Introduction à

Wild

Décrire des composants électroniques

Après avoir étudié les concepts fondamentaux des systèmes d'exploitation, il nous font revenir à la conception hardware et commencer à réfléchir à la programmation de notre FPGA. Pour cela, nous entamons une nouvelle étape avec la découverte pas à pas du VHDL, langage de haut niveau permettant de coder ce style de puce…

Un peu d'histoire pour débuter...

VHDL signifie Very High Speed Integrated Circuit Hardware Description Language. C'est un langage de synthétisation des composants électroniques né de l'impérieuse nécessité d'uniformiser les différents langages de description matériel déjà existants. La première norme VHDL fut la IEEE 1076-87, parue en décembre 1987.

En effet, jusqu'à cette date, chaque société spécialisée dans la Conception Assistée par Ordinateur proposait son langage propriétaire (c'était le cas avec M chez Mentor Graphics, Verilog chez Cadence etc.) mais également des langages pour la synthèse et d'autres pour le test. Bref, à l'époque, il s'agissait d'une vraie « jungle ».

Ainsi, au début des années 1980, le DOD (Department Of Defense – Département de la Défense Américaine) confia à trois sociétés (Intermetrics, I.B.M. et Texas Instruments) de mettre au point un nouveau langage apte à assurer à tous une indépendance vis à vis des sociétés tierces existantes et de leurs outils de développement. C'est donc de cette manière que VHDL est né.

Les normes se sont succédés : soulignons ainsi l'existence, dés 1994, de la version IEEE 1076-93 puis de l'IEEE 1164 et enfin de la norme IEEE 1076.4. Très récemment encore, des modifications ont eu lieues avec l'IEEE 1076.1-1999 qui démontre bien la vitalité du langage VHDL qui est, depuis sa mise au point, un réel succès.

Les avantages du langage VHDL

En premier lieu, VHDL est un langage portable. Ensuite, le langage a un aspect généraliste ce qui permet une forte abstraction de ce que l'on veut créer ce qui permet la modélisation. Il est certain qu'on fur et à mesure de votre avancée au sein de VHDL, vous découvrirez d'autres avantages. N'hésitez pas à venir nous en faire part dans la rubrique hardware du forum en ligne de votre magazine préféré ! ! Il est désormais temps de se lancer à l'assaut de ce mystérieux langage…

Un premier survol...

Un code VHDL se compose toujours, lors d'une approche minimaliste, de deux parties distinctes.

La première s'appelle l'ENTITY et correspond à la description des entrées-sorties du circuit. La seconde, regroupant la description du circuit, se dénomme ARCHITECTURE.

VHDL permet la manipulation d'un certain nombre d'opérateurs décrits dans le tableau n° 1

OPERATEURS LOGIQUES	
And	Nand
Or	Nor
Xor	Xnor
Not	
OPERATEURS RELATIONNELS	
=	Egal
>	Supérieur
>=	Supérieur ou égal
/=	Différent
<	Inférieur
<=	Inférieur ou égal
OPERATEURS ARITHMETIQUES	
+	Addition
-	Soustraction
*	Multiplication
/	Division
mod	Modulo
abs	Valeur absolue

Tableau n° 1 – Les opérateurs avec VHDL

Il est également possible de manipuler des affectations, comme dans tous les langages

VHDL

de haut niveau. Ainsi, une variable s'affecte par := (dans un process ; ne vous inquiétez pas si le terme est obscur, nous l'éclaircirons au fil de nos aventures !) tandis qu'un signal se met en évidence par <=.

Entrons maintenant dans la partie ENTITY dont nous avons déjà précisé l'existence un peu plus haut. Elle correspond à la description de notre puce avec le monde extérieur c'est à dire l'intégralité des E/S ou I/O (Entrées-Sorties ou Input/Output).

L'entête du programme est de la forme :

```
ENTITY nomprog IS
 --nomprog est le
   nom de programme
PORT (
 --Description des
   entrées ici
);
END nomprog ;
```

Pour la description des entrées sorties, il est possible d'user de deux méthodes.

La première est dite « classique » et s'avère très simple à mettre en ?uvre. En effet, pour chaque entrée, on utilise IN BIT où IN veut dire que c'est une entrée et BIT qui veut dire que l'on a affaire à une seule entrée ne pouvant prendre que la valeur 0 ou 1. Cette dernière désignant l'état actif, la première l'état inactif.

Lorsque l'on doit définir une sortie, le IN cède sa place à un OUT et le BIT signifie toujours la même chose.
Voici un court exemple illustrant cela :

```
ENTITY nomprog IS
PORT (
  D0, D1, D2, D3 :
   IN BIT ;
  a, b, c , d :
   OUT BIT),
END nomprog ;
```

Le programme ci-dessus déclare donc quatre entrées (D0 à D3) et tout autant de sorties labellisées a à d.

La seconde méthode nécessite l'utilisation de bits vectors. Elle permet la définition rapide de plusieurs bits. Il faut définir le nombre de bits à l'intérieur de chaque vecteur ainsi que le sens de la numérotation. Afin d'éviter la confusion possible dans le sens des bits, il faut commencer par la valeur égale à nbits-1 et finir à 0 en utilisant downto.

Voici l'exemple d'un bit vector de 8 bits d'entrée appelé A :

```
A : IN BIT_VECTOR
  (7 downto 0) ;
```

Il est aisé d'utiliser des entrées comme des sorties comme le démontre le petit exemple ci-après :

```
ENTITY nomprog IS
PORT(
A :IN BIT_VECTOR
    (3 downto 0) ;
 -- Les quatre entrées
B : OUT BIT_VECTOR
    (7 downto 0) ;
 -- Les quatre sorties
END nomprog ;
```

L'architecture, quant à elle, correspond à la partie la plus complexe, celle qui au cœur du programme et qui se place juste après l'ENTITY.

Elle débute toujours de la même manière :

```
ARCHITECTURE
 nomarchitecture OF
   nomprog IS

BEGIN
```

Elle se termine toujours par :

```
END nomarchitecture ;
```

L'architecture correspond à la description de la fonctionnalité de notre FPGA. Il existe deux types de description qui peuvent cohabiter. La première est dite parallèle. En son sein, l'intégralité des opérations se déroule de manière concurrente. La seconde, dite séquentielle, permet à toutes les opérations de se dérouler dans l'ordre où elles sont décrites, au sein d'un process. Nous verrons cela en détail dès le prochain numéro. En attendant n'oubliez pas de réagir sur le forum du site internet du magazine.

Google

- vhdl tutorial
- vhdl gpl simulator

WIKIPÉDIA
L'encyclopédie libre

- VHDL
- Bascule
- FPGA

Si vous suivez le développement du CPCNG depuis le début et si vous avez réalisé un montage, envoyez-nous vos photos !