

Solution

1. Le pic utilisé est le 16F877 , et on doit le déclarer par : `list p=16F877`
2. On définit les noms des variables réservées au pic par : `#include <p16f877.inc>`
3. Pour configurer les modes de fonctionnement du pic, on doit répondre à des questions telles que :

Doit-on utiliser un Code Protection ? NON : `_CP_OFF`

A la mise sous tension du μC , doit-on lancer une temporisation après laquelle est effectué un RESET interne ? OUI : `_PWRTE_ON`

Est-ce qu'on a besoin d'utiliser le Watchdog ? NON : `_WDT_OFF`

Quel est le type d'horloge utilisé? Horloge RC : `_RC_OSC`

On obtient, alors: `_config_CP_OFF & _PWRTE_ON & WDT_OFF & _RC_OSC`

4. Les registres PORTC et PORTB se trouvent dans la banque 0 (bank 0) de la mémoire de données, et les registres TRISC et TRISB se trouvent dans la banque 1 (bank 1) de la mémoire de données.
5. Les interrupteurs sont des entrées et les leds sont des sorties, donc RB0, RB1, RB2 et RB3 doivent être configurés en entrée et RC0, RC1, RC2 et RC3 doivent être configurés en sortie.

Pour configurer RB0, RB1, RB2 et RB3 en entrée, il faut mettre des uns(1) dans TRISB

TRISB

x	x	x	x	1	1	1	1
---	---	---	---	---	---	---	---

Pour configurer RC0, RC1, RC2 et RC3 en sortie, il faut mettre des zéros (0) dans TRISC

TRISC

x	x	x	x	0	0	0	0
---	---	---	---	---	---	---	---

Puisque les autres lignes du PORTB et du PORTC ne sont pas utilisés on va prendre :

TRISB

1	1	1	1	1	1	1	1
---	---	---	---	---	---	---	---

TRISC

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

6. Pour accéder aux registres TRISA et TRISB ; dans le cas du 16F877 , il faut mettre le les bits RP01 et RP0 du registre STATUS respectivement à 0 et à 1

```
bcf      STATUS , RP1
bsf      STATUS ,RP0 ; sélectionner bank 1
```

et pour accéder aux registres PORTA et PORTB lorsqu'on est dans la banque1, il suffit de mettre le bit RP0 à 0.

```
bcf      STATUS ,RP0 ; sélectionner bank 0
```

7. Instructions de configuration des PORTC et PORTB

```
bcf      STATUS , RP1
bsf      STATUS ,RP0 ; sélectionner bank 1

movlw    0xff          ; configure les lignes du PORTB en entrée
movwf    TRISB

clrf     TRISC          ; configure les lignes du PORTC en sortie

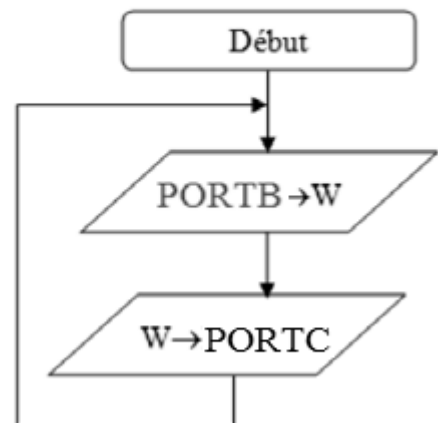
bcf      STATUS ,RP0 ; sélectionner bank 0
```

8. Algorithme et organigramme du programme principal

Algorithme

- 1) lire le PORTB : mettre PORTB dans W
- 2) transférer W sur le PORTC
- 3) recommencer (aller à 1)

Organigramme



9. Programme principal

```
boucle  movf    PORTB, W      ; met PORTB dans W
        movwf   PORTC        ; met W dans PORTC
        goto    boucle
```

10.

```
List p=16F877
#include <16F877.inc>

__CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC

org    0x00                ; début du programme

bcf     STATUS, RP1        ; sélectionner bank 1
bsf     STATUS, RP0

movlw   0xff               ; configure les lignes du PORTB
movwf   TRISB              ; PORTB en entrée

clrf    TRISC              ; configure les lignes du PORTC

boucle  movf   PORTB, W     ; met PORTB dans W
        movwf  PORTC        ; met W dans PORTC
        goto   boucle
end
```