Université de Paris-Saclay
Faculté des sciences d'Orsay
L3 E3A

Électronique numérique

UE-308

TD1 : Bases et arithmétiques de numérations, représentations des nombres pour le numérique

Exercice 1 : conversions diverses

1. Convertir en décimal (signé ou non) les nombres binaires (naturels ou signés en complément à deux) suivants : $(01011)_2$; $(01011)_{C2}$; $(11011)_2$ et $(11011)_{C2}$.

2. Pour les nombres décimaux (signé ou non) suivants, prévoir le nombre de bits nécessaires aux codages respectifs en binaire naturel et en binaire signé (complément à deux) puis effectuer la conversion : $(10)_{10}$; $(-10)_{10s}$; $(128)_{10}$; $(128)_{10s}$ et $(-131)_{10}$.

3. Convertir en décimal (non-signé) les nombres hexadécimaux suivants : $(F3)_{16}$ et $(54)_{16}$.

4. Pour les nombres décimaux non-signés suivants, prévoir le nombre de chiffre nécessaires à leur codage en hexadécimal puis effectuer la conversion : $(17)_{10}$; $(128)_{10}$ et $(131)_{10}$.

5. Convertir les nombres en binaire naturel suivants en hexadécimal : $(010)_2$ et $(0110100)_2$.

6. Convertir les nombres hexadécimaux suivants en binaire naturel : $(F3)_{16}$ et $(54)_{16}$.

7. Convertir 108Mo en Mio puis en Gio et ko.

8. Convertir 230Gio en Go puis en Mio.

9. Convertir $(12)_{10}$ en code Gray.

10. Convertir « J'ai eu 12/20 en math !! » en code ASCII (codage décimal).

Exercice 2 : système de numération octal [1]

Le système de numération octal est le système de numération de base $B = 8$, et utilise les chiffres α_k de 0 à 7.

1. Donner en base 10 le nombre $(702)_8$.

2. Soit le nombre $(123)_{10}$, combien faudra t'il de chiffre en base octal pour le représenter.

3. A l'aide de vos connaissances, proposer deux méthodes pour passer d'un nombre décimal (non-signé) à un nombre en base 8.

Exercice 3 : addition en binaire

1. Soient deux nombres en binaire naturel de même taille égale à T bits. Prévoir, en passant par le décimal, une règle pour détecter un dépassement (ou overflow).

2. Pour les couples de nombres suivants, dire si l'addition va amener à un dépassement puis effectuer l'addition :$(0100)_2 + (0011)_2$ et $(1100)_2 + (0111)_2$.

3. En utilisant un codage en complément à 2, effectuer en binaire et sur 8 bits les cinq opérations suivantes $1 - 2$; $51 + 127$; $-3 - 127$; $-127 + 127$; $-63 - 63$. Préciser, pour chaque opération, s'il y a dépassement.

1. Ce système a longtemps été utilisée aux débuts des ordinateurs, puis abandonnée au profit de l'hexadécimal. Les Yuki (tribu autochtone d'Amérique du nord du 19ème siècle utilisaient le système octal car ils utilisaient l'espace entre leurs doigts pour compter). Plusieurs autres anecdotes sur ce système peuvent être consulter à l'adresse suivante :https ://en.wikipedia.org/wiki/Octal

Exercice 4 : code BCD et addition

1. Convertir les nombres non signés suivant en BCD : $(231)_{10}$; $(3A)_{16}$ et $(0011101)_2$.

2. Pour additionner deux nombres décimaux codés en BCD, on procède par l'addition en binaire naturel des unités puis des dizaines, centaines etc. Pour chaque groupe, si le résultat est supérieur (strictement) à 9 alors on ajoute 0110 au résultat obtenu puis on passe au groupe suivant. Utiliser cet algorithme pour calculer en BCD les sommes suivantes : $(13)_{10} + (25)_{10}$ puis $(17)_{10} + (25)_{10}$. Justifier pourquoi il est nécessaire d'ajouter 0110 au résultat obtenu pour que l'algorithme fonctionne.

Exercice 5 : addition en hexadécimal

1. Effectuer l'addition suivante avec un résultat en hexadécimal (on pourra s'aider du passage au décimal) : $(3AF)_{16} + (B2)_{16}$.

2. On parle de code hexadécimal signé lorsque l'on substitue son code binaire naturel par un code en complément à 2. Autrement dit, pour obtenir un nombre décimal signé à partir d'un nombre hexadécimal supposé signé, on effectue les opération suivantes $(X)_{16s} \rightarrow (Y)_2 \equiv (Z)_{C2} \rightarrow (W)_{10s}$. Donner la valeur en décimal signé du nombre en hexadécimal signé $(FC)_{16s}$. On considère tout les nombres de deux chiffres en hexadecimal signé. Quel est l'intervalle de nombres représentant des décimaux négatifs.

Exercice 6 : gestion de la virgule et norme IEEE 754

1. Soit la représentation virgule fixe en complément à 2 sur 16 bits, où la virgule est à droite du bit de poids fort et où les 15 bits à droite de la virgule correspondent aux puissances 2^{-k} avec $k \in \{1, \ldots, 15\}$. Quelle est l'ensemble des nombres décimaux représenté et avec quelle précision ? Convertir les nombres décimaux $(0.1)_{10}$ et $(0.2)_{10}$ dans ce système puis effectuer l'addition en binaire. Retrouve [2] t'on le nombre $(0.3)_{10}$?

La norme IEEE 754, apparue en 1985, est la norme la plus employée pour la représentation des nombres à virgule flottante dans le domaine informatique [3].

2. Coder le nombre $(-1027,625)_{10}$ dans la norme IEEE 754 en simple précision.

3. Soit le nombre $(01010101010101010101010101010101)_2$ dans la norme IEEE 754 en simple précision. Combien représente ce nombre en décimal.

4. Quel est le plus grand nombre pouvant être codé en simple précision avec la norme IEEE 754 ? En déduire le plus petit nombre pouvant être codé. (Attention aux valeurs particulières).

2. Ceci explique pourquoi un logiciel comme Matlab ou Python nous renvoi l'affirmation logique « faux » si nous lui demandons d'effectuer $0.1 + 0.2 == 0.3$

3. https ://fr.wikipedia.org/wiki/IEEE_754