Charlotte Junod et Nicolas Fuchs
charlotte.junod@edu.hefr.ch et nicolas.fuchs@edu.hefr.ch
Fribourg, le 19.01.2017

Systèmes embarqués I – journal

TP06

Heures de travail en dehors des cours : $2^1/_2$

Feedback :

Dans ce TP, nous avons créé un pilote qui contrôle le module température du Beaglebone. Les boutons permettent de définir les registres haut et bas. Nous avons pu compléter nos connaissances sur la gestion des périphériques du Beaglebone ainsi que sur les structures et pointeurs C. Cependant, le codage du fichier epwm1.c n'a pas été facile; la documentation est difficile à comprendre (heureusement nous avons pu demander de l'aide à nos camarades).

Questions :

Quelle est la fonction des 5 registres internes du thermomètre TMP102 ?

Pointer Register :	Pointeur de 8 bits qui sert à pointer sur le registre (data) sur lequel on va lire ou écrire.
Temperature Register :	Registre de 12/13 bits qui permet de lire la température
Configuration Register :	Registre de 16 bits (lecture et écriture) qui stocke des bits pour le contrôle des modes d'opérations du capteur thermique.
T_{LOW} Register :	Registre de 16 bits qui stocke la valeur de la limite de température inférieure.
T_{HIGH} Register :	Registre de 16 bits qui stocke la valeur de la limite de température supérieure.

Comment le TMP102 génère-t-il le signal d'alarme ALERT ?

Le master envoie une commande d'alerte de 8 bits sur le bus. Si la pin ALERT est active, le périphérique répond en envoyant son adresse sur la ligne SDA. Le huitième bit (LSB) de cette adresse indique si l'alerte est due au fait que la température est au-dessus de T_{HIGH} au-dessous de T_{LOW}. (bit à low si température >=T_{HIGH}, bit à high si température <=T_{LOW}).

Pour quelle raison le TMP102 dispose d'un registre T_{HIGH} et T_{LOW} ?

Le but premier est de pouvoir définir un intervalle de température acceptable et de déclencher l'alarme si la température captée sort de cet intervalle.
L'usage que nous en avons fait dans ce TP est un peu différent:
lorsque la température dépasse le seuil fixé, le signal d'alarme est déclenché. Si on avait pris un seul seuil et que le signal de température venait à osciller autour de ce seuil, l'alarme s'allumerait et s'éteindrait en boucle.
En utilisant deux seuils au lieu d'un, on évite cette oscillation du signal d'alarme. Il faut que la température descende en dessous de T_{LOW} pour désactiver l'alarme et qu'elle remonte au-dessus de T_{HIGH} pour réactiver l'alarme.

Charlotte Junod et Nicolas Fuchs
charlotte.junod@edu.hefr.ch et nicolas.fuchs@edu.hefr.ch

Quels sont les domaines d'application des pointeurs de fonctions ?

L'appel de différentes fonctions est beaucoup plus flexible et modulable. On peut s'en servir pour stocker des fonctions dans des tableaux et automatiser certaines opérations. On les utilise aussi très souvent pour concevoir des interfaces avec des fonctions de rappel (appelé aussi Callback). On arrive également avec ces pointeurs de fonctions à concevoir des algorithmes génériques pour les traitements de données.

Comment déclare-t-on un pointeur de fonction ?

On peut le faire avec un typedef :

```
typedef return_type (*pointer_type) (function_parameters);
pointer_type function_name;
```

ou bien sans :

```
return_type (*function_name) (function_parameters);
```

Comment utilise-t-on un pointeur de fonction ?

Pour déclarer un pointeur de fonction:

pointer_type function_name = &function; (le caractère '&' n'est pas obligatoire)

Pour appeler la fonction :

```
type value = (*function_name) (function_parameters);
```

ou

```
type value = function_name (function_parameters);
```

Si la fonction renvoie void, on omet simplement value :

```
(*function_name) (function_parameters);
```

ou

```
function_name (function_parameters);
```

Comment le compilateur implémente-t-il un pointeur de fonction en assembleur ?

Ce sont des adresses mémoires.
Quand on utilise notre pointeur de fonction, on est redirigé vers l'adresse du pointeur puis vers l'adresse qu'il contient, où la fonction est effectivement stockée.