Architecture des ordinateurs - TD 01

21 septembre 2020

1 Conversions de base

1. Convertissez les nombres décimaux suivants en binaire :
   255_{10}, 104_{10}, 2010_{10}
2. Convertissez les nombres décimaux suivants en base 5 :
   250_{10}, 78_{10}, 33_{10}, 622_{10},
3. Convertissez les nombres suivants en décimal :
   1234_5, 1234_7, 1234_9
4. Convertissez les nombres binaires suivants en hexadécimal puis en octal :
   110_2, 1011_2, 11110101100_2, 1100000011011110_2
5. Convertissez les nombres hexadécimaux suivants en décimal :
   400_{16}, FFF_{16}, $7FF_{16}$, $A000_{16}$
6. Quel est le plus grand entier positif codable sur 9 bits en binaire ? Combien faut-il de chiffres pour l'écrire en octal ? Et en hexadécimal ?

2 Additions et Multiplications en binaire

1. Additionnez les entiers positifs suivants directement en binaire, indiquez les cas qui produisent un overflow de la représentation 8 bits :
   $00101001_2 + 11001010_2$
   $10101011_2 + 11001010_2$
   $11111111_2 + 11111111_2$
2. Multipliez les entiers positifs suivants, indiquez les cas qui produisent un overflow de la représentation 8 bits :
   $00001001_2 \times 00001010_2$
   $10101011_2 \times 11001010_2$
3. Quel est l'entier le plus grand représentable sur n bits ?
4. Additionnons un entier x codé en n bits et un entier y codé en m bits, avec $n \geq m$, sur combien de bits faut-il coder le résultat pour éviter un overflow ?
5. Soit un entier x codé en n bits et un entier y codé en m bits avec $n \geq m$. Montrez que $n+m$ bits suffisent pour représenter $x \times y$.
   (Bonus plus difficile : montrez que le résultat ne peut pas être codé sur $n+m-1$ bits si $m \geq 2$.

3 Représentation des binaires codés en décimal

1. Donnez la représentation en binaire codé en décimal des nombres suivants :
   89
   2048
   1984

2. Donnez la valeur décimale des nombres codés en binaire suivants :
   01000010, 0010000000010001, 010100010010
3. Estimez la rapport entre le nombre de bits nécessaires pour coder un nombre en binaire et en binaire codé décimal. Rappel de cours : pour coder l'entier N, il faut $\lceil log_b(N+1) \rceil$ bits en base b. Quelle est la représentation la plus compacte ?

4 Représentation en virgule fixe

Dans la représentation en virgule fixe, un nombre décimal s'écrit avec n chiffres pour la partie entière et q chiffres pour la partie fractionnaire. Voici quelques nombres décimaux lorsque $n = 2$ et $q = 3$: $12,345$; $05,217$.

1. Arrondissez les rationnels suivants à la valeur décimale la plus proche en virgule fixe ($n = 2$ et $q = 3$) : 1/3 15/7
2. Quelle est l'erreur maximale obtenue lors d'un arrondi dans la représentation décimale en virgule fixe (n,q) ?
3. Quel est le résultat de l'opération suivante : dans les réels ? en virgule fixe ($n = 2$ et $q = 3$) ?
   $0,14 \times 0,99$

Architecture des ordinateurs - TD 02

1 Représentation d'entiers signés

1. Donnez sur 8 bits les représentations signe plus valeur absolue, complément à 1 et complément à 2 des nombres décimaux suivants :
   36_{10}, -123_{10}, -45_{10}
2. Quelle est la valeur en décimal des nombres binaires dont la représentation en complément à 2 sur 8 bits est :
   11111111_2, 01001000_2, 10001111_2
   Quelles seraient ces valeurs si ces nombres binaires représentaient un complément à 1 ? Un signe plus valeur absolue ?
3. Considérons les représentations en complément à 2 de x, y et z sur 8 bits :
   10010101_2, 00010010_2, 11101011_2
   Sans calculer la valeur de x, y et z en décimal, donnez leur représentation en complément à 2 sur 16 bits.
   Quel est le nombre minimal de bits pouvant représenter x en complément à 2 ? Même question pour y et z.
4. Comment identifierez-vous un nombre pair sous sa représentation binaire en signe + valeur absolue ? En complément à 1 ? En complément à deux ?

2 Additions / soustractions d'entiers signés

1. Transformez la soustraction de deux entiers signés en une addition dans la représentation en complément à 2 puis calculez $120_{10} - 45_{10}$.
2. Sans passer par leur représentation décimale, effectuez les opérations suivantes sur des nombres de 8 bits en complément à deux :
   $00100100_2 + 01000000_2$
   $00011001_2 + 11001110_2$
   $11110110_2 + 10100110_2$
   $00101101_2 + 01101111_2$
   $10000001_2 + 11000000_2$

3 Les masques binaires

Les masques binaires utilisent les opérateurs AND et OR pour ne récupérer qu'une partie des bits d'un champs de bits. En langage C, ils sont représentés respectivement par les symboles & et |. L'opérateur AND permet de ne garder que les bits à 1 communs aux deux champs de bits et l'opérateur OR permet de combiner les deux champs de bits.

1. Effectuer les opérations binaires suivantes :
   11011001 AND 10101010
   11011001 OR 10101010
   11011001 AND (10001000 OR 00100010)

2. Ecrire la fonction C qui affiche la parité d'un nombre donné en paramètre en utilisant les opérateurs & et | (vous ne pouvez pas utiliser l'opérateur modulo) La fonction aura le prototype suivant :

   ```
   void affiche_parite (uint32_t N)
   ```

   Tester avec N = 4 (pair), N = 5 (impair).

3. Ecrire la fonction C qui renvoie 1 si le premier bit, le dernier bit ou les deux sont à 1, sinon elle renvoie 0.

   ```
   int verifie_premier_dernier (uint32_t N)
   ```

   Tester avec N = 164 (doit renvoyer 1), N = 35 (doit renvoyer 1), N = 110 (doit renvoyer 0).

4 Un peu de programmation : cardinal d'un ensemble représenté avec un champs de bits

Soit un ensemble $E \in \mathcal{P}([|0; 3OA1|])$ par exemple $E' = \{4, 5, 8, 15\}$. Une représentation très compacte de cet ensemble est basée sur l'utilisation d'un champs de bit (*bit set* en anglais). Dans cette représentation on encode E sous la forme d'un mot de 32 bits $M = (m_{31}m_{30}\dots m_0)_2$. On utilise la convention suivante : $m_n = 1$ si et seulement si $n \in E$. Par exemple, pour représenter l'ensemble E' on utilisera le mot $M' = 00000000000000001000000100110000_2$.

Nous souhaitons écrire une fonction `char get_size(uint32_t M);` qui calcule le cardinal de l'ensemble E. Cela revient à compter le nombre de bits à 1.

1. Ecrire la fonction `get_size` en utilisant une boucle et l'opérateur >>. Dans le pire cas vous effectuerez 32 itérations.
2. Considérez le mot M' ci-dessus. Calculez $M'' = M'\&(M' - 1)$. Calculez $M''' = M''\&(M'' - 1)$. Que remarquez vous ?
3. Utilisez le résultat de la question précédente pour réécrire la fonction `get_size` en utilisant une boucle qui effectue au maximum $card(E)$ itérations.
4. Question ouverte : voyez vous des algorithmes plus efficaces pour calculer $card(E)$?

Architecture des ordinateurs - TD 03

1 Position du bit a 1 le plus a gauche

1. Ecrire une fonction C qui retourne la position du bit a 1 le plus a gauche dans un mot de 32 bits en utilisant une boucle et l'opeerateur `>>`. Quel est le nombre maximum d'iteerations effectueees ?
2. Nous souhaitons acceeleerer l'algorithme en utilisant une meethode de recherche dichotomique, dont l'algorithme est donnee ci-dessous :
   (a) Soit un mot M composee de n bits. On coupe le mot en deux parties : M_g qui contient les $n/2$ bits les plus a gauche et M_d qui contient les $n/2$ bits les plus a droite.
   (b) Si $M_g = 0$ alors le bit a 1 le plus a gauche est contenu dans M_d et sa position est comprise entre 0 et $n/2$. On recommence ce processus en remplacant M par M_d.
   (c) Si $M_g > 0$ alors le bit a 1 le plus a gauche est contenu dans M_g et sa position est comprise entre $n/2$ et n. On recommence ce processus en remplacant M par M_g.

   Ecrire cet algorithme en C. Pour seelectionner la partie gauche d'un mot de 32 bits vous pouvez utiliser l'opeeration `M & 0xFFFF0000` par exemple ou un deecalage `M >> 16`.

2 Code 2 parmi 5

Un code 2 parmi 5 repreesente chaque chiffre de 0 a 9 sur 5 bits. Chaque mot de 5 bits a exactement 2 bits a 1. Voici un exemple de code 2 parmi 5 :

0	01100
1	11000
2	10100
3	10010
4	01010
5	00110
6	10001
7	01001
8	00101
9	00011

Par la suite, on notera C_n le code pour le chiffre n.

1. Le code 2 parmi 5 peut-etre utilisee pour encoder des codes barres. En utilisant la table ci-dessus, lire la valeur du code barre suivant :

2. La distance de Hamming $d(M, N)$ entre deux mots M et N repreesente le nombre de bits diffeerents entre les mots. Par exemple, $d(1001, 1010) = 2$. Calculer $d(C_6, C_9)$ et $d(C_5, C_6)$.
3. La distance de Hamming d_C d'un code est deefinie comme la plus petite distance de Hamming possible entre deux mots distincts du code, soit $d_C = min\{\forall i \neq j, \quad d_h(C_i, C_j)\}$. Quelle est la distance de Hamming pour le code 2 parmi 5 ?
4. La distance de Hamming d'un code permet de calculer combien d'erreurs peuvent e[a]tre deetecteees e_d et combien d'erreurs peuvent e[a]tre corrigeees e_c. On deefinit $e_d = d_C - 1$ et $e_c = \lfloor \frac{d_C - 1}{2} \rfloor$. Combien d'erreurs peuvent e[a]tre deetecteees par le code 2 parmi 5 ? Combien d'erreurs peuvent e[a]tre corrigeees ?

3 Code avec bit de paritee

On veut envoyer des mots $(a_3 \dots a_0)$ de 4 bits sur un canal. Le code de paritee consiste a ajouter un bit p devant le mot $pa_3 \dots a_0$ qui signale la paritee de la somme des bits du mot $(p = a_3 \oplus a_2 \oplus a_1 \oplus a_0)$. Si la somme est paire, $p = 0$, sinon $p = 1$.

1. Encoder les mots 1010 et 1000 avec un bit de paritee.
2. Calculer la distance de Hamming du code avec bit de paritee.
3. Combien d'erreurs peuvent e^{a}tre deetecteees ? Combien peuvent e^{a}tre corrigeees ?

4 Code a reepeetition

Le code a reepeetition 3 consiste a remplacer chaque 0 du mot initial par 000 et chaque 1 par 111.

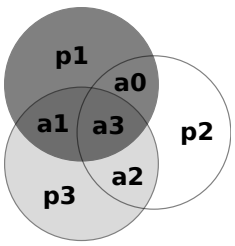
1. Calculer la distance de Hamming du code a reepeetition 3.
2. Combien d'erreurs peuvent e^{a}tre deetecteees ? Combien peuvent e^{a}tre corrigeees ?
3. Vous recevez les messages suivants :
   111000111
   110000011011
   Indiquez si des erreurs se sont produites et corrigez-les le cas eecheeant.
4. Vous disposez d'un canal de communications dont la probabilitee d'intervertir un bit du message est de $p_e = 0.001$ (pour tout n et m distincts les probabilitees d'avoir une erreur sur le bit n et une erreur sur le bit m sont consideereees indeependantes). Vous utilisez un code a reepeetition 3 pour envoyer un message original de 100 bits. Quelle est la probabilitee pour que le destinataire ree§oive le bon message avant correction ? Apreš correction ?

5 Code de Hamming (7,4)

Le code de Hamming (7,4) transmet 4 bits d'informations en utilisant 3 bits de paritee. Pour encoder le mot $a_3a_2a_1a_0$ on ecrira $p_1p_2a_0p_3a_1a_2a_3$ avec $p_1 = a_0 \oplus a_1 \oplus a_3$, $p_2 = a_0 \oplus a_2 \oplus a_3$ et $p_3 = a_1 \oplus a_2 \oplus a_3$.

On peut reesumer la couverture des bits de paritee avec la table ou avec le diagramme de Venn suivants :

	a_0	a_1	a_2	a_3
p_1	1	1	0	1
p_2	1	0	1	1
p_3	0	1	1	1



1. En s'appuyant sur le diagramme de Venn, si l'on change un bit des donneees (a_0 a a_3), combien de bits de paritee changent neecessairement ?
2. Quelle est donc la distance de Hamming du code ? Combien d'erreurs peuvent e^{a}tre deetecteees ? Combien peuvent e^{a}tre corrigeees ?
3. Vous recevez les messages suivants :
   0001111
   1110010
   1101110
   0011111.
   Indiquez si une erreur s'est produite et corrigez le mot ree§u le cas eecheeant.

Architecture des ordinateurs - TD 04

1 Codage excédent 127

La représentation sur 8 bits en excédent 127 consiste à encoder la valeur $x + 127$ en binaire. Par exemple pour encoder -27 on prendra $1100100_2 = 100_{10}$.

1. Quelle est la valeur minimale et maximale codable en excédent 127 sur 8 bits ?
2. Donner la représentation en excédent 127 des nombres suivants : 10, 23 et -40
3. Calculer $1001_{exc} + 11000011_{exc}$ en excédent 127 :
   (a) en convertissant chaque terme en décimal, puis en convertissant le résultat en excédent 127.
   (b) proposer une méthode qui ne nécessite pas le passage en décimal.

2 Représentation des réels

1. Donnez la plus grande et la plus petite valeur strictement positives représentables en simple précision normalisé, c'est-à-dire sur un mot de 32 bits dont 1 bit de signe, 8 bits d'exposant et 23 bits de mantisse. Même question avec un nombre dénormalisé.
   Pour simplifier, dans la suite on utilisera la représentation suivante :

Signe	Exposant	Mantisse
1 bit	4 bits	11 bits

2. Exprimez les nombres décimaux suivants (on utilisera l'arrondi par défaut si nécessaire) :
   1.5
   -0.125
   153.75
   -0.2
3. Convertissez en décimal les nombres flottants suivants :
   0 1110 00000110010
   1 0001 00101100000
   0 0000 00100100000
   0 1111 10000000001
4. Quelle est la représentation en simple précision (sur 32 bits) des nombres suivants, exprimés en double précision (sur 64 bits) :
   4004000000000000_{16}
   $37E8000000000000_{16}$
   $C800000000000000_{16}$

3 Sommes de flottants

1. Effectuez les calculs suivants en utilisant la représentation des réels :
   0 0101 $10111000011 + 0$ 0011 00011110101
   1 1101 $10010001000 + 0$ 0110 10000000000

4 Multiplication de flottants

1. Effectuez les multiplications suivantes :
   0 1001 00110000000×0 0111 10000000000
   1 1100 00000000000×0 0110 01000000000
2. Représentez les nombres 3 et $\frac{1}{3}$ en utilisant les arrondis au plus près, vers 0, vers $+\infty$ et vers $-\infty$. Donnez ensuite les valeurs décimales correspondantes.
3. Donnez le résultat de la multiplication de ces deux nombres.

Architecture des ordinateurs - TD 05

1 Algèbre de Boole : simplifications

1. Démontrer les propriétés suivantes :

$$X(\overline{X}+Y) = XY \tag{1}$$
$$(X+Y)(X+Z) = X+YZ \tag{2}$$
$$XY + X\overline{Y} = X \tag{3}$$
$$(X+Y)(X+\overline{Y}) = X \tag{4}$$
$$XY + \overline{Y} = X + \overline{Y} \tag{5}$$

2. Simplifier les expressions suivantes :

$$AB\overline{C} + \overline{(AB\overline{C})} \tag{6}$$
$$(AB + C\overline{D})(AB + \overline{D}E) \tag{7}$$
$$A + \overline{B}C + \overline{D}(A + \overline{B}C) \tag{8}$$
$$A\overline{B}(C+D) + \overline{(C+D)} \tag{9}$$
$$(\overline{(EF)} + AB + \overline{CD})(EF) \tag{10}$$
$$(AB+C) + (D+EF)\overline{(AB+C)} \tag{11}$$

3. Donner le tableau de Karnaugh correspondant à la table de vérité ci-dessous. Simplifier F.

P	Q	F
0	0	1
0	1	1
1	0	0
1	1	1

4. Donner le tableau de Karnaugh correspondant à la table de vérité ci-dessous. Simplifier F.

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Architecture des ordinateurs - TD 06

1 Afficheur 7 segments

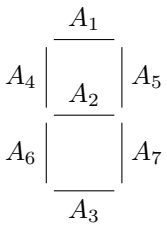
On souhaite réaliser un circuit pour afficher les chiffres de 0 à 9 sur un afficheur 7 segments (voir figure ci-dessous). Un afficheur 7 segments est composé de 7 leds commandées par 7 signaux distincts $A_1 \dots A_7$. Le chiffre à afficher est codé en BCD, chaque bit correspond à un signal b_3 à b_0.

Nous souhaitons réaliser le circuit permettant de commander le signal A_3 correspondant à l'allumage du segment inférieur.

Codage BCD

déc.	$b_3b_2b_1b_0$
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Afficheur 7 segments



1. Écrire la table de vérité du signal A_3. Lorsque la valeur de A_3 est indeterminée écrire X (par exemple pour B=13).
2. Écrire le tableau de Karnaugh correspondant à la table de vérité de A_3.
3. Simplifier l'expression. Proposer un circuit pour commander A_3.

2 Circuits basiques

1. Fabriquer avec uniquement des portes NOR les circuits suivants :
   — porte NON
   — porte OU
   — porte ET
   Qu'en deduisez vous ?
2. Fabriquer avec uniquement des portes NAND les circuits suivants :
   — porte NON
   — porte OU
   — porte ET
   — porte OU à quatre entrées
   — porte ET à quatre entrées
3. Donner un circuit permettant de tester l'égalité de deux nombres de 4 bits.
4. Soit deux nombres positifs : $A = a_3a_2a_1a_0$ et $B = b_3b_2b_1b_0$, implémenter le circuit de la fonction $C(A, B) = A < B$.

3 Circuits additioneurs

1. Un demi additionneur est composé de :
   — deux entrées : les bits a et b
   — deux sorties : s la somme des deux bits et r l'éventuelle retenue.
   Donner les expressions booléennes de s et r en fonction de a et b.
2. Proposer un circuit pour le demi additionneur.
3. Un additionneur complet est composé de :
   — trois entrées : les bits a et b et r_0 la retenue propagé par l'additionneur précédent.
   — deux sorties : $s = a + b + r_0$ et r_1 l'éventuelle retenue lors du calcul de s.
   Construire un circuit pour l'additionneur complet. Pour cela utiliser deux demi-additionneur ainsi qu'une porte OU.
4. Additionneur 4 bits. Un additionneur 4 bits possède 8 signaux en entrée et 5 signaux en sortie :
   — entrées : $A = a_3a_2a_1a_0$ et $B = b_3b_2b_1b_0$ deux nombres codés sur 4 bits chacun.
   — sorties : $C = c_3c_2c_1c_0$ et r. C est la somme de A et B et r l'éventuelle retenue.
   En utilisant les circuits des questions précédentes proposer un circuit pour l'additionneur 4 bits.