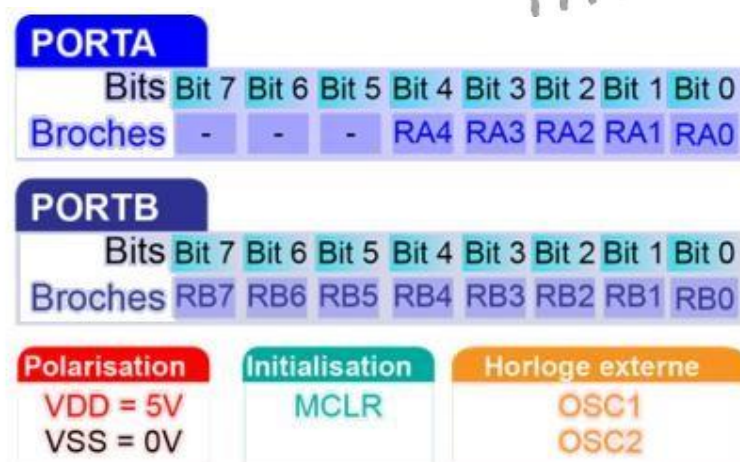
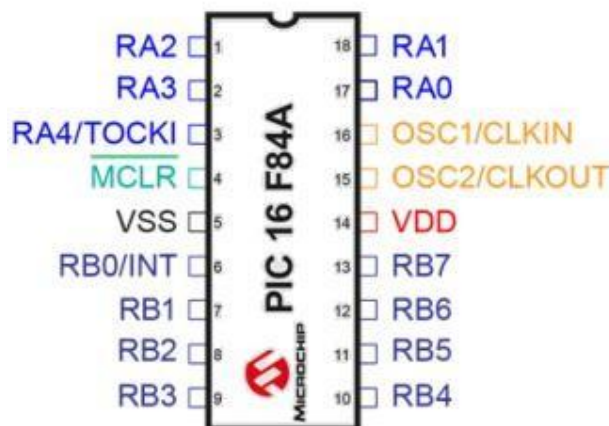


Thème 6 : LES MICROCONTRÔLEURS

Exercice 1 :

La figure ci-dessous présente le microcontrôleur du type PIC16F84A à partir :

- du schéma de brochage ;
- de la répartition des broches par port et par fonction.



1- Cocher la bonne réponse :

- Le pic16F84A est un circuit intégré ☐ Combinatoire ☐ Séquentiel ☐ Programmable
- Fabriquant ☐ MICROCHIP ☐ INTEL ☐ MOTOROLA
- Famille ☐ Base line ☐ Mid-range ☐ High-end
- Type de sa mémoire programme ☐ Flash ☐ ROM ☐ RAM

2- Compléter le tableau ci-dessous par la désignation et numéro de la broche :

FONCTION	DESIGNATION	N° broches
Entrées d'horloge externe		
Alimentation (2V à 5,5V continue)		
Masse (0V)		
Initialisation du microcontrôleur		

3- Déterminer les numéros des broches représentant les entrées/sorties du PIC :

-	-	-	RA4	RA3	RA2	RA1	RA0
							17
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
				9			

4- Compléter le tableau ci-dessous :

	Nombre de ports	Nombre d'entrées/sorties	Nombre de broches
PIC16F84A			

5- Quel est le rôle des registres TRISA et TRISB ?

.....

6- A quel niveau logique doit-on porter la broche MCLR pour initialiser le microcontrôleur.

.....

7- Il est possible d'accéder à un seul bit d'un registre en MicroC Pro. Barrer ce qui est faux.

TRISB.B0=1 ; TRISB.0=0 ; TRISB.RB0=0 ; TRISB.F0=0 ; trisB0_bit=0 ;
 RB0_bit=0 ; PORTB.B0=1 ; PORTB.RB0=0 ; PORTB.F0=1 ; PortB.0=1 ;

8- Barrer la syntaxe qui n'est pas acceptée par le compilateur MikroC Pro.

sbit A at RB0_bit ; sbit A at PORTB.B0 ; sbit A at PORTB.0 ;
 sbit A at PORTB.F0 ; sbit A at PORTB.RB0 ; sbit A : at PORTB.b0 ;
 bit at RB0_bit ;

Exercice 2 :

La figure 1 représente une architecture simplifiée d'un microcontrôleur du type PIC16F84.

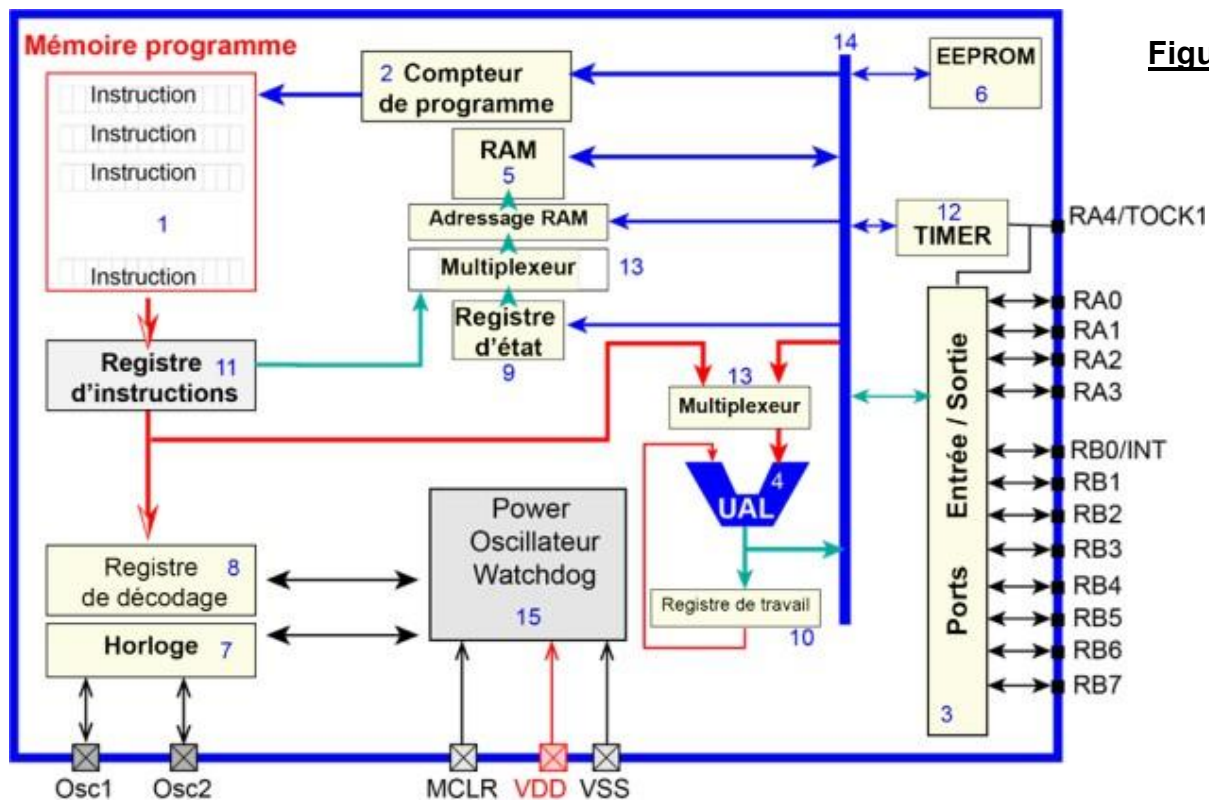


Figure 1

Compléter le tableau suivant par le nom de chaque élément évoqué dans cette architecture.

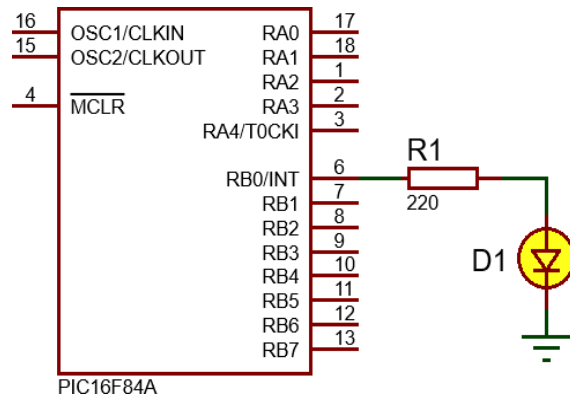
Nom	Fonction
	Stocker le programme.
	Choisir l'instruction à exécuter (pointeur).
	Pour la communication des informations.
	Exécuter les opérations de calcul.
	Mémoire vive : enregistrement temporaire des données.
	Mémoire morte programmable et effaçable électriquement.
	Rythmer le cycle d'exécution du microcontrôleur.
	Décoder une instruction.
	Enregistrer le dernier état du calculateur.
	Enregistrer le dernier résultat de calcul de l'UAL
	Stocker une instruction à exécuter.
	Fournir ou recevoir le résultat de comptage.
	Suite à une autorisation, il cède le passage à une information parmi plusieurs présentes en entrée.
	Initialiser le microcontrôleur.

Exercice 3 :

On désire faire clignoter une Led « D1 » branchée au portB (par exemple bit 0 du PORTB), du microcontrôleur PIC avec une période de 2 secondes (1 seconde allumée et une seconde éteinte). Période : $T=2s$, fréquence : $f = 0,5Hz$

1- A partir du schéma de simulation ci-contre, compléter les 2 tableaux suivants par :

- Entrée, sortie, non utilisée : pour le PortB.
- Pour le registre TrisB, une broche non utilisée sera considérée comme entrée.(sous forme d'instructions)



	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
PortB								

	Décimal	Binaire	Octal	Hexadécimal
TRISB				

2- Compléter le programme en mikroC Pro conformément à l'algorithme donné :

Algorithme	Programme
Mot clé du début du programme	
Début du programme	
Configurer le port B	
Initialiser les sorties	
TANT QUE (vraie)	
DÉBUT	
portB ← 1	
Délai (1s)	
portB ← 0	
Délai (1s)	
FIN TANT QUE	
FIN PROGRAMME	

Autres méthodes :

• Utilisation du caractère Négation logique !	• Utilisation de la variable sbit
<pre>void main() { TRISB=0; PORTB=0; while (1) { =; delay_ms(1000); } }</pre>	<pre>.....; void main() { TRISB=0; PORTB=0; while (1) { =; delay_ms(1000); } }</pre>

Exercice 4 :

Ecrire un programme en MikroC permettant de clignoter tout le PORTB avec une période 2s

Solution 1	Solution 2	Schéma de simulation
<pre>void main() { TRISB=0; PORTB=0; while (1) { PORTB=; delay_ms(1000); PORTB=; delay_ms(1000); } }</pre>	<pre>void main() { TRISB=0; PORTB=0; while (1) { PORTB=...; delay_ms(500); } }</pre> <i>Utilisation de l'opérateur tilde « ~ »</i>	

Remarque :

Si on utilise l'instruction « **PORTB= ! PORTB ;** » on aura seulement la broche **RB0** qui clignote puisque l'opérateur **!** (**NOT**) utilisé réalise la fonction **NON** juste pour un seul bit, elle nous renvoie une valeur **booléenne**.

Exercice 5 :

1- Compléter le schéma du circuit (figure 2) en appliquant les consignes suivantes :

- ✗ brancher L1 sur RA3 ;
- ✗ brancher S1 sur RA2 ;
- ✗ polariser le microcontrôleur ;
- ✗ brancher le quartz sur les entrées de l'oscillateur externe.

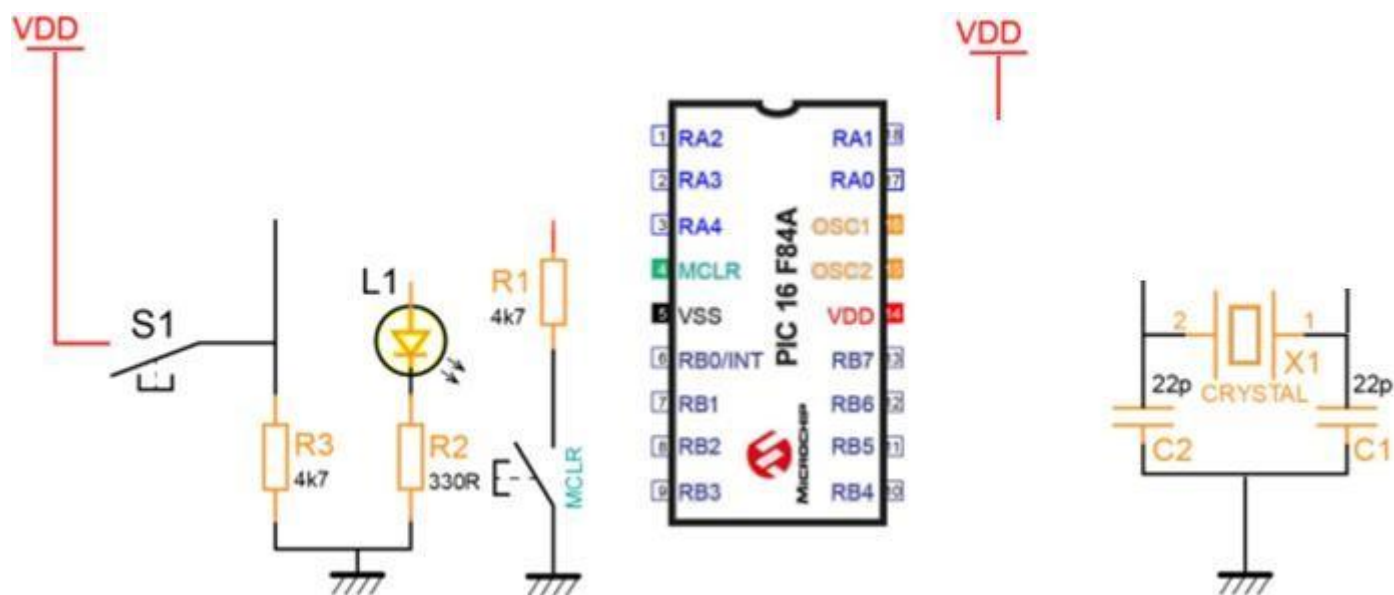


Figure 2

2- La diode L1 s'allume quand le bouton-poussoir est actionné.

- ✗ Etablir un algorithme ;
- ✗ Traduire cet algorithme en langage mikroC.

Structure	Algorithme	Programme
Déclaration	S1 type bit → RA2	
Début Programme	Mot clé du début du programme Début du programme	
Orientation des ports et initialisation des variables de sortie	Configurer le port A	
	Initialiser L1 à 0	
Corps du programme	Boucle TANT QUE (vraie)	
	DEBUT	
	Si (S1=1) Alors faire L1 ← SINON FAIRE L1 ← Fin TANT QUE	
FIN PROGRAMME	Fin programme	

Exercice 6 :

On désire commander une **LED** en utilisant le bouton poussoir BP : une impulsion sur le bouton poussoir BP allume la **LED** durant 3s puis elle s'éteint.

Programme

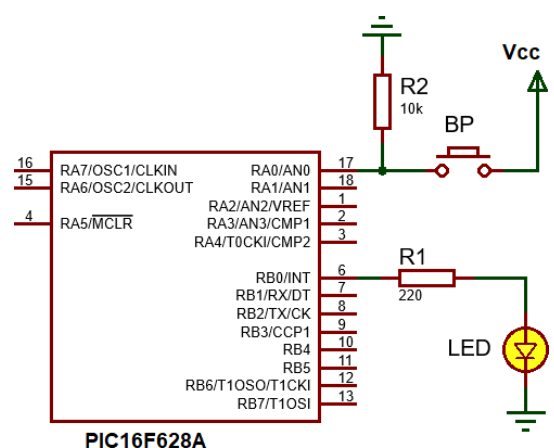
```

.....; //déclarer une variable
      LED du type bit branchée sur RB0.
.....; //déclarer une variable
      BP du type bit branchée sur RA0.

void main() {
    .....; //configuration de la broche RA0.
    .....; //configuration de la broche RB0.
    CMCON= 0x07; // PortA numérique
    while (1) // début de la boucle
    {
        if (BP.....) //ou .....
        {
            .....;
            .....;
        }
        else LED=.....;
    } }

```

Schéma de simulation



Exercice 7 :

On désire commander une LED D1 en utilisant deux boutons poussoirs BP1 et BP2. Une impulsion sur le bouton poussoir BP1 allume la LED D1 et une impulsion sur le bouton poussoir BP2 éteint la LED D1.

Programme

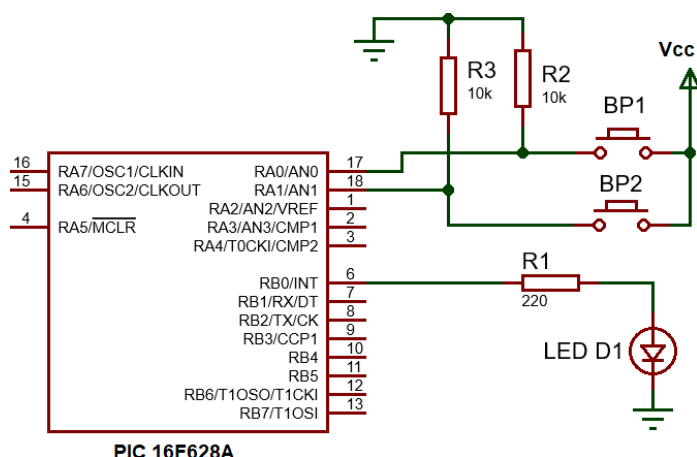
```

sbit BP1 at RA0_bit;
sbit BP2 at RA1_bit;
sbit LED_D1 at RB0_bit;

void main() {
    trisa= .....; // en décimal
    trisb= .....; // en décimal
    portb=0; .....;
    while (1) {
        if (.....; // or if (BP1)
        if (.....; // or if (BP2)
    } }

```

Schéma de simulation

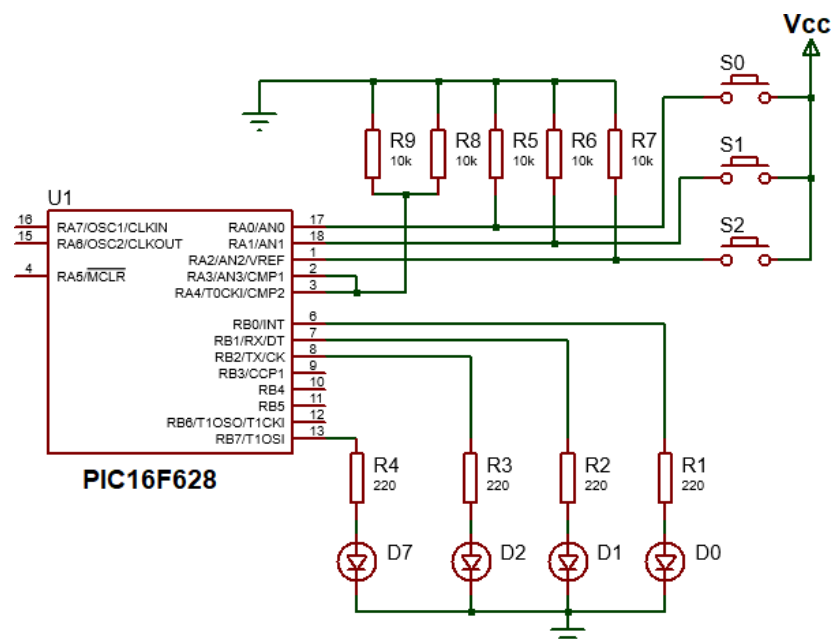


Exercice 8 :

Un circuit à base de 16F628A, comporte 3 boutons poussoirs (S0, S1 et S2 reliés respectivement à RA0, RA1 et RA2) et trois LED (D0, D1 et D2 reliées respectivement à RB0, RB1 et RB2).

L'action sur l'un des boutons poussoirs entraîne l'allumage de la diode de même indice et l'extinction des autres. Si aucun des boutons n'est actionné, une LED (D7) branchée sur la broche RB7 s'allume.

Ecrire un programme mikroC qui répond à ce cahier de charges.



```

sbit D0 at RB0_bit;
sbit D1 at RB1_bit;
sbit D2 at RB2_bit;
sbit D7 at RB7_bit;

void main() {

```

Poids								
PortA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
Entrée	-	-	-	-	-	S2	S1	S0

```

    Trisa=0x .....; // configuration du portA en entrée
    Trisb=0b.....; // configuration du portB en sortie
    PORTB=0; .....; // initialisation PortB et PortA numérique
    while (1) { // debut de la boucle
        switch (.....) {
            case 1: D0=.....; D1=.....; D2=.....; D7=.....; .....;
            case ....: D0=.....; D1=.....; D2=.....; D7=.....; .....;
            case ....: D0=.....; D1=.....; D2=.....; D7=.....; .....;
            default: D0=.....; D1=.....; D2=.....; D7= .....;
        }
    }
}

```

```
sbit S0 at RA0_bit;
sbit S1 at RA1_bit;
sbit S2 at RA2_bit;
sbit D0 at RB0_bit;
sbit D1 at RB1_bit;
sbit D2 at RB2_bit;
sbit D7 at RB7_bit;
void main()
{
    TrisA=0xFF;
    Trisb=0;
    PORTB=0;
```

Page 7/8

TANT QUE (vraie) DEBUT K ← PORTB SELON (K) DEBUT Cas 0 : L1 ← ; L2 ← ; sauter Cas.....L1 ← ; L2 ← ; sauter Cas : L1 ← ; L2 ← ; sauter Défaut : L1 ← ; L2 ← ; sauter FIN SELON FIN TANT QUE FIN du programme	while (1) {;;;;; } }
---	---

Exercice 10 :

Les sorties d'un système combinatoire sont régies par les équations logiques ci-dessous :

$$S1 = A + B + C$$

$$S2 = B \cdot D$$

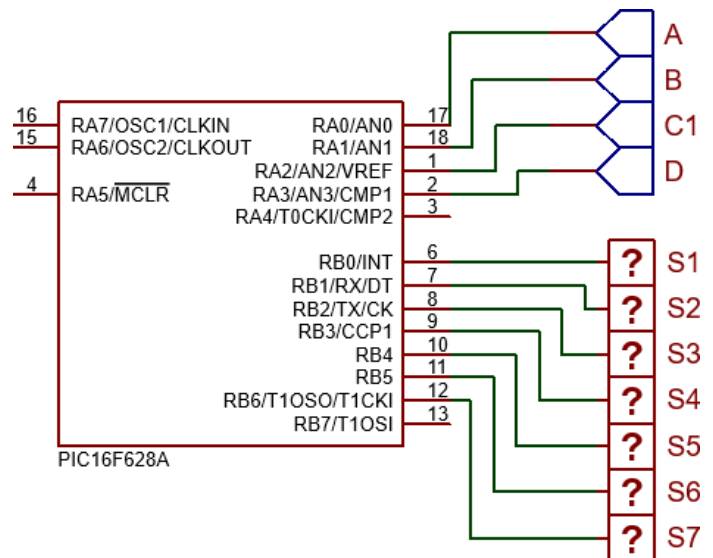
$$S3 = A \oplus B$$

$$S4 = A \cdot \overline{B} + D$$

$$S5 = A \odot B$$

$$S6 = \overline{A \cdot B}$$

$$S7 = \overline{A + B}$$



N.B : la variable « C » n'est pas acceptée par le compilateur, on la change par « C1 ».

Etablir un programme en mikroC

.....
.....

sbit C1 at RA2_bit;

sbit D at RA3_bit;

sbit S1 at RB0_bit;

sbit S2 at RB1_bit;

sbit S3 at RB2_bit;

sbit S4 at RB3_bit;

sbit S5 at RB4_bit;

sbit S6 at RB5_bit;

sbit S7 at RB6_bit;

void main() {

TrisA=0x.....;

TrisB=0x.....;

portB=0;

.....;

while (1) {

if; else S1=0;

if; else S2=0;

if; else S3=0;

if; else S4=0;

if; else S5=0;

if; else S6=0;

if; else S7=0;

} }

Autre solution:

while(1) {

S1=; S2 =; S3 =.....;

S4=; S5 =

S6=; S7 =

}