Codage des informations numériques

I. Les systèmes de numération

I.1. Numération décimale

Ce système de numération, usuel dans la vie quotidienne, dispose de dix symboles (les chiffres) :

0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

On travaille alors en **base 10**.

Exemple : $7239 = 7.10^3 + 2.10^2 + 3.10^1 + 9.10^0$

De manière générale, un nombre s'écrivant $N = a_{n-1}...a_1a_0$ (les a_i représentent les n chiffres) dans une base B (on dispose de B symboles) s'écrit :

$$N = a_{n-1}B^{n-1} + a_{n-2}B^{n-2} + \ldots + a_1B^1 + a_0B^0$$

On note alors $N = (a_{n-1}...a_1a_0)_B$. La base B est notée en indice, codée en décimal.

I.2. Numération binaire

La numération en base deux (ou **numération binaire**) utilise deux symboles 0 et 1. Cette base est très commode pour distinguer les deux états logiques fondamentaux.

On écrit :

$(a_{n-1}a_{n-2}...a_1a_0)_2 = a_{n-1}.2^{n-1} + \ldots + a_0.2^0$ (expression de droite écrite dans la base 10 et $a_i \in \{0, 1\}$).

Exemple : $(4)_{10} = 1.2^2 + 0.2^1 + 0.2^0 = (100)_2$

Un nombre à n chiffres en base deux distingue 2^n états.

Un état binaire est appelé **bit** (contraction de *binary digit*). Un bit prend les valeurs 0 ou 1.

Les puissances successives de 2 (1, 2, 4, 8, 16, 32, 64, 128, 256,...) sont appelées **poids binaires**. En général, le poids du bit de rang n est 2^n (**attention, on commence toujours au rang 0**).

Le bit de poids le plus fort est appelé MSB (***M**ost **S**ignificant **B**it*).

Le bit de poids le plus faible est appelé LSB (***L**ess **S**ignificant **B**it*).

I.3. Numération octale

Ce système utilise 8 symboles : 0, 1, 2, 3, 4, 5, 6, 7. Il n'est plus guère employé aujourd'hui, puisqu'il servait au codage des nombres dans les ordinateurs de première génération.

$$(N)_8 = a_{n-1}..a_0, \ (N)_{10} = a_{n-1}8^{n-1} + a_{n-2}8^{n-2} + \ldots + a_18^1 + a_08^0$$

I.4. Numération hexadécimale

Le développement des systèmes microprogrammés (mini- et micro-ordinateurs) a favorisé l'utilisation de ce code. Il comporte 16 symboles : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

$$(N)_{16} = a_{n-1}..a_0, \ (N)_{10} = a_{n-1}16^{n-1} + a_{n-2}16^{n-2} + \ldots + a_116^1 + a_016^0$$

Un **quartet**, ou digit hexadécimal, évolue entre 0 et 15 (en base 10) soit 0 et F en hexadécimal. L'assemblage de 2 quartets forme un **octet** qui varie de 0 à 255 (en décimal).

Pour indiquer la base 16, on peut la noter en indice suivant la manière générale. Mais dans la pratique on utilise une autre notation. On place le caractère $ (dollar) devant le nombre ou H derrière.

Exemple

$(AA)_{16}$ = AAH = $AA = $A.16^1+A.16^0 = 10.16+10.1 = (170)_{10}$

II. Changements de base - Conversions

II.1. Conversion décimal vers binaire

Il existe plusieurs moyens d'effectuer une telle conversion :

- par soustractions successives des poids binaires dans la différence de l'étape précédente ;
- par divisions successives par 2 du dividende de l'étape précédente. Les restes correspondants sont les bits consécutifs. C'est terminé au premier dividende nul (que l'on de compte pas).

II.2. Conversion décimal vers hexadécimal

On reprend les deux méthodes précédentes avec des poids hexadécimaux ou en divisant par 16.

II.3. Décomposition d'un nombre décimal en octal

Les mêmes principes s'appliquent aussi.

II.4. Toutes les conversions vers le décimal

Dans tous les cas, il n'y a rien de particulier à ajouter. Le principe de conversion est directement attaché à la manière dont on écrit un nombre dans une base donnée (Cf. définition).

$$(N)_B = a_{n-1}.B^{n-1} + \ldots + a_0.B^0 \text{ où } B \text{ est codé en décimal}$$

La conversion est réalisée automatiquement dans la mesure où le résultat est écrit directement dans la base dix.

II.5. Les conversions directes (sans passer par le décimal)

Dans les bases usuelles (2, 8 et 16) utilisées dans les systèmes numériques, les conversions peuvent être réalisées par exploitation de propriétés particulières aux nombres de ces bases.

II.5.1. Binaire vers hexadécimal

Un nombre hexadécimal est « découpable » en quartets facilement codables en binaire. Donc, pour convertir du binaire en hexadécimal, on divise le nombre binaire en « tranches de quatre » en partant de la droite. Chacun des « paquets » est ensuite converti en hexadécimal. Cette méthode revient à fractionner en décompositions successives.

Exemple : $(110101110001)_2 = (\underline{1101}\ \underline{0111}\ \underline{0001})_2 = D71H$

Explication : la mise en paquet revient à effectuer une série de factorisations partielles de la base de destination. Ici c'est $16 = 2^4$. Les résidus constituent les chiffres de la conversion. Dans l'exemple précédent, cela donne :

$$\begin{aligned}(110101110001)_2 &= 1.2^{11} + 1.2^{10} + 0.2^9 + 1.2^8 + 0.2^7 + 1.2^6 + 1.2^5 + 1.2^4 + 0.2^3 + 0.2^2 + 0.2^1 + 1.2^0 \\ &= \{1.2^3+1.2^2+0.2^1+1.2^0\}.(2^4)^2 + \{0.2^3+1.2^2+1.2^1+1.2^0\}.2^4 + \{0.2^3+0.2^2+0.2^1+1.2^0\} \\ &= 13.16^2 + 7.16^1 + 1.16^0\end{aligned}$$

II.5.2. Hexadécimal vers binaire

C'est le processus directement inverse, on écrit chaque quartet sur 4 bits en complétant éventuellement avec des zéros.

Exemple : BC34H = $(1011[B]\ 1100[C]\ 0011[3]\ 0100[4])_2 = (1011\ 1100\ 0011\ 0100)_2$

II.5.3. Binaire vers octal et inversement

On reprend les mêmes principes, sachant que $8 = 2^3$ (en factorisant $8 = 2^3$).

III. Codage des nombres

Un code constitue une correspondance entre des symboles et des objets à désigner. Les codes du §I sont pondérés : dans une base de travail donnée, la valeur d'un rang donné est un multiple par la base de celle du rang inférieur. D'autres codes ne sont pas pondérés, c'est à dire que la position d'écriture ne correspond pas à un poids des autres. Ils ne permettent d'effectuer d'opérations arithmétiques.

III.1. Codes pondérés

III.1.1. Code naturel

Le code binaire naturel et ses dérivés (octal et hexadécimal) répondent aux règles classiques de l'arithmétique des nombres positifs (on peut calculer).

III.1.2. Code décimal codé binaire (DCB)

Dans ce codage (BCD, *Binary Coded Decimal* en anglais), chaque digit décimal est écrit en binaire puis tous sont juxtaposés. Cette représentation est commode pour traiter les nombres dans le mode de représentation le plus adapté à l'opérateur humain (lors d'un affichage par exemple).

Exemple : 7239 = $(0111\ 0010\ 0011\ 1001)_{DCB}$. = $(1110001000111)_2$.

III.1.3. Représentation des nombres négatifs : code complément à 2

Le codage en complément à 2 permet d'effectuer des soustractions.

Le **complément à 1** ou **complément restreint** (CR) d'un nombre est obtenu en inversant chaque bit. On a donc $x + CR(x) = 2^n-1$ (n est le nombre de bits de x).

Exemple : CR(22) = CR(10110) = 01001 (= $9 = 2^5 - 1 - 22$).

Il vient donc : $x + CR(x) + 1 = 2^n$. 2^n correspond à un cycle et n'est pas interprété (le bit extérieur à la taille de codage est ignoré). Par conséquent, la valeur négative $-x$ est représentée par $CR(x) + 1$.

La valeur $CR(x) + 1$ est appelée complément vrai (CV) ou complément à 2 de x.

La codage par **complément à 2** (***Tableau 1***) permet la représentation des nombres négatifs utilisée dans les calculateurs. Le signe est le bit de poids fort : 0 pour le signe + (compatible avec le codage non signé) et 1 pour le signe négatif.

	c_3	c_2	c_1	c_0
7	0	1	1	1
6	0	1	1	0
5	0	1	0	1
4	0	1	0	0
3	0	0	1	1
2	0	0	1	0
1	0	0	0	1
0	0	0	0	0
-1	1	1	1	1
-2	1	1	1	0
-3	1	1	0	1

Tableau 1

Exemple : 7 – 4 = 7 + CV(4) = 0111 + CV(0100) = 0111 + 1011 + 1 = (1)0011 = +3.

III.2. Codes non pondérés

III.2.1. Code cyclique : code binaire réfléchi ou code Gray

Dans ce code, un seul bit change entre deux valeurs adjacentes. Il est employé dès que l'on doit représenter une l'évolution réelle des variables où une seule change à un instant (exemple dans les tables de Karnaugh).

III.2.2. Codes redondants : détecteur et correcteurs d'erreurs (pour info)

Ces codes sont utilisés pour contrôler les transmissions. Ils permettent de détecter une erreur de bit lors d'une transmission et parfois même une correction de l'erreur. Pour l'essentiel, ces codes ne sont pas pondérés.

IV. Extension aux codes non numériques

Pour manipuler d'autres éléments que des nombres, il est nécessaire de les coder. Le plus connu de ces codes, et le plus utilisé en particulier dans le monde informatique, est le code ASCII (*American Standard Code for Information Interchange*) présenté dans le ***Tableau 2***.

Table des caractères de contrôle (00 à 31)

ASCII	Caract.	Signification	ASCII	Caract.	Signification
00	NUL	*null*, nul	16	DLE	*data link escape*, échap. liaison données
01	SOH	*start of heading*, début d'en-tête	17	DC1	*device control 1*, commande unité 1
02	STX	*start of text*, début de texte	18	DC2	*device control 2*, commande unité 2
03	ETX	*end of text*, fin de texte	19	DC3	*device control 3*, commande unité 3
04	EOT	*end of transmission*, fin de transmission	20	DC4	*device control 4*, commande unité 4
05	ENQ	*enquiry*, interrogation	21	NAK	*negative acknowledge*, acc. récep. nég.
06	ACK	*acknowledge*, accusé de réception	22	SYN	*synchronous idle*, inactif synchronisé
07	BEL	*bell*, sonnerie	23	ETB	*end of transmission block*, fin tran. bloc
08	BS	*backspace*, espacement arrière	24	CAN	*cancel*, annuler
09	HT	*horizontal tabulation*, tabulation horiz.	25	EM	*end of medium*, fin du support
10	LF	*line feed*, saut de ligne	26	SUB	*substitute*, substitut
11	VT	*vertical tabulation*, tabulation verticale	27	ESC	*escape*, échappement
12	FF	*form feed*, saut de page	28	FS	*file separator*, séparateur de fichiers
13	CR	*carriage return*, retour chariot	29	GS	*group separator*, séparateur de groupes
14	SO	*shift out*, hors code	30	RS	*record separator*, sép. d'enregistr.
15	SI	*shift in*, en code	31	US	*unit separator*, séparateur d'unités

Table des caractères imprimables (32 à 127) — ou table ASCII standard

ASCII	Caractère.
32	SP (*space*, espace)
33	!
34	"
35	#
36	$
37	%
38	&
39	'
40	(
41	)
42	*
43	+
44	,
45	-
46	.
47	/
48	0
49	1
50	2
51	3
52	4
53	5
54	6
55	7
56	8
57	9
58	:
59	;
60	<
61	=
62	>
63	?

ASCII	Caractère
64	@
65	A
66	B
67	C
68	D
69	E
70	F
71	G
72	H
73	I
74	J
75	K
76	L
77	M
78	N
79	O
80	P
81	Q
82	R
83	S
84	T
85	U
86	V
87	W
88	X
89	Y
90	Z
91	[
92	\
93	]
94	^
95	_

ASCII	Caractère
96	`
97	a
98	b
99	c
100	d
101	e
102	f
103	g
104	h
105	i
106	j
107	k
108	l
109	m
110	n
111	o
112	p
113	q
114	r
115	s
116	t
117	u
118	v
119	w
120	x
121	y
122	z
123	{
124	|
125	}
126	~
127	DEL (*delete*, sup.)

Tableau 2 : codage ASCII.