INSTITUT d'OPTIQUE GRADUATE SCHOOL

RAPPELS D'ELECTRONIQUE ANALOGIQUE ET NUMERIQUE

1ère Année

Franck DELMOTTE

Année 2011/2012

CONTENU

PARTIE A : Electronique analogique

PARTIE B : Electronique numérique

PARTIE A : Electronique analogique

1) Définitions et lois élémentaires

Commençons avec quelques rappels de vocabulaire. On appelle **réseau** un ensemble de dipôles reliés les uns aux autres, un dipôle étant un composant électrique avec deux bornes.

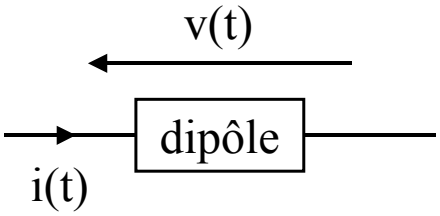


Figure 1 : exemple d'un dipôle passif (convention récepteur).

Le courant i(t) qui traverse le dipôle et la tension v(t) à ses bornes caractérisent le fonctionnement du dipôle (figure 1). Un réseau est dit **linéaire** si, pour chacun des dipôles le constituant, la relation fonctionnelle $\phi(v,i) = 0$ est linéaire et invariante dans le temps. Cette relation fonctionnelle pourra donc toujours se mettre sous la forme suivante :

$$a_n \frac{d^n v}{dt^n} + ... + a_1 \frac{dv}{dt} + a_0 v + b_p \frac{d^p i}{dt^p} + ... + b_1 \frac{di}{dt} + b_0 i = 0 \qquad \text{(eq.1)}$$

Dans un réseau, on distingue des **branches**, des **nœuds** et des **mailles**.

Une **branche** est un ensemble de dipôles connectés en série sans aucune dérivation de courant. Tous les dipôles d'une même branche sont donc traversé par un courant identique.

Considérons par exemple le réseau représenté sur la figure 2 : les chemins EFAB ou BCDE sont des branches. Par contre, les chemins ABC ou DEF ne sont pas des banches car il y a dérivation de courant en B et en E.

Tout point du réseau auquel à lieu une dérivation de courant est un **nœud**. Dans l'exemple précédent, les seuls nœuds du réseau sont les points B et E.

Enfin, on appelle **maille** du réseau tout chemin fermé. Sur la figure 2, le réseau comporte trois mailles :ABCDEF, ABEF et BCDE.

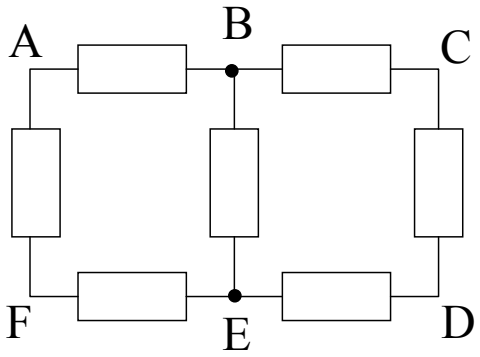


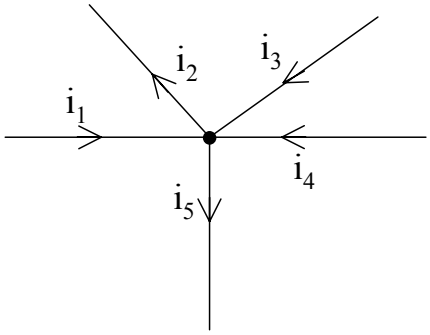
Figure 2 : exemple de réseau.

Lois de Kirchoff

Les lois de Kirchoff sont la base de tout calcul des circuits électriques. La **loi des nœuds** traduit tout simplement la conservation de la charge électrique : en un nœud, la somme algébrique des courants est nulle.

Il est souvent plus pratique de l'utiliser sous la forme suivante :

en un nœud, la somme des courants entrants est égale à la somme des courants sortants.

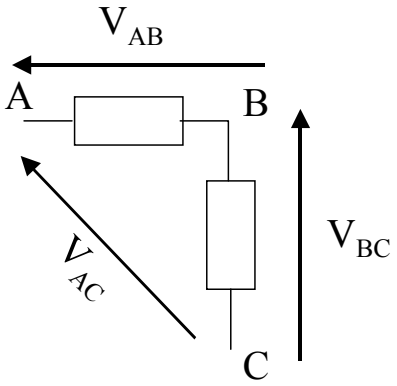


$$i_1 + i_3 + i_4 = i_2 + i_5$$

Figure 3 : La loi des noeuds.

La **loi des mailles** traduit l'additivité des tensions :

la tension aux bornes d'une branche est égale à la somme algébrique des tensions aux bornes de chacun de ses dipôles.



$$V_{AC} = V_{AB} + V_{BC}$$

Figure 4 : La loi des mailles.

Analyse nodale, analyse maillée

L'analyse nodale permet d'éviter d'écrire des courants intermédiaires. La méthode consiste à écrire la loi des nœuds à l'aide des potentiels voisins de ce nœud.
Sur l'exemple représenté en figure 5, on peut écrire au nœud A l'équation suivante :

$$\frac{E - V_A}{R_1} - \frac{V_A}{R_2} + \frac{V_S - V_A}{R_3} = 0$$

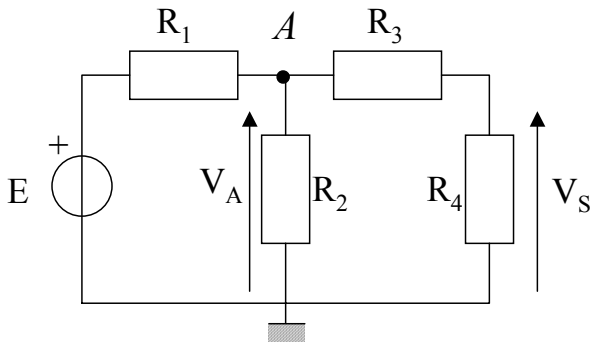
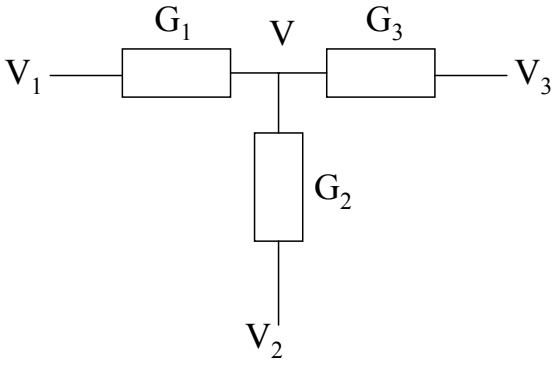


Figure 5 : exemple d'analyse nodale.

Le **théorème de Millman** est une forme particulière et très utile de cette méthode. Son énoncé est le suivant :

> lorsque les branches s'appuyant sur un nœud ne sont que des résistances,
> le potentiel du nœud est la moyenne des potentiels des nœuds voisins,
> pondéré par les conductances des résistances respectives.



$$(G_1 + G_2 + G_3)\, V = G_1\, V_1 + G_2\, V_2 + G_3\, V_3$$

Figure 6 : illustration du théorème de Millman.

La généralisation de cet exemple donne la formule mathématique suivante :

$$V = \frac{\sum_i G_i V_i}{\sum_i G_i}$$

où les V_i représentent les potentiels des i nœuds voisins, et les G_i les conductances des résistances respectives.

La formule du pont **diviseur de tension** n'est qu'une application de ce théorème (figure 7). En effet, d'après le théorème précédent de Millman, on peut écrire :

$$S = \frac{\frac{E}{R_1} + \frac{0}{R_2}}{\frac{1}{R_1} + \frac{1}{R_2}},$$

soit $\quad S = \frac{R_2}{R_1 + R_2} \times E\,.$

On retrouve bien évidemment la formule classique du diviseur de tension (à connaître par cœur). On peut vérifier avec cette formule que si R_2 devient nulle (i.e. on la remplace par un fil dans le schéma) S passe à 0.

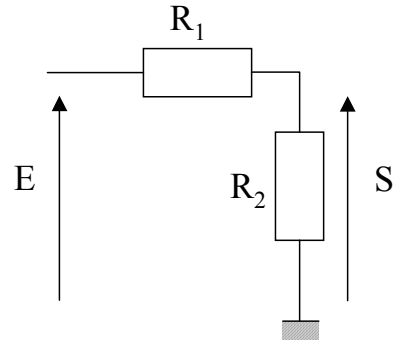


Figure 7 : Le pont diviseur de tension.

L'analyse maillée permet d'éviter d'écrire des tensions intermédiaires. La méthode consiste à écrire la loi des mailles à l'aide des courants circulant dans les branches d'une maille. Un exemple est traité sur la figure 8. Par cette méthode on peut écrire directement l'équation suivante correspondant à la maille fléchée :

$$R_2 (I_S - J) + R_3 I_S + R_4 I_S = 0.$$

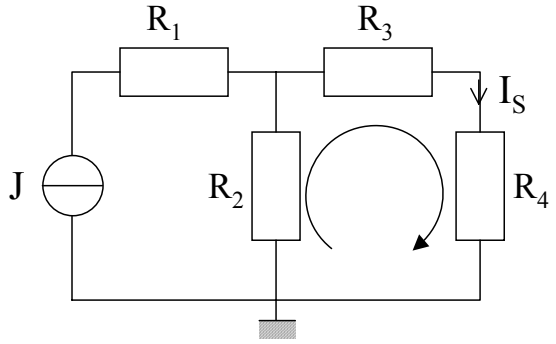


Figure 8 : exemple d'analyse nodale.

La formule du pont **diviseur de courant** est une application courante de cette méthode (figure 9). En effet, on obtient directement : $R_1 (I_2 - J) + R_2 I_2 = 0$,

$$\text{soit} \qquad I_2 = \frac{R_1}{R_1 + R_2} \times J .$$

On retrouve la formule classique du diviseur de courant (à connaître par cœur). On peut vérifier avec cette formule que si R_2 tend vers l'infini (i.e. on la remplace par circuit ouvert dans le schéma) I_2 passe à 0.

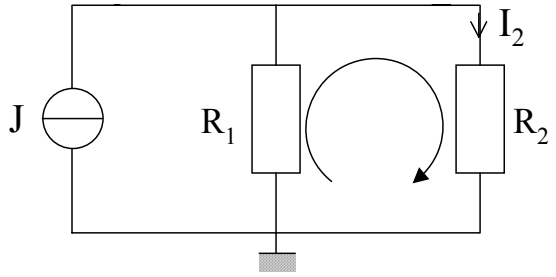


Figure 9 : Le pont diviseur de courant.

2) Dipôles actifs

Convention de sens de courant

Pour les dipôles « actifs » on utilise la **convention générateur** : le sens du courant à l'extérieur du dipôle suit celui de la tension générée.

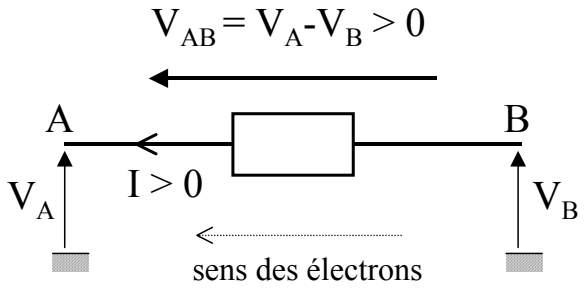


Figure 10 : dipôle actif : convention générateur.

Source de tension

Une **source de tension idéale** est caractérisée par sa force électromotrice (fem) exprimée en volt et par la relation : dV/dI = 0.

Le symbole utilisé pour représenter ce dipôle et sa caractéristique I(V) sont donnés sur la figure 11.

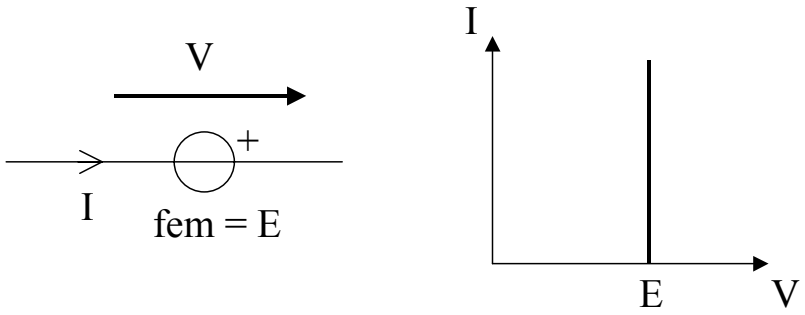


Figure 11 : Source de tension idéale.

Un générateur de tension est généralement modélisé par une source de tension idéale de fem E en série avec une résistance R_g. La branche est alors caractérisée par la relation : $V = E - R_g I$. La caractéristique est représentée sur la figure 12. Plus **la résistance R_g est faible** plus on se rapproche du cas idéal.

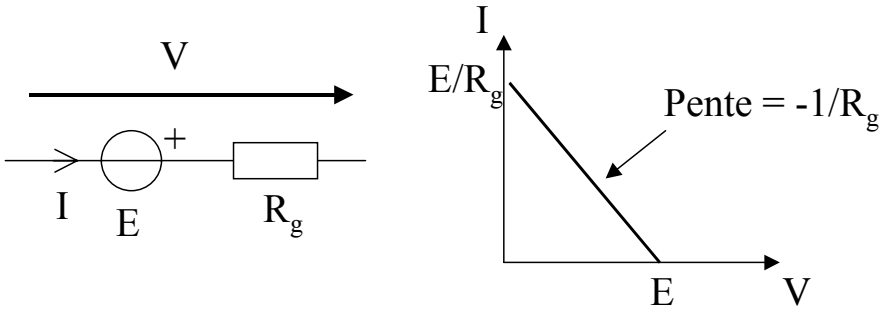


Figure 12 : Source de tension réelle.

Source de courant

Une **source de courant idéale** est caractérisée par son courant électromotrice (cem) exprimée en ampère et par la relation : dI/dV = 0.

Le symbole utilisé pour représenter ce dipôle et sa caractéristique I(V) sont donnés sur la figure 13.

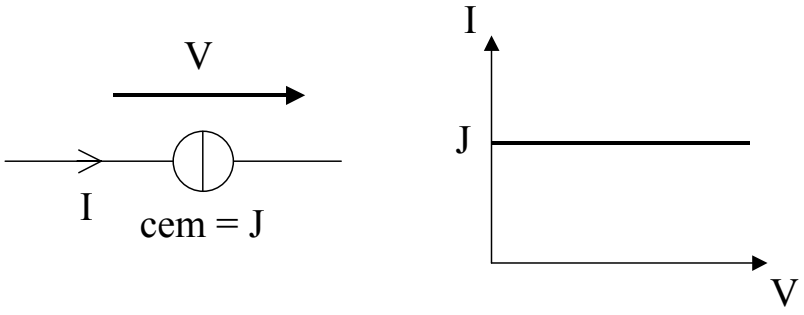


Figure 13 : Source de courant idéale.

Un générateur de courant est généralement modélisé par une source de courant idéale de cem J en parallèle avec une résistance R_g. La branche est alors caractérisée par la relation : $I = J - V/R_g$. La caractéristique est représentée sur la figure 14. Plus **la résistance R_g est élevée** plus on se rapproche du cas idéal.

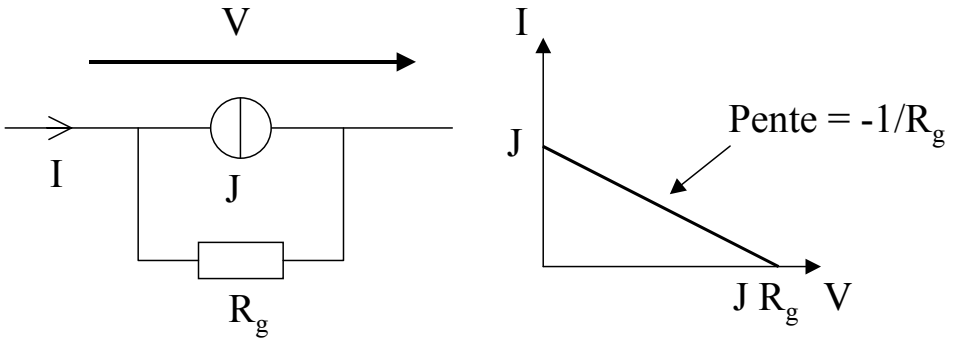


Figure 14 : Source de courant réelle.

3) Dipôles passifs idéaux

Convention de sens de courant

Pour les dipôles « passifs », on utilise la **convention récepteur** : le sens du courant est opposé à celui de la tension appliquée (figure 15).

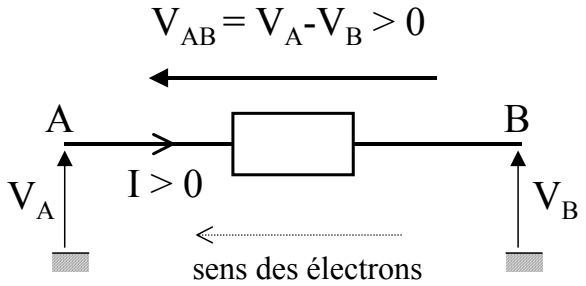
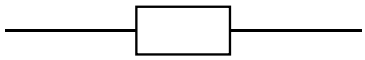


Figure 15 : dipôle passif : convention récepteur.

La résistance parfaite

Pour un dipôle purement résistif, la tension est proportionnelle au courant qui le traverse. Cette propriété remarquable est décrite la loi d'Ohm : $v(t) = R\ i(t)$ ou R est la résistance exprimée en Ohm (Ω). On utilise parfois également la conductance $G = 1/R$ exprimée en Siemens (S).

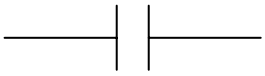


Valeurs usuelles : de 10 Ω à 1 MΩ.
Valeur typique : 1kΩ.

Le condensateur parfait

D'après les lois de l'électrostatiques, la charge présente sur une armature d'un condensateur est proportionnelle à la tension à ses bornes : $q(t) = C\ v(t)$. Le coefficient de proportionnalité C est la capacité exprimée en Farad (F).
On obtient par dérivation, la relation fonctionnelle du condensateur parfait : **$i(t) = C\ dv(t)/dt$**.
Le condensateur joue le rôle d'un réservoir de charge. Le nombre de charge sur une armature ne pouvant pas varier de manière discontinue, on en déduit la propriété essentielle suivante :

la tension aux bornes d'un condensateur évolue de façon continue.



Valeurs usuelles : de 1 pF à 10 µF.
Valeur typique : 1 nF.

La self ou l'inductance parfaite

Dans une self, le flux d'induction magnétique est proportionnelle au courant : $\phi(t) = L\ i(t)$. Le coefficient de proportionnalité L est l'inductance exprimée en Henry (H).
On obtient par dérivation, la relation fonctionnelle de la self parfaite: **$v(t) = L\ di(t)/dt$**.
La self joue le rôle d'un réservoir de courant et possède la propriété essentielle suivante :

le courant traversant une self évolue de façon continue.

Valeurs usuelles : de 100 nH à 10 mH.
Valeur typique : 1 µH.

4) Calcul des circuits

Théorème de superposition

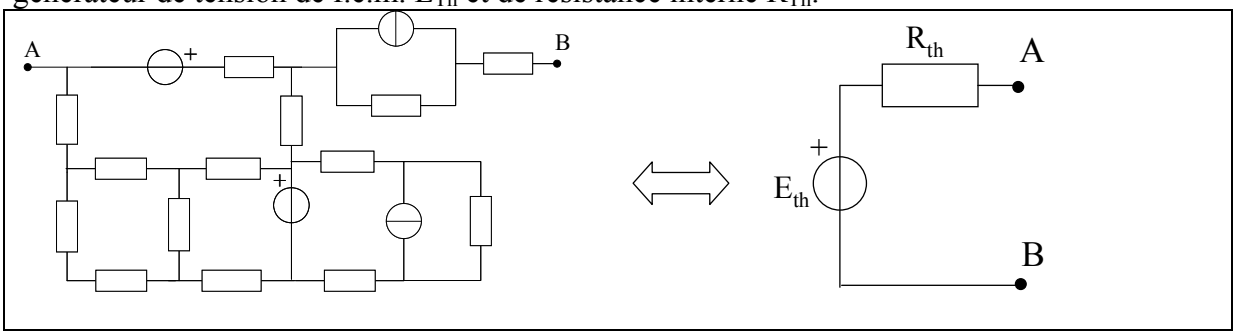
Lorsque le circuit est trop compliqué, on le découpe en circuits simples.

Si un circuit linéaire comprend plusieurs générateurs, le courant circulant dans une branche de ce circuit est la somme des courants qui seraient créés par chaque générateur pris isolément (les autres générateurs ayant été remplacés par leur impédance interne).

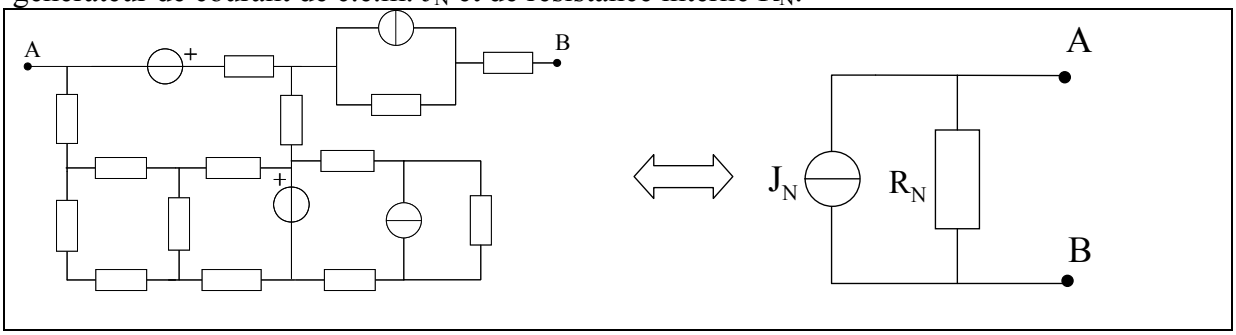
Théorème de Thévenin et Norton

Tout réseau linéaire de pôles A et B se comporte comme un simple générateur

Thévenin : un réseau linéaire pris entre deux nœuds quelconques (A et B) est équivalent à un générateur de tension de f.é.m. E_{Th} et de résistance interne R_{Th}.[1]



Norton : un réseau linéaire pris entre deux nœuds quelconques (A et B) est équivalent à un générateur de courant de c.é.m. J_N et de résistance interne R_N.[2]



E_{Th} est la tension à vide du réseau c'est-à-dire la tension que l'on mesure entre A et B lorsque le circuit extérieur est ouvert (I = 0).
R_{Th} (= R_N) est la résistance équivalente du réseau lorsque l'on éteint les sources indépendantes.
J_N est le courant de court-circuit c'est-à-dire le courant que l'on mesure entre A et B lorsque le circuit extérieur est remplacé par un court-circuit.
On montre facilement que (il suffit d'appliquer le théorème de Norton à un générateur Thévenin) :

$$J_N = E_{Th} / R_{Th} ;$$

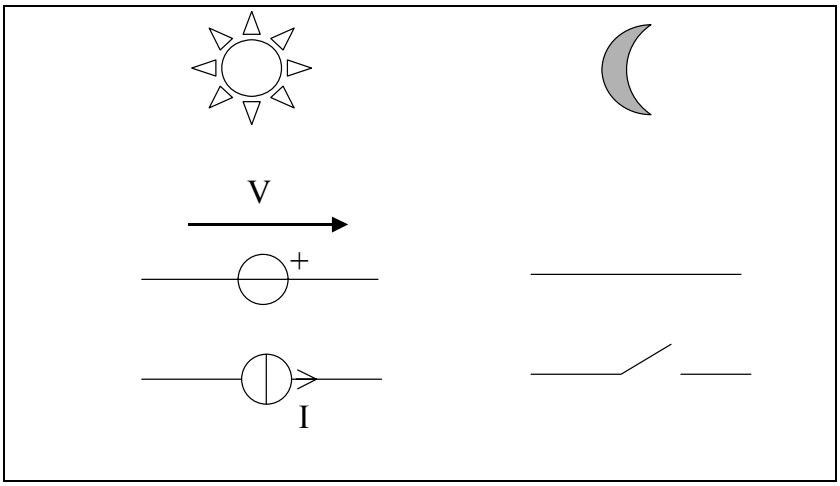
et que les résistances internes des deux générateurs (Thévenin et Norton) sont égales :

$$R_N = R_{Th} .$$

Extinction des sources

Une source éteinte n'apporte plus d'énergie extérieure au réseau. Pour une source de tension on annule la fem (V=0), ce qui revient à remplacer la source de tension par un court-circuit. Pour une source de courant, on annule le cem, ce qui revient à remplacer la source de courant par un circuit ouvert.

[1] A l'exclusion des sources de courant idéales qui n'ont pas d'équivalent Thévenin.
[2] A l'exclusion des sources de tension idéales qui n'ont pas d'équivalent Norton.



Une **source est dite commandée ou liée** lorsque sa grandeur caractéristique (fem ou cem) est fonction d'une grandeur électrique (tension ou courant) du réseau.

On ne peut pas éteindre une source liée.

En effet, l'énergie qu'apporte une telle source n'est pas extérieure au réseau. Il s'agit d'énergie interne au réseau.

5) Réseaux linéaires en régime harmonique

Représentation de Fresnel

En régime harmonique on considère que le signal électrique (tension ou courant) est une fonction sinusoïdale du temps. Ce signal s(t) s'exprime donc sous la forme :

$$s(t) = S \cos(\omega t + \varphi)$$

où S est son amplitude (en V ou en A), ω sa pulsation (en rad/s) et φ son angle de phase (en rad). La représentation temporelle du signal s(t) (figure 16) ne donne pas d'information immédiate sur la phase ni sur la pulsation du signal. Par contre elle donne accès graphiquement aux différents temps caractéristiques : période $T = 2\pi / \omega$, et délai $= \varphi / \omega$.

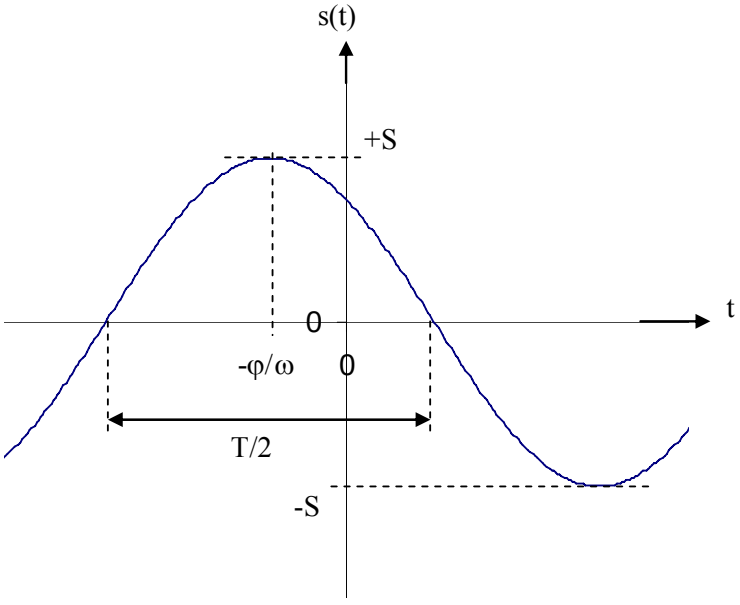


Figure 16 : Représentation temporelle de s(t).

Lorsque l'on souhaite avoir une représentation graphique des pulsations et des phases, notamment pour comparer deux signaux sinusoïdaux entre eux, il est commode d'utiliser la représentation en vecteur tournant, ou représentation de Fresnel. On trace dans un repère (x,y) un vecteur de norme S faisant un angle ($\omega t + \varphi$) avec l'axe des abscisses (figure 17a). Ce vecteur tourne dans le sens trigonométrique autour de l'origine du repère lorsque le temps s'écoule (ω est d'ailleurs appelé également vitesse angulaire). Le signal s(t) est simplement la projection du vecteur sur l'axe des abscisses.

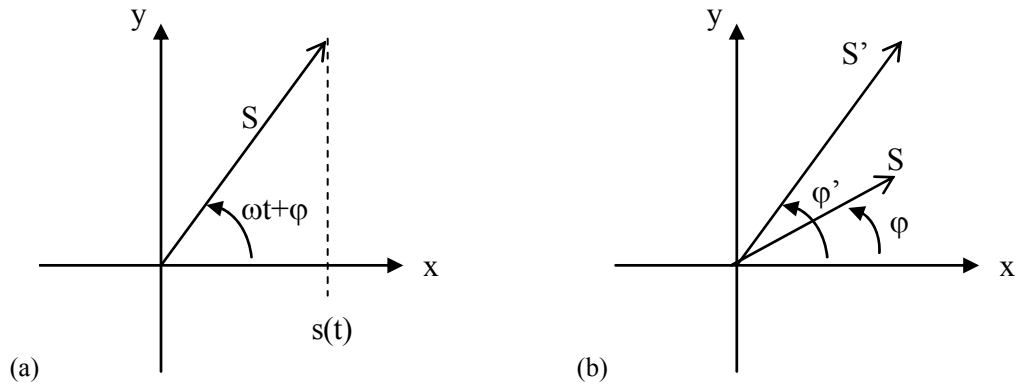


Figure 17 : (a) Représentation de Fresnel de s(t), (b) Comparaison de deux signaux.

Pour comparer deux signaux sinusoïdaux, il suffit de les tracer dans le même repère. La figure 17b représente par exemple deux signaux de même pulsation mais d'amplitude et de phase différentes pour un temps t=0. On appelle **(φ'-φ) le déphasage de s'(t) par rapport à s(t)** ou encore **retard de s(t) sur s'(t)**. Dans le cas de la figure 17b, (φ'-φ) est positif et s(t) est bien en retard sur s'(t).

Notation complexe

Définition

Considérons la représentation de Fresnel d'un signal sinusoïdal. On peut associer 2 grandeurs réelles au vecteur tournant : sa projection sur l'axe x, s1(t), et sa projection sur l'axe y, s2(t). Cela revient en fait à associer à notre signal $s1(t) = S \cos(\omega t + \varphi)$ un deuxième signal $s2(t) = S \sin(\omega t + \varphi)$ On définit alors un signal complexe $\underline{s}(t)$ associé au vecteur tournant par : $\underline{s}(t) = s1(t) + j\, s2(t) = S \exp(j(\omega t + \varphi))$.

Ce signal complexe peut encore s'écrire : $\underline{s}(t) = S \exp(j\varphi) \exp(j\omega t)$.

Le terme $\underline{S} = S \exp(j\varphi)$ est appelé amplitude complexe du signal.

Intérêt

Considérons maintenant une branche d'un réseau linéaire parcourue par un signal i(t). Il apparait alors à ses bornes une tension v(t). En régime harmonique, i(t) et v(t) sont des signaux sinusoïdaux et de même pulsation. Le rapport v(t)/i(t) n'est pas significatif pour caractériser cette branche du réseau car il évolue dans le temps (sauf si les signaux sont en phase). En notation complexe on peut écrire :

$\underline{v}(t) = \underline{V} \exp(j\omega t)$

et $\underline{i}(t) = \underline{I} \exp(j\omega t)$.

le rapport $\underline{V}/\underline{I}$ est un nombre complexe indépendant du temps. Il caractérise le comportement de la branche pour le régime sinusoïdal de pulsation ω.

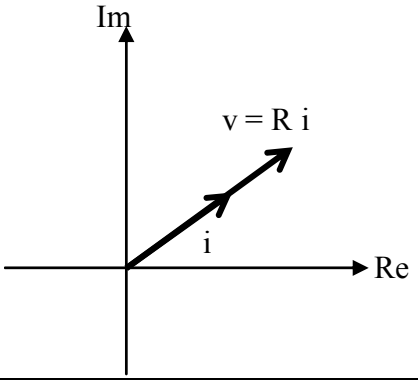
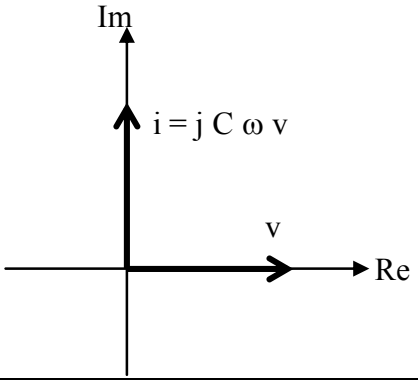
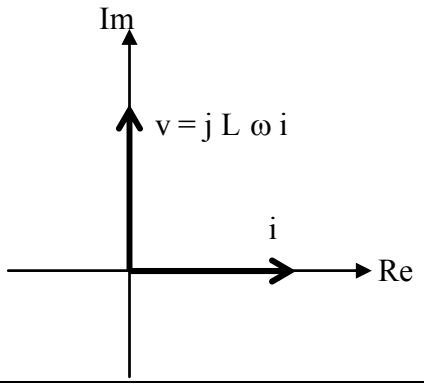
Quelques termes utiles

Impédance complexe : $Z = \underline{V}/\underline{I} = R + j\,X$

R est appelé réactance et X susceptance

Admittance complexe : $Y = \underline{I}/\underline{V} = 1 / Z$

Cas des dipôles passifs idéaux

Nature du dipôle	Impédance complexe	Représentation de Fresnel
Résistance R	R **L'intensité et la tension sont en phase**	
Condensateur C	$1 / j\,C\,\omega$ **L'intensité est en avance sur la tension de $\pi/2$**	
Inductance L	$j\,L\,\omega$ **La tension est en avance sur l'intensité de $\pi/2$**	

De façon formelle, lorsque l'on fait l'analyse d'un réseau linéaire avec des amplitudes complexes, tout se passe comme si les impédances complexes jouaient le rôle de résistances dans un réseau permanent. Toutes les lois élémentaires de calcul des circuits vues au 4) restent donc valides.

Fonction de transfert et diagramme de Bode

On considère un réseau linéaire comme une « boîte noire », à l'entrée et à la sortie de laquelle on mesure respectivement les tensions ve et vs, dont on prend les représentations complexes $\underline{ve}$ et $\underline{vs}$. On définit la **fonction de transfert** en régime harmonique du système, notée $H(j\omega)$, par : $H(j\omega) = \underline{vs} / \underline{ve}$.

La fonction de transfert est une caractéristique du système, dont la valeur ne dépend que de la fréquence du signal d'entrée.

Remarque : Il est également possible de définir une fonction de transfert comme le rapport de la tension de sortie et du courant d'entrée par exemple, auquel cas cette grandeur a la dimension d'une résistance, mais le plus souvent il s'agit du rapport de deux tensions, quantité sans dimension.

La fonction de transfert contient deux informations : l'évolution de l'amplitude du signal entre l'entrée et la sortie est donnée par le module de H ; l'évolution de la phase du signal est donnée par l'argument de H. Le **diagramme de Bode** permet de représenter graphiquement ces deux informations en fonction de la pulsation.

Un diagramme de Bode comporte donc deux graphiques. L'un représentant le gain ($|H|$) en fonction de la pulsation et le second représentant la phase (Arg(H)) en fonction de la pulsation. Compte tenu des grandes plages d'excursion possible pour les pulsations et pour les gains dans les circuits électroniques, on choisit une représentation logarithmique pour ces deux grandeurs. La pulsation est représentée en abscisse par une échelle logarithmique tandis que le module du gain est représenté en ordonnée en décibels : on ne trace pas $|H|$ mais $|H|_{dB} = 20 \log (|H|)$.

Cela a de plus une conséquence pratique très utile : si on exprime la fonction de transfert sous la forme d'un produit de fonctions élémentaires alors son diagramme de Bode (en gain et en phase) est la somme des diagrammes de Bode de chaque fonction élémentaire.

Si $H = H1 \cdot H2 \cdot H3$

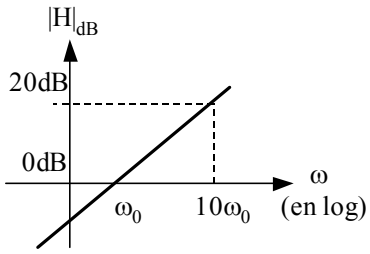
alors $|H|_{dB} = |H1|_{dB} + |H2|_{dB} + |H3|_{dB}$

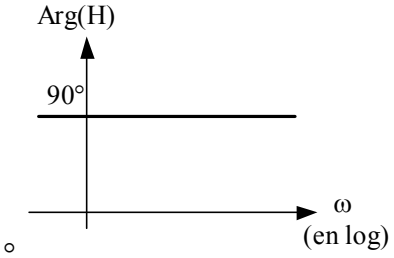
et $\text{Arg}(H) = \text{Arg}(H1) + \text{Arg}(H2) + \text{Arg}(H3)$

Dans la pratique, on commence par tracer le diagramme **asymptotique** de Bode en étudiant ce qui se passe aux basses fréquences puis aux hautes fréquences. Notez que, dans le cas général, la courbe réelle s'éloigne des asymptotes, notamment au niveau des fréquences de coupures.

Quelques exemples à retenir

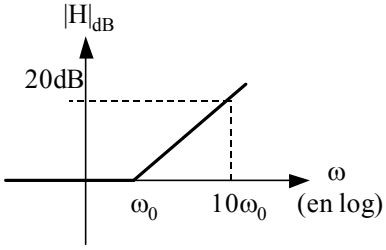
$H(j\omega) = j\omega / \omega_0$ $\quad |H| = \omega / \omega_0$ $\quad Arg(H) = 90°$

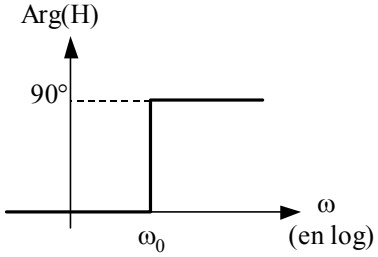




Le diagramme asymptotique et le diagramme réel sont confondus

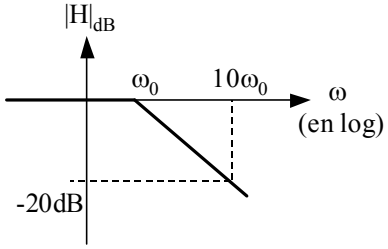
$H(j\omega) = 1 + j\omega / \omega_0$ $\quad |H| = (1 + \omega^2/\omega_0^2)^{1/2}$ $\quad Arg(H) = \arctan(\omega/\omega_0)$

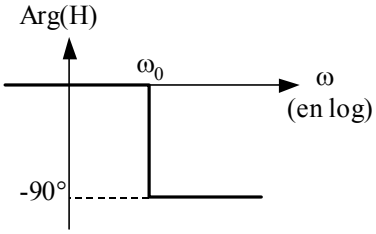




Le diagramme réel passe par : $|H(j\omega_0)| = 3$ dB et $Arg(H(j\omega_0)) = 45°$.

$H(j\omega) = 1/(1 + j\omega / \omega_0)$ $\quad |H| = (1 + \omega^2/\omega_0^2)^{-1/2}$ $\quad Arg(H) = -\arctan(\omega/\omega_0)$





Le diagramme réel passe par : $|H(j\omega_0)| = -3$ dB et $Arg(H(j\omega_0)) = -45°$.

Remarque importante : compte tenu de l'échelle logarithmique pour les pulsations, le point ω=0 (c'est à dire le régime continu) n'est pas représenté. Sa valeur (en gain ou en phase) constituera une **asymptote** de la courbe réelle.

Quelques valeurs de décibels à retenir

	signal atténué				signal amplifié		
$\|H\|$	0.1	0.5	$1/\sqrt{2}$	1	$\sqrt{2}$	2	10
$\|H\|_{dB}$	-20 dB	-6 dB	-3 dB	0 dB	3 dB	6 dB	20 dB

6) Dipôles passifs réels

Notion de composants réels

Un composant réel ne peut jamais se modéliser complètement par un seul dipôle R, C ou L idéal. On est donc amené à définir des **modèles** constitués de plusieurs dipôles parfaits, connectés en série ou en parallèle. Sous certaines conditions expérimentales, on pourra simplifier ces modèles et considérer le composant réel comme un dipôle idéal. Le comportement d'un composant réel peut dépendre de plusieurs grandeurs telles que :

- la fréquence du signal appliqué,
- la température de fonctionnement,
- la tension à ses bornes ou le courant le traversant.

De plus, chaque composant présente une dissipation maximale de puissance qu'il ne faut en aucun cas dépasser sous peine d'endommager le composant.

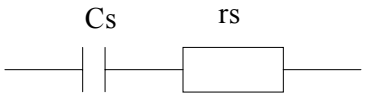
La résistance est le composant le plus parfait : on peut généralement le modéliser par une résistance idéale. L'influence de la fréquence est en effet négligeable pour les fréquences inférieures au mégaHertz. L'influence de la température sur la valeur de la résistance est relativement faible : les résistances les plus stables présentent des variations de 100ppm/°C. La puissance maximale pour les résistances standard est de l'ordre du Watt. Elle peut monter jusqu'à 1000W pour les résistances dite « de puissance ».

Le condensateur présente peu de défauts hormis une résistance « parasite ». En régime continu, il s'agit d'une résistance de fuite qui est généralement très supérieure au mégaOhm. En régime alternatif, le condensateur réel se modélise par un condensateur parfait en parallèle ou en série avec une résistance. On parle de **modèle série** ou de **modèle parallèle**.

La self est de loin le composant passif qui présente le plus de défauts. En régime continu, elle se comporte quasiment comme une self idéale mais elle est très peu utilisée à cause de son encombrement. Aux fréquences moyennes, elle se modélise par un modèle RL série ou parallèle faisant intervenir une résistance parasite. Pour les fréquences élevées il faut en plus tenir compte des capacités parasites entre spires : la self se modélise alors par un circuit RLC.

Le facteur de qualité d'un condensateur et d'une self en régime alternatif

En régime alternatif, le condensateur peut être modélisé par un circuit RC série tel que représenté ci-dessous.



L'impédance équivalente est donc $Zs = rs + 1 / j \omega Cs$. Les défauts du condensateur sont ici modélisés par rs. Pour estimer leur importance, il faut comparer la valeur de rs à $(1 / Cs \omega)$. Il est alors commode de faire apparaitre une autre grandeur, appelé facteur de qualité, en factorisant par rs dans l'expression de Zs : $Zs = rs (1 – j Q)$.

Le **facteur de qualité d'un condensateur** est défini par $\mathbf{Q = 1 / rs\ Cs\ \omega}$. Graphiquement, dans le plan complexe, le facteur de qualité est représenté par la tangente de l'angle ϕ (figure 18) : $Q = \tan \phi$. Pour un condensateur parfait rs est nulle donc ϕ vaut 90° et Q est infini.

On utilise également, pour caractériser les défauts du condensateur, l'angle de perte δ défini par : $\mathbf{\tan \delta = 1 / Q = rs\ Cs\ \omega}$. Pour un condensateur parfait, nous avons vu précédemment

que l'intensité est en avance de $\pi/2$ sur la tension. Dans le cas d'un condensateur réel, **l'intensité est en avance de $(\pi/2)-\delta)$ sur la tension**.

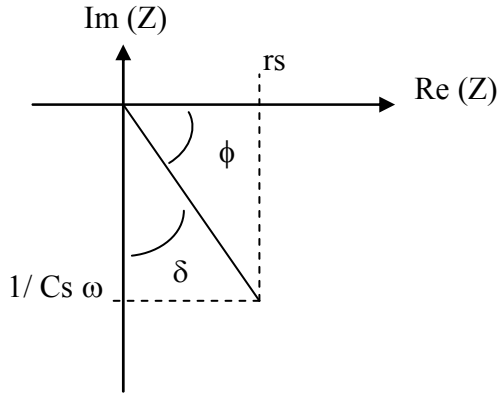


Figure 18 : Représentation de Zs dans le plan complexe.

Remarque : attention, le facteur de qualité ainsi que l'angle de perte dépendent de la fréquence. Q n'est donc pas inversement proportionnel à ω et tan δ n'est pas proportionnel à ω.

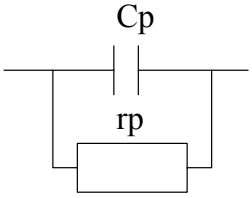
Exemple numérique : Pour un condensateur polyester de 0.1µF le constructeur donne les valeurs suivantes.

fréquence	100 Hz	10 kHz
rs	38 Ω	1.25 Ω
tan δ	2×10^{-3}	8×10^{-3}
Q	500	125

Sur cet exemple on remarque que rs est bien plus élevé à basse fréquence qu 'à haute fréquence. Pourtant ce condensateur se comportera plus comme un condensateur idéal à basse fréquence car son facteur de qualité est plus grand.

De même, la self en régime alternatif peut être modélisée par un circuit RL série.
Par un raisonnement analogue, on défini le **facteur de qualité d'une self** par **$Q = Ls\,\omega / rs$**.

<u>Equivalence modèle série// modèle parallèle</u>
Le condensateur réel peut également être modélisé par un circuit RC parallèle. On parle alors du modèle parallèle du condensateur.



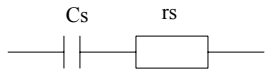
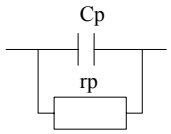
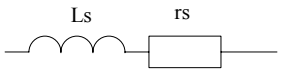
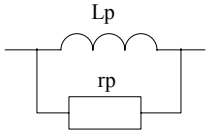
L'impédance complexe du circuit est alors : $Zp = rp / (1 + j\, rp\, Cp\, \omega) = rp / (1 + j\, Qp)$
$Q = rp\, Cp\, \omega$ est le facteur de qualité du condensateur.
On montre par identification avec le modèle série que :

$$\mathbf{rp = rs\,(1+Q^2)}$$
$$\mathbf{Cp = Cs\,Q^2 / (1+Q^2)}$$

Remarque : le facteur de qualité caractérise le condensateur réel indépendamment du modèle choisi. En effet, le facteur de qualité du modèle parallèle est bien identique à celui du modèle série : $Q = rp\ Cp\ \omega = 1 / rs\ Cs\ \omega$.

Ces équations de **transformation série/parallèle** se simplifie pour Q>>1 (i.e. typiquement Q>10). On obtient alors : $rp \approx rs\ Q^2$ et $Cp \approx Cs$.

Le tableau ci-dessous récapitule les différents modèles et les lois de transformation pour le condensateur et pour la self.

Modèle	Impédance	Facteur de qualité	Transformation
	$rs + 1 / j\ Cs\ \omega$	$Q = 1 / rs\ Cs\ \omega$	Pour Q > 10 : $rp \approx rs\ Q^2$ $Cp \approx Cs$
	$rp / (1 + j\ rp\ Cp\ \omega)$	$Q = rp\ Cp\ \omega$	
	$rs + j\ Ls\ \omega$	$Q = Ls\ \omega / rs$	Pour Q > 10 : $rp \approx rs\ Q^2$ $Lp \approx Ls$
	$j\ rp\ Lp\ \omega / (rp + j\ Lp\ \omega)$	$Q = rp / Lp\ \omega$	

7) Le système international d'unités (SI)

(source : Bureau International des Poids et Mesures http://www.bipm.fr/)

Le système international d'unités, appelé encore SI est un système cohérent d'unités adopté par la 11ème conférence générale des poids et mesures (CGPM) en 1960. Le SI comprend :

- des unités de base,
- des unités dérivées, y compris les unités supplémentaires.

L'utilisation de ce système est obligatoire en France.

Le SI comporte sept unités de base correspondant aux sept grandeurs de base :

Grandeur	Nom de l'unité de base	Symbole de l'unité
Longueur	Mètre	m
Masse	Kilogramme	Kg
Temps	Seconde	s
Intensité du courant	Ampère	A
Température thermodynamique	Kelvin	K
Quantité de matière	Mole	mol
Intensité lumineuse	Candela	cd

Quelques unités dérivées couramment utilisées en électronique sont données ci-dessous :

Grandeurs	**Dénomination**	**Symbole**	**Expression**
Fréquence	Hertz	Hz	1Hz = 1 s^{-1}
Energie	Joule	J	1J = 1 N.m
Puissance	Watt	W	1W = 1 J/s
Charge électrique	Coulomb	C	1C = 1 A.s
Potentiel électrique	Volt	V	1V = 1 J/C
Capacité électrique	Farad	F	1F = 1 C/V
Résistance électrique	Ohm	Ω	1Ω = 1 V/A
Conductance	Siemens	S	1S = 1 Ω^{-1}
Flux magnétique	Weber	Wb	1Wb = 1V.s
Inductance	Henry	H	1H = 1 Wb/A
Force	Newton	N	1N = 1kg.m/s-2

Le tableau ci-dessous rappelle les principaux multiples et sous-multiples des unités SI.

Symbole	**Préfixe**	**Facteur**
P	Peta	10^{15}
T	Téra	10^{12}
G	giga	10^{9}
M	méga	10^{6}
k	kilo	10^{3}
--	--	1
m	milli	10^{-3}
μ	micro	10^{-6}
n	nano	10^{-9}
p	pico	10^{-12}
f	femto	10^{-15}
A	atto	10^{-18}

Quelques constantes physiques :

Charge électronique élémentaire : $e = 1{,}6.10^{-19}$ C
Constante de BOLTZMANN : $k = 1{,}38.10^{-23}$ $J.K^{-1}$
Permittivité du vide : $\varepsilon 0 = 8{,}85.10^{-12}$ $F.m^{-1}$
Perméabilité du vide : $\mu = 4.\pi.10^{-7}$ $H.m^{-1}$

PARTIE B : Electronique logique

1) L'information logique : le bit

Généralités sur les signaux numériques

Un signal numérique est une grandeur qui n'a que 2 valeurs significatives. On peut représenter ce signal numérique par une variable binaire, encore appelée bit (contraction de « binary digit »).

De nombreux appareils, dits « numériques », utilisent des signaux numériques : téléphone, appareils photo, caméra, ordinateurs, automates, lecteurs de CD audio, CD rom, DVD ...

Ces appareils diffèrent des appareils dits analogiques par la nature même des informations qu'ils utilisent ou véhiculent.

Le « monde » analogique est directement accessible à l'homme par ces différents sens (vue, ouïe, ...). Il est caractérisé par des grandeurs variant de manière continue dans le temps et pouvant prendre une infinité de valeurs.

Le monde numérique n'est pas directement accessible : une suite de valeurs binaires ne représente a priori rien pour l'homme. Il est caractérisé par des grandeurs qui sont des fonctions discontinues du temps et qui ne peuvent prendre qu'un nombre fini de valeurs.

Le passage d'un monde à l'autre se fait à l'aide de convertisseur analogique-numérique et numérique-analogique (voir cours d'électronique). Les deux étapes essentielles qui permettent de passer d'un signal analogique à un signal numérique sont illustrées sur la figure 1.
Il s'agit de l'échantillonnage (discrétisation du temps) et de la numérisation, ou quantification (discrétisation de l'amplitude).

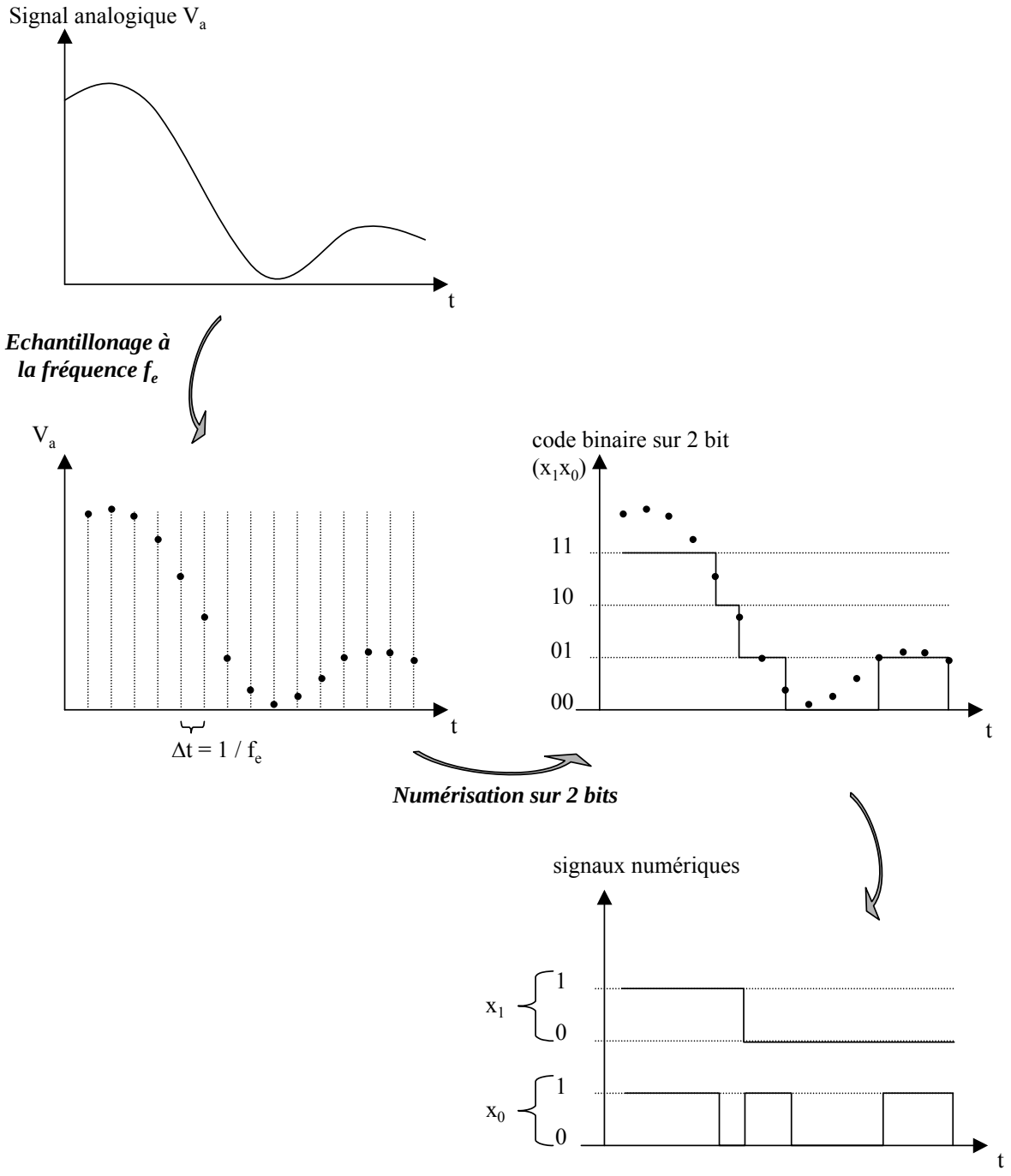


Figure 1 - Passage d'un signal analogique à un signal numérique

Bit et signaux électriques

Dans les circuits numériques, l'information logique est véhiculée par un signal électrique, le plus souvent une tension. La valeur du signal numérique se mesure donc en volt. Par convention les deux états permis pour ce signal sont appelés H (pour High) et L (pour Low).

H représente l'état haut, c'est à dire celui qui a la tension la plus élevée en valeur algébrique.

L représente l'état bas, c'est à dire celui qui a la tension la plus basse en valeur algébrique.

Par exemple, à l'entrée d'un circuit logique de technologie TTL, une tension de 3V sera considérée comme un état H et une tension de 0V comme un état L.

Attention, dans certains cas une tension de 0V peut représenter un état H. Par exemple en technologie ECL, 0V est considéré comme un état H et -1.6V comme un état bas.

Les correspondances entre les états H et L et les valeurs de tension suivant les technologies de circuits utilisés seront détaillées plus tard dans le cours.

Conventions logiques

Dans un circuit numérique, la valeur (H ou L) d'une entrée ou d'une sortie peut avoir diverses significations :

- un chiffre en base 2 : 0 ou 1
- la valeur d'une variable logique : Faux ou Vrai
- l'état d'un opérateur : inactif ou actif
- l'état d'un moteur : arrêt ou marche
- etc.

Une convention logique permet d'associer les valeurs H et L du signal numérique aux valeurs de ce qu'il représente.

Attention, il n'y a pas de convention par défaut. Traditionnellement, on qualifie de **convention logique positive** l'association de H à 1, Vrai ou actif et **convention logique négative** l'association de H à 0, Faux ou inactif.

convention logique positive	
L	H
0	1
Faux	Vrai
inactif	actif

convention logique négative	
L	H
1	0
Vrai	Faux
actif	inactif

Avantages et limitations du numérique

L'un des intérêt majeurs des signaux numériques est leur grande robustesse vis à vis des perturbations extérieures. Cette immunité au bruit provient de deux mécanismes qui se complètent pour rendre le système numérique plus robuste.

- Une protection au niveau du signal élémentaire (le bit) qui repose sur le fait que l'amplitude d'un signal numérique n'est pas l'image de l'information à transmettre. La seule contrainte est que le système soit capable de différencier sans ambiguïté un niveau H d'un niveau L. L'écart entre ces deux niveaux étant grand, seule une perturbation de grande amplitude (de l'ordre de 0.4V pour un circuit TTL) pourra provoquer une erreur.
- Une protection au niveau du système, par le jeu du codage (codes détecteurs et correcteurs d'erreurs).

La principale limitation des systèmes numériques provient de la très faible quantité d'information véhiculée par un signal élémentaire (0 ou 1). Il est donc nécessaire d'utiliser un grand nombre de signaux élémentaires pour coder une information intéressante (image, musique, données alphanumériques, ...).
D'un point de vue pratique, les systèmes numériques nécessitent un débit important pour la transmission d'une part et un volume et/ou une densité de stockage élevés d'autre part.

2) Le codage de l'information : du bit au mot

On appelle mot un ensemble de bit. Pour donner à un mot une signification, on utilise un code déterminé. Il existe bien entendu un nombre important de codes différents selon la nature de l'information que l'on désirer codée. Nous nous contenterons dans ce cours de décrire les codes les plus utilisés.

On considère ici un mot de n bits : $(a_{n-1}, ..., a_1, a_0)$, tous les a_i appartenant à l'ensemble {0 ;1}.
a_{n-1} est le bit de poids fort : **MSB** en anglais pour Most Significant Bit.
a_0 est le bit de poids faible : **LSB** en anglais pour Less Significant Bit.

Entiers naturels (entiers non signés)

Code binaire naturel

Dans le code binaire naturel, un mot de n bits représente un entier A écrit en base 2 :

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

Par exemple, pour n = 4 :
0000 représente l'entier 0
0001 représente l'entier 1
0010 représente l'entier 2
0011 représente l'entier 3
...
1111 représente l'entier 15

Comme on le voit sur cet exemple, le codage en binaire naturel ne donne pas accès à l'ensemble des entiers naturels mais à un sous ensemble fini de cet ensemble. En effet, pour un mot de n bits, on a : $\mathbf{0 \leq A \leq 2^n - 1}$.

On donne dans le tableau suivant les valeurs de la borne supérieure de A pour les types de mots les plus courants (la borne inférieure de A est toujours 0):

type de mots	nombre de bits (n)	borne supérieure de A (2^n-1)
octet	8	255
entier court	16	65 535 (= 64 K*)
entier long	32	4 294 967 295 (= 4096 M)

* Notez que dans le monde du numérique (et de l'informatique) 1 K = 2^{10} = 1024 et 1 M = 1024 K. Cet usage perdure malgré la norme internationale proposée par l'IEC en 2000 (voir annexe).

Notation hexadécimale

Lorsque n est un multiple de 4, on peut écrire A sous forme de nombre hexadécimal (c'est à dire en base 16).

Le tableau suivant donne la correspondance entre le code binaire sur 4 bits, le code hexadécimal et la valeur décimale :

code binaire naturel	code hexadécimal	valeur décimale
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

Pour convertir un mot binaire de plus de 4 bits, il suffit de faire des blocs de 4 bits en partant du LSB puis de convertir bloc par bloc en utilisant le tableau ci-dessus; et réciproquement pour convertir de l'hexadécimal en binaire.

Dans certains cas, il est préférable d'utiliser un code différent du code binaire naturel.

Code binaire réfléchi (ou code Gray)

Il existe des systèmes, où l'on a avantage à ce que d'une valeur à l'autre, il n'y ait qu'un seul bit qui varie. Ce n'est pas le cas du binaire, où pour passer de 1 à 2 par exemple, deux bits changent. Si un capteur produit une information codée, les transitions ne sont pas simultanées et on peut lire :

1 (001) → 3 (011) → 2 (010)
ou bien:
1 (001) → 0 (000) → 2 (010).

Le code binaire réfléchi est construit de manière à ce que le passage d'une valeur à la valeur suivante s'effectue en modifiant un seul bit. Ce code se construit de manière symétrique par rapport à des axes Y et cyclique par rapport à des axes X (voir exemple ci-après).
Nous le rencontrerons notamment dans les méthodes de simplification des fonctions logiques.

Exemple : Code binaire réfléchi pour un mot de 4 bits.

valeur décimale	code binaire réfléchi	
0	0000	
1	0001	
2	0011	
3	0010	Y1
4	0110	
5	0111	
6	0101	
7	0100	X
8	1100	
9	1101	
10	1111	
11	1110	Y2
12	1010	
13	1011	
14	1001	
15	1000	

Code BCD

Le code décimal codé binaire (Binary-Coded Decimal) consiste à coder en binaire chaque digit du code décimal. Par exemple, pour coder le nombre 529 :
529 = 5*100 + 2*10 + 9 (décimal) = 0101 0010 1001 (BCD).
Ce code est pratique pour afficher en décimal des nombres. Par contre les calculs arithmétiques (addition, soustraction, ...) sont bien plus complexe qu'en code binaire naturel. Il est essentiellement utilisé pour les applications ou l'affichage des chiffres joue un rôle prépondérant.

code binaire naturel	code BCD	valeur décimale
0000	0 0000	0
0001	0 0001	1
0010	0 0010	2
0011	0 0011	3
0100	0 0100	4
0101	0 0101	5
0110	0 0110	6
0111	0 0111	7
1000	0 1000	8
1001	0 1001	9
1010	1 0000	10
1011	1 0001	11
1100	1 0010	12
1101	1 0011	13
1110	1 0100	14
1111	1 0101	15

Entiers relatifs (entiers signés)

Remarque importante : un mot de n bits ne peut fournir que 2^n valeurs différentes.

Pour coder des entiers signés, il faudra donc restreindre la plage des valeurs accessibles pour la valeur absolue du nombre.
Pour un entier relatif A codé sur n bits, on a : $\mathbf{-2^{n-1} \leq A \leq 2^{n-1}-1}$.

Voici quelques valeurs pour les mots les plus courants :

type de mots	nombre de bits (n)	valeurs de A
octet	8	-128 à +127
entier relatif court	16	-32 768 à +32 767
entier relatif long	32	-2 147 483 648 à +2 147 483 647

Une manière simple de coder un entier signé est d'utiliser le bit de poids fort (MSB) comme bit de signe : un MSB à 0 désignera un nombre positif et un MSB à 1 un nombre négatif. Les n-1 bits restants servent alors à coder la valeur absolue du nombre. On peut, par analogie à notre système décimale, coder la valeur absolue du nombre en code binaire naturel. On aurait alors :

0010 = +2
0001 = +1
0000 = 0
1001 = -1
1010 = -2
...

Ce code « signe-valeur absolue » engendre de telles complications au niveau de l'arithmétique qu'il n'est jamais utilisé pour les calculs sur des entiers relatifs.

On utilise le plus souvent le code complément à 2, et parfois le code binaire décalé. Voyons comment sont construits ces deux codes.

Code complément à 2

C'est le code utilisé pour représenter les nombres entiers signés dans un ordinateur. Il présente l'intérêt majeur de se prêter à une arithmétique simple.
La construction du code complément à deux sur n bits découle de la définition modulo 2^n des nombres. Etant donné un nombre A :

- **Si $A \geq 0$, le code de A est l'écriture en binaire naturel de A, éventuellement complétée à gauche par des 0.**
- **Si $A < 0$, le code de A est l'écriture en binaire naturel de 2^n+A, c'est à dire de $2^n-|A|$.**

Par exemple +3 codé sur 8 bits s'écrit : 00000011 et -3 s'écrit : 11111101 qui est la représentation en binaire naturel de $2^8 - 3 = 253$.

Remarques :

- Avec ce code complément à 2, le bit de poids fort (MSB) est aussi le bit de signe, avec la convention suivante :
  MSB à 0 = nombre positif,
  MSB à 1 = nombre négatif.

- Pour écrire en code complément à 2 un nombre négatif -A (A>0), il suffit d'écrire en binaire naturel A, de prendre son complément à 1 (obtenu en remplaçant les 1 par des 0 et vice-versa) et de lui ajouter 1.
  En effet, d'après la définition le code complément à 2 de -A est égal au code binaire naturel de : $2^n - A = 2^n - 1 - A + 1$
  $= \overline{A} + 1$ où $\overline{A}$ désigne le complément à 1 de A ($\overline{A} = 2^n - 1 - A$).

  On peut donc écrire : $\boxed{-A = \overline{A} + 1}$.

- L'addition de deux nombres codés en compléments à 2 est l'addition binaire classique. Elle est indépendante du signe de ces nombre. Prenons deux exemples sur 4 bits :
  (-3) + 2 = ?

```
  1101
  0010
  ------
  1111 --> -1
```

  3 + (-2) = ?

```
  0011
  1110
  ------
  0001 --> 1
```

Comme nous le montre ces exemples simples, le code compléments à 2 se prête bien aux calculs. Par contre il complique les opérations de comparaison car la relation d'ordre entre les nombres binaires n'est pas la même qu'en code binaire naturel. La figure 2 illustre cette rupture de la relation d'ordre entre les deux codes.

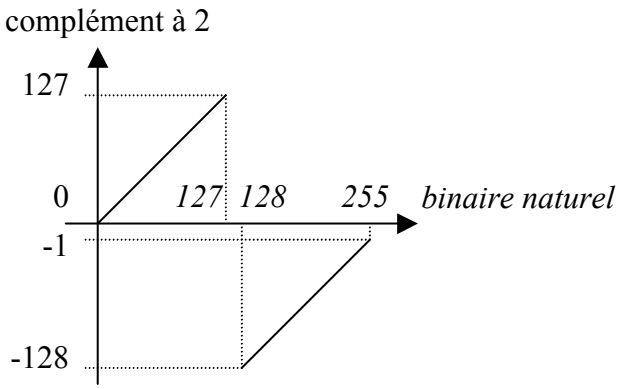


Figure 2 - Relation entre les codes binaire naturel et complément à 2 sur 8 bits

Pour pallier cet inconvénient on peut être amené à utiliser le code binaire décalé.

Code binaire décalé

Ce code possède la même relation d'ordre que le code binaire naturel (voir figure 3). On l'obtient à partir du code binaire naturel par simple décalage de l'origine.

Soit un nombre A tel que $-2^{n-1} \leq A \leq 2^{n-1}-1$:

le code binaire décalé de A est le code binaire naturel de $A + 2^{n-1}$.

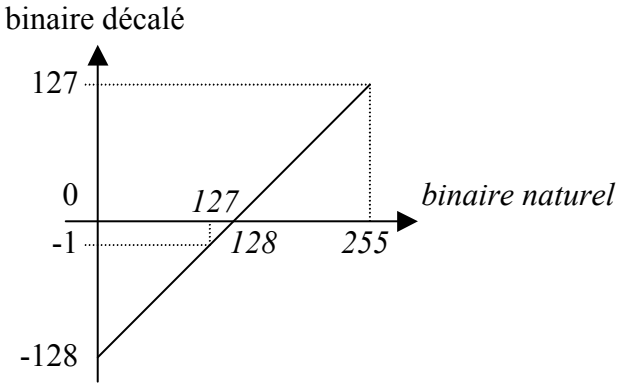


Figure 3 - Relation entre les codes binaire naturel et binaire décalé sur 8 bits

On rencontre ce type de code dans certains convertisseurs numériques-analogiques et dans la représentation de l'exposant des nombres flottants (dans ce cas le décalage est de $\mathbf{2^{n-1}-1}$, voir description au paragraphe suivant).

Dans le tableau suivant, nous comparons les codes complément à 2 et binaire décalé sur 4 bits. On passe du code binaire décalé au code complément à 2 en complémentant le bit de signe (MSB).

code binaire décalé	code complément à 2	valeur décimale
0000	1000	-8
0001	1001	-7
0010	1010	-6
0011	1011	-5
0100	1100	-4
0101	1101	-3
0110	1110	-2
0111	1111	-1
1000	0000	0
1001	0001	1
1010	0010	2
1011	0011	3
1100	0100	4
1101	0101	5
1110	0110	6
1111	0111	7

Nombres réels

Virgule fixe et virgule flottante

Le codage des nombres réels à virgule fixe (nombre de digits après la virgule fixe) s'apparente au codage des entiers (un entier peut être représentatif d'un nombre fractionnaire si l'on connaît la place de la virgule). Nous ne détaillerons pas ce type de codage dans ce cours.

Un nombre flottant, c'est à dire dont le nombre de digits après la virgule n'est pas fixé à priori, peut être écrit sous la forme suivante :

$$A = (-1)^S . b^E . M,$$

où S représente le signe de A avec la convention S=0 pour + et S=1 pour -,
b est la base de numération,
E est l'exposant,
M est la mantisse, qui s'écrit pour $A \neq 0$: N,xxx...xx avec $N \neq 0$.

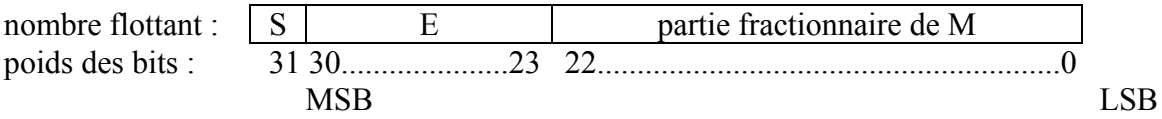
Le format général de ce codage est le suivant :

signe (S)	gestion de la virgule (E)	chiffres significatifs (M)

Pour le codage des flottants en binaire (en base 2), on code uniquement la partie fractionnaire de la mantisse : N est omis car il n'apporte aucune information (il vaut nécessairement 1 puisque $N \neq 0$ par définition). Il existe des formats normalisés dont les plus courants sont les flottants simple précision codés sur 32 bits et les flottants double précision codés sur 64 bits (norme IEEE 754-1985).

Flottants simple précision

Le format binaire des nombres flottants simple précision est le suivant :



S est le bit de signe : 0 pour + et 1 pour -.

L'exposant E est un entier signé compris entre –126 (-2^7+2) et +127 (2^7-1). Il est codé sur 8 bits en binaire décalé de 2^7-1: **on code E+127 en binaire naturel**. Les deux valeurs extrêmes de l'exposant (valeurs –127 et +128 correspondant aux codes 0 et 2^8-1) sont réservées (voir description ci-dessous).

La partie fractionnaire de la mantisse M est un entier non signé codé en binaire naturel sur 23 bits. M s'écrit donc : $1, m_{22} \ldots m_0$

Le réel A vaut : $(-1)^S (2^E + m_{22} 2^{E-1} + \ldots + m_0 2^{E-23})$

De plus, il existe des **combinaisons réservées** pour le zéro, l'infini (+ et -) et NAN (Not A Number) :

- pour zéro : E = -127 (codé 0) et M = 0,
- pour l'infini : E = 128 (codé 2^8-1) et M = 0,
- pour NAN : E = 128 (codé 2^8-1) et $M \neq 0$.

Flottants double précision

Les flottants double précision sont codés suivant un format similaire à celui des flottants simple précision mais avec l'exposant codé sur 11 bits (valeurs comprises entre –1022 et +1023, 0 et +1024 étant réservées comme ci-dessus) et la partie fractionnaire de la mantisse codée sur 52 bits.

Remarques

- Ce mode de codage permet la représentation de nombres très grands et/ou très petits. Par contre la résolution est fonction de l'exposant les nombres sont donc représentés avec une précision relative constante.
- L'arithmétique des nombres flottants et celle des entiers font appel à des algorithmes radicalement différents. Les entiers obéissent à une arithmétique euclidienne clairement définie alors que les flottants obéissent à une arithmétique approchée.
- Le test d'égalité de deux nombres, qui a un sens clair pour des entiers, fournit un résultat aléatoire dans le cadre des flottants. Seule une majoration de l'écart entre ces deux nombres conduit à un résultat prévisible.

Caractères : les codes ASCII

Nous avons vu jusqu'à présent uniquement des codes permettant de représenter des nombres. Lorsque l'on désire échanger des informations sous forme de texte, il est nécessaire de disposer de **codes alphanumériques**.

Code ASCII (sur 7 bits)

Il existe un standard de code alphanumérique, quasi universellement adopté : il s'agit du code ASCII pour « American Standard Code for Information Interchange ».

Ce code permet de restituer sous forme binaire sur 7 bits, soit 128 valeurs possibles :

- les caractères de l'alphabet romain (minuscules et majuscules) sans les lettres accentuées (voir plus loin),
- les 10 chiffres décimaux,
- les caractères de ponctuations, les parenthèses, les crochets et les accolades,
- les symboles arithmétiques les plus courants,
- des commandes et des caractères spéciaux.

Les codes de valeurs inférieures à 32 en décimal (c'est à dire 20 en hexadécimal) sont les commandes et caractères spéciaux. Comme commandes on peut citer, à titre d'exemple, LF (nouvelle ligne, « line feed ») codée par 10 en décimal ou CR (retour chariot, « carriage return ») codé par 13 en décimal. Comme caractères spéciaux, on peut citer SP (caractère espace, « blank space character ») codé par 32 en décimal ou FS (séparateur de fichier, « file separator »).

Les codes des valeurs supérieures à 32 en décimal sont donnés dans le tableau suivant. Dans chaque bloc, la colonne de gauche représente la valeur du code en décimal et celle de droite le caractère correspondant.

Table des codes décimaux des caractères ASCII

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

Code ASCII étendu (sur 8 bits ou plus)

L'ajout d'un huitième bits au code ASCII standard permet d'intégrer 128 caractères supplémentaires. Ces caractères supplémentaires peuvent être utilisé pour représenter les caractères non Anglo-saxon (notamment les caractères accentués), des symboles graphiques ou encore des symboles mathématiques.

De nombreux codes ASCII étendus différents ont été proposé par des sociétés ou des organisations. Le plus répandu dans le monde de l'informatique est sans doute le **code ISO Latin 1**.

Citons également le code **UNICODE** développé dans les années 1990 qui est un code sur 16 bits, intégrant le code ASCII standard, et contenant les alphabets de nombreux autres pays.

Codes correcteurs d'erreur

L'objectif de ces codes est de détecter et/ou de corriger les éventuelles erreurs qui peuvent survenir lors d'une transmission (minitel, par exemple) ou lors de la lecture de données stockées (sur un CD par exemple). Cela est possible en ajoutant à chaque mot à transmettre un ou plusieurs bits codés suivant un algorithme particulier. Prenons un exemple simple, utilisé pour les transmissions, dans lequel un seul bit est ajouté à chaque mot à transmettre. Ce bit supplémentaire (généralement placé à la gauche du MSB) est calculé par l'émetteur de telle façon que pour chaque mot transmis le nombre de bit à 1 soit pair (respectivement impair). On parle alors de parité paire (respectivement impaire). Si émetteur et récepteur utilise la même convention de parité, le récepteur est capable de détecter une faute de transmission tant qu'il n'y a pas plus d'une erreur par mot.

Pour pouvoir en plus corriger l'erreur détecter il faut utiliser des codes plus complexes que nous ne détaillerons pas ici.

3) Algèbre de Boole et portes logiques

Quelques définitions

L'algèbre **logique** est l'art de construire un raisonnement au moyen de propositions qui sont soit vraies soit fausses.
L'algèbre de Boole (Boole est un mathématicien Anglais du 19ème siècle) est un algèbre binaire à la base des systèmes logiques. Il permet de donner un caractère **algébrique** aux relations logiques par attribution de valeurs numériques aux propositions :

à une proposition vraie on associe la valeur binaire 1,
à une proposition fausse on associe la valeur binaire 0.

$$VRAI \equiv 1$$
$$FAUX \equiv 0$$

On appelle **variable logique** une grandeur x qui ne peut prendre que deux valeurs (0 ou 1) :

si $x \neq 0$ alors $x = 1$,
si $x \neq 1$ alors $x = 0$.

On appelle **fonction logique** un groupe de variables logiques liées entre elles par des **opérateurs logiques**. Une fonction ne peut prendre que deux valeurs (0 et 1).

Une **table de vérité** est un tableau donnant les valeurs d'une fonction pour toutes les combinaisons des variables qui la composent.

Les opérations logiques élémentaires

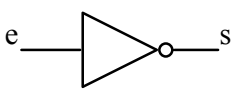
Il existe trois opérateurs logiques élémentaires. Ils sont génériques : toutes les fonctions logiques peuvent être écrites à partir de ces trois opérateurs élémentaires.

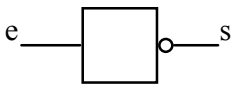
L'opérateur PAS[3] (NOT)

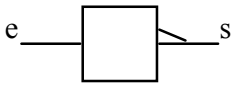
Table de vérité

e	s
0	1
1	0

Symboles







Notations

$s = \overline{e}$

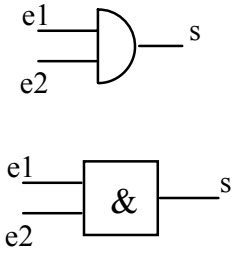
[3] Aussi appelé opérateur INVERSION

L'opérateur ET (AND)

Table de vérité

e1	e2	s
0	0	0
0	1	0
1	0	0
1	1	1

Symboles



Notations

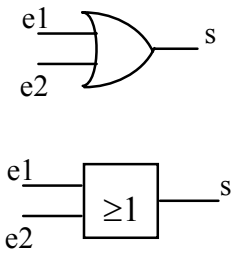
$s = e1 \,.\, e2$
$s = e1 \,\&\, e2$
$s = e1 \wedge e2$

L'opérateur OU (OR)

Table de vérité

e1	e2	s
0	0	0
0	1	1
1	0	1
1	1	1

Symboles



Notations

$s = e1 + e2$
$s = e1 \mid e2$
$s = e1 \vee e2$

Propriétés des opérateurs élémentaires

Dans ce paragraphe, A, B et C désignent des variables logiques ou des fonctions logiques.

Associativité et commutativité

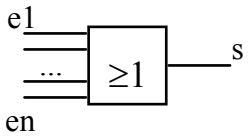
Les opérateurs ET et OU possèdent les propriétés d'associativité :

$$A + (B + C) = (A + B) + C$$
$$A \,.\, (B \,.\, C) = (A \,.\, B) \,.\, C$$

et de commutativité :

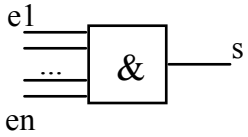
$$A + B = B + A$$
$$A \,.\, B = B \,.\, A$$

Une conséquence de ces deux propriétés est qu'une expression du type (e1+e2+e3+...+en) est définie de manière univoque. On peut donc généraliser l'opérateur OU pour n entrées ($n \geq 2$) :



$$s = e1+e2+e3+...+en$$

De même, on peut généraliser l'opérateur ET pour n entrées (n≥2) :



$$s = e1.e2.e3. \ldots .en$$

Double distributivité

Les opérateurs ET et OU sont mutuellement distributifs l'un par rapport à l'autre.

$$A . (B + C) = (A . B) + (A . C)$$
$$A + (B . C) = (A + B) . (A + C)$$

Attention, en algèbre binaire, la distributivité est valable dans les deux sens contrairement aux opérateurs arithmétiques SOMME et PRODUIT.

Ordre de priorité

Bien que leur propriétés soit parfaitement symétriques, on définit un ordre de priorité entre les opérateurs ET et OU :

$$A . B + A .C = (A . B) + (A . C)$$

l'opérateur ET est prioritaire devant l'opérateur OU.

Quelques relations à connaître

OU	ET
$A + 0 = A$ (0 élément neutre)	$A . 1 = A$ (1 élément neutre)
$A + 1 = 1$ (1 élément absorbant)	$A . 0 = 0$ (0 élément absorbant)
$A + A = A$	$A . A = A$
$A + \overline{A} = 1$	$A . \overline{A} = 0$

Théorèmes de DE MORGAN

$$\overline{A + B} = \overline{A} . \overline{B}$$
$$\overline{A . B} = \overline{A} + \overline{B}$$

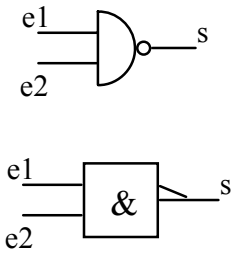
Les autres opérateurs logiques

NON ET (NAND)

Table de vérité

e1	e2	s
0	0	1
0	1	1
1	0	1
1	1	0

Symboles



Notations

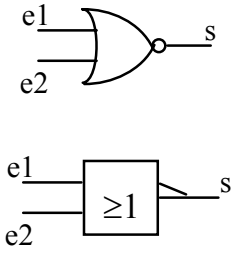
$s = \overline{e1 \cdot e2}$

NON OU (NOR)

Table de vérité

e1	e2	s
0	0	1
0	1	0
1	0	0
1	1	0

Symboles



Notations

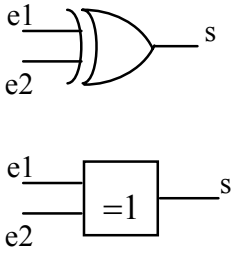
$s = \overline{e1 + e2}$

OU exclusif (XOR)

Table de vérité

e1	e2	s
0	0	0
0	1	1
1	0	1
1	1	0

Symboles



Notations

$s = e1 \oplus e2$

Les opérateurs NON ET et NON OU sont des opérateurs complets car l'une de ces fonctions permets de réaliser toutes les autres fonctions logiques.

L'opérateur OU exclusif est notamment utilisé pour l'addition (somme modulo 2), pour les tests de parité (contrôle d'erreurs) ou encore pour la détection d'égalité (comparaison).

4) Simplification des fonctions logiques

Définition et objectifs de la simplification

Comme nous l'avons évoqué précédemment, toute fonction logique peut s'écrire sous la forme d'une expression polynomiale qui fait intervenir les trois opérateurs élémentaires de l'algèbre de Boole : la somme logique (OU), le produit logique (ET) et la complémentation (PAS).
En général, une fonction est exprimée soit par une somme de produits logiques :
par exemple $F_1(x,y,z) = x.y + x.\overline{z} + \overline{x}.\overline{y}.z$,

soit par un produit de sommes logiques :
par exemple $F_2(x,y,z) = (x+y).(\overline{x}+z).(\overline{x}+y+\overline{z})$.

Une fonction est dite sous forme normale (ou canonique) si chaque terme (produit ou somme logique) contient toutes les variables sous un de leurs aspects (complémenté ou non). Il existe donc deux formes normales, simplement nommées première et deuxième formes normales.

La première forme normale correspond à l'écriture de la fonction comme somme de produits logiques. Par exemple :
$F_3(x,y,z) = x.y.z + \overline{x}.\overline{y}.z + x.\overline{y}.\overline{z}$.
Chaque terme est appelé un minterme.
Pour obtenir cette forme, il suffit d'énumérer les cas où la fonction prend la valeur 1. Dans chaque cas, on écrit le produit logique des variables complémentées ou non : on prend le complément lorsque la variable vaut 0.
Cette forme est la plus couramment utilisée. Une grande majorité de circuits programmables ont une architecture interne qui reproduit, en trois couches logiques (PAS - ET - OU), un développement en première forme normale.

La deuxième forme normale correspond à l'écriture de la fonction comme produit de sommes logiques. Par exemple :
$F_4(x,y,z) = (x+y+z).(\overline{x}+\overline{y}+z).(\overline{x}+y+\overline{z})$.
Chaque terme est appelé un maxterme.
Pour obtenir cette forme, il suffit d'énumérer les cas où la fonction prend la valeur 0. Dans chaque cas, on écrit la somme logique des variables complémentées ou non : on prend le complément lorsque la variable vaut 1.

Exemple
Regardons comment s'écrit la fonction OU exclusif sous ces deux formes normales :
la première forme normale donne : $x \oplus y = x.\overline{y} + \overline{x}.y$,

la deuxième forme normale donne : $x \oplus y = (x+y).(\overline{x}+\overline{y})$.

La simplification d'une fonction logique consiste à obtenir une expression de cette fonction contenant :

- le nombre minimal de terme
- le nombre minimal de variables dans chaque terme.

Certaines fonctions peuvent être exprimées par plusieurs expressions sous forme minimale différentes. Ces formes sont dites <u>équivalentes</u>.

Le but de la simplification est de réduire le nombre de portes élémentaires nécessaire à la synthèse de la fonction souhaitée. Elle permet :

- une réduction du coût du matériel,
- une augmentation de la fiabilité du système,
- une augmentation de la vitesse du système (réduction des temps de propagation),
- une augmentation de la fonctionnalité des composants dans le cas de composants logiques programmables.

En pratique, il faut disposer de méthodes efficaces qui permettent d'éliminer les redondances présentes dans la forme normale de la fonction considérée.

Nous décrirons dans ce cours trois types de méthodes qui tentent de résoudre ce problème, de la plus simple à la plus sophistiquée.

<u>Méthode algébrique</u>

La méthode algébrique consiste à utiliser les théorèmes et les règles de l'algèbre de Boole. Elle se fait par :

- regroupement des termes et mises en facteur,
- addition de termes déjà existants,
- suppression des termes superflus en recherchant les consensus (voir ci-après),
- passage d'une forme normale à l'autre en utilisant les théorèmes de De Morgan.

La simplification, ou minimisation, des fonctions logiques passe donc par la recherche de consensus. Ces derniers peuvent être classés en fonction de leur ordre.
Dans la suite x, y et z désignent des variables logiques et f_c une fonction logique.

Consensus d'ordre 0 : (f_c , f_c)

Les consensus d'ordre 0 sont des termes identiques. On peut les simplifier en appliquant les relations suivantes :
$f_c + f_c = f_c$
$f_c \,.\, f_c = f_c$

Règle n°1 : dans une somme (OU), ou un produit (ET), on peut supprimer un des deux termes identiques.

<u>Exemple</u> : Simplifions l'expression $f = x + y + x\,\overline{y}$

$f = x\,(y+\overline{y}) + y + x\,\overline{y}$ *ajout d'un terme = 1 dans un produit*

$f = x\,y + x\,\overline{y} + y + x\,\overline{y}$

$f = x\,y + x\,\overline{y} + y$ *simplification du consensus d'ordre 0*

$f = x\,(y+\overline{y}) + y$

$f = x + y$

Consensus d'ordre 1 : ($x\ f_c$, $\overline{x}\ f_c$)

Les consensus d'ordre 1 sont des termes où une seule variable change d'état. La simplification se fait suivant les relations suivantes :

$x\ f_c + \overline{x}\ f_c = f_c$

$x\ f_c\ .\ \overline{x}\ f_c = 0$

Règle n°2 : dans une somme (OU), si une seule variable change elle disparaît.

Règle n°3 : dans un produit (ET), si une seule (ou plusieurs) des variables change(nt), le terme peut être supprimé.

Consensus d'ordre 2 : ($x\ y\ f_c$, $\overline{x}\ \overline{y}\ f_c$) ou ($x\ \overline{y}\ f_c$, $\overline{x}\ y\ f_c$)

Les consensus d'ordre 2 sont des termes où deux variables changent d'état. La simplification se fait suivant les relations suivantes :

$x\ y\ f_c + \overline{x}\ \overline{y}\ f_c = f_c\ .\ (x\ y + \overline{x}\ \overline{y})$

$f_c\ .\ (\overline{x \oplus y})$

$x\ \overline{y}\ f_c + \overline{x}\ y\ f_c = f_c\ .\ (x\ \overline{y} + \overline{x}\ y)$

$f_c\ .\ (x \oplus y)$

Règle n°4 : dans une somme (OU), lorsque deux variables changent apparaît un ou exclusif (inverse).

Pour les consensus d'ordre 2 et supérieure dans les produits (ET), la simplification est la même que pour l'ordre 1 (voir la règle n°3).

Consensus d'ordre 3 et supérieur : ($x\ y\ z\ f_c$, $\overline{x}\ \overline{y}\ \overline{z}\ f_c$) ...

Par exemple : $x\,\overline{y}\,z\ f_c + \overline{x}\,y\,\overline{z}\ f_c = f_c\ .(x\,\overline{y}\,z + \overline{x}\,y\,\overline{z})$

Pour les consensus d'ordre 3 et supérieur dans une somme (OU), il n'existe pas d'opérateur standard. Il n'y a donc pas de simplification possible.

Outre les consensus, la simplification peut se faire en utilisant les propriétés algébriques des opérateurs élémentaires que nous avons déjà vues précédemment (double distributivité, éléments neutres, éléments absorbants).
On en déduit notamment les **relations d'absorption** suivantes (A et B désignant des variables ou des fonctions logiques):

$$A + AB = A$$
$$A + \overline{A}B = A + B$$

Méthodes graphiques

La recherche systématique des consensus d'ordre 1 et 2 peut se faire par des méthodes graphiques. La méthode la plus répandue est celle des tableaux de Karnaugh.
Une fonction de n variables est représenté par un tableau à deux dimensions contenant 2n cases. A chaque case est associée une combinaison algébrique des variables et le passage d'une case à sa voisine se fait par changement d'une seule variable à la fois (utilisation du code binaire réfléchi).

Exemple pour 3 variables :

x \ yz	00	01	11	10
0	$\overline{x}\ \overline{y}\ \overline{z}$	$\overline{x}\ \overline{y}\ z$	$\overline{x}\ y\ z$	$\overline{x}\ y\ \overline{z}$
1	$x\ \overline{y}\ \overline{z}$	$x\ \overline{y}\ z$	$x\ y\ z$	$x\ y\ \overline{z}$

Dans le tableau de Karnaugh on note dans chaque case la valeur de la combinaison correspondante. Si cette valeur est 1, la combinaison correspond à un MINTERME de la fonction.
Le principe de la simplification repose sur le regroupement des cases à 1 adjacentes. Les règles à respecter sont les suivantes.
- Ce regroupement se faire par ensemble de 2^k cases en ligne, colonne, carré ou rectangle.
- Il faut utiliser toutes les cases à 1 au moins une fois.
- On considère comme adjacentes 2 cases dont les combinaisons correspondantes ne diffèrent que d'une variable. Pour les tableaux à 2, 3 ou 4 variables, cela correspond aux cases qui ont un coté en commun mais aussi aux cases extrêmes. Pour les tableaux à plus de 4 variables, il faut tenir compte en plus des règles de symétrie du code binaire réfléchi.

Pour une fonction à n variables :
- un regroupement de 1 case seule correspond à un terme à n variables,
- un regroupement de 2 cases correspond à un terme à (n-1) variables,
- un regroupement de 4 cases correspond à un terme à (n-2) variables,
...
- un regroupement de 2^k cases correspond à un terme à (n-k) variables.

Exemple : simplifions par la méthode de Karnaugh l'expression

$$\overline{x}\ y\ z + x\ \overline{y}\ \overline{z} + x\ y\ z + x\ y\ \overline{z}$$

Le tableau de Karnaugh de cette expression est le suivant :

x \ yz	00	01	11	10
0	0	0	1	0
1	1	0	1	1

La simplification donne directement : $yz + x\overline{z}$.

Fonctions incomplètement spécifiées.

La méthode des tableaux de Karnaugh est particulièrement utile lorsque la fonction à synthétiser n'est pas complètement spécifiée.
Nous avons toujours supposé jusqu'ici que les fonctions logiques étaient entièrement spécifiées, c'est à dire que pour n'importe quelle combinaison des variables d'entrées, nous connaissions la valeur de la fonction. Cependant, il peut arriver que pour certaines combinaisons de ces variables, la valeur de la fonction ne soit pas définie parce que, par exemple :

- cette combinaison n'est pas possible physiquement,
- cette combinaison peut avoir lieu mais la valeur de la fonction est indifférente à ce moment là car elle n'intervient pas dans le résultat global du système.

Dans ces deux cas, le concepteur peut choisir les valeurs non spécifiées à sa convenance, pour tenter de minimiser les équations qui en résultent. Une valeur non spécifiée par le cahier des charges est traditionnellement notée Φ dans le tableau de Karnaugh. Notez que, une fois les équations choisies, les valeurs non spécifiées initialement sont déterminées et fixées.

Exemple :
Soit un système S commandé par quatre contacts a, b, c et d. Le système S délivre un signal de sortie (S=1) si :

- d est fermé, a et c sont ouverts,
- d est ouvert, a et c sont fermés,
- a est fermé, b, c et d sont ouverts.

D'autre part :

- c et d ne sont jamais fermés ensemble (--> impossibilité)
- si a et b sont fermés alors que c est ouvert le signal de sortie est indifférent.

La convention retenue pour ce système est la suivante : un contact ouvert correspond à un signal à 0 et un contact fermé à un signal à 1.

Le tableau de Karnaugh de se système est le suivant :

ab \ cd	00	01	11	10
00	0	1	Φ	0
01	0	1	Φ	0
11	Φ	Φ	Φ	1
10	1	0	Φ	1

Les mintermes des cases Φ sont introduits dans la fonction dans la mesure où ceux-ci permettent d'obtenir une forme minimale plus simple.
La fonction simplifiée sans tenir compte des cases Φ serait :

$$S = \bar{a}\,\bar{c}\,d + a\,c\,\bar{d} + a\,\bar{b}\,\bar{d}$$

En prenant en compte certaine des cases Φ, on obtient le forme minimale suivante :

$$S' = \bar{a}\,d + a\,\bar{d}$$

Conclusion sur les méthodes graphiques

Les méthodes graphiques, et en particulier celle de Karnaugh, sont très pratiques pour simplifier des fonctions de 2 à 6 variables mais elles ne donnent pas d'une manière systématique toutes les formes minimales équivalentes.
Ces méthodes sont également utilisée par résoudre les problèmes de discontinuité dans les circuits logiques (ou aléas de commutation). Nous en reparlerons lors de l'étude des circuits logiques séquentiels.

Méthodes algorithmiques ou itératives

Les méthodes algorithmiques sont une réponse aux difficultés d'utilisation des méthodes graphiques. On les applique surtout sur des fonctions contenant un grand nombre de variables. Elles sont programmables sur ordinateur et présentent l'avantage de donner toutes les formes minimales équivalentes d'une fonction. Nous citerons simplement ici deux algorithmes, parmi les plus utilisés.

L'algorithme de Quine-McCluskey date de 1956. Il utilise, de manière systématique et non graphique, des tables qui décrivent tous les mintermes possibles d'une fonction.
Tout comme la méthode des tableaux de Karnaugh, la taille des données à manipuler augmente de manière exponentielle avec le nombre de variables d'entrées. Cette méthode exhaustive ne sera donc pas bien adaptée pour simplifier des systèmes complexes à grand nombre de variables d'entrées.

Le deuxième type d'algorithme qui a vu le jour dans les années 1980, avec l'avènement des logiciels de synthèse de circuits logiques, ne repose pas sur une table des mintermes. Ces algorithmes (Expresso par exemple) manipule l'expression algébrique d'une fonction, en tentant de la transformer, de proche en proche, pour aboutir à une expression plus simple. Ce type d'algorithme ne garantie pas l'obtention d'une simplification optimale, mais il laisse espérer une simplification importante en un temps de calcul raisonnable. La plupart des logiciels de CAO utilise ce type d'algorithme pour les calculs de minimisation d'expressions logiques.

5) Annexe : Préfixes pour les multiples binaires

D'après la norme IEC 60027-2 (voir http://physics.nist.gov/cuu/Units/binary.html), les préfixes à utiliser pour les multiples binaires sont les suivants.

Factor	Name	Symbol	Origin	Derivation
2^{10}	kibi	Ki	kilobinary: $(2^{10})^1$	kilo: $(10^3)^1$
2^{20}	mebi	Mi	megabinary: $(2^{10})^2$	mega: $(10^3)^2$
2^{30}	gibi	Gi	gigabinary: $(2^{10})^3$	giga: $(10^3)^3$
2^{40}	tebi	Ti	terabinary: $(2^{10})^4$	tera: $(10^3)^4$
2^{50}	pebi	Pi	petabinary: $(2^{10})^5$	peta: $(10^3)^5$
2^{60}	exbi	Ei	exabinary: $(2^{10})^6$	exa: $(10^3)^6$

Examples and comparisons with SI prefixes	
one **kibibit**	1 Kibit = 2^{10} bit = **1024 bit**
one **kilobit**	1 kbit = 10^3 bit = **1000 bit**
one **mebibyte**	1 MiB = 2^{20} B = **1 048 576 B**
one **megabyte**	1 MB = 10^6 B = **1 000 000 B**
one **gibibyte**	1 GiB = 2^{30} B = **1 073 741 824 B**
one **gigabyte**	1 GB = 10^9 B = **1 000 000 000 B**

6) Bibliographie

Ouvrages généraux sur l'électronique analogique :
« Circuits fondamentaux de l'électronique analogique. », TRAN TIEN Lang (disponible à la bibliothèque de l'IO, Cote : B1200)

« Principes et applications de l'électronique T1 : calcul des circuits & fonctions. », DIEULEVEULT (François) De ; FANET (Hervé). , 1997. - X - 331p., ISBN 2-10-003162-7 (disponible à la bibliothèque de l'IO, Cote : B1200)

"The art of electronics", Horowitz and Hill, Cambridge Univ. Press

"Principes d'électronique", Albert Paul Malvino, Mc-Graw-Hill.

Ouvrages généraux sur l'électronique logique :

T.T. Lang, *Electronique numérique*, Masson – 1995 (disponible à la bibliothèque de l'IOTA, cote B1265).

P. Horowitz et W. Hill, *Traité théorique et pratique d'électronique - Volume 2 : Techniques numériques et analogiques,* Elector - 1996 (disponible à la bibliothèque de l'IOTA, cote B1200).

Cours avec exercices corrigés :

J. P. Vabre et J. C. Lafont, *Cours et problèmes d'électronique numérique*, Ellipses – 1998 (disponible à la bibliothèque de l'IOTA, cote B1200).