TD2_bis Microcontrôleur

Exercice 1

1. Donner la signification de chaque instruction et son mode d'adressage dans le deux cas a) et b).
   a) Movlw 0x55
      Movwf 0x40
      Movwf 0x41
      Movwf 0x42
      Movwf 0x43

   b) Movlw 0x55
      Lfsr 0x40
      Movwf INDF0
      Incf FSR0L,f
      Movwf INDF0
      Incf FSR0L,f
      Movwf INDF0
      Incf FSR0L,f
      Movwf INDF0
2. Que réalisent les séquences a) et b) ?

Exercice 2

1. Quel est le rôle du registre BSR ?
2. Ecrire le fragment de code assembleur permettant de charger la valeur 0x99 dans la case mémoire d'adresse 0x202 ;
3. Ecrire le fragment de code assembleur permettant de copier la contenu de la case mémoire d'adresse 0x202 dans la case mémoire d'adresse 0x203 ;
4. Ecrire la séquence assembleur qui permet de transférer le contenu du PORTB dans WREG.
5. Quel est le registre qu'on utilise comme pointeur de données en adressage indirect ?
6. Donner les capacités des mémoires RAM et Flash du microcontrôleur PIC18F4520.
7. Quel est l'avantage de mettre des données dans la mémoire Flash ?
8. Quels sont les registres à utiliser pour accéder à une donnée chargée dans la mémoire Flash ?
9. Donner les étapes nécessaires pour lire une information située dans la mémoire Flash à l'adresse 0x2000.
10. Dans quel mémoire (RAM ou Flash) doit-on sauvegarder les données suivantes :
    a) Température
    b) Date et heure
    c) Version du programme
11. Ecrire le code assembleur qui calcule l'expression : k = i –j . les variables i, j et k ont pour adresses 0x400, 0x401 et 0x402.
12. Donner la valeur de la case mémoire affectée après exécution de chaque instruction. En prend pour états initiales :
    WREG = 0x3D ; BSR = 0x00 ;

Valeurs initiales		Instructions
Adresse	**Contenu**	subwf 0x5C,f
0x5A	0x3B	movlw 0x5C
0x5B	0xA2	addwf 0x5C,f
0x5C	0xF4	addwf 0x5A,f
		incf 0x5B,w

13. Traduire en assembleur le fragment du code suivant :

```
Char i,j ;
If(i>j)
   i = j ;
```

Exercice 3

On considère des nombres signés codés sur 8 bits, le 8ème bit est un bit de signe.

Donner le résultat ainsi que l'état des indicateurs N, Z, et C dans les cas suivants :

0x01 + 0xFF ; 0x80 + 0x7F ; 0xF2 – 0x20

Exercice 4

Soit le programme suivant :

```
signed char MyNum[]={+1, -1, +2, -2, +3, -3, +4, -4} ;
char z ;
void main()
{     TRISD =0 ;
      for(z =0 ; z < 8 ; z++)
            PORTD = MyNum[z] ;
      while(1) ;
}
```

Donner dans un tableau les valeurs (en binaire) à la sortie du PORTD en fonction de z.

Exercice 5

Donner les valeurs des ports B, C et D après exécution.

```
void main()
{     TRISB =0 ; TRISC = 0 ; TRISD = 0 ;
      PORTB = 0x35 & 0x0F ;   PORTC = 0x04 | 0x68 ;   PORTD = 0x54 ^0x78 ;
      PORTB = ~0x55 ;         PORTC |= (0x9A>> 3) ;   PORTD &= 0x06 ;
      While(1) ;
}
```

Exercice 6

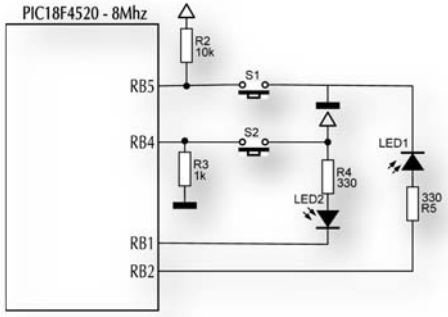
```
char MyDATA[] = {0X25, 0x62, 0x3F, 0x52} ;
char sum =0 ;
char z, Checksum ;
void main()
{       TRISC = 0 ;
        For(z = 0 ; z < 4 ; z++)
        {       PORTC = MyDATA[z] ;
                Sum = sum +MyDATA[z] ;
        }
        Checksum = ~sum+1 ; PORTC = Checksum ;
        While(1) ;
}
```

Donner les valeurs des variables « sum » et « Checksum » après exécution du programme.

A quoi sert la variable Checksum ?

Exercice 7

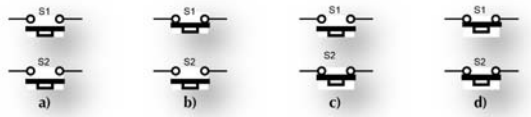
On donne le schéma du montage suivant :



1. Comment configurer le PORTB, sachant les bits RB7, RB6, RB3 et RB0 seront considérés comme des entrées.
2. Dans le programme C, on trouve l'instruction suivante : **`BPs = PORTB & 0x30`**.

Quel doit être le type de la variable **BPs** ?

3. Donner pour les cas suivants les valeurs possibles de **BPs** en Hexadécimal.



4. En utilisant l'instruction à choix multiples « switch … case », écrire le programme C permettant d'allumer une LED lorsque le bouton correspondant est enfoncé.

Exercice 8

1. En utilisant les masques, écrire l'instruction qui permet :
   - De mettre à 1 les bits RB3 et RB5 du PORTB
   - De mettre à 0 les bits RB1 et TB7 du PORTB
   - D'inverser le bit RB6 du PORTB
   - De mettre le bit RB2 à 1 et RB6 à 0.
2. Ecrire les expressions booléennes sur les tests suivants : Expression vraie si,
   - RB2 à 1 et RB0 à 0
   - RB3 à 1 ou RB1 à 0
3. Donner les valeurs possibles de l'expression suivante :

```
((p1 & 0x80) >>3)^(p1 & 0x10)
```

Exercice 9

1. Traduire en langage C l'algorithme suivant :

```
Début
      PORTC ← 0x81 ;
      Répéter toujours
            PORTC ← (PORTC << 1)|(PORTC >> 7));
            Attente_1000ms;
      Fin répéter
fin
```

2. Donner dans un tableau, les états possibles du PORTC

RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0
1	0	0	0	0	0	0	1
			⋮				

3. Quelle est la fonction réalisée par l'instruction :

```
PORTC ← (PORTC << 1)|(PORTC >> 7));
```

Exercice 10

1. Ecrire un programme permettant de faire clignoter la LED connectée à la broche RC0 au rythme de 500ms allumée; 500ms éteinte.

On utilisera la fonction prédéfinie delay_ms().

2. Modifier le programme pour faire clignoter toutes les LEDs du PORTC en même temps au rythme de 500ms allumée; 500ms éteinte.

3. Modifier le programme pour faire clignoter les 8 leds par paquet de 4 au rythme de 500ms allumée ; 500ms éteinte.
4. Modifier le programme pour faire le décalage à gauche puis à droite. Vous pouvez utiliser l'algorithme suivant :

```
Debut
PORTC ← 0X01
  Tant que (1) Faire
        Pour i allant de 0 à 6 Faire
              PORTC ← PORTC << 1
              Attente 500ms
        FinFaire
        Pour i allant de 0 à 6 Faire
              PORTC ← PORTC >> 1
              Attente 500ms
        FinFaire
  Fin Tant que
fin
```

Exercice 11

Ecrire un programme C permettant d'allumer la Led (RC2) selon le tableau suivant :

RB1	RB0	Led
0	0	Led ← 0
0	1	Led ← 1
1	0	Led clignote à une fréq. De 1 Hz
1	1	Led clignote à une fréq. De 0,5 Hz

Ali Hmidene
ISET Sousse 2014