TP.04 : Mise en oeuvre d'un timer hardware en C

Objectifs

A la fin du laboratoire, les étudiant-e-s seront capables de

- Concevoir et réaliser un programme modulaire (plusieurs fichiers) en C
- Débugger un programme en C
- Concevoir et réaliser un pilote de périphérique en C
- Concevoir le pilote pour le contrôleur DMTimer du µP AM3358
- Intégrer des composants développés lors de travaux précédents
- Etudier le datasheet d'un composant d'un microprocesseur

Durée du travail pratique

- 1 séance de laboratoire (4 heures) + travail personnel

Rapport à rendre

- un journal de laboratoire avec le code source sur le dépôt centralisé

Travail à réaliser

Ce TP a pour objectif la conception et réalisation d'un jeu permettant de mesurer le temps de réaction d'une personne lors de l'apparition d'un signal lumineux.

Pour la réalisation de cette application, il faudra mettre en oeuvre un bouton poussoir, l'écran LCD OLED-C ainsi qu'un timer du µP TI AM3358.

Spécifications de l'application

- Amorce
  - Une pression sur un bouton poussoir permettra d'armer le jeu
  - Un message s'affichera sur l'écran LCD indiquant que le jeu démarre
- Réflexe
  - Après un temps aléatoire entre 500 et 2500 ms, un nouveau message s'affichera sur l'écran LCD
  - Un premier compteur prendra la valeur courante d'un timer du µP
  - Dès la relâche du bouton poussoir, un 2e compteur prendra la nouvelle valeur du timer
  - La différence entre ces 2 compteurs donnera le temps de réaction
- Résultat
  - Le temps de réaction devra être affiché en ms sur l'écran LCD
  - Une nouvelle pression sur le bouton poussoir réamorcera le jeu

Horloge

- Le timer DMTimer2 du µP AM3358 servira d'horloge pour l'application
- Le pilote de périphérique permettra de contrôler les 6 timers du µP, DMTimer 2 à 7, et offrira les services suivants
  - Méthode pour initialiser un timer
  - Méthode pour lire la valeur actuelle d'un timer
  - Méthode pour lire la fréquence à laquelle un timer est cadencé
- Le bon fonctionnement du pilote devra être validé

Aspects pratiques

Voici quelques points qui devraient faciliter la réalisation de ce travail pratique.

Les timers du µP AM3358

Le µP dispose de 8 timers distincts (DMTimer 0 à 7). Les timers 2 à 7 sont identiques. La figure "integration" ci-dessous montre leur intégration sur le µP TI AM3358.

Comme indiqué sur la figure "integration", les timers reçoivent leur horloge du module *PRCM*. Ce module propose 3 horloges différentes, soit une horloge *CLK_M_OSC* à 24MHz, une horloge *CLK_32KHz* et une horloge externe. Le module `<am335x_clock.h>` fournit les services nécessaires pour enclencher les horloges des différents contrôleurs du µP. La méthode `am335x_clock_enable_timer_module(enum am335x_clock_timer_modules module)` permettra de sélectionner l'horloge *CLK_M_OSC* et ainsi de cadencer le timer spécifié lors de l'appel de la méthode.

Figure 20-3. Timer2-7 Integration

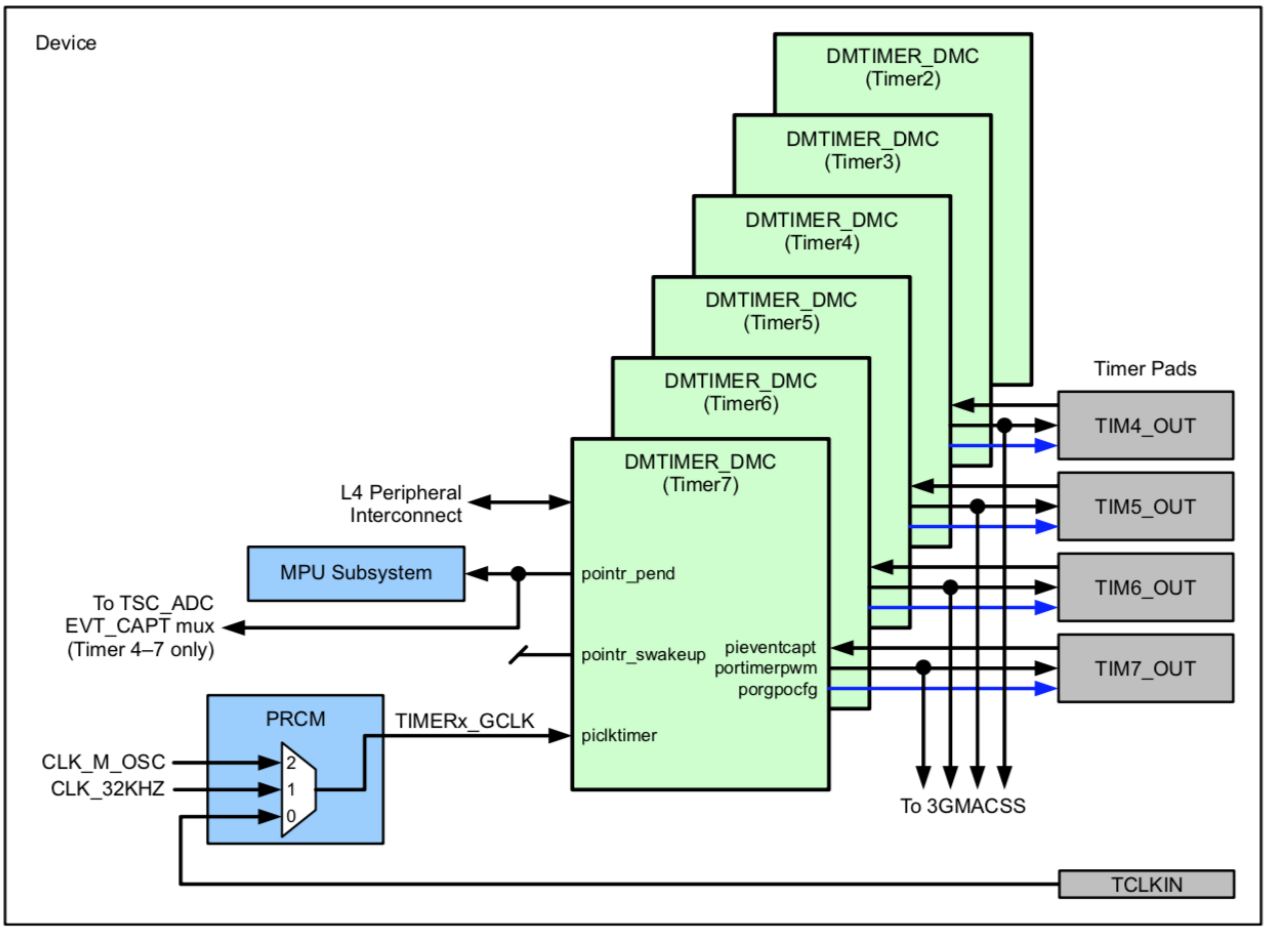

Figure 1: integration

(ref: se12/docs/01_datasheets/01_am335x/06_am335x_technical_reference_manual.pdf, page 4328)

Le schéma fonctionnel des DMTimer 2 à 7

La figure "dmtimer" ci-dessous montre le schéma fonctionnel des DMTimer 2 à 7 du µP AM3358. Seule la fonction *counter* avec auto-reload sera nécessaire à la réalisation de ce projet, soit les blocs *Prescaler* et *Timer Counter*, ainsi que les registres *tclr*, *ttgr*, *tldr*, *tcrr* et *tiocp_cfg*. La logique d'interruption ne sera, quant à elle, pas utile au projet.

(ref: se12/docs/01_datasheets/01_am335x/06_am335x_technical_reference_manual.pdf, page 4326)

Figure 20-1. Timer Block Diagram

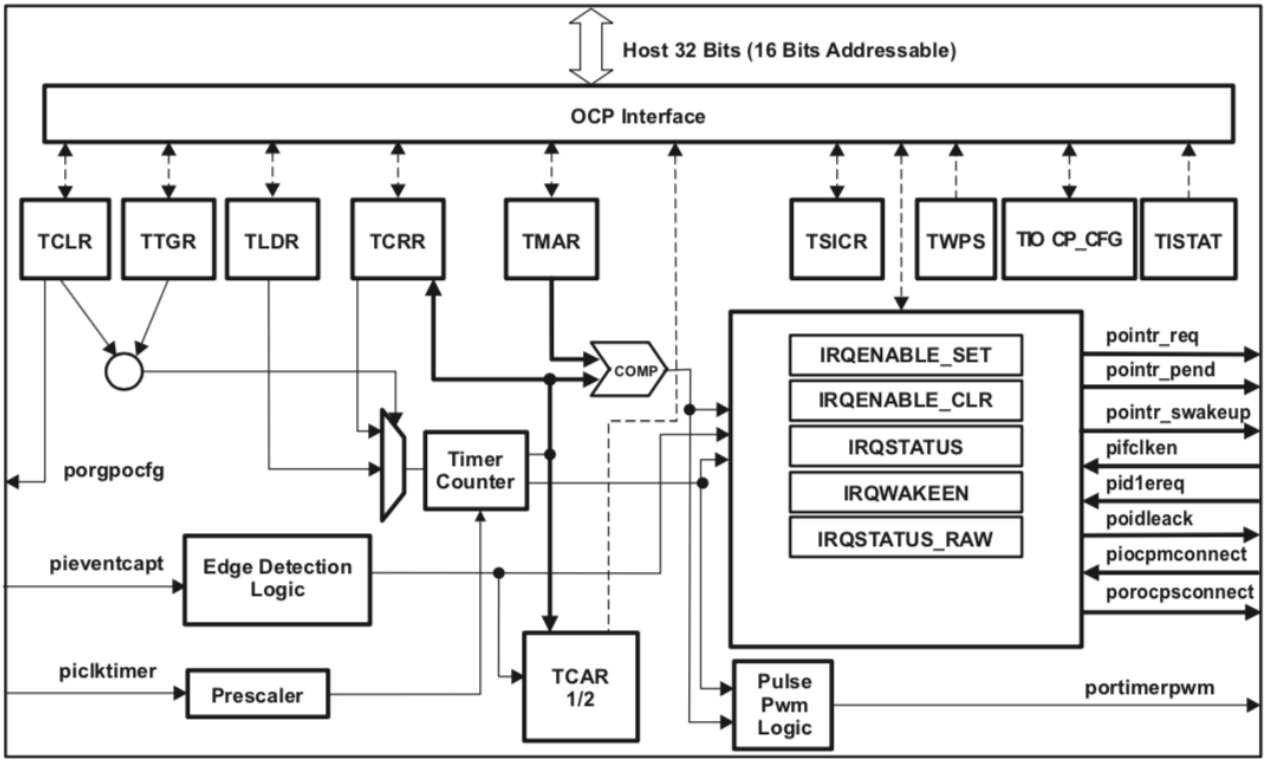

Figure 2: dmtimer

Le mode compteur du DMTimer

La figure “mode compteur” ci-dessous montre le mode *compteur* des DMTimer. Dans ce mode, le timer incrémente le contenu du registre *tccr* à la fréquence de l’horloge (dans note cas à 24MHz), jusqu’à la valeur d’overflow 0xffff’ffff avant de s’arrêter.

Si l’on configure le registre *tclr* avec le bit *AR* à 1, le contrôleur rechargera le registre *tccr* avec la valeur du registre *tldr*. Ce mode de fonctionnement permet d’obtenir un timer qui compte indéfiniment sur une période d’environ 3 minutes.

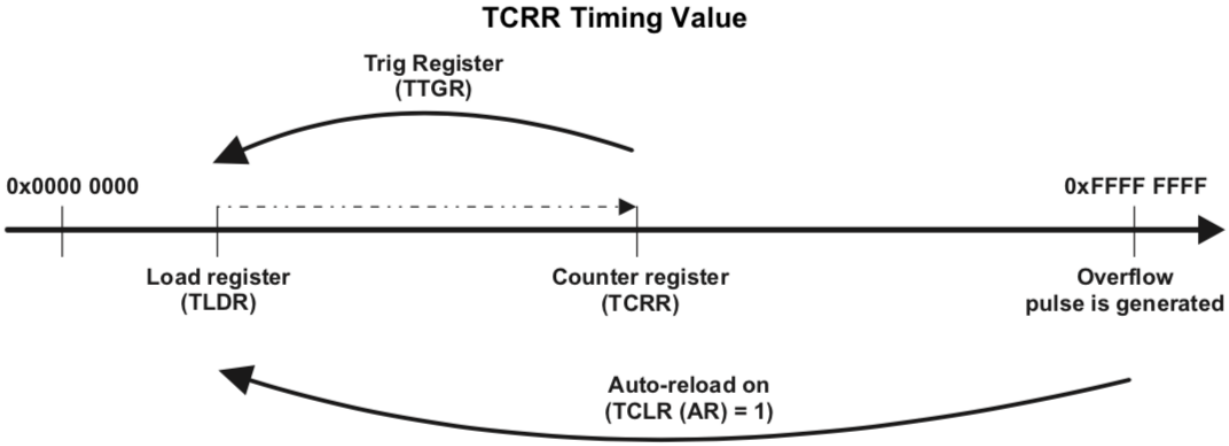

Figure 3: mode compteur

(ref: se12/docs/01_datasheets/01_am335x/06_am335x_technical_reference_manual.pdf, chapitre 20.1.3.1, page 4331)

Les registres des DMTimer 2 à 7

La figure “registers” ci-dessous montre les registres des DMTimers 2 à 7.

Ces registres sont placés dans l’espace adressable du µP aux adresses suivantes (voir pages 181-182)

- DMTimer2 : `0x4804_0000 - 0x4804_0FFF`
- DMTimer3 : `0x4804_2000 - 0x4804_2FFF`
- DMTimer4 : `0x4804_4000 - 0x4804_4FFF`
- DMTimer5 : `0x4804_6000 - 0x4804_6FFF`
- DMTimer6 : `0x4804_8000 - 0x4804_8FFF`
- DMTimer7 : `0x4804_A000 - 0x4804_AFFF`

(ref: se12/docs/01_datasheets/01_am335x/06_am335x_technical_reference_manual.pdf, page 4340)

Questions

- Quelle est la signification du qualificatif `volatile` et quelle est son utilité quand il est associé à un pointeur ?
- Comment sont placés les champs (membres) d’une structure dans la mémoire ?
- Comment peut-on efficacement définir les registres d’un contrôleur de périphérique situés dans l’espace d’adressage du µP ainsi que leur contenu en C ?
- Comment peut-on accéder ces registres ?
- Comment générer des nombres aléatoires ?

Table 20-10. TIMER Registers

Offset	Acronym	Register Name	Section
0h	TIDR	Identification Register	Section 20.1.5.1
10h	TIOCP_CFG	Timer OCP Configuration Register	Section 20.1.5.2
20h	IRQ_EOI	Timer IRQ End-of-Interrupt Register	Section 20.1.5.3
24h	IRQSTATUS_RAW	Timer Status Raw Register	Section 20.1.5.4
28h	IRQSTATUS	Timer Status Register	Section 20.1.5.5
2Ch	IRQENABLE_SET	Timer Interrupt Enable Set Register	Section 20.1.5.6
30h	IRQENABLE_CLR	Timer Interrupt Enable Clear Register	Section 20.1.5.7
34h	IRQWAKEEN	Timer IRQ Wakeup Enable Register	Section 20.1.5.8
38h	TCLR	Timer Control Register	Section 20.1.5.9
3Ch	TCRR	Timer Counter Register	Section 20.1.5.10
40h	TLDR	Timer Load Register	Section 20.1.5.11
44h	TTGR	Timer Trigger Register	Section 20.1.5.12
48h	TWPS	Timer Write Posting Bits Register	Section 20.1.5.13
4Ch	TMAR	Timer Match Register	Section 20.1.5.14
50h	TCAR1	Timer Capture Register	Section 20.1.5.15
54h	TSICR	Timer Synchronous Interface Control Register	Section 20.1.5.16
58h	TCAR2	Timer Capture Register	Section 20.1.5.17

Figure 4: registers

- A la fréquence maximale (24MHz), le compteur du timer ne permet de compter le temps que sur un intervalle de 3 minutes environ. Décrivez l'algorithme à mettre en place si l'on souhaite compter sur plusieurs années avec la même granularité.

Mises à jour

- Pour mettre à jour la bibliothèque spécialisée du Beaglebone

```
$ cd ~/workspace/se12/tp
$ git pull upstream master
$ make -C ~/workspace/se12/tp/bbb/source
```

- Pour mettre à jour les paths des includes dans eclipse
  - ouvrir *Properties* de votre projet
  - aller *C/C++ General* –> *Paths and Symbols*
  - ouvrir *Includes* –> *GNU C*
  - ajouter */home/lmi/workspace/se12/tp/bbb/source*

Conditions

- Rendu
  - Le code et le rapport seront rendus au travers du dépôt Git centralisé
    - sources: *.../tp/tp.04*
    - rapport: *.../tp/tp.04/doc/report.pdf*
- Délai
  - Le journal et le code doivent être rendus au plus tard 7 jours après le TP à 23h59