Formation Automatique et Informatique Industrielle

Master 1 S2

Matière : Systèmes Embarqués et Systèmes Temps Réel SE-STR

Par : ATOUI Hamza

Plan du cours

- Le cas général d'un µP.
  - Architecture de Von-Neumann et Harvard.
  - Codage CISC et RISC.
  - Exemple de codage CISC vs RISC.
  - Phases et cycle machine.

Le cas général

- Maintenant, imaginez qu'on a plus de 5 **cases mémoires**. La même histoire pour les **entrées/sorties**. L'algorithme qu'on va exécuter prend plus de 1000 lignes avec un **ensemble d'opérations** assez nombreux.
- **Donc, comment démarrer de la logique programmée vers une architecture générique qui exécute n'importe quel algorithme ?????**

Le cas général

- d'après l'introduction, notre logique programmée (**microprocesseur**) manipule une zone contient l'algorithme, une zone contient les cases mémoires temporaires et une zone contient les entrées/sorties, dont les noms techniques des ces zones sont :
  1. **CODE MEMORY** (contient l'algorithme).
  2. **DATA MEMORY** (contient les cases mem tmp).
  3. **I/O SPACE** (les entrées/sorties).

Le cas général

- **Le µP est le résultat de l'intégration de plusieurs circuits LSI/VLSI pour exécuter un traitement (SOFTWARE) existe dans une mémoire.**
- **Le traitement est un ensemble d'instructions l'une après l'autre (séquence), dont le µP prend un temps pour exécuter.**
- **Chaque instruction prend une phase ou plusieurs pour exécuter; on général, ces phase sont (FETCH/DECODE,LECTURE DES OPERANDES, EXECUTION, ECRITURE).**
  - **FETCH/DECODE : phase de recherche de l'instruction dans la mémoire puis décoder .**
  - **LECTURE DES OPERANDES : phase optionnelle, si l'instruction a besoin des opérandes, le µP charge ces opérandes à partir de la mémoire ou un port (exemple : adresse d'une case mémoire, constante…).**
  - **EXECUTION : phase de réalisation de l'opération demander dans l'instruction (exemple : addition, soustraction…).**
  - **ECRITURE : phase optionnelle, si l'instruction a besoin d'écrire le résultat dans une case mémoire ou un port.**

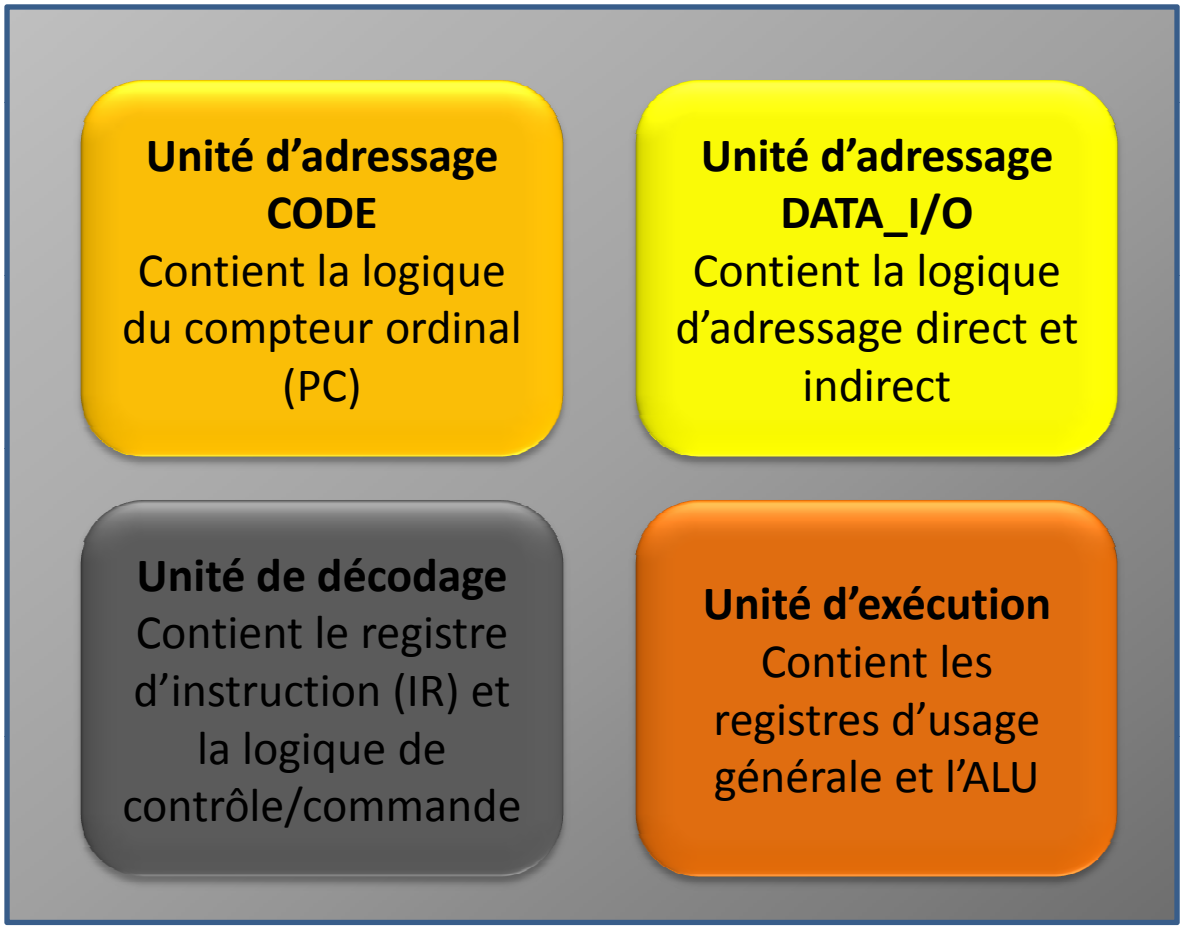
  N.B : chaque phase prend un ou plusieurs top d'horloge.

Le cas général

- Les deux célèbres architectures des µP sont :
  - L'architecture de **Von-Neumann** : se base sur le partage de buses entre les différentes zones mémoires (CODE, DATA et I/O).
  - L'architecture de **Harvard** : se base sur la séparation de buses entre les différentes zones mémoires.
- Dans les 2 cas le µP se divise sur **4 unités** sont :
  1. **Unité d'adressage de la zone DATA_I/O.**
  2. **Unité d'adressage de la zone CODE.**
  3. **Unité de décodage.**
  4. **Unité d'exécution.**

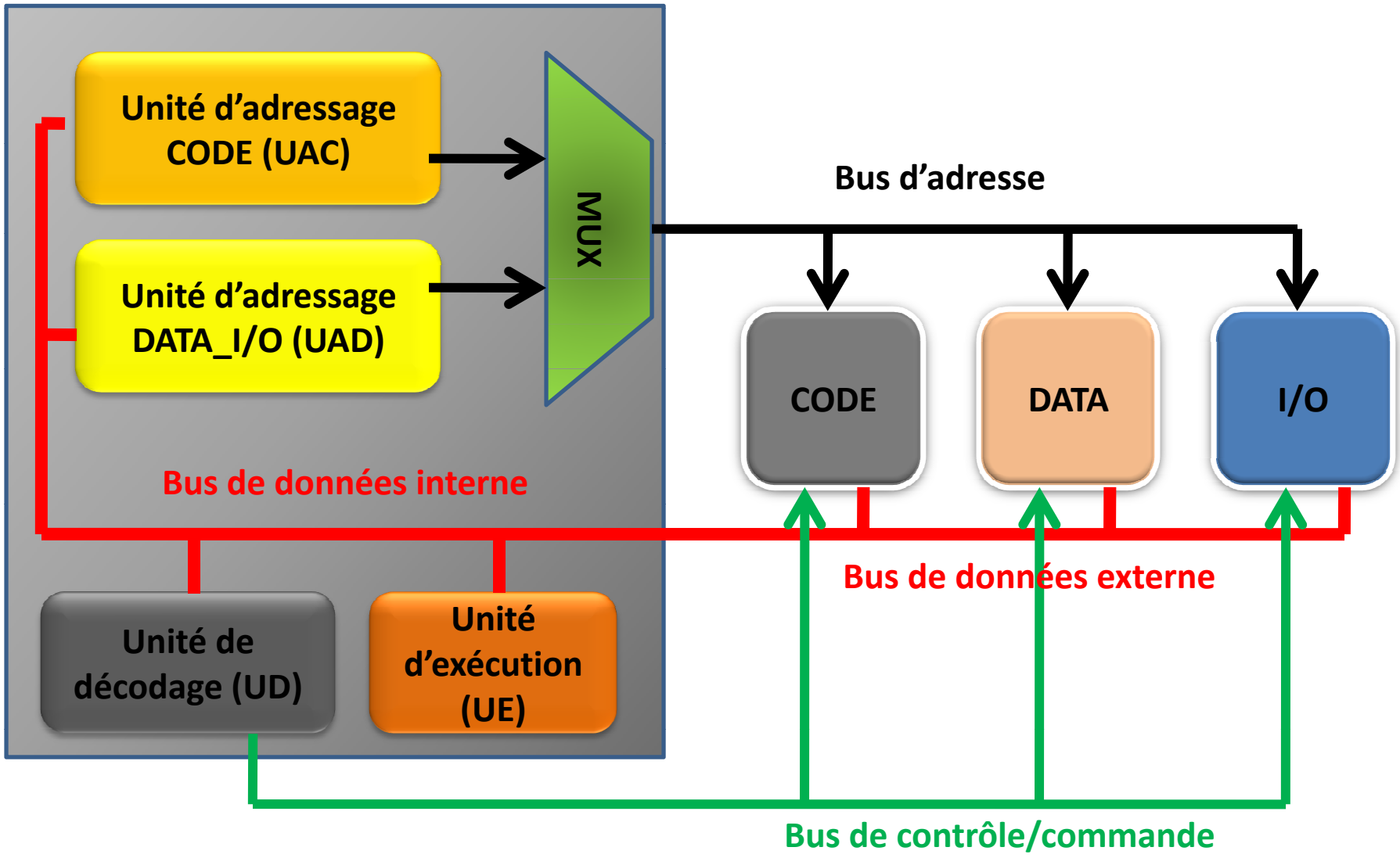
Le cas général

- La figure suivante présente un µP avec les différentes unités qui compose.



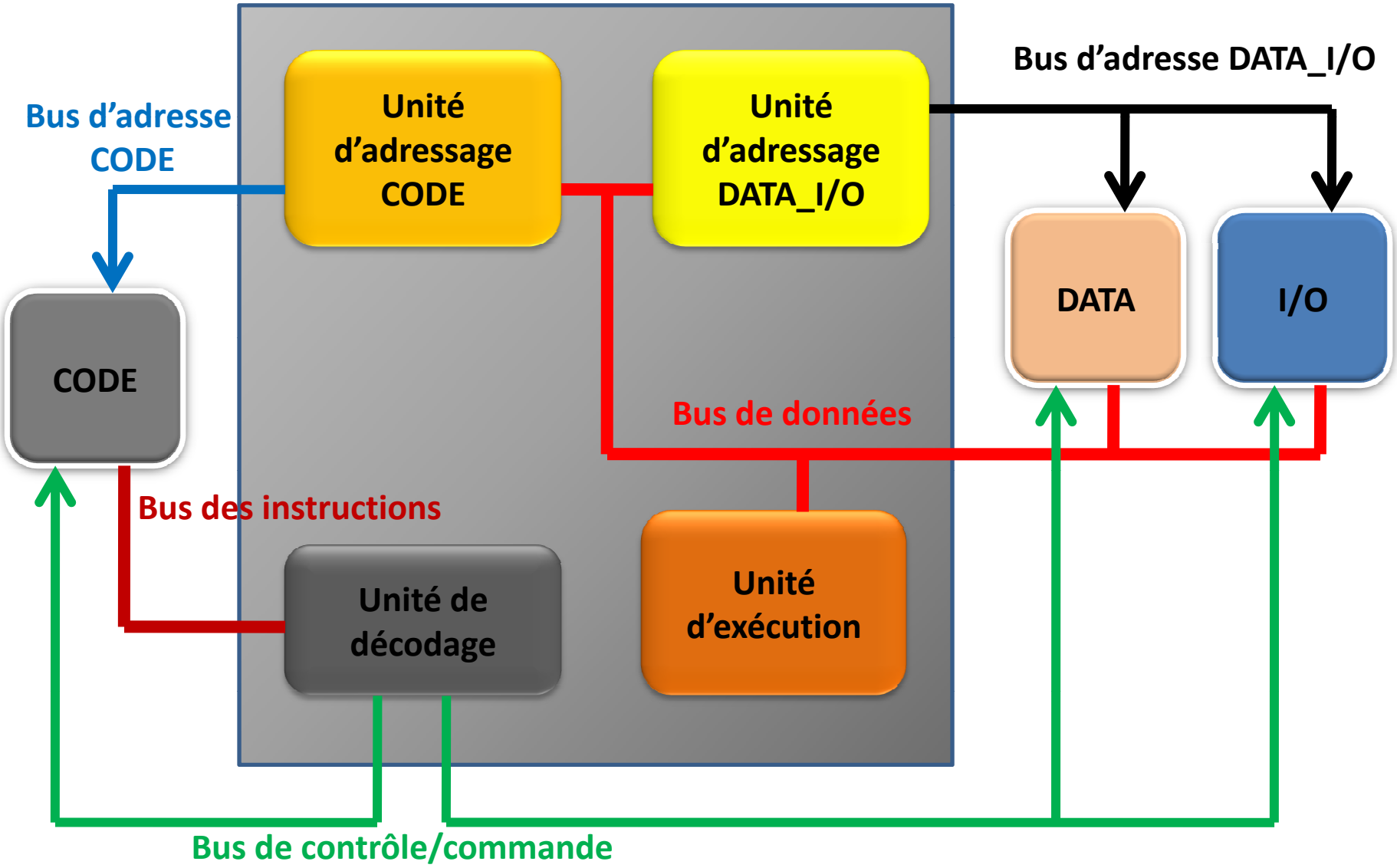
Le cas général

- Le cas de **Von-Neumann :**



Le cas général

- Le cas de **Harvard :**



Le cas général

RISC

- Reduced Instruction Set Computer.
- Instruction de taille statique
- Nombre d'instructions < 130
- Facile à comprendre
- Code plain de tournures
- Développement au niveau bas (Assembleur et un C adapté au µC).

CISC

- Complex Instruction Set Computer
- Instruction de taille dynamique
- Nombre d'instruction >130
- Difficile à comprendre
- Code simple
- Développement au besoin (ASM/ C/ PASCAL/ MATLAB…).

Le cas général

- Exemple de codage CISC vs RISC:
- CISC.
  - ADD 04h, MAT[R2+R3]
- RISC.
  - MV @MAT, R1
  - ADD R1, R2, R1
  - ADD R1, R3, R1
  - LD *R1, R4
  - ADD 04h, R4, R4
  - ST R4, *R1

Le cas général

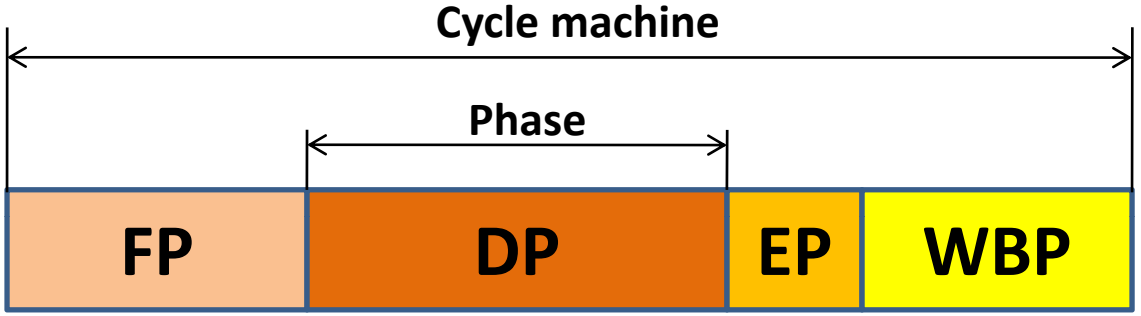
- Jusqu'à présent, **tout va bien.**

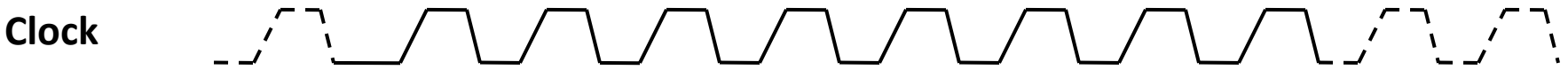
- Mais, j'ai une petite question à posé :

- **Quel est le mécanisme suivi par l'unité de décodage pour exécuter une instruction?????**

Le cas général

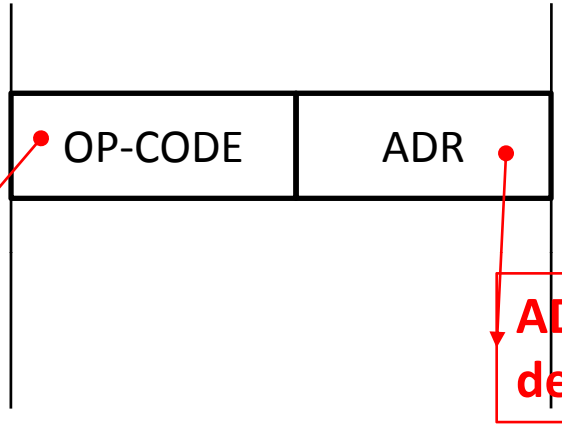
- L'unité de décodage fait un ensemble de phases pour exécuter une instruction. Ces phases sont :
  1. La recherche de l'instruction qu'on va exécuter (**FETCH PHASE -FP-**).
  2. Décodage et lecture des opération (**DECODE PHASE -DP-**).
  3. Réalisation de l'opération demandée par l'instruction (**EXECUTE PHASE -EP-**).
  4. Sauvegarde de résultat dans la destination (**WRITE-BACK PHASE -WBP-**).
  - On appel un **cycle machine** est le temps nécessaire pour exécuter une instruction, donc :
  - **Cycle machine = FP + DP + EP + WBP**.
  - Chaque phase peut prendre un ou plusieurs top d'horloge.





Le cas général

- Exemple pratique : on désire de réaliser l'opération suivante : **M(#3) ← M(#3) + R0** (addition entre la case mémoire d'adresse 3 dans la zone DATA et le registre interne R0 et le résultat sera chargé dans la même case mémoire).
- La première étape est la traduction de l'opération à une instruction en mnémonique comme : **ADD M(#3), R0**.
- La deuxième étape est le codage de l'instruction. Par exemple :

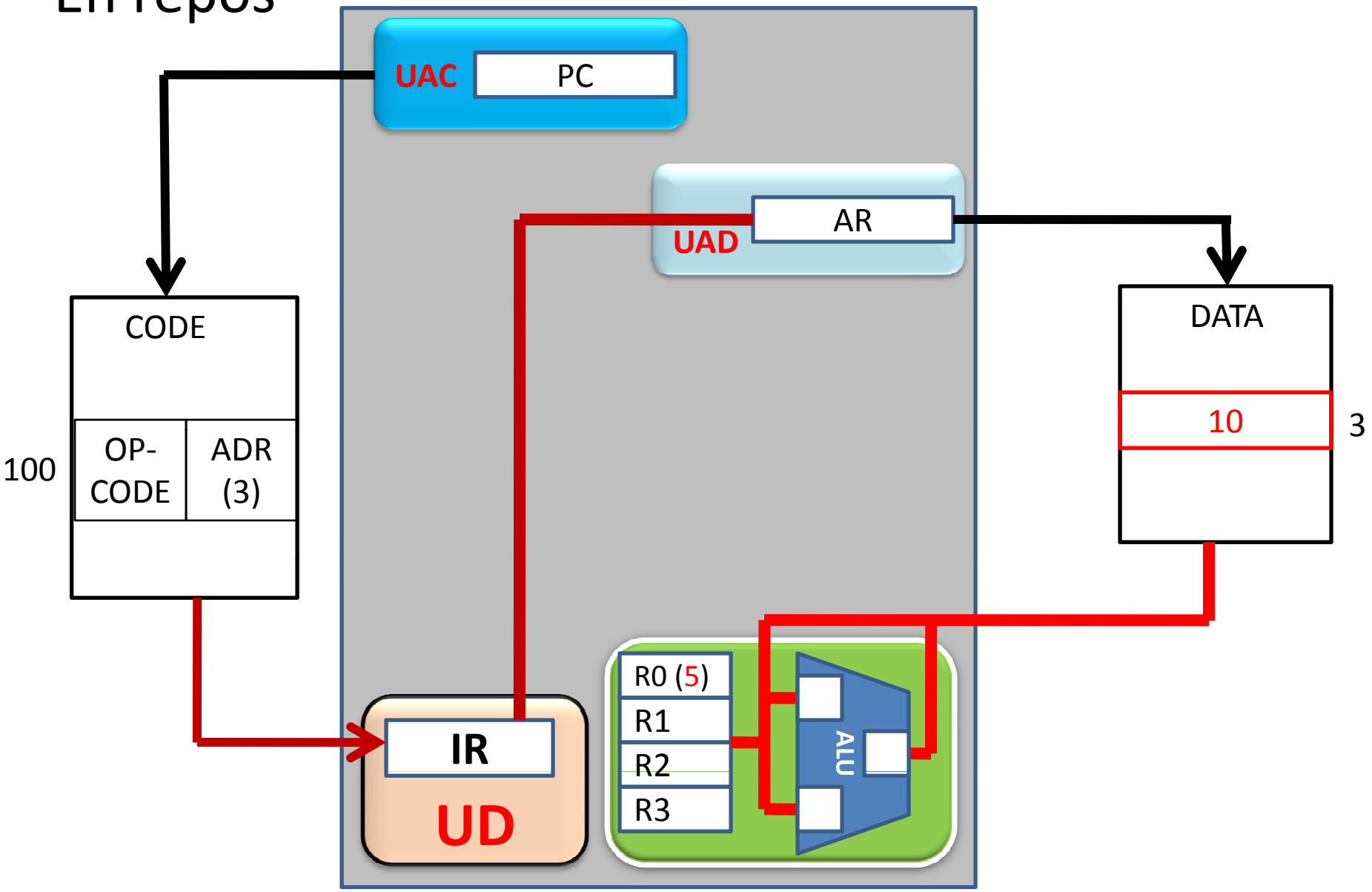


OP-CODE : présente le codage de l'instruction d'addition entre une case mem avec le registre R0

ADR : présente l'adresse de la case mémoire

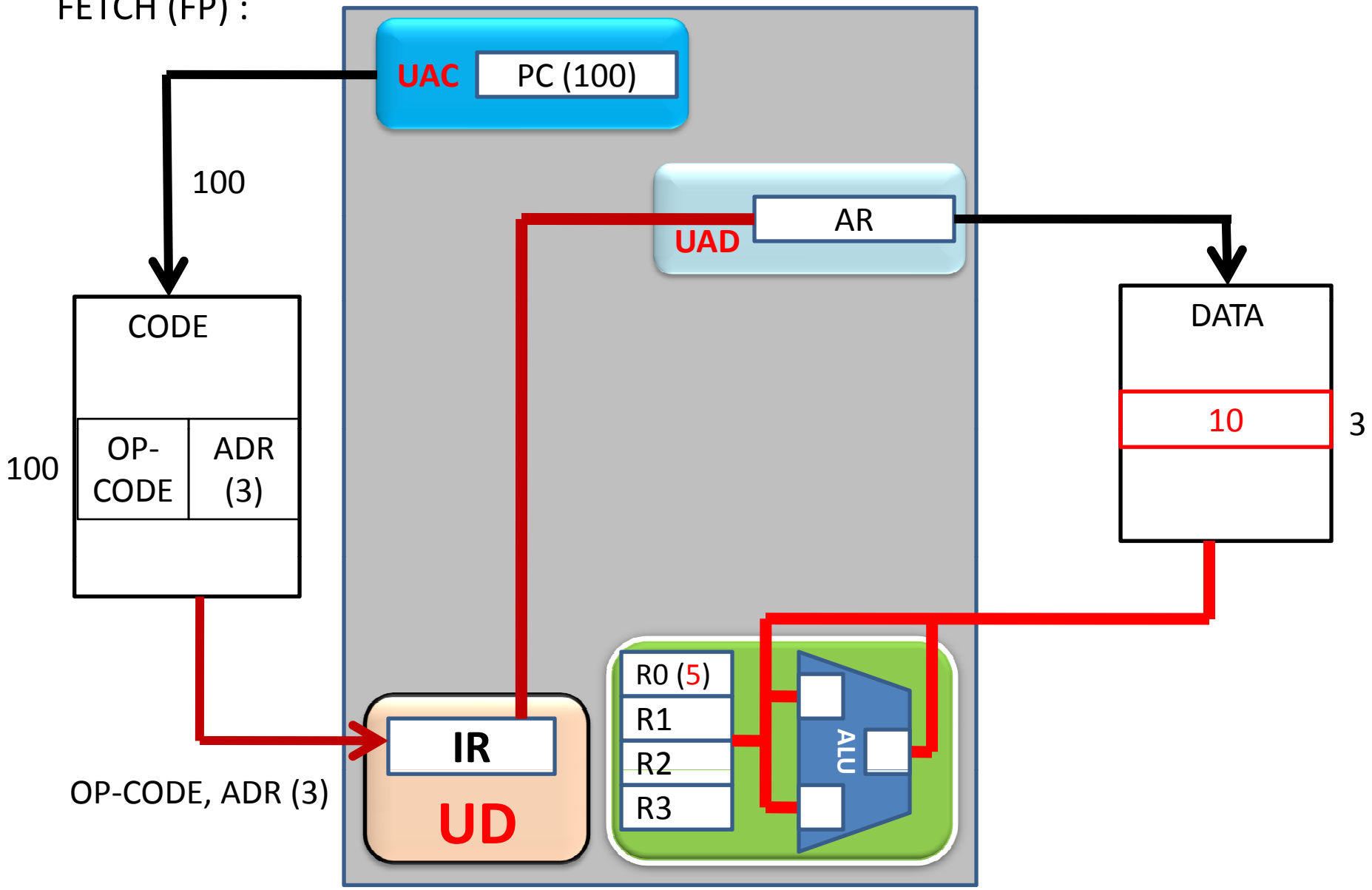
Le cas général

- En repos



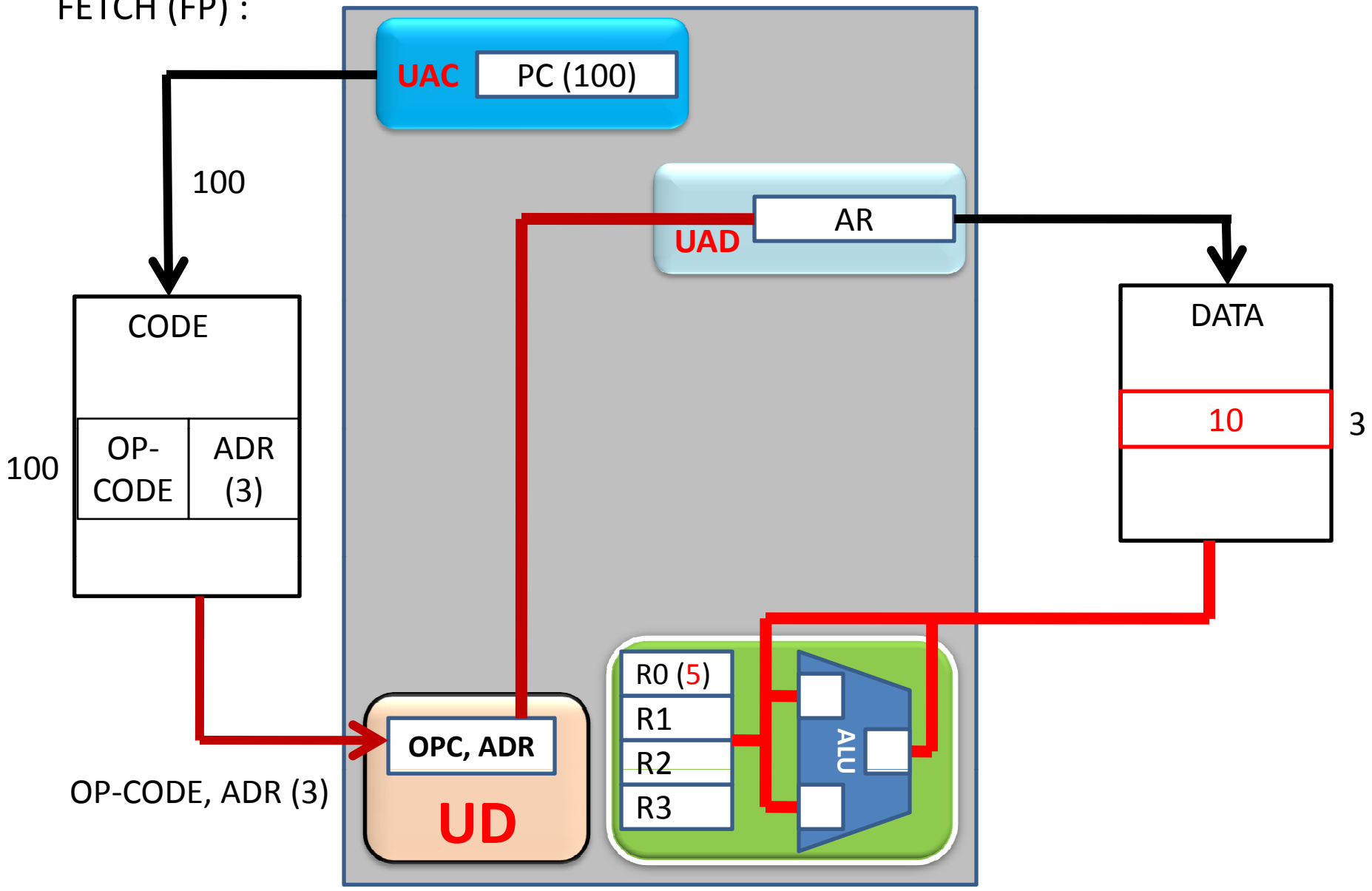
Le cas général

- La phase FETCH (FP) :



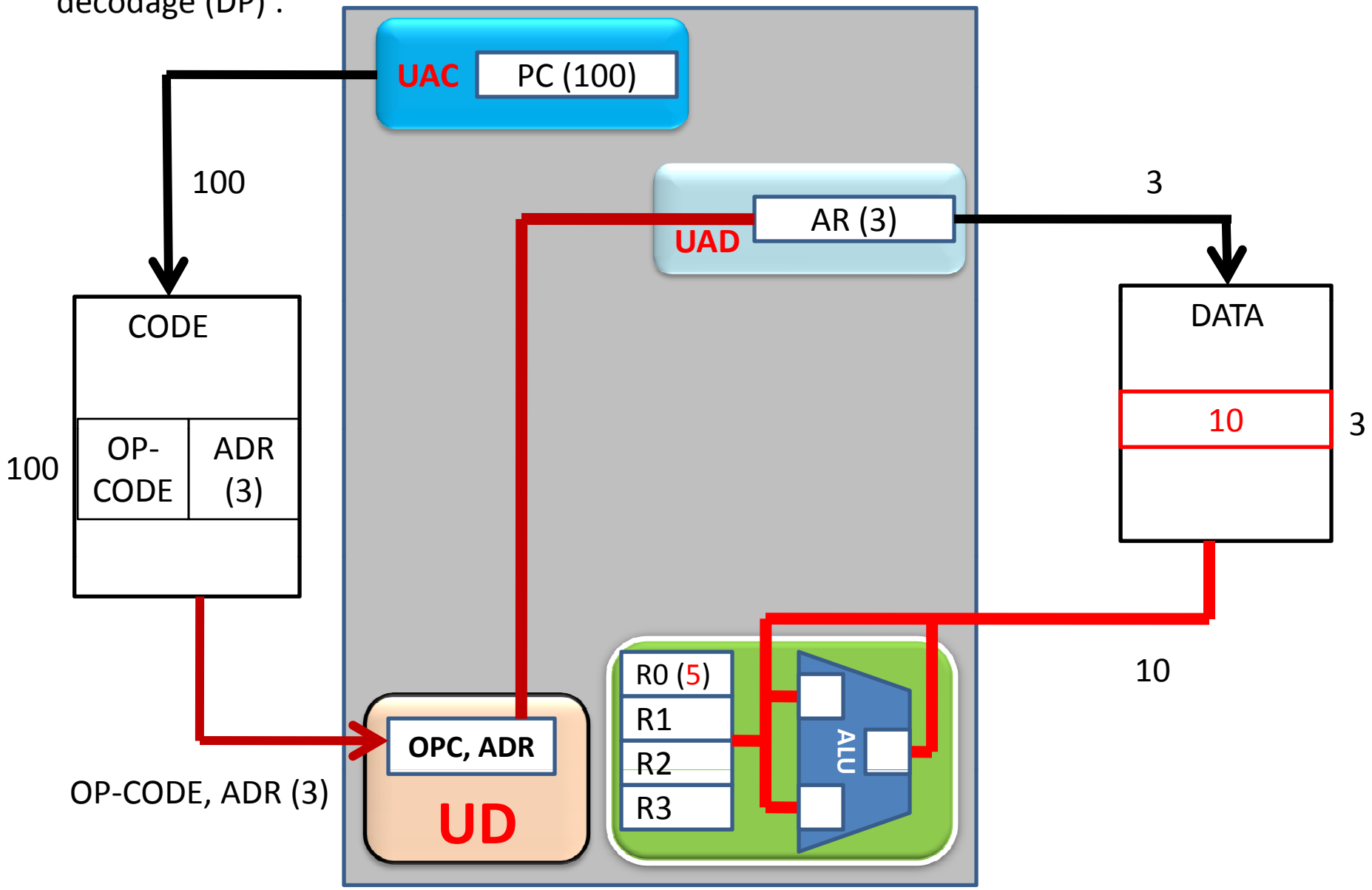
Le cas général

- La phase FETCH (FP) :



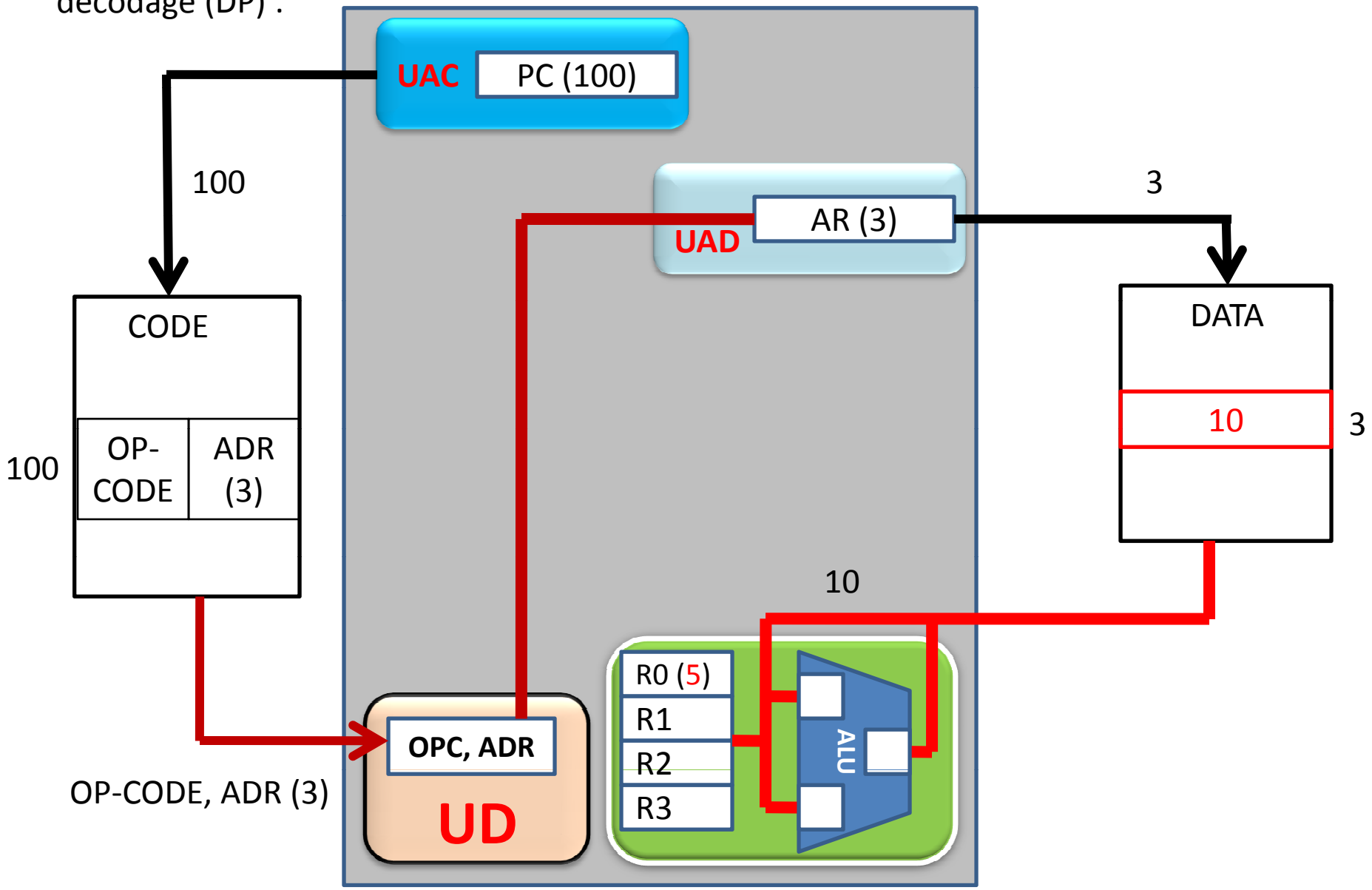
Le cas général

- La phase décodage (DP) :



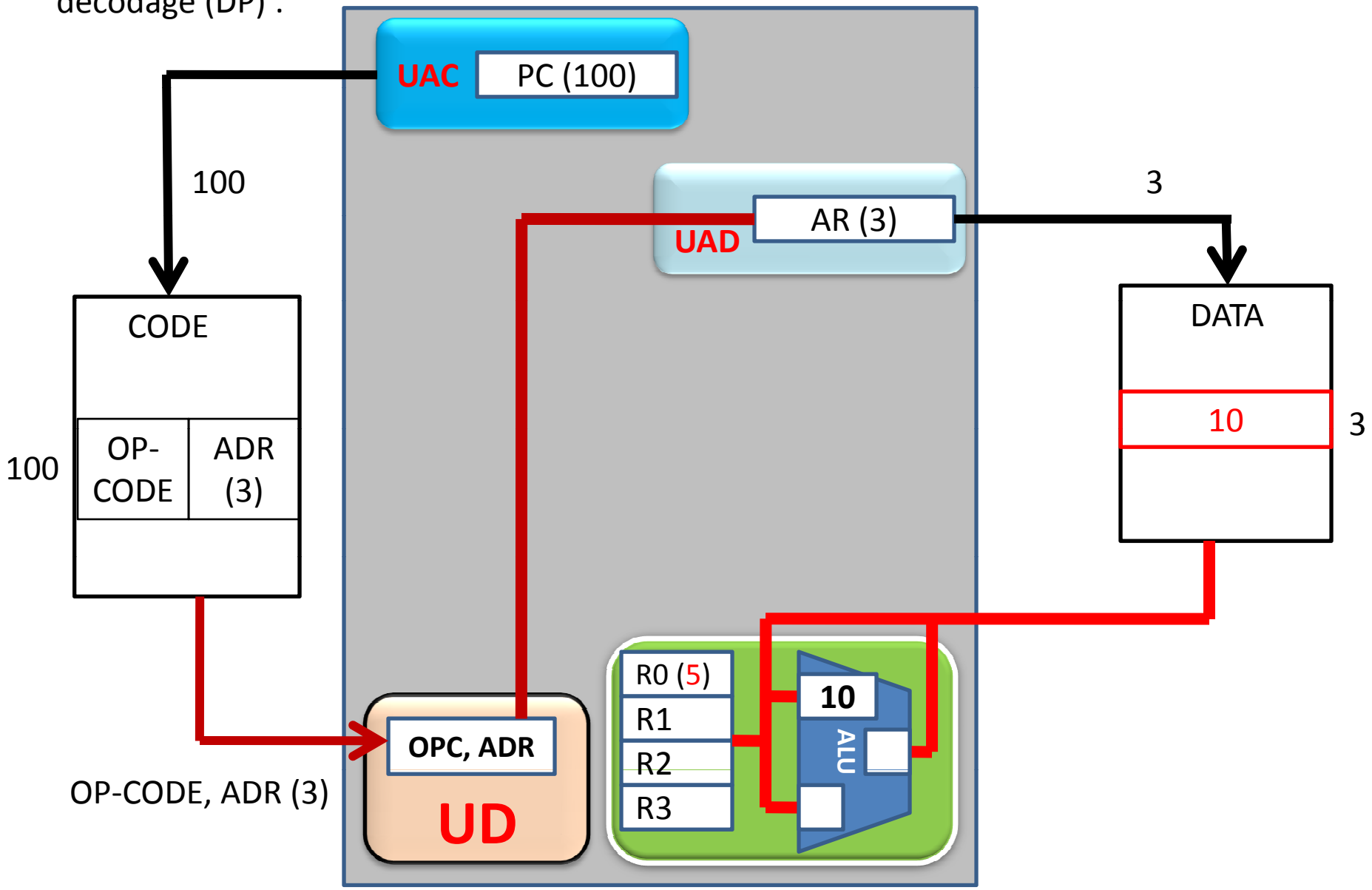
Le cas général

- La phase décodage (DP) :



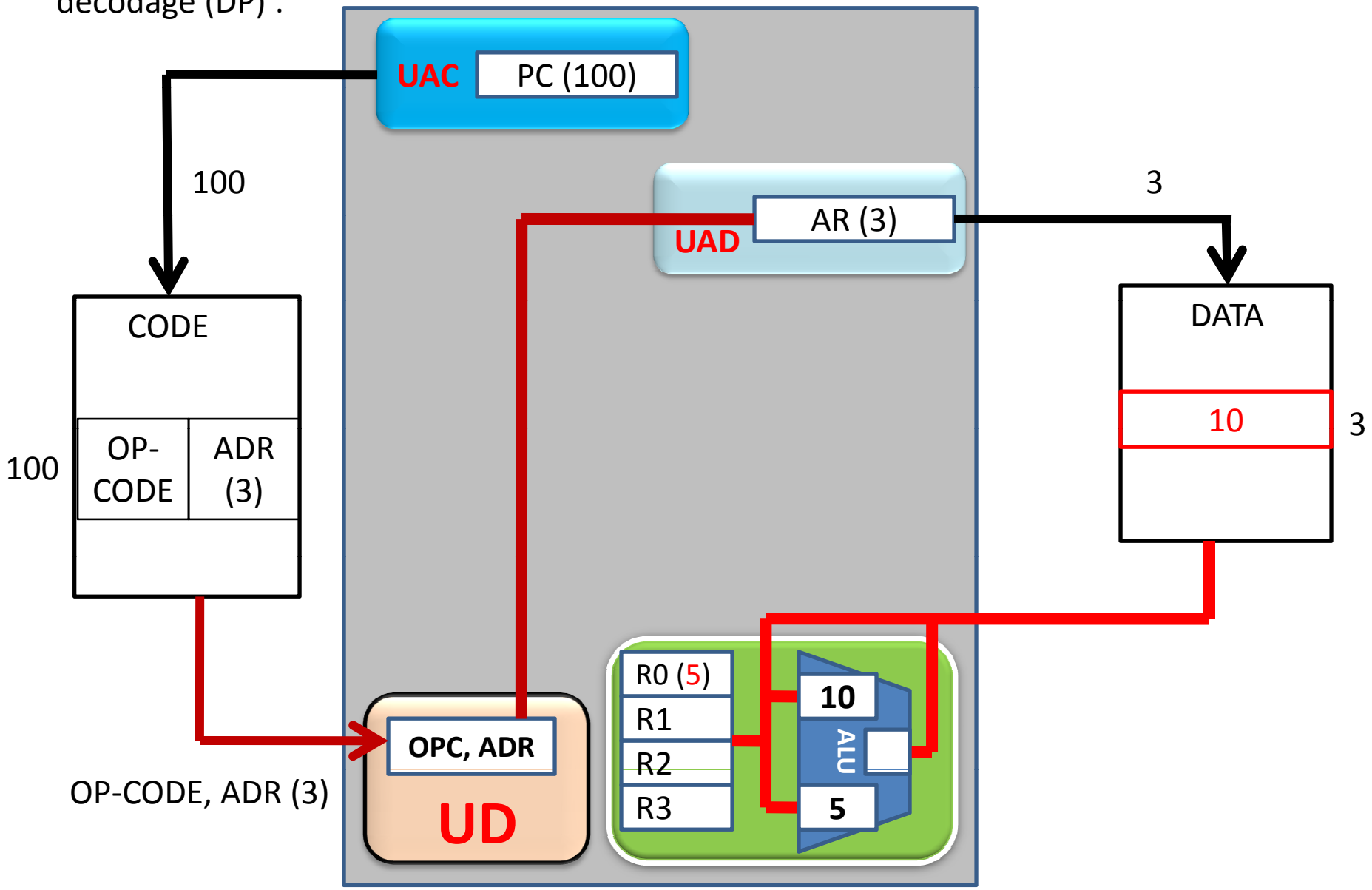
Le cas général

- La phase décodage (DP) :



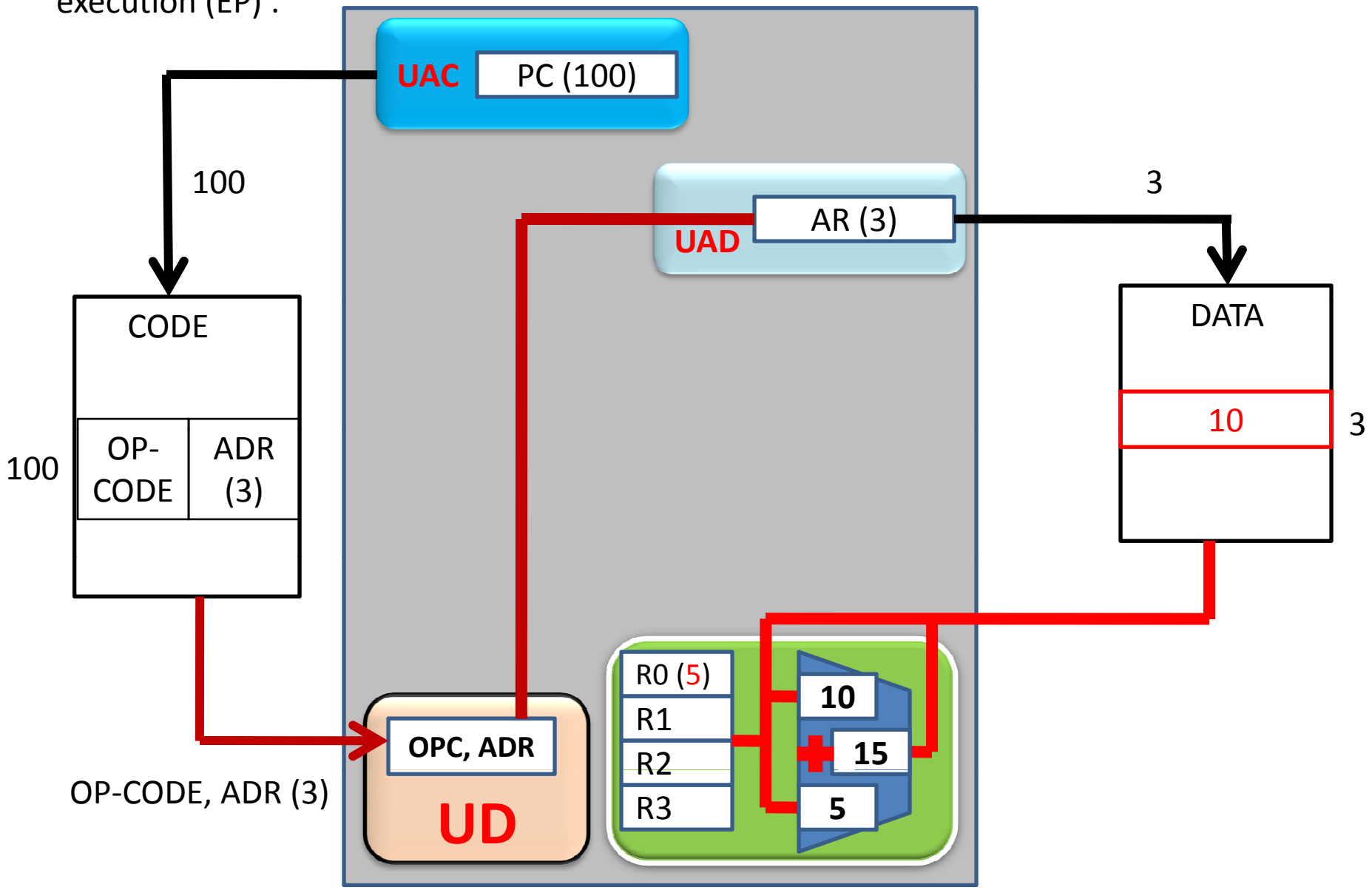
Le cas général

- La phase décodage (DP) :



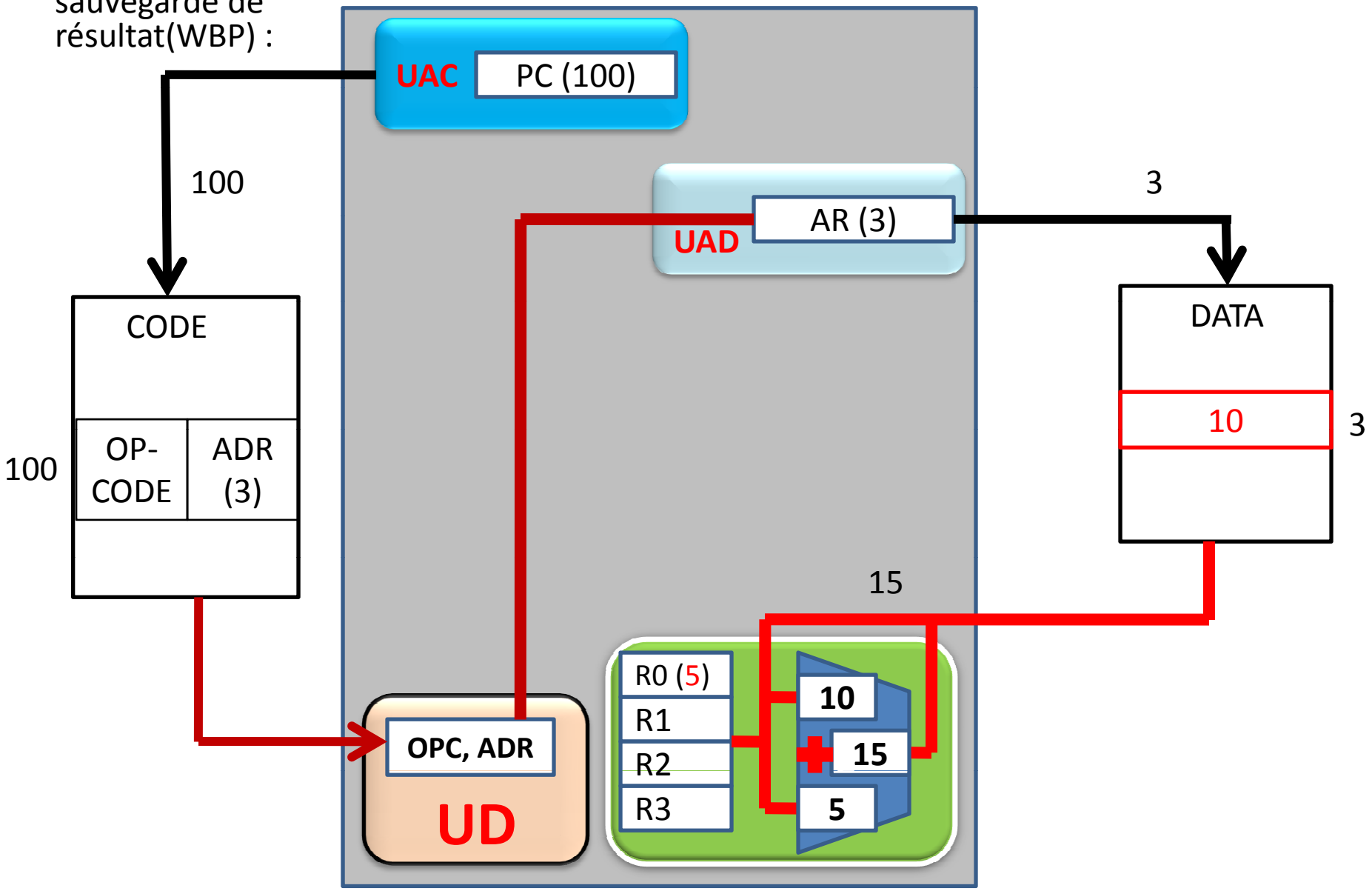
Le cas général

- La phase exécution (EP) :



Le cas général

- La phase sauvegarde de résultat(WBP) :



Le cas général

- La phase sauvegarde de résultat(WBP) :