```
#define _XTAL_FREQ      4000000
// Lcd pinout settings
sbit LCD_RS at RA3_bit;
sbit LCD_EN at RA2_bit;
sbit LCD_D4 at RA1_bit;
sbit LCD_D5 at RA0_bit;
sbit LCD_D6 at RA7_bit;
sbit LCD_D7 at RA6_bit;

// Pin direction
sbit LCD_RS_Direction at TRISA3_bit;
sbit LCD_EN_Direction at TRISA2_bit;
sbit LCD_D4_Direction at TRISA1_bit;
sbit LCD_D5_Direction at TRISA0_bit;
sbit LCD_D6_Direction at TRISA7_bit;
sbit LCD_D7_Direction at TRISA6_bit;

// MAX_TIMECOUNT sets the number of loops after which the power is switched off
// one loop is aproximately 1.15 seconds long and the time period will be T = MAX_TIMECOUNT x 1.15

#define MAX_TIMECOUNT   208

int f1;
char PowerOFF_flag;


// inerrupt procedure
void interrupt() {
   T0IF_bit = 0;                    // clear T0IF bit
   f1++;
   if (RBIF_bit) {                  // if there is change on PORTB
    if (!PORTB.F5)                  // check if Power button is pressed
      PowerOFF_flag = 1;            // sets the PowerOFF_flag
    RBIF_bit = 0;                   // clear the interrupt flag
   }
}

// displays the frequency with commas after each 3 digits
void display_freq(long f) {
 char i, n, k, digit[9];
 n = 0;
 // separating digits
 do {
  digit[n] = '0' + f % 10;
  f /= 10;
  n++;
 } while (f > 0 && n < 9);

 k = 11;                  // number of symbols to be displayed: 9 digits + 2 commas
 for(i = 0; i < 9; i++) {
  if (k == 4 || k == 8) {
   if (i < n)
    Lcd_Chr(1,k+2,',');   // displays commas at 4-th and 8-th positions
   else
    Lcd_Chr(1,k+2,' ');   // displays ' ' if number is shorter
   k--;
  }
  if (i < n || i == 0) {
   Lcd_Chr(1,k+2,digit[i]); // displays digit
  }
  else {
   Lcd_Chr(1,k+2,' ');     // displays ' ' if number is shorter
  }
  k--;
 }
}


void main() {
  long freq;
  int f2, prescaler, timecount;
  char i, prescaler_bits;
```

```
CMCON = 0b00000111;                     // comparator off
T1CON = 0b00001110;                     // TMR1 prescale 1:1, osc = on
TRISA = 0b00110000;
TRISB = 0b11101111;
PORTB.F4 = 1;                           // sets the power on
Lcd_Init();
Lcd_Cmd(_LCD_CLEAR);
Lcd_Cmd(_LCD_CURSOR_OFF);
Lcd_Out(1, 1, "FrequencyCounter");
delay_ms(1000);
Lcd_Cmd(_LCD_CLEAR);
Lcd_Out(1, 1, "DIYfan.blogspot.com");
delay_ms(1000);
for(i = 0; i < 3; i++) {
  Lcd_Cmd(_LCD_SHIFT_LEFT);
  delay_ms(700);
}
Lcd_Cmd(_LCD_RETURN_HOME);
Lcd_Cmd(_LCD_CLEAR);
Lcd_Out(1, 1, "F=");
Lcd_Out(1, 15, "Hz");
PowerOFF_flag = 0;
timecount = 0;

while (1) {                             // main loop
  OPTION_REG = 0b00100111;              // set TMR0 prescale = 256
  // Seting the initial value of TMR1 to 62851
  // Timer1 will run for 2684 cycles = 0.08191 seconds
  TMR1L = 0b10000011;
  TMR1H = 0b11110101;
  TMR1IF_bit = 0;
  T0IF_bit = 0;
  f1 = 0;
  INTCON = 0b10101000;                  // Enable interupt at Timer0
  TMR1ON_bit = 1;                       // start Timer1
  TMR0 = 0;                             // start counting input signal

  while (!TMR1IF_bit) {}                // loop for 0.08191 seconds

  INTCON = 0;                           // Disable interupts
  TMR1ON_bit = 0;                       // stop Timer1

  prescaler = 256;                      // sets the initial value of prescaler
  prescaler_bits = 0b00000111;          // sets the initial value of prescaler bits

  // calculating prescaler
  // f1 holds the input frequency divided by 256*128 and multiplied by time period 0.08191s
  // if the frequency is above 102.4MHz prescaler = 256
  // 102.4MHz .. 51.2MHz -> prescaler = 128
  // 51.2MHz .. 25.6MHz -> prescaler = 64
  // 25.6MHz .. 12.8MHz -> prescaler = 32
  // 12.8Mhz .. 6.4MHz -> prescaler = 16
  // 6.4MHz .. 3.2MHz -> prescaler = 8
  // 3.2MHz .. 1.6Mhz -> prescaler = 4
  // 1.6MHz .. 800kHz -> prescaler = 2
  // 800khz .. 0 -> prescaler = 1
  f1 <<= 1;
  while (f1 < prescaler) {
    prescaler >>= 1;                    // divide prescaler by 2
    prescaler_bits--;
  }
  if (prescaler < 2) {
    prescaler = 1;
    OPTION_REG = 0b00101000;            // TMR0 prescale 1:1
    prescaler_bits = 0;
  }
  else {
    OPTION_REG = 0b00100000 | prescaler_bits; // TMR0 prescale
    prescaler_bits++;
  }
  // Seting the initial value of TMR1 to 32768
  // Timer1 will run for 32768 cycles = 1 second
```

```
      // Changing the initial value can tweak the time base.
      // 1 cycle = 1/32768 sec = 30 microseconds
      TMR1L = 0b00000000;
      TMR1H = 0b10000000;
      TMR1IF_bit = 0;
      T0IF_bit = 0;
      f1 = 0;
      INTCON = 0b10100000;                  // Enable interupt for Timer0
      TMR1ON_bit = 1;                       // start Timer1
      TMR0 = 0;                             // start counting input signal

      while (!TMR1IF_bit) {}                // loop for 1 second

      INTCON = 0;                           // Disable interupts
      TMR1ON_bit = 0;                       // stop Timer1
      f2 = TMR0;

      // calculating frequency
      freq = (long)f1;
      freq <<= 8;                           // multiply by 256
      freq += (long)f2;
      freq <<= prescaler_bits;              // multiply by prescaler
      display_freq(freq);                   // displaying frequency

  //  uncomment bellow to display the prescaller value on the second row
  //    i = 0;
  //    Lcd_Out(2, 1, "Prescaler:");
  //    for(i = 0; i < 3; i++) {
  //      if (prescaler) {
  //        Lcd_Chr(2,14-i,prescaler%10+'0');
  //        prescaler /= 10;
  //      }
  //      else Lcd_Chr(2,14-i,' ');
  //    }

      timecount++;

      // Shut down sequence
      if (PowerOFF_flag || timecount > MAX_TIMECOUNT) {
        INTCON = 0;                         // disable all interrupts
        Lcd_Cmd(_LCD_CLEAR);
        Lcd_Out(1, 2, "Power OFF...");
        delay_ms(1000);
        PORTB.F4 = 0;                       // switch off the power
        while (1) {}                        // endless loop (until power is off)
      }
    }
  }
```