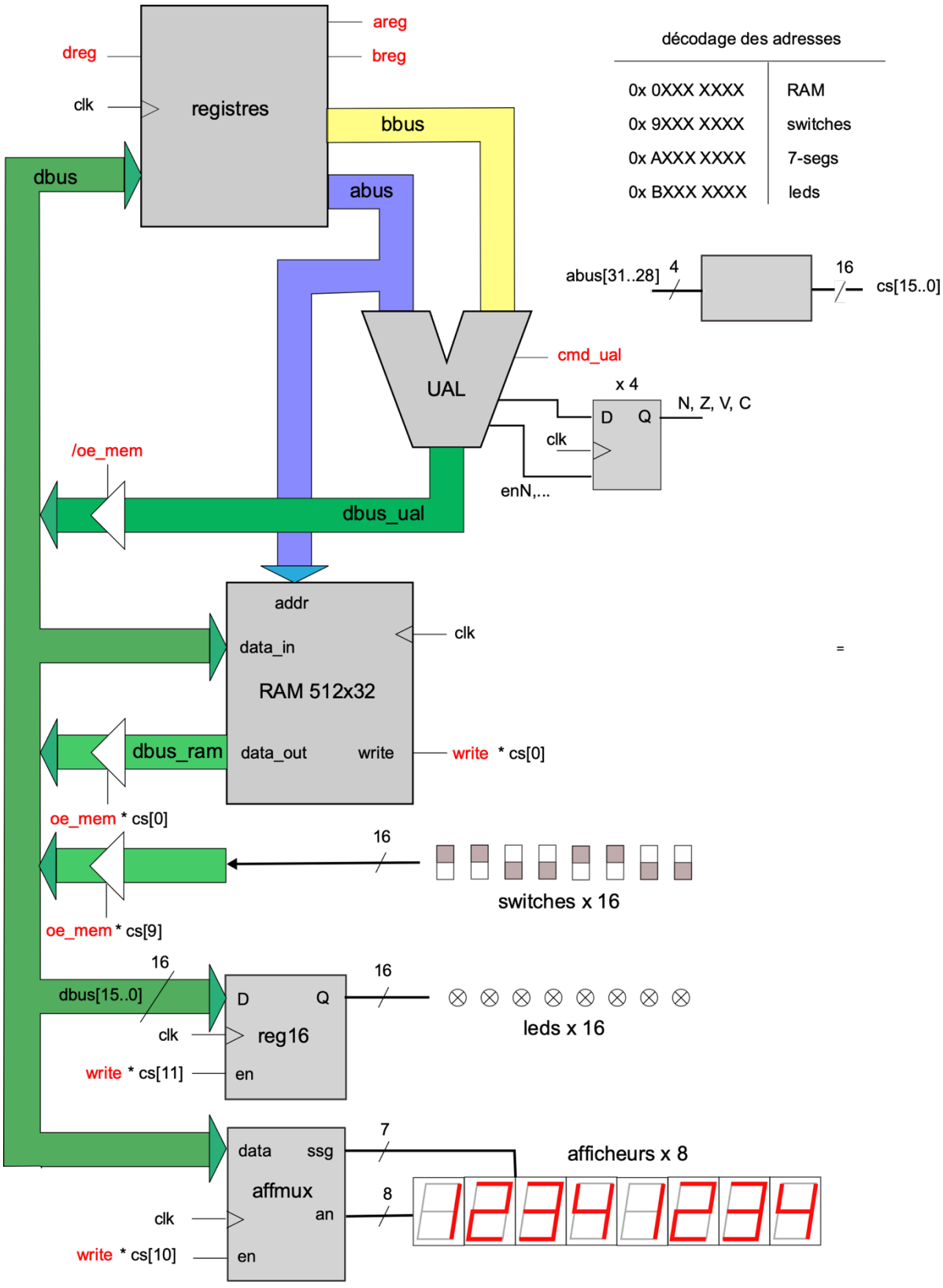
TD3 - Entrées/sorties ; interruptions

1. Entrées/sorties



a) Entrées/sorties mappées en mémoire (15 mn)

Expliquer à partir du schéma de la micromachine comment :

- une lecture avec l'instruction `ld` à l'adresse 0x90000000 copie dans le registre destination l'état des 16 switches (sur les 16 bits de poids faibles)
- une écriture avec l'instruction `st` d'un registre à l'adresse 0xB0000000 projette les 16 bits de poids faibles de cette valeur sur les 16 leds

b) Exercice (15 mn)

Écrire un programme qui ne termine jamais ; à chaque itération il lit la valeur des 16 switches, l'élève au carré, puis affiche le résultat sur les 16 leds.

c) Exercice (15 mn)

Programme analogue au b), mais qui fait la somme entre les 8 bits de poids faibles et les 8 bits de poids forts des switches. (voir instructions *bit-wise :* `and, or, sll, slr`)

2. Interruptions

a) Explications (15 mn)

Une interruption est un front qui arrive sur une ligne distincte (qui ne fait pas partie des lignes d'entrées/sorties). Un processeur dispose généralement de plusieurs lignes d'interruption, chacune étant associée à l'occurrence d'un type d'événement particulier : événement d'un périphérique d'entrée (clavier, souris, etc.), disque dur, réseau, horloge de séquencement.

Une interruption est dite *asynchrone*, car elle peut intervenir à tout moment de l'exécution d'un programme. Lors de son occurrence, un mécanisme spécifique déclenche l'appel à un *sous-programme d'interruption* sans que cela n'affecte l'exécution du programme interrompu.

CRAPS dispose d'une seule ligne d'interruption appelée *IT* ; le mécanisme de prise en compte d'une interruption arrivant sur *IT* est le suivant :

- Un front montant sur *IT* déclenche la mémorisation de l'interruption dans une bascule *pendingIT*
- L'instruction en cours d'exécution est terminée, jusqu'au retour du séquenceur à l'état FETCH
- Lorsque le séquenceur est dans l'état FETCH et que *pendingIT* = 1, le séquenceur ne va pas dans l'état DECODE ; il suit une autre séquence d'états durant laquelle :
  - *pendingIT* est remis à 0
  - la valeur courante de PC est empilée
  - les flags sont empilés
  - PC ← 1
  - Retour à l'état FETCH

Ainsi, l'occurrence d'une interruption provoque un branchement à l'adresse 1. À cette adresse, un *sous-programme d'interruption* doit être implanté, terminé par l'instruction *reti*. L'exécution de *reti* a l'effet suivant :

- Le sommet de la pile est dépilé dans les flags
- Le sommet de la pile est dépilé dans PC

L'exécution de *reti* provoque donc le retour dans le programme interrompu, juste après l'instruction durant laquelle l'interruption a eu lieu.

b) Comptage des événements sur IT (45 mn)

- Écrire un programme qui ne termine jamais ; à chaque itération il lit la valeur de la case mémoire d'adresse 0x100 et l'affiche sur les leds

- Ajouter à ce programme un sous-programme d'interruption, qui incrémente le contenu de la case mémoire 0x100 à chaque occurrence d'une interruption sur IT