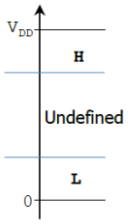
Note de COURS
3 : Technologie

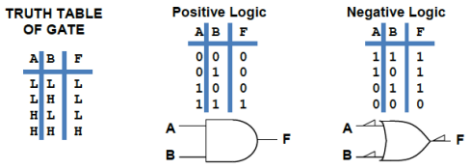
Niveau Logique



- Les valeurs logiques sont représentées par VRAIE OU FAUX. Dans les circuits numériques ces valeurs sont représentées en utilisant des tensions : H pour High ou VRAIE, et L pour Low ou FAUX.
- La valeur de V_{DD} dépend des I/O standard. Par exemple : TTL (5V), LVTTL (3.3V), LVCMOS(2.5V), LVCMOS(0.7V)
- Il existe 2 façons d'assigner les valeurs '1' et '0' :
  - **Logique positive** : le symbole '1' correspond à H et '0' à L
  - **Logique négative** : le symbole '0' correspond à H et '1' à L

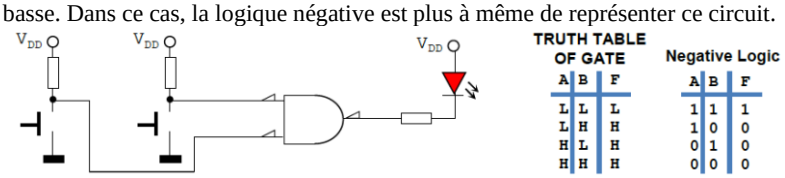
Exemple :

- Si l'on considère le circuit logique avec la table de vérité suivante, il correspond à une porte ET en logique positive et à une porte OU en logique négative. Pour différencier la logique négative, on positionne des petits triangles sur les entrées et sorties des portes.

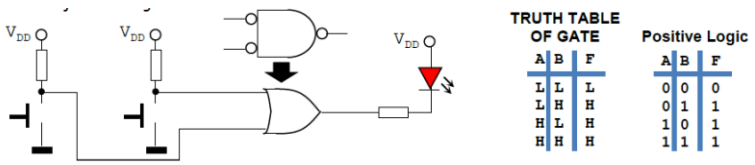


- Les entrées et sorties peuvent être activées par des valeurs Haute (H) ou Basse (L). Dans ce cas la distinction entre la logique positive et négative est importante.

- Supposons que l'on souhaite un circuit qui active une LED lorsque les entrées sont activées. Les entrées (switches) et sorties (LED) sont actives en valeur basse. Dans ce cas, la logique négative est plus à même de représenter ce circuit.



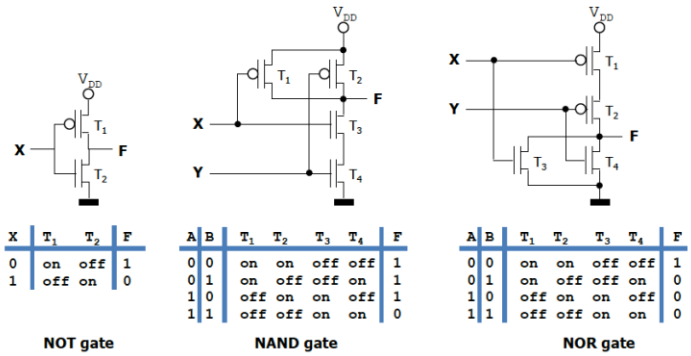
Si l'implémentation est basée sur de la logique positive, il suffit de remplacer les triangles par des inverseurs, et l'on a plus besoins que d'une porte OU.



- Il est courant de rencontrer une grande variété de ports d'E/S dont certains sont actifs en valeur basse et d'autre en valeur haute. Pour passer de portes logiques actives en valeur basse à des portes actives en valeur haute, il suffit d'inclure des inverseurs et d'utiliser la logique positive.

Porte logique CMOS

- CMOS (Complementary Metal Oxide Semiconductor) : ce type de porte repose sur l'utilisation de transistors PMOS et NMOS.
- On peut implémenter les fonctions logiques en utilisant juste des transistors. C'est une façon économique de traiter les fonctions, cependant pour la conception de gros circuit, on utilise des bibliothèques de portes logiques.
- **Exemple :** Porte NON, NAND NOR



X	T_1	T_2	F
0	on	off	1
1	off	on	0

A	B	T_1	T_2	T_3	T_4	F
0	0	on	on	off	off	1
0	1	on	off	off	on	1
1	0	off	on	on	off	1
1	1	off	off	on	on	0

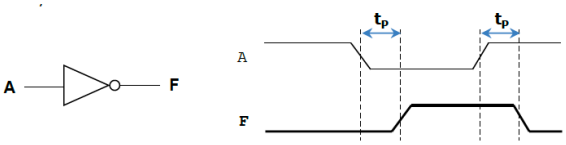
A	B	T_1	T_2	T_3	T_4	F
0	0	on	on	off	off	1
0	1	on	off	off	on	0
1	0	off	on	on	off	0
1	1	off	off	on	on	0

Aspects pratiques :

- Les circuits logiques sont des circuits analogiques !

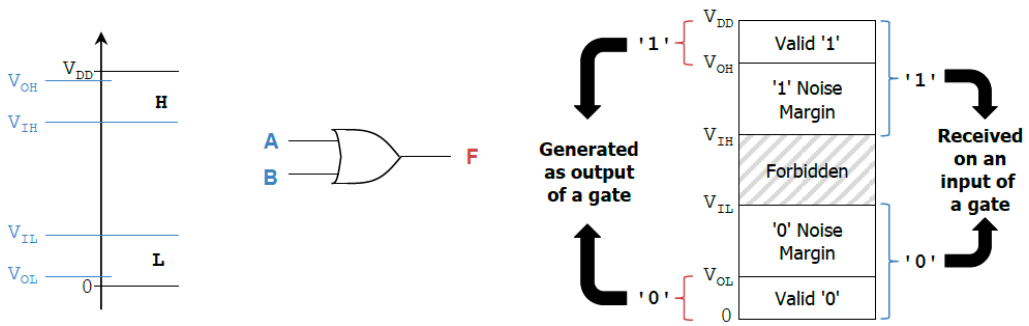
Temps de propagation :

- **t_P** : Temps de propagation



Marges de bruit

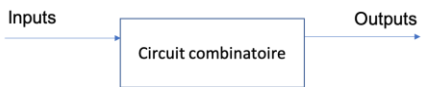
- Définition des tensions : Les tensions V_{IL}, V_{IH} sont supposées être générées par la sortie d'autre portes.
  - **V_{OH}** : Tension produit par la porte lorsque la sortie est Haute
  - **V_{OL}** : Tension produit par la porte lorsque la sortie est Basse (Low)
  - **V_{IH}** : Tension en entrée maximum que la porte interprètera comme Haute
  - **V_{IL}** : Tension en entrée minimum que la porte interprètera comme Basse (Low)
- **Marge au bruit** : Tolérance d'une porte au bruit. Les circuits électroniques sont en permanence sujet à des perturbations aléatoires, appelées bruits, qui peuvent altérer la tension en sortie de porte. Cependant, ce bruit ne doit pas perturber le fonctionnement des portes qui reçoivent ce signal bruité.
  - **Marge de bruit basse :** $N_{ML}=V_{IL}-V_{OL}$
  - **Marge de bruit haute :** $N_{MH}=V_{OH}-V_{IH}$

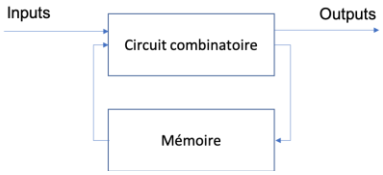


Classification des circuits numériques

- **Circuits combinatoires** : C'est un circuit dont les valeurs en sorties dépendent uniquement des valeurs en entrées. Les circuits résultant de l'implémentation de fonctions booléennes appartiennent à cette catégorie.

- **Circuits Séquentiels** : Un circuit logique séquentiel est un circuit logique possédant des entrées et des sorties et présentant un comportement où les sorties ne dépendent pas seulement des entrées, mais également des séquences des entrées passées. Les circuits séquentiels inclus des éléments de stockage afin de mémoriser les précédentes valeurs. Les circuits séquentiels peuvent implémenter une plus grande variété de circuit. Les circuits logiques séquentiels peuvent être groupés en deux grandes catégories : les circuits logiques séquentiels synchrones et les circuits logiques séquentiels asynchrones. La différence fondamentale entre ces deux catégories se situe dans l'introduction de la notion de temps dans les circuits. En effet, un circuit logique séquentiel synchrone incorpore une horloge qui sert à enclencher les actions, tandis que les circuits logiques séquentiels asynchrones n'en présentent pas.

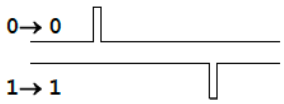




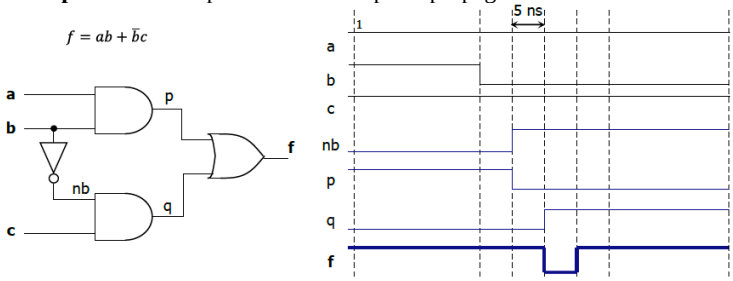
Circuit Combinatoire **Circuit Séquentiel**

Aléas

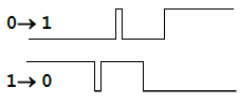
- Si on considère le chronogramme des signaux lors d'une transition dite de front montant (transition de 0 → 1), il apparaît que le signal en sortie prend la valeur 1 durant un très court laps de temps. On nomme cette valeur transitoire un aléa, *hazard* en anglais. Dans la pratique, un tel phénomène est également désigné par son nom anglais moins formel : *glitch*. Il existe 2 types d'aléas :
  - **Aléa statique** : Se produit lorsque les temps de propagation ne sont pas équilibrés. Ce problème peut être résolu en ajoutant tous les impliquants premiers d'une fonction. Ils se produisent lorsque les entrées changent alors que les sorties ne doivent pas changer. (2 types 0 => 0 et 1=>1).



Exemple : toutes les portes ont un temps de propagation de 5 ns.



  - **Aléa dynamique** : Ils sont le résultat de la structure du circuit, sont difficiles à détecter et à résoudre. Ils peuvent se produire à différents niveaux. Pour les éviter, on peut utiliser exclusivement des circuits à 2 niveaux et s'assurer qu'il n'y a pas d'aléa statique. (2 types : 1 => 0 et 0 => 1).



- **Implications :**
  - **Circuits combinatoires** : Les aléas ne constituent pas un problème car les sorties ne dépendent que des entrées, et ce du moment que le temps entres deux changements dans les entrées soit supérieur au temps de propagation dans le circuit ce qui est généralement le cas.
  - **Circuit asynchrones :** Ils sont très vulnérables aux aléas et peuvent rendre le circuit inutilisable.
  - **Circuit synchrones :** Ils ne posent pas de gros problèmes car ils reposent sur l'utilisation de registres.

Synthèse logique

- Consiste à partir d'une table de vérité ou d'une expression booléenne, à spécifier les opérateurs matériels permettant l'implémentation de la table ou de l'expression correspondante.

Logique « anarchique »

- Consiste à "implanter" la fonction booléenne à l'aide d'un ensemble minimal de portes de base.
- On commence par simplifier l'expression complète déduite de la forme disjonctive normale pour obtenir un nombre minimum de portes, avec un nombre minimum d'entrées pour ces portes.
- Utile uniquement pour des fonctions « simples » lié à la difficulté de simplifier les expressions booléennes.

Logique « structurée »

- Consiste "implanter" la fonction dans une structure régulière, prédéfinie à l'avance, et dont la surface ne dépendra pas de la configuration particulière des 0 et des 1 propre à une fonction, mais uniquement du nombre de variables d'entrées (structure ROM), ou du nombre de variables d'entrée et de termes produit de la fonction (structure PLA ou PAL).

Structure ROM

- Composé d'un décodeur (fonctionnement vue plus tard) et d'un OU logique des termes produits pour lesquels la fonction a pour valeur 1.
- La ROM est une mémoire à lecture seule.
- Le nombre de bits d'adresse correspond au nombre de variables des fonctions logiques et le nombre de bit en sortie correspond au nombre de fonctions logiques différentes implantées.
- **Inconvénient** : la ROM nécessite un décodeur complet alors que les fonctions logiques à grand nombre d'entrées n'utilisent généralement qu'un nombre réduit de termes produit : en d'autres termes, la fonction a beaucoup plus de 0 que de 1. Il est possible d'utiliser uniquement un décodeur partiel, ce qui est fait dans les réseaux logiques programmables (ou PLA)

Structure PLA et PAL

- Composé de 2 demi-PLA : Une matrice de ET et une matrice de OU.
- **Exemple :**
  - $s = \bar{a}b\overline{r_e} + \bar{a}br_e + a\bar{b}\overline{r_e} + abr_e$;
  - $r_s = ab + ar_e$;
  - $inf = \bar{a}b$;
  - $eq = \bar{a}\bar{b} + ab$;
  - $sup = a\bar{b}$

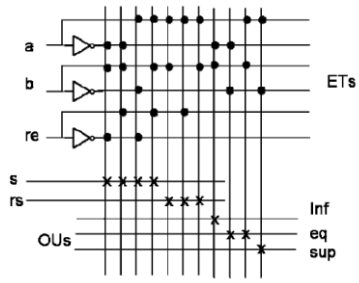
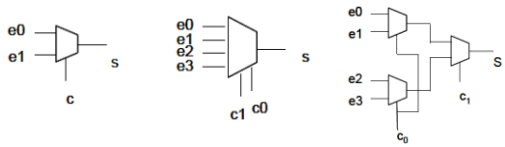


Figure 3 : Exemple de PLA

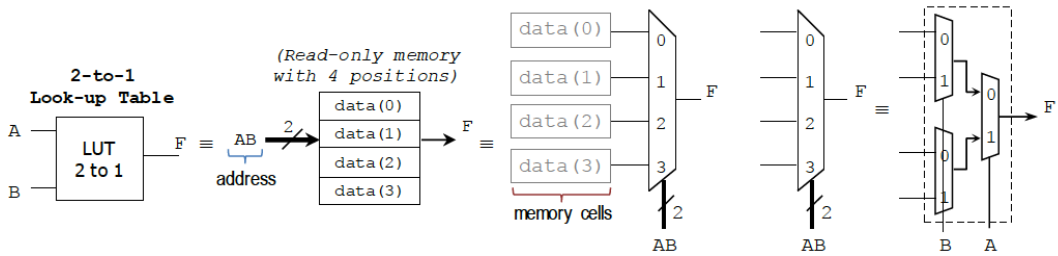
Multiplexeurs (MUX)

- C'est un opérateur logique à 2^n entrées et 1 sortie contrôlée par n fils de commande. L'entrée n° i est reliée à la sortie si la commande correspond à i codé sur n bits.
- **Exemples** :
  - Multiplexeurs à 2 et 4 entrées
  - $s = e_0\bar{c} + e_1c$; $s = e_0\overline{c_1}\overline{c_0} + e_1\overline{c_1}c_0 + e_2c_1\overline{c_0} + e_3c_1c_0$



Tables préprogrammées

- Basée sur des LUT (Look-up table) correspondant à des cellules mémoires.
- Permettent d'implanter des tables de vérité de 2, 3 ou 4 variables. (Les FPGA modernes ont par exemple des LUT à 6 entrées, Xilinx ARTIX-7)
- Une 2-LUT peut donc implanter n'importe laquelle des 16 fonctions logiques différentes à 2 entrées. Elle est donc équivalente à un MUX 4vers1 dont les entrées sont le contenus des cellules mémoires, elles même correspondant aux minterm de la fonction *f(A,B)*.



- 3 exemples de fonction dépendant du contenu des cellules mémoires :

	A	*B*	Minterms	F1	F2	F3
0	0	0	$data(0) = m_0 = \bar{A}\,\bar{B}$	0	1	0
1	0	1	$data(1) = m_1 = \bar{A}B$	1	0	1
2	1	0	$data(2) = m_2 = A\bar{B}$	1	1	1
3	1	1	$data(3) = m_3 = AB$	0	1	1

- La synthèse sur FPGA a une logique différente de la synthèse anarchique puisqu'il s'agit de minimiser le nombre de LUT.