Chapitre 2 : codage binaire des nombres et des textes

Animation
1. convertisseur binaire / décimal / hexadécimal

Table des matières

I) codage des nombres

1) codage des nombres entiers positifs

Les nombres entiers positifs sont codés avec leur valeur en **base 2**. Un entier est codé sur **un Mot**. On peut donc représenter les entiers sur des intervalles qui dépendent du type de microprocesseur.
mot de 16 bits : [0 , $2^{16} - 1$]
mot de 32 bits : [0 , $2^{32} - 1$]
mot de 64 bits : [0 , $2^{64} - 1$]
On ne peut donc représenter qu'un petit sous-ensemble de N avec ce système.
Par exemple pour un microprocesseur 16 bits (mot de 16 bits) le chiffre binaire correspondant à $(277)_{10}$ sera : $(0000\ 0001\ 0001\ 0101)_2$

Ex 1 : quel est, pour un microprocesseur de 64 bits, le nombre entier positif maximal que l'on peut coder ?
Réponse : environ $1{,}8 \times 10^{20}$.

2) addition de 2 nombres positifs

Comme en base 10, l'addition de deux nombres binaires s'effectue avec retenue : 1 + 1 = 0 et on retient 1. Le calcul s'effectue du bit de poids le plus faible vers le bit de poids le plus fort.

Exemple : soit un ordinateur manipulant des mots de 8 bits. Quand on effectue la somme de 12 + 5 = 17 l'ordinateur effectue la somme suivante :

```
  0000 1100
+
  0000 0101

=  0001 0001
```

ex 2 : multiplication par deux de 2 nombres entiers positifs
Comment effectuer la multiplication par deux ?
Exemple : nombre sur 4 bits.
en décimal **2x4** = 8
$2_{10} = 0010_2$
$4_{10} = 0100_2$
$(8)_{10} = (1000)_2$
Pour multiplier par 2 un nombre binaire il suffit de rajouter un zéro à droite du nombre !

3) codage des entiers négatifs

La solution choisit pour coder les nombres entiers négatifs est appelé **le complément à 2.**

- on fixe le nombre n de bits
- on sépare le premier bit à gauche noté s et les n-1 bits à droite. Les n-1 bits représentent, en binaire, une valeur positive noté v. **Si s =0**, la valeur du nombre est alors v.
- si **s = 1**, la valeur du nombre vaut $-2^{n-1} + v$.

ex 3 : compléter le tableau suivant pour n = 4

0000	0001	0010	0011	0100	0101	0110	0111
0	1	2	3	4	5	6	7

1000	1001						
-8	-7						

ex 4 : effectuer l'opération (3) + (-4) en binaire et vérifier qu'on trouve le nombre -1

```
  0011
+
  1101
=
```

$(1111)_2 = -2^{n-1}+v = -2^3 + 7 = (-1)_{10}$

Autre manière de voir le complément à 2
$0111_2 = 7_{10}$
On prend la négation de chaque bit (non 1 = 0, opération réalisée avec des portes logiques) et on ajoute 1 à cette valeur
La négation de 0111 est 1000
on ajoute 1: $1000 + 1 = 1001_2 = -7_{10}$

4) les nombres réels

Un ordinateur à une mémoire restreinte, il ne peut, par exemple, stocker tous les chiffres du résultat de $2^{1/2}$. Il a une représentation interne des réels **en notation scientifique**. On appelle ce nombre un nombre à **virgule flottante**. Ces nombres et les calculs sont régit par un standard, **l'IEEE-754** (voir wikipédia pour la définition précise du standard)

Le codage d'un nombre en virgule flottante se fait sous la forme d'un triplet :

- signe s (si s = 0 le nombre entier est positif, si s = 1 le nombre entier est négatif)

- une mantisse M

- un exposant E sous la forme 2^E

Un nombre flottant normalisé a une valeur *v* donnée par la formule suivante :

$$v = s \times 2^E \times m.$$

s = ±1 représente le signe (selon le bit de signe) ;
E est l'exposant avant son décalage de 127 ;
m = 1+*mantisse* M . m représente la partie significative (en binaire), d'où $1 \leq m < 2$ (la *mantisse* M étant la partie décimale de la partie significative, comprise entre 0 et 1)

Exemple : pour le nombre réel $1{,}25 \times 2^{23}$
le signe est positif (s = 0)
la mantisse M = 0,25
l'exposant E = 23
m = 1,M = 1,25

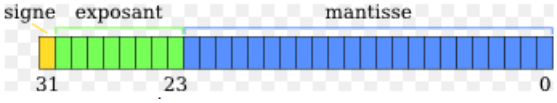
Suivant la puissance du calculateur on code le nombre réel sur :

- 32 bits : $(-1)^s \times 1{,}M \times 2E+-127$. Ce type de codage est appelé codage simple précision.

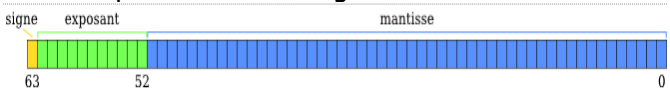
- 64 bits : $(-1)^s \times 1{,}M \times 2E+-1023$. Ce type de codage est appelé codage double précision.

Le nombre de bits pour le signe, la mantisse et l'exposant sont :

- Simple précision : 23 bits de mantisse, 8 bits d'exposant, 1 bit de signe



- Double précision : 52 bits de mantisse, 11 bits d'exposant, 1 bit de signe



Exemple du codage d'un nombre réel avec la norme IEEE 754
On veut coder le nombre réel (-51,375) sur 32 bits (simple précision)

1) le signe négatif est mis dans le bit s = 1

2) on sépare le nombre en deux: la partie entière (51) et la partie fractionnaire ou décimale (0,375)

3) on convertit la partie entière en nombre binaire : $51_{10} = 110011_2$

4) on convertit la partie décimale (ou fractionnaire) en nombre binaire de la façon suivante : on multiplie la partie décimale par 2 jusqu'à obtenir 0

0,375x2 = **0**,75
0,75x2 = **1**,5
0,5x2 = **1**,0
0,0x2 = **0**,0

En effet $0{,}375 = 0\times2^{-1} + 1\times2^{-2} + 1\times2^{-3} + 0\times2^{-4}$
Le nombre réel en base 10 s'écrit en base 2 :
$51{,}375_{10} = 110011{,}0110_2$

5) on décale la virgule à gauche de 5 rangs pour l'écrire sous la forme 1,M. L'exposant de la puissance de 2 vaut 5: E =5
l'écriture du nombre binaire est 1, 10011011000..
la mantisse est M = $(100110110000. \ . \ .)_2$
par conséquent le nombre
$(51{,}375)_{10} = (1, 10011011000)_2 \times 2^5$
Retrouvons ce résultat :
$(1\times2^0 + 1\times2^{-1}+1\times2^{-4} + 1\times2^{-5} + 1\times2^{-7}+1\times2^{-8})\times2^5 = 32 + 16 + 2 + 1 + 0{,}25 + 0{,}125 = 51{,}375$

Pour respecter la norme IEEE 754 on ajoute 127 à l'exposant avant de le coder :
$E = 5 + 127 = (132)_{10} = (10000100)_2$
On complète la mantisse à 23 bits (en simple précision, le nombre réel étant codé en tout avec 32 bits)
$M = (10011011000000000000000)_2$
Le nombre (-51,375)10 s'écrit avec la norme IEEE 754 en simple précision :

1	10000100	10011011000000000000000
signe	exposant	mantisse

Ex 5 : coder le nombre 6,875 avec la norme IEEE 754 en simple précision.
Étape 1 : le signe s = 0 car le nombre est positif
Étape 2 : codage en binaire de la partie entière
$(6)_{10} = (110)_2$
Étape 3 : codage de la partie fractionnaire (après la virgule)
0,875x2 =**1**,75
0,75x2= **1**,5
0,5x2 =**1**,0
0x2= **0**,0
0x2=**0**,0
etc..
Étape 4 : décalage de la virgule pour obtenir l'exposant E et la mantisse:
$(6{,}875)_{10} = (110{,}1110\ldots)_2$, il faut décaler de 2 rangs vers la gauche la virgule pour obtenir un nombre de la forme 1,M
l'exposant E = 2. pour répondre à la norme IEEE 754 il faut ajouter 127 à l'exposant :
$E = 2+127 = (129)_{10} = (10000001)_2$
La mantisse M vaut :
$M = (10111000000000000000000)_2$ (on rappelle que la mantisse est codée sur 23 bits)

Le nombre 6,875 vaut en codage IEEE 754 :

0	10000001	10111000000000000000000
signe	exposant	mantisse

II) codage du texte

1) les différents types de code

Le texte est constitué de caractères (lettre, chiffre signe de ponctuation). Chaque caractère est représenté par un entier. Il existe de nombreux codages des caractères ; les principaux codages pour les occidentaux sont :

1) Le code ASCII (ISO 646)
2) Les codes ISO 8859-1 / ISO 8859-15 (symbole e)
3) Le code Unicode
4) Les codes UTF-8 / UTF-16 / UTF-32

2) le code ASCII

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

L'ASCII (American standard code for information interchange) a été créé au début des années 60. Son principe consiste à associer à chaque lettre, chiffre ou caractère d'un clavier d'ordinateur un entier compris entre 0 et 127, donc représentable sur 7 bits. Avec ce code on peut représenter les chiffres, les lettres latines, et les principaux symboles de ponctuation. Exemple : lorsqu'on tape dans un texte sur la barre d'espace (space) , l'ordinateur enregistre dans sa mémoire vive (RAM) le code hexadécimal $(20)_{hex}$. Le codage binaire sur 7 bits correspondant est : $(010\ 0000)_2 = (32)_{10}$

Remarque: comment insérer un caractère ASCII dans un document Excel?
Outre la saisie d'un caractère au clavier, vous pouvez également utiliser la code de caractère du symbole comme raccourci clavier ou lorsqu'un symbole n'est pas disponible sur le clavier que vous utilisez.
Pour insérer un caractère ASCII des tables ci-dessous, appuyez sur la touche Alt et maintenez-la enfoncée tout en tapant l'équivalent numérique décimal.
Par exemple, pour insérer le symbole du point d'exclamation ! , appuyez sur la touche Alt et maintenez-la enfoncée tout en tapant 33 sur le pavé numérique.

Ex 6: retrouver à l'aide du tableau ci-dessus le codage hexadécimal, décimal et en déduire le nombre binaire correspondant à la lettre L.
Réponse : $(4C)_{hex} = (100\ 1100)_2 = (76)_{10}$

3) le codage ISO 8859-1

Le codage ASCII suffit pour coder un texte anglais mais ne suffit pas pour les autres langues. Par exemple, les lettres accentuées ne figurent pas dans le code ASCII. Le code **ISO 8859-1** a été créé dans les années 80, il est appelé également **'latin-1'** . Il permet de représenter les caractères accentués. A chaque caractère alphanumérique

(lettre, chiffre, ponctuation etc..) est associé un nombre compris entre 0 et 255. On a donc besoin de 8 bits pour représenter l'ensemble des caractères.
Exemple : le codage du point d'interrogation est $(3F)_{hex} = (0011\ 1111)_2$

ex 7: retrouve à l'aide du tableau ci-dessous le code hexadécimal, décimal et binaire de la lettre à.
Réponse : le code de la lettre à est : $(E0)_{hex} = (1110\ 0000)_2 = (224)_{10}$

ISO-8859-1																
	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1x	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2x	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL
8x	PAD	HOP	BPH	NBH	IND	NEL	SSA	ESA	HTS	HTJ	VTS	PLD	PLU	RI	SS2	SS3
9x	DCS	PU1	PU2	STS	CCH	MW	SPA	EPA	SOS	SGCI	SCI	CSI	ST	OSC	PM	APC
Ax	NBSP	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬		®	¯
Bx	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
Cx	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Dx	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
Ex	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
Fx	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

4) L'Unicode

D'autres codages ont été définis pour les autres langues : le chinois, l'arabe etc.. Ces codes sont compatibles avec l'ASCII mais ne le sont pas entre eux. Comment écrire un texte multilingue ? On a créé un **unique codage universel l'Unicode.** Les valeurs entières associées étaient initialement comprises entre 0 et 65235 donc codées sur 16 bits. Cependant ce codage soufre de nombreux défauts (car mis en place par l'homme). Maintenant il est codé sur 32 bits. Toutes les langues connues (sur Terre) sont représentées dans Unicode.
Problème! le é peut se représenter soit par le caractère 'é' du latin-1, soit par la suite de caractère « 'e »

Exemple : de table de caractères UNICODE (Plan multilingue de base (PMB, 0000 à FFFF))

Points de code		Nom officiel du bloc	Commentaires
Début	Fin		
0000	007F	Latin de base	voir norme ISO 646, code ASCII
0080	009F	Non-utilisé	voir plage non-utilisé norme ISO 8859 et ISO 8859-1
00A0	00FF	Supplément Latin-1	voir norme ISO 8859, code ISO 8859-1
0100	017F	Latin étendu A	
0180	024F	Latin étendu B	
0250	02AF	Alphabet phonétique international (API)	Alphabet phonétique international
02B0	02FF	Lettres modificatives avec chasse	
0300	036F	Diacritiques	voir Diacritique
0370	03FF	Grec et copte	
etc...			

Pour pallier les problèmes de l' UNICODE on a construits des codages tels que l'UTF-8 qu'on ne verra pas dans ce cours.

(livre de Mr Dowek)

Ex 8: définir un codage binaire pour coder une information courante comme par exemple les dates de naissance des personnes de la classe. Rechercher un code qui utilise un minimum de bits compte tenu des données de la classe. Puis rechercher comment coder la date de naissance de n'importe quel élève du lycée.

Ex 9: chercher dans Wikipédia combien il y a de caractères codés en UTF-8 pour toutes les langues du monde. Expliquer en 2 lignes le lien entre le codage ASCII et UTF-8.

Ex 10 : rechercher sur internet comment effectuer la multiplication de 2 nombres binaires.

Programme officiel

savoirs	**capacités**
Numérisation L'ordinateur manipule uniquement des valeurs numériques. Une étape de numérisation des objets issus du monde physique est donc indispensable.	Coder un nombre, un caractère au travers d'un code standard, un texte sous forme d'une liste de valeurs numériques.