Ludovic MIRANDA
Muhammad Haziq KAMARUL-AZMAN 23/05/12

TP 3 : Modulation de largeur d'impulsion

Réalisation d'un signal par le PWM

- D'après le manuel PAGE 52 – TABLE 24, l'horloge périphérique *pclk* est la même que celle du processeur *cclk* si les bits 1 et 0 du registre *VPBDIV* sont à '01'. *« 0 1: VPB bus clock is the same as the processor clock. »*

- D'après le manuel PAGE 78 – TABLE 48, pour avoir configurer la sortie *PWM6*, il faut que les bits 19 et 18 soient à '10'. On pourra également voir la sortie du signal sur le port 0.8.

- D'après le manuel PAGE 154 – TABLE 119, pour activer le *PWM4*, le bit 12 doit être à 1 : *« When one, enables the PWM4 output. »*, et pour le configurer en mode *« simple edge »*, le bit 4 doit être à 0 : *When zero, selects single edge controlled mode for PWM4. ».*

- D'après le manuel PAGE 153 – TABLE 118, pour remettre à zéro le *PWMMTC* quand la valeur *PWMMR0* est atteinte, il faut que le bit 1 soit à '1' : *« When one, the PWMTC will be reset if PWMMR0 matches it ».*

- D'après le manuel PAGE 146 – TABLE 119, on doit utiliser *MR0* pour fixer la période, car on a configuré le *PWM4* en mode *« simple edge »*.
  Comme la fréquence de l'horloge est de 10MHz, il faut mettre *MR0* à 10000 (il va compter de 0 à 10000) pour obtenir un signal de fréquence 1KHz.

- D'après le manuel PAGE 146 – TABLE 119, on doit utiliser *MR4* pour fixer le rapport cyclique, car on a configuré le *PWM4* en mode *« simple edge »*.
  Pour avoir un rapport cyclique de 50 %, on met *MR4* à 5000.

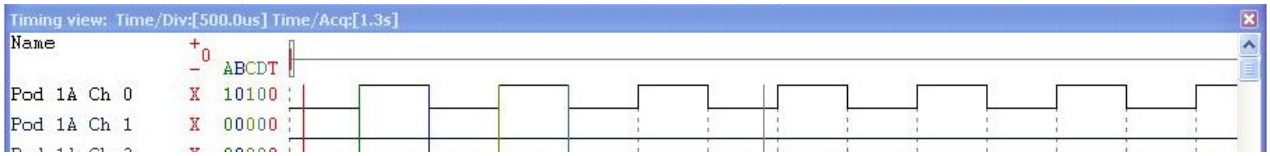
- D'après le manuel PAGE 152 – TABLE 117, il faut que le bit 1 du *PWMTCR* soit à 1 pour faire une remise à zéro du compteur et du *prescaler* : *« When one, the PWM Timer Counter and the PWM Prescale Counter are synchronously reset [...] ».*

- D'après le manuel PAGE 152 – TABLE 117, pour activer le compteur, et donc démarrer l'exécution du *PWM*, il faut que le bit 3 soit à 1 : *« When one, PWM mode is enabled. ».*

Code de la fonction pwm_init() :

```
void pwm_init(){
	SCB_VPBDIV = 0x1; // on met l'horloge périphérique à la même fréquence que le processeur

	PCB_PINSEL0 = 0x00020000; // on met les bits 17 et 16 à '10'

	PWM_PCR = 0x00001000; // sélection du mode "simple edge"

	PWM_MCR = 0x2; // mise à zéro du PWM_MTC

	PWM_MR0 = 10000; // fixer la fréquence à 1KHz

	PWM_LER = 0x1	// 'reseter'

	PWM_MR4 = 5000; // fixer le rapport cyclique à 50%

	PWM_LER = 0x10	// 'reseter'

	PWM_MTCR = 0x2; // ràz de PWMTCR et "prescaler"

	PWM_MTCR = 0x9; // activation du compteur
}
```

Afin de visualiser les signaux générés, on branche la carte à l'analyseur logique. On obtient alors la figure suivante, sur laquelle on voit bien un signal de fréquence 1KHz, et de rapport cyclique 50% :



Graphique 1: Signal vu sur la led 1

Application à la commande d'un moteur à pas

Dans cette partie, nous allons créer une fonction qui devra générer les quatre signaux de commande d'un moteur pas à pas. Pour générer ces signaux logiques, nous allons utiliser les quatre sorties du *PWM* configurées comme suit :

- ✗ *PWM4* en *simple edge* pour *Q1*
- ✗ *PWM2* en *double edge* pour *Q2*
- ✗ *PWM6* en *double edge* pour *Q3*
- ✗ *PWM3* en *double edge* pour *Q4*

Pour cela, nous allons procéder de la même manière que dans la partie précédente :

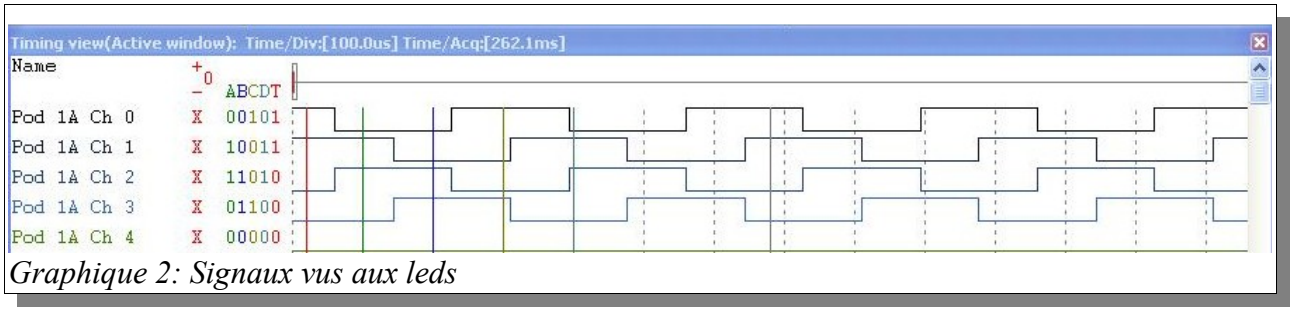
- Il faut d'abord changer la direction du *PINSEL0*, pour que celui-ci soit connecté sur les ports *P0.8 P0.7 P0.9 P0.1*.
  D'après le manuel PAGE 78 – TABLE 48, il faut changer les bits 3,4,15,16,17,18,19,20.

- Il faut ensuite changer la valeur du *PWMMCR* pour configurer *PWM2, PWM6* et *PWM3* en mode *double edge*.
  D'après le manuel PAGE 154 – TABLE 119, il faut changer les bits 3,4,5,7 pour selectionner le mode *double edge*, et les bits 11,12,13,15, pour activer les ports.

- Pour fixer les différentes fréquences, et le rapport cyclique de chaque signal, on utilise le manuel PAGE 146 – TABLE 119, pour avoir les correspondances.

A l'aide de l'analyseur logique, on a pu visualiser tous les signaux crées :

On voit bien qu'on obtient bien tous les signaux souhaités, avec les bonnes fréquences et les bons rapports cycliques :



Graphique 2: Signaux vus aux leds

Code de la fonction :

```
void pwm_step_motor_control(){

  SCB_VPBDIV = 0x1; // on met l'horloge périphérique à la même fréquence que le processeur

  PCB_PINSEL0 = 0xA8008; // Pour la deuxieme partie, PWM2, 3, 4, 6 sont utilisés

  PWM_PCR = 0x5C4C; // séléction du mode "simple edge", pour la deuxieme partie, PWM4 reste en simple edge et
les autres sont en double edge

  PWM_MCR = 0x2; // mise à zéro du PWM_MTC

  PWM_MR0 = 10000; // fixer la fréquence à 1KHz

  PWM_LER = 0x01; // 'reseter'

  PWM_MR4 = 5000; // fixer le rapport cyclique à 50% pour PWM4, commence à 0

  PWM_LER = 0x10; // 'reseter'

  PWM_MR1 = 2500; // fixer le rapport cyclique à 50% pour PWM2, commence à 2500

  PWM_LER = 0x02; // 'reseter' PWM2

  PWM_MR5 = 5000; // fixer le rapport cyclique à 50% pour PWM6, commence à 5000

  PWM_LER = 0x04; // 'reseter' PWM6

  PWM_MR3 = 2500; // fixer le rapport cyclique à 50% pour PWM3, commence à 7500

  PWM_LER = 0x08; // 'reseter' PWM3

  PWM_MR5 = 5000;

  PWM_LER = 0x20;

  PWM_MR6 = 10000;

  PWM_LER = 0x40;

  PWM_TCR = 0x02; // ràz de PWMTCR et "prescaler"

  PWM_TCR = 0x09; // activation du compteur

}
```