3ème année
Génie électrique - Unité d'enseignement 6
Informatique Industrielle – Systèmes numériques

Optronique - Unité d'enseignement 5
Electronique- Systèmes numériques

Chapitre 1

Numérations – Algèbre de BOOLE - Quantification

Etienne Faure, Joëlle Maillefert

OBJECTIFS DU COURS

Il s'agit d'étudier les fondements de l'électronique numérique:

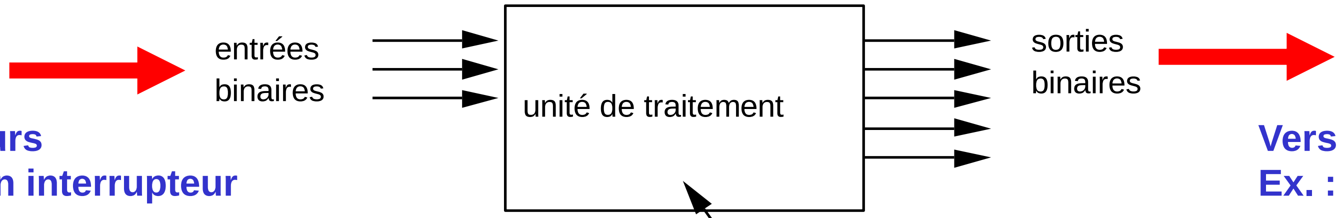
- les concepts mis en oeuvre
- les techniques de réalisation

Des approfondissements seront vus en $4^{ème}$ et en $5^{ème}$ selon les parcours

QU’EST-CE-QUE L’ELECTRONIQUE NUMERIQUE ?

L’électronique numérique traite (entrées) et fournit (sorties) des informations **BINAIRES**:

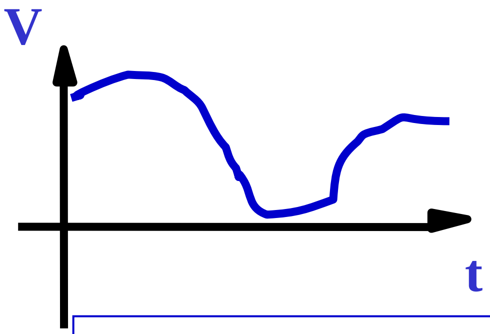
Deux valeurs	**0**	**ou**	**1**
Tension	**0 Volt**	**ou**	**5 Volt (3,3 Volt ou 1,8 Volt)**
Interrupteur	**ouvert**	**ou**	**fermé**
Voyant lumineux	**éteint**	**ou**	**allumé**



Capteurs
Ex. : un interrupteur

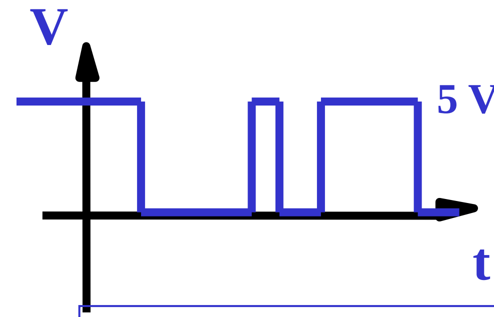
Vers les actionneurs
Ex. : un voyant

Comment réaliser cette “boîte” ?
Quelle technique choisir ?



Signal analogique

Par opposition, l'électronique analogique traite des informations continues (en amplitude)



Signal numérique

Disque vinyl
Radio FM

CD-Audio, MP3, JPEG
Carte SIM
carte de paiement
- Info codée
- Info mémorisée

Propriétés des techniques numériques

- plus robustes
- moins coûteuses
- adaptables
- plus lentes

Exemple:

Base 10	**Base 2**
0	**0**
1	**1**
2	**1 0**
3	**1 1**
4	**1 0 0**
5	**1 0 1**
6	**1 1 0**
7	**1 1 1**
8	**1 0 0 0**
9	**1 0 0 1**
10	**1 0 1 0**
11	**1 0 1 1**

NUMERATION DECIMALE

Base de calcul = base 10
10 CHIFFRES: 0 1 2 … 9

NUMERATION BINAIRE

Base de calcul = base 2
2 CHIFFRES: 0 et 1

Définitions: 1 chiffre = **1 bit**

1 nombre de 8 bits = **1 octet (byte)**

1 nombre de 16 bits = **1 mot de 16 bits (word)**

le bit de poids faible

le bit de poids fort

1 0 1 1 0 0 0 1

En pratique 1 chiffre sera matérialisé par un fil mis au potentiel 0 Volt ou 5 Volt (3,3 Volt ou 1,8 Volt aujourd'hui)

En électronique numérique, on indique toujours :

- **le nombre de bits (ou de fils !) avec lesquels on travaille**
- **le nombre d'entrées et le nombre de sorties**

Généralisation à la numération sur n bits

Un nombre X sur n bits est tel que

$0 \leq X \leq 2^n - 1$

Sur 4 bits : $0 \leq X \leq 15$

Sur 8 bits : $0 \leq X \leq 255$

Sur 16 bits : $0 \leq X \leq 2^{16} - 1 = 65535$

Conversion Base 2 / Base 10:

Soit A un nombre exprimé en binaire sur n bits,
tel que $A = (a_{n-1}\, a_{n-2} \,.........\, a_1\, a_0)_2$ par exemple $A = (10010011)_2$

Alors
$A = a_{n-1} \times 2^{n-1} + ... + a_2 \times 2^2 + a_1 \times 2^1 + a_0 \times 2^0$

Exemples sur 8 bits:

$A = (10010011)_2$ $A = 128 + 0 + 0 + 16 + 0 + 0 + 2 + 1 = 147$
$A = (00110110)_2$ $A = 0 + 0 + 32 + 16 + 0 + 4 + 2 + 0 = 54$

CONVERSION BASE 10 / BASE 2:

On recherche les puissances de 2 présentes dans le nombre, par ordre décroissant:

Exemples sur 8 bits:

A = 674 --> impossible car A>255

A = 123 A = 64 + 59 A = 64 + 32 + 27

A = 64 + 32 + 16 + 11 A = 64 + 32 + 16 + 8 + 3

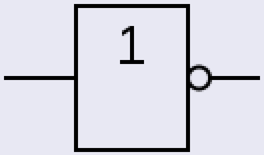
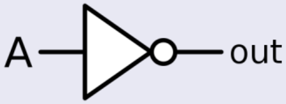
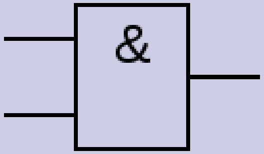
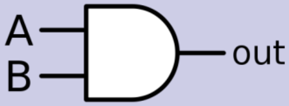
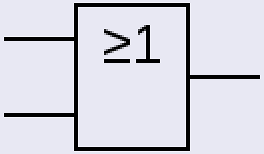
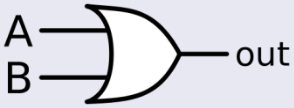
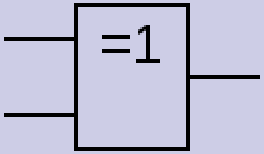
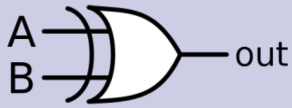
A = 64 + 32 + 16 + 8 + 2 + 1 $A = (01111011)_2$

A = 200

A = 128 + 72 A = 128 + 64 + 8 $A = (11001000)_2$

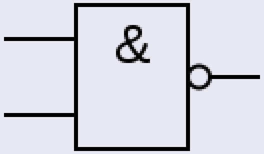
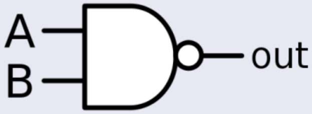
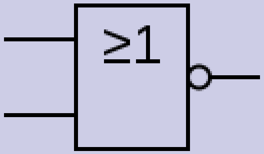
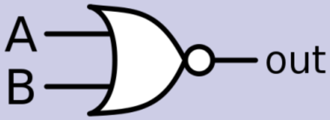
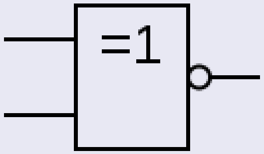
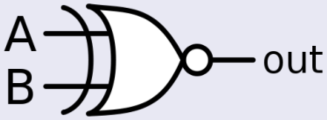
OPERATEURS LOGIQUES (1)

Les opérateurs logiques sont définis pour des nombres de 1 bit, tant pour les opérandes que pour le résultat.

NON	$\bar{A}$		
ET	$A.B$		
OU	$A + B$		
OU Exclusif	$A \oplus B$		

OPERATEURS LOGIQUES (2)

Opérateurs à sortie complémentée

NON ET NAND	$\overline{A.B}$		
NON OU NOR	$\overline{A+B}$		
NON OU Exclusif NXOR	$\overline{A \oplus B}$		

Tables de vérité

AND		
A	B	A . B
0	0	0
0	1	0
1	0	0
1	1	1

OR		
A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

XOR		
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

NAND		
A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

NOR		
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

XNOR		
A	B	$\overline{A \oplus B}$
0	0	1
0	1	0
1	0	0
1	1	1

OPERATIONS LOGIQUES ENTRE NOMBRES DE n BITS

On effectue l'opération bit à bit, soit n opérations entre nombres de 1 bit

Exemple: X = 1 1 0 0 1 0 0 1
Y = 0 0 1 0 1 1 0 0

/X= 0 0 1 1 0 1 1 0

X+Y= 1 1 1 0 1 1 0 1

X.Y= 0 0 0 0 1 0 0 0

/(X+Y)= 0 0 0 1 0 0 1 0

PROPRIETES DES OPERATEURS LOGIQUES

Commutativité : *Tous les opérateurs logiques sont commutatifs*

$$a \,.\, b = b \,.\, a \qquad a + b = b + a \qquad \overline{a \,.\, b} = \overline{b \,.\, a}$$

Démonstration **:** ***Les tables de vérité sont symétriques***

Associativité : *Seuls les opérateurs ET, OU , OU EXCLUSIF sont associatifs*

$$a + b + c = (a + b) + c \qquad a \,.\, b \,.\, c = (a \,.\, b) \,.\, c$$

Démonstration à faire (table de vérité)

Distributivité du ET par rapport au OU :

Forme générale : $(a+b) \,.\, (c+d) = a.c + a.d + b.c + b.d$

Démonstration à faire (table de vérité sur la forme simple (a+b).c)

LOIS DE DE MORGAN

Elles permettent de « casser » la barre des opérateurs NAND ET NOR, ces opérateurs n'étant pas associatifs.

Lois de DE MORGAN
$\overline{a+b} = \bar{a}.\bar{b}$
$\overline{a.b} = \bar{a}+\bar{b}$
$\overline{a+b+c} = \bar{a}.\bar{b}.\bar{c}$
$\overline{a.b.c} = \bar{a}+\bar{b}+\bar{c}$
$\overline{a+b+c+\cdots+x+y+z} = \bar{a}.\bar{b}.\bar{c}.\ldots.\bar{x}.\bar{y}.\bar{z}$
$\overline{a.b.c.\ldots.x.y.z} = \bar{a}+\bar{b}+\bar{c}+\cdots+\bar{x}+\bar{y}+\bar{z}$

Relations à connaître

$a + 1 = 1$	$a + 0 = a$	$a + a = a$	$a + \overline{a} = 1$
$a.1 = a$	$a.0 = 0$	$a.a = a$	$a.\overline{a} = 0$
$a \oplus 1 = \overline{a}$	$a \oplus 0 = a$	$a \oplus a = 0$	$a \oplus \overline{a} = 1$

MISE EN EQUATIONS LOGIQUES: CAHIER DES CHARGES

Le cahier des charges décrit « en langage courant » le système logique à réaliser (pour notre exemple, un distributeur de café) :

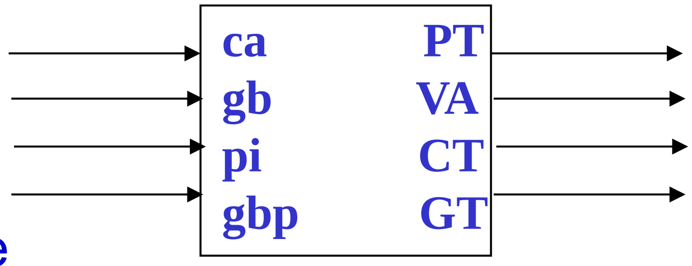
- « La pièce est avalée si elle est présente et s'il reste du café et des gobelets »
- « Le gobelet tombe lorsque la pièce est avalée »
- « Le café tombe si un gobelet non plein est en place sous le bec verseur»
- « Un voyant rouge s'allume s'il manque du café, ou des gobelets, ou qu'un gobelet plein est en place sous le bec verseur. »

MISE EN EQUATIONS LOGIQUES: ENTREES / SORTIES

1ère tâche : identifier les entrées et les sorties :

Entrées :
ca : vaut 1 s’il reste du café
gb : vaut 1 s’il reste des gobelets
pi : vaut 1 si la pièce est présente
gbp : vaut 1 si un gobelet plein est présent sous le bec verseur

Sorties :
PT : mise à 1 pour que la pièce tombe
VA : mise à 1 pour que le voyant s’allume
CT : mise à 1 pour que du café coule
GT : mise à 1 pour qu’un gobelet tombe



« boîte » à réaliser

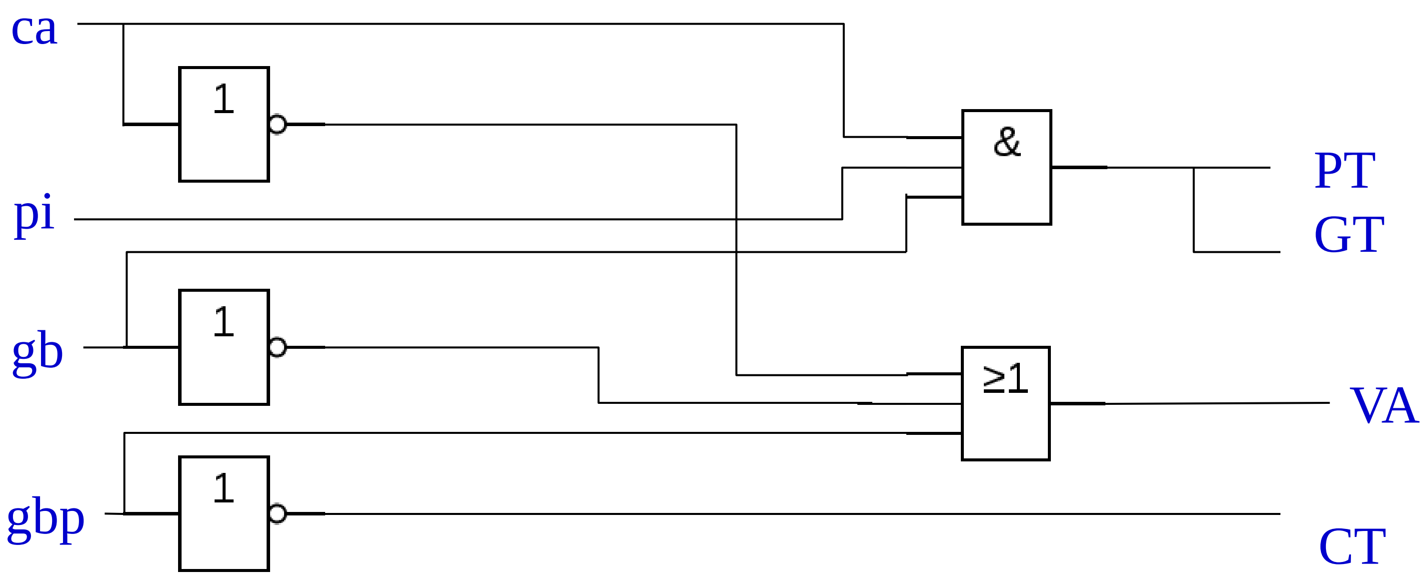
MISE EN EQUATIONS LOGIQUES: LES EQUATIONS

2ème tâche : mettre les sorties en équations

PT = ca . pi . gb   GT = PT = ca . pi . gb

CT = /gbp   VA = /ca + /gb + gbp

3ème tâche : représenter le logigramme



NUMERATION HEXADECIMALE (base 16)

16 chiffres :
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f.

Si un nombre A s'écrit **avec p chiffres** en base 16 alors,

$$\mathbf{A = h_{p-1} \times 16^{p-1} + \ldots\ldots + h_1 \times 16 + h_0}$$

Exemple d'un nombre de 2 chiffres en base 16 :

$$\mathbf{A = 25 = 1x16 + 9 = 0x19}$$

h_1 h_0

NUMERATION HEXADECIMALE (base 16)

1 chiffre	2 chiffres	3 chiffres
$0 \leq A \leq 15$	$0 \leq A \leq 255$	$0 \leq A \leq 16^3 - 1$

Base 10	Base 16	
25	0x19	25 = 16 + 9
32	0x20	32 = 2 x 16 + 0
45	0x2D	45 = 2 x 16 + 13
154	0x9A	154 = 9 x 16 + 10
155	0x9B	

Lien entre la base 2 et la base 16

Soit $A = (a_{n-1}\ a_{n-2}\ \ldots\ldots\ldots\ a_1\ a_0)_2$ par exemple $A = (10010011)_2$

$$A = 128a_7 + 64a_6 + 32a_5 + 16a_4 + 8\,a_3 + 4a_2 + 2a_1 + a_0$$

$$A = 16(\underbrace{8a_7 + 4a_6 + 2a_5 + a_4}_{\text{1 chiffre hexa}}) + \underbrace{8\,a_3 + 4a_2 + 2a_1 + a_0}_{\text{1 chiffre hexa}}$$

$$\underbrace{1001}_{9}\ \underbrace{0011}_{3}$$

A = 0x93

Pour passer de la base 2 à la base 16, on regroupe les bits 4 par 4 en commençant à droite, puis on identifie chaque chiffre de la base 16.

Lien entre la base 2 et la base 16 - quelques exemples

Base 10	Base 2 (4 bits)	Base 16 (1 chiffre)
8	1000	0x8
12	1100	0xc
15	1111	0xf

Base 10	Base 2 (8 bits)	Base 16 (2 chiffres)
8	0000 1000	0x08
12	0000 1100	0x0c
15	0000 1111	0x0f
142	1000 1110	0x8e
255	1111 1111	0xff

De nombreuses calculatrices connaissent l'hexadécimal

CODAGE EN BINAIRE SIGNE

Problème : comment coder en binaire les nombres négatifs ?

A est un octet – On calcule A plus /A plus 1

$$
\begin{array}{ccccccccccc}
 & & 1 & 1 & 1 & 1 & 1 & 1 & 1 & & \\
 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & & A \\
+ & /a_7 & /a_6 & /a_5 & /a_4 & /a_3 & /a_2 & /a_1 & /a_0 & & /A \\
+ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & & 1 \\
\hline
(1) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & &
\end{array}
$$

Si on ne tient pas compte de la retenue, on obtient 0.

CODAGE EN BINAIRE SIGNE

On appelle *<u>complément à 2</u>* d'un nombre A le nombre $(\overline{A} + 1)$

Ce nombre est *<u>l'opposé de A</u>*

Exemples :

Complément à 2 de A =10101010 : /A =01010101 /A plus 1 = 01010110

Complément à 2 de A =11111111 : /A =00000000 /A plus 1 = 00000001

Complément à 2 de A =00000000 : /A =11111111 /A plus 1 = 00000000

Complément à 2 de A= 0xff : /A = 0x00 /A plus 1 = 0x01

Complément à 2 de A=0x7f: /A =10000000 /A plus 1 = 10000001 = 0x81

A=01111111

CODAGE EN BINAIRE SIGNE

Règle de codage:

Si A$\geq$ 0 alors le codage en BS = le codage en BN

Si A< 0 alors
- **le codage en BN n ’existe pas**
- **pour obtenir le codage en BS, on code |A| (positif !) puis on calcule /|A| plus 1**

Il faut définir au départ la taille des nombres

On ne “voit” pas qu’un nombre est codé en BS, il faut que ce soit annoncé

CODAGE EN BINAIRE SIGNE

Autre règle :
Que A soit positif ou négatif, on passe de A à –A en calculant /A + 1

Exemples sur 8 bits :

A = 19 en base 10 = $(0001\ 0011)_2$ = 0x13 en binaire signé

-A = /A +1 = $(1110\ 1100)_2$ + $(0000\ 0001)_2$ = $(1110\ 1101)_2$ = 0xED

0xED est le codage en binaire signé de -19

A = -50 en base 10 = $(1100\ 1110)_2$ = 0xCE en binaire signé

-A = /A +1 = $(0011\ 0001)_2$ + $(0000\ 0001)_2$ = $(0011\ 0010)_2$ = 0x32

0x32 est le codage en binaire signé de +50

CODAGE EN BINAIRE SIGNE

Binaire signé	Décimal
1000	-8
1001	-7
1010	-6
1011	-5
1100	-4
1101	-3
1110	-2
1111	-1

Binaire signé	Décimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

CODAGE EN BINAIRE SIGNE

En binaire signé sur n bits, un nombre A est tel que

$$-2^{n-1} \leq A \leq 2^{n-1} - 1$$

Exemples :

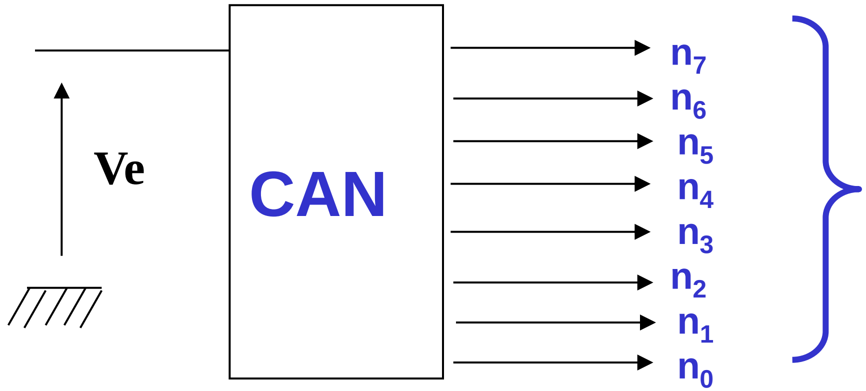
n = 8 : $-128 \leq A \leq +127$

n = 16 : $-2^{15} \leq A \leq 2^{15} - 1$ $\quad - 32768 \leq A \leq +32767$

QUANTIFICATION

Comment passer du monde analogique au monde numérique ?

Convertisseur Analogique Numérique p bits (ici p = 8)



$$0 \leq N \leq 2^p - 1$$

(ici 255)

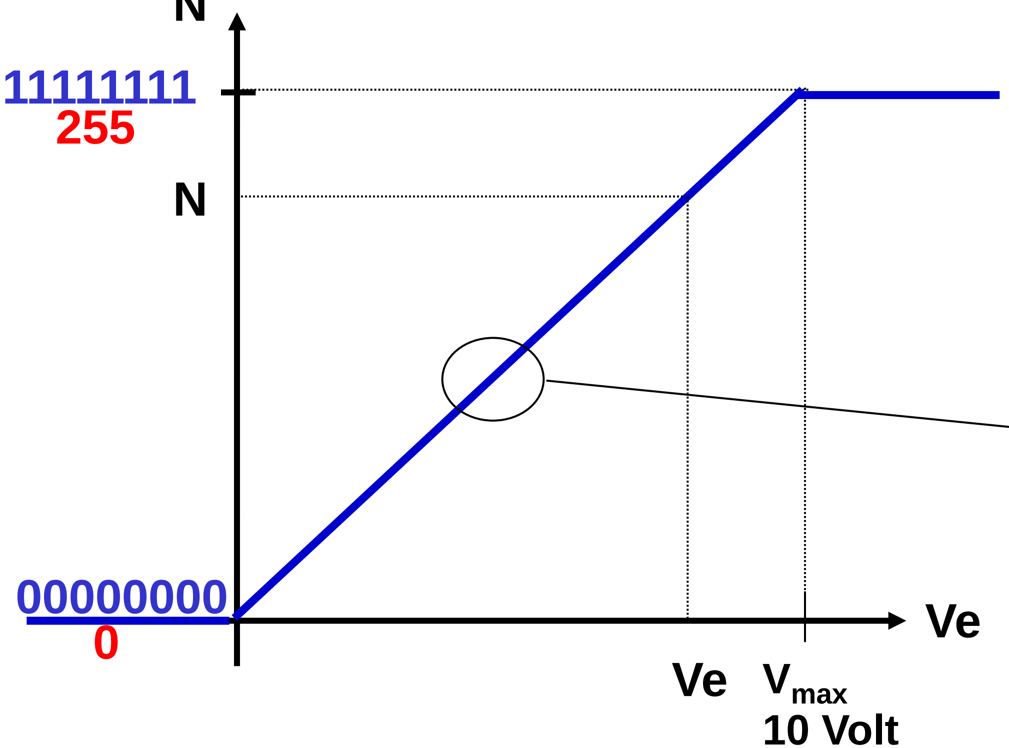
Ve est une tension analogique, comprise entre 0 et V_{max} (par exemple 10Volt)

Le convertisseur analogique/numérique fournit p (ici 8) signaux binaires $n_{p-1}\ n_{p-2}\ \ldots\ n_5\ n_4\ n_3\ n_2\ n_1\ n_0$ représentant un nombre N compris entre 0 et 2^p-1

=> On dit que l'on quantifie le signal Ve

QUANTIFICATION

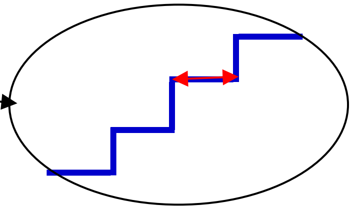
Fonction de transfert



$$N = (2^P - 1) . \frac{Ve}{Vmax}$$

ici p=8 et Vmax=10V

N est un nombre entier



pour N=1 $\Delta Ve = \frac{V_{max}}{2^p-1}$

exemples :

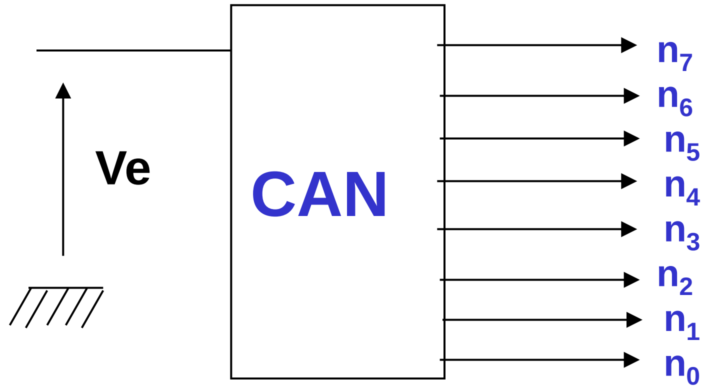
Ve = 2,8 V N = 71 = $(01000111)_2$

Ve = 7,3 V N = 186 = $(10111010)_2$

QUANTIFICATION

Le convertisseur peut fonctionner en binaire signé

Convertisseur Analogique Numérique p bits (ici p = 8)



$$-2^{p-1} \leq N \leq 2^{p-1} - 1$$

$$(\text{ici } -128 \leq N \leq +127)$$

Ve est une tension analogique, comprise entre V_{min}<0 et V_{max}>0 (par ex ±5 Volt)

Le convertisseur analogique/numérique fournit p (ici 8) signaux binaires $n_{p-1}\ n_{p-2} \ldots\ n_5\ n_4\ n_3\ n_2\ n_1\ n_0$ représentant un nombre N compris entre -2^{p-1} et $2^{p-1}-1$

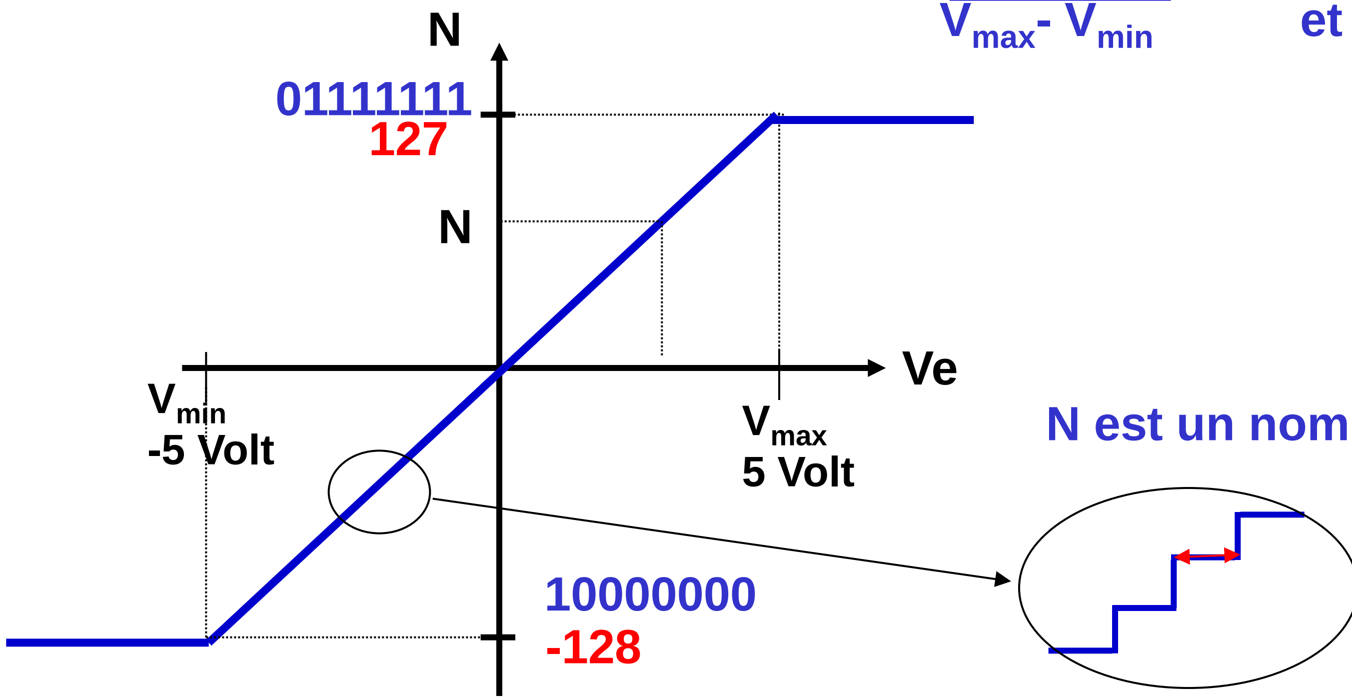
QUANTIFICATION

Fonction de transfert

$$N = \frac{(2^p - 1) \times Ve}{V_{max} - V_{min}}$$

ici p=8

et $V_{max} - V_{min} = 10V$



N est un nombre entier

pour N=1 $\Delta Ve = \frac{V_{max} - V_{min}}{2^p - 1}$

<u>exemples :</u>

Ve = 2,8 V N =71 = $(01000111)_2$

Ve = -3,7 V N =-95 = $(10100001)_2$