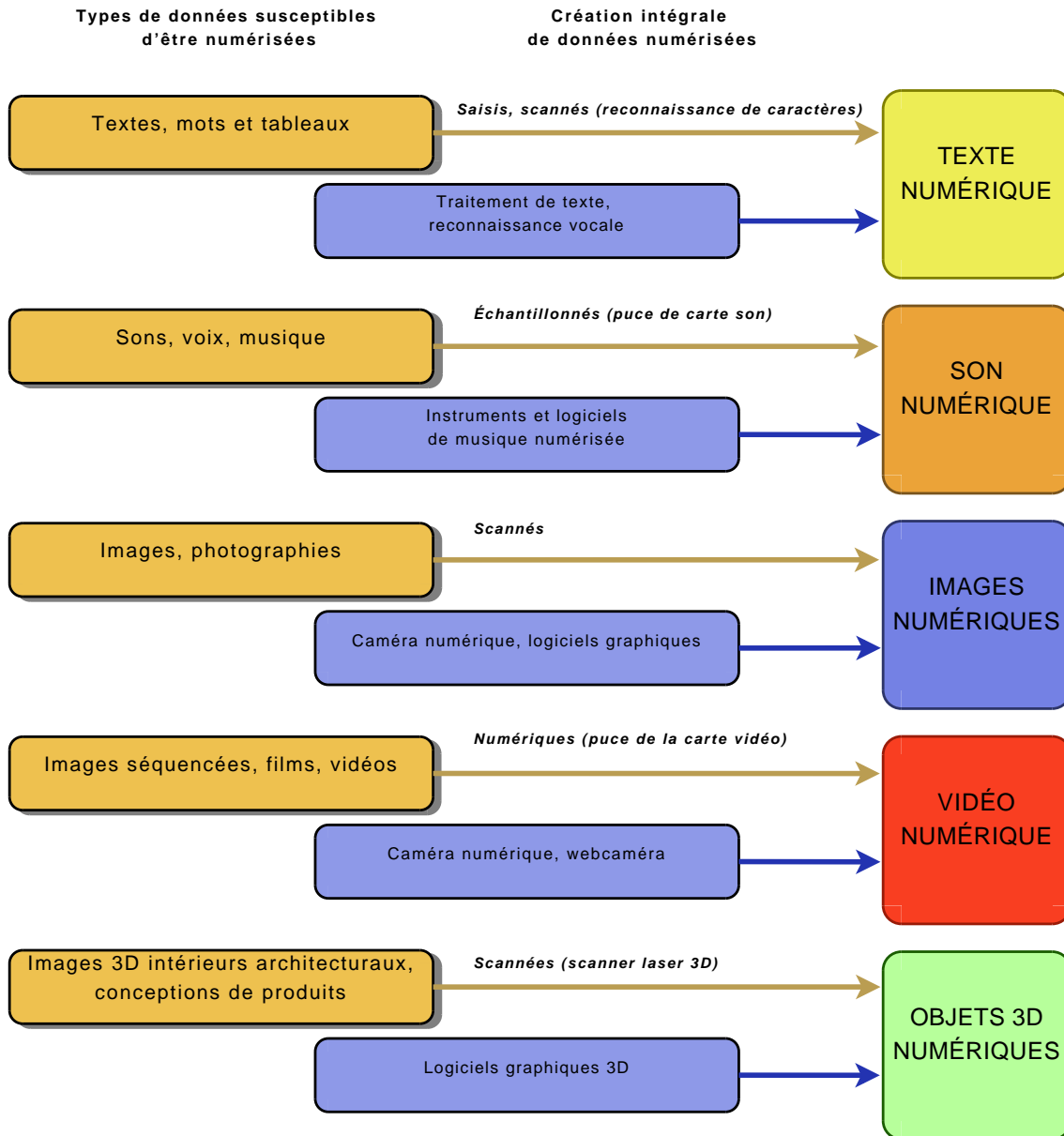


1.1 Le monde du numérique

Les appareils numériques que nous utilisons tous les jours ont tous un point commun : 2 chiffres seulement, 0 et 1, sont utilisés pour coder l'information. En effet il est possible, à partir de ces deux seuls symboles, de traiter et de stocker des nombres, des textes, des images et du son.



D'après Michael Wright et Mukul Patel. How Things Work Today. Gründ, 2001. 54 p. ISBN 2-7000-5042-8

La gamme de divers produits et médias créés à partir de données numérisées est immense et touche tous les domaines, des produits d'enseignement et de loisirs aux bases de données économiques et aux études environnementales modélisées.

Afin de faire circuler cette énorme quantité d'informations, une révolution à l'échelle mondiale est en cours qui concerne le type et les capacités de matériels utilisés dans les systèmes de communication.

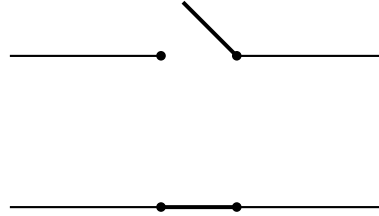
1.2 Le système binaire

a. principe

Un chiffre binaire 0 ou 1 se dit en anglais « **Binary digit** », c'est ce qu'on appelle un **Bit**.

Chaque bit dans la mémoire centrale d'un appareil numérique peut être symbolisé par un petit interrupteur capable de laisser passer le courant ou pas.

Comme un interrupteur classique, en position « ouvert off » (le courant ne passe plus) ou en position « fermé on » (le courant passe). La valeur 0 correspond à l'interrupteur ouvert et 1 à l'interrupteur fermé



Avant d'étudier comment sont traités et représentés les textes, les images, les sons etc, intéressons-nous aux nombres.

Tous les nombres entiers peuvent être représentés uniquement à l'aide de 0 et de 1.

Pour comprendre le système binaire, il faut bien comprendre le système « usuel », le système décimal et positionnel.

Le système de numérotation indo-arabe repose sur deux idées principales :

- dix symboles dont un symbole, le zéro 0, représente l'absence de quantité,
- la valeur d'un symbole dépend de sa position, pour le nombre 1231 le « 1 » à droite et le « 1 » à gauche ne représentent pas la même quantité, ce qui n'est pas le cas par exemple pour les nombres romains : pour le nombre « XXV », le symbole « X » vaut 10 quelle que soit sa position.

Ainsi le nombre $3024 = 3 \times 10^3 + 0 \times 10^2 + 2 \times 10^1 + 4 \times 10^0$

Comptons en binaire d'abord 0, puis 1, ensuite 10, et viennent 11, 100, 101, 110, 111, 1000, 1001 ... soit

Décimal	0	1	2	3	4	5	6	7	8	9
Binaire	0	1	10	11	100	101	110	111	1000	1001

Un ensemble de 8 bits s'appelle un octet.

b. Arithmétique binaire

Les règles et principes de l'arithmétique sont similaires à celle du système décimal.

Pour l'addition il n'y a que deux tables, la table du zéro et celle du un.

Tables d'addition

+	0	1
0	0	1
1	1	10

Posons quelques additions, attention aux retenues !

$$\begin{array}{r} + 11 \\ \hline 1 \end{array} \quad \begin{array}{r} + 111 \\ \hline 101 \end{array} \quad \begin{array}{r} + 10101 \\ \hline 110 \end{array} \quad \begin{array}{r} + 1011 \\ \hline 1001 \end{array}$$

Pour la multiplication, on a :

Tables de multiplication

×	0	1
0	0	0
1	0	1

Ensuite on utilise la distributivité de multiplication par rapport à l'addition.

Exemple

$$101 \times 11 = 101 \times 10 + 101 \times 1 = 1010 + 101 = 1111$$

c. Du binaire au décimal

Pour convertir un nombre binaire dans le système décimal on utilise les puissances de 2.

Ainsi le nombre binaire 110 1010 correspond à 106 dans le système décimal, en effet :

$$110\ 1010_{(2)} = 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

$$110\ 1010_{(2)} = 1 \times 64 + 1 \times 32 + 0 \times 16 + 1 \times 8 + 0 \times 4 + 1 \times 2 + 0 \times 1$$

$$110\ 1010_{(2)} = 64 + 32 + 8 + 2$$

$$110\ 1010_{(2)} = 106_{(10)}$$

► Exercice 1

En s'inspirant de l'exemple ci-dessus convertir les nombres binaires suivants dans le système décimal.

a) 1 0011

c) 100 1001

b) 10 1010

d) 11 0110

► Exercice 2

On souhaite automatiser la conversion « binaire → décimal » à l'aide d'un tableur.

- Réaliser une feuille de calculs (voir ci-dessous) dans laquelle lorsqu'on saisit un nombre binaire de maximum 8 bits (un caractère par cellule) on obtient sa valeur dans le système décimal

	A	B	C	D	E	F	G	H	I	J	K
1	position	7	6	5	4	3	2	1	0		
2	nombre binaire	0	1	1	0	1	0	1	0	valeur décimale	106

- Améliorer cette feuille en utilisant des cases à cocher. Ainsi une case cochée représentera un « 1 » une case non cochée représentera un « 0 ».

	A	B	C	D	E	F	G	H	I	J	K
1	position	7	6	5	4	3	2	1	0		
2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
3	nombre binaire	0	1	1	0	1	0	1	0	valeur décimale	106

d. Du décimal au binaire

On souhaite déterminer l'écriture binaire du nombre 172. On va tout d'abord encadrer 172 par deux puissances de 2 consécutives. On a $128 < 172 < 256$ soit $2^7 < 172 < 2^8$.

Puis on soustrait successivement, quand c'est possible, les puissances de 2 dans l'ordre décroissant.

$$172 = 128 + 44,$$

il reste 44, on ne peut pas soustraire $2^6 = 64$, par contre on peut soustraire $2^5 = 32$

$$44 = 32 + 12,$$

il reste 12, on ne peut pas soustraire $2^4 = 16$, par contre on peut soustraire $2^3 = 8$

$$12 = 8 + 4,$$

il reste 4, on peut soustraire $2^2 = 4$

$$4 = 4 + 0$$

il reste 0 d'où

$$172 = 128 + 32 + 8 + 4 = 2^7 + 2^5 + 2^3 + 2^2 = 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$$

Concrètement lorsque l'on peut soustraire une puissance de 2 cela correspond à un 1 dans l'écriture binaire et à un 0 dans le cas contraire.

On obtient donc $172_{(10)} = 10101100_{(2)}$.

► Exercice 3

En s'inspirant de l'exemple ci-dessus convertir les nombres suivants dans le système binaire.

a) 18

b) 45

c) 233

d) 666

► Exercice 4

Généralisons un peu en analysant la procédure de conversion. On se limite aux nombres n inférieurs ou égaux à 255 c'est-à-dire $n < 2^8$.

1. Ci-dessous un algorithme en langue naturelle. Décrire et commenter cet algorithme.

```
Entrées
  Saisir un entier n (compris entre 0 et 255)
Traitement
  Affecter à i la valeur 7
  Affecter à A la valeur ""
  Tant que i >= 0 :
    Si n - 2^i >= 0 alors :
      Affecter à A la valeur A + "1"
      Affecter à n la valeur n - 2^i
    Sinon
      Affecter à A la valeur A + "0"
    Fin de Si
  Affecter à i la valeur i - 1
  Fin de Tant que
Sorties
  Afficher A
```

2. On souhaite réaliser à l'aide de cet algorithme un programme **Dec2Bin** capable de convertir une écriture décimale en écriture binaire.

a) Concevoir le programme **Dec2Bin** à l'aide du logiciel **AlgoBox**.

b) Le tester d'abord en mode pas à pas, puis en mode normal avec de nombreux exemples.

3. Voici ci-dessous le même algorithme en langage dit algorithmique.

```
Saisir un entier  $n$  compris entre 0 et 255
i=7
A=""
tant que  $i \geq 0$  faire
| si  $n - 2^i \geq 0$  alors
| | A = A + "1"
| |  $n = n - 2^i$ 
| sinon
| | A = A + "0"
| fin
|  $i = i - 1$ 
fin
Afficher A
```

Analyser l'algorithme en complétant les phrases suivantes, l'algorithme est composé :

- d'une demande d'entrée
- d'une boucle
- d'un test
- d'une sortie

4. En Python, un langage de programmation la syntaxe est la suivante :

```
1: n = input("Donnez n = ")
2: i = 7
3: A = ""
4: while i >= 0 :
5:     if n - 2**i>=0 :
6:         A = A + "1"
7:         n = n - 2**i
8:     else :
9:         A = A + "0"
10:    i = i -1
11: print A
```

- Établir les correspondances entre le programme en Python et l'algorithme.
- À l'aide de l'éditeur **Geany**, saisir, sauvegarder et tester ce programme.

1.3 Coder l'alphabet

a. Le code morse

Le Morse est le premier codage qui permet une communication à longue distance. Dans ce code, les lettres sont représentées par un système de points (impulsion électrique brève) et de tirets (impulsion électrique longue), un codage binaire en quelque sorte.

Tableau des caractères morse

A	• <u> </u>	B	<u> </u> •••	C	<u> </u> • <u> </u> •	D	<u> </u> ••	E	•
F	•• <u> </u> •	G	<u> </u> <u> </u> •	H	••••	I	••	J	• <u> </u> <u> </u> <u> </u>
K	<u> </u> • <u> </u>	L	• <u> </u> ••	M	<u> </u> <u> </u>	N	<u> </u> •	O	<u> </u> <u> </u> <u> </u>
P	• <u> </u> <u> </u> •	Q	<u> </u> <u> </u> • <u> </u>	R	• <u> </u> •	S	•••	T	<u> </u>
U	•• <u> </u>	V	••• <u> </u>	W	• <u> </u> <u> </u>	X	<u> </u> •• <u> </u>	Y	<u> </u> • <u> </u> <u> </u>
Z	<u> </u> <u> </u> ••								

► **Exercice 5**

- Décoder ce message écrit en morse ••• ••• ••• • ••
- Coder le message suivant « HELLO WORLD » en morse.

b. code ASCII

La mémoire de l'ordinateur conserve toutes les données sous forme numérique. Il n'existe pas de méthode pour stocker directement les caractères. Chaque caractère possède donc son équivalent en code numérique : c'est le code ASCII (American Standard Code for Information Interchange - traduisez « Code Américain Standard pour l'Echange d'Informations »). Le code ASCII de base représente les caractères sur 7 bits de 000 0000 à 111 1111 (c'est-à-dire 128 caractères possibles, de 0 à 127).

- Les codes 0 à 31 ne sont pas des caractères. On les appelle caractères de contrôle.
- Les codes 65 à 90 représentent les majuscules
- Les codes 97 à 122 représentent les minuscules

Ainsi le « A » correspond à 65 en base 10, dans la mémoire de l'ordinateur il sera donc stocké 100 0001 en binaire. Tandis que 110 0001 en binaire, 97 en base 10 correspond au caractère « a ».

► **Exercice 6**

- À l'aide du tableur générer une table de correspondance nombre base 10 → caractère ASCII. On utilisera la fonction **CAR()** qui à un nombre compris entre 0 et 127 associe le caractère ASCII correspondant.
- Donner le code ASCII correspondant à votre prénom.

► **Exercice 7**

À partir de la feuille de calculs du tableur de l'exercice 2, on souhaite lorsqu'on saisit un nombre binaire de maximum 7 bits obtenir le caractère ASCII correspondant.

	A	B	C	D	E	F	G	H	I	J	K	L
1	position	7	6	5	4	3	2	1	0			
2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>			
3	nombre binaire	0	1	1	0	1	0	1	0	Code ASCII	106	j

1.4 Le système hexadécimal

Le système hexadécimal est un système de numération de positionnel de base 16. Il est composé de 16 symboles les 10 chiffres de 0 à 9 et des 5 lettres de A à F.

Table de correspondance

Base 10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Base 16	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Exemple : pour le nombre A5 en base 16 on a $A5 = 10 \times 16 + 5$ soit 165 en base 10.

Binaire, décimal et hexadécimal

L'écriture et la manipulation de grands nombres binaires peut-être fastidieuse, on utilise alors l'écriture hexadécimale. La conversion binaire \leftrightarrow hexadécimale est relativement simple car 16 est une puissance de 2.

Par exemple dans un octet (8 bits), il est d'usage de grouper les bits par paquet de 4 ce qui facilite la conversion en hexadécimal.

Table de correspondance

Base 2	0000	0001	0010	0011	0100	0101	0110	0111
Base 16	0	1	2	3	4	5	6	7
Base 16	1000	1001	1010	1011	1100	1101	1110	1111
Base 2	8	9	A	B	C	D	E	F

Exemple : le nombre binaire 1101 1001 s'écrit en hexadécimal D9 (car 1101 correspond à D, 1001 correspond à 9). Soit en base 10, $13 \times 16 + 9 = 217$. Le système hexadécimal est très pratique comme intermédiaire entre les systèmes décimal (homme) et binaire (machine).

Bibliographie

Michael Wright et **Mukul Patel** How Things Work Today. Gründ, 2001. ISBN 2-7000-5042-8

Codage binaire- <http://www.commentcamarche.net/contents/base/binaire.php3>