

LE BINAIRE ET LE CODAGE DES INFORMATIONS

CORRECTION

I LE SYSTEME BINAIRE

1) Le binaire



2) Découverte du langage binaire

Initiation au langage binaire

Nombre décimal
≤ 255

56

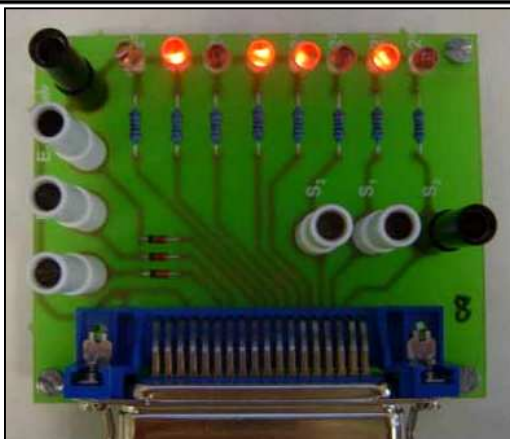
Visualisation en binaire du
nombre décimal

Module Entrée - Sortie à 8 DEL permettant de visualiser en binaire, le nombre décimal envoyé sur le port imprimante

Visualisation en binaire d'un nombre décimal

Décimal	Binaire (octet)							
	2 ⁷	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	1	0
3	0	0	0	0	0	0	1	1
4	0	0	0	0	0	1	0	0
5	0	0	0	0	0	1	0	1
6	0	0	0	0	0	1	1	0
7	0	0	0	0	0	1	1	1
8	0	0	0	0	1	0	0	0
9	0	0	0	0	1	0	0	1
10	0	0	0	0	1	0	1	0
11	0	0	0	0	1	0	1	1
20	0	0	0	1	0	1	0	0
50	0	0	1	1	0	0	1	0
90	0	1	0	1	1	0	1	0
100	0	1	1	0	0	1	0	0
150	1	0	0	1	0	1	1	0
200	1	1	0	0	1	0	0	0
255	1	1	1	1	1	1	1	1



90 en binaire

3) Conversion décimal-binaire

- fichier "Conversion élève.xls".

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	
1	Conversion Décimal-Binaire 8 bits																	
2																		
3	Nombre décimal à convertir (N < 256): N =										10							
4																		
5			7	6	5	4	3	2	1	0								
6			128	64	32	16	8	4	2	1								
7		10	10	10	10	10	2	2	0	0								
8	Nombre binaire	0 0 0 0 1 0 1 0																
9																		
10																		

- a) L'écriture binaire d'un nombre décimal pair se termine par un 0.
L'écriture binaire d'un nombre décimal impair se termine par un 1.
- b) Si $N = 2^6$, l'écriture décimale de N est $N = 64$ et l'écriture binaire de ce nombre est 1 000 000.

4) Conversion Binaire - Décimal

11	Conversion Binaire-Décimal 8 bits								
12									
13									
14	0	0	0	0	1	0	1	0	Nombre binaire: $a_7a_6a_5a_4a_3a_2a_1a_0$ Numéro n du bit Poids du bit: 2^n Terme $a_n \cdot 2^n$
15	7	6	5	4	3	2	1	0	
16	128	64	32	16	8	4	2	1	
17	0	0	0	0	8	0	2	0	
18	Nombre décimal	10	□						
19									

- a) Le nombre binaire le plus grand que l'on puisse écrire est: 1111 1111.
Il correspond au décimal $N_{max} = 255$.
- b) Ce nombre peut s'écrire sous la forme: $255 = 2^8 - 1$ avec $n = 8$.

5) Le bit d'information

a) tableau complété:

Nombre de bits	Nombre d'états	États
1	2	0 1
2	4	00 01 10 11
3	8	000 001 010 011 100 101 110 111
4	16	0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

- b) **nombre d'états** = $2^{\text{nombre de bits}}$.
Pour **n bits**, le **nombre d'états différents** est: 2^n .
- c) Dans la conversion du paragraphe précédent, les nombres binaires étaient écrits sur "**8 bits**": il y a donc $2^8 = 256$ états différents entre **0** (000 0000) et **255** (1111 1111).

6) L'octet et ses multiples

- a) Pour un **octet** il y a $2^8 = 256$ états différents.
- b) Un **kilo-octet** vaut **1 Ko** = 2^{10} octets. Le décimal correspond à 2^{10} est **1024**.
Donc: **1 Ko** \neq **1000 octets**.
- c) Un **méga-octet** vaut **1 Mo** = 2^{20} octets. Le décimal correspond à 2^{20} est **1 048 576**.
Donc: **1 Mo** \neq **1 000 000 octets**.

d) **1 Go = 2³⁰ octets.**

e) Le premier ordinateur familial (ZX Sinclair) avait **1 Ko** de RAM ... La RAM des ordinateurs aujourd'hui est d'environ **1024 Mo**.

f) L'ordre de grandeur de la capacité de stockage:

- des disquettes 3 ½ : 1,44 Ko
- d'un CD-ROM: 640 Mo
- d'une clé USB: 1 à 2 Go
- d'un disque dur: quelques centaines de Go.

II LE CODAGE DES CARACTERES EN BINAIRE

1) Sur **un seul octet** on peut coder 256 caractères: cela est largement suffisant pour coder tous les caractères du clavier (une centaine en tout).

• Un code a été créé, **le code ASCII** (American Standard Code for Informatic Information). A chaque valeur d'octet correspond un caractère ou une commande du clavier. En voici un extrait:

32		48	0	64	@	80	P	96	`	112	p	128		144	
33	!	49	1	65	A	81	Q	97	a	113	q	129		145	
34		50	2	66	B	82	R	98	b	114	r	130	é	146	
35	"	51	3	67	C	83	S	99	c	115	s	131		147	
36	#	52	4	68	D	84	T	100	d	116	t	132		148	
37	\$	53	5	69	E	85	U	101	e	117	u	133	à	149	
38	%	54	6	70	F	86	V	102	f	118	v	134		150	
39	&	55	7	71	G	87	W	103	g	119	w	135	ç	151	ù
40	'	56	8	72	H	88	X	104	h	120	x	136		152	
41	(57	9	73	I	89	Y	105	i	121	y	137		153	
42)	58	:	74	J	90	Z	106	j	122	z	138	è	154	
43	*	59	;	75	K	91	[107	k	123	{	139		155	
44	,	60	<	76	L	92	\	108	l	124		140		156	£
45	-	61	=	77	M	93]	109	m	125	}	141		157	
46	.	62	>	78	N	94	^	110	n	126	~	142			
47	/	63	?	79	O	95	_	111	o	127	Δ	143			

Remarque: Le caractère 32 est "l'espace". Les autres cases vides correspondent à des caractères non représentés.

2) Nombre binaire	Nombre décimal	Caractère
0100 0010	66	B
0101 0010	82	R
0100 0001	65	A
0101 0110	86	V
0100 1111	79	O
0010 0001	33	!

3) **Prénom ERIC** en binaire:

Caractère	Nombre décimal	Nombre binaire
E	69	0100 0101
R	82	0101 0010
I	73	0100 1001
C	67	0100 0011

4) Le fichier **1_caractère.txt** est codé sur **1 octet** et le fichier **10_caractère.txt** est codé sur 10 octets. Ainsi **un caractère est codé sur un octet**.

5) Combien de pages de **40** lignes, comportant chacune 80 caractères, devrait-on pouvoir enregistrer sur une disquette de **1,44 Mo** ?

Une ligne comporte 80 caractères chacun codé sur 1 octet soit en tout 80 octets.

Une page comporte 40 lignes avec $40 \times 80 = 3200$ caractères soit 3200 octets.

Or $1,44 \text{ Mo} = 1,44 \times 2^{20}$ octets

Le nombre N de pages que l'on peut enregistrer sur une disquette 1,44 Mo est alors:

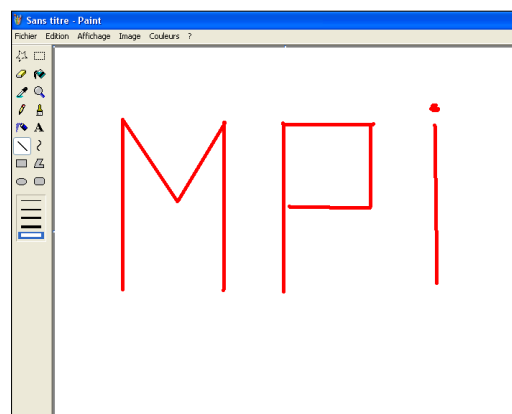
$$N = 1,44 \times 2^{20} / 3200 \approx \mathbf{472 \text{ pages}}$$

III. LE CODAGE DES IMAGES

1) Image bitmap

a) **largeur: 640 pixels**
hauteur: 512 pixels

c) Le nombre de pixels du cadre image est:
 $640 \times 512 = \mathbf{327\ 680 \text{ pixels}}$



c)

Nom du fichier	Taille (en Ko)
img_monochrome.bmp	41
Img_16couleurs.bmp	161
Img_256couleurs.bmp	322
Img_24bits.bmp	961

2) Vérification de la taille des images par le calcul

• La **place mémoire** occupée par **un pixel** dépend du nombre de couleurs que l'on souhaite :

Nombre de couleurs	Nombre de bits	Place mémoire en Ko
Noir et blanc: 2 couleurs	1	40
16 couleurs	4	160
256 couleurs	8	320
63 536 couleurs	16	640
≈ 16,7 millions de couleurs	24	960

a) Nombre **de bits** pour coder une image pour laquelle **chaque pixel** peut prendre:

2 couleurs:	1 bit	$2^1 = 2$
16 couleurs	4 bits	$2^4 = 16$
256 couleurs:	8 bits	$2^8 = 256$
63 536 couleurs:	16 bits	$2^{16} = 63\,536$
16,7 millions de couleur:	24 bits	$2^{24} = 16\,777\,216$

c) Les images enregistrées ont **327 680 pixels**.

Or: **1 octet = 8 bits**

1 Ko = 1024 octets

- image noir et blanc sur 1 bit:
 $327\,680 \times 1 = 327\,680$ bits soit $327\,680 / 8 = 40\,960$ octets soit $40\,960 / 1024 = \mathbf{40\ Ko}$
- image 16 couleurs sur 4 bits:
 $327\,680 \times 4 = 1\,310\,720$ bits soit $1\,310\,720 / 8 / 1024 = \mathbf{160\ Ko}$ ($40 \times 4 = \mathbf{160\ Ko}$)
- image 256 couleurs sur 8 bits: $8 \times 40\ Ko = \mathbf{320\ Ko}$.
- image 63 536 couleurs sur 16 bits: $16 \times 40\ Ko = \mathbf{640\ Ko}$.
- image 16,7 millions de couleurs sur 24 bits: $24 \times 40\ Ko = \mathbf{960\ Ko}$.

c) A 1 ou 2 Ko près, la **taille en Ko** de chaque image du 1) correspond à la place en mémoire en Ko calculée.