TRAITEMENT NUMÉRIQUE

Objectifs du COURS :

Ce cours traitera essentiellement les points suivants :

- Définitions : unité de codage, unité de transfert et mots binaires
- Codage :
  - décimale vers binaire
  - décimale vers hexadécimale
  - décimale vers Binaire Codé Décimale (BCD)
  - des caractères (ASCII)
  - Unicode
  - des images
  - de la couleur
  - de la profondeur d'image

Aujourd'hui nos ordinateurs, téléphones et autres appareils savent manipuler aussi bien des nombres et du texte que des images, de la vidéo ou de la musique… Mais comment représenter, au sein d'un système numérique, cette diversité des objets du monde réel ou virtuel ?
Quelles sont les techniques utilisées pour représenter numériquement les grandeurs qui nous entourent ?

DÉFINITIONS

UNITÉ DE CODAGE

Les composants constituant un système informatique réagissent, de manière interne, à des signaux « **tout ou rien** ». On représente les deux états stables ainsi définis par les symboles « **0** » et « **1** » ou encore par « **L** » (Low) et « **H** » (High).

Le système de numération adaptée à la représentation de tels signaux est **la base 2**, on parle alors de **codage binaire.**

L'unité de codage de l'information est un élément ne pouvant prendre que les valeurs 0 ou 1 ; le **bit** (contraction de Binary Digit).

UNITÉ DE TRANSFERT

Pour les échanges de données, les informations élémentaires (bits) sont manipulées par groupes qui forment ainsi des mots binaires. La taille de ces mots est le plus souvent un multiple de $8 = 2^3$.

L'unité de transfert utilisée pour les échanges de données est le mot de 8 bits appelé **octet**.

Exemples : (2 octets)

1111 0011
1010 1111

Remarque :

Un octet est **un byte** (chiffre binaire) particulier contenant 8 bits.

Pour faciliter les manipulations, un octet peut être divisé en deux mots de 4 bits que l'on appelle des **quartets** : celui situé à gauche est le quartet de poids fort, **MSQ** (**M**ost **S**ignificant **Q**uartet), et celui situé à droite, le quartet de poids faible, **LSQ** (**L**ess **S**ignificant **Q**uartet).

Exemple :

MSQ							LSQ
1	0	0	1	1	1	1	0
quartet de poids fort				quartet de poids faible			
octet							

Remarque :

Un quartet est un byte particulier contenant 4 bits.

MOTS BINAIRES

Dans un mot binaire, le bit situé le plus à gauche est le bit le plus significatif, **MSB** (**M**ost **S**ignificant **B**it), celui situé le plus à droite est le bit le moins significatif, **LSB** (**L**ess **S**ignificant **B**it).

Exemple :

MSB															LSB
1	0	0	0	1	1	1	0	1	0	0	0	0	0	1	1
octet de poids fort								octet de poids faible							
mot (16 bits)															

Dans les notations de quantité binaires « kilo », « méga », … sont utilisés pour exprimer des multiples en puissances de 2, mais cet usage est contraire aux normes SI (Système International).

ko (kB) =	kilo-octet (kiloByte)	=	10^3 octets	=	1000 octets
Mo (MB) =	Méga-octet (MegaByte)	=	10^6 octets	=	1000 ko
Go (GB) =	Giga-octet (GigaByte)	=	10^9 octets	=	1000 Mo
To (TB) =	Téra-octet (TeraByte)	=	10^{12} octets	=	1000 Go
kio (kiB) =	kibi-octet (kibiByte)	=	2^{10} octets	=	1024 octets
Mio (MiB) =	Mébi-octet (MebiByte)	=	2^{20} octets	=	1024 ko
Gio (GiB) =	Gibi-octet (GibiByte)	=	2^{30} octets	=	1024 Mo
Tio (TiB) =	Tébi-octet (TebiByte)	=	2^{40} octets	=	1024 Go

(k, M, G, T, … = multiple du système international , b=bit, B=Byte, bi=binary)

La capacité en octets des différents constituants tels que circuits mémoires, disques durs, … est souvent importante : il devient indispensable d'utiliser **des unites multiples de l'octet.**
En dehors de l'unité de transfert (octet), des regroupements plus importants sont couramment utilisés : **le mot de 16 bits** = 2 octets (word), **le mot de 32 bits** = 4 octets (double word), et **le mot de 64 bits** = 8 octets (quad word)…

EXERCICES D'APPLICATIONS

EXERCICE N°1

La fiche technique d'un disque dur indique une capacité de 320 GB.

Question :

Exprimer cette capacité en Mio.

320 GB = 320.10^9 octets = $\frac{320.10^9}{2^{20}}$ = 305 176 Mio

EXERCICE N°2

Votre FAI vous annonce un débit descendant de 8 192 kibits/s.
Vous faites une mesure de débit réel et vous trouvez une moyenne de 3 280 kibits/s.

Question :

Quelle sera le temps théorique minimal de téléchargement d'une application de taille égale à 25 Mo ?

3 280 kibits/s = $3\,280.2^{10}$ bits/s = $\frac{3280.2^{10}}{8}$ octets/s = 419 840 octets/s

25 Mo = 25.10^6 octets ; $\frac{25.10^6}{419840}$ = 59,5 s

CODAGE

DÉCIMALE → BINAIRE

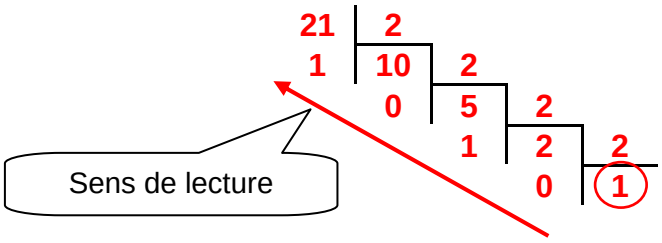
Notation :

Des indices ou un préfixe peuvent être utilisés pour les nombres binaires : $110011_{(2)}$, $1101_{(BIN)}$, %111000.

Pour coder un nombre décimal en binaire, on peut utiliser plusieurs méthodes.

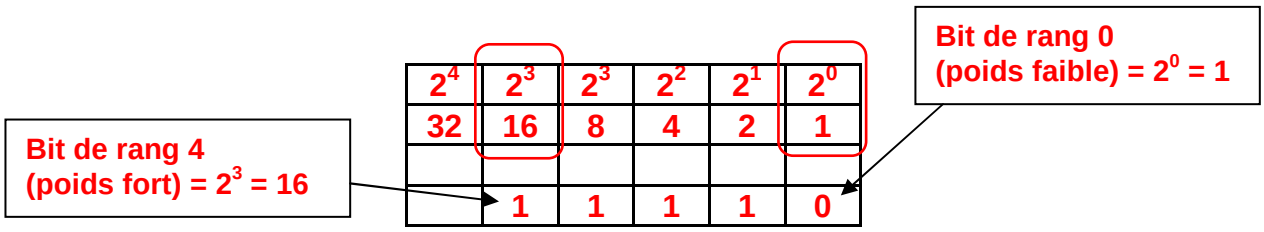
Exemple : codage des nombres $21_{(10)}$ et $30_{(10)}$ en binaire.

1ère méthode : la division successive par 2



$21_{(10)} = 10101_{(2)}$

2ème méthode : le tableau



$30 = 16x1 + 8x1 + 4x1 + 2x1 + 1x0$

$30_{(10)} = 11110_{(2)}$

DÉCIMALE → HEXADÉCIMALE

Le binaire, s'il est très représentatif du codage interne des machines, reste très délicat et fastidieux à manipuler. Les programmeurs ont très vite ressenti la nécessité d'utiliser une représentation plus rapide des nombres binaires.
Imaginons de représenter chaque quartet binaire par un unique symbole. Un quartet permettant de coder 2^4 valeurs (soit de 0 à 15). Il faut donc trouver une base de représentation disposant de 16 symboles : il s'agit de **la base 16** (appelée hexadécimale).

Notation :

Des indices ou un préfixe peuvent être utilisés pour les nombres hexadécimaux : $15_{(16)}$, $23_{(HEX)}$, 0x55F, $AF4, &h38, #44B.

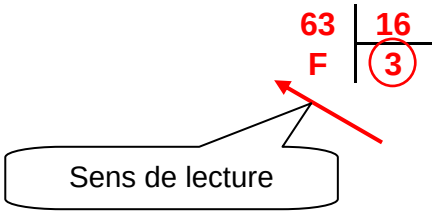
Remarques :

Le préfixe 0x est utilisé dans le langage C, C++ et JAVA ;le $ est utilisé dans le langage Pascal ; le &h dans le langage Basic et le # dans le HTML.
Une autre écriture courante est l'ajout du suffixe « h » à la fin du nombre (F15Ah par exemple).
Dans le système hexadécimale les dix premiers symboles correspondent à ceux utilisés dans le système décimal : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9, et les six derniers correspondent aux premières lettres de l'alphabet latin : A, B, C, D, E et F, lesquelles valent respectivement 10, 11, 12, 13, 14 et 15 en base 10.

Pour coder un nombre décimal en hexadécimale, on peut utiliser plusieurs méthodes.

Exemple : codage des nombres $63_{(10)}$ et $80_{(10)}$ en hexadécimale.

1ère méthode : la division successive par 16



$63_{(10)} = 3F_{(16)}$

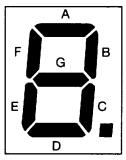
2ème méthode : le tableau

16^2	16^1	16^0
256	16	1
	5	0

$80 = 5 \times 16 + 0 \times 1$
$80_{(10)} = 50_{(16)}$

DÉCIMALE → BCD (OU DCB)

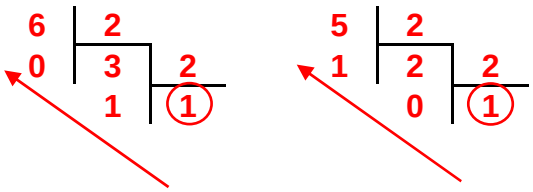
DCB signifie Décimal Codé en Binaire. Ce code est utilisé principalement pour les afficheurs 7 segments.
Il faut ici coder les chiffres décimaux individuellement afin d'obtenir pour chaques chiffres décimaux son équivalent codé en binaire sur 4 bits (quartet).



Pour coder un nombre décimal en BCD, on peut utiliser plusieurs méthodes.

Exemple : codage des nombres $65_{(10)}$ et $78_{(10)}$ en BCD.

1ère méthode : la division successive par 2



$65_{(10)} = 01100101_{(BCD)}$

2ème méthode : le tableau

2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0
8	4	2	1	8	4	2	1
0	1	1	1	1	0	0	0

7 = 0x8 + 1x4 + 1x2 + 1x1
8 = 1x8 + 0x4 + 0x2 + 0x1

$78_{(10)} = 01111000_{(BCD)}$

DES CARACTÈRES (ASCII)

Une grosse part des informations manipulées par les systèmes numériques concerne le langage parlé ou écrit matérialisé sous formes de textes, eux-mêmes constitués de caractères typographiques. Comment coder universellement ces caractères et permettre ainsi l'échange d'informations entre machines et/ou utilisateurs, quelle que soit la langue utilisée ?

Remarque :

Le morse inventé en 1844 est le premier codage à permettre une communication orientée caractère à longue distance. Ce code est composé de points et de tirets (une sorte de codage binaire).
SOS : ●●● — — — ●●●

Le jeu de caractères codés ASCII (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) ou code américain normalisé pour l'échange d'informations, inventé par Bob BERNER en 1961, est la norme de codage de caractères en informatique la plus connue, la plus ancienne et la plus largement compatible.

Le code ASCII est un code sur 7 bits (valeurs 0 à 127), il permet de définir :
- des caractères imprimables universels : lettres minuscules et majuscules, chiffres, symboles,…
- des codes de contrôle non imprimables : indicateur de saut de ligne, de fin de texte, codes de contrôle de périphériques, …

Table ASCII sur 7 bits :

Binaire					b6	0	0	0	0	1	1	1	1
					b5	0	0	1	1	0	0	1	1
					b4	0	1	0	1	0	1	0	1
				Hexadécimal		0	1	2	3	4	5	6	7
b3	b2	b1	b0		Décimal	0	16	32	48	64	80	96	112
0	0	0	0	0	+0	NUL	TC7 (DEL)	SP	0	@	P	`	p
0	0	0	1	1	+1	TC1 (SOH)	DC1	!	1	A	Q	a	q
0	0	1	0	2	+2	TC2 (STX)	DC2	"	2	B	R	b	r
0	0	1	1	3	+3	TC3 (ETX)	DC3	#	3	C	S	c	s
0	1	0	0	4	+4	TC4 (EOT)	DC4	$	4	D	T	d	t
0	1	0	1	5	+5	TC5 (ENQ)	TC8 (NAK)	%	5	E	U	e	u
0	1	1	0	6	+6	TC6 (ACK)	TC9 (SYN)	&	6	F	V	f	v
0	1	1	1	7	+7	BEL	TC 10 (ETB)	'	7	G	W	g	w
1	0	0	0	8	+8	FE0 (BS)	CAN	(	8	H	X	h	x
1	0	0	1	9	+9	FE1 (HT)	EM	)	9	I	Y	i	y
1	0	1	0	A	+10	FE2 (LF)	SUB	*	:	J	Z	j	z
1	0	1	1	B	+11	FE3 (VT)	ESC	+	;	K	[	k	é
1	1	0	0	C	+12	FE4 (FF)	IS4 (FS)	,	<	L	\	l	ù
1	1	0	1	D	+13	FE5 (CR)	IS3 (GS)	-	=	M	]	m	è
1	1	1	0	E	+14	SO	IS2 (RS)	.	>	N	^	n	-
1	1	1	1	F	+15	SI	IS1 (US)	/	?	O	_	o	DEL

EXERCICES D'APPLICATIONS

À l'aide de la table ASCII ci-dessus :

EXERCICE N°1

Question :

Décrypter la chaîne ASCII ci-dessous représentée sous la forme d'une suite d'octets :

0011 0001 0101 0011 0100 1001

1SI

EXERCICE N°2

Question :

Retrouver la chaîne ASCII sous la forme décimale du texte page suivante.

J'aime la SI.

74 39 97 105 109 101 32 108 97 32 83 73 46

Remarque :

Il existe une table ASCII dite « étendue ou élargie » sur 8 bits (http://www.ascii-code.com/).

UNICODE

Pour coder de manière universelle l'ensemble des symboles utilisés quelque soit la langue (anglais, français, grec, chinois,…) il faut attribuer à tout caractère ou symbole de n'importe quel système d'écriture de langue un nom et un identifiant numérique, et ce de manière unifiée, quelle que soit la plate-forme informatique ou le logiciel.

C'est ce que propose la norme Unicode (www.unicode.org).

Exemples :

Chaque symbole d'écriture est représenté par un nom et une valeur hexadécimale préfixée par « **U+** ».

A	« lettre majuscule latine A »	**U+0041**
é	« lettre minuscule latine e accent aigü »	**U+00E9**
€	« symbole euro »	**U+20AC**

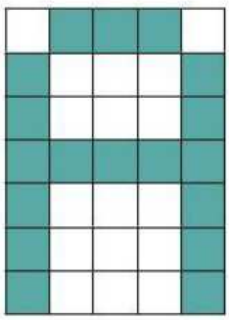
Pour stocker sur un support informatique un texte constitué de caractères Unicode, il faut encore choisir un procédé transformant chaque définition Unicode en une suite d'octets et réciproquement… C'est le **processus d'encodage.**

Actuellement un des systèmes d'encodage couramment utilisés (Unix, Internet, …) est **UTF-8** (**U**nicode **T**ransformation **F**ormat).

DES IMAGES

Après le texte, l'image est le support le plus utilisé pour communiquer.

Dans le domaine du numérique, les images sont constituées d'une **matrice L x H** (Largeur x Hauteur) de points élémentaires que l'on nomme généralement **des pixels** (abréviation de Picture Element).
Chaque pixel a une couleur codée sur un nombre plus ou moins grand de bits.
Le périphérique de sortie (écran, imprimante, …) se doit de restituer les pixels de manière ordonnée en fonction de leur position respective (x, y) et de leur couleur.



Lettre pixelisée

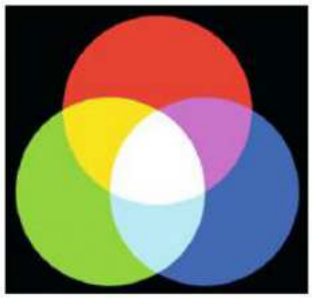
Exemple :

Le caractère ASCII $41 peut être simplement représenté sur une matrice 5 x 7 en « allumant » les pixels adéquats. La « couleur » peut-être ici matérialisée par un unique bit (pixel allumé ou éteint).

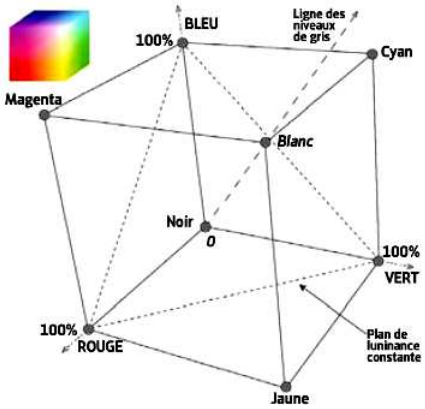
Le codage d'une image peut donc dans un premier temps se résumer en la succession des codages des pixels suivant un ordre bien défini (balayage lignes-colonnes en partant du coin supérieur gauche).

DE LA COULEUR

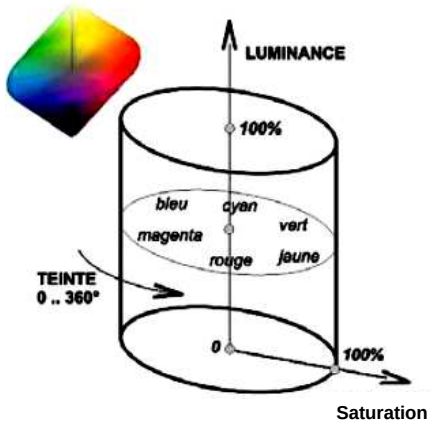
Le mode de représentation **RVB** (**R**ouge, **V**ert et **B**leu, ou en anglais RGB) correspond à celui fourni par la plupart des caméras couleur, il est naturellement utilisé pour la reproduction de couleurs sur écran (base noire). C'est le mode de composition des couleurs basé sur le principe des couleurs **additives** : le rouge, le vert et le bleu sont les trois primaires utilisés dans la constitution de couleurs à partir de sources lumineuses.



Synthèse additive



Modèle RVB



Modèle TSL

Le modèle **TSL** (**T**einte-**S**aturation-**L**uminance) est un autre modèle plus proche de la perception humaine des couleurs. Ses coordonnées se calculent à partir des proportions RVB. La composition de la teinte et de la saturation est appelée **chrominance.**

Exemple : (codage RVB)

Pixel blanc → codage RVB = $FFFFFF

http://primatice.net/logiciels/chromoweb/aide/codage.htm

DE LA PROFONDEUR D'IMAGE

Les images codées sur 24 bits sont dites en vraies couleurs (true color) ; une composante Alpha permettant d'inclure une information de transparence peut être ajoutée à ce type de codage, chaque pixel est alors codé sur 32 bits.

Le terme de profondeur d'image est utilisé pour spécifier le nombre de bits alloué au codage de chaque pixel ; les valeurs courantes de profondeur sont 1 (image binaire), 8 (256 couleurs ou niveaux de gris) et 32 (vraies couleurs avec canal alpha).

EXERCICES D'APPLICATIONS

EXERCICE N°1

Question 1 :

Quelle est la taille (en octets) d'une image non compressée, de définition 640 x 480 et de profondeur 24 bits (RVB) ?

Taille de l'image = 640 x 480 x 3 octets = 921 600 octets

L'image est incorporée à un document destiné à être distribué sous forme de photocopies noir et blanc.

Question 2 :

Quelle économie de taille réalisez-vous en convertissant l'image en 256 niveaux de gris ?

Taille de l'image en niveaux de gris = 1 octet par pixel soit 640 x 480 x 1 = 307 200 octets
Économie = 921 600 - 307 200 = 614 400 octets soit 66,6 %

EXERCICE N°2

Votre ordinateur affiche sans problème des images de définition 1024 x 768 et de profondeur 32 bits (RVBA).

Question :

Que pouvez-vous en déduire quant à la taille mémoire de votre carte vidéo ?

La taille mémoire vidéo doit être supérieure ou égale à 1024 x 768 x 4 = 3 145 728 octets.