

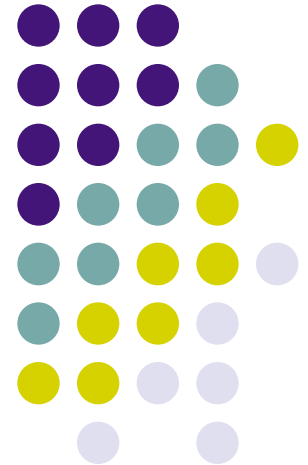
Université Mohammed V
Faculté des Sciences
Département d'Informatique

Cours M6 pour SMIA

Introduction à l'Informatique

M. El Marraki
N. El Khattabi
2020 – 2021

Cours N°9



Sommaire



- I. La Filière SMIA (SMI / SMA)
- II. Histoire de l'informatique et Structure des ordinateurs
- III. Histoires des Langages de programmation
- IV. Algèbre de Boole
- V. **Le codage**
 - Introduction
 - Système de numération décimale, binaire, octale et hexadécimale
 - Codage des nombres entiers
 - **Codage des nombres réels**
 - Codage des caractères
 - Codages des images et du son
- VI. Le langage HTML

Conversion décimale - IEEE754 (Codage d'un réel)



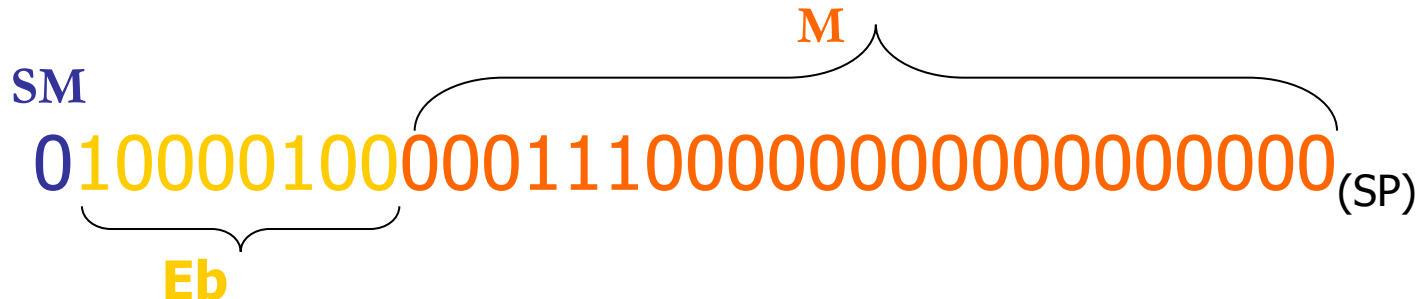
$$35,5_{(10)} = ?_{(\text{IEEE 754 simple précision})}$$

Nombre positif, donc SM = 0

$$\begin{aligned} 35,5_{(10)} &= 100011,1_{(2)} \quad (\text{virgule fixe}) \\ &= 1,000111 \cdot 2^5_{(2)} \quad (\text{virgule flottante}) \end{aligned}$$

$$E_b = E + 127 = 5 + 127, \text{ donc } E_b = 132$$

$$1,M = 1,000111 \quad \text{donc } M = 00011100\dots$$



Conversion décimale - IEEE754 (Codage d'un réel)



$$- 240.125_{(10)} = ?_{(SP)}$$

Nombre négatif, donc **SM = 1**

$$240,125_{(10)} = 11110000,001_{(2)} \text{ (virgule fixe)}$$

$$0,125 \times 2 = 0,25 \rightarrow 0$$

$$0,25 \times 2 = 0,5 \rightarrow 0$$

$$0,5 \times 2 = 1,0 \rightarrow 1$$

$$240,125_{(10)} = 1,1110000001 \cdot 2^7_{(2)} \text{ (virgule flottante)}$$

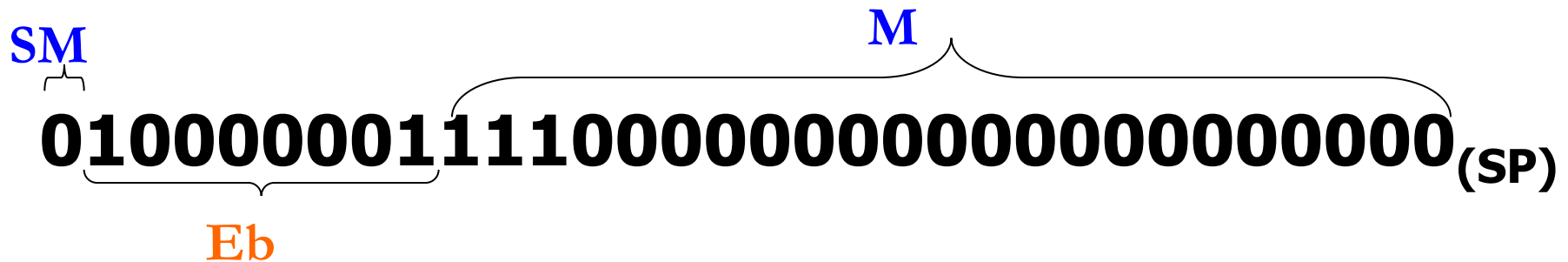
$$\text{Exposant : } E_b = 127 + 7 = 134 = 128 + 6 = 10000110_2.$$

SM

M

1 **10000110** **11100000010000000000000000**_(SP)
Eb

Conversion IEEE754 - décimale (Évaluation d'un réel)



0 10000001 11100000000000000000000

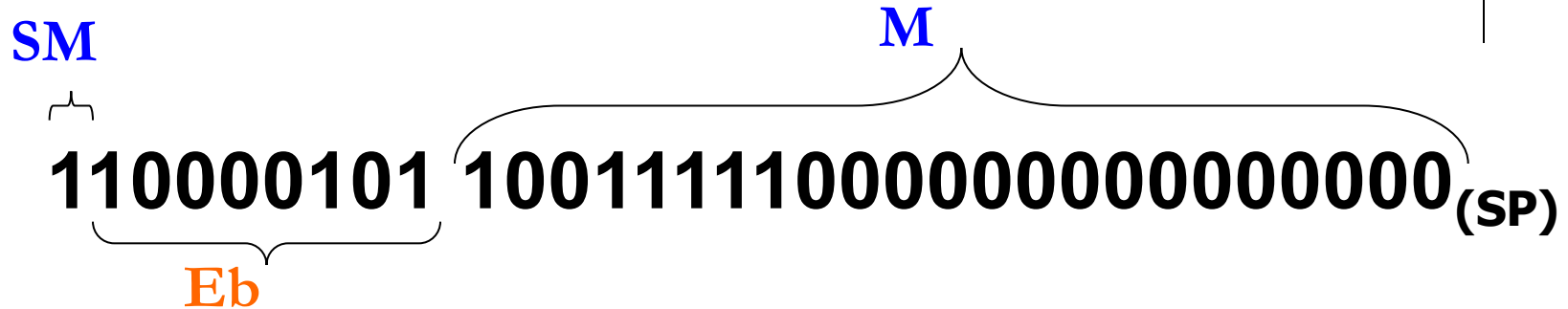
SM = 0, donc nombre positif

Eb = 129, donc exposant E = Eb - 127 = 2

1,M = 1,111

+ 1,111 . 2²₍₂₎ = 111,1₍₂₎ = 7,5₍₁₀₎

Conversion IEEE754 - décimale (Évaluation d'un réel)



1 10000101 10011111000000000000000000000000

$$E_b = 10000101_2 = 128 + 5 = 133,$$

$$\text{donc } E = 133 - 127 = 6$$

$$x = -1,10011111 \cdot 2^6 = -1100111,11$$

$$1100111_2 = 103 \text{ et } 0,11_2 = 0,75$$

$$\text{donc } x = -103,75$$

Convertir le nombre réel -193.125 dans le format IEEE 754 simple précision.



Le nombre est négatif donc SM=1

$$193 = 96 \times 2 + 1 = 3 \times 64 + 1 = 11_2 \times 1000000_2 + 1_2 \\ = 11000001_2$$

$$0,125 = 0,001_2 \quad \left(\begin{array}{l} 0,125 \times 2 = 0,25 \rightarrow 0 \\ 0,25 \times 2 = 0,5 \rightarrow 0 \\ 0,5 \times 2 = 1,0 \rightarrow 1 \end{array} \right)$$

$$193,125 = 11000001,001_2 = 1,1000001001_2 \times 2^7$$

$$E_b = 7 + 127 = 134 = 10000110_2$$

$$-193,125 = 1 \ 10000110 \ 100000100100...0$$

Conversion IEEE754 - décimale (Évaluation d'un réel)



$$x = \overset{\text{SM}}{\underbrace{101111101}_{\text{Eb}}} \overset{\text{M}}{\underbrace{10101000000000000000000000000000}_{\text{(SP)}}}$$

$$1 \quad 01111101 \quad 10101000000000000000000000000000$$

$$E_b = 01111101_2 = 125,$$

$$\text{donc } E = E_b - 127 = 125 - 127 = -2$$

$$x = -1, 10101 \cdot 2^{-2}$$

$$= -0,0110101_2 = -(2^{-2} + 2^{-3} + 2^{-5} + 2^{-7})$$

$$= -0,4140625$$

Sommaire



- I. La Filière SMIA (SMI / SMA)
- II. Histoire de l'informatique et Structure des ordinateurs
- III. Histoires des Langages de programmation
- IV. Algèbre de Boole
- V. **Le codage**
 - Introduction
 - Système de numération décimale, binaire, octale et hexadécimale
 - Codage des nombres entiers
 - Codage des nombres réels
 - **Codage des caractères**
 - Codages des images et du son
- VI. Le langage HTML



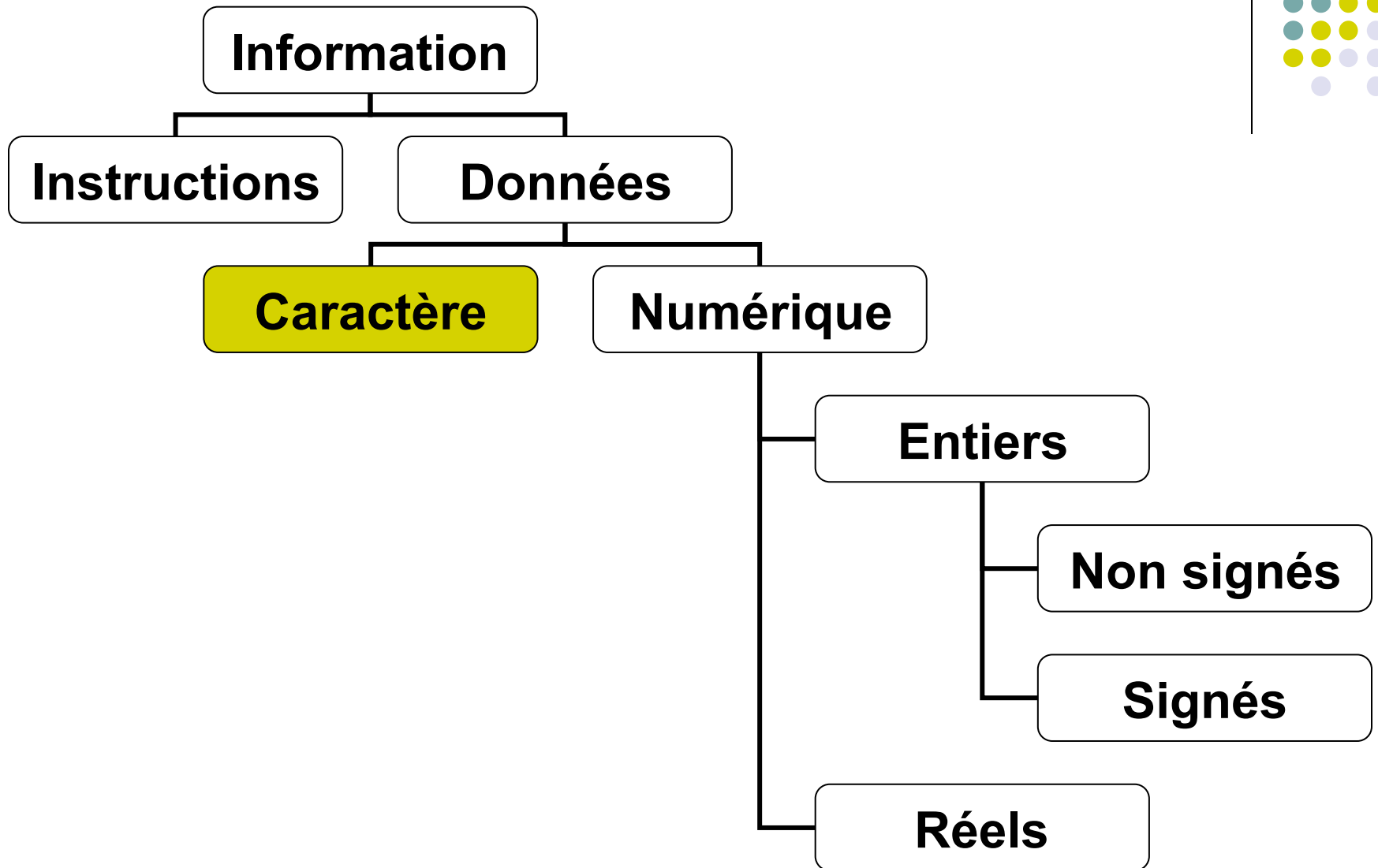
V. Le codage

Introduction

Système d'énumération

Codage des nombres réels

Codage des caractères





Codage des caractères

Caractères : **Alphabétique** (A-Z , a-z), **numérique** (0 ,..., 9), **ponctuation, spéciaux** (&, \$, %, ...) ...
etc.

Données **non numérique** (addition n' a pas de sens)

Comparaison ou **tri** → **très utile**

Codage revient à créer une **Table de correspondance** entre les **caractères** et **des nombres**.

Codage des caractères

Les standards



Code (ou Table) **ASCII** (American Standard Code for Information Interchange)

- **7 bits** pour représenter **128** caractères (0 à 127)
- **48** à **57** : **chiffres** dans l'ordre (**0**,1,...,**9**)
- **65** à **90** : les **alphabets majuscules** (**A** ,..., **Z**)
- **97** à **122** : les **alphabets minuscule** (**a** ,..., **z**)

Codage des caractères

Les standards



Table **ASCII Etendu**

- **8 bits** pour représenter **256** caractères (0 à 255)
- Code les caractères **accentués** : à, è,...etc.
- **Compatible** avec **ASCII**

Code **Unicode** (mis au point en 1991)

- **16 bits** pour représenter **65 536** caractères (0 à 65 535)
- **Compatible** avec **ASCII**
- **Code** la plupart des **alphabets** : **Arabe**, **Chinois**,
- On en a défini environ **50 000** caractères pour l' instant






















Code ASCII Etendu



DECIMAL VALUE	➡	0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
↩	HEXA DECIMAL VALUE	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	BLANK (NULL)	▶	SP	0	@	P	‘	p	Ç	É	á	⋮	⌌	⌌	∞	≡
1	1	😊	◀	!	1	A	Q	a	q	ü	æ	í	⋮	⌌	⌌	β	±
2	2	😬	↑	"	2	B	R	b	r	é	Æ	ó	⋮	⌌	⌌	Γ	≥
3	3	♥	!!	#	3	C	S	c	s	â	ô	ú	⋮	⌌	⌌	π	≤
4	4	♦	¶	\$	4	D	T	d	t	ä	ö	ñ	⋮	⌌	⌌	Σ	∫
5	5	♣	§	%	5	E	U	e	u	à	ò	Ñ	⋮	⌌	⌌	σ	∫
6	6	♠	—	&	6	F	V	f	v	å	û	ä	⋮	⌌	⌌	μ	÷
7	7	BEL	↓	'	7	G	W	g	w	ç	ù	ó	⋮	⌌	⌌	τ	≈
8	8	BS	↑	(8	H	X	h	x	ê	ÿ	ı	⋮	⌌	⌌	ϕ	°
9	9	HT	↓)	9	I	Y	i	y	ë	Ö	ƒ	⋮	⌌	⌌	θ	•
10	A	LF	→	*	:	J	Z	j	z	è	Ü	ƒ	⋮	⌌	⌌	Ω	•
11	B	VT	←	+	;	K	I	k	{	ï	ç	½	⋮	⌌	⌌	δ	√
12	C	FF	FS	,	<	L	\	l	!	î	£	¼	⋮	⌌	⌌	∞	n
13	D	CR	GS	—	=	M	J	m	}	ì	¥	ı	⋮	⌌	⌌	φ	²
14	E	♪	RS	.	>	N	^	n	~	Ä	Ŕ	«	⋮	⌌	⌌	€	■
15	F	⚙	US	/	?	O	—	o	Δ	Å	ƒ	»	⋮	⌌	⌌	∩	NAME

Unicode

پ	ڈ	ع	گ	ق	ی	اُ	و
0680	0690	06A0	06B0	06C0	06D0	06E0	06F0
خ	ز	ف	ح	ط	ي	اُو	اِ
0681	0691	06A1	06B1	06C1	06D1	06E1	06F1
ط	ز	ب	ح	اِ	اِ	اُو	اِ
0682	0692	06A2	06B2	06C2	06D2	06E2	06F2

						
2600	2610	2620	2630	2640	2650	2660
						
2601	2611	2621	2631	2641	2651	2661
						
2602	2612	2622	2632	2642	2652	2662

€	Д	Φ	д	φ	€	€	✓
0404	0414	0424	0434	0444	0454	0464	0474
S	E	X	e	x	s	€	✓
0405	0415	0425	0435	0445	0455	0465	0475
I	Ж	Ц	ж	ц	i	А	✓
0406	0416	0426	0436	0446	0456	0466	0476

Ce ne sont que des bits !!!



01001001 01001110 01000110 01001111
01010010 01001101 01000001 01010100
01001001 01010001 01010101 01000101

Ce ne sont que des bits !!!



Caractères codés en ASCII Etendu (8 bits)

01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

INFORMATIQUE

Ce ne sont que des bits !!!



Entiers codés en binaire pur sur 1 octets

01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

73 ; 78 ; 70 ; 79 ; 82 ; 77 ;
65 ; 84 ; 73 ; 81 ; 85 ; 69 (base 10)

Ce ne sont que des bits !!!



Entiers codés en binaire pur sur 2 octets

01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

18766 ; 17999 ; 21069 ;
16724 ; 18769 ; 21829 (base 10)

Ce ne sont que des bits !!!



Entiers codés en binaire pur sur 4 octets

01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

1 229 866 575 ;

1 380 794 708 ;

1 230 067 013 (base 10)

Ce ne sont que des bits !!!



Nombres en flottant simple précision (32 bits)

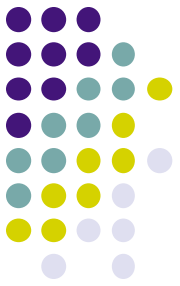
01001001 01001110 01000110 01001111 01010010 01001101
01000001 01010100 01001001 01010001 01010101 01000101

$$\begin{aligned} &+ (1,10011100100011001001111) \cdot 2^{19} ; \\ &+ (1,10011010100000101010100) \cdot 2^{37} ; \\ &+ (1,10100010101010101000101) \cdot 2^{19} ; \end{aligned}$$

844 900,9375;

220 391 079 936 ;

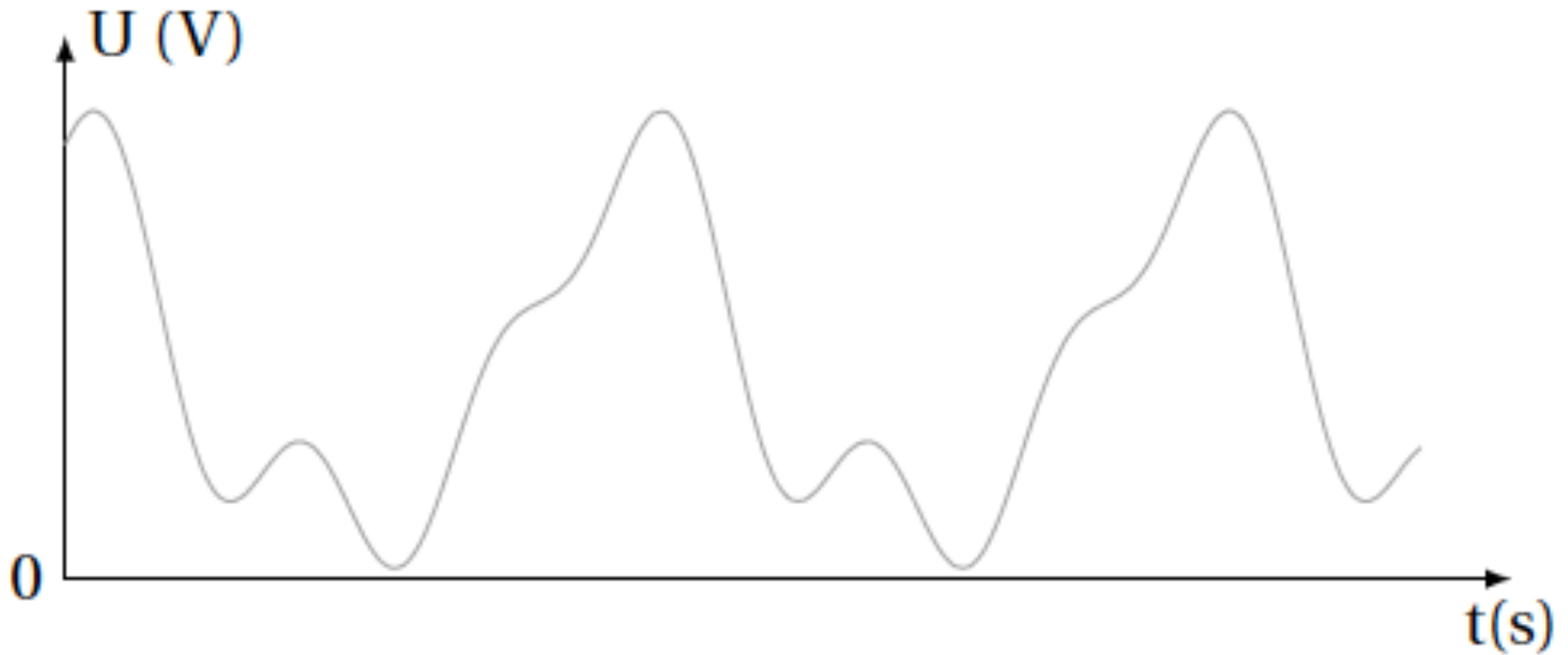
857 428,3125 (base 10)



Codage de la musique



Codage d'un signal

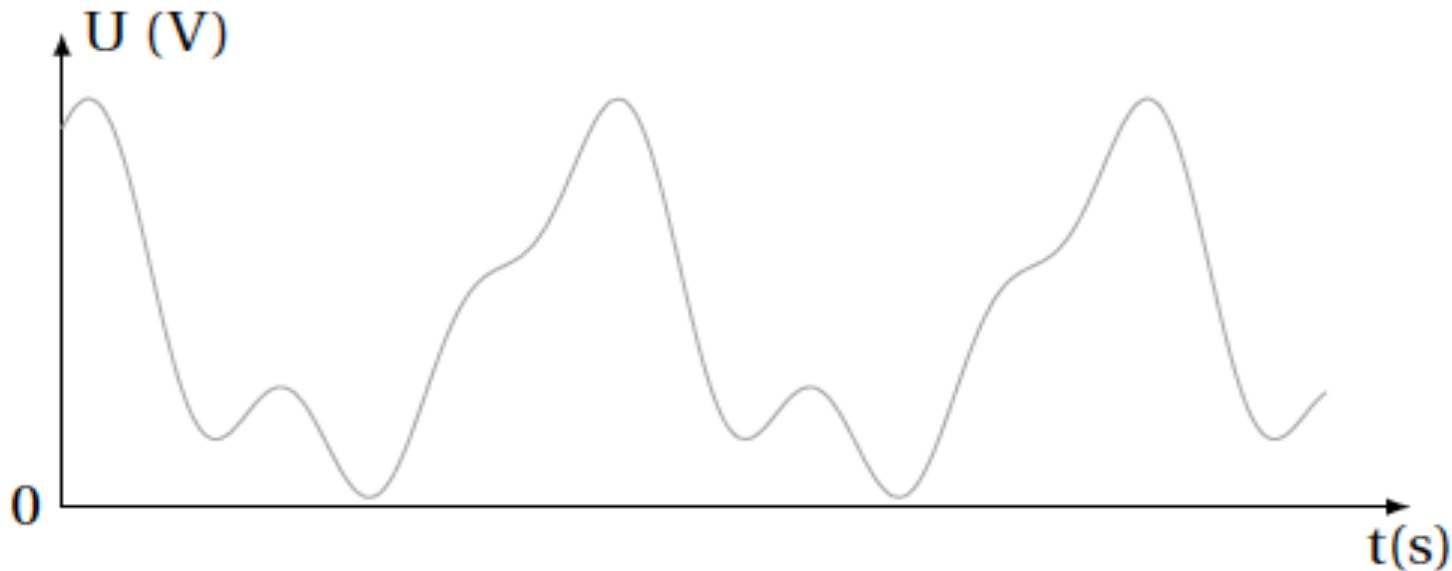


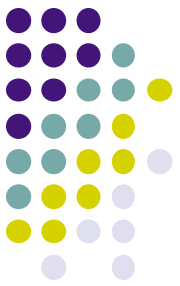


Signal analogique

Définition

Un signal **analogique** est un signal **continu** au cours du temps.

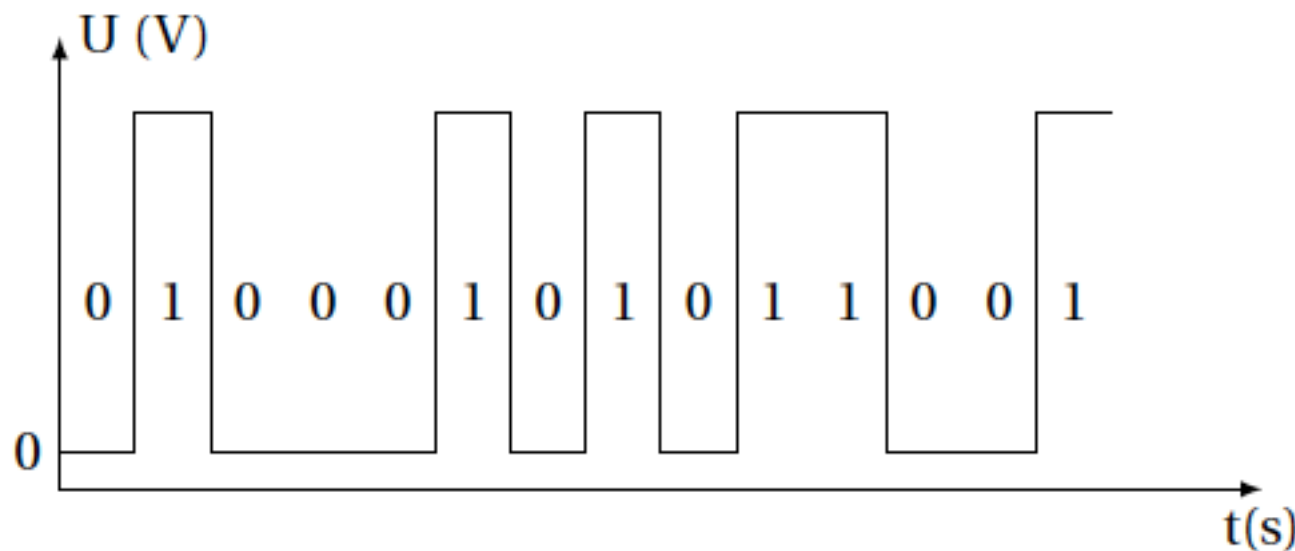




Signal numérique

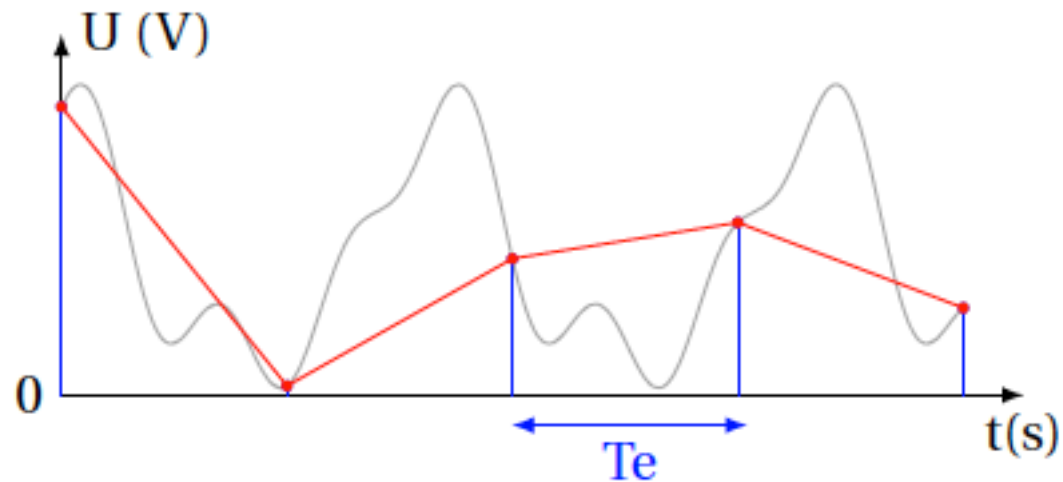
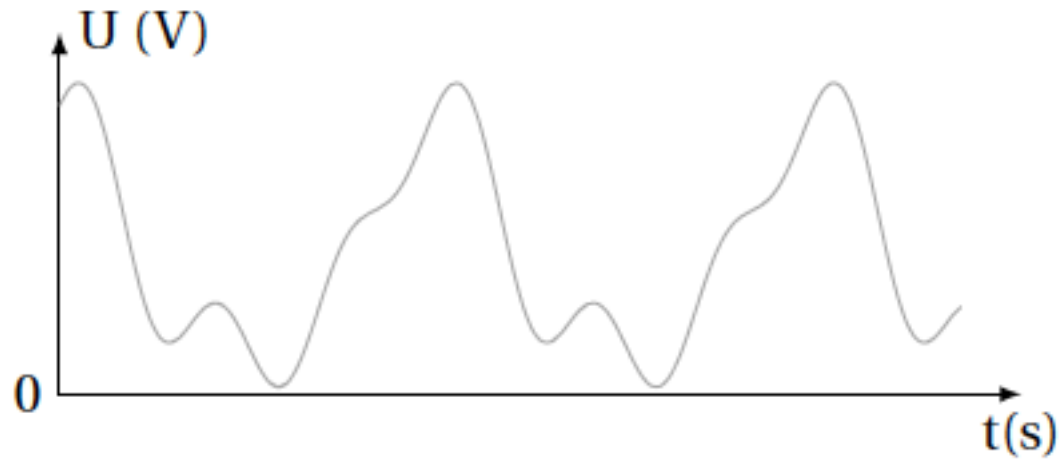
Définition

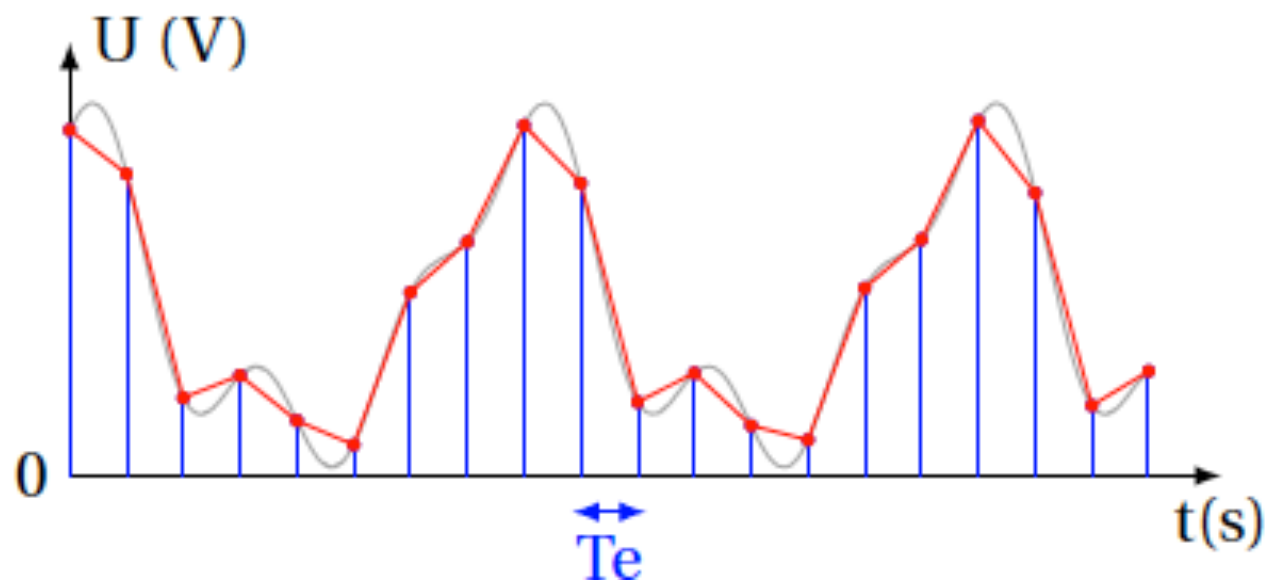
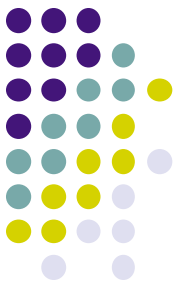
Un signal **numérique** est une suite de 0 et de 1 logiques.



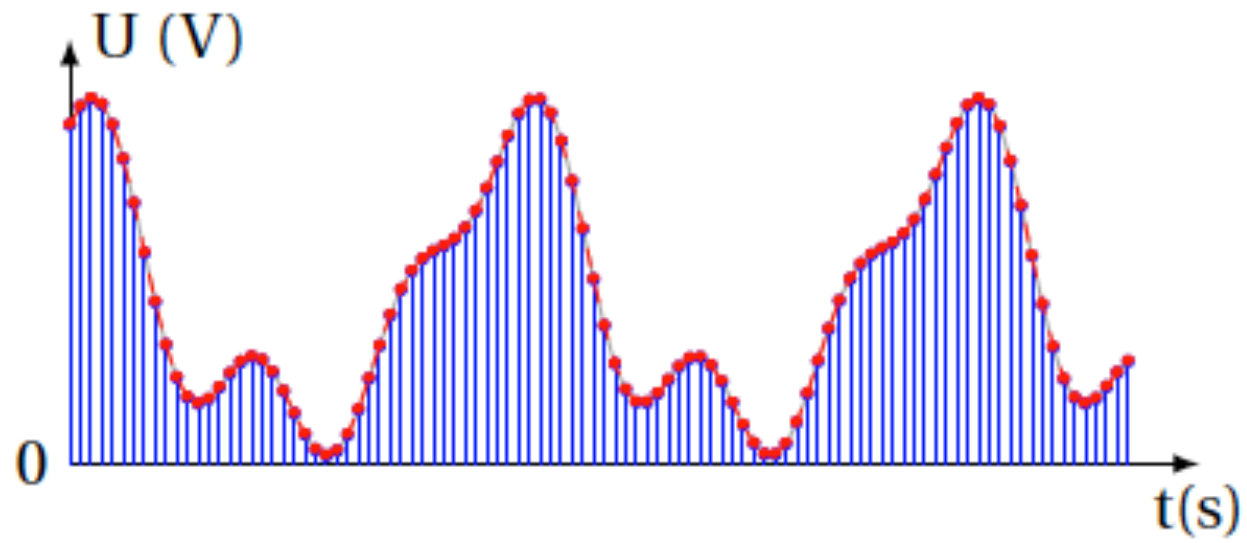


Signal analogique





$$F_e = \frac{1}{T_e}$$





Théorème de Shannon

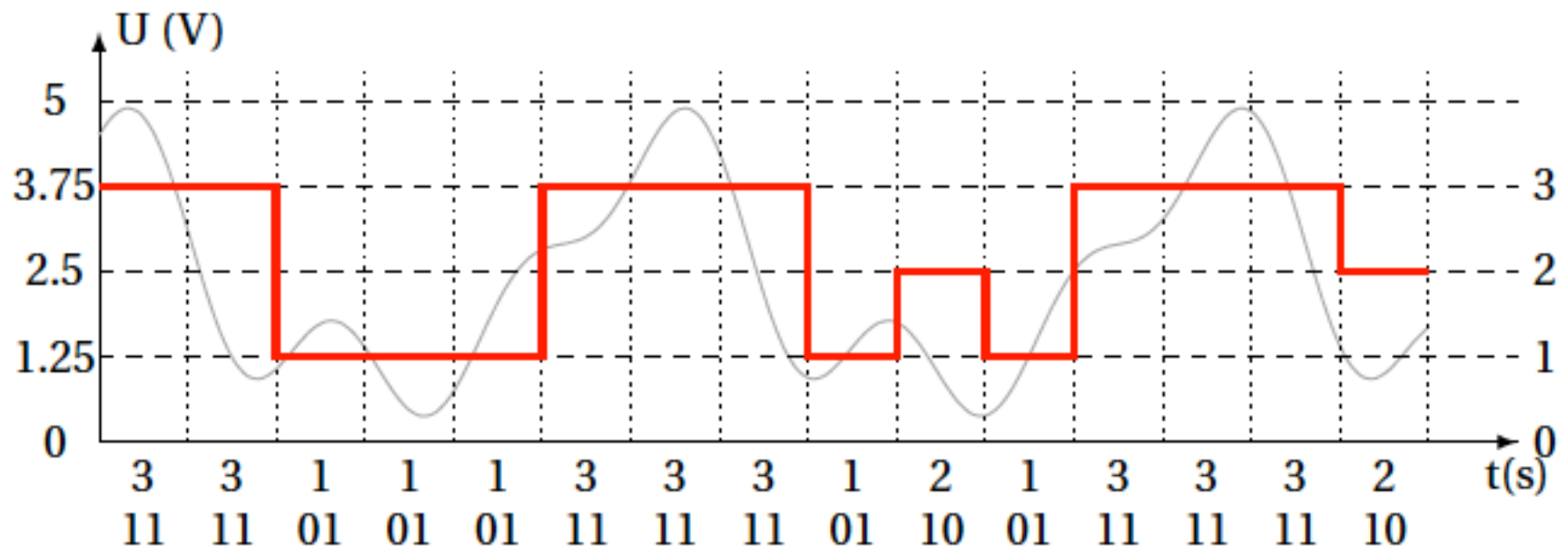
Pour que le signal puisse être entièrement reconstruit à partir des échantillons, il faut et il suffit que : La fréquence d'échantillonnage doit être strictement supérieure à deux fois la plus grande fréquence présente dans le spectre du signal continu (condition de Nyquist-**Shannon**).

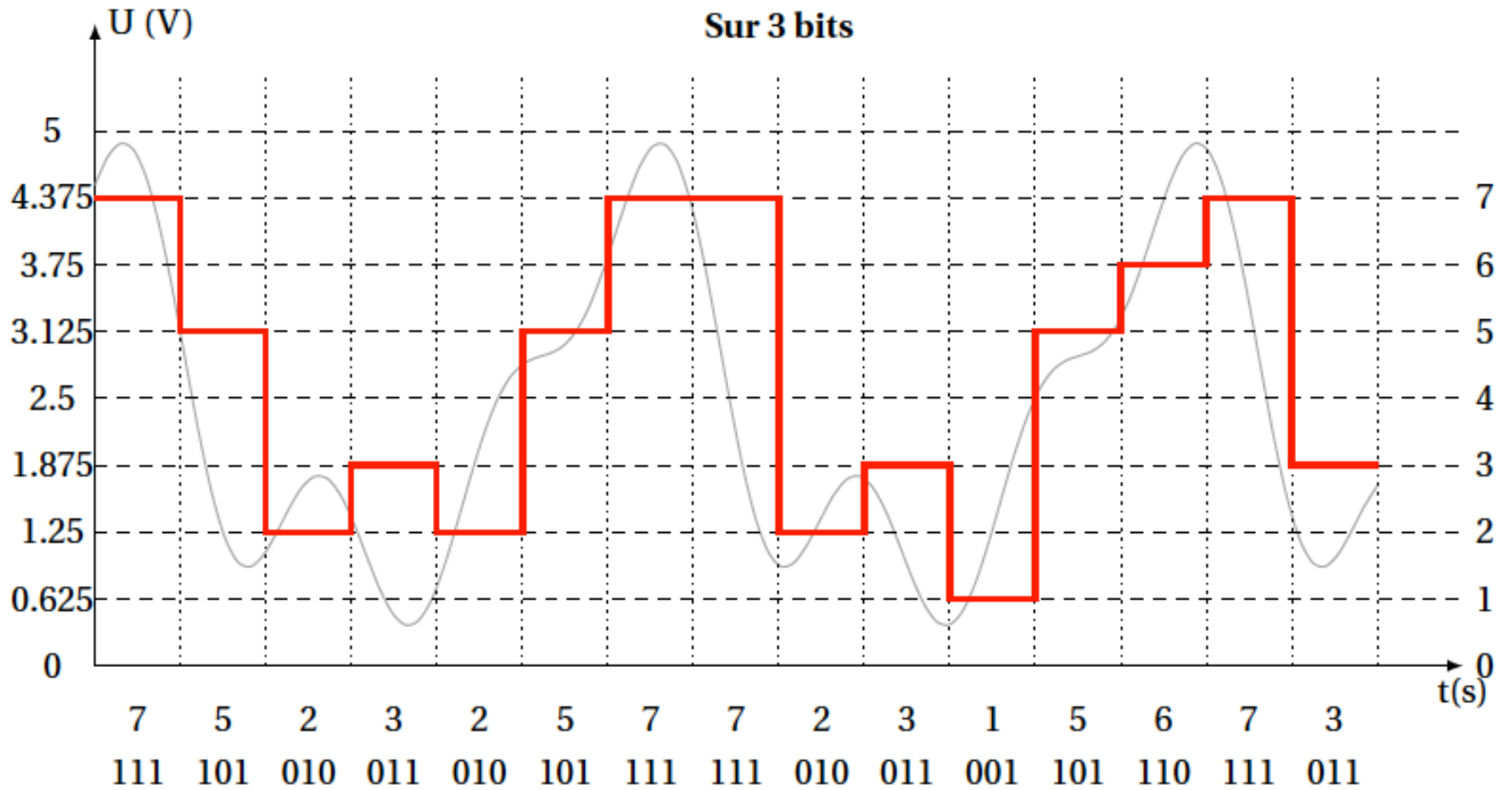
$$F_e \geq 2F_{Max}$$

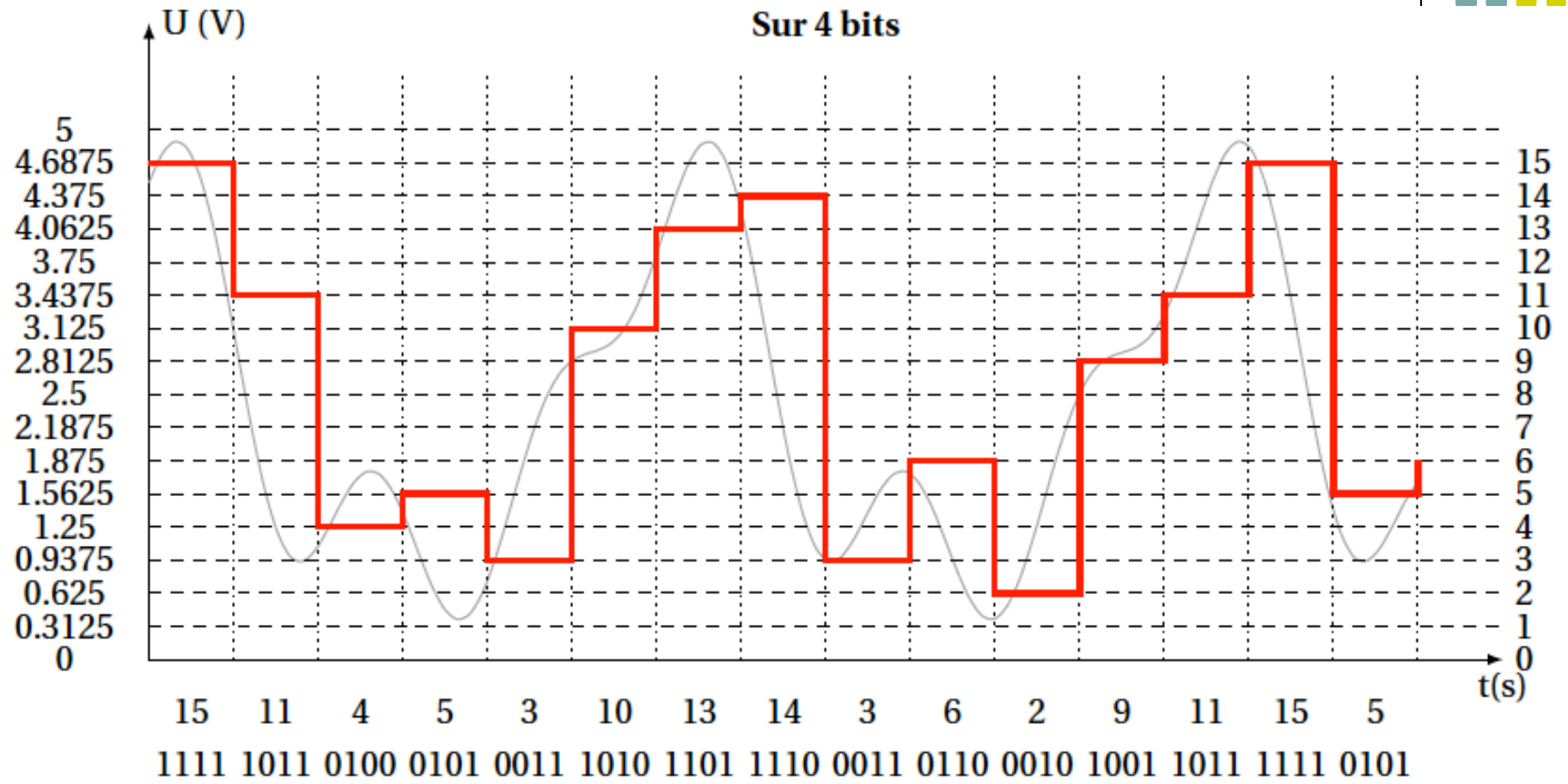
Codé sur 2 bits :

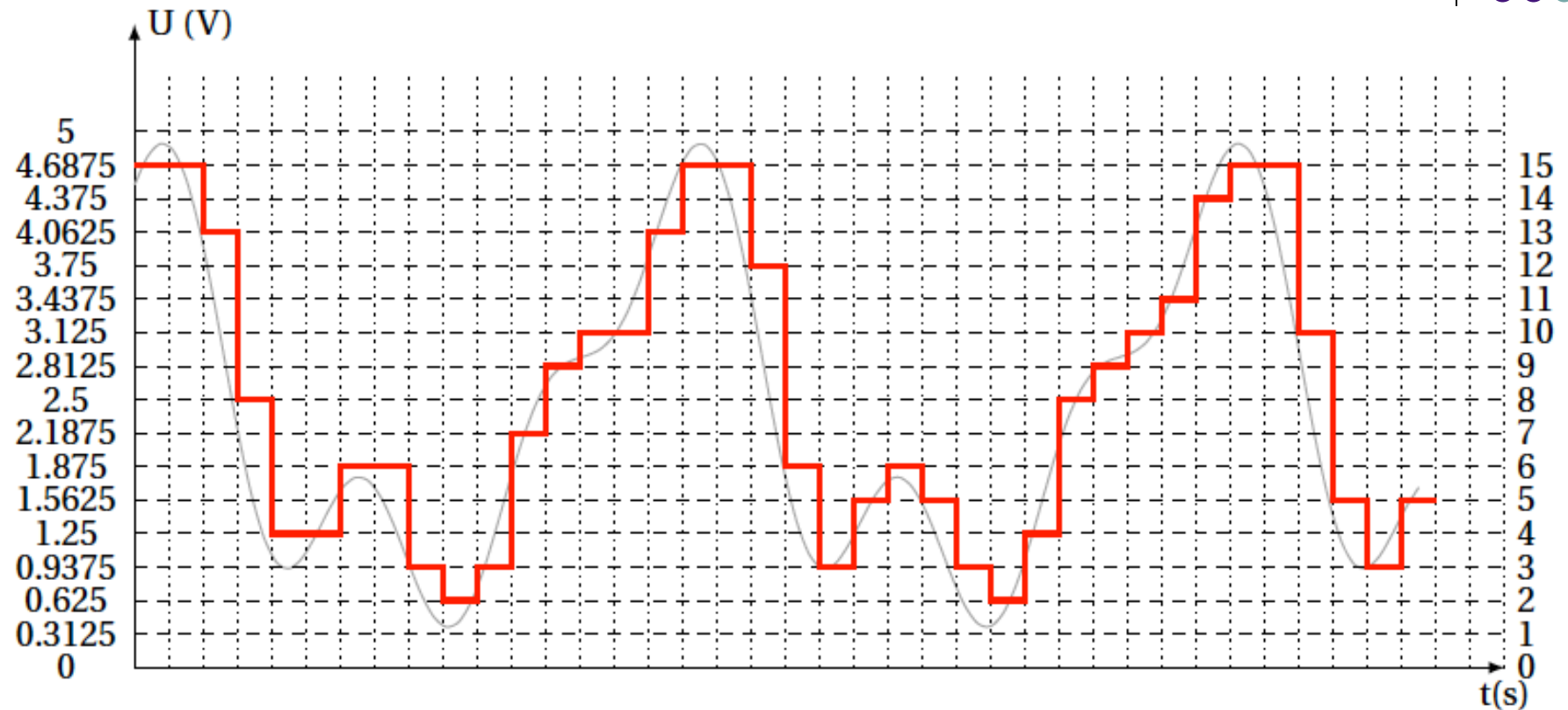
On obtient les correspondances :

tension (V)	Décimal	Binaire
0	0	00
1,25	1	01
2,5	2	10
3,75	3	11







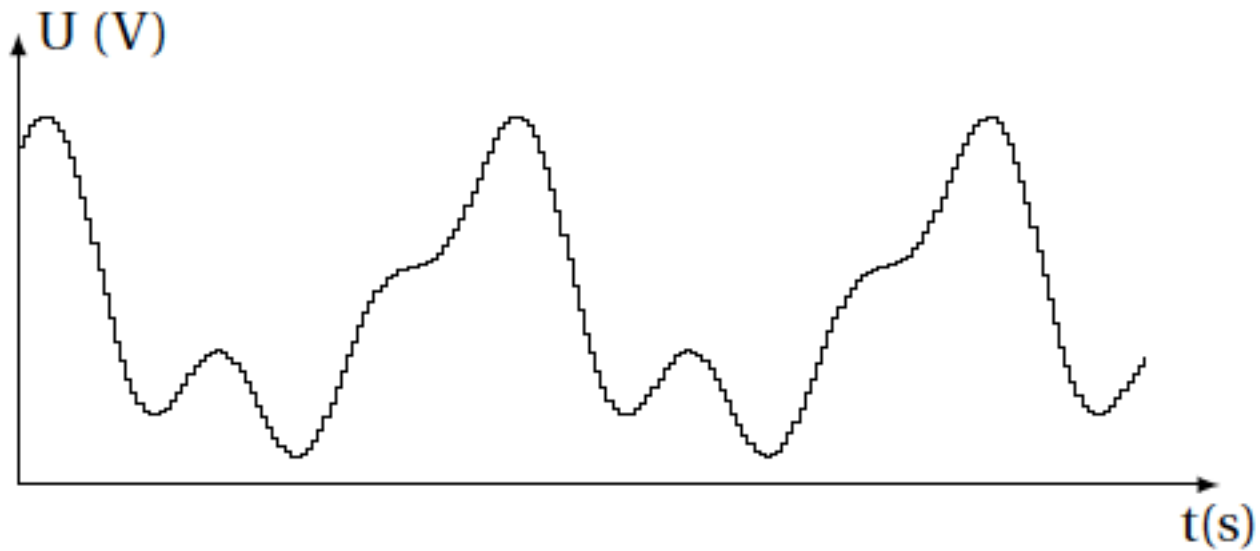
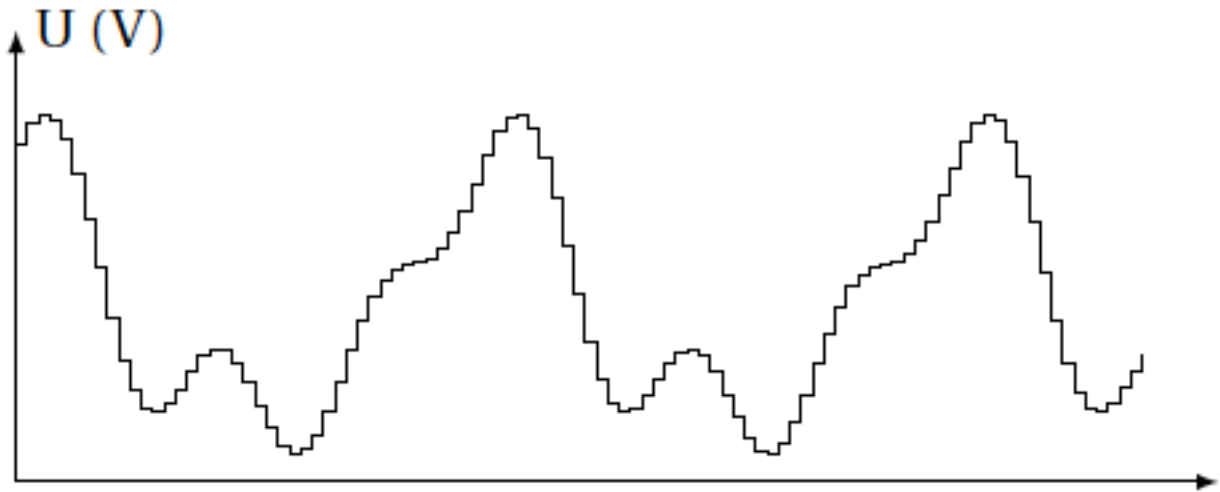


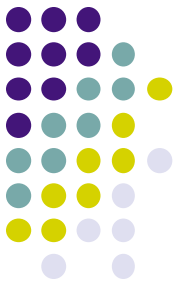
Je vous laisse vérifier que le codage est bien :

```
111111111101100001000100011001100011001000110111100110101010110111111111111000
110001101010110010100110010010010001001101010111110111111111010010100110101
```



On obtient





Fin du cours