



<b>STI2D</b> Option SIN / Terminale	<h1>Révisions</h1>	
	<b>Partie 3 : binaire, numération, CNA, CAN, logique, ...</b>	

<b>Pré-requis :</b>
Les bases de la numération (binaire, décimal et hexadécimal)
Les propriétés des fonctions logiques
Le principe des convertisseurs analogiques-numériques et numériques-analogiques

<b>Compétences visées :</b>
<i>Être capable de convertir des signaux de natures différentes (analogique, numérique, binaire, décimal, ...)</i>
<i>Être capable d'analyser un montage à base de circuits logiques</i>

<b>Plan de la séance :</b>
Exercice 1 : révision du binaire et du décimal à partir des adresses IPv4
Exercice 2 : révision du binaire, du décimal et de l'hexadécimal à partir des adresses IPv6
Exercice 3 : analyse d'un convertisseur analogique/numérique
Exercice 4 : circuits logiques
En bonus :
Exercice 5 : nombre au format float
Exercice 6 : circuit logique inconnu



**Toutes les informations utiles sont présentes dans les cours et TP de 1ère STI SIN.**

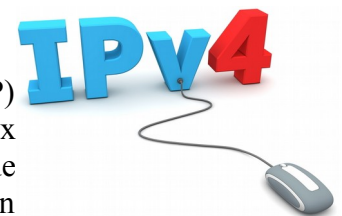


**Vous devez détailler la méthode de résolution des exercices (le résultat seul importe peu car il peut souvent être obtenu facilement par des programmes informatiques présents dans les calculatrices ou ordinateurs). L'intérêt n'est pas de donner le résultat mais de montrer comment vous arrivez au résultat (et donc de comprendre les structures, règles et démarches).**

### Exercice 1 : révision du binaire et du décimal à partir des adresses IPv4

Présentation de IPv4 :

IPv4 (Internet Protocol version 4) est la première version d'Internet Protocol (IP) à avoir été largement utilisée aussi bien pour internet que pour les réseaux informatiques en général. Elle permet une définition commune (mondialement) de la manière d'écrire les adresses des machines (ordinateur, serveur, ...) reliées à un réseau informatique.



La plage des adresses va de 0.0.0.0 à 255.255.255.255 (l'adresse est en 4 parties séparées par des points, chaque nombre est en général donné en décimal)

#### **Exemple 1 : le réseau d'un particulier**

Le réseau des particuliers est en général le suivant :  
192.168.1.x où x est l'adresse des éléments connectés au réseau

1. 'x' pouvant prendre comme valeur 0 à 255 (en décimal), sur combien de bits est-il codé en binaire ?
2. L'adresse de mon ordinateur étant 192.168.1.2, écrivez cette adresse en binaire.
3. Combien de matériels différents puis-je relier sur ce réseau ?

### Exemple 2 : le réseau d'une petite entreprise

Le réseau de l'entreprise est : 192.168.x.y où x et y sont les octets codant l'adresse des éléments connectés au réseau

4. 'x' et 'y' pouvant chacun aller de 0 à 255 (en décimal), sur combien de bits au total l'adresse des éléments est-elle codée ?
5. Combien de matériels différents puis-je relier sur ce réseau ?

### Généralisation

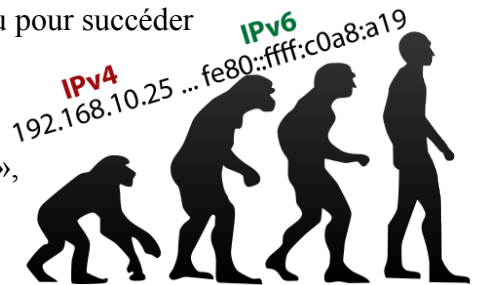
6. Si tous les éléments de l'adresse peuvent être choisis (w.x.y.z), chacun pouvant aller de 0 à 255 (en décimal), sur combien de bits au total l'adresse est-elle codée ?
7. Combien de matériels différents puis-je relier sur le réseau internet avec cette norme IPv4 ?

## Exercice 2 : révision du binaire, du décimal et de l'hexadécimal à partir des adresses IPv6

IPv6 (Internet Protocol version 6) est le protocole réseau qui a été conçu pour succéder à l'IPv4.

La plage des adresses va de 0:0:0:0:0:0:0:0 à FFFF:FFFF:FFFF : ..... :FFFF (l'adresse est en 8 parties séparées par des « deux points », chaque nombre est donné en hexadécimal )

Exemple d'une adresse IPv6 :  
2001:0db8:0000:85a3:0010:0a0b:8001:ec1f



### Préliminaires :

1. Convertissez le chiffre hexadécimal 'E' en décimal puis en binaire
2. Combien faut-il de bits pour coder un 'chiffre' hexadécimal ?

### Étude d'un des 8 éléments composant une adresse IPv6 :

Cet élément peut avoir comme valeur 0000 à ffff. Prenons l'élément 'ec1f'

3. Convertissez ce nombre hexadécimal en décimal
4. Convertissez ce nombre hexadécimal en binaire
5. Sur combien de bits est-il codé ?

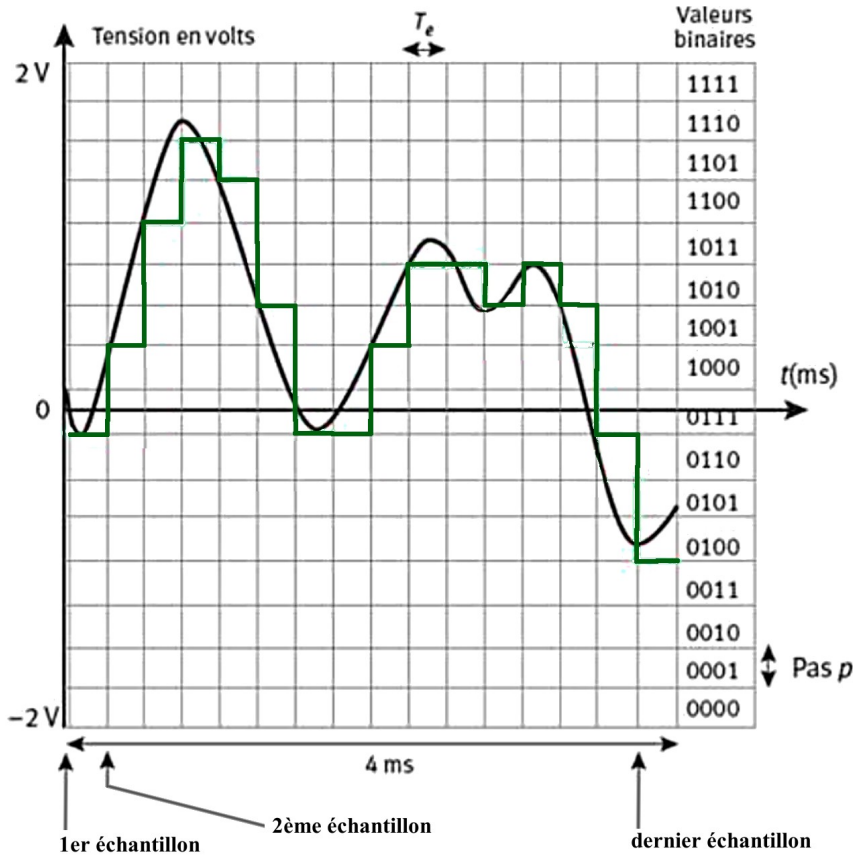
### Généralisation :

6. Sur combien de bits une adresse IPv6 complète est-elle codée ?
7. Combien de matériels différents puis-je relier sur le réseau internet avec IPv6 ?

### Exercice 3 : analyse d'un convertisseur analogique/numérique

#### Partie 1 :

Soit le courbe analogique suivante (en noir) et le résultat de sa conversion numérique (en vert) :



1. Sur combien de bits travaille ce convertisseur A/N (amplitude) ?
2. Combien de valeurs différentes sont-elles possibles (amplitude) ?
3. Déterminez la période d'échantillonnage  $T_e$  utilisée par ce convertisseur.
4. Calculez la fréquence d'échantillonnage.
5. Sachant que les valeurs max et min du signal analogique sont 2V et -2V, calculez le pas de ce convertisseur
6. Pour mémoriser le signal numérique correspondant à ces 4ms, combien de bits de données va-t-on avoir?
7. Si le signal total dure maintenant 5 minutes, combien de bits de données seront nécessaires pour mémoriser numériquement le signal?
8. Sachant qu'un kilo octet vaut 1024 octet (et oui 1ko n'est pas 1000 octets. Ceci est malheureusement une mauvaise habitude très répandue du milieu informatique et contraire au système international), donnez le résultat précédent en ko (kilo-octet).

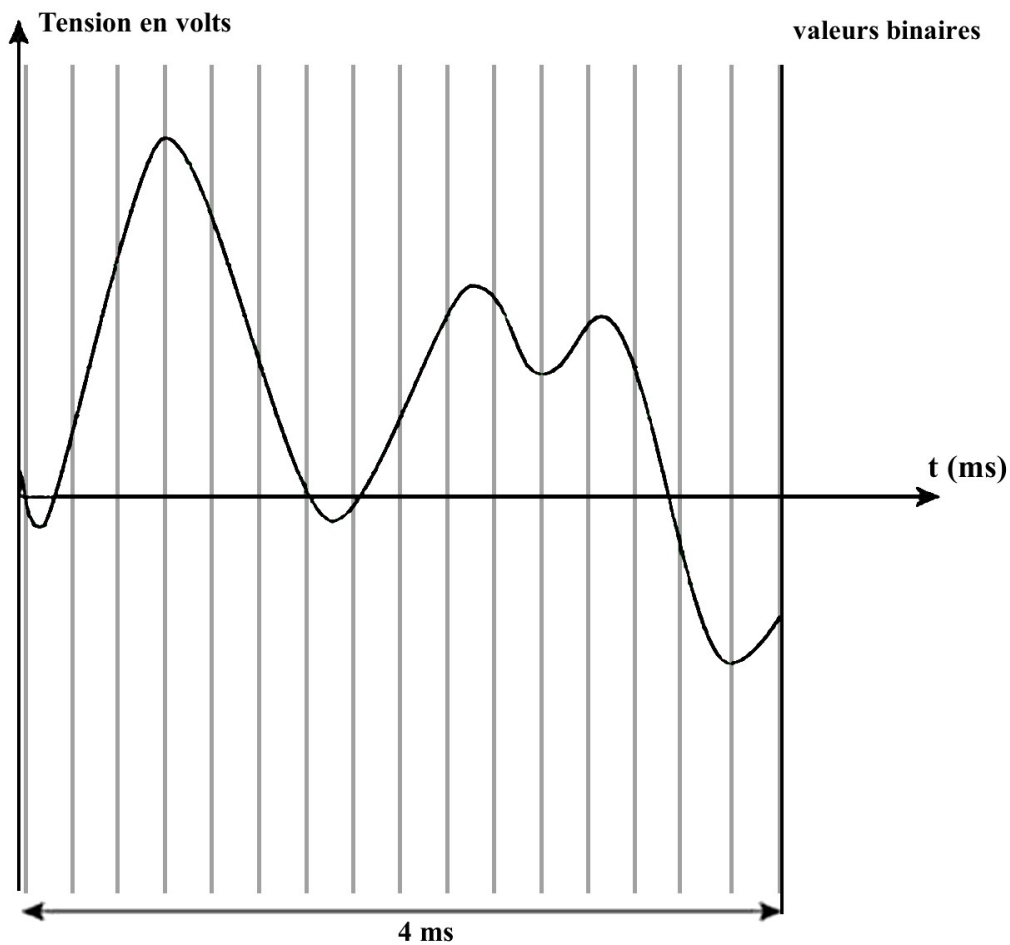
## Partie 2 : améliorations

On va essayer d'améliorer ce convertisseur A/N.

### Etape 1 : amélioration de la définition (amplitude)

On va passer le convertisseur sur 5 bits

1. Combien de valeurs différentes sont-elles possibles maintenant (amplitude) ?
2. Sachant que les valeurs max et min du signal analogique sont  $2V$  et  $-2V$ , calculez le pas de ce convertisseur
3. Tracer, au crayon, sur le dessin ci-dessous, la grille (quadrillage) de l'amplitude.
4. Tracer sur le dessin ci-dessous, en couleur, la courbe numérisée

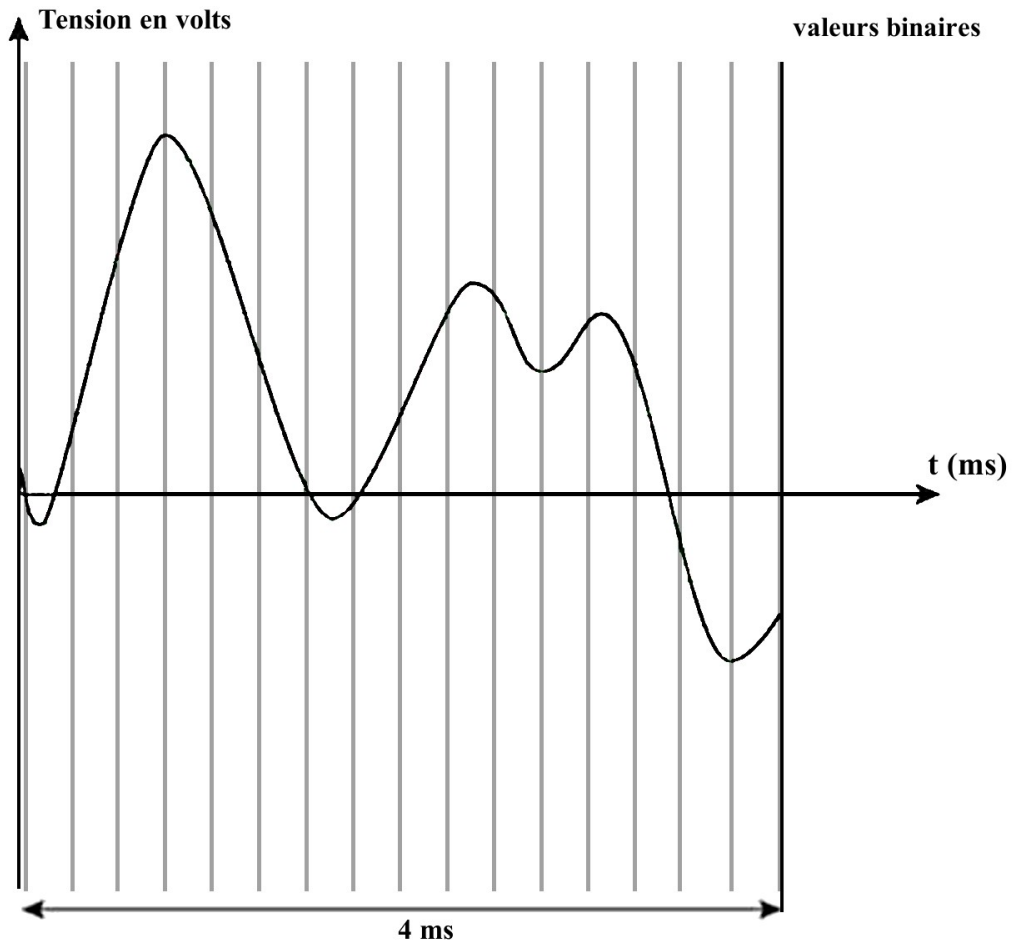


5. Pour mémoriser le signal numérique correspondant à ces 4ms, combien de bits de données va-t-on avoir?
6. Si le signal total dure maintenant 5 minutes, combien de bits de données seront nécessaires pour mémoriser numériquement le signal (donnez le résultat en bits, en octets et en ko)?

## Etape 2 : amélioration de la fréquence d'échantillonnage

Tout en laissant le convertisseur sur 5 bits, on va doubler la fréquence d'échantillonnage.

1. Tracez, au crayon, sur le dessin ci-dessous, la grille (quadrillage) de l'amplitude et de l'échantillonnage.
2. Tracez sur le dessin ci-dessous, en couleur, la courbe numérisée



3. Pour mémoriser le signal numérique correspondant à ces 4ms, combien de bits de données va-t-on avoir?
4. Si le signal total dure maintenant 5 minutes, combien de bits de données seront nécessaires pour mémoriser numériquement le signal (résultat en bits, en octets et en ko)?

## Exercice 4 : circuits logiques

### 1. Étude préliminaire : révision de quelques portes logiques élémentaires :

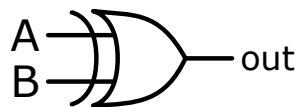
Porte OU :



a) Compléter la table de vérité :

A	B	Q

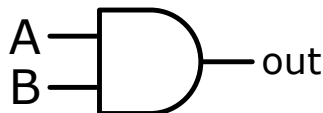
Porte OU Exclusif (XOR):



b) Compléter la table de vérité :

A	B	out

Porte ET :

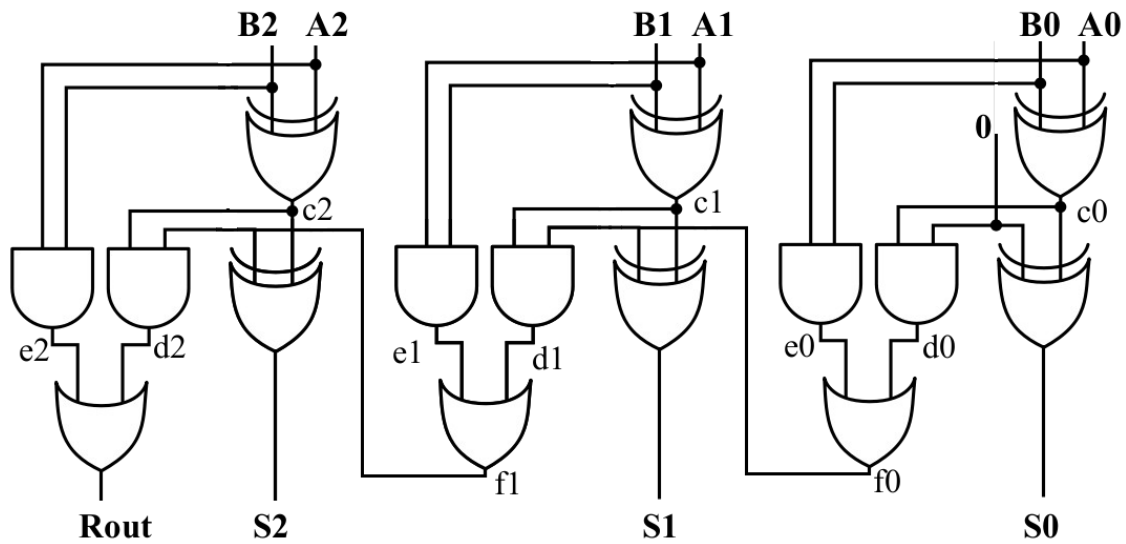


c) Compléter la table de vérité :

A	B	out

### 2. Un circuit logique inconnu à analyser.

soit le circuit suivant :



d) Complétez la table de vérité suivante (des variables intermédiaires ont été ajoutée comme  $c_0, e_0, \dots$  afin de faciliter la résolution)

	B2	B1	B0	A2	A1	A0	$c_0$	$d_0$	$e_0$	$f_0$	$c_1$	$d_1$	$e_1$	$f_1$	$c_2$	$d_2$	$e_2$	S2	S1	S0	$R_{out}$		
Cas 1	0	0	1	0	0	1																	
Cas 2	0	1	0	0	1	1																	
Cas 3	1	1	0	0	0	1																	
Cas 4	1	1	1	0	0	1																	

Essayons de comprendre ce que réalise ce circuit logique.

$B_0, B_1$  et  $B_2$  représentent un chiffre codé sur 3 bits (de même que  $A_0, A_1$  et  $A_2$  et aussi  $S_0, S_1$  et  $S_2$ ).

e) Complétez le tableau suivant avec les valeurs décimales de ces 3 chiffres

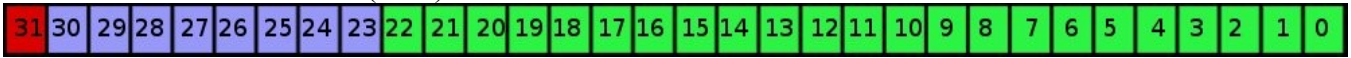
	Chiffre B	Chiffre A	Chiffre S
Cas 1			
Cas 2			
Cas 3			
Cas 4			

f) Déduisez des 3 premiers cas du tableau précédent la fonction réalisée par ce circuit logique

g) En vous aidant du cas 4, expliquez à quoi sert la sortie  $R_{out}$  ?

**Exercice 5 : nombre au format float**

Un nombre codé sur 32 bits (float) est de la forme:



- Les bits 0 à 22 (la mantisse) servent à coder le nombre sans tenir compte de la virgule.
- Les bits 23 à 30 servent à donner l'exposant : -126 à 127. Cette manière de coder est un peu bizarre. Pour avoir l'exposant il faut faire  $exp = \text{nombre(bits 23 à 30)} - 127$  (décalage)
- Le bit 31 sert à préciser le signe. (0= positif, 1=négatif)

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

$$\text{valeur} = \text{signe} \times (1, \text{mantisse})_{10} \times 2^{(\text{exposant} - \text{décalage})}$$

**exemple :** 0 10000010 110000000000000000000000

- signe = 0 : positif
- mantisse : 110000000000000000000000
- exposant : 10000010 = 128 + 2 = 130

mantisse :  $(1,110\dots)_2 = 1 \times 2^0 + (1 \times 2^{-1}) + (1 \times 2^{-2}) + (0 \times 2^{-3}) + \dots = 1, (0,5 + 0,25 + 0 + 0 + \dots) = 1,75$  (en décimal)

valeur = + 1,75 × 2<sup>(130 - 127)</sup> = 1,75 \* 2<sup>3</sup> = 14

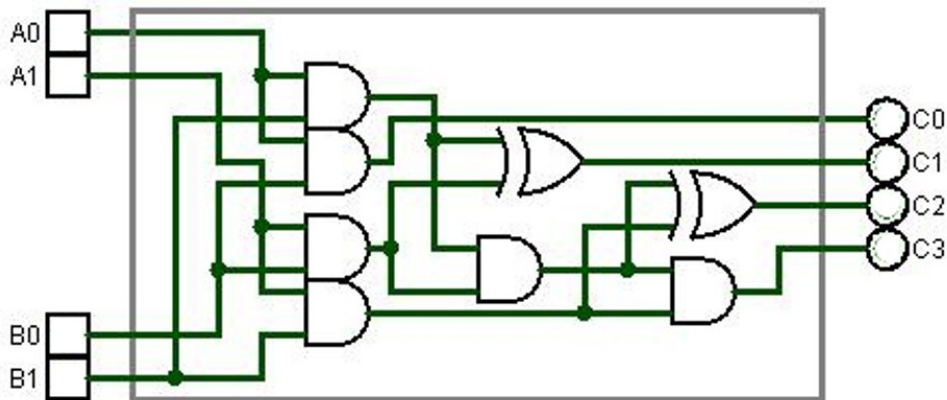
C'est à vous maintenant:

**Retrouvez la valeur binaire des nombres flottants suivants :**

- a) 1 10000010 111101000000000000000000
- b) 0 10001000 011011000010000000000000
- c) 1100001000001110000000000000000000

**Exercice 6 : circuit logique inconnu**

Soit le schéma suivant :



Appliquez une démarche similaire à l'exercice 4 pour trouver (et démontrer) le rôle de ce circuit logique.