TP Chapitre 04 : Programmation Assembleur A32

Objectifs pratiques

- Écrire et exécuter des programmes assembleur
- Utiliser le simulateur A32
- Maîtriser les boucles et conditions
- Manipuler la mémoire et les périphériques

Durée estimée : 2h

Prérequis : TD Chapitre 04 terminé

Préparation

Accès au Simulateur

👉 **Ouvrir le Simulateur A32**

Sélectionnez **A32 Assembly** dans le menu.

Interface :

- Zone code (gauche)
- Registres (droite)
- Mémoire (bas)
- Contrôles : Run, Step, Reset

Exercice 1 : Premier Programme

Objectif : Vérifier que l'environnement fonctionne

Code

```
; Mon premier programme A32
.text
.global _start

_start:
    MOV R0, #42       ; R0 = 42
    MOV R1, #10       ; R1 = 10
    ADD R2, R0, R1    ; R2 = R0 + R1
    HALT
```

Validation

👉 **Exécuter dans le simulateur**

✅ Après exécution, R2 doit contenir **52**

Exercice 2 : Opérations Arithmétiques

Objectif : Pratiquer ADD, SUB, MUL

Code à compléter

```
.text
.global _start

_start:
    ; a = 15, b = 7
    MOV R0, #15
    MOV R1, #7

    ; c = a + b
    ; TODO: R2 = ?

    ; d = a - b
    ; TODO: R3 = ?

    ; e = a * b (utiliser MUL)
    ; TODO: R4 = ?

    HALT
```

Validation

✅ R2 = 22, R3 = 8, R4 = 105

👉 **Exécuter**

Solution Exercice 2

▸ Solution

Exercice 3 : Condition Simple

Objectif : Utiliser CMP et branchements

Problème

Calculer le maximum de R0 et R1, stocker dans R2.

```
.text
.global _start

_start:
    MOV R0, #25
    MOV R1, #17

    ; TODO: R2 = max(R0, R1)
    ; Utiliser CMP, B.GT (ou B.LT), et des labels

    HALT
```

👉 **Ouvrir l'exercice**

Solution avec branchement

▸ Solution

Solution avec prédication

- Solution alternative

Exercice 4 : Boucle - Somme de 1 à N

Objectif : Implémenter une boucle

Problème

Calculer la somme 1 + 2 + ... + N (N dans R0).

```
.text
.global _start

_start:
    MOV R0, #10       ; N = 10
    MOV R1, #0        ; sum = 0
    MOV R2, #1        ; i = 1

    ; TODO: Boucle while (i <= N)
    ;       sum += i
    ;       i++

    ; R1 devrait contenir 55
    HALT
```

👉 Ouvrir l'exercice

Solution Exercice 4

▸ Solution

Exercice 5 : Fibonacci

Objectif : Boucle avec deux variables

Problème

Calculer le N-ième nombre de Fibonacci.
F(0)=0, F(1)=1, F(n)=F(n-1)+F(n-2)

```
.text
.global _start

_start:
    MOV R0, #10       ; N = 10

    ; TODO: Calculer F(10) dans R3
    ; F(10) = 55

    HALT
```

👉 Ouvrir l'exercice

Solution Fibonacci

▸ Solution

Exercice 6 : Accès Mémoire

Objectif : Utiliser LDR et STR

Problème

Lire une valeur en mémoire, la doubler, la réécrire.

```
.data
    value: .word 21   ; Variable en mémoire

.text
.global _start

_start:
    LDR R0, =value    ; R0 = adresse de value

    ; TODO:
    ; 1. Lire value dans R1
    ; 2. Doubler R1
    ; 3. Écrire R1 dans value

    HALT
```

👉 Ouvrir l'exercice

Solution Exercice 6

▸ Solution

Exercice 7 : Parcours de Tableau

Objectif : Indexation mémoire

Problème

Trouver le maximum dans un tableau de 5 éléments.

```
.data
    array: .word 12, 45, 7, 89, 23
    size:  .word 5

.text
.global _start

_start:
    LDR R0, =array    ; Adresse du tableau
    LDR R1, =size
    LDR R1, [R1]      ; Taille

    ; TODO: Trouver le max dans R2

    HALT              ; R2 devrait être 89
```

👉 Ouvrir l'exercice

Solution Tableau

▸ Solution

Exercice 8 : Dessiner un Pixel

Objectif : Utiliser le MMIO

Rappel

- Écran : `0x00400000`
- 320×200 pixels, 1 bit/pixel
- 40 octets par ligne

Code

```
.text
.global _start

_start:
    ; Allumer pixel (0, 0)
    LDR R0, =0x00400000
    MOV R1, #0x80          ; Bit 7
    STRB R1, [R0]

    ; TODO: Allumer pixel (8, 0)
    ; TODO: Allumer pixel (0, 1)

    HALT
```

👉 **Ouvrir l'exercice**

Solution Pixels

▸ Solution

Exercice 9 : Dessiner une Ligne (Défi)

Objectif : Combiner boucle et MMIO

Dessinez une ligne horizontale de 32 pixels à la ligne 10.

▸ Indice

👉 **Ouvrir l'exercice**

Solution Ligne

▸ Solution

Récapitulatif

Programmes écrits

Exercice	Concept
Premier programme	MOV, ADD
Arithmétique	ADD, SUB, MUL
Maximum	CMP, B.cond, prédication
Somme	Boucle while
Fibonacci	Variables multiples
Mémoire	LDR, STR
Tableau	Indexation
Pixel	MMIO
Ligne	Boucle + MMIO

Validation Finale

Checklist

- [ ] Exécuter un programme simple
- [ ] Utiliser les conditions (CMP, B.EQ, etc.)
- [ ] Écrire une boucle
- [ ] Accéder à la mémoire (LDR/STR)
- [ ] Dessiner sur l'écran (MMIO)

Prochaine étape

➡️ **Chapitre 05 : CPU** — Construire le processeur qui exécute ces instructions !

📚 **Référence :** Livre Seed, Chapitre 04 - Architecture