Professeur : Etienne Messerli
Assistant : Anthony Convers

Laboratoire 06 – Mesure du temps de réaction

Objectifs du laboratoire

Ce laboratoire a pour but de concevoir une application de mesure du temps de réaction avec l'utilisation de plusieurs afficheurs externe. Le laboratoire comprendra plusieurs étapes :

- Mise en œuvre d'une interface UART du HPS, afin d'envoyer des messages dans un terminal distant.
- Développement et interfaçage d'un émetteur série asynchrone 20 bits, afin de piloter l'affichage des leds de la carte Max10_leds.
- Mise en œuvre des interruptions du HPS générées par l'appui d'un bouton depuis la partie FPGA.
- Implémentation et interfaçage d'un compteur avec une précision de 20ns dans le FPGA.
- Génération d'un nombre aléatoire.

L'objectif final est d'assembler toutes les étapes citées ci-dessus dans l'application qui réalise une mesure du temps de réaction. Les spécifications concernant l'application sont données ci-après.

Spécifications

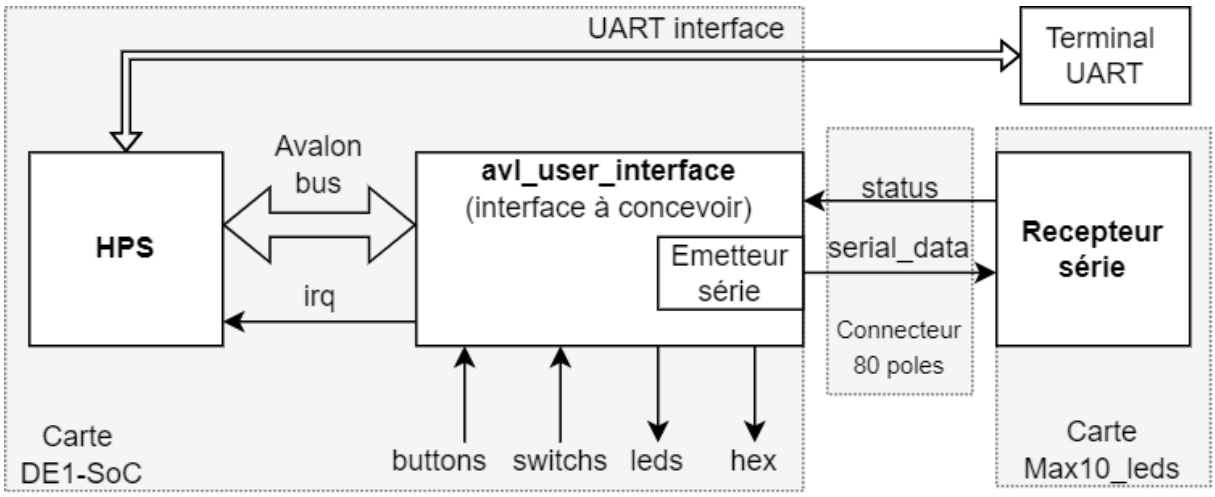
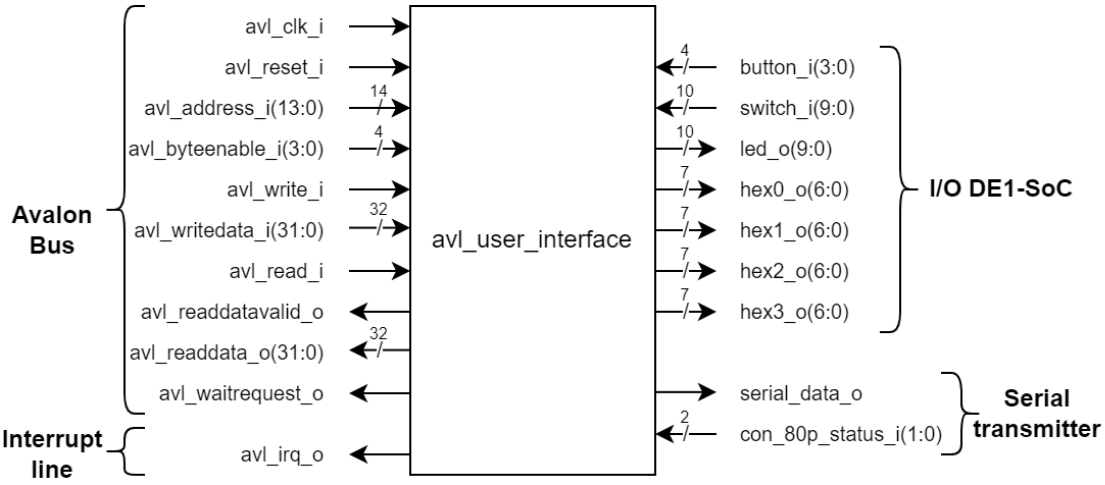


Schéma du system

Le composant VHDL "avl_user_interface.vhd" sera utilisée pour implémenter l'interface connecté sur le bus Avalon afin de communiquer avec l'émetteur série asynchrone, le compteur, le gestionnaire d'interruption et les différents périphériques. Le composant avl_user_interface sera aussi capable de générer des interruptions du HPS.

L'horloge du bus Avalon est à 50Mhz. Voici le symbole du composant avl_user_interface :



Symbole du composant avl_user_interface

L'interface développée doit permettre d'accéder, à travers le bus Avalon, aux périphériques suivants :

- Interface constant ID sur 32 bits (disponible à l'offset 0)
- 4 Boutons (Key) DE1-SoC
- 10 Interrupteurs (Switch) DE1-SoC
- 10 Leds DE1-SoC
- 4 Afficheurs 7 segments (Hex) DE1-SoC
- Emetteur série asynchrone, liaison vers la carte Max10_leds.
- Compteur avec une précision de 20 ns (interne à l'interface avl_user_interface).
- Gestionnaire des interruptions lors de l'appuie sur le bouton KEY0.

Interface UART du HPS

Utilisation du module UART0 du HPS.
Le module UART0 est configuré dans le mode "No flow control".
L'horloge l4_sp_clk est à une fréquence de 100MHz.

Configurer le module UART0 avec les caractéristiques :

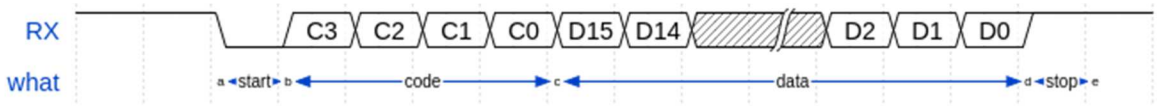
- Baudrate : 9600
- Bit de données : 8
- Bit de parité désactivé
- Bit de stop : 1
- Activer les buffers FIFO en émission et réception

Après avoir connecté la carte DE1-SoC (connecteur : UART to USB, J4) à un ordinateur avec le câble mini-USB, vous pouvez utiliser la commande suivante pour démarrer un terminal UART :

sudo picocom -b 9600 /dev/ttyUSBx x : numéro du port, en principe 0

Emetteur série asynchrone

Dans votre interface, réaliser un émetteur série asynchrone qui permet de communiquer avec la carte Max10_leds et ainsi de piloter les différentes Leds. Les 2 cartes sont reliées par le connecteur 80 pôles et communique seulement par la liaison série asynchrone de 20 bits sans parité. Un signal de statut sur 2 bits est également disponible et permet d'informer que la carte Max10_leds est correctement connecté et dans le bon mode de fonctionnement. Voici ci-après les caractéristiques du récepteur de la liaison série asynchrone utilisée.



Chronogramme liaison série asynchrone 20bits sans parité

A l'état de repos, la ligne est à l'état haut. Le début d'une transmission commence par la mise à l'état bas de la ligne, c'est le bit start. Ensuite, les 20 bits sont transmis, MSB en premier. Les 4 premiers bits reçus correspondent au code qui définit la zone de leds concernée. Les 16 bits suivants correspondent à la donnée qui donne l'état de chaque led dans la zone sélectionnée. La transmission se termine par l'envoi du bit stop, à l'état haut. Puis la ligne est de retour à l'état de repos.
La transmission se fait à un débit de 9600 bauds. L'horloge du système ne nous permet pas d'avoir la valeur exacte du débit, mais vous utiliserez la valeur la plus proche.
Les codes définissants les différentes zones de leds sont présentés dans le tableau ci-dessous.

code (3..0)	Spécification de Data (15..0)
0000	D15 not used, Leds secondes DS15...01
0001	D15 not used, Leds secondes DS30...16
0010	D15 not used, Leds secondes DS45...31
0011	D15 not used, Leds secondes DS60...46
0100	Ligne leds DL15....0
0101	Ligne leds DL31...16
0110	D15 not used, Carré de leds DM35-31/25-21/15-11
0111	D15-10 not used, Carré de leds DM55-51/45-41
1000	Leds heures DH04-01 Int(3..0), 4 bits d'intensité par led
1001	Leds heures DH08-05 Int(3..0), 4 bits d'intensité par led
1010	Leds heures DH12-09 Int(3..0), 4 bits d'intensité par led
1011	réservé
…	…
1111	réservé

En annexe, vous avez l'explication sur le codage des 4 bits d'intensité pour les leds des heures.

Le signal de statut (con_80p_status_i) sur 2 bits qui vient de la carte Max10_leds est défini de la manière suivante :
- 00 : configuration non valide.
- 01 : configuration valide.
- 1X : réservé

Compteur avec une précision de 20ns

Dans votre interface, vous devez implémenter un compteur sur 32bit avec une précision de 20ns.
Votre interface doit supporter les fonctionnalités suivantes :
- Accéder à la valeur actuelle du compteur.
- Activer/désactiver le compteur.
- Remettre à zéro le compteur.

Gestionnaire des interruptions de l'interface avl_user_interface

L'interface avl_user_interface dispose du signal "avl_irq_o" qui permet de générer une interruption lorsque ce signal est actif ('1'). Ce signal d'interruption est connecté au HPS sur la ligne qui porte le nom FPGA_IRQ0.

Votre interface doit supporter les fonctionnalités suivantes :
- Générer une interruption lors d'un appuie sur KEY0 (détection de flanc).
- Pouvoir masquer/démasquer l'interruption.
- Mécanisme qui permet d'acquitter l'interruption.
- Prévoir un statut qui informe sur l'état de la détection de flanc montant du bouton et l'état de l'interruption.

Plan d'adressage

Voici le plan d'adressage qui est spécifié pour le bus AXI lightweight HPS-to-FPGA du projet fourni.

Offset on bus AXI lightweight HPS-to-FPGA (relative to BA_LW_AXI)	**Fonctionnalités**
0x00_0000 – 0x00_0003	Design standard ID 32 bits (Read only)
0x00_0004 - 0x00_FFFF	reserved
0x01_0000 - 0x01_00FF	Zone disponible pour votre interface dans la FPGA
0x01_0100 - 0x01_FFFF	Zone disponible pour d'autres interfaces développées dans la FPGA
0x02_0000 - 0x1F_FFFF	not used

Voici le plan d'adressage proposé pour votre interface qui est à compléter.

Adresse (offset)	Read	Write
0x00	[31..0] Interface user ID	not used
0x04	[31..4] "0..0" ; [3..0] buttons	not used
0x08	[31..10] "0..0" ; [9..0] switchs	not used
0x0C	[31..10] "0..0" ; [9..0] leds	[31..10] reserved ; [9..0] leds
0x10	[31..28] "0..0" ; [27-21] hex3 ; [20-14] hex2 ; [13-7] hex1 ; [6-0] hex0	[31..28] reserved ; [27-21] hex3 ; [20-14] hex2 ; [13-7] hex1 ; [6-0] hex0
0x14 …. 0xFF	available for news functionality	available for news functionality

Spécifications du programme

Le but est de réaliser une application qui permet de mesurer le temps de réaction lors de l'appuie sur le bouton KEY0. Différent afficheurs externe seront utilisés pour cette application. La spécification du fonctionnement est la suivante :

Au démarrage, le programme doit remplir les conditions suivantes :

- Afficher la constante design standard ID du bus AXI lightweight HPS-to-FPGA au format hexadécimal dans la console de ARM-DS.
- Afficher la constante interface user ID du bus Avalon au format hexadécimal dans la console de ARM-DS.
- Vérifier que le statut de la carte Max10_leds est une configuration valide. Sinon afficher un message d'erreur dans la console ARM-DS et quitter le programme.
- Les 10 leds DE1-SoC sont éteintes.
- Les 4 afficheurs 7 segments affichent la valeur 0000.
- Les leds de la carte Max10_leds sont éteintes.
- Afficher dans le terminal UART un message texte qui décrit comment utiliser l'application de mesure du temps de réaction.

Ensuite pendant l'exécution du programme, à tout instant les actions suivantes doivent être respectées :

- Une pression sur KEY0 (par interruption) permet de stopper la mesure de temps de réaction lorsque la procédure a été démarrer (détails ci-après) et de faire un toggle de la Led9 de la carte DE1-SoC.
- Une pression sur KEY1 (par scrutation) permet de débuter la procédure pour la mesure du temps de réaction (détails ci-après).

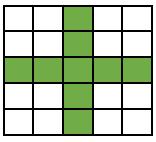
- L'état de SW3-0 permet de choisir la valeur afficher sur les 4 afficheurs 7 segments :
    - SW0=1 : affichage du meilleur temps de réaction en ms.
    - SW1=1 : affichage du plus mauvais temps de réaction en ms.
    - SW2=1 : affichage du nombre d'erreur lors de la mesure du temps de réaction (erreur = appuie sur KEY0 avant le symbole de début).
    - SW3=1 : affichage du nombre total de tentative de mesure du temps de réaction.
    - Sinon : affichage du dernier temps de réaction obtenue en ms.

Le début de la procédure pour la mesure du temps de réaction est la suivante :
- Afficher dans le terminal UART un message sur le début d'une mesure du temps de réaction et donner les actions à faire par l'utilisateur.
- Attendre un temps aléatoire de 1 à 4s et afficher le symbole d'attente sur le carré de leds de la carte Max10_leds.
- Afficher le symbole du début de mesure sur le carré de leds de la carte Max10_leds et démarrer le compteur pour la mesure du temps de réaction.

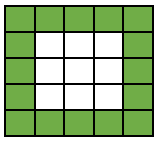
Ensuite lorsque la procédure a été stoppé par la pression sur KEY0 :
- Afficher le symbole de fin de mesure sur le carré de leds de la carte Max10_leds.
- Afficher le résultat de la mesure en ms sur les afficheurs 7 segments.
- Afficher dans la terminal UART un message avec le temps de réaction obtenue en ms, et les informations sur les scores obtenus jusqu'à présent : le meilleur, le plus mauvais, le nombre d'erreur, le nombre total de tentative.

Lors de la procédure, si un appuie anticipé sur KEY0, avant l'affichage du symbole de début, la tentative est comptabilisée comme une erreur et la procédure est stoppé. Un message d'information sur l'arrêt de la mesure et la mise à jours des scores sont affiché dans le terminal UART et le carré de leds de la carte Max10_leds sera éteint.

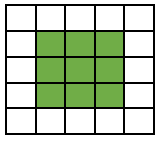
Voici les symboles à afficher sur le carré de leds de la carte Max10_leds :



Symbole d'attente



Symbole du début



Symbole de fin

À rendre

Ce laboratoire est évalué. Il y a un rapport à rédiger à l'issu de ce laboratoire contenant les explications sur les différentes étapes de la réalisation de votre système. Vous devez rendre une archive avec les sources du projet pour Quartus et le programme C. Utiliser le Makefile à la racine du projet pour générer votre archive à rendre en tapant « make zip » dans un terminal.

Les fichiers sont à rendre sur Cyberlearn à la date indiquée.

Travail demandé

1) Mise en œuvre de l'interface UART0 du HPS suivant les spécifications données précédemment et de la documentation du HPS (cyclone-v_hps_trm_5v4.pdf). Prévoir une fonction c qui permet d'envoyer une chaine de caractère dans l'UART. La gestion de l'envoie d'une chaine de caractère doit être optimiser pour garantir son bon fonctionnement et un débit le plus élevé possible.

2) Etablir un schéma bloc de votre composant avl_user_interface qui sera adapté au fur et à mesure du laboratoire. De même vous devez compléter le plan d'adressage de votre interface afin de répondre aux spécifications. Cette définition doit permettre d'avoir un accès facile aux différents périphériques.

3) Concevoir l'émetteur série asynchrone de 20 bits sans parité suivant les spécifications données précédemment. Simuler votre émetteur pour vérifier son bon fonctionnement, vous pouvez utiliser votre TB développé pendant le laboratoire VSE.
   Puis, concevoir et simuler une interface pour votre émetteur qui est implémenté dans le composant avl_user_interface. Penser à l'accès de bits de statut concernant votre émetteur.
   Tester votre émetteur avec la carte Max10_leds. Ecrire les fonctions C associer à l'utilisation de votre émetteur série.

4) Concevoir un gestionnaire d'interruption comme d'écrit dans les spécifications. Puis mettre en œuvre l'interruption qui provient de votre interface avl_user_interface et connecté sur la ligne FPGA_IRQ0. Ecrire les fonctions d'initialisation et de configuration des interruptions : config_GIC(), __cs3_isr_irq(), fpga_ISR(). La routine dédiée aux interruptions du FPGA sera fpga_ISR() et contiendra au moins le toggle de la Led9 de la DE1-SoC. Tester les interruptions sur la carte DE1-SoC.

5) Développer et interfacer un compteur avec une précision de 20ns dans votre interface avl_user_interface. Le compteur doit répondre aux spécifications décrites précédemment. Ecrire les fonctions C associés à l'utilisation du compteur. Tester son fonctionnement sur la carte DE1-SoC.

6) Proposer une solution afin de générer une vraie valeur aléatoire. Puis réaliser cette solution et tester son fonctionnement.

7) Développer l'application principale suivant les spécifications ci-dessus en utilisant les différentes parties développées précédemment.
   Tester votre application avec les cartes DE1-SoC et Max10_leds.

8) Faire valider votre programme par le professeur ou l'assistant.

9) Question : Pourrait on utiliser un module UART du HPS pour piloter la liaison série asynchrone 20 bits vers la carte Max10_leds ? Justifier votre réponse.

Annexe

I. Spécifications de la carte Max10-leds

La carte Max10-leds fourni un statut sur 2 bits permettant de confirmer que la configuration demandée est valide. Le signal de statut (con_80p_status_i) de 2 bits est défini de la manière suivante :

Statut	Description
"00"	Configuration non valide
"01"	Configuration valide
"1x"	Réservé

La ligne série permet d'envoyer un code de 20 bits permettant d'accéder à l'ensemble des leds disponibles sur la carte Max10-leds. Voici les différentes tables décrivant le codage du code de 20 bits.

Voici les codes définissants les différentes zones de leds accessibles :

code (3..0)	**Spécification de Data (15..0)**
0000	D15 not used, Leds secondes DS15...01
0001	D15 not used, Leds secondes DS30...16
0010	D15 not used, Leds secondes DS45...31
0011	D15 not used, Leds secondes DS60...46
0100	Ligne leds DL15....0
0101	Ligne leds DL31...16
0110	D15 not used, Carré de leds DM35-31/25-21/15-11
0111	D15-10 not used, Carré de leds DM55-51/45-41
1000	Leds heures DH04-01 Int(3..0), 4 bits d'intensité par led
1001	Leds heures DH08-05 Int(3..0), 4 bits d'intensité par led
1010	Leds heures DH12-09 Int(3..0), 4 bits d'intensité par led
1011	réservé
…	…
1111	réservé

Contrôle de l'intensité pour les leds-heures :

Les Leds des heures sont des leds RGB. Chaque Led-heure dispose de 3 leds de couleur, soit : R=rouge, G=vert et B=bleu. La commande de ces leds est réalisée avec un PWM afin de pouvoir varier la luminosité de chaque couleur. Il est ainsi possible d'obtenir différentes couleurs. Chaque Led-heure est dès lors pilotée avec une commande de 4 bits nommée : Int(3..0)

Le codage d'intensité est décomposé en deux parties, soit :
- Int(3..2) : choix de la couleur
- Int(1..0) : choix de l'intensité

Tableau pour le choix de la couleur pour les Leds-heures :

Int(3..2)	Color	Description
"00"	red	couleur rouge
"01"	green	couleur verte
"10"	blue	couleur bleue
"11"	white	couleur blanche, soit les 3 leds RGB

Tableau pour le choix de l'intensité pour les Leds-heures :

Int(1..0)	Intensité	Intensité sélectionnée
"00"	off	éteint
"01"	weak	faible
"10"	medium	moyenne
"11"	strong	forte