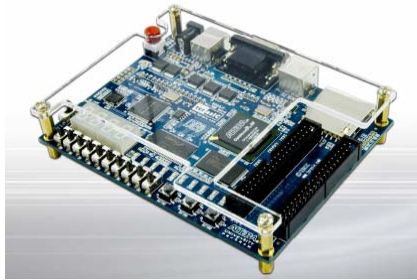


## TUTORIEL DU LOGICIEL QUARTUS D'ALTERA

version du 23 octobre 2012



### Table des matières

|  |           |
|--|-----------|
| <b>1 Créer un projet</b>   | <b>2</b>  |
| 1.1 Création d'un dossier Windows spécifique à votre projet . . . . .              | 2         |
| 1.2 Lancer le logiciel Quartus . . . . .   | 2         |
| 1.3 Créer un projet sous Quartus avec l'assistant de création . . . . .            | 2         |
| <b>2 Créer le ou les fichiers décrivant votre projet</b>                           | <b>3</b>  |
| 2.1 Création de fichier sous forme schématique . . . . .                           | 3         |
| 2.2 Création de fichier sous forme de fichier VHDL . . . . .                       | 5         |
| <b>3 Schéma associé à une fonction</b>   | <b>5</b>  |
| 3.1 Création de schéma associé à une fonction sous forme de fichier VHDL . . . . . | 5         |
| 3.2 Utilisation du schéma associé à une fonction . . . . .                         | 6         |
| <b>4 Travailler avec plusieurs fichiers</b>  | <b>6</b>  |
| 4.1 Travail avec plusieurs fichiers . . . . .                                      | 6         |
| 4.2 Ajouter ou éliminer des fichiers du projet . . . . .                           | 7         |
| <b>5 Compiler le projet</b>  | <b>7</b>  |
| <b>6 Configurer le composant</b>   | <b>7</b>  |
| <b>7 Tester le fonctionnement du composant configuré</b>                           | <b>8</b>  |
| <b>8 Compléments</b>   | <b>8</b>  |
| 8.1 Simulation . . . . .   | 8         |
| 8.2 Quelques astuces . . . . .   | 11        |
| <b>Interfaçage carte d'acquisition - Maquette DE0</b>                              | <b>12</b> |
| <b>Tableau des noms des broches sur la maquette DE0</b>                            | <b>16</b> |

L'environnement de développement permettant de programmer les composants reconfigurables d'Altera se nomme **Quartus**. Il permet de réaliser l'intégralité des étapes permettant de configurer un composant programmable. Ce tutoriel est écrit pour la version 9.0 de **Quartus** avec laquelle fonctionne la carte de développement DE0.

# 1 Créer un projet

## 1.1 Création d'un dossier Windows spécifique à votre projet

Pour chacun des projets ou des versions de projets sur lesquelles vous allez travailler, il est très fortement conseillé de créer un dossier Windows spécifique, dans lequel seront sauvegardés le projet et les différents fichiers créés. On veillera donc à changer de dossier dès qu'on change de projet ou de version de projet.

## 1.2 Lancer le logiciel Quartus

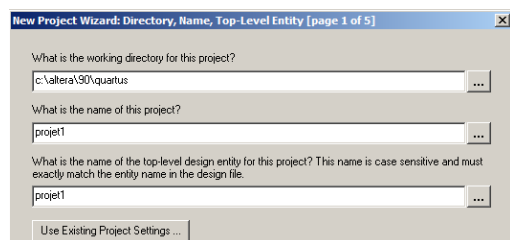
Lancez le logiciel **Quartus II 9.0 Web Edition**.

## 1.3 Créer un projet sous Quartus avec l'assistant de création

↪ Quartus fonctionnant par projet il est nécessaire de commencer par créer un projet à chaque début de conception.



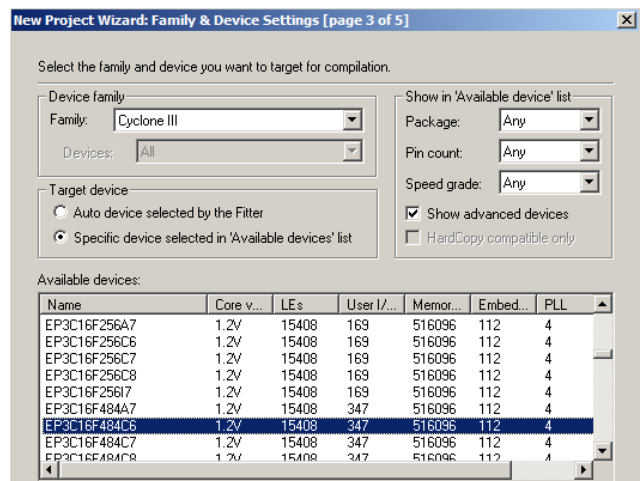
↪ Entrer ensuite dans le *Project Wizard* le nom choisi pour votre projet :



↪ Cliquer 2 fois sur **Next >**

↪ Choisir le FPGA présent sur votre carte en sélectionnant :

- La famille de FPGA : **Cyclone III**
- Le modèle : **EP3C16F84C6**




↪ Cliquer 2 fois sur **Next >** puis sur **Finish** .  
Le projet est créé.

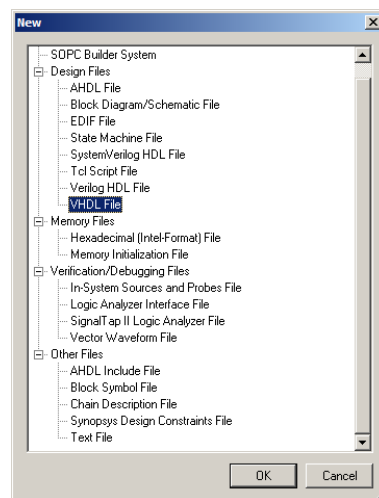
## 2 Créer le ou les fichiers décrivant votre projet


Un projet est généralement constitué de plusieurs fichiers décrivant chacun une ou plusieurs fonctions numériques. Parmi les possibilités proposées par Quartus pour créer des fichiers, nous allons en voir deux :

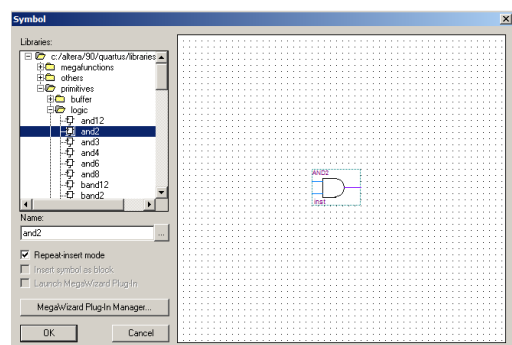
1. Création sous forme schématique : on parle alors de *Block Diagramm/Schematic File* (extension *.bdf* des fichiers), traitée au paragraphe 2.1.
2. Création sous forme textuelle en utilisant un langage de programmation de circuits : Langage VHDL (extension *.vhd*) ou bien langage Verilog (extension *.v*, langage non enseigné à l'IOGs), voir le paragraphe 2.2.

### 2.1 Création de fichier sous forme schématique

↪ *File>New* (ou *Ctrl N*) ou icône , puis sélectionner le type de fichier : *Block Diagram/Schematic file*




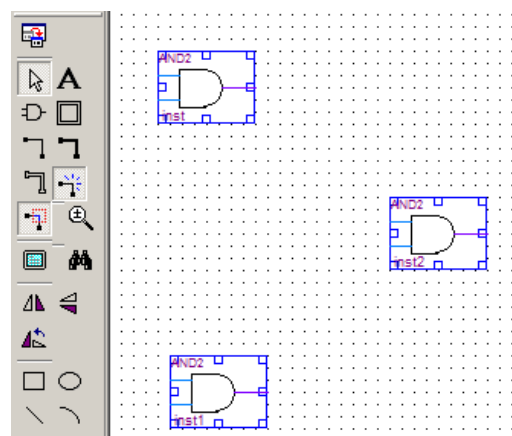
↪ Dans le fenêtre qui s'ouvre, sélectionner l'outil de sélection de composants , puis, dans la fenêtre *Symbol*, pour accéder par exemple, à une porte ET à 2 entrées, il faut descendre ensuite dans le menu de sélection de composants en choisissant : *C :/altera/quartus/librairies/primitives/logic*.




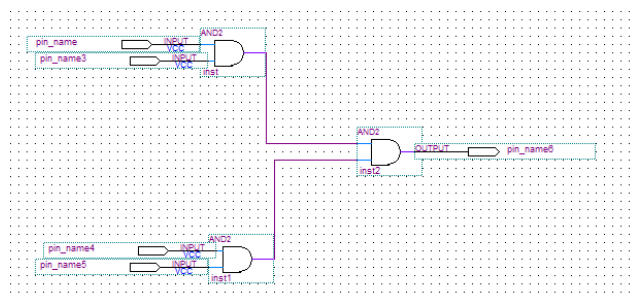
↪ Sélectionner le composant, puis cliquer sur *Ok*.

↪ On bascule alors sur le schéma ; chaque clic souris dessine une nouvelle porte ; on sort de ce mode par la touche *Echap* (ou *Esc*).

↪ Pour tracer des connexions, on sélectionne l'outil , et on relie les 2 terminaisons concernées entre elles :



↪ Il faut maintenant dessiner les entrées/sorties de la fonction logique (les **in** ou les **out** de la description VHDL). Pour cela sélectionner à nouveau l'outil  puis, dans la fenêtre **Symbol**, choisir le chemin *C :/altera/quartus/librairies/primitives/pin* et choisir *input* pour les entrées ou *output* pour les sorties. Reprendre les mêmes opérations que pour les portes pour la fin de ce schéma. On obtient alors :



À ce niveau,

**soit** vous souhaitez utiliser cette fonction dans d'autres fichiers (voir la commande **component** en VHDL, et la notion d'instanciation), dans ce cas lire le paragraphe **3**.

**soit** vous voulez relier votre fonction numérique à des broches physiques de la carte DE0. C'est l'objet de la suite de ce paragraphe.

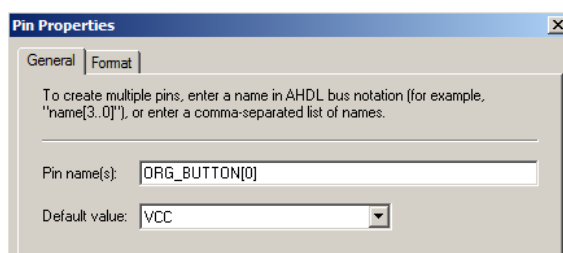
Afin de faire correspondre les entrées/sorties avec les signaux de la maquette il faut choisir leur nom se sorte qu'ils correspondent à des broches physiques de la carte. Pour cela, il existe un fichier descriptif de la carte, *DE0\_Pins\_Assignments.xls*. Ce fichier décrit les 484 broches du composant (d'où son nom) et nous permet de sélectionner facilement les broches de la carte. Le tableau **2** page **16** indique les noms des broches et leur correspondance sur la maquette DE0.

↪ Ce fichier est dans le répertoire */DE0 pour TPELEC/* que vous devez recopier à partir du serveur dans vos documents.

↪ Pour relier la 1ère entrée du schéma au bouton poussoir 0, on double-clique sur le *pin name* du connecteur d'*Input*, puis dans la fenêtre apparaissant on renseigne le nom en indiquant *ORG\_BUTTON[0]*. Puis on valide.

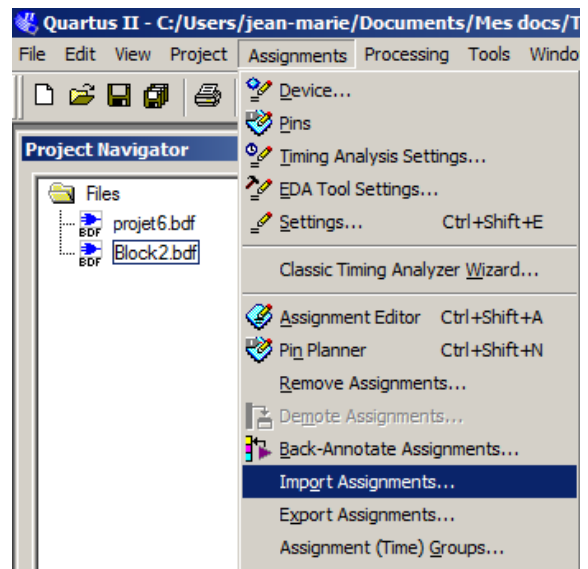
↪ Pour relier la 1ère entrée du schéma au Switch 8, on renseigne le nom en indiquant *SW[8]*.

↪ Pour relier la sortie du schéma à la Led verte 3 on renseigne le nom du connecteur *Output* en indiquant *LEDG[3]*.




Pour permettre au logiciel de faire la correspondance entre les noms des broches de la carte, et les broches du composant. Il faut importer le fichier *DE0\_Pins\_Assignments* dans le projet.

- ↪ Pour cela, choisir **Assignments > Import Assignments**, puis aller sélectionner le fichier à inclure.



- ↪ La conception de votre 1er circuit sous forme graphique se termine ici. Reste à compiler le projet puis à la télécharger sur la carte. Vous pouvez donc, pour ce 1er circuit, passer directement au paragraphe 5.

## 2.2 Création de fichier sous forme de fichier VHDL

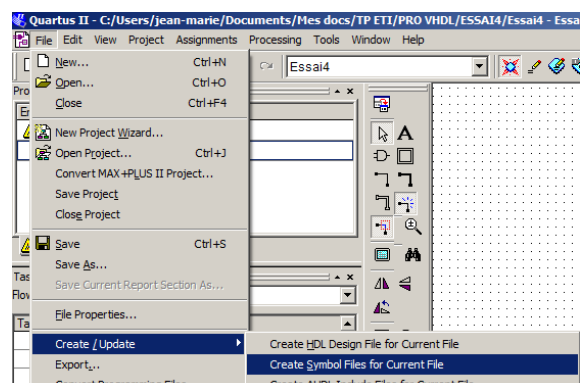
- ↪ **File>New** (ou **Ctrl N**) ou icône , puis sélectionner le type de fichier : **VHDL File**.
- ↪ Tapez le code VHDL.
- ↪ Pour associer les Entrées /sorties de votre code VHDL à des broches physiques du composant, le plus simple est de créer un schéma bloc associé à votre code VHDL (voir paragraphe 3)
- ↪ Puis de relier les broches de ce schéma bloc aux broches physique comme indiqué dans le paragraphe 2.1. Il vous faut donc d'abord passer à la partie 3.

## 3 Schéma associé à une fonction

Quartus proposant une interface graphique, il est utile de pouvoir associer n'importe quelle fonction à un schéma; cela s'applique indifféremment aux fonctions décrites sous forme de schéma, ou sous forme de code VHDL. On peut ensuite associer entre elles les nouvelles fonctions décrites sous formes de schémas, et ainsi de suite. On voit donc apparaître la notion de conception hiérarchique.

### 3.1 Création de schéma associé à une fonction sous forme de fichier VHDL

- ↪ Ouvrir le fichier (schéma ou code VHDL)
- ↪ Sélectionner **Create/Update > Create Symbol File for current File**, voir la figure ci-contre (nb : pour pouvoir créer un schéma associé à un fichier, le fichier doit avoir été sauvegardé)
- ↪ Sauvegarder le schéma (fichier **.bdf**) ainsi créé dans le dossier du projet en lui donnant un nom explicite différent du nom du fichier VHDL ainsi que de celui de l'entité (*entity*).



## 3.2 Utilisation du schéma associé à une fonction

↪ Créer un nouveau fichier Block Diagram/Schematic (ou ouvrir un fichier existant)

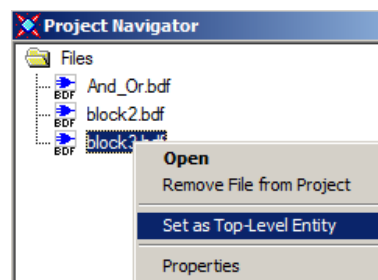
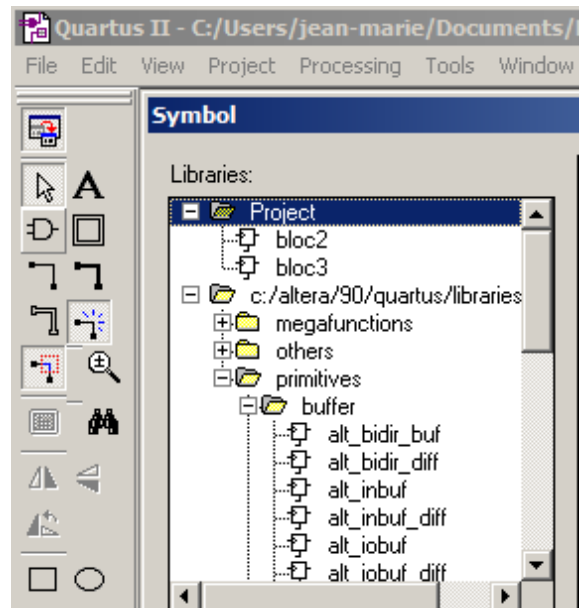
↪ Sélectionner l'outil Symbol Tool 

↪ Ouvrir le dossier *Project* et sélectionner le composant par son nom

↪ Se servir ensuite de ce composant comme d'un composant standard dans des *Block Diagram/Schematic* de plus haut niveau.

↪ On peut par exemple associer les broches physiques de la carte aux Entrées/sorties d'un code écrit en VHDL.

↪ Ne pas oublier de désigner le fichier "schéma" (.bdf) comme étant le fichier de plus haut niveau. Pour cela, sélectionner dans le *Project Navigator* le fichier de plus haut niveau ; faire un clic droit et choisir *Set as Top-Level Entity* comme sur la figure ci-contre.



## 4 Travailler avec plusieurs fichiers

Cette partie ne concerne pas vos tous premiers développements.

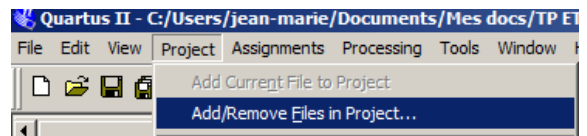
### 4.1 Travail avec plusieurs fichiers

Pour le développement d'un projet, on est très rapidement amené à avoir plusieurs fichiers. **Quartus** nous permet d'utiliser des fichiers de tous types (graphiques ou textuels). Pour les associer entre eux, le plus simple est d'associer chaque fichier textuel à un schéma puis d'associer les schémas entre eux, et enfin de nommer les entrées-physiques sur le schéma global avec les noms adéquats (voir tableau 2 page 16). Le mode opératoire est donc :

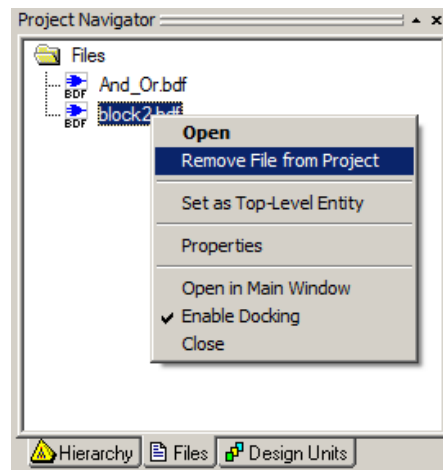
1. Créer les différents fichiers
2. Associer chaque fichier à un schéma
3. Associer les schémas entre eux
4. Désigner le fichier de plus haut niveau (*Set as Top-Level Entity*)

## 4.2 Ajouter ou éliminer des fichiers du projet

↪ Pour ajouter (ou d'ailleurs éliminer) des fichiers au projet, on sélectionne **Project > Add/Remove Files in Project** puis on choisit les divers fichiers à inclure.



↪ Pour éliminer des fichiers du projet, on peut aussi aller dans la fenêtre de *Project Navigator*, sélectionner le fichier, puis **Remove File from Project**.



A noter qu'un fichier éliminé du projet n'est nullement effacé de l'ordinateur et qu'on pourra le ré-inclure dans ce projet ou dans un autre ultérieurement.

## 5 Compiler le projet

La compilation va générer la configuration du composant programmable qui permettra la réalisation concrète du projet. Pour cela :

- sélectionner l'icône idoine (la flèche violette) dans le menu



- ou dans la barre de menu, choisir *Processing > Start Compilation*,
- ou encore en tapant **Ctrl L**.

On peut ne pas lire le rapport de compilation dans la fenêtre de Messages (en bas de la fenêtre **Quartus**) lors des premières programmations. Si la compilation n'est pas couronnée de succès, il va falloir trouver et corriger les erreurs dans les fichiers sources. Sinon, on peut passer à la programmation du composant, paragraphe 6.

Remarques sur la fenêtre de *messages* et celle de *Flow Summary* : de façon générale, un grand nombre d'informations vous sont données sur le bon (ou mauvais déroulement de la compilation), mais dans un 1er temps leur lecture n'est pas indispensable. Vous y verrez un nombre important de warnings : ils ne sont pas forcément critiques. Enfin la fenêtre de *Flow Summary* vous donne un résumé sur les ressources occupées par votre fonction sur votre composant.

## 6 Configurer le composant

Pour configurer le composant afin d'obtenir le fonctionnement décrit par le projet, il faut impérativement :

- que la maquette DE0 soit reliée à l'ordinateur par câble USB,

- qu'elle soit sous tension (bouton rouge),
- que le commutateur RUN / PROG soit positionné sur RUN

Il est de plus préférable d'avoir testé la validité de la chaîne de développement ainsi que celle de la connexion USB, à l'aide du programme de test fourni, *DEO Control Panel* ou par la programmation de la carte à l'aide d'un programme déjà réalisé et présent sur DEO pour TPELEC\New Files, *de0\_debounce\_cnt*.

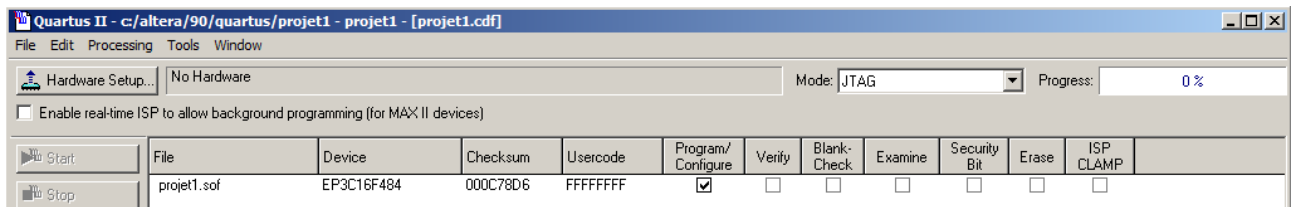
Pour configurer le composant de la maquette DEO :

↪ sélectionner l'icône de programmation,



↪ ou bien Sélectionner *Tools > Programmer*

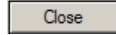
↪ La fenêtre de "programmation" s'ouvre alors.



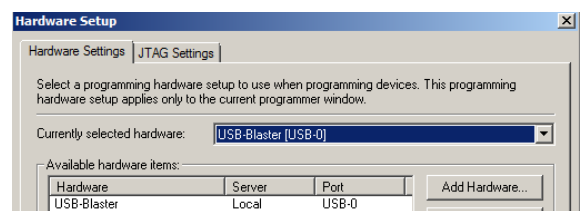
↪ Cliquer sur *Hardware Setup*

↪ Puis dans la nouvelle fenêtre sélectionner *USB-Blaster* à la place de *No Hardware* dans le menu *Currently Selected Hardware*.

Fermer cette fenêtre



↪ Enfin, cliquer sur *Start* pour lancer la configuration du circuit de la maquette DEO.



## 7 Tester le fonctionnement du composant configuré

Pour valider la conception, il faut tester le comportement électronique et le comparer au cahier des charges. En cas de fonctionnement non-conforme, il faudra (sûrement) revenir à la conception des fonctions du projet.

## 8 Compléments

### 8.1 Simulation

La simulation est l'étape qui vous permet de valider un design avant de le programmer sur la carte. Il est en effet souvent bien plus facile de trouver une erreur sur des résultats de simulation plutôt que sur le circuit. De plus, programmer une configuration de circuit qui ne correspond pas à ce que vous deviez concevoir est une perte de temps. En dehors des cas les plus simples, il faut donc simuler après avoir compilé et avant de programmer.

Pour simuler, vous allez devoir définir précisément le type de simulation voulu, la durée de cette simulation, les entrées et les sorties que vous voulez observer.

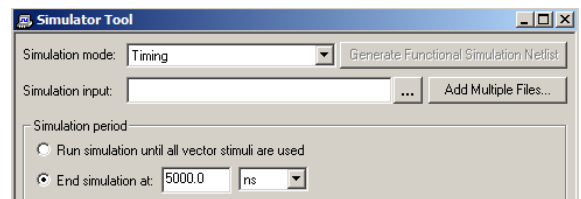


Il faudra aussi choisir les signaux qu'on veut observer, et imposer une configuration aux entrées. Cela consiste à définir les "vecteurs de tests", ce que nous ferons en configurant un fichier *Vector Wave File* (extension *.vwf*). Enfin se pose la question de savoir si les tests sont exhaustifs : c'est un problème qui peut être (très) compliqué et qui n'est pas abordé dans ce document.

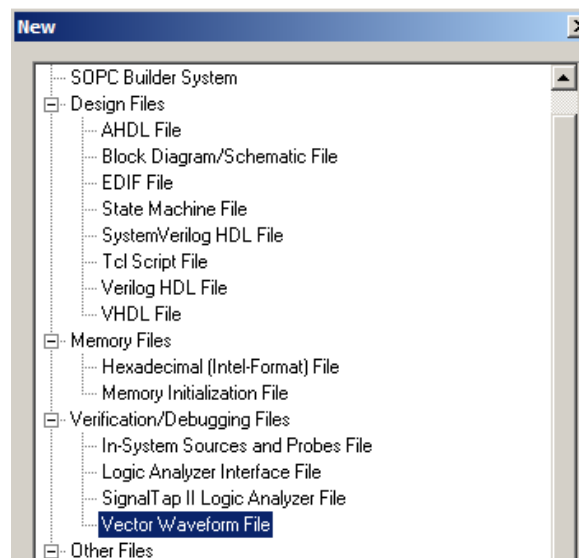
## Configurer le simulateur

↪ Lancer le simulateur par le menu *Processing > Simulator Tool*,

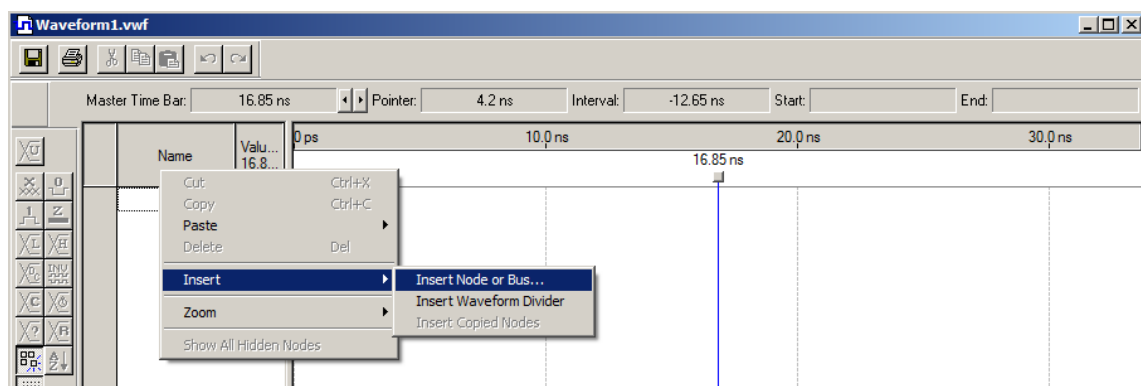
↪ Compléter les champs *Simulation mode* et *End Simulation*. Au niveau des modes de simulation, *Functionnal* correspond à une simulation avec des portes idéales, et *Timing* à une simulation en tenant compte des retards technologiques. On vous conseille de choisir ce dernier mode.

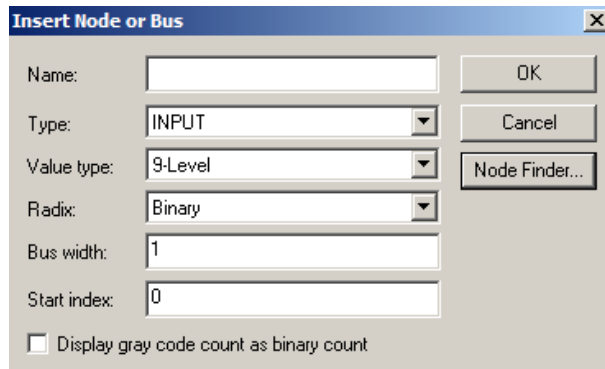


↪ Il va falloir créer un fichier de vecteur de tests. On va pour cela, aller dans *File > New* (ou *Ctrl N*) et sélectionner *Vector Wave File*.

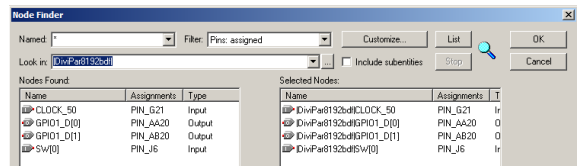


↪ Dans la nouvelle fenêtre, clic droit sur *Name* pour ajouter des signaux, puis sélectionner *Insert > Insert Node or Bus* et sélectionner le menu *Node Finder*.

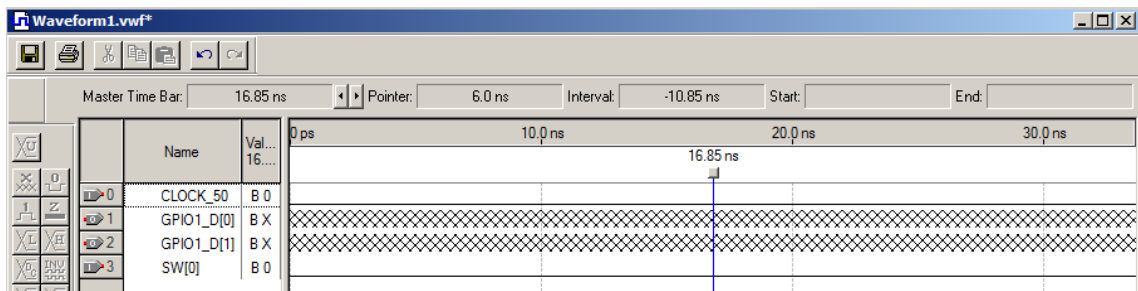




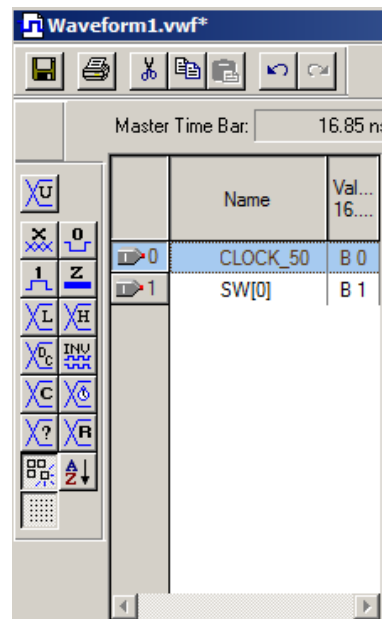
↪ On ajoute les signaux qui nous intéressent en les faisant apparaître par double-clic (ou encore, par sélection puis >).



L'interface du simulateur s'ouvre alors :



↪ Il faut maintenant imposer une valeur pour chaque entrée. On sélectionne un signal, puis les icônes de la colonne de gauche passent en bleu, et on impose des valeurs, ou bien par Clic Droit, on choisit dans le menu contextuel.



**Quelques remarques :** – Pour imposer un 0 ou un 1 pendant juste une durée, sélectionner cette durée dans le panneau de droite puis cliquer sur l'icône associée dans le menu.  
 – Pour l'horloge, sélectionner le signal voulu puis cliquer sur l'icône de signal périodique.

- Quelques raccourcis clavier du simulateur :
  - Ctrl W** : affiche tout l'intervalle de simulation
  - Ctrl Espace** : Zoom
  - Ctrl Shift Espace** : Dézoom

## Lancer la simulation

- ↪ Lancer la simulation par l'icône *Start* du *Simulator Tool*
- ↪ Pour observer les résultats, cliquer sur *Report*
- ↪ Pour modifier le fichier de vecteur de tests, cliquer sur les résultats, cliquer sur *Open*.



## 8.2 Quelques astuces

### Projets

Lorsqu'on n'est pas directement au démarrage de **Quartus**, pour créer un nouveau projet, on peut sélectionner le menu *File > New Project Wizard*.

Pour ouvrir un projet existant, on peut faire *File > Open Project*, ou bien *Ctrl J*.

Pour fermer un projet ouvert, faire *File > Close Project*.

### Commandes de schéma

Pour déplacer un symbole, cliquer dessus et le déplacer.

Pour déplacer l'ensemble du schéma dans la fenêtre active, faire *Ctrl A* et déplacer


Pour copier un symbole, procéder par *Ctrl C* puis *Ctrl V*.

Pour détruire un symbole, procéder sélection puis *Suppr*.

Pour modifier le nom d'un signal, double cliquer sur le nom puis modifier.

### Notion de bus

Les signaux numériques se présentent souvent sous forme de bus (associations de plusieurs fils par exemple lorsqu'il représentent un mot numérique. Il est aussi parfois judicieux d'associer plusieurs signaux booléens et de créer un bus.

Sous l'éditeur de schéma de **Quartus**, on trace les bus avec l'outil .

Pour nommer ces bus on utilisera par exemple `:HEX2_D[6..0]` pour nommer les 6 fils du bus d'entrée d'un des afficheurs 7 segments (donc de sortie du FPGA), ou encore `SW[2..0]` pour le bus associé à 3 switches. On choisit la taille souhaitée pour le bus.

### Du schéma au VHDL

A partir d'un schéma, **Quartus** peut générer un code VHDL. Pour cela, dans l'éditeur de schéma, sélectionner le menu *Create/Update > Create HDL Design* : Le code VHDL sera ensuite généré automatiquement.

# Interfaçage carte d'acquisition - Maquette DE0

## Bornier partagé

Le bornier permet de relier la carte d'acquisition NI USB 6009 au monde physique par l'intermédiaire de câbles BNC. Ce même bornier va aussi être utilisé pour relier la carte DE0 au monde physique.

On a donc 3 modes de fonctionnement :

1. DE0 reliée au bornier (carte d'acquisition déconnectée)
2. carte d'acquisition reliée au bornier (carte DE0 déconnectée)
3. carte d'acquisition et DE0 reliées aux bornier, donc reliées ensemble.

Tout dépend de la façon dont vous avez choisi de relier les connecteurs en nappe.

Au niveau de la DE0, 2 connecteurs d'extension sont disponibles, totalisant jusqu'à 72 entrées-sorties possibles :

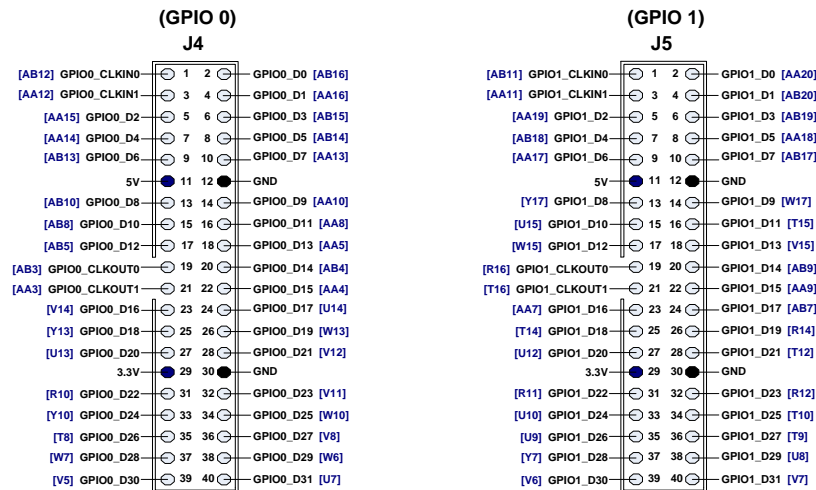


FIGURE 1 – Brochage des connecteurs d'extension de la carte DE0

En ce qui nous concerne, nous ne nous servirons que du connecteur GPIO1, et seulement d'un sous-ensemble de ce connecteur :

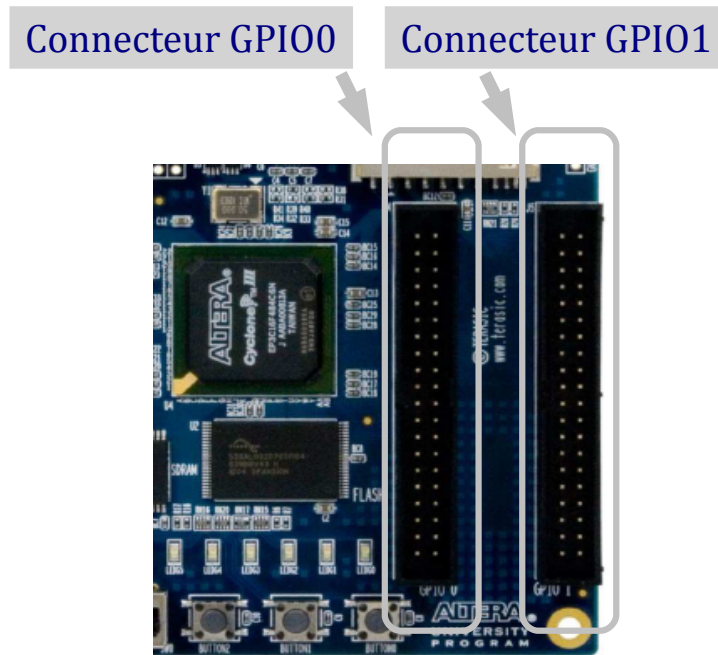


FIGURE 2 – Connecteurs d’extension de la carte DEO

| GPIO1 vers NI USB |         | NI USB vers GPIO1 |            |
|-------------------|---------|-------------------|------------|
| GPIO1             | NI 6009 | NI 6009           | GPIO1      |
|                   | port 0  | port 0            |            |
|                   |         | P0.0              | -          |
| D[0]              | P0.3    | P0.1              | CLK_OUT[0] |
| D[1]              | P0.5    | P0.2              | CLK_OUT[1] |
| D[2]              | P0.6    | P0.3              | D[0]       |
| D[3]              | P0.7    | P0.4              | CLK_IN[1]  |
| CLK_OUT[0]        | P0.1    | P0.5              | D[1]       |
| CLK_OUT[1]        | P0.2    | P0.6              | D[2]       |
| CLK_IN[1]         | P0.4    | P0.7              | D[3]       |
|                   | port 1  | port 1            |            |
| D[4]              | P1.0    | P1.0              | D[4]       |
| D[5]              | P1.1    | P1.1              | D[5]       |
| D[6]              | P1.2    | P1.2              | D[6]       |
| D[7]              | P1.3    | P1.3              | D[7]       |

TABLE 1 – Connexions entre la carte DEO et la carte d’acquisition NI USB 6009 sur le bornier des TPs.

Pour faciliter les connexions, le bornier à une face avant nommant explicitement les entrées/sorties, avec la convention :

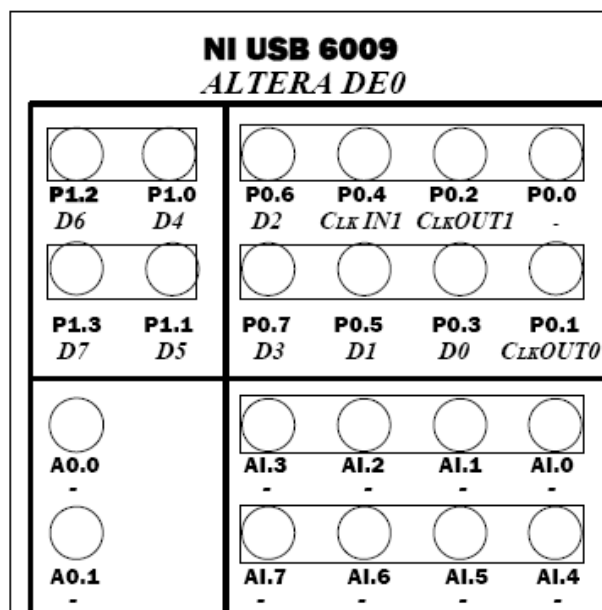


FIGURE 3 – Bornier

**Logiciel Measurement&Automation** Ce logiciel permet d’acquérir ou de générer des signaux analogiques ou numériques sur la carte d’acquisition NI USB 6009

La carte comprend comprend 4 blocs bien distincts :

- un bloc d’entrées analogiques : AI, au nombre de 8,
- un bloc de sorties analogiques : AO, au nombre de 2,
- un bloc d’entrées/sorties numériques : Port 0 (8 E/S) ou Port 1 (4 E/S) pouvant être configurées en entrée ou en sortie à la demande,
- un bloc E/S de compteur : PFIO

Une fois le logiciel lancé, cliquer sur « panneau de test », puis choisir dans un des 4 onglets (correspondant aux blocs cités ci-dessus, une (ou plusieurs) voie(s), et la (les) configurer.

**Pour les entrées analogiques,** noter le mode différentiel ou RSE (soit référencé à la masse), les valeurs maxi des entrées, et le mode (sur demande, en continu, ou fini). Pour le mode en continu fini, préciser la fréquence d’échantillonnage et le nombre de points à acquérir.

**Pour les sorties analogiques,** il n’y a que les limites à fixer.

**Pour les entrées-sorties numériques,** choisir le nom du port (0 ou 1), configurer bit à bit entrée ou sortie, sélectionner (pour les sorties) la valeur des différents bits.



## Tableau des noms des broches sur la maquette DE0

| Nom  | Mode             | Commentaires   |
|--|------------------|--|
| Broches du connecteur d'extension n°0 (GPIO0)<br>(GPIO = General Purpose Input Output)               |                  |  |
| GPIO0_CLKIN[1]   | Entrée           | Entrée n°1 d'un signal d'horloge   |
| GPIO0_CLKIN[0]   | Entrée           | Entrée n°0 d'un signal d'horloge   |
| GPIO0_CLKOUT[1]  | Sortie           | Sortie n°1 d'un signal d'horloge   |
| GPIO0_CLKOUT[0]  | Sortie           | Sortie n°0 d'un signal d'horloge   |
| GPIO0_D[i]   | Bidirectionnelle | Données d'entrée ou de sortie n°i (i varie de 0 à 31).<br>Exemple : GPIO0_D[0] pour le bit n°0.                                      |
| Broches du connecteur d'extension n°1 (GPIO1)  |                  |  |
| GPIO1_CLKIN[1]   | Entrée           | Entrée n°1 d'un signal d'horloge   |
| GPIO1_CLKIN[0]   | Entrée           | Entrée n°0 d'un signal d'horloge   |
| GPIO1_CLKOUT[1]  | Sortie           | Sortie n°1 d'un signal d'horloge   |
| GPIO1_CLKOUT[0]  | Sortie           | Sortie n°0 d'un signal d'horloge   |
| GPIO1_D[i]   | Bidirectionnelle | Données d'entrée ou de sortie numéro i (i varie de 0 à 31).<br>Exemple : GPIO0_D[0] pour le bit n°0.                                 |
| Afficheurs 7 segments (de 0 à 3)<br>7 LEDs (de 0 à 6) par afficheur + 1 DP (dot point) par afficheur |                  |  |
| HEXj_D[i]  | Sortie           | LED n°i de l'afficheur 7 segments n°j<br>Exemples :<br>HEX0_D[6] : Led 6 de l'afficheur n°0,<br>HEX3_D[6] : Led 6 de l'afficheur n°3 |
| HEXj_DP  | Sortie           | LED de Dot Point de l'afficheur 7 segments n°j<br>Exemple :<br>HEX4_DP : Led DP de l'afficheur n°4                                   |
| Switchs (de 0 à 9)   |                  |  |
| SW[i]  | Entrée           | Switch n°i (i variant de 0 à 9)<br>Exemple :<br>SW[6] pour le switch n°6   |
| Boutons poussoirs (de 0 à 2)   |                  |  |
| ORG_BUTTON[i]  | Entrée           | Bouton poussoir n°i (i variant de 0 à 2)<br>Exemple :<br>ORG_BUTTON[2] pour le switch n°2  |
| Leds Vertes (de 0 à 9)   |                  |  |
| LEDG[i]  | Sortie           | LED verte n°i (i variant de 0 à 2)<br>Exemple :<br>LEDG[6] pour la LED n°6   |

TABLE 2 – Noms des broches du circuit de la maquette DE0.