La logique binaire

Table des matières

La logique binaire

L'algèbre de Boole

L'algèbre de Boole est la partie des mathématiques, de la logique et de l'électronique qui s'intéresse aux opérations et aux fonctions sur les variables logiques.

Le nom provient de George Boole. George Boole est le fondateur de la logique moderne. L'algèbre de Boole est une algèbre permettant de traduire des signaux (tout ou rien) en expressions mathématiques en remplaçant chaque signal élémentaire par des variables logiques et leur traitement par des fonctions logiques.

L'algèbre de Boole permet de résoudre des équations logiques afin de réaliser des fonctions sur des signaux numériques. Ces fonctions seront appelées fonctions combinatoires.

L'algèbre de Boole des fonctions logiques permet de modéliser des raisonnements logiques, en exprimant un « état » en fonction de conditions.

Un mathématicien britannique qui, durant le milieu du XIXe siècle, restructura complètement la logique en un système formel.

Plus spécifiquement, l'algèbre booléenne permet d'utiliser des techniques algébriques pour traiter les expressions à deux valeurs.

Aujourd'hui, l'algèbre de Boole trouve de nombreuses applications en informatique et dans la conception des circuits électroniques.

État des contacts et des récepteurs.

Un circuit électrique, électronique ou pneumatique, peut avoir 2 états logiques. Ces états peuvent prendre les valeurs 1 ou 0. Ces états sont fonctions de l'état des composants en série dans le circuit.

État 0 :

Les actionneurs tels que : moteurs, vérins sont à l'état 0 (ou niveau bas) lorsqu'ils ne sont pas alimentés. Le circuit est alors ouvert. Pour un circuit pneumatique ceci correspond à une absence de pression. Pour un circuit électrique cela correspond à une absence de différence de potentiel entre les bornes du circuit. Pour un contact ou un distributeur, c'est l'absence d'action physique intervenant sur un contact qui représente l'état 0.

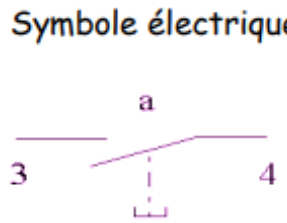
État 1 :

Les actionneurs sont à l'état 1 (niveau haut) lorsqu'ils sont alimentés. Pour un circuit pneumatique ou hydraulique ceci correspond à une pression d'air ou d'huile dans le circuit. Pour un circuit électrique cela correspond à une différence de potentiel entre les bornes du circuit. Pour un contact ou un distributeur ils sont actionnés, c'est à dire qu'une action physique est prise en compte.

Définitions

Contact à fermeture :

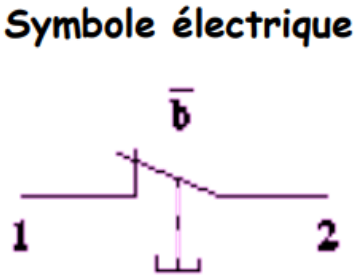
C'est un contact qui est normalement ouvert (Normally Open) au repos. Il se ferme lorsqu'il est actionné. On désigne ce type de contact par des lettres minuscules a, b, c ... Ses bornes sont repérées par des chiffres.



Contact ouvert au repos (NO)

Contact à ouverture :

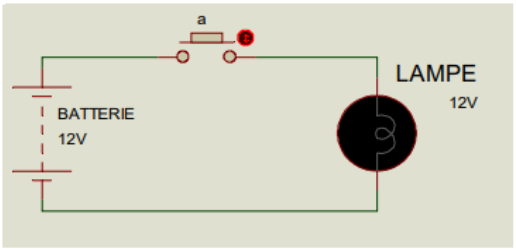
C'est un contact qui est normalement fermé (Normally Closed) au repos et qui s'ouvre lorsqu'il est actionné. On désigne ce type de contact par des lettres $\overline{a}$, $\overline{b}$, $\overline{c}$, a/, b/, c/ ($\overline{a}$ se lit "a barre"). Ses bornes sont repérées par des chiffres, 1 et 2 ici.



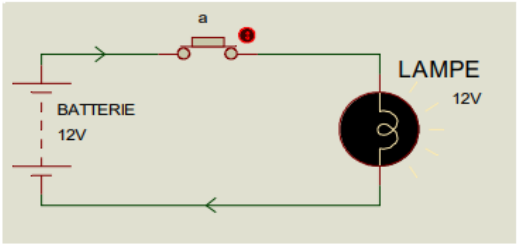
Contact fermé au repos (NC)

État d'un circuit électrique :

Un circuit électrique est dit passant, ou fermé, lorsqu'un courant électrique circule dans ce circuit. Cela implique qu'il y ait continuité de ce circuit, c'est à dire que le contact établisse le circuit. Un circuit électrique est non passant, ou ouvert, si le courant ne peut pas circuler dans ce circuit. Un circuit électrique comprend au minimum, une source d'énergie, un récepteur et un contact.



Circuit ouvert, pas de courant



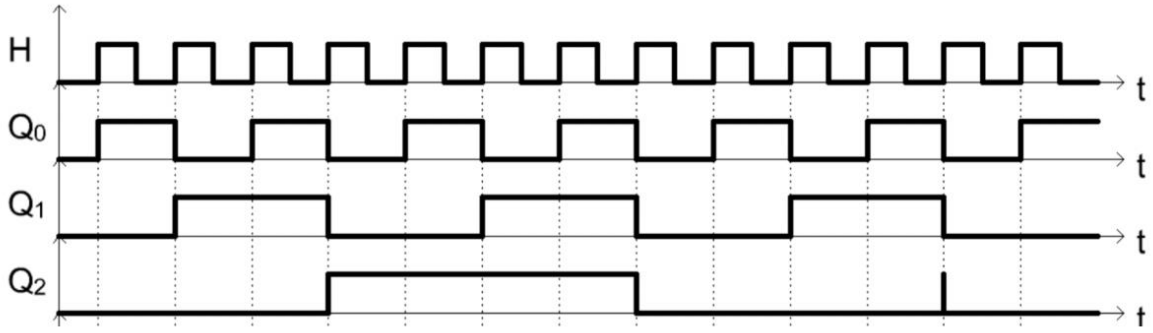
Circuit fermé, circulation d'un courant

logique_fig_01_mp4

logique_fig_01.DSN

Chronogramme :

Un chronogramme est une représentation schématique temporelle de l'évolution d'un système automatisé en fonction des variations d'état d'une ou plusieurs entrées.
Cette évolution est représentée sous la forme suivante :



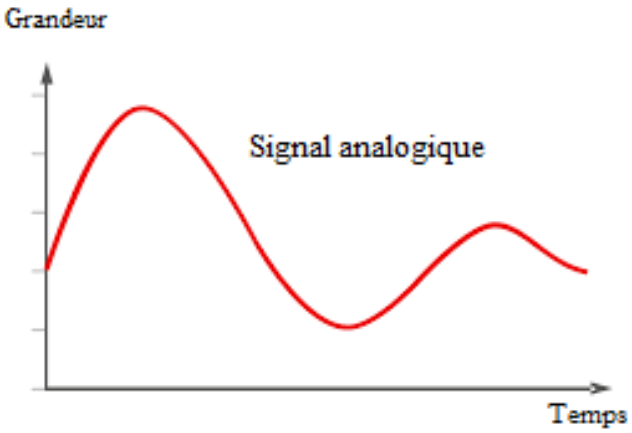
Types de signaux

Différents types de signaux :

Dans les systèmes trois types de signaux sont utilisés principalement. Les signaux analogiques, numériques et tout ou rien.

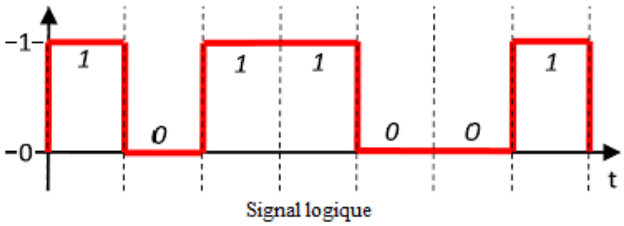
Signal analogique

Un signal analogique est un signal qui représente la variation continue (continue au sens mathématique) d'une certaine grandeur (ex : température, vitesse ...).



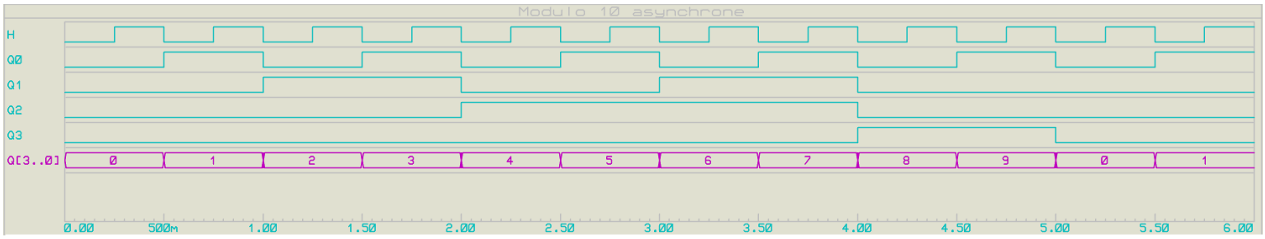
Signal logique

Un signal logique (tout ou rien) est un signal qui représente l'état binaire (vrai, non vrai) d'une variable d'un système (ex : un contacteur d'un circuit électrique est soit actionné soit non actionné).
Un signal logique ne peut prendre que 2 valeurs, un niveau haut (en anglais "high" = "H"), et un niveau bas (en anglais "low" = "L").



Signal numérique

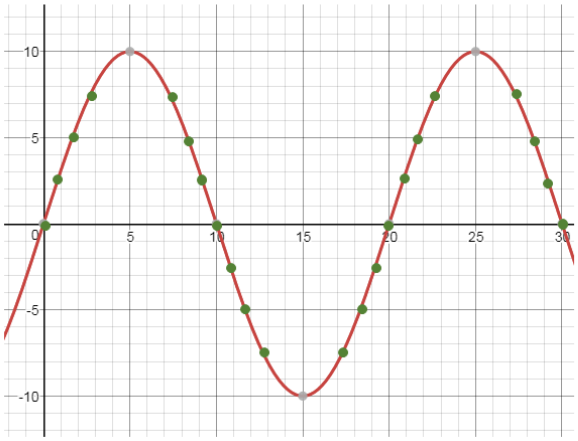
Les signaux numériques sont également des signaux logiques mais qui représentent des valeurs numériques.



Ils sont le plus souvent transmis en série entre deux équipements.

Conversion analogique / numérique

Signal analogique



Numérisation 3 bits

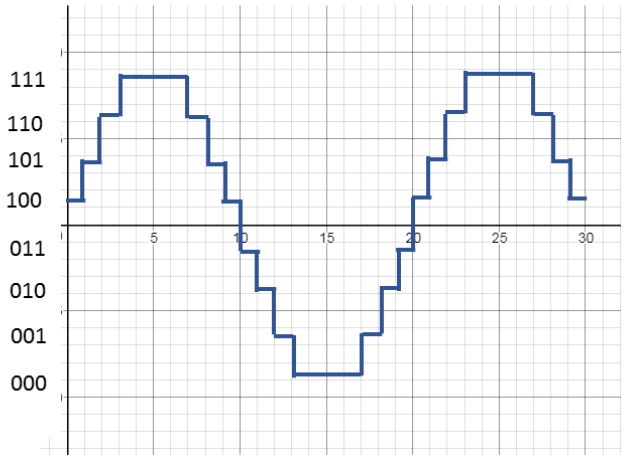
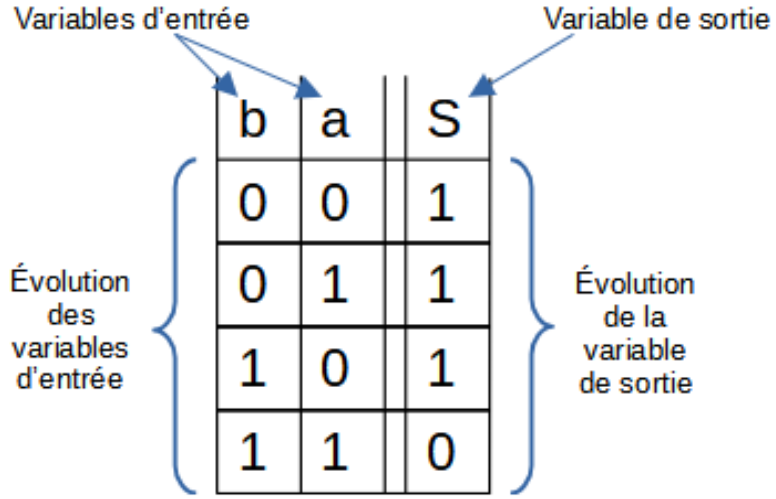


Table de vérité

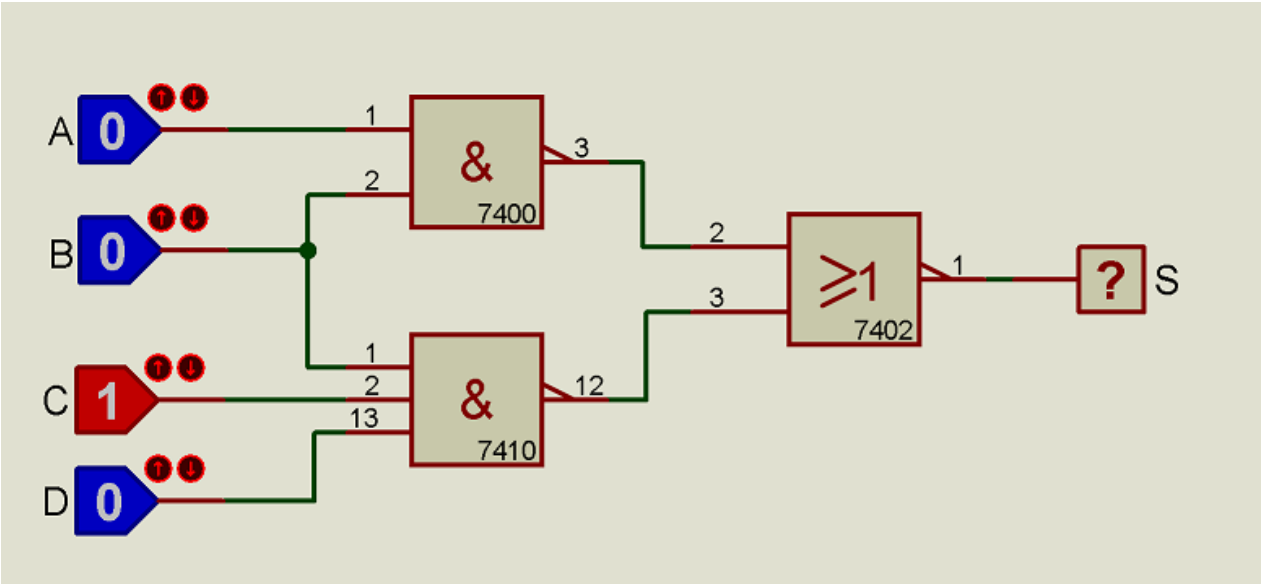
Table de vérité :

Une table de vérité est la représentation de l'évolution du comportement d'un système automatisé en fonction des variations de ses entrées. Chacune des variables est représentée sous une écriture binaire. Une table de vérité s'utilise principalement en logique combinatoire.
Elle est représentée sous la forme suivante :



Logigramme :

Un logigramme est un schéma structurel représentant une succession de symboles logiques permettant d'obtenir par combinaison de variables d'entrées la sortie recherchée. Attention, les fonctions logiques sont des opérateurs logiques et non des opérateurs mathématiques. Le résultat obtenu sera un résultat logique et non un résultat mathématique.



logique_fig_02.DSN

Codes binaires

Codes binaires :

Le code binaire pur :

Le code binaire pur est une représentation numérique en base deux. Cette représentation permet de représenter des nombres sous forme de 1 et de 0, ou de décrire l'évolution des variables vraies ou non vraies d'un système automatisé, c'est cette possibilité que nous allons utiliser.

Le nombre de combinaisons possibles des variables se calcule de la façon suivante :

- 1 variable d'entrée ==> $2^1 = 2$ combinaisons (0, 1)
- 2 variables d'entrée ==> $2^2 = 4$ combinaisons (00, 01, 10, 11)
- 3 variables d'entrée ==> $2^3 = 8$ combinaisons (000, 001, …)
- 4 variables d'entrée ==> $2^4 = 16$ combinaisons (0000, …)
- n variables d'entrée ==> 2^n combinaisons possibles

Puissance de 2 :	2^4	2^3	2^2	2^1	2^0	
					0	
					1	← Épuisement
Puissance supérieure →				1	0	
				1	1	← Épuisement
Puissance supérieure →			1	0	0	
			1	0	1	
			1	1	0	
			1	1	1	← Épuisement
Puissance supérieure →		1	0	0	0	
		1	0	0	1	
		1	0	1	0	
		1	0	1	1	
		1	1	0	0	
		1	1	0	1	
		1	1	1	0	
		1	1	1	1	← Épuisement
Puissance supérieure →	1	0	0	0	0	

Le code binaire réfléchi :

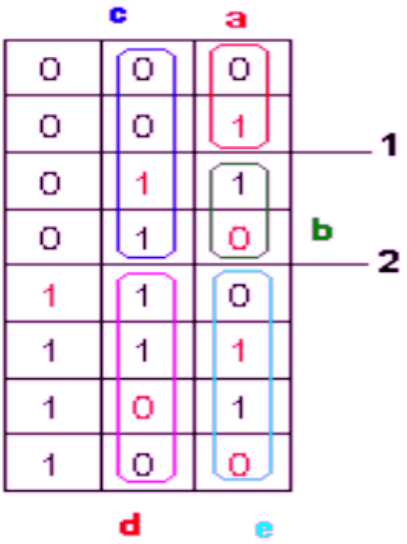
Lorsque l'on regarde ligne par ligne l'évolution du code binaire pur, on remarque que pour passer d'une ligne à l'autre, plusieurs variables peuvent être amenées à changer de valeur simultanément. Ceci est très gênant lorsque l'on cherche à analyser le comportement d'un système en fonction de ses entrées.

Un autre code binaire a été mis au point, c'est le code binaire réfléchi ou code GRAY du nom de son inventeur. Ce code permet de passer d'une ligne à l'autre de la description d'un système avec l'évolution d'une seule variable à la fois. Ce code permettra de définir l'évolution d'un système automatisé. En aucun cas il ne pourra servir de base de comptage comme le binaire pur. On utilisera ce codage ultérieurement dans le cours pour la définition des tableaux de Karnaugh.

Binaire pur

0	0	0	
0	0	1	1 changement
0	1	0	2 changements
0	1	1	1 changement
1	0	0	3 changements
1	0	1	1 changement
1	1	0	2 changements
1	1	1	1 changement

Binaire réfléchi



Le binaire réfléchi est construit par symétrie de lignes. Le groupement "a" est reproduit en "b" symétriquement par rapport à la ligne 1. Les groupements "a" et "b" sont reproduits en "e" symétriquement par rapport à la ligne 2. La même règle prévaut pour les groupements "c" et "d". La suite se construit à l'identique. On remarque alors qu'une seule variable, la variable rouge, évolue d'une ligne à l'autre.

Le code BCD :

(BCD pour Binary-Coded Decimal, DCB Décimal Codé en Binaire)
Le code BCD permet de coder un nombre décimal (en base 10) à l'aide du binaire, en effet chaque chiffre composant ce nombre est codé en binaire à l'aide de 4 bits.
exemple:
$(1958)_{10}$ ==> $(0001\ 1001\ 0101\ 1000)_{BCD}$

Le complément d'une variable

Nous avons vu précédemment qu'une variable «e» avait deux états, l'état 0 et l'état 1.
Si on admet qu'il peut exister une variable «$\bar{e}$» qui a l'état inverse de la variable «e», alors on pourra dire que «$\bar{e}$» est le complément de «e».

« $\bar{e}$ » est le complément de « e »

Fonctions logiques

Fonctions logiques

Les fonctions logiques sont des opérateurs logiques. C'est à dire qu'en fonction d'une ou plusieurs variables données (entrées), ils vont répondre par une sortie particulière.

Symbolisation des fonctions logiques (norme IEEE)
voir document Explanation of Logic Symbols

Afin de faciliter la lecture de schéma structurel en électroniques, l'IEEE *(Institute of Electrical and Electronic Engineers)* à développé dans les années 1970 un nouveau standard de symbolisation logique. Cette symbolisation permet de montrer la relation entre chaque entrée d'un circuit et ses sorties sans qu'il soit nécessaire d'expliciter la composition interne de ce circuit.

Symbolisation

Un symbole comprend un contour ou une combinaison de contours avec un ou plusieurs symboles de qualification (Fig. 1). Le but des symboles de qualification générale est de décrire avec précision la fonction logique de l'élément, et les plus utilisés sont listés dans la table 1. la direction privilégiée du flux de signal à travers les symboles et les circuits associés est de la gauche vers la droite; les entrées sont à gauche et les sorties à droite. Les exceptions à cette convention sont indiquées par des flèches sur les lignes de signaux montrant la direction du flux du signal.

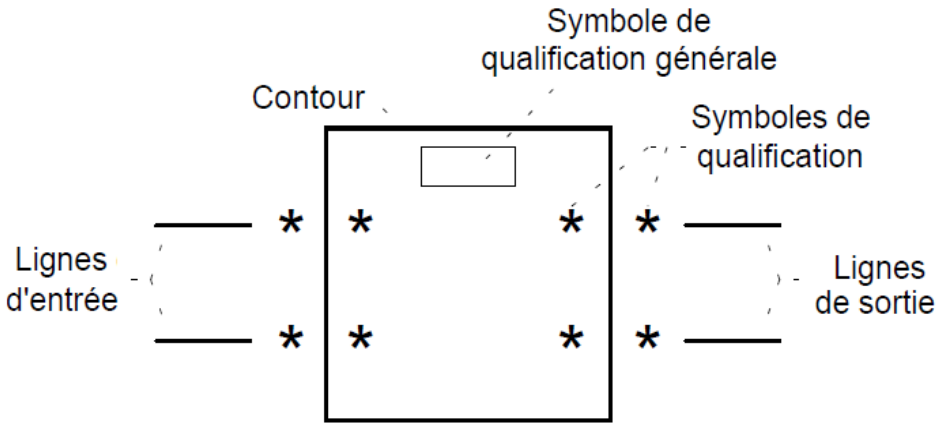


Fig. 1 Composition d'un symbole

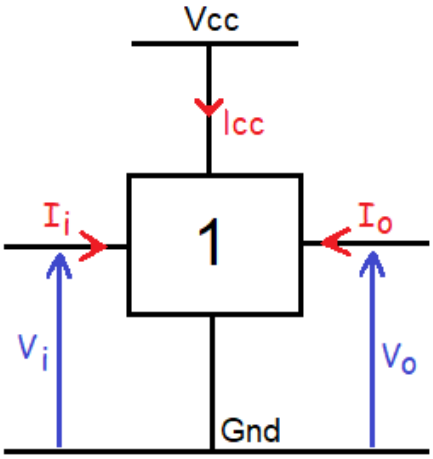
Symboles de qualification générale (Table 1)

Symbole	Description
&	ET logique
≥1	OU logique
=1	OU Exclusif (seule une entrée doit être active pour activer la sortie)
▷	Sortie amplifiée (buffer)
⎍	Trigger de Schmitt (élément avec hystérésis)
X/Y	Codeur, convertisseur de codes… (BIN/7SEG, DEC/BCD…)
MUX	Multiplexeur
DMUX ou DX	Démultiplexeur
⎍	Monostable redéclenchable
1 ⎍	Monostable non redéclenchable
CTRm	Compteur, m = nombre de bits (longueur du cycle = 2^m)
CTR DIVm	Compteur don la longueur du cycle est m.
SRGm	Registre à décalage de m bits *(Shift ReGister)*.
ROM	Mémoire à lecture seule *(Read Only Memory)*.
RAM	Mémoire à lecture écriture *(Random Access Memory)*.
ALU	Unité arithmétique et logique *(Arithmetic Logic Unit)*.
[..]	Information non standard.

Symboles de qualification des entrées - sorties (Table 2)

Symbole	Description
—○	Négation logique en entrée.
○—	Négation logique en sortie.
⎍	Entrée active à l'état bas *(équivalent à —○ en logique positive)*.
⎍	Sortie active à l'état bas *(équivalent à ○— en logique positive)*.
◀—	Sens du signal de la droite vers la gauche (non indiqué sinon).
◀—▶	Signal bidirectionnel.
—▷	Entrée dynamique (active sur front montant).
⎍▷	Entrée dynamique (active sur front descendant).
—○▷	Entrée dynamique (active sur transition de 1 vers 0).
—×|	Connexion non logique (composant …)
∩|	Entrée analogique (sur un composant numérique).
#|	Entrée numérique (sur un composant analogique).

Caractéristiques électriques



Niveaux de tension des entrées et des sorties

Un 1 logique et un 0 logique ne peuvent représenter une valeur unique de tension : il s'agit forcément d'une zone de tension. Ainsi on trouvera systématiquement entre O et la tension d'alimentation (Vcc) trois zones : la zone correspondant au 1 logique, celle du zéro et entre les deux une zone où la valeur logique ne peut pas être considérée comme sûre.

Ces zones définissent 4 tensions : V_{IHmin}, V_{ILmax}, V_{OHmin} et V_{OLmin} (on garde ici la notation anglo-saxonne I=input et O=output). Pour être compatible, c'est-à-dire que l'on puisse relier une entrée à une sortie, il faut respecter un certain nombre de conditions sur ces tensions. Commençons par définir ces tensions :

Définition

- V_{IH} (High Level Input Voltage) tension d'entrée de niveau haut

  → V_{IHmin}, permet de spécifier la plage de tension en entrée qui sera considérée comme un 1 logique (entre V_{IHmin} et Vcc),

- V_{IL} (Low Level Input Voltage) tension d'entrée de niveau bas

  → V_{ILmax}, permet de spécifier la plage de tension en entrée qui sera considérée comme un 0 logique (entre V_{ILmax} et 0),

- V_{0H} (High Level Output Voltage) tension de sortie de niveau haut

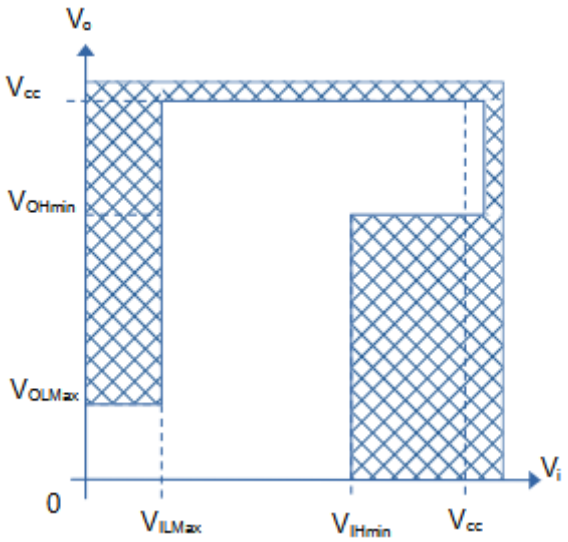
  → V_{OHmin}, permet de spécifier le domaine de tension qu'un circuit aura en sortie s'il est sensé réaliser un 1 logique,

- V_{0L} (Low Level Output Voltage) tension de sortie de niveau bas

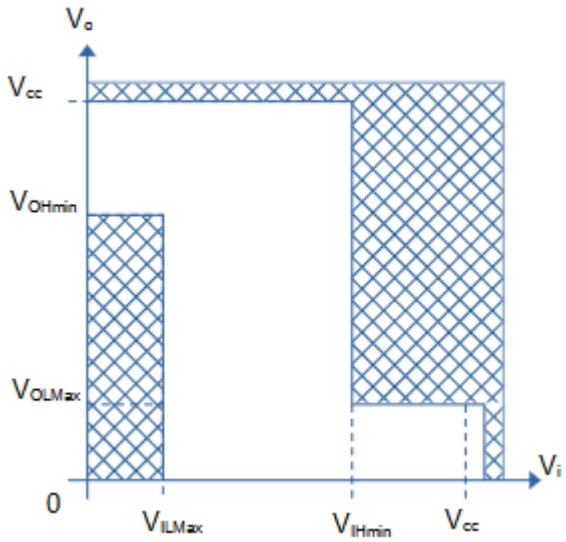
  → V_{OLmin}, permet de spécifier le domaine de tension qu'un circuit aura en sortie s'il est sensé réaliser un 0 logique,

La fonction de transfert $V_o = f(V_i)$ doit donc s'intégrer, selon la fonction, dans l'un des gabarits suivant:

Gabarit d'un Non-Inverseur

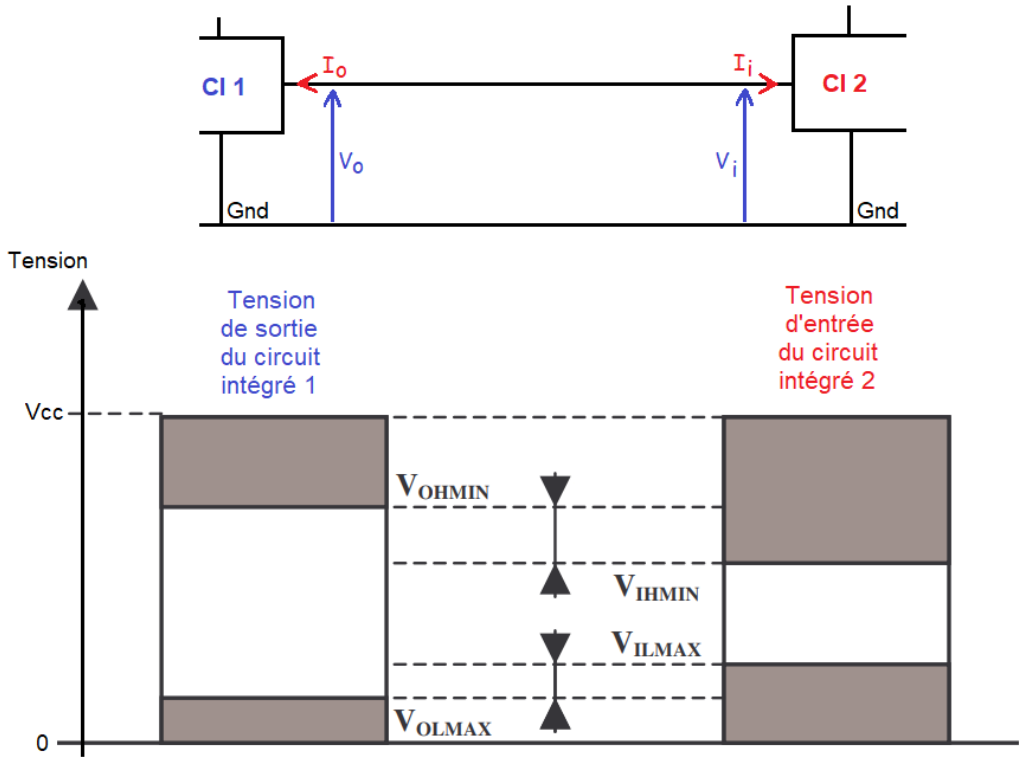


Gabarit d'un Inverseur



La compatibilité en tension signifie que les inégalités suivantes doivent être respectées :

$$V_{IHmin} < V_{OHmin} \text{ et } V_{ILmax} > V_{OLmax}$$



Mais la compatibilité en tension n'est pas suffisante. Il faut aussi être compatible en courant.

Courants de sortie et d'entrée

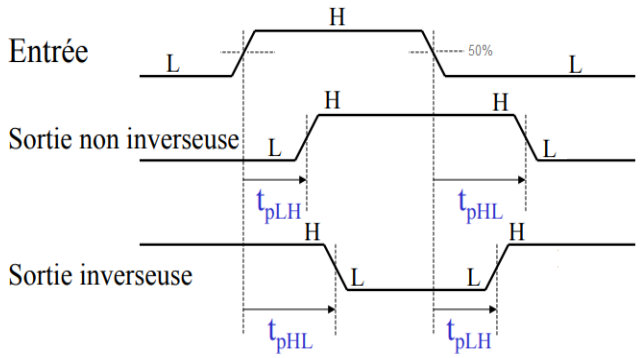
Définition

- I_{IH} (High Level Input Current) courant d'entée de niveau haut
- I_{IL} (Low Level Input Current) courant d'entée de niveau bas
- I_{0H} (High Level Output Current) courant de sortie de niveau haut
- I_{0L} (Low Level Output Curent) courant de sortie de niveau bas

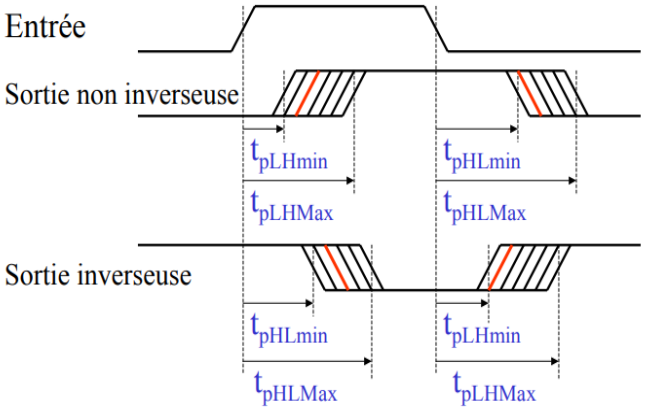
Les problèmes de courant sont liés aux problèmes de tension. Si une entrée consomme trop de courant la tension de sortie risque de se trouver dans la zone indéterminée. Des conditions sur les courants doivent donc être respectées.
Comme il est courant de relier une sortie à plusieurs entrées on définit un moyen simple de calculer le maximum de portes que l'on peut relier ensemble. On pose pour cela la convention suivante : l'entrance (fan-in) vaut 1 pour une porte ET-NON.
La sortance (fan-out) est le nombre maximal d'entrées qu'une sortie peut piloter : c'est le plus petit des rapports I_{OH}/I_{IH} et I_{OL}/I_{IL}.
L'assemblage des circuits suit une règle très simple : il suffit que la la sortance d'un circuit soit supérieure ou égale à la somme des entrances des circuits qu'il commande. Pour cela il faut aussi avoir à l'esprit que l'entrance varie d'un composant à l'autre.

Caractéristiques temporelles (dynamiques)

Les temps de propagation



Valeurs mini et Maxi (données constructeur)

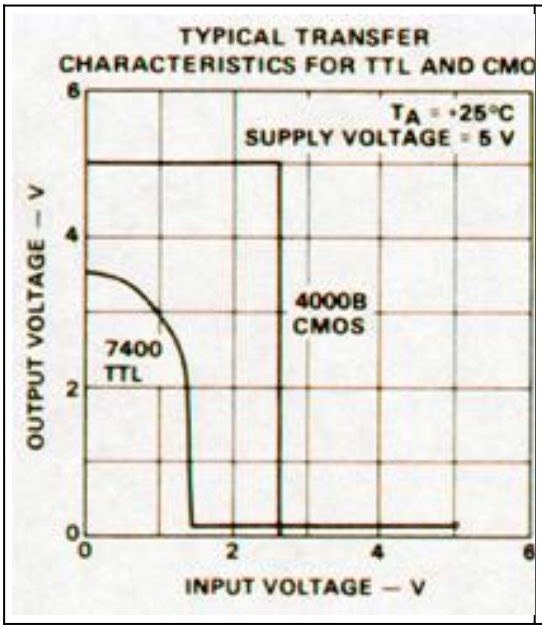


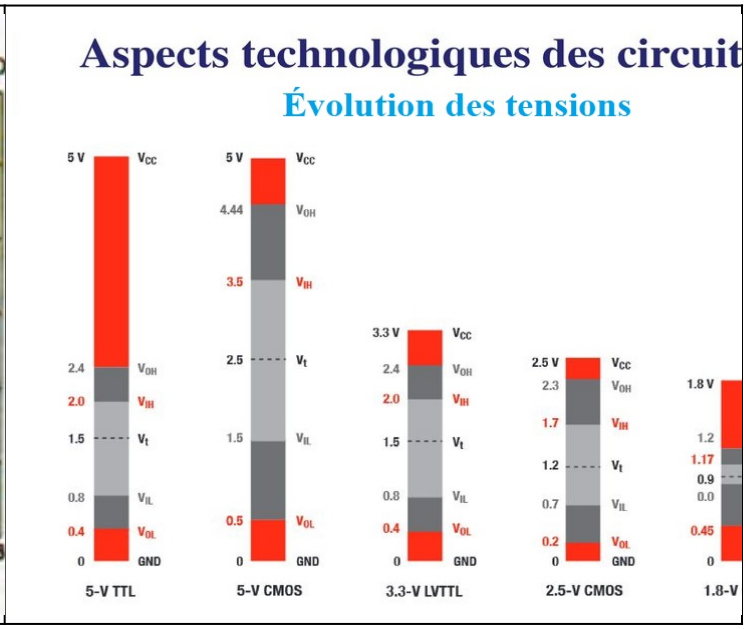
Technologie

TTL (*Transistor-Transistor Logic)* est une famille de circuits logiques utilisée en électronique inventée dans les années 1960. Cette famille est réalisée avec la technologie du transistor bipolaire et tend à disparaître du fait de sa consommation énergétique élevée (comparativement aux circuits CMOS).

CMOS *(Complementary Metal Oxide Semiconductor)* est une technologie de fabrication de composants électroniques et, par extension, les composants fabriqués selon cette technologie. Ce sont pour la plupart des circuits logiques (NAND, NOR, etc.) comme ceux de la famille Transistor-Transistor Logic (TTL) mais, à la différence de ces derniers, ils peuvent être aussi utilisés comme résistance variable.

ECL *(Emitter Coupled Logic ou Logique à émetteurs couplés)* est une technologie de circuits logiques permettant un niveau de performances supérieur à la technologie TTL moyennant une consommation bien plus importante. Pour la conception de circuits logiques, la technologie ECL est aujourd'hui totalement dépassée ; elle a eu son heure de gloire à l'époque du supercalculateur Cray, entièrement réalisé en logique ECL. Elle constitue cependant encore la seule alternative crédible pour la réalisation de portes logiques très rapides, typiquement au-delà de 10 Gbit/s, et trouve de nombreuses applications dans le cadre des télécommunications sur fibre optique.



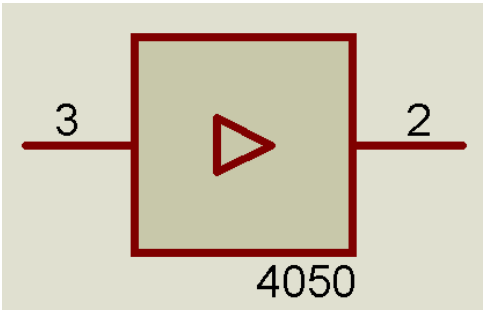
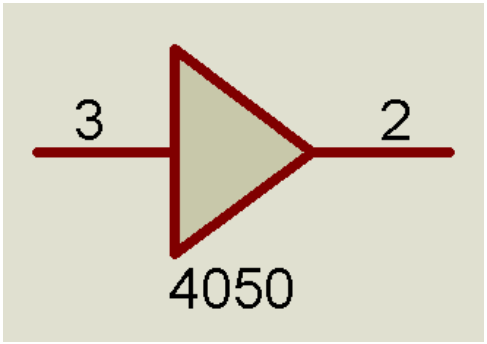


Fonction OUI

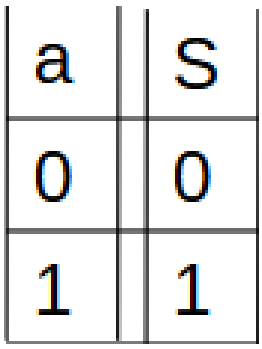
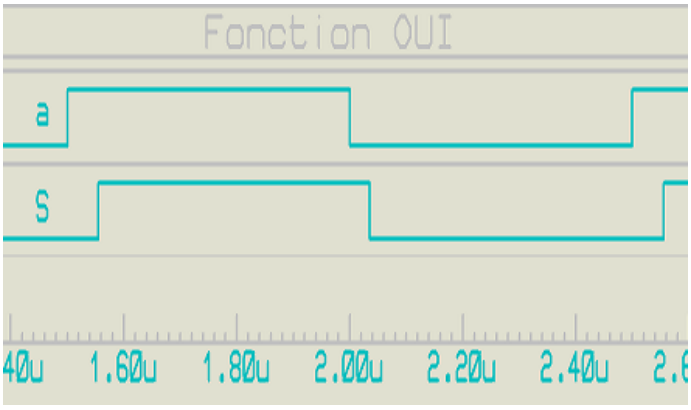
Fonction OUI

(non-inverting)

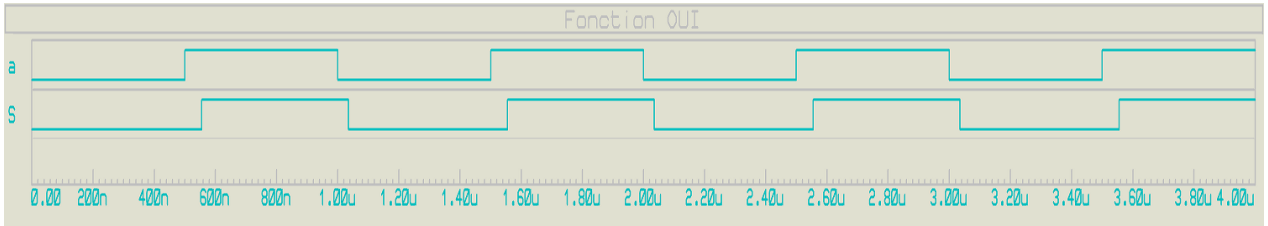
- Équation : S = a ; a ⇒ S
- S reproduit la variable d'entrée a.

Symbole électrique (IEEE/IEC)	Symbole électrique (au siècle dernier avant 1970)
	

Tampon (Oui) : il est connu sous le nom de "buffer" ou de porte directe, car sa sortie aura le même état que son entrée. Bien que cela puisse sembler inutile, dans de nombreux circuits logiques, il est souvent utilisé comme amplificateur de courant afin d'augmenter la sortance ou comme suiveur de tension.

Table de vérité	Chronogramme
	

a	S
0	0
1	1



Fonction_OUI.DSN
Fonction_OUI.pdsprj

Fonction NON

Fonction NON

(inverting)

- Équation : S = $\overline{a}$; $\overline{a} \Rightarrow S$
- S reproduit l'inverse de la variable d'entrée a.

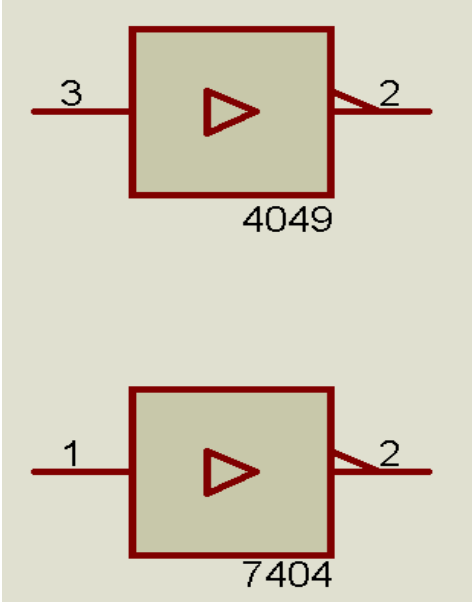
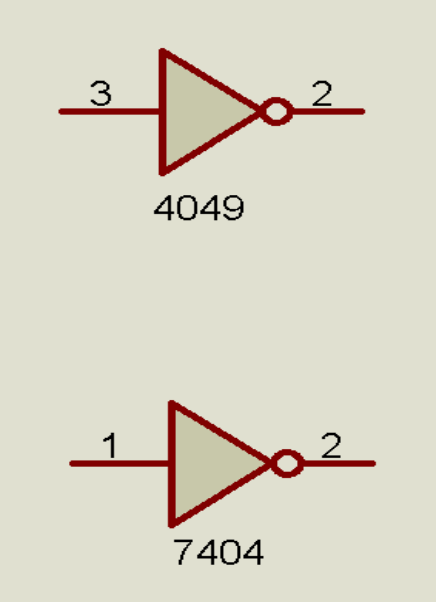
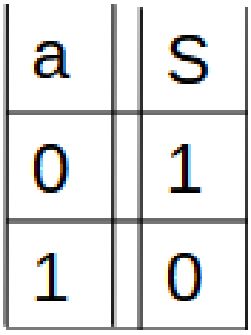
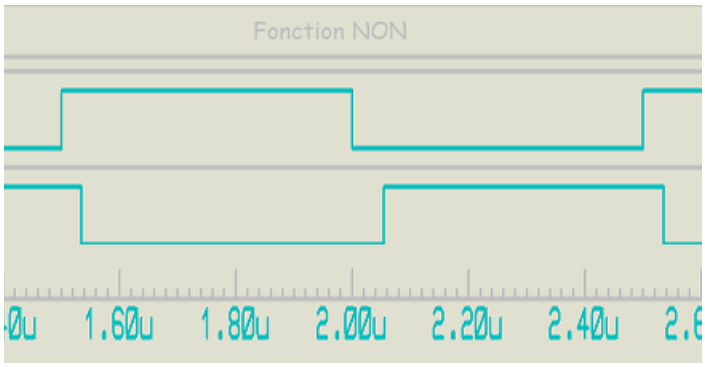
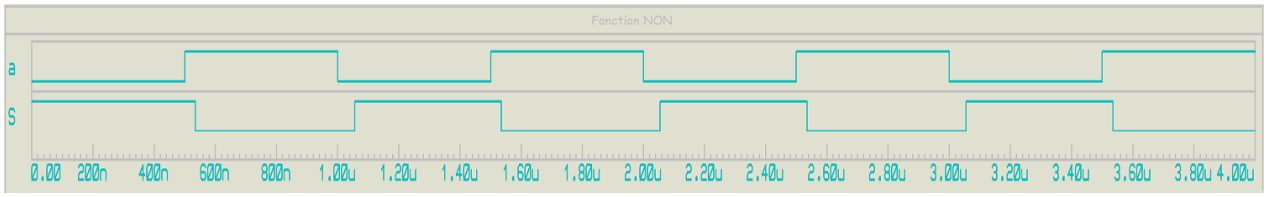
Symbole électrique (IEEE/IEC)	Symbole électrique (avant 1970)
	

Table de vérité	Chronogramme
	



Fonction_NON.DSN
Fonction_NON.pdsprj

Fonction ET

Fonction ET

(AND)

- Équation : S = a . b (lire S égal a ET b)
- S reproduit le produit des variables d'entrée a et b.

La sortie S est au niveau haut, si a = 1 **ET** b = 1, sinon S est au niveau bas

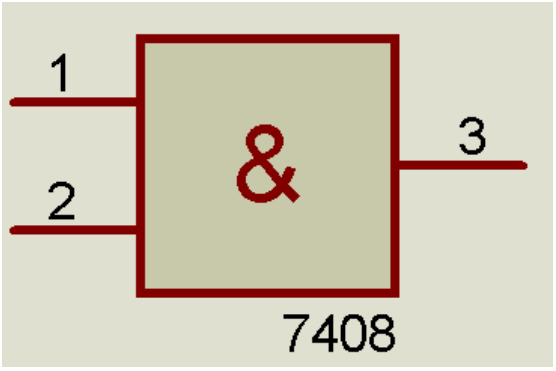
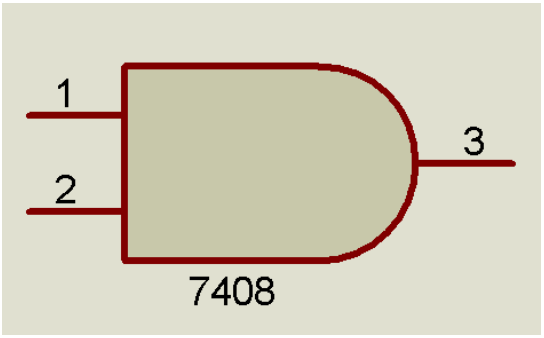
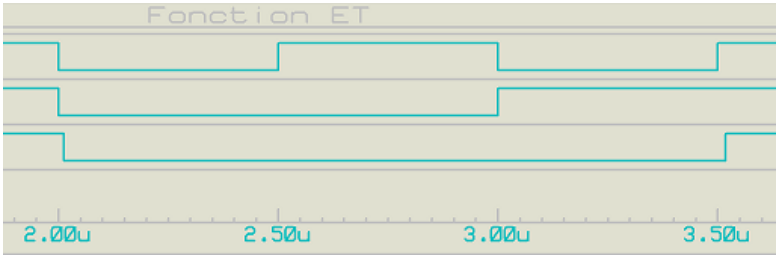
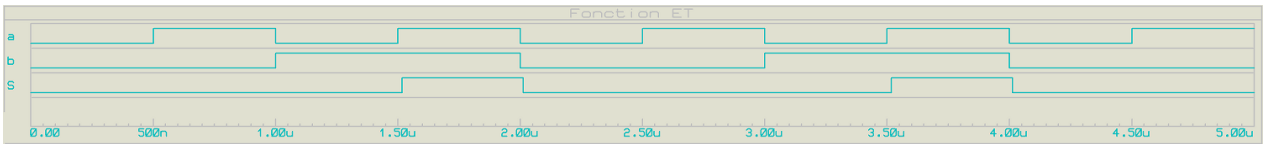
Symbole électrique (IEEE/IEC)	Symbole électrique (< 1970)
	

Table de vérité	Chronogramme

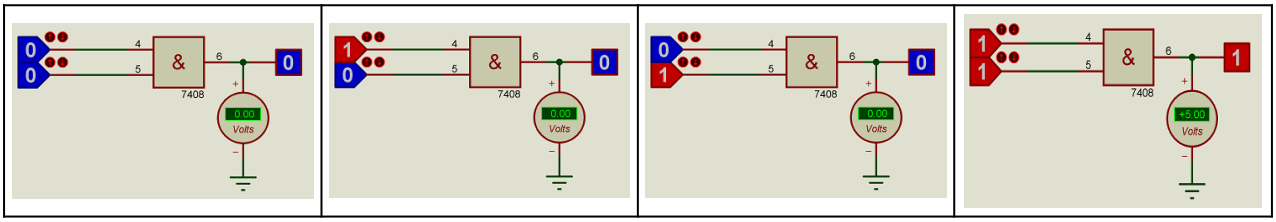
b	a	S = a.b
0	0	0
0	1	0
1	0	0
1	1	1

ET à 2 entrées





Fonction_ET.DSN
Fonction_ET.pdsprj



Porte ET à 3 entrées

- Équation : S = a . b . c (lire S égal a ET b ET c)

  La sortie S est au niveau haut, si a = 1 **ET** b = 1 **ET** c = 1 , sinon S est au niveau bas

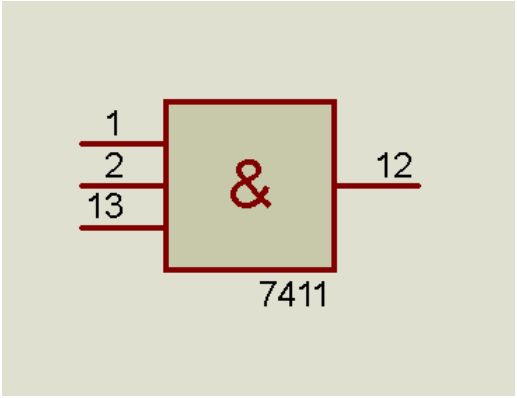
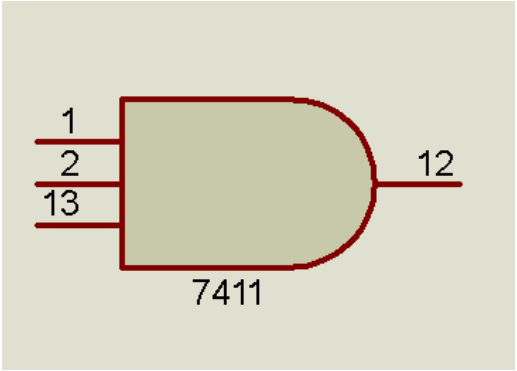
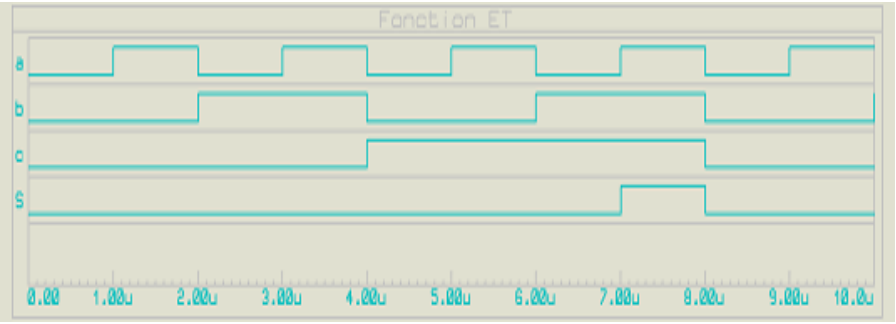
Symbole électrique (IEEE/IEC)	Symbole électrique (< 1970)
	

Table de vérité	Chronogramme

c	b	a	S = a.b.c
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

ET à 3 entrées



remarque:

- la fonction ET est commutative a . b = b . a
- la fonction ET est associative (a . b) . c = a . (b . c) = a . b . c

Q: comment réaliser une fonction ET à 3 variables à l'aide de portes ET à 2 entrées ?
Q: comment réaliser une fonction ET à 4 variables à l'aide de portes ET à 2 entrées ?

Fonction OU

Fonction OU

(OR)

- Équation : S = a + b (lire S égal a OU b)
- S reproduit la somme logique des variables d'entrée a et b.

La sortie S est au niveau haut, si a = 1 **OU** b = 1, sinon S est au niveau bas
Attention le OU est inclusif, la sortie est à 1 si au moins une des entrées est à 1, ce qui justifie l'emploi du symbole « ≥ 1 »

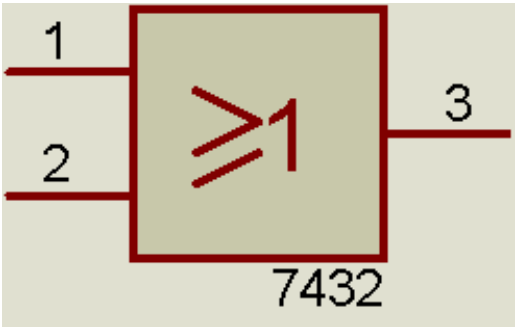
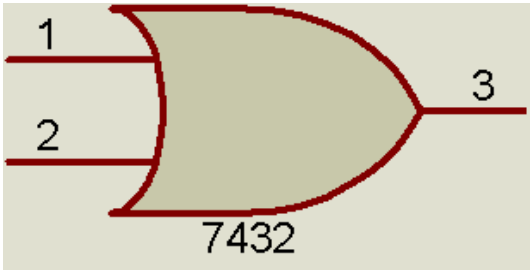
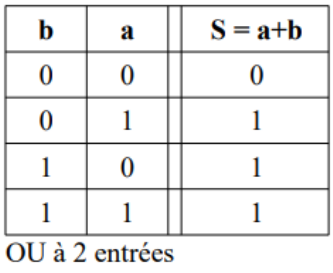
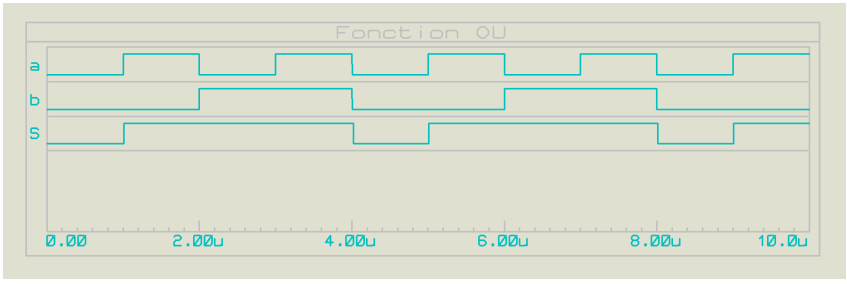
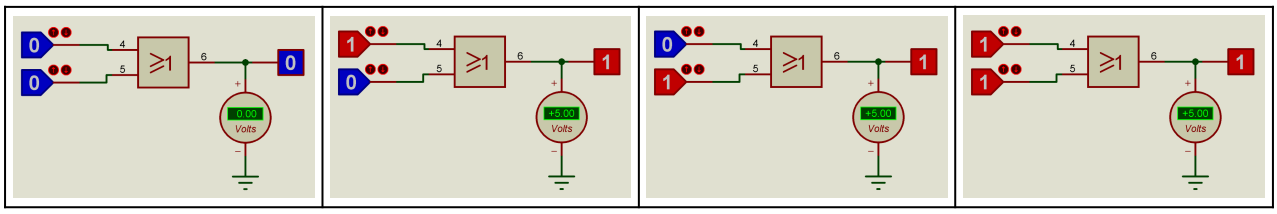
Symbole électrique (IEEE/IEC)	Symbole électrique (< 1970)
	

Table de vérité	Chronogramme
	

b	a	S = a+b
0	0	0
0	1	1
1	0	1
1	1	1

OU à 2 entrées

Fonction_OU.DSN
Fonction_OU.pdsprj



Porte OU à 3 entrées

- Équation : S = a + b + c (lire S égal a OU b OU c)
  La sortie S est au niveau haut, si a = 1 **OU** b = 1 **OU** c = 1 , sinon S est au niveau bas

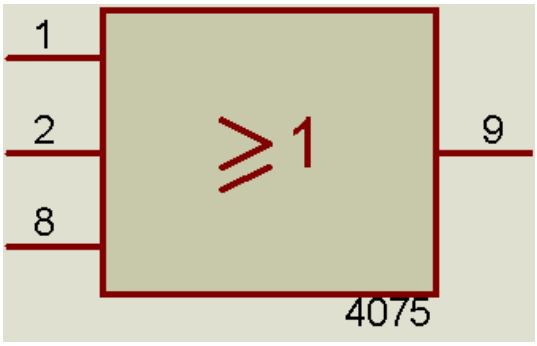
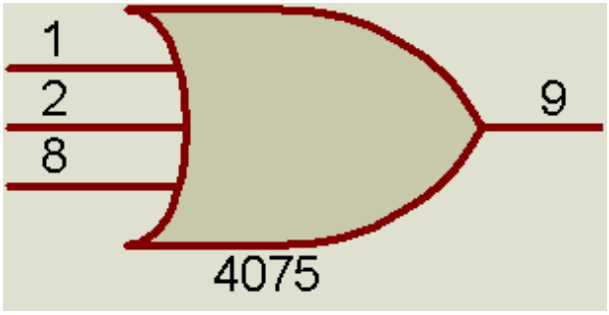
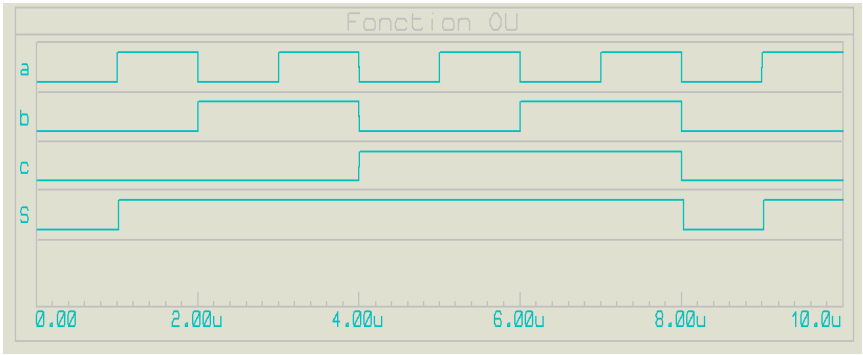
Symbole électrique (IEEE/IEC)	Symbole électrique (< 1970)
	

Table de vérité	Chronogramme

c	b	a	S = a+b+c
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

OU à 3 entrées



remarque:

- la fonction OU est commutative a + b = b + a
- la fonction OU est associative (a + b) + c = a + (b + c) = a + b + c

Q: comment réaliser une fonction OU à 4 variables à l'aide de portes OU à 2 entrées ?

Fonction ET-NON

Fonction ET-NON

(NAND, No-AND)

- Équation : $S = \overline{a \cdot b}$ (lire S égal a ET b le tout barre)
- S reproduit l'inverse du produit des variables d'entrée a et b.

La sortie S est au niveau bas, si a = 1 **ET** b = 1, sinon S est au niveau haut

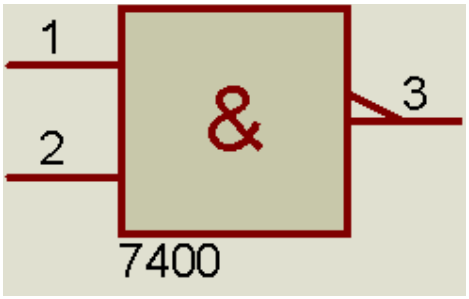
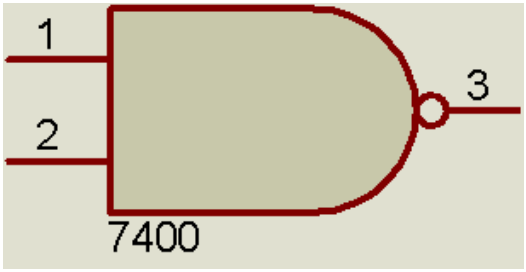
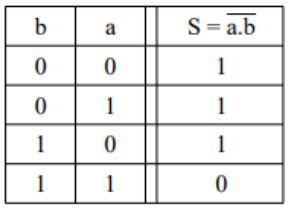
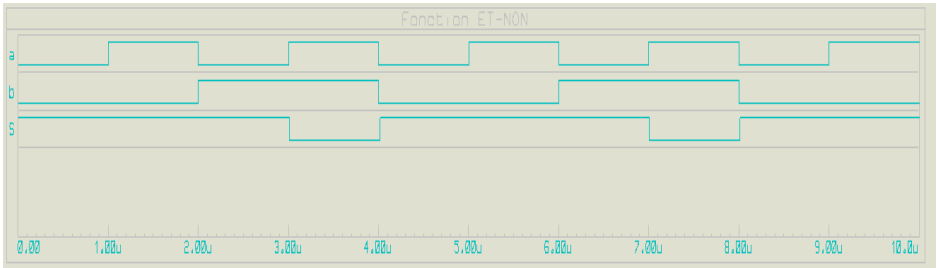
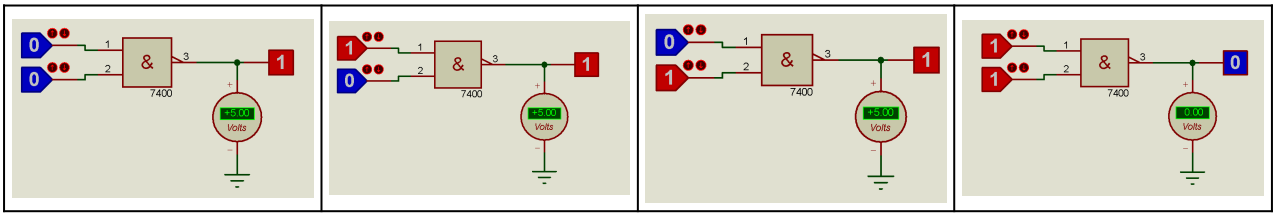
Symbole électrique (IEEE/IEC)	Symbole électrique (< 1970)
	

Table de vérité	Chronogramme
	

b	a	$S = \overline{a.b}$
0	0	1
0	1	1
1	0	1
1	1	0

ET-NON à 2 entrées

Fonction_ET-NON.DSN
Fonction_ET-NON.pdsprj



Porte ET-NON à 3 entrées

- Équation : $S = \overline{a \cdot b \cdot c}$ (lire S égal a ET b ET c le tout barre)

  La sortie S est au niveau bas, si a = 1 **ET** b = 1 **ET** c = 1 , sinon S est au niveau haut

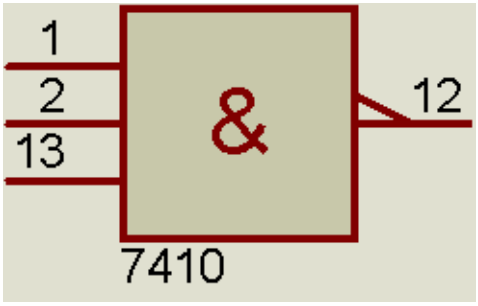
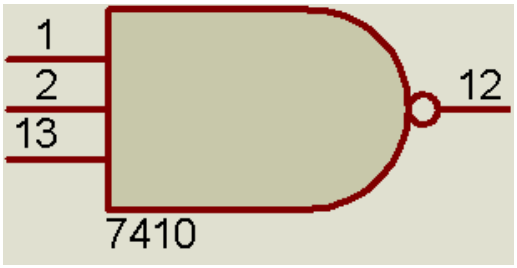
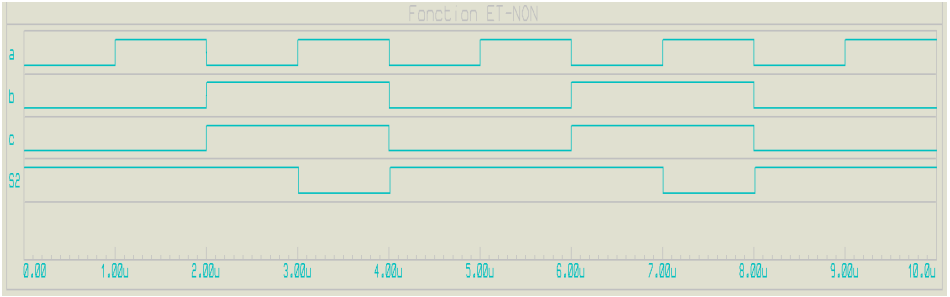
Symbole électrique (IEEE/IEC)	Symbole électrique (< 1970)
	

Table de vérité	Chronogramme

c	b	a	$S = \overline{a.b.c}$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

ET-NON à 3 entrées



Remarque:
La fonction ET-NON, est une fonction universelle, c'est à dire que les fonctions OUI, NON, OU, ET peuvent être réalisées avec uniquement des opérateurs ET-NON.
Ainsi toute fonction même complexe peut être réalisée avec uniquement des portes NAND.

Exemples

a) Réaliser une porte NON à l'aide d'une porte ET-NON
b) Réaliser une porte Et à 2 entrées à l'aide de portes ET-NON
c) Réaliser une porte OU à 2 entrées à l'aide de portes ET-NON

Fonction OU-NON

Fonction OU-NON

(NOR, No-OR)

- Équation : $S = \overline{a + b}$ (lire S égal a OU b le tout barre)
- S reproduit l'inverse de la somme logique des variables d'entrée a et b.

La sortie S est au niveau bas, si a = 1 **OU** b = 1, sinon S est au niveau haut

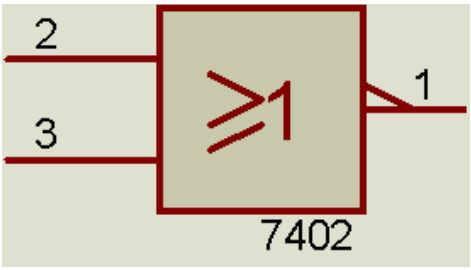
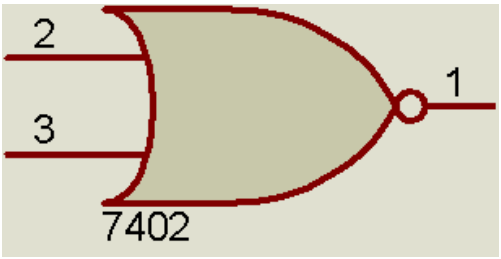
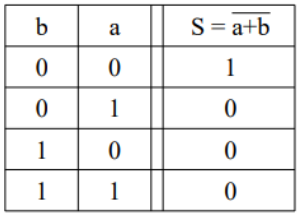
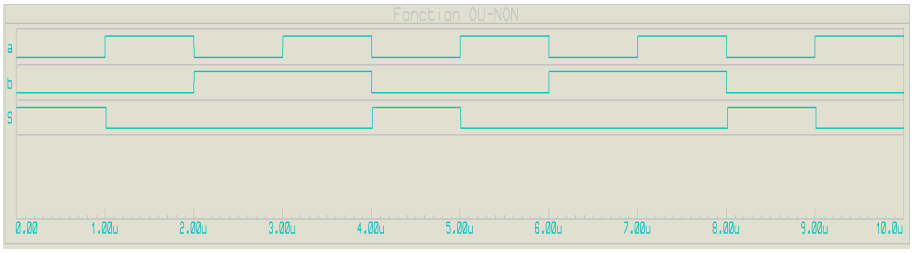
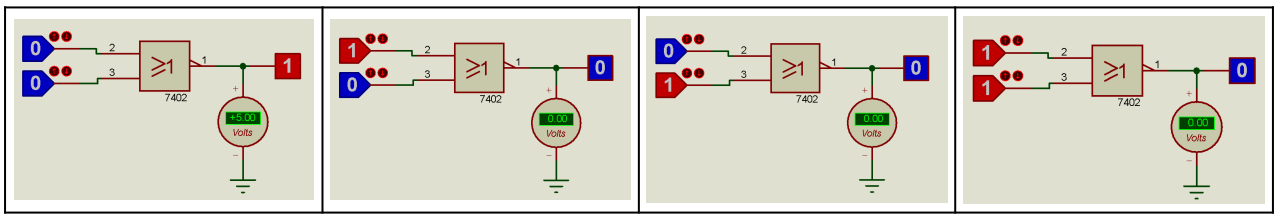
Symbole électrique (IEEE/IEC)	Symbole électrique (< 1970)
	

Table de vérité	Chronogramme
	

b	a	$S = \overline{a+b}$
0	0	1
0	1	0
1	0	0
1	1	0

OU-NON à 2 entrées

Fonction_OU-NON.DSN
Fonction_OU-NON.pdsprj



Porte OU_NON à 3 entrées

- Équation : $S = \overline{a + b + c}$ (lire S égal a OU b OU c le tout barre)

  La sortie S est au niveau bas, si a = 1 **OU** b = 1 **OU** c = 1 , sinon S est au niveau haut

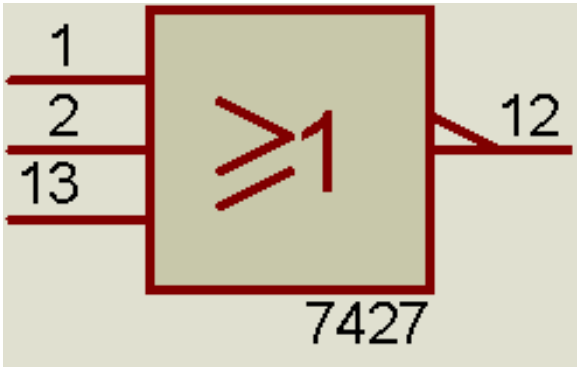
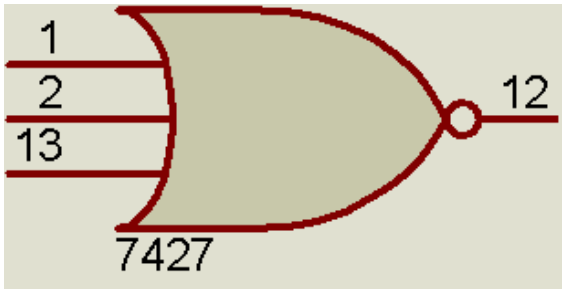
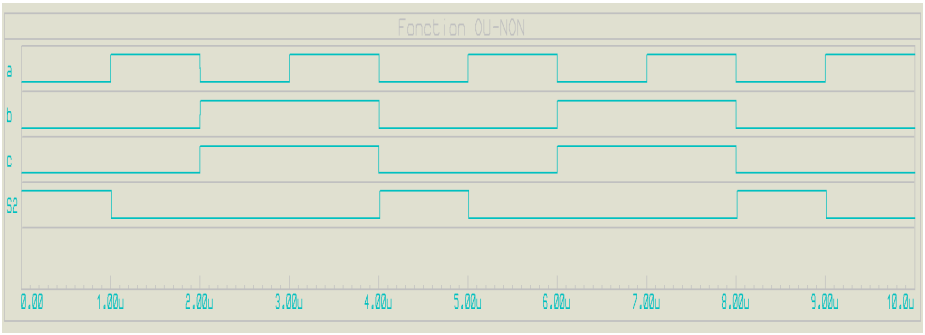
Symbole électrique (IEEE/IEC)	Symbole électrique (< 1970)
	

Table de vérité	Chronogramme

c	b	a	$S = \overline{a+b+c}$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

OU-NON à 3 entrées



Remarque:
La fonction OU-NON, est aussi une fonction universelle, c'est à dire que les fonctions OUI, NON, OU, ET, ET-NON peuvent être réalisées avec uniquement des opérateurs OU-NON.
Ainsi toute fonction même complexe peut être réalisée avec uniquement des portes NOR.

Exemples

a) Réaliser une porte NON à l'aide d'une porte OU-NON
b) Réaliser une porte Et à 2 entrées à l'aide de portes OU-NON
c) Réaliser une porte OU à 2 entrées à l'aide de portes OU-NON
d) Réaliser une porte ET-NON à 2 entrées à l'aide de portes OU-NON

Fonction OU-Exclu

Fonction OU-Exclusif

(XOR, eXclusif OR)

- Équation : $S = a \oplus b$ $\quad S = \bar{a}.b + a.\bar{b}$ (lire S égal a OU exclusif b)
- S reproduit la somme des variables d'entrée a et b.

La sortie S est au niveau haut, si l'une des deux entrées est à 1, sinon S est au niveau bas

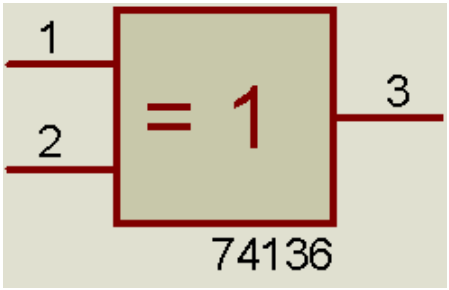
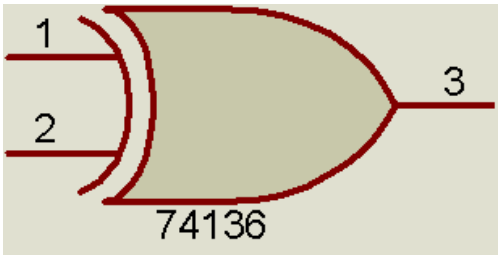
Symbole électrique (IEEE/IEC)	Symbole électrique (< 1970)
	

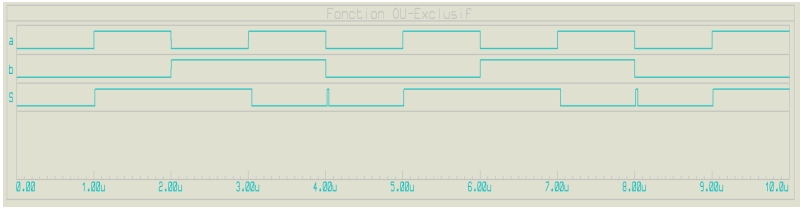
Table de vérité	Chronogramme

b	a	S = a ⊕ b
0	0	0
0	1	1
1	0	1
1	1	0

Remarque:

$S = a \oplus b$

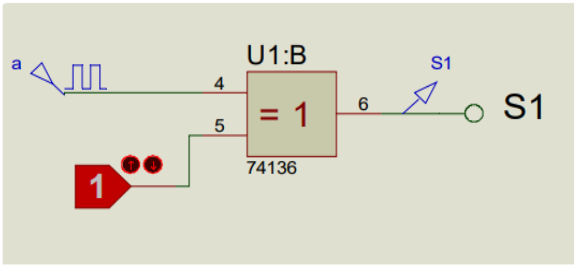
$S = (a \,.\, \bar{b}) + (\bar{a} \,.\, b)$



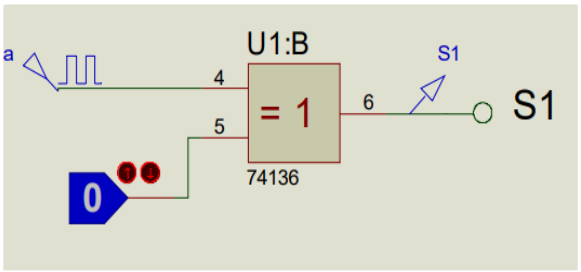
exercices

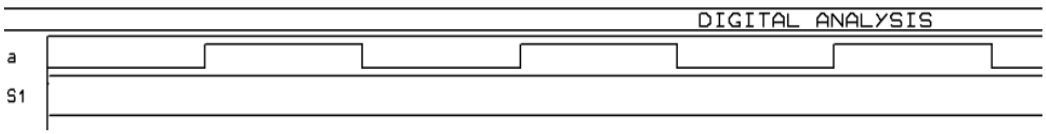
Compléter les chronogrammes suivants

montage a)

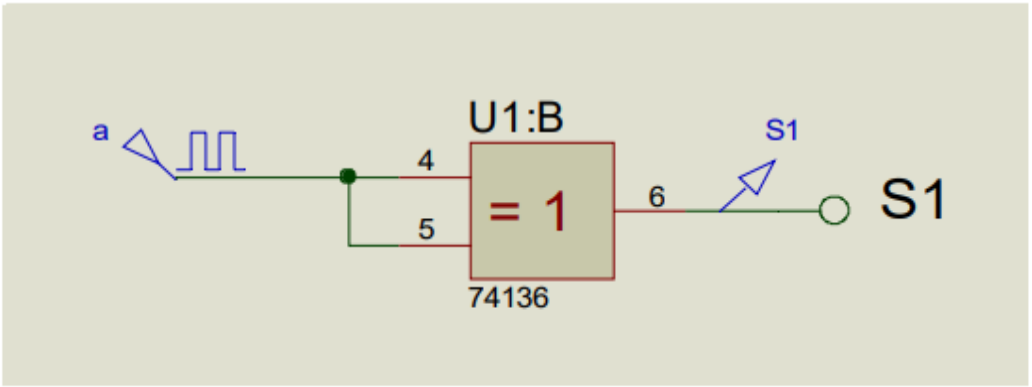


montage b)





montage c)



Relations logiques

Relations caractéristiques de la logique booléenne :

Sommes logiques	Produits logiques
$a + 1 = 1$	$a \cdot 1 = a$
$a + 0 = a$	$a \cdot 0 = 0$
$a + a = a$	$a \cdot a = a$
$a + \overline{a} = 1$	$a \cdot \overline{a} = 0$
$\overline{a + b} = \overline{a} \cdot \overline{b}$	$\overline{a \cdot b} = \overline{a} + \overline{b}$

Commutativité : $a \cdot b = b \cdot a$ $a + b = b + a$	**Associativité :** $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ $(a + b) + c = a + (b + c)$	**Distributivité :** $a \cdot (b + c) = a.b + a.c$ $a + (b \cdot c) = (a+b) \cdot (a+c)$
Théorème d'inclusion $a \cdot b + a \cdot \overline{b} = a$ $(a+b) \cdot (a+\overline{b}) = a$	**Théorème d'allègement** $a \cdot (\overline{a} + b) = a \cdot b$ $a + \overline{a}.b = a + b$	**Théorème d'absorption** $a \cdot (a + b) = a$ $a + a.b = a$

Théorèmes d'Augustus De Morgan :

Le complément d'un produit logique de variables est égal à la somme logique des compléments de variables.

$$(\overline{a \cdot b}) = \overline{a} + \overline{b}$$

$$(\overline{a \cdot b \cdot c}) = \overline{a} + \overline{b} + \overline{c}$$

Le complément d'une somme logique de variables est égal au produit logique des compléments de variables.

$$(\overline{a + b}) = \overline{a} \cdot \overline{b}$$

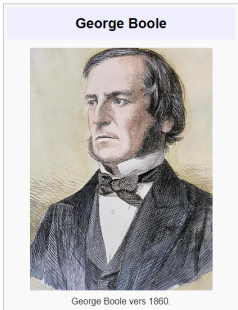
$$(\overline{a + b + c}) = \overline{a} \cdot \overline{b} \cdot \overline{c}$$

Bonus

$$a.\overline{b} + \overline{a}.b = (a + b) \cdot (\overline{a} + \overline{b})$$
$$\ldots\ldots\ldots = (a + b) \cdot (\overline{a.b})$$
$$\ldots\ldots\ldots = a.(\overline{a.b}) + b.(\overline{a.b})$$
$$\ldots\ldots\ldots =$$

ou comment faire un OU-exclusif avec 4 portes NAND

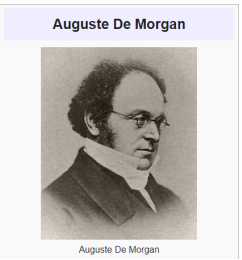
Annexe



George Boole, né le 2 novembre 1815 à Lincoln (Royaume-Uni) et mort le 8 décembre 1864 à Ballintemple (Irlande), est un logicien, mathématicien et philosophe britannique. Il est le créateur de la logique moderne, fondée sur une structure algébrique et sémantique, que l'on appelle algèbre de Boole en son honneur.

Source WIKIPÉDIA L'encyclopédie libre

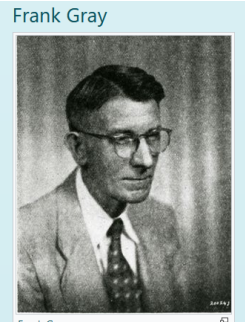
https://fr.wikipedia.org/wiki/George_Boole



Auguste (ou Augustus) De Morgan (27 juin 1806 à Madurai (Tamil Nadu) - 18 mars 1871) est un mathématicien et logicien britannique, né en Inde.
Il est le fondateur avec George Boole de la logique moderne ; il a notamment formulé les lois de De Morgan.

Source WIKIPÉDIA L'encyclopédie libre

https://fr.wikipedia.org/wiki/Auguste_De_Morgan

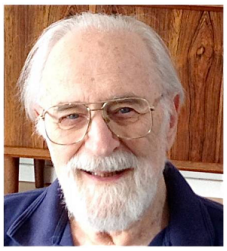


Frank Gray (13 septembre 1887, Alpine dans l'Indiana – 23 mai 1969), physicien et chercheur américain aux Laboratoires Bell.

Il est à l'origine de plusieurs innovations en télévision, tant d'un point de vue mécanique qu'électronique, pour avoir déposé, en 1953, un brevet concernant un ancien code binaire qu'il avait nommé « code binaire réfléchi » et que les autres chercheurs baptisèrent Code de Gray.

Ce code de Gray, souvent utilisé en électronique, a également de nombreuses applications en mathématiques.

https://fr.wikipedia.org/wiki/Frank_Gray_(physicien_et_chercheur)



Maurice Karnaugh (4 octobre 1924 à New York), est un ingénieur en télécommunications.

Il a développé la Table de Karnaugh aux laboratoires Bell en 1953. De nationalité américaine, c'est un spécialiste de physique et mathématique logique (algèbre de Boole).

https://fr.wikipedia.org/wiki/Maurice_Karnaugh

https://en.wikipedia.org/wiki/Binary_number
https://en.wikipedia.org/wiki/Numeral_system