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

Systèmes embarqués I - Journal
TP01 : Introduction

Heures de travail en dehors des cours : 2

Synthèse et feedback

Ce TP nous a permis de nous familiariser avec l'environnement de travail et avec cette notion de registres utilisés pour manipuler des données, en plus de nous donner un premier aperçu du langage assembleur.

Nous avons compris que lorsqu'on code en assembleur, en opposition aux langages de plus haut niveau dont nous avons l'habitude, il faut prendre soin de charger dans les registres les valeurs stockées dans la RAM et de retransmettre les résultats sur les variables.

Par contre, nous n'avons pas saisi la signification de toutes les instructions (bne par exemple), mais nous pensons que nous y parviendrons à force de TP et de cours théoriques.

Questions

Quelle est la taille de chacune des variables ?

Nos trois variables, res, incr et i, sont initialisées de la manière suivante :

```
res:    .long 16
incr:   .short 32
i: .space 4
```

res est un long, il occupera donc 32 bits.
Incr est un short, il occupera donc 16 bits.
Pour i, on demande de réserver un espace de 4 octets, soit 32 bits.

Quelle est la taille du code ?

10396 octets.

Comment procéder pour obtenir ces tailles ?

Pour les variables, il suffit de regarder dans le code leur type ou l'espace qu'on demande de leur attribuer. Pour la taille du code, nous nous sommes servis du fichier app_a.map qui nous indique notamment la taille complète du code en hexadécimal (en l'occurrence 0x289c = 10396 en décimal).

Où se trouve chaque variable en mémoire (valeur absolue) ?

incr : 0x80003b04
res : 0x80003b00
i : 0x80003d00

Où se trouve le code en mémoire ?

Grâce au débogueur, en regardant les registres, on voit au début de l'exécution que le pointeur d'instruction (pc) a pour valeur 0x80001124. Cette valeur correspond à l'adresse du début des instructions dans la RAM.

Est-il possible d'améliorer l'algorithme ?

Oui, nous avons une version optimisée disponible dans le code source du TP.

Nous l'avons testée et avons vérifié qu'elle fonctionne. Les améliorations sont les suivantes : on utilise moins de registres (5 au lieu de 6) et on procède à moins d'échanges entre les registres et la RAM.