CAPÍTULO 3

3. DISEÑO DEL SOFTWARE.

La programación del PIC se realizó en lenguaje C, usando el compilador mikroC PRO for PIC. Este compilador se utilizó, ya que tiene una versión demo muy buena, además tiene una variedad de características tales como: IDE fácil de usar, un código muy compacto y eficiente, muchas librerías y la documentación completa de ayuda; además, en su sitio web se puede encontrar numerosos ejemplos.

La programación en lenguaje C es muy flexible, fácil de entender y ejecutar, pero se debe tener en cuenta que se tiene un límite del programa ejecutable de 2K palabras en este demo, pero es suficiente para esta aplicación.

3.1 Programa del Microcontrolador.

Las principales funciones del microcontrolador son: adquirir por medio del A/D y también, mostrar mensajes de pesos y voltaje en el LCD.

El microcontrolador requiere ser configurado de acuerdo con los recursos a utilizar, que para este caso son los pines del LCD y el conversor análogo

digital. Todos estos recursos se configuran con instrucciones especiales en el compilador, que determinan en que pin van a ser utilizados y el protocolo a seguir.

Las tareas que debe realizar el microcontrolador son las siguientes:

- Configuración de recursos.
- Presentación de mensajes.
- Adquisición de datos.
- Validaciones.
- Filtrado digital.
- Presentación en la pantalla LCD.

Estas tareas en la programación se las diseñó como subrutinas o partes del programa principal que se ejecuta. Más adelante se explica cómo se encuentran estructuradas estas subrutinas.

3.1.1 Programa Principal.

El programa principal cuenta con los siguientes procesos:

1. Configuración de los recursos del microcontrolador donde se configuran los pines de los periféricos, como por ejemplo: la pantalla LCD y el conversor A/D.
2. Encendido de la pantalla LCD y mensajes de inicio.
3. Inicialización del conversor análogo digital.
4. Adquisición y estabilización de la señal de entrada en el convertidor.

5. Conversión del valor del voltaje de entrada en gramos, por medio de una ecuación matemática.
6. Validación del peso.
7. Mostrar en la pantalla LCD el voltaje y el peso.

Esta sección del programa es la principal donde convergen los principales procesos y subrutinas; aquí se inicia con tres subprogramas importantes para que todos los periféricos y puertos funcionen correctamente. La subrutina Inicio_Var, es la encargada de la configuración de los puertos y registros del microcontrolador. Luego se tiene el subprograma Mensajes_Ini con el cual se presenta un mensaje de inicio en el LCD, y seguido se tiene el inicio del conversor A/D.

La señal que proviene del filtro e ingresa al microcontrolador debe ser procesada antes de que este valor pueda ser visualizado en la pantalla LCD. Para esto se debe tener en consideración los siguientes correctivos:

- La señal debe ser filtrada, lo cual se logra tomando un conjunto de muestras y sacando el promedio de las mismas. La función encargada de esto se llama Filtrado_Digital; en este subprograma se realiza un muestreo de la señal de entrada, para obtener un mejor valor a la salida del peso en gramos. Además, aquí se realiza la conversión del voltaje de entrada por medio de una ecuación que se obtuvo en la prueba de funcionamiento de la celda de carga, para luego mostrar el peso en gramos y el voltaje de entrada al PIC.
- Una de las validaciones que se debe de tener en cuenta además de que un peso no puede ser negativo y tampoco tiene que sobrepasar el peso máximo permitido por la celda de carga.

El programa principal del microcontrolador con sus subrutinas se resume en el diagrama de flujo de la figura 3.1.

3.1.1.1 Diagrama de Flujo del Programa Principal.

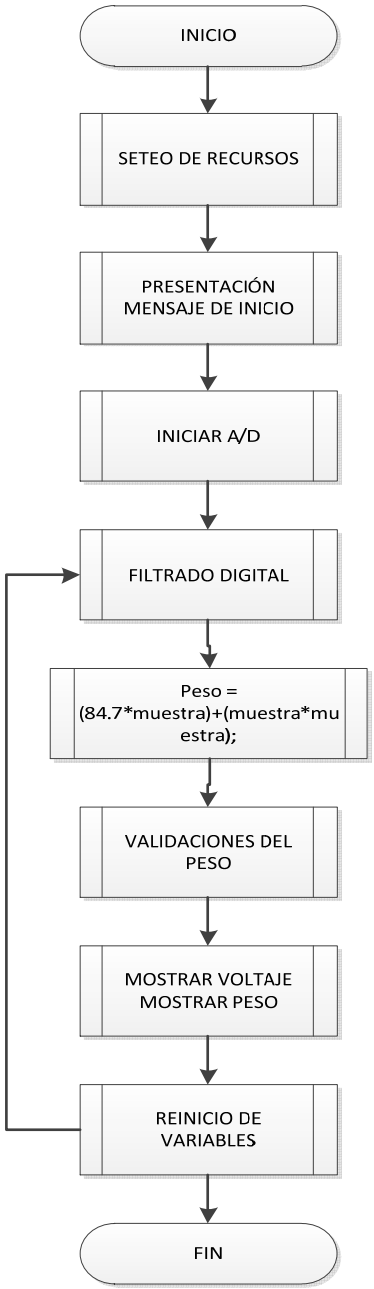

Figura 3.1 Diagrama de Flujo del Programa Principal.

3.1.1.2 Codificación del Programa Principal en C.

```
void main(void){                                    //Inicio del programa principal
InitMain();                                         //Fijar puertos y registros
Mensaje_Ini();                                      //Mensaje de Inicio
ADC_Init();                                         //Inicialización del conversor
while(1){
muestra =  Filtrado_Digital();                      //Muestreo
PesoB = (84.7*muestra)+(muestra*muestra);//Ecuación del peso
pru =((float)ADC_Read(1)*500)/1023;                 //Voltaje de entrada
if( ban == 1 ){                                     //Validación de inicio
PesoM=PesoB;                                        //Peso neto = Peso Bruto
}
else{
PesoTotal=PesoB-PesoM;                              //Peso total=Peso bruto–Peso
Neto
if(pru <= 0.10)                                     //Validación del voltaje
{PesoTotal=0;}
}
Delay_ms(15);
Mostrar_Voltaje(pru);                               //Mostrar el valor del voltaje
Mostrar_Peso(Pesototal);                            //Mostrar el valor del peso en
gramos
PesoTotal=0;                                        //Reinicio de variables
PesoM=0;
ban=0;
}
}                                                   //Fin del programa principal
```

3.1.2 Programación para el convertidor A/D.

En la programación del convertidor analógico digital se debe primero reiniciar los puertos como entradas o salidas; en este trabajo, el puerto A (PORTA) se configuró como entrada; luego se procedió a configurar los registros de control del PIC. El funcionamiento del convertidor A/D está bajo el control de los registros ADCON0, registro de control 0; y ADCON1, registro de control 1.

El PIC 16F877A tiene un convertidor analógico digital de 10 bits. La ecuación que utiliza el convertidor del PIC, empleada en el cálculo de la conversión es la siguiente:

$$\text{Binario} = \frac{(2^{n} - 1)\text{x Vint}}{\text{Vref}}$$

$$\text{Vint} = \frac{5\text{x Binario}}{1023}$$

Este voltaje Vint, es el voltaje resultante que está presente en el pin del microcontrolador, el cual va a formar parte de la ecuación para el cálculo del peso en gramos. Esta ecuación forma parte de la subrutina Filtrado Digital. Para la obtención de esta ecuación, se realiza una prueba de funcionamiento con la celda de carga, que se estudiará en el capítulo siguiente.

3.1.2.1 Filtrado Digital.

La principal función de este proceso es adquirir, por medio del convertidor A/D y realizar un registro de la señal proveniente del sensor, para obtener un mejor resultado final; es decir, se realiza un muestreo de la señal que ingresa al PIC.

Este muestreo fue realizado con un cierto intervalo de tiempo; se sacaó un promedio general para mostrar una mejor estabilidad del peso a la salida; esta función se realizó, porque la señal no era estable y esto causaba que en la pantalla LCD, el valor del peso variara de igual manera, haciendo que esto afecte el resultado final.

El muestreo se realiza en cuestión de milisegundos; en este caso el microcontrolador 16F877A, tiene un tiempo entre tomar un dato y el siguiente de 20u seg; en la codificación del programa se escogió un retardo de 4ms, que es el tiempo que se utiliza para dar entre muestra y muestra un intervalo al conversor AD, para que adquiera un nuevo valor y no se lea el mismo. En el siguiente apartado se muestra el diagrama de flujo y la codificación de esta subrutina (Ver figura 3.2).

3.1.2.1.1 Diagrama de Flujo de Filtrado Digital.

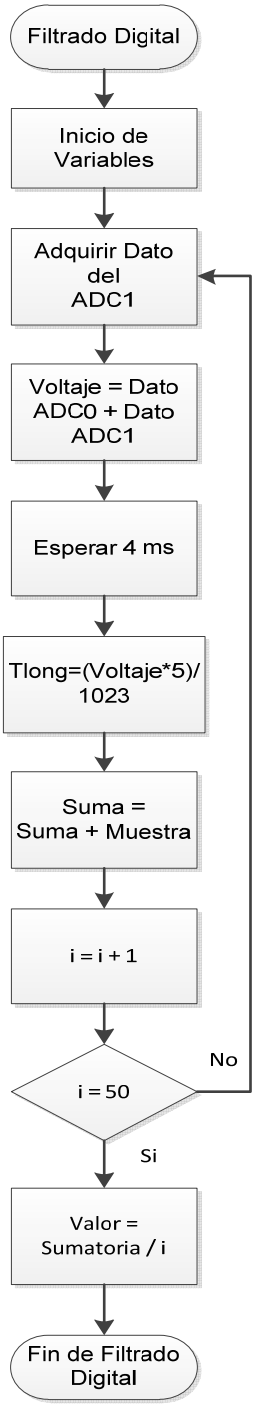

Figura 3.2 Diagrama de Flujo del Muestreo y Filtrado de la Señal.

3.1.2.1.2 Codificación de Muestreo y Filtrado en C.

```
float Filtrado_Digital (void){              // Inicio de la función

int i=0;                                    // Inicialización de variables
float valor=0, voltaje=0;
float  suma =0;

for(i=0;i<50;i++){
voltaje=((float)ADC_Read(1)*5)/1023;  // Muestreo de entrada al canal ADC
Delay_ms(4);                                // Retardo de 4ms
suma += tlong;                              // Suma de todas las muestras
}
valor = suma/50;                            // Promedio de todos los valores
return valor;                               // Retorno del valor promedio
}                                           // Fin de la función
```

3.1.3 Programación para visualizar en el LCD.

Para visualizar los valores y mensajes en este proyecto, lo primero que se hace es configurar el puerto B (PORTB), para la comunicación, luego se tiene que fijar el puerto B con el módulo de conexiones de los pines de la pantalla; esto se lo hace ligando el pin del microcontrolador con el pin de la pantalla LCD. Lo que muestra este proyecto es el mensaje de inicio cuando se enciende el circuito (Ver figura 3.3).

Figura 3.3 Mensaje de Inicio.

Este compilador tiene incorporadas algunas funciones para su uso. Haciendo que la programación en la comunicación del PIC con la pantalla LCD, sea mucho más sencilla; por ejemplo: se tiene funciones donde el texto que se envía desde el PIC, se desplace de izquierda a derecha con una sola función. A continuación se muestra el módulo de conexiones y la codificación que reglamentariamente tiene que estar en el programa.

3.1.3.1 Codificación para visualizar el LCD.

```
// Conexiones del Módulo LCD
sbit LCD_RS at RB4_bit;
sbit LCD_EN at RB5_bit;
sbit LCD_D4 at RB0_bit;
sbit LCD_D5 at RB1_bit;
sbit LCD_D6 at RB2_bit;
sbit LCD_D7 at RB3_bit;
sbit LCD_RS_Direction at TRISB4_bit;
```

```
sbit LCD_EN_Direction at TRISB5_bit;
sbit LCD_D4_Direction at TRISB0_bit;
sbit LCD_D5_Direction at TRISB1_bit;
sbit LCD_D6_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB3_bit;


void Mensaje_Ini(void){
  Lcd_Init();                          // Inicio de subprograma
  Lcd_Cmd(_LCD_CLEAR);                 // Borrar LCD
  Lcd_Cmd(_LCD_CURSOR_OFF);            // Apagar cursor
  Lcd_Out(1,4,"Iniciando..");          // Mensaje de inicio
  Delay_ms(2000);                      // Esperar 2 seg.
  Lcd_Cmd(_LCD_CLEAR);                 // Borrar LCD
  return;
 }
```

3.1.3.2 Programación para mostrar el voltaje.

Este subprograma muestra el valor del voltaje, que está presente en la entrada del microcontrolador, donde se realiza la conversión. Este valor también se muestra en la pantalla LCD. Este mensaje se visualiza en la primera línea del LCD (Ver figura 3.4).

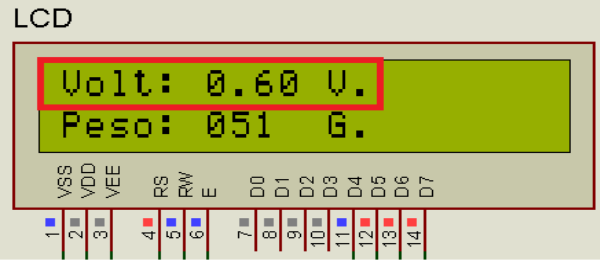

Figura 3.4 Visualización del Voltaje.

A continuación se muestra la codificación de esta subrutina:

```
void Mostrar_Voltaje(unsigned int y)          // Inicio de la Función
 {

  unsigned char pt=0;                         // Variable a imprimir
  Delay_ms(15);                               // Retardo
  Lcd_Out(1,1,"Volt: ");                      // Escribir Volt
  pt=(y/100)%10;                              // Extraer centenas
  Lcd_Chr(1,7,48+pt);                         // Escribir el ASCII
  Lcd_Out(1,8,".");                           // Escribimos el punto
  pt=(y/10)%10;                               // Extraer decenas
  Lcd_Chr(1,9,48+pt);                         // Escribir el ASCII
  pt=(y/1)%10;                                // Extraer unidades
  Lcd_Chr(1,10,48+pt);                        // Escribir el ASCII
  Lcd_Chr(1,11,32);                           // Escribir el ASCII
  Lcd_Chr(1,12,86);                           // Escribir el ASCII
  Lcd_Chr(1,13,46);                           // Escribir el ASCII

 }                                            // Fin de la Función
```

3.1.3.3 Programación para mostrar el peso.

Este subprograma cumple la misma función que Mostrar_Voltaje; indica el valor del peso en gramos en la segunda línea del LCD (figura 3.5). La diferencia con la función Mostrar_Voltaje es que, tiene una limitación en cuanto al peso: si es mayor que 405 gramos entonces, se imprime un mensaje indicando que es el peso máximo. Esta validación se realiza en esta sección del programa.

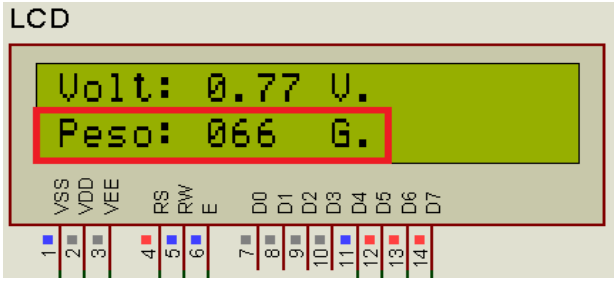

Figura 3.5 Visualización del Peso.

La codificación de este proceso es la siguiente:

```
void Mostrar_Peso(unsigned int x){              // Inicio de la Función

unsigned char ch=0;
if(x > 405){                                    // Límite del peso
Lcd_Out(1,1,"    --PESO--  ");                  // Mostrar Mensaje
Lcd_Out(2,1,"  ---MAXIMO---  ");
}
else{
Lcd_Out(2,1,"Peso: ");                          // Escribir peso
```

```
ch=(x/100)%10;                          // Extraer centenas
Lcd_Chr(2,7,48+ch);                     // Escribir el ASCII
ch=(x/10)%10;                           // Extraer decenas
Lcd_Chr(2,8,48+ch);                     // Escribir el ASCII
ch=(x/1)%10;                            // Extraer unidades
Lcd_Chr(2,9,48+ch);                     // Escribir el ASCII
Lcd_Chr(2,10,32);                       // Escribir el ASCII
Lcd_Chr(2,11,32);                       // Escribir el ASCII
Lcd_Chr(2,12,71);                       // Escribir el ASCII
Lcd_Chr(2,13,46);                       // Escribir el ASCII
 }

}                                       // Fin de la Función
```

3.1.4 Programa completo.

```
// INICIO DEL PROGRAMA

sbit LCD_RS at RB4_bit;
sbit LCD_EN at RB5_bit;
sbit LCD_D4 at RB0_bit;
sbit LCD_D5 at RB1_bit;
sbit LCD_D6 at RB2_bit;
sbit LCD_D7 at RB3_bit;
sbit LCD_RS_Direction at TRISB4_bit;
sbit LCD_EN_Direction at TRISB5_bit;
sbit LCD_D4_Direction at TRISB0_bit;
sbit LCD_D5_Direction at TRISB1_bit;
sbit LCD_D6_Direction at TRISB2_bit;
```

```
sbit LCD_D7_Direction at TRISB3_bit;

float muestra=0,PesoM=0,PesoB=0,tlong=0;
 unsigned int PesoTotal=0,ban=1,pru=0;

void Inicio_Var(void) {
PORTA  = 0xFF;
PORTB  = 0xFF;
INTCON = 0;
TRISA  = 0xFF;
TRISB  = 0;
ADCON1 = 0x88;
ADCON0 = 0x88;
return;
}

void Mensaje_Ini(void){
  Lcd_Init();
  Lcd_Cmd(_LCD_CLEAR);
  Lcd_Cmd(_LCD_CURSOR_OFF);
  Lcd_Out(1,4,"Iniciando..");
  Delay_ms(2000);
  Lcd_Cmd(_LCD_CLEAR);
  return;
 }

void Mostrar_Voltaje(unsigned int y)
 {
  unsigned char pt=0;
  Delay_ms(15);
```

```
  Lcd_Out(1,1,"Volt: ");
  pt=(y/100)%10;
  Lcd_Chr(1,7,48+pt);
  Lcd_Out(1,8,".");
  pt=(y/10)%10;
  Lcd_Chr(1,9,48+pt);
  pt=(y/1)%10;
  Lcd_Chr(1,10,48+pt);
  Lcd_Chr(1,11,32);
  Lcd_Chr(1,12,86);
  Lcd_Chr(1,13,46);
 }

void Mostrar_Peso(unsigned int x){
unsigned char ch=0;
if(x > 405){
Lcd_Out(1,1,"    --PESO--  ");
Lcd_Out(2,1,"  ---MAXIMO---  ");
}
else{
Lcd_Out(2,1,"Peso: ");
ch=(x/100)%10;
Lcd_Chr(2,7,48+ch);
ch=(x/10)%10;
Lcd_Chr(2,8,48+ch);
ch=(x/1)%10;
Lcd_Chr(2,9,48+ch);
Lcd_Chr(2,10,32);
Lcd_Chr(2,11,32);
Lcd_Chr(2,12,71);
```

```
Lcd_Chr(2,13,46);
 }
}

float Filtrado_Digital (void){
int i=0;
float valor=0, voltaje=0;
float  suma =0;
for(i=0;i<50;i++){
voltaje=((float)ADC_Read(1)*5)/1023;
Delay_ms(4);
suma += tlong;
}
valor = suma/50;
return valor;
}
void main(void){
InitMain();
Mensaje_Ini();
ADC_Init();
while(1){
muestra =  Filtrado_Digital();
PesoB = (84.7*muestra)+(muestra*muestra);
pru =((float)ADC_Read(1)*500)/1023;
if( ban == 1 ){
PesoM=PesoB;
}
else{
PesoTotal=PesoB-PesoM;
if(pru <= 0.10)
```

```
{PesoTotal=0;}
}
Delay_ms(15);
Mostrar_Voltaje(pru);
Mostrar_Peso(Pesototal);
PesoTotal=0;
PesoM=0;
ban=0;
}
}
// FIN DEL PROGRAMA
```