Rapport – Mini Projet Radar

Architecture de Processeur

CHEN Pascal
KO Roqyun

Table des matières

1 Introduction

Dans le cadre du cours architecture de processeur, nous réalisons un mini radar avec un télémètre HC-SR04 et un servomoteur. Dans ce projet, nous apprenons la communication FPGA/HPS en utilisant la carte DE0-Nano-SoC.

2 Partie 1 – Télémètre:HC-SR04

2.1 Simulation

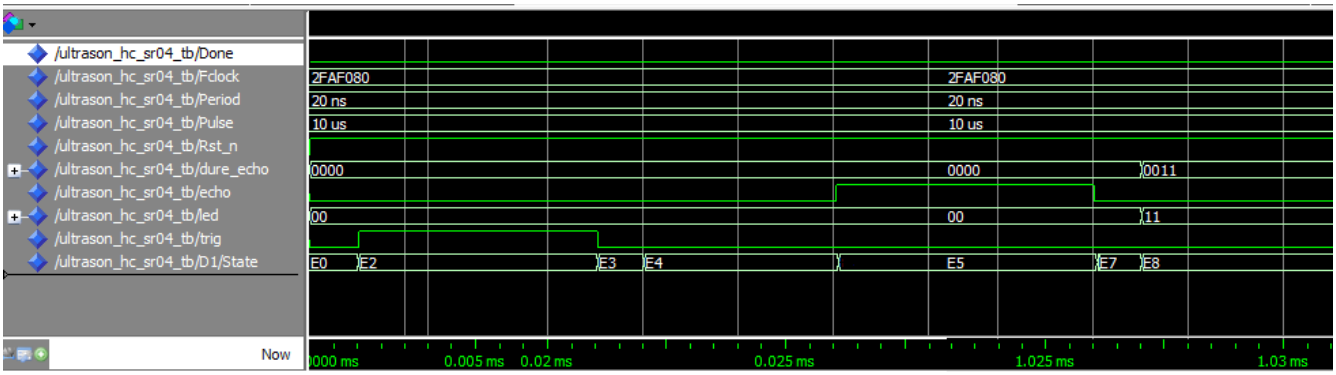


Figure 1: Le capture d'écran du résultat de la simulation (Image a été modifiée pour la lisibilité)

2.1.1 Fonctionnement

Le télémètre est codé en machine en état. Il y a 8 états au total.
Pour comprendre la machine à l'état, il faut comprendre le fonctionnement du télémètre :

1. Pour activer le télémètre, on doit envoyer un signal (trigger) de 1 pendant au moins 10 µs.
2. Le télémètre mesure la distance entre l'obstacle qui est devant et le signal ccho devient 1.
3. Le signal echo est remis à 0 lorsque la mesure de la distance est finie. Donc, la distance et la longueur temporelle du signal echo est proptionnelle.
4. Le télémètre envoie le signal echo vers la carte FPGA.

Dans ce projet, on répète ce procédure automatiquement.
Si le signal echo dure environ 40 ms, on considère qu'il n'y a pas d'obstacles et on redémarre le processus.

2.1.2 Machine en état

Afin de réaliser ce processus, une machine en état a été implémenté.

- E0
  - Initialisation : Remettre le compteur à 0 et mettre le signal trigger à 1.
- E2 :

    - Incrémenter le compteur à chaque 1 µs et attendre jusqu'à ce qu'il atteint 20 µs.
    - Si 20 µs est passé, mettre le signal trigger à 0 pour activer la mesure.
- E3 :
    - Attente jusqu'au prochain coup d'horloge.
- E4 :
    - Attente jusqu'à l'activation du télémètre (il est activé quand le signal echo est à 1)
    - Initialiser le compteur
- E5 :
    - Incrémenter le compteur à chaque 1 µs et attendre jusqu'à ce que le signal écho est remis à 0.
- E6 :
    - Si la longueur du signal echo est inférieur à 30 ms, passer à l'état E7
    - Si la longueur du signal echo est supérieur à 30 ms, passer à l'état E8
- E7 :
    - Au prochain coup d'horloge, calculer la distance entre le télémètre et l'obstacle.
    - Passe à l'état E8
- E8 :
    - Incrémenter le compteur à chaque 1 µs et attendre 30ms avant de redémarrer. Passer à l'état E0

2.1.3 Résultat

Il y a 2 cas
Un obstacle est présent :
E0 → E2 → E3 → E4 → E5 → E6 → E7 → E8
Un obstacle n'est présent :
E0 → E2 → E3 → E4 → E5 → E6 → E8

2.2 Calcul de la distance

Pour calculer la distance, on prend en compte de la vitesse du son, la durée d'impulsion est en dizaine et enfin la longueur du signal echo :

La vitesse du son, v = 340 m/s = 34 000 cm / 1000 000µs
La durée d'impulsion, t_{imp} = 10µs
la longueur du signal echo, L_{echo}

Le son fait un aller-et-retour, donc il faut diviser la longueur du signal echo par 2.

Enfin, la distance est donnée par la formule suivant :
$d = [v * (t_{imp} * L_{echo})] / 2 = 17 / 100 * L_{echo} = L_{echo} / 58$

3 Partie 2 – Servomoteur

Le servomoteur est commandé par un PWM (Pulse Width Modulation). Sa position dépend du rapport cyclique. La fréquence du PWM est 50Hz et la période est 20ms. Le rapport cyclique minimal est 5 % et le maximum est 12,5% (0,5ms ~ 2,5ms). Donc la formule pour trouver le rapport cyclique, **α**, afin de changer la position est donnée par :

$T_{duty\_cycle}(\theta) = (\theta / 180°) * (2{,}5ms – 0{,}5ms)$

$\alpha = T_{duty\_cycle} / 20\ ms$

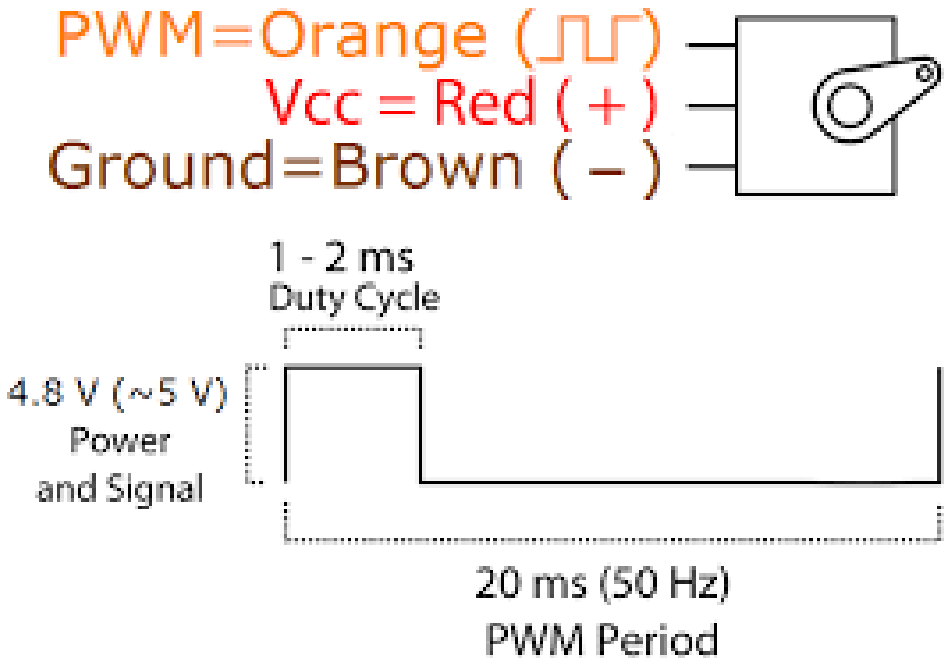


Figure 2: https://electronics.stackexchange.com/questions/346603/driving-servo-motor-with-pwm-signal?rq=1

Dans notre IP servomoteur, la position est commandé par deux switch, SW[2] et SW[3]

SW[2]	SW[3]	Position
0	0	0°
0	1	45°
1	0	135°
1	1	180°

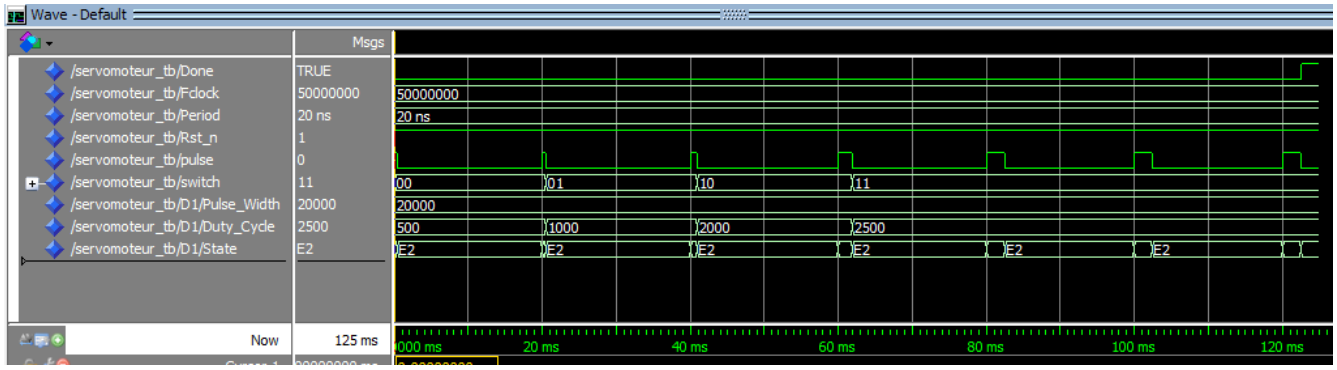
4 Machine en état

L'IP servomoteur est réalisé par une machine en état dont le nombre d'états est 3.

- E0 :
  - Initialisation. Remettre le compteur à 0 et envoie un signal pulse de 1 au servomoteur.
- E1 :
  - Attendre pendant T_{duty_cycle}. Puis, remettre le signal pulse à 0 et passer à E2
- E2 :
  - Attendre pendant (20ms - T_{duty_cycle}). Repasser à E0.

4.1 Simulation

https://youtu.be/nlnNZwyBEh4



La largeur du pulse devient de plus en plus grande en fonction des switchs. On remarque que le rapport cyclique (duty cycle) change au bout de chaque 20ms sauf le dernier 60 ms où les valeurs de switchs ne changent pas. On vérifie que l'IP fonctionne correctement.

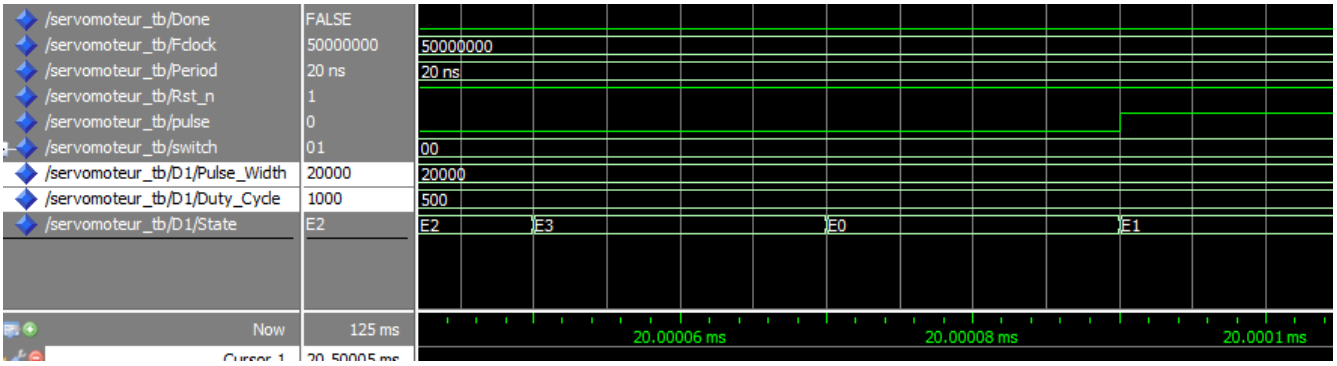


Figure 3 Les transitions de E0 à E1 et de E2 à E3 sont très rapides

4.2 IP Avalon Servomoteur

L'IP Avalon servomoteur est codé pour que l'écriture est autorisé quand le signal address est à 1, le signal write_n à 1 et le signal chip_select est égale à 1. Après, on écrit la position dans le signal WriteData depuis le terminal de l'ordinateur. Le code correspondant en C est ci-dessous :

```
h2p_lw_servomoteur_addr=(unsigned int *)(h2p_lw_virtual_base + SERVOMOTEUR_BASE);
h2p_lw_switch_addr =(unsigned int *)(h2p_lw_virtual_base + DIPSW_PIO_BASE);

for (i=0 ; i<1000000 ; i++) {
        sleep(1);
        sw = ioread32(h2p_lw_switch_addr);
        do {
                printf("Position du servomoteur (0~180) ? ");
                scanf("%u%c", &angle,&ch);
        }while(!(0<= angle && angle <= 180));
        iowrite32(angle & 0xFF, h2p_lw_servomoteur_addr);
        printf("Switch = %x \t Angle = %d \n", sw&0xF, angle & 0xFF);
}
```