Informatique Embarquée, TP n°6b

Luxon JEAN-PIERRE

Temps estimé : 5 heures
Temps passé : 4 heures

À propos de *typeof* et *offsetof* :

typeof est un opérateur qui permet d'avoir le type de l'élément donné en paramètre.
offsetof est un opérateur qui permet d'avoir l'offset d'un membre d'une structure ou d'une union. Cette opérateur prend en paramètre le nom de la structure, et le membre de cette structure.

À propos du code exemple :

Il est possible d'insérer un objet dans 2 listes doublement chaînées au sein d'un même programme. En effet, dans le code donné en exemple, il suffit que le champs *list_all* soit un tableau de *struct head*, afin d'avoir différent nodes. Chaque node pouvant être dans une liste spécifique.

HashList :

J'ai choisi de définir l'interface suivante :

```
struct hash
{
    struct list_head ** hash_table;
    size_t size;
};

#define INIT_HASH_TABLE(htable) htable = htable_make();

#define DESTROY_HASH_TABLE(htable) free(htable->hash_table); free(htable);

struct hash * htable_make();

bool htable_add(struct list_head * node, struct hash * htable);

bool htable_del(struct list_head * node, struct hash * htable);

struct list_head * htable_find(size_t node_id, struct hash * htable);
```

Ma structure va contenir un double pointeur vers *list_head*, de sorte que chaque élément de ma table soit un pointeur vers une liste doublement chaînée telle qu'elle a été définie avant. Au niveau de la mémoire, je ne gère que la création et la destruction de la table, pas des listes. Ces listes seront gérées par l'utilisateur de ma table de hachage. Ma table aura une taille qui, dans le cadre du TP, sera de 10. Une extension possible serait de donner, lors de la construction de la table, une taille qui sera définie par l'utilisateur.
La fonction de hachage est définie de la manière suivante :

```
index = node_id % htable->size;
```

Voir code source.