ANEXO II. PROGRAMACIÓN DEL INTEGRADO PIC16F87XA

1. FICHEROS DE CABECERA

```
#include <16F876A.h>
#device adc=8

#FUSES NOWDT                    //No Watch Dog Timer
#FUSES HS                       //High speed Osc (> 4mhz)
#FUSES PUT                      //Power Up Timer
#FUSES NOPROTECT                //Code not protected from reading
#FUSES NODEBUG                  //No Debug mode for ICD
#FUSES BROWNOUT                 //Reset when brownout detected
#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=4000000)
#use rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_C7,bits=8)
```

2. CÓDIGO FUENTE

```
// POWER MONITOR V1.0
// 16 de Junio de 2010
//-------------------------------------------------------------------------------------
// Descripción:
// Monitor DC que cumple la norma MIL-PRF-24021K TABLA III y Figura 9
// Siempre que se cumpla dicha norma, el relé de salida está activado,
// de lo contrario el relé se desactiva y el tipo de error es informado con los LEDs:
// LED1 -> Tensión baja
// LED2 -> La tensión esta entre 10 y 24 y ha superado 2.5s entre estos niveles
// LED3 -> La tensión esta entre 30 y 50v y ha superado los tiempos definidos en la tabla
// LED4 -> Tensión alta
// LEDs 1 al 4 -> Rizado máximo admitido superado (los 4 encendidos a la vez)
//-------------------------------------------------------------------------------------

#include <pwrmon00.h>
#define LoadVMAX 250        //50v    | Por encima de esta tensión SIEMPRE da error
#define Load29V 133         //29v    | Entre estas 2 tensiones no hay error, fuera
#define Load24V 111         //24v    | se amplifica la tabla de abajo
#define LoadVMIN 51         //10v    | Por debajo de esta tensión SIEMPRE da error
                            // Sólo se usa cuando el relé fue activado
#define ResetVMAX 137       //29.9v  | Rango de tensiones para encendido o rearme
#define ResetVMIN 105       //23v
                            // Rango a cumplir antes del encendido o después de un error
                            de las anteriores
#define RELE PIN_B4
#define LED1 PIN_B0
#define LED2 PIN_B1
#define LED3 PIN_B2
#define LED4 PIN_B3
#define BUTTON PIN_C0
#define MAXRIPPLE 23        // Rizado máximo permitido (4Vpp)
#define NOERROR 0           // LEDs apagados
#define LOWVERROR 1         // LED1
#define LOWVERRORT 2        // LED2
#define HIGHVERRORT 3       // LED3
#define HIGHVERROR 4        // LED4
#define RIPPLEERROR 5       // Los 4 LEDs encendidos
                            // Lista de errores posibles para la variable ERROR
#use fast_io(B)

// Tabla que cumple la curva A de la Figura 9 de la MIL-PRF-24021

const int8 Limites[]={240, 145, 105, 82, 68, 57, 49, 43, 38, 34, 31, 29, 26, 24, 23, 22, 20, 19,
          17, 16, 14, 14, 14, 13, 13, 12, 12, 11, 11, 10, 10, 9, 9, 9, 9, 9, 9,
          9, 8, 8, 8, 8, 8, 8, 8, 7, 7, 7, 7, 7, 7, 7, 7, 6, 6, 6, 6, 6, 6, 6,
          6, 6, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
          4, 4, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3};

int8 ContInt = 5;               // Se usa para multiplicar el tiempo del temporizador HW x5
int1 TOMARMUESTRA = FALSE;      // SI = TRUE se ha producido 5 veces la interrupción
                                WH (ContInt=5)
int8 ERROR = 0;                 // Almacena el tipo de error
int1 ESTADO_RELE = 0;           // Diferencia los 2 estados de funcionamiento del Power
                                monitor, avión conectado o desconectado
```

```
int8 MUESTRA = 0;          // Almacena el valor de la conversión del ADC
int8 CONTADOR_MAL = 0;
int8 CONTADOR_OK = 0;
int16 CONTADOR_LOW = 0;    // Contador para 2.5s (500 x 5ms)

// Interrupcion del Timer configurado para 1ms, se activa un flag cuando ha ocurrido 5 veces (5ms).
// Se toman muestras del ADC cada 5ms.

#int_RTCC
void  RTCC_isr(void)
{
  if(--ContInt==0)
  {
    TOMARMUESTRA = TRUE;
    ContInt = 5;
  }
}

// Toma una muestra del nivel de rizado de la tensión y se compara con el valor máximo admitido
// Devuelve un 0 si se supera dicho valor.

int1 RizadoOK(){

int8 MUESTRARIPPLE;

  set_adc_channel(1);
  delay_us(10);
  MUESTRARIPPLE=read_adc();

  if(MUESTRARIPPLE>MAXRIPPLE)
    return 0;
  else
    return 1;
}

void main()
{
  setup_adc_ports(AN0_AN1_AN3);
  setup_adc(ADC_CLOCK_DIV_2);
  setup_spi(SPI_SS_DISABLED);
  setup_timer_0(RTCC_INTERNAL|RTCC_DIV_4);       // Temporizador a 1ms
  setup_timer_1(T1_DISABLED);
  setup_timer_2(T2_DISABLED,0,1);
  setup_comparator(NC_NC_NC_NC);
  setup_vref(FALSE);
  enable_interrupts(INT_RTCC);           // Interrupción por desbordamiento del timer
  set_tris_B( 0b11100000 );
  output_B(0x00);

// Se espera a pulsar el botón ON/RESET
```

```
  delay_ms(3000);
  enable_interrupts(GLOBAL);
  while(!input(BUTTON))
  {}
  while(input(BUTTON))
  {}

  while(TRUE)
  {
    if(TOMARMUESTRA == TRUE)      // Entramos sólo si desbordo el timer 5 veces (5ms)
    {
      TOMARMUESTRA = FALSE;       // Reseteamos el contador para volver a contar 5ms
      set_adc_channel(0);
      delay_us(10);
      MUESTRA = read_adc();

      if(ESTADO_RELE == 0)        // Estado inicial o después de un error
      {
        if((MUESTRA > ResetVMIN)&&(MUESTRA < ResetVMAX)&&(RizadoOK()))

// Condiciones de la TablaIII part1, MIL-PRF24021

        {
          output_high(RELE);
          ESTADO_RELE = 1;
          ERROR = NOERROR;
          CONTADOR_OK = 0;
          CONTADOR_MAL = 0;
          CONTADOR_LOW = 0;
          output_low(LED1);
          output_low(LED2);
          output_low(LED3);
          output_low(LED4);
        }
        else if(MUESTRA <= ResetVMIN)        // Si alguna de las condiciones del if anterior no
                                             se cumplió, averiguamos cual
          ERROR = LOWVERROR;
        else if(MUESTRA >= ResetVMAX)
          ERROR = HIGHVERROR;
        else
          ERROR = RIPPLEERROR;
      }
      else                  // CON RELÉ ACTIVADO, Condiciones de la Tabla III parte 2.
      {
        if(!RizadoOK())
          ERROR = RIPPLEERROR;
        if(MUESTRA >= LoadVMAX)
          ERROR = HIGHVERROR;
        else if(MUESTRA < LoadVMIN)
          ERROR = LOWVERROR;
        else if((MUESTRA > LoadVMIN)&&(MUESTRA < Load24V))
        {
          CONTADOR_LOW++;
```

```
      if(CONTADOR_LOW > 488)          // Duración máxima permitida (2.5s) para
                                      tensiones entre 24 y 10v.
        ERROR = LOWVERRORT;
    }
    else if((MUESTRA > Load24V)&&(MUESTRA < Load29V))     // Rango de tensiones
                                                          normal
    {
      CONTADOR_LOW = 0;                 // Se resetean contadores de error
      CONTADOR_MAL = 0;
      ERROR = NOERROR;
    }
    else if((MUESTRA>=Load29V)&&(MUESTRA<LoadVMAX))
    {
      if(++CONTADOR_MAL > Limites[MUESTRA-Load29V])
        ERROR = HIGHVERRORT;
    }
  }
  if(ERROR!=NOERROR)
  {
    output_low(RELE);               // Apagamos el relé ya que se produjo un error
    ESTADO_RELE = 0;                // Pasamos el estado de relé desactivado
    switch(ERROR){                  // Averiguamos que tipo de error dio y encendemos el
                                    LED para dicho error
      case HIGHVERROR:
        output_high(LED4);
        output_low(LED1);
        output_low(LED2);
        output_low(LED3);
      break;
      case HIGHVERRORT:
        output_high(LED3);
        output_low(LED1);
        output_low(LED2);
        output_low(LED4);
      break;
      case LOWVERRORT:
        output_high(LED2);
        output_low(LED1);
        output_low(LED3);
        output_low(LED4);
      break;
      case LOWVERROR:
        output_high(LED1);
        output_low(LED2);
        output_low(LED3);
        output_low(LED4);
      break;
      case RIPPLEERROR:
        output_high(LED1);
        output_high(LED2);
        output_high(LED3);
        output_high(LED4);
      break;
    }
```

```
            while(!input(BUTTON))
            {}
            while(input(BUTTON))            // Una vez producido un error se necesita un rearme
            {}
            ERROR = NOERROR;
            CONTADOR_OK = 0;
         }
      }
   }
}
```

3. COMPILACIÓN

CCS PCM C Compiler, Version 4.065, 38112 04-jun-10 17:45

Filename: C:\Documents and Settings\mrflower-laptop\Mis documentos\Proyectos\PowerMon\pwrmon00\pwrmon00.lst

ROM used: 544 words (7%)
Largest free fragment is 2048
RAM used: 24 (7%) at main() level
25 (7%) worst case
Stack: 2 worst case (1 in main + 1 for interrupts)

```
*
0000:  MOVLW  00
0001:  MOVWF  0A
0002:  GOTO   0DE
0003:  NOP
0004:  MOVWF  7F
0005:  SWAPF  03,W
0006:  CLRF   03
0007:  MOVWF  21
0008:  MOVF   0A,W
0009:  MOVWF  20
000A:  CLRF   0A
000B:  MOVF   04,W
000C:  MOVWF  22
000D:  MOVF   77,W
000E:  MOVWF  23
000F:  MOVF   78,W
0010:  MOVWF  24
0011:  MOVF   79,W
0012:  MOVWF  25
0013:  MOVF   7A,W
0014:  MOVWF  26
0015:  MOVF   7B,W
0016:  MOVWF  27
0017:  BCF    03.7
0018:  BCF    03.5
0019:  BTFSS  0B.5
001A:  GOTO   01D
001B:  BTFSC  0B.2
001C:  GOTO   030
001D:  MOVF   22,W
001E:  MOVWF  04
001F:  MOVF   23,W
0020:  MOVWF  77
0021:  MOVF   24,W
0022:  MOVWF  78
0023:  MOVF   25,W
0024:  MOVWF  79
0025:  MOVF   26,W
0026:  MOVWF  7A
0027:  MOVF   27,W
0028:  MOVWF  7B
0029:  MOVF   20,W
```

```
002A:  MOVWF  0A
002B:  SWAPF  21,W
002C:  MOVWF  03
002D:  SWAPF  7F,F
002E:  SWAPF  7F,W
002F:  RETFIE
0030:  BCF    0A.3
0031:  BCF    0A.4
0032:  GOTO   0A3

.....        // POWER MONITOR V1.0
.....        // 16 de Abril 2009
.....        //-------------------------------------------------------------------------------
.....        // Descripción:
.....        // Monitor DC que cumple la norma MIL-PRF-24021K TABLA III y Figura 9
.....        // Siempre que se cumpla dicha norma, el relé de salida está activado,
.....        // de lo contrario el relé se desactiva y el tipo de error es informado
.....        // con los LEDs:
.....        // LED1 -> Tensión baja
.....        // LED2 -> La tensión esta entre 10 y 24 y ha superado 2.5s entre estos niveles
.....        // LED3 -> La tensión esta entre 30 y 50v y ha superado los tiempos definidos
                en la tabla
.....        // LED4 -> Tensión alta
.....        // LEDs 1 al 4 -> Rizado máximo admitido superado (los 4 encendidos a la vez)
.....        //-------------------------------------------------------------------------------
.....
..... #include <pwrmon00.h>
..... #include <16F876A.h>
..... //////// Standard Header file for the PIC16F876A device ////////////////
..... #device PIC16F876A
..... #list
.....
..... #device adc=8
.....
..... #FUSES NOWDT                    //No Watch Dog Timer
..... #FUSES HS                       //High speed Osc (> 4mhz)
..... #FUSES PUT                      //Power Up Timer
..... #FUSES NOPROTECT                //Code not protected from reading
..... #FUSES NODEBUG                  //No Debug mode for ICD
..... #FUSES BROWNOUT                 //Reset when brownout detected
..... #FUSES NOLVP                    //No low voltage programming, B3(PIC16) or
                                        B5(PIC18) used for I/O
..... #FUSES NOCPD                    //No EE protection
..... #FUSES NOWRT                    //Program memory not write protected
.....
..... #use delay(clock=4000000)
*
00AC:  MOVLW  32
00AD:  MOVWF  04
00AE:  BCF    03.7
00AF:  MOVF   00,W
00B0:  BTFSC  03.2
00B1:  GOTO   0C0
00B2:  MOVLW  01
```

```
00B3:  MOVWF  78
00B4:  CLRF   77
00B5:  DECFSZ 77,F
00B6:  GOTO   0B5
00B7:  DECFSZ 78,F
00B8:  GOTO   0B4
00B9:  MOVLW  4A
00BA:  MOVWF  77
00BB:  DECFSZ 77,F
00BC:  GOTO   0BB
00BD:  GOTO   0BE
00BE:  DECFSZ 00,F
00BF:  GOTO   0B2
00C0:  BCF    0A.3
00C1:  BCF    0A.4
00C2:  GOTO   144 (RETURN)
..... #use rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_C7,bits=8)
.....
.....
..... #define LoadVMAX 250     // 50v Por encima de esta tensión SIEMPRE da error
..... #define Load29V 133      // 29v Entre estas 2 tensiones no hay error, fuera
..... #define Load24V 111      // 24v se amplifica la tabla de abajo
..... #define LoadVMIN 51      // 10v Por debajo de esta tensión SIEMPRE da error
                               // Sólo se usan cuando el relé fue activado.
..... #define ResetVMAX 137    // 29.9v Rango de tensiones para encendido o rearme
..... #define ResetVMIN 105    // 23v
                               // Rango a cumplir antes del encendido o después de
                               un error de las anteriores
..... #define RELE PIN_B4
..... #define LED1 PIN_B0
..... #define LED2 PIN_B1
..... #define LED3 PIN_B2
..... #define LED4 PIN_B3
..... #define BUTTON PIN_C0
..... #define MAXRIPPLE 23          // Rizado máximo permitido (4Vpp)
..... #define NOERROR 0             // LEDs apagados
..... #define LOWVERROR 1           // LED1
..... #define LOWVERRORT 2          // LED2
..... #define HIGHVERRORT 3         // LED3
..... #define HIGHVERROR 4          // LED4
..... #define RIPPLEERROR 5         // Los 4 LEDs encendidos
                                    // Lista de errores posibles para la variable
                                    ERROR
..... #use fast_io(B)
.....
..... // Tabla que cumple la curva A de la Figura 9 de la MIL-PRF-24021
..... const int8 Limites[]=
                {240, 145, 105, 82, 68, 57, 49, 43, 38, 34, 31, 29, 26, 24, 23, 22, 20, 19,
.....           17, 16, 14, 14, 14, 13, 13, 12, 12, 11, 11, 10, 10, 9, 9, 9, 9, 9, 9,
.....           9, 8, 8, 8, 8, 8, 8, 8, 7, 7, 7, 7, 7, 7, 7, 6, 6, 6, 6, 6, 6, 6,
.....           6, 6, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
.....           4, 4, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3};
.....
..... int8 ContInt = 5;          // Se usa para multiplicar el tiempo del temporizador HW x5
```

```
*
00F1:  MOVLW  05
00F2:  BCF    03.5
00F3:  MOVWF  28
..... int1 TOMARMUESTRA = FALSE; // SI = TRUE se ha producido 5 veces la
                                    interrupción WH (ContInt=5)
00F4:  BCF    29.0
..... int8 ERROR = 0;            // Almacena el tipo de error
00F5:  CLRF   2A
..... int1 ESTADO_RELE = 0;      // Diferencia los 2 estados de funcionamiento
                                    del Power monitor, avión conectado o
                                    desconectado
00F6:  BCF    29.1
..... int8 MUESTRA = 0;          // Almacena el valor de la conversión del ADC
00F7:  CLRF   2B
..... int8 CONTADOR_MAL = 0;
00F8:  CLRF   2C
..... int8 CONTADOR_OK = 0;
00F9:  CLRF   2D
..... int16 CONTADOR_LOW = 0;    // Contador para 2.5s (500 x 5ms)
00FA:  CLRF   2E
00FB:  CLRF   2F
.....
.....     // Interrupción del Timer configurado para 1ms, se activa un flag cuando ha
             ocurrido 5 veces (5ms).
.....     // Se toman muestras del ADC cada 5ms.
..... #int_RTCC
..... void  RTCC_isr(void)
..... {
.....    if(--ContInt==0)
*
00A3:  DECFSZ 28,F
00A4:  GOTO   0A8
.....    {
.....       TOMARMUESTRA = TRUE;
00A5:  BSF    29.0
.....       ContInt = 5;
00A6:  MOVLW  05
00A7:  MOVWF  28
.....    }
..... }
.....
.....     // Toma una muestra del nivel de rizado de la tensión y se compara con el valor
             máximo admitido
.....     // Devuelve un 0 si se supera dicho valor.
00A8:  BCF    0B.2
00A9:  BCF    0A.3
00AA:  BCF    0A.4
00AB:  GOTO   01D
..... int1 RizadoOK(){
.....
..... int8 MUESTRARIPPLE;
.....
.....    set_adc_channel(1);
```

```
*
00C3:  MOVLW  08
00C4:  MOVWF  78
00C5:  MOVF   1F,W
00C6:  ANDLW  C7
00C7:  IORWF  78,W
00C8:  MOVWF  1F
.....................    delay_us(10);
00C9:  MOVLW  03
00CA:  MOVWF  77
00CB:  DECFSZ 77,F
00CC:  GOTO   0CB
.....................    MUESTRARIPPLE=read_adc();
00CD:  BSF    1F.2
00CE:  BTFSC  1F.2
00CF:  GOTO   0CE
00D0:  MOVF   1E,W
00D1:  MOVWF  31
.....................    if(MUESTRARIPPLE>MAXRIPPLE)
00D2:  MOVF   31,W
00D3:  SUBLW  17
00D4:  BTFSC  03.0
00D5:  GOTO   0DA
.....................       return 0;
00D6:  MOVLW  00
00D7:  MOVWF  78
00D8:  GOTO   0DD
.....................    else
00D9:  GOTO   0DD
.....................       return 1;
00DA:  MOVLW  01
00DB:  MOVWF  78
00DC:  GOTO   0DD
..................... }
00DD:  RETLW  00
.....................
..................... void main()
..................... {
00DE:  CLRF   04
00DF:  BCF    03.7
00E0:  MOVLW  1F
00E1:  ANDWF  03,F
00E2:  MOVLW  19
00E3:  BSF    03.5
00E4:  MOVWF  19
00E5:  MOVLW  A6
00E6:  MOVWF  18
00E7:  MOVLW  90
```