

IFT209 – Programmation système
Université de Sherbrooke

Laboratoire 2

Enseignant: Michael Blondin
Date de remise: dimanche 7 février 2021 à 23h59
À réaliser: en équipe de deux
Modalités: remettre en ligne sur Turnin

Date de remise: dimanche 7 février 2021 à 23h59

À réaliser: en équipe de deux

Modalités: remettre en ligne sur **Turnin**

Le but de ce laboratoire est de mettre en pratique l'arithmétique signée ainsi que la manipulation de nombre de différentes tailles.

Problème. Vous devez écrire un programme, en langage d'assemblage de l'architecture ARMv8, qui:

- lit deux entiers signés a et b de 16 bits;
- affiche $a + b$ (calculé sur 32 bits);
- indique s'il y aurait eu débordement si la somme avait été calculée sur 16 bits;
- affiche $a \cdot b$ (calculé sur 32 bits).

Observons qu'il n'est pas possible ici d'utiliser les codes de condition afin d'identifier un débordement, puisqu'il s'agit d'arithmétique sur 16 bits plutôt que 64 ou 32 bits. Il faut donc le vérifier manuellement.

Afin de calculer le produit, utilisez l'algorithme « simple » de multiplication, c.-à-d. celui qui:

- étend a et b à 32 bits avec les bons bits de signe (voir section 6.6.2 des notes de cours au besoin);
- effectue la multiplication des deux nombres sur 64 bits à l'aide de l'addition itérée de 32 termes;
- garde les 32 bits de poids faible de la somme obtenue.

Par exemple, pour $a = 5$ et $b = -7$, l'algorithme effectue ces calculs:

[illegible]

Comme vous n'avez pas accès directement à la représentation binaire de a et b , pensez à une façon arithmétique de décaler a progressivement vers la gauche et de consommer les bits de b progressivement à partir de la droite.

Tests. Par exemple, dans un terminal, vous devriez obtenir:

5	8448	-30567
-7	123	-25542
-2	8571	-56109
sans débordement	sans débordement	débordement
-35	1039104	780742314

où les deux premières lignes sont les nombres entrés au clavier, et les **trois dernières lignes** correspondent à la sortie affichée (avec sauts de ligne).

Directives.

- Votre programme doit être obtenu en complétant le code partiel ci-dessous;
- Votre programme doit être remis dans un seul fichier nommé `labo2.s`;
- Vous ne pouvez *pas* utiliser d'instruction de multiplication comme « **mul** » et « **madd** »;
- La lecture d'un nombre signé de 16 bits s'effectue avec le format « `%hd` »;
- Ne modifiez pas le point d'entrée ainsi que le format des entrées et sorties;
- Supposez que les valeurs en entrée sont valides.

Pointage. Vous pouvez obtenir jusqu'à 10 points répartis ainsi:

- 1 point si votre programme lit deux entiers signés de 16 bits;
- 2 points pour l'addition (vous aurez au moins 1 point si vous passez les trois tests de la page précédente);
- 2 points pour l'identification du débordement (vous aurez au moins 1 point si vous passez les trois tests);
- 3 points pour la multiplication (vous aurez au moins 1,5 point si vous passez les trois tests);
- 2 points pour la lisibilité du code (indentation, commentaires et usage des registres).

Code partiel.

```
.global main

// ...
main:
    /*
       Code ici
    */

    mov     x0, 0
    bl      exit

.section ".rodata"
fmtEntree:      .asciz  "%hd"
fmtSortie:      .asciz  "%d\n"
msgDebordement: .asciz  "débordement\n"
msgSansDebordement: .asciz  "sans débordement\n"
```