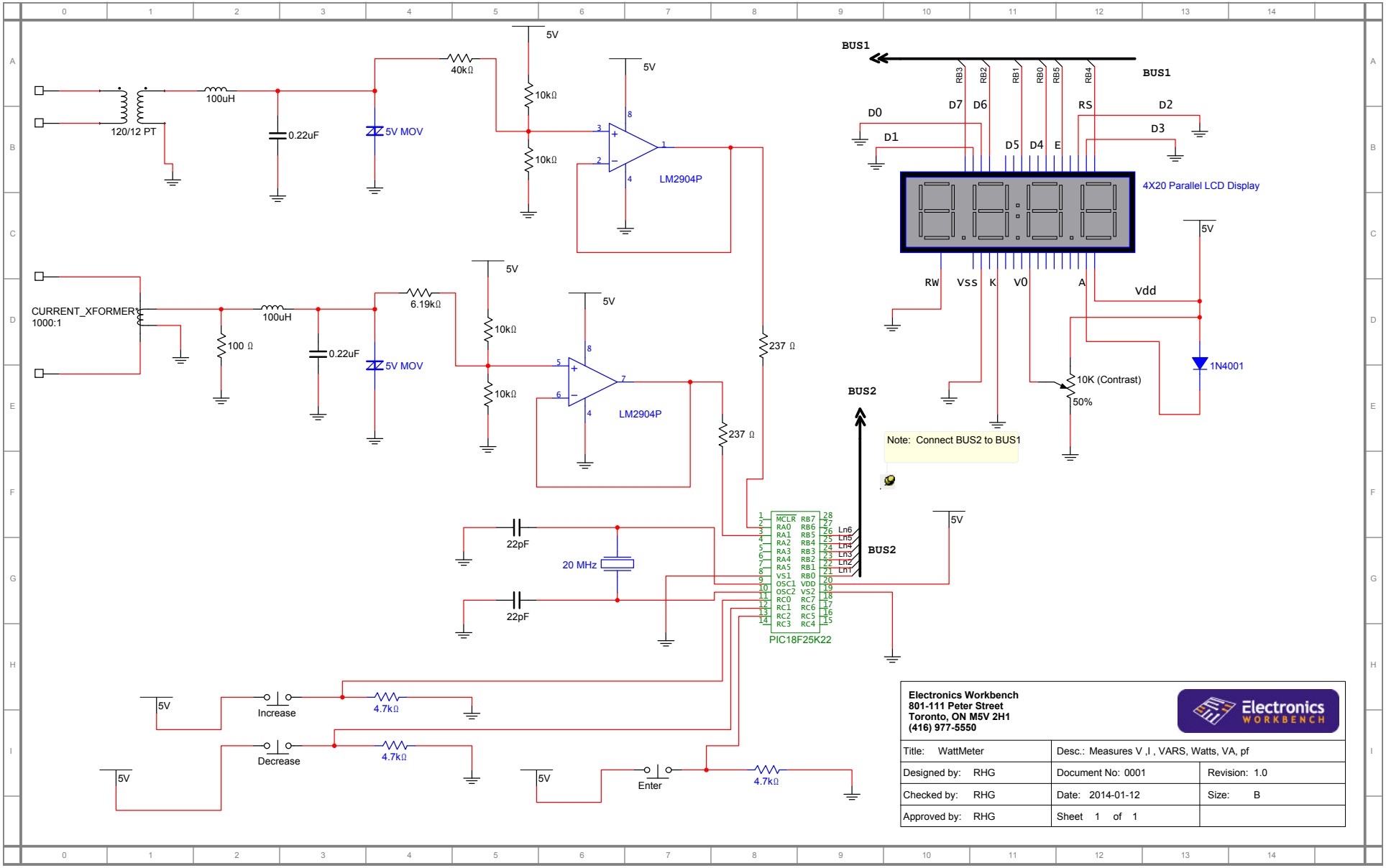

Electronics Workbench **801-111 Peter Street** **Toronto, ON M5V 2H1** **(416) 977-5550**		
Title: WattMeter	Desc.: Measures V ,I , VARS, Watts, VA, pf	
Designed by: RHG	Document No: 0001	Revision: 1.0
Checked by: RHG	Date: 2014-01-12	Size: B
Approved by: RHG	Sheet 1 of 1	

```
program Roy_Wattmeter
'
'*****************************************************************
'*  Name    : Wattmeter                                          *
'*  Author  : Roy H. Guerra Jr.                                  *
'*  Notice  : Copyright (c) 2013                                 *
'*          : All Rights Reserved                                *
'*  Date    : 12/23/2013                                         *
'*  Version : 1.0                                                *
'*  Notes   : Use PIC18F25K22                                    *
'*          : Use 20MHZ ceramic resonator                        *
'*****************************************************************
' Notes:
' 1) Circuit uses a PT & CT to provide isolation.  However, only symetrical
'    components, not asymetrical can be measured (no DC).
' 2) Harmonics up to (2 * sampling freq) can be measured accurately.
' 3) Total data collection should be >=2 times fundemantal frequency of 60HZ.
'    - A/D conversion using for 1 TAD = 4uS (Max) using "FRC" from PIC Datasheet
'    - ADC function uses 12 TAD or 48uS (4uS*12).
'    - There are two channels (Voltage & Current). This equates to 96uS.
'    - Acquisition time for one cycle = [(VTAD+ITAD+Delay)*(Number of Samples)].
'    - Choose a delay time of 10uS, and 0.0166 = [(48uS+48uS+10uS)*(Num_Samples)
'    - solve for the number of samples for one cycle = 156.6 or 157 samples.
'    - 60 Hz=16.6mS.  Nyquist criteria would be (2X) aquisition time or 0.0333,
'    - using 314 samples(157*2).  This would be the minimum.
'    - This circuit will sample at (10X), and use 300 samples.  The total
'    - acquisition time will be (0.0166*10)=[(48uS+48uS+delayuS)*(300)], solve
'    - for delayuS, and get "457".  So 10 full 60 HZ cycles are sampled.
' 4) The start of data collection begins with an "auto zero" upon power up
'    which is stored in EEPROM as a voltage & current offset.
' 5) A 20MHZ ceramic resonator with PLL enabled to give a clock frequency of
'    60MHZ.
' 6) rms = SQRT[(square each sample while adding them together) / (#samples)].
' 7) VA = [Magnitude (Vrms / #samples) * Magnitude (Irms / #samples)].
' 8) Watts = Average [sum (Vrms * Irms) / (#samples)].
' 9) pf = (Watts / VA).
' 10) VARS = SQRT[SQ(VA) - SQ(Watts)].
' 11) AC voltage and current swings negative (microcontroller is offset to take)
' 12) AC input is scaled via LM2904P circuit (not shown) to provide approx. 0-5V
'     for voltage and current. The digital offset = [(2^10/5 - 1)*(2.5)] = 512
'     this is for 10 bit A/D conversion and is adjusted upon power-up.
' 13) This circuit due to PT & CT is meant for sinusiod waveforms only. If you
'     remove the PT & CT isolation and use resistors, it can handle any
'     waveform, and also the DC component.
' 14) The accuracy of this circuit is typicaly 1%, but can be improved using
'     precision resistors in the scaling circuits, more accurate PT & CT, and
'     a faster sampling frequency with more samples (a different PIC IC), and
'     a more stable voltage supply.
' 15) Refresh rate, to begin new samples is around 1 Second.
' 16) The LCD is a 4 bit parallel mode and 4 lines X 20 characters.
' 17) A software calibration routine is used to set the span, and the span
'     for the voltage and current are stored in EEPROM.  To initiate press
'     the "Increase" and "Decrease" buttons and hold for "3" seconds.
' 18) At time of programming, enter in EEPROM the following:
'     - For voltage (address $5 = 78; address $6 = 00), this is 120 decimal
'     - For current (address $7 = 0A; address $8 = 00), this is 10 decimal
' 19) Minimum thresholds to maintain 1% accuracy is 10% nom values for Amps and
```

```
58: '     Volts (rms).  Nom. Amps is 20rms, and volts is 120rms.  Readings under 10%
59: '     nominal are allowed, but accuracy is affected.
60: ' 20) Maximum CT & PT Inputs are 22 amps(rms) and 132 volts(rms).
61: ' 20) Ensure PT & CT are wired "In-Phase", so samples are taken correctly.  One
62: '     way to check is to place wires in series, the voltage should be a max, not
63: '     min value (with CT & PT loaded). Or look at "pf" using a resistance load.
64: '
65: ' Hardware Declarations
66: ' ====================
67: '
68: ' LCD Module
69: ' -----------
70: ' LCD is parallel 4 bit data bus with HD44780 or equivalent Interface
71: ' LCD 4 Bit Data Bus (PORTB.0,1,2,3)to LCD (D4, D5, D6, D7)
72: ' LCD (RS) Bit to (PORTB.4)
73: ' LCD Enable Biit (E) to (PORTB.5)
74: ' LCD R/W tied to Ground with LCD Data pins(D0, D1, D2, D3)
75: ' Vdd = +5 V, A = +5v for backlight
76: ' Vss = Gnd, K = Gnd for backlight, Vo = LCD display contrast
77: '
78: ' PT & CT Inputs
79: ' --------------
80: ' Scaled PT input (0-4v) is PORTA.0 and Gnd = (-120 to +120) Volts rms input
81: ' Scaled CT input (0-4v) is PORTA.1 and Gnd = (-20 to + 20) Amps rms input
82: '
83: ' Software Declarations Section
84: ' ============================
85: '
86: ' Configuration Bits
87: ' ------------------
88: ' Oscillator = HS Oscilator
89: ' 4 X PLL = Always Enabled
90: ' Primary Clk = Always Enabled
91: ' Fail Safe Clk Monitor = Disabled
92: ' INT/EXT Switchover = Disabled
93: ' Brown out Detection = HW/SW Disabled
94: ' Power up Timer = Disabled
95: ' Watch Dog Timer = Disabled in hardware (H/S)
96: ' MCLR = Disabled, RE3 enabled
97: ' CCP2 B Output MUX Bit = RD2
98: ' T3CMX = RC0
99: ' HF Internal Fast Startup = Output is not delayed
100: ' CCP3 MUX Bit = MUX with RE0
101: ' PORTB A/D Enable Bit = RB<4:0> Digital
102: ' CCP2 MUX Bit =  RC1
103: ' In Circuit Debogger = Disabled
104: ' Extended Instruction Set = Disabled
105: ' Low Voltage Programming = Disabled
106: ' Stack Overflow Reset = Disabled
107: '
108: ' Constants:
109: ' ----------
110: const NUM_SAMPLES = 300     ' 300 measurements for each voltage / current sample
111:       VDD = 5.0             ' PIC supply voltage
112:
113: ' Variables:
114: ' ----------
```

```
115: dim Str1 as string[23]                          ' Display array for results
116:     Vsample, Isample as word[NUM_SAMPLES]       ' Array for storing V&I samples
117:     Vscale, Iscale as float                ' Scaled voltage and current numbers
118:     Vsquare, Isquare, VReal, IReal as float     ' Equation Variables
119:     VRms, IRms, PF, VAR, VA, Pwatt, PRms as float      ' Equation Variables
120:     Vspan, Ispan as word                        ' Calibration variables
121:     Voffset, Ioffset as word                    ' A/D offset variables
122:     I as word                                   ' Counting Variable
123:     Pos as word                                 ' String formatting variable
124:     flag_1, flag_2 as byte                      ' Calibration program flags
125: dim Sw_1 as sbit at PORTC.0                     ' (+) Increase Button
126: dim Sw_2 as sbit at PORTC.1                     ' (-) Decrese Button
127: dim Sw_3 as sbit at PORTC.2                     ' (enter) button
128: '
129: ' Lcd Module connections
130: ' ------------------------
131: dim LCD_RS as sbit at LATB4_bit                 ' LCD pins
132:     LCD_EN as sbit at LATB5_bit
133:     LCD_D4 as sbit at LATB0_bit
134:     LCD_D5 as sbit at LATB1_bit
135:     LCD_D6 as sbit at LATB2_bit
136:     LCD_D7 as sbit at LATB3_bit
137:
138: dim LCD_RS_Direction as sbit at TRISB4_bit      ' LCD data direction register
139:     LCD_EN_Direction as sbit at TRISB5_bit
140:     LCD_D4_Direction as sbit at TRISB0_bit
141:     LCD_D5_Direction as sbit at TRISB1_bit
142:     LCD_D6_Direction as sbit at TRISB2_bit
143:     LCD_D7_Direction as sbit at TRISB3_bit
144:
145: sub procedure InitMain()
146:   INTCON.7 = 0             ' Disable Global Interrupts
147:   INTCON2.7 = 0            ' Disable PORTB Pull Up's
148:   ANSELA = %00111111       ' Set PORTA to Analog
149:   ANSELC = %00000000       ' Set PORTC to Digital
150:   TRISA = %11111111        ' Set PORTA to Input
151:   TRISC = %00001111        ' Set PORTC to Output(4-7) and Input(0-3)
152:   ADC_Init                 ' Initialize A/D Converter
153:   delay_ms(200)            ' 200mS delay
154:   Lcd_Init()                     ' Initialize Lcd
155:   Delay_ms(200)                  ' LCD requires 100ms minimum
156:   Lcd_Cmd(_LCD_CLEAR)            ' Clear display
157:   Lcd_Cmd(_LCD_CURSOR_OFF)       ' Cursor off
158:   Lcd_Out(1,1,"Power Meter")     ' Write text in first row
159:   Lcd_Out(2,1,"By Roy Guerra")   ' Write text in second row
160:   Delay_ms(2000)                 ' Delay  2S
161:   VRms = 0                       ' Set all initial values to "0"  or a number
162:   IRms = 0
163:   Vsquare = 0
164:   Isquare = 0
165:   VReal = 0
166:   IReal = 0
167:   PF = 0
168:   VAR = 0
169:   VA = 0
170:   Pwatt = 0
171:   PRms = 0
```

```
172:   flag_1 = 0
173:   flag_2 = 0
174:  end sub
175:
176: sub procedure acquisition()
177:    Vscale = ((VDD / 1023.0) * Vspan)      ' 120Vrms~ nominal @ 5 volt input
178:    Iscale = ((VDD / 1023.0) * Ispan)      ' 10A~ nominal @ 5 volt input
179:
180:    for I = 0 to (NUM_SAMPLES - 1)       ' Starting at "0", so subtract "1"
181:       Vsample[I] = ADC_Read(0)          ' Sample Each Voltage; takes 48 us
182:       Isample[I] = ADC_Read(1)          ' Sample each Current; takes 48 us
183:       Delay_us(457)              ' makes a total of 0.166, = 10 full 60Hz cycles)
184:     next I
185:
186:       Vsquare = 0                    ' Set all initial values to "0"
187:       Isquare = 0
188:       Pwatt = 0
189:
190:     for I = 0 to (NUM_SAMPLES - 1)
191:       VReal = Vscale * float(integer(Vsample[I] - Voffset)) ' scale the voltage
192:       Vsquare = Vsquare + (VReal * VReal)            ' sum the squared voltages
193:
194:       IReal = Iscale * float(integer(Isample[I] - Ioffset)) ' scale the current
195:       Isquare = Isquare + (IReal * IReal)            ' sum the squared currents
196:
197:       Pwatt = Pwatt + (VReal * IReal)        ' Calculate sum of all power
198:     next I
199:                         ' Irms: square root out of the sum of voltage squared
200:     VRms = Sqrt(Vsquare / NUM_SAMPLES)
201:          if Vrms < 1 then              ' This keeps noise out of reading at "0"
202:            Vrms = 0
203:          end if
204:                        ' Irms: square root out of the sum of current squared
205:     IRms = Sqrt(Isquare / NUM_SAMPLES)
206:          if Irms < 0.1 then            ' This keeps noise out of reading at "0"
207:            Irms = 0
208:          end if
209:     PRms = (Pwatt / NUM_SAMPLES)        ' Power = average power
210:          if Prms < 2 then              ' This keeps noise out of reading at "0"
211:            Prms = 0
212:          end if
213:     VA  = (VRms * IRms)                 ' Apparent power (complex power)
214:     PF = (PRms / VA)                    ' PowerFactor
215:          if VA = 0 then                 ' This keeps from dividing by zero
216:            PF = 1.00
217:          end if
218:     VAR = Sqrt((VA * VA) - (PRms * PRms))    ' VARS
219: end sub
220:
221: sub procedure FormatStr(dim byref Str as string, dim Len as byte)' Format String
222:   Pos = strchr(Str, ".")           ' Searches string for decimal point
223:   Str[Pos + 3] = 0                 ' Truncates to 2 decimal places
224:   while StrLen(Str) < Len   ' Test length, and compare to suggested lengths below
225:    StrAppendPre(" ", Str)          ' Add a null to string begining
226:   wend
227: end sub
228:
```

```
229: sub procedure autozero()
230:   Lcd_Cmd(_LCD_CLEAR)             ' Clear display
231:   Lcd_Out(1,1,"Auto Zero")        ' Write text in first row
232:   Lcd_Out(2,1,"Remove Load")      ' Write text in second row
233:   Delay_ms(2000)                  ' Delay  2S
234:   Voffset = ADC_Read(0)           ' Perform A/D on PORTA pin 1
235:   Ioffset = ADC_Read(1)           ' Perform A/D on PORTA pin 2
236:   EEPROM_Write(0x00,Lo(Voffset)) ' Write Low Byte to EEPROM Address "0"
237:   Delay_ms(20)                    ' Allow 20mS to complete write
238:   EEPROM_Write(0x01,Hi(Voffset)) ' Write High Byte to EEPROM Address "1"
239:   Delay_ms(20)                    ' Allow 20mS to complete write
240:   EEPROM_Write(0x02,Lo(Ioffset)) ' Write Low Byte to EEPROM Address "2"
241:   Delay_ms(20)                    ' Allow 20mS to complete write
242:   EEPROM_Write(0x03,Hi(Ioffset)) ' Write High Byte to EEPROM Address "3"
243:   Delay_ms(20)                    ' Allow 20mS to complete write
244:   Lcd_Cmd(_LCD_CLEAR)             ' Clear display
245:   Lcd_Out(1,1,"Auto Zero")        ' Write text in first row
246:   Lcd_Out(2,1,"Complete")         ' Write text in second row
247:   Delay_ms(2000)                  ' Delay  2S
248:  end sub
249:
250: main:
251: ' Main Program
252: ' ============
253:   InitMain()               ' Sub-Procedure  for register / port initialization
254:   autozero()               ' Sub-Procedure  for A/D auto zero & EEPROM store
255:   Lo(Voffset) = EEPROM_Read(0x00)       ' Read Low Byte from EEPROM Address "0"
256:   Delay_ms(20)                          ' Allow 20mS to complete read
257:   Hi(Voffset) = EEPROM_Read(0x01)       ' Read High Byte from EEPROM Address "1"
258:   Delay_ms(20)                          ' Allow 20mS to complete read
259:   Lo(Ioffset) = EEPROM_Read(0x02)       ' Read Low Byte from EEPROM Address "2"
260:   Delay_ms(20)                          ' Allow 20mS to complete read
261:   Hi(Ioffset) = EEPROM_Read(0x03)       ' Read High Byte from EEPROM Address "3"
262:   Delay_ms(20)                          ' Allow 20mS to complete read
263:
264:   Lo(Vspan) = EEPROM_Read(0x05)         ' Read Low Byte from EEPROM Address "5"
265:   Delay_ms(20)                          ' Allow 20mS to complete read
266:   Hi(Vspan) = EEPROM_Read(0x06)         ' Read High Byte from EEPROM Address "6"
267:   Delay_ms(20)                          ' Allow 20mS to complete read
268:   Lo(Ispan) = EEPROM_Read(0x07)         ' Read Low Byte from EEPROM Address "7"
269:   Delay_ms(20)                          ' Allow 20mS to complete read
270:   Hi(Ispan) = EEPROM_Read(0x08)         ' Read High Byte from EEPROM Address "8"
271:   Delay_ms(20)                          ' Allow 20mS to complete read
272:
273:   ' Start data acquisition & conversion algorithims
274:   ' ------------------------------------------------
275:   while true
276:     acquisition()                    ' Go to Sub-Procedure acquisition
277:
278:     Lcd_Cmd(_LCD_CLEAR)              ' Clear LCD Display
279:
280:     FloatToStr(VRms, Str1)           ' Display RMS Voltage
281:     FormatStr(Str1, 6)               ' Go to Sub-Procedure Format, length = 6
282:     LCD_Out(1,1, Str1 + " V")       ' Display at LCD line 1, cursor position #1
283:
284:     FloatToStr(IRms, Str1)           ' Display RMS Current
285:     FormatStr(Str1, 5)               ' Go to Sub-Procedure Format, length = 5
```

```
286:     LCD_Out(1,12, Str1 + " A")       ' Display at LCD line 1, cursor position #12
287:
288:     FloatToStr(PRms, Str1)             ' Display Real Power
289:     FormatStr(Str1, 6)                 ' Go to Sub-Procedure Format, length = 6
290:     LCD_Out(2,1, Str1 + " W")         ' Display at LCD line 2, cursor position #1
291:
292:     FloatToStr(PF, Str1)               ' Display Power Factor
293:     FormatStr(Str1, 5)                 ' Go to Sub-Procedure Format, length = 5
294:     LCD_Out(2,12, Str1 + " pf")      ' Display at LCD line 2, cursor position #12
295:
296:     FloatToStr(VA, Str1)               ' Display RMS Voltage
297:     FormatStr(Str1, 6)                 ' Go to Sub-Procedure Format, length = 6
298:     LCD_Out(3,1, Str1 + " VA")        ' Display at LCD line 3, cursor position #1
299:
300:     FloatToStr(VAR, Str1)              ' Display RMS Current
301:     FormatStr(Str1, 6)                 ' Go to Sub-Procedure Format, length = 5
302:     LCD_Out(4,1, Str1 + " VAR")       ' Display at LCD line 4, cursor position #1
303:
304:     Delay_ms(1000)                     ' Allow 1 S to complete read
305:                            ' This entire last section is for "Auto Span"
306:     if ((Sw_1 = 1 ) and (Sw_2 = 1)) and (flag_1 = 0) then  '+ and - starts menu
307:       Delay_ms(3000)                                       ' 3 second delay
308:       if ((Sw_1 = 1 ) or (Sw_2 = 1)) and (flag_1 = 0) then
309:           flag_1 = 1
310:           flag_2 = 1
311:           Lcd_Cmd(_LCD_CLEAR)              ' Clear LCD Display
312:           Lcd_Out(1,1,"Auto Span")         ' Write text in first row
313:           Lcd_Out(2,1,"Calibration")       ' Write text in second row
314:           Delay_ms(2000)                   ' Delay  2S
315:           Lcd_Cmd(_LCD_CLEAR)              ' Clear LCD Display
316:           Lcd_Out(1,1,"Attach 120Vrms AC")      ' Write text in first row
317:           Lcd_Out(2,1,"Calibrated Source")      ' Write text in second row
318:           Lcd_Out(3,1,"Attach 20Arms AC")       ' Write text in third row
319:           Lcd_Out(4,1,"Calibrated Source")      ' Write text in fourth row
320:           Delay_ms(4000)                        ' Delay 4S
321:           Lcd_Cmd(_LCD_CLEAR)                   ' Clear LCD Display
322:           Lcd_Out(1,1,"Get Ready To")           ' Write text in first row
323:           Lcd_Out(2,1,"Calibrate")              ' Write text in second row
324:           Lcd_Out(3,1,"Press Increase")         ' Write text in third row
325:           Lcd_Out(4,1,"Press Decrease")         ' Write text in fourth row
326:           Delay_ms(2000)                        ' Delay 2S
327:           Lcd_Cmd(_LCD_CLEAR)                   ' Clear LCD Display
328:           acquisition()                         ' Go to this Sub-Procedure
329:       end if
330:    end if
331:    while flag_1 = 1                  ' Stay in loop until complete
332:          if flag_2 = 1 then                  ' Only display volts
333:             acquisition()                    ' Go to this Sub-Procedure
334:             Lcd_Cmd(_LCD_CLEAR)              ' Clear LCD Display
335:             FloatToStr(VRms, Str1)           ' Display RMS Voltage
336:             FormatStr(Str1, 6)          ' Go to Sub-Procedure Format, length = 6
337:             LCD_Out(1,1, Str1 + " Vrms")        ' Display at LCD line 1
338:             LCD_Out(2,1, "You Read 120V Yet?") ' Display Text on second Line
339:             LCD_Out(3,1, "Then Press Enter")   ' Display Text on third Line
340:             Delay_ms(250)                       ' 250 ms delay
341:           end if
342:          if flag_2 = 2 then                ' Only dispaly amps
```

```
343:              acquisition()                    ' Go to this Sub-Procedure
344:              Lcd_Cmd(_LCD_CLEAR)               ' Clear LCD Display
345:              FloatToStr(IRms, Str1)              ' Display RMS Current
346:              FormatStr(Str1, 5)          ' Go to Sub-Procedure Format, length = 5
347:              LCD_Out(1,1, Str1 + " Arms")       ' Display at LCD line 1
348:              LCD_Out(2,1, "You Read 20A Yet?")  ' Display Text on second Line
349:              LCD_Out(3,1, "Then Press Enter")   ' Display Text on third Line
350:              Delay_ms(250)                      ' 250 ms delay
351:           end if
352:          if flag_2 = 3 then                 ' Only display when completed
353:            Lcd_Cmd(_LCD_CLEAR)              ' Clear display
354:            Lcd_Out(1,1,"Auto Span")         ' Write text in first row
355:            Lcd_Out(2,1,"Calibration Complete")       ' Write text in second row
356:            Lcd_Out(3,1,"Turn Power off/on")          ' Write text in third row
357:            Lcd_Out(4,1,"For New Settings")           ' Write text in fourth row
358:            Delay_ms(2000)                            ' Delay  2S
359:            flag_2 = 0                                 ' Reset program flag
360:            flag_1 = 0                              ' Reset program flag to break loop
361:          end if
362:            if (Sw_1 = 1) and (flag_2 = 1) then      ' Volts section
363:                Delay_ms(100)                        ' 100 ms delay
364:                 if (Sw_1 = 1) and (Vspan < 301) and (flag_2 = 1) then ' Increase
365:                  inc(Vspan)                         ' Increase by "1"
366:                  Delay_ms(10)                       ' 10 ms delay
367:                 end if
368:              end if
369:            if (Sw_2 = 1) and (flag_2 = 1) then       ' Volts section
370:                Delay_ms(100)                       ' 100 ms delay
371:                 if (Sw_2 = 1) and (Vspan <> 0) and (flag_2 = 1) then  ' Decrease
372:                  dec(Vspan)                         ' Decrease by "1"
373:                  Delay_ms(10)                       ' 10 ms delay
374:                 end if
375:              end if
376:            if (Sw_3 = 1) and (flag_2 = 1 ) then          ' Enter Selection
377:               Delay_ms(500)                              ' 500 ms delay
378:                if (Sw_3 = 1) and (flag_2 = 1) then       ' Next section
379:                   EEPROM_Write(0x05,Lo(Vspan)) ' Write Low Byte to EEPROM Address
380:                   Delay_ms(20)                 ' Allow 20mS to complete write
381:                   EEPROM_Write(0x06,Hi(Vspan)) ' Write High Byte to EEPROM Address
382:                   Delay_ms(20)                 ' Allow 20mS to complete write
383:                   Lcd_Cmd(_LCD_CLEAR)           ' Clear display
384:                   Lcd_Out(1,1,"Voltage Span")       ' Write text in first row
385:                   Lcd_Out(2,1,"Calibration Complete") ' Write text in second row
386:                   Delay_ms(2000)                            ' Delay  2S
387:                   while (Sw_3 = 1) and (flag_2 = 1)  ' Wait for Enter key "up"
388:                     nop                               ' No operation
389:                   wend
390:                     flag_2 = 2                  ' Change program flag
391:                end if
392:              end if
393:            if (Sw_1 = 1) and (flag_2 = 2) then      ' Amps section
394:                Delay_ms(100)                        ' 100 ms delay
395:                 if (Sw_1 = 1) and (Ispan < 30) and (flag_2 = 2) then ' Increase
396:                  inc(Ispan)                         ' Increase by "1"
397:                  Delay_ms(500)                      ' 500 ms delay
398:                 end if
399:              end if
```

```
400:           if (Sw_2 = 1) and (flag_2 = 2) then      ' Amps section
401:               Delay_ms(100)                        ' 100 ms delay
402:                if (Sw_2 = 1) and (Ispan <> 0) and (flag_2 = 2) then
403:                 dec(Ispan)                          ' Decrease by "1"
404:                 Delay_ms(500)                       ' 500 ms delay
405:                end if
406:             end if
407:           if (Sw_3 = 1) and (flag_2 = 2 ) then          ' Enter Selection
408:              Delay_ms(500)                              ' 500 ms delay
409:              if (Sw_3 = 1) and (flag_2 = 2) then        ' Next section
410:                 EEPROM_Write(0x07,Lo(Ispan)) ' Write Low Byte to EEPROM Address
411:                 Delay_ms(20)                 ' Allow 20mS to complete write
412:                 EEPROM_Write(0x08,Hi(Ispan)) ' Write High Byte to EEPROM Address
413:                 Delay_ms(20)                 ' Allow 20mS to complete write
414:                 Lcd_Cmd(_LCD_CLEAR)            ' Clear display
415:                 Lcd_Out(1,1,"Current Span")       ' Write text in first row
416:                 Lcd_Out(2,1,"Calibration Complete") ' Write text in second row
417:                 Delay_ms(2000)                          ' Delay  2S
418:                 while (Sw_3 = 1) and (flag_2 = 2)   ' Wait for Enter key "up"
419:                    nop                                  ' No operation
420:                 wend
421:                flag_2 = 3                               ' Change program flag
422:              end if
423:           end if
424:     wend
425:   wend                    ' End of Main Program
426: end.                     ' Kept for program overflow (just in case)
```