OVERVIEW

A solid state thermostat, using a PIC12C508 as the measuring and control device. This circuit demonstrates the PIC12C508's ability to measure an unknown resistance, and either communicate its value to another device through a serial connection, or to provide direct control outputs, as has been done in this case. The resistance could be from a light dependent resistor, a temperature dependent resistor, or any other variable resistance.

APPLICATION OPERATION

This circuit takes advantage of the schmitt trigger input of GP2 to monitor the voltage on a capacitor while it is discharged - first through a known reference resistor, then through an unknown resistance. The time taken to discharge the capacitor in each case is recorded, and used to determine a course of action. For this application, the two times obtained are simply compared to one another (with some hysteresis added to avoid chatter), but they could also be used to determine the value of the unknown resistance, by noting that the ratio of the two resistances is equal to the ratio of the two discharge times for each.

The thermostat has been designed as a backup thermostat to the main one in our house. It operates a relay below a fixed setpoint of 10°C or 50°F, should the primary heating control fail. Since the output is from a relay, though, it could be put to a variety of uses. By reversing the logic (or the contact), the output could indicate cooling failure, whether from air conditioning, or from a food freezer (that should remain below 0°C or 32°F).

The circuit is constructed as shown in the block diagram. The output relay is powered by 12V, while the PIC12C508 is powered from 5V through a 78L05 regulator. The transistor is a standard 2N3904, and the diode is a 1N4001. The main components of interest, however, are the Rtemp, Rset and 10K resistors, as well as the 0.1μF capacitor. These four components make up the measuring circuit.

A measurement is conducted by charging the capacitor to +5V through the 10K resistor, then switching GP2 to an input, and discharging the capacitor through the setpoint resistor (Rset - 18K), then repeating the process using the unknown resistor (Rtemp - a 10K at 25°C thermistor such as the 271-110A from Radio Shack). If the time taken to discharge the capacitor through Rtemp is greater than the time for Rset, for three consecutive counts, then GP0 is set high, energizing the output relay. The relay will remain energized until a temperature of 12°C or 54°F is measured. The 18K (Rset) value was chosen as it represents a temperature of 10°C or 50°F. A value of 30K could be chosen for a freezer alarm, or 8.2K for an A/C alarm.
This circuit uses the watchdog timer to wake up at maximum (2.3 sec) intervals, then make a measurement. During each measurement, the LED is turned on, providing some visual feedback to show there is activity. Should the temperature be found to be out of limits, however, (likely due to a bad sensor) the error condition is shown by a reduction in the watchdog period, effectively causing a faster (warning) flash rate.

One problem that occurred during the design of this circuit is that the WDT seemed to reset the IO port state to all inputs upon wakeup from sleep, so I couldn't just check the present state with a BTFS instruction. Rather than rely on file storage to determine the required state of the output I believe I took a novel approach by using the timing capacitor as a memory storage element. Upon wakeup, the level at GP2 is checked, and if it's high, then the output to the relay is driven high as well. The small blip in coil voltage isn't even noticed by the relay.

### BILL OF MATERIALS

<table>
<thead>
<tr>
<th>Part#</th>
<th>Manufacture</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC12C508</td>
<td>Microchip</td>
</tr>
<tr>
<td>10K@25°C</td>
<td>One source is Radio Shack</td>
</tr>
<tr>
<td>Thermistor</td>
<td>271-110A</td>
</tr>
</tbody>
</table>
APPENDIX A: SOFTWARE LISTING

1 ;
2 ; PICostat
3 ----------
4 ; by Jim Nagy, July 1997
5 ;
6 ; A solid state thermostat, using a PIC12C508 as the measuring and control
7 ; device. This circuit serves to demonstrate the ‘508s ability to measure an
8 ; analog quantity, and make decisions based on it.
9 ;
10 ; The PICostat has been designed as a backup heating thermostat. It operates
11 ; at a fixed temperature, when the measured resistance of a thermistor is
12 ; equal to a reference resistance. Hysteresis is added on the shutoff to
13 ; avoid hunting.
14 ;
15 ; Circuit connections are as follows:
16 ; - Output is active high from GP0 (pin 7)
17 ; - An ‘activity’ LED output (active high) is from GP1 (pin 6)
18 ; - the voltage on a grounded.1uF capacitor is monitored at GP2 through
19 ; a 10K resistor
20 ; - A thermistor (10K@25C) is connected from GP5 (pin 2) to the capacitor
21 ; - A reference resistor is connected from GP4 (pin 3) to the capacitor
22 ; - GP3 (pin 4) is configured as an active low MCLR, with internal pullup
23 ; - +5V is connected to pin 1, gnd to pin 8
24 ;
25 ; The reference resistor is 18K (providing operation at 10C or 50F)
26 ;
27 ; **************************************************************************
28 ; Program equates
29 - 0026 Hyst EQU H’26’ ; Hysteresis for turnoff (about 2deg C at a 10C setpoint)
30 - 0003 Cycles EQU 3 ; #times that temp must be stable for, before output change
31 ; Standard Equates
32 ;
33 - 0000 W EQU 0
34 - 0001 F EQU 1
35 - 0007 GPWUF EQU 7
36 - 0005 PA0 EQU 5
37 - 0004 TO EQU 4
38 - 0003 PD EQU 3
39 - 0002 Z EQU 2
40 - 0002 Zero EQU 2
41 - 0001 DC EQU 1
42 - 0000 C EQU 0
43 - 0000 Carry EQU 0
44 - 0000 MCLRDisabled EQU 0
45 - 0010 MCLREnabled EQU H’10’
46 - 0000 CodeProtect EQU 0
47 - 0008 NoCodeProtect EQU H’08’
48 - 0000 WDTDisabled EQU 0
49 - 0004 WDTEnabled EQU H’04’
50 - 0002 IntRCOsc EQU H’02’
51 - 0003 ExtRCOsc EQU H’03’
52 - 0001 XTOsc EQU H’01’
53 - 0000 LPOsc EQU 0
54 ; ‘508 Registers
55 - 0000 INDF EQU H’00’
56 - 0001 TMRO EQU H’01’
57 - 0002 PCL EQU H’02’
58 - 0003 STATUS EQU H’03’
59 - 0004 FSR EQU H’04’
60 - 0005 OSCCAL EQU H’05’
65   = 0006 GPIO     EQU H’06’
66   ; program variables
67   = 0007 TRefLo    EQU H’07’    ; Lo byte of 7uS counter - ref resistor
68   = 0008 TRefHi    EQU H’08’    ; Hi byte of “    “
69   = 0009 TMeasLo   EQU H’09’    ; Lo byte of measurement counter
70   = 000A TMeasHi   EQU H’0A’    ; Hi byte of “    “
71   = 000B OnCount   EQU H’0B’    ; delay for output turn-on
72   = 000C OffCount  EQU H’0C’    ; delay for output turn-off
73
74   ;      *********************************************
75   ;       Setting the ID words...
76
77   ORG H’0200’
78 0200   0000    ID0       Data.W  H’0000’
79 0201   0000    ID1       Data.W  H’0000’
80 0202   0000    ID2       Data.W  H’0000’
81 0203   0005    ID3       Data.W  H’0005’
82
83   ;      *********************************************
84   ;       and the Fuses...
85
86   ORG H’0FF0’
87 0FF0   01E     CONFIG  Data.W  MCLREnabled + NoCodeProtect + WDTEnabled + IntRCOsc
88
89
90   ;      *********************************************
91   ;               Charge
92   ;       Charges up the capacitor, and waits 10 time constants
93   ;       GP2 is left as an OUTPUT afterward, and GP4,5 are inputs
94   ;
95   0007   0C38    Charge  MOVLW   B’00111000’ ; turn GP4 and GP5 off (inputs), and GP2 on
96   0008   0006    TRIS    GPIO
97   0009   0546    BSF     GPIO,2  ; start charging, but wait ~10mS
98   000A   006D    MOVWF   H’0D’    ; outer loop counter
99   000B   002A    MOVWF   TMeasHi   ; (OK to trash these regs right now)
100  000C   0069    ch1 CLRF   TMeasLo
101  000D   02E9    ch2 DECFSZ  TMeasLo,F  ; wait 256*3uS (repeated 13*)
102  000E   0A0D    GOTO   ch2
103  000F   02EA    DECF SZ  TMeasHi,F
104  0010   0A0C    GOTO   ch1
105  0011   0800    RETLW  0
106
107   ;      *********************************************
Simple counting loop that waits for GP2 to go low
(each count is approx. 7uS)

Measure CLRF TMeasHi ; clear the counters
CLRF TMeasLo
m1 BTFSS GPIO,2 ; check if the cap is discharged
RETlw 0 ; if so, we’re done
INCFSZ TMeasHi,F
GOTO OOLimits ; somethings wrong with the sensor...

*********************************************
OOLimits
Timing count is Out of Limits! Overflow of counter
occurs at approx –110C, so circuit must be open.

*********************************************
Init
reset the ‘on’ counter
and pretend that we’ve turned off properly
CLRWDT
switch the prescaler to \(/8
(all outputs off
GP0-2 are outputs
TRIS GPIO
SLEEP ; then bail

*********************************************
Main
turn the LED on and start charging the cap
but check if the cap is already charged,
and if so, also turn the relay on
GP0, GP1, and GP2 are outputs,
GP3, GP4, and GP5 are inputs

OOLimits may have changed things, so...
first, measure the reference resistor
charge up the capacitor
BP4 starts at +5V
make Rref (GP4) an output
and GP2 an input
drop GP4 to 0V and measure the time
no pullups, and no wakeup on change

;

CALL Charge ; charge up the capacitor
BSF GPIO,4
MOVlw B’00101100’ ; make Rref (GP4) an output
TRIS GPIO
BCF GPIO,4
CALL Measure
MOVF TMeasLo,W ; tuck away the results
MOVWF TRefLo
MOVF TMeasHi,W
MOVWF TRefHi
now, measure the unknown resistor

;  get the cap ready

;  will need GP5 at +5V

;  make Rtemp (GP5) an output

;  and GP2 an input

;  drop GP5 to 0V and measure the time

;  add temp offset to reading (hysteresis) if the output is currently on

;  is output on?

;  yes, add hysteresis

;  yes, add hysteresis

;  else, treat as if Tmeas>Tref

;  if clear, Tmeas>Tref

;  That’s all...

;  turn the LED off