# IFT6800–Atelier en Informatique
# (Devoir #1)

Louis Salvail[1]

Université de Montréal (DIRO), QC, Canada
salvail@iro.umontreal.ca
Bureau: Pavillon André-Aisenstadt, #3369

## 1 Devoir #1

Il s'agit du premier devoir pour le cours. La date de remise est:

Vendredi, 30 septembre 2016, 12:30,

à la période de démo. Ce devoir doit être fait individuellement.

### 1.1 Algèbre de Boole (50pts)

1. (5pts) Additionnez les deux mots de 32 bits écrits en hexadécimal:

$$D40CCB41_{16} + 5720A51F_{16} \ .$$

   Donnez votre réponse en hexadécimal sur 32 bits. Vous éliminez la retenue finale.
2. (5pts) Est-ce que le calcul précédent à débordé (*overflow*)?
3. (5pts) Est-ce que le résultat obtenu en **??**. est strictement plus grand que 0 en complément à 2?
4. (10pts) Triez les éléments suivants en ordre croissant (en tenant compte des signes). Il s'agit de 5 mots de 32 bits exprimés en complément à 2:

$$C00045BB_{16}, 70F0F0F0_{16}, 81347000_{16}, 70000000_{16}, 5FFFFFFF_{16} \ .$$

5. (5pts) Combien de nombres sont négatifs (selon le complément à deux) dans la liste de 5 nombres hexadécimaux de 32 bits donnée plus haut?
6. (5pts) Multipliez le nombre hexadécimal de 64 bits $4720AABCB732A705_{16}$ par $16_{10}$ (i.e. ce qui est $10_{16}$). Donnez simplement votre réponse en hexadécimal. Inspirez-vous de la façon que nous multiplions par 10 en décimal.
7. (10pts) Donnez un circuit arithmétique fait de portes logiques ET ("$\wedge$"), OU ("$\vee$"), OUX ("$\oplus$") et NON ("$\neg$") qui calcule la formule booléenne

$$(x_1 \wedge x_2 \wedge \neg x_3) \oplus (\neg(x_3 \vee x_2 \vee x_0) \oplus \neg(x_0 \wedge x_1)) \ ,$$

   où $x = x_0, x_1, x_2, x_3$ est un registre de 4 bits. L'expression contient des parenthèses que vous devez respecter. Par exemple. $\neg(x_3 \vee x_2 \vee x_0)$ est la négation du résultat de $x_3 \vee x_2 \vee x_0$.
8. (5pts) Trouvez une valeur $x = x_0, x_1, x_2, x_3$ explicite pour les 4 bits de $x$ qui retourne 1 lorsqu'appliquée à la formule précédente. Trouvez une autre valeur $x' = x'_0, x'_1, x'_2, x'_3$ qui retourne 0 dans les mêmes conditions.

### 1.2 Programmer la X-TOY (50pts)

1. (0pt) Installez le simulateur X-TOY sur votre ordinateur. Vous le trouverez à l'adresse:

   `http://introcs.cs.princeton.edu/xtoy/`.

2. (10pts) Écrivez un programme X-TOY qui boucle à l'infini (qui ne termine jamais). Donnez le programme en hexadécimal et en pseudocode de la façon vue en cours et en démo. Votre programme est logé à partir de l'adresse $10_{16}$. Expliquez pourquoi le programme ne temine jamais.
3. (10pts) Écrivez un programme X-TOY dont la premère instruction est logée à l'adresse $10_{16}$ qui accepte un mot de 16 bits en entrée (via le `stdin`) et qui affiche `0000` si le nombre est positif ou nul ou affiche `00FF` s'il est négatif (i.e. en compplément à deux). Donnez votre programme en hexadécimal et en pseudocode de la façon vue en cours et en démo. Expliquez son fonctionnement.
4. (10pts) Donnez un programme qui charge trois mots de 16 bits à partir du `stdin`, additionne les deux premiers avant de soustraire le troisième et de donner le résultat final à l'écran (i.e. sur `stdout`).Donnez votre programme en hexadécimal et en pseudocode de la façon vue en cours et en démo.
5. (10pts) En cours, nous avons vu un programme, `re-program`, qui accepte un mot du `stdin` et qui remplace une instruction du programme par celle-ci. Le programme est donc modifié par l'instruction acceptée en entrée. Indiquez un mot, qui lorsque donné en entrée sur `stdin`, fait en sorte que le programme `re-program` boucle à l'infini. Expliquez votre réponse.
6. (10pts) Écrivez un programme, logé à l'adresse $10_{16}$, qui charge dans `R[1]` la valeur hexadécimale `CAFE` (i.e. ce qui vous demandera possiblement quelques instructions et l'aide d'autres registres) et qui l'imprime ensuite à l'écran. À la fin de l'exécution, le registre `R[1]` doit toujours contenir la valeur `CAFE`. Donnez le programme en hexadécimal et en pseudocode de la façon vue en cours et en démo. Expliquez son fonctionnement.