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

Prof. Yann Thoma

# Laboratoire VSN
semestre de printemps 2020 - 2021

# Exercices de vérification
# Quelques réflexions

## VHDL

Dans le contexte d'un banc de test VHDL, nous nous intéressons à vérifier la validité de certaines assertions. Pour chacune de ces assertions, écrire le code VHDL permettant de la vérifier. Il peut s'agit d'une assertion concurrente ou d'un processus.

1. Lorsque `A` passe de '0' à '1', `B` doit ensuite rester stable pendant 100 ns.
2. `A` ne doit jamais avoir la même valeur que `B`.
3. Lorsque, sur un flanc d'horloge, `A` vaut '1', alors `B` doit valoir '1' pendant les deux cycles d'horloge suivant. Utilisez le signal `clk` comme horloge.
4. Lorsque le signal `wr` passe à '0' au temps $t$, le signal **`data`** doit avoir été stable entre le temps $t - ta$ et $t - tf$. ($ta > tf > 0$)

## Contraintes SystemVerilog

Soit la classe SystemVerilog suivante :

```
class Test;
    rand logic wr;
    rand logic rd;
    rand logic cs;
    rand logic[1:0] type;
    rand logic[7:0] address;
    rand logic[31:0] data;
    rand logic parity;

    // Votre code
endclass : Test
```

Nous désirons générer aléatoirement des objets de cette classe selon les contraintes suivantes :

1. Si `cs` vaut 0, alors `address` doit aussi valoir 0
2. Si `wr` vaut 1, alors `rd` doit valoir 0
3. Si `rd` vaut 1, alors `wr` doit valoir 0
4. Si **`type`** vaut 0, alors `address`$< 16$
5. Si **`type`** vaut 1, alors $16 \leq$`address`$< 128$
6. Si **`type`** vaut 2, alors $128 \leq$`address`
7. `parity` doit valoir 0 si la somme des bits à 1 de **`data`** est pair, et doit valoir 1 sinon

Ecrivez le code nécessaire à la génération aléatoire afin de respecter ces contraintes. Votre code doit permettre une simulation la plus efficace possible en termes de temps d'exécution. Partez du principe que votre code sera placé à l'endroit du commentaire `// votre code`.

## Groupes de couverture SystemVerilog

Nous désirons mettre en place un groupe de couverture pour la classe suivante :

```
class Test;
    rand logic wr;
    rand logic rd;
    rand logic cs;
    rand logic[1:0] type;
    rand logic[7:0] address;
    rand logic[31:0] data;
    rand logic parity;
endclass : Test
```

Le groupe de couverture doit permettre de détecter les combinaisons suivantes :

1. L'ensemble des types doivent avoir été observés en lecture (rd à 1) et en écriture (wr à 1)
2. Toutes les adresses entre 0 et 3 et entre 252 et 255 doivent avoir été observées

Ecrivez le groupe de couverture correspondant.

## Assertions SystemVerilog

Ecrivez les assertions suivantes en SystemVerilog, pour un système ayant une horloge `clk` et un reset `rst` :

1. Si une requête arrive (`req`=1), alors un acknowledge (`ack`=1) doit être observé au plus tard 4 cycles après
2. Si `A` a été à 1 pendant 3 cycles, et que `B` a été à 1 pendant les deux dernier cycles où `A` était à 1, alors `C` doit être à 0 au cycle suivant et passer de 0 à 1 au plus tard 4 cycles après