MI41 final

Durée 1h30. Les fonctions écrites en assembleur ARM doivent respecter la convention d'appel AAPCS, les accès mémoire doivent être minimisés (sauvegardes/restaurations des registres).

1. Exercices ASM (3x4 pts)

1. Ecrire une fonction permettant de comparer deux tableaux d'entiers de taille N, la fonction renvoie 1 si les 2 tableaux sont identiques, 0 sinon.

Prototype :

```
int compare_tab(int* tab1, int* tab2, int N);
```

2. Appel de fonction : donner le code assembleur de la fonction fct1 donnée ci-après :

```
extern int fct2 (int a, int b) ; // prototype de fonction2 appelée par fonction

int fct1 (int a, int b, int n)
{
        int x,i ;
        for (i = 0 ; i<n ; i++){
                x = x + fonction2 (a+b, a-b) ;
        }
        return x ;
}
```

3. Ecrire le code d'une fonction random qui renvoie une valeur aléatoire calculée à partir d'une valeur initiale. La valeur initiale est une variable globale appelée seed (int32_t seed) qui est également mise à jour par la fonction random. La suite des valeurs construites par les appels successif à la fonction random forme une suite dite pseudo aléatoire (SBPA).

Le calcul de la nouvelle valeur aléatoire est donné par :

D0 = Q31 ⊕ Q21 ⊕ Q1 ⊕ Q0

les autres bits sont obtenus par décalage à gauche de la valeur initiale (ancienne valeur) :

D1 = Q0

Di = Qi-1

D31 = Q30

avec Di ième bit de la valeur nouvelle aléatoire et Qi, ième bit de la valeur initiale seed (ancienne valeur aléatoire).

la valeur initiale seed est également mise à jour avec la nouvelle valeur aléatoire calculée.

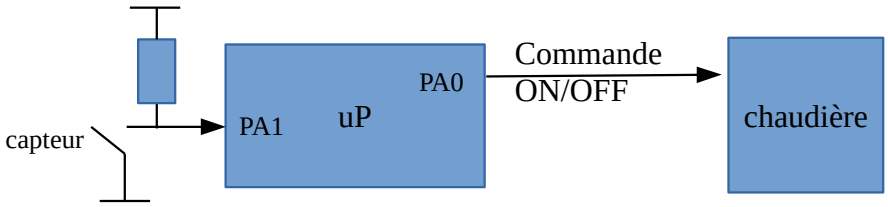
Prototype :

```
int random(void);
```

2. Interface : Commande d'une chaudière à l'aide d'un microprocesseur (8pts)

2.1. Description du système

On utilise une interface PIO pour commander une chaudière. La chaudière est commandée par une valeur unique ON/OFF. Le schéma est donné ci-dessous :



Le principe consiste à récupérer la valeur de consigne '*consigne*' et la température ambiante *'temperature'*. Si la température ambiante est de 5 unités inférieure à la valeur de consigne la sortie de commande est ON (1

logique) sinon elle est de 5 unités supérieure la commande est OFF (0 logique)
Par ailleurs, un capteur externe de type ON/OFF, si à 0, force la commande de la chaudière à OFF, dans le cas contraire (entrée de capteur à 1) le fonctionnement est normal.

L'interface utilisée pour la génération de la commande et l'acquisition de la valeur du capteur est une interface de type GPIO que l'on nommera PA.
La commande est connectée à la broche PA0 associée au bit 0 de PA et le capteur est connecté à la broche PA1 associée au bit 1 de PA.

Le GPIO PA fonctionne avec deux registres 32 bits :

- le registre DIR permettant de définir la direction de chaque broche, le bit 0 définit la direction de la broche PA0, le bit 1, la direction de la broche PA1, et ainsi de suite.
  Si le bit i du registre DIR est à 1 la broche associée est en entrée, si le bit i du registre DIR est à 0 la broche associée est en sortie.
- Le registre DATA permet de lire ou de fixer l'état des broche du microcontroleur selon la valeur du registre DIR (direction des broche). Ainsi pour chaque bit i du registre DATA il est possible en fonction de la direction choisie pour la broche associée :
  - de lire si la broche i est à la tension haute ou basse (entrée ou sortie)
  - de placer la broche i à la tension haute ou basse (uniquement si la broche est en sortie)

Le registre DIR se trouve à l'adresse 0xA0000000
Le registre DATA se trouve à l'adresse 0xA0000004

La température est acquise à l'aide d'une sonde de température reliée au microcontrôleur via un convertisseur analogique numérique.
Le convertisseur est accessible via 3 registres 32 bits mappés en mémoire :
ADC_CNTRL : configuration et contrôle, adresse 0xA0000010
ADC_STATE : état de l'interface, adresse 0xA0000014
ADC_DATA : donnée 32 bits signée (int32_t), adresse 0xA0000018
Pour lancer une conversion, il est nécessaire d'écrire un 1 dans le bit 0 du registre ADC_CNTRL. Les autres bits de ce registre ne doivent pas être modifiés. La conversion analogique/numérique n'étant pas instantanée, l'interface signal une fin de conversion par la mise à 1 du bit 0 du registre ADC_STATE. Une nouvelle donnée est alors disponible pour la lecture dans le registre ADC_DATA.
Une lecture de la donnée présente dans ADC_DATA remet le bit 0 du registre ADC_STATE à 0 ainsi que le bit 0 du registre ADC_CNTRL.

La consigne est obtenue par l'appel de la fonction *int32_t get_consigne(void)*

2.2. Réalisation (6pts)

En commentant soigneusement votre code, écrivez une fonction (boucle infinie) permettant d'acquérir la température et de commander la chaudière selon de cahier des charges donné.

2.3. Fonctionnement en mode automatique sous IRQ (2pts + 2pts bonus)

Le convertisseur analogique numérique a été configuré de manière à ce qu'une conversion soit lancée automatiquement toute les secondes. Lorsque que le bit 0 du registre ADC_STATE est à 1 (fin de conversion) une requête d'interruption est générée par l'interface. La routine de traitement des interruptions *void ADC_Handler(void)* est alors lancée.
Tout le traitement est réalisé au sein de cette routine. Donnez le code soigneusement commenté de cette routine de traitement des interruptions.

2.4. Acquisition de la température

La température est acquise à l'aide d'une sonde de température reliée au microcontrôleur via un convertisseur analogique numérique. Le convertisseur peut fonctionner en mode manuel ou automatique.
Le convertisseur est accessible via 3 registres 32 bits mappés en mémoire :
ADC_CNTRL : configuration et contrôle
ADC_STATE : état de l'interface
ADC_DATA : donnée.
L'interface est configurée en mode automatique, une conversion est lancée toutes les secondes. Lorsqu'une nouvelle conversion est disponible, le bit 0 du registre ADC_STATE passe à 1. Une lecture de la donnée présente dans ADC_DATA remet ce bit à 0.

L'interface est également configurée pour travailler avec interruption. Ainsi lorsqu'une conversion est terminée et que le bit 0 du registre ADC_STATE passe à 1, une requête d'interruption est générée et si les interruptions sont autorisées, la fonction adc_handler est lancée.