Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Systèmes embarqués 1 : TP.05 Pilote d'un timer du AM335x

Horner Frédéric / Pharisa Valentin, HEIA Fribourg le 06.11.2017

Synthèse

Non-acquis : L'utilisation des volatiles, la manipulation des registres de contrôleurs, les champs membres d'une structure,…

Acquis mais à exercer : Le fonctionnement des fichiers button.c et wheel.c et de leurs entêtes respectifs

Parfaitement acquis : La manipulation des structures en c et de la majorité des variables

Questions

1. **Quelle est la signification du qualificatif « volatile » ?**
   Il désigne un type de variable pouvant être modifié à l'extérieur de l'implémentation de notre code par exemple par le matériel ou encore le système d'exploitation. Le compilateur ne va optimiser cette partie du code afin de laisser cette variable accessible en tout temps.
2. **Quelle est l'utilité du qualificatif « volatile », quand il est associé à un pointeur ?**
   Cela indique à notre compilateur que la variable est changeante et peut-être modifiée par un autre processus hormis l'exécution en cours. C'est utile lors de la manipulation des I/O par exemple.
3. **Comment peut-on efficacement définir les registres d'un contrôleur de périphérique situés dans l'espace d'adressage du µP et leur contenu en C ?**
   Nous avons défini les registres du DMTimer dans le fichier « dmtimer1.c » de la manière suivante :

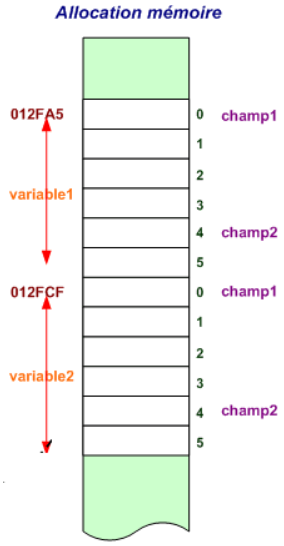
```
struct timer1_ctrl {
        uint32_t tidr;          // 00
        uint32_t res1[3];       // 04-0c
        uint32_t tiocp_cfg;     // 10
        …

};
```

4. **Comment peut-on accéder ces registres ?**
   Il faut consulter la datasheet de notre appareil pour trouver à quel adresse mémoire est lié notre composant. Une fois fait, il faut utiliser le « include <am335…>. Nous devons ensuite implémenter la structure comme fait au point 3.

5. **Comment sont placés les champs (membres) d'une structure dans la mémoire ?**
   Chaque structure dispose d'un certain espace mémoire. Une fois les champs déclarés, ceux-ci sont insérés les uns après les autres dans la structure
   Exemple code :

   struc Type{
   type 1, champ 1
   type 2, champ 2, champ 3
   …
   } variable 1, variable 2



6. **Quelle est l'utilité des conversions de type (type casting) en C ?**
   Elle nous permet, à l'instar de Java, de convertir des variables afin de les traiter dans un format voulu et de ne pas perdre de l'information.
   Exemple : double division = maVariable / (double)taVariable;
7. **Comment réalise-t-on un "type cast" en C ?**
   Comme dans l'exemple plus haut, il faut rajouter le type entre parenthèse.
   Exemple : on veut caster un int en double
   maVariable -> (double)maVariable

Remarques

Le C est très intéressant à manipuler mais nécessite encore de l'exercice afin de comprendre certaines finesses telles que les pointeurs, les volatiles,… Nous avons encore de la peine à comprendre comment manipuler nos périphériques.

Nombre d'heures passées

Pharisa Valentin : 5h- 6h

Horner Frédéric : 5h – 6h

Feedback

Ce TP nous a permis de mieux comprendre l'interaction entre l'hardware de notre beaglebone et le code C. Nous avons trouvé ce TP très intéressant mais ne comprenons pas encore tout le fonctionnement de notre code. Il sera nécessaire de poser des questions pour combler les lacunes liées à la manipulation des périphériques et l'espace d'adressage.