Principes de fonctionnements des machines binaires (PF1)
Partiel du samedi 20 novembre 2010. Durée : 2 heures
Aucun document autorisé.
Ni calculette, ni ordinateur. Téléphone éteint

Les exercices sont indépendants et peuvent être traités dans un ordre quelconque
Réfléchissez avant de vous lancer dans des calculs peut-être inutiles ...
Vos réponses doivent être justifiées

Exercice 1. Imaginons une machine où les mots seraient constitués de 13 bits dans lesquels les nombres entiers seraient représentés comme sur la plupart des machines, c'est-à-dire en base deux, avec un bit de signe et en complément à deux. Quel serait l'intervalle des nombres représentables ?

Exercice 2. Soit le nombre dont la représentation usuelle (c'est-à-dire en base dix) est `1112`.

2.1. Quelles sont ses représentations dans les bases deux, quatre, huit et seize ?

2.2. Dire pour chacun des types `byte`, `short`, `int` et `long` si il y est représentable.

2.3. Pour le type de plus petite taille pour lequel il est représentable, quelle est la représentation de son opposé (c'est-à-dire `-1112`) ?

Exercice 3. On considère le nombre n dont la représentation octale (en base huit) est `3456701356710400000`.

3.1. Quels sont les plus grands entiers p, q, r et s tels que n est un multiple de 2^p, de 4^q, de 8^r et de 16^s ?

3.2. Quels sont les quotients et restes de la division euclidienne de n par 2^{p+1}, par 4^{q+1}, par 8^{r+1} et par 16^{s+1} ?

Exercice 4. Soient les nombres m et n dont les représentations hexadécimales sont respectivement `BC670893B` et `A0BC57A`. Effectuer à la main et directement en base seize, en faisant apparaître les retenues, les opérations $m + n$ et $m - n$.

Exercice 5. On considère l'instruction

```
float n = 147 + 1.0/256 + 1.0 / 2048 + 1.0/4096 + 1.0/8192
```

5.1. Quelle est la représentation en machine[1] de `n` ?

5.2. Quelle sera la représentation en machine de `x` après la séquence `float x = n + (16 * 1024);` ?

5.3. Quelle sera la représentation en machine de `y` après la séquence `float y = n + 1.0/(8 * 1024 * 1024);` ?

---

[1] On rappelle la représentation en machine du type `float` :
Un bit pour le signe, 8 bits pour l'exposant (valeur de l'exposant vrai augmentée de 127) et 23 bits pour la mantisse

Exercice 6. On considère une carte graphique possédant une mémoire de 2Mo (Mega-octets). On rappelle que 1Ko=1024Ko et 1Mo=1024Ko.

6.1. On souhaite pouvoir afficher une image ayant une définition de 1024×768 pixels où chaque pixel possède 256 nuances de couleur. Quelle est le poids de l'image ? La carte graphique peut elle contenir celle-ci ?

6.2. Compte-tenu de la mémoire de la carte graphique, combien de bits peut-on utiliser par pixel si l'on travaille avec une définition de 1280*1024 pixels ?

6.3. On souhaite travailler avec 32 bits par pixels. Si l'on conserve le rapport standard pour l'image de 4/3 (largeur/hauteur), quelle définition peut-on atteindre au maximum en stockant l'image dans la mémoire de la carte ?

Exercice 7. L'UTF-8 est une norme qui permet de coder un ensemble universel de caractères et étend l'ASCII. Il utilise un nombre d'octets variant de 1 à 4 (les caractères codés sur un octet sont exactement les caractères US-ASCII).

Le nombre d'octets utilisé pour coder un caractère est fixé par le premier octet de la représentation. Les règles pour représenter les caractères sont indiquées dans le tableau suivant (`x` signifie que le bit correspondant peut être `0` ou `1`).

nombre d'octets	1er octet	2nd octet	3ème octet	4ème octet
1	`0xxxxxxx`			
2	`110xxxxx`	`10xxxxxx`		
3	`1110xxxx`	`10xxxxxx`	`10xxxxxx`	
4	`11110xxx`	`10xxxxxx`	`10xxxxxx`	`10xxxxxx`

Dans la suite de l'exercice, on appelle code UTF-8 une suite de 1 à 4 octets de la forme indiquée par le tableau ci-dessus.

7.1. Combien de caractères différents les codes UTF-8 permettraient-ils théoriquement de représenter sur (a) un octet ? (b) deux octets ? (c) trois octets (d) 4 octets ? (justifiez vos réponses et vous pouvez vous contenter d'un résultat approché pour (c) et (d)).

7.2. On veut représenter chaque code UTF-8 par un entier dont l'écriture binaire est obtenue par juxtaposition des octets dans l'ordre, le premier étant de poids le plus fort. Pour $i \in \{1, 2, 3, 4\}$, donnez, *en hexadécimal*, le plus petit entier p_i, et le plus grand entier g_i qui représentent un code UTF-8 sur i octets.

7.3. Tout entier entre p_i et g_i représente-t-il un code UTF-8 ? (discuter suivant i, et si la réponse est non, produire un contre-exemple).

Une suite d'octets représente un texte en UTF-8 si elle est obtenue par concaténation (mise bout à bout) de codes UTF-8 (du poids fort vers le poids faible).

7.4. Donner un exemple de suite d'octets qui ne représente pas un texte en UTF-8.