

# TD1- Microcontrôleur

## Exercice 1

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 2

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 3

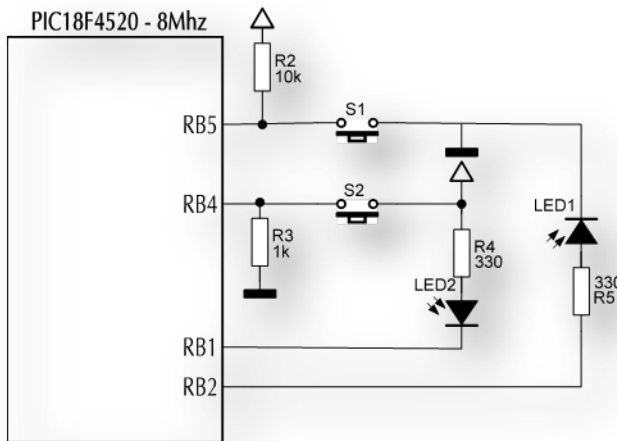
```
char MyDATA[] = {0x25, 0x62, 0x3F, 0x52} ;
char sum =0 ; char z, Checksum ;
void main()
{
    TRISC = 0 ;
    For(z = 0 ; z < 4 ; z++)
    {
        PORTC = MyDATA[z] ;
        Sum = sum +MyDATA[z] ;
    }
    Checksum = ~sum+1 ; PORTC = Checksum ;
    While(1) ;
}
```

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

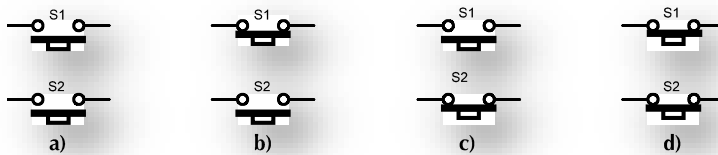
A quoi sert la variable Checksum ?

#### Exercice 4

Etant donné le schéma 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 5

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 RB7 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

#### Exercice 6

1. Traduire en langage C l'algorithme suivant :

Début

PORTC  $\leftarrow$  0x81 ;

Répéter toujours

PORTC  $\leftarrow$  (PORTC  $\ll$  1)|(PORTC  $\gg$  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 :

$\text{PORTC} \leftarrow (\text{PORTC} \ll 1) | (\text{PORTC} \gg 7);$

### Exercice 7

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

$\text{PORTC} \leftarrow 0x01$

Tant que (1) Faire

Pour i allant de 0 à 6 Faire

$\text{PORTC} \leftarrow \text{PORTC} \ll 1$

Attente 500ms

FinFaire

Pour i allant de 0 à 6 Faire

$\text{PORTC} \leftarrow \text{PORTC} \gg 1$

Attente 500ms

FinFaire

Fin Tant que

fin

### Exercice 8

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

| RB1 | RB0 | Led                                |
|-----|-----|------------------------------------|
| 0   | 0   | Led $\leftarrow$ 0                 |
| 0   | 1   | Led $\leftarrow$ 1                 |
| 1   | 0   | Led clignote à une fréq. De 1 Hz   |
| 1   | 1   | Led clignote à une fréq. De 0,5 Hz |

## CORRIGE DU TD LES PORTS E/S

### Exercice 1

Z = 0      PORTD = 0b00000001  
Z = 1      PORTD = 0b11111111  
Z = 2      PORTD = 0b00000010  
Z = 3      PORTD = 0b11111110  
Z = 4      PORTD = 0b00000011  
Z = 5      PORTD = 0b11111101  
Z = 6      PORTD = 0b00000100  
Z = 7      PORTD = 0b11111100

### Exercice 2

PORTB = 0x05      ; PORTC = 0x6C      ;      PORTD = 0x2C  
PORTB = 0xAA      ; PORTC = 0x7F      ;      PORTD = 0x04

### Exercice 3

Sum = 0x25 + 0x62 + 0x3F + 0x52 = 0x18 (C = 1)

Chechsum = 0xE8

Cette variable est utilisée pour la vérification d'erreur dans le cas de transmission des données.

### Exercice 4

1. TRISB = 0b11111001 ;
2. BPs est codée sur 8 bits, c'est le type « char »
3.
  - a) BPS = 0x20      b) BPS = 0x00      c) BPS = 0x30      d) BPS = 0x10

4.

```
char BPs ;
void main()
{   TRISB = 0xF9 ;
    while(1)
    {   BPs = PORTB & 0x30 ;
        switch(BPs)
        {   case 0x20 : {RB1 = 1 ; RB2 = 0 ;}break ;
            case 0x00 : {RB1 = 1 ; RB2 = 1 ;}break ;
            case 0x30 : {RB1 = 0 ; RB2 = 0 ;}break ;
            case 0x01 : {RB1 = 0 ; RB2 = 1 ;}break ;
        }
    }
}
```

### Exercice 5

1.
  - De mettre à 1 les bits RB3 et RB5 du PORTB ; PORTB = PORTB | 0x28 ;
  - De mettre à 0 les bits RB1 et TB7 du PORTB ; PORTB = PORTB | 0x7D ;

- D'inverser le bit RB6 du PORTB;  $PORTB = PORTB \wedge 0x40$ ;
  - De mettre le bit RB2 à 1 et RB6 à 0. ;  $PORTB = (PORTB \& 0xBF) | 0x40$ ;
2. Expression vraie si,
- RB2 à 1 et RB0 à 0 ;  $if((RB2 == 1) \&\& (RB0 == 0))$
  - RB3 à 1 ou RB1 à 0 ;  $if((RB3 == 1) || (RB1 == 0))$

### Exercice 6

1. Traduire en langage C l'algorithme suivant :

```
void main()
```

```
{
    PORTC = 0x81;
    while(1)
    {
        PORTC = (PORTC << 1) | (PORTC >> 7);
        delay_ms(1000);
    }
}
```

2. les états possibles du PORTC

| RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 1   | 0   | 0   | 0   | 0   | 0   | 0   | 1   |
| 0   | 0   | 0   | 0   | 0   | 0   | 1   | 1   |
| 0   | 0   | 0   | 0   | 0   | 1   | 1   | 0   |
| 0   | 0   | 0   | 0   | 1   | 1   | 0   | 0   |
| 0   | 0   | 0   | 1   | 1   | 0   | 0   | 0   |
| 0   | 0   | 1   | 1   | 0   | 0   | 0   | 0   |
| 0   | 1   | 1   | 0   | 0   | 0   | 0   | 0   |
| 1   | 1   | 0   | 0   | 0   | 0   | 0   | 0   |
| 1   | 0   | 0   | 0   | 0   | 0   | 0   | 1   |

3. la fonction réalisée par l'instruction : Rotation du PORTC

### Exercice 7

- 1.

```
void main()
```

```
{
    TRISC = 0xFE;
    while(1)
    {
        RC2 = !RC2;
        delay_ms(500);
    }
}
```

- 2.

```
void main()
{
    TRISC = 0x00; PORTC = 0;
    while(1)
    {
        PORTC = ~PORTC;
        delay_ms(500);
    }
}
```

```

    }
}

3.
void main()
{
    TRISC = 0x00; PORTC = 0x0F;
    while(1)
    {
        PORTC = ~PORTC;
        delay_ms(500);
    }
}

4. void main()
{
    char i;
    TRISC = 0x00; PORTC = 0x01;
    while(1)
    {
        for(i = 0; i < 7; i++){
            PORTC = PORTC << 1;
            delay_ms(500);
        }
        for(i = 0; i < 7; i++){
            PORTC = PORTC >> 1;
            delay_ms(500);
        }
    }
}

```

### **Exercise 8**

```

void main()
{
    TRISC = 0x00; PORTC = 0x01;
    while(1)
    {
        if((RB1 == 0) && (RB0 == 0))
            RC2 = 0;
        if((RB1 == 0) && (RB0 == 1))
            RC2 = 1;
        if((RB1 == 1) && (RB0 == 0)){
            RC2 = !RC2;
            delay_ms(1000);
        }
        if((RB1 == 1) && (RB0 == 1)){
            RC2 = !RC2;
            delay_ms(2000);
        }
    }
}

```