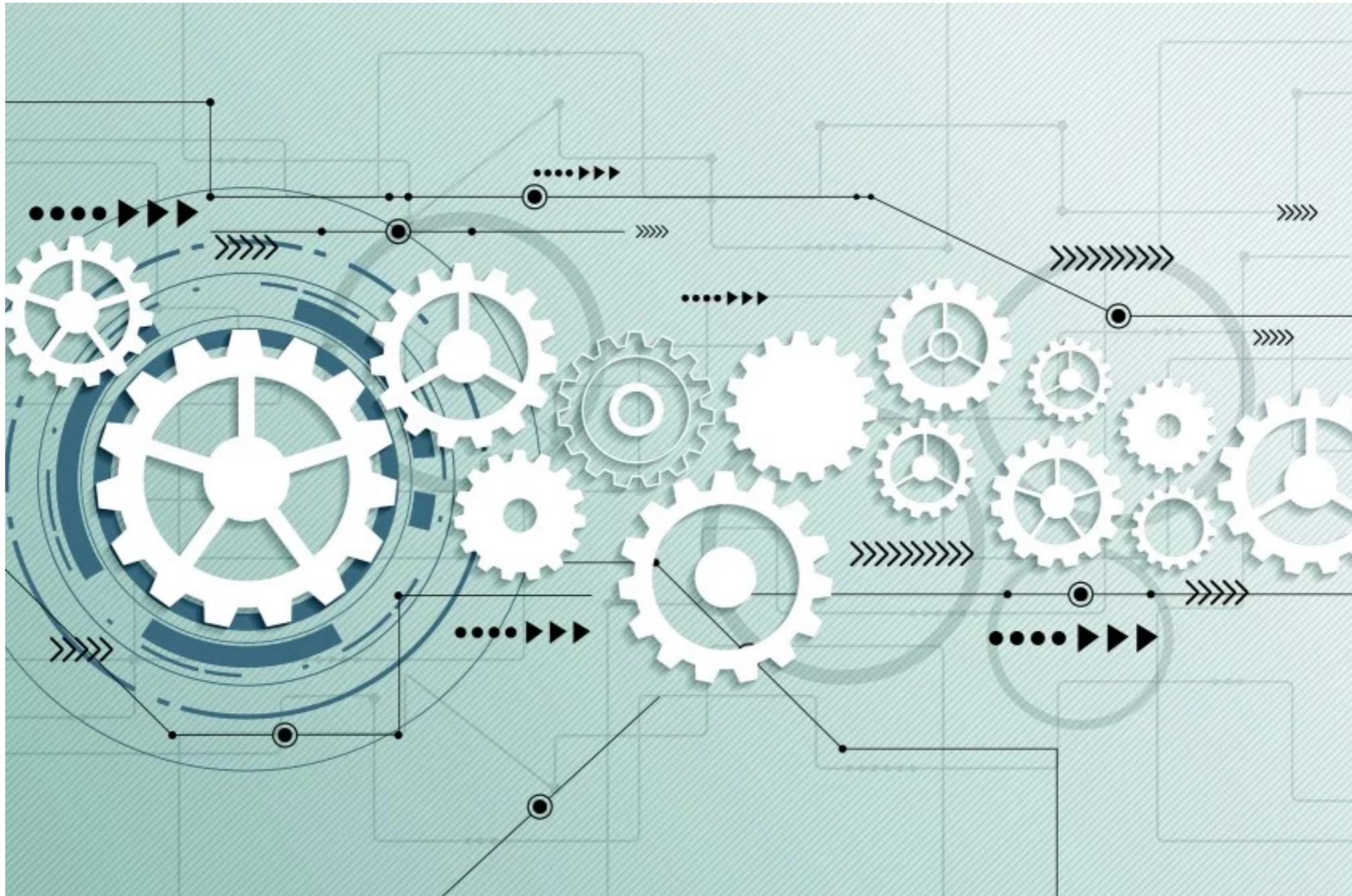


Introduction à l'assembleur ARM: variables et assembleur



GIF-1001 Ordinateurs: Structure et Applications, Hiver 2018
Jean-François Lalonde

Accès mémoire avec variables

- Les instructions LDR et STR sont utilisées avec la syntaxe suivante pour accéder aux variables:

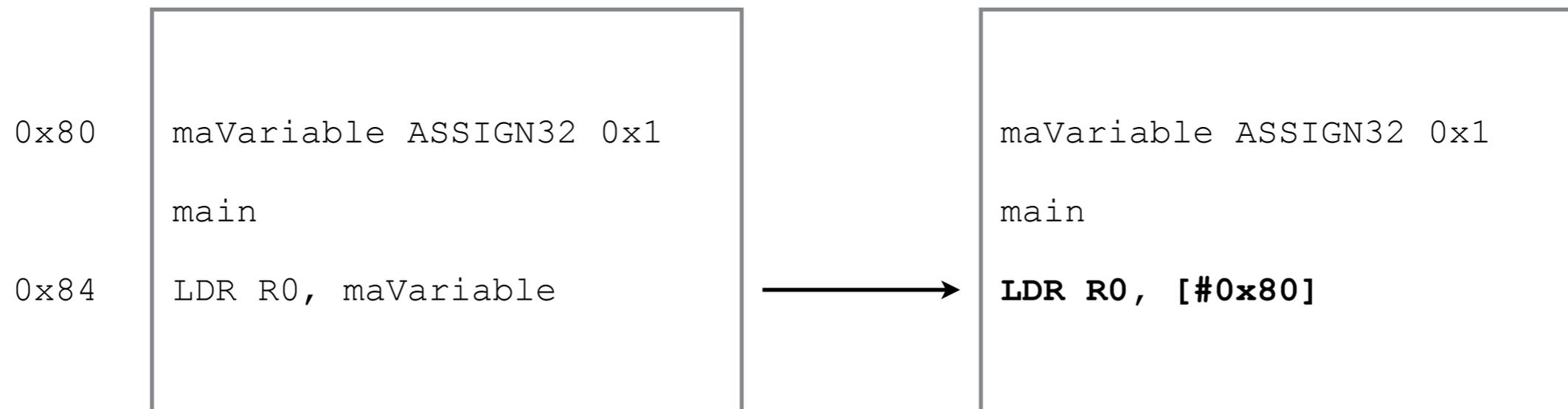
```
LDR Rd, maVariable    ;Met la valeur de la variable dans Rd  
LDR Rd, =maVariable  ;Met l'adresse de la variable dans Rd
```

- L'assembleur doit traduire chacune de ces lignes en *une seule* instruction du processeur
- Comment fait-il?

Accès mémoire avec variables

Par quoi l'assembleur remplace-t-il `LDR R0, maVariable`?

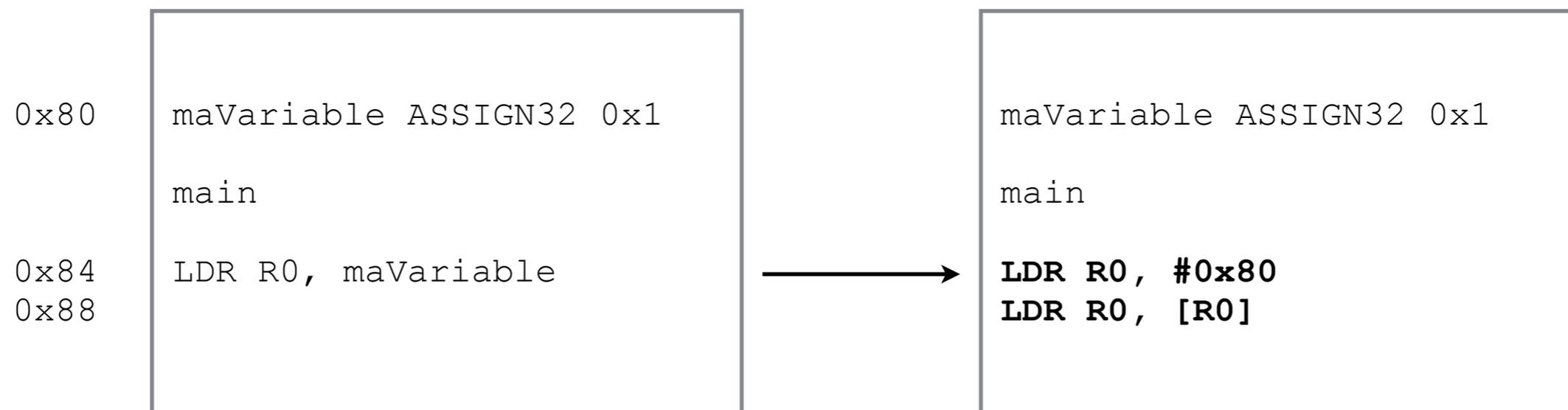
- L'assembleur connaît déjà l'adresse de `maVariable` (`0x80`)
Essayons d'aller lire le contenu mémoire à cette adresse:



- Question:
 - pouvons-nous effectuer cette opération en ARM?

Question

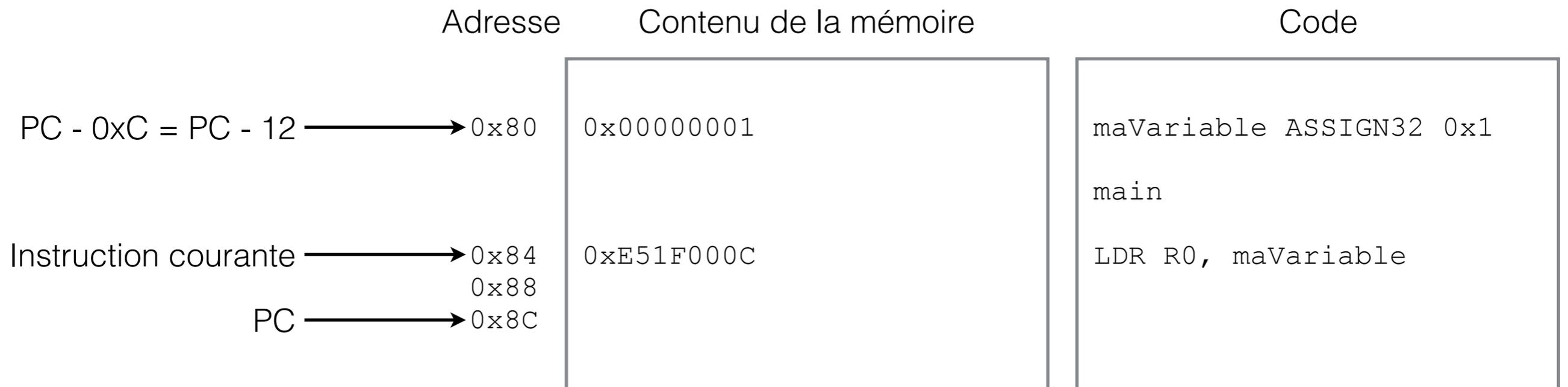
- Pouvons-nous effectuer cette opération en ARM?
 - NON! LDR demande toujours un registre de base...
 - Mais pour avoir un registre de base, il nous faudrait 2 instructions



- Solution?
 - PC a une valeur connue (instruction courante + 8)!
 - Utiliser une instruction relative à PC!

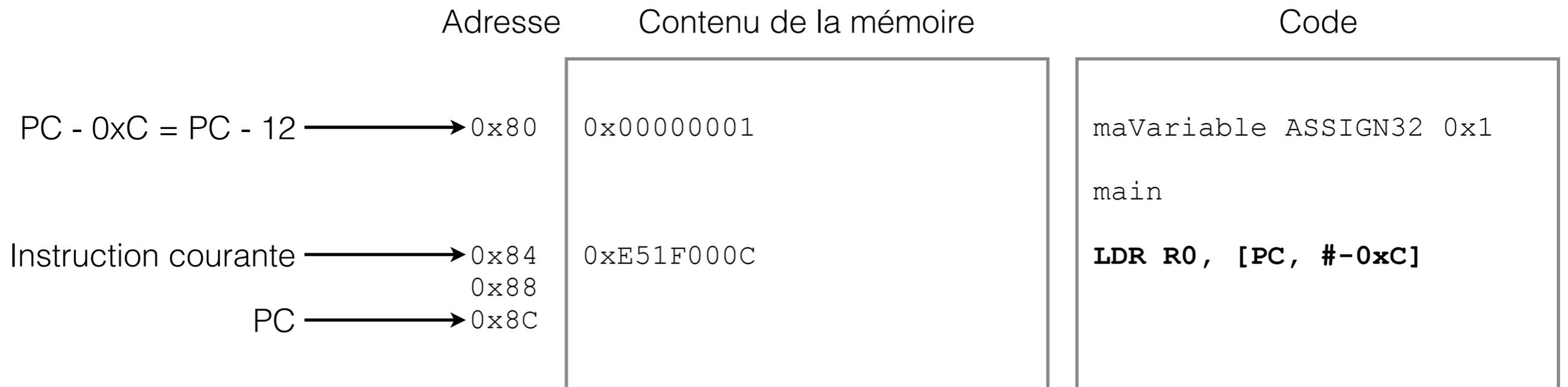
Calculs sur PC

Par quoi l'assembleur remplace-t-il `LDR R0, maVariable`?



Calculs sur PC

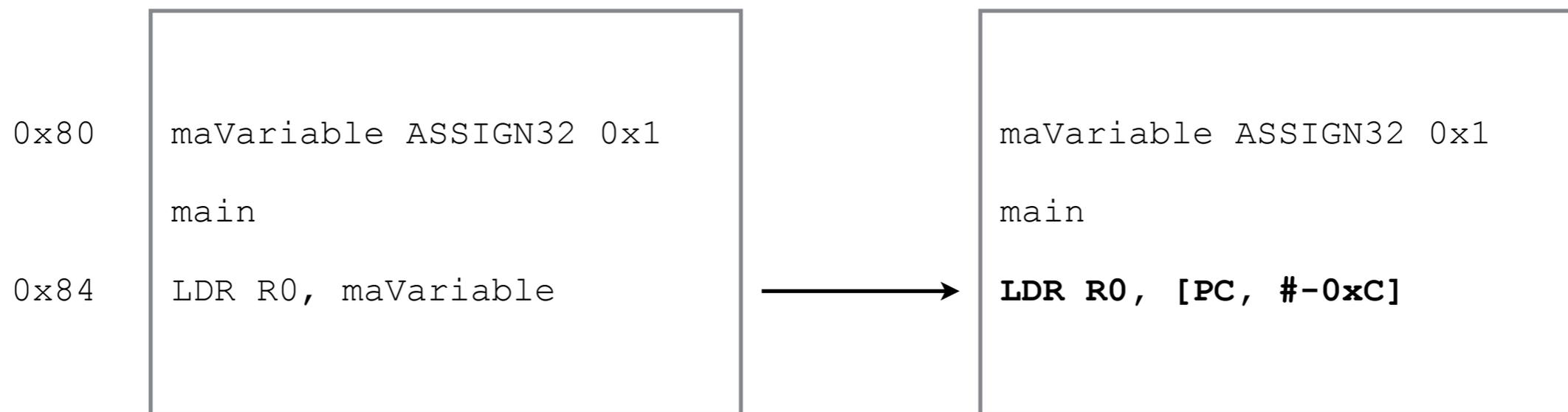
Par quoi l'assembleur remplace-t-il `LDR R0, maVariable`?



Accès mémoire avec variables

Par quoi l'assembleur remplace-t-il `LDR R0, maVariable`?

- Remplaçons par un déplacement *relatif à PC*. Déplacement de combien?
 - L'instruction est à l'adresse `0x84`. Cela veut dire que PC contient `0x8C`.
 - La variable est à l'adresse `0x80`.
 - Nous voudrions que $PC + \text{déplacement} = 0x80$.
 - Donc, $\text{déplacement} = 0x80 - PC = 0x80 - 0x8C = -0xC$.

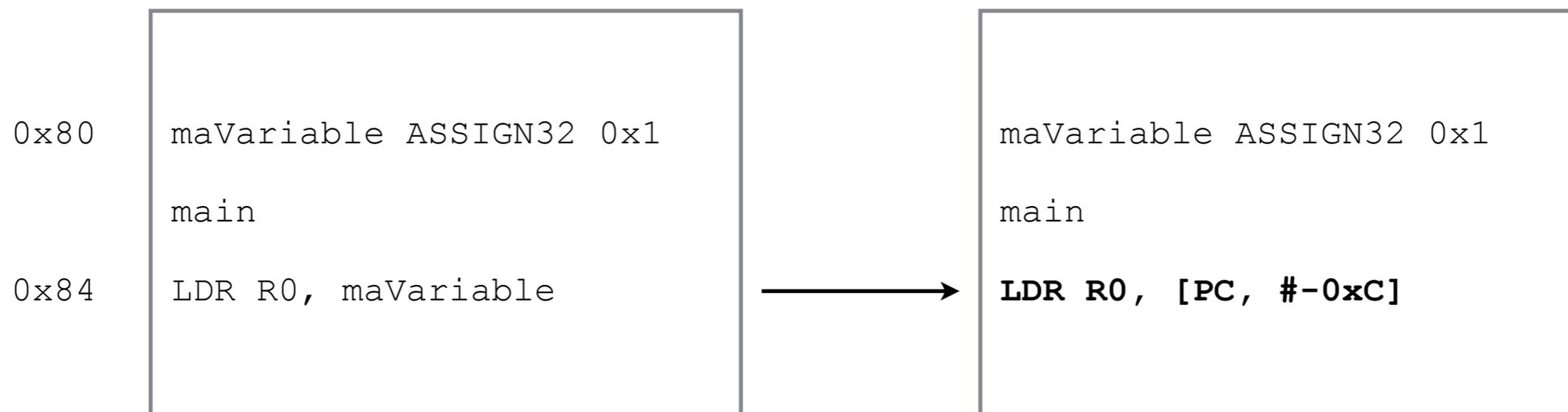


Accès mémoire avec variables

Par quoi l'assembleur remplace-t-il `LDR R0, maVariable`?

- Remplaçons par un déplacement *relatif à PC*. Déplacement combien?
 - L'instruction est à l'adresse `0x84`. Cela veut dire
 - La variable est à l'adresse `0x80`.
 - Nous voudrions que $PC + \text{déplacement} = 0x80$.
 - Donc, $\text{déplacement} = 0x80 - PC = 0x80 - 0x8C = -0xC$.

Attention!
Le déplacement maximal permis est de 4096 octets (`0x1000`).



Démonstration

(Accès mémoire avec variable (1))

Adresse de la variable

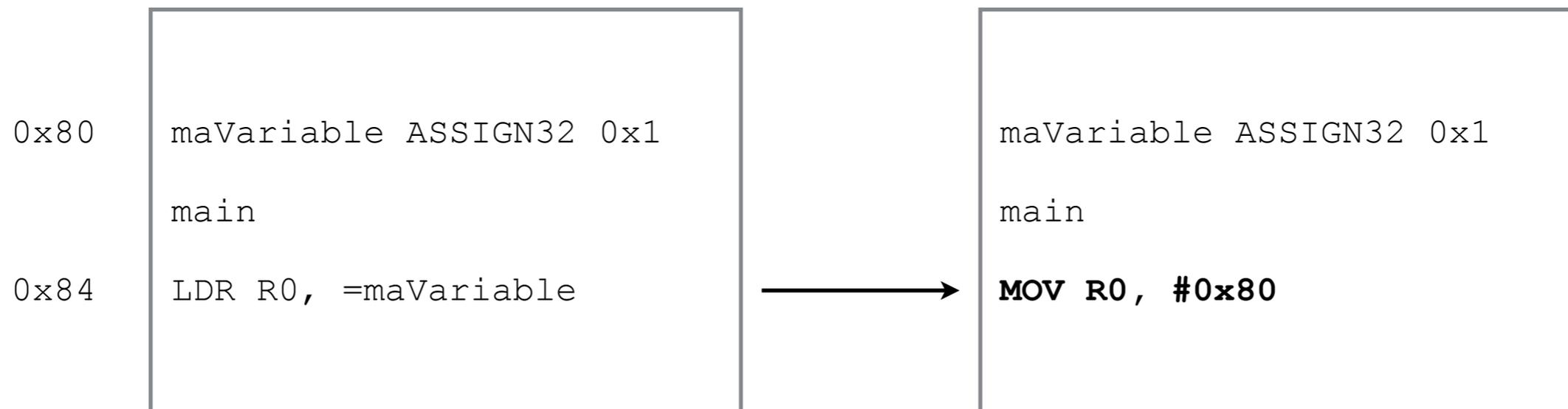
Adresses	Code	Mémoire
0x0	SECTION INTVEC B main	0xEA00001F
0x80	SECTION CODE ; Assignons une variable (adresse 0x80) maVariable ASSIGN32 0x1	0x00000001
0x84	main LDR R0, =maVariable	0xE51F000C
0x88	SECTION DATA	0x00000080

Par quelle instruction l'assembleur remplace-t-il
LDR R0, =maVariable?

Accès mémoire avec variables

Par quoi l'assembleur remplace-t-il `LDR R0, =maVariable`?

- L'assembleur connaît déjà l'adresse de `maVariable` (`0x80`). Essayons de charger l'adresse dans `R0` avec `MOV`:



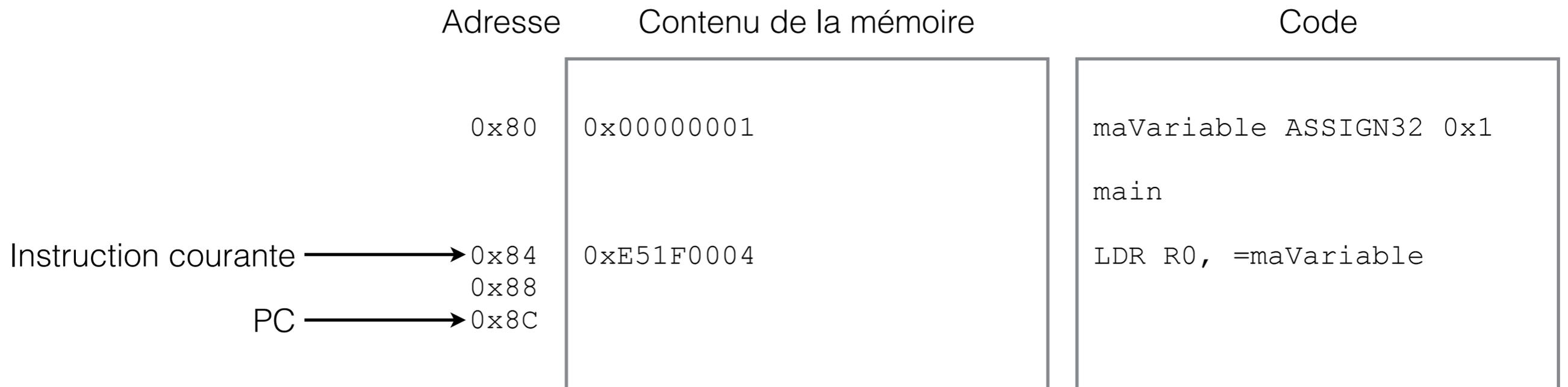
- Question:
 - Combien de bits (en général) avons-nous besoin pour représenter une adresse?

Question

- Combien de bits avons-nous besoin pour représenter une adresse?
 - 32
 - C'est le même nombre que pour l'instruction elle-même... impossible de mettre une adresse de 32 bits dans une instruction de 32 bits!
 - Solution? Utiliser un déplacement par rapport à PC. Nous aurons besoin de moins de 32 bits pour encoder ce déplacement...
- Cependant... où se trouve l'adresse de maVariable?
 - Nulle part! Il faut que l'assembleur l'écrive à qq part en mémoire
- Où peut-il l'écrire?

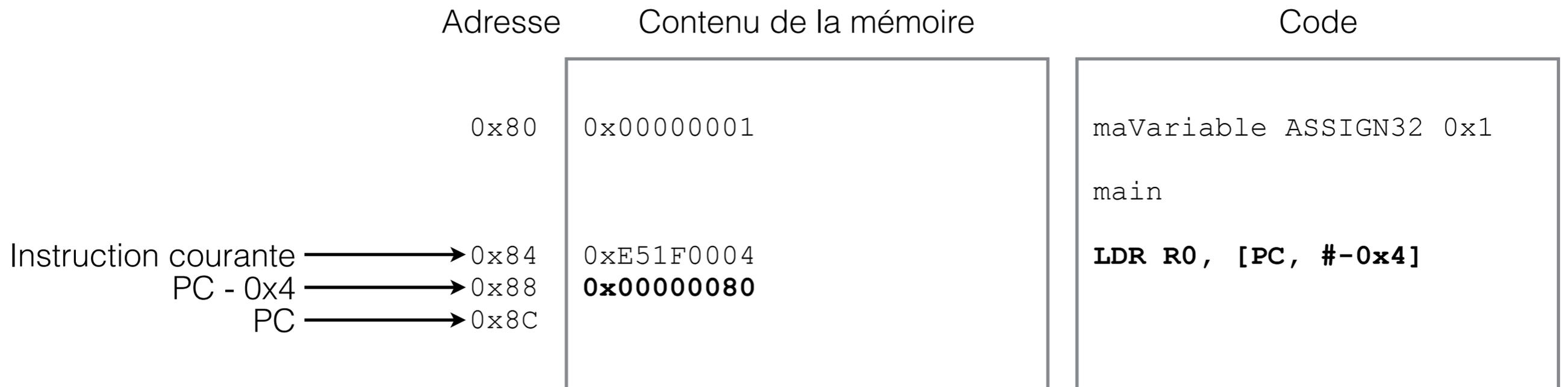
Calculs sur PC

Par quoi l'assembleur remplace-t-il `LDR R0, =maVariable`?



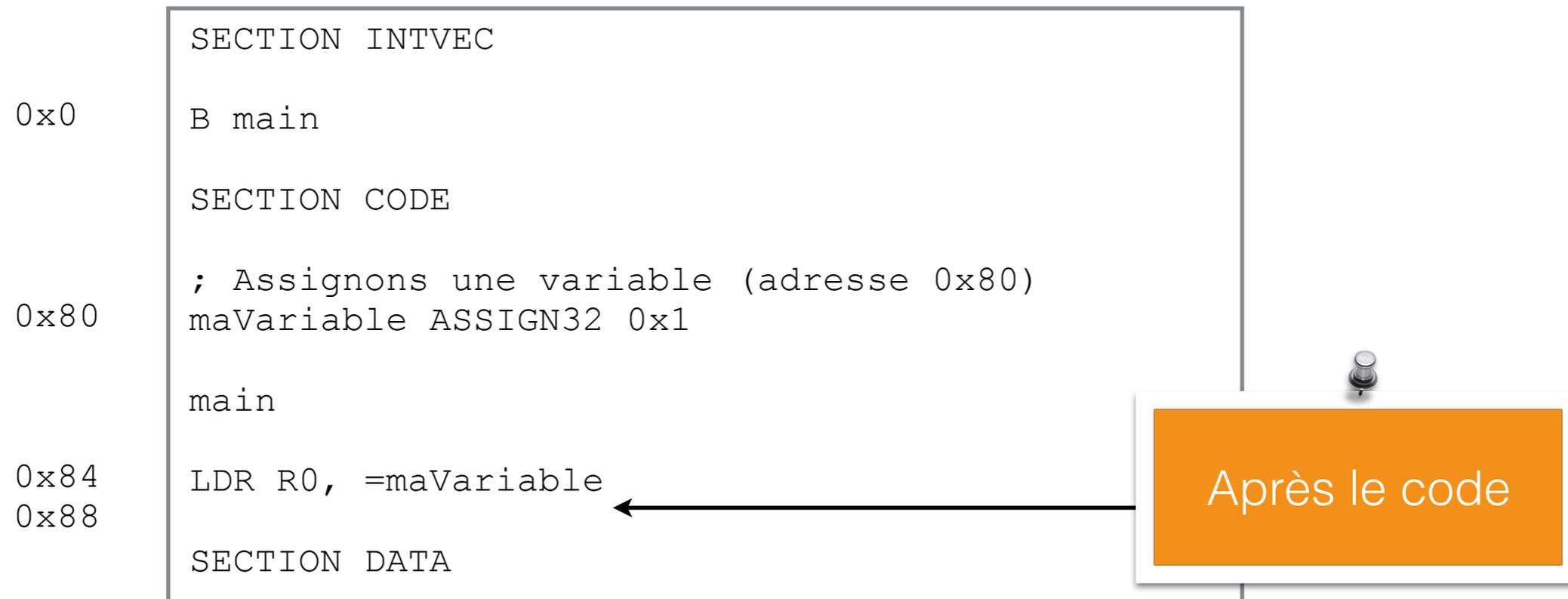
Calculs sur PC

Par quoi l'assembleur remplace-t-il `LDR R0, =maVariable`?



Où écrire l'adresse...

Où l'assembleur peut-il écrire l'adresse de maVariable?

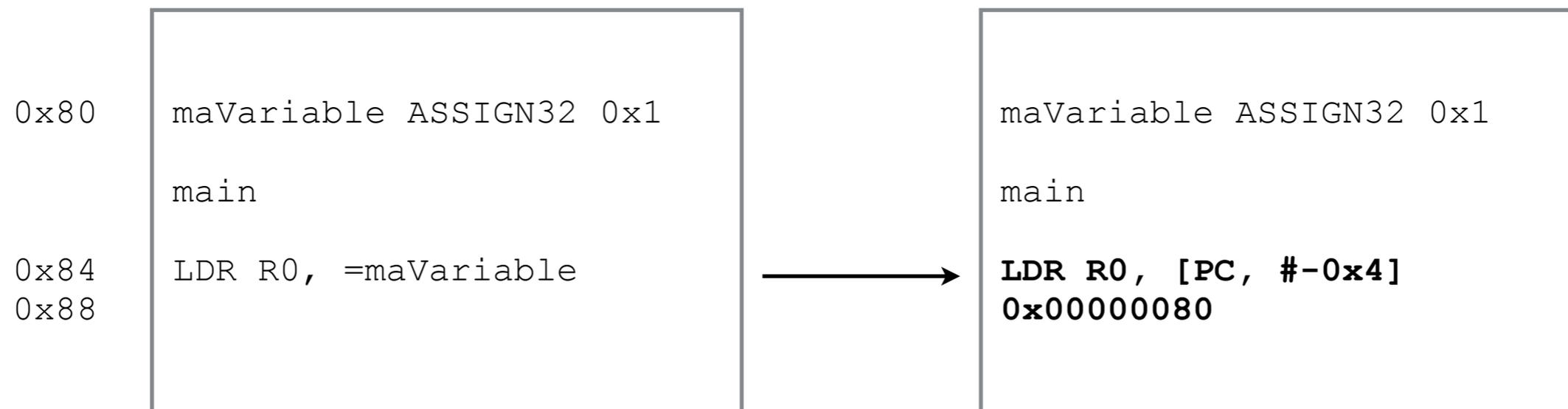


- Typiquement, l'assembleur rajoute l'adresse de la variable *après* le code.

Accès mémoire avec variables

Par quoi l'assembleur remplace-t-il `LDR R0, maVariable`?

- Plaçons l'adresse de la variable *après* le code, soit à l'adresse 0x88.
- Remplaçons par un déplacement *relatif* à PC. Déplacement de combien?
 - L'instruction est à l'adresse 0x80. Cela veut dire que PC contient 0x88.
 - L'adresse de variable est à l'adresse 0x84.
 - Nous voudrions que $PC + \text{déplacement} = 0x84$.
 - Donc, $\text{déplacement} = 0x84 - PC = 0x84 - 0x88 = -0x4$



Démonstration

(Accès mémoire avec variable (2))