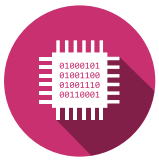


Multiplicateur Câblé

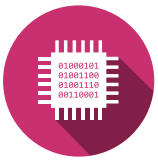
Laboratoire Conception Numérique

Contenu

1 Objectif

Dans ce laboratoire, vous vous penchez sur la mise en œuvre de multiplicateurs pour les nombres naturels et arithmétiques. Vous apprenez comment les algorithmes de multiplication sont réalisés en matériel et quelles particularités doivent être prises en compte lors du traitement de nombres binaires.

En développant progressivement un multiplicateur pour les nombres naturels et en l'étendant aux nombres arithmétiques en complément à deux, vous acquérez une compréhension approfondie des circuits numériques. Vous utilisez différents blocs logiques et analysez leur comportement en tenant compte des retards de calcul.



2 Multiplieur pour nombres naturels

Dans cette section, vous développerez un multiplieur pour les nombres naturels, c'est-à-dire pour les entiers non négatifs stockés sous forme binaire.

2.1 Algorithme

La Figure 1 présente l'algorithme de multiplication de deux nombres binaires. Il est basé sur le calcul de produits partiels qui sont ensuite additionnés. Cette méthode correspond à la multiplication écrite dans le système décimal, sauf qu'ici des chiffres binaires sont utilisés. Chaque produit partiel est obtenu en multipliant un nombre par un seul chiffre binaire de l'autre nombre. Les produits partiels résultants sont ensuite décalés d'un bit vers la droite et additionnés pour obtenir le résultat final.

				a_3	a_2	a_1	a_0
				$\times\ b_3$	b_2	b_1	b_0
				$b_0{*}a_3$	$b_0{*}a_2$	$b_0{*}a_1$	$b_0{*}a_0$
			$b_1{*}a_3$	$b_1{*}a_2$	$b_1{*}a_1$	$b_1{*}a_0$	
		$b_2{*}a_3$	$b_2{*}a_2$	$b_2{*}a_1$	$b_2{*}a_0$		
	$b_3{*}a_3$	$b_3{*}a_2$	$b_3{*}a_1$	$b_3{*}a_0$			
p_7	p_6	p_5	p_4	p_3	p_2	p_1	p_0

Figure 1 - Algorithme de multiplication

2.2 Analyse

Nous allons d'abord analyser la multiplication de 2 nombres naturels (unsigned) codés sur 4 bits. La multiplication de 2 nombres de 4 bits donne un produit de 8 bits. Comme illustré dans la Figure 1.

Déterminez le nombre le plus grand et le plus petit que l'on peut atteindre avec cette multiplicateur.

Déterminez également le nombre de bits nécessaires pour le produit de deux nombres binaires naturels codés respectivement sur n_1 et n_2 bits.

2.3 Circuit

La Figure 2 montre le circuit d'un multiplieur qui fonctionne selon l'algorithme donné ci-dessus.

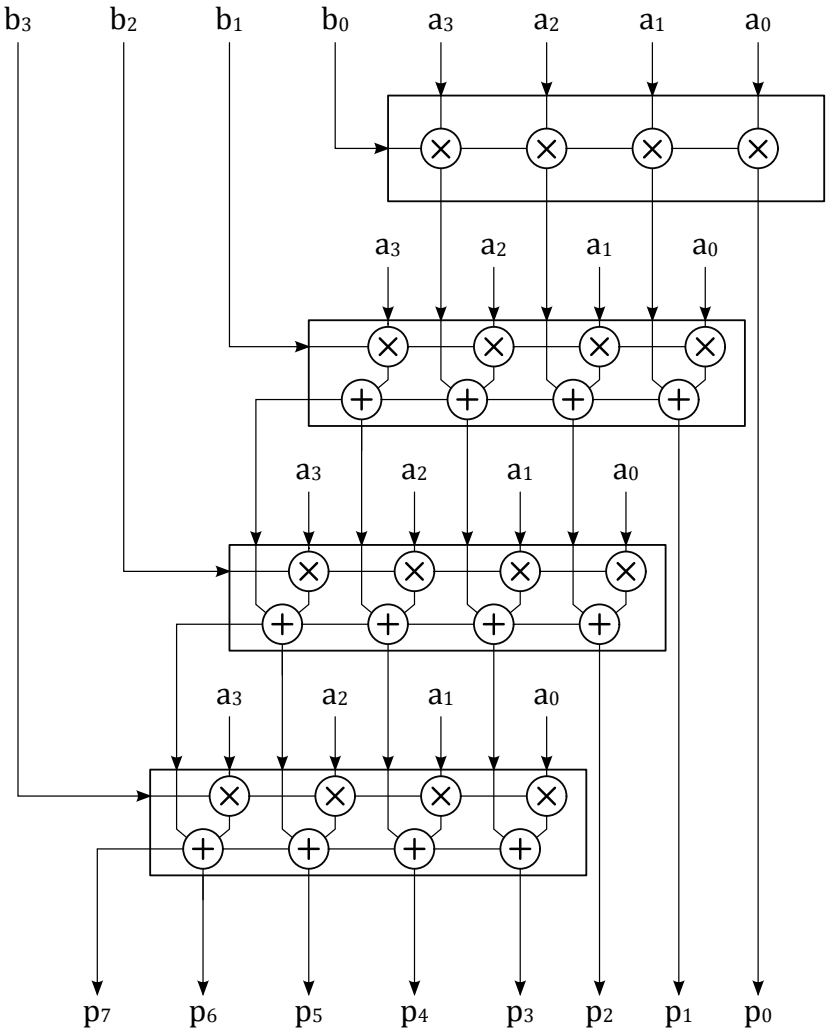
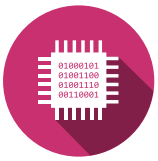


Figure 2 - Architecture du multiplicateur

2.4 Réalisation

Réalisez le circuit du multiplieur de la Figure 2. Certains éléments de circuit sont déjà donnés dans le bloc **`MUL/mul4Unsigned`**. Complétez les éléments manquants pour garantir le fonctionnement du multiplieur.

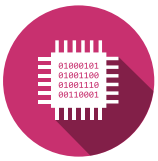
À l'aide de portes INV, ET, OU et XOR, complétez le schéma hiérarchique du multiplieur de la Figure 2.



2.5 Simulation

Chaque circuit implémenté doit être correctement testé. Le banc d'essai **`MUL_test/mul4Unsigned_tb`** doit vérifier le fonctionnement **complet** du circuit. Comment cela est-il réalisé dans le cas du testeur **`MUL_test/mul4Unsigned_tester`**?

Simulez le banc d'essai **`MUL_test/mul4Unsigned_tb`** avec le fichier de simulation **`$SIMULATION_DIR/MUL1.do`**.



3 Multiplieur pour nombres arithmétiques

Dans cette section, vous étendrez le multiplicateur précédemment implémenté aux nombres arithmétiques, c'est-à-dire aux nombres représentés en complément à deux. Cela permet la multiplication de nombres entiers positifs et négatifs.

3.1 Algorithme

La Figure 3 présente l'algorithme de Baugh-Wooley pour la multiplication de deux nombres arithmétiques (signed) codés en complément à deux avec le même nombre de bits. Par rapport à Figure 1, plusieurs additions et inversions supplémentaires sont effectuées ici pour obtenir le résultat correct.

$$
\begin{array}{cccccccc}
 & & & & a_3 & a_2 & a_1 & a_0 \\
 & & & & \times\ b_3 & b_2 & b_1 & b_0 \\
\hline
 & & & 1 & \overline{b_0{*}a_3} & b_0{*}a_2 & b_0{*}a_1 & b_0{*}a_0 \\
 & & & \overline{b_1{*}a_3} & b_1{*}a_2 & b_1{*}a_1 & b_1{*}a_0 & \\
 & & \overline{b_2{*}a_3} & \overline{b_2{*}a_2} & \overline{b_2{*}a_1} & b_2{*}a_0 & & \\
1 & b_3{*}a_3 & \overline{b_3{*}a_2} & \overline{b_3{*}a_1} & \overline{b_3{*}a_0} & & & \\
\hline
p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0
\end{array}
$$

Figure 3 - Algorithme de multiplication de nombres signés

3.2 Analyse

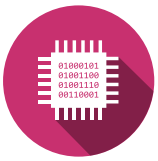
Nous allons d'abord analyser la multiplication de deux nombres arithmétiques (signed) codés en complément à deux sur 4 bits. Contrairement à la multiplication de nombres naturels (unsigned), nous devons ici tenir compte des nombres négatifs.

> Déterminez le nombre le plus grand et le plus petit que l'on peut atteindre avec cette multiplication.

3.3 Réalisation

Réalisez le circuit du multiplicateur de la Figure 3. Certains éléments de circuit sont déjà fournis dans le bloc **`MUL/mul4Signed`**. Complétez les éléments manquants pour garantir le bon fonctionnement du multiplicateur.

> À l'aide de portes INV, ET, OU et XOR, complétez le schéma hiérarchique du multiplicateur de la Figure 3.

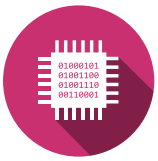


3.4 Simulation

Chaque circuit implémenté doit être correctement testé. La testbench **`MUL_test/mul4Signed_tb`** doit vérifier le **fonctionnement complet** du circuit.

Simulez la testbench **`MUL_test/mul4Signed_tb`** avec le fichier de simulation **`$SIMULATION_DIR/MUL2.do`**.

Comment expliquez-vous les nombreux glitches qui apparaissent dans la simulation?



4 Analyse

En supposant que tous les portes logiques ont la même durée de propagation de 1ns, analysez le délai de calcul maximal du multiplicateur pour les nombres naturels (unsigned), représenté dans Figure 2.

Le délai résulte du nombre de chemins de portes en cascade que le signal doit traverser avant d'atteindre le résultat final.

Déterminez le délai de calcul maximal du multiplicateur Figure 2

Proposez une architecture optimisée pour réduire le délai de calcul.