TP2

Le but de ce TP est de se familiariser avec d'autres circuits combinatoires que l'on utilisera par la suite. Tout circuit réalisé dans Logisim peut être réutilisé dans un autre circuit. Ce mécanisme permet de réutiliser le travail déjà fait et de hiérarchiser la conception des circuits (cela correspond un peu à l'idée des fonctions). Dans ce TP, on prend comme exemple celui des additionneurs binaires. Créer un nouveau projet, que vous enregistrerez dans `adder.circ`, puis travaillez toujours dans ce même fichier.

2.1. Un demi-additionneur

1) Commencez par créer un circuit `DA` : pour cela vous pouvez faire `Project→Add circuit`, puis donner son nom au circuit. Assurez vous que vous éditez bien le circuit `DA`, en double-cliquant sur son nom : une loupe doit apparaître dessus. Réalisez le circuit représenté à gauche sur la Figure 2.1.
2) On crée une instance de `DA` dans le circuit principal : éditer le circuit `main` en double-cliquant sur son nom. Ensuite, déposez une instance de `DA` comme vous l'avez déjà fait pour des composants de la bibliothèque, et complétez le circuit comme indiqué à droite sur la Figure 2.1. Notez que l'on retrouve les entrées (a, b), et les sorties (r_s, s), « disposition » que lors de la création du circuit : vous pouvez faire s'afficher leurs noms en passant le pointeur dessus.
3) Complétez la table de vérité ci-dessous. Pourquoi appelle-t-on ce circuit un demi-additionneur ?

a	b	r_s	s
0	0		
0	1		
1	0		
1	1		

.....

.....

.....

.....

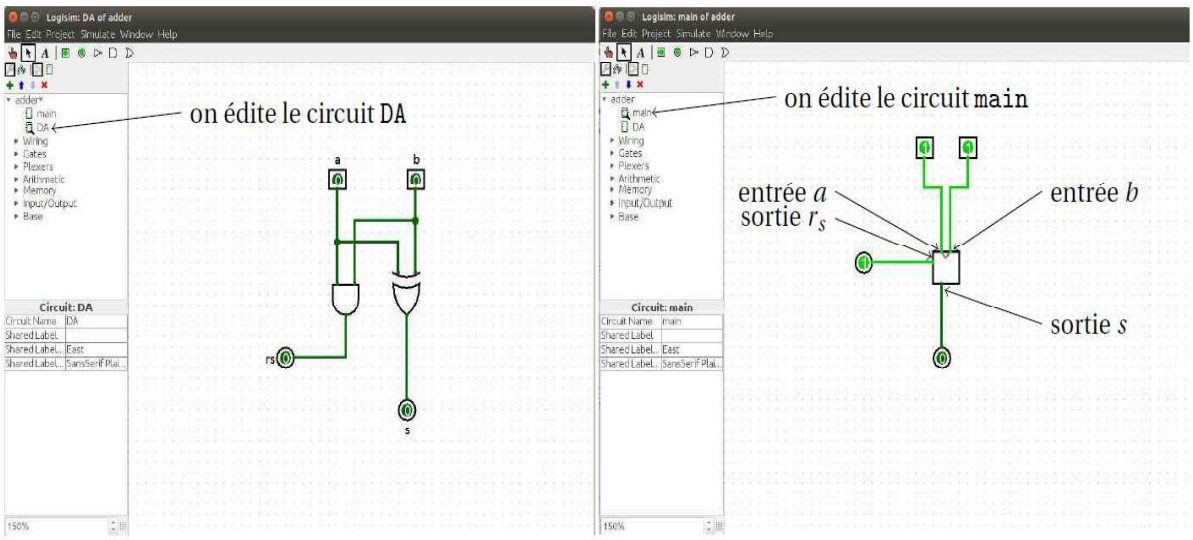
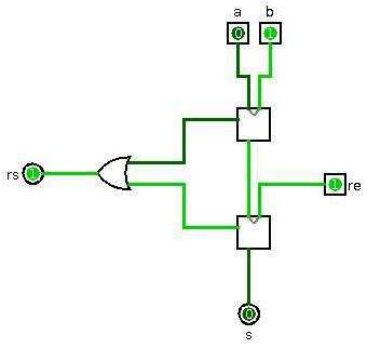


FIGURE 2.1 – Création (à gauche) et instanciation (à droite) du circuit DA.

2.2. Additionneur 1 bit complet

Un additionneur 1 bit complet prend en entrée trois bits a, b et r_e, et produit deux bits de sortie r_s et s tels que $2 \times r_s + s = a + b + r_e$ (au sens des entiers naturels).

1) En faisant la table de vérité d'un additionneur 1 bit complet, trouver comment réaliser ce circuit à l'aide de deux demi-additionneur et d'une porte `OR`.



2) Implantez un circuit `AC` réalisant un additionneur 1 bit complet.

3) Veillez à bien tester votre circuit en vous basant sur sa table de vérité.

2.3. Additionneur 4 bits

Vous devez réaliser maintenant un additionneur 4 bits, dans un circuit `add4`, qui devra :

— prendre en deux entiers binaires a et b, chacun sur 4 bits, ainsi qu'une retenue entrante r_e sur 1 bit,

— produire en sortie l'entier binaire s sur 4 bits, et une retenue sortante r_s sur 1 bit.

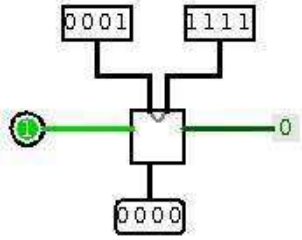
Les entrées et sorties doivent vérifier $2^4 \times r_s + s = a + b + r_e$. Vous utiliserez :

— un splitter pour obtenir les 4 bits qui composent l'entrée a, un autre pour décomposer b,

— un splitter regrouper les 4 bits qui composent la sortie s,

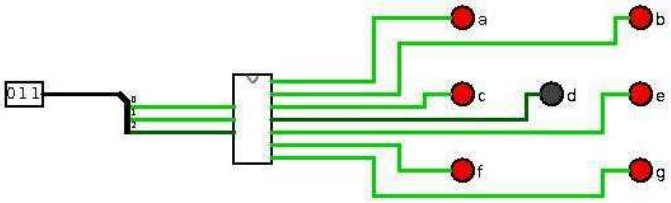
— 4 instances de votre circuit `AC`.

Testez ensuite votre circuit dans le `main`, en réalisant un circuit ressemblant à celui ci-dessous. Pour la retenue entrante, on utilise une constante égale à 0 (`Wiring`, `Constant`). Avec votre circuit, calculez $(0101)_2 + (0011)_2$, $(1111)_2 + (0001)_2$, $(1010)_2 + (1011)_2$. Dans chacun des cas, vérifiez à la main que vous obtenez bien le bon résultat. Si vous deviez tester tous les cas possibles, combien y en aurait-il ?



2.4. Le Dé électronique

On souhaite implanter dans Logisim l'afficheur « dé électronique » vu en TD. Le circuit principal aura l'allure suivante :



Entre l'entrée sur 3 bits placée sur la gauche, et les 7 LED permettant l'affichage, un transcodeur permet d'effectuer la conversion, de façon à ce que les bonnes LED soient allumées en fonction de l'entier indiqué. Pour fabriquer le transcodeur, nous allons utiliser un outil de Logisim permettant de créer automatiquement des circuits combinatoires d'après leur table de vérité.

1) Allez dans `Windows`→`Combinational Analysis` :
- dans l'onglet `Input`, indiquez les variables d'entrée du transcodeur (`x`, `x`, `z`) ;
- dans l'onglet `Output`, indiquez les variables de sortie (`a`, ..., `g`) ;
- dans `Table`, vous pouvez entrer la table de vérité de chaque sortie ;
- dans `Expression`, vous obtenez une expression booléenne pour chaque de sortie, et dans `Minimized` une expression simplifiée.

Une fois que vous avez entré toutes les tables de vérité, vous pouvez utiliser le bouton `Build circuit`, pour construire un circuit auquel vous donnerez le nom de `transcoder`. Pour tous les détails, voir dans l'aide de Logisim : `Guide to being a Logisim User` → `Combinational analysis`.

2) Utilisez le circuit `transcoder`, ainsi que des LED (`Input/Output` → `LED`) pour construire le circuit demandé. Testez le soigneusement !

3) Vous pouvez ensuite expérimenter avec le composant `Memory`→`Random Genrator` pour simuler un lancer de dé !