TP n°3

Informatique embarquée

Vincent Ruello

À rendre avant vendredi 31/10/2024 - 20:00 (CET)

Le but de ce TP est de faire clignoter la LED intégrée d'une carte Arduino UNO R3 à l'aide de *timers* en utilisant `avr-libc`.

Contexte

La carte Arduino UNO est une plateforme open-source de développement permettant de programmer un ATmega328P. sLa documentation du microcontrôleur ATmega328P est disponible ici[1].
AVR Libc est une bibliothèque open-source dont le but est de fournir une libc utilisable sur l'architecture AVR avec GCC. Sa documentation est disponible ici[2].

Partie 1. Programmation en C avec avr-libc

Cette partie a pour but de prendre en main la programmation en C d'un ATmega328P avec avr-libc.

1. Écrire un programme en C qui fait clignoter la LED intégrée toutes les 500ms.

A la place d'utiliser `avr-as` (assembleur) puis `avr-ld` (éditeur de lien), on utilisera uniquement `avr-gcc`. Il est nécessaire de fournir à `avr-gcc` les options suivantes qui permettent au compilateur de connaître la fréquence d'exécution des instruction et le modèle de MCU utilisé ainsi que de spécifier le type d'optimisations souhaité :

```
-DF_CPU=16000000UL -mmcu=atmega328p -Os
```

De plus, on pourra utiliser la fonction `_delay_ms` de avr-libc fournie dans `<util/delay.h>`.

- N'exécuter que le pré-processeur (option `-E` de `avr-gcc`) et observer le code généré. Comment sont traduites les macros représentant les registres I/O (`PORTB`, `DDRB`, ...) ?

- Quelles sont les sections présentes dans l'exécutable ELF produit par `avr-gcc` ? On pourra utiliser l'outil `avr-readelf`. Quelle est la signification des octets présents dans la section `.text` ? et dans la section `.data` ?

1 http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061A.pdf
2 https://www.nongnu.org/avr-libc/user-manual/index.html

- Observer attentivement la séquence de démarrage du programme. Que se passe-t-il avant l'appel de votre fonction `main` ?

2. Écrire un programme en C qui fait clignoter la LED intégrée en boucle avec le motif suivant :

```
=.=.=...===.===.===...=.=.=........
```

où :

`=` représente la LED allumée pendant 500ms
`.` représente la LED coupée pendant 500ms

Partie 2. Utilisation d'un timer

Cette partie a pour but de préparer l'utilisation d'un timer avec interruptions pour remplacer l'attente active avec `_delay_ms`.

1. Quelle instruction AVR permet d'activer la gestion globale des interruptions ? Comment effectuer cette opération avec avr-libc ?

2. On suppose qu'on utilise le timer 8-bit `Counter0` du MCU ATmega328P. L'horloge utilisée par le timer, $\texttt{clk}_{\texttt{IO}}$, a une fréquence de 16MHz. Le timer est utilisé en *Normal Mode*. Quelle est la durée entre deux interruptions d'overflow (TOV) émises par le timer ?

3. Il est possible de réduire cette fréquence en configurant le *prescaler.* Celui-ci propose plusieurs modes : /8, /64, /256, /1024. Quelle est la durée maximale configurable entre deux interruptions d'overflow (TOV) émises par le timer Counter0 en mode Normal ?

4. Reprendre les questions 2 et 3 avec le timer 16-bit Counter1.

5. Le mode CTC permet de paramétrer la résolution du timer. Trouver un couple (OCR1A, prescaler) tel qu'une interruption COMPA soit émise toutes les 500ms.

6. On observe que, même avec le prescaler /1024 et le timer 16-bit, la durée maximale entre deux interruptions est de quelques secondes. Comment faire pour attendre plusieurs minutes ? heures ? jours ?

Partie 3. Clignotement de la LED intégrée à l'aide de Counter1

A l'aide des étapes précédentes, utiliser le timer Counter1 pour reproduire le motif de clignotement de la partie 1. Le MCU devra être mis en sommeil (dans le mode adéquat) dès que possible.
On prendra soin de désactiver les modules non utilisés du MCU afin d'optimiser la consommation énergétique.

Quels sont les avantages et inconvénients d'utiliser un périphérique timer à la place d'une attente active ?

Partie 4. Clignotement de la LED intégrée à l’aide du WDT

Le but de cette partie est d’utiliser le Watch Dog Timer pour reproduire le motif de clignotement de la partie 1. Le MCU devra être mis en sommeil (dans le mode adéquat) dès que possible.
On prendra soin de désactiver les modules non utilisés du MCU afin d’optimiser la consommation énergétique.

Quels sont les avantages et inconvénients d’utiliser le WDT à la place d’un simple périphérique timer ?