

Chapitre 1 : Solutions

1.3 Conversions

$$1.3.1.a) \quad B4F,D5_{(16)} = B_{(16)} * 16^2 + 4_{(16)} * 16^1 + F_{(16)} * 16^0 + D_{(16)} * 16^{-1} + 5_{(16)} * 16^{-2}$$

$$B4F,D5_{(16)} = 11 * 256 + 4 * 16 + 15 * 1 + 13 * 0,0625 + 5 * 0,00390625$$

$$B4F,D5_{(16)} = 2816 + 64 + 15 + 0,8125 + 0,01953125$$

$$B4F,D5_{(16)} = 2895,83203125$$

$$1.3.1.b) \quad 324,21_{(5)} = 3_{(5)} * 5^2 + 2_{(5)} * 5^1 + 4_{(5)} * 5^0 + 2_{(5)} * 5^{-1} + 1_{(5)} * 5^{-2}$$

$$324,21_{(5)} = 3 * 25 + 2 * 5 + 4 * 1 + 2 * 0,2 + 1 * 0,04$$

$$324,21_{(5)} = 75 + 10 + 4 + 0,4 + 0,04$$

$$324,21_{(5)} = 89,44$$

1.3.1.c) Il est possible de faire ce problème en passant directement de la base 7 à la base 2. Cela est par contre peu pratique puisqu'il faut faire des opérations mathématiques en base 2 et que la majorité de nos calculatrices ne sont pas adaptées pour cela. La méthode utilisée est donc de faire la conversion vers la base 10 pour ensuite aller à la base 2.

$$125_{(7)} = 1 * 7^2 + 2 * 7^1 + 5 * 7^0 = 49 + 14 + 5 = 68$$

Méthode itérative :

$$68 / 2 = 34 \text{ restant } 0$$

$$34 / 2 = 17 \text{ restant } 0$$

$$17 / 2 = 8 \text{ restant } 1$$

$$8 / 2 = 4 \text{ restant } 0$$

$$4 / 2 = 2 \text{ restant } 0$$

$$2 / 2 = 1 \text{ restant } 0$$

$$1 / 2 = 0 \text{ restant } 1$$

$$125_{(7)} = 1000100_{(2)}$$

$$1.3.1.d) \quad 73 / 16 = 4 \text{ restant } 9$$

$$4 / 16 = 0 \text{ restant } 4$$

$$73_{(10)} = 49_{(16)}$$

$$1.3.1.e) \quad 73_{(8)} = 00111011_{(2)} = 00111011_{(2)} = 3B_{(16)}$$

$$1.3.1.f) \quad 101101_{(2)} = 55_{(8)}$$

1.4 : Arithmétique binaire

1.4.1 Le « format » complément à 2 est une représentation binaire qui permet de représenter à la fois des valeurs positives et négatives. Les valeurs négatives sont représentées comme le complément à 2 de leur valeur absolue.

L'opération de complément à 2 (« faire » le complément à 2) est une transformation que l'on peut effectuer sur un nombre pour le représenter avec une origine décalée. Lorsqu'un nombre est représenté en format complément à 2, faire le complément à 2 de ce nombre est l'équivalent d'inverser le nombre.

1.4.2 Il est utile d'utiliser le format complément à 2 en arithmétique binaire puisqu'il facilite la manipulation de valeurs négatives. En format complément à 2, il est possible d'additionner des valeurs positives et négatives avec une opération d'addition normale. Pour faire une soustraction, il suffit de faire une addition avec le complément à 2 de la valeur que l'on veut soustraire.

1.4.3.a) Pour représenter ce nombre dans le format complément à 2, on peut d'abord trouver la représentation de sa valeur absolue en binaire non signé, et ensuite en faire le complément à 2.

$$\begin{aligned} |-29_{(10)}| &= 29_{(10)} = 0011101_{(2)} \\ \overline{0011101} &= \sim 0011101 + 1 = 1100010 + 1 = 1100011 \end{aligned}$$

1.4.3.b)

$$\begin{aligned} |-29,5_{(10)}| &= 29,5 = 000\ 0011101,100 \\ \overline{0000011101,100} &= \sim 0000011101,100 + 0,01 = 1111100010,011 + 0,01 = 1111100010,100 \end{aligned}$$

1.4.3.c)

$$\begin{aligned} -4D,3_{(16)} &= \overline{0100\ 1101,0011} = \overline{0001001101,0011} \\ -4D,3_{(16)} &= \sim 0001001101,0011 + 0,0001 = 1110110010,1100 + 0,0001 \\ -4D,3_{(16)} &= 1110110010,1101 \end{aligned}$$

1.4.3.d)

$53 / 2 = 26$ reste 1	$0,375 * 2 = 0,75$
$26 / 2 = 13$ reste 0	$0,75 * 2 = 1,5$
$13 / 2 = 6$ reste 1	$0,5 * 2 = 1,0$
$6 / 2 = 3$ reste 0	
$3 / 2 = 1$ reste 1	
$1 / 2 = 0$ reste 1	

$$53,375 = 0110101,011$$

1.4.3.e) Lorsqu'un nombre est positif complément à 2, la représentation est identique qu'en binaire naturel si le bit de gauche vaut zéro (autrement, il faut le rajouter). C'est pourquoi e) et f) ont la même solution:

$$+53,375 = 0110101,011$$

1.4.3.f) 110100,10 (complément à 2, 7 bits, 2bits) → (base 10)

1.4.4.a)

$$\begin{array}{r} 01101110 \\ +00011001 \\ \hline 10000111 \end{array}$$

L'addition de deux entiers positifs donne une valeur négative : il y a dépassement

1.4.4.b)

$$\begin{array}{r} 01101110 \\ +10110011 \\ \hline \cancel{1}00100001 \\ +00011001 \\ \hline 00111010 \end{array}$$

La retenue est
tout simplement
éliminée

première addition : positif + négatif : pas de dépassement

deuxième addition : positif + positif = positif : pas de dépassement

Il n'y a donc pas de dépassement.

1.4.4.c)

$$\overline{N1} = \sim 01101110 + 1 = 10010001 + 1 = 10010010$$

$$N2 - N1 = N2 + \overline{N1}$$

$$\begin{array}{r} 00011001 \\ +10010010 \\ \hline 10101011 \end{array}$$

positif + négatif : pas de dépassement

1.4.4.d) Les compléments à 2 et à 1 de N1 sont réutilisés de c)

$$N3 - N1 = N3 + \overline{N1}$$

$$\begin{array}{r} 10110011 \\ +10010010 \\ \hline \cancel{1}01000101 \end{array}$$

Dans les deux cas :

négatif + négatif = positif : dépassement

1.4.4.e)

$$N1 * 2 = 01101110 * 2 = 01101110 \ll 1 = 011011100$$

On passe d'une valeur positive à une valeur négative : dépassement

1.4.4.f)

$$N1 / 2 = 01101110 / 2 = 01101110 \gg 1 = \overset{\curvearrowright}{0}01101110$$

Élimination d'un 0 : pas de troncature

1.4.4.g)

$$N3 * 2 = 10110011 * 2 = 10110011 \ll 1 = \cancel{0}01100110$$

On passe d'une valeur négative à une valeur positive : dépassement

1.4.4.h)

$$N3 / 2 = 10110011 / 2 = 10110011 \gg 1 = \cancel{1}1011001$$

Élimination d'un 1 : troncation

1.4.4.i)

$$N2 * 4 = 00011001 * 4 = 00011001 \ll 2 = \cancel{00}01100100$$

Élimination de 0 : pas de dépassement

1.4.4.j)

$$N2 / 4 = 00011001 / 4 = 00011001 \gg 2 = \cancel{00000}1100$$

Élimination d'un 1 : troncation

1.5 Codes

1.5.1 Il y a huit réponses possibles :

- a) 001, 010, 100
- b) 000, 011, 101
- c) 000, 011, 110
- d) 001, 010, 111
- e) 000, 101, 110
- f) 001, 100, 111
- g) 010, 100, 111
- h) 011, 101, 110

1.5.2 *NON!* Si on dispose de 6 bits, on ne peut avoir plus de 2 mots différents pour avoir une distance minimale de 6.

1.5.3 Il existe plusieurs façon de solutionner ce problème : une solution est de construire la séquence de code de grey pour obtenir la réponse et deuxième façon correcte est d'utiliser un algorithme de conversion grey à binaire naturel (mais ce n'est pas montré en classe). Voici les réponse finale :

- a) 28 b) 2 c) 14 d) 1 e) 21 f) 3

1.5.4

- a) 1 b) 0 c) 0 d) 1 e) 1 f) 0 g) 1 h) 0 i) 0 j) 1

1.6 Contrôle des erreurs

1.6.1

Distance de Hamming entre les codes	Détecter 1 erreur	Détecter 2 erreurs	Détecter 3 erreurs	Détecter 4 erreurs
2	X			
3	X	X		
4	X	X	X	
5	X	X	X	X

Distance de Hamming entre les codes	Corriger 1 erreur	Corriger 2 erreurs	Corriger 3 erreurs	Corriger 4 erreurs
2				
3	X			
4	X			
5	X	X		

1.6.2

1.6.2.1) Réponse : b)

1.6.2.2) Réponse:

0	0	0	1	1
1	0	0	1	0
0	0	0	1	1
1	0	0	1	0
0	0	0	1	1
0	0	0	1	1

1.6.2.3a) On retrace la grille

0	0	0	1	1
1	0	0	1	0
0	0	0	1	1
1	0	0	1	0
0	0	0	1	1
0	0	1	1	1

La parité n'est pas respectée. Cependant, comme toutes les parités sur la colonne de droite sont justes, on déduit que le message original est bien

0001 1001 0001 1001 0001

1.6.2.3b)

On retrace la grille

0	0	0	1	1
1	0	0	1	0
0	1	0	1	1
1	0	0	1	0
0	0	0	1	1
0	0	0	1	1

Deux parités ne sont pas respectées. La première se trouve sur la deuxième colonne, la seconde sur la troisième ligne. On en déduit que le bit à l'intersection est erroné, et on corrige le message obtenu pour obtenir :

0001 1001 0001 1001 0001

1.6.2.3c)

On retrace la grille

0	0	0	1	1
1	0	0	1	0
0	1	1	1	1
1	0	0	1	0
0	0	0	1	1
0	0	0	1	1

La parité n'est pas respectée sur deux colonnes (2 et 3). Cependant, toutes les parités sont correctes sur les lignes. Le message est donc entaché d'erreur, mais il nous est impossible de le corriger. C'est là une illustration du concept $M-1 = C+D$

M étant ici 3, on voit bien que nous sommes en mesure de détecter 2 erreurs ($D=2$), mais il apparaît alors que $C=0$ (impossible de corriger).

1.6.2.4)

i) $3 \times 5 = 15$

ii) Un code de Hamming permet d'envoyer $2^n - n - 1$ bits d'information (par déduction en observant le code de Hamming) ou n est le nombre de bits de parité. Pour $n = 3$, on peut envoyer 4 bits, pour $n = 4$, 13, pour $n = 5$, 26, etc. Donc pour envoyer 20 bits, il faut 5 bits de parité (6 bits du code ne seront tout simplement pas utilisés).

iii) L'avantage de l'envoyer en 5 mots de 4 bits est que si une erreur survient dans plusieurs de ces groupes de 4 bits, il est possible de faire la correction à l'intérieur de chacun de ces groupes. En utilisant un seul mot de 20 bits, si plusieurs erreurs surviennent, il devient impossible de les corriger. L'avantage d'utiliser un mot de 20 bits par contre est que le nombre de bits de parité à transmettre est beaucoup plus faible.

1.6.2.5) *L'utilisation du code de Hamming produit toujours une distance de Hamming de 3.*

1.6.3

1.6.3.1) 1

1.6.3.2) *La plus petite distance de Hamming entre deux mots du code BCD étant 1, l'équation*

$$M - 1 = C + D \text{ donne}$$

$$M - 1 = 1 - 1 = 0 = C + D$$

$$D'ou\ C = D = 0$$

Une erreur peut donc survenir (lecture 0000, réception 0001 par exemple) sans que le système ne puisse la détecter.

Notez cependant que certaines erreurs peuvent être détectées néanmoins. En effet, si on a le schéma (lecture 1001, réception 1011) le système détecte une erreur puisque 1011 ne fait pas partie du code BCD 8421. Dans ce cas, il sera incapable de savoir quel était le chiffre lu, puisqu'il peut tout aussi bien être 0011 que 1001.

1.6.3.3) 2

1.6.3.4) *M étant égal à 2*

$$M - 1 = 2 - 1 = 1 \text{ On trouve donc } D = 1 \text{ } C = 0$$

Une erreur peut être détectée, mais elle ne pourra pas être corrigée.

1.6.3.5)

0: 1110010

5: 1001110

1: 1100110

6: 1010000

2: 1101100

7: 1000100

3: 1000010

8: 1001000

4: 1011100

9: 1110100

1.6.3.6) *Après avoir effectué un complément à 1 sur un ensemble de mots binaires, les distances de Hamming les séparant demeurent les mêmes. La plus petite reste donc 2.*

1.6.3.7) *Elle est forcément paire puisque tous les bits ont été inversés, et que le nombre de bits par mot est impair (il est de 7).*

1.6.3.8) *Elle serait demeuré la même : impaire.*