**ELN 2 - TP**

PROJET CHRONOSCORE

SELLAMI Nadim, SAYAQUE Benjamin, CORNATON Maxime - Groupe C - Equipe 6

[**INTRODUCTION**](#_st4hot8ulb31) **2**

[**OBJECTIFS**](#_pzsyedqsyus9) **2**

[**PRÉSENTATION DE LA CARTE**](#_2rpaehd6wyw3) **3**

[La carte](#_wk43cge7erop) 3

[Bus de configuration/Alimentation](#_ljv33glvd2l6) 3

[Boutons utilisés dans le projet](#_gui1ok89le5j) 4

[**ARCHITECTURE**](#_wmjnv1jrpemd) **5**

[Fonctions spécifiques](#_a2ot55r1ofot) 5

[Schéma général (modules instanciés dans Chronoscore)](#_nozk5wqc6ynz) 5

[Schéma Display](#_4rmba3dv24p5) 6

[Fonction transcoder\_3v8:](#_6cgxib48lf76) 7

[Fonction transcoder\_7segs:](#_p6qanut6a7wa) 10

[Fonction mux\_8x1x1b:](#_tyy1h87olbq4) 13

[Fonction mux\_8x1x4b:](#_4zm9puno5wnk) 15

[Fonction register\_8b:](#_kaiexqy4rih7) 16

[Fonction Tregister\_1b:](#_412yug4ozm84) 19

[Fonction counter\_3b\_E:](#_5wzy4g33iuz9) 20

[Schéma Score](#_nkpybt199g41) 22

[Fonction register\_1b:](#_8v30bvjuevil) 23

[Fonction register\_1b\_E:](#_iv075ohshpvp) 24

[Fonction counterDec\_4b\_RE:](#_bliybdnnak5u) 25

[Schéma Chronometer](#_wlhayhl9sko9) 26

[Fonction equ45min:](#_r1hys3gilx9a) 27

[Fonction register\_1b\_R:](#_vaplp9l7i6nh) 28

[Fonction counterSen\_4b\_RE:](#_j06ofzmavypp) 29

[Fonction counterDec\_4b\_RE:](#_kay0pd2ltgsv) 30

[**MÉTHODE D'IMPLÉMENTATION / TEST**](#_yhv8zhriuv83) **31**

[**CONCLUSION**](#_wtuyjal8dv21) **32**

[Retour d’expérience/Opinion sur le projet](#_j3x9xpczmm5q) 32

[Discussion sur de potentiels problèmes rencontrés / solutionnés ou pas](#_vws3kpx9bsor) 32

[Proposition d’ajout ou modification au projet pour améliorer chronoscore](#_2njsgobglv5g) 32

[**ANNEXES**](#_d2lom41ciisl) **33**

[Codes VHDL:](#_czg23zlkylh3) 33

# INTRODUCTION

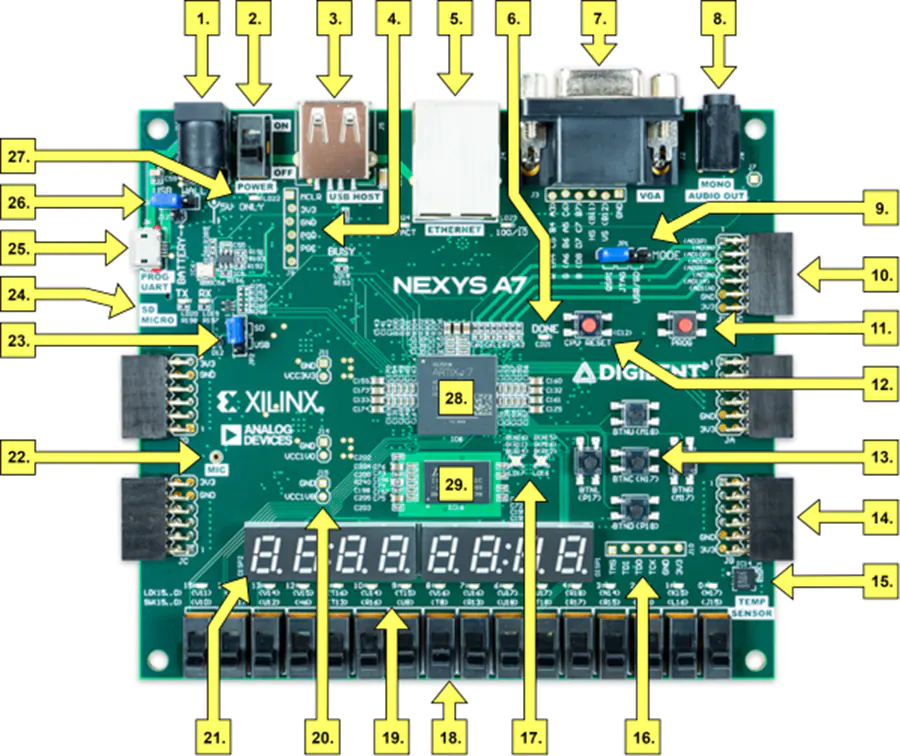
Durant 3 séances de 4h de TP, nous nous sommes penchés sur un projet d’affichage de la durée et du score d’un match de football à l’aide d’un FPGA.

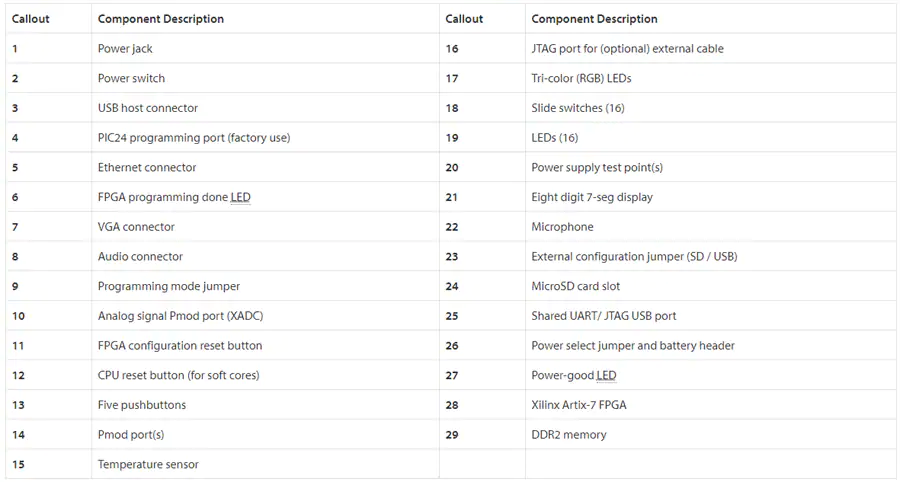
# OBJECTIFS

* Mettre en œuvre un système d’affichage multiplexé
* Prendre en main la carte NEXYs A7 de Digilent et le FPGA : Artix 7 XC7A100T CSG324-1
* Consolider l’utilisation de l’outil ISE (Xilinx)
* Consolider l’utilisation d’instructions concurrentes VHDL
* S’approprier l’utilisation d’instructions séquentielles VHDL
* Vérifier le fonctionnement du projet à partir de chronogrammes

# PRÉSENTATION DE LA CARTE

## La carte

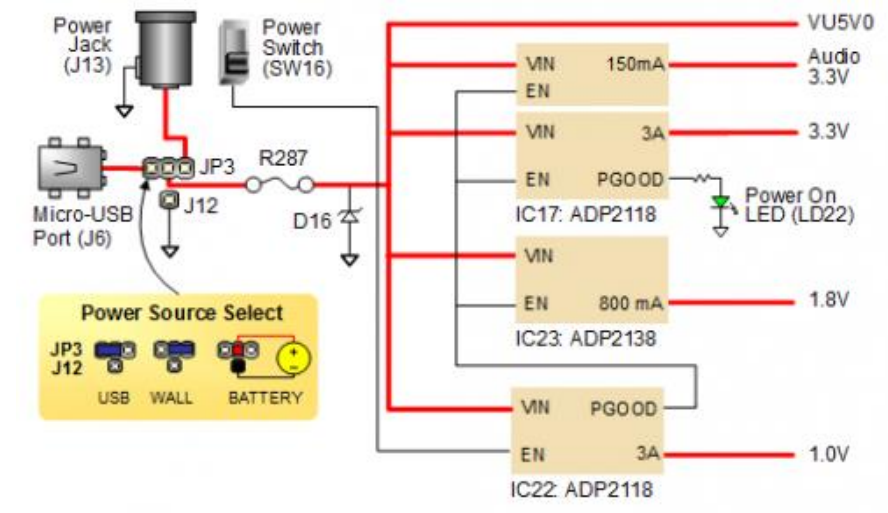
La carte utilisée pour ce projet est la NEXYS A7 de la société Digilent. Cette carte est très utile au vu de ses nombreuses caractéristiques: LEDs, Switches, tri-color LEDs, sortie VGA 12 bits, capteur de température, accéléromètre, port microSD, 2 afficheurs 7 segments de 4 digits etc… En réalité, nous nous servirons évidemment pas de toutes ses fonctionnalités mais le fait que cette carte contient déjà un grand nombre d’éléments nécessaires à notre projet font d’elle une carte simple d’utilisation.



## Bus de configuration/Alimentation

La carte Nexys A7 peut être alimentée par le port Digilent USB-JTAG (J6) ou par une alimentation externe.

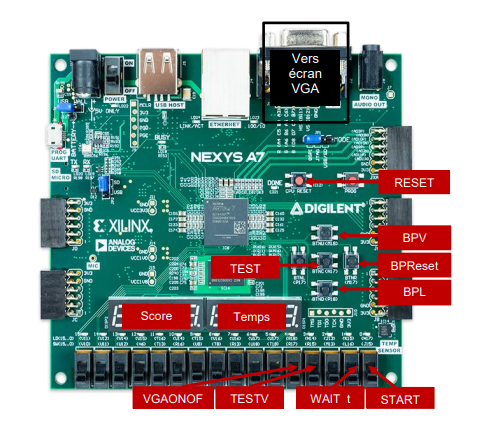
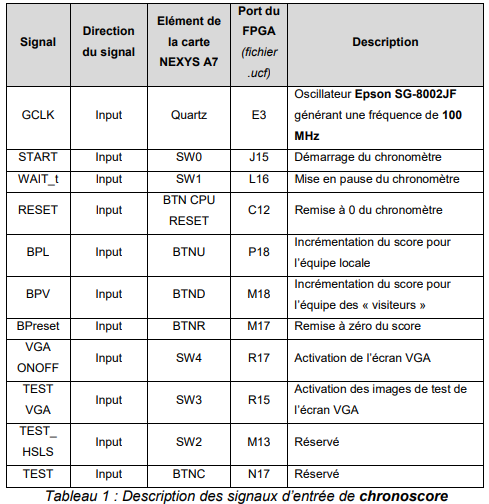
Le cavalier JP3 (près de la prise d'alimentation) détermine quelle source est utilisée.

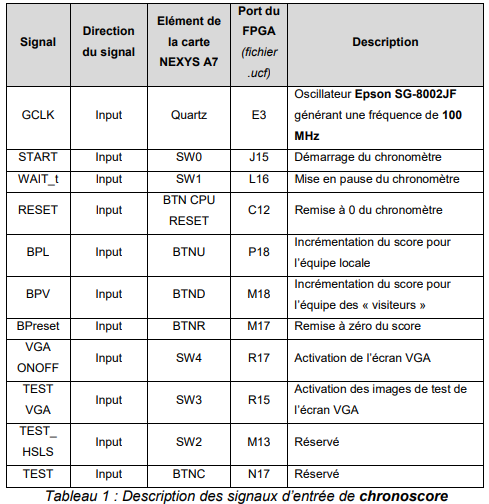
Toutes les alimentations Nexys A7 peuvent être allumées et éteintes par un seul interrupteur d'alimentation de niveau logique (SW16). Une LED (LD22), pilotée par la sortie "power good" (PGOOD) de l'alimentation ADP2118, indique que les alimentations sont allumées et fonctionnent normalement. 

Une alimentation externe peut être utilisée en la branchant à la prise d'alimentation (J13) et en réglant le cavalier JP3 sur "WALL".

Nous utiliserons seulement le port Micro-USB (J6) pour alimenter notre carte.

## Boutons utilisés dans le projet





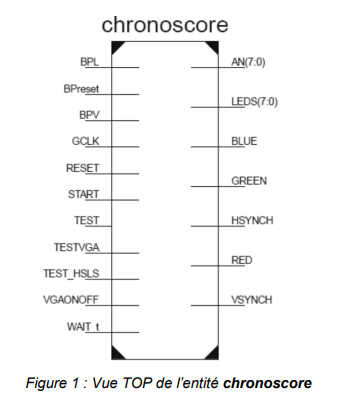
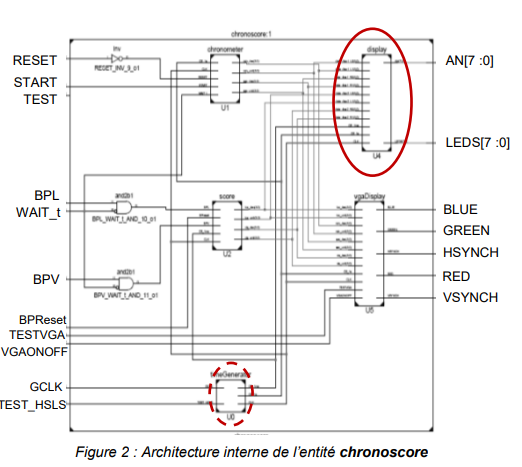
# ARCHITECTURE

## Fonctions spécifiques

On veut pouvoir compter les scores de deux équipes et les afficher, avoir un chronomètre affichant secondes et minutes et s’arrêtant au bout de 45 minutes (représentant la mi-temps d’un match de foot) et pouvoir tout réinitialiser toutes ces valeurs avec un bouton reset.

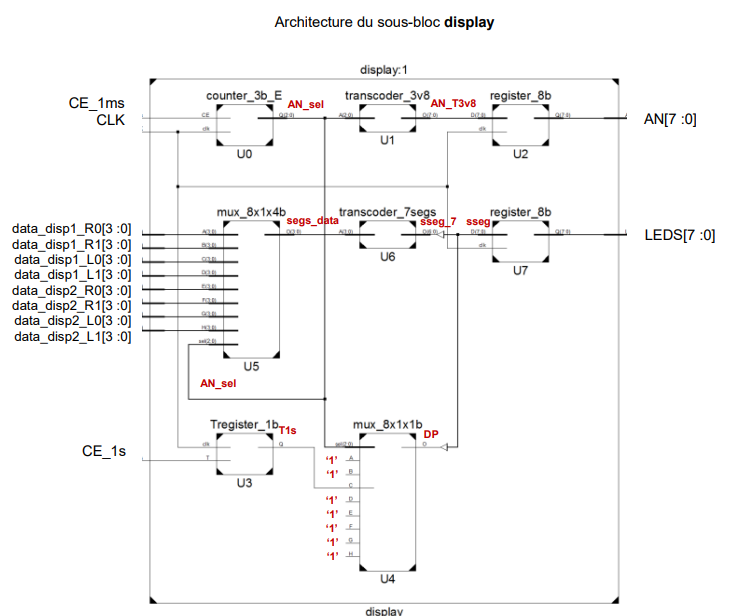
## Schéma général (modules instanciés dans Chronoscore)

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

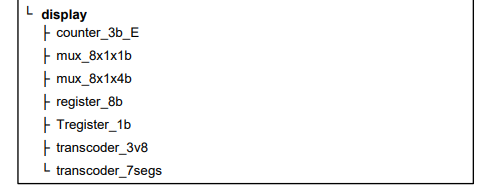


## Schéma Display

Le sous-bloc display gère les données à afficher sur les 8 afficheurs 7- segments de la carte NEXYX A7. Les quatre afficheurs de droite (disp1 : afficheurs 0 à 3) doivent indiquer le temps écoulé en minutes et secondes. 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.



Le sous-bloc display comporte sept fonctions :

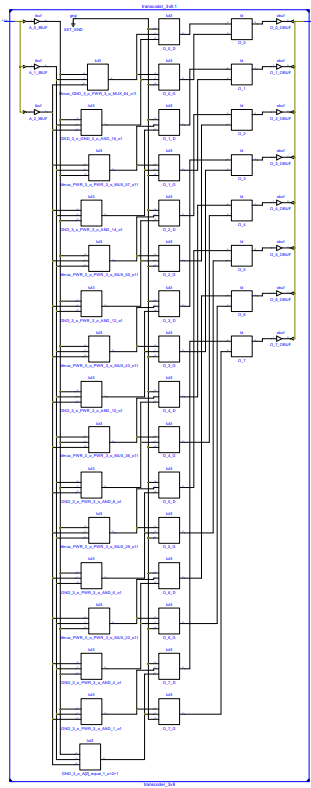
* counter\_3b\_E 
* mux\_8x1x4b
* mux\_8x1x1b
* register\_8b
* Tregister\_1b
* transcoder\_3v8
* transcoder\_7segs

Il y a 4 fonctions de type combinatoire et 3 fonctions de type séquentiel.

### Fonction transcoder\_3v8:

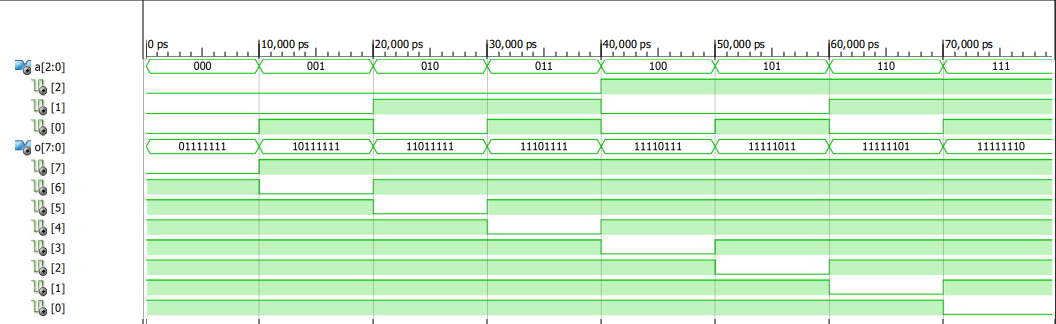
La fonction transcoder\_3v8 est en réalité un décodeur qui met à l’état BAS le numéro du signal de sortie, correspondant à l’équivalent décimal du code binaire sur trois bits appliqué sur ses entrées. L’entrée de l’entité VHDL est définie comme un vecteur A de dimension 3. La sortie de l’entité VHDL est définie comme un vecteur O de dimension 8.

Vue technologique du composant:

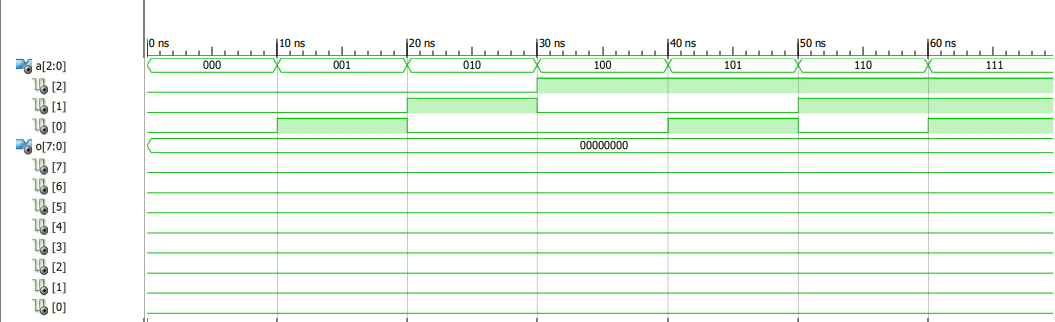


Étude comportementale du composant :

Simulation behavorial :

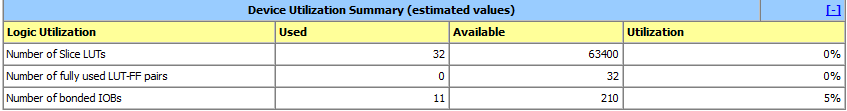


Simulation Post-Route:



Les simulations nous montrent bien que le signal de sortie met à l'état bas le numéro sélectionné par l’entrée (exemple 010 qui représente le 2 en binaire met le 3ème numéro (car le 0 est compté ici) en état bas)

Utilisation du composant:

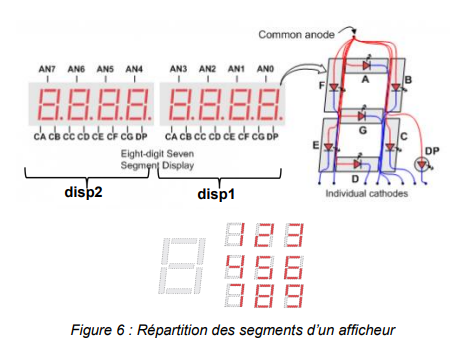


Conclusion sur le composant :

Le transcoder\_3v8 reçoit l’information donnée par le counter\_3b\_E sur le numéro désigné pour pour les 8 emplacements du chronoscore et s’occupe de mettre en sortie à l'état bas le numéro désigné sur une série de 8 bit, tous à l'état haut. Cette sortie sera désignée en entrée du register\_8b.

### Fonction transcoder\_7segs:

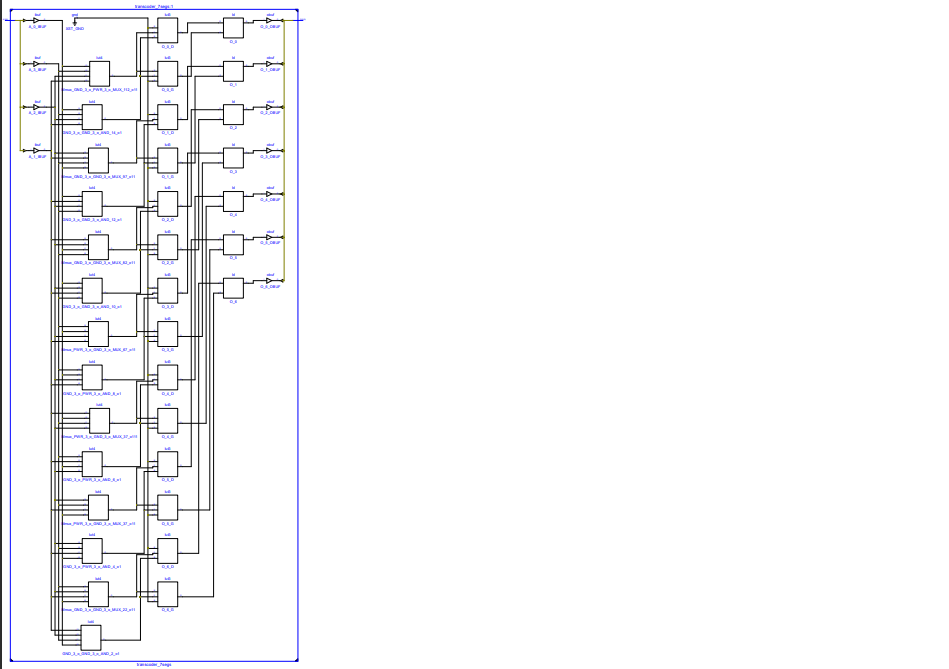
La fonction transcoder\_7segs est un transcodeur qui associe à un code binaire sur 4 bits, le code 7-segments correspondant à la valeur hexadécimale de l’entrée. La figure 6 ci-dessous précise le nom des segments en fonction de leur répartition sur l’afficheur ainsi que la répartition des segments éclairés pour les chiffres de 1 à 9.

Les segments éclairés sont pilotés par un signal à l’état BAS, les segments éteints par un signal à l’état HAUT. 

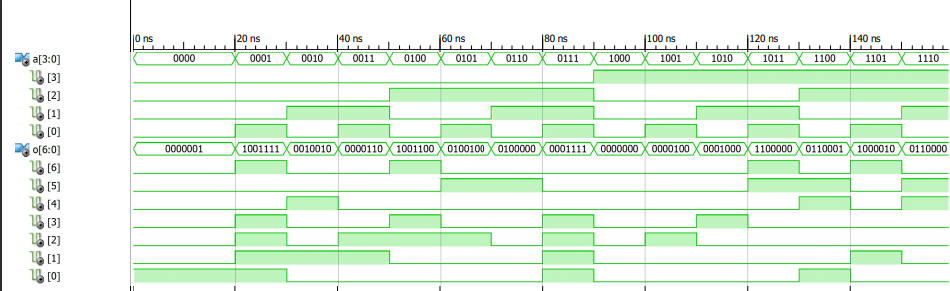
L’entrée de l’entité VHDL est définie comme un vecteur A de dimension 4. La sortie de l’entité VHDL est définie comme un vecteur O de dimension 7.

Le segment « a » correspond à O[0] et le segment « g » à O[6].

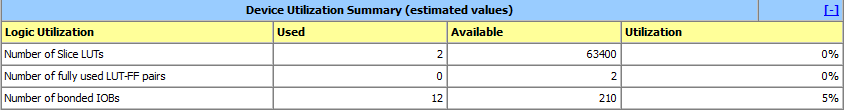
Vue Technologique du composant:



Étude comportementale du composant : (behavorial)



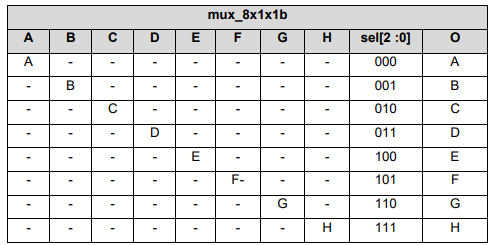
La simulation montre que les lettres concernées pour les nombres désignés en entrée sont à l'état bas lorsqu’ils sont demandés. Par exemple, pour le chiffre 1 (001 en binaire, le premier chiffre de la combinaison désigne le point), les deux barres concernant ce chiffre (soit b=O[1] et c=O[2]) sont à l’état bas. Rappel : O[7] désigne le point

Utilisation du composant:

Conclusion sur le composant :

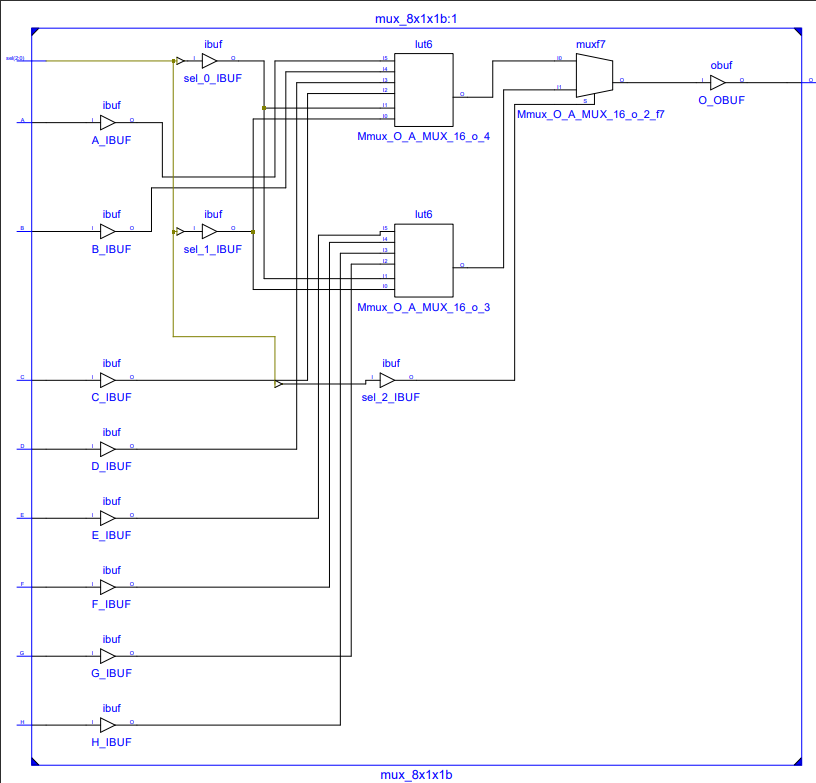
Le transcodeur\_7segs reçoit l’information transmise par le multiplexeur sur le chiffre à afficher sur chacune des anodes et s’occupe de faire la conversion entre ce même nombre et les numéros des leds concernées afin d’afficher ce chiffre. Il renvoie alors un segment de 8 bit ou les numéros des leds concernées sont à l'état bas.

### Fonction mux\_8x1x1b:

La fonction mux\_8x1x1b est un multiplexeur de 8 signaux d’entrée de 1 bit vers 1 signal de sortie de 1 bit. Il dispose de trois signaux de sélection.

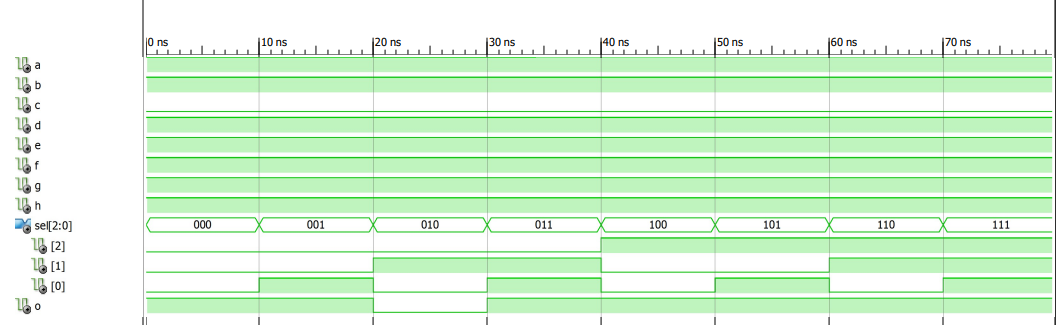
Les entrées de l’entité VHDL sont définies comme des signaux A, B, C, D, E, F, G, H et comme un vecteur sel de dimension 3. La sortie de l’entité VHDL est définie comme un signal : O.

Vue technologique du composant :

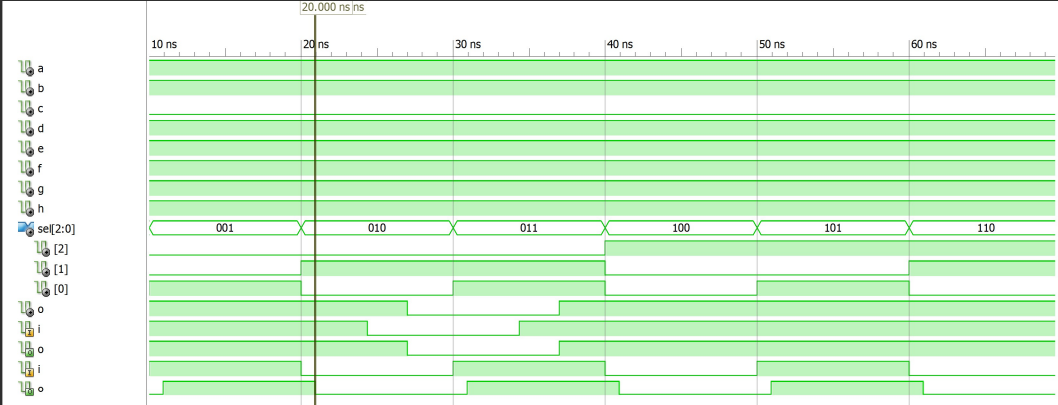


Étude comportementale du composant :

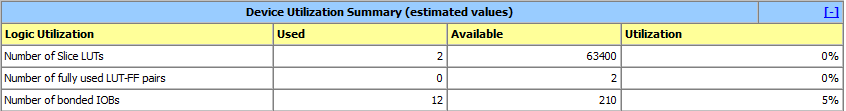
Simulation behavorial :



Simulation post-route :



Utilisation du composant:

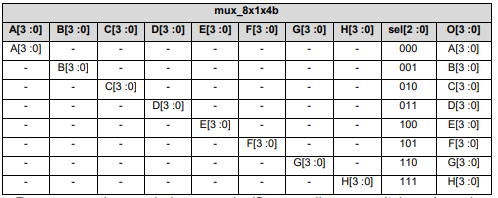


Conclusion sur le composant :

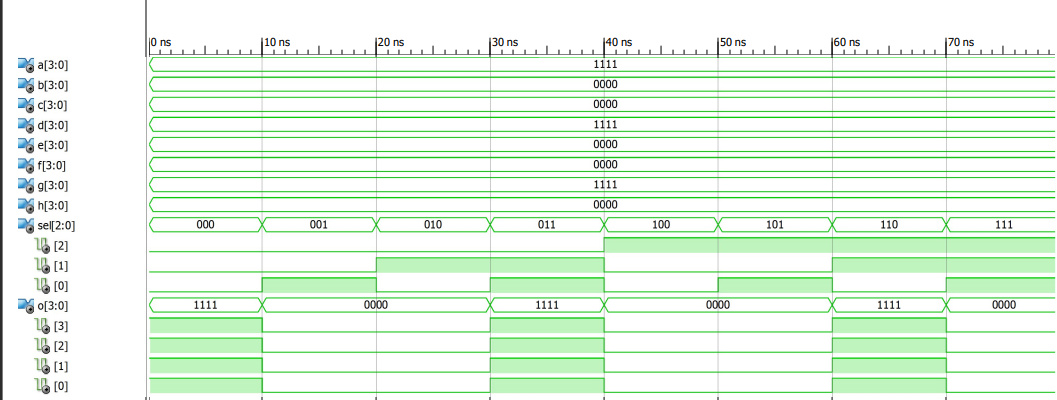
Le multiplexeur permet en fonction de la valeur attribuée sur l'entrée de sélection, le multiplexeur renverra la valeur attribuée à l’entrée correspondante, soit 0, soit 1 (données par les entrée A à H qui seront sélectionnées). On obtient alors les deux points séparants les minutes des secondes sur le compteur.

### Fonction mux\_8x1x4b:

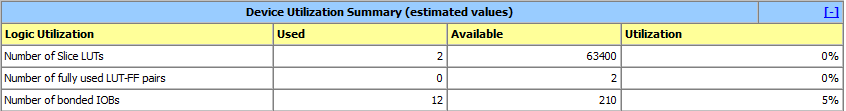
La fonction mux\_8x1x4b est un multiplexeur de 8 signaux d’entrée de 4 bits vers 1 signal de sortie de 4 bits. Il dispose de trois signaux de sélection.

Les entrées de l’entité VHDL seront définies comme des vecteurs A, B, C, D, E, F, G, H de dimension 4 et comme un vecteur sel de dimension 3.. La sortie de l’entité VHDL sera définie comme un vecteur O de dimension 4.

Étude comportementale du composant :



Utilisation du composant:

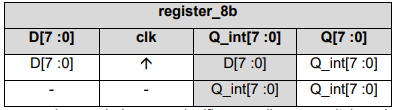


Conclusion sur le composant :

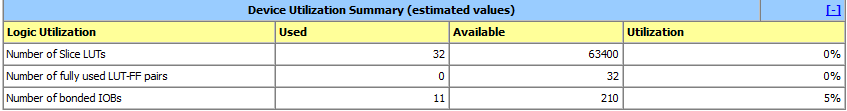
Le mux\_8x1x4b reçoit en entrées des vecteurs de A à H de dimension 4 qui correspondent aux valeurs en binaire (de 0 a 9) de chacun des nombres contenus dans les différentes places proposés dans le tableau d’affichage. En effet, le vecteur A correspond a l’entré data\_disp\_R0 soit le premier nombre a droite du tableau d’affichage (dizaine de l’equipe à domicile). Sachant que ce numéro change de manière perpétuelle, on introduit le vecteur selection (sel[2:0]) qui indiquera alors la sur quelle rotation nous somme (de A a H il y a 8=2 puissance 3 valeur possible de rotation, d’où un vecteur de dimension 3).

### Fonction register\_8b:

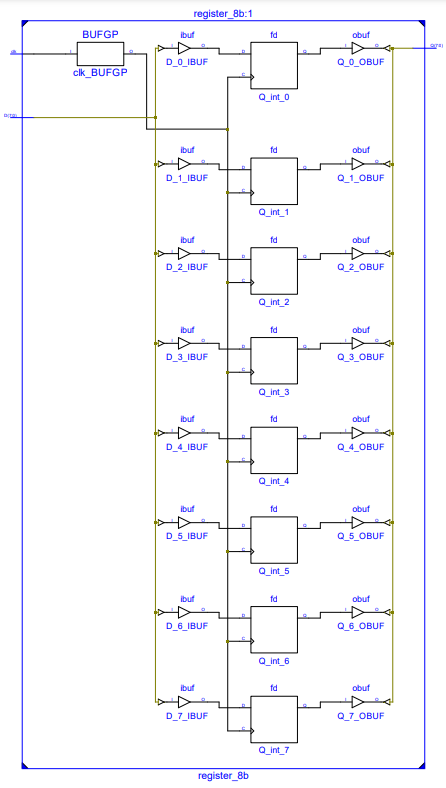
La fonction register\_8b est un registre synchrone constitué de 8 bascules D actives sur front montant du signal d'horloge.

Les entrées de l’entité VHDL sont définies comme un signal clk et comme un vecteur D de dimension 8. La sortie de l’entité VHDL est définie comme un vecteur Q de dimension 8. 

Cette table fait apparaître l’état du signal interne Q\_int[7 :0] nécessaire pour la description VHDL de l'état du registre.

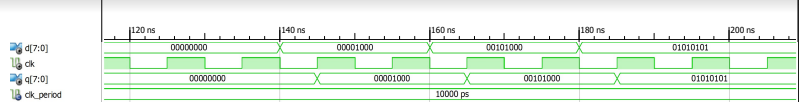


Vue technologique du composant :

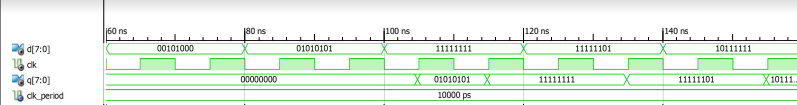


Réponse comportementale du composant :

Behavorial :



Post-Route:



Les simulations comportementale du composant nous montre que lors d’une consigne d’un affichage est en entrée, le bloc attend le prochain front montant de la clock pour pouvoir afficher l’entrée en sortie, si le front montant apparaît avant la prochaine consigne, la sortie reste la même (exemple entre 100 et 120ns).

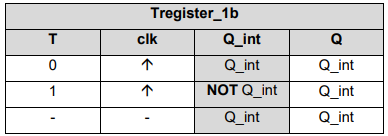
Conclusion sur le composant :

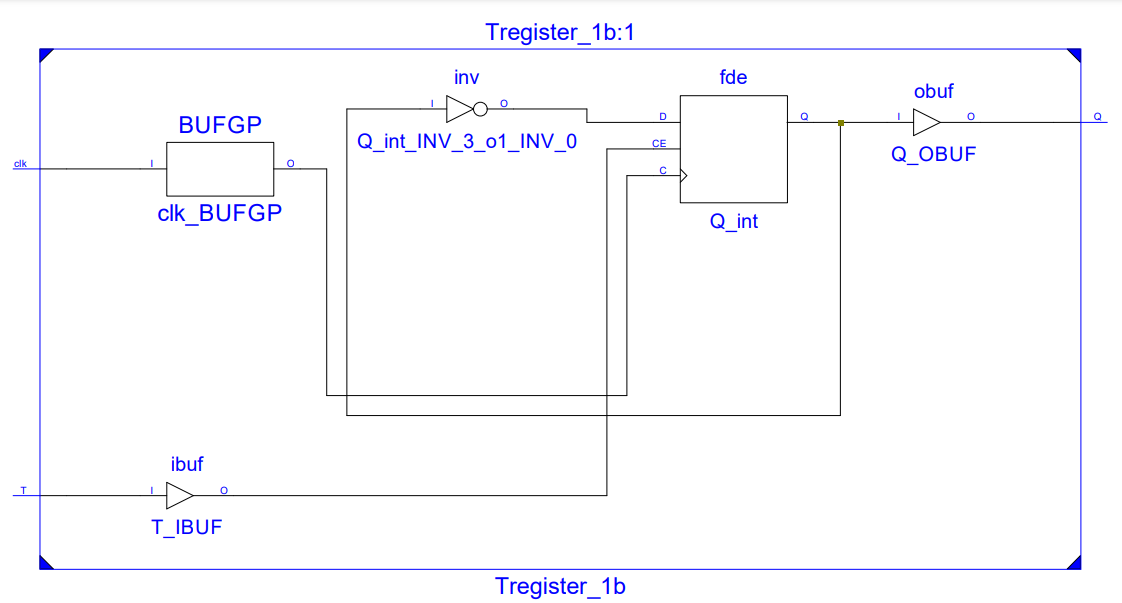
Le register\_8b permet, dans le bloc display, de contrôler l’affichage du chronoscore. On utilise dans display deux blocs register\_8b qui vont contrôler tout d’abord quelle partie du tableau d’affichage on veut contrôler (équipe à domicile ou visiteur + temps) puis le second bloc détermine quel est le chiffre que l’on veut afficher à l'endroit que l’on a décidé ( 8 bit car on a besoin de 7 leds pour afficher un chiffre et le 8ème contrôle le point). Cette gestion de l’affichage sera contrôlée par les sorties LEDS[7:0] (pour les chiffres) et AN[7:0] (pour la sélection sur le tableau d’affichage). Ils font alors la conversion série parallèle entre les entrées et l’affichage des anodes.

### Fonction Tregister\_1b:

La fonction Tregister\_1b est une bascule T synchrone active sur front montant du signal d'horloge. L’état de sortie de la bascule est inversé lorsque l’entrée T est à l’état HAUT.

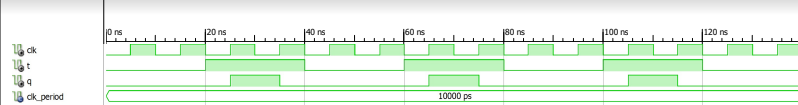
Les entrées de l’entité VHDL sont définies comme un signal clk et comme un signal T. La sortie de l’entité VHDL est définie comme un signal Q.

Cette table fait apparaître l’état du signal interne Q\_int nécessaire pour la description VHDL de l'état du registre.

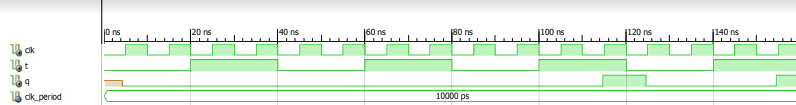
Vue technologique du composant :

Réponse comportementale du composant :

Behavorial :



Post-Route :



L’entrée T change toutes les secondes a l’aide du clock enable de periode 1s .

Conclusion sur le composant :

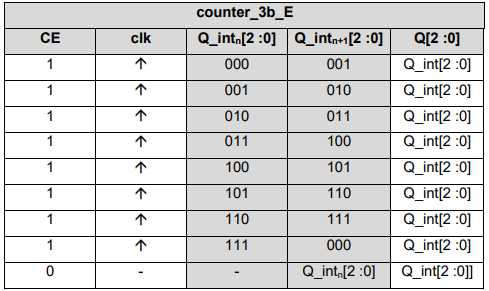
Le Tregister\_1b envoie toutes les secondes un 1 ou un 0 et les transferts au multiplexeur 1b.

### 

### Fonction counter\_3b\_E:

La fonction counter\_3b\_E est un compteur 3 bits actif sur front montant du signal d'horloge. Il dispose d’une entrée de validation CE (Clock Enable) synchrone et active à l'état HAUT.

Les entrées de l’entité VHDL sont définies comme un signal clk et comme un signal CE. Les sorties de l’entité VHDL sont définies comme un vecteur Q de dimension 3.

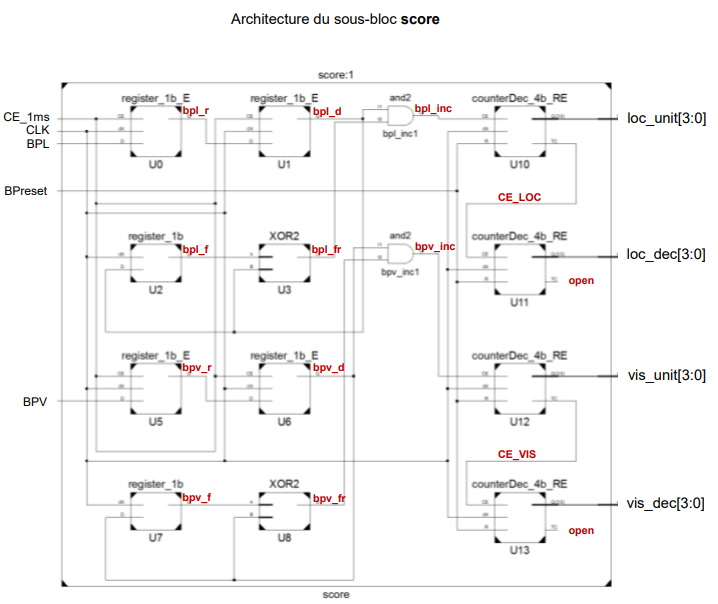
Cette table fait apparaître l’état du signal interne Q\_int[2 :0] nécessaire pour la description VHDL de l’état du compteur.

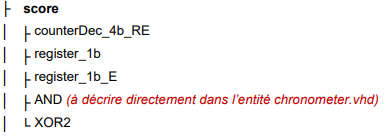
Conclusion sur le composant:

La fonction counter\_3b\_E est un compteur qui permet de sélectionner quelle anode est sélectionnée pour l’affichage en attribuant à chacun un chiffre de 0 à 7 en binaire soit les 8 anodes disponibles pour le score et le temps. Ce compte est réitéré toutes les millisecondes grâce a la clock CE\_1ms qui permettra, lorsque la clock est a l’état haut, de passer a la prochaine valeur de 0 à 7 et de réitérer le processus.

## Schéma Score

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



Le sous-bloc score comporte 5 fonctions : 

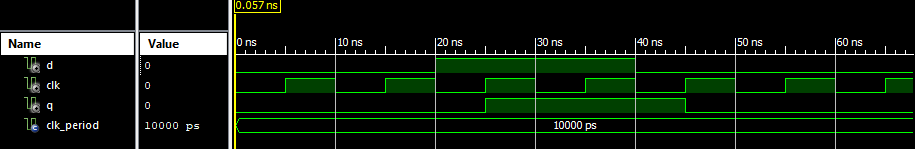
Il y a 2 fonctions de type combinatoire et 3 fonctions de type séquentiel dont une est également présente dans chronometer (counterDec\_4b\_RE).

### Fonction register\_1b:

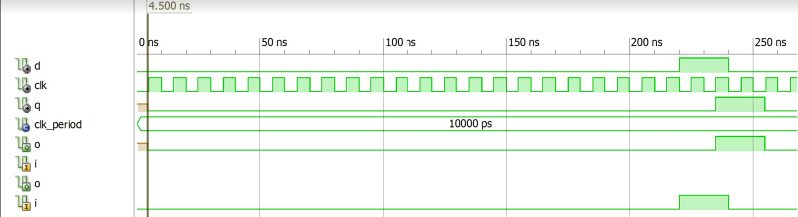
La fonction register\_1b est un registre synchrone constitué d’une bascule D active sur front montant du signal d'horloge clk.

Réponse comportementale du composant :

Behavorial :



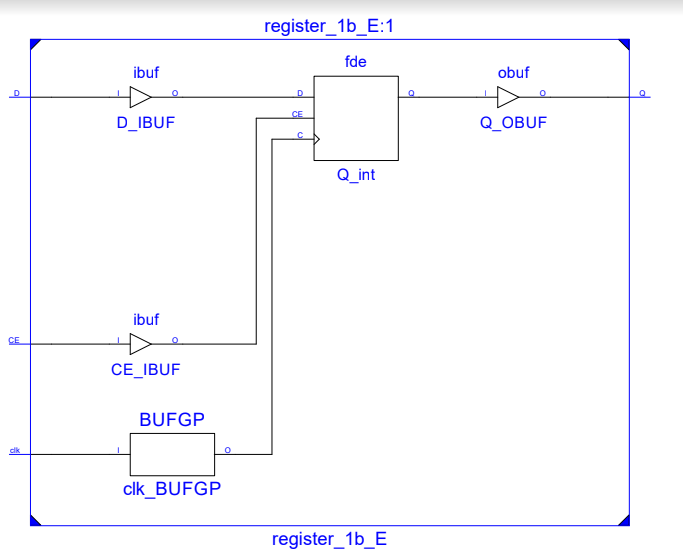
Post-Route:



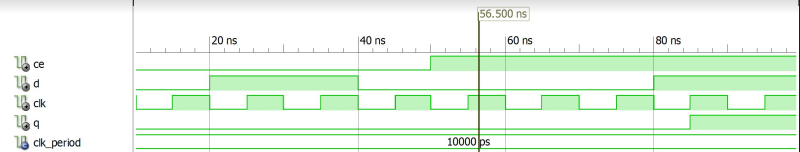
### Fonction register\_1b\_E:

La fonction register\_1b\_E est un registre synchrone constitué d’une bascule D active sur front montant du signal d'horloge clk. Il dispose d’une entrée de validation CE (Clock Enable) synchrone active à l'état HAUT.

Vue technologique du composant :



Réponse comportementale du composant : (behavorial)



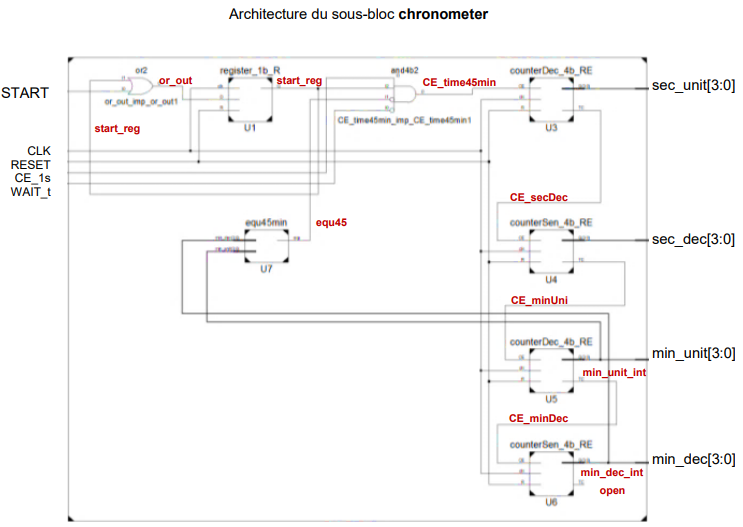
### Fonction counterDec\_4b\_RE:

La fonction counterDec\_4b\_RE est un compteur 4 bits qui compte de 0 à 9 et qui est actif sur front montant du signal d'horloge clk. Il dispose d’une entrée de remise à zéro R (Reset) asynchrone active à l'état HAUT et d’une entrée de validation CE (Clock Enable) synchrone active à l'état HAUT.

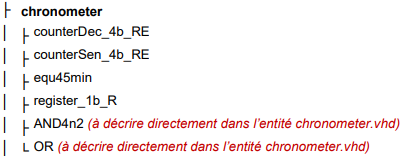
Il dispose également d’un signal de sortie TC (Terminal Count) qui passe à l’état HAUT pendant une période d’horloge en fin de comptage.

## Schéma Chronometer

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



Le sous-bloc chronometer comporte six fonctions :

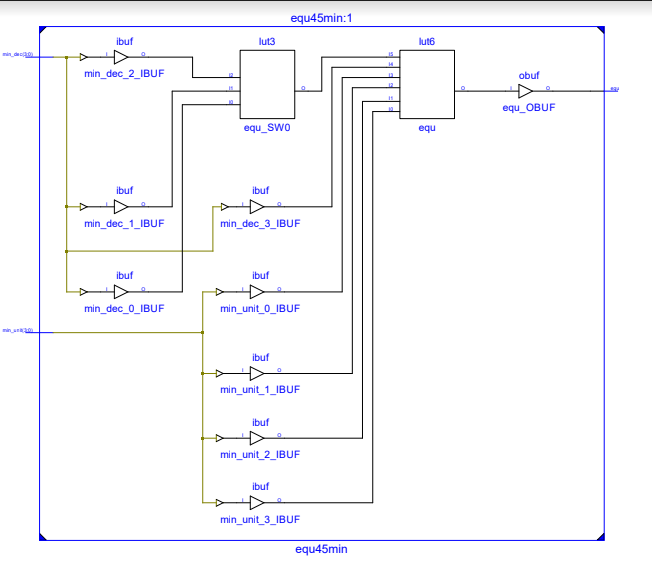
* OR 
* AND4n2
* equ45min
* register\_1b\_R
* counterSen\_4b\_RE
* counterDec\_4b\_RE

Il y a 3 fonctions de type combinatoire et 3 fonctions de type séquentiel.

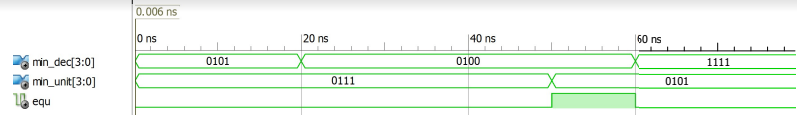
### Fonction equ45min:

La fonction equ45min est un comparateur logique. Lorsque l'entrée des unités vaut « 5 », et que celle des dizaines vaut « 4 », la sortie vaut « 1 ». Sinon, elle vaut « 0 ».

Vue technologique du composant :



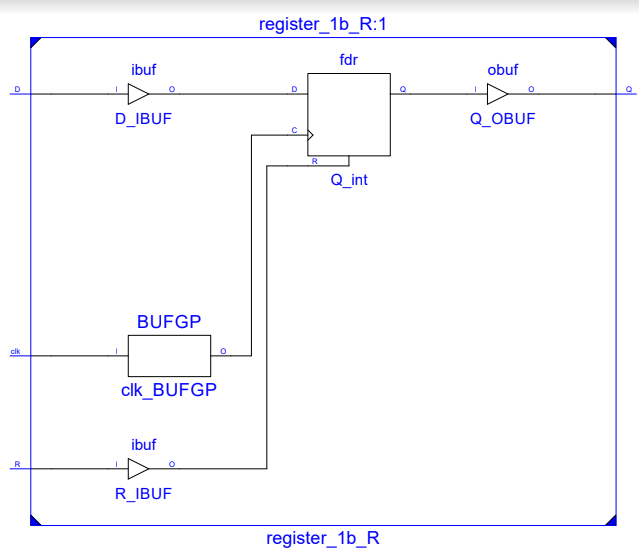
Réponse comportementale du composant (behavorial):



### Fonction register\_1b\_R:

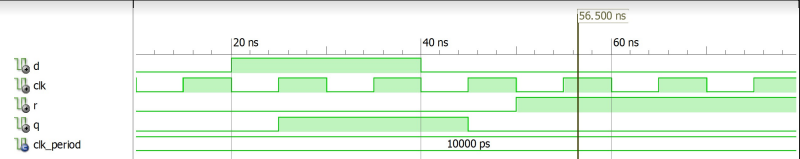
La fonction register\_1b\_R est un registre synchrone constitué d’une bascule D active sur front montant du signal d'horloge clk. Il dispose d’une entrée de remise à zéro R (Reset) synchrone active à l'état HAUT.

Vue technologique du composant :

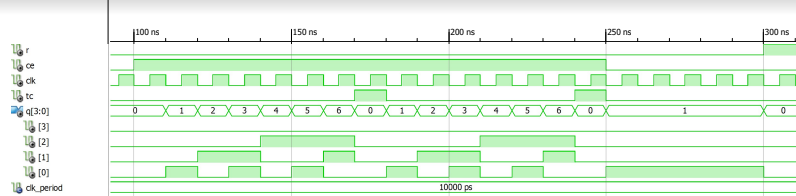


Réponse comportementale du composant :

Behavorial :



Post-Route :

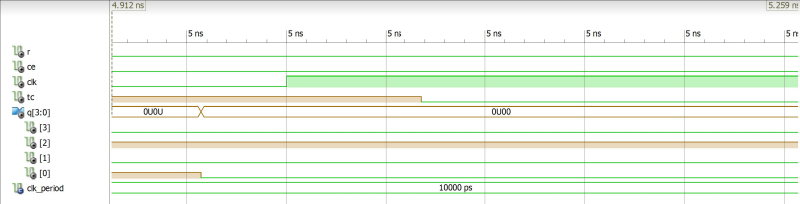


### Fonction counterSen\_4b\_RE:

La fonction counterSen\_4b\_RE est un compteur 4 bits qui compte de 0 à 5 et qui est actif sur front montant du signal d'horloge clk. Il dispose d’une entrée de remise à zéro R (Reset) asynchrone active à l'état HAUT et d’une entrée de validation CE (Clock Enable) synchrone active à l'état HAUT.

Il dispose également d’un signal de sortie TC (Terminal Count) qui passe à l’état HAUT pendant une période d’horloge en fin de comptage.

Réponse comportementale du composant (Post-Route):

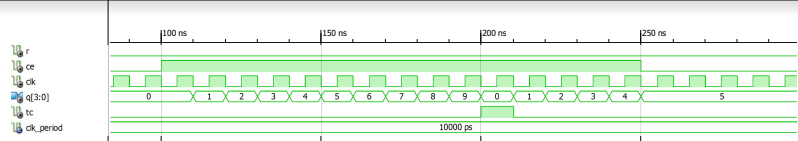


### Fonction counterDec\_4b\_RE:

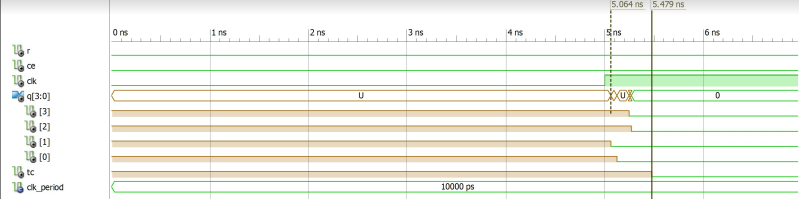
Voir sous-bloc Score.

Réponse comportementale du composant :

Behavorial :



Post-Route :



# MÉTHODE D'IMPLÉMENTATION / TEST

Nous avons consacré la phase 1 du projet au développement sous-bloc display dédié à la gestion des afficheurs 7-segments de la carte. Le test de ce sous-bloc a permis de valider le bon fonctionnement de chaque fonction réalisée jusqu’à présent.

De la même manière, à la fin de la seconde phase, consacrée à la gestion du chronomètre et du score, nous avons testé ces deux sous-blocs pour valider le fonctionnement des nouvelles fonctions.

Les blocs timeGenerator et vgaDisplay ont été mis à notre disposition.

Pour la validation globale du projet, des fichiers .ngc des sous-blocs chronometer score et display nous ont été fournis.

Nous avons donc créé le projet final en portant l’ensemble des fichiers .ngc sur la carte à partir d’iMPACT et avons validé son bon fonctionnement.

De là, nous avons remplacé le fichier chronometer.ngc par l’ensemble de ses entités VHDL. Nous en avons testé la réalisation à l’aide de la carte de développement et fait valider par un assistant. Nous avons fait de même avec le score.ngc puis display.ngc.

Finalement, avec l’ensemble de nos blocs créés en VHDL, le projet fonctionne et respecte le cahier des charges.

# CONCLUSION

## Retour d’expérience/Opinion sur le projet

On a vraiment apprécié ce projet, d’une part parce que évidemment on touche à quelque chose de concret avec un projet qui est abouti. D’autre part, on a pu se rendre compte de certaines subtilités lors de la conception de ce genre de projet (anti-rebond pour les boutons poussoirs…) et de la quantité de scénarios auxquels ont doit penser que ce soit en amont de la création de fonction ou même en aval pour corriger certains problèmes possibles.

## Discussion sur de potentiels problèmes rencontrés / solutionnés ou pas

Lors de la réalisation du chronoscore nous n’avons rencontré aucun problème. C’est au moment du téléversement sur la carte que nous avons réalisé qu’une de nos fonctions dans le bloc display contenait une erreur. Effectivement, les affichages des scores et du temps nous paraissaient faux, nous nous étions trompés sur le bit de poid fort dans la définition des fonctions du bloc display.

## Proposition d’ajout ou modification au projet pour améliorer chronoscore

On pourrait penser à afficher le nombre de remplacements de joueurs déjà effectués par chaque équipe, sous forme de numéro comme le score ou alors de points qui s'allumeraient.

# ANNEXES

## Codes VHDL:

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

