Université de Cergy-Pontoise

Rapport

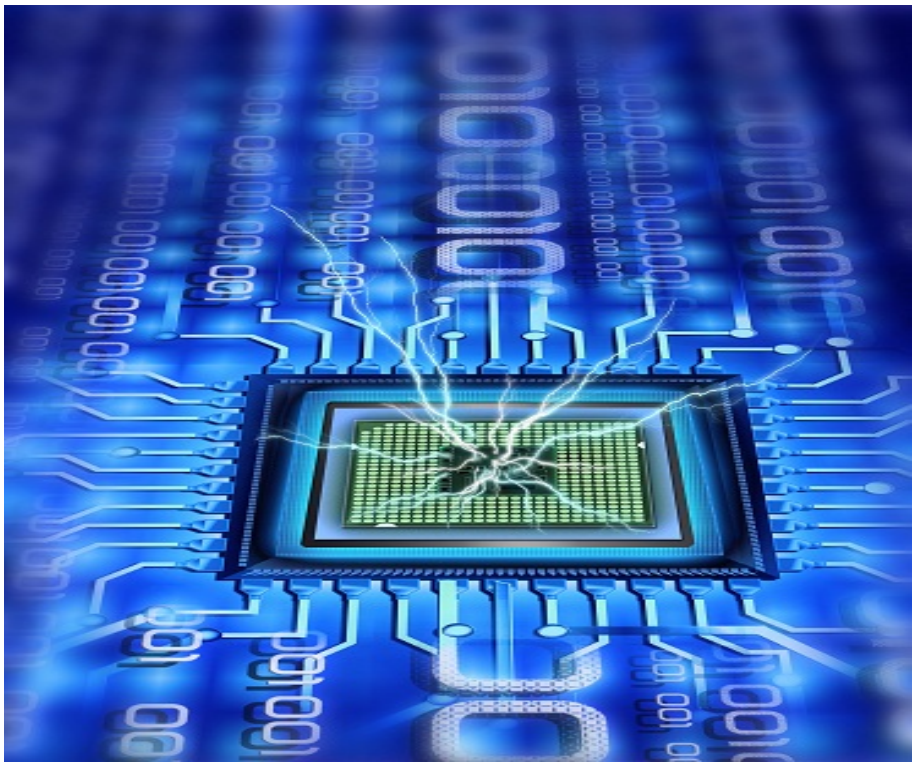
pour le projet d'architecture des ordinateurs
Licence 3 informatique

sur le sujet

Conception d'un processeur 16 bits en VHDL

réalisé par

HACHOUD Rassem et METIDJI Fares



2018/2019

Table des matières

Remerciements

Tout d'abord, il apparaît opportun d'adresser nos remerciements à tout ceux qui nous ont aidé pour la réalisation de ce projet, notamment notre encadrant Monsieur Jordane Lorandel, auprès duquel nous avons pu bénéficier d'un grand soutien.

1 Introduction

Dans le cadre de nos études en Licence 3 informatique, nous devons réaliser un projet pour le module d'architecture des ordinateurs. Il consiste à concevoir un processeur en VHDL, et de l'implémenter sur une carte FPGA. Ce processeur permet d'effectuer des opérations basiques (addition, soustraction ...) sur des mots de 16 bits.

2 Spécification

2.1 Architecture générale :

Le processeur réalisé est composé principalement de 4 blocs : une ALU pour effectuer les différentes opérations, un banc de 8 registres pour le stockage des données, un multiplexeur pour la sélection des registres, et enfin, une unité de contrôle qui pilote les différents composants.

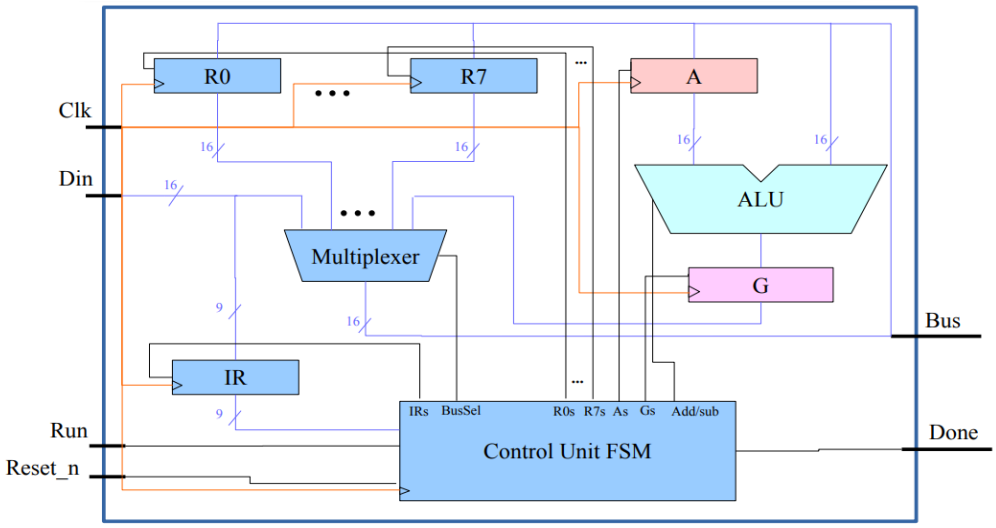


FIGURE 1 – Vue d'ensemble de processeur

2.2 Principe de fonctionnement

Tout d'abord, les données entrantes par le bus de 16 bits peuvent être rangées dans l'un des registres en sélectionnant le chemin approprié dans le multiplexeur. Pour effectuer une opération dans l'ALU, l'unité de contrôle positionne la première opérande sur le bus à l'aide de multiplexeur. Cette dernière sera ensuite placée dans le registre A. Puis la seconde opérande sera placée sur le bus au cycle suivant de l'horloge. L'opération est alors réalisée entre le contenu de registre A et le contenu du bus. Enfin, le résultat est rangé dans le registre G et peut être envoyé vers la sortie de processeur ou bien stocké dans l'un des registres.

2.3 Jeux d'instruction :

Les instructions chargées dans le registre RI, sont encodées sur 9 bits suivant 2 formats distincts :

— **Format registre :** [XXX YYY ZZZ]
  XXX = code opération
  YYY = Numéro du registre RX
  ZZZ = le code le registre RY

— **Format immédiat :** [XXX YYY 000]
  XXX = code opération
  YYY = Numéro du registre RX

Mnémonique	Code	Opération
ADD	000	Addition
SUB	001	Soustraction
MULT	010	Multiplication
AND	011	ET logique
OR	100	OU logique
NOT	101	NON logique
MV	110	Copie de contenu de Rx dans Ry
MVI	111	Copie de DIN dans Rx

3 Réalisation

3.1 L'unité arithmétique et logique

L'ALU (Arithmetical and Logical Unit) est l'unité en charge de faire les calculs et les opérations logiques. Elle prend comme entré 2 mots de 16 bits et effectue l'une des opérations disponibles (voir le tableau précédant). Le choix de l'opération se fait par une entrée de sélection de 3bits.

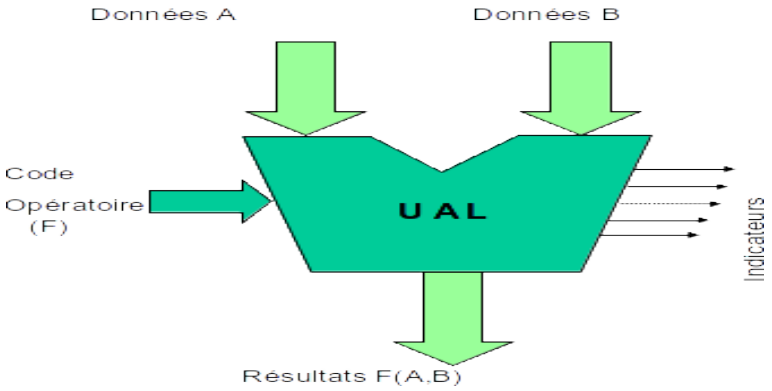


FIGURE 2 – Unité arithmétique et logique

Les instructions qui utilisent l'ALU nécessitent plusieurs cycles pour s'exécuter à cause des multiples transferts à opérer sur le bus. C'est l'unité de contrôle qui dirige ces opérations de transfert à chaque cycle.

3.2 Banc de registres et multiplexeur

Le banc de registres est l'unité du CPU qui va stocker provisoirement les données. Il est composé de 8 registres (R) de 16 bits. Le multiplexeur est commandé par l'unité de contrôle pour sélectionner l'un de ces registres, soit pour lire une données ou bien pour la stocker. Le fonctionnement de ce multiplexeur est décrit dans le tableau suivant :

Sélection	Registre
0000	R0
0001	R1
0010	R2
0011	R3
0100	R4
0101	R5
0110	R6
0111	R7
1000	G
1001	Din

3.3 Registres particuliers

En plus des 8 registres 16 bits constituant le banc de registres, 3 autres registres sont nécessaires pour le bon fonctionnement de ce processeur.

Registre I : Il sert à stocker les instructions venant de bus Din. À chaque fois qu'il est activé, cela revient à demander l'instruction suivante.

Registre A : Lorsque l'on veut effectuer une opération sur deux opérandes, l'architecture de ce processeur pose problème, puisque on ne dispose que d'un seul bus véhiculant les données. C'est pour cela, un registre A (16 bits) est mis en place sur une entrée de l'ALU, de manière qu'elle puisse effectuer l'opération entre le contenu de ce registre et celui de bus principal. Avant chaque opération à 2 opérandes, utilisant l'ALU, il faut donc d'abord stocker la première opérande dans le registre A, envoyer la deuxième opérande dans le bus principal et enfin activer l'ALU en lui envoyant le signal correspondant à l'opération en question.

Registre G : Afin de ne pas perdre le résultat d'une opération utilisant l'ALU, nous devons l'écrire dans l'un des registres R. Or, à la fin d'une instruction, le contenu du bus principal disparaît. Nous sommes donc obligés d'ajouter un registre (G) juste après la sortie de l'ALU, dans lequel le résultat sera systématiquement stocké. L'unité de contrôle peut choisir de faire passer le contenu du registre G dans le bus principal. On peut donc sauvegarder le résultat d'une opération dans l'un des registres R. Lorsque le bit correspondant à G vaut 1 le contenu du bus sortant de l'ALU y est stocké.

3.4 L'unité de contrôle

L'unité de contrôle est une machine à états finis (FSM) qui pilôte les différents composants de processeur. Elle envoi les bons signaux aux bons composants en fonction de code opératoire. Elle choisie dans un premier temps, la donnée qui circulera sur le bus principal parmi 3 possibilités : le signal d'entrée Din, le contenu d'un registre R, ou le résultat de l'ALU. Ceci se fait à l'aide de multiplexeur. Ensuite elle envoi les différents signaux permettant l'exécution de l'instruction.

3.4.1 Composition et interface

Puisque l'unité de contrôle est une FSM, elle est donc composée de 3 parties : un registre d'états pour stocker l'état actuel de la FSM, une fonction de transition calculant l'état futur en fonction de l'état présent et des entrées, et un fonction de génération calculant les sorties en fonction de l'état actuel et des entrées. L'interface externe de l'unité de contrôle est composée principalement de signaux suivants :

- IR_s = Signal de sortie sur 1 bit activant l'écriture dans le registre d'instruction
- A_s = Signal de sortie sur 1 bit activant l'écriture dans le registre A , premier termes de l'ALU.
- G_s = Signal de sortie sur 1 bit activant l'écriture dans le registre G , résultat de l'ALU.
- RI_s = Signal de sortie sur 8 bits activant l'écriture dans un des registre du banc de registre.
- Bus_sel = Signal de sortie sur 4 bits activant l'entrée correspondante dans le multiplexeur.
- ALU_sel = Signal de sortie sur 3 bits sélectionnant l'opération sur l'ALU.
- Done = Signal de sortie sur 1 bit signalant la fin de l'exécution de l'instruction courante.
- Ir = Signal entrant sur 9 bits, contenant l'instruction à exécuter.
- Run = Signal entrant activé par le programeur, indiquant la présence d'une nouvelle donnée sur le bus d'entrée Din
- Reset = Signal entrant sur 1 bit permettant la réinitialisation l'état de la FSM

3.4.2 La machine à états

La FSM est composée de 7 états :

— **IR :** Lecture de registre d'instructions.
— **DONE :** Fin de l'exécution de l'instruction courante.
— **MV :** Exécution d'une instruction Mouve
— **MVI :** Exécution d'une instruction Mouve Immédiat.
— **ALU0 :** Exécution de premier cycle d'une instruction sur l'ALU.
— **ALU1 :** Exécution de deuxième cycle d'une instruction sur l'ALU.
— **ALU0 :** Exécution de dernier cycle d'une instruction sur l'ALU.

Les différentes transitions sont expliquées dans la tableau suivant :

État présent	**État future**	**condition de transition**
any	IR	reset = 1
IR	MV	(CODOP = 110) et reset = 0
IR	MVI	(CODOP = 111) et reset = 0
IR	ALU0	CODOP != 110 et CODOP != 111 et reset = 0
MV	DONE	reset = 0
MVI	DONE	reset = 0
ALU0	ALU1	reset = 0
ALU1	ALU2	reset = 0
ALU2	DONE	reset = 0
DONE	IR	run

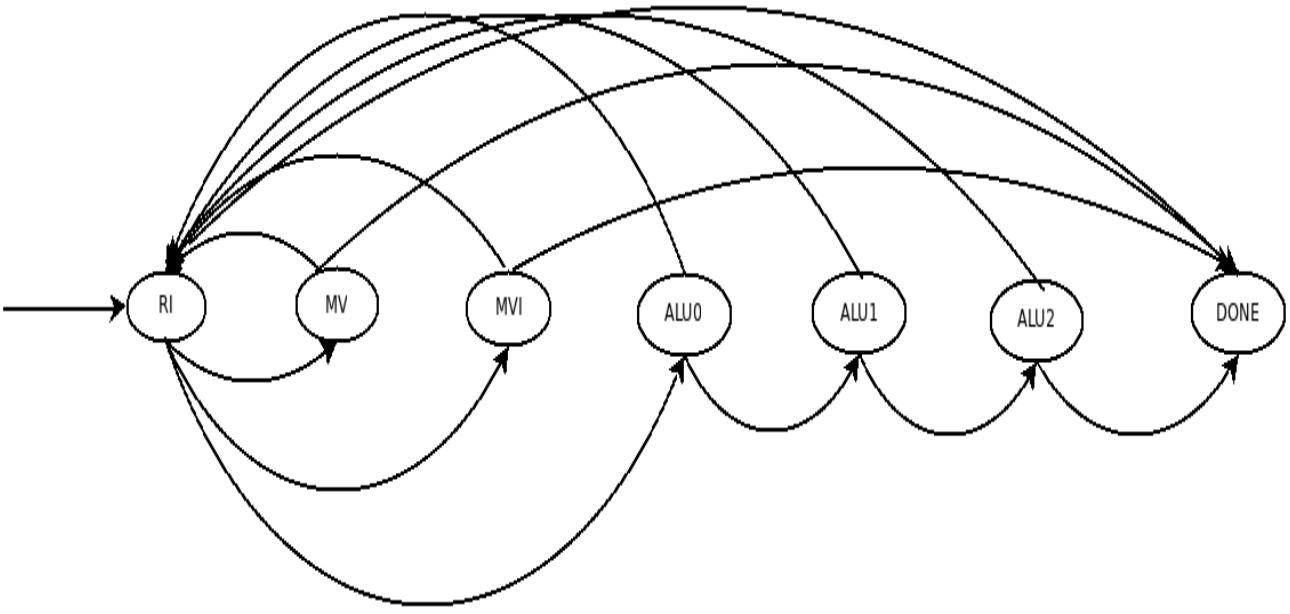


FIGURE 3 – Graphe d'états de la FSM

4 Déroulement du projet

4.1 Répartition des tâches

HACHOUD Rassem	**METIDJI Fares**
- Unité arithmétique et logique - Unité de contrôle	- Registres et multiplexeur - Unité de contrôle

4.2 Problèmes rencontrés

— Difficultés pour la Réalisation de l'unité de contrôle (FSM)
— Difficultés de tests sur la carte FPGA

5 Conclusion

Au final nous avons donc réalisé un processeur en VHDL et nous l'avons implanté sur la carte FPGA. Ce processeur permet d'effectuer des opérations basiques (addition, soustraction ...) sur des mots de 16 bits.

Ce projet nous a été très intéressant puisqu'il nous a permis de mettre en œuvre les connaissances que nous avons acquises tout au long du module d'architecture des ordinateurs, notamment la maitrise de langage de description matériel VHDL.