TP Chapitre 05 : Exploration du CPU

Objectifs pratiques

- Explorer le CPU Visualizer
- Observer le cycle d'exécution
- Comprendre les signaux de contrôle
- Implémenter les composants du CPU

Durée estimée : 2h

Prérequis : TD Chapitre 05 terminé

Partie 1 : CPU Visualizer

Accès au Visualizer

👉 **Ouvrir le CPU Visualizer**

Interface :

- Vue pipeline (5 étapes)
- Panneau registres (R0-R15)
- Panneau flags (N, Z, C, V)
- Code source avec surlignage

Exercice 1 : Première Exploration

Objectif : Se familiariser avec l'interface

Étapes

1. Cliquez sur **Demo 1: Addition simple**
2. Cliquez sur **Reset** pour initialiser
3. Observez l'interface :
   - Où sont les registres ?
   - Où est le code ?
   - Où sont les étapes du pipeline ?

Validation

✅ Vous identifiez les 4 zones principales

Exercice 2 : Exécution Pas-à-Pas

Objectif : Observer le cycle d'exécution

Étapes

1. Avec **Demo 1: Addition**, cliquez sur **Step** (ou touche N)
2. Observez quelle étape s'illumine
3. Continuez en pas-à-pas
4. Notez :
   - Quand R0 change-t-il ?
   - Quand R1 change-t-il ?
   - Quand R2 change-t-il (résultat) ?

Code de la démo

```
MOV R0, #5      ; R0 = 5
MOV R1, #3      ; R1 = 3
ADD R2, R0, R1  ; R2 = R0 + R1 = 8
```

👉 **Exécuter**

Questions

1. Combien de cycles pour l'instruction MOV ?
2. À quelle étape le registre est-il modifié ?
3. Que vaut R2 à la fin ?

▸ Réponses

Exercice 3 : Observer les Flags

Objectif : Comprendre N, Z, C, V

Étapes

1. Chargez **Demo 6: Flags CPU**
2. Exécutez pas-à-pas
3. Observez les flags après chaque instruction

Questions

- Quelle instruction met Z = 1 ?
- Quelle instruction met N = 1 ?
- Quelle instruction met C = 1 ?

👉 **Exécuter Demo Flags**

▸ Indice

Exercice 4 : Branchement Conditionnel

Objectif : Observer un branchement

Étapes

1. Chargez **Demo 2: Boucles**
2. Exécutez jusqu'à l'instruction B.LE
3. Observez :
   - Les flags après CMP
   - Le PC change-t-il après B.LE ?
   - Combien de fois la boucle s'exécute ?

Code de la démo

```
    MOV R0, #0        ; sum = 0
    MOV R1, #1        ; i = 1
loop:
    ADD R0, R0, R1    ; sum += i
    ADD R1, R1, #1    ; i++
    CMP R1, #5
    B.LE loop         ; si i <= 5, boucler
```

👉 **Exécuter**

Questions

1. Pourquoi utilise-t-on CMP avant B.LE ?
2. Que vaut R0 à la fin (somme de 1 à 5) ?

▸ Réponses

Exercice 5 : Accès Mémoire

Objectif : Observer LDR et STR

Étapes

1. Chargez **Demo 3: Accès mémoire**
2. Exécutez pas-à-pas
3. Observez :
    - L'étape **Memory** s'active-t-elle ?
    - Quand la valeur apparaît-elle dans le registre ?

👉 **Exécuter Demo Mémoire**

Partie 2 : Implémentation HDL (Optionnel)

Accès au Simulateur HDL

👉 **Ouvrir le Simulateur HDL**

Allez dans **HDL Progression → Projet 5 : CPU**

Exercice 6 : Le Décodeur

Objectif : Extraire les champs de l'instruction

Spécification

```
entity Decoder is
  port(
    instr : in bits(31 downto 0);
    cond  : out bits(3 downto 0);
    class : out bits(2 downto 0);
    op    : out bits(3 downto 0);
    s_bit : out bit;
    rn    : out bits(3 downto 0);
    rd    : out bits(3 downto 0);
    rm    : out bits(3 downto 0);
    imm12 : out bits(11 downto 0)
  );
end entity;
```

Code à compléter

```
architecture rtl of Decoder is
begin
  cond  <= instr(31 downto 28);
  class <= instr(27 downto 25);
  -- TODO: Compléter les autres champs
end architecture;
```

▸ Solution

Exercice 7 : CondCheck

Objectif : Vérifier les conditions

Spécification

```
entity CondCheck is
  port(
    cond : in bits(3 downto 0);
    n, z, c, v : in bit;
    ok : out bit
  );
end entity;
```

Logique

cond	Condition	Test
0000	EQ	Z = 1
0001	NE	Z = 0
1010	GE	N = V
1011	LT	N ≠ V
1110	AL	toujours

👉 **Ouvrir l'exercice**

Solution CondCheck

▸ Solution

Exercice 8 : Charger Votre Code

Objectif : Exécuter un programme personnel

Étapes

1. Dans le CPU Visualizer, cliquez sur **Charger fichier**
2. Créez un fichier `test.asm` avec ce code :

```
; Calcul de 10 + 20 + 30
MOV R0, #10
MOV R1, #20
MOV R2, #30
ADD R3, R0, R1
ADD R3, R3, R2
HALT
```

1. Chargez et exécutez
2. Vérifiez que R3 = 60

👉 **Charger dans le Visualizer**

Récapitulatif

Composants explorés

Composant	Fonction
Pipeline	5 étapes d'exécution
Registres	R0-R15, mise à jour en WB
Flags	N, Z, C, V après opérations
Branchements	Modifient le PC
Mémoire	LDR/STR à l'étape MEM

Validation Finale

Checklist

- [ ] Exécuter une démo pas-à-pas
- [ ] Observer les flags changer
- [ ] Comprendre un branchement conditionnel
- [ ] Voir l'étape Memory s'activer pour LDR
- [ ] (Optionnel) Implémenter le Decoder

Prochaine étape

➡️ **Chapitre 06 : Assembleur** — Traduire le code assembleur en binaire !

📚 **Référence :** Livre Seed, Chapitre 05 - CPU