ELE3311 – Systèmes logiques programmables
Projet n°3

SIMON - JEU DE MÉMOIRE - AFFICHAGE OLED

Hiver 2020

Introduction

Le but de ce projet de laboratoire est d'approfondir des concepts fondamentaux des systèmes logiques programmables. Ce projet consiste à réaliser **des éléments** d'un jeu de mémoire de type Simon avec un affichage sur l'écran OLED et l'utilisation des boutons de la carte de développement. Les éléments choisis dans le cadre du projet 3 seront réalisés indépendamment du projet 2 afin de **limiter la durée du projet à 2 semaines**.

Les concepts approfondis lors de ce projet sont :

- la resynchronisation du signal de remise à zéro asynchrone (*reset*);
- la description comportementale de mémoires (une RAM et une ROM) et
- la création d'un banc d'essai autonome (*self-checking testbench*).

Description du projet

Le projet consiste à écrire dans une mémoire de trame (*frame buffer*) pour l'écran OLED et à envoyer la charte des pixels (*character map*) des caractères correspondant au système en aval, représenté ici par le banc d'essai. Chaque caractère est composé de 64 pixels (8x8). Un compteur d'adresse parcours la mémoire de trame une ligne à la fois et pour chaque caractère, une colonne à la fois dans la charte.

Le tableau ci-dessous donne la correspondance entre l'adresse de la mémoire de trame et la position sur l'écran OLED **:**

0	1	2												14	15
16	17	18			Chaque case du tableau correspond à un caractère 8x8 sur l'écran OLED.									30	31
32	33	34												46	47
48	49	50												62	63

Le tableau ci-dessous donne un exemple de caractère 8x8 **:**

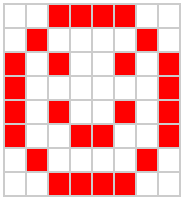
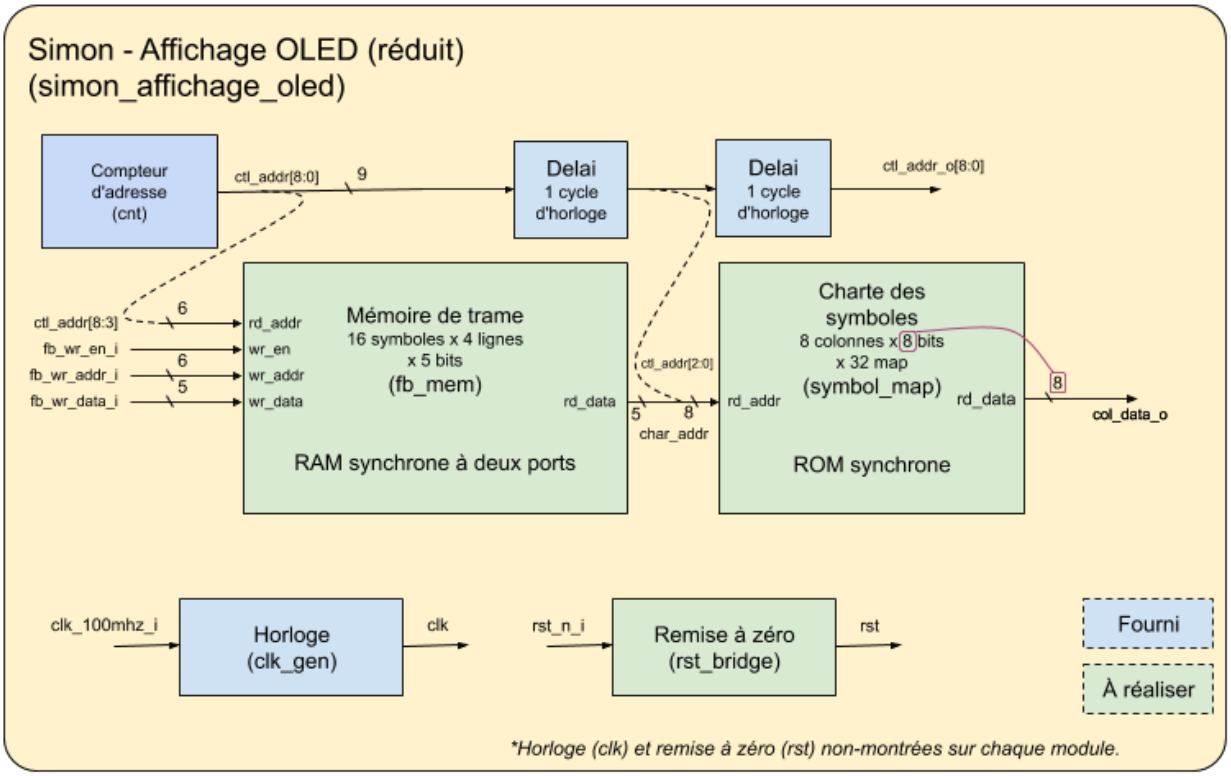


Schéma-bloc

Le schéma ci-dessous montre l'architecture du système. Les modules en bleu sont fournis. Les modules en vert doivent être implémentés.



Les entrées et sorties du niveau hiérarchique supérieur sont données dans le tableau ci-dessous.

Nom	Largeur	Direction	Description
rst_n_i	1	entrée	Réinitialisation du système. Bouton *cpu_resetn*.
clk_100mhz_i	1	entrée	Horloge 100 MHz.
fb_wr_en_i	1	entrée	Activation de l'écriture dans la mémoire de trame
fb_wr_addr_i	6	entrée	Adresse pour l'écriture dans la mémoire de trame
fb_wr_data_i	5	entrée	Donnée à écrire dans la mémoire de trame
ctl_addr_o	9	sortie	Compteur d'adresse
col_data_o	8	sortie	Pixels actifs (OLED) pour l'adresse donnée

Description des modules

Horloge (fourni)

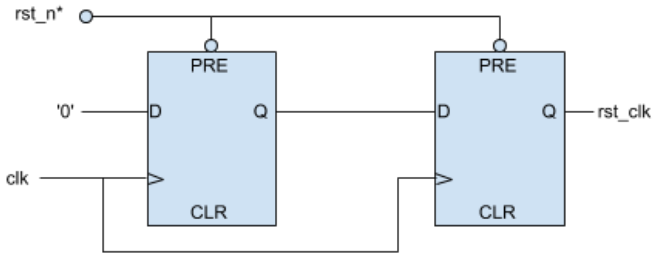
Une horloge de 100 MHz est disponible sur la carte Nexys Video. Cette horloge ***clk_100mhz_i*** est propagée à l'ensemble du systèmes à l'aide d'un arbre de distribution (BUFG) qui est normalement instancié dans le niveau hiérarchique supérieur.

Nom	Largeur	Direction	Src/Dst	Description
clk_100mhz_i	1	entrée	port	Horloge 100 MHz
clk	1	sortie	multiples	Horloge 100 MHz (*buffered*)

Remise à zéro (à réaliser)

La remise à zéro ***rst_n_i*** est utilisée pour le compteur d'adresse seulement. La remise à zéro à l'allumage (*power-on reset*) généré à l'extérieur du FPGA est asynchrone et doit être resynchronisée, ce qui n'a pas été fait dans le cadre du projet 2. On vous demande d'implémenter un pont de resynchronisation asynchrone pour la remise à zéro (**rst_bridge**). La sortie du pont de resynchronisation devient active de façon **asynchrone** lorsque le signal de remise à zéro est actif (à '0') et est désactivé de façon **synchrone** lorsque le signal de remise à zéro est relâché (à '1').

Le schéma ci-dessous montre la **structure** à réaliser. Assurez-vous que les niveaux logiques spécifiques au projet soient respectés.



Nom	Largeur	Direction	Src/Dst	Description
rst_n_i	1	entrée	port	Remise à zéro asynchrone (active basse)
clk_i	1	entrée	clk_gen	Horloge 100 MHz
rst_clk_o	1	sortie	cnt	Remise à zéro, déassertion resynchronisée

Compteur d'adresse (fourni)

Le compteur d'adresse (**cnt**) est un simple compteur séquentiel avec remise à zéro asynchrone. Le compteur du projet 2 est réutilisé.

Nom	Largeur	Direction	Src/Dst	Description
clk_i	1	entrée	clk_gen	Horloge 100 MHz
rst_i	1	entrée	rst_bridge	Remise à zéro synchrone
set_1_i	1	entrée	'0'	Fixe à la valeur du compteur à 1
inc_i	1	entrée	'1'	Incrémente la valeur du compteur de 1
cnt_o	WIDTH	sortie	multiples	Valeur du compteur

IMPORTANT : Dans le rapport, vous devez fournir l'évaluation des ressources nécessaires à la réalisation de ce compteur, soit le nombre de LUT, de FF et de ports d'entrée/sortie.

Mémoire de trame (à réaliser)

La mémoire de trame (**fb_mem**) conserve le contenu à afficher sur l'écran OLED. L'écran de 128 x 32 pixels est divisé en symboles de 8 x 8 pixels. Ainsi la mémoire doit conserver 16 symboles de large x 4 symboles de haut pour un total de 64 symboles. L'architecture choisie permet de différencier 32 symboles, soit 5 bits de données. La mémoire de trame contient donc, pour chacun des 64 symboles à afficher, les 5 bits identifiant le "caractère" dans la **charte des symboles**.

Le contenu des 64 adresses de la mémoire est modifié par le port d'écriture. Pour modifier une donnée, il faut activer l'écriture (***wr_en_i*** = '1') et avoir une adresse et une donnée valide au prochain front montant de l'horloge.

La récupération des données pour affichage se fait à travers le port de lecture. La donnée visée par l'adresse de lecture ***rd_addr_i*** est transférée sur le port de sortie ***rd_data_o*** à chacun des fronts montants de l'horloge.

Nom	Largeur	Direction	Src/Dst	Description
clk_i	1	entrée	clk_gen	Horloge
rd_addr_i	6	entrée	cnt	Port de lecture - adresse
rd_data_o	5	sortie	symbol_map	Port de lecture - donnée
wr_en_i	1	entrée	port	Port d'écriture - activation (1 = vrai)
wr_addr_i	6	entrée	port	Port d'écriture - adresse
wr_data_i	5	entrée	port	Port d'écriture - donnée

Pour décrire la mémoire de trame, référez-vous aux *Language Templates* disponibles dans l'outil Vivado Design Suite et copiée dans le répertoire **template** du projet.

Charte des symboles (à réaliser)

La charte des symboles (**symbol_map**) est une mémoire morte qui contient la matrice d'affichage de chaque symbole utilisé dans le cadre du projet. Le port de sortie ***rd_data_o*** est mis à jour à chaque front montant de l'horloge, en fonction de la valeur sur ***rd_addr_i***. Afin de limiter le travail redondant, vous devez décrire 3 symboles seulement :

Adresse 0	Espace (aucun pixel allumé)
Adresse A	Premier symbole de votre choix
Adresse A+1	Deuxième symbole de votre choix

L'adresse A est choisie par chaque équipe et est comprise entre 1 et 30.

Nom	Largeur	Direction	Src/Dst	Description
clk_i	1	entrée	clk_gen	Horloge 100 MHz
rd_addr_i	8	entrée	multiples	Port de lecture - adresse
rd_data_o	8	sortie	port	Port de lecture - donnée

Délai (fourni)

Afin d'aligner la valeur du compteur d'adresse avec la sortie ***col_data_o***, il est nécessaire d'ajouter deux cycles d'horloge de délai (1 cycle par mémoire). Cette fonctionnalité est décrite directement dans le niveau hiérarchique supérieur.

Banc d'essai (à réaliser)

Le banc d'essai contrôle les entrées et vérifie les sorties du DUT (*Device Under Test*). Il doit écrire le code d'un symbole valide (0, A ou A+1) dans la mémoire de trame et vérifier que les sorties du système correspondent aux valeurs attendues suite à la modification de la mémoire de trame. Afin de vérifier les valeurs attendues, vous devez utiliser des déclarations de type **assert**.

```
assert <condition> report <string> severity <severity_level>;
```

Exemple :

```
assert col_data=X"F2" report "1er symbole, 2e colonne" severity error;
```

La section suivante donne un aperçu de ce que devrait faire le banc d'essai.

Plan de vérification

Le plan de vérification liste les requis et la façon de les vérifier. C'est une étape nécessaire à la création d'un environnement de vérification, ici, le banc d'essai. Afin de limiter le nombre de document à produire au cours du projet, on vous demande d'ajouter le plan de vérification sous forme de commentaires dans le banc d'essai.

Votre plan de vérification doit spécifier, entre autre, les adresses choisies (A, A+1), les opérations faites sur la mémoire de trame et les vérifications à faire (à quel moment et quelle valeur) sur les sorties.

N'hésitez pas à discuter tôt avec l'équipe de chargés/répétiteurs pour valider votre plan de vérification lors de la première séance de laboratoire.

Fichiers fournis

Des fichiers sources sont fournis et toutes les parties à compléter sont indiquées en commentaires.

Travail au laboratoire

Avant la première séance, vous devez vous assurez d'avoir accès à un outil de simulation. Trois options sont disponibles :

- Installer Vivado Webpack sur votre PC
- Accéder à Vivado sur un PC au laboratoire
- Créer un compte sur EDA Playground

Voici le travail attendu lors de chacune des deux séances de laboratoire :

Séance 1 : Plan de vérification. Description HDL de la **mémoire de trame** et du **banc d'essai**. Simulation et validation de la fonctionnalité de la **mémoire de trame**.

Séance 2 : Description HDL de la **charte des symboles** et de la **remise à zéro** (*reset bridge*). Simulation et validation de la fonctionnalité.

Remises en ligne

Vous devez remettre votre rapport de laboratoire et faire le quiz individuel (Moodle Quiz sur le site Web du cours) avant vendredi le 17 avril à 18h00.

Remarques

- Tout le travail doit être fait en VHDL en utilisant les fichiers fournis pour ce laboratoire.
- Veuillez ajouter le barème de la page suivante dans le rapport.
- Les points du barème de la page suivante indiquent les éléments à mettre dans le rapport. Ne mettez aucun autre texte qui n'est pas explicitement demandé.
- De l'information supplémentaire sur la plateforme de développement est disponible sur les liens suivants :
    - https://reference.digilentinc.com/reference/programmable-logic/nexys-video/reference-manual

ELE3311 – Systèmes logiques programmables – Hiver 2020

BARÈME D'ÉVALUATION DU PROJET 3 : SIMON - AFFICHAGE OLED

No d'équipe : _____

Noms : _____ Matricules : _____

_____ _____

_____ _____

A)	**Évaluations intermédiaires**	**/4**
	Séance 1: Plan de vérification (**simon_affichage_oled**) /2	
	Simulation partielle (**simon_affichage_oled, fb_mem**) /2	
B)	**Fonctionnement complet (séance 2)**	**/5**
	Séquence d'écriture de symbol en mémoire (**fb_mem, banc d'essai**) /1	
	Lecture de données (**symbol_map**) et par le banc d'essai (**banc d'essai)** /1	
	Vérification des symboles lus en fonction des accès (**banc d'essai**) /1	
	Remise à zéro resynchronisée (**rst_bridge**) /1	
	Exactitude des résultats /1	
C)	**Rapport**	**/7**
	Plan de vérification final (**simon_affichage_oled**) /1	
	Simulation commentée (**simon_affichage_oled**) /1	
	Description VHDL commentée (t**ous les modules**) /3	
	Vérification manuelle pour les latches /1	
	Évaluation manuelle des ressources (**FF, LUT, IO**) (**cnt**) /1	
D)	**Quiz individuel -** sur Moodle Quiz, avant de remettre le rapport	**/4**
TOTAL :		***/20***

Note: Ce rapport doit être remis au plus tard le 17 avril à 18h00 (pour tous les groupes). Les retards aux évaluations sont pénalisés.