# TD 11
# Modification d'un processeur

L'objectif de ce TD est de montrer les conséquences de l'ajout d'instructions sur la micro-architecture d'un processeur et sa machine d'état. Le processeur utilisé est le même que dans les séances précédentes.

***Méthodologie***

## Ex. 1 : Chargement immédiat : intérêt et modification sur le processeur

Pour simplifier l'accès aux constantes dans un programme, on propose d'ajouter une instruction qui permet de charger un registre (8 bits) avec une constante (8 bits également). Cette nouvelle instruction sera notée `li IMM, rd`. Généralement nommée *load immediate* en anglais, elle effectue l'affectation suivante : $rd \leftarrow IMM$.

**Question 1** Indiquer où doit se trouver la constante à charger dans le registre et proposer un encodage de l'instruction `li` qui s'insère simplement dans l'encodage existant (voir le TD 10 pour le jeu d'instruction).

**Question 2** Est-il nécessaire de modifier la partie opérative (voir le TD 8) ? Si oui, effectuer les modifications. Si non, justifier votre réponse.

**Question 3** Étendre la machine d'état de la PC obtenue au TD 9 de manière à supporter cette nouvelle instruction.

## Ex. 2 : Branchement : intérêt et modification sur le processeur

Le programme ci-dessous calcule le produit de la constante `X` (située à l'adresse 0x2000) et la constante `Y` (0x2001) et stocke le résultat à l'adresse 0x1230.
Pour pouvoir exécuter ce programme sur notre processeur, on a besoin de rajouter des instructions de branchement :

- `jmp @mem` fait « sauter » l'exécution à l'adresse `@mem` (*i.e.* `PC := @mem`) ;
- `jz @mem` provoque un branchement à l'adresse `@mem` ssi $Z = 1$ (où $Z$ est l'indicateur de l'UAL valant 1 ssi le résultat de la dernière opération était nul).

```
Cpt := Y;
Res := 0;
while Cpt /= 0 loop
  Res := Res + X;
  Cpt := Cpt - 1;
end loop;
```

```
0x4000: ld 0x2000, r0
0x4003: ld 0x2001, r1
0x4006: li 1, r2
0x4008: xor r3, r3
0x4009: or r1, r1
0x400A: jz 0x____
0x400D: add r0, r3
0x400E: sub r2, r1
0x400F: jmp 0x____
0x4012: st r3, 0x1230
```

**Question 1** Associer les lignes de l'algorithme écrit en pseudo-code (ci-dessus à gauche) à

celles du programme écrit en langage machine (ci-dessus à droite), justifier les adresses des instructions et compléter les champs manquants après les instructions de branchement.

**Question 2** Que faut-il ajouter à la partie opérative de notre processeur pour supporter ces nouvelles instructions ?

**Question 3** Modifier la PC en conséquence et proposer un codage pour ces instructions.