Systèmes Embarqués 1 & 2

Classes T-2/I-2 // 2017-2018

p.02 – Interruptions (2e partie)

Solutions

Exercice 1

Concevez un programme qui utilise les interruptions pour mesurer le temps qui s'écoule entre deux pressions de touche successives. La valeur minimale, la valeur maximale et le nombre de pressions de touche doivent être calculées.

Deux périphériques sont à disposition:

(a) Une horloge temps réel (real-time clock).
Ce périphérique émet une interruption à intervalle régulier

(b) Un clavier ou une souris.
Le périphérique utilisé émet une interruption à chaque pression d'une touche

La routine de service de l'horloge temps réel incrémente, à chaque interruption, une variable globale clock_count.

Solution:

```
// Une solution possible, en faisant abstraction de l'attachement des
// routines de traitement d'interruptions à leur vecteurs

uint32_t clock_count = 0;
uint32_t keyclick_count  = 0;
uint32_t last_clock_count   = 0;
uint32_t min_interval = -1;
uint32_t max_interval = 0;

void clock_isr()
{
  clock_count++;
}

void keyclick_isr()
{
    keyclick_count++;

    uint32_t clock_val = clock_count;
    uint32_t delta = clock_val - last_clock_count;
    if (keyclick_count > 1) {
      if (delta > max_interval) max_interval = delta;
      if (delta < min_interval) min_interval = delta;
    }
    last_clock_count = clock_val;
 }
```

Exercice 2

Quel est le résultat de l'instruction ci-dessous si l'interruption « irq_h » survient lors de son exécution ?

```
int len = 0;

len += 2;   // irq_h survient ici ?

void irq_h()
{
  len += 4;
}
```

Solution:

L'instruction len+=2; se décompose comme suit

```
        // <-- si irq arrive ici --> len == 6
ldr r0, =len
        // <-- si irq arrive ici --> len == 6
ldr r1, [r0]
        // <-- si irq arrive ici --> len == 2 !!!
add r1, r1, #2
        // <-- si irq arrive ici --> len == 2 !!!
str r1, [r0]
        // <-- si irq arrive ici --> len == 6
```

Exercice 3

Le µP est en train d'exécuter une application dans le mode utilisateur et les interruptions FIQ et IRQ sont autorisées. Que se passe-t-il et/ou quel est l'état du processeur, si

(a) une instruction SVC est appelée par l'application

(b) une interruption IRQ est levée puis une interruption FIQ est levée

(c) une interruption FIQ est levée puis une interruption IRQ est levée

Solution:

- une instruction SVC est appelée par l'application
  - Processeur entre dans le mode superviseur (SVC)
  - CPSR sauvé dans SPSR_svc, adresse de retour sauvée dans LR_svc
  - IRQ sont désactivées
- une interruption IRQ est levée puis une interruption FIQ est levée
  - Processeur entre dans le mode interrupt (IRQ)
  - CPSR sauvé dans SPSR_irq, adresse de retour + 4 sauvée dans LR_irq
  - IRQ sont désactivées
  - Processeur entre dans le mode fast interrupt (FIQ)
  - CPSR sauvé dans SPSR_fiq, adresse de retour + 4 sauvée dans LR_fiq
  - FIQ sont désactivées
- une interruption FIQ est levée puis une interruption IRQ est levée
  - Processeur entre dans le mode fast interrupt (FIQ)
  - CPSR sauvé dans SPSR_fiq, adresse de retour + 4 dans LR_fiq
  - FIQ et IRQ sont désactivées
  - IRQ n'a aucun effet, doit attendre la fin de traitement de la FIQ

Exercice 4

Expliquer la différence dans le traitement d'une interruption vectorisée et le traitement d'une interruption non-vectorisée. Quels sont les avantages et désavantages de ces deux types de traitement.

Solution:

Vectorisée

- Avantages
  - Simplicité logicielle (vecteur d'interruption en une lecture)
  - Performance (pas de temps de scrutation)
- Désavantages
  - Complexité matériel

Non-vectorisée

- Avantages
  - Simplicité matériel (peu onéreux)
- Désavantages
  - Complexité des algorithmes logiciels
  - Temps de scrutation élevé