Micro-architecture (UE S3)

Dossier de Projet

Serpentin 7-segment programmable (FPGA)

Etudiants :
Afizullah Rahmany
Romain Pereira

Enseignant :
M. Augé

25/10/2018

Table des matières

Préambule

Ce projet a été réalisé dans le cadre de nos études à l'ENSIIE (Ecole National Supérieur d'informatique pour l'industrie et l'entreprise) d'Evry.

Ce projet est l'aboustissement de nos cours en Micro-architecture.

Nous programmions sur un FPGA d'Altera (gamme Cyclone), en VHDL et à l'aide du logiciel Quartus.

L'objectif a été de réalisé un serpentin programmable, et affichable sur un 7-segment.

1 Introduction

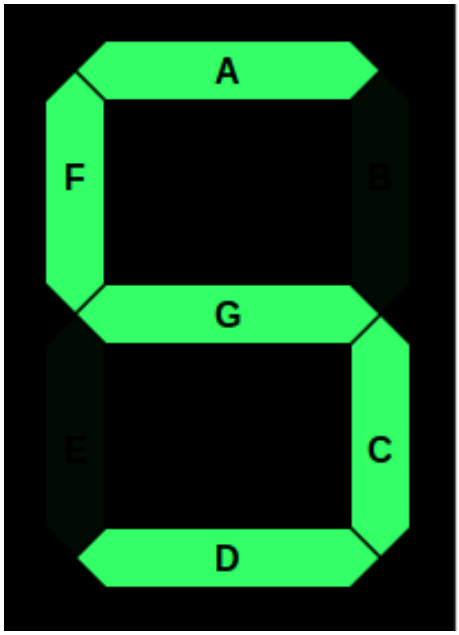
Vous cherchez un cadeau de noël pour vos enfants ou vos grands parents ?
Ne cherchez plus, voici **le Serpentin 3000**
Cette modeste plaquette electronique passionera les petits comme les grands enfants.

Amusez vous !
Modifier la vitesse et le type de défilement.

Exprimez votre créativité !
Grâce à l'interface 'Programmation 3000', vous pouvez créer votre propre serpentin.

Accessible Pour la modique somme de 29.99, offrez vous un

SERPENTIN 3000



2 Manuel utilisateur

Bouton de droite 'Bouton reset' , réinitialise l'état du serpentin dans son état initial

Interrupteurs 'IVDF2 IVDF1 IVDF0' permettent de régler la vitesse de défilement

// TODO : graph des vitesses

Diodes d'interrupteur 'DVDF2 DVDF1 DVDF0' allumé ou éteintes selon que les interrupteurs IVDF2, IVDF1 ou IVDF0 sont actifs ou inactifs.

Diode défilement 'DVDF' cette diode clignote à la vitesse de défilement

Selectionneur serpentin 'SEL1 SEL2' permettent de selectionner le type de serpentin

// TODO graph des serpentins

BUS - IA BUS-IA, prends un message pour communiquer avec le serpentin 3.2

3 Présentation générale

3.1 Fonctionnement

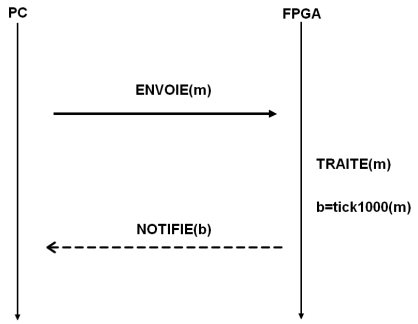


FIGURE 1 – MSC des actions utilisateur

m est un *message*. Il s'agit d'une suite de **44 bits** avec le format :

- **bits 43 à 24** Ce sont les bits du contrôle du BUS-IA (voir Bus-IA)
- **bits 23 à 0** Ce sont les bits du message

Un *message* doit avoir le format :

- **bits 23 à 21** Ce sont les bits coddant le *type* du *message*
- **bits 20 à 0** Ce sont les données du message, le format varie selon le *type*

3.2 Format des messages

Ci-dessous, voici les commandes supportées par le processeur, et le format des messages permettant d'executer ces commandes.

noop Ne rien faire

- **bits 23 à 21** : "000" (= (bit 23, bit 22, bit 21))
- **bits 20 à 0** : *non utilisées*

h-init(n) Génère un tick sur H100 tous les n coups d'horloge maître

- **bits 23 à 21** : "001"
- **bits 20 à 0** : *non utilisées*

h-check-ON() Demande au processeur d'envoyer au PC un message TICK1000 tous les 1000 tickets de H100.

- **bits 23 à 21** : "010"
- **bits 20 à 0** : *non utilisées*

h-check-OFF() Demande au processeur d'arrêter d'envoyer au PC un message TICK1000

- **bits 23 à 21** : "011"

- **bits 20 à 0** : *non utilisées*

clr() Indique au processeur d'afficher un serpentin vide sur le 7-segment (**n** à 32 et met à 0 toutes les *valeurs* programmées

- **bits 23 à 21** : "100"

- **bits 20 à 0** : *non utilisées*

set-N(n) Indique au processeur le nombre de **valeurs** à faire défiler en boucle sur le 7-segments. (**n** compris entre 1 et 32)

- **bits 23 à 21** : "101"

- **bits 20 à 6** : *non utilisées*

- **bits 5 à 0** : **n** coddé sur 6 bits ($2^6 = 64$)

set-val(i, v) Indique au processeur la i-ème valeur du 7-segments est **v**

- **bits 23 à 21** : "110"

- **bits 20 à 13** : *non utilisées*

- **bits 12 à 6** : **v** les 7 bits correspond aux 7 segments de l'afficheur (0 => éteinds, 1 => allumé)

- **bits 5 à 0** : **n** coddé sur 6 bits ($2^6 = 64$)

4 Description matérielle

4.1 Schéma général

// TODO SCHEMA GENERAL

5 Présentation générale

5.1 Schéma général

5.2 Description des blocs

5.2.1 Wrapper

Il s'agit de l'interface de communication dans le BUS-IA.

Ce bloc suit le protocole 'handshake' dans le BUS-IA.

Il récupère les messages qui lui sont destinés, puis les renvoie vers le processeur du serpentin.

Entrées

- clk, reset : la master clock et le reset

- busin : entrée dans le BUS-IA (44 bits, protocole handshake)

- busin_valid : entrée dans le BUS-IA (1 bit, protocole handshake)

- busout_eated : entrée dans le BUS-IA (1 bit, protocole handshake)

Sorties

- busout : sortie vers le BUS-IA (44 bits, protocole handshake)

- busout_valid : sortie vers le BUS-IA (1 bit, protocole handshake)

- busin_eated : sortie vers lle BUS-IA (1 bit, protocole handshake)

- busmsg : sortie vers le processeur du serpentin (24 bits, pas de protocole)

5.2.2 Dispatcher

Il s'agit du bloc de répartition.

Il récupère les *messages* qui lui sont destinés, les interprète, et modifie ses sorties en fonction.

Entrées

- clk, reset : la master clock et le reset

- busmsg : le message a interpreté (24 bits)

Sorties

- bus_ss : sortie vers le 7-segment (32 * 7 = 224 bits), coddant les configurations possibles du 7-segments.

- bus_n : sortie vers le 7-segment (6 bit), coddant 'N' (3.2)

- bus_n_clock : sortie vers le générateur de ticks (20 bits), coddant le nombre de **kilo master clock** à attendre pour générer un tick. (la master clock étant à 50MHz, pour une entrée

$$(1001110001000)_2 kHz = (5000)_{10} kHz = (5000000) Hz$$

On aura 100 ticks seront générés par secondes.)

- bus_tick1000 : sortie vers le générateur de tick, '1' ou '0' selon que le générateur doit envoit un message au PC tous les 1000 ticks.

5.2.3 H100

Il s'agit du générateur de tick.

Il récupère un nombre *N_ clock*, puis compte *N_ clock* **kilo master clock** avant de générer un tick.

Lorsqu'un tick est généré, sa sortie passe à '1' pendant un master clock, sinon elle vaut '0'

Entrées

- clk, reset : la master clock et le reset
- N_clock : le nombre de **kilo master clock** (20 bits)

Sorties

- T : '1' ou '0' si un tick est généré ou non (1 bit)

5.2.4 H10

Il s'agit d'un compteur de tick.

Il module le signal du tick en un signale de clignotement.

Sa sortie alterne entre '1' et '0' tous les 10 ticks.

Une diode est branché sur ce bloc, elle clignote avec une fréquence de 10 ticks.

Entrées

- clk, reset : la master clock et le reset
- T : un tick (1 bit)

Sorties

- S : le signal de sortie (1 bit)

5.2.5 Moduleur

Il s'agit d'un moduleur de tick programmable.

Il prends un entier X coddé sur 3 bits en entrée (avec donc 7 valeurs possibles)

Il génère un tick tous les Y ticks de H100, avec la correspondance :

X	"000"	"001"	"010"	"011"	"100"	"101"	"110"	"111"
Y	2	5	7	10	12	15	17	20

Ce tick est envoyé dans l'entrée du serpentin programmable.

Entrées

- clk, reset : la master clock et le reset
- E2 E1 E0 : l'entier coddé sur 3 bits (3 x 1 bit)
- T : un tick (1 bit)

Sorties

- S : le tick de sortie (1 bit)

5.2.6 Serpentin Programmable

Il s'agit du serpentin programmable.

On appelle *frame* 7 bits configurant l'état des segments sur le 7-segment.

Si le i-ème bit est à '1', cela indique que le i-ème segment doit être allumé, sinon il est éteinds.

Ce bloc prends en entrée un tableau de 32 *frames*.

Il selectionne successivement (modulo N) les frames 1 à N, lorsqu'un tick est reçu.

Entrées

- clk, reset : la master clock et le reset
- T : un tick venant du moduleur (1 bit)
- frames : les images programmées sur le 7-segment (7 * 32 = 224 bits)
- N : le nombre de frames sur lesquelles le 7-segment doit boucler (entre 1 et 32, 6 bits)

Sorties

- S6 S5 S4 S4 S3 S2 S1 S0 : les 7 bits coddant la frame en cours.

5.2.7 Serpentin clignottant

Il s'agit du serpentin pré-programmé.

Il sort la configuration du serpentin correspondant à un clignotement.

Sorties

- N : "000010" (2, le serpentin boucle sur 2 frames)
- frames : "000...000 1111111" (toutes les frames éteintes, et une allumé)

5.2.8 Serpentin horaire

Il s'agit du serpentin pré-programmé.

Il sort la configuration du serpentin correspond à une rotation horaire.

Sorties

- N : "000110" (6, le serpentin boucle sur 6 frames)
- frames : la configuration correspondante

5.2.9 Serpentin anti-horaire

Il s'agit du serpentin pré-programmé.

Il sort la configuration du serpentin correspond à une rotation anti-horaire.

Sorties

- N : "000110" (6, le serpentin boucle sur 6 frames)
- frames : la configuration correspondante

6 Implémentation du bloc ? ? ? ?