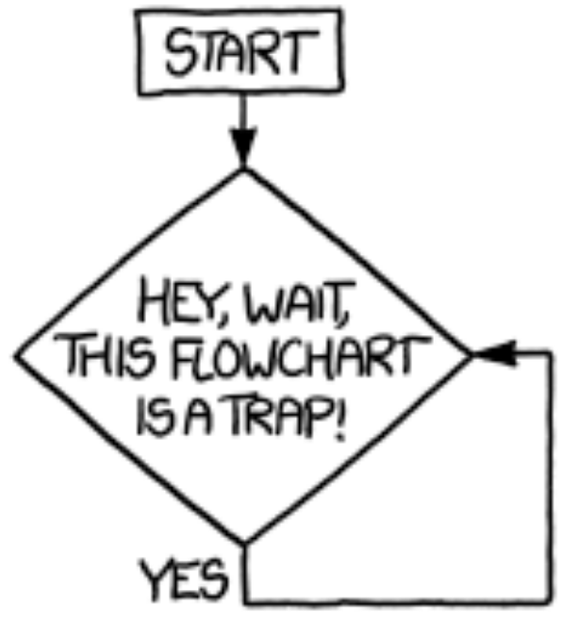
Assembleur ARM:
Séquence d'exécution et branchements



xkcd.com

GIF-1001 Ordinateurs: Structure et Applications

Jean-François Lalonde

Merci à Yves Roy

Modification de la séquence d'exécution

- Par défaut, les instructions s'exécutent séquentiellement et PC est incrémenté automatiquement par le microprocesseur entre chaque instruction
  - PC = PC + 4 (taille d'une instruction)
- Dans nos programme, il arrive que l'on veuille exécuter autre chose que la prochaine instruction:
  - Saut direct à une instruction
  - Énoncé conditionnel "si" (`if`)
  - Boucle: "répète N fois" (`for`) ou "répète tant que" (`while`)
  - Appel de fonction
- Il est possible de contrôler la séquence d'exécution, en assembleur, avec des instructions qui modifient PC.

Modifications à PC

- PC est un registre et peut être modifié comme les autres registres, avec la plupart des instructions

- Modifier la valeur de PC correspond à effectuer un saut absolu à une adresse.

- Exemples:

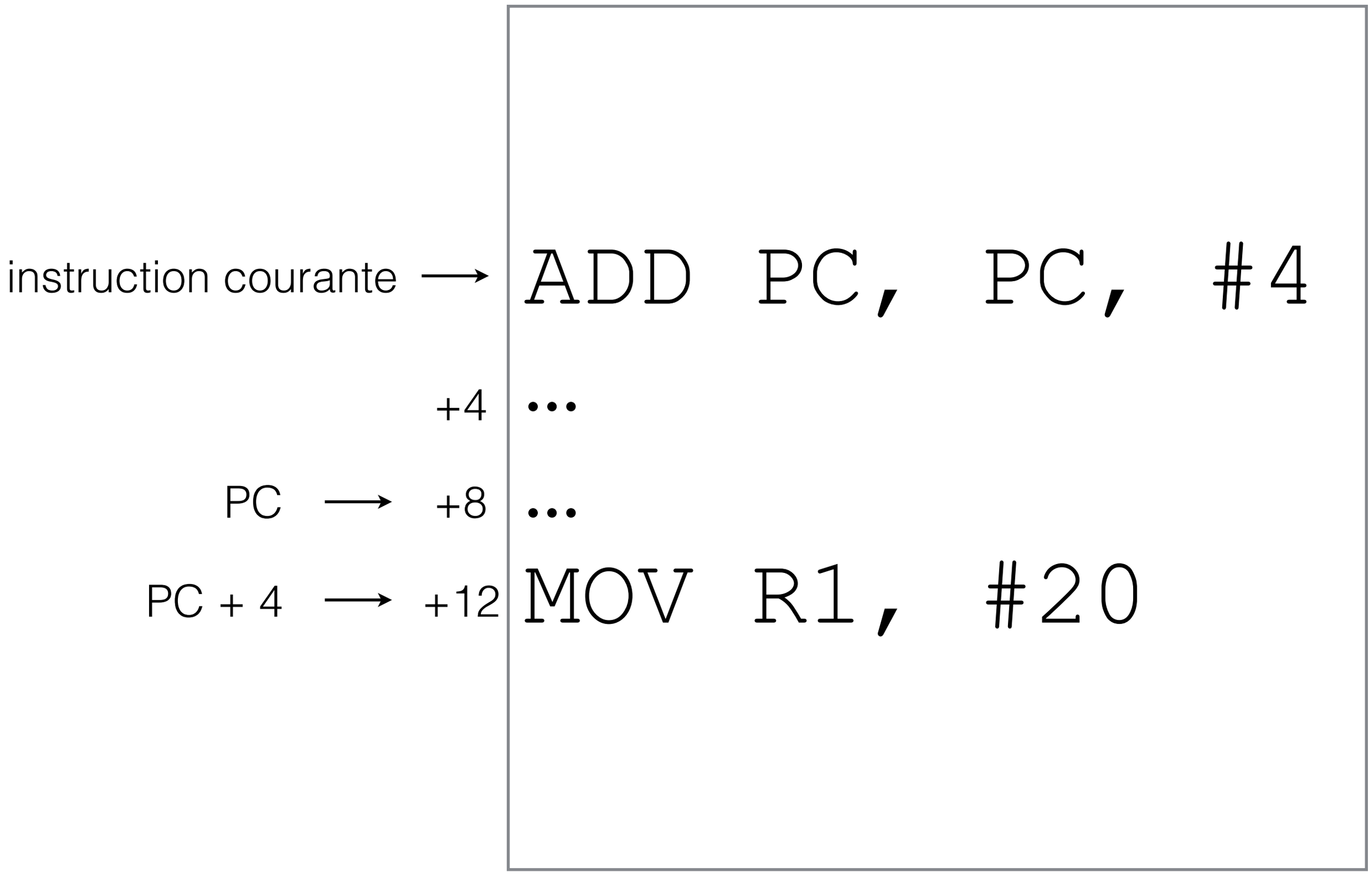
Lorsque l'on modifie PC,
le microprocesseur n'effectue pas PC = PC + 4.

```
MOV PC, #0x80          ; PC = 0x80
MOV PC, R0             ; PC = R0
LDR PC, [R0]           ; PC = Memoire[R0]
ADD PC, R0, #4         ; PC = R0 + 4
```

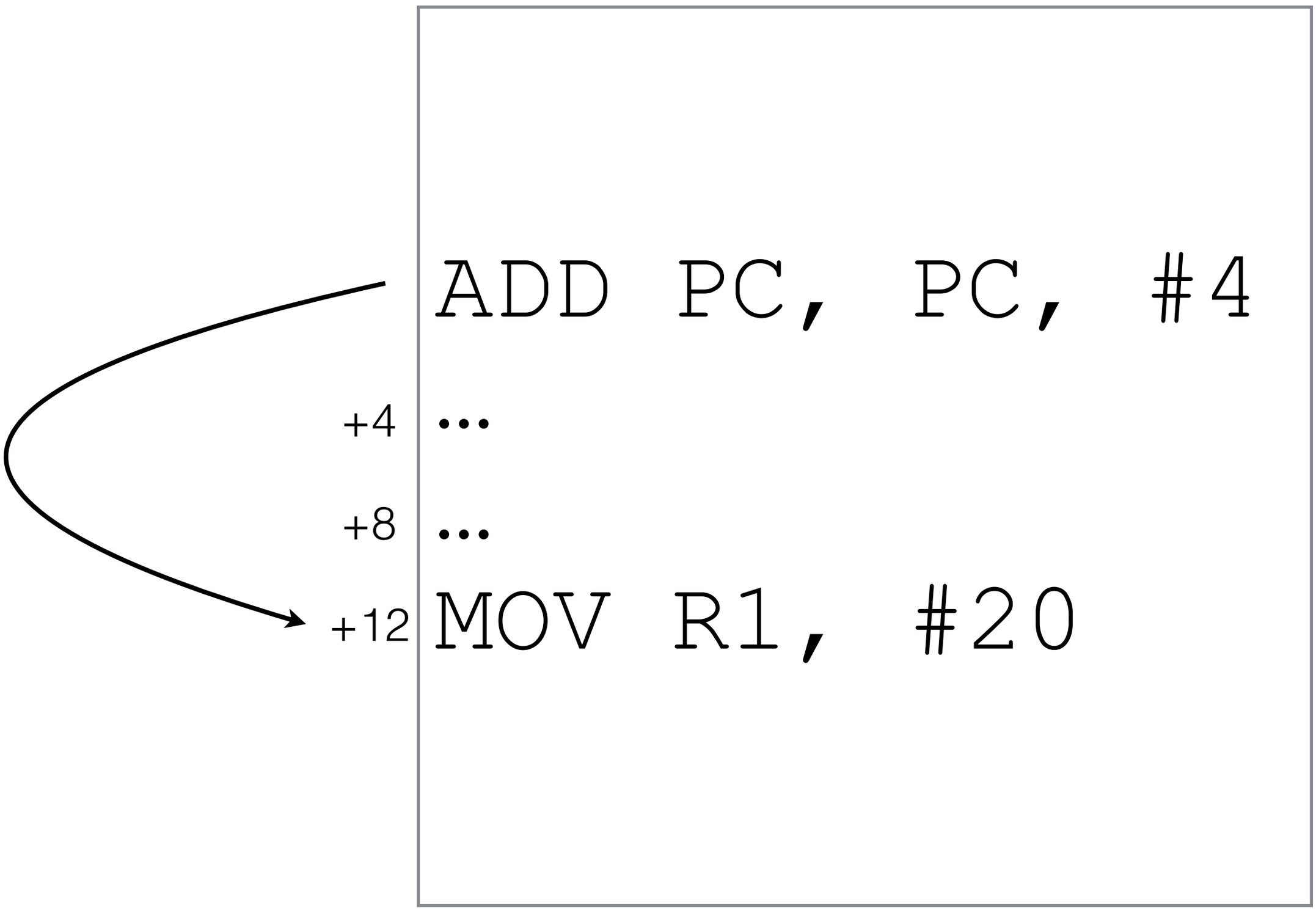
Modifications à PC

```
ADD PC, PC, #4
…
…
MOV R1, #20
```

Modifications à PC



Modifications à PC



Démonstration

(Modifications à PC)

Branchements *inconditionnels*

- Il n'est pas très pratique d'avoir à déterminer l'adresse directement

- L'instruction B (Branch) saute à une « étiquette » déterminée par le programme:

```
B etiquette
```

- L'assembleur convertit l'étiquette par son adresse relative à PC.

```
B main ; saute à l'adresse indiquée par
l'étiquette "main"
...

main
MOV R0, #1
```

Branchements *conditionnels*

- On peut également utiliser la version *conditionnelle* de l'instruction `B` (`BNE`, `BEQ`, `BLT`, etc.)

- Exemple: Si `maVar` vaut 4, exécute une tâche:

exemple en C

```
if (maVar == 4) {
  // exécute une tâche…
}

// le programme continue…
```

Branchements *conditionnels*

exemple en C

```
if (maVar == 4) {
  // exécute une tâche…
}

// le programme continue…
```

Branchements *conditionnels*

exemple en C

```
if (maVar == 4) {
  // exécute une tâche…
}

// le programme continue…
```

```
LDR   R0, maVar
CMP   R0, #4      ; Effectue la comparaison

BNE   PasEgal     ; NE = "Not Equal"
; execute une tâche…

PasEgal
; le programme continue…
```

Autre exemple: if/else

exemple en C

```
if (maVar == 4) {
  // exécute une tâche…
} else {
  // exécute une autre tâche…
}
// le programme continue…
```

```
LDR   R0, maVar
CMP   R0, #4      ; Effectue la comparaison
BNE   PasEgal     ; NE = “Not Equal”
; execute une tâche (dans le “if”)


PasEgal
```

Autre exemple: if/else

exemple en C

```
if (maVar == 4) {
  // exécute une tâche…
} else {
  // exécute une autre tâche…
}
// le programme continue…
```

assembleur ARM

```
LDR   R0, maVar
CMP   R0, #4      ; Effectue la comparaison
BNE   PasEgal     ; NE = “Not Equal”
; execute une tâche (dans le “if”)

B Continue
PasEgal
; exécute une autre tâche (dans le “else”)

Continue
; le programme continue...
```

Démonstration

(if/else avec branchements)

Boucles

- Une boucle (répète N fois ou tant que) est constituée de trois opérations:
  1. initialiser la boucle (point de départ)
  2. condition d'arrêt
  3. opération mathématique qui mènera à la réalisation de la condition d'arrêt (progression)

- Voici un exemple de boucle qui se répète cinq fois:

exemple en C

```
for (int i = 0; i < 5; i++) {
  // tâche à l'intérieur de la boucle
}
// le programme continue…
```

Point de départ

Condition d'arrêt

Progression vers la condition d'arrêt

Boucles

exemple en C

```
for (int i = 0; i < 5; i++) {
  // tâche à l'intérieur de la boucle
}
// le programme continue…
```

Boucles

exemple en C

```
for (int i = 0; i < 5; i++) {
  // tâche à l'intérieur de la boucle
}
// le programme continue…
```

assembleur

```
MOV R4, #0                ; Point de départ

DebutDeBoucle
CMP R4, #5                ; Condition d'arrêt
BGE FinDeBoucle

; tâche à l'intérieur de la boucle

ADD R4, R4, #1            ; Progression
B DebutDeBoucle           ; Effectue la boucle
FinDeBoucle

; le programme continue…
```

Démonstration

(Tableaux avec boucle)