# Systèmes Embarqués

# Journal - TP.07 : Interfaçage assembleur-C

## Temps effectué hors des heures de classe

Nous avons effectué 6h de plus en dehors des quatre périodes mises à disposition pour ce TP.

## Synthèse des acquis

Acquis :
- Utilisation de code assembleur au sein d'un projet codé en C

Acquis, mais à exercer encore :
- Langage assembleur & C

## Questions

### L'affichage du contenu des registres du µP peut être réalisé de différentes manières. Citez différentes méthodes et donnez les avantages et désavantages de ces dernières.

La première méthode est celle que nous avons utilisée dans notre code. Il s'agit de récupérer l'état des registres et de l'afficher dans le terminal directement en assembleur. L'avantage de cette manière est qu'elle est rapide, pour autant qu'elle soit réalisée correctement bien sûr. Son défaut est que plus de lignes sont nécessaires afin de réaliser cette opération qu'il en serait nécessaire en utilisant du C.

Une deuxième méthode consiste à récupérer l'état des registres en assembleur, puis de l'afficher en C. Cette méthode est plus simple à écrire du fait que le C est un langage de plus haut niveau que l'assembleur, mais elle est en revanche légèrement plus longue à exécuter.

### Existe-t-il une autre technique que de passer par un fichier en assembleur pour accéder ses registres spéciaux du µP ? Si oui, décrivez-la succinctement.

La première manière consiste à écrire du code rassembleur directement dans un fichier C :

```
1 __asm__("movl %edx, %eax\n\t"
2         "addl $2, %eax\n\t");
```

Cette méthode est appelée *Inline Assembly* et est utilisée pour optimiser certaines opérations, faire des *System Calls* ou, comme il aurait été possible lors de ce TP, d'accéder aux instructions spécifiques à l'assembleur et notamment à la manipulation des registres.

Une autre méthode est d'utiliser le mot-clé *register* du langage C. Cependant, cette manière de faire est maintenant déconseillée malgré le fait qu'elle fonctionne toujours puisque nous l'avons testée.

```
1 int printR6() {
2     register int r6 asm("r6");
3   printf("r6 value: %d\n", r6);
4 }
```

## Remarques

Il est nécessaire d'utiliser un "\n" en fin de *printf* afin que l'affichage se fasse en console.

La séquence "Ctrl-A", puis "C" permet de clean la console Minicom.

## Feedback

Il est intéressant de connaître l'interfaçage assembleur - C afin d'utiliser des possibilités qu'offre l'assembleur et dont le C ne dispose pas, comme l'accès aux registres. Le fait d'avoir créer une librairie nous permettant de lire les registres nous sera certainement utile lors de futurs projets.

Il est également très bien d'avoir à disposition un jour de plus pour rendre le rapport. Faire six heures (minimum) de système embarqué sur le même jour peut s'avérer comme étant un peu trop. Le fait d'avoir une journée supplémentaire nous permet de rendre un travail que nous trouvons de meilleur qualité qu'il en aurait été si nous avions dû le rendre le soir même.