TP5-6 - Entrées/sorties ; interruptions

1. Comptage d'événements en scrutant une entrée

Écrire un programme qui ne termine jamais ; à chaque itération il vérifie si `sw[0]` est passé de 0 à 1 ; si c'est le cas il incrémente la valeur de la case mémoire d'adresse 0x100 et l'affiche sur les leds

2. Comptage d'événements par interruption

- 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

3. Buffer d'entrée

- Écrire des sous-programmes gérant un buffer (file) de mots de 32 bits. On doit pouvoir connaître sa taille, ajouter un mot, retirer un mot
- Écrire un sous-programme d'interruption qui, à chaque occurrence de IT, ajoute la valeur de `sw[15..0]` au buffer
- Écrire un programme qui ne termine jamais et qui attend une centaine de cycles entre chaque itération ; il affiche sur les leds le cumul de toutes les valeurs qui ont été introduites dans le buffer

4. Commutation de tâches

Modifier le code suivant pour qu'il permette d'exécuter les deux programmes prog0 et prog0 en alternance, le changement ayant lieu à chaque appui sur IT. A la commutation, l'état du programme interrompu sera sauvegardé dans sa propre pile.

```
// tâche 0 = comptage sur ld[3..0] ; tâche 1 = comptage sur ld[7..4]
// À chaque appui sur IT, changement de tâche

LD      =     0xb0000000
STACK0  =     0x100
STACK1  =     0x200

        ba    start

        .org  1
it:     // ...
        reti

start:  // initialisation contexte tâche 1
        setq  STACK1, %sp
        // ...
        // démarrage sur la tâche 0
        setq  STACK0, %sp
        ba    prog0

// tâche 0
```

```
prog0:   set   LD, %r3
         clr   %r1
loop0:   inc   %r1
         sll   %r1, 8, %r2
         st    %r2, [%r3]
         ba    loop0

// tâche 1
prog1:   set   LD, %r3
         clr   %r1
loop1:   inc   %r1
         st    %r1, [%r3]
         ba    loop1

tasknum: .word 0          // numéro de la tâche courante (0 ou 1)
tabsp:   .word 0, 0       // table des pointeurs de pile
```