HEIG VD | REDS Institut Reconfigurable and Embedded Digital Systems

Prof. Romuald Mosqueron

Prof. Olivier Auberson

Laboratoire d'architecture des ordinateurs

semestre printemps 2024 - 2025

Microarchitecture FORWARDING Analyse

Informations générales

Le rendu pour ce laboratoire sera **par groupe de deux**, chaque groupe devra rendre son travail sur Cyberlearn.

Le rendu du laboratoire sera **noté**. Tout retard impactera la note obtenue.

> ⚠ **N'oubliez pas de sauvegarder et d'archiver votre projet à chaque séance de laboratoire**

NOTE : Nous vous rappelons que si vous utilisez les machines de laboratoire situées au niveau A, il ne faut pas considérer les données qui sont dessus comme sauvegardées. Si les machines ont un problème nous les remettons dans leur état d'origine et toutes les donnés présentes sont effacées.

Objectif du laboratoire

L'objectif est de comprendre le fonctionnement d'un processeur dit pipeliné avec forwarding. Vous recevrez le processeur complet en version pipeliné avec forwarding et des programmes à exécuter. Il vous sera demandé d'analyser le fonctionnement du processeur avec le pipeline et forwarding.

Vous devez exécuter les différents programmes, tracer les chronogrammes correspondant et répondre aux questions.

Outils

Pour ce laboratoire, vous devez utiliser les outils disponibles sur les machines de laboratoire (A07/A09) ou votre ordinateur personnel avec la docker fournie par le REDS.

Fichiers

Vous devez télécharger à partir du site Cyberlearn un ".zip" contenant un répertoire «workspace» où vous trouverez :

— **processeur_ARO2_hazard_detection_student.circ** : Le fichier de travail Logisim
— **main.S** : fichier source du code assembleur pour tester le circuit
— **Makefile** : fichier contenant les directives d'assemblage
— **0X_main.S** : Les différents programmes à tester.

Attention : Vous ne devez pas utiliser les fichiers du précédent laboratoire. Créez un nouveau répertoire.

⚠ **Le fichier Makefile ne doit pas être modifié !**

⚠ **Respectez l'architecture hiérarchique présentée dans le cours.**

Rendu

Pour ce laboratoire, vous devez rendre un fichier au format *.pdf* contenant :

— les réponses aux questions posées,
— les chronogrammes **annotés et commentés**.

1 Analyse et test du processeur avec forwarding

1.1 Analyse du processeur

Le processeur qui vous est fourni est pipeliné avec forwarding. Il a été modifié à partir du processeur que vous avez implémenté dans les laboratoires précédents. Certains changements ont dû être opérés pour pouvoir supporter le forwarding.

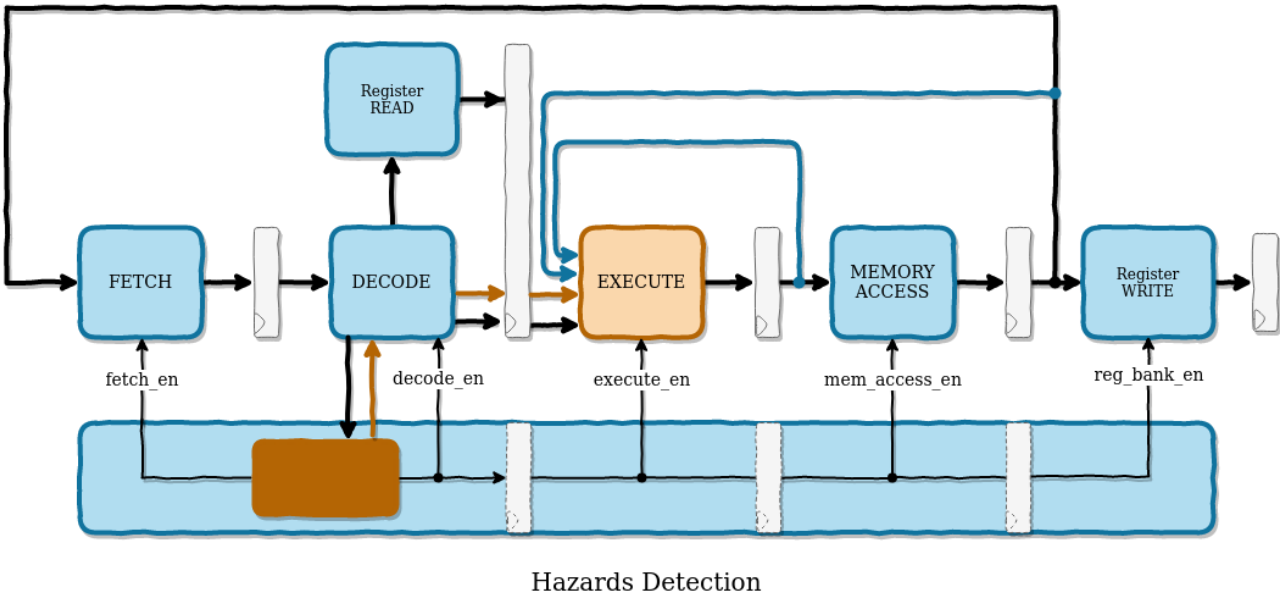

FIGURE 1 – Croquis du processeur pipeliné avec forwarding

Afin d'optimiser la gestion des aléas, des chemins pour les données ont été ajoutés au circuit afin d'obtenir la nouvelle valeur des registres avant qu'elle soit écrite dans les registres comme sur la Figure 1.
Le circuit dispose de 2 modes de fonctionnement. Le mode peut être changé dans le composant **data_hazard**, pour cela modifier la valeur de la constante du signal **hazard_detection_mode**.

Indications :

Les valeurs qui proviennent d'une instruction mémoire ne peuvent pas être forwardées.

1.2 Test du processeur avec un programme

Utiliser le programme (01_main.S) et tester l'exécution du programme dans le circuit Logisim pour les 2 modes de fonctionnement. Vous stopperez le chronogramme lorsque la dernière instruction a écrit sa valeur dans le registre.

Dans les chronogrammes vous placerez les signaux suivants en respectant l'ordre :

- **clk**, **fetch/instr_data_o**,

Ainsi que les signaux dans le composant decode/main_control_unit :

- **dec_instr**, **exec_instr**, **mem_instr**, **wrb_instr**,
- **fetch_en_o**, **decode_en_o**, **execute_en_o**, **memory_access_en_o**, **reg_bank_en_o**,

puis les signaux dans le composant decode/main_control_unit/hazard_detection/data_hazard :

- **no_data_hazard_o**, **sel_op1_forward_o**, **sel_op2_forward_o**, **sel_op_mem_forward_o**,

puis les signaux dans la banque de registre :

- **R0**, **R1**, **R2**, **R3**, **R4**, **SP**, **LR**, **PC**.

Questions

1. Tracer les chronogrammes de l'exécution du programme (01_main.S) pour les 2 modes de fonctionnement.
2. Annoter sur les chronogrammes le moment correspondant à la fin de l'exécution complète du programme, les instructions **"nop"** ne sont pas considérées comme faisant partir du programme.
3. Quel est l'IPC du programme (01_main.S) dans le circuit avec "hazard_detection_mode = 0" ?
4. Quel est l'IPC du programme (01_main.S) dans le circuit avec "hazard_detection_mode = 1" ?
5. Expliquer les 2 modes de fonctionnement ?
6. Utiliser le mode avec la valeur de la constante =1 et tracer le chonogramme de l'exécution du programme (03_main.S).
7. Pourquoi observe t'on un arrêt pendant l'exécution du programme avec le mode = 1 sélectionné ? Est-t-il normal qu'on observe un arrêt ? Annoter l'arrêt sur le chronogramme (03_main.S).

1.3 Circuit data_hazard

Des modifications ont été apportées au circuit data_hazard pour créer les signaux **sel_op1_forward_s**, **sel_op2_forward_s** et **sel_op_mem_s**. Ces signaux prennent les valeurs suivantes.

Valeur	Description
0	Il n'y pas de data forwarding possible ou nécessaire pour cette instruction
1	Data forwarding depuis le bloc EXECUTE
2	Data forwarding depuis le bloc MEMORY_ACCESS
3	Data forwarding depuis le bloc WRITE_BACK

sel_op1_forward_s fait la sélection pour l'opérande 1. **sel_op2_forward_s** fait la sélection pour l'opérand 2 et **sel_op_mem_forward_s** fait la sélection de la donnée pour les instructions mémoire.

Questions

8. Annoter sur le chronogramme (03_main.S) les moments où les valeurs des signaux **sel_opX...** sont utilisés. Expliquer ces valeurs.
9. Lors de la détection pour le forwarding, quelle est l'utilité du signal sel_mem_i dans la détection ? et pourquoi ce signal est utile ?
10. Est-il nécessaire de faire un data forwarding depuis le stage WRITE_BACK et pourquoi ?
11. Quelles sont les conditions pour que le forwarding puisse avoir lieu ?
12. Quelles sont les avantages et inconvénients du forwarding sur la gestion des aléas de données ?

1.4 Circuit execute

Le circuit Execute a été modifié pour que **operand_1_s** et **operand_2_s** soient séléctionnés avec les signaux **sel_op1_forward_s** et **sel_op2_forward_s** respectivement. La même chose est faite pour le signal **reg_mem_read_data_s** grâce au signal **sel_op_mem_forward_s**.

Questions

13. Que permet de réaliser les signaux **sel_opX_forward_s** dans le circuit Execute ?
14. Dans le circuit Execute, pourquoi il y a un registre connecté à l'entrée **memory_data_out_i** ?