TP 7 Timers et PWM

Le but de ce TP est d'implémenter l'animation "*glow*" sur les LEDs de notre carte : leur luminosité augmente progressivement jusqu'à atteindre leur maximum, puis diminue progressivement jusqu'à être totalement éteintes, puis ce cycle recommencera. Pour faire varier la luminosité des LEDs, on utilisera le technique de *modulation de largeur d'impulsion* (PWM).

1 PWM en polling

Dans un premier temps, faisons varier l'intensité des LEDs "à la main", en les faisant clignoter dans la boucle principale du `main`.

1. Déclarer une variable globale `carrier` de type `uint16_t` [1]. Dans la boucle principale :
   — incrémenter sa valeur de 1, et
   — allumer les LEDs si `carrier < 10000`.
   — ajouter un petit `delay()` très court (compter jusqu'à 20 par exemple)

   Les LEDs devraient clignoter.
2. Changer le seuil (la valeur 10000 dans la question précédente) en 1000, puis en 100, puis en 50000. Qu'observez-vous ?
3. Changer l'incrément (1 précédemment) en 10, 100 et 1000. Qu'observez-vous ?
4. Fixer l'incrément à 1000, et faire du seuil une variable global `uint16_t modulator = 0`. Incrémenter cette variable dans la boucle. Qu'observez-vous ?
5. Maintenant, recompilez le programme en augmentant le niveau d'optimisation de `gcc` à 2 dans `makefile.inc` (option `-O`). Quel effet cela a-t-il sur l'animation ?

2 PWM avec timers

L'implémentation de 1. occupe le CPU à 100%. Ici nous allons réduire son taux d'usage en laissant le maximum de travail aux *timers*.

1. Convertissons d'abord la variable `carrier` en un timer, et laissons donc le hardware s'occuper de l'incrémenter régulièrement. Initialiser et démarrer pour cela TIM2, avec une période de 65535 et pas de prescaler ; et remplacer `carrier` par la valeur TIM2. Quelle est la fréquence d'*overflow* de ce timer ? [2] Observer le résultat.
2. Jusqu'à présent, les LEDs sont toujours rafraîchies dans le `main`, c'est à dire à la vitesse maximum possible du CPU. Initialiser et démarrer TIM3 avec activation de l'interruption ; il nous servira de timer de rafraîchissement des LEDs ; période de 255, pas de prescaler (quelle est sa fréquence d'*overflow* ?). Mettre à jour l'état des LEDs dans son *callback* et observer le résultat.
3. Finalement, le modulateur est encore incrémenté dans le `main`, et on va le remplacer par un troisième timer. Initialiser et démarrer TIM4 à une fréquence de 1Hz et en mode de comptage `TIM_COUNTERMODE_CENTERALIGNED1` (le timer compte de 0 jusqu'à sa période, puis redescend jusqu'à 0 etc.). Remplacer `modulator` par la valeur de TIM4, et observer le résultat.

1. Les valeurs de ce type peuvent sont comprises entre 0 et `INT16_MAX`, qui vaut 65535.
2. Par défaut, les timers de notre carte sont alimentées par l'horloge du bus APB à 4MHz.