HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Prof. Yann Thoma

Laboratoire CSF
FIFO de taille générique
semestre printemps 2016 - 2017

Objectifs pédagogiques

Ce laboratoire a pour but de réaliser une mémoire de type FIFO, en exploitant une mémoire et des paramètres génériques.

Cahier des charges

Il vous est demandé de réaliser une mémoire de type FIFO (First In First Out) de taille générique. Pour ce faire, n'hésitez pas à séparer votre système en sous-modules et à exploiter un composant mémoire (à vous de le développer). Pour ce composant mémoire, faites que la lecture de cette mémoire soit synchrone, c'est-à-dire que la donnée lue soit disponible un cycle d'horloge après que l'adresse soit valide.
Le nombre de mots mémorisés par la FIFO, ainsi que la taille de ces mots, est générique. Pour ce faire, deux paramètres génériques seront fournis :

- `FIFOSIZE` : Définit le nombre d'éléments que la FIFO doit contenir
- `DATASIZE` : Définit la taille des données

Les entrées/sorties sont :

Nom	Taille	Direction	Description
`clk_i`	1	in	Horloge du système
`rst_i`	1	in	Reset du système
`wr_i`	1	in	Indique une écriture
`rd_i`	1	in	Indique une lecture
`wr_data_i`	`DATASIZE`	in	Donnée d'écriture
`rd_data_o`	`DATASIZE`	out	Donnée de lecture
`full_o`	1	out	Indique que la FIFO est pleine
`empty_o`	1	out	Indique que la FIFO est vide

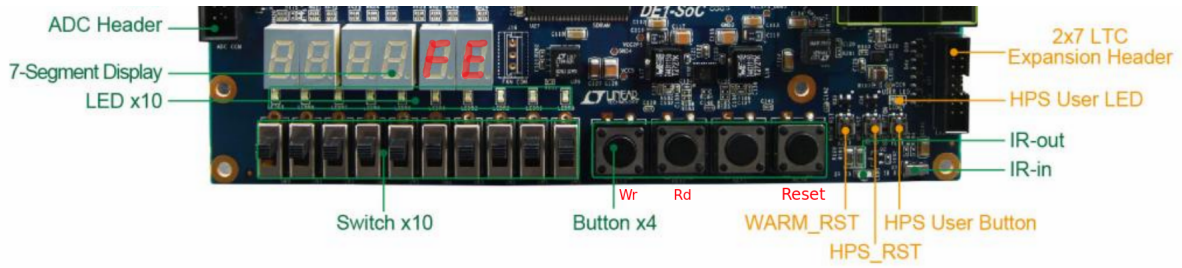
La FIFO doit indiquer si elle est vide, pleine, ou partiellement remplie, et ce en activant les sorties `empty_o` et `full_o`.
Si une écriture est faite alors que la FIFO est pleine, la donnée est perdue. De même lire un élément d'une FIFO vide n'a pas de sens.
La donnée en sortie doit être disponible dès que la FIFO n'est plus vide, et l'activation du signal de lecture modifie cette sortie au prochain coup d'horloge.
Si vous en avez besoin, sachez que vous pouvez déclarer un entier défini dans un certain range de valeurs :

```
signal counter_s  : integer range 0 to FIFOSIZE-1;
```

Travail à réaliser

1. Proposer un schéma du système.

2. Pour les entités à réaliser, décrire leur architecture en VHDL. Vous disposez déjà de l'entité de la FIFO, dans `src_vhdl/fifo.vhd`.
3. Simuler votre système à l'aide des fichiers fournis.
   — Dans le répertoire `sim`, tapez `do ../scripts/sim.do` depuis la console Questasim.
   — Vous pouvez essayer plusieurs cas en ajoutant la taille de la FIFO suivi de la largeur des mots, à la manière suivante (exemple pour une profondeur de 16 et des mots de 32 bits) : `do ../scripts/sim.do 16 32`
4. Intégrer la FIFO sur la carte DE1-SoC. Pour ce faire créez et compilez un nouveau projet Quartus nommé **de1_top** (référez vous au laboratoire précédent pour plus de détails). Les fichiers fournis sont :
   — `src_vhdl/de1_top.vhd` - Le code du top.
   La profondeur de la FIFO peut être changée dans ce fichier.
   — `src_vhdl/hex_7_seg.vhd` - Le décodeur sept segments.
   — `synth/de1_top_pin_assignments.csv` - L'assignation des pins.
   Les flags empty et full sont affichés à l'aide de deux afficheurs sept segments. L'entrée, un mot de 8 bits, se programme grâce aux switches 0 à 7, la sortie est affichée sur les leds 0 à 7. Le bouton 0 est le reset du système, Le bouton 2 correspond à un read et le bouton 3 correspond à un write. Comme illustré ci-dessous.
5. Définir et effectuer une série de tests pertinents pour montrer le bon fonctionnement de la FIFO. (Tester l'usage normal et les cas limites).



Travail à rendre

Une fois achevé[1], chaque groupe rendra un mini rapport contenant une description de ses choix ainsi que les sources VHDL, sous format électronique.
Faite également valider votre système sur la DE1-SoC par le professeur ou l'assistant.

Notes

Si le serveur JTAG ne voit pas votre carte (et que tout est bien branché, allumé), relancez le avec les commandes suivantes :

```
$ sudo killall -9 jtagd
$ sudo /opt/EDA/quartus16/quartus/bin/jtagd
```

Et vérifiez avec :

```
$ jtagconfig
```

1. le travail, pas l'exécutant