

Chapitre 1

Fonctions combinatoires

Les fonctions combinatoires sont indispensables au fonctionnement de tout système numérique. La table de vérité d'une fonction combinatoire constitue son ADN, elle fournit des informations claires sur son mode de fonctionnement.

En pratique, toute fonction combinatoire est représentée par des portes logiques simples ET, OU, INV. Pour utiliser le moins de portes possible, on passe par la table de KARNAUGH pour simplifier la fonction.

1.1 Additionneur 1 bit

On souhaite réaliser un additionneur 1 bit simple.

1. **Compléter** la table de vérité de l'additionneur :

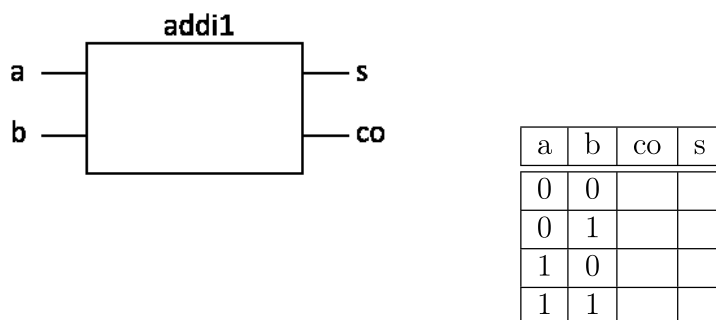


FIGURE 1.1 – Entité d'un additionneur 1 bit + table de vérité

2. En **déduire** l'équation logique des sorties s et co.

La réalisation pratique se fera sur un composant programmable de type FPGA. On utilisera le logiciel Logisim-Evolution pour décrire le circuit ou fournir la table de vérité. La table de vérité pourra être traduite en portes logiques par l'outil de synthèse. L'implémentation va générer un fichier de configuration qui, une fois chargé dans le composant, va générer les portes logiques nécessaires et les connecter comme il se doit pour réaliser l'additionneur.

1.1.1 Création d'un projet

- **Démarrer** le logiciel Logisim-Evolution. Le logiciel s'ouvre en créant un nouveau projet qu'il faut sauvegarder.
- *Fichier >> Enregistrer*

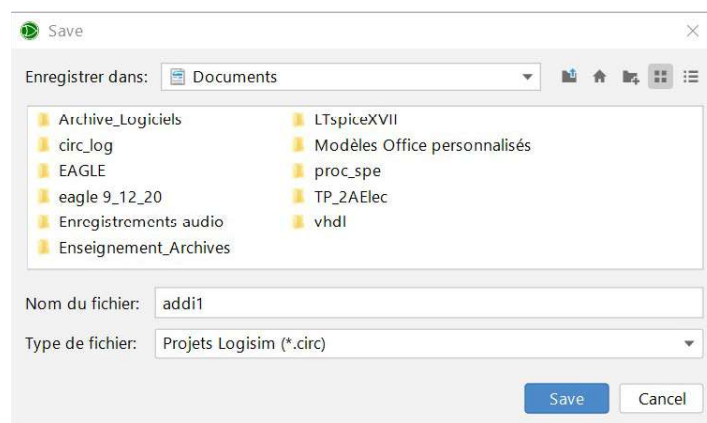


FIGURE 1.2 – Création du Projet

1.1.2 Création d'un nouveau circuit (composant)

- **Renommer** le circuit *main* en *addi1* : *Propriétés >> Nom du circuit >> addi1*

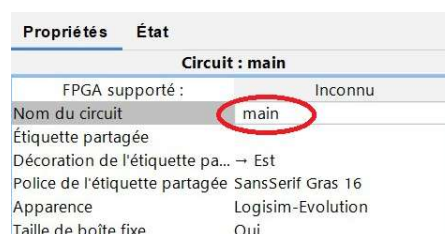


FIGURE 1.3 – Création du circuit

- **Dessiner** le schéma du circuit additionneur 1 bit. Le panneau de navigation permet d’accéder à tous les composants des bibliothèques pour dessiner un circuit logique. Par exemple, dans la bibliothèque *Portes logiques*, se trouvent les portes nécessaires.

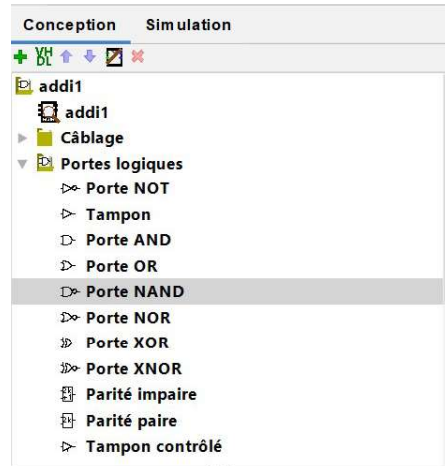


FIGURE 1.4 – Bibliothèques de Logisim

- **Sélectionner** et **positionner** les portes nécessaires.
- Puis **sélectionner** les broches d’entrée et de sortie dans la barre d’outils. Pour chaque entrée ou sortie, il faut **indiquer** son nom dans l’onglet *Étiquette*.

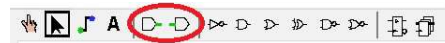


FIGURE 1.5 – Entrées/Sorties

Propriétés	État
Broche (80,110)	
FPGA supporté :	Pris en charge
Orientation	→ Est
Sortie ?	Non
Largeur données	1
Trois états ?	Non
Comportement	Inchangé
Étiquette	a
Police de l'étiquette	SansSerif Gras 16
Base	Binaire
Apparence	Formes de flèche

FIGURE 1.6 – Modification du nom des entrées/sorties

- Enfin, il faut réaliser les connectiques entre les différents éléments en sélectionnant l’outil *Câbles* dans la barre d’outils.



FIGURE 1.7 – Outil Câblage

Remarque : il est possible d’adapter le nombre d’entrées des portes logiques dans l’onglet *Nombre d’entrées*.

Propriétés	État
Porte OR (380,380)	
FPGA supporté :	Pris en charge
Orientation	→ Est
Largeur données	1
Dimension dessin	moyen
Nombre d’entrées	3
Valeur de sortie	0/1
Étiquette	
Police de l’étiquette	SansSerif Gras 16
Inverseur 1 (↑ Haut)	Non
Inverseur 2 (↓ Bas)	Non

FIGURE 1.8 – Changement du nombre d’entrées d’une porte logique

1.1.3 Vérification du fonctionnement du circuit (Simulation sous Logisim-Evolution)

- **Sélectionner** l’onglet simulation, et **vérifier** que le simulation tourne en continu (premier bouton).

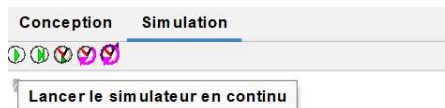


FIGURE 1.9 – Onglet Simulation

- **Sélectionner** dans la barre d’outils l’onglet qui permet de changer les valeurs des entrées dans le circuit. **Vérifier** la table de vérité en changeant les valeurs des entrées a et b.



FIGURE 1.10 – Changement des valeurs d’entrée

- **Fixer** les valeurs des entrées pour tester toutes les combinaisons possibles des entrées a et b et **valider** le fonctionnement du circuit.

1.1.4 Création du circuit à partir de la table de vérité

On souhaite maintenant réaliser un additionneur complet à la place de l’additionneur simple.

- **Supprimer** le circuit précédent dessiné dans *addi1*.

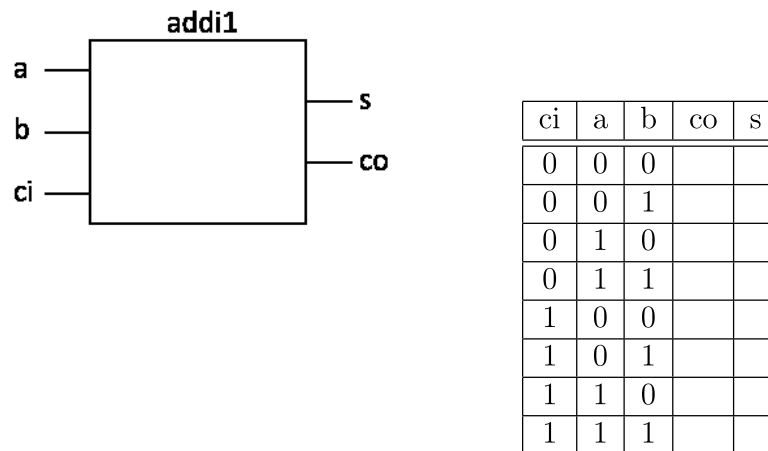


FIGURE 1.11 – Additionneur 1 bit complet

- **Compléter** la table de vérité de l'additionneur complet.
- Logisim propose la création de circuit logique à partir d'une table de vérité. On renseigne la table de vérité et on laisse à Logisim le soin de générer et dessiner le circuit correspondant. Pour cela :
Projet >> Analyser le circuit
- Dans l'onglet *Entrées & Sorties* : **Renseigner** les entrées et les sorties.

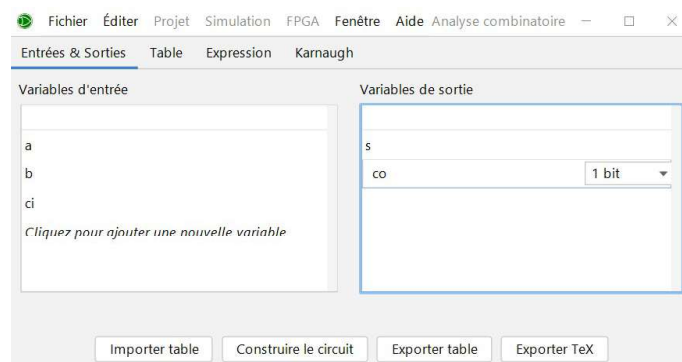


FIGURE 1.12 – Entrées et Sorties de la table de vérité

- Dans l'onglet *Table* : **Écrire** la table de vérité. Les combinaisons des entrées sont déjà écrites.

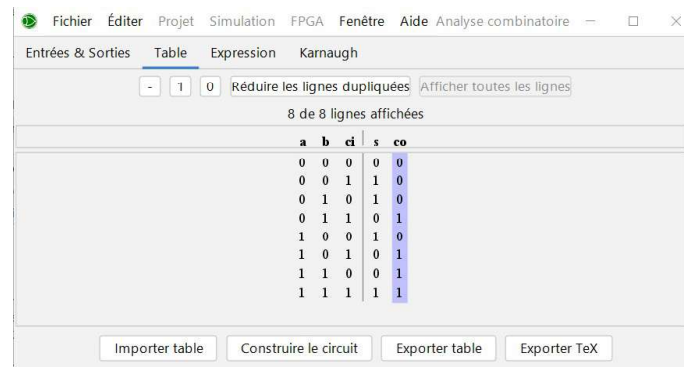


FIGURE 1.13 – Ecriture de la table de vérité

- **Vérifier** l'équation logique dans l'onglet *Expression* de cette fenêtre, ainsi que les tables de Karnaugh des deux sorties dans l'onglet *Karnaugh*.
- **Générer** le circuit : *Construire le circuit* >> *OK*

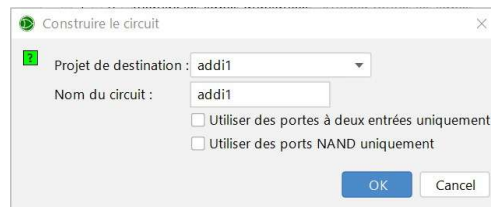


FIGURE 1.14 – Génération du circuit

- **Simuler** le circuit et vérifier son fonctionnement

1.1.5 Implémentation du circuit sur maquette Cyclone V GX Starter Kit

On souhaite implémenter ce circuit sur FPGA.

- **Choisir** un dossier dans lequel sera généré le projet : *Fichier* >> *Préférences*
- Se **placer** dans l'onglet *FPGA Commander paramètres* et indiquer l'emplacement du projet.

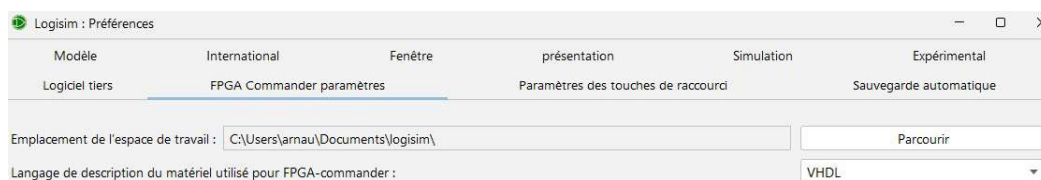


FIGURE 1.15 – Choix de l'emplacement de l'espace de travail

- **Sélectionner** *FPGA* >> *Synthétiser et télécharger*.
- **Choisir** la *carte Cible* : ENSGTI_Cyclone_V.

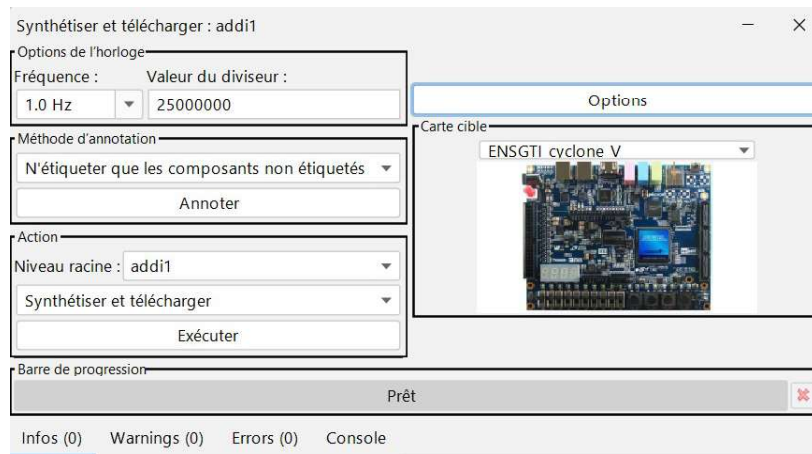


FIGURE 1.16 – Synthèse sur Cyclone V GX Starter Kit

- Cliquer sur *Annoter*.
- Dans le cadre *Actions*, sélectionner
 - le niveau racine correct (*addi1*),
 - *Synthétiser et télécharger*
- Cliquer sur *Exécuter*.
- **Réaliser** le mappage : **Associer** les noms d'entrées et de sorties aux périphériques d'entrées et de sorties
 - Entrées -> Switchs
 - Sorties -> LEDs

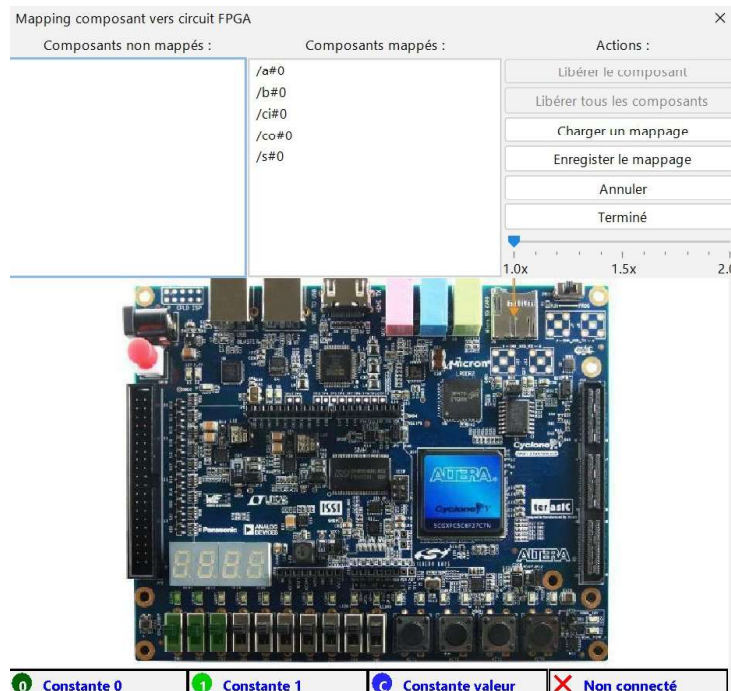


FIGURE 1.17 – Synthèse et mappage sur Cyclone V GX Starter Kit

- Cliquer sur *Terminé*.
- Connecter la carte Cyclone V GX Starter Kit puis télécharger :

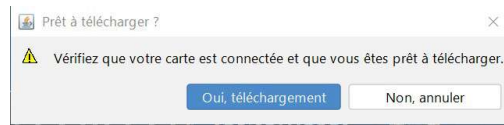


FIGURE 1.18 – Téléchargement sur Cyclone V GX Starter Kit

- Tester sur la maquette.

1.1.6 Visualisation de l'implémentation avec Quartus

- Ouvrir un explorateur de fichier et se déplacer dans le dossier sélectionné précédemment
- Ouvrir et analyser les fichiers *.vhd* du dossier */vhdl/gates*
- Ouvrir le projet *logisimTopLevelShell.qpf* généré par Logisim dans le dossier sélectionné précédemment. Le projet se trouve dans *Sandbox*. Le projet s'ouvre avec le logiciel Quartus.
- Dans Quartus, choisir *Assignments >> Pin Planner*.

Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate
<<new node>>									
s_0	Output	PIN_L7	8A	B8A_NO	PIN_L7	2.5 V		12mA (default)	1 (default)
co_0	Output	PIN_K6	8A	B8A_NO	PIN_K6	2.5 V		12mA (default)	1 (default)
ci_0	Input	PIN_AD13	4A	B4A_NO	PIN_AD13	2.5 V		12mA (default)	
b_0	Input	PIN_AE10	4A	B4A_NO	PIN_AE10	2.5 V		12mA (default)	
a_0	Input	PIN_AC9	3B	B3B_NO	PIN_AC9	2.5 V		12mA (default)	

FIGURE 1.19 – Visualisation des broches sur Quartus

- Identifier dans la colonne *Location* les noms des broches du FPGA choisies. Vérifier la correspondance sur le schéma de la datasheet de la Cyclone V GX Starter Kit (voir Annexe).
- Cliquer sur *Tools >> Netlist Viewers >> Technology Map Viewer (Post Fitting)* pour visualiser l'implémentation réelle sur le composant FPGA.
- Faire un clic droit sur un port de sortie et choisir *Locate Node >> Locate in Chip Planner*
- Zoomer sur l'élément *I/O (IOE)* et double cliquer dessus pour voir l'intérieur : *Pads, Buffers*
- Revenir dans le Chip Planner et choisir *View >> Generate Fan-In Connections* en ayant sélectionné l'IOE
- Dézoomer et identifier l'*adaptive logic module (ALM)* utilisé. De la même façon que précédemment, double cliquer dessus pour voir l'intérieur. *Registers, mux, adder, LUTs* : Blocs de mémoire dans lequel est implémentée la fonction logique.
- Vérifier le fonctionnement de l'*ALM* correspondante.