

# UTILISATION DU SIMULATEUR PIC : PICSIMLAB<sup>1</sup>

1. Introduction.....	2
2. Protocole d'utilisation.....	2
3. Board Mclab2 (PIC16F877A).....	2
4. Exemple de code.....	5
4.1 Test.....	6
4.2 Port UART.....	6
4.3 Oscilloscope.....	8
5. Création d'un board personnel.....	9
5.1 Le board.....	9
5.2 Utilisation du LCD.....	10
5.2.1 Exemple de texte fixe.....	10
a ) Explication.....	10
b ) Code.....	10
5.2.2 Exemple de texte avec variable formatée.....	11
a ) Explication.....	11
5.3 Test moteur pas à pas sur 'Spare parts'.....	12
5.3.1 Schéma.....	12
5.3.2 Explication.....	12
5.3.3 Code.....	12
5.3.4 Conclusion.....	15
6. Annexes.....	16
6.1 Code du lcd.h.....	16
6.2 Code du lcd.c.....	18
6.3 Code complet du programme !;.....	21

---

<sup>1</sup>By S.B. v08/05/20-23:56

## 1. Introduction

Ce simulateur permet de tester des programmes compilés pour différents PIC.

Le programme HEX est chargé et utilisé sur différents boards.

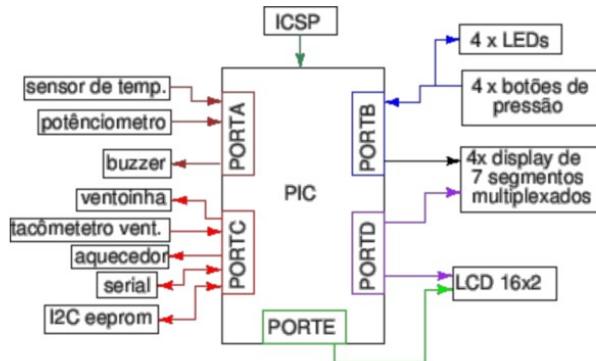
**ATTENTION** : la version 64 bits plante. La version 32 bits fonctionne bien.

## 2. Protocole d'utilisation

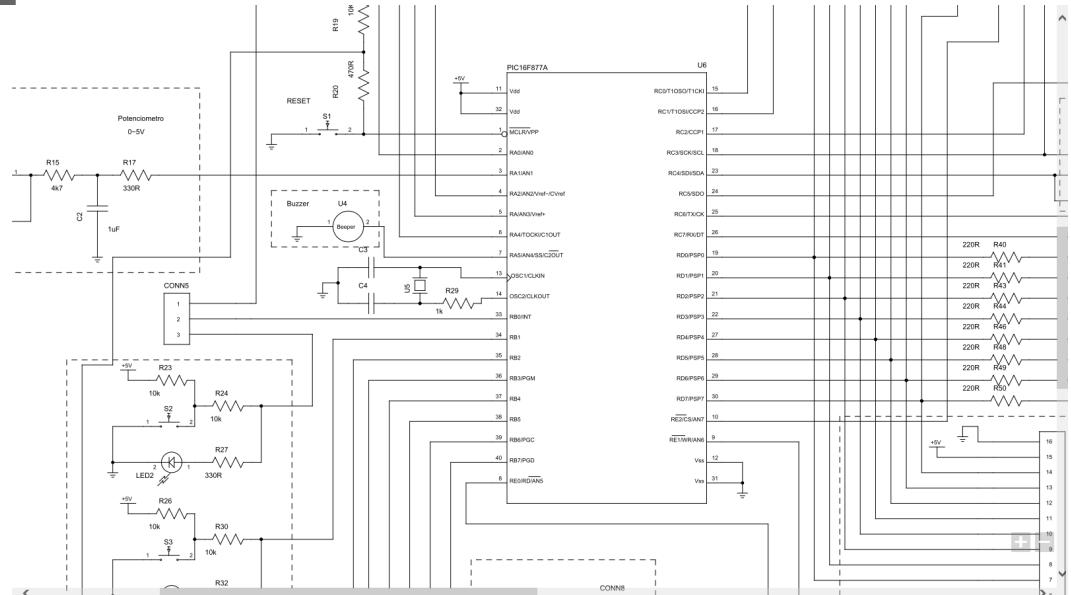
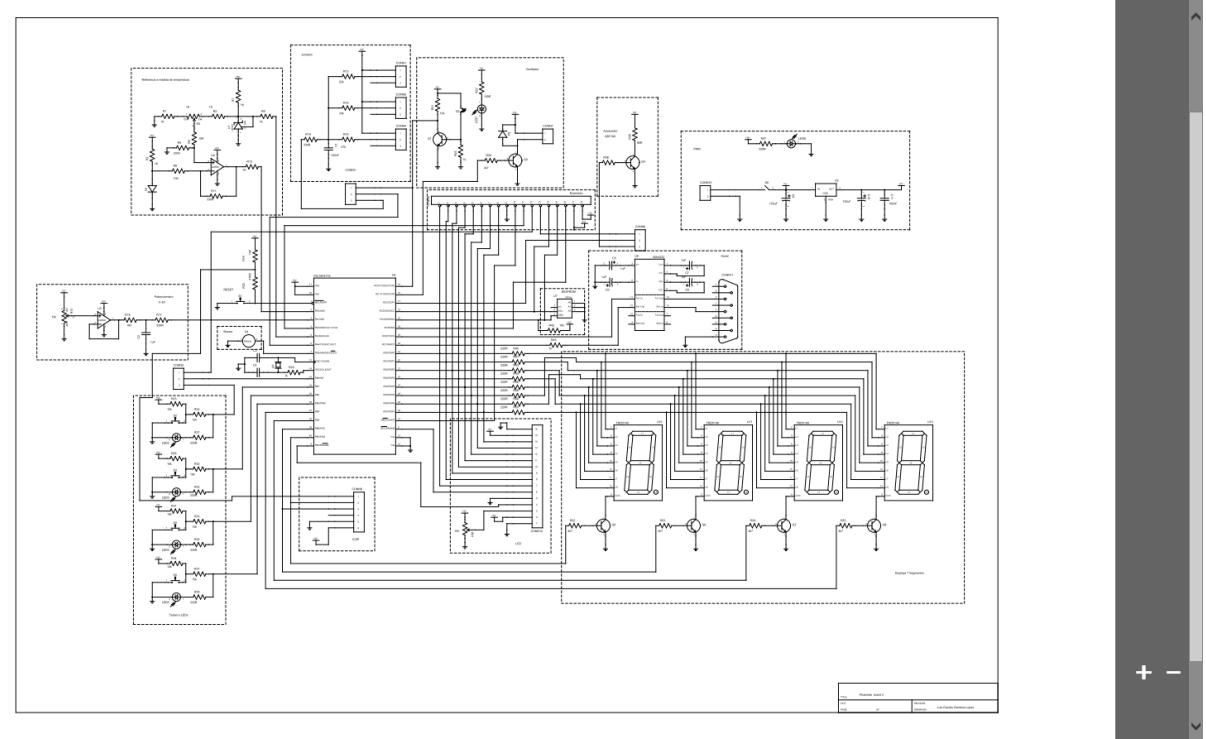
1. Écrire le programme en C en suivant les broches du schéma de la board
2. Compiler le programme en utilisant PICC ou MPLABX
3. Charger l'HEX dans la board adaptée.
4. Tester le bon fonctionnement de votre code.

La board utilisée dans ce tuto est la McLab2 en utilisant un 16F877A. **Le schéma en pdf est disponible sur 'stssnsb.free.fr'-rubrique logiciels indispensables – 'picsimlab'.**

## 3. Board McLab2 (PIC16F877A)



## Utilisation du simulateur PIC : PicSimLab



La board contient de nombreux éléments :

LED, switch, un potentiomètre P2, 4 afficheurs 7 segments multiplexé commandés par transistors, afficheur LCD parallèle.

Le schéma donne toutes les informations pour créer les programmes.

Des erreurs entre les noms du board et le schéma sont présentes :

- sur le board S1,S2,S3,S4 correspondent à S2,S3,S4 et S5 sur le schéma : décalage de 1.
- attention en rajoutant des éléments externes à ne pas faire de court circuit : broches complètement libres :

portA	portB	portC	portD	portE
A0: capteur température A1 : pot P2 <b>A2 : libre</b> A3 : ref A4 : pont diviseur controlé par les switch OU capteur lumière A5 : buzzer (attention problème car pic16F877 ne peut pas avoir A4 en analogique!!!!)	B0 : libre OU switch S2(S1 et led L1) B1 : sw S3 (S2 et led L2) B2 : sw S4 (S3 et led L3) B3 : sw S5 (S4 et led L4) B4 : mux AFFQ5(7SEG gauche) B5 : mux AFFQ6 B6 : mux AFFQ7 B7 : mux AFFQ8(7SEG droit)	C0 : état RX infrarouge C1 : commande Q3 CON7 C2 : LIBRE ou Q3 heater par CON7 C3 : SCL ou libre C4 : SDA ou libre <b>C5 : LIBRE</b> C6 : TX C7 : RX	D0 à D7 : Libre en sortie OU segments des afficheurs D0=a, D1=f, D2=b ,D3=g ,D4=e ,D5=c ,D6=d ,D7= dp.	E0 : libre <b>E1 : libre</b> <b>E2 : libre</b>

### 4. Exemple de code

```
#include <16F877A.h>
#device adc=8

#FUSES NOWDT           //No Watch Dog Timer
#FUSES HS              //High speed Osc (> 4mhz for PCM/PCH) (>10mhz for PCD)
#FUSES NOPUT            //No Power Up Timer
#FUSES NOPROTECT        //Code not protected from reading
#FUSES NODEBUG           //No Debug mode for ICD
#FUSES NOBROWNOUT       //No brownout reset
#FUSES NOLVP             //No low voltage prgming, B3(PIC16) or B5(PIC18) used for I/O
#FUSES NOCPD             //No EE protection
#FUSES NOWRT             //Program memory not write protected

#use delay(clock=20000000)
#define BP0_B0  PIN_B0

#define BP_L1 PIN_B1
#define BP_L2 PIN_B2
#define L3 PIN_B3
#define AFFQ8 PIN_B4
#define AFFQ7 PIN_B5
#define AFFQ6 PIN_B6
#define AFFQ5 PIN_B7
#define 7SEGA PIN_D0
#define 7SEG_B PIN_D1
#define 7SEG_C PIN_D2
#define 7SEG_D PIN_D3
#define 7SEG_E PIN_D4
#define 7SEG_F PIN_D5
#define 7SEG_G PIN_D6
#define 7SEG_DP PIN_D7
#define P2 PIN_A1

#define TX PIN_C6
#define RX PIN_C7
#use rs232(baud=9600,parity=N,xmit=TX,rcv=RX,bits=8)

void main()
{
int16 valeurA0=0;
```

```
setup_adc_ports(AN0_AN1_AN2_AN3_AN4);
setup_adc(ADC_CLOCK_INTERNAL);
setup_psp(PSP_DISABLED);
setup_spi(SPI_SS_DISABLED);
setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
setup_timer_1(T1_DISABLED);
setup_timer_2(T2_DISABLED,0,1);
setup_comparator(NC_NC_NC_NC);
setup_vref(FALSE);

printf("Debut...");delay_ms(1000);
// TODO: USER CODE!!
while(1{
    //Test LED
    output_high(BP_L1);delay_ms(1000);output_low(BP_L1);
    output_toggle(BP_L2);
    //Test 7SEG
    output_low(AFFQ5);output_low(AFFQ6);output_low(AFFQ7);output_low(AFFQ8);
    output_high(AFFQ8);
    output_d(0b00000110);
    //Test INTER
    if (input(BP_L1)) printf("A");
    //Test Analogique A1
    set_adc_channel(1);
    valeurA0=read_adc();
    printf("valA0= %lu",valeurA0);

};//while

}//main
```

### 4.1 Test

Ce programme affiche 1 sur un des afficheurs 7 segments:ok

Mesure la valeur sur AN1 et l'envoie sur le port série: ok

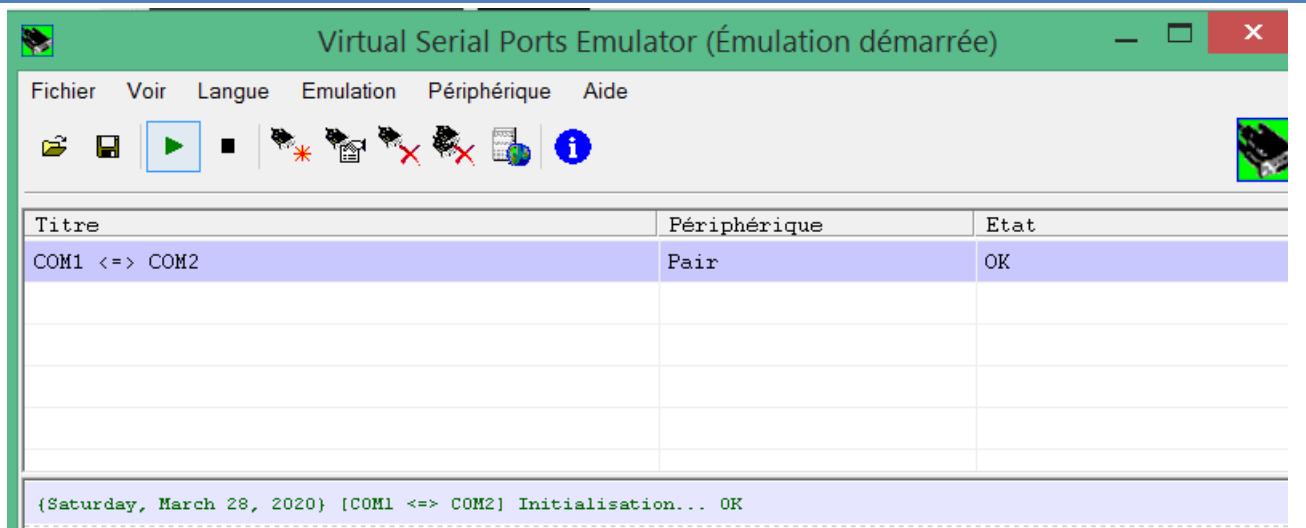
La LED L2 clignote : ok

Un appui sur BP\_L1 envoie 'A' sur le port série. : ok

### 4.2 Port UART

Le test de la liaison série nécessite un émulateur de port série afin de diriger les données du simulateur de pic vers un terminal série.

## Utilisation du simulateur PIC : PicSimLab



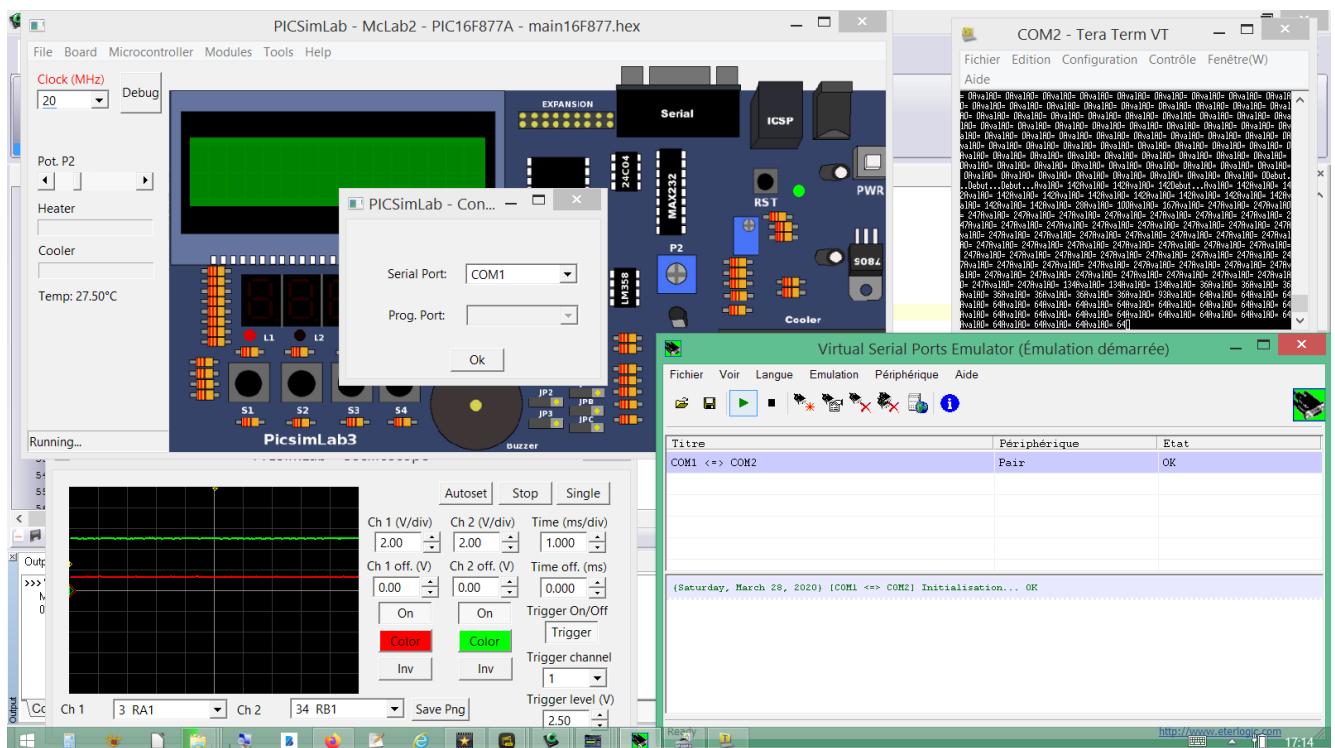
On installe 'virtual serial ports emulator' (gratuit) avec lequel on crée 2 ports virtuels : COM1 et COM2.

Sur PicSimLab : on configure l'UART sur COM1 (9600 – 8N1)

Sur un terminal (ici Teraterm) : on configure le COM2 (9600-8N1)

**Attention : lancer VSPE et Teraterm avant PicSimLab.**

Les printf du code C sont alors envoyés vers Teraterm :



### **4.3 Oscilloscope**

Un oscilloscope permet de visualiser les broches du Pic :

#### **Modules + Oscilloscope**

Faire les réglages classiques : Calibre des tension, base de temps et déclenchement

Choisir les broches à visualiser.

## 5. Crédit d'un board personnel

**ATTENTION : Enregistrer les boards avec un nom différent à chaque sauvegarde.  
Utiliser un même nom en écrasant le précédent fait PLANTER le logiciel !!!**

Un outil permet de créer son propre board :

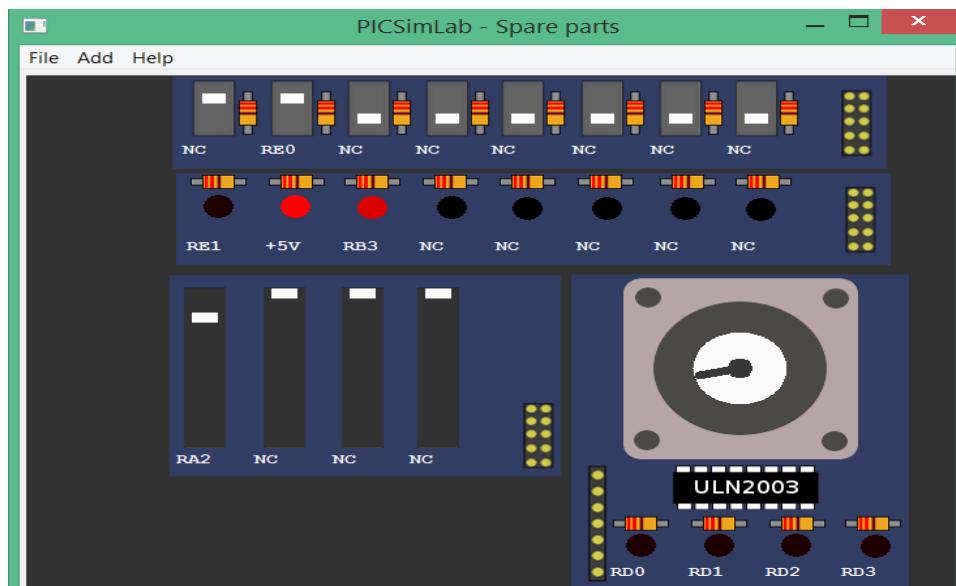
**Modules + Spare parts**

Un doc. En anglais est disponible dans : Help + Contents

### 5.1 Le board

Pour faire fonctionner le board 'perso' il faut lancer aussi le board McLab2 (qui gère le PIC)

Le board 'perso' permet d'ajouter des périphériques adaptés à votre projet.



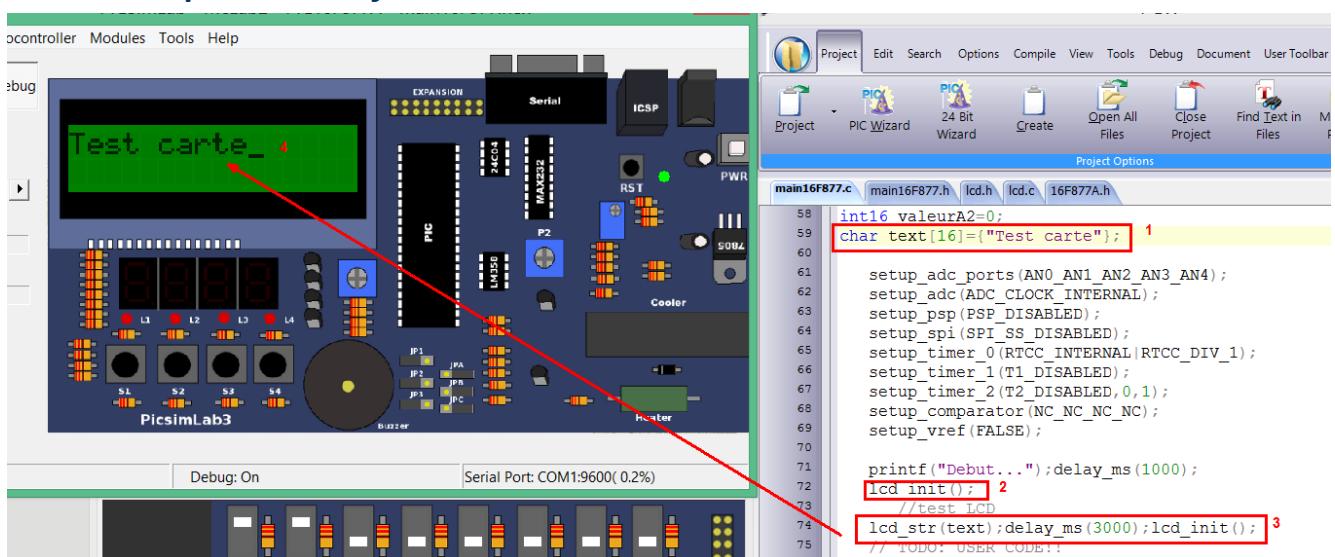
## 5.2 Utilisation du LCD

L'usage du LCD du PicSimLab3 nécessite l'utilisation du driver fournit pour Mplabx et modifié.(Présent dans le zip sur stssnsb.free.fr – logiciels -PicSimLab code)

Trois fonctions sont utiles :

```
lcd_init() : initialise le LCD (efface et retour au début)
lcd_str(text) : affiche le texte contenu dans le tableau de char 'text'
lcd_dat() : permet d'afficher un texte contenant des variables.
```

### 5.2.1 Exemple de texte fixe



#### a) Explication

On déclare un tableau de char : text[16].

On initialise le lcd par lcd\_init() ;

On écrit le text sur le lcd par lcd\_str() ;

#### b) Code

```
char text[16]={"Test carte"};

setup_adc_ports(AN0_AN1_AN2_AN3_AN4);
setup_adc(ADC_CLOCK_INTERNAL);
setup_psp(PSP_DISABLED);
setup_spi(SPI_SS_DISABLED);
setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
setup_timer_1(T1_DISABLED);
setup_timer_2(T2_DISABLED,0,1);
```

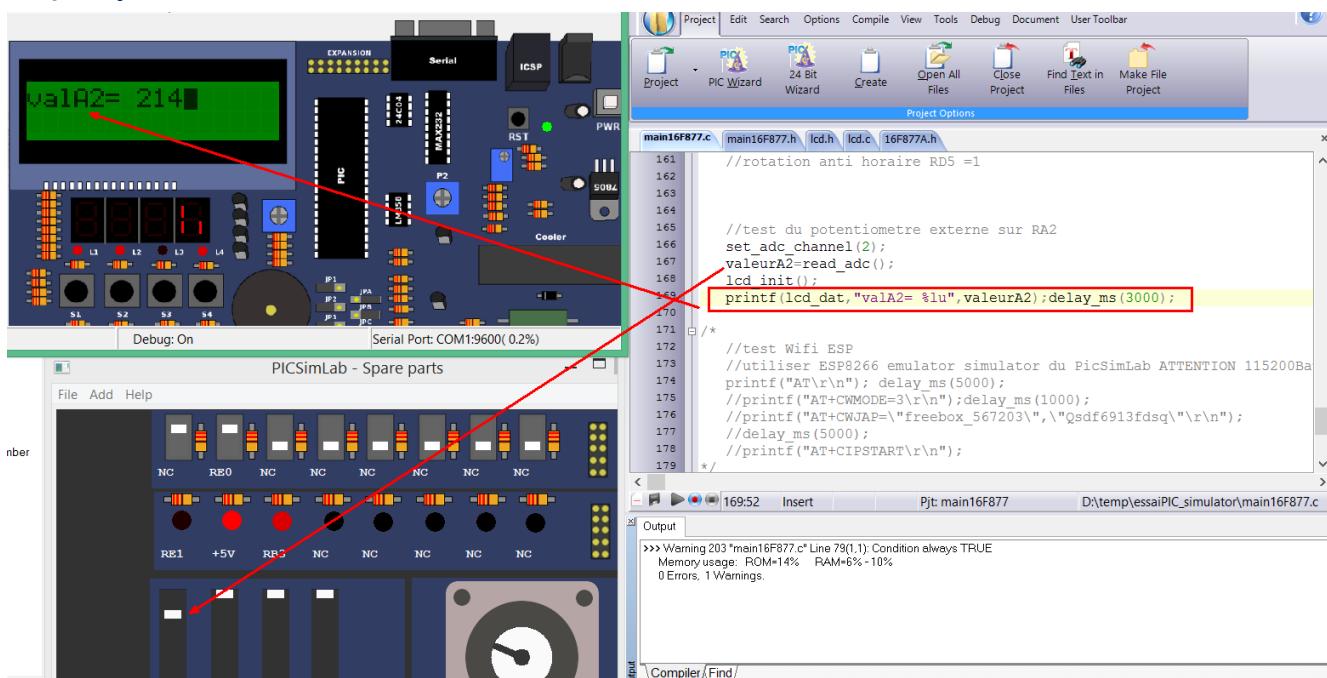
## Utilisation du simulateur PIC : PicSimLab

```
setup_comparator(NC_NC_NC_NC);  
setup_vref(FALSE);  
  
printf("Debut...");delay_ms(1000);  
lcd_init();  
//test LCD  
lcd_str(text);delay_ms(3000);lcd_init();
```

### 5.2.2 Exemple de texte avec variable formatée

La figure ci dessous montre la mesure sur le portA2 analogique (ajouter avec 'spare parts') et son affichage sur le LCD.

#### a ) Explication



on utilise la fonction 'lcd\_dat' dans le printf afin de profiter du formatage du printf..

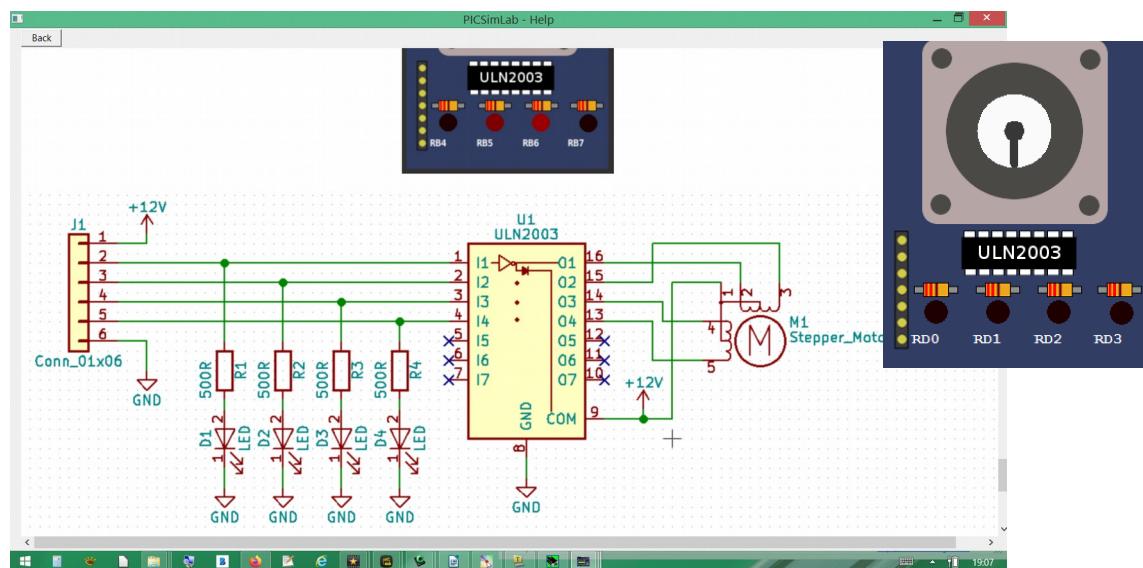
**Attention : cet usage envoie les données vers le lcd et non vers le port série comme habituellement.**

Les sources de lcd.h et lcd.c sont dans le zip fournit.

Remarque : l'affichage sur la première ligne se fait après un lcd\_init(). L'affichage sur la seconde est en cours d'étude !!

## 5.3 Test moteur pas à pas sur 'Spare parts'

### 5.3.1 Schéma



### 5.3.2 Explication

On dispose d'un MPP 4 bobines.

Les bobines sont respectivement contrôlées par RD0 RD1 RD2 RD3.

### 5.3.3 Code

Cette exemple est développé avec PICC et fournit un .HEX (Vous pouvez aussi utiliser MPLABX : PicSimLab s'intègre dans le logiciel – voir l'aide)

```
#include <16F877A.h>
#device adc=8

#FUSES NOWDT          //No Watch Dog Timer
#FUSES HS              //High speed Osc (> 4mhz for PCM/PCH) (>10mhz for
PCD)
#FUSES NOPUT           //No Power Up Timer
#FUSES NOPROTECT       //Code not protected from reading
#FUSES NODEBUG          //No Debug mode for ICD
#FUSES NOBROWNOUT      //No brownout reset
#FUSES NOLVP           //No low voltage prgming, B3(PIC16) or B5(PIC18)
used for I/O
#FUSES NOCPD           //No EE protection
#FUSES NOWRT            //Program memory not write protected
```

## Utilisation du simulateur PIC : PicSimLab

```
#use delay(clock=20000000)

#define BP0_B0    PIN_B0

#define BP_L1 PIN_B1
#define BP_L2 PIN_B2
#define L3 PIN_B3
#define AFFQ8 PIN_B4
#define AFFQ7 PIN_B5
#define AFFQ6 PIN_B6
#define AFFQ5 PIN_B7
#define 7SEGA PIN_D0
#define 7SEGB PIN_D1
#define 7SEGC PIN_D2
#define 7SEGD PIN_D3
#define 7SEGE PIN_D4
#define 7SEGF PIN_D5
#define 7SEGG PIN_D6
#define 7SEGDP PIN_D7
#define P2 PIN_A1

//defin epour le moteur pas a pas MPP
#define D0a1 output_high(PIN_D0)
#define D0a0 output_low(PIN_D0)
#define D1a1 output_high(PIN_D1)
#define D1a0 output_low(PIN_D1)
#define D2a1 output_high(PIN_D2)
#define D2a0 output_low(PIN_D2)
#define D3a1 output_high(PIN_D3)
#define D3a0 output_low(PIN_D3)

#define TX PIN_C6
#define RX PIN_C7
#use rs232(baud=9600,parity=N,xmit=TX,rcv=RX,bits=8)

void main()
{
int16 valeurA0=0;
    setup_adc_ports(AN0_AN1_AN2_AN3_AN4);
    setup_adc(ADC_CLOCK_INTERNAL);
```

## Utilisation du simulateur PIC : PicSimLab

```
setup_psp(PSP_DISABLED);
setup_spi(SPI_SS_DISABLED);
setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
setup_timer_1(T1_DISABLED);
setup_timer_2(T2_DISABLED,0,1);
setup_comparator(NC_NC_NC_NC);
setup_vref(FALSE);

printf("Debut...");delay_ms(1000);

while(1){
    //Test LED
    output_high(BP_L1);delay_ms(1000);output_low(BP_L1);
    output_toggle(BP_L2);
    //Test 7SEG
    output_low(AFFQ5);output_low(AFFQ6);output_low(AFFQ7);output_low(AFFQ8);
    output_high(AFFQ8);
    output_d(0b00000110);
    //Test INTER
    if (input(BP_L1)) printf("A");
    //Test Analogique A1
    set_adc_channel(1);
    valeurA0=read_adc();
    printf("valA0= %lu",valeurA0);

    //Test Spareparts board avec inter (RD4RD5) pasapas(RD0RD1RD2RD3) LED(RE5)
    //Test INTER RD4 et LED RE2
    printf("\r\nTest LEDRE2 INTER RD4");
    if (input(PIN_D4)) output_high(PIN_E2);
    //delay_ms(1000);
    //Test PASAPAS RD0RD1RD2RD3 (ULN003)
    printf("\r\nTest PAS A PAS\r\n");

    printf("\r\nMPP ... \r\n");
    //D0a1;D1a0;D2a0;D3a0;delay_ms(50);
    D0a0;D1a1;D2a0;D3a0;delay_ms(50);
    D0a0;D1a0;D2a1;D3a0;delay_ms(50);
    D0a0;D1a0;D2a0;D3a1;delay_ms(50);
```

```
//D0a1;D1a0;D2a0;D3a0;delay_ms(50);  
D0a0;D1a1;D2a0;D3a0;delay_ms(50);  
D0a0;D1a0;D2a1;D3a0;delay_ms(50);  
D0a0;D1a0;D2a0;D3a1;delay_ms(50);  
  
D0a1;D1a0;D2a0;D3a0;delay_ms(50);  
D0a0;D1a1;D2a0;D3a0;delay_ms(50);  
D0a0;D1a0;D2a1;D3a0;delay_ms(50);  
D0a0;D1a0;D2a0;D3a1;delay_ms(50);  
  
D0a1;D1a0;D2a0;D3a0;delay_ms(50);  
D0a0;D1a1;D2a0;D3a0;delay_ms(50);  
D0a0;D1a0;D2a1;D3a0;delay_ms(50);  
D0a0;D1a0;D2a0;D3a1;delay_ms(50);  
  
} //while  
  
}//main
```

### 5.3.4 Conclusion

L'enclenchement de l'interrupteur RD4 affiche un segment sur l'afficheur 7 segment de droite : ok

Le moteur tourne : ok (un pas en arrière et 3 en avant : revoir la séquence des pas).

## 6. Annexes

### 6.1 Code du lcd.h

```
/* #####PICsim - PIC simulator http://sourceforge.net/projects/picsim/
#####
Copyright (c) : 2015 Luis Claudio Gamb?a Lopes

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2, or (at your option)
any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

For e-mail suggestions : lcgamboa@yahoo.com
##### */



//#define LENA PORTEbits.RE1
#define LENA PIN_E1
//#define LDAT PORTEbits.RE0
#define LDAT PIN_E0


//#define LPORT PORTD


#define L_ON 0x0F
#define L_OFF 0x08
#define L_CLR 0x01
#define L_L1 0x80
#define L_L2 0xC0
#define L_CR 0x0F
#define L_NCR 0x0C
```

## Utilisation du simulateur PIC : PicSimLab

---

```
#define L_CFG 0x38

void lcd_init(void);
void lcd_cmd(unsigned char val);
void lcd_dat(unsigned char val);
void lcd_str(char* str); //void lcd_str(const char* str);
```

### 6.2 Code du lcd.c

```
/* #####PICsim - PIC simulator http://sourceforge.net/projects/picsim/
#####
Copyright (c) : 2015 Luis Claudio Gamb?a Lopes

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2, or (at your option)
any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

For e-mail suggestions : lcgamboa@yahoo.com
#####
*/
//#include <xc.h>
#include "lcd.h"

void atraso_ms(unsigned int valor)
{
    unsigned int i;
    unsigned char j;

    for (i =0; i< valor; i++)
    {
        for (j =0 ; j < 200; j++)
        {
            #asm
            NOP
        }
    }
}
```

## Utilisation du simulateur PIC : PicSimLab

```
NOP
NOP
NOP
NOP
#endasm;
}
}

void lcd_wr(unsigned char val)
{

// LPORT=val;
output_d(val);
}

void lcd_cmd(unsigned char val)
{

output_high(LENA);//LENA=1;
lcd_wr(val);
output_low(LDAT);//LDAT=0;
atraso_ms(3);
output_low(LENA);//LENA=0;
atraso_ms(3);
output_high(LENA);//LENA=1;
}

void lcd_dat(unsigned char val)
{

output_high(LENA);//LENA=1;
lcd_wr(val);
output_high(LDAT);//LDAT=1;
atraso_ms(3);
output_low(LENA);//LENA=0;
atraso_ms(3);
output_high(LENA);//LENA=1;
}

void lcd_init(void)
{
output_low(LENA);//LENA=0;
output_low(LDAT);//LDAT=0;
atraso_ms(20);
output_high(LENA);//LENA=1;

lcd cmd(L CFG);
```

## Utilisation du simulateur PIC : PicSimLab

```
atraso_ms(5);
lcd_cmd(L_CFG);
    atraso_ms(1);
lcd_cmd(L_CFG); //configura
lcd_cmd(L_OFF);
lcd_cmd(L_ON); //liga
lcd_cmd(L_CLR); //limpa
lcd_cmd(L_CFG); //configura
    lcd_cmd(L_L1);
}

void lcd_str(char* str)
{
    unsigned char i=0;

    while(str[i] != 0 )
    {
        lcd_dat(str[i]);
        i++;
    }
}
```

### 6.3 Code complet du programme !;

```
#include <16F877A.h>
#device adc=8

#FUSES NOWDT      //No Watch Dog Timer
#FUSES HS        //High speed Osc (> 4mhz for PCM/PCH) (>10mhz for PCD)
#FUSES NOPUT      //No Power Up Timer
#FUSES NOPROTECT  //Code not protected from reading
#FUSES NODEBUG    //No Debug mode for ICD
#FUSES NOBROWNOUT //No brownout reset
#FUSES NOLVP      //No low voltage prgming, B3(PIC16) or B5(PIC18) used for I/O
#FUSES NOCPD      //No EE protection
#FUSES NOWRT      //Program memory not write protected

#use delay(clock=20000000)

#include "lcd.c" //lcd.c adapté à PICC

#define BP0_B0 PIN_B0
#define BP_L1 PIN_B1
#define BP_L2 PIN_B2
#define L3 PIN_B3
#define AFFQ8 PIN_B4
#define AFFQ7 PIN_B5
#define AFFQ6 PIN_B6
#define AFFQ5 PIN_B7
#define 7SEGA PIN_D0
#define 7SEG_B PIN_D1
#define 7SEGC PIN_D2
#define 7SEGD PIN_D3
#define 7SEGE PIN_D4
#define 7SEGF PIN_D5
#define 7SEGG PIN_D6
#define 7SEGDP PIN_D7
#define P2 PIN_A1
#define POTEXTA2 PIN_A2
#define BUZZER PIN_A5

#define D0a1 output_high(PIN_D0)
#define D0a0 output_low(PIN_D0)
```

## Utilisation du simulateur PIC : PicSimLab

---

```
#define D1a1 output_high(PIN_D1)
#define D1a0 output_low(PIN_D1)
#define D2a1 output_high(PIN_D2)
#define D2a0 output_low(PIN_D2)
#define D3a1 output_high(PIN_D3)
#define D3a0 output_low(PIN_D3)

#define TX PIN_C6
#define RX PIN_C7
#use rs232(baud=9600,parity=N,xmit=TX,rcv=RX,bits=8)

void main()
{
    int16 valeurA0=0;
    int16 valeurA2=0;
    char text[16]={"Test carte"};

    setup_adc_ports(AN0_AN1_AN2_AN3_AN4);
    setup_adc(ADC_CLOCK_INTERNAL);
    setup_psp(PSP_DISABLED);
    setup_spi(SPI_SS_DISABLED);
    setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
    setup_timer_1(T1_DISABLED);
    setup_timer_2(T2_DISABLED,0,1);
    setup_comparator(NC_NC_NC_NC);
    setup_vref(FALSE);

    printf("Debut...");delay_ms(1000);
    lcd_init();
    //test LCD
    lcd_str(text);delay_ms(3000);lcd_init();
    // TODO: USER CODE!!

    while(1{
        lcd_init();
        //Test LED
        output_high(BP_L1);delay_ms(1000);output_low(BP_L1);
        output_toggle(BP_L2);
        //Test 7SEG
        output_low(AFFQ5);output_low(AFFQ6);output_low(AFFQ7);output_low(AFFQ8);
        output_high(AFFQ8);
        output_d(0b00000110);
        //Test INTER
        if (input(BP_L1)) printf("A");
    }
}
```

## Utilisation du simulateur PIC : PicSimLab

---

```
//commande du heater (Q4 = RC2 avec CON9)
output_high(PIN_C2);delay_ms(1000);

//mesure sur A1 potentiometre P2
set_adc_channel(1);
valeurA0=read_adc();
printf(lcd_dat,"P2= %lu",valeurA0);

//commande ventilateur (C1)
output_high(PIN_C1);delay_ms(1000);

//commande du heater (Q4 = RC2 avec CON9)
output_low(PIN_C2);delay_ms(1000);

//buzzer NE MARCHE PAS CAR problème avec RA5 qui est AN4 donc analogique en entrée
output_high(BUZZER);delay_ms(1);output_low(BUZZER);delay_ms(1);
output_high(BUZZER);delay_ms(1);output_low(BUZZER);delay_ms(1);
output_high(BUZZER);delay_ms(1);output_low(BUZZER);delay_ms(1);
output_high(BUZZER);delay_ms(1);output_low(BUZZER);delay_ms(1);

//Test Spareparts board avec inter (RD4RD5) pasapas(RD0RD1RD2RD3) LED(RE5)
//Test INTER RD4 et LED RE2
printf("\r\nTest LEDRE0 INTER RE1");
if (input(PIN_E1)==1) output_high(PIN_E0);
else output_low(PIN_E0);
//delay_ms(1000);

//Test PASAPAS RDORD1RD2RD3 (ULN003)
printf("\r\nTest PAS A PAS\r\n");
//rotation horaire RD5 = 0
//if (input(PIN_E0==0))
//{
    printf("\r\nMPP ... \r\n");
    //D0a1;D1a0;D2a0;D3a0;delay_ms(50);
    D0a0;D1a1;D2a0;D3a0;delay_ms(50);
    D0a0;D1a0;D2a1;D3a0;delay_ms(50);
    D0a0;D1a0;D2a0;D3a1;delay_ms(50);

    //D0a1;D1a0;D2a0;D3a0;delay_ms(50);
    D0a0;D1a1;D2a0;D3a0;delay_ms(50);
    D0a0;D1a0;D2a1;D3a0;delay_ms(50);
    D0a0;D1a0;D2a0;D3a1;delay_ms(50);

    D0a1;D1a0;D2a0;D3a0;delay_ms(50);
    D0a0;D1a1;D2a0;D3a0;delay_ms(50);
```

## Utilisation du simulateur PIC : PicSimLab

---

```
D0a0;D1a0;D2a1;D3a0;delay_ms(50);  
D0a0;D1a0;D2a0;D3a1;delay_ms(50);  
  
D0a1;D1a0;D2a0;D3a0;delay_ms(50);  
D0a0;D1a1;D2a0;D3a0;delay_ms(50);  
D0a0;D1a0;D2a1;D3a0;delay_ms(50);  
D0a0;D1a0;D2a0;D3a1;delay_ms(50);  
  
D0a1;D1a0;D2a0;D3a0;delay_ms(50);  
D0a0;D1a1;D2a0;D3a0;delay_ms(50);  
D0a0;D1a0;D2a1;D3a0;delay_ms(50);  
D0a0;D1a0;D2a0;D3a1;delay_ms(50);  
  
/*  
D0a1;D1a0;D2a0;D3a0;delay_ms(200);  
D0a0;D1a1;D2a0;D3a0;delay_ms(200);  
D0a0;D1a0;D2a1;D3a0;delay_ms(200);  
D0a0;D1a0;D2a0;D3a1;delay_ms(200);  
  
D0a1;D1a0;D2a0;D3a0;delay_ms(200);  
D0a0;D1a1;D2a0;D3a0;delay_ms(200);  
D0a0;D1a0;D2a1;D3a0;delay_ms(200);  
D0a0;D1a0;D2a0;D3a1;delay_ms(200);  
*/  
//}//if  
//rotation anti horaire RD5 =1  
  
  
  
//test du potentiometre externe sur RA2  
set_adc_channel(2);  
valeurA2=read_adc();  
lcd_init();  
printf(lcd_dat,"valA2= %lu",valeurA2);delay_ms(3000);  
  
/*  
//test Wifi ESP  
//utiliser ESP8266 emulator simulator du PicSimLab ATTENTION 115200Bauds sinon marche pas  
printf("AT\r\n"); delay_ms(5000);  
//printf("AT+CWMODE=3\r\n");delay_ms(1000);  
//printf("AT+CWJAP=\"freebox_567203\",\"Qsdf6913fdsq\"\r\n");  
//delay_ms(5000);  
//printf("AT+CIPSTART\r\n");  
*/  
  
}//while
```

## Utilisation du simulateur PIC : PicSimLab

---

}//main