

FORMATION SUR MAQUETTE DE DÉVELOPPEMENT PIC 1 6FXXX - LS-PIC 1



Sommaire

l-	Pre	ésen	ntation et caractéristiques techniques	3	
	1-	Pré	sentation :	3	
;	2-	Car	ractéristiques techniques	3	
;	3-	Les	s modules principaux	4	
	3-1	1-	Schéma électrique de carte développement (18 broches):	4	
	3-2	2-	Schéma électrique de carte développement (28 broches : 16F876) :	4	
	3-3	3-	Schéma électrique de carte développement (40 broches : 16F877) :	5	
	3-4	1-	Interface des entrées TOR :	5	
	3-5	5-	Interface des sorties LED :	6	
	3-6	-	Interface de puissance avec relais	6	
	3-7	7-	Interface d'isolation galvanique (opto-coupleur TLP521-4)	7	
	3-8	3-	Carte LCD 1602	8	
	3-9	9-	Carte afficheur 7 seg	8	
	3-1	10-	Clavier matriciel 4 x 4	8	
	3-1	11-	Carte d'interfaçage RS232-USB (TX et RX)	9	
II-	En	viro	nnement de programmation :	10	
	1 -	ET	APES DE PROGRAMMATION :	10	
	ACT	ACTIVITÉ N° 1: DÉCOUVERTE DE L'ENVIRONNEMENT PROGRAMMATION TEXTUELLE 11			
	ACT	IVITÉ	É N° 2: COMPTEUR MODULO 10	15	
	ACTIVITÉ N° 3: GESTION D'UN AFFICHEUR LCD			17	
	ACT	ACTIVITÉ N° 4: CAN - VOLTMÈTRE			
	ACT	IVITÉ	É Ν° 5: MLI – CONVERSIO N	19	
	Х СТ І	IX/ITÉ	ÉNO SUNTEREACE HART + LARVIEW	20	

I- Présentation et caractéristiques techniques

1- Présentation :

La valise **LS-PIC1** est une maquette de développement PIC 16Fxxx permettant l'étude sur des différentes type de microcontrôleur PIC (microchip) forme PDIP :

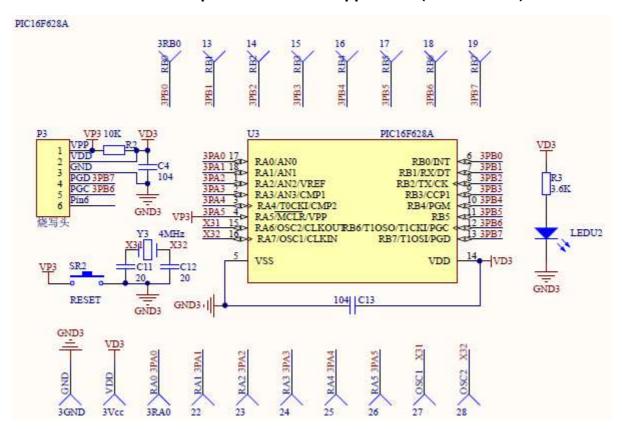
- **a-** PIC de 18 broches (16F84, 16F628, 16F88....)
- **b-** PIC de 28 broches (16F876..)
- **c-** PIC de 40 broches (16f877 ...)

Elle est principalement composée de :

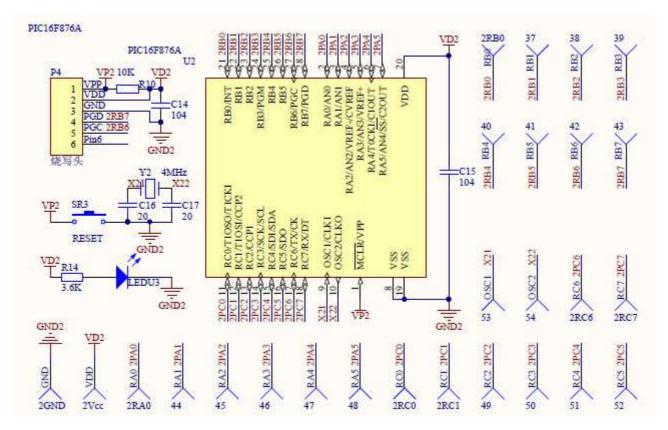
- a- Trois cartes de développement (PIC 18, 28 and 40 broches)
- b- Programmateur Microchip PicKit 2
- c- Deux Interface entrées TOR (12 interrupteurs et 16 boutons poussoirs : 12 NO et 4 NC)
- d- Interface sorties LED
- e- Interface de puissance avec relais
- **f-** Interface opto-coupleurs
- g- Carte LCD 1602
- h- Carte afficheur 7 seg
- i- Carte d'entrée analogique (utilisation de module CAN)
- j- Clavier matriciel 4 x 4
- **k-** Carte d'interfaçage RS232-USB (TX et RX)
- I- Plaque d'essai large
- 2- Caractéristiques techniques
- > Tension d'alimentation : 220ACV±5% 50Hz
- > Tension d'alimentation des circuits : +5VDC
- > Puissance de l'ensemble : 5W
- ➤ Environnement : température -10°C~+60°C Humidité<90%(25°C)

3- Les modules principaux

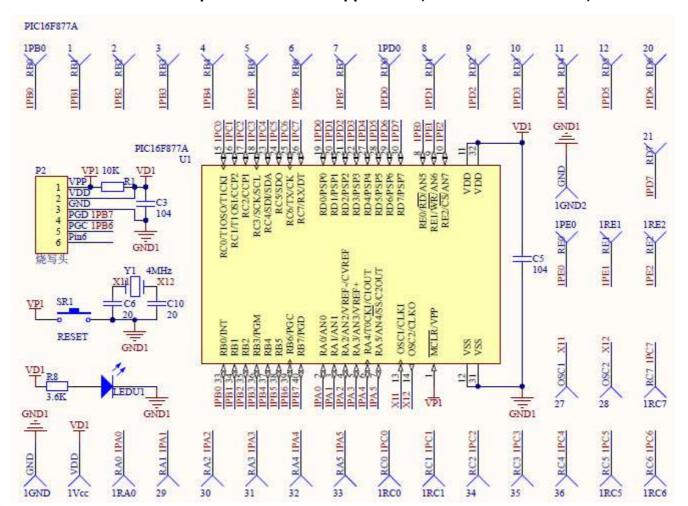
3-1- Schéma électrique de carte développement (18 broches):



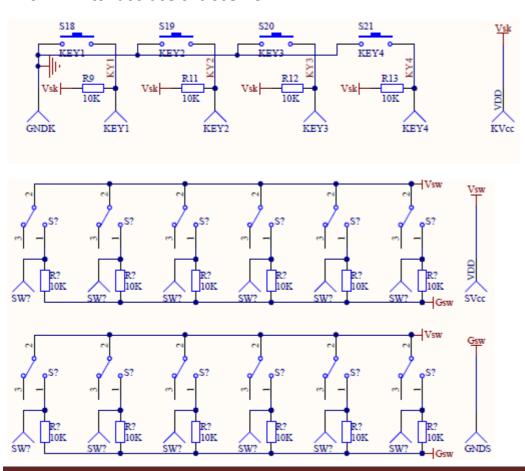
3-2- Schéma électrique de carte développement (28 broches : 16F876) :

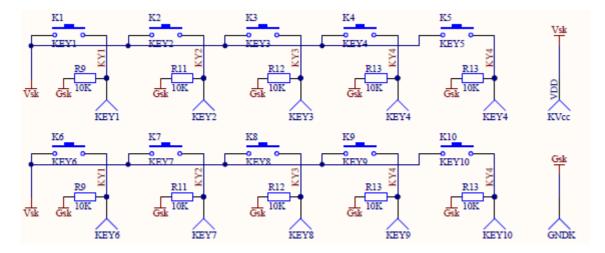


3-3- Schéma électrique de carte développement (40 broches : 16F877) :

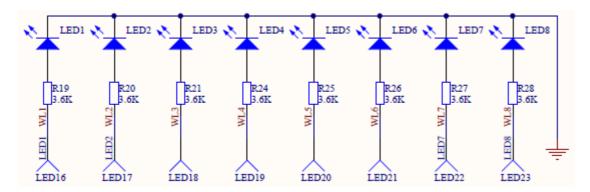


3-4- Interface des entrées TOR :

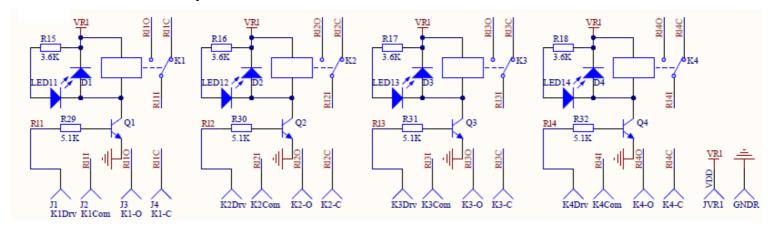




3-5- Interface des sorties LED:



3-6- Interface de puissance avec relais



Description: Relais 5v-220 10A

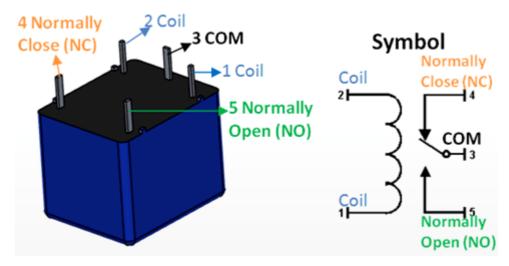
Un relais électronique est un interrupteur qui se commande avec une tension continue de faible puissance. La partie interrupteur sert à piloter des charges secteur de forte puissance (jusqu'à 10A couramment).

Fiche technique Model: SRD-5VDC-SL-C:

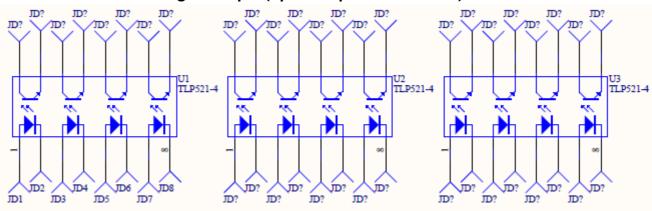
- 1. Coil voltage: DC 5V
- 2. Coil resistance: $70\Omega \sim 80\Omega$
- 3. Contact resistance: 100Ω Max
- 4. Operation time: 10msec Max
- 5. Release time: 5msec Max
- 6. Operating current: 43mA ~ 46mA
- 7. Release current: 15mA ~ 18mA

- 8. Pin: 5Pin
- 9. Insulation resistance: ≥100M (Ohm)
- 10. Between coil & contacts: AC 1500V 50HZ ~ 60HZ/min
- 11. Between contacts: AC 1000V 50HZ ~ 60HZ/min
- 12. Rated load:: 10A 250V AC / 10A 125V AC / 10A 30V DC / 10A 28V DC

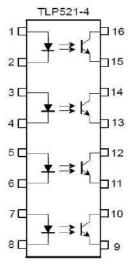
Branchement du relais



3-7- Interface d'isolation galvanique (opto-coupleur TLP521-4)



Le TLP521-4 est un isolateur traversant couplé optiquement dans un boîtier DIP 16 broches L'isolateur se composent de diodes émettrices de lumière infrarouge et de photo transistor NPN silicium. Cet isolateur est largement utilisé pour les terminaux informatiques, les contrôleurs de système industriel, la transmission du signal entre les systèmes de différents potentiels et impédances.

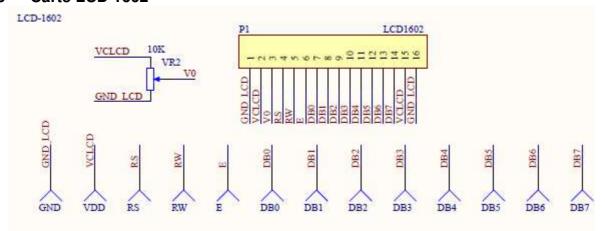


- Un ratio élevé de transfert de courant de 50%
- ➤ Haute tension d'isolement de 7,5K
- > BVCEO élevé de 55V(min)
- > Reconnu UL E91231
- Sélections personnalisée électriques disponibles
- Courant If (Forward) de 50mA
- Quad canaux

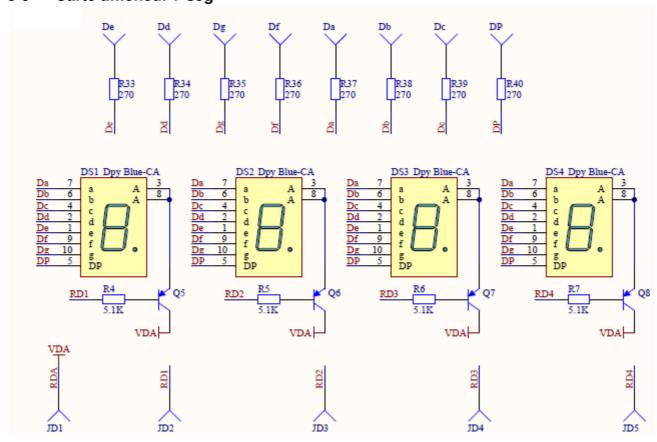
Applications

> Industrie, Test et Mesure, Traitement du Signal

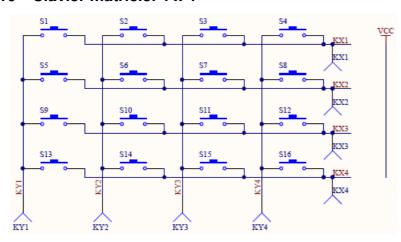
3-8- Carte LCD 1602



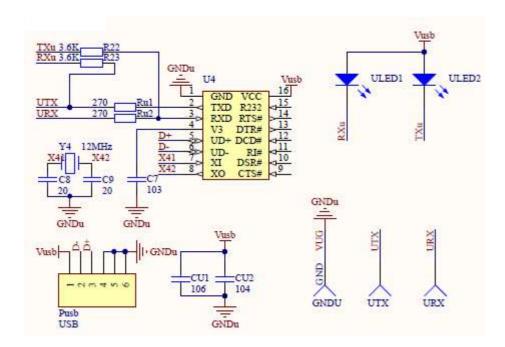
3-9- Carte afficheur 7 seg



3-10- Clavier matriciel 4 x 4

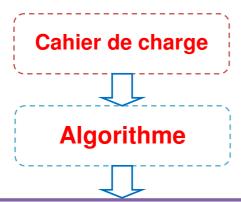


3-11- Carte d'interfaçage RS232-USB (TX et RX)



II- Environnement de programmation :

1- ETAPE DE PROGRAMMATION:



Programme

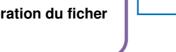
Traduction de l'algorithme en un programme en language évolué (Pascal) En utilisant compilateur MIKRPASCAL





Compilation

Transformation du programme en language machine et génération du ficher (.hex)



Simulation

Simuler le fonctionnement sur PC à l'aide d'un logiciel de simulation (ISIS..)





Chargement

Charger le ficher (.hex) dans la mémoire de microcontrôleur en utilisant logiciel PICkit 2



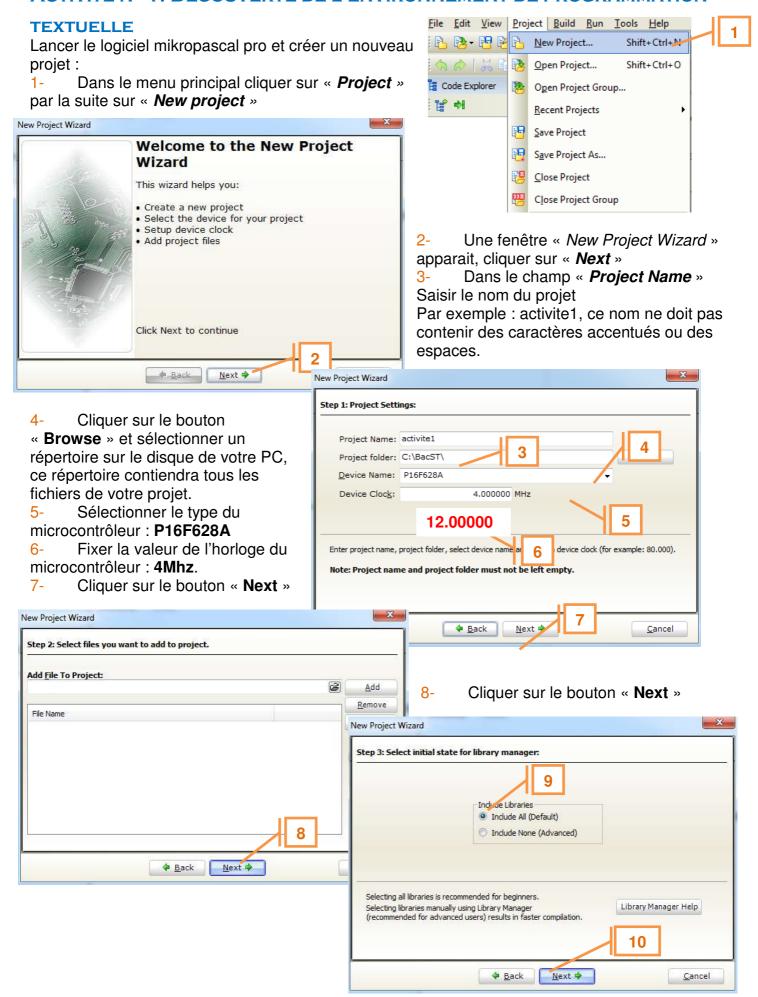


Mise en oeuvre

Mise en marche et exploitation du système à base de microcontrôleur



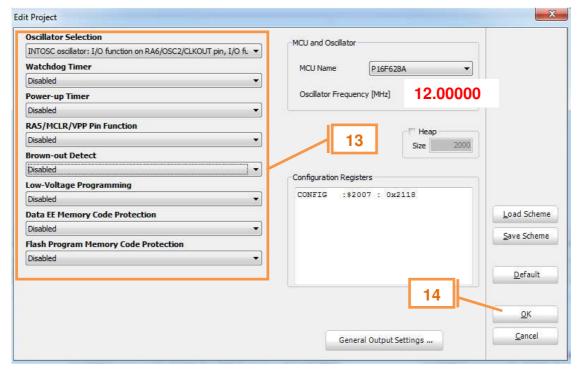
ACTIVITÉ N° 1: DÉCOUVERTE DE L'ENVIRONNEMENT DE PROGRAMMATION



- 9- Cocher la case « **Include All** » cette option permet d'utiliser les fonctions prédéfinies de Mikropascal
- 10- Cliquer sur le bouton « Next »



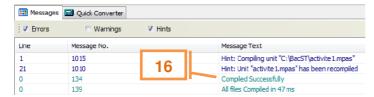
- 11- Cocher la case « *Open Edit Project window to set Configuration bits* »
 - 12- Cliquer sur le bouton « Finish »



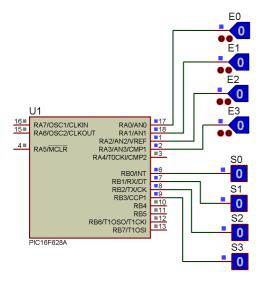
- 13- Assurer les réglages des bits de configurations comme indiqué dans la figure précédente
- 14- Cliquer sur le bouton « OK »

```
Start Page 🖾 📄 activite 1.mpas 🔀
            program activite1;
                                                      15
            Var
            E0 : sbit at RAO_bit:
            E1 : sbit at RA1_bit;
            E2 : sbit at RA2_bit;
            E3 : sbit at RA3_bit:
            SO : sbit at RBO_bit;
            S1 : sbit at RB1_bit;
           S2 : sbit at RB2_bit;
S3 : sbit at RB3_bit;
          pbegin
               TRISA:-$FF; // port A entrées
TRISB:=$F0; // RBO à RB3 sorties
CMCON:=$07; // désactivation du comparateur
               while true do
                 begin
                 S0:= E0 XOR E1;
S1:- E1 XOR E2;
S2:= E2 XOR E3;
                  S3:= E3;
                  end;
```

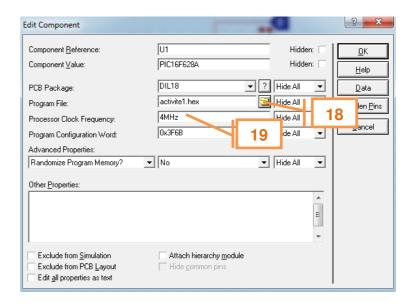
- 15- Saisir le programme ci-contre
- 16- Compiler le projet et vérifiez que la compilation a réussi : message « *Compiled Successfuly* » dans l'onglet messages.



17- Lancez le logiciel « ISIS » et saisissez le schéma de simulation.



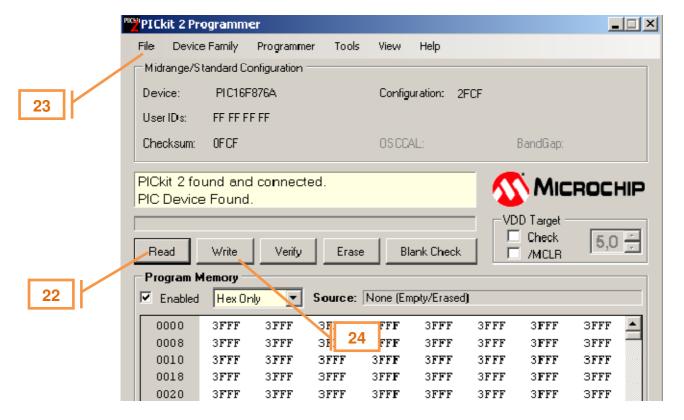
- 18- Dans la fenêtre « Edit Component » indiquer le nom et le chemin du fichier .hex
 - 19- Régler l'horloge à 12MHz



20-Bancher le programmateur PICKIT 2 au PC via le câble USB et connecteur de programmation de la valise

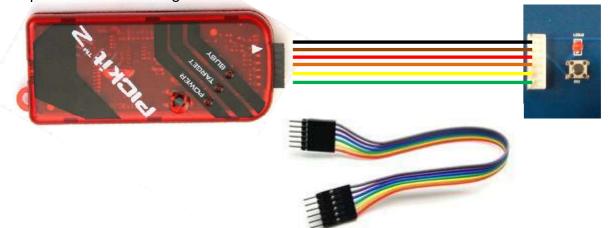


- 21- Lancer le logiciel PICkit 2.
- 22-Cliquer sur l'icône de Read pour détecter le type du microcontrôleur.
- 23- Dans le menu principal cliquer sur : File... import ... fichier.hex
- 24- Programmer le microcontrôleur PIC16F876A



Remarque : le câble ICSP doit être connecté au PICkit2 et connecteur de programmation de microcontrôleur de cette manière :

Le fil noir au indicateur de coté PICkit2 de l'autre de partie de connecteur de programmation prés de la diode rouge

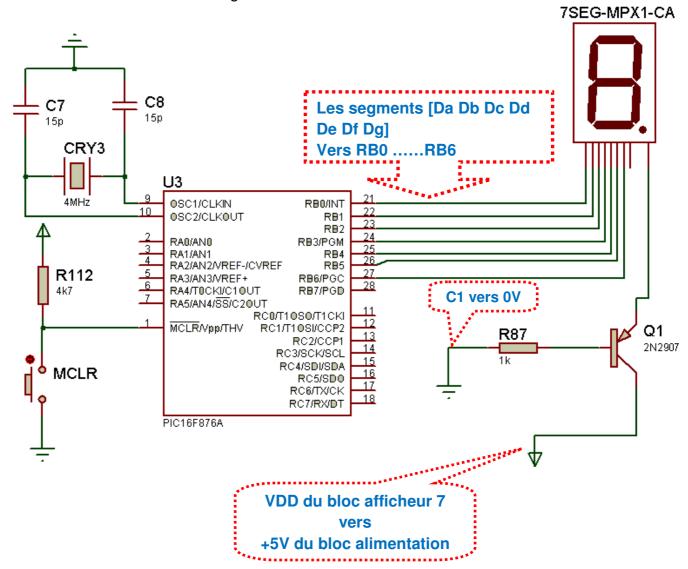


Attention il ne faut surtout pas avoir à la fois une alimentation par PICkit 2 et une alimentation externe : RISQUE DE COURT CIRCUIT ! Et

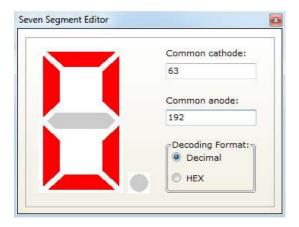
DESTRUCTION de PICKIT !

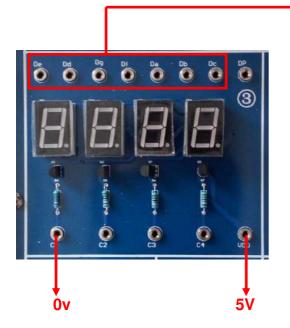
ACTIVITÉ N° 2: COMPTEUR MODULO 10

On souhaite réaliser un compteur modulo 10 en utilisant la maquette de développement PIC16FXXX conformément au montage suivant :



- 1. Reliez C1 du bloc afficheur 7 segments à 0V du bloc alimentation
- 2. Reliez VDD du bloc afficheur 7 segments à 5V du bloc alimentation
- Reliez les entrées des segments [Da Db Dc Dd De Df Dg] respectivement aux broches RB0, RB1, RB2, RB3, RB4, RB5 et RB6 du microcontrôleur PIC16F876A.
- 4. Essayer d'utiliser le gestionnaire d'afficheur 7 segment



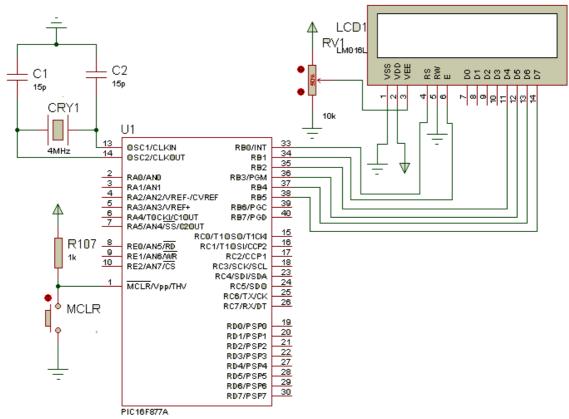




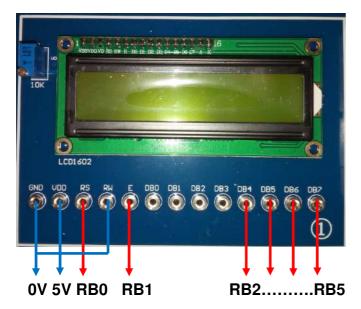
Programme N°1 **Programme N°2** program compteur10; program COMPTEUR2; begin var i : integer; const chiffre : array[10] of byte =(192, 249, 164,176, Trisb:=0; 153, 146, 130, 248, 128,144); portb:=0; while (1=1) do begin begin //segments [g f e d c b a] Trisb:=0; portb:=192; // chiffre 0 portb:=0; delay_ms(1000); // attente d'une seconde while (1=1) do portb:=249; // chiffre 1 begin for i := 0 to 9 do // compteur delay_ms(1000); begin portb:=164; // chiffre 2 portb:=chiffre[i]; // affichage du chiffre delay ms(1000); // attente d'une seconde delay_ms(1000); portb:=176; // chiffre 3 end: delay ms(1000); end: portb:=153; // chiffre 4 end. delay_ms(1000); portb:=146; // chiffre 5 delay ms(1000); portb:=130; // chiffre 6 delay_ms(1000); portb:=248; // chiffre 7 delay_ms(1000); portb:=128; // chiffre 8 delay_ms(1000); portb:=144; // chiffre 9 delay ms(1000); end: end.

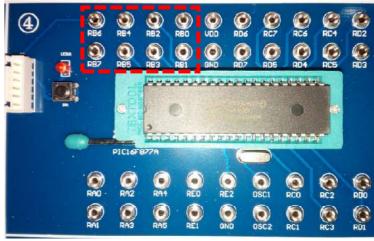
ACTIVITÉ N° 3: GESTION D'UN AFFICHEUR LCD

Soit le montage suivant :



```
Programme N°1
Program affichage;
// Connections du module Lcd
var LCD RS: sbit at PORTB.0;
var LCD EN: sbit at PORTB.1;
var LCD D4 : sbit at PORTB.2;
var LCD D5 : sbit at PORTB.3;
var LCD D6: sbit at PORTB.4;
var LCD D7: sbit at PORTB.5;
var LCD RS Direction: sbit at TRISB.0;
var LCD EN Direction: sbit at TRISB.1;
var LCD_D4_Direction: sbit at TRISB.2;
var LCD D5 Direction: sbit at TRISB.3;
var LCD D6 Direction: sbit at TRISB.4;
begin
LCD_init();
LCD_CMD(_LCD_CURSOR_OFF);
while true do
  begin
  LCD_out(1,1,'FORMATION PIC');
  end:
end.
```

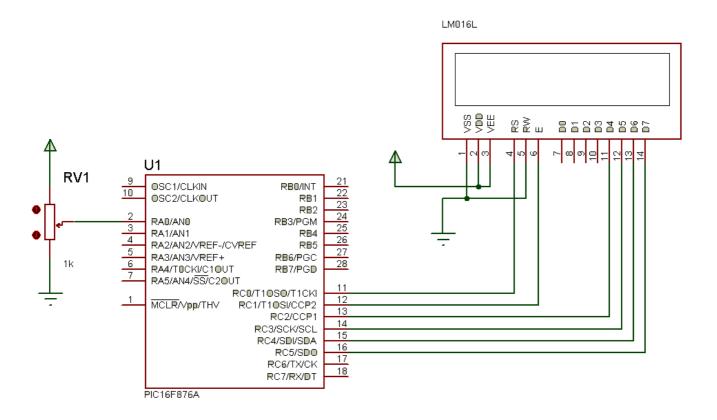




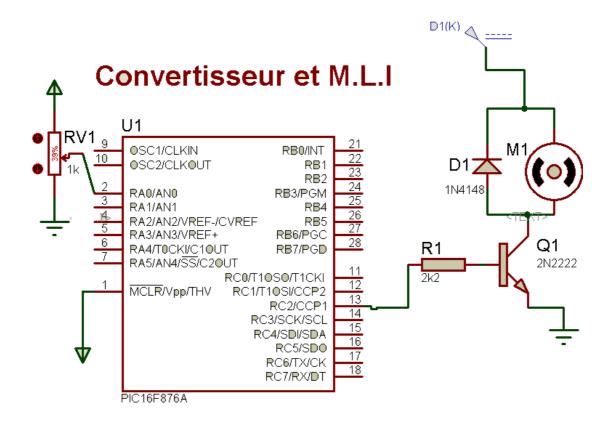
ACTIVITÉ N° 4: CAN - VOLTMÈTRE

On désire de réaliser un voltmètre en utilisant CAN et afficheur LCD

```
Programme
program voltmetre;
var
N:word;
Cal: real;
T: word;
valeur affichage: string[3];
LCD_RS: sbit at portc.0;
LCD EN: sbit at portc.1;
LCD_D4: sbit at portc.2;
LCD D5: sbit at portc.3;
LCD_D6: sbit at portc.4;
LCD D7: sbit at portc.5;
LCD_RS_Direction: sbit at TRISC.0;
LCD EN Direction: sbit at TRISC.1;
LCD D4 Direction: sbit at TRISC.2;
LCD D5 Direction: sbit at TRISC.3;
LCD_D6_Direction : sbit at TRISC.4;
LCD D7 Direction: sbit at TRISC.5;
begin
adcon1:=%10000000;
lcd_init();
lcd_cmd(_LCD_CURSOR_OFF);
lcd_out(1,1,'V=');
adc init();
while true do
begin
N := adc read(0);
Cal := (N*5000)/1023;
T:= word(Cal);
wordToStr(T,valeur_affichage);
lcd out(1,3,valeur affichage);
lcd out(1,10,'mV');
delay_ms(100);
end; end.
```



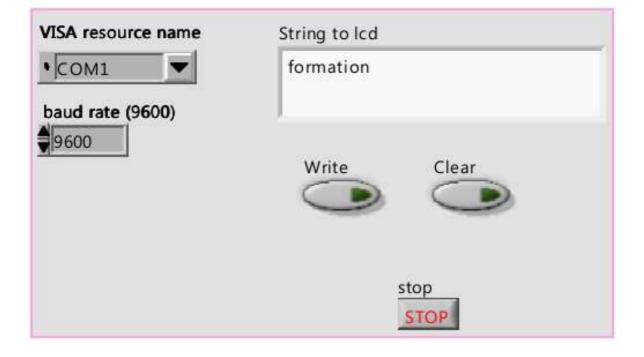
ACTIVITÉ N° 5: MLI – CONVERSIO N

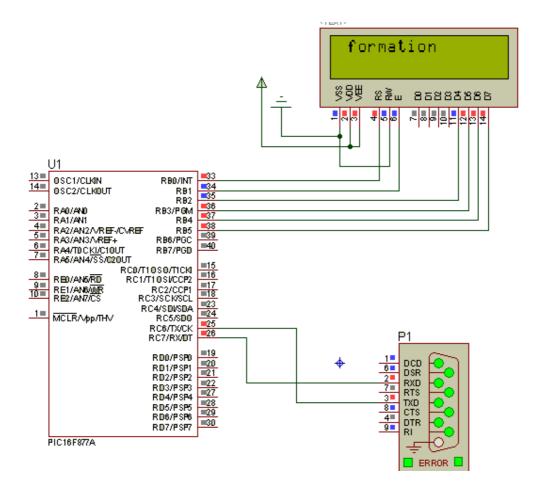


Programme program MLI_conversion; var K: byte; N:word; begin PWM1 init(250); // Initialiser le module PWM1 et choix de la fréquence de PWM1= 250 Hz ADCON1:=\$80; // \$8E Configuration des entrées du porta comme entrées analogiques y compris PWM1_start; // démarrage du module PWM1 while true do // boucle infinite begin N:= adc read(0); // lecture de la conversion K:=N/4; // calcul PWM1_set_duty (K); // changement du rapport cyclique : le rapport cyclique est K/ 255 end; end.

ACTIVITÉ N° 6: INTERFACE UART + LABVIEW

On désire de créer une interface en **LABVIEW** pour commander un afficher LCD :





```
Programme
program testlcd;
  var i : char;
var LCD RS: sbit at RB0 bit;
var LCD EN: sbit at RB1 bit;
var LCD D4: sbit at RB2 bit;
var LCD_D5: sbit at RB3_bit;
var LCD_D6 : sbit at RB4_bit;
var LCD D7: sbit at RB5 bit;
var LCD RS Direction: sbit at TRISB0 bit;
var LCD EN Direction : sbit at TRISB1 bit;
var LCD D4 Direction : sbit at TRISB2 bit;
var LCD D5 Direction : sbit at TRISB3 bit;
var LCD D6 Direction : sbit at TRISB4 bit;
var LCD_D7_Direction : sbit at TRISB5_bit;
begin
Lcd Init();
                        // Initialize LCD
  Lcd_Cmd(_LCD_CLEAR);
                                 // Clear display
  Lcd_Cmd(_LCD_CURSOR_OFF);
                                        // Cursor off
uart1_init (9600); // init uart
delay_ms (100);
while true do
begin
if ( uart1_data_ready() <> 0 ) then
begin
i:= uart1_read();
if (i = 13) then
lcd_cmd (_lcd_clear);
LCD chr cp(i);
  end;
 end:
 end.
```

