Synthèse logique de CORDIC

La lecture du cours est nécessaire pour comprendre les principes de l'algorithme CORDIC utilisé pour la rotation.

Objectifs

Dans ce TME, vous devez

- compléter le modèle VHDL du circuit CORDIC à partir d'une description de l'algorithme en décrit en C.
- valider le fonctionnement avec des patterns que vous pouvez produire à la main, ou avec genpat, pour en créant un *test bench* en vhdl.
- faire la synthèse logique sur SXLIB.
- écrire le compte-rendu de vos travaux et décrire les résultats obtenus.
- Le même circuit sera repris les deux semaines suivantes alors vous pourrez ne rendre qu'un seul compte rendu reprenant les étapes.

Fonction

Le circuit réalise la rotation d'un vecteur (x,y) par un angle a et produit le vecteur (nx,ny)

- Le circuit prend en entrée
  - ♦ les coordonnées x_p et y_p qui sont des nombres entiers signés de -127 à +127.
  - ♦ L'angle a_p est exprimé en radian et il est représenté par un nombre en virgule fixe **non** signé 3-7.
    - ◊ À l'intérieur du circuit, c'est un nombre en virgule fixe 1-8-7.
    - ◊ Mais, à l'interface, j'ai choisi une représentation non signée 3-7 (port a_p) pour avoir des angles entre 0 et presque 8 radians. La conversion se fait dans le circuit en recopiant les 10 bits de a_p dans les 10 bits de poids faible d'un registre de 16 bits représentant l'angle, puis en complétant avec des `0`à gauche. C'est un choix pour réduire le nombre de broches, mais vous pouvez faire un choix plus "propre" en codant l'angle en 1-3-7 et faire une conversion avec extension du signe.
  - ♦ le circuit reçoit aussi une horloge et un signal reset.
- Le circuit produit en sortie les coordonnées (nx_p, ny_p) du vecteur après rotation.
- Le protocole de communication en entrée et en sortie est FIFO.

Fichiers fournis

- **Spécification de haut niveau**

  La spécification de haut niveau (non RTL) est donnée dans un programme C.
  C'est ce programme qui sert de référence pour le modèle RTL.

  - ♦ cercle.c : contient deux fonctions qui calculent un cercle en faisant tourner vecteur (127,0). La première fonction utilise les fonctions `sin()` et `cos()`, la seconde utilise l'algorithme cordic(). Les deux fonctions produisent un fichier .dat avec les coordonnées que gnuplot peut afficher.

  - ♦ Makefile : compile et execute cercle.c puis lance gnuplot pour afficher les deux cercles.

- **Modèle VHDL 1 core**

  Vous allez modéliser le circuit, mais comme ce n?est pas si simple (ça prend du temps), je vous donne un fichier vhdl incomplet, qui va vous aidez (en principe). Dans ce fichier, j'ai fait des choix que vous êtes

libres de changer, mais vous devez avoir en tête que ce modèle 1 core, sera découpé en deux parties la semaine prochaine : une partie contrôle et une partie chemin de données. Le modèle que je vous propose me semble assez simple à couper.

Parmi les choix qui peuvent faire débat, il y a la description de l'automate. J'utilise un codage One-Hot parce que je trouve que c'est lisible et c'est très efficace (en fait tous les circuits que auxquels j'ai participé ont utilisé cette approche...).

Je ne vous donne pas de patterns. Je vous conseille de créer un *test benchs* qui produit et consomme des données et que vous devez connecter à votre circuit. Si vous n'y arrivez pas, je vous donnerai de quoi valider le circuit la semaine prochaine.

Notez que si vous avez une manière plus élégante de faire des shifters, allez-y, mais il faut que `vasy`accepte, or j'ai bien peur que les for generate (solution possible) ne soient pas acceptés.

- cordic_cor.vhd