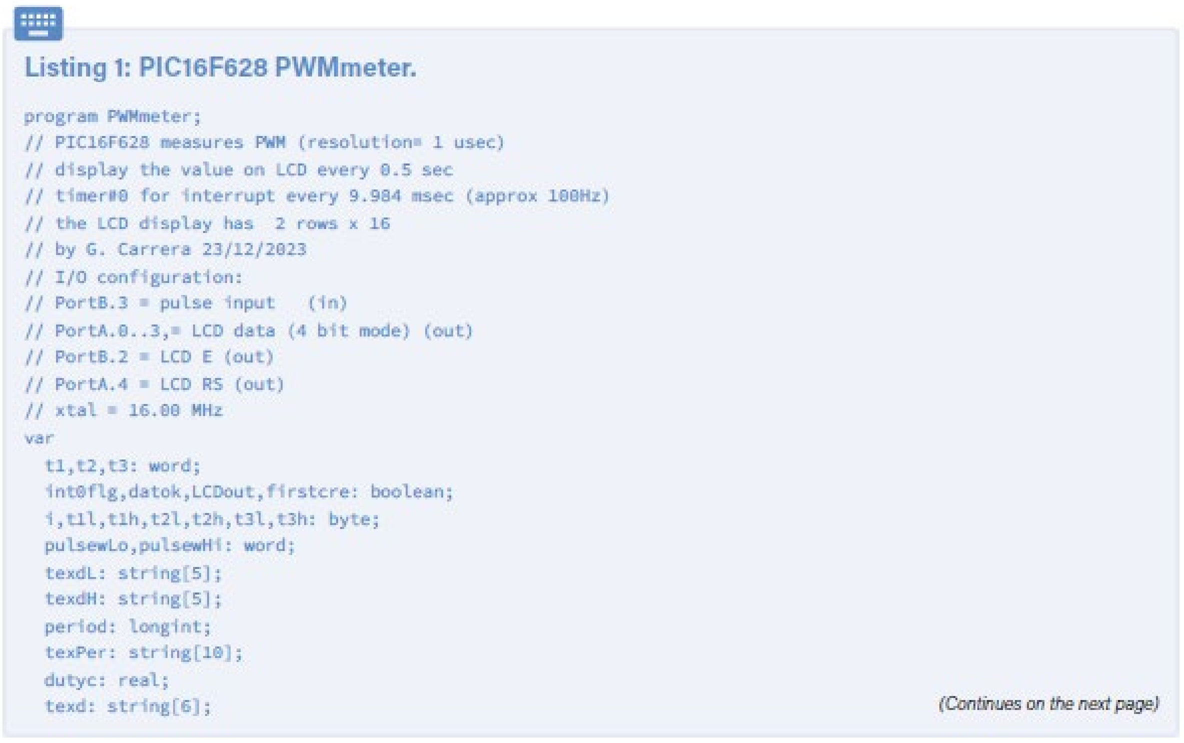
G. Carrera, “PWM Measurement with a PIC,” Elektor May/June 2024. https://www.elektormagazine.com/magazine-archive

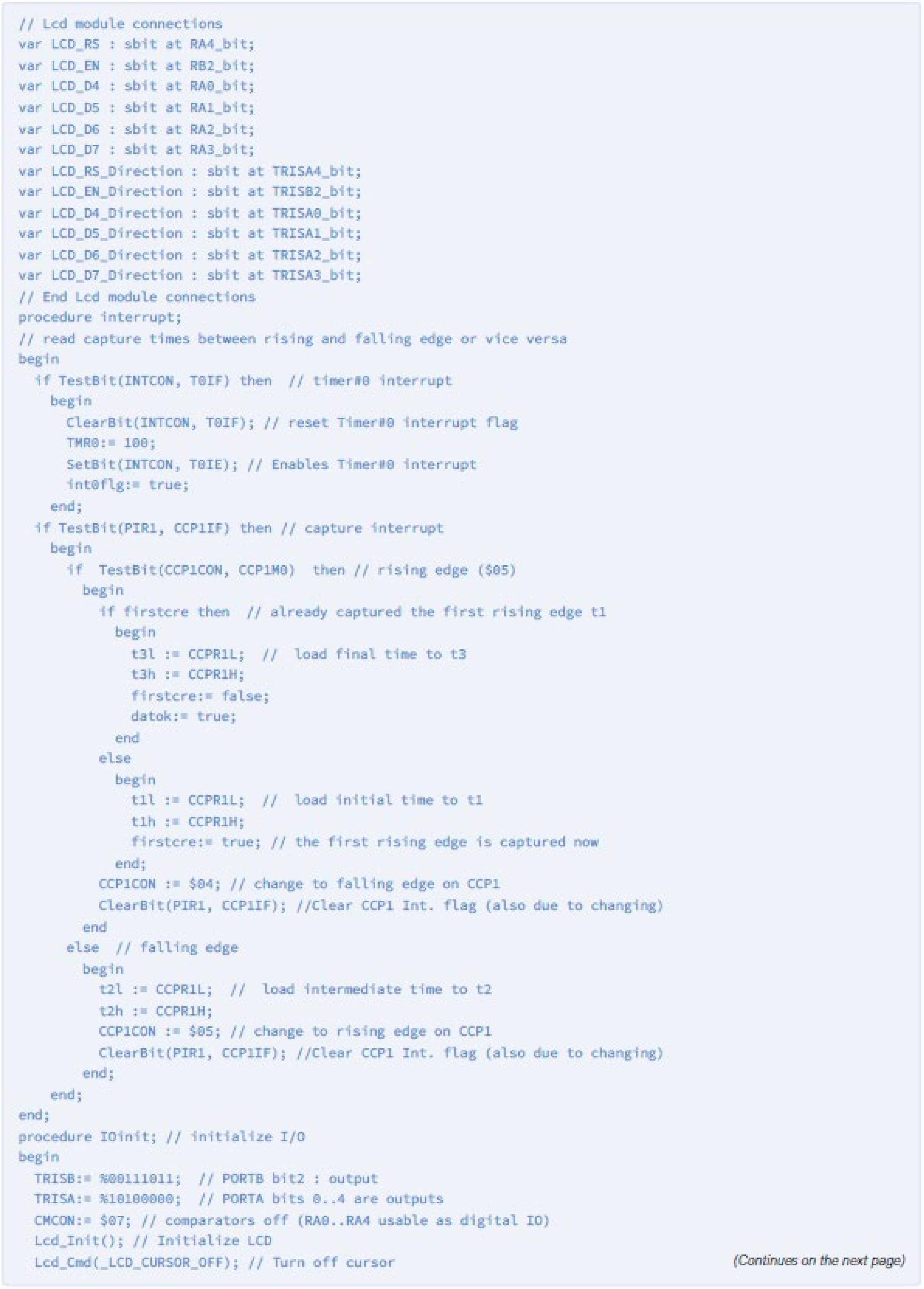


Listing 1: PIC16F628 PWMmeter.

```
program PWMmeter;
// PIC16F628 measures PWM (resolution= 1 usec)
// display the value on LCD every 0.5 sec
// timer#0 for interrupt every 9.984 msec (approx 100Hz)
// the LCD display has  2 rows x 16
// by G. Carrera 23/12/2023
// I/O configuration:
// PortB.3 = pulse input   (in)
// PortA.0..3,= LCD data (4 bit mode) (out)
// PortB.2 = LCD E (out)
// PortA.4 = LCD RS (out)
// xtal = 16.00 MHz
var
  t1,t2,t3: word;
  int0flg,datok,LCDout,firstcre: boolean;
  i,t1l,t1h,t2l,t2h,t3l,t3h: byte;
  pulsewLo,pulsewHi: word;
  texdL: string[5];
  texdH: string[5];
  period: longint;
  texPer: string[10];
  dutyc: real;
  texd: string[6];
```

(Continues on the next page)

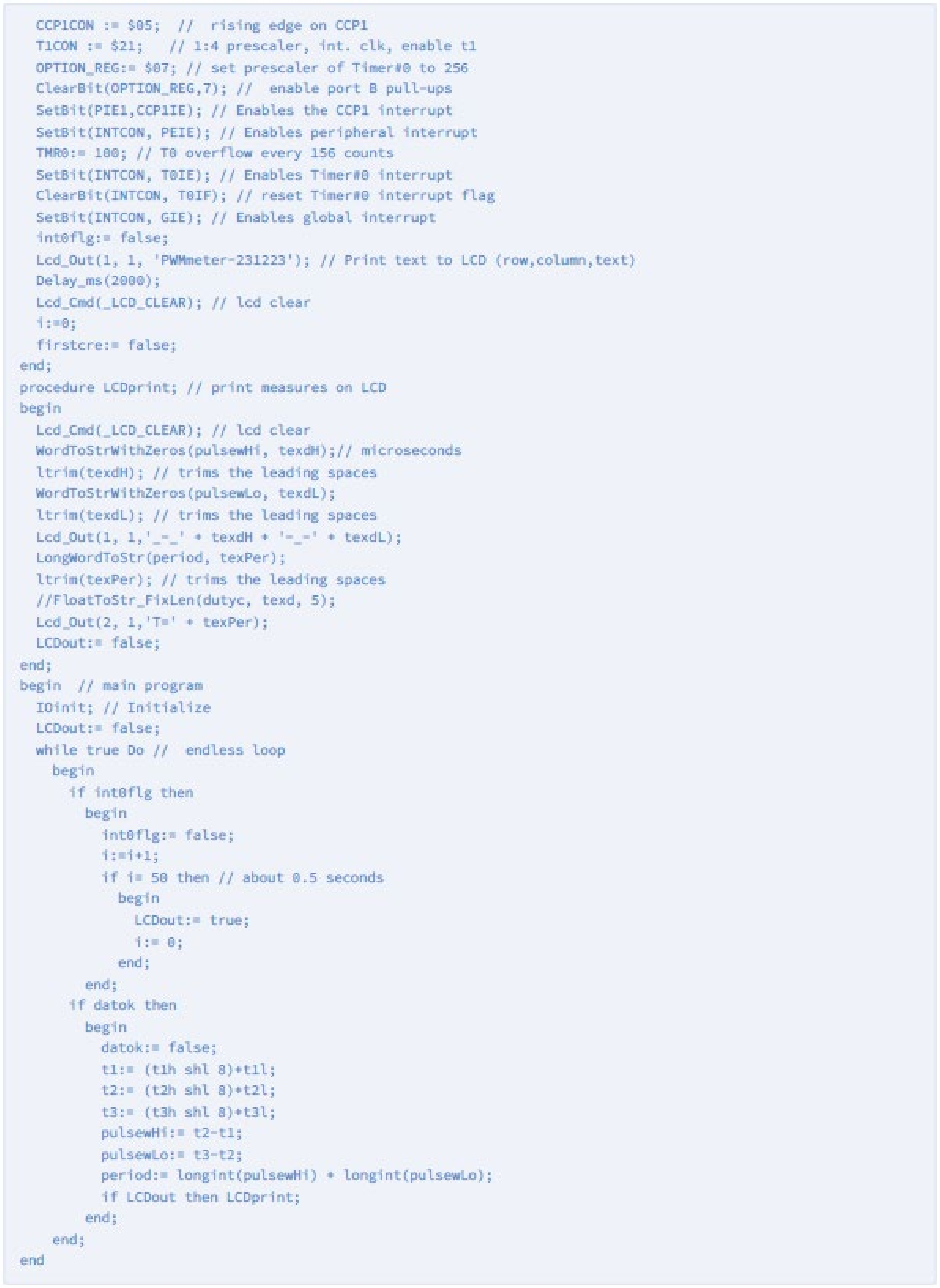
G. Carrera, “PWM Measurement with a PIC,” Elektor May/June 2024. https://www.elektormagazine.com/magazine-archive



```
// Lcd module connections
var LCD_RS : sbit at RA4_bit;
var LCD_EN : sbit at RB2_bit;
var LCD_D4 : sbit at RA0_bit;
var LCD_D5 : sbit at RA1_bit;
var LCD_D6 : sbit at RA2_bit;
var LCD_D7 : sbit at RA3_bit;
var LCD_RS_Direction : sbit at TRISA4_bit;
var LCD_EN_Direction : sbit at TRISB2_bit;
var LCD_D4_Direction : sbit at TRISA0_bit;
var LCD_D5_Direction : sbit at TRISA1_bit;
var LCD_D6_Direction : sbit at TRISA2_bit;
var LCD_D7_Direction : sbit at TRISA3_bit;
// End Lcd module connections
procedure interrupt;
// read capture times between rising and falling edge or vice versa
begin
  if TestBit(INTCON, T0IF) then  // timer#0 interrupt
    begin
      ClearBit(INTCON, T0IF); // reset Timer#0 interrupt flag
      TMR0:= 100;
      SetBit(INTCON, T0IE); // Enables Timer#0 interrupt
      int0flg:= true;
    end;
  if TestBit(PIR1, CCP1IF) then // capture interrupt
    begin
      if  TestBit(CCP1CON, CCP1M0)  then // rising edge ($05)
        begin
          if firstcre then  // already captured the first rising edge t1
            begin
              t3l := CCPR1L;  //  load final time to t3
              t3h := CCPR1H;
              firstcre:= false;
              datok:= true;
            end
          else
            begin
              t1l := CCPR1L;  //  load initial time to t1
              t1h := CCPR1H;
              firstcre:= true; // the first rising edge is captured now
            end;
          CCP1CON := $04; // change to falling edge on CCP1
          ClearBit(PIR1, CCP1IF); //Clear CCP1 Int. flag (also due to changing)
        end
      else  // falling edge
        begin
          t2l := CCPR1L;  //  load intermediate time to t2
          t2h := CCPR1H;
          CCP1CON := $05; // change to rising edge on CCP1
          ClearBit(PIR1, CCP1IF); //Clear CCP1 Int. flag (also due to changing)
        end;
    end;
end;
procedure IOinit; // initialize I/O
begin
  TRISB:= %00111011;  // PORTB bit2 : output
  TRISA:= %10100000;  // PORTA bits 0..4 are outputs
  CMCON:= $07; // comparators off (RA0..RA4 usable as digital IO)
  Lcd_Init(); // Initialize LCD
  Lcd_Cmd(_LCD_CURSOR_OFF); // Turn off cursor
```

(Continues on the next page)

G. Carrera, “PWM Measurement with a PIC,” Elektor May/June 2024. https://www.elektormagazine.com/magazine-archive



```
  CCP1CON := $05;  //  rising edge on CCP1
  T1CON := $21;   // 1:4 prescaler, int. clk, enable t1
  OPTION_REG:= $07; // set prescaler of Timer#0 to 256
  ClearBit(OPTION_REG,7); //  enable port B pull-ups
  SetBit(PIE1,CCP1IE); // Enables the CCP1 interrupt
  SetBit(INTCON, PEIE); // Enables peripheral interrupt
  TMR0:= 100; // T0 overflow every 156 counts
  SetBit(INTCON, T0IE); // Enables Timer#0 interrupt
  ClearBit(INTCON, T0IF); // reset Timer#0 interrupt flag
  SetBit(INTCON, GIE); // Enables global interrupt
  int0flg:= false;
  Lcd_Out(1, 1, 'PWMmeter-231223'); // Print text to LCD (row,column,text)
  Delay_ms(2000);
  Lcd_Cmd(_LCD_CLEAR); // lcd clear
  i:=0;
  firstcre:= false;
end;
procedure LCDprint; // print measures on LCD
begin
  Lcd_Cmd(_LCD_CLEAR); // lcd clear
  WordToStrWithZeros(pulsewHi, texdH);// microseconds
  ltrim(texdH); // trims the leading spaces
  WordToStrWithZeros(pulsewLo, texdL);
  ltrim(texdL); // trims the leading spaces
  Lcd_Out(1, 1,'_-_' + texdH + '-_-' + texdL);
  LongWordToStr(period, texPer);
  ltrim(texPer); // trims the leading spaces
  //FloatToStr_FixLen(dutyc, texd, 5);
  Lcd_Out(2, 1,'T=' + texPer);
  LCDout:= false;
end;
begin  // main program
  IOinit; // Initialize
  LCDout:= false;
  while true Do //  endless loop
    begin
      if int0flg then
        begin
          int0flg:= false;
          i:=i+1;
          if i= 50 then // about 0.5 seconds
            begin
              LCDout:= true;
              i:= 0;
            end;
        end;
      if datok then
        begin
          datok:= false;
          t1:= (t1h shl 8)+t1l;
          t2:= (t2h shl 8)+t2l;
          t3:= (t3h shl 8)+t3l;
          pulsewHi:= t2-t1;
          pulsewLo:= t3-t2;
          period:= longint(pulsewHi) + longint(pulsewLo);
          if LCDout then LCDprint;
        end;
    end;
end
```