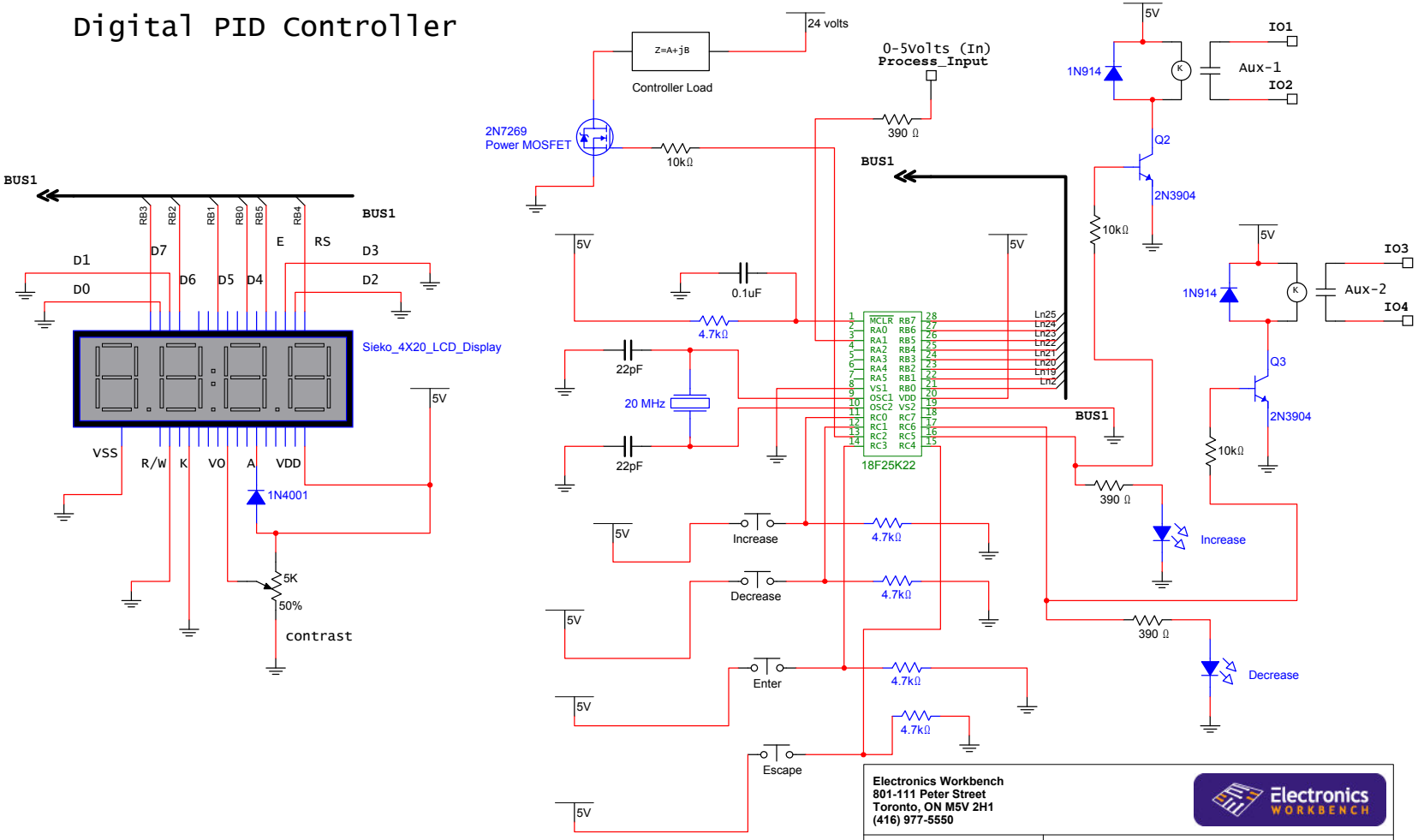
Digital PID Controller



Electronics Workbench **801-111 Peter Street** **Toronto, ON M5V 2H1** **(416) 977-5550**		
Title: Digital PID Controller	Desc.: For Slow / Medium Processes	
Designed by: RHG	Document No: 0001	Revision: 1.0
Checked by: RHG	Date: 7/6/14	Size: B
Approved by: RHG	Sheet 1 of 1	

```
program PID_Controller
'
'*****************************************************************
'*  Name    : PID Controller                                     *
'*  Author  : Roy H. Guerra Jr.                                  *
'*  Notice  : Copyright (c) 2014                                 *
'*          : All Rights Reserved                                *
'*  Date    : 7/6/2014                                           *
'*  Version : 1.0                                                *
'*  Notes   : Use PIC18F25K22                                    *
'*          : Use 20MHZ ceramic resonator                        *
'*****************************************************************
'
'   PID Theory(general concept)
'   ---------------------------
'         previous_error = 0
'         integral = 0
'       start:
'        error = setpoint - Input (process)
'        integral = integral + (error*dt)
'        derivative = (error - previous_error)/dt
'         output = (Kp*error) + (Ki*integral) + (Kd*derivative)
'         previous_error = error
'       wait(dt)
'       goto start
'
' Notes:
' ------
' - Output = (Kp*error) + (Ki*integral) + (Kd*derivative)
' - Output is based on HPWM so it runs continiously in background while code
'   is being executed, and is scaled (0-256) = 0-100% Controller output
'   which equals 0-100% Duty Cycle @ 5VDC).
' - To start the setup and configuration of the controller press the "Increase"
'   or "Decrease" pushbuttons.
' - To change from "Auto" to "Manual" operation, and back again, press and
'   hold the "Enter" and "Escape" pushbuttons when the controller is
'   In-Service.  Note- The increase & decrease pushbuttons are always
'   active in manual mode (changes output), and the LED's are turned off.
' - Designed for slow to medium-fast processes.
' - If Setpoint is not reached, then increase the "gain(s)"
' - Setpoint, Input (process), and Output are in "%" Units.
' - Contains false derivitive Error code to aviod a huge derivitive value
'   on a power up situation.
' - Contains an escape function if entered the menu function by mistake
'
' Description:
' ============
' This program is a Digital PID Controller that provides the following:
' - Menu driven operation on the LCD Display
' - EEPROM storage of setpoints, gains, & program functions (on a loss of power)
' - The ability to choose proportional control, derivitive control, integral
'   control, or any combination of the above.
' - Output indication (LED Indication for up or down control and a relay)
' - PWM with variable duty cycle that runs independent of program code, but is
'   updated by program (used for driving an FET Transistor to a Load).
' - An integrator limiter algorithim to prevent a "windup" condition, and a
'   timed reset for +/- process crossing the setpoint for error control.
```

```
58: ' - A Controler output limiter algorithim to prevent output saturation
59: '   on positive & negative swings.
60: ' - A timeout routine if no menu selection is made, to return to process
61: ' - Normal / Reverse Controller operation
62: ' - Adjustable Setpoint 0% to 100%
63: ' - All Gains are adjustable from (1-100)
64: ' - A 1% deadband around setpoint to minimize unnecessary cycling.
65: ' - Auto / Manual Operation
66: ' - Once tuned, accuracy is 1% of setpoint.  Can be made more accurate
67: '   by removing deadband of +/- 1 (around setpoint)
68: '
69: ' Note - Keep in mind that when you activate a menu function,
70: '         you are disconneted from the process.
71: '
72: ' Hardware Declarations
73: ' ======================
74: '
75: ' Note - All switch inputs are N/O and go high when depressed.
76: ' Increase = (PORTC0) (use 4.7K pulldown to Gnd)
77: ' Decrease = (PORTC1) (use 4.7K pulldown to Gnd)
78: ' Enter = (PORTC3) (use 4.7K pulldown to Gnd)
79: ' Escape = (PORTC4) (use 4.7K pulldown to Gnd)
80: ' LED_Up = (PORTC5) (with a 390 ohm series resistor)(can also use a relay)
81: ' LED_Dwn = (PORTC6) (with a 390 ohm series resistor)(can also use a relay)
82: ' PWM Signal (PORTC2). Controller output sinal used to drive FET Transistor,etc
83: ' Process siggnal input (PORTA1) (0-5 volt signal representing the process for
84: ' 0-100%)
85: '
86: ' LCD Module (4 rows X 20 characters)
87: ' ----------------------------------
88: ' LCD is parallel 4 bit data bus with HD44780 or equivalent Interface
89: ' LCD 4 Bit Data Bus (PORTB.0,1,2,3)to LCD (D4, D5, D6, D7)
90: ' LCD (RS) Bit to (PORTB.4)
91: ' LCD Enable Biit (E) to (PORTB.5)
92: ' LCD R/W tied to Ground with LCD Data pins(D0, D1, D2, D3)
93: ' Vdd = +5 V, A = +5v for backlight
94: ' Vss = Gnd, K = Gnd for backlight, Vo = LCD display contrast
95: '
96: ' Software Declarations Section
97: ' ==============================
98: '
99: ' Configuration Bits
100: ' ------------------
101: ' Oscillator = HS Oscilator
102: ' 4 X PLL = Always Enabled
103: ' Primary Clk = Always Enabled
104: ' Fail Safe Clk Monitor = Disabled
105: ' INT/EXT Switchover = Disabled
106: ' Brown out Detection = HW/SW Disabled
107: ' Power up Timer = Disabled
108: ' Watch Dog Timer = Disabled in hardware (H/S)
109: ' MCLR = Disabled, RE3 enabled
110: ' CCP2 B Output MUX Bit = RD2
111: ' T3CMX = RC0
112: ' HF Internal Fast Startup = Output is not delayed
113: ' CCP3 MUX Bit = MUX with RE0
114: ' PORTB A/D Enable Bit = RB<4:0> Digital
```

```
115: ' CCP2 MUX Bit =  RC1
116: ' In Circuit Debogger = Disabled
117: ' Extended Instruction Set = Disabled
118: ' Low Voltage Programming = Disabled
119: ' Stack Overflow Reset = Disabled
120: '
121: ' Store these Initial Values in EEPROM at Programming Time:
122: ' ======================================================
123: ' 0X01 = 10   'Initial KP Value  of 10
124: ' 0X02 = 01   'Initial KI Value  of 1
125: ' 0X03 = 01  'Initial KD Value  of 1
126: ' 0X04 = 32  'Initial Setpoint Value of 50%
127: ' 0X05 = 00  'Initial Flag_m Value of "0" for Proportional Control Only
128: ' 0X06 = 00  'Initial Flag_o Value of "0" for Normal Controller Operation
129: '
130: ' Constants:
131: ' ----------
132: const INT_TC as byte = 10                  'Time constant for the integral time
133: const PID_SUM_LIMIT as byte = 100          'Clamp the final output to 100%
134: const PID_SUM_LIMIT1 as integer = -100     'Clamp the final output to -100%
135: '
136: ' Variables:
137: ' ----------
138: dim Str1 as string[20]                ' Display array
139: dim text1 as string[20]               ' Display array
140: dim text2 as string[20]               ' Display array
141: dim text3 as string[20]               ' Display array
142: dim Inc_1 as sbit at PORTC.0          ' (+) Increase Button
143: dim Dec_1 as sbit at PORTC.1          ' (-) Decrease Button
144: dim Ent_1 as sbit at PORTC.3          ' (enter) Button
145: dim Esc_1 as sbit at PORTC.4          ' (abort) Button
146: dim LED_Up as sbit at PORTC.5     'Turns on when Setpoint > (Process + Deadband)
147: dim LED_Dwn as sbit at PORTC.6    'Turns on when Setpoint < (Process - Deadband)
148: dim Cntr_out as sbit at PORTC.2   'Adjustable duty cycle Controller output signal
149: dim Z_tim as byte                 'Menu Timeout variable
150: dim Flag_m as byte                'Mode flag (P,P+I,P+D,PID)
151: dim Flag_o as bit                 '0 = Normal; 1 = Reverse Controller Operation
152: dim Flag_am as bit                '0 = Auto; 1 = Manual
153: dim Process as float              'Process Value (equates to 0-100%)
154: dim Adval as word                 'A/D Value (0-1023)
155: dim Setpoint as word              'Setpoint(0-100%)
156: dim KP as byte                    'Proportional gain (1-100)
157: dim KI as byte                    'Integral gain (1-100)
158: dim KD as byte                    'Derivative gain (1-100)
159: dim Error as float                'Error input. (setpoint - process) for normal
160: dim PID_Sum as float              'PID summation of all gains (P + I + D)
161: dim PID_Outm as byte              'Manual Output summation
162: dim PID_P as float                'Proportional value
163: dim PID_I as float                'Integral value
164: dim PID_D as float                'Derivative value
165: dim Der_LastError as float        'Last error, used for calculating Derivitive
166: dim Int_Acc as float              'Integral accumulator.
167: dim Int_Count as word             'Counter for the integrator, matches against Ti
168: dim Flag1 as byte                 'Controller Setup menu Flag permissive
169: dim Pos as word                   'String formatting variable
170: dim D_First_Time as boolean       'Derivitive flag to aviod 1rst time large value
171: dim duty_cycle as byte            'Duty cycle variable for PWM value
```

```
'
' Lcd Module connections
' ----------------------
dim LCD_RS as sbit at LATB4_bit                ' LCD pins
    LCD_EN as sbit at LATB5_bit
    LCD_D4 as sbit at LATB0_bit
    LCD_D5 as sbit at LATB1_bit
    LCD_D6 as sbit at LATB2_bit
    LCD_D7 as sbit at LATB3_bit

dim LCD_RS_Direction as sbit at TRISB4_bit     ' LCD data direction register
    LCD_EN_Direction as sbit at TRISB5_bit
    LCD_D4_Direction as sbit at TRISB0_bit
    LCD_D5_Direction as sbit at TRISB1_bit
    LCD_D6_Direction as sbit at TRISB2_bit
    LCD_D7_Direction as sbit at TRISB3_bit

sub procedure InitMain()
  INTCON.7 = 0             ' Disable Global Interrupts
  INTCON2.7 = 0            ' Disable PORTB Pull Up's
  ANSELA = %00111111       ' Set PORTA to Analog
  ANSELC = %00000000       ' Set PORTC to Digital
  TRISA = %11111111        ' Set PORTA to Input
  TRISC = %00011011        ' Set PORTC to Output(2,3,6,7) and Input(0,1,4,5)
  ADC_Init                 ' Initialize A/D Converter
  PWM1_Init(5000)          ' Initialize PWM to 5KHz
  PWM1_Start()             ' Start PWM module
  delay_ms(200)            ' 200mS delay
  Lcd_Init()                     ' Initialize Lcd
  Delay_ms(200)                  ' LCD requires 100ms minimum
  Lcd_Cmd(_LCD_CLEAR)            ' Clear display
  Lcd_Cmd(_LCD_CURSOR_OFF)       ' Cursor off
  Lcd_Out(1,1,"PID Controller")  ' Write text in first row
  Lcd_Out(3,1,"By Roy Guerra")   ' Write text in third row
  Delay_ms(2000)                 ' Delay  2S
  PID_P = 0                'Reset P, I & D variables.
  PID_I = 0
  PID_D = 0
  Flag1 = 0                'Set menu flag permissive
  Flag_am = 0              '0 = Auto
  Z_tim = 0                'Set timeout to zero
  LED_Up = 0               'Turn off Indication
  LED_Dwn = 0              'Turn off Indication
  Int_Count = 0            'Reset the Integrator counter.
  Int_Acc = 0              'Reset the Integrator accumulator
  PID_Outm = 0             'Start Controller at zero initial value
  Der_LastError = 0        'Starts this value at zero.
  D_First_Time = true      'Sets derivitive initial flag
  duty_cycle = 5           ' sets initial value of duty cycle
 end sub

sub procedure acquisition()
If Flag_am = 0 then                         'Auto mode of operation
    if Flag_o = 0 then
       Error = float(Setpoint) - Process        'Calculate Error value(Normal)
    end if
    if Flag_o = 1 then
```

```
229:        Error = Process - float(Setpoint)          'Calculate Error value(Reverse)
230:     end if
231:   If (float(Setpoint) >= (Process + 1)) then    'Turn on indication for +error
232:     if Flag_o = 0 then
233:        LED_Up = 1               'Turn on Indication
234:        LED_Dwn = 0              'Turn off Indication
235:     end if
236:     if Flag_o = 1 then
237:        LED_Up = 0               'Turn off Indication
238:        LED_Dwn = 1              'Turn on Indication
239:     end if
240:   end if
241:
242:  If (float(Setpoint) <= (Process - 1)) then    'Turn on indication for -error
243:     if Flag_o = 0 then
244:        LED_Up = 0               'Turn on Indication
245:        LED_Dwn = 1              'Turn off Indication
246:     end if
247:     if Flag_o = 1 then
248:        LED_Up = 1               'Turn off Indication
249:        LED_Dwn = 0              'Turn on Indication
250:     end if
251:  end if
252:
253: ' This is a deadband of 1%, do nothing
254:   If (float(setpoint) <= Process + 1) and (float(setpoint) >= Process - 1) Then
255:      Error = 0
256:      LED_Up = 0                                          'Turn off Indication
257:      LED_Dwn = 0                                         'Turn off Indication
258:    end if
259:
260:    '(Calculate the Proportional Section)
261:        PID_P = Error                                     ' Proportional error
262:
263:    '(Calculate the Integral Section)
264:        Int_Acc = Int_Acc + Error          'Add error to acumulator.
265:        Int_Count = inc(Int_Count)         'Increment the reset-time counter
266:        If Int_Count >= Int_TC then        'Is it time to update the I-term?
267:          Int_Acc = (Int_Acc / float(Int_TC))     'Divide by reset time
268:          PID_I = PID_I + Int_Acc  'Integration inial value + accumulated value
269:          Int_Count = 0                    'Reset the Integrator counter.
270:          Int_Acc = 0                      'Reset the Integrator accumulator.
271:        end if
272:      if PID_I >= 80 then          'Prevents +Integrator Windup
273:        PID_I = 80
274:      end if
275:      if PID_I <= -80 then          'Prevents -Integrator Windup
276:        PID_I = -80
277:      end if
278:
279:    '(Calculate the Derivative Section)
280:      if D_First_Time = true then    'to avoid a huge DiffValue the first time
281:        D_First_Time = false
282:        Der_LastError = Error
283:      end if
284:      PID_D = Error - Der_LastError   'calculate derivitive based on cycle time
285:      Der_LastError = PID_D           'Store error for next Derivitive calc.
```

```
286:
287:      if PID_D >= 80 then          'limits process noise by clamping
288:        PID_D = 80
289:      end if
290:      if PID_D <= -80 then          'limits process noise by clamping
291:       PID_D = -80
292:      end if
293:
294:    'Calculate the Total Output Signal
295:
296:     Select Case Flag_m                'Case is based on mode of operation
297:       Case 0
298:         PID_sum = (PID_P * float(KP))
299:       Case 1
300:         PID_sum = (PID_P * float(KP))+(PID_I * float(KI))
301:       Case 2
302:         PID_sum = (PID_P * float(KP))+(PID_D * float(KD))
303:       Case 3
304:         PID_sum = (PID_P * float(KP))+(PID_I * float(KI))+(PID_D * float(KD))
305:     End Select
306:
307:       If PID_sum >= PID_SUM_LIMIT Then      'See if +Saturation limit is reached
308:          PID_sum = PID_SUM_LIMIT
309:        end if
310:
311:        If PID_sum <= PID_SUM_LIMIT1 Then    'See if -Saturation limit is reached
312:           PID_sum = PID_SUM_LIMIT1
313:        end if
314:
315:   'Set duty cycle
316:     duty_cycle = (PID_sum * 2.55)     ' Scaled at 0-255 = 0-100%
317:     If PID_sum <= 0 then          'Clamp at "0", no negative numbers allowed
318:         duty_cycle = 0
319:     end if
320:
321:     PWM1_Set_Duty(duty_cycle)             ' Set duty cycle to scaled value
322:  end if
323: end sub
324:
325: sub procedure FormatStr(dim byref Str as string, dim Len as byte)' Format String
326:   Pos = strchr(Str, ".")           ' Searches string for decimal point
327:   Str[Pos + 3] = 0                 ' Truncates to 2 decimal places
328:   while StrLen(Str) < Len   ' Test length, and compare to suggested lengths below
329:    StrAppendPre(" ", Str)          ' Add a null to string begining
330:   wend
331: end sub
332:
333: sub procedure menu()
334:   If Flag_am = 0 then
335:     if (Inc_1 = 1) or (Dec_1 = 1) then       'Initiates menu
336:     Delay_ms(100)                            'debounce
337:       if (Inc_1 = 1) or (Dec_1 = 1) then
338:       Lcd_Cmd(_LCD_CLEAR)               ' Clear display
339:       Lcd_Out(1,1,"Disconnected")       ' Write text in first row
340:       Lcd_Out(2,1,"From Process")       ' Write text in second row
341:       Lcd_Out(3,1,"Wait or")            ' Write text in third row
342:       Lcd_Out(4,1,"Press Escape")       ' Write text in fourth row
```

```
343:       Delay_ms(2000)                    ' 2 second delay
344:       Z_tim = 0                         ' Set timeout to zero
345:       Flag1 = 1                         ' Set Flag 1
346:     end if
347:   end if
348:  ' Setpoint Adjust
349:  ' ---------------
350:   While Flag1 = 1                       'Stay in loop while true
351:     if (Esc_1 = 1) and (Flag1 > 0) then             'Go back
352:       Delay_ms(100)                     ' Delay  100mS
353:      if (Esc_1 = 1) and (Flag1 > 0) then
354:        Flag_am = 0                    'Send controller to Auto
355:        Lcd_Cmd(_LCD_CLEAR)             ' Clear display
356:        Lcd_Out(1,1,"Terminated")       ' Write text in first row
357:        Delay_ms(1000)                  ' Delay  1S
358:        Flag1 = 0                       ' Reset Menu Permissive flag
359:        exit                            ' break out of routine
360:      end if
361:    end if
362:      Lcd_Cmd(_LCD_CLEAR)               ' Clear display
363:      Lcd_Out(1,1,"Setpoint Adjust")    ' Write text in first row
364:      byteToStr(Setpoint,text1)         ' Convert Setpoint to LCD format
365:      Lcd_Out(3,1,"Setpoint = " + text1 + "%")       ' Write text1 in third row
366:      Delay_ms(300)                          ' 300mS delay
367:      inc(Z_tim)                             ' Check for a timeout condition
368:      If (Z_tim = 35) then                   ' 10 seconds of no key depresses
369:       Lcd_Cmd (_LCD_CLEAR)                  ' Clear LCD
370:       Lcd_Out (1,1,"Timed Out")             ' Write text in first row
371:       Lcd_Out (3,1,"No User Response")      ' Write text in third row
372:       Delay_ms (2000)                       ' Wait 2s
373:       Z_tim = 0                             ' Reset Z
374:       Flag_am = 0                           ' Send controller to Auto
375:       Flag1 = 0                             ' Reset flag
376:       exit                                  ' break out of routine
377:     end if
378:     If (Inc_1 = 1) and (Setpoint <> 100) Then        ' switch debounce
379:         Delay_ms(100)                                ' 100mS delay
380:        If (Inc_1 = 1) and (Setpoint <> 100) Then
381:           inc(Setpoint)                             'increment by 1
382:           Z_tim = 0                                 'Reset Z
383:        end if
384:     end if
385:    If (Dec_1 = 1) And (Setpoint <> 0) Then           'switch debounce
386:        Delay_ms(100)                                 '100mS delay
387:       If (Dec_1 = 1) And (Setpoint <> 0) Then
388:           dec(Setpoint)                             'decrement by 1
389:           Z_tim = 0                                  'Reset Z
390:       end if
391:    end if
392:    If (Ent_1 = 1) Then                               'switch debounce
393:        Delay_ms(100)                                 '100mS delay
394:       If (Ent_1 = 1) Then
395:          Z_tim = 0                                  'Reset Z
396:          Flag1 = 2                                  'Change flag mode
397:        end if
398:     end if
399:   wend
```

```
400: ' Proportional Gain Adjust
401: '------------------------
402:  While Flag1 = 2                        ' Stay in loop while true
403:    if (Esc_1 = 1) and (Flag1 > 0) then              'Go back
404:       Delay_ms(100)                   ' Delay  100mS
405:      if (Esc_1 = 1) and (Flag1 > 0) then
406:        Flag_am = 0                                  'Send controller to Auto
407:        Lcd_Cmd(_LCD_CLEAR)             ' Clear display
408:        Lcd_Out(1,1,"Terminated")       ' Write text in first row
409:        Delay_ms(1000)                  ' Delay  1S
410:        Flag1 = 0                       'Reset Menu Permissive flag
411:        exit                            ' break out of routine
412:      end if
413:    end if
414:    Lcd_Cmd(_LCD_CLEAR)                 ' Clear display
415:    Lcd_Out(1,1,"Proportional Gain")        ' Write text in first row
416:    byteToStr(KP,text1)                 ' Convert Setpoint to LCD format
417:    Lcd_Out(3,1,"Prop. Gain = " + text1)      ' Write text1 in third row
418:    Delay_ms(300)                       ' 300mS delay
419:      inc(Z_tim)                             'Check for a timeout condition
420:      If (Z_tim = 35) then                   '10 seconds of no key depresses
421:       Lcd_Cmd (_LCD_CLEAR)                          ' Clear LCD
422:       Lcd_Out (1,1,"Timed Out")                ' Write text in first row
423:       Lcd_Out (3,1,"No User Response")         ' Write text in third row
424:       Delay_ms (2000)                          ' Wait 2s
425:       Z_tim = 0                                ' Reset Z
426:       Flag_am = 0                              'Send controller to Auto
427:       Flag1 = 0                                ' Reset flag
428:       exit                                     ' break out of routine
429:     end if
430:     If (Inc_1 = 1) and (KP <> 100) Then         'switch debounce
431:      Delay_ms(100)                             ' 100mS delay
432:        If (Inc_1 = 1) and (KP <> 100) Then
433:         inc(KP)                                 'increment by 1
434:         Z_tim = 0                               'Reset Z
435:        end if
436:     end if
437:    If (Dec_1 = 1) And (KP <> 0) Then            'switch debounce
438:     Delay_ms(100)                              ' 100mS delay
439:       If (Dec_1 = 1) And (KP <> 0) Then
440:         dec(KP)                                 'decrement by 1
441:         Z_tim = 0                               'Reset Z
442:       end if
443:    end if
444:    If (Ent_1 = 1) Then                         'switch debounce
445:       Delay_ms(100)                            ' 100mS delay
446:       If (Ent_1 = 1) Then
447:         Z_tim = 0                              'Reset Z
448:         Flag1 = 3                              'Change flag mode
449:        end if
450:    end if
451:  wend
452: ' Integral Gain Adjust
453: ' --------------------
454:   While Flag1 = 3                    ' Stay in loop while true
455:    if (Esc_1 = 1) and (Flag1 > 0) then              'Go back
456:       Delay_ms(100)                   ' Delay  100mS
```

```
457:      if (Esc_1 = 1) and (Flag1 > 0) then
458:        Flag_am = 0                                  'Send controller to Auto
459:        Lcd_Cmd(_LCD_CLEAR)              ' Clear display
460:        Lcd_Out(1,1,"Terminated")        ' Write text in first row
461:        Delay_ms(1000)                   ' Delay  1S
462:        Flag1 = 0                        ' Reset Menu Permissive flag
463:        exit                             ' break out of routine
464:      end if
465:    end if
466:    Lcd_Cmd(_LCD_CLEAR)                ' Clear display
467:    Lcd_Out(1,1,"Integral Gain")       ' Write text in first row
468:    byteToStr(KI,text1)                ' Convert Setpoint to LCD format
469:    Lcd_Out(3,1,"Int. Gain = " + text1)    ' Write text1 in third row
470:    Delay_ms(300)                      ' 300mS delay
471:      inc(Z_tim)                             'Check for a timeout condition
472:      If (Z_tim = 35) then                   '10 seconds of no key depresses
473:       Lcd_Cmd (_LCD_CLEAR)                  ' Clear LCD
474:       Lcd_Out (1,1,"Timed Out")             ' Write text in first row
475:       Lcd_Out (3,1,"No User Response")      ' Write text in third row
476:       Delay_ms (2000)                        ' Wait 2s
477:       Z_tim = 0                              ' Reset Z
478:       Flag_am = 0                            ' Send controller to Auto
479:       Flag1 = 0                              ' Reset flag
480:       exit                                   ' break out of routine
481:     end if
482:     If (Inc_1 = 1) and (KI <> 100) Then          ' switch debounce
483:      Delay_ms(100)                               ' 100mS delay
484:        If (Inc_1 = 1) and (KI <> 100) Then
485:         inc(KI)                                  'increment by 1
486:         Z_tim = 0                                'Reset Z
487:        end if
488:     end if
489:    If (Dec_1 = 1) and (KI <> 0) Then             'switch debounce
490:     Delay_ms(100)                                ' 100mS delay
491:       If (Dec_1 = 1) and (KI <> 0) Then
492:         dec(KI)                                 'decrement by 1
493:         Z_tim = 0                                'Reset Z
494:       end if
495:    end if
496:    If (Ent_1 = 1) Then                           'switch debounce
497:       Delay_ms(100)                              ' 100mS delay
498:       If (Ent_1 = 1) Then
499:         Z_tim = 0                                'Reset Z
500:         Flag1 = 4                                ' Change flag mode
501:        end if
502:    end if
503:  wend
504: ' Derivitive Gain Adjust
505: ' ----------------------
506:   While Flag1 = 4                      ' Stay in loop while true
507:    if (Esc_1 = 1) and (Flag1 > 0) then             'Go back
508:       Delay_ms(100)                  ' Delay  100mS
509:      if (Esc_1 = 1) and (Flag1 > 0) then
510:        Flag_am = 0                                  'Send controller to Auto
511:        Lcd_Cmd(_LCD_CLEAR)              ' Clear display
512:        Lcd_Out(1,1,"Terminated")        ' Write text in first row
513:        Delay_ms(1000)                   ' Delay  1S
```

```
514:        Flag1 = 0                          ' Reset Menu Permissive flag
515:        exit                               ' break out of routine
516:      end if
517:    end if
518:    Lcd_Cmd(_LCD_CLEAR)                    ' Clear display
519:    Lcd_Cmd(_LCD_CURSOR_OFF)               ' Cursor off
520:    Lcd_Out(1,1,"Derivitive Gain")         ' Write text in first row
521:    byteToStr(KD,text1)                    ' Convert Setpoint to LCD format
522:    Lcd_Out(3,1,"Der. Gain = " + text1)       ' Write text1 in third row
523:    Delay_ms(300)                          ' 300mS delay
524:      inc(Z_tim)                           'Check for a timeout condition
525:      If (Z_tim = 35) then                 '10 seconds of no key depresses
526:       Lcd_Cmd (_LCD_CLEAR)                ' Clear LCD
527:       Lcd_Out (1,1,"Timed Out")           ' Write text in first row
528:       Lcd_Out (3,1,"No User Response")    ' Write text in third row
529:       Delay_ms (2000)                      ' Wait 2s
530:       Z_tim = 0                            ' Reset Z
531:       Flag_am = 0                          ' Send controller to Auto
532:       Flag1 = 0                            ' Reset flag
533:       exit                                 ' break out of routine
534:     end if
535:     If (Inc_1 = 1) and (KD <> 100) Then        'switch debounce
536:      Delay_ms(100)                             ' 100mS delay
537:        If (Inc_1 = 1) and (KD <> 100) Then
538:         inc(KD)                                 'increment by 1
539:         Z_tim = 0                               'Reset Z
540:        end if
541:     end if
542:    If (Dec_1 = 1) And (KD <> 0) Then           'switch debounce
543:     Delay_ms(100)                              ' 100mS delay
544:       If (Dec_1 = 1) And (KD <> 0) Then
545:         dec(KD)                                 'decrement by 1
546:         Z_tim = 0                               'Reset Z
547:       end if
548:    end if
549:    If (Ent_1 = 1) Then                         'switch debounce
550:       Delay_ms(100)                            ' 100mS delay
551:       If (Ent_1 = 1) Then
552:         Z_tim = 0                               'Reset Z
553:         Flag1 = 5                               ' Change flag mode
554:        end if
555:    end if
556:  wend
557: ' Mode Adjust
558: ' -----------
559:   While Flag1 = 5                       ' Stay in loop while true
560:     if (Esc_1 = 1) and (Flag1 > 0) then              'Go back
561:       Delay_ms(100)                                  ' Delay  100mS
562:      if (Esc_1 = 1) and (Flag1 > 0) then
563:        Flag_am = 0                     ' Send controller to Auto
564:        Lcd_Cmd(_LCD_CLEAR)             ' Clear display
565:        Lcd_Out(1,1,"Terminated")       ' Write text in first row
566:        Delay_ms(1000)                  ' Delay  1S
567:        Flag1 = 0                       ' Reset Menu Permissive flag
568:        exit                            ' break out of routine
569:      end if
570:    end if
```

```
571:      inc(Z_tim)                                  'Check for a timeout condition
572:      If (Z_tim = 35) then                        '10 seconds of no key depresses
573:       Lcd_Cmd (_LCD_CLEAR)                       ' Clear LCD
574:       Lcd_Out (1,1,"Timed Out")
575:       Lcd_Out (3,1,"No User Response")
576:       Delay_ms (2000)                             ' Wait 2s
577:       Z_tim = 0                                   ' Reset Z
578:       Flag_am = 0                                 ' Send controller to Auto
579:       Flag1 = 0                                   ' Reset flag
580:       exit                                        ' break out of routine
581:     end if
582:    select case Flag_m                    ' Choose mode of operation
583:     case 0
584:      Lcd_Cmd(_LCD_CLEAR)                 ' Clear display
585:      Lcd_Out(1,1,"Prop Control")         ' Write text in first row
586:      Lcd_Out(2,1,"To Change Press")      ' Write text in second row
587:      Lcd_Out(3,1,"To INC or DEC")        ' Write text in third row
588:      Lcd_Out(4,1,"Then Press Enter")     ' Write text in fourth row
589:      Delay_ms(300)                       ' Delay  300mS
590:     case 1
591:      Lcd_Cmd(_LCD_CLEAR)                 ' Clear display
592:      Lcd_Out(1,1,"P+I Control")          ' Write text in first row
593:      Lcd_Out(2,1,"To Change Press")      ' Write text in second row
594:      Lcd_Out(3,1,"To INC or DEC")        ' Write text in third row
595:      Lcd_Out(4,1,"Then Press Enter")     ' Write text in fourth row
596:      Delay_ms(300)                       ' Delay  300mS
597:     case 2
598:      Lcd_Cmd(_LCD_CLEAR)                 ' Clear display
599:      Lcd_Out(1,1,"P+D Control")          ' Write text in first row
600:      Lcd_Out(2,1,"To Change Press")      ' Write text in second row
601:      Lcd_Out(3,1,"To INC or DEC")        ' Write text in third row
602:      Lcd_Out(4,1,"Then Press Enter")     ' Write text in fourth row
603:      Delay_ms(300)                       ' Delay  300mS
604:     case 3
605:      Lcd_Cmd(_LCD_CLEAR)                 ' Clear display
606:      Lcd_Out(1,1,"P+I+D Control")        ' Write text in first row
607:      Lcd_Out(2,1,"To Change Press")      ' Write text in second row
608:      Lcd_Out(3,1,"To INC or DEC")        ' Write text in third row
609:      Lcd_Out(4,1,"Then Press Enter")     ' Write text in fourth row
610:      Delay_ms(300)                       ' Delay  300mS
611:    end select
612:     If (Inc_1 = 1) and (Flag_m <> 3) Then               'switch debounce
613:       Delay_ms(100)                                     ' Delay  100mS
614:        If (Inc_1 = 1) and (Flag_m <> 3) Then
615:          inc(Flag_m)                                  'increment mode by 1
616:          Z_tim = 0                                    'Reset Z
617:        end if
618:     end if
619:    If (Dec_1 = 1) And (Flag_m <> 0) Then          'switch debounce
620:       Delay_ms(100)                               ' Delay  100mS
621:     If (Dec_1 = 1) And (Flag_m <> 0) Then
622:         dec(Flag_m)                               'decrement mode by 1
623:         Z_tim = 0                                 'Reset Z
624:        end if
625:     end if
626:    If (Ent_1 = 1) Then                               'switch debounce
627:       Delay_ms(100)                                  ' Delay  100mS
```

```
628:         If (Ent_1 = 1) Then
629:            Z_tim = 0                                   'Reset Z
630:            Flag1 = 6                                   ' Change flag mode
631:         end if
632:     end if
633:     Delay_ms(200)                                      ' 200mS delay
634:   wend
635: ' Operation Mode Subroutine:
636: ' -------------------------
637:   While Flag1 = 6                                 ' Stay in loop while true
638:    if (Esc_1 = 1) and (Flag1 > 0) then            ' Go back
639:       Delay_ms(100)                               ' Delay  100mS
640:      if (Esc_1 = 1) and (Flag1 > 0) then
641:        Flag_am = 0                               ' Send controller to Auto
642:        Lcd_Cmd(_LCD_CLEAR)                       ' Clear display
643:        Lcd_Out(1,1,"Terminated")                 ' Write text in first row
644:        Delay_ms(1000)                            ' Delay  1S
645:        Flag1 = 0                                 ' Reset Menu Permissive flag
646:        exit                                      ' break out of routine
647:      end if
648:    end if
649:      inc(Z_tim)                                 ' Check for a timeout condition
650:      If (Z_tim = 35) then                       ' 10 seconds of no key depresses
651:       Lcd_Cmd (_LCD_CLEAR)                      ' Clear LCD
652:       Lcd_Out (1,1,"Timed Out")
653:       Lcd_Out (3,1,"No User Response")
654:       Delay_ms (2000)                           ' Wait 2s
655:       Z_tim = 0                                 ' Reset Z
656:       Flag_am = 0                               ' Send controller to Auto
657:       Flag1 = 0                                 ' Reset flag
658:       exit                                      ' break out of routine
659:     end if
660:    select case Flag_o                    ' Choose normal or reverse operation
661:     case 0
662:      Lcd_Cmd(_LCD_CLEAR)                 ' Clear display
663:      Lcd_Cmd(_LCD_CURSOR_OFF)            ' Cursor off
664:      Lcd_Out(1,1,"Normal Output")        ' Write text in first row
665:      Lcd_Out(2,1,"To Change Press")      ' Write text in second row
666:      Lcd_Out(3,1,"INC or DEC")           ' Write text in third row
667:      Lcd_Out(4,1,"Then Press Enter")     ' Write text in fourth row
668:      Delay_ms(300)                       ' Delay  300mS
669:     case 1
670:      Lcd_Cmd(_LCD_CLEAR)                 ' Clear display
671:      Lcd_Cmd(_LCD_CURSOR_OFF)            ' Cursor off
672:      Lcd_Out(1,1,"Reverse Output")       ' Write text in first row
673:      Lcd_Out(2,1,"To Change Press")      ' Write text in second row
674:      Lcd_Out(3,1,"To INC or DEC")        ' Write text in third row
675:      Lcd_Out(4,1,"Then Press Enter")     ' Write text in fourth row
676:      Delay_ms(300)                       ' Delay  300mS
677:    end select                            'Return to process
678:    If (Inc_1 = 1) and (Flag_o <> 1) Then             'switch debounce
679:        Delay_ms(100)                                 ' Delay  100mS
680:          If (Inc_1 = 1) and (Flag_o <> 1) Then
681:            Flag_o = Flag_o + 1                       'increment by 1
682:            Z_tim = 0                                 'Reset Z
683:          end if
684:    end if
```

```
685:    If (Dec_1 = 1) And (Flag_o <> 0) Then              'switch debounce
686:       Delay_ms(100)                                   ' Delay  100mS
687:         If (Dec_1 = 1) And (Flag_o <> 0) Then
688:           Flag_o = Flag_o - 1                         'decrement by 1
689:           Z_tim = 0                                   'Reset Z
690:         end if
691:    end if
692:    If (Ent_1 = 1) Then                              'switch debounce
693:       Delay_ms(100)                                 ' Delay  100mS
694:         If (Ent_1 = 1) Then
695:           Z_tim = 0                                 'Reset Z
696:           Flag1 = 7                                 ' Change mode flag
697:         end if
698:    end if
699:    Delay_ms(200)
700:   wend
701:   While Flag1 = 7                                  ' Stay in loop while true
702:      if (Esc_1 = 1) and (Flag1 > 0) then           ' Go back
703:       Delay_ms(100)                                ' Delay  100mS
704:      if (Esc_1 = 1) and (Flag1 > 0) then
705:        Flag_am = 0                                 ' Send controller to Auto
706:        Lcd_Cmd(_LCD_CLEAR)                         ' Clear display
707:        Lcd_Out(1,1,"Terminated")                   ' Write text in first row
708:        Delay_ms(1000)                              ' Delay  1S
709:        Flag1 = 0                                   ' Reset Menu Permissive flag
710:        exit                                        ' break out of routine
711:      end if
712:    end if
713:      Lcd_Cmd(_LCD_CLEAR)                           ' Clear display
714:      Lcd_Out(1,1,"Storing Data")                   ' Write text in first row
715:      Lcd_Out(3,1,"In EEPROM")                      ' Write text in third row
716:      Delay_ms(2000)                                ' 2 second delay
717:      Z_tim = 0
718:      EEPROM_Write(0x01,KP)             ' Write some data at address 1
719:      Delay_ms(20)                      ' 20mS EEPROM write time
720:      EEPROM_Write(0x02,KI)             ' Write some data at address 2
721:      Delay_ms(20)                      ' 20mS EEPROM write time
722:      EEPROM_Write(0x03,KD)             ' Write some data at address 3
723:      Delay_ms(20)                      ' 20mS EEPROM write time
724:      EEPROM_Write(0x04,Setpoint)       ' Write some data at address 4
725:      Delay_ms(20)                      ' 20mS EEPROM write time
726:      EEPROM_Write(0x05,Flag_m)         ' Write some data at address 5
727:      Delay_ms(20)                      ' 20mS EEPROM write time
728:      EEPROM_Write(0x06,Flag_o)         ' Write some data at address 6
729:      Delay_ms(20)                      ' 20mS EEPROM write time
730:      Flag1 = 0                         ' Reset program flag
731:   wend
732:  end if
733: end sub
734: '
735: ' Manual Subroutine
736: ' -----------------
737: sub procedure manual()
738:    If Flag1 = 0 then                 'Check if controller is in operation
739:       If (Ent_1 = 1) and (Esc_1 = 1) Then          ' To enable manual mode
740:           Delay_ms(100)                            ' Delay  100mS
741:             If (Ent_1 = 1) and (Esc_1 = 1) Then    ' debounce & delay
```

```
742:                  Flag1 = 8                              ' Set menu flag out of range
743:                 Flag_am = Flag_am + 1                   ' swap flag between 0 and 1
744:           end if
745:         end if
746:       end if
747:    if Flag_am = 1 then                                  'Manual control mode
748:       LED_Up = 0                                         'Turn off LED's
749:       LED_Dwn = 0
750:         If Flag_o = 0 Then                               ' Normal operation
751:            If (Inc_1 = 1) and (PID_Outm <> 100) Then      'switch debounce
752:               Delay_ms(100)                              ' Delay  100mS
753:               If (Inc_1 = 1) and (PID_Outm <> 100) Then
754:                  inc(PID_Outm)                           'increment by 1
755:               end if
756:            end if
757:            If (Dec_1 = 1) And (PID_Outm <> 0) Then        'switch debounce
758:              Delay_ms(100)                               ' Delay  100mS
759:                If (Dec_1 = 1) And (PID_Outm <> 0) Then
760:                  dec(PID_Outm)                           'decrement by 1
761:               end if
762:             end if
763:        end if
764:
765:       If Flag_am = 1 then                                'Manual control mode
766:         LED_Up = 0                                        'Turn off LED's
767:         LED_Dwn = 0
768:         If Flag_o = 1 Then                               'Reverse operation
769:             If (Inc_1 = 1) and (PID_Outm <> 0) Then       'switch debounce
770:                 Delay_ms(100)                            ' Delay  100mS
771:                   If (Inc_1 = 1) and (PID_Outm <> 0 ) Then
772:                      dec (PID_Outm)                      'decrement by 1
773:                   end if
774:             end if
775:          If (Dec_1 = 1) And (PID_Outm <> 100) Then        'switch debounce
776:             Delay_ms(100)                                ' Delay  100mS
777:             If (Dec_1 = 1) And (PID_Outm <> 100) Then
778:               inc (PID_Outm)                             'increment by 1
779:             end if
780:          end if
781:         end if
782:       end if
783:
784:       duty_cycle = float(PID_Outm) * 2.55        ' Scaled at 0-255 = 0-100%
785:       PWM1_Set_Duty(duty_cycle)                  ' Set duty cycle to scaled value
786:    end if
787: end sub
788: '
789: ' Main Program
790: ' ============
791: main:
792:    InitMain()                 ' Sub-Procedure for register / port initialization
793:    KP = EEPROM_Read(0x01)              ' Read data from address 1
794:    Delay_ms(20)                        ' 20mS EEPROM read time
795:    KI = EEPROM_Read(0x02)              ' Read data from address 2
796:    Delay_ms(20)                        ' 20mS EEPROM read time
797:    KD = EEPROM_Read(0x03)              ' Read data from address 3
798:    Delay_ms(20)                        ' 20mS EEPROM read time
```

```
799:     Setpoint = EEPROM_Read(0x04)        ' Read data from address 4
800:     Delay_ms(20)                        ' 20mS EEPROM read time
801:     Flag_m = EEPROM_Read(0x05)          ' Read data from address 5
802:     Delay_ms(20)                        ' 20mS EEPROM read time
803:     Flag_o = EEPROM_Read(0x06)          ' Read data from address 6
804:     Delay_ms(20)                        ' 20mS EEPROM read time
805:
806:    ' Start data acquisition & conversion algorithims
807:    ' ------------------------------------------------
808:     Lcd_Cmd(_LCD_CLEAR)                  ' Clear display
809:     Lcd_Out(1,1,"Press Increase")        ' Write text in first row
810:     Lcd_Out(2,3,"or Decrease")           ' Write text in second row
811:     Lcd_Out(3,1,"To Adjust")             ' Write text in third row
812:     Delay_ms(2000)                       ' Delay  2S
813:
814:    while true
815:     Flag1 = 0                            ' Flag is reset
816:     Adval = ADC_Read(1)                  ' A/D result 0-1023 (2^10-1)
817:     Process = (float(Adval)/1023)*(100)  ' equates to (0 to 100 F.S.)
818:     acquisition()                        ' Sub-Procedure acquisition
819:     menu()                               ' Sub-Procedure for menu control
820:     manual()                             ' Sub-Procedure for manaul control
821:     Lcd_Cmd(_LCD_CLEAR)                  ' Clear LCD Display
822:
823:      if Flag_am = 0 then                ' Automatic Control Mode
824:       Lcd_Out(1,1,"Auto Control")       ' Write text in first row
825:       byteToStr(setpoint,text1)                  ' Convert Output to LCD format
826:       Lcd_Out(2,1,"Setpoint = " + text1 + "%")    ' Write text1 in second row
827:       FloatToStr(Process, Str1)         ' Display Control output
828:       FormatStr(Str1, 5)                ' Go to Sub-Procedure Format, length = 5
829:       Lcd_Out(3,1,"Process = " + Str1 + "%")     ' Write text1 in third row
830:       FloatToStr(PID_Sum, Str1)        ' Display Control output
831:       FormatStr(Str1, 5)                ' Go to Sub-Procedure Format, length = 5
832:       LCD_Out(4,1, "Control = " + Str1 + "%")        ' Display at LCD line 4
833:      end if
834:
835:     if Flag_am = 1 then                 ' Manual Control Mode
836:       Lcd_Cmd(_LCD_CLEAR)               ' Clear display
837:       Lcd_Out(1,1,"Manual Control")     ' Write text in first row
838:       FloatToStr(Process, Str1)         ' Display Control output
839:       FormatStr(Str1, 5)                ' Go to Sub-Procedure Format, length = 5
840:       Lcd_Out(3,1,"Process = " + Str1 + "%")     ' Write text1 in third row
841:       byteToStr(PID_Outm,text3)                  ' Convert Output to LCD format
842:       Lcd_Out(4,1,"Output = " + text3 + "%")     ' Write text1 in third row
843:     end if
844:
845:     delay_ms (300)                                '300mS delay
846:
847:    wend
848: end.
```