Axel papillier
Aymeric Bagamboula Loundi

Rapport projet S2 informatique embarquée

Partie 1 :

Nous devions réaliser une fonction qui permettait d'encoder une chaine de caractères saisie par l'utilisateur. Nous avons réalisé cette fonction sur codeblocks mais pour la partie 5 nous avons dû modifier quelques points de la fonction pour la « transférer sur µvision.

La fonction : int base64_encode(char *dst, int size, char *src)

Cette fonction prend en argument un tableau dst qui sera le tableau ou sera stocké notre chaine de caractères encodé, size sera la taille a alloué pour dst, src sera le tableau ou sera stocké la chaine entrée par l'utilisateur.

Pour réaliser cette fonction nous avons dû faire des « paquets » de 3 char soit 3 octets que nous avons nommé oct_a oct_b et oct_c. Nous avons donc mis les 3 octets dans un int car 3 octets = 24 bits donc il fallait au moins 32 bits pour stocker nos 3 octets.

Nous utilisons ensuite ceci dans une boucle pour encoder notre chaine.

```
dst[j++] = tab[(triple >> 18) & 63];
dst[j++] = tab[(triple >> 12) & 63];
dst[j++] = tab[(triple >> 6) & 63];
dst[j++] = tab[(triple) & 63];
```

Nous remplaçons les bits vides par des « = » comme ceci :

```
for (i = 0; i < mod_table[size % 3]; i++)
   dst[size-1 - i] = '=';
```

Et étant donné que nous avons malloc dst, on n'oublie pas de rajouter le \0 à la fin de la chaine et de free dst.

Nous n'avons pas réussi à réaliser la fonction décodage, mais avec l'utilisation que nous en faisons, elle ne nous sera pas utile pour la suite.

Partie 2 :

Pour la partie 2, il y avait des fonctions fournies permettant la liaisons port série entre la carte et le pc mais lorsque nous compilions sur µvision, il y avait des erreurs, malgré avoir rajouté tous les includes, µvision ne trouvais pas les bibliothèques données.

Nous avons donc opté pour une solution qui marche mais qui a un problème :

```
while (mess[0]=='\0'){
HAL_UART_Receive(&huart2, (uint8_t*)mess, sizeof(mess), 1000);
}
```

Nous avons créé un tableau de char « mess » initialisé avec que des \0 et tant que la première case de mess est égale a \0 alors la carte va receive. L'inconvenant, nous avons seulement le temps du timeout pour écrire, nous ne devons pas appuyer sur entrer. Nous utiliserons cette méthode pour tous les prochains exercices.

Axel papillier
Aymeric Bagamboula Loundi

Partie 3 :

Pour cette partie, on ne s'intéressera, dans ce rapport, uniquement à la partie encodage cesar sur l'alphabet uniquement étant donné que c'est très similaire à l'encodage sur toute la table ascii ?

Comme pour le 1, nos fonctions encode et decode prennent en argument deux tableau de char dst et src et la taille de dst.

Pour obtenir la clé, nous avons préféré la faire de façon Radom avec les fonctions srand et rand plutôt que le demander à l'utilisateur car c'est plus simple de limiter la valeur de la clé et car l'utilisateur ne sait pas forcément à quoi cela correspond.

Une fois rentré dans notre fonction, on met tous les caractères de notre chaine non encodée grâce a la fonction toupper.

Après nous avons plus qu'à faire cela :

```
while(src[j] != '\0')
{
    if (src[j] >= 'A' && src[j] <= 'Z' ){
        rang = src[j] - 'A';
        rang = (rang + clef) % 26;
        if(rang < 0) rang += 26;
        dst[j] = 'A' + rang;
    }
    else {
        dst[j] = src[j];
    }
    j++;
}
dst[j] = '\0';
```

Nous avons fait un while qui parcourt src (notre chaine non encodée) on regarde grâce au if si le caractère est une lettre de l'alphabet, si s'en est un, alors on trouve son rang dans l'alphabet et on ajout la valeur de la clé (ou on la soustrait pour le décodage). Si c'est un caractère, alors on le laisse comme tel.

Pour l'encodage XOR, nous avons également pris une valeur random pour la clé (de 0 à 255) soit la valeur max sur 8 bits. Une fois la clé initialisée, il suffisait simplement de faire une boucle qui parcours notre chaine non encodée et faire un XOR avec le caractère et notre clé.

Axel papillier
Aymeric Bagamboula Loundi

Partie 4 :

Dans cette partie nous devions simuler un capteur de température dans un four et ajuster la température en fonction de la température demandée par l'utilisateur. Pour cela nous allons imaginer que notre capteur possède un signal sur 12bit.

Pour résoudre ce problème nous avons généré une valeur aléatoire comprise entre 0 et 4095. Cette valeur fait office de valeur analogique du capteur, plus concrètement, se serait la valeur du capteur pour la température initiale du four. Puis nous demandons à l'utilisateur la température qu'il désire régler dans son four.

Ensuite on convertit la valeur numérique, renseigné préalablement par l'utilisateur, en valeur analogique. Puis on affiche les 2 valeurs analogiques, celle du capteur et celle de l'utilisateur. C'est les premières valeurs qui s'affichent lorsqu'on lance le programme.

Ensuite on convertit la valeur analogique du capteur, en valeur numérique. Puis on compare les deux valeurs, et si la température perçue par le capteur est supérieure à celle renseignée par l'utilisateur : on refroidit le four ; et vice versa, on chauffe. Et si les deux températures sont égales on annonce que le four est à bonne température. Et bien-sûr on annonce la température à chaque changement de température.

Partie 5 :

Cette partie regroupe globalement tous les exercices, donc nous ne reviendrons pas sur les parties déjà expliqué précédemment. Néanmoins, nous avons dû faire certaines choses que nous allons détailler :

Premièrement : l'énoncé nous informait qu'il fallait que l'utilisateur s'identifie. Il n'était pas précisé si le mot de passe et l'identifiant était à déclarer en dur dans le programme donc nous avons pris le parti de permettre à l'utilisateur de s'inscrire avant. C'est pourquoi nous avons créé un menu ou l'utilisateur choisit entre s'inscrire, se connecter, ou quitter le programme. La première fois, l'utilisateur s'inscrit, l'identifiant et le mot de passe qui sera d'ailleurs directement après le saisie encrypté en base 64 seront stockés dans une structure de taille 10. Ce qui signifie que 10 utilisateur pourront d'inscrire, après le programme renvoie un message d'erreur et retourne au menu.

Une fois l'utilisateur inscrit, il peut se connecter, nous lui demandons alors son identifiant, s'il existe, alors il peut saisir son mot de passe sinon on lui redemande de le saisir. Lors de la saisie du mot de passe, nous encryptons immédiatement, le mot de passe qui vient d'être saisie et nous comparons le mot de passe crypté avec ceux également crypté dans notre structure. Nous n'avons malheureusement pas eu le temps de faire le double encodage, mais nous l'aurons lors de la soutenance. Si son mot de passe correspond alors il est connecté, il a 3 tentatives avant d'être renvoyé vers le menu en cas d'échec.

Si l'utilisateur se connecte, il est envoyé vers un autre menu, qui indique soit la déconnexion soit la possibilité de régler la température du four (confert exercice 4). Une fois ceci fait, l'utilisateur est renvoyé dans le menu principal.