

Information Numérique Projet : Calculateur arithmétique



1. Introduction

Il s'agit de réaliser un calculateur arithmétique composé de :

1. Additionneur à propagation de retenue (Ripple Carry Adder).
2. Additionneur à retenue anticipée (Carry Look Ahead Adder).

2. Organisation, évaluation

1. Le projet se déroule en trois séances de 3 heures.
2. Les étudiants sont au maximum en binôme.
3. Les questions marquées P (préparation) et d'un astérisque (*) sont à rédiger individuellement avant la séance et seront pointées par les encadrants en début de chaque séance.
4. Les questions marquées Q sont à traiter dans le rapport final (schéma renseigné du circuit étudié, chronogramme renseigné obtenu par simulation MULTISIM, etc...).
5. Les questions marquées VQ sont à traiter de la même manière mais doivent aussi être validées par les encadrants et pour 1 binôme pendant chaque séance.
6. Toute absence sera pénalisée.
7. Le rapport final par binôme est à rendre 2 jours ouvrables après la dernière séance.
8. Les montages seront réalisés par des portes 'NAND' ou 'ETNON'.

Additionneur à propagation de retenue

3. Introduction

3.1 Demi-Additionneur

L'algorithme d'addition de 2 opérandes A ($a_{n-1} \dots a_1 a_0$) et B ($b_{n-1} \dots b_1 b_0$) de n chiffres est le même quel que soit la base. Il commence par les chiffres (a_0 et b_0) de poids faible des opérandes dont l'addition (somme arithmétique) produit deux chiffres sortants : la Somme (S_0) et la Retenue (R_0). C'est le Demi Additionneur (Half Adder).

3.2 Additionneur Complet

Pour un poids intermédiaire, les 3 chiffres entrants de l'additionneur complet (Full Adder) sont les 2 chiffres de poids correspondant provenant des 2 opérandes (a_i et b_i) et la retenue sortante R_{i-1} du calcul de poids inférieur devenant ainsi la retenue entrante du poids correspondant ($r_i = R_{i-1}$).

Ce principe est à la base des unités de calcul dans l'unité arithmétique et logique (UAL) des microprocesseurs modernes.

On se propose de réaliser des additionneurs binaires de 4 bits par deux méthodes :

1. Par propagation de la retenue.
2. Par anticipation du calcul de la retenue.

3.3 Propagation de retenue

Exemple de l'addition de deux nombres $S = A + B$ binaires ou hexadécimaux :

Retenues :	1 0111 1000 ₂	11010 ₁₆
Opérande A :	1011 1101 ₂	F749 ₁₆
Opérande B :	+ 1011 0100 ₂	+ 7958 ₁₆
	-----	-----
Somme Totale :	= 1 0111 0001 ₂	= 170A1 ₁₆

3.3.1 Quelle est la retenue entrante de poids 0 ? Quel est le nombre de chiffres du résultat ?

3.3.2 Quel est le déplacement de la retenue ?

4. Réalisation d'un additionneur à propagation de retenue

Le Demi Additionneur (DA) sera utilisé comme cellule élémentaire.

4.1 Demi-Additionneur

4.1.1 P1* : Établir la table de vérité d'un demi-additionneur de 2 opérandes binaires A et B de 1 bit, calculant la Somme (S) et la Retenue (R). Donner l'équation et el schéma logiques de S et R

4.1.2 Q1 : Vérifier en séance, avec le Logic Converter du logiciel, les équations et les schémas logiques de S et de R, puis simuler (Word Generator + Logic Analyzer) le fonctionnement du demi additionneur

4.2 Additionneur complet

Il s'agit de l'additionneur de poids intermédiaire i. Soient a_i et b_i les bits de poids i des opérandes A et B. Soit r_i la retenue entrante (la retenue sortante R_{i-1} provenant du calcul de poids i-1 immédiatement inférieur). Soient S_i la somme et R_i la retenue sortante.

4.2.1 P2* : Établir la table de vérité de l'additionneur complet et en déduire les équations logiques simplifiées de $S_i = f(a_i, b_i, r_i)$ et $R_i = f(a_i, b_i, r_i)$. Montrer qu'un additionneur complet est formé de deux demi additionneurs et d'une porte logique à définir

4.2.2 Q2 : Vérifier la question précédente en séance avec le Logic Converter

Rappel : pour construire un sous circuit :

1. Établir le logigramme du circuit,
2. Placer les entrées/sorties et les connecter au circuit !
3. Sélectionner le circuit complet,
4. Placer la sélection comme sous circuit.

4.2.3 VQ3 : Établir et simuler le sous circuit du D A. Construire le sous circuit l'AC à partir de celui i du DA et de la porte définie plus haut et le simuler

4.3 Additionneur de 2 nombres de 4 bits à propagation de retenue

La structure de ce type d'additionneur est répétitive. Une cellule élémentaire peut être utilisée pour chaque poids. L'addition de deux nombres de n éléments binaires nécessite dans ce cas 1 demi additionneur (DA) pour le calcul de poids 0 et n-1 additionneurs complets (AC) pour les poids supérieurs.

- 4.3.1 VQ4 : En utilisant la structure hiérarchique de MULTISIM, établir le schéma d'un additionneur de deux mots de quatre bits (A4b) avec 1 sous circuit DA et 3 sous circuits AC. Simuler le circuit A4b (Word Generator + Logic Analyzer + afficheurs 7 segments adéquats placés sur les opérandes et sur les résultats)

4.4 Additionneur de 2 nombres de 8 bits à propagation de retenue

- 4.4.1 Q5 : Proposer le montage d'un additionneur de deux mots de 8 bits en utilisant le sous circuit A4b et un sous circuit à 4 AC en expliquant pourquoi. Simuler le fonctionnement de l'A8b à propagation de retenue.

Dans cet additionneur la retenue est propagée. Soit t le temps de calcul pour un étage. Dans le pire cas, le temps de calcul total est $n.t$. Pour diminuer le temps de calcul au prix d'une augmentation de la complexité (nombre de portes) on propose d'étudier l'additionneur à retenue anticipée.

Additionneur à retenue anticipée

5. Introduction

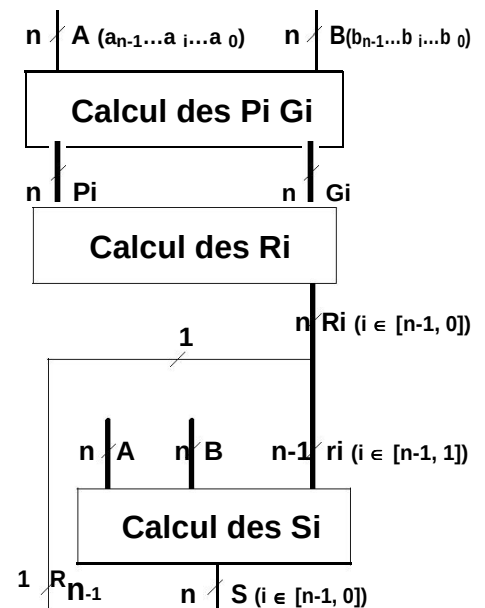
La structure complète de l'additionneur n bits à retenue anticipée est la suivante. Le symbole  désigne un bus de n fils.

Cette méthode consiste à calculer toutes les retenues en parallèle directement à partir des opérandes (données d'entrées) sans calculer les sommes partielles. Les temps de calcul de chaque étage dépendent moins du nombre de bits des opérandes.

En repartant des équations de la somme et de la retenue d'un additionneur complet, nous établirons les équations logiques de récurrence des retenues pour des opérandes de 4 bits en posant :

- $G_i = a_i \cdot b_i$ (G_i est le terme de génération),
- $P_i = a_i + b_i$ (P_i est le terme de propagation).

- 5.1.1 P3* : Montrer que les expressions des retenues sortantes R_i peuvent s'écrire $R_i = G_i + P_i r_i$ où r_i est la retenue entrante ($r_i = R_{i-1}$: retenue sortante R_{i-1} de poids $i-1$) et les sommes $S_i = a_i \oplus b_i \oplus r_i$.



6. Additionneur à retenue anticipée de 4 bits

- 6.1.1 P4* : En utilisant la relation de récurrence $r_i = R_{i-1}$ (retenue entrante : r_i et retenue sortante : R_{i-1}) exprimer les R_i (i valant de 0 à 3) en fonction exclusivement des $G_i P_i$.