

# ÉLECTRONIQUE NUMÉRIQUE

TP 1 - Introduction

Logique combinatoire

Multisim

[charles.meunier@u-bourgogne.fr](mailto:charles.meunier@u-bourgogne.fr)

## OBJECTIFS

L'objectif de ce TP est de s'assurer que les aspects théoriques étudiés en cours se vérifient dans la pratique et de prendre en main le logiciel Multisim.

**Un compte-rendu détaillé vous sera demandé en fin de séance**

N'hésitez pas à poser vos questions à votre enseignant

### Notes

*Les actions à réaliser sont toujours précéder du symbole ➤.*

*Lorsqu'il vous sera demandé de réaliser un schéma, comprenez qu'il faudra réaliser un dessin propre sur votre compte-rendu.*

*Lorsqu'il vous sera demandé de réaliser un circuit, il vous faudra alors utiliser votre platine de tests et les composants nécessaires.*

## RETOUR AUX ORIGINES

On a déjà du vous le demander un paquet de fois, mais comme vous avez quatre heures devant vous, autant en profiter pour reprendre les choses à leur commencement et ensuite pouvoir avancer sereinement.

### NAND

- Commencez par énoncer la table de vérité de la porte logique NON-ET.
- Une fois ceci fait, étudiez la documentation du composant qui vous a été fourni pour le TP.
- **Tracez le schéma** du circuit vous permettant de vérifier par l'expérimentation la table de vérité précédente.
- **Réalisez** ce circuit et testez.

### Important

*Les broches des composants, contrairement à ce que l'on pourrait croire au premier abord, ne sont pas des pattes qui permettraient à la puce de se déplacer par elle-même. Par conséquent, inutile de tenter d'y trouver le moindre degré de liberté en les tordant dans tous les sens. Merci.*

### NOT

- On y retourne avec la table de vérité de l'inverseur (NOT).
- Toujours à partir du composant précédent, **tracez le schéma** puis **réalisez** le circuit vous permettant de tester le fonctionnement décrit précédemment.
- Testez.

**AND, OR, XOR**

Il est possible de faire de même avec les autres pour logiques comme OU et OU Exclusif. Mais vous avez compris le principe, donc nous ne vous demanderons pas de réitérer l'exercice avec ces portes. D'autant que vous connaissant, vous allez vous contenter d'aller chercher le schéma sur Internet...

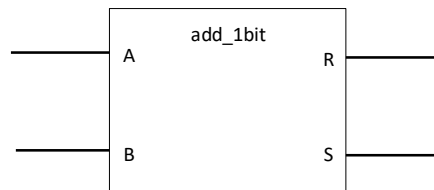
**CONCLUSION**

Vous venez de poser le doigt sur le concept de la logique combinatoire. On retrouve la même notion en programmation orientée objets où vous combinez des briques logicielles élémentaires pour réaliser des algorithmes complexes. En électronique, c'est la même chose et vos briques sont des portes logiques.

A présent, tentons quelque chose de plus "audacieux".

**ARITHMETIQUE****ETAPE 1**

Il est connu et reconnu que les élèves ingénieurs se vouent corps et âme à la discipline mathématique. C'est pourquoi nous allons vous donner l'occasion d'exprimer vos talents dans la conception d'un additionneur de deux mots... de 1 bit :

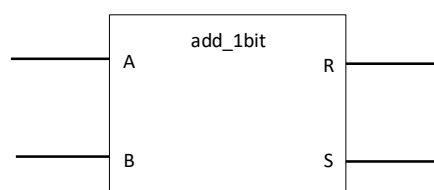


- Comme d'habitude, commencez par énoncer la table de vérité d'un tel dispositif. Vous expliquerez ce que représente chacune des entrées et sorties de cet additionneur.
- **Tracez le schéma** d'un circuit ayant la fonction décrite précédemment, uniquement à l'aide de composants utilisés précédemment.

**ETAPE 2**

Vous conviendrez aisément que faire la somme de deux mots de 1 bit peut être pratique mais que cela reste assez limité. C'est pourquoi nous vous proposons d'accéder à la cour des grands en réalisant un additionneur de... 2 mots de 2 bits !

Mais vous n'allez pas réinventer la roue et considérer que vous disposez de composants qui représentent des additionneurs de deux mots de 1 bit :



- A partir de ces composant `add_1bit` donnés et de portes logiques additionnelles, réalisez le schéma permettant de réaliser l'addition de deux mots de 2 bits.
- Quelle modification serait-il judicieux d'apporter au composant `add_1bit` ?
- Donnez la table de vérité de cette nouvelle version du composant `add_1bit`.
- Donnez les équations simplifiées des sorties R et S.
- **Tracez le schéma** permettant de réaliser une addition de deux mots de 4 bits à l'aide de cette nouvelle version de `add_1bit`.

Il ne reste plus qu'à tester que cela fonctionne bien. Et comme vous avez précédemment vérifié la solidité des principes de fonctionnement des portes logiques, nous vous proposons de délaissier temporairement les platines de tests au profit d'un outil de simulation informatique : Multisim

## SIMULATION

Afin de vous familiariser avec cet outil, vous allez réaliser la version évoluée de l'additionneur de 2 mots de 1 bit à partir des équations trouvées précédemment.

- Démarrez Multisim v11

### Note

*Il est possible que les fonctionnalités décrites ci-dessous diffèrent légèrement de celles disponibles dans votre version de Multisim, le sujet ayant été réalisé avec une version 14 du logiciel.*

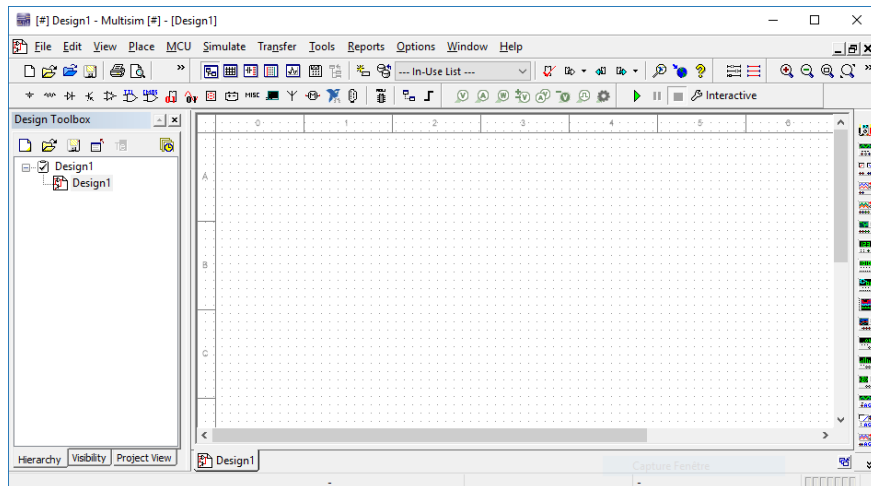
### Important


*Comme avec n'importe quel logiciel, pensez à sauvegarder régulièrement votre travail.*

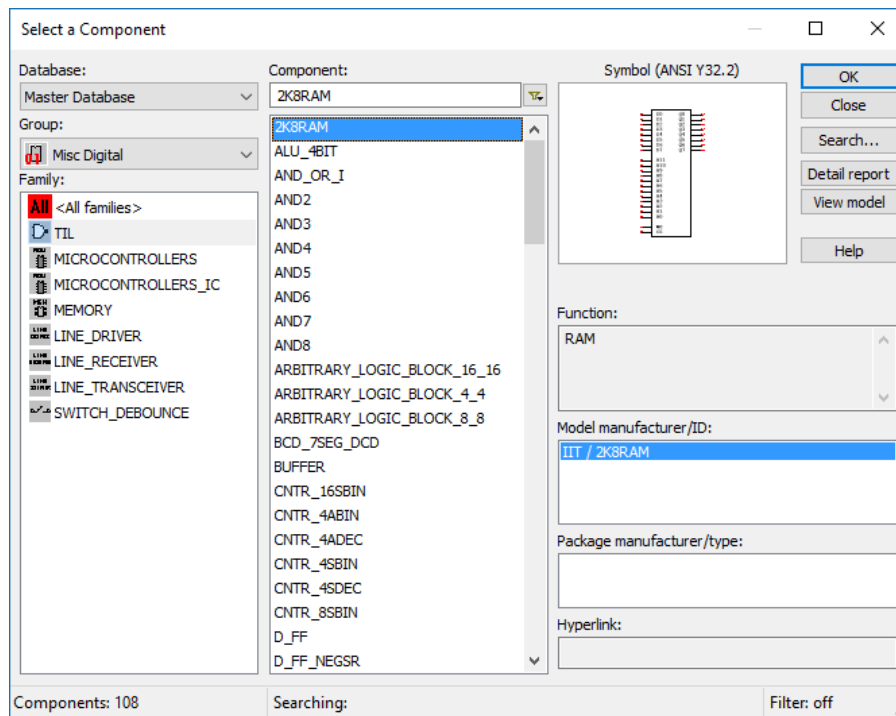
## MISE EN PLACE DES COMPOSANTS ET SIMULATION

Lorsque Multisim démarre, il crée automatiquement un nouveau schéma sur lequel vous allez pouvoir réaliser votre circuit.





- Dans la barre d'outils, cliquez sur "Place Misc Digital" (  )



- Sélectionnez la famille "TIL", puis le type de porte que vous souhaitez placer.
- Cliquez sur OK.
- Placez votre porte sur le schéma.
- Répétez l'opération jusqu'à ce que vous ayez placé toutes les portes nécessaires à la réalisation du circuit.
- Ajoutez une alimentation (Vcc) et une masse (Ground) via l'outil "Place Source"
- Ajouter deux interrupteurs (Place Source > Digital Sources > Interactive Digital Constant) qui permettront de faire varier les états des entrées de l'additionneur, ainsi que des leds (Place Indicator > Probe > Probe **DIG** XXX) pour voir l'état des sorties.
- Connectez tous les éléments.
- Testez le fonctionnement de votre circuit (Outil "Run" ou F5).

## SOUS-CIRCUIT

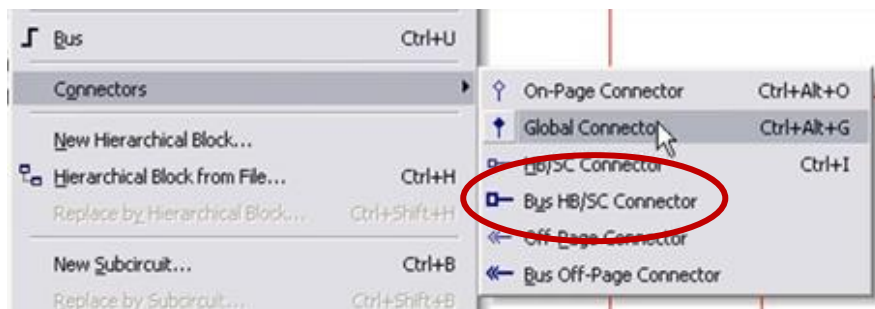
Nous l'avons vu plus tôt dans ce TP, la logique combinatoire permet d'associer des briques élémentaires afin de réaliser des dispositifs plus complexes. Il est également possible de combiner des dispositifs sophistiqués pour obtenir des éléments toujours plus élaborés. Comme en programmation orientée objets.

Multisim nous permet de faire cela grâce à la création de sous-circuits ou, comme il les appelle, des Blocs hiérarchiques.

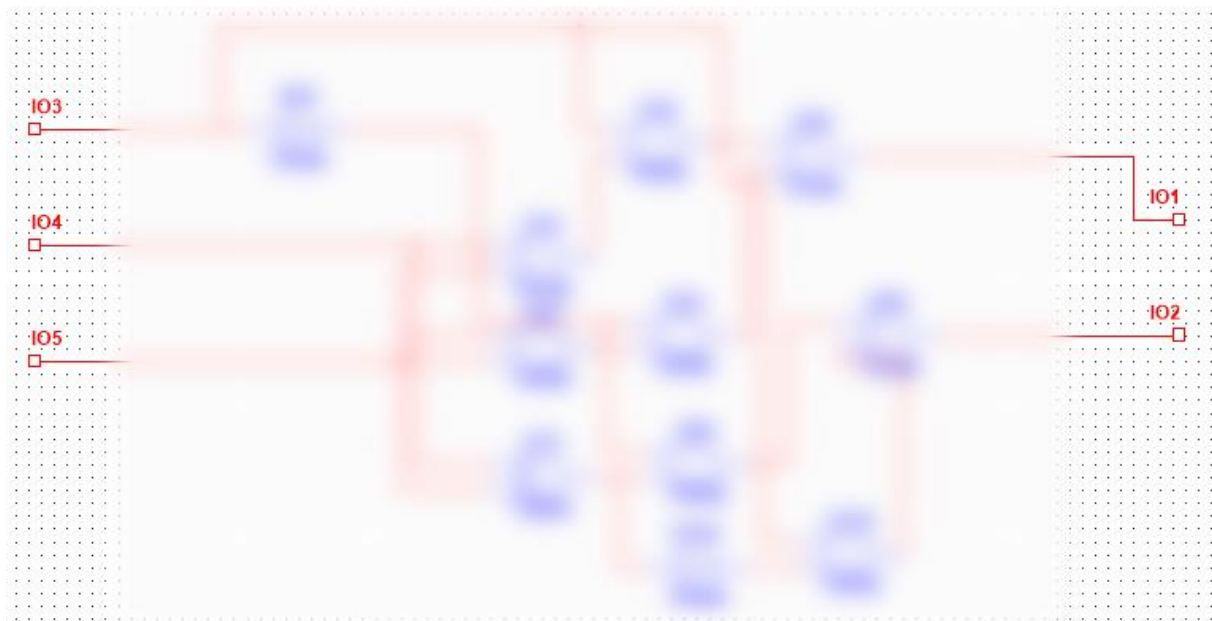
Dans la suite du sujet, vous allez créer un sous-circuit pour l'additionneur de 2 mots de 1 bit que vous venez de créer.

### Entrées/sorties

- Afin que la transformation s'opère de la meilleure façon possible, vous allez commencer par retirer l'alimentation et la masse.
- Puis, vous allez remplacer les switches et les leds par des connecteurs d'entrée/sortie (Menu Place > Connectors > HB/SC connectors).



Vous devriez obtenir quelque chose comme ceci :

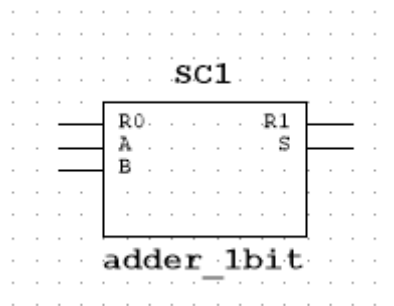


- A présent, faites un clic-droit sur chaque entrée/sortie et sélectionnez "Properties" dans le menu contextuel.

- Renommez le connecteur avec le nom de l'entrée ou de la sortie qu'il représente.
- Indiquez également la direction du connecteur : Input ou Output.

#### Création du sous-circuit

- Sélectionnez l'ensemble de votre schéma.
- Faites un clic-droit sur un des éléments sélectionnés.
- Sélectionnez "replace by hierarchical block" dans le menu contextuel.
- Saisissez le nom de votre composant : `adder_1bit`.
- Validez et cliquez sur votre schéma pour déposer votre composant où vous le souhaitez.



#### Réutilisation du composant

- A partir du sous-circuit réalisé précédemment, réalisez un additionneur de 2 mots de 4 bits.
- Testez.

## QUAND TOUT BASCULA...

Dans la première partie du TP, vous avez réalisé des circuits utilisant la logique combinatoire, logique dans laquelle l'état des sorties dépend directement de l'état des entrées.

Nous allons voir, à présent, qu'il est possible de briser cette dépendance.

### BASCULE RS

- **Tracez le schéma** de la bascule RS (à partir de porte NAND).
- **Réalisez le circuit** précédent et testez.
- Réalisez le même circuit en simulation avec Multisim.

### BASCULE D

- Sur un nouveau projet Multisim placez et testez une bascule D (utilisez le composant D-FF de Multisim).
- Expliquez son fonctionnement (à quoi correspondent les entrées et les sorties et ce qui se passe en fonction de l'état des entrées).

## LES MEMOIRES SE FORGERENT...

### REGISTRE 4 BITS

- A l'aide de bascules D vues précédemment, créez une mémoire permettant de conserver un mot de 4 bits.
- Créez un "Hierarchical block" à partir de ce schéma.

## ET LE TEMPS S'ECOULA...

Les bascules sont des pièces maîtresses de la logique séquentielle. Comme vu précédemment, elles permettent de mémoriser des états et donc de briser la dépendance directe entre les entrées et les sorties.

La gestion du temps devient possible et les machines à état ne sont plus très loin...

### COMPTEUR ASYNCHRONE

- Créez un nouveau projet Multisim
- A l'aide de bascules D, réalisez un compteur asynchrone sur 4 bits.
- Branchez un oscilloscope à quatre entrées sur votre schéma afin de visualiser les changements d'état de chaque bascule.
- Que constatez-vous ?
- Double-cliquez sur une des bascules D
- Dans l'onglet "Value" cliquez sur "Edit model"
- Remplacez tous les "1n" (1 nanoseconde) par "1m" (1 milliseconde)
- Testez à nouveau. Qu'observez-vous ?