : Respecter les noms imposés dans les spécifications.
3. Ecrire le code VHDL de son architecture.
   Préciser dans la zone d'entête de chaque fichier source les noms des auteurs (par ordre alphabétique) :
   -- Engineer: **NOM1_NOM2** (ajouter **NOM3** si trinôme).

4. Faire la synthèse.
5. Effectuer la simulation **Behavioral**.

   Le signal **CLK**, sera défini de la manière suivante :

   - **CLK** : période 10 ns, durée à l'état HAUT 5 ns,

   La description **VHDL** de ce signal est fournie en annexe 6.
6. Mettre en évidence les différents états de la machine d'états **hsynchFSM** et vérifier les durées des différentes étapes du signal de synchronisation de la norme **VGA**.

   Le tableau 3 ci-dessous précise ces durées pour la résolution d'écran de 800x600 utilisée.

Mode	Nom	Clock	Pixel Clock MHz	Front Porch		Pulse width ou Synch Pulse		Back Porch		Active video		Line period	
		Hz	MHz	µs	pixel	µs	pixel	µs	pixel	µs	pixel	µs	pixel
VGA	800x600	56	36	0.58	21	2.00	72	3.47	125	22.39	806	28.44	1024
VGA	800x600	60	40	0.93	37	3.2	128	2.13	85	20.15	806	26.4	1056
VGA	800x600	72	50	1.06	53	2.4	120	1.22	61	16.12	806	20.8	1040

*Tableau 3 : Durée de chacune des quatre zones identifiées lors de l'affichage d'**une ligne***

A quoi correspondent les signaux hblank et hsynch ?

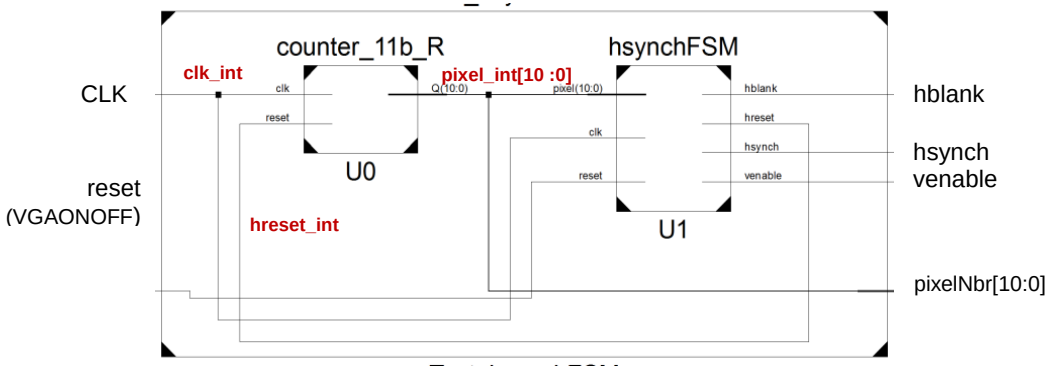


*Figure 10 : Architecture de l'entité **Test_hsynchFSM***

Imprimer et **valoriser** (c'est-à-dire commenter) les documents suivants :

- le fichier source,
- la vue **RTL** développée,
- le chronogrammes de simulations **Behavorial**.

3.3.2 Entité **Test_vsynchFSM**

Les entrées de l'entité **VHDL** sont définies comme des signaux : **CE**, **reset**, **clk**.

Les sorties de l'entité **VHDL** sont définies comme des signaux et un vecteur de dimension 10 : **vsynch**, **vblank**, **lineNbr**[9 :0].

L'entité comporte également les signaux internes suivants : **clk_int**, **vreset_int**, **line_int**[9 :0].

L'architecture de l'entité **Test_vsynchFSM** est donnée figure 11 :

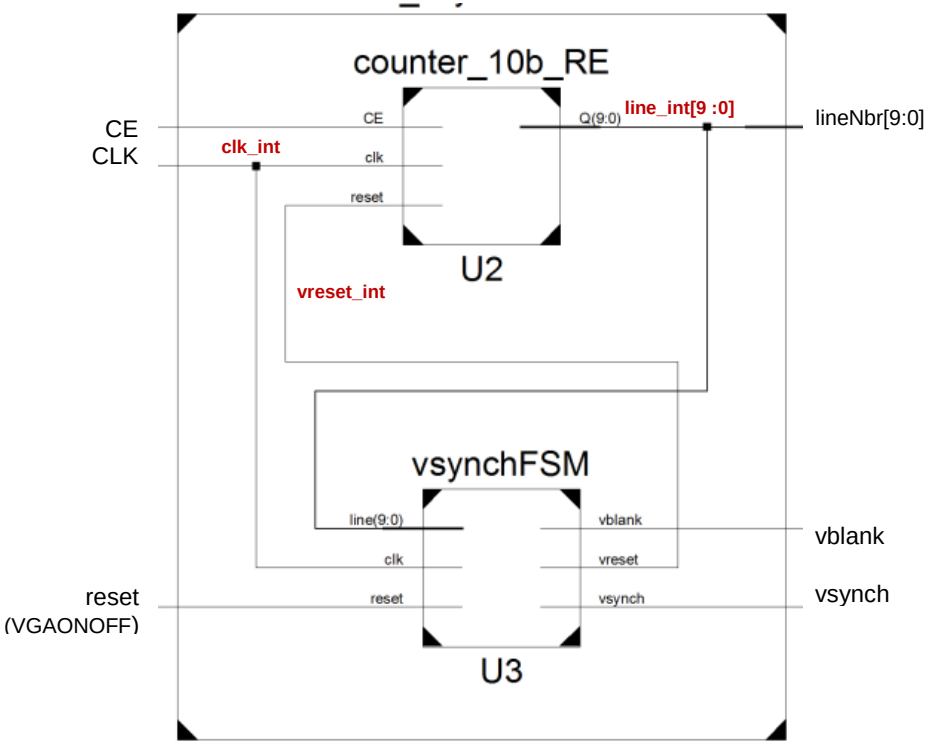


*Figure 11 : Architecture de l'entité **Test_vsynchFSM***

Le signal **CE** appliqué en entrée du compteur **counter_10b_RE** correspondant à la sortie **venable** de la machine d'état **hsynchFSM**.

1. Créer l'entité* VHDL correspondante (Projet → New Source).
2. Définir ses signaux d'entrées* et sorties*.
   (*) : Respecter les noms imposés dans les spécifications.
3. Ecrire le code VHDL de son architecture.
   Préciser dans la zone d'entête de chaque fichier source les noms des auteurs (par ordre alphabétique) :
   -- Engineer: **NOM1_NOM2** (ajouter **NOM3** si trinôme).
4. Faire la synthèse.
5. Effectuer la simulation **Behavorial**.
   Les signaux **CLK** et **CE** seront définis de la manière suivante :
   - **CLK** : période 10 ns, durée à l'état HAUT 5 ns,
   - **CE** : période 20.8 us (sous multiple de la durée réelle), durée à l'état HAUT 10 ns,

   La description **VHDL** de ces signaux est fournie en annexe 6.
   Mettre en évidence les différents états de la machine d'états **vsynchFSM** et vérifier les durées des différentes étapes du signal de synchronisation de la norme **VGA**.
   Le tableau 4 ci-dessous précise ces durées pour la résolution d'écran de 800x600 utilisée.

Mode	Nom	Clock	Line time	Front Porch		Synch Pulse		Back Porch		Active video		Frame period	
		Hz	µs	µs	ligne	µs	ligne	µs	ligne	ms	ligne	ms	ligne
VGA	800x600	56	28.44		- 1	56	1	568	20	17.177	604	17.775	625
VGA	800x600	60	26.4		- 1	106	4	554	21	15.945	604	16.579	628
VGA	800x600	72	20.8	728	35	125	6	436	21	12.563	604	13.853	666

*Tableau 4 : Durée de chacune des quatre zones identifiées lors de l'affichage d'**une page***

A quoi correspondent les signaux vblank et vsynch ?

Imprimer et **valoriser** (c'est-à-dire commenter) les documents suivants :

le fichier source,

la vue **RTL** développée,

le chronogrammes de simulations **Behavorial**.

3.4 Réalisation du sous-bloc **vgaCore**

1. Créer l'entité* VHDL correspondante (Projet → New Source).
2. Définir ses signaux d'entrées* et sorties*.
   (*) : Respecter les noms imposés dans les spécifications.
3. Ecrire le code VHDL de son architecture à partir de ses sous-fonctions.
   Préciser dans la zone d'entête de chaque fichier source les noms des auteurs (par ordre alphabétique) :
   -- Engineer: **NOM1_NOM2** (ajouter **NOM3** si trinôme).
   L'entité **vgaCore** comporte les signaux internes suivants : **clk_int**, **reset_int**, **venable_int**, **hreset_int**, **pixel_int**[10 :0], **vreset_int**, **line_int**[9 :0].
   Ces signaux sont identifiés sur le schéma fourni en fin de document.
4. Faire la synthèse.

Une fois l'entité jugée valide (vue **RTL** correcte après synthèse) :

- imprimer et **valoriser** (c'est-à-dire commenter) les documents suivants :

  le fichier source,

  la vue **RTL** développée,

- compléter le tableau* de caractérisation de **vgaCore** (fourni(1) sur CPe-Campus) en précisant les informations(2) suivantes :

  le nombre de SLICES,

  le nombre de SLICES LUTS,

  le nombre de SLICES REGISTERS ou IOB FLIP-FLOP,

  le nombre d'IOBS (entrées / sorties).

(1) Préciser le groupe de TP et l'équipe dans l'entête du fichier.

(2) Les informations à rassembler dans le tableau sont disponibles dans le document **Place and Route Report** une fois la phase d'implémentation réussie. Si la page **HTML** ne se met pas à jour, on peut les trouver dans le fichier d'extension « **.par »** créé par l'outil **ISE** et disponible dans le répertoire du projet.

Vérifier la cohérence de l'ensemble des résultats. Commenter.

3.4 Réalisation du sous bloc **vgaDisplay**

A partir du cours ELN2 sur **CPe-Campus**, télécharger les fichiers relatifs au sous-bloc **vgaDisplay** :

- vgaDisplay.vhd,

- squareTestFunc.vhd,

- timeScoreFunc.vhd et dataROM.vhd,
- mux_2x3b.vhd,
- register_5b.vhd.

Placer ces fichiers dans le répertoire du projet **chronoscore_phase3**.
Ajouter les fichiers au projet **chronoscore_phase3** (Projet → Add Source).
Faire la synthèse du sous-bloc **vgaDisplay** et imprimer la vue **RTL** développée.

3.5 Finalisation du système **chronoscore_phase3**

A partir du cours ELN2 sur **CPe-Campus**, télécharger les fichiers relatifs à **chronoscore_phase3** :

- chronoscore.vhd,
- timeGenerator.ngc,
- display.ngc,
- chronometer.ngc,
- score.ngc.
- chronoscore.ucf,

Placer ces fichiers dans le répertoire du projet **chronoscore_phase3**.
Ajouter les fichiers au projet **chronoscore_phase3** (Projet → Add Source).

Pour le fichier **chronoscore.ucf** penser à mettre **chronoscore_phase3.vhd** en « **top-module** » avant.

Exécuter l'ensemble du flot de conception pour générer le fichier binaire qui servira à configurer les LUTs, bascules et interconnexions du **FPGA**.
Implanter **chronoscore** dans le **FPGA** à l'aide de l'outil **iMPACT**.
Tester la réalisation à l'aide de la carte de développement et de votre écran **VGA**.

5 En fin de séance

Dépôt de fichiers :

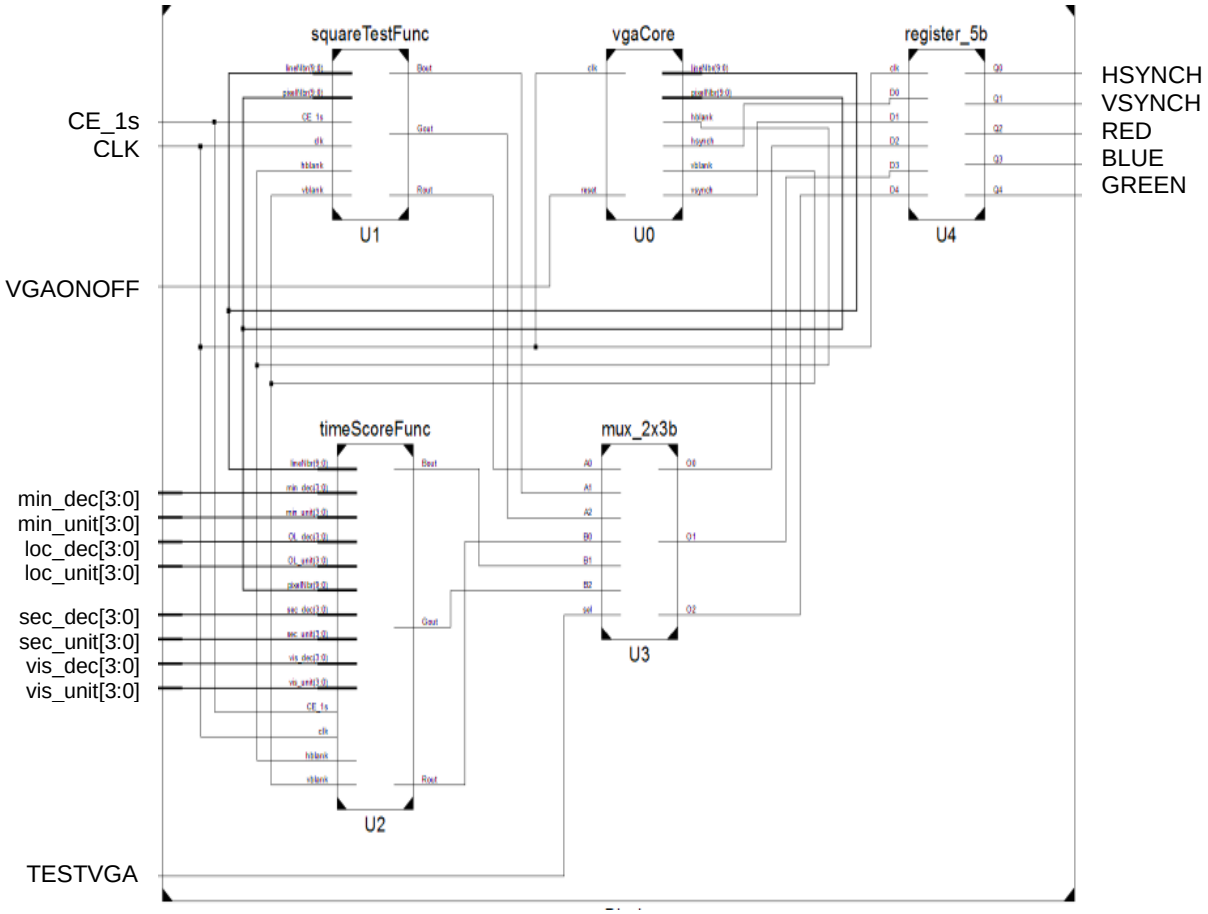
- Faire un répertoire **« .zip »** des fichiers sources créés et du tableau Excel complété.
  Les répertoires au format tar ou 7z ne seront pas acceptés.
  Le nom du répertoire devra être formaté de la manière suivante ;
  ELN2-X-NOM1-NOM2-… :
  X, groupe de TP
  NOM1, nom étudiant(e) 1
  NOM2, nom étudiant(e) 2
  etc …
  } Mettre les noms des membres de l'équipe par **ordre alphabétique**
- Le dépôt du répertoire devra être effectué sur CPe-Campus par l'étudiant(e) n°1 dans l'ordre alphabétique.

Documents papier à rendre :

- Pour chaque entité créée, rendre :
  Impression du code source,
  Impression commentée de la vue **RTL** ou **Technologique** selon le cas,
  Impression valorisée de la simulation **Behavioral**,
- Tableau de caractérisation du sous-blocs **vgaCore**.

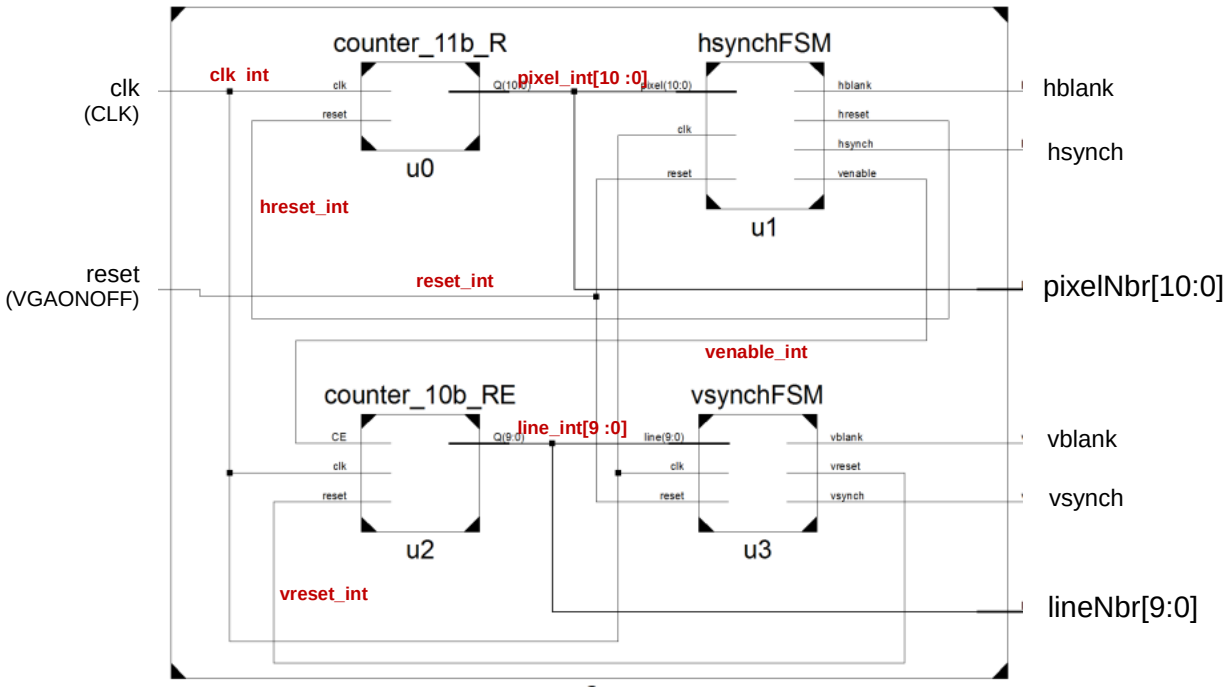
Annexe 1

Architecture du sous-bloc **vgaDisplay**



Annexe 2

Architecture de **vgaCore**



L'entité **vgaCore** comporte les signaux internes suivants :

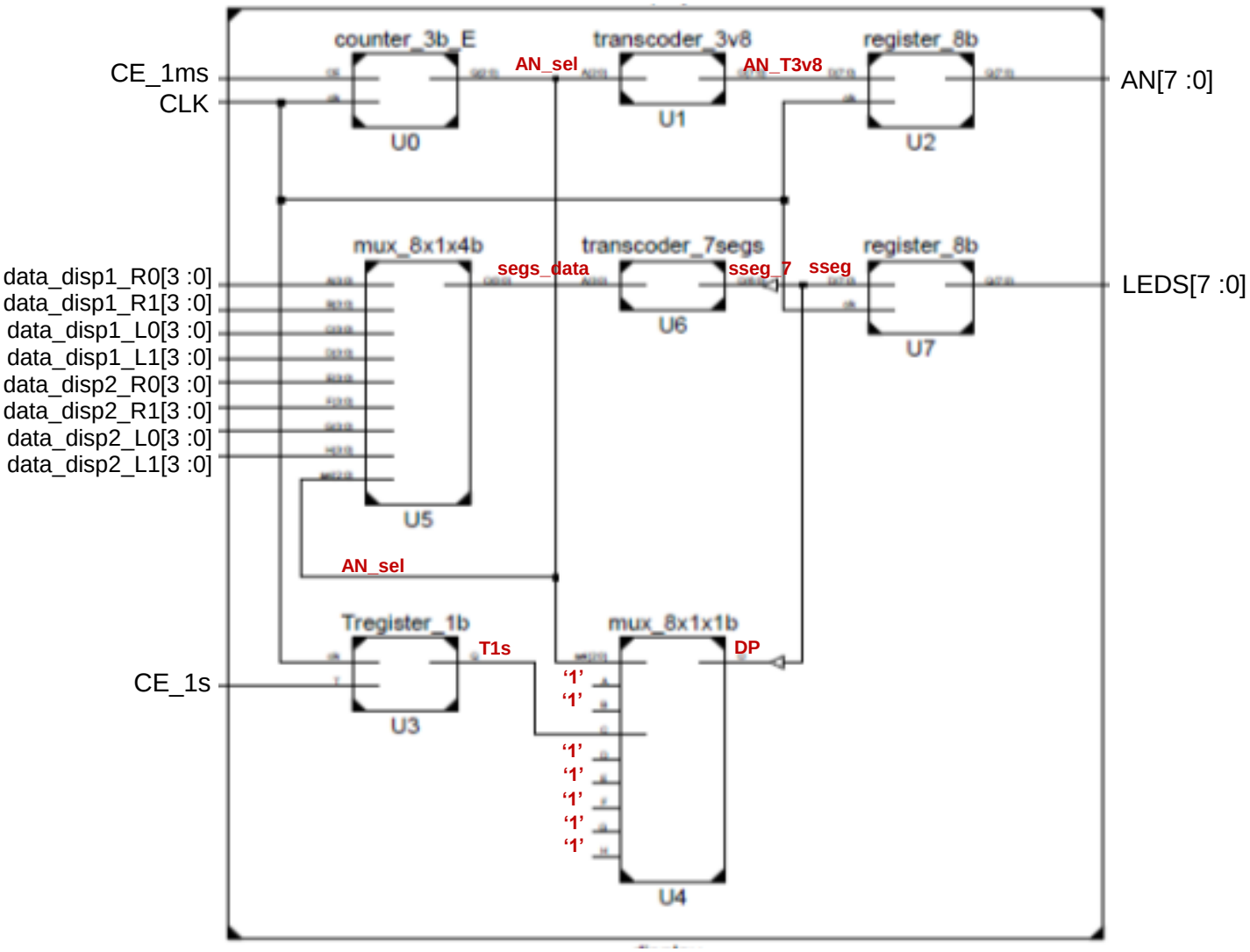
clk_int, **reset_int**,

hreset_int, **venable_int**, **vreset_int**,

pixel_int[10 :0], **line_int[9 :0]**.

Annexe 3

Architecture de l'entité **display**



L'entité **display** comporte des signaux internes suivants :

AN_sel, **AN_T3v8**, **T1s**, **DP**, **segs_data**, **sseg_7**, **sseg**.

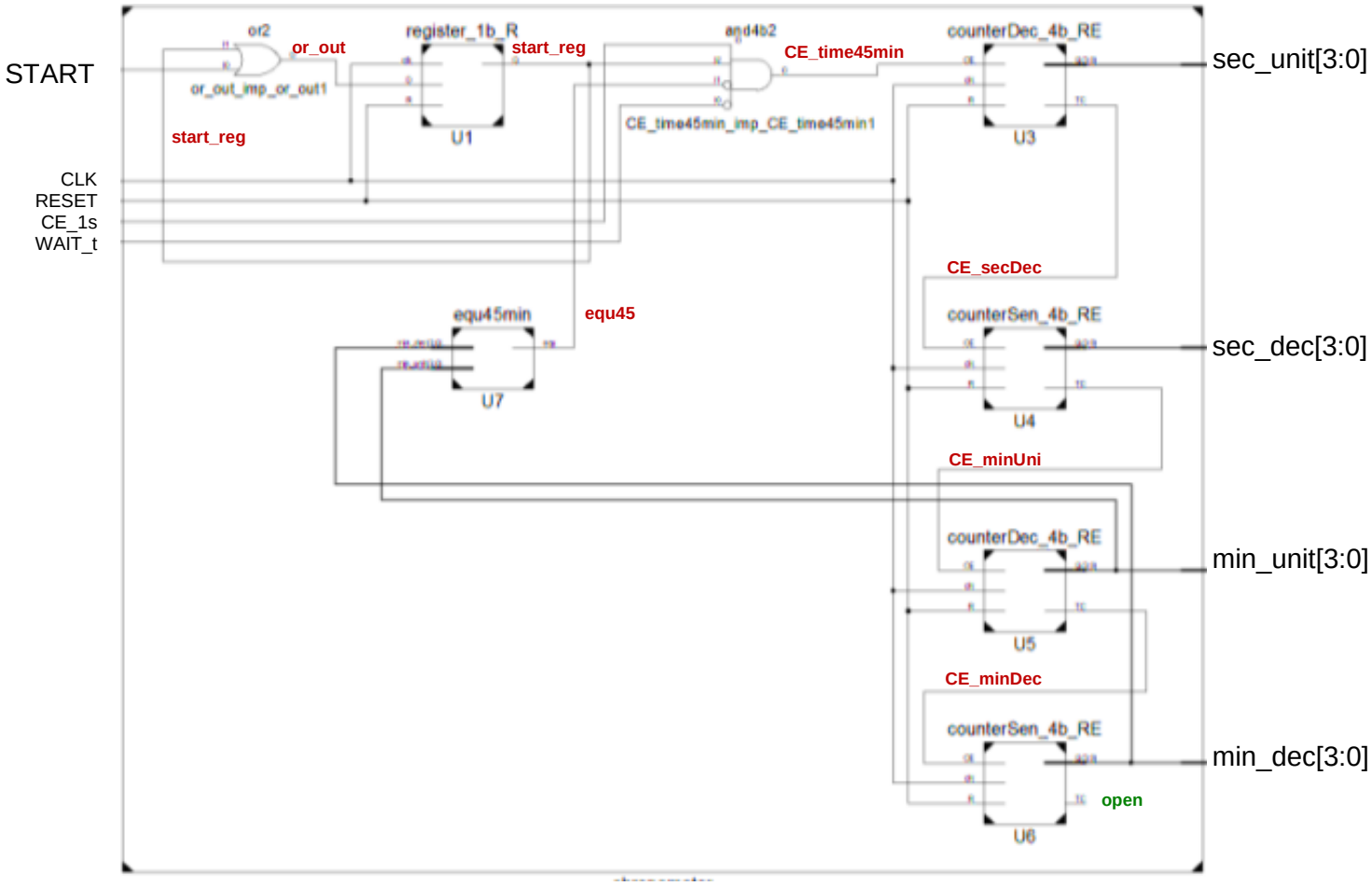
Le signal **sseg** est obtenu par concaténation de **DP** et **sseg_7** :

sseg(7 downto 0) <= DP & sseg_7(6 downto 0);

Les entrées non utilisées de l'entité mux_8x1x1b (**A**, **B**, **D, E, F, G, H**) sont positionnées à **'1'**.

Annexe 4

Architecture du sous-bloc **chronometer**



L'entité **chronometer** comporte les signaux internes suivants :

start_reg, **or_out**, **equ45**, **CE_time45min**,

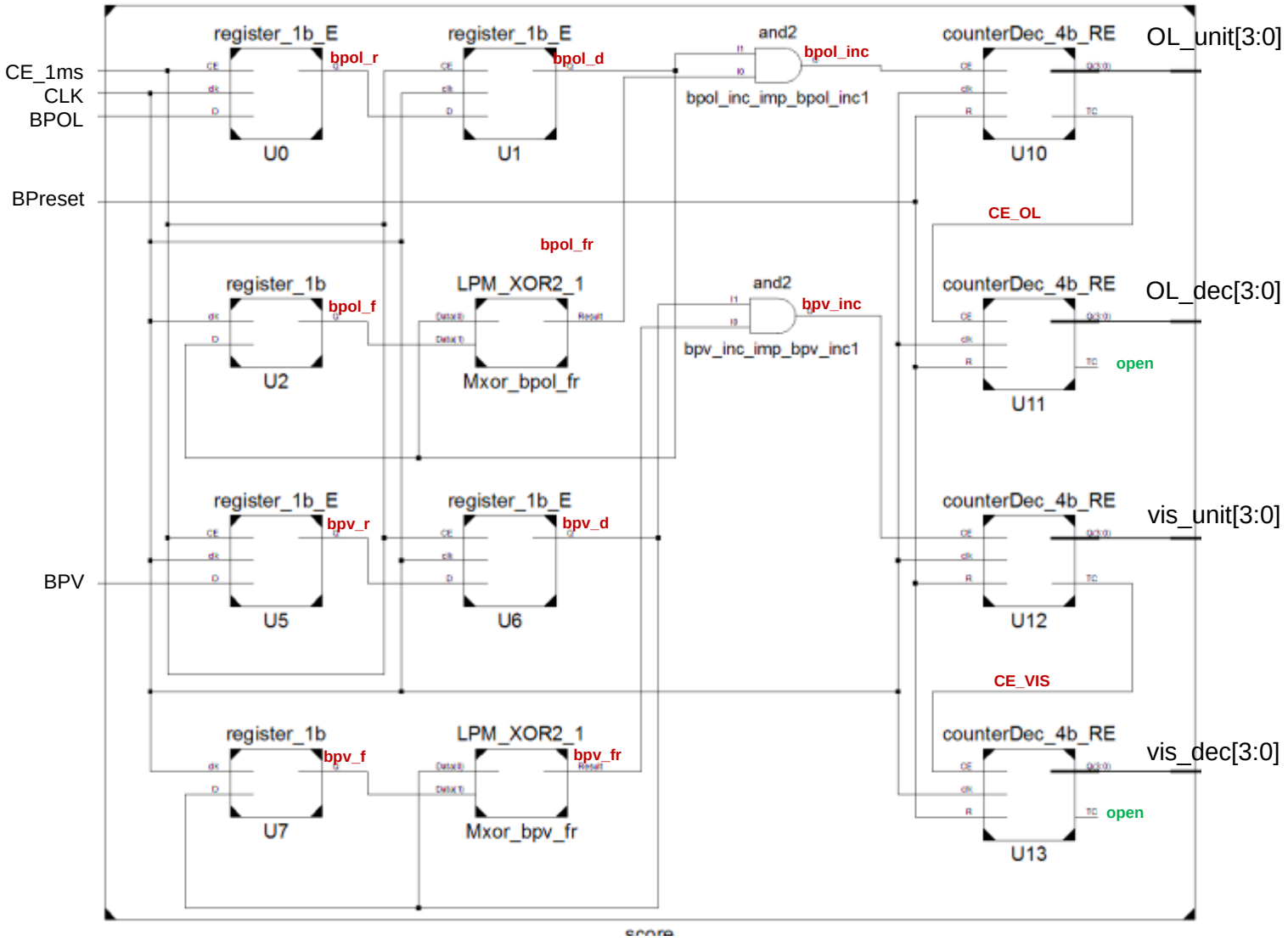
CE_secDec, **CE_minUni**, **CE_minDec**,

min_unit_int, **min_dec_int**.

La sortie **TC** non utilisée de l'entité **counterSen_4b_RE** des minutes (U6) est positionnée à **'open'**.

Annexe 5

Architecture du sous-bloc **score**



L'entité **score** comporte les signaux internes suivants :

bpl_r, **bpl_d**, **bpl_f**, **bpl_fr**, **bpl_inc**, **CE_LOC**,

bpv_r, **bpv_d**, **bpv_f**, **bpv_fr**, **bpv_inc**, **CE_VIS**.

Les sorties non utilisées des entités **counterDec_4b_RE** (U11 et U13) sont positionnées à **'open'**.

Annexe 6

Code **VHDL** des signaux **clk**, **CLK**, **CE** à utiliser pour les simulations

-- Clock period definitions (à placer dans la zone de déclarations du fichier _tb.vhd)

```
constant clk_period : time := 10 ns;
constant CLK_period : time := 10 ns;
constant CE_period : time := 20.8 us;
```

-- Clock process definitions (à placer entre BEGIN et END)

```
clk_process :process                    – ou CLK_process : process
begin
    clk <= '0';                         -- ou CLK <= ‘0’;
    wait for clk_period/2;
    clk<= '1';                          -- ou CLK <= ‘1’;
    wait for clk_period/2;
end process;

CE_process :process
begin
    CE <= '0';
    wait for 20790 ns;
    CE <= '1';
    wait for 10 ns;
end process;
```