

**Laboratoire de systèmes numériques / Systèmes logiques**

Auteurs : A. Lescourt, Prof. F. Vannel, Prof. A. Upegui (Hepia)

Modifications : Y.Saugy, S.Krieg, A.Convers, S.Masle (HEIG-VD). v.1.4

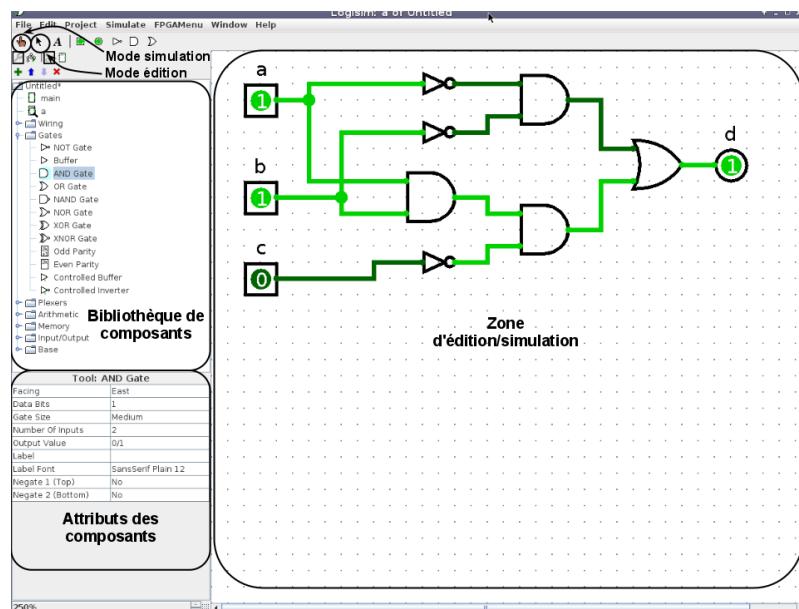
**Introduction à l'utilisation de Logisim**

## 1 Introduction

Logisim est un logiciel open-source permettant de concevoir et de simuler des circuits logiques<sup>1</sup>.

Ce document est un tutoriel qui décrit comment établir un système numérique à l'aide de cet éditeur de schéma. Nous expliquerons les démarches nécessaires afin de concevoir, simuler et implémenter un projet sur une carte Altera MAX\_V.

Il existe différentes façons de décrire formellement les systèmes numériques : des langages de description du matériel (HDL), des tables de vérité, des graphes d'états, ou des schémas. Logisim permet uniquement de travailler sur des schémas. Le premier chapitre expliquera comment réaliser un premier schéma. Vous pouvez voir l'interface de Logisim sur la Figure 1.



**FIGURE 1.** Interface de Logisim

1. La dernière version de Logisim peut être téléchargée à l'adresse <http://reds-data.heig-vd.ch/logisim-evolution/logisim-evolution.t.jar>. Logisim contient un mécanisme d'auto-update : dès qu'une nouvelle version sort, vous recevrez une notification au démarrage du logiciel et vous aurez la possibilité de le mettre à jour.

Une des particularités de Logisim est de pouvoir éditer et simuler un circuit en même temps. Nous expliquerons plus tard dans ce document comment simuler un circuit, puis comment l'implémenter sur la carte du laboratoire.

A l'ouverture de Logisim, une fenêtre demande l'introduction du nom d'utilisateur – voir Figure 2. Il sera inscrit dans chaque composant créé, afin d'empêcher le plagiat.



**FIGURE 2.** Fenêtre de login

Lors de la première utilisation, il faut ajouter un utilisateur comme en Figure 3 :

1. Cliquer sur Change user.
2. Introduire votre prénom et votre nom dans la case Add new user, sous la même forme que votre login Heig-vd, sans caractères spéciaux, séparé par un \_.
3. Cliquer sur Add.
4. Cliquer sur Close.
5. Cliquer sur Accept conditions.



**FIGURE 3.** Ajout d'un utilisateur

La liste des utilisateurs est enregistrée sur la machine. Pour les utilisations ultérieures, il suffit de cliquer sur Accept conditions pour sélectionner l'utilisateur actif ou de le sélectionner dans la liste.

## 2 Mode édition

1. Pour utiliser le mode édition, il faut simplement sélectionner la flèche comme indiqué en haut de la figure 1.
2. On peut alors choisir un composant dans la bibliothèque sur la gauche. Pour l'ajouter dans son schéma, il suffit de cliquer sur le composant désiré, puis de cliquer sur le schéma.
3. Chaque composant que vous utiliserez aura des attributs modifiables dans la zone inférieure gauche de Logisim. Par exemple, si l'on pose une porte AND, on peut modifier le nombre de signaux d'entrée, ou encore de mettre un inverseur sur une de des entrées.

4. Il est aussi possible de faire des copier/coller d'un ou plusieurs composants. Dans ce cas, les composants conserveront aussi tous les attributs préalablement définis.
5. Voici un descriptif des éléments que vous allez avoir besoin pour ce tutoriel :
  - Pour les entrées, l'élément Pin de Wiring.
  - Pour les sorties, l'élément Pin de Wiring avec l'attribut `output?=yes`.
  - Les portes logiques sont présentes dans le répertoire Gates.
  - Le splitter de Wiring.
  - Le ground et power de Wiring.
6. Une fois que l'on a posé tous les composants, il faut alors les connecter. Pour cela, il suffit de placer le curseur de la souris sur un des ports à connecter et, en gardant pressé le bouton gauche de la souris, le déplacer jusqu'au port de destination.

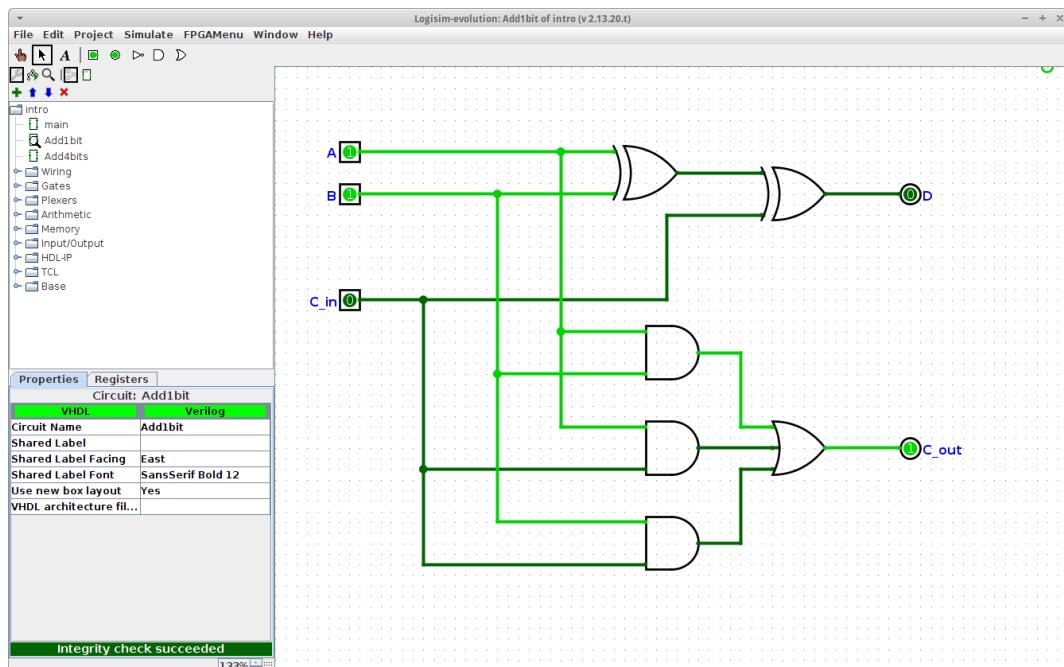
### 3 Création d'un premier circuit

Tous les circuits réalisés dans Logisim peuvent être réutilisés dans d'autres circuits. Afin de créer un nouveau circuit, il faut aller dans Project -> Add circuit... -> nommer le circuit. Le circuit créé devient un composant disponible dans la bibliothèque.

#### 3.1 Add1bit

Réaliser le schéma en Figure 4 et nommer le Add1bit. Remarques :

- Le circuit en cours d'édition est celui qui comporte une petite loupe en dessous du nom du projet.
- Ne pas prendre en compte la couleur des fils ni la valeur des Pin d'entrées (ces dernières sont un X bleu par défaut).
- Vous pouvez changer l'orientation des composants en modifiant l'attribut Facing.

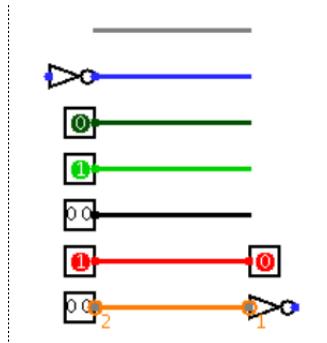


**FIGURE 4.** Additionneur 1 bit

## 4 Mode simulation

Logisim est capable de simuler le circuit en affichant les valeurs des signaux directement sur le schéma. L'utilisateur peut alors définir les valeurs des bits en entrée et observer la comportement du design.

1. Pour utiliser le mode simulation, il faut sélectionner la main en haut à gauche de Logisim (cf Figure 1)
2. Il est alors possible de contrôler l'état des différentes entrées en cliquant directement dessus. Le X bleu des Pin d'entrées représente l'état haute impédance. Dans ce laboratoire, nous travaillerons uniquement avec des états haut ou bas. Pour supprimer cet état de haute impédance, il faut modifier les attributs de ces Pin d'entrées de façon à ce que la ligne Three-state) soit égale à No.
3. En cliquant sur une entrée, la valeur doit alterner entre '0' ou '1'.
4. Voici un descriptif des couleurs utilisées pour les signaux en mode simulation.



**FIGURE 5.** Couleurs des fils en simulation

- **Gris** : La taille du fil est inconnue. Le fil n'est relié à aucune entrée ou sortie.
  - **Bleu** : Le fil comporte une valeur, cependant elle est inconnue.
  - **Vert foncé** : Le fil comporte la valeur '0'.
  - **Vert clair** : Le fil comporte la valeur '1'.
  - **Noir** : Le fil comporte plusieurs bits (BUS).
  - **Rouge** : Le fil comporte une erreur.
  - **Orange** : Les composants reliés au fil n'ont pas la bonne taille.
5. Tester le bon fonctionnement de votre additionneur 1 bit.

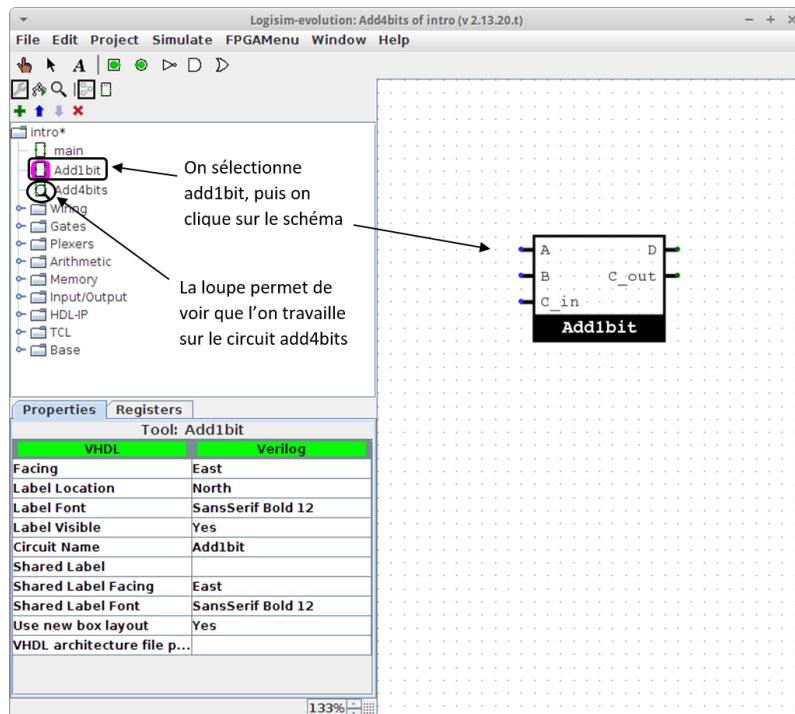
## 5 Design hiérarchique

La méthodologie de design que l'on vient d'utiliser est valable pour la conception de systèmes numériques relativement simples, c'est-à-dire avec un nombre de portes logiques plutôt faible. Lorsque l'on vise des systèmes plus complexes, le risque de voir le nombre de portes et de connexions augmente rapidement. Dans ce cas, l'introduction d'erreurs devient très importante.

La clé pour gérer correctement une complexité plus grande est d'utiliser le design hiérarchique. Grâce au design hiérarchique, on peut travailler à différents niveaux d'abstraction. D'abord on décrit des blocs de base à l'aide des portes logiques, pour ensuite utiliser ces blocs de base comme parties d'un système plus large. Dans le cas de notre additionneur 4 bits, l'utilisation de quatre additionneur 1 bit sera nécessaire. Ce nouveau bloc pourra aussi être utilisé dans autre système plus grand.

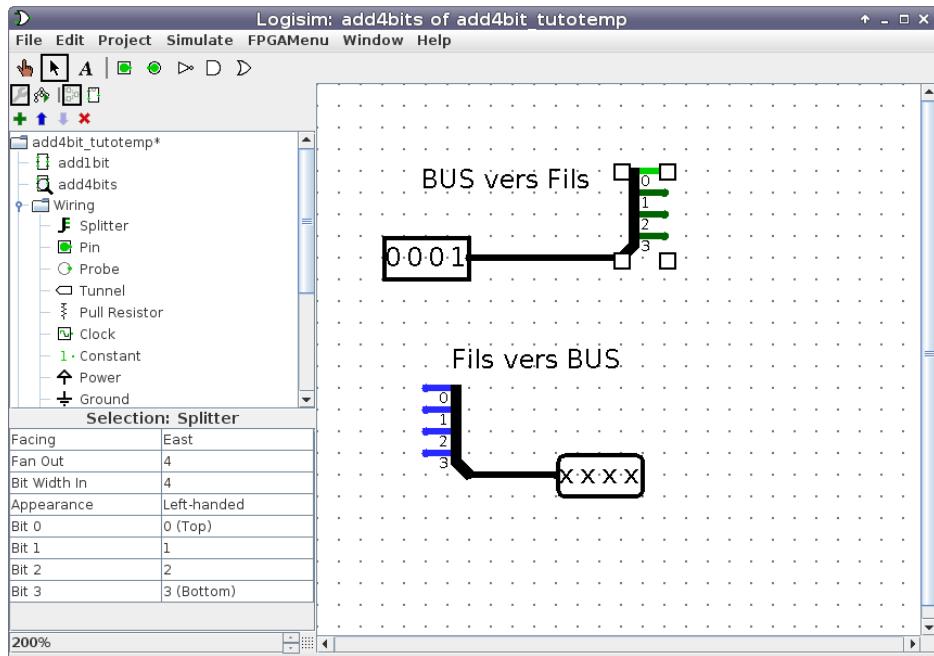
Pour créer un design hiérarchique, en incluant l'additionneur 1 bit que l'on a déjà conçu, il faudra suivre les pas suivants :

- Créer un nouveau circuit comme déjà expliqué dans la section 3 et nommer le Add4bits. Pour passer de l'édition d'un circuit à l'autre, il suffit de double-cliquer sur le nom de celui désiré dans le menu de gauche.
- Il est alors possible d'ajouter un sous circuit add1bit de la même manière que l'utilisation d'un composant quelconque. On clique sur Add1bit dans le menu indiqué sur la Figure 6, puis on le place en cliquant sur le circuit.
- Si le circuit Add1Bit a été créé correctement, alors il devrait être représenté par un petit bloc, avec sur sa gauche trois points bleus correspondant aux entrées et deux points verts sur sa droite correspondant aux sorties.



**FIGURE 6.** Sous circuit

- Si les sorties apparaissent en bleu et non en vert sur le schéma, vérifier que vous avez bien affecté l'attribut `output? = yes` dans les pins de sorties.
- Pour l'implémentation de l'additionneur 4 bits, il vous faut 4 additionneurs 1 bit, donc compléter le schéma en incluant les ports d'entrée et sortie. Les différences entre les circuits pour les additionneurs 1 et 4 bits, sont les entrées et les sorties et plus précisément leurs tailles, l'un avec des fils et l'autre avec des bus. Par exemple, pour définir l'entrée A comme un bus de 4 bits, il faut ajouter un élément `pin` et définir sa taille via l'attribut `Data bits = 4`.
- Lorsque l'on tire un fil de l'une de ces entrées, ce n'est plus un simple signal mais un bus de 4 bits. Pour pouvoir connecter les éléments de ce bus aux entrées des additionneurs 1 bits, on va devoir séparer les différents fils du bus afin de pouvoir les traiter un par un. L'élément `splitter` de `wiring` permet d'effectuer ces conversions dans les deux sens : d'un bus de 4 bits vers 4 fils, et de 4 fils vers un bus de 4 bits – voir Figure 7.



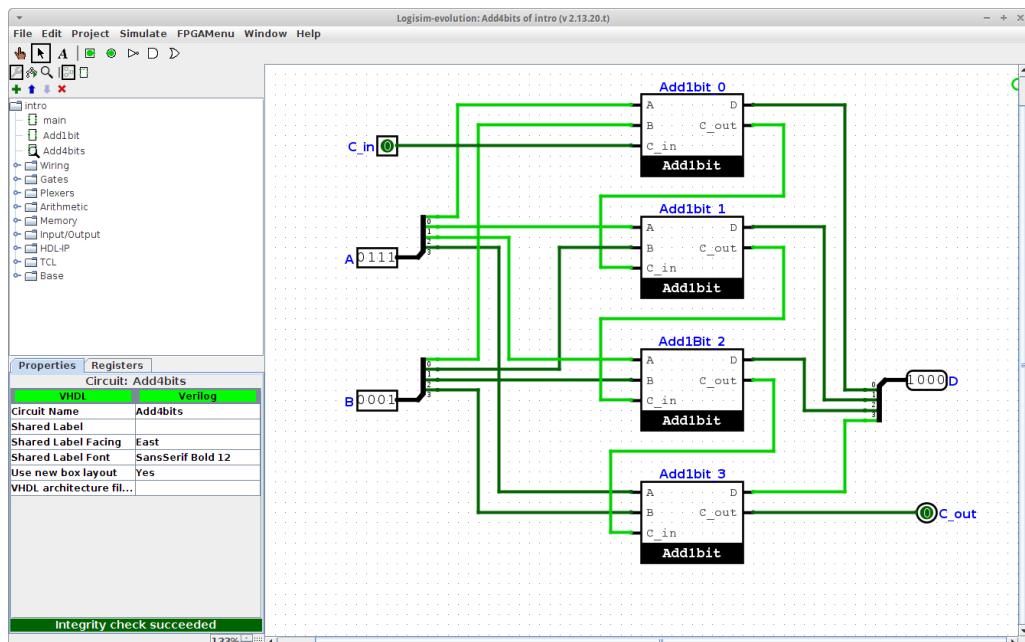
**FIGURE 7.** Exemples splitters

Il faut définir les tailles d’entrées et de sorties du splitter via les attributs `Fan out` et `Bit Width In`. Dans notre cas on définit les deux valeurs à 4.

Note : Le bit de poids faible est indexé à 0 en sortie du splitter.

## 6 Additionneur 4 bits

Réaliser l’additionneur 4 bits en Figure 8, puis vérifier son bon fonctionnement en simulation. Afin de faciliter la lecture des valeurs d’entrée/sortie, il est possible de changer le radix (binaire / octal / décimal) dans les options de la pin.



**FIGURE 8.** Additionneur 4 bits

## 7 Test avec un testbench

Un testbench (ou banc de test) est un outil permettant de vérifier de manière automatique un design spécifique. Un testbench ne teste que les cas pour lesquels il a été prévu. Ceci implique que si le testbench ne vous retourne pas d'erreur, cela ne signifie pas de manière irréfutable que le design testé est sans faille.

Le testbench se situe sur un serveur interne à la HEIG-VD, et nécessite donc d'être dans l'enceinte de la HEIG-VD ou connecté au VPN. Le nom du fichier .circ a fournir au testbench doit **impérativement** être le même que celui fourni, il **ne faut pas le modifier** sous peine de voir le testbench ne rien faire. Le nom des composants déjà fournis dans le .circ ainsi que leurs entrées/sorties ne doivent pas non plus être modifiés car cela posera problème lors de la compilation du testbench.

Voici le protocole pour utiliser le testbench :

- Etre à la HEIG-VD ou se connecter au VPN.
- Ouvrir la page web : <http://reds-calculator/logisim-validator/>.
- Sélectionner le fichier.circ à tester à l'aide du bouton "Choisir un fichier".
- La simulation doit débuter automatiquement.
- A la fin de la simulation, si le circuit fonctionne sans erreur, le message "Tests successfully passed" doit apparaître. Sinon la liste des erreurs s'affiche.
- Pour relancer une simulation, faire un rafraîchissement de la page web puis sélectionner de nouveau le circuit à tester.

Afin de tester que vous arriviez correctement à lancer une simulation, un testbench pour l'additionneur 4 bits du point précédent a été réalisé.

## 8 Programmation de la carte MAX\_V avec la console

Logisim permet de programmer directement la carte MAX\_V. Il faudra alors associer les Pin en entrées à des boutons, et ceux des sorties à des LEDs.

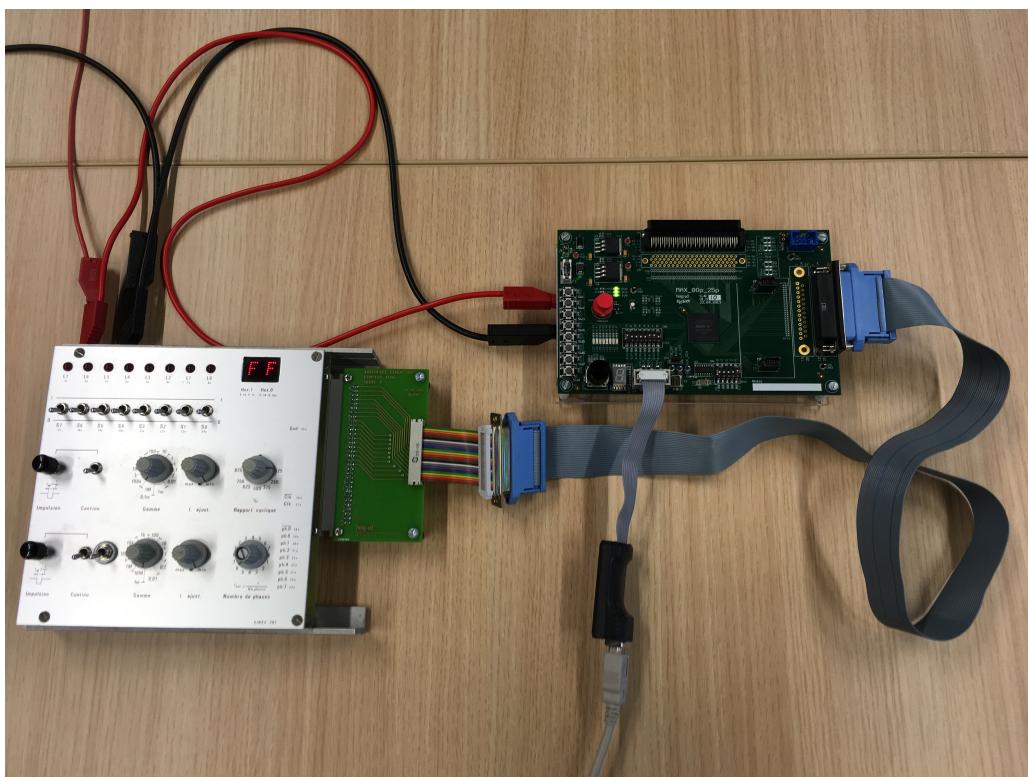
La partie programmation ne peut se faire uniquement qu'avec les ordinateurs mis à votre disposition dans les salles de laboratoire. Même avec Logisim installé sur votre ordinateur personnel, il est impossible de programmer une carte.

### 8.1 Connexion

La carte MAX\_V et la console doivent être alimentées avec une alimentation de 5 volts. La programmation de la carte MAX\_V se fait via le module USB-Blaster. Les éléments doivent être branchés comme sur la Figure 9.

**Matériel nécessaire :**

- Carte MAX\_V,
- Console,
- Carte d'interface de la console,
- Cable de liaison interface-MAX\_V,
- le module USB-Blaster,
- Cables d'alimentation.



**FIGURE 9.** Connexion de la carte MAX\_V avec la console

### 8.2 Configuration

1. Afin de pouvoir programmer la carte, il est nécessaire de nommer chaque entrée/sortie en leur affectant l'attribut Label. On utilisera les termes A, B et D respectivement pour les deux bus d'entrées et le bus de sorties. Les retenues seront nommées avec C\_in et C\_out.

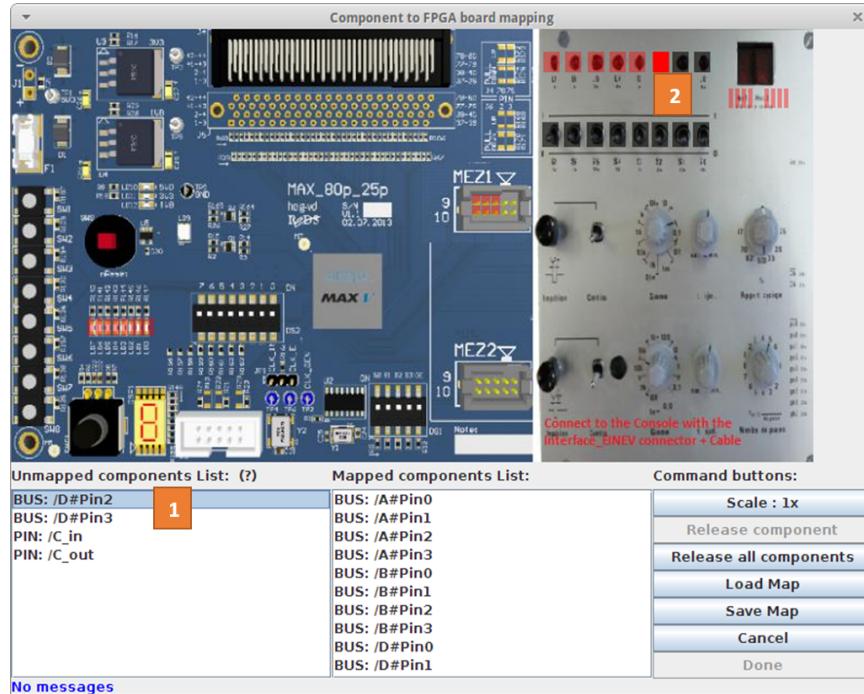
Il faut aussi annoter tous les composants avec des identifiants uniques. Les quatre additionneurs 1 bit auront alors pour label add1bit\_X où X est un numéro unique.

2. On peut ensuite ouvrir l'interface de programmation via FPGAMenu -> FPGA Commander comme sur la Figure 10.



**FIGURE 10.** *FPGA Commander*

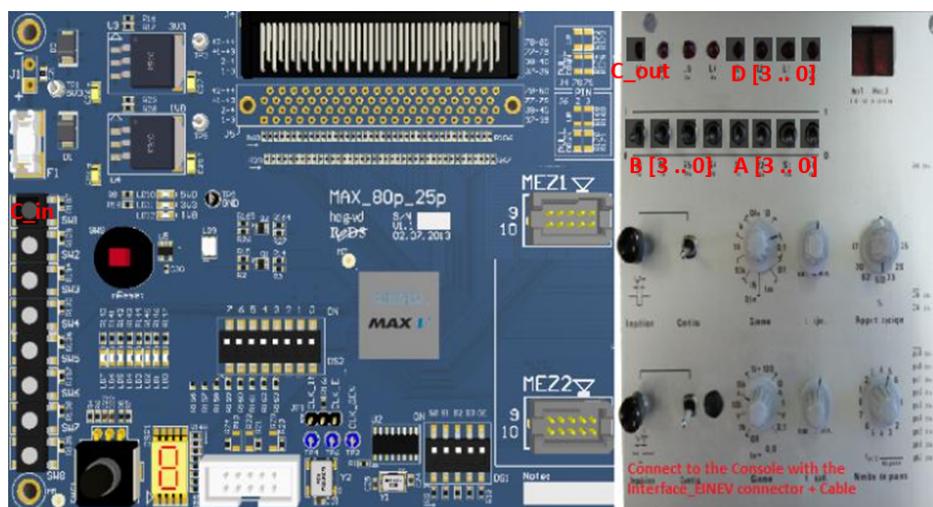
3. Plusieurs paramètres doivent alors être définis :
  - **Choose main circuit** : Le circuit top level que nous voulons programmer. Dans notre cas c'est Add4bits.
  - **Choose target board** : La carte que nous allons utiliser. Il faut choisir la carte du laboratoire MAX\_V\_CONSOLE.
  - **Toolpath** : Il faut s'assurer qu'il pointe vers /opt/EDA/altera/13.0/quartus/bin
  - **Workspace** : Il doit être défini vers /home/redsfuser/logisim\_workspace
  - Les autres paramètres sont laissés à leur valeur par défaut.
4. Afin de programmer la carte MAX\_V\_CONSOLE, Logisim va tout d'abord générer plusieurs fichiers .vhd. Il faut pour cela cliquer sur **Annotate**, puis sur le bouton **Download**. Tous les fichiers générés sont affichés dans l'onglet **Infos** du FPGA Commander.
5. Il faut ensuite associer les entrées/sorties de l'additionneur aux boutons et aux LEDs de la carte comme sur la Figure 11 et la Figure 12.



**FIGURE 11.** Console Board Mapping

6. Toutes les entrées/sorties que l'on a définies sont visibles dans la zone **Unmapped components list**. Si l'on veut par exemple mettre le bit 0 de l'entrée A sur l'interrupteur tout en bas à droite de la carte, il faut sélectionner PIN: /A<0>, puis cliquer sur l'interrupteur dans l'image. Si le mapping est possible, lorsque l'on passe le curseur sur l'interrupteur, ce dernier devient rouge. Pour les pins C\_in et C\_out, il faut double-cliquer sur la ligne pour passer du type pin\_in/out au type switch/led.
7. Placez les éléments de la manière suivante (**cf Figure 12**) :
  - Les 4 bits de l'entrée A sur les 4 premiers switches (S0 à S3).
  - Les 4 bits de l'entrée B sur les 4 switches suivants (S4 à S7).
  - L'entrée C\_in sur le bouton-poussoir (SW1).
  - Les 4 bits de la sortie D sur les 4 premières LEDs (L0 à L3).
  - La sortie C\_out sur la LED L7.

Une fois que toutes les entrées et sorties ont été définies, cliquer sur Done .



**FIGURE 12.** Position des entrées/sorties

- Après la compilation, une fenêtre apparaît pour demander à l'utilisateur si la carte est branchée. Brancher le connecteur usb du programmeur jtag au pc et le connecteur jtag à la carte. Allumer l'alimentation 5V puis cliquer sur Yes, download.  
Une fois le téléchargement terminé, l'onglet Errors de la fenêtre de FPGA Commander indique si il y a eu des erreurs.

## 9 Chronogramme

En plus de pouvoir simuler en temps réel un schéma logique, Logisim peut représenter cette simulation sous la forme d'un chronogramme. Si cette visualisation est peu adaptée aux systèmes purement combinatoires, elle est très pratique dans le cas des systèmes logiques séquentiels (utilisation d'horloges, ex : compteur, registres...). Afin de vérifier le fonctionnement de l'additionneur en fonction du temps, il faut ajouter un compteur au circuit.

1. Créer un nouveau circuit dans le projet.
2. Insérer votre additionneur 4 bits dans le nouveau circuit.
3. Ajouter un compteur (le compteur se trouve dans les composants Memory). Editer le paramètre `Data bits`, et fixer sa valeur à 4.
4. Câbler le compteur et l'additionneur comme indiqué sur l'image suivante (les constantes se trouvent dans `Wiring`).

Le circuit final doit ressembler à celui en Figure 13.

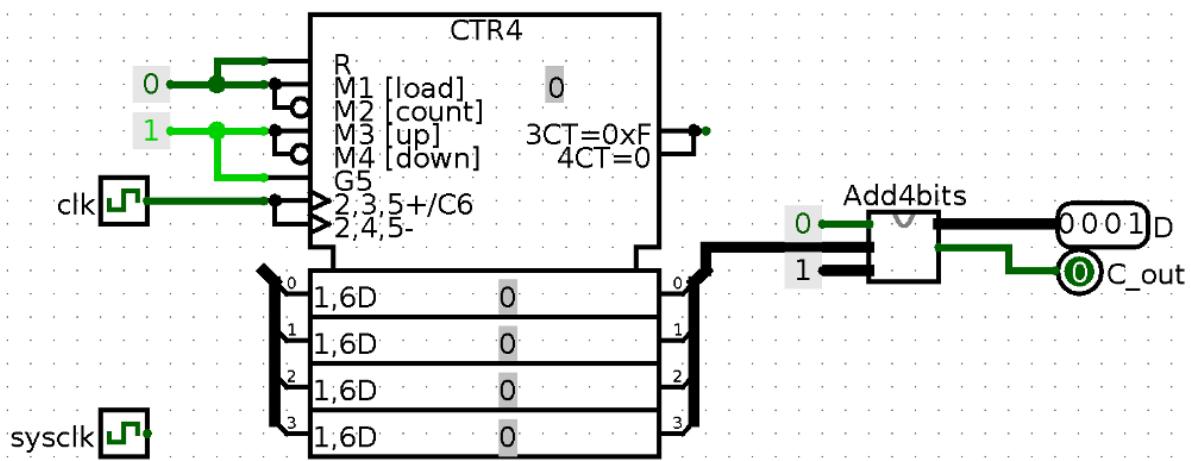


FIGURE 13. Ajout du compteur

### 9.1 Configuration du Chronogramme

Pour pouvoir afficher le chronogramme, il faut définir une fréquence d'échantillonnage qui indiquera au système le moment où il doit afficher les valeurs de chaque signal. Cette fréquence doit être défini par une horloge d'échantillonnage ajouté au circuit simulé, qui ne doit être reliée à aucun composant.

#### 9.1.1 Horloge d'échantillonnage

Pour créer l'horloge d'échantillonnage il faut :

- Aller dans le liste de composant, prendre `Wiring` -> `clock` et l'ajouter au schéma.
- Aller dans les attributs de cette horloge et la nommer `sysclk` (Attribut Label)
- Assurez-vous que les attributs `High Duration` et `Low duration` sont bien définis à un coup d'horloge.
- **Ne reliez cette horloge à aucun composants !**
- Pour définir la fréquence de cette horloge : `Simulate` -> `Tick Frequency`, prendre par exemple 4Hz.

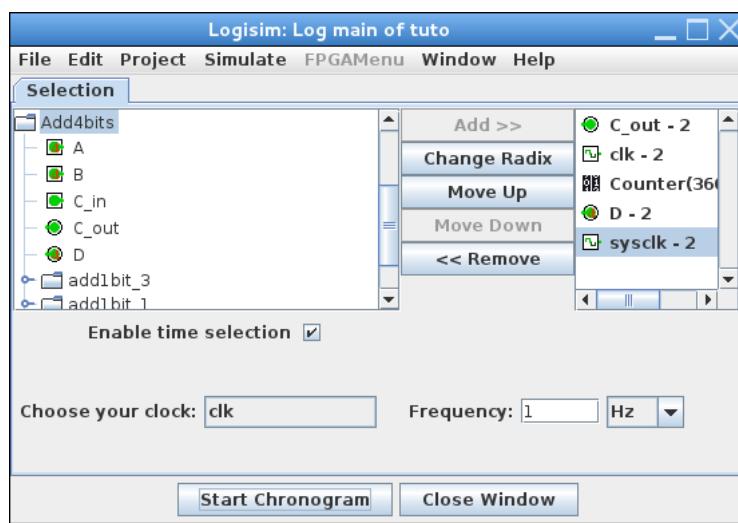
### 9.1.2 Ajout d'une horloge interne

Placer une nouvelle horloge nommée `clk`. Logisim nécessite le nom `clk` pour le chronogramme, un autre nom ne convient pas. L'horloge interne de votre système doit être plus lente que `sysclk`. Elle s'ajoute de la même façon que l'horloge d'échantillonnage : `Wiring -> clock`. Usuellement, on nomme l'horloge de son système `clk`.

Le chronogramme va rechercher l'horloge dans le main circuit. Pour changer votre circuit en main circuit, il faut effectuer un clic droit sur votre composant (dans la liste des composants de la colonne de gauche) puis sélectionner `Set As Main Circuit`. Pour définir sa fréquence, il faudra modifier ses attributs afin de régler combien de "ticks" cette horloge a besoin avant de changer d'état. Par exemple, si vous avez choisi une fréquence d'échantillonnage de 4Hz, mettre les attributs `High duration` et `Low duration` de votre horloge à 4 ticks. Votre horloge changera alors d'état une fois par seconde.

### 9.1.3 Sélection des signaux

Il faut ensuite définir quels signaux nous voulons afficher dans le chronogramme, comme en Figure 14.

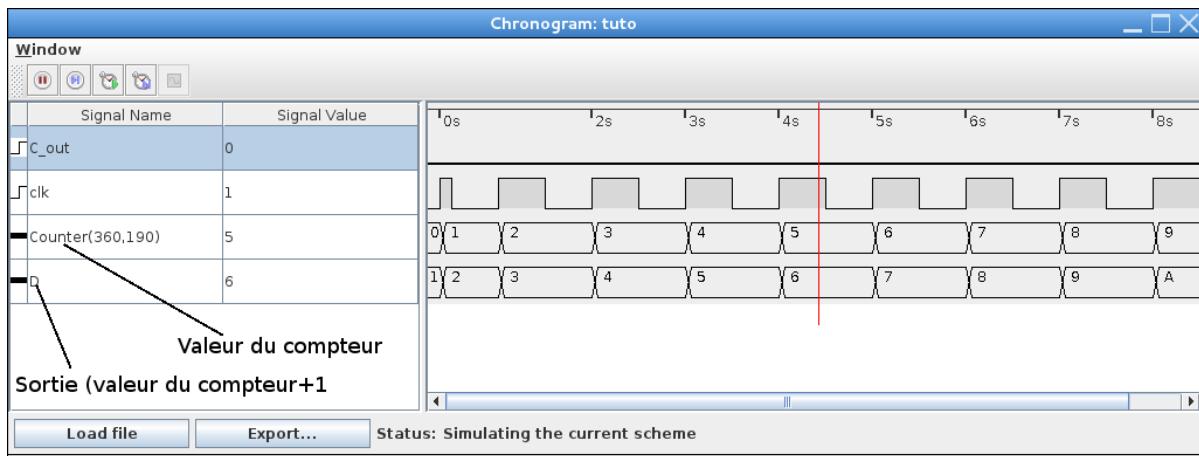


**FIGURE 14.** Sélection des signaux

- Ouvrir le menu `Simulate` puis sélectionnez `Chronogram`. La liste des signaux disponibles apparaît.
- Sélectionner les signaux que vous voulez et cliquez sur `Add >>`
- Il est indispensable que l'horloge d'échantillonnage `sysclk` soit aussi ajoutée.

Il est possible (mais pas indispensable) d'afficher une base de temps, selon la fréquence d'un signal d'horloge. Il faut activer la case `Enable time selection`, puis choisir l'horloge de votre système ainsi que sa fréquence.

Démarrer le chronogramme via le bouton `Start Chronogram`, puis organiser les fenêtres afin d'être capable de voir à la fois le schéma et le chronogramme. Le chronogramme final doit ressembler à celui en Figure 15.



**FIGURE 15.** Chronogramme

## 9.2 Utilisation du chronogramme

A chaque changement d'état de l'horloge sysclk, le chronogramme se mettra à jour.

Il est aussi possible de démarrer et d'arrêter l'horloge : Simulate -> Ticks Enabled (Raccourci : Ctrl+k). La touche F2 permet d'avancer en mode pas à pas.

Contrôles du chronogramme :

- Valeur à l'instant  $t$  : En cliquant sur un signal dans la partie droite du chronogramme, la valeur des données à cet instant sera affichée à côté du nom du signal.
- Utilisez la molette sur le chronogramme pour zoomer/dézoomer (focus sur la zone sélectionnée).
- Click droit sur un bus ->Format pour changer son format d'affichage.
- Click droit sur un bus ->Expand afficher chaque signal composant le bus.

### 9.2.1 Sauvegarde et chargement

Une fois la simulation terminée, vous pouvez exporter ce chronogramme dans un logFile via le bouton Export. Créer un fichier .txt ou .log à l'emplacement souhaité.

Le chargement d'un de ces chronogrammes se fait via le bouton Load External File. Dans ce cas, le chronogramme ne prend plus en compte les signaux du schéma courant. Il affiche simplement l'état des signaux tels qu'ils étaient lors de l'exportation. Par ailleurs, si vous voulez utiliser le chronogramme uniquement pour visualiser un fichier préalablement exporté, il faut simplement lancer le chronogramme avant de charger le fichier : (Simulate -> Chronogram -> Start Chronogram -> Load file).

## 10 Liens utiles

Logisim est un logiciel open-source. C'est à dire que les sources sont accessibles par tous le monde.

1. Le dépôt git est accessible à l'adresse :  
<https://github.com/red5-heig/logisim-evolution>
2. Une FAQ est accessible dans le wiki du dépôt  
<https://github.com/red5-heig/logisim-evolution/wiki/Logisim-evolution---usage-and-FAQ>