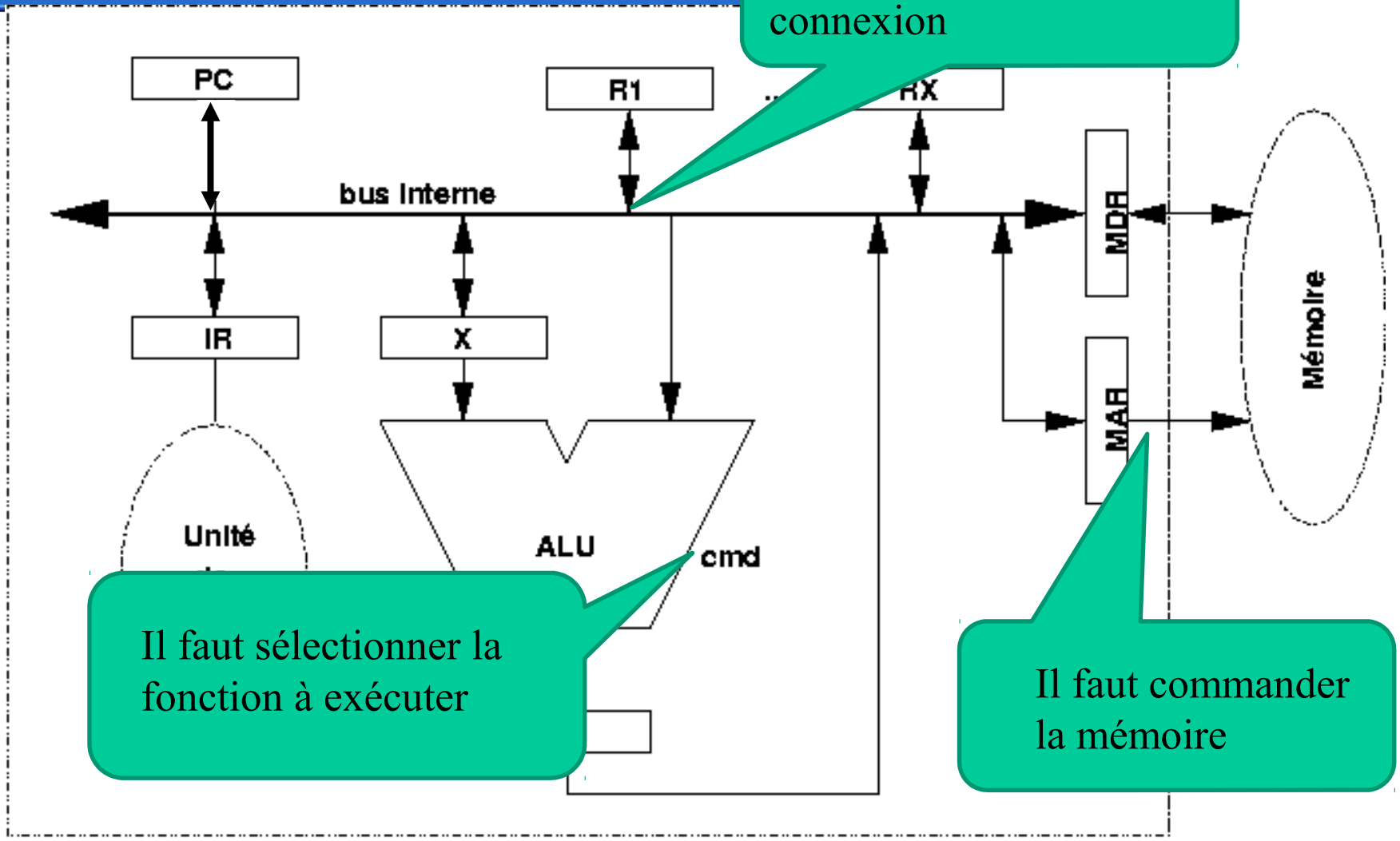
AE 9

De l’instruction à la micro programmation

Objectifs

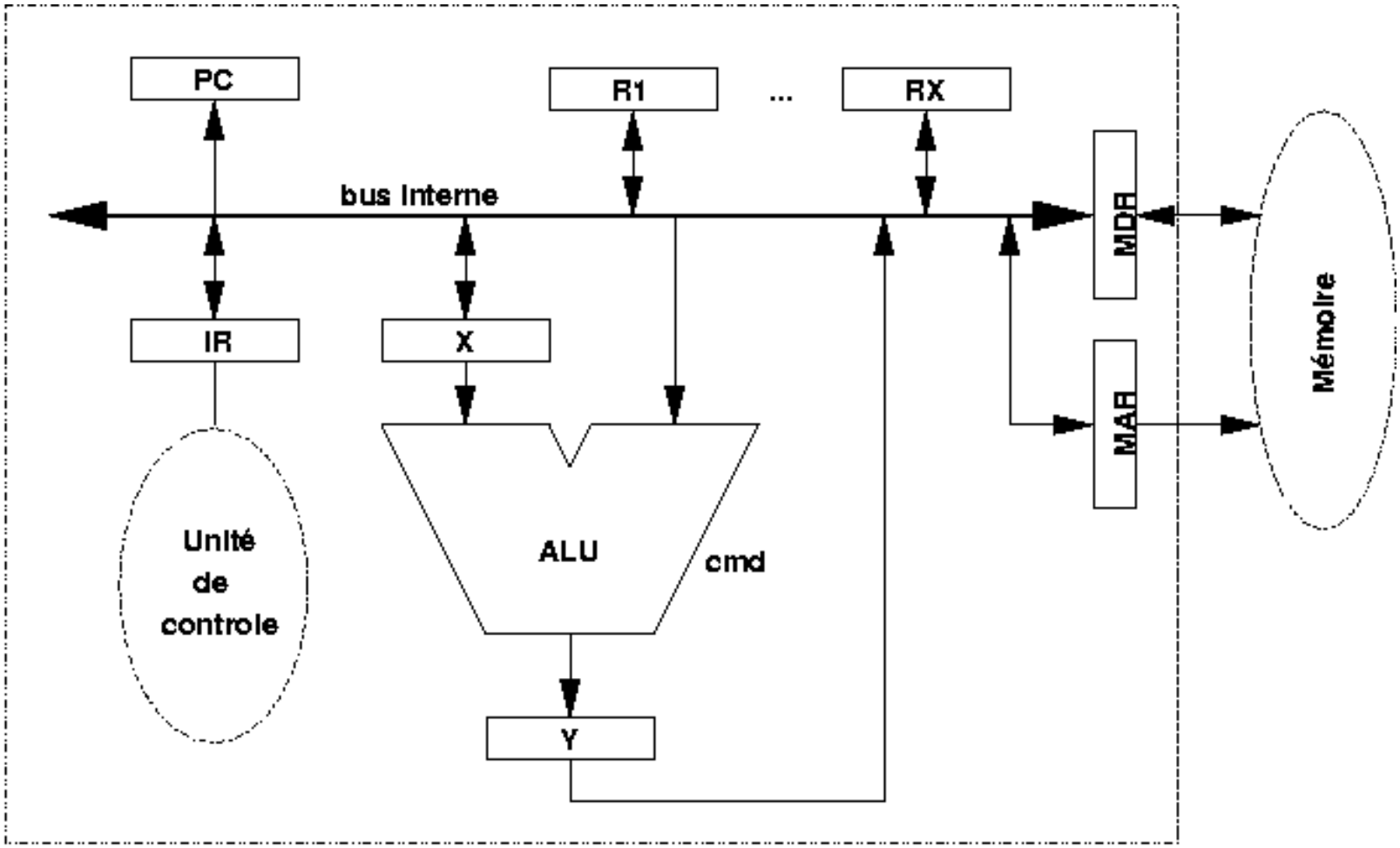
- Une vue globale du déroulement d'une instruction
- Du RI vers l'exécution
  - Décodages d'instruction
  - séquencement des micro-opérations
  - l'exécution des micro-opérations
    - Notion de signaux/commandes
    - Micro architecture

Une machine simple



Phase de chargement

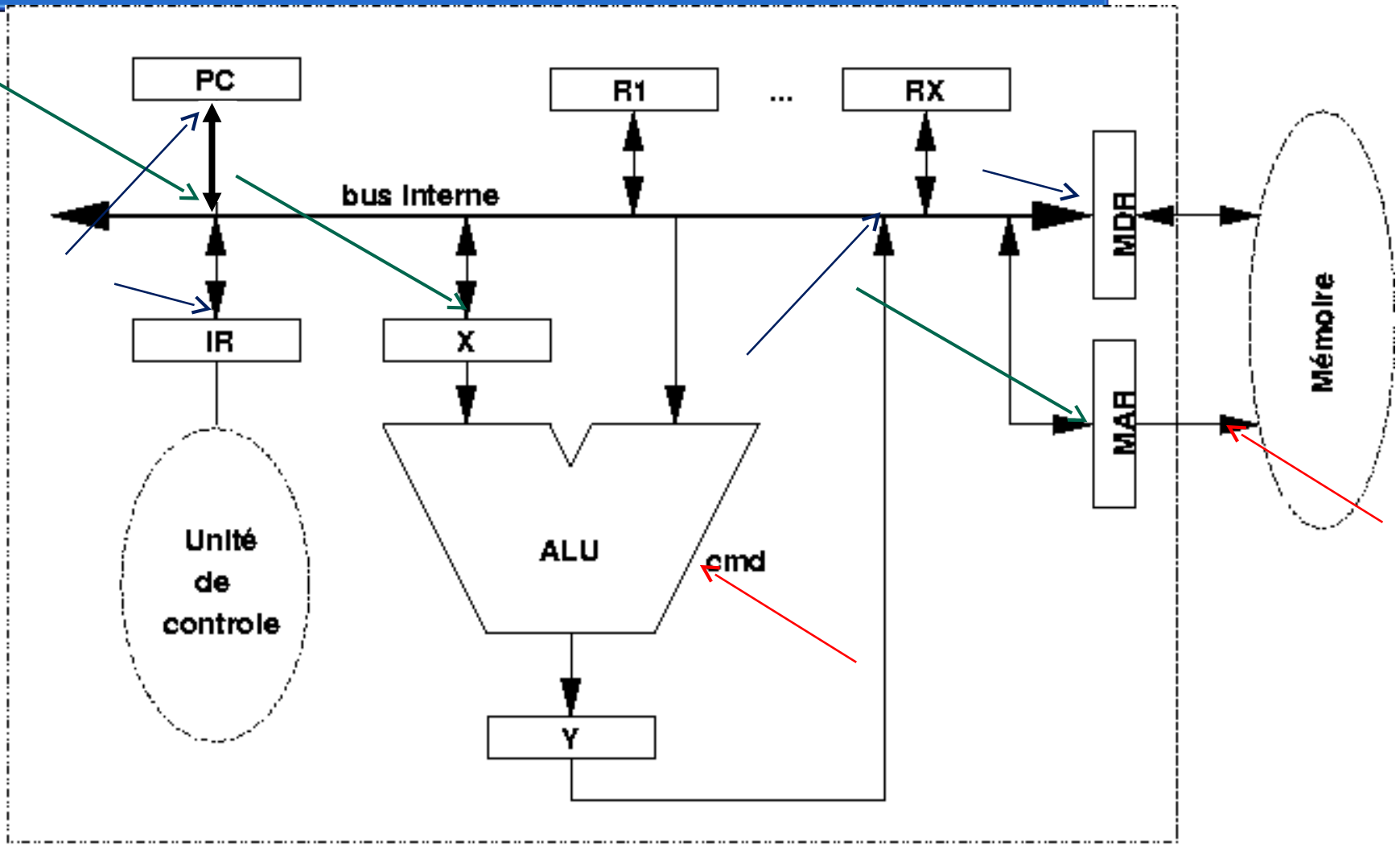
- **Phase 1 – Chargement de l'instruction**
  - PC pointe vers la case mémoire contenant la prochaine instruction à exécuter: ADD R1, 4
    - Pour ce processeur : codé sur 2 mots : ADD R1 et 4
  - **Mettre PC dans MAR et donner l'ordre de lecture**
    - PCout – MARin (En même temps!!!)
    - Read – (WaitMemory)
  - **Placer la valeur de MDR dans IR pour que l'instruction soit décodée**
    - MDRout – Irin (En même temps!!!)


PC
R1
...
RX
bus interne
MDR
Mémoire
IR
X
MAR
Unité
de
controle
ALU
cmd
Y

Phase de chargement

- **Phase 1 (suite)**
  - **On profite du fait que la lecture en mémoire est lente pour incrémenter PC**
  - **PC pointe vers l'emplacement mémoire suivant**
  - **« code » de la phase 1 :**
    - PCout – MARin – Xin
    - Read – INCX
    - Yout – PCin – (WaitMemory)
    - MDRout – IRin
- **Prêt pour décoder !!!! ADD R1 est dans le RI**

Une machine simple



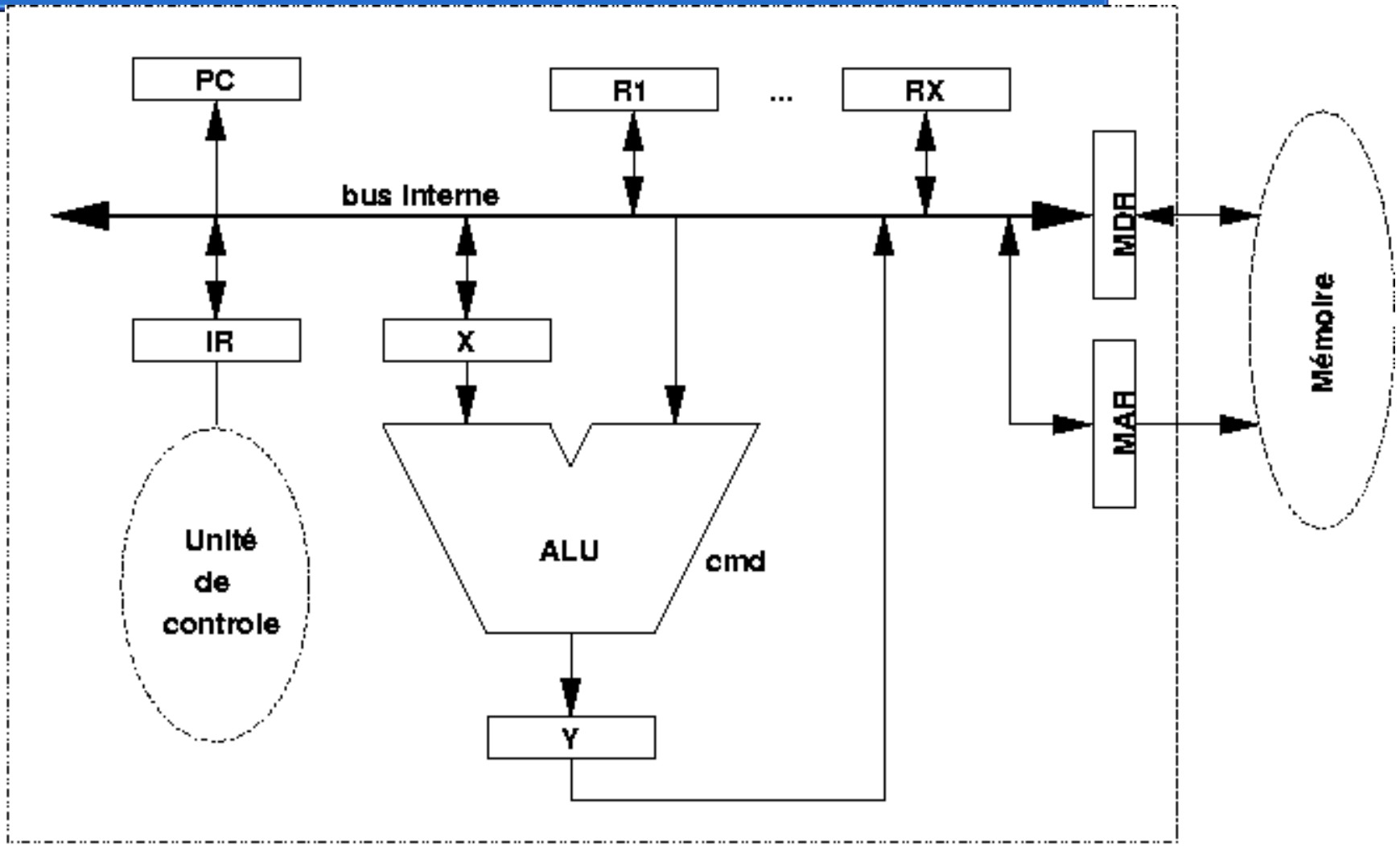
Phase d'exécution

- Phase 2 – décodage de l'instruction
  - Pris en charge par l'UC (unité de contrôle - FSM)
- Phase 3 – exécution
  - Deux sous-phases :
    - 3.1 Récupérer les arguments éventuellement
    - 3.2 Exécution

Phase d'exécution

- **3.1 lecture argument + incrémentation PC**
  - PCout – MARin – Xin
  - Read – INCX
  - Yout – PCin – (WaitMemory)
- **3.2 exécuter l'instruction (ici ADD R1,4)**
  - R1out – Xin
  - MDRout – ADD
  - Yout – R1in

codé sur 2 mots : ADD R1 et 4

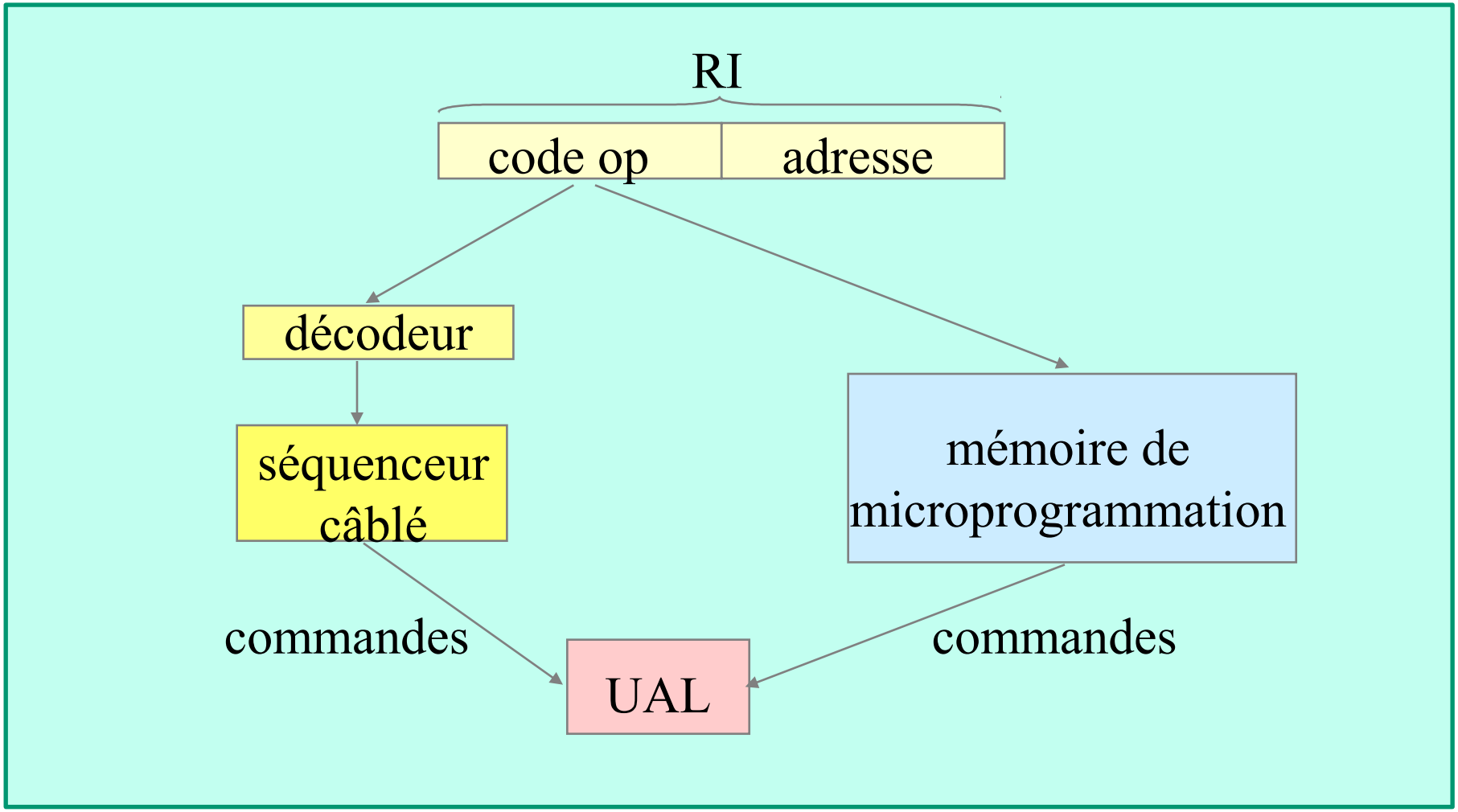

PC
R1
...
RX
bus Interne
MDR
Mémoire
IR
X
MAR
Unité
de
controle
ALU
cmd
Y

Décodage et contrôle

- Comment sont réalisées les Unités de Contrôle dans les microprocesseurs ?
- Automates d'états finis
  - Circuits logique combinatoire
    - Prise en compte des signaux de contrôle et registre condition
    - codeur multiplexeur etc..

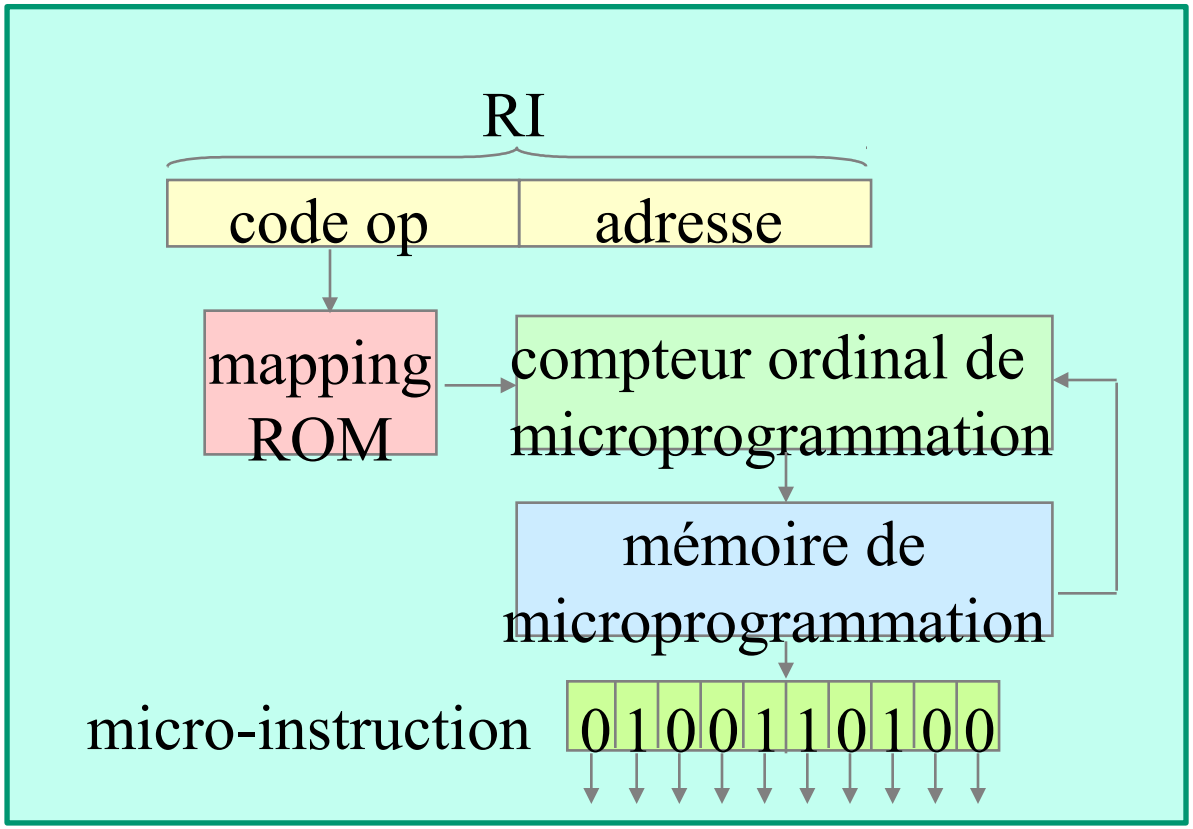
  - Micro architecture
    - Micro programmation
    - Mémoire morte chaque case contient les signaux à activer à chaque top d'horloge

Du RI aux commandes ou signaux



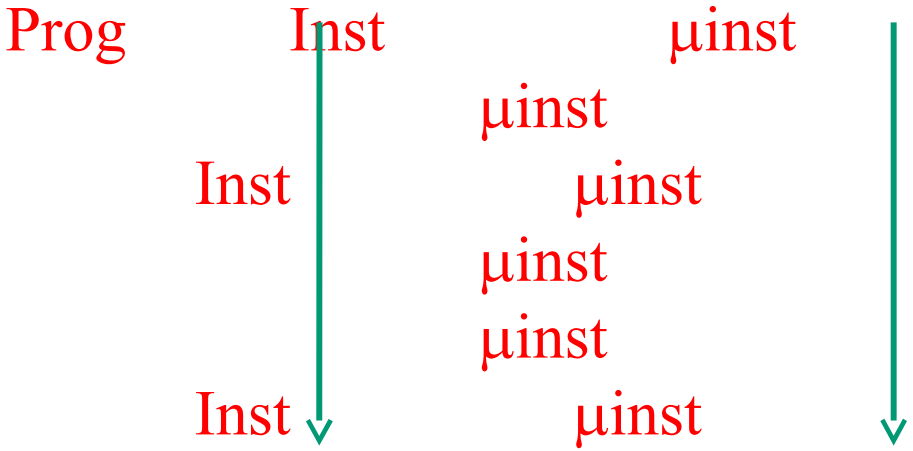
Le modèle avec mémoire morte

- Le microprogramme peut être stocké dans une ROM ou une EPROM.



Instruction vs Micro-instruction

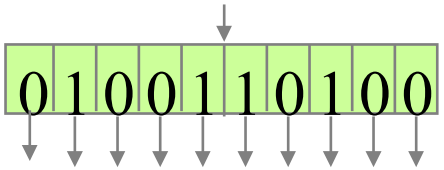
- Les instructions machines(macro-instructions) sont interprétées par le micro-séquenceur qui exécute le microcode.
- Plusieurs micro-instructions sont exécutées séquentiellement à chaque top d'horloge pour réaliser une (macro-)instruction



Microprogrammation «horizontale»

Chaque ligne de la mémoire de microprogramme est une micro-instruction

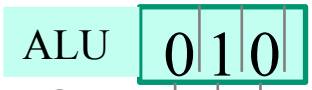
- **Chaque bit du mot est relié à un signal de contrôle**



- **Toutes les combinaisons sont possibles mais çà prend beaucoup de place**

Microprogrammation «verticale»

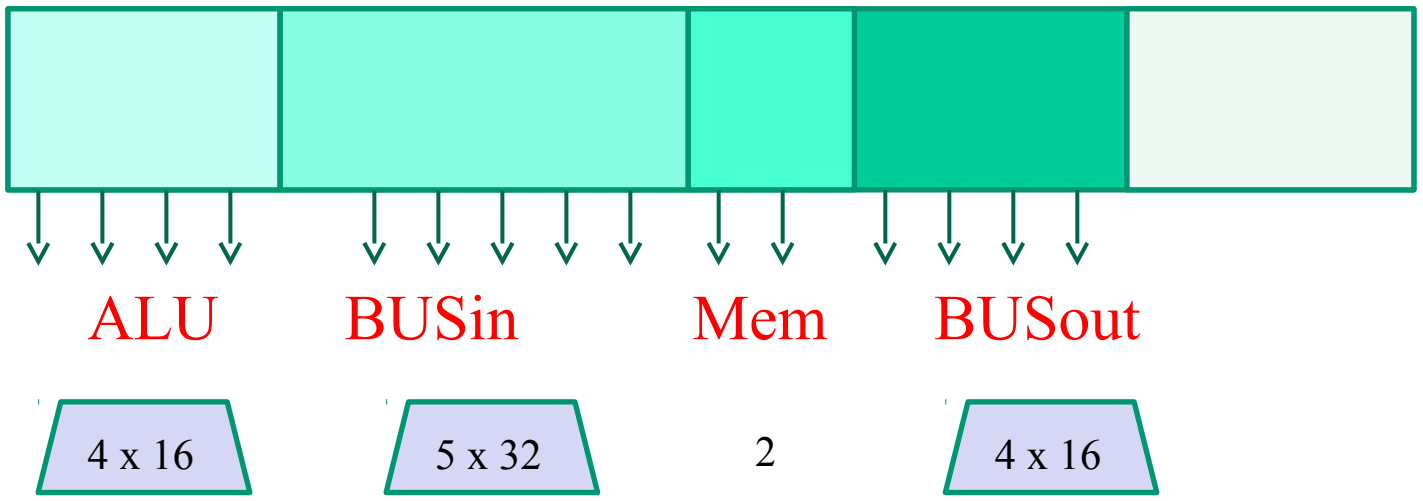
- Une µinstruction est associée à un groupe de commande
  - Par exemple un groupe
    - lect/ecr mémoire
    - Fonction ALU ADD, INCX ….
- Il faut alors identifier le groupe correspondant aux commandes déclenchées



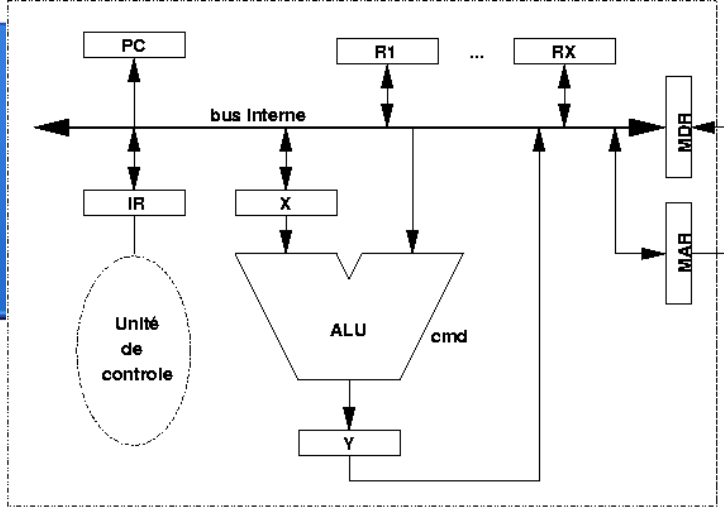
- Permet de déclencher 8 commandes avec 3 bits (et un décodeur)

Microprogrammation par champs

- On mélange horizontale et verticale
  - Un ensemble de champs et un codage par groupe dans chaque champs



Une machine simple



- **ALU Cmd**
  - add sub shift rot inc…
- **BUSin**
  - Pcin, R0in..Rxin, MDRin, MARin, Xin, IRin
- **BUSout**
  - Pcout, R0out..Rxout, MDRout, Yout, IRout
- **Mem**
  - Lect, Ecr, (waitmemory)

- La µinst doit correspondre à des signaux sur le CPU