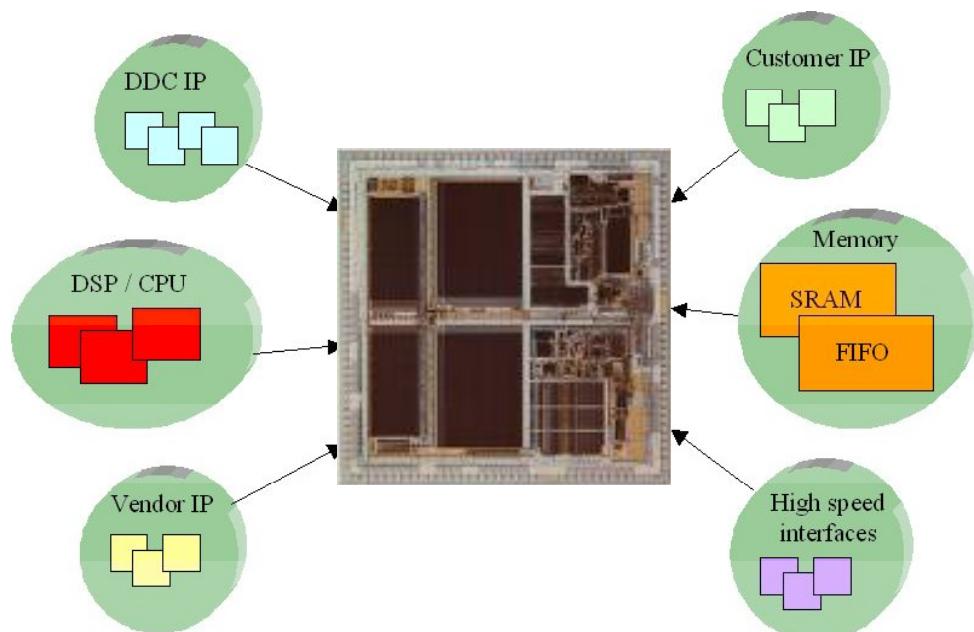


# Mini-Projet

## Radar 2D



# Objectifs

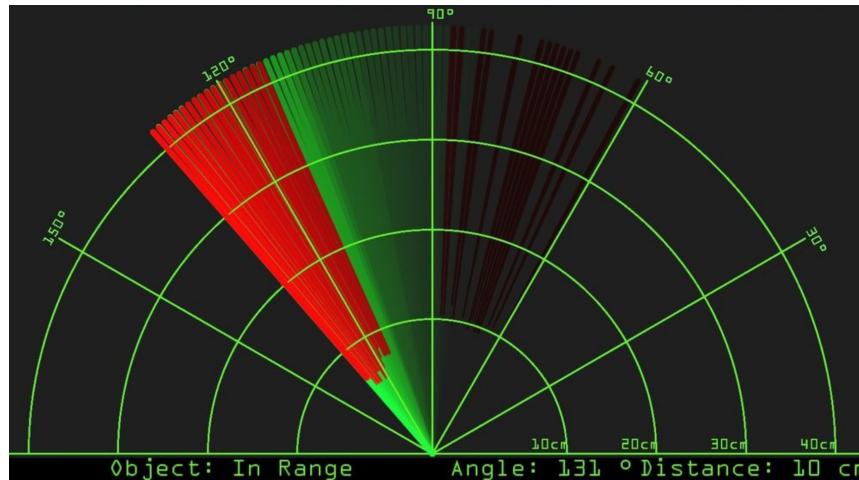
L'objectif de ce mini-projet est de cartographier une scène (Radar 2D) à l'aide d'un télémètre ultrason et un servomoteur.

L'organigramme de ce mini-projet est le suivant :



Dans un premier temps, il sera nécessaire de concevoir une IP télémètre à ultrasons. Cette étape commencera par le développement de la partie opérative (sans intégration avec le bus Avalon) et sa validation à l'aide d'une simulation sur Modelsim, suivie d'un test sur le FPGA pour consolider la vérification de cette partie. Par la suite, une interface avec le bus Avalon sera ajoutée pour intégrer l'IP dans un système basé sur Nios II en tant que périphérique personnalisé (Custom Peripheral).

Ensuite, une IP dédiée au contrôle d'un servomoteur devra être développée. Enfin, il sera nécessaire de concevoir une application permettant de cartographier une scène en faisant tourner le servomoteur tout en effectuant des mesures régulières grâce au télémètre à ultrasons.



#### **Modalités d'évaluations :**

Il faudra faire valider les différentes étapes par l'enseignant lorsque c'est demandé.

Il faut rédiger un compte rendu de projet qui répond aux différentes questions posées dans le sujet et qui illustre les différentes manipulations, codes (C et VHDL) et simulations que vous allez réaliser pour atteindre les objectifs du projet.

## Etape 1 : Implémentation de l'IP télémètre ultrason HC SR04 (5 points)

### 1.1 Introduction : Présentation du capteur HC SR04

#### Caractéristiques

- Plage de mesure : 2 cm à 400 cm
- Résolution de la mesure : 0.3 cm
- Angle de mesure efficace : 15 °
- Largeur d'impulsion sur l'entrée de déclenchement : 10 µs (Trigger Input Pulse width)

#### Broches de connection

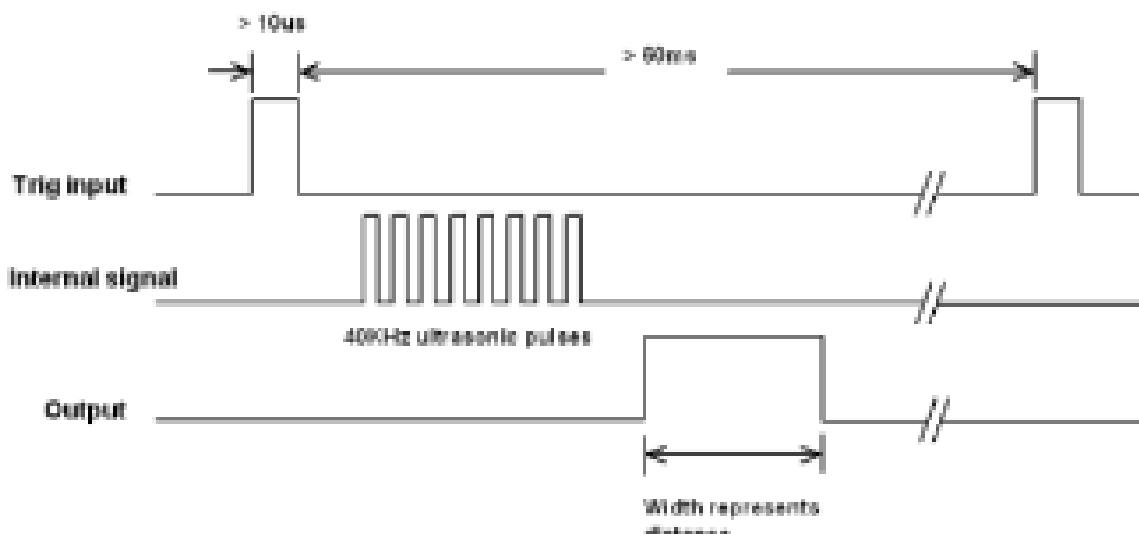
- Vcc = Alimentation +5 V DC
- Trig = Entrée de déclenchement de la mesure (Trigger input)
- Echo = Sortie de mesure donnée en écho (Echo output)
- GND = Masse de l'alimentation

#### Spécifications et limites

Paramètre	Min	Type	Max	Unité
Tension d'alimentation	4.5	5.0	5.5	V
Courant de repos	1.5	2.0	2.5	mA
Courant de fonctionnement	10	15	20	mA
Fréquence des ultrasons	-	40	-	kHz

#### Fonctionnement

Pour déclencher une mesure, il faut présenter une impulsion "high" (5 V) d'au moins 10 µs sur l'entrée "Trig". Le capteur émet alors une série de 8 impulsions ultrasoniques à 40 kHz, puis il attend le signal réfléchi. Lorsque celui-ci est détecté, il envoie un signal "high" sur la sortie "Echo", dont la durée est proportionnelle à la distance mesurée.



Note : A grande distance, la surface de l'objet à détecter doit mesurer au moins 0.5 m<sup>2</sup>. S'il n'y a pas d'obstacle, l'écho dure environ 40 ms. Il faut attendre au moins 60 ms entre chaque nouvelle mesure.

## 1.2 Travail à réaliser

### 1.2.1 Simulation de l'IP

Dans un premier temps, vous allez décrire la partie opérative de l'IP et la simuler dans Modelsim en écrivant un banc de test et un script de simulation.



*Reporter dans le compte-rendu de projet les captures d'écrans de la simulation de l'IP avec les explications sur son fonctionnement et les tests réalisés.*

*Remarque : On choisit un reset (rst\_n) actif à l'état bas pour faciliter la suite.*

### 1.2.2 Test de l'IP sur carte

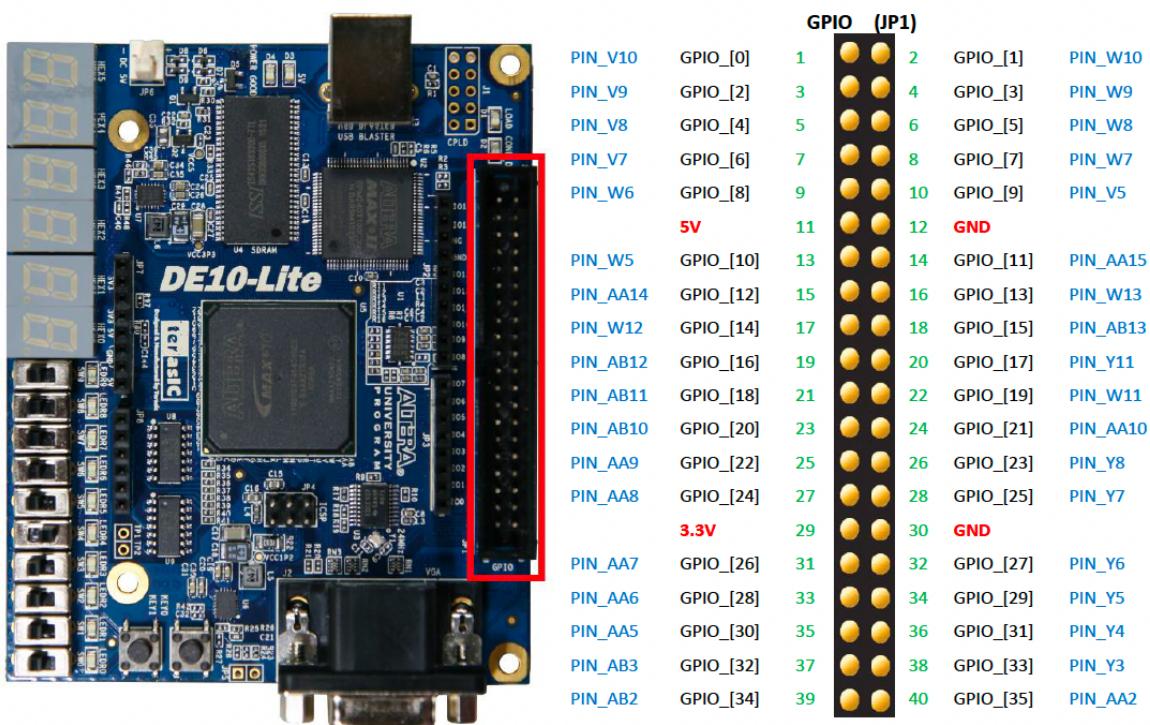
Avant de rajouter l'IP dans le SoPC architecturé autour du Nios2 à l'aide de Platform Designer, vous allez tester l'IP uniquement sur la partie FPGA (sans Nios2) pour vérifier qu'elle fonctionne bien en « standalone ».

Pour cela, vous allez partir d'un nouveau projet Quartus pour la carte DE10\_Lite. Vous allez y rajouter le fichier source de votre télémètre qui sera considéré comme le Top Level .

Connecter les entrées-sorties de cette IP de la manière suivante :

Signaux de l'entité telemetre_us_HC_SR04	Signaux de la carte DE10_Lite (voir User Manual)	Pinout
Rst_n	KEY0	PIN_B8
CLK	MAX10_CLK1_50	PIN_P11
Trig	GPIO[1]	PIN_W10
Echo	GPIO[3]	PIN_W9
Dist_cm (9 downto 0)	LEDR[9..0]	Voir page 27 du DE10-Lite_User_Manual.pdf

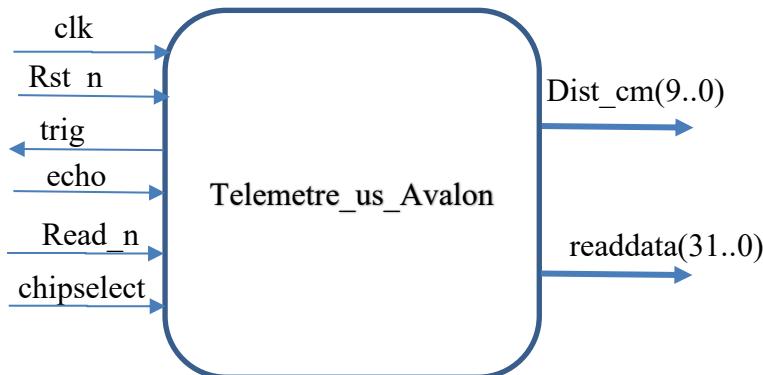
Pour alimenter le télémètre HC\_SR04, il faut utiliser les pins 5V et GND du connecteur GPIO (JP1) (voir figure ci-dessous).



Testez votre IP sur la carte DE10\_Lite. Vérifiez que les valeurs des LEDs évoluent en fonction de la distance d'un obstacle tenu devant le télémètre. Prenez des photos à rajouter dans votre compte rendu pour montrer que cette partie fonctionne bien.

### 1.2.3 Intégration de l'IP Télémètre dans Platform Designer

Rajouter l'interface Avalon (voir les signaux de l'entité ci-dessous) à votre IP Télémètre Ultrason afin de pouvoir le connecter au processeur NIOS II.



Dans un premier temps, tester votre IP par simulation pour vous assurer que l'interface Avalon fonctionne bien.

*Reporter dans le compte-rendu de projet les captures d'écrans de la simulation de l'IP avec les explications sur les tests réalisés.*

Ensuite, il faut intégrer votre IP dans Platform Designer (Custom Peripheral, cf TP2) en prenant le projet source qui se trouve sur Moodle : Source DE10\_Lite\_Computer System qui intègre le processeur et l'ensemble des périphériques nécessaires.

**Pour utiliser ce projet correctement, il faut utiliser la version 18.1 de Quartus ainsi que l'exécutable fournit par Intel-Altera University program : « intel\_fpga\_upds\_setup.exe » Si vous n'utilisez pas la version 18.1 et que vous n'avez pas installé l'exécutable de Intel-Altera, celui-ci est disponible à partir du lien suivant :**

<https://dropsu.sorbonne-universite.fr/s/98M6RrPFJ3pOpTS>

L'intégration d'une nouvelle IP dans un système SoC-FPGA existant se fait par l'intermédiaire d'un Custom Peripheral. Cela consiste à connecter les E/S de votre IP version Avalon avec les signaux disponibles sur le bus Avalon.

Dans Platform Designer, ajouter une IP de type Custom Peripheral (comme vu pendant les TP). Dans le menu **File → New component**, importez fichier VHDL de votre IP (onglet Files).

Connectez les signaux d'entrées / sorties de votre IP avec les signaux du bus Avalon (Onglet Signals ans Interfaces).

Il faut rajouter une interface **conduit\_end** en cliquant sur <<add interface>> pour les signaux Trigger, Echo et Dist\_Cm.

Générer le nouveau système. Modifier le code VHDL du top-level, synthétisez et implémentez le système sur le FPGA.

#### *1.2.4 Programmation logicielle et test de l'IP*

Maintenant que vous avez intégré la nouvelle IP Télémètre, vous allez programmer le processeur Nios2 pour qu'il acquiert une mesure de distance et qu'il l'affiche dans le terminal de la console du NIOS II.

Dans le code en langage C, il faut utiliser la fonction IORD(adresse de votre IP) ou plus concrètement la fonction IORD(IP\_TELEMETRE\_AVALON\_BASE,0) ;

Dans un second temps, vous pouvez afficher la distance en cm sur les afficheurs 7 segments.

Prenez des photos à rajouter dans votre compte rendu pour montrer que cette partie fonctionne bien.

## Etape 2 : Conception de l'IP Servomoteur (4 points)

Vous devez réaliser une IP qui permet de contrôler un servomoteur. Cette IP permettra de commander la position d'un servomoteur depuis le NIOS II.

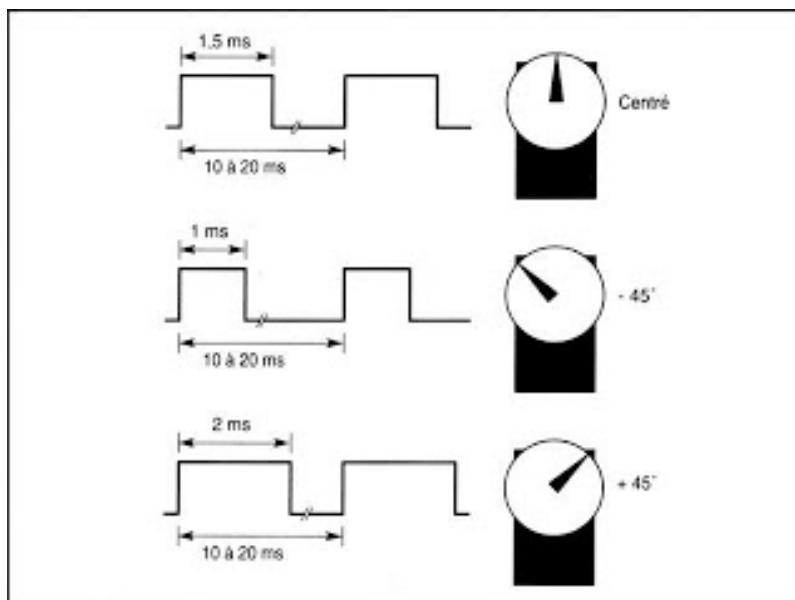
### 2.1 Comment fonctionne un servomoteur

Le principe de base est assez simple. Il suffit d'envoyer une impulsion et c'est le temps que durera cette impulsion qui déterminera l'angle du servo-moteur. Ce temps d'impulsion est de quelques millisecondes et doit être répété à intervalle régulier (toutes les 10 à 20 ms).

Si le temps d'impulsion varie d'un fabricant à l'autre, les valeurs suivantes sont assez standard :

- 1 ms = 0 degré
- 1.50 ms = 45 degrés
- 2 ms = 90 degrés

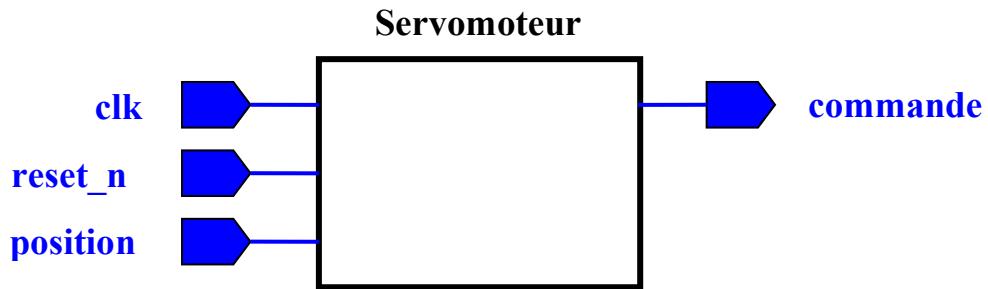
Le graphique ci-dessous montre la correspondance entre la longueur d'impulsion et l'angle du servo-moteur.



### 2.2 Conception de la partie opérationnelle du composant Servomoteur

Dans un premier temps vous allez réaliser un composant VHDL qui doit permettre de commander le servomoteur depuis les interrupteurs (switch) afin de vérifier la partie opérationnelle de votre IP.

L'entité à développer en VHDL devra être conforme aux spécifications données ci-dessous :



L'IP devra comporter en entrée :

- un signal position sur 8 ou 10 bit qui permet de choisir la position du servomoteur en degrés.
- un signal reset actif à l'état bas.
- une horloge système à 50 MHz.

L'IP devra comporter en sortie :

- un signal commande d'un bit connecté sur l'entrée du servo-moteur.

### *2.2.1 Développement de l'IP Servomoteur*

Vous êtes libre de concevoir cette IP comme vous le souhaitez. C'est-à-dire, soit de manière comportementale, soit de manière structurelle, soit avec une machine à état.

Expliquez l'architecture de votre IP sur votre compte rendu.

### *2.2.2 Simulation et validation*

Effectuer la simulation de l'IP avec Modelsim. Il est demandé d'écrire un banc de test et un script de simulation qui doit permettre de vérifier rapidement la validité du fonctionnement de l'IP.

Reportez vos résultats de simulation sur votre compte-rendu de projet.

### *2.2.3 Vérification sur la carte DE1-SoC*

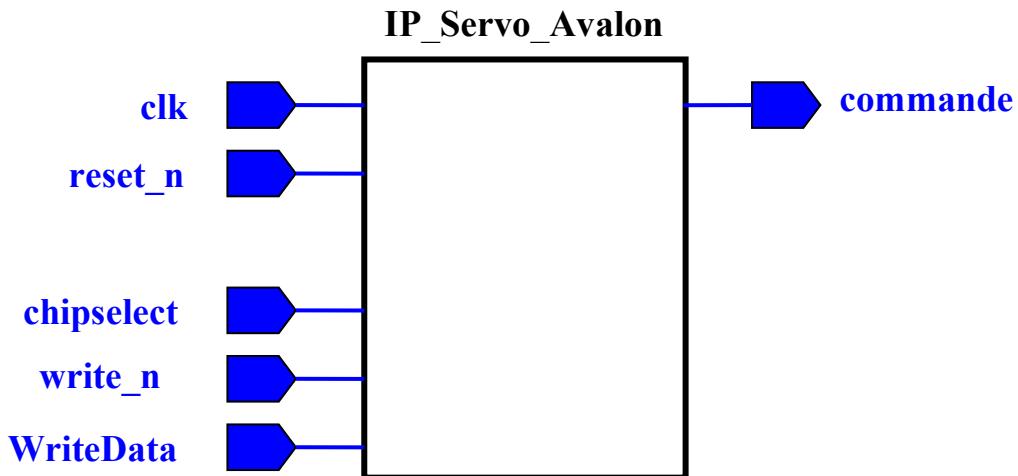
Tester votre IP sur la carte DE10-Lite. Pour cela, vous devez commander la position du servomoteur à partir des interrupteurs (switch). Configurez la sortie commande du servomoteur sur la broche PIN\_V10 du FPGA, ce qui correspond à la broche GPIO\_[0].

A l'aide d'un oscilloscope, vérifiez que le timing de la sortie commande correspond bien aux valeurs attendues et ensuite testez avec un servomoteur.

**Faites vérifier cette étape par l'enseignant.**

## 2.3 Extension de l'IP Servomoteur vers une version connectable au bus Avalon

Dans cette partie, nous allons modifier l'IP conçue auparavant afin que son interface prenne en compte les signaux disponibles sur le bus Avalon. L'entité à développer en VHDL devra être conforme aux spécifications données ci-dessous :



L'IP devra comporter en entrée :

- un signal chipselect : sélection du périphérique sur le bus Avalon
- un signal write\_n : signal d'autorisation d'écriture dans les registres de l'IP
- un signal reset\_n : remise à zéro des registres de l'IP actif à l'état bas.
- une horloge système à 50 MHz
- un signal WriteData sur 8, 16 ou 32 bits, qui permet d'écrire dans l'IP

L'IP devra comporter en sortie :

- un signal commande : entrée du servomoteur

### 2.3.1 Développement de l'IP

Apporter à la première version de l'IP les modifications nécessaires afin qu'elle prenne en compte les signaux du bus Avalon.

### 2.3.2 Simulation et validation

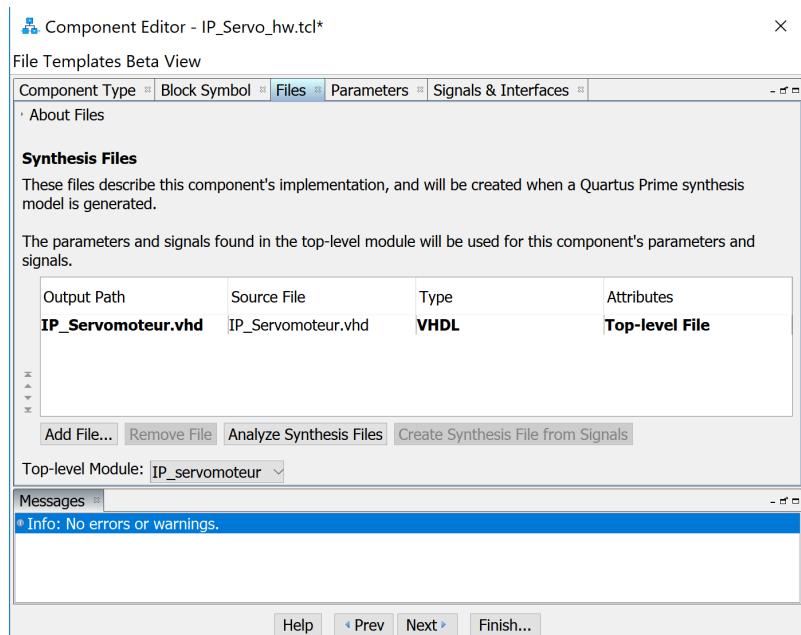
Effectuer les simulations du contrôleur avec Modelsim. Il est demandé d'écrire un banc de test et un script de simulation qui doivent permettre de vérifier rapidement la validité du fonctionnement de l'IP.

Reportez vos résultats de simulation sur votre compte-rendu de projet.

## 2.4 Intégration matérielle de l'IP Servomoteur

L'intégration d'une nouvelle IP dans un système SoC-FPGA existant se fait par l'intermédiaire d'un Custom Peripheral. Cela consiste à connecter les E/S de votre IP version Avalon avec les signaux disponibles sur le bus Avalon.

Dans Platform Designer, ajouter une IP de type Custom Peripheral (comme vu pendant les TP). Dans le menu **File → New component**, importez fichier VHDL de votre IP (onglet Files).



Connectez les signaux d'entrées / sorties de votre IP avec les signaux du bus Avalon (Onglet Signals ans Interfaces).

Il faut rajouter une interface **conduit\_end** en cliquant sur <<add interface>> pour le signal de Commande.

Rajouter l'IP Servomoteur ainsi créé dans votre projet et connecter les signaux en prenant pour exemple l'IP télémètre.

Générer le nouveau système. Modifier le code VHDL du top-level, synthétisez et implémentez le système sur le FPGA.

## 2.5. Programmation logicielle et test de l'IP Servomoteur

Maintenant que vous avez intégré la nouvelles IP Servomoteur, vous allez programmer le processeur pour qu'il commande le servomoteur en position depuis les interrupteurs (Switch). Pour cela, vous pouvez prendre comme point de départ le programme donné pour l'IP télémètre.

Consignez votre programme dans votre Compte Rendu.

**Prenez une photo du résultat à mettre dans votre compte-rendu.**

## **Etape 3 : Affichage des obstacles (2 points)**

Dans cette étape il est demandé de détecter les obstacles en faisant tourner le servomoteur sur  $180^{\circ}$  tout en faisant des mesures régulières avec le télémètre ultrason.

Affichez les résultats dans le terminal de NIOS2 SBT.

Par exemple sous la forme suivante :

1° -> 60 cm

2° -> 61 cm

3° -> 60 cm

4° -> 61 cm

...

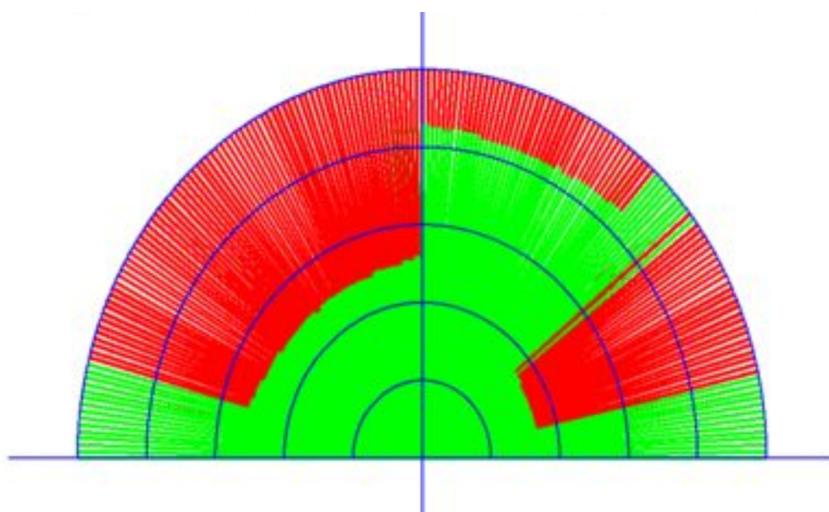
Consignez votre programme et vos résultats dans votre Compte Rendu.

**Faites vérifier cette étape par l'enseignant.**

*Pour la suite vous n'êtes pas obligé de respecter l'ordre du sujet. Vous pouvez soit commencer par la partie 4 et finir par les parties 5 et 6. Soit faire les parties 5 et 6 puis la partie 4.*

## **Etape 4 : Affichage des obstacles sur un écran VGA (3 points)**

Dans cette étape il est demandé d'afficher les obstacles sur un écran VGA en faisant tourner le servomoteur sur  $180^{\circ}$  tout en faisant des mesures régulières avec le télémètre ultrason. Il faudra faire un affichage sur écran VGA de la cartographie 2D comme sur la figure ci-dessous :



Pour cette partie, vous pouvez consulter le document DE10-Lite-Computer-NiosII.pdf (disponible sur Moodle) qui pourra vous donner des pistes pour la partie VGA.

**Pour utiliser ce projet correctement, il faut utiliser la version 18.1 de Quartus ainsi que l'exécutable fournit par Intel-Altera University program : « intel\_fpga\_upds\_setup.exe »**

**Si vous n'utilisez pas la version 18.1 et que vous n'avez pas installé l'exécutable de Intel-Altera, ceci sont disponible à partir du lien suivant :**

<https://dropsu.sorbonne-universite.fr/s/98M6RrPFJ3pQpTS>

Consignez votre programme et vos résultats dans votre Compte Rendu.

**Faites vérifier cette étape par l'enseignant.**

## **Etape 5 : Conception de l'IP UART (3 points)**

Ajouter un périphérique UART qui doit permettre d'exporter les données cartographiées vers un Terminal sur un PC (Putty ou TeraTerm sur Windows ou Minicom et Screen sur Linux/MacOS).

Ajouter des commandes UART pour configurer certains paramètres du radar (plage de balayage et vitesse de balayage).

Pour réaliser cette IP, vous devez reprendre toutes les étapes qui ont été détaillé dans les IP précédemment décrites (télémètre ultrason et servomoteur).

Ce qui se traduit par la simulation d'un composant VHDL qui permet d'envoyer ou recevoir des données vers un PC à travers l'UART et un convertisseur série/USB.

Ensuite il faut rajouter l'interface Avalon de votre IP UART et enfin l'intégrer au système autour du Nios II.

Consignez votre programme, vos simulations et vos résultats dans votre Compte Rendu.

**Faites vérifier cette étape par l'enseignant.**

## **Etape 6 : Conception de l'IP Neopixel (3 points)**

Pour cette partie, pour rajouter un peu d'animation à votre radar 2D vous devez réaliser une IP qui permet de commander un Ring de 12 LED Neopixel.

Voici quelques liens pour avoir plus d'informations sur le fonctionnement des LEDs NeoPixel :

<https://cdn-shop.adafruit.com/datasheets/WS2812.pdf>

<https://cdn-learn.adafruit.com/downloads/pdf/adafruit-neopixel-uberguide.pdf>

Pour réaliser cette IP, vous devez reprendre toutes les étapes qui ont été détaillé dans les IP précédemment décrites (télémètre ultrason et servomoteur).

Ce qui se traduit par la simulation d'un composant VHDL qui permet de commander une ou plusieurs LED Neopixel puis tester cette IP sur carte FPGA.

Ensuite il faut rajouter l'interface Avalon de votre IP NeoPixel et enfin l'intégrer au système autour du Nios II.

Consignez votre programme, vos simulations et vos résultats dans votre Compte Rendu.

**Faites vérifier cette étape par l'enseignant.**