

## Travaux Pratiques 1 - Logique combinatoire et séquentielle

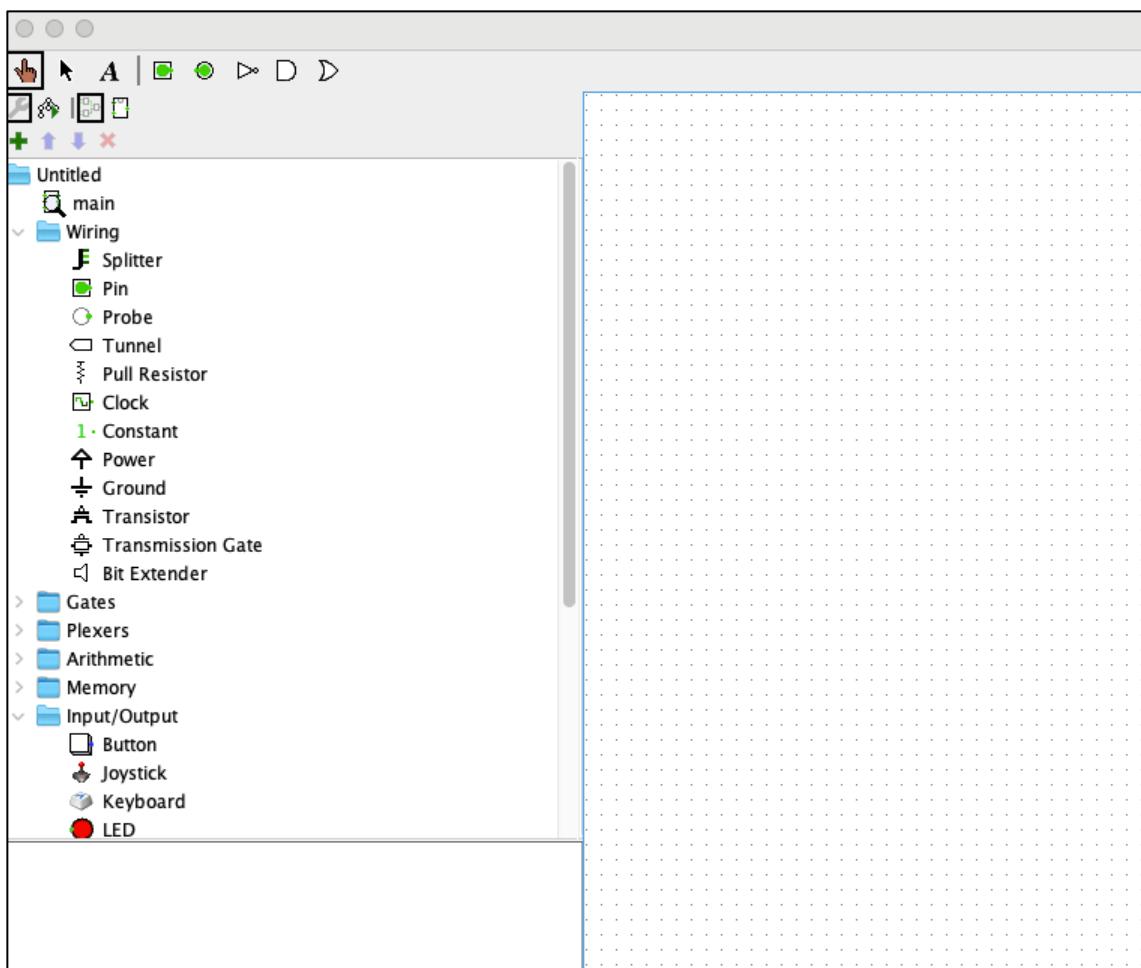
Pour ce TP nous utiliserons un logiciel permettant de concevoir des circuits numériques : LogiSim.

Vous pouvez utiliser ce logiciel depuis : Logisim ou un outil en ligne tels que <https://circuitverse.org/simulator>

### 1. Prise en main du logiciel :

Le logiciel comprend deux parties principales :

- La partie **Composant** (où l'on trouve les composants électroniques utilisables pour les circuits)
- La partie **Schéma** (où l'on assemble les composants).



Pour ce premier TP, nous utiliserons principalement les composants suivants :

### Les fils (wiring tool)

- Le fil de liaison permet la connexion des composants entre eux.

### Les niveau logiques (wiring) :

- « Power » fournit le niveau logique 1.  
1 = active = Vcc = 5V = ON = High  
Les fils au niveau logique 1 sont représentés en **vert**
- « Ground » fournit le niveau logique 0.  
0 = inactif = Masse = 0V = OFF = Low  
Les fils au niveau logique 0 sont représentés en **noir**

### Les boutons (Input/Output) :

- Le bouton poussoir (Button) : Ce type de bouton permet le passage du courant (Vcc) uniquement lorsqu'il est enfoncé. Dès qu'il est relâché, le circuit est interrompu.
- Le bouton à bascule (Switch) : Contrairement au bouton poussoir, ce bouton permet de maintenir le passage du courant (Vcc) après avoir été activé. Il reste en position une fois pressé jusqu'à ce qu'on le désactive manuellement.

### Les portes (Gates -> NON/AND/NAND/OR/NOR/XOR/XNOR) :

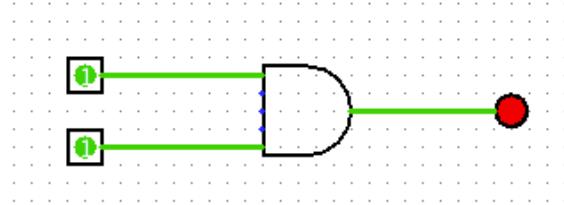
- **NON** : La sortie de cette porte est au niveau logique 1 si son entrée est au niveau logique 0. La sortie de cette porte est au niveau logique 0 si son entrée est au niveau logique 1.
- 1. **AND** : La sortie de cette porte est au niveau logique 1 si toutes les entrées sont au niveau logique 1. La sortie de cette porte est au niveau logique 0 si l'une des entrées est au niveau logique 0. Donc, en logique négative, une porte ET devient un OU.
- **NAND** : La sortie de cette porte est au niveau logique 0 si toutes les entrées sont au niveau logique 1. La sortie de cette porte est au niveau logique 1 si l'une des entrées est au niveau logique 0.
- **OR** : La sortie de cette porte est au niveau logique 1 si l'une des entrées est au niveau logique 1. La sortie de cette porte est au niveau logique 0 si toutes les entrées sont au niveau logique 0. Donc, en logique négative, une porte OU devient un ET.
- **NOR** : La sortie de cette porte est au niveau logique 0 si l'une des entrées est au niveau logique 1. La sortie de cette porte est au niveau logique 1 si toutes les entrées sont au niveau logique 0.
- **XOR** : La sortie de cette porte est au niveau logique 0 si les 2 entrées sont au même niveau logique. La sortie de cette porte est au niveau logique 1 si les 2 entrées sont à des niveaux logiques opposés.
- **XNOR** : La sortie de cette porte est au niveau logique 1 si les 2 entrées sont au même niveau logique. La sortie de cette porte est au niveau logique 0 si les 2 entrées sont à des niveaux logiques opposés.

NB : Pour lancer et arrêter la simulation appuyez sur le bouton de la simulation (Simulate -> Simulation Enabled)

## 2. Pratique :

Expérimenter pour chacune des portes logiques et visualiser leur sortie en utilisant une LED pour les portes logiques NON, AND, NAND, OR, NOR, XOR XNOR

Exemple avec la porte AND :



NB : pour changer l'état de votre bouton à bascule vous devez utiliser le « poke tool » dans base.



1. Pour chacune des portes, testez toutes les combinaisons possibles pour 2 et 3 entrées (sauf pour la porte NON qui prend une entrée uniquement) et remplissez les tables de vérité correspondantes. On appellera l'entrée 1 « X », l'entrée 2 « Y », l'entrée 3 « Z » et la sortie « S ».

**NON :**

X	S
0	1
1	0

**AND :**

X	Y	S
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	Z	S
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

**NAND :**

X	Y	S
0	0	1
0	1	1
1	0	1
1	1	0

X	Y	Z	S
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

**OR :**

X	Y	S
0	0	0
0	1	1
1	0	1
1	1	1

X	Y	Z	S
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

**NOR :**

X	Y	S
0	0	1
0	1	0
1	0	0
1	1	0

X	Y	Z	S
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

**XOR :**

X	Y	S
0	0	0
0	1	1
1	0	1
1	1	0

X	Y	Z	S
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

**XNOR :**

X	Y	S
0	0	1
0	1	0
1	0	0
1	1	1

X	Y	Z	S
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

2. En comparant les portes AND et NAND, ainsi que les portes OR et NOR, quelle porte logique devons-nous utiliser pour transformer une porte AND en NAND et une porte OR en NOR ?

Il suffit d'utiliser la porte NOT qui inverse les sorties  
.....  
.....

3. Faites les circuits, qui permettent de vérifier les théorèmes suivants :

- (a)  $A + A = A$
- (b)  $A \cdot A = A$
- (c)  $A + 1 = 1$
- (d)  $A \cdot 0 = 0$
- (e)  $A + (A \cdot B) = A$
- (f)  $A \cdot (A + B) = A$
- (g)  $A + (\bar{A} \cdot B) = A + B$
- (h)  $A \cdot (\bar{A} + B) = A \cdot B$
- (i)  $(\bar{\bar{A}}) = A$
- (j)  $\text{not}(A + B) = \bar{A} \cdot \bar{B}$
- (k)  $\text{not}(A \cdot B) = \bar{A} + \bar{B}$
- (l)  $A \oplus B = \bar{A} \cdot B + A \cdot \bar{B}$

4. Construire les circuits pour :

- $F(x,y) = (x + y) \cdot (x + z) \cdot (y + z)$
- $G(x,y,z) = (x \cdot y) + (x \cdot z) + (y \cdot z)$

5. Construire les circuits pour :

- $G(x,y,z) = x'y + xyz' + xyz$

Quelle est l'expression simplifiée de  $G(x,y,z)$  ?

$G(x,y,z) = y(x' + xz' + xz) = y(x' + x(z' + z)) = y$   
.....  
.....

6. Construire les circuits pour :

- $F(x,y,z) = y'z + xz$
- $G(x,y,z) = xy'z + x'y'z + xyz$

Que remarquer vous ?

Une partie des deux fonctions est similaire et la deuxième pourrait être simplifiée.  
.....  
.....