Compte Rendu Partiel : VLSI ARM Project

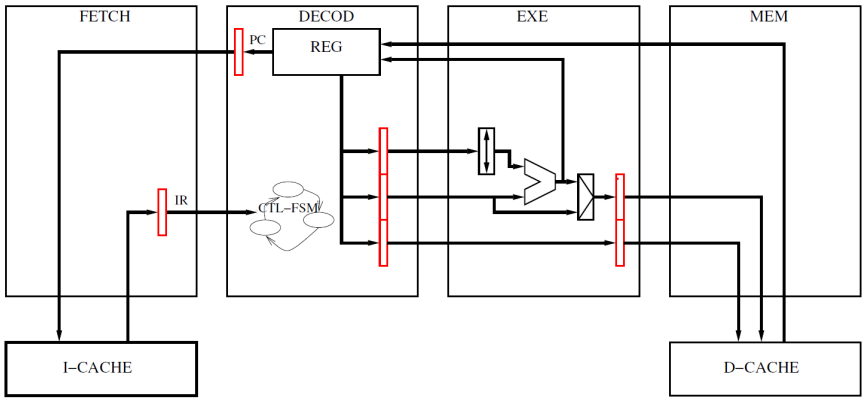
Tao Schreiner
Louis Geoffroy Pitailler
Timothée Le Berre

December 2021

1 Introduction

Au cours de ce projet, on cherche à décrire le cœur d'un processeur basé sur une architecture ARM. Pour ce faire on utilise le langage de description matérielle VHDL.
Le CPU que l'on modélise est un processeur pipeliné sur 4 étages :
- Fetch : récupère l'instruction en mémoire et l'envoie à l'étage decode,
- Decode : récupère l'instruction chargé par Fetch et procède à son décodage afin de sélectionner les opérandes, registres et calculs nécessaire à son exécution,
- Exe : effectue les opérations arithmétiques de bases,
- Mem : effectue des accès mémoires si l'instruction exécutée en nécessite.

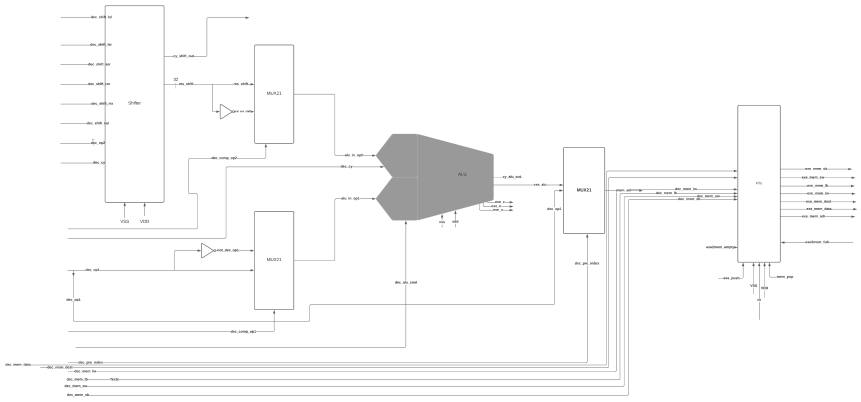
Voici un schéma simplifié du pipeline que l'on va modéliser :



2 EXE

2.1 Introduction :

La première étape de notre modélisation a été l'étage EXE, c'est en effet le plus simple. Il est constitué de 2 parties primordiales : l'ALU (*arithemic logic unit*) et le shifter. Voici un schéma de son architecture :



2.2 ALU : Arithmetic logic Unit

La modélisation est assez simple, on envoie un signal de commande sur 2 bits à l'ALU et on sélectionne ainsi l'opération à faire : and, or, xor, somme.
Dans le but de pouvoir faire des soustractions, on ajoute des inverseurs dans l'architecture de l'ALU, un sur la sortie du Shifter et un sur l'opérande 1. Cela va nous permettre de choisir si l'on veut le complément à deux de l'un des opérandes dans le but de faire une soustraction.

2.3 Shifter

Le shifter est commandé par 5 bits indiquant le type de shift que l'on fait. Une fois que le type de shift est sélectionné on récupère la valeur de shift sur 5 bits et on fait une disjonction des cas. Si l'un des bits de la valeur de shift est à 1 cela signifie que l'on fait un décalage de cette valeur.
Nous avons donc créé des entités effectuant des shits de 1,2,4,8 et 16 que l'on map en fonction de la valeur de shift que l'on veut.
De même pour la rotation, on effectue une concaténation de sorte à obtenir le résultat escompté.

3 DECOD :

L'étage DECOD est composé de deux parties principales : d'un côté, le banc de registre, et de l'autre, le décodage des instructions et le contrôle général du processeur.

3.1 REG : Registre Bank

Sur cette architecture ARM, le banc de registre fait partie de l'étage DECOD.

Il contient 16 registres, ainsi que les flags C, Z, N et V. Le banc de registres peut gérer deux écritures (dont une prioritaire sur l'autre en cas d'adresse identique), 3 lectures 32-bits et une lecture 5-bits servant uniquement pour la lecture du registre codant la valeur d'un shift.
À chaque registre est associé un bit de validité. Un registre est invalidé par DECOD quand une instruction qui écrit dedans est lancée, et redevient valide quand le résultat de l'instruction est écrit dans le registre.
Les 4 flags partagent un bit de validité.
En pratique, l'étage REG reçoit les registres à invalider en entrée, et chaque registre redevient valide dès qu'une écriture est effectuée dessus. Les lectures sont accompagnées du bit de validité du registre correspondant.
En VHDL, notre banc de registre est représenté par un tableau de std_logic_vector, et les bits de validité par un tableau de bits. Lors d'une lecture ou d'une écriture, les adresses sont converties en entiers pour être utilisées comme index pour les tableaux.
L'écriture dans les registres se fait à chaque front montant de l'horloge, en donnant bien priorité au premier registre écrit s'il y a un conflit d'adresse.

REG s'occupe aussi de gérer PC : si un flag de contrôle vaut 1, PC est incrémenté de 4, et sinon il garde sa valeur.

4 Conclusion :

Il reste à faire toute la partie "décodage des instructions", ainsi que le contrôle du processeur, de l'étage DECOD. C'est une partie assez grosse, avec beaucoup de branchements, ainsi qu'une machine à états.
Il restera ensuite le branchement final des différents étages. Nous avons réalisé une FIFO générique (donc d'une profondeur arbitraire), et même les GENERIC ne sont pas synthétisables, il nous suffira de les remplacer par des entiers littéraux pour avoir une FIFO de longueur 4 ou 8 par exemple.
Pour pouvoir synthétiser notre processeur, il faudra sans doute refaire une partie des étages (par exemple, les conversions en entiers dans l'ALU et dans REG ne sont probablement pas synthétisables).