

Compte-Rendu de mini-projet Radar 2D

Table des matières

1. Implémentation de l'IP télémètre ultrason HC SR04	3
1.1 Introduction	3
1.2 Travail à réaliser	3
1.2.1 Simulation de l'IP	3
1.2.2 Test de l'IP sur la carte	3
1.2.3 Intégration de l'IP Télémètre dans Platform Designer	3
1.2.4 Programmation logicielle et test de l'IP	4
2. Conception de l'IP Servomoteur	5
2.1 Comment fonctionne un servomoteur	5
2.2 Conception de la partie opérationnelle du composant Servomoteur	5
2.2.1 Développement de l'IP servomoteur	5
2.2.2 Simulation et validation	5
2.2.3 Vérification sur la carte DE1-SoC	5
2.3 Extension de l'IP Servomoteur vers une version connectable au bus Avalon	6
2.3.1 Développement de l'IP	6
2.4 Intégration matérielle de l'IP Servomoteur	6
2.5. Programmation logicielle et test de l'IP Servomoteur	7
3. Affichage des obstacles	8

1. Implémentation de l'IP télémètre ultrason HC SR04

1.1 Introduction

L'objectif dans cette partie est de créer l'IP permettant d'utiliser le télémètre par implémentation directe sur le FPGA. Ensuite, nous nous pencherons sur une deuxième IP à développer pour utiliser la première avec le processeur NIOS II par le bus Avalon en l'ajoutant sur Platform Designer. On utilisera enfin cette IP sur Eclipse avec le bus Avalon.

1.2 Travail à réaliser

1.2.1 Simulation de l'IP

On décrit l'IP avec les données du sujet. J'ai choisi d'utiliser une machine à états, ça me semblait la méthode la plus facile pour l'implémentation.

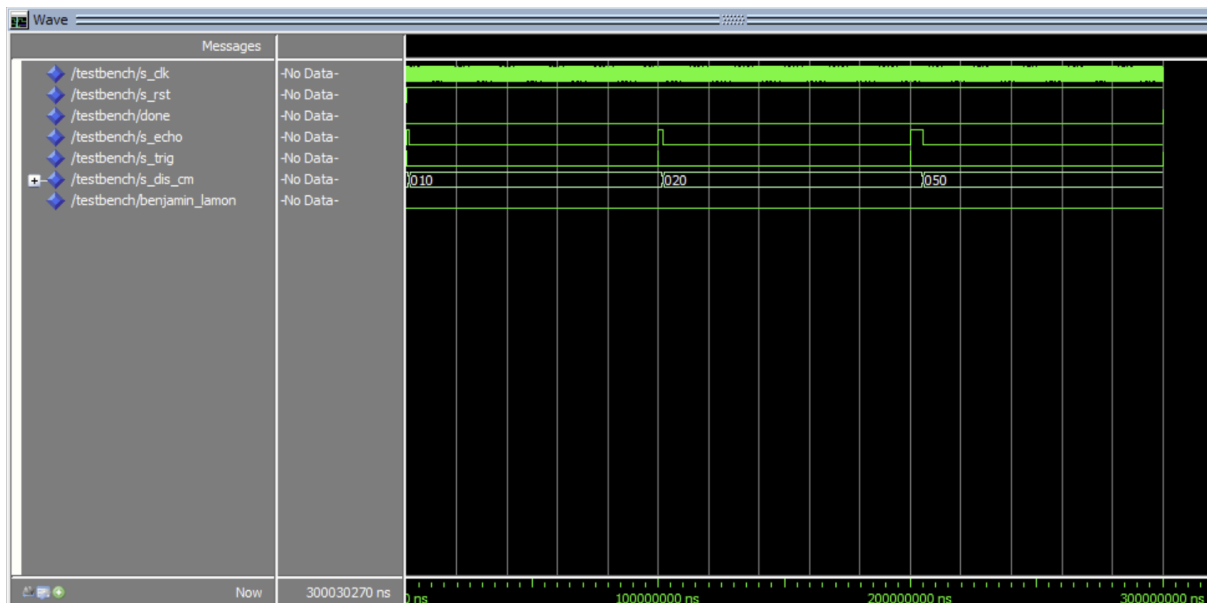


Figure 1: Simulation de l'IP télémètre.

On observe que plus l'écho est grand, plus la distance est grande. Ces valeurs sont purement des simulations de la sortie du télémètre.

1.2.2 Test de l'IP sur la carte

On implémente ensuite l'IP en modifiant les pinouts. Pour ce faire, il faut ouvrir le pin planner, supprimer les pins déjà utilisés pour les réorienter vers les signaux de l'IP. On branche ensuite le télémètre en branchant VCC sur une sortie 5V, Gnd sur un pin GND de la carte, Trig sur GPIO(1) et Echo sur GPIO(3).

https://youtube.com/watch?v=ysj_6euBZiU

1.2.3 Intégration de l'IP Télémètre dans Platform Designer

En ajoutant les signaux de cette manière, on pourra utiliser l'IP sur Eclipse avec le processeur NIOS II.

```

Name
└─ avalon_slave_0 Avalon Memory Mapped Slave
    └─ chipselect [1] chipselect
    └─ read_n [1] read_n
    └─ readdata [32] readdata
        <<add signal>>
└─ clock Clock Input
    └─ clk [1] clk
└─ conduit_end Conduit
    └─ dis_cm [10] dist_cm
    └─ echo [1] echo
    └─ trig [1] trig
        <<add signal>>
└─ reset_n Reset Input
    └─ rst_n [1] reset_n
        <<add signal>>
<<add interface>>

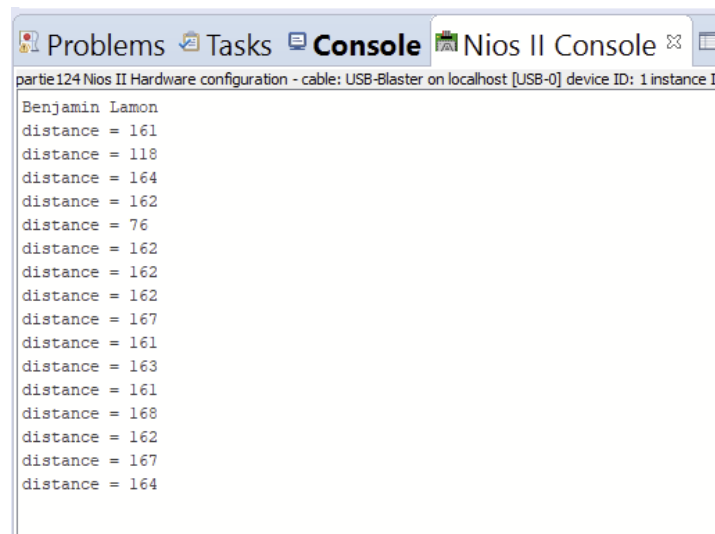
```

Figure 2: Noms et localisation des signaux de l'IP Télémètre Avalon dans Platform Designer.

1.2.4 Programmation logicielle et test de l'IP

Une fois l'implémentation faite, on prend le template d'instanciation donné par Platform Designer. On modifie ensuite le fichier DE10_Lite_Computer.vhd du projet fourni en annexe au sujet. On compile, puis on lance Eclipse pour pouvoir utiliser le softcore NIOS II.

On obtient enfin le résultat suivant lors de la lecture de la distance sur le télémètre.



```

partie124 Nios II Hardware configuration - cable: USB-Blaster on localhost [USB-0] device ID: 1 instance 1
Benjamin Lamon
distance = 161
distance = 118
distance = 164
distance = 162
distance = 76
distance = 162
distance = 162
distance = 162
distance = 167
distance = 161
distance = 163
distance = 161
distance = 168
distance = 162
distance = 167
distance = 164

```

Figure 3: Console NIOS II lisant la distance grâce au télémètre.

2. Conception de l'IP Servomoteur

2.1 Comment fonctionne un servomoteur

On utilisera le servomoteur entre 0 et 90°.

2.2 Conception de la partie opérationnelle du composant Servomoteur

2.2.1 Développement de l'IP servomoteur

On utilisera une machine à états. Deux états y figureront : un état d'émission de l'impulsion en fonction de "position", et un état d'attente jusqu'à 10ms. J'utiliserai deux compteurs, un global pour compter jusqu'à 10ms et un autre pour compter l'impulsion.

2.2.2 Simulation et validation

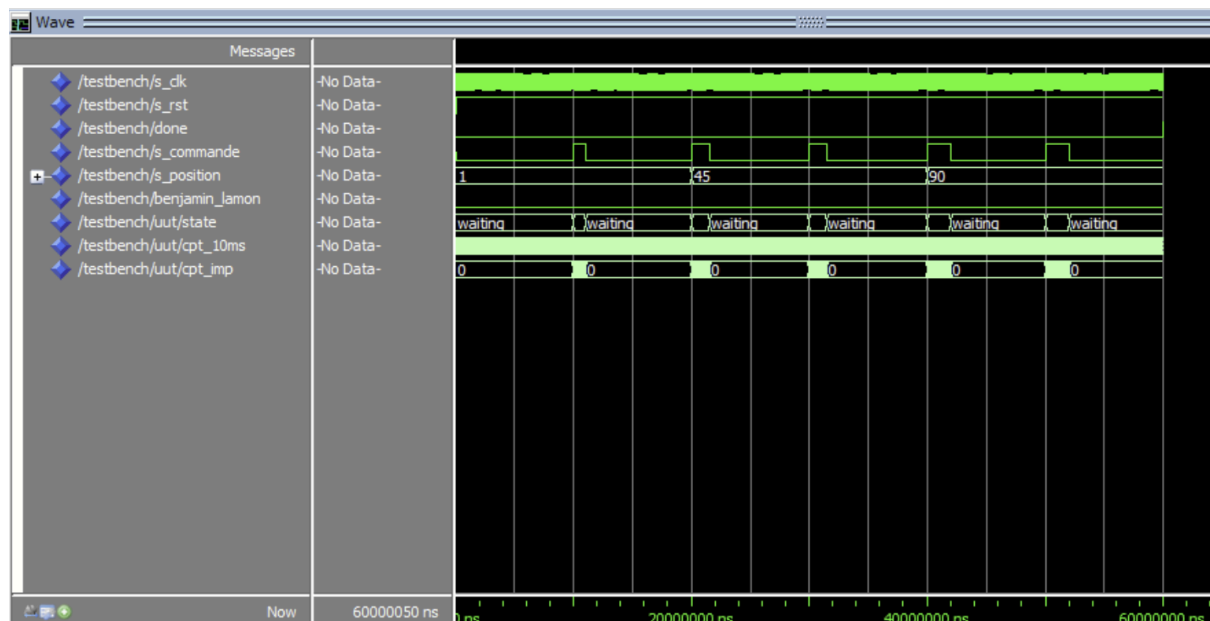


Figure 4: Simulation de l'IP servomoteur.

Sur cette simulation, nous observons la différence d'impulsion en fonction des différentes valeurs. À savoir qu'une impulsion de 1.5ms est bien représentée lorsque position vaut 45 et 2ms lorsque position vaut 90, ce qui est conforme aux explications du sujet.

2.2.3 Vérification sur la carte DE1-SoC

Voir vidéo : <https://youtube.com/watch?v=AXbyqnmqXvg>

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

Avalon

2.3.1 Développement de l'IP

On développe maintenant une IP qui utilisera celle développée en 2.2 en respectant les consignes du sujet. J'ai choisi un signal writeData sur 32 bits.

2.3.2 Simulation et validation

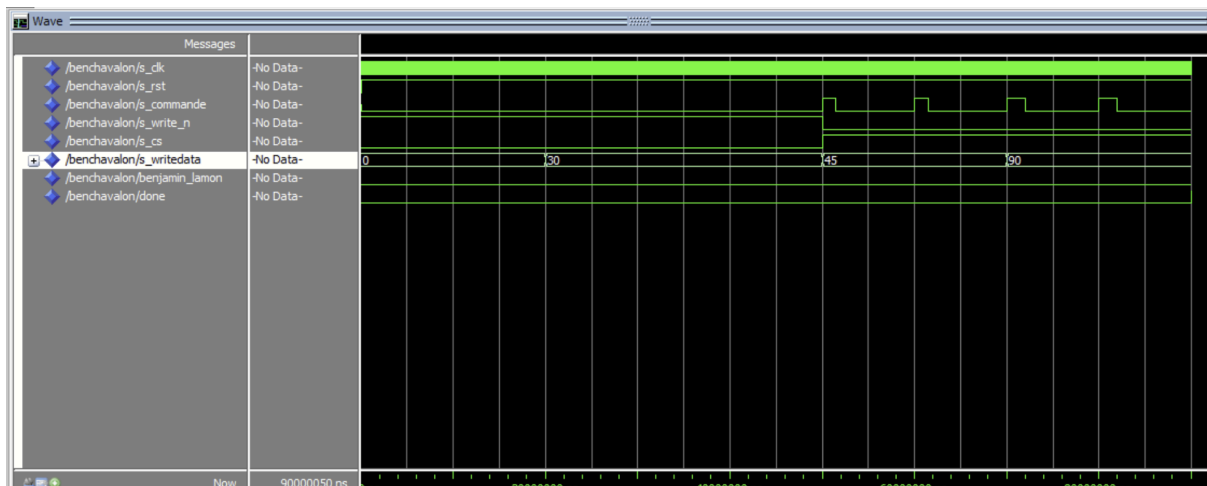


Figure 5: Simulation de l'IP servomoteur pour utilisation sur le bus Avalon.

On observe que malgré le changement de commande (ici sur WriteData), la commande n'est changée que lorsque "Chip Select" est à 1 et "Write_n" est à 0 ce qui est conforme.

2.4 Intégration matérielle de l'IP Servomoteur

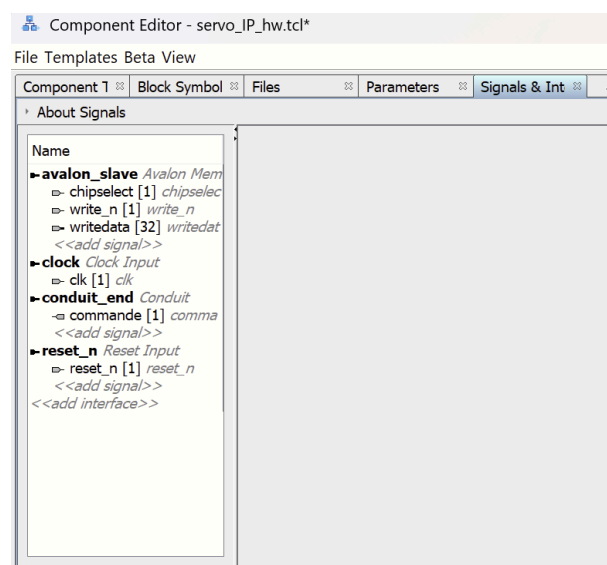


Figure 6: Ajout de l'IP servomoteur dans Platform Designer.

On modifie alors le Top-Level du projet fourni avec l'exemple d'instanciation généré par Platform Designer.

2.5. Programmation logicielle et test de l'IP Servomoteur

```
dernier NIOS II Hardware Configuration - Cable: USB-Diagster  
Benjamin Lamon  
distance = 329 ; angle 16  
distance = 328 ; angle 16  
distance = 322 ; angle 16  
distance = 322 ; angle 16  
distance = 330 ; angle 16  
distance = 327 ; angle 16  
distance = 325 ; angle 16  
distance = 328 ; angle 16  
distance = 328 ; angle 16  
distance = 324 ; angle 16  
distance = 325 ; angle 16  
distance = 324 ; angle 16  
distance = 324 ; angle 16  
distance = 324 ; angle 16  
distance = 324 ; angle 16  
distance = 337 ; angle 16  
distance = 325 ; angle 16  
distance = 325 ; angle 16  
distance = 328 ; angle 16
```

Figure 7: Terminal de l'IP servomoteur avec en entrée les switchs de la carte. Ici, seul le switch 5 est actif.

3. Affichage des obstacles

L'objectif ici est de faire tourner le servomoteur dans un sens, puis dans l'autre tout en mesurant la distance. Je l'ai fait sur 90°.

```
Benjamin Lamon
distance = 323 ; angle 5
distance = 323 ; angle 10
distance = 323 ; angle 15
distance = 323 ; angle 20
distance = 323 ; angle 25
distance = 324 ; angle 30
distance = 323 ; angle 35
distance = 324 ; angle 40
distance = 324 ; angle 45
distance = 323 ; angle 50
distance = 326 ; angle 55
distance = 324 ; angle 60
distance = 324 ; angle 65
distance = 324 ; angle 70
distance = 323 ; angle 75
distance = 323 ; angle 80
distance = 324 ; angle 85
distance = 324 ; angle 90
distance = 328 ; angle 85
distance = 324 ; angle 80
distance = 327 ; angle 75
distance = 325 ; angle 70
distance = 325 ; angle 65
distance = 324 ; angle 60
distance = 324 ; angle 55
distance = 327 ; angle 50
distance = 327 ; angle 45
distance = 326 ; angle 40
distance = 325 ; angle 35
distance = 325 ; angle 30
distance = 325 ; angle 25
distance = 325 ; angle 20
distance = 324 ; angle 15
distance = 326 ; angle 10
distance = 323 ; angle 5
distance = 323 ; angle 0
```

Figure 8: Affichage des obstacles dans le terminal NIOS II.

On note que lorsque l'angle atteint 90 (sous-entendu 90°), on descend jusqu'à 0°, et on augmente jusqu'à 90° etc.