Modèle VHDL de l'étage DECOD partie 2 semaine 7

Objectif(s)

- ✯ Faire un premier modèle comportemental de l'étage `DECOD`.
- ✯ Tester ce modèle.

Exercice(s)

Exercice 1 – Machine à état de DECOD

Le contrôle du flot d'instruction par DECOD est plus complexe que pour EXEC. DECOD communique avec ses deux voisins IFETCH et EXEC, suivant une unique séquence qui suivant l'instruction à exécuter pourra être incomplète :

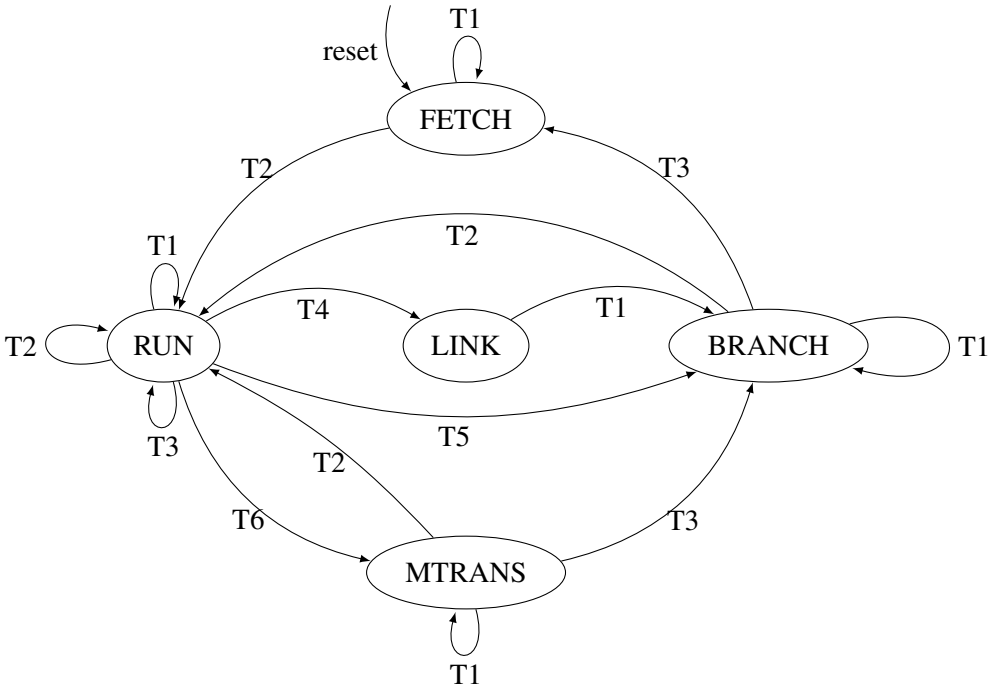
1. Envoyer l'adresse d'une instruction dans la fifo `dec2if` ;
2. Lire l'instruction chargée par `IFETCH` et stockée dans la fifo `if2dec` ;
3. Décoder l'instruction chargée ;
4. Écrire le résultat du décodage (opérandes, opérations à réaliser ...) dans la fifo `dec2exec` à destination de `EXEC`.

La fonction principale de cette machine à état consiste à contrôler la lecture et l'écriture dans les trois fifos. Trois signaux vont être dédiés à ces actions dans l'ordre de la séquence précédemment exposée : `dec2if_push`, `if2dec_pop` (correspond au port `dec_pop`) et `dec2exe_push`. Ces signaux sont affectés par la machine à état.

De par leur nature et sans considérer l'état de remplissage des différentes fifos certaines instructions vont être traitées en plusieurs cycles : les branchements et les instructions de transfert multiples. Généralement le simple décodage combinatoire des instructions va être suffisant pour générer les opérandes et commandes à destination de `EXEC`, ce n'est pas le cas pour les instructions nécessitant plusieurs cycles.

Dans le cas des appels de fonction (*branch and link*) mon implémentation utilise un premier cycle pour la sauvegarde de l'adresse de retour. Au moment du décodage d'une instruction *bl*, PC a déjà progressé de 2 instructions ($PC+8$) et donc pour calculer l'adresse de retour 4 est envoyé comme second opérande, PC comme premier opérande et `EXEC` reçoit la commande correspondant à effectuer une soustraction. Tout cela est commandé par le signal `blink` généré par la machine à état.

Les signaux `mtrans_shift` et `mtrans_loop_adr` sont générés par la machine à état et utiles dans le cas des transferts multiples.



Nous allons détailler pour chaque état à quelle configuration des entrées correspond une transition :

FETCH : État de démarage (après reset) `DECOD` envoie une première adresse vers `IFETCH`.

- **T1** : La fifo vers `IFETCH` est pleine.
- **T2** : Une première valeur de PC a peu être écrite dans `dec2if`.

RUN : État général de fonctionnement pour toutes les instructions traitées par [DECOD] dans le cycle.

- **T1** : `if2dec` vide, `dec2exe` pleine ou prédicat invalide, une nouvelle valeur de PC est envoyée vers `IFETCH` si la fifo `dec2if` n'est pas pleine.
- **T2** : Le prédicat est faux, l'instruction doit être jetée.
- **T3** : Le prédicat est vrai, l'instruction est exécutée.
- **T4** : L'instruction est un appel de fonction.
- **T5** : L'instruction est un branchement.
- **T6** : L'instruction est un transfert multiple.

LINK :

- **T1** : Transition prise systématiquement, commande d'`EXEC` pour calculer une nouvelle valeur de PC.

BRANCH : Purge de l'instruction suivant un branchement pris.

- **T1** : `if2dec` est vide.
- **T2** : `if2dec` n'est pas vide.

MTRANS : traitement des instructions de transfert multiple ...

Cette machine à état étant de type *Mealy* la valeur des sorties va dépendre des transitions. À vous de déterminer la valeur des sorties pour chacune des transitions. Vous allez dans un premier temps décrire le modèle comportemental du banc de registre `REG` qui va constituer un des composants de `DECOD`.

Question 1

Complétez le modèle `VHDL` du bloc `DECOD`.

Question 2

Écrire un fichier *test_bench* permettant de valider votre modèle.

