3 – Modulation de largeur d'impulsion

Objectifs : *gestion de périphériques (PWM), génération de signaux modulés en largeur d'impulsion, utilisation d'un analyseur logique.*

PREPARATION : LE CODE DE LA FONCTION `pwm_init` DE LA PARTIE **1.2** DEVRA ETRE PREPARE ET SERA VERIFIE AU DEBUT DE LA SEANCE DE TP.

REALISATION : LES PARTIES **1.2** et **1.3** DOIVENT ETRE VALIDEES PAR L'ENCADRANT AU COURS DE LA SEANCE.

1.1 Description

La modulation en largeur d'impulsion (MLI) est une solution très pratique pour permettre le contrôle d'un système électrique à partir d'une commande numérique générée par un processeur. Le principe est de générer un signal logique (valant 0 ou 1), à fréquence fixe mais dont le rapport cyclique est contrôlé numériquement (Figure 1). Une application typique est le contrôle de circuits électriques par une commande numérique.

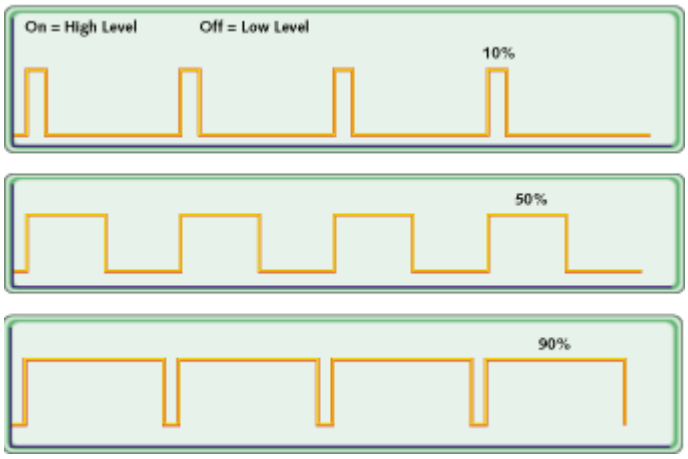


Figure 1. signaux PWM et différent rapports cycliques

Dans ce TP nous illustrerons l'utilisation du périphérique de génération de signaux MLI de la carte LPC2106 (PWM) pour générer les signaux de commande d'un moteur pas à pas.

1.2 Réalisation d'un signal par le PWM

Dans un premier temps on va chercher à réaliser un signal carré simple (rapport cyclique 50%) de fréquence 1KHz.

Configurer un projet IAR comme suit :

- Créer un fichier `main.c`, dans lequel vous écrirez une fonction `main()` appelant une fonctions : `pwm_init()`, celle-ci devra initialiser le PWM et le configurer pour qu'il génère un signal carré sur la sortie `PWM4`.

Reportez-vous au manuel utilisateur pour la description exhaustive du fonctionnement du PWM. Les étapes à suivre pour configurer correctement le PWM sont les suivantes :

- Mettre l'horloge périphérique `pclk` à la valeur de l'horloge processeur `cclk`, qui est à 10 MHz (`VPBDIV`).
- Configurer les broches de la carte pour connecter la sortie `PWM4` (`PINSEL0`). Ceci permettra de récupérer le signal sur une des broches disponibles (déterminer laquelle) pour visualisation.
- Activer `PWM4` qui sera configuré en mode « mode simple edge » (`PWM_PCR`).
- Générer une remise à zéro du compteur `PWMTC` lorsque la valeur `PWMMR0` est atteinte (`PWM_MCR`).
- Fixer la période du `PWM` (`PWMMR0`).
- Fixer le rapport cyclique du `PWM` (`PWMMR4`).
- Remise à zéro du compteur et du « prescaler » (`PWMTCR`).
- Activation du `PWM` et du compteur (`PWMTCR`). Ceci correspond à démarrer l'exécution du PWM.

Vous pourrez ainsi vérifier le fonctionnement correct du programme en visualisant le signal en sortie à l'aide de l'analyseur logique (vérifier en particulier que la fréquence du signal est bien à 1KHz).

Ajouter le chronogramme obtenu et le source de votre programme dans vos compte-rendu avec des explications sur votre façon de procéder.

Annexe : utilisation de l'analyseur logique

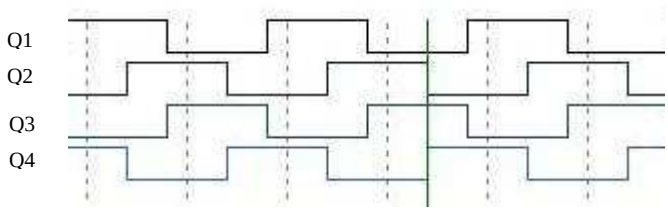
L'analyseur logique comme son nom l'indique, est un outil très pratique pour observer des signaux logiques. Pour l'utiliser, il suffit de connecter les signaux à visualiser à l'entrée d'un Logic Pod et de lancer une capture par les commandes Go/Stop dans le menu Trigger. Ouvrir l'analyseur logique avec le fichier de configuration `TP4_PWM.INI`, qui contient tous les préréglages adaptés pour ce TP.

1.3 Application à la commande d'un moteur pas à pas

Nous allons maintenant ajouter et écrire le code d'une fonction `pwm_step_motor_control()` qui devra générer les quatre signaux de commande d'un moteur pas à pas. Le moteur pas à pas est étudié en travaux pratiques d'électronique numérique (pour le principe de fonctionnement, se rapporter au tp1). Pour produire une rotation du moteur dans le sens horaire, il faut produire la séquence d'excitation suivante :

Q_4	Q_3	Q_2	Q_1
1	0	0	1
0	0	1	1
0	1	1	0
1	1	0	0

Celle-ci correspond aux chronogrammes suivants :



L'objectif est donc de générer ces quatre signaux logiques, pour cela nous pouvons utiliser quatre sorties du PWM configurées de la manière suivante :

- `PWM4` configuré en simple edge pour `Q1`.
- `PWM2` configuré en double edge pour `Q2`.
- `PWM6` configuré en double edge pour `Q3`.
- `PWM3` configuré en double edge pour `Q4`.

Compléter la fonction `pwm_step_motor_control()` pour configurer correctement le PWM.

Remarque : à chaque écriture dans un PWM Match register (`PWM_MR0` - `PWM_MR6`), il faut rendre effective la modification en écrivant dans le registre `PWMLER`.

Vérifier un à un que chaque canal génère un signal correct à l'aide de l'analyseur logique.

Testez ensuite le fonctionnement du programme dans son ensemble en vérifiant les quatre chronogrammes à l'analyseur logique. Complétez vos compte-rendu avec des captures d'écran de l'analyseur logique.