Université de Batna 2
جامعـــة باتنـــة 2

Cours Microcontrôleur
PIC

Chapitre 4
Module Convertisseur Analogique Digital (ADC)

Proposé par: N. Ghoggali

PIC16F877A
ADC

Convertisseur analogique Numérique

- Tous les exemples traités jusqu'à ce moments si des exemples conçus de tel sorte a utiliser uniquement les ports du Microcontrôleur comme des entrées (Bouton) ou sortie comme (Leds, aff 7 segment, LCD);
- Dans la réalité les données à traiter par le microcontrôleur sont fournies par ds capteurs dans la majorités des cas;
- Les signaux délivrer par les capteur sont de nature Analogique

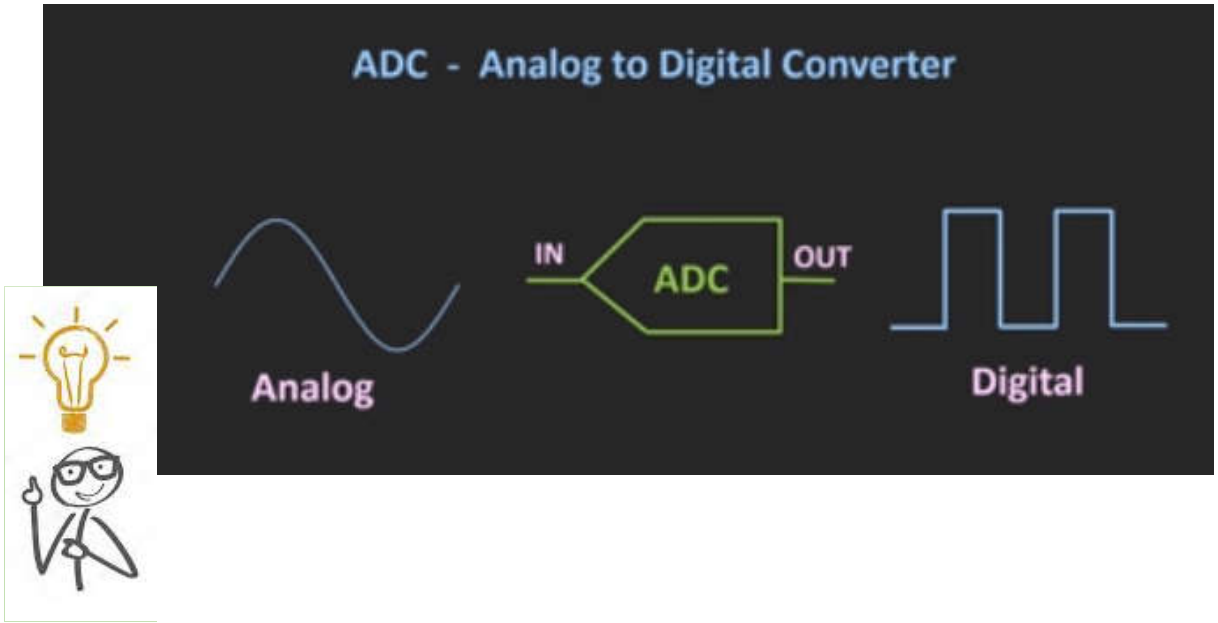
Problème

Un signal Analogique ne peut être traiter par un circuit programmable Numérique tel que le microcontrôleur PIC16F877A

PIC16F877A
ADC

Solution

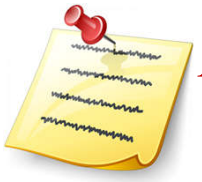
Utiliser un convertisseur Analogique Digitale pour transformer le signal Analogique en un signal Digital



PIC16F877A
ADC

ADC MODULE

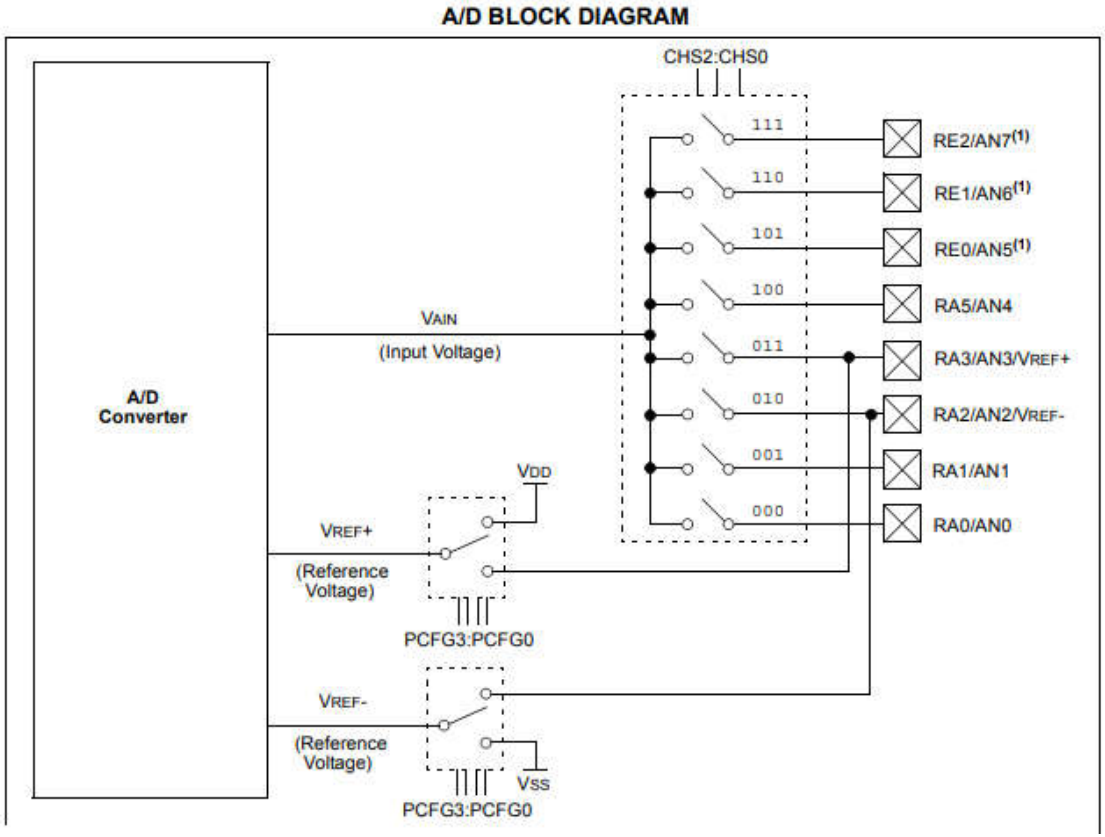
- Le PIC16F877A possède un module ADC intégré, ce module peut être utiliser et exploiter par le programmeur,
- Pour utiliser correctement le module ADC du PIC16F8777A il faut manipuler 4 registres à savoir .
    - ADCON0
    - ADCO1
    - ADRESH
    - ADRESL



Note

Pour plus de détail voir le datasheet Section 11 page 129-135

PIC16F877A
ADC



PIC16F877A
ADC

Le registre ADCON0

ADCON0							
7	6	5	4	3	2	1	0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON

ADCS2-ADCS0:A/D Conversion Clock Select bits

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	FRC (clock derived from the internal A/D RC oscillator)

CHS2-CHS0: Analog Channel Select bits

000 = Channel 0 (AN0)
001 = Channel 1 (AN1)
010 = Channel 2 (AN2)
011 = Channel 3 (AN3)
100 = Channel 4 (AN4)
101 = Channel 5 (AN5)
110 = Channel 6 (AN6)

PIC16F877A
ADC

Le registre ADCON0

GO/DONE: A/D Conversion Status bit
When ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion which is automatically cleared by hardware when the A/D conversion is complete)
0 = A/D conversion not in progress

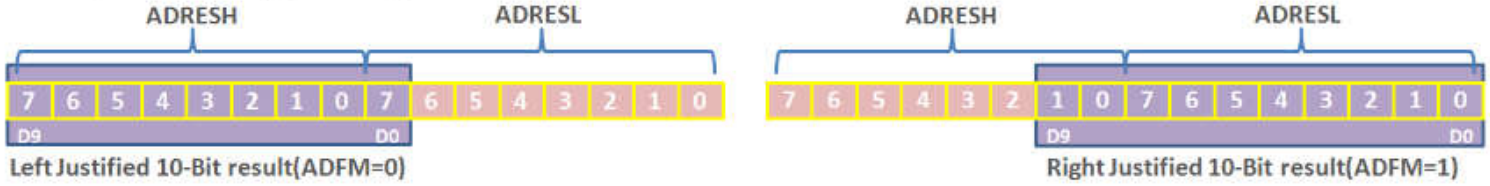
ADON: A/D On bit
1 = A/D converter module is powered up
0 = A/D converter module is shut-off and consumes no operating current

PIC16F877A
ADC

Le registre ADCON1

ADCON1							
7	6	5	4	3	2	1	0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0

ADFM: A/D Result Format Select bit
1 = Right justified. Six (6) Most Significant bits of ADRESH are read as '0'.
0 = Left justified. Six (6) Least Significant bits of ADRESL are read as '0'.



ADCS2: A/D Conversion Clock Select bit
Check ADCS1:ADCS0 of ADCON0 register.

PIC16F877A
ADC

Le registre ADCON1

PCFG3-PCFG0: A/D Port Configuration Control bits

PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C/R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	AN3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	AN3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	AN3	VSS	2/1
011x	D	D	D	D	D	D	D	D	—	—	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	AN3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1/2

A = Analog input D = Digital I/O
C/R = # of analog input channels/# of A/D voltage references

PIC16F877A
Afficheur LCD

Exemple 1: conception d'un voltmètre digital

Concevoir un voltmètre digital avec un PIC16F877A en utilisant son Module ADC et un potentiomètre (résistance variable)?

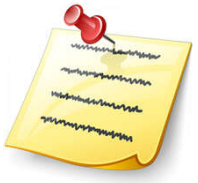
Solution

Il existe deux méthodes pour programmer le Module ADC du PIC16F877A

- Utiliser les fonction prédéfinie dans Mikroc

  ADC_Init() et

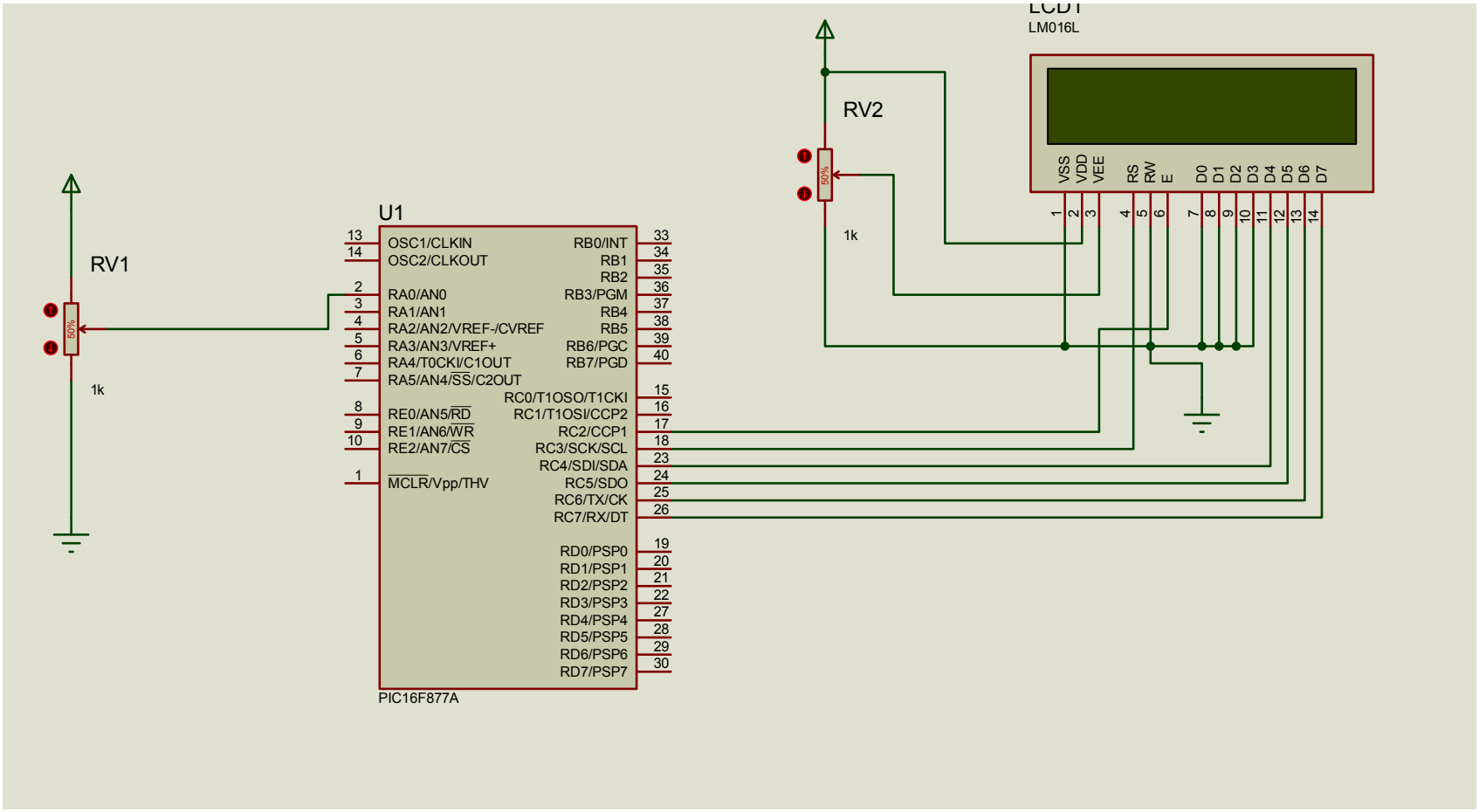
  ADC_Read(0) //Lire depuis Pin AN0 la valeur du signal analogique
- Utiliser ces propres fonctions comme par exemple ADCInit() et ADCRead(0)



Note ***le role des deux fonctions précédente s et d'initialiser les registres ADCON0, ADCON1 par les valeurs adéquate pour le bon fonctionnement du module ADC ainsi que choisir comment récupérer le résultat (Right jutified ou Left justified)***

PIC16F877A
Afficheur LCD

Solution: Partie Montage électronique



PIC16F877A
ADC

Solution: Partie Programme

- Dans cette solution nous allons utilisés les fonctions prédéfinies dans Mikroc

```
// LCD module connections
sbit LCD_RS at RC3_bit;
sbit LCD_EN at RC2_bit;
sbit LCD_D4 at RC4_bit;
sbit LCD_D5 at RC5_bit;
sbit LCD_D6 at RC6_bit;
sbit LCD_D7 at RC7_bit;

sbit LCD_RS_Direction at TRISC3_bit;
sbit LCD_EN_Direction at TRISC2_bit;
sbit LCD_D4_Direction at TRISC4_bit;
sbit LCD_D5_Direction at TRISC5_bit;
sbit LCD_D6_Direction at TRISC6_bit;
sbit LCD_D7_Direction at TRISC7_bit;
// End LCD module connections

```

PIC16F877A
ADC

Solution: Partie Programme Suite

```
void main()
{
    char txt[9]="";
    float voltage;
    char voltagetxt[6]="";
    unsigned  adcValue=0;
    Lcd_Init();                         // Initialize LCD
    Lcd_Cmd(_LCD_CLEAR);                // Clear display
    Lcd_Cmd(_LCD_CURSOR_OFF);           // Cursor off
    ADC_Init();              //Initialize the ADC module
    while(1)
    {
       adcValue = ADC_Read(0);       // Read the ADC value of channel zero  AN0
       Lcd_Out(1,1,"binaire=");                 // Write text in first row
       IntToStr( adcValue,txt);       //   adcValue =23    txt="23"
       Lcd_Out(1,9,txt);
       //conversion du binare (10bists)  vers  tension (volt)
       //  5v=5000mv        ----------> 1023         binaire
       // voltage=? ----------> adcValue        binaire
       //xvoltage=(adcValue*5.0)/1023
       Lcd_Out(2,1,"V=");
       voltage=(adcValue*5.0)/1024.0;
       FloatToStr(voltage,voltagetxt);
              // Write text in first row
       Lcd_Out(2,4,voltagetxt);                 // Write text in first row
       delay_ms(300);
       Lcd_Cmd(_LCD_CLEAR);               // Clear display
    }
}
```

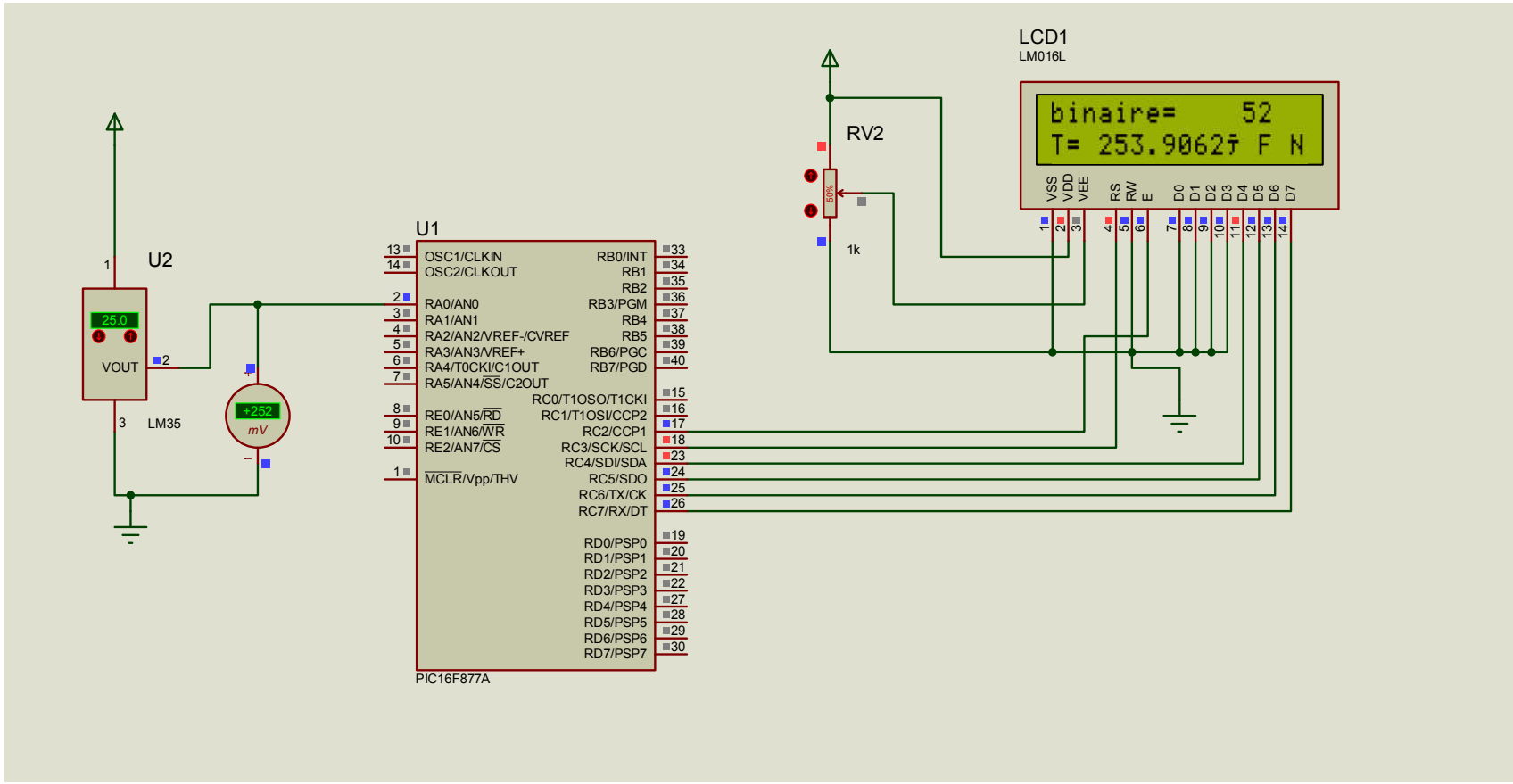
PIC16F877A
ADC

Exemple 2: conception d'un thermomètre digital

Concevoir un thermomètre digital avec un PIC16F877A en utilisant son Module ADC et un afficheur LCD 2x16?

PIC16F877A
ADC

Solution: Partie Montage électronique



PIC16F877A
ADC

Solution: Partie Programme

```
// LCD module connections
sbit LCD_RS at RC3_bit;
sbit LCD_EN at RC2_bit;
sbit LCD_D4 at RC4_bit;
sbit LCD_D5 at RC5_bit;
sbit LCD_D6 at RC6_bit;
sbit LCD_D7 at RC7_bit;

sbit LCD_RS_Direction at TRISC3_bit;
sbit LCD_EN_Direction at TRISC2_bit;
sbit LCD_D4_Direction at TRISC4_bit;
sbit LCD_D5_Direction at TRISC5_bit;
sbit LCD_D6_Direction at TRISC6_bit;
sbit LCD_D7_Direction at TRISC7_bit;
// End LCD module connections

```

PIC16F877A
ADC

Solution: Partie Programme Suite

```
void main()
{
     char txt[9]="";
     float voltage;
     char voltagetxt[6]="";
     unsigned  adcValue=0;
     Lcd_Init();                          // Initialize LCD
     Lcd_Cmd(_LCD_CLEAR);                 // Clear display
     Lcd_Cmd(_LCD_CURSOR_OFF);            // Cursor off
     ADC_Init();              //Initialize the ADC module
     while(1)
     {
        adcValue = ADC_Read(0);       // Read the ADC value of channel zero  AN0
        Lcd_Out(1,1,"binaire=");                  // Write text in first row
        IntToStr( adcValue,txt);       //   adcValue =23    txt="23"
        Lcd_Out(1,9,txt);
        //conversion du binaire (10bists)  vers  tension (volt)
        //  5v=5000mv        ----------> 1023         binaire
        // voltage=? ----------> adcValue        binaire
        //xvoltage=(adcValue*5.0)/1023
        Lcd_Out(2,1,"T=");
        voltage=(adcValue*5000.0)/1024.0;
        FloatToStr(voltage,voltagetxt);
                // Write text in first row
        Lcd_Out(2,4,voltagetxt);                  // Write text in first row
        delay_ms(300);
        Lcd_Cmd(_LCD_CLEAR);               // Clear display
     }
}
```

PIC16F877A
ADC

Utilisation des fonctions utilisateur

Une deuxième solution consiste à utiliser la section 129-135 datasheet pour définir nos propres fonctions ADCInit() et ADCRead()

```
void ADCInit()
{
  ADCON0=0b00000000;      //0x00;
  ADCON1=0b10000000;      //0x80;
}

int ADCRead(int adcChannel)
{
   ADCON0 = 0b00000001;   //ADCON0.ADON=1
   delay_ms(1000);                    //Acquisition Time(Wait for Charge Hold Capacitor to get charged )
   ADCON0.GO_DONE=1;                            // Start ADC conversion
   while(ADCON0.GO_DONE==1);                 // attendre jusqu'a done =1 => GO =0
                                  // GO_DONE bit will be cleared once conversion is complete
   return((ADRESH<<8) | ADRESL);   // return right justified 10-bit result
}
```

PIC16F877A
ADC

Utilisation des fonctions utilisateur

Une deuxième solution consiste à utiliser la section 129-135 datasheet pour définir nos propre fonctions

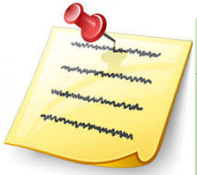
```
int main()
{
    int adcValue=0;
    float adcval;

    ADCInit();              //Initialize the ADC module

    while(1)
    {
        adcValue = ADCRead(0);       // Read the ADC value of channel zero
        //PORTB = (adcValue & 0xff);    //Adc value displayed on LEDs connected to PORTB,PORTD
        //PORTD = (adcValue>>8) & 0x03; // PORTB will display lower 8-bits and PORTD higher 2-bits
        adcval=(adcValue*5.0)/1023.0;
        FloatToStr(adcval,volttxt);
        LCD_Out(1,8,volttxt);
    }
}
```

PIC16F877A
ADC

Solution: Partie Programme Suite

```
void main()
{
     char txt[9]="";
     float voltage;
     char voltagetxt[6]="";
     unsigned  adcValue=0;
     Lcd_Init();                          // Initialize LCD
     Lcd_Cmd(_LCD_CLEAR);                 // Clear display
     Lcd_Cmd(_LCD_CURSOR_OFF);            // Cursor off
     ADC_Init();              //Initialize the ADC module
     while(1)
     {
        adcValue = ADC_Read(0);       // Read the ADC value of channel zero  AN0
        Lcd_Out(1,1,"binaire=");                  // Write text in first row
        IntToStr( adcValue,txt);       //   adcValue =23    txt="23"
        Lcd_Out(1,9,txt);
        //conversion du binaire (10bists)  vers  tension (volt)
        //  5v=5000mv        ----------> 1023         binaire
        // voltage=? ----------> adcValue        binaire
        //xvoltage=(adcValue*5.0)/1023
        Lcd_Out(2,1,"T=");
        voltage=(adcValue*5000.0)/1024.0;
        FloatToStr(voltage,voltagetxt);
                // Write text in first row
        Lcd_Out(2,4,voltagetxt);                  // Write text in first row
        delay_ms(300);
        Lcd_Cmd(_LCD_CLEAR);               // Clear display
     }
}
```

PIC16F877A
ADC



Note

Pour pouvoir exécuter la simulation des fichiers ADC,rar et Thermomettre,rar il faut que vous changer la dans les propriétés PIC16F877A dans ISIS pour choisir l'endroit correct ou le .hex est localiser