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

Prof. Yann Thoma

Mémoire cache
Implémentation sur FPGA
semestre printemps 2015 - 2016

Introduction

Afin d'améliorer les performances d'un système nécessitant beaucoup de mémoire nous sommes intéressés à réaliser une mémoire cache interne à la FPGA de manière à accélérer l'accès aux données présents dans une mémoire externe (DDR).
La mémoire cache que nous allons réaliser devra être interfacée avec un bus simple. Elle sera un slave vue par le système FPGA, et un master pour le contrôleur mémoire. Le code qui vous est fourni comprend un paquetage contenant les types de données présentes sur les ports du composant. Vous avez également à disposition l'entité de la mémoire cache et un banc de test très minimaliste.
Vous devrez réaliser une mémoire à correspondance directe et blocs de 16 mots.

Interface

Le composant à réaliser a des ports présentés sous forme de `record` qui sont déclarés dans le paquetage `cmf_pkg` :

```
type agent_to_cache_t is
record
    wr          : std_logic;
    rd          : std_logic;
    addr        : std_logic_vector;
    data        : std_logic_vector;
end record agent_to_cache_t;

type cache_to_agent_t is
record
    dready      : std_logic;
    data        : std_logic_vector;
    busy        : std_logic;
end record cache_to_agent_t;

type cache_to_mem_t is
record
    wr          : std_logic;
    rd          : std_logic;
    burst_range : std_logic_vector;
    burst       : std_logic;
    addr        : std_logic_vector;
    data        : std_logic_vector;
end record cache_to_mem_t;

type mem_to_cache_t is
record
    rxd         : std_logic; --rx done
    dready      : std_logic; --data ready
    data        : std_logic_vector;
end record mem_to_cache_t;
```

Les deux premiers représentent les interactions avec la partie interne de la FPGA et les deux derniers sont voués à interagir avec un contrôleur mémoire pour accès externe.
Lors d'une écriture depuis l'agent, il faut activer le signal `wr` en même temps que l'adresse et la donnée sont présentés. Ces trois signaux doivent rester activés tant que `busy` est actif. Ce dernier signal permet à la mémoire cache d'indiquer qu'elle n'est pas prête à être accédée.

Pour une lecture, il faut activer le signal `rd` en plaçant l'adresse demandée. La donnée sera alors présente et légale lorsque le signal `dready` sera activé. De ce fait la mémoire cache pourra prendre plusieurs cycles d'horloge pour fournir la donnée. Il est toutefois clair que ce temps devrait être le plus court possible.
Du côté du contrôleur mémoire externe, la mémoire cache pourra faire des accès en écriture en activant le signal `wr` et en plaçant l'adresse et la donnée correspondantes. Des accès en burst sont également possibles via l'activation du signal `burst` et en indiquant dans `burst_range` le nombre de mots à transmettre. L'écriture est valide si le signal `busy` n'est pas activé par le contrôleur mémoire.
La lecture est déclenchée en activant `rd` et peut se faire en burst de la même manière que les écritures. Dans ce cas, le signal `burst` doit être activé et la première adresse doit être fournie. Ensuite les données sont fournies dans l'ordre croissant d'adresse par le contrôleur mémoire. A chaque fois qu'une donnée est prête le signal `dready` doit être activé. Là encore le signal `busy` permet au contrôleur d'indiquer qu'il n'est pas prêt à supporter une lecture.
A noter également que le signal busy n'a pas besoin d'être activé pendant un burst en lecture ou écriture, c'est à la mémoire cache de savoir qu'elle est déjà en train de faire un accès.
L'adressage de la mémoire cache se fait au mot, un mot étant de taille `DATA_SIZE`.

Politique d'écriture

Vous devrez implémenter une politique d'écrasement de type write-back, qui est plus performante. Un bit `dirty` sera donc nécessaire pour chaque ligne et chaque ensemble afin de savoir si la donnée est en adéquation avec la mémoire RAM. L'accès à la mémoire externe se fera via des bursts de 16 mots.

Banc de test

Un banc de test sera nécessaire à votre mémoire. Pour ce faire il faudra émuler le contrôleur mémoire sur lequel votre bloc viendra s'interfacer. De l'autre côté il faudra imaginer des scénarios d'accès lecture/écriture permettant de vérifier le comportement de votre mémoire.

Intégration

Une fois votre mémoire validée en simulation, vous pourrez l'intégrer sur une carte REPTAR. Un programme de test vous sera fourni et permettra de vérifier le fonctionnement de votre mémoire.

Conseils

Travaillez par groupe de 2.
Réfléchissez ensemble à une architecture de votre mémoire. Répartissez ensuite bien les tâches. Il y aura vraisemblablement plusieurs blocs à réaliser et des bancs de tests.
N'oubliez pas de faire des **synthèses régulièrement** afin de vérifier un composant synthétisable !