Travaux pratique : Construction d'une Unité Arithmétique et Logique simple

Année universitaire 2021–2022

1 Présentation générale du sujet

Le travail consiste à construire une Unité Arithmétique et Logique (UAL) simple, capable de réaliser 8 opérations différentes sur des données de 16 bits (addition, soustraction, négation, et, ou, ou-exclusif, décalage à gauche, et décalage arithmétique à droite), et capable de produire 4 bits de status (résultat négatif, résultat nul, retenue sortante, débordement). Le modèle du circuit sera développé à l'aide du logiciel Logisim, dans sa version *italian fork*.

Le travail s'étale sur 4 séances et 2 semaines. La première semaine sera consacrée à la prise en main de Logisim et à la construction d'un additionneur 4 bits avec anticipation de la retenue. Ce circuit a été étudié à l'occasion du TD n° 1. La deuxième semaine sera consacrée à la construction d'un additionneur 16 bits avec anticipation de la retenue, puis à la construction de l'UAL.

2 Évaluation

2.1 Modalités

D'après le livret de l'étudiant, les modalités de contrôle des connaissances pour la ressource R1.03 impliquent deux notes : une note de DS (devoir surveillé), et une note de TD (travaux dirigés). Cette dernière a pour but d'évaluer votre investissement et la régularité de votre travail. Ce TP compte pour $\frac{3}{4}$ de la note de TD, selon les modalités suivantes :

— Chaque semaine, à l'issue de la deuxième séance, vous déposez votre fichier Logisim sur un dossier réseau dont le lien vous sera donné par votre enseignant(e).

— Une semaine après votre dernière séance, vous déposez un rapport décrivant votre travail et justifiant les choix de conception que vous avez réalisé.

2.2 Triche, plagiat

En cas de triche, l'étudiant(e) concerné(e) aura la note 0. En cas de plagiat (reprise partielle ou totale du travail d'un(e) autre étudiant(e)), l'un des travaux sera noté et l'ensemble des étudiant(e)s concerné(e)s seront invité(e)s à se partager à les points.

Vous pouvez discuter avec vos camarades des problèmes que vous rencontrez. Par contre, vous ne pouvez pas partager vos solutions.

3 Première semaine

3.1 Installation et prise en main de Logisim

Logisim est un logiciel permettant la conception et la simulation de circuits électroniques combinatoires et séquentiels simples. Il est développé avant tout dans un objectif pédagogique. Pour le développement de circuits complexes on utilisera plutôt des langages de description dédiés, comme par exemple VHDL.

Logisim peut être téléchargé ici : https://logisim.altervista.org/. Se rendre en bas de la page, cliquer sur le bouton étiqueté `JAR`, puis sur le bouton `DOWNLOAD`. Une fois le logiciel téléchargé, la commande suivante, utilisée dans le dossier où se trouve le fichier, permet de le lancer :

```
$ java -jar Logisim-ITA.jar
```

Une fois Logisim téléchargé et lancé, se signaler à l'enseignant(e) et attendre la démonstration.

3.2 Additionneur 1 bit

La première étape consiste à construire un additionneur 1 bit. Ce circuit doit avoir trois entrées et trois sorties, toutes ayant une largeur de 1 bit. Les trois entrées sont :

- A, la première opérande.
- B, la seconde opérande.
- C_{in}, la retenue entrante.

Les trois sorties sont :

- S, le résultat de l'addition.
- P, qui est vraie ssi les valeurs des entrées sont telles qu'une retenue entrante sera propagée en sortie.
- G, qui est vraie ssi les valeurs des entrées sont telles qu'une retenue sortante est nécessairement engendrée.

Les équations booléennes permettant le calcul des sorties à partir des entrées ont été vues (TD n° 1, exercice 3). Créer le circuit puis utiliser les différentes fonctionnalités de Logisim pour le tester et le valider. En cas de doute, demander à l'enseignant(e) de valider le travail.

3.3 Unité de calcul anticipé de la retenue

La deuxième étape consiste à construire une unité de calcul anticipé de la retenue pour une addition sur 4 bits. Le principe général de cette unité, ainsi que la formule permettant de calculer la retenue sortante de l'addition de 4 bits à partir des entrées, ont été vues (TD n° 1, exercice 3). En utilisant ce travail préliminaire, il faut :

1. Identifier les entrées et les sorties du circuit.
2. Déterminer les équations permettant de calculer les retenues intermédiaires.
3. Construire le circuit.
4. Tester et valider le circuit. En cas de doute, demander à l'enseignant(e) de valider le travail.

Indication (nombre d'entrées d'une porte) Dans Logisim, une porte logique peut avoir plus de deux entrées. Lorsque un composant de l'espace de travail est sélectionné, une boîte de dialogue de configuration s'ouvre en bas à gauche de l'écran. Lorsque le composant est une porte logique, dans cette boîte de dialogue, l'attribut `Nombre d'entrées` correspond, comme son nom l'indique, au nombre d'entrées de la porte. Cet attribut peut donc prendre une valeur supérieure à deux. Par exemple, pour traduire un terme du type $a.b.c.d$, on utilisera une porte ET avec quatre entrées.

3.4 Additionneur 4 bits

La troisième étape pour cette première semaine consiste à construire un additionneur 4 bits avec calcul anticipé de la retenue. Pour cela, il faut utiliser quatre additionneurs 1 bit, une unité de calcul anticipé de la retenue, et reprendre les étapes habituelles :

1. Identifier les entrées et les sorties du circuit.
2. Construire le circuit.
3. Tester et valider ce circuit. En cas de doute, demandez à l'enseignant(e) de valider votre travail.

L'étape consistant à déterminer les équations n'est pas nécessaire ici car le travail de construction est principalement un travail d'interconnexion.

Indication (largeur d'une connexion) Dans Logisim, une entrée (ou une sortie, ou n'importe quel point de connexion) peut avoir une largeur en bit supérieure à 1. Cette largeur se configure par l'attribut `DataBits` dans la boîte de dialogue de configuration du composant. Pour passer d'une largeur à une autre, il est possible d'utiliser le composant `Séparateur (splitter)` disponible sous le nœud `Cablage`. Il est possible de configurer la largeur de chaque « dent du peigne », ainsi que la correspondance entre les positions des bits du peigne et celles des bits de l'autre extrémité du `Séparateur`.

3.5 Additionneur 4 bits avec sorties super-P et super-G

Selon votre avancement, cette quatrième étape peut être réalisée en fin de première semaine ou début de deuxième semaine. Il s'agit de modifier le circuit conçu à l'étape précédent pour lui ajouter deux sorties :

— P est un « super-signal » de propagation, qui indique si une retenue entrante sur l'additionneur 4 bits sera propagée en sortie.

— G est un « super-signal » de génération, qui indique si une retenue sortante est engendrée par l'additionneur 4 bits.

1. Écrire les équations permettant de calculer P et G et les faire valider par l'enseignant(e).
2. Construire le circuit correspondant.
3. Tester et valider ce circuit. En cas de doute, demander à l'enseignant(e) de valider le travail.

4 Deuxième semaine

4.1 Additionneur 16 bits

En utilisant l'additionneur 4 bits avec sorties super-P et super-G de l'étape précédente, et l'unité de calcul anticipé de la retenue, il faut construire dans cette étape un additionneur 16 bits. Cela doit normalement être assez rapide puisque ce circuit se construit sur le modèle de l'additionneur 4 bits sans les super-sorties. Comme toujours, en cas de doute sur la validité de la solution mise en œuvre, il est possible de demander à l'enseignant(e).

4.2 Réaliser des soustractions avec un additionneur

Le circuit additionneur mis en place à l'étape précédente permet également de réaliser des soustractions. Il suffit pour cela d'utiliser deux propriétés :
— Le fait que $x - y = x + (-y)$.
— La technique de construction de l'opposé d'un nombre en complément à 2 vue dans le TD n° 2 (exercice 6).

Indication (multiplexeur) Si vous n'avez pas eu le temps de faire l'exercice 4 du TD n° 1, il vous faut maintenant comprendre ce qu'est un `Multiplexeur`, et la façon dont on s'en sert. Dans Logisim, ce composant est disponible sous le nœud `Plexers`.

Un multiplexeur a 2^k entrées de données, k entrées de contrôle, et une sortie. Fonctionnellement, le circuit recopie sur la sortie le signal d'entrée dont le numéro est codé par les bits de l'entrée de contrôle. Ainsi, si un multiplexeur a deux entrées de données, E_0 et E_1, et une entrée de contrôle C, la sortie S sera égale à E_0 quand C vaut 0, et E_1 quand C vaut 1. Dans un circuit, c'est un composant qui sert à réaliser des « aiguillages » conditionnels.

Le circuit à concevoir à cette étape dispose d'une entrée supplémentaire par rapport à l'additionneur : 1 bit de commande qui permet de déterminer si l'opération à effectuer est une addition ou une soustraction. Par convention, il s'agira d'une addition lorsque le bit de commande est à 0 et d'une soustraction dans le cas opposé.

Construire le circuit, le tester et le valider. En cas de doute, demander à l'enseignant(e) de valider le travail.

4.3 Ajout de fonctions à l'UAL

L'étape suivante consiste à étendre l'UAL, qui supporte pour l'instant uniquement deux « instructions », afin qu'elle en supporte 8. Une telle UAL a donc une entrée de contrôle qui a une largeur de 3 bits. L'association entre la valeur de l'entrée de contrôle et la fonction à calculer est :
— 000 : $S = A + B$.
— 001 : $S = A - B$.
— 010 : $S = \overline{A}$.
— 011 : $S = A.B$.
— 100 : $S = A + B$.
— 101 : $S = A \oplus B$.
— 110 : décalage à gauche de A de 1 bit (cf. indication ci-dessous).
— 111 : décalage à droite arithmétique de A de 1 bit.

Indication (décalage) Le décalage à gauche consiste à décaler tous les bits de l'opérande de n crans vers la gauche. Ainsi, les n bits de poids forts sont évincés, et n bits à 0 sont insérés. Cela revient à multiplier l'opérande par 2^n.

Le décalage arithmétique à droite consiste à décaler tous les bits de l'opérande de n crans vers la droite. Ainsi, les n bits de poids faible sont évincés, et n bits sont insérés, dont la valeur est égale au bit de poids fort de l'opérande avant décalage. Recopier le bit de poids fort permet de préserver le signe de l'opérande si cette dernière est codée en complément à 2. Un décalage arithmétique à droite de n bits est équivalent à une division par 2^n.

Dans Logisim, le composant `Décalage` est disponible sous le nœud `Arithmétique`. La boite de dialogue permet de contrôler le sens du décalage, et, dans le cas d'un décalage à droite, la nature du décalage (arithmétique ou logique). Le composant a deux entrée : l'opérande à décaler, et une entrée permettant d'indiquer la valeur de n, c'est-à-dire l'amplitude du décalage. Dans notre cas, cette valeur est fixée à 1 et cette entrée peut donc être connectée à une `Constante` (disponible sous le nœud `Cablage`).

Indication (tunnel) Un circuit complexe ressemble vite à un plat de spaghettis. Pour limiter ce problème, il est possible d'utiliser le composant `Tunnel` disponible sous le nœud `Cablage`. Un `Tunnel` est une étiquette que l'on utilise aux extrémités de deux ou plus connexions et qui indique que ces connexions sont en fait reliées.

En s'inspirant de l'étape précédente, en particulier l'utilisation du multiplexeur, et en utilisant les composants arithmétiques et logiques proposés par Logisim (sauf l'additionneur : il faut utiliser celui conçu lors des étapes précédentes), créer une UAL avec les fonctions demandées. Construire le circuit, le tester et le valider. En cas de doute, demander à l'enseignant(e) de valider le travail.

4.4 Ajouter des informations de status

Pour permettre de traduire les structures de contrôle conditionnelles des langages de programmation, les langages machines proposent la plupart du temps des instructions particulières que l'on appelle des branchements conditionnels. Ces instructions permettent de se rendre à un certain point du programme lorsqu'une condition donnée est satisfaite. La condition est exprimée sur le résultat du dernier calcul réalisé par l'UAL, c'est pourquoi cette dernière produit, en plus du résultat, des bits de status. Cette dernière étape consiste à ajouter les bits de status suivants à l'UAL construite à l'étape précédente :

— Z : la sortie vaut 0 (tous les bits de la sortie sont à 0).

— N : la sortie, interprétée en complément à 2, est négative.
— C : retenue sortante.
— V : interprété en complément à 2, le résultat du calcul déborde de la capacité de codage.

1. Pour chaque bit de status, donner l'équation permettant de le calculer à partir des informations qui circulent dans l'UAL.
2. Modifier l'UAL pour ajouter le calcul des bits de status. Tester et valider ces modifications. En cas de doute, demander à l'enseignant(e) de valider le travail.