Correction devoir surveillé

Semestre : 1 ☐ 2 ■

Session : Principale ■ Rattrapage ☐

Unité d'enseignement : UP embarqué

Module : Architecture des microcontrôleurs

Classe(s) : 2A, 2P Nombre de pages : ENONCE (4 pages) +ANNEXE (1 pages)

Documents autorisés : ☐ Oui ■ Non Calculatrice autorisé : ■ Oui ☐ Non

Date : 06/04/2024 Heure : 09H00 Durée : 1H

QCM (10 points) : Cocher la ou les bonne(s) réponse(s) :

1. **Le PIC-16F84 appartient à la famille Mid-range 14 bits. Cette valeur correspond à :**

   A. La taille du registre program counter
   B. L'adresse d'un registre
   C. La taille d'une instruction
   D. La taille d'une donnée en EEPROM

2. **Le PIC 16F84 possède un registre de travail interne W :**

   A. De taille 14 bits permettant de coder les instructions.
   B. De taille 13 bits, permet de stocker l'adresse de retour suite à l'exécution d'un sous-programme.
   C. De taille 8 bits, joue le rôle d'accumulateur pour le chargement des résultats intermédiaires.
   D. De taille 16 bits car le microcontrôleur est de famille mid range.

3. **On a un programme qui nécessite 0.5 µs (CI) pour exécuter une instruction. Déterminer la fréquence du microcontrôleur $F_{\mu c}$ et le temps nécessaire pour exécuter ce programme T_E, sachant que le nombre de cycle d'instruction NB= 250.**

   A. $F_{\mu c}$ = 4 MHz ; T_E = 250 µs
   B. $F_{\mu c}$ = 11 MHz ; T_E = 90 µs
   C. $F_{\mu c}$= 8 MHz ; T_E = 125 µs
   D. Aucunes réponses

4. **Quelle est la taille de la mémoire de programme qu'on a le droit de manipuler du pic16F84**
   A. 2^{13} mots ×14 bits
   B. 2^{10} mots × 13 bits
   C. 1024 mots × 14 bits
   D. 2^{10} mots × 14 bits

5. **Quelle est la fonction du registre PC (Program Counter) dans le microcontrôleur Pic 16F84?**
   A. Stocker le résultat des opérations arithmétiques et logiques
   B. Stocker l'adresse de la prochaine instruction à exécuter
   C. Stocker les constantes numériques
   D. Stocker les données d'entrée

6. **La réalisation d'une opération logique par le processeur de notre microcontrôleur pic16F84 a un effet sur :**
   A. Le bit Carry
   B. Le bit Digit Carry
   C. Le bit Zéro
   D. Toutes les réponses sont correctes.

7. **Nous avons le bout de code suivant :**

```
Unsigned char A, B, Somme;
A = 158 ;
B = 98 ;
Somme = A + B ;
```

   Cochez-la ou les proposition(s) correcte (s) :

   A. C=0 ; Z=0 ; Somme = 256
   B. C=1 et Z=1 ; Somme = 0
   C. Elle n'affecte pas le registre STATUS
   D. C=1 et Z=1 ; Somme =256

8. **Nous avons le bout de code suivant :**

```
Unsigned char b=276, a=2;
while(a<=5)
        {  a++;
           b=b - a; }
```

Au bout de 4 itérations de la boucle while :
   A. Z =0, C=0, b= 258
   B. Z =1, C=1, b=2.
   C. Z =0, C=1, b=2
   D. Z =0, C=1, b= 258

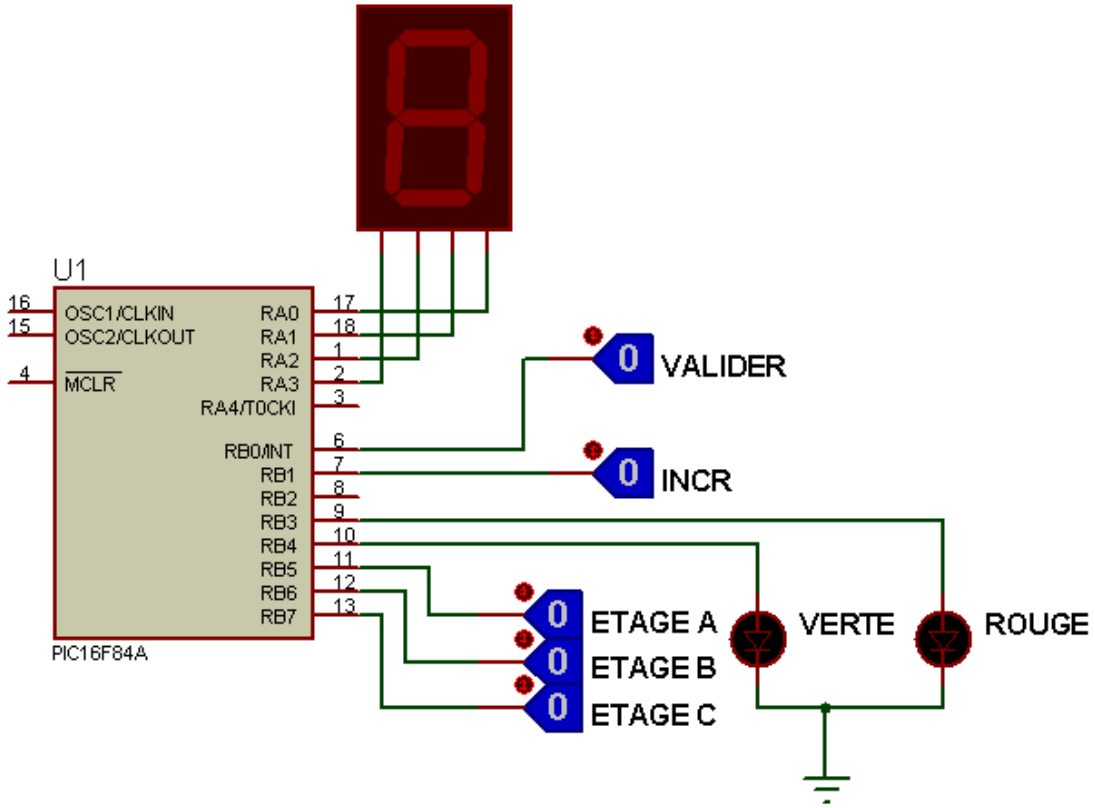
Problème (10 points) :

Dans une cabine d'ascenseur conçue pour accueillir jusqu'à 4 personnes, **la présence de ces 4 occupants est nécessaire pour démarrer**. Nous souhaitons réaliser un système de gestion intégré permettant aux utilisateurs de connaître le nombre de places disponibles avant de choisir l'étage souhaité.

Le système fonctionne à base d'un PIC16F84 et dispose de :

- **Un bouton INCRE** permettant d'incrémenter le nombre de personnes disponibles dans la cabine.
- **Deux lampes LEDs : LED_R & LED_V**, permettant d'indiquer l'état de la cabine (pleine ou pas encore).
- **Trois boutons (ETAGE_A, ETAGE_B et ETAGE_C)** permettant de sélectionner l'étage choisi.
- Un bouton **VALIDER** permettant de valider le choix de l'étage sélectionné.
- **Un afficheur BCD** indiquant :
    - Le nombre de personnes disponibles dans la cabine
    - L'étage sélectionné

Le montage ci-dessous décrit les composants du système :



Fonctionnement :

- Au départ, lorsque le système s'active, il affiche le nombre de personnes présentes dans la cabine. À ce stade, la valeur de la variable « **NbPers** » est **nulle**. La LED rouge s'allume tandis que la LED verte s'éteint, signalant ainsi qu'aucune personne n'est présente dans la cabine et qu'elle n'est pas encore pleine.

- Le bouton **INCRE** simule l'entrée d'une personne dans la cabine, ce qui entraîne l'incrémentation de la variable **« NbPers » qui est constamment affichée sur le BCD**. Si le nombre de personnes dans la cabine atteint la valeur 4, **la LED_V s'allume** et **la LED_R s'éteint.**

- Une fois que l'ascenseur est plein, l'une des personnes disponibles peut **sélectionner l'étage** désiré en **appuyant sur l'un des 3 boutons** (ETAGE_A, ETAGE_B et ETAGE_C). Dès qu'elle appuie sur l'un de ces boutons, cela déclenche l'affichage de la lettre correspondant à l'étage choisi. Notons que chaque appui sur un bouton envoie un signal logique de 1.

- Pour **valider ce choix**, il suffit d'appuyer sur le bouton (**VALIDER=0).** Cette action entraîne immédiatement le clignotement des deux LEDs verte et rouge 4 fois, d'une durée totale de 2 secondes. Ensuite, le système revient à son état de départ.

Travail demandé :

Réaliser le code C qui correspondent au fonctionnement du système tout en passant par les étapes suivantes :

A. Les directives : **(1 point)**
   - Defines (0.5 point)

```
#define verte rb4
#define rouge rb3
#define valider rb0
#define incr rb1
#define etageA rb5
#define etageB rb6
#define etageC rb7
```

   - Macros (0.5 point)

```
#define set_high(pin)(pin=1)
#define set_low(pin)(pin=0)
#define display(value)(porta=value)
```

B. La fonction principale main : **(5 points)**

- La configuration des entrées / sorties. (1 point)

```
TRISA=0;
TRISB=0b11100011;
```

- La configuration de l'interruption. (1 point)

```
GIE=1;
INTEDG=0;
```

- L'initialisation. (1 point)

```
RB3=1;
RB4=0;
PORTA=0;
NBPER=0;
//delay_mss(2000);
```

- Le programme principal. (2 points)

```
while(1)
{
//delay_mss(2000);
if(     RB1==1)
{
        NBPER++;
PORTA=NBPER;
delay_mss(500);
if(NBPER==4)
{       NBPER=0;
        PORTA=4;
        RB4=1;
        RB3=0;
        RBIE=1;
        delay_mss(500);}
            }
```

C. La fonction d'interruption **(4 points)**

- La sélection de l'étage (2 points)

```
if(RBIF==1)
{if(RB5==1){
PORTA=0X0A;
NBPER=0;
INTE=1;}
if(RB6==1){
PORTA=0X0B;
NBPER=0;
INTE=1;}
```

```
if(RB7==1){
PORTA=0X0C;
NBPER=0;
INTE=1;}
RBIF=0;
}
```

- La validation du choix (2 points)

```
if(INTF==1){
for(i=0;i<4;i++){
RB3=1;
RB4=1;
delay_ms(2000);
RB3=0;
RB4=0;
delay_ms(2000); }
INTF=0;
PORTA=0;
RB3=1;
delay_ms(2000); }
```

NB : nous disposons d'une fonction **delay_ms()** qui assure les attentes.

BON TRAVAIL