AG43 Examen Final

Automne 2016

Horloge et Radio-Réveil

Présentation

On propose d'étudier la programmation d'un microcontrôleur gérant un radio-réveil. Ce microcontrôleur est en liaison avec les éléments suivant :

- Un clavier simplifié
- Un afficheur une ligne
- Un buzzer
- la commande alimentation du récepteur radio
- La commande volume du récepteur radio

Les fonctions principales que gèrera le microcontrôleur sont :

- La mise à jour de la date et de l'heure toutes les minutes
- La gestion de l'alarme
- Le réglage de la date, de l'heure, de l'alarme
- La lecture du clavier
- La gestion du volume de la radio

Variables mémoire

La date est stockée sous la forme d'un tableau de 3 octets `Date[]`, entiers sur un octet. L'année (`Date[2]`) est notée de 0 à 99, représentant une année comprise entre 2000 et 2099.

L'heure est également stockée sous la forme d'un tableau de 3 octets nommé `Heure`. `Heure[0]` représente le jour de la semaine, 0 pour Lundi et 6 pour Dimanche ; `Heure[1]` représente les heures (0.. 23), `Heure[2]` les minutes.

L'alarme comporte deux alarmes (`Al1[]` et `Al2[]`, tableaux de 3 octets) pouvant être sélectionnées séparément. Leur validation se fait par le premier élément de chaque tableau.

On dispose d'une variable de type octet, appelée `Alarme`, qui contient 1 si on veut activer uniquement le buzzer, 2 si on veut activer uniquement la radio, et 3 si on veut activer les deux.

A la mise sous tension, les variables précédentes sont initialisées comme suit :

- date : 01/01/00
- heure : 0h00
- jour de la semaine : 0
- Alarme1 et Alarme2 : 0h00
- Alarmes non activées : `Al1[0], Al2[0] = 0`
- type d'alarme par défaut : buzzer

Fonctionnement

La mise à jour de l'horloge se fait par une fonction interruption, déclenchée toutes les minutes. Dans cette fonction interruption, il est procédé à la mise à jour de la date et de l'heure. Le déclenchement de l'alarme est également géré dans cette interruption.

La boucle principale du programme surveille le clavier (boutons de réglage) et exécute les commandes du clavier (mise à l'heure, réglage alarmes, gestion radio (Marche/Arret)…)

Travail demandé

Il s'agit d'une étude partielle des différentes fonctions gérant le fonctionnement du radio réveil. Les réglages et gestion de la radio ne sont pas pris en compte dans cette première partie.

Exercice 1 : Initialisation

Ecrire la partie du programme qui procède à l'initialisation des variables décrites précédemment. Faire figurer la déclaration des variables. Cette partie de programme est exécutée une fois, lors de la mise sous tension de l'appareil.

Exercice 2 : Etude de la procédure de type Interruption, qui gère la date et l'heure

Dans un premier temps, on ne s'occupera pas de la gestion de l'alarme. Ecrire la procédure de type interruption, nommée `Gestion()`, (elle sera appelée automatiquement toutes les minutes), qui gère la date et l'heure, en incrémentant à chaque déclenchement les minutes `Heure[2]`. Elle mettra également à jour les autres éléments des tableaux `Heure` et `Date`.

On dispose de :

- La fonction `bis(annee)` qui retourne 1 si l'année contenue dans l'argument entier `annee` est bissextile. Ex : `bis(2017)` retourne 0.
- Du tableau d'entiers sur 1 octet `TM[]` indicé par rapport à 0, variable globale, qui indique la durée en jours de chaque mois.

Gestion de l'alarme

Le radio-réveil possède deux alarmes. Chaque alarme est représentée par un tableau (`Al1[]` et `Al2[]`) constitué comme suit :

- `Alx[0]` : Octet de validation. On utilise 7 des 8 bits de cet octet. Chaque bit représente un jour de la semaine. L'organisation est logiquement la suivante : `[0,D,S,V,J,M,M,L]`. `L` représente le bit 0, bit de poids faible. Lorsqu'un bit est à 1, l'alarme est validée pour le jour correspondant. Ainsi, si on programme une alarme devant sonner uniquement le Lundi, Mardi, Jeudi et Vendredi, `Alx[0]` aurait comme valeur binaire `00011011b`. En conséquence, lorsque `Alx[0] = 0`, on sait que l'alarme x n'est pas activée.
- `Alx[1]` : Octet indiquant l'heure de l'alarme (0… 23)
- `Alx[2]` : Octet indiquant les minutes de l'alarme (0… 59)

On dispose de l'opérateur décalage à gauche "`<<`", utilisé comme suit :

```
a = a << n.
```

Cet opérateur procède au décalage simultané de tous les bits de la variable `a` vers la gauche d'une valeur de n bits. Si l'on pose :

```
a = 1 (00000001b)
```

puis :

```
a = a << 3
```

on obtiendrait `a = 8 (00001000b)`.

On dispose également de l'opération binaire ET qui réalise un ET logique bit à bit entre deux octets, ou entre un octet et une constante. Cette opération permet de savoir si un bit donné d'un octet est à 1 ou à 0. Elle permet également de mettre à 0 certains bits d'un octet.

Exercice 3 : Fonction de test activation Alarme 1 ou 2

Ecrire une fonction appelée `IsActivated(tab[])`, admettant comme argument le tableau `Alx`[], et retournant 1 si l'alarme est activée pour le jour courant, 0 si elle n'est pas activée. Vous disposez de la variable globale `Heure[]` et des opérateurs indiqués précédemment.

Exercice 4 : Fonction d'autorisation de mise en route d'alarme

On propose d'écrire une fonction `Alarme()`, n'admettant aucun argument, et retournant un entier, qui sera appelée depuis la procédure d'interruption `Gestion()`. Cette fonction dispose des variables globales `Al1`, `Al2` et `Heure` et de la fonction `IsActivated()`. Cette fonction, vérifie si une des deux alarmes doit sonner, et dans ce cas retourne 1 ou 2 selon l'alarme effective (Al1 ou Al2), et 0 si aucune alarme n'est à actionner.