ISN S1 : Codage des informations : représentation des nombres

Vus de l'extérieur, les ordinateurs et les programmes que nous utilisons tous les jours permettent de mémoriser, de transmettre et de transformer des nombres, des textes, des images, des sons, etc.

Un ordinateur ne manipule que des 0 et des 1. Les nombres, les textes, les images, les sons, etc. mémorisés, transmis, transformés par les ordinateurs doivent être représentés comme des suites de 0 et de 1. Une telle valeur, 0 ou 1, s'appelle un **booléen**, un **chiffre binaire** ou encore un **bit** (*binary digit)*. Une suite de bits, par exemple 0000110100 , est appelé un mot.

1. Les systèmes de numérations : représentation des entiers naturels:

a) Numération décimale :

La numération décimale utilise 10 chiffres : **0, 1, 2, 3, 4, 5, 6, 7, 8 et 9**.

L'écriture du nombre $(275)_{10}$ se traduit par $275= 2x10^2 + 7x10^1 + 5*10^0$, on dit que 275 est la représentation en base 10 appelée écriture décimale .

1) Ecrire une égalité semblable pour les nombres 1 234.

b) Numération binaire :

L'informatique utilise des courants électriques, des aimantations, des rayons de lumière...

Chacun de ces phénomènes met en jeu deux états possibles. :

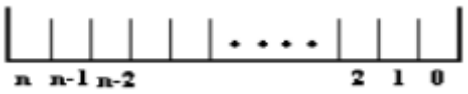
- Tension nulle ou tension non nulle (5V par ex),
- Aimantation dans un sens ou dans l'autre sens,
- Lumière ou pas de lumière.

Il suffit de deux chiffres pour traduire ces états : c'est la numération binaire qui utilise les chiffres **0** et **1**.

L'écriture du nombre $(1110)_2$ se traduit par $1110 = 1x2^3+1x2^2+1x2^1+0x2^0=8+4+2+0=14$, on dit que 1110 est la représentation en base 2 de 14.

Chaque chiffre binaire se nomme BIT (de BInary digiT)

Voici le schéma d'une mémoire à n+1 bits (au maximum 8 bits dans un micro-ordinateur):



Les cases du schéma représentent les bits, le chiffre marqué en-dessous d'une case indique la puissance de 2 à laquelle est associé ce bit (on dit aussi rang du bit).

- Le bit de **rang 0** est appelé le bit de **poids faible.**
- Le bit de **rang n** est appelé le bit de **poids fort**.

2) Traduire en écriture décimale les nombres binaires 0101.

Un nombre binaire de huit chiffres est un octet.

3) Quel est le plus grand nombre que l'on peut représenter avec un octet ?

4) Justifier qu'avec un **mot** de **n** bits, on peut représenter les nombres de 0 à 2^n - 1.

Pour multiplier par dix un entier naturel exprimé en base 10, il suffit d'ajouter un 0 à sa droite, par exemple 12 x 10 = 120

5) Quelle est l'opération équivalente pour les entiers naturels exprimés en base 2? Exprimer en base 2 les nombres 3, 6, et 12 pour illustrer cette réponse.

Les multiples et sous-multiples des unités employées en informatique

	Valeur exacte	en pratique dans le système international SI	Exemples d'utilisation
1 kilo-octet(Ko)	2^{10} = 1024 octets	10^3 octets	un modem téléphonique débite à 56,6 Kbps, un téléphone GSM à 9.6 Kbps une page de texte correspond environ 2 Ko.
1 Méga-octet(Mo)	2^{10} Ko = 1 048 576 octets	10^6 octets	un CDROM contient 640 Mo
1 Giga-octet (Go)	2^{10} Mo =1 073 741 824 octets	10^9 octets	1 DVD a une capacité de stockage de 4,7 Go 10 mètres de livres dans une bibliothèque tiendraient sur environ 1 Go un étage rempli complètement de journaux équivaut à 100 Go, c'est à dire au disque dur d'un PC
1 Téra-octet (To)	2^{10} Go =1 099 511 627 776 octets	10^{12} octets	une grande bibliothèque publique tiendrait sur 1 To environ l'ordinateur Blue Gene/L d'IBM calcule à la vitesse de 36Tflops (36 Téra opérations par seconde) YouTube annonce plus de mille milliards de vidéos vues

c) Numération hexadécimale :

La numération hexadécimale utilise 16 chiffres : **0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E et F**.
A est donc le nombre "10", B le nombre "11"....
L'écriture du nombre $(23A)_{16}$ se traduit par $23A= 2x16^2 + 3x16^1 + 10*16^0=512+48+10=570$ on dit que 23A est la représentation en base 16 de 570 .

6) Traduire en nombre décimale $(B8C)_{16}$

Cette numération est utilisée pour les adresses des mémoires. Exemple : B8AC 000F
Elle est aussi utilisée pour coder les couleurs :

000000	0000FF	00FF00	FF0000	FFFF00	FFFFFF
Noir	*Bleu*	*Vert*	*Rouge*	*Jaune*	*Blanc*

7) A quel nombre décimal correspond le nombre FF?

8) Quelle remarque faites-vous par rapport à l'octet?

La taille rapidement encombrante de l'écriture binaire a conduit les informaticiens à faire un usage intensif du système hexadécimal.

2. Changement de base :

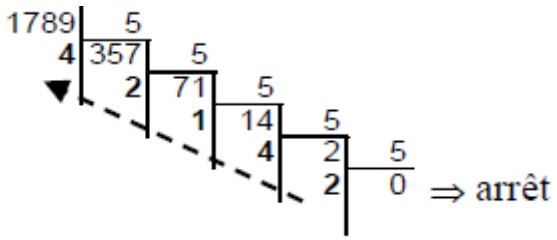
a) Passage de la base b à la base 10:

$(a_k a_{k-1}....a_2 a_1 a_0)_b$ (en base b) désigne l'entier $a_k \times b^k + a_{k-1} \times b^{k-1} + ...+ a_2 \times b^2 + a_1 \times b^1 + a_0 \times b^0$.

9) Donner l'écriture décimale de $(76401)_8$ et $(BA9865)_{12}$.

b) Passage de la base 10 à la base b:

Pour écrire les entiers naturels en base *b*, on a besoin de *b* chiffres. Quand on a *n* objets, on les groupe par paquets de *b*, qu'on regroupe à leur tour en paquets de *b* paquets, etc.
Autrement dit, on fait une succession de divisions par *b*, jusqu'à obtenir un quotient égal à 0.



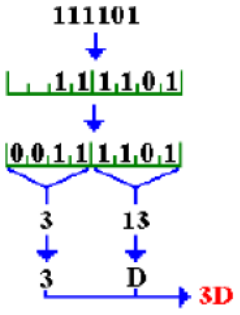
donc 1789 s'écrit $(24124)_5$

10) Trouver la représentation en base 16 puis en base 2 des nombres 6725 et 18 379 (écrits en base 10) (écrits en base 10)

c) Passerelle entre la base 2 et la base 16:

Pour passer de la base 2 à la base 16 :

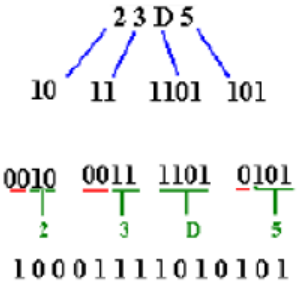
- On décompose ce nombre par tranches de 4 bits à partir du bit de poids faible ($2^4 = 16$)
- On complète la dernière tranche (celle des bits de poids forts)par des 0 s'il y a lieu.
- On convertit chaque tranche en son symbole de la **base 16.**
- On réécrit à sa place le nouveau symbole par changements successifs de chaque groupe de 4 bits.



Donc $(111101)_2 = (3D)_{16}$

Pour passer de la base 16 à la base 2 :
Cette conversion est l'opération inverse de la précédente.

- On convertit chaque symbole hexadécimal du nombre en son écriture binaire (nécessitant au plus 4 bits).
- Pour chaque tranche de 4 bits, on complète les bits de poids fort par des 0 s'il y a lieu.
- On regroupe toutes les tranches de 4 bits à partir du bit de poids faible, sous forme d'un seul nombre binaire.



Donc $(23D5)_{16} = (10001111010101)_2$

11) Convertir $(1100011010)_2$ en base 16 et $(7CF21)_{16}$ en base 2.

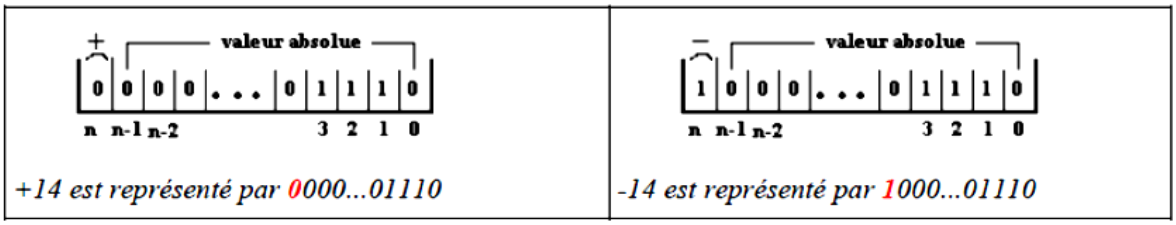
3. Représentation des entiers relatifs (entiers signés):

Pour représenter les entiers relatifs en notation binaire, on doit étendre la représentation des positifs aux nombres négatifs.

1ère solution avec les premières machines :

Une solution est de réserver le bit de poids fort pour le signe de l'entier à représenter (1 pour – et 0 pour +) et d'utiliser les autres pour représenter sa valeur absolue:

Exemple du codage en binaire signé des nombres **+14** et **-14** :

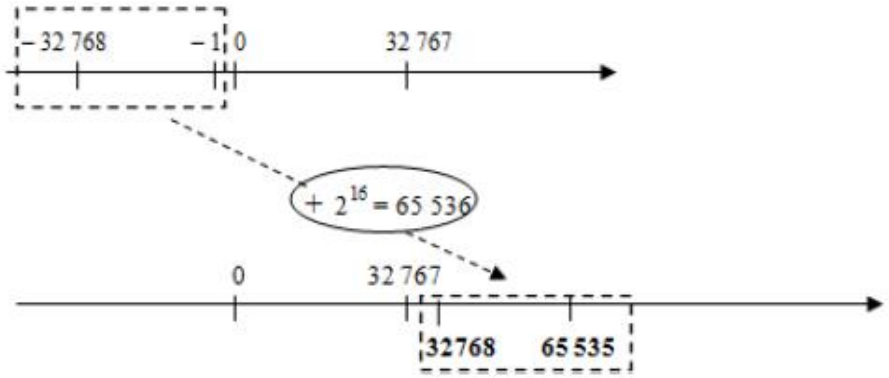


12) Ainsi, avec des mots de 16 bits, en utilisant 1 bit pour le signe et 15 pour la valeur absolue, quels sont les entiers que l'on peut représenter?

Ce codage n'est pas utilisé en pratique. En effet le traitement spécifique du signe coûte cher en circuits électroniques et en temps de calcul. C'est une version améliorée qui est utilisée dans la plupart des calculateurs : elle se nomme le complément à deux.

2ème solution : codage sur 16 bits :

Si on utilise des mots de 16 bits, cette fois-ci les nombres 1000 0000 0000 0000 et 0000 0000 0000 0000 représente 2 nombres distincts donc on peut représenter les entiers relatifs compris entre $-2^{15} = -32\,768$ et $2^{15}-1 = 32\,767$. L'idée est de transporter les entiers strictement négatifs dans le monde des entiers naturels et de laisser intact les entiers positifs (voir ci-dessous) :



Avec cette méthode, tous les entiers relatifs sont donc représentés par un entier naturel.

13) Comment est alors représenté l'entier négatif -1?

Cette manière de représenter les entiers relatifs s'appelle la notation en **complément à deux**.

14) Quels entiers relatifs peut-on représenter avec des mots de 8 bits? Combien sont-ils?

4. Codage d'un texte sur un ordinateur :

Un texte est une liste de caractères: les lettres majuscules et minuscules, les chiffres, les signes de ponctuation, des symboles mathématiques...

Au commencement, chaque caractère était identifié par un code unique qui est un entier naturel et la correspondance entre le caractère et son code était appelée un Charset.

Le code n'étant pas utilisable tel quel par un ordinateur qui ne comprend que le binaire, il fallut donc représenter les codes par des octets, et cela fut appelé Encoding.

Dans de nombreux grimoires anciens on découvre le code ASCII qui était utilisé pour représenter du texte en informatique. ASCII signifiait American Standart Code for Information Interchange.

Il paraît que ce code est toujours en usage...

Le code ASCII (standardisé en 1963) a été conçu pour représenter des textes écrits en anglais, il n'y a donc pas d'accents, de tréma...

En ASCII on code 95 caractères: les 10 chiffres, les 26 lettres minuscules, les 26 lettres majuscules, 32 symboles (@, <, >..), l'espace; et 33 symboles de mise en page (passage à la ligne, saut de page...), soit 128 caractères.

Il faut donc 7 bits, mais on code sur 1 octet, le premier bit étant toujours 0 et sert au contrôle de parité (pour éviter des erreurs).

Dans un texte le premier octet donne la longueur du texte, le dernier est 00000000 indique la fin du texte. Un texte de n caractères occupe donc une place de n octets.

Le code ASCII est donné sur la page suivante.

15) Dans notepad++, écrire le texte suivant « Codage de texte sur un ordinateur »

Enregistrer sous codetxt avec l'extension .text, regarder ses propriétés et la place occupée.

Recopier dans un éditeur de texte (word ou openoffice.org) regardé la place occupée et conclure.

Sur la page suivante, vous avez un tableau correspondant au code Ascii. Quel est le code binaire correspondant à la lette I ?

A quoi correspond le code 01010011 ?

Coder en binaire le mot : Codage.

Que signifie : 01000010 01110010 01100001 01110110 01101111 00100001.

Avec le développement du web, pour permettre de coder toutes les langues on a ensuite créé la norme Unicode (L'Unicode est une table de correspondance Caractère-Code (Charset), et l'UTF-8 est l'encodage correspondant (Encoding) le plus répandu).

UTF : Universal Transformation Format

Le code est compatible avec la norme ASCII.

En UTF-8 les caractères sont codés sur 1, 2 ,3 ou 4 octets (les plus courants étant codés sur 1 octet). Les bits de poids forts (les premiers) vont indiquer le nombre d'octets utilisés, par exemple 01100001 représente le a (commence par 0 car a est codé sur 1 octet).

Le caractère ε est codé U+025B ce qui veut dire que le nombre attribué est 025B en hexadécimal soit en binaire : 10 0101 1011 ce nombre va donc s'écrire sur 2 octets.

Soit en UTF-8: 1100 1001 1001 1011; on écrit au début 110 pour dire que ce caractère est codé sur 2 octets, puis 10 commence le deuxième octet, puis les autres chiffres donnent le nombre 10 01011011; et le 0 vert complète le nombre de bits.

16) Coder en UTF-8 le caractère ♫ (codage U+266B)

Pour les plus rapides, voici un jeu Binary-Game

Dec	Hex	Char	Remarque	Dec	Hex	Char	Remarque	Dec	Hex	Char	Remarque	Dec	Hex	Char	Remarque
0	0	NA	Fin de chaîne	64	40	@		128	80	NA		192	C0	À	
1	1	NA		65	41	A		129	81	NW		193	C1	Á	
2	2	NA		66	42	B		130	82	‚	True type	194	C2	Â	
3	3	NA		67	43	C		131	83	ƒ	True type	195	C3	Ã	
4	4	NA		68	44	D		132	84	„	True type	196	C4	Ä	
5	5	NA		69	45	E		133	85	…	True type	197	C5	Å	
6	6	NA		70	46	F		134	86	†	True type	198	C6	AE	
7	7	NA		71	47	G		135	87	‡	True type	199	C7	Ç	
8	8	NA		72	48	H		136	88	ˆ	True type	200	C8	È	
9	9	NA		73	49	I		137	89	‰	True type	201	C9	É	
10	0A	NA	Fin de ligne	74	4A	J		138	8A	Š	True type	202	CA	Ê	
11	0B	NA		75	4B	K		139	8B	‹	True type	203	CB	Ë	
12	0C	NA		76	4C	L		140	8C	OE	True type	204	CC	Ì	
13	0D	NA		77	4D	M		141	8D	NW		205	CD	Í	
14	0E	NA		78	4E	N		142	8E	Ž		206	CE	Î	
15	0F	NA		79	4F	O		143	8F	NW		207	CF	Ï	
16	10	NA		80	50	P		144	90	NW		208	D0	Ð	
17	11	NA		81	51	Q		145	91	‘		209	D1	Ñ	
18	12	NA		82	52	R		146	92	’		210	D2	Ò	
19	13	NA		83	53	S		147	93	“	True type	211	D3	Ó	
20	14	NA		84	54	T		148	94	”	True type	212	D4	Ô	
21	15	NA		85	55	U		149	95	•	True type	213	D5	Õ	
22	16	NA		86	56	V		150	96	–	True type	214	D6	Ö	
23	17	NA		87	57	W		151	97	—	True type	215	D7	×	
24	18	NA		88	58	X		152	98	˜	True type	216	D8	Ø	
25	19	NA		89	59	Y		153	99	™	True type	217	D9	Ù	
26	1A	NA		90	5A	Z		154	9A	š	True type	218	DA	Ú	
27	1B	NA		91	5B	[		155	9B	›	True type	219	DB	Û	
28	1C	NA		92	5C	\		156	9C	oe	True type	220	DC	Ü	
29	1D	NA		93	5D	]		157	9D	NW		221	DD	Ý	
30	1E	NW		94	5E	^		158	9E	ž		222	DE	Þ	
31	1F	NA		95	5F	_		159	9F	Ÿ	True type	223	DF	ß	
32	20	"espace"	Barre espace	96	60	`		160	A0	NW		224	E0	à	
33	21	!		97	61	a		161	A1	¡		225	E1	á	
34	22	"		98	62	b		162	A2	¢		226	E2	â	
35	23	#		99	63	c		163	A3	£		227	E3	ã	
36	24	$		100	64	d		164	A4	¤		228	E4	ä	
37	25	%		101	65	e		165	A5	¥		229	E5	å	
38	26	&		102	66	f		166	A6	¦		230	E6	ae	
39	27			103	67	g		167	A7	§		231	E7	ç	
40	28	(		104	68	h		168	A8	¨		232	E8	è	
41	29	)		105	69	i		169	A9	©		233	E9	é	
42	2A	*		106	6A	j		170	AA	ª		234	EA	ê	
43	2B	+		107	6B	k		171	AB	«		235	EB	ë	
44	2C	,		108	6C	l		172	AC	¬		236	EC	ì	
45	2D	-		109	6D	m		173	AD	-		237	ED	í	
46	2E	.		110	6E	n		174	AE	®		238	EE	î	
47	2F			111	6F	o		175	AF	¯		239	EF	ï	
48	30	0		112	70	p		176	B0	°		240	F0	ð	
49	31	1		113	71	q		177	B1	±		241	F1	ñ	
50	32	2		114	72	r		178	B2	²		242	F2	ò	
51	33	3		115	73	s		179	B3	³		243	F3	ó	
52	34	4		116	74	t		180	B4	´		244	F4	ô	
53	35	5		117	75	u		181	B5	µ		245	F5	õ	
54	36	6		118	76	v		182	B6	¶		246	F6	ö	
55	37	7		119	77	w		183	B7	·		247	F7	÷	
56	38	8		120	78	x		184	B8	¸		248	F8	ø	
57	39	9		121	79	y		185	B9	¹		249	F9	ù	
58	3A	:		122	7A	z		186	BA	º		250	FA	ú	
59	3B	;		123	7B	{		187	BB	»		251	FB	û	
60	3C	<		124	7C	|		188	BC	¼		252	FC	ü	
61	3D	=		125	7D	}		189	BD	½		253	FD	ý	
62	3E	>		126	7E	~		190	BE	¾		254	FE	þ	
63	3F	?		127	7F	NW		191	BF	¿		255	FF	ÿ	