DigiTherm - Digital Thermostat

Designed by: Levente Kincses (levy_21@freemail.hu)

Table of contents

Introduction

This project isn't a conventional one. In the past the thermostats was totally mechanical, with a built in mechanical switch. In the function of temperature, two kinds of irons bend and that caused the switch to turn on-off. Nowadays the thermostats are half- or full digital, which means that they contain a microprocessor to process the value of the temperature. Half digital system has an analogue sensor, which converts the temperature to resistance, and then an analogue to digital (A/D) converter converts it to binary value.In my project I'll describe a full digital system, which uses a digital temperature sensor.It's output is binary coded value of the temperature.

There is various kind of digital sensors in the commercial market, from different manufacturers across the output interface type e.g. (I^2C, SPI, 1-Wire-Bus etc.).I used DS18S20 model delivered from Dallas Semiconductor, which has a special feature of multiple devices on the same 1-Wire Bus.The details of the 1-Wire Bus protocol is completely described in DS18S20 datasheet, so I won't write about in detail.

To display the actually temperature and the menu items, I used five piece of seven-segment displays(I know that I could use Matrix LCD rather than LED, but it's still cheaper and brighter).The displays are driven by a shift register, which makes it possible to use only three wires to display numbers, even letters.But it's not all.Driving five displays through eight wires still impossible if they wouldn't multiplexed, which uses five driver wire for display's common pin.

Installing DigiTherm

Generally the thermostats are mounted on the wall of the room.This device is designed as a wall-thermostat also.There are two mounting holes on the back side of the device, which stand for fixing.After securing DigiTherm, there is one more thing you should do, the electrical installation.DigiTherm requires an external power from the main power line, that means you will need a wall-socket near the device.The second thing that you will need is the wires from the properly wired controlled load or device.

Working description

After turning on the main switch, an initial beep sound hearable and on the displays appear the current ambient temperature. Every three seconds the thermostat's control state is displayed for a half-second.Pressing the Mode button, you'll enter the thermostat's setup mode.Here are the thermostat's min and max temperature settings, that can be adjusted with the two other buttons, plus and minus.

Mods: (Tip: You can change mode by pressing Mode button :-)

1.Indoor temperature monitor with flashing thermostat's control state monitor

This is the mostly used mode, normally the device operates in this.The indoor temperature is displayed and every three seconds the thermostat's control state is appear on seven segment displays for a half second.When the *P off* appears on the display, that means the controlled device is turned off, and when it writes *P on* that's the opposite, the device is on.The thermostat works with three and half digits. The three digits are used to show the integer part of the temperature and the half digit stands for displaying the half part of a Celsius degree.

2.Setup: Set Minimum temperature

This value is the thermostat's minimum temperature, and changeable with plus and minus buttons.When the sensor returns less or equal to previously set minimum value, the thermostat will turn on the controlled load.

3.Setup: Set Maximum temperature

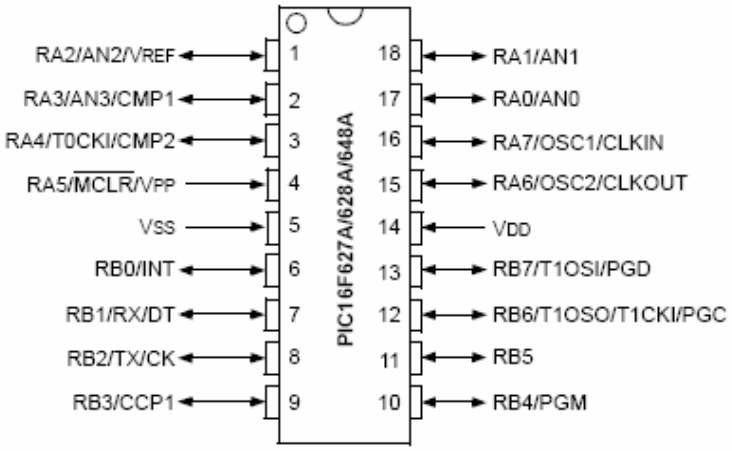
Thermostat's maximum value, also changeable with the plus and minus buttons.When the ambient's (sensor's) temperature reaches or surpass this value, the controlled load will be turned off.

DigiTherm in detail

Hardware:

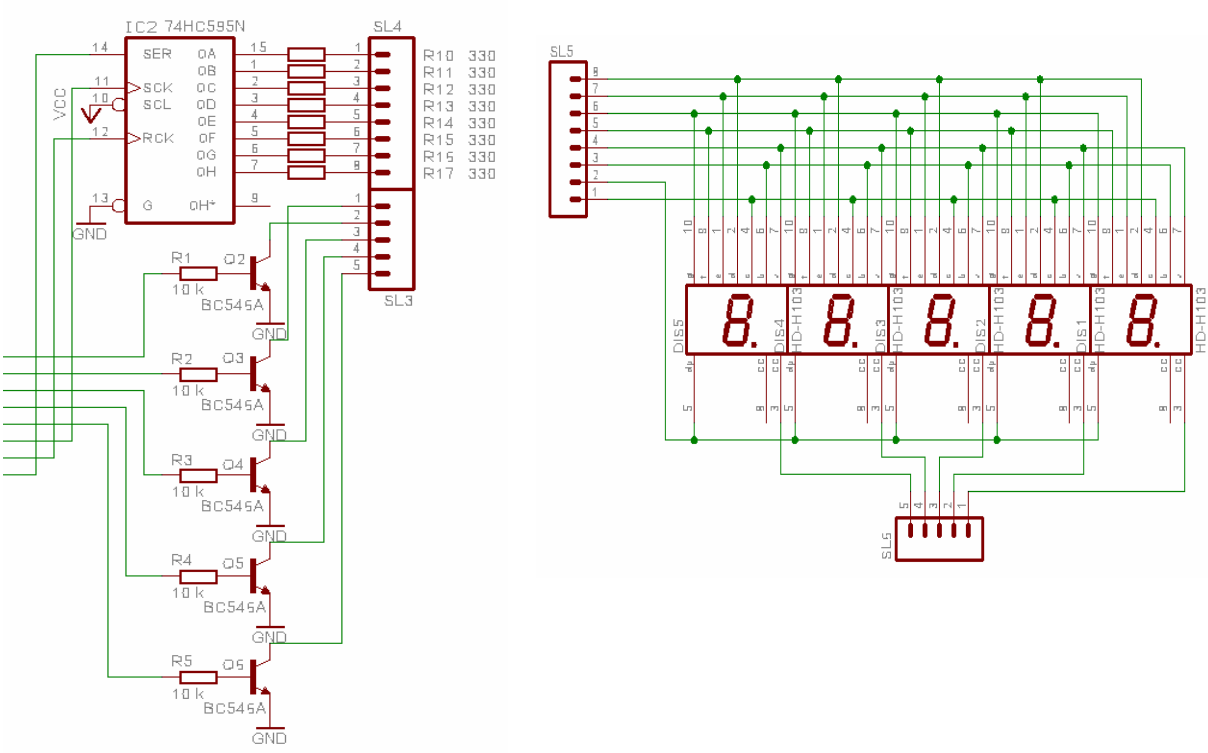
MCU and pinout:

Port	Function
RA 0	Speaker
RA 1	DS18S20
RA 2	Relay
RA 3	Button
RA 4	Button
RA 5	Button
OSC 1	Xtal
OSC 2	Xtal
VSS	GND
VDD	VCC
RB 0	S.R. Serial Data
RB 1	S.R. RCK
RB 2	S.R. Clock
RB 3	Displ. 5 CC
RB 4	Displ. 4 CC
RB 5	Displ. 3 CC
RB 6	Displ. 1 CC
RB 7	Displ. 2 CC

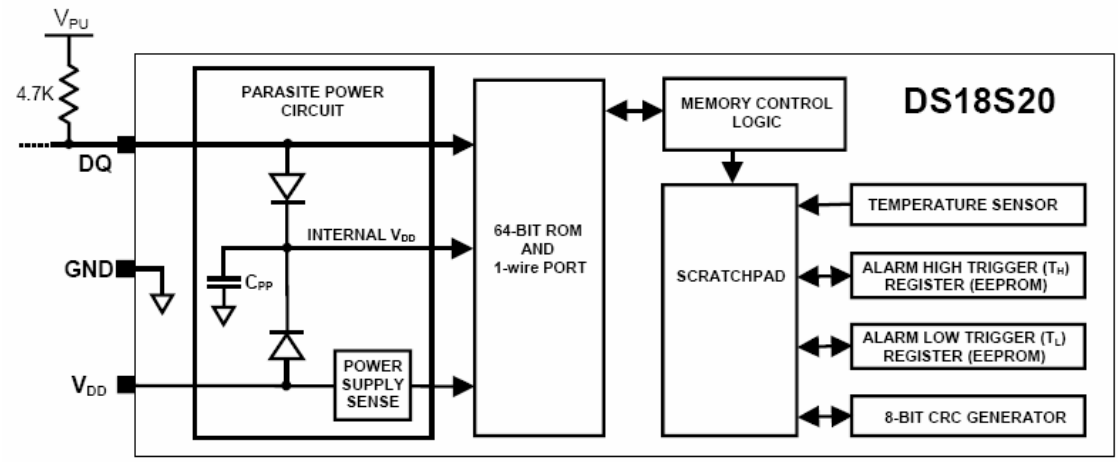


We should start with the display driver part.The main concept was that the thermostat could display the full spectrum of temperature (from -55°C to +125°C) represented by the digital sensor with 0.5°C accuracy.That means minimum four seven-segment displays will be required for displaying, but I added an expansion one to display the sign °C.

Like I wrote before, the target MCU is too small to drive five seven-segment display directly.To decrease the driver lines, the displays are multiplexed.That means in one displaying cycle only one display is showing a number or letter, but the other ones are turned off.In the next cycle the next display will be turned on and so on.The refresh rate for five displays should be between the human eye's disability, which's minimum vary from 15 to 25 refreshes per second.The multiplexing isn't the only thing that can make the MCU's ports to exploit. Here I used an eight bit shift (serial to paralell) register to drive the displays seven segment and a dot point on each digit.This wiring uses only three data port, not eight, like when the shift register isn't presented.



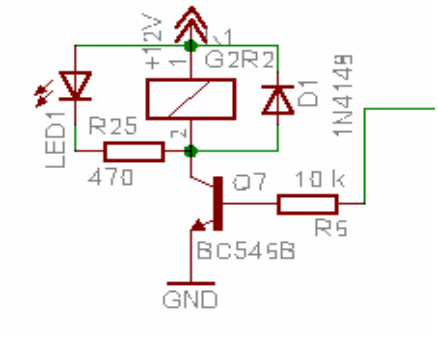
Display section



DS18S20 Temperature sensor block diagram

The DS18S20 digital temperature sensor is connected to the MCU through a data line, which is a two way 1 wire bus.The details about 1-Wire Bus protocol are described in the DS18S20's datasheet.

The load is controlled through a relay switch, that is drived by a transistor.



Relay driver section

Software:

The MCU has few tasks to do during the operation of the device.These are:

1.Communcating with DS18S20 (Issue conversion, Get Temperature value)
2.Temperature value handling, converting to digits
3.Display Refresh (MUX + Shift 5 x Seven Segment Display)
4.Control a load through a relay switch
5.Watch for pressed button and mode
6.Store operation values for switching (EEPROM)

1.Communcating with DS18S20 (Issue conversion, Get Temperature value)

The DS18S20 digital temperature sensor has it's own structure and protocol.Fist of all the MCU should issue a temperature conversion at the sensor:

```
IssueTempConv           ;Routine for Starting Temperature Conversion
call OW_RESET           ;Send Reset Pulse and read for Presence Detect Pulse
btfss PDBYTE,0          ;1 = Presence Detect Detected
goto NOPDPULSE
movlw SKPROM
call DSTXBYTE           ;Send Skip ROM Command (0xCC)
movlw ConvertT
call DSTXBYTE           ;Issue Temperature Converting
return
```

When the sensor finished the conversion, the MCU ask the result and stores it in a GPR:

```
GetTemperature          ;Temperature Measuring
call OW_RESET           ;Send Reset Pulse and read for Presence Detect Pulse
btfss PDBYTE,0          ;1 = Presence Detect Detected
goto NOPDPULSE
movlw SKPROM
call DSTXBYTE           ;Send Skip ROM Command (0xCC)
movlw RDScrPad
call DSTXBYTE           ;Issue Read ScratchPad
call DSRXBYTE           ;Get TemperatureLSB
movfw IOBYTE
movwf TemperatureLSB
call DSRXBYTE           ;Get TemperatureMSB
movfw IOBYTE
movwf TemperatureMSB
call OW_RESET           ;Send Reset Pulse and read for Presence Detect Pulse
btfsc PDBYTE,0          ;1 = Presence Detect Detected
return
NOPDPULSE               ;Add some error processing here!
return                  ;Return from Temperature measuring
```

2. Temperature value handling, converting to digits

After the MCU received the temperature value, a procedure has to process it.It's because the DS18S20 codes the negative temperature value with complement binary numbers. The first byte of the result shows the sign of the temperature value. (0 means positive and 1 means negative temperature) But the LSB bit of the result's second byte shows the half part of a Celsius degree. (0 means 0.0 °C, 1 means 0.5 °C)

```
TempHandling                ;Determine + or - Temperature and handle half Degrees
movlw d'35'                 ;mask of Celsius Degree
movwf Digit1
movfw TemperatureLSB
movwf TempLSB
btfss TemperatureMSB,0      ;when MSB = 1, Temperature is negative
goto Positive
comf TempLSB                ;Complement of Temperature
movlw d'1'
addwf TempLSB
Positive
movlw d'0'
movwf Digit2
btfss TempLSB,0             ;Detect half degree
goto NoHalf
movlw d'5'
movwf Digit2
NoHalf
movfw TempLSB
movwf Temperature
rrf Temperature             ;Convert to integer
bcf Temperature,7
return                      ;Return from Temperature Handling
```

Before displaying the temperature value, the MCU converts the binary value into digits, that the display refresh procedure can handle.

```
Byte2Digit
clrf Digit3
clrf Digit4
movfw Temperature
movwf DigitTemp
movlw d'100'
DoAgain0
        incf Digit5
        subwf DigitTemp
        btfsc STATUS,C
        goto DoAgain0
        decf Digit5
        addwf DigitTemp
        movfw Temperature
        movwf DigitHundreds
```

```
        movlw d'100'
        subwf DigitHundreds
        btfsc STATUS,C
        goto LeaveDsply4ON
        movlw d'46'                     ;Display OFF
        movwf Digit5
        btfss TemperatureMSB,0
        goto LeaveDsply4ON
        movlw d'48'                     ;mask of minus
        movwf Digit5
LeaveDsply4ON
        movlw d'10'
DoAgain1
        incf Digit4
        subwf DigitTemp
        btfsc STATUS,C
        goto DoAgain1
        decf Digit4
        addwf DigitTemp
        movfw Temperature
        movwf DigitHundreds
        movlw d'100'
        subwf DigitHundreds
        btfsc STATUS,C
        goto LeaveDsply3ON
        movfw Digit4
        xorwf d'0'
        btfss STATUS,Z
        goto LeaveDsply3ON
        movlw d'46'                     ;Display OFF
        movwf Digit4
LeaveDsply3ON
        movlw d'1'
DoAgain2
        incf Digit3
        subwf DigitTemp
        btfsc STATUS,C
        goto DoAgain2
        decf Digit3
        addwf DigitTemp
        movlw d'36'                     ;Offset between 0 and 0.
        addwf Digit3
return
```

3.Display Refresh (MUX + Shift 5 x Seven Segment Display)

DigiTherm has five seven-segments displays to communicate with the user.This means that 5 displays x 7+1 segments = 40 LEDs to control! It's too much, because the PIC has only 6 + 8 I/O pins.The question is - HOW is it possible to drive 40 LEDs over 8 I/O pins? Answer is quite simple, like I mentioned above, they are paralell connected and working alternately, but the segments are driven by a shift register.The conclusion is that we need five display driver and three shift register driver I/O pins.This way we spared 32 I/O pins, but the software becomes more complex.

The following code is for sending data to shift register, which will drive the segments.

```
DataSender                ;Routine for SENDING 8 BITS to the SHIFT register
movlw d'8'                ;8 bits will be sent
movwf BitCnt              ;Bit Counter variable for counting the current bit

SendNext                  ;Send Next Bit
btfsc D2Send,0            ;If the 0.BIT is SET then
bsf SSDsplSIN             ;SET the SHIFT reg.'s SERIAL INPUT
btfss D2Send,0            ;If the 0.BIT is CLEAR then
bcf SSDsplSIN             ;CLEAR the SHIFT reg.'s SERIAL INPUT
bsf SSDsplSCLK            ;Generate clock on SHIFT reg.'s SERIAL CLOCK
nop                       ;Delay to SHIFTT Reg. to realize the clock
bcf SSDsplSCLK            ;The clock's 2. half period
rrf D2Send                ;Shift right the content of D2Send, and lose the 0.bit
decfsz BitCnt             ;Decrement Bit Counter,
goto SendNext             ;In case that not all bits sent, continue the sending
bcf SSDsplSIN             ;Clear SHIFT Reg's SERIAL INPUT
bsf SSDsplSEN             ;En. SHIFT Reg's STORAGE to appear the data on disp.
nop                       ;Delay to SHIFTT Reg. to realize the STORAGE Enable
bcf SSDsplSEN             ;Disabel SHIFT Reg's STORAGE to fix data on display
return                    ;Return from data sending routine
```

The displayable characters are stored in a table, called DataTable:

```
DataTable                 ;Table of Data, represents masked value of the characters
addwf PCL                 ;Jump to the proper value, through the Prog. Cnt. Latch
retlw b'10111011'         ;mask of 0
retlw b'10100000'         ;mask of 1
retlw b'00110111'         ;mask of 2
retlw b'10110101'         ;mask of 3
retlw b'10101100'         ;mask of 4
retlw b'10011101'         ;mask of 5
retlw b'10011111'         ;mask of 6
retlw b'10110000'         ;mask of 7
retlw b'10111111'         ;mask of 8
retlw b'10111101'         ;mask of 9
…
Other characters, like letters and some special signs, like °C
…
return
```

Each time the display refreshes, the following procedure will be called:

```
DisplayRefresh                ;Disp Refresh Routine for driving 7Segment Displays
movf Digit1,W                 ;Digit1 to display on DISPLAY 1
call DataTable                ;Find in the D.TABLE the masked value of CHAR
movwf D2Send                  ;Send masked value to the SHIFT register
call DataSender               ;Call Data Sender routine
bsf SSDspl1En                 ;Enable DISP 1
call DisplayDelay             ;Time to display the character
bcf SSDspl1En                 ;Disable DISP 1
….
Four more times the code represented above
….
return
```

This subroutine first calls the DataTable procedure to get the displayable character's binary code (which segments should be on and off). This will be stored in W register after return. The next thing to do is, that send the binary code to the shift register (call DataSender).When the segments are properly configured, it's time to enable the display. Some short delay (few ms) should be to ensure the appearance of the characters.

4.Control a load through a relay switch

The MCU run through the relay switching routine in regular intervalls to keep up-to-date the contolled load's state.

```
RelaySwitching
movfw TemperatureMSB
xorwf WMinMSB
btfsc STATUS,Z
call CheckMinLSB
movfw TemperatureMSB
xorwf WMaxMSB
btfsc STATUS,Z
call CheckMaxLSB
return
CheckMinLSB
movfw TemperatureLSB
subwf WMinLSB
btfsc STATUS,C
bsf RelayPORT
return
CheckMaxLSB
decf WMaxLSB
movfw TemperatureLSB
subwf WMaxLSB
btfss STATUS,C
bcf RelayPORT
return
```

5.Watch for pressed button and mode

In case that the Mode button pressed, the next subroutine will be executed:

```
Mode                    ;Mode routine
BEEP 0xAF, 0x8          ;Mode Button's Sound
movlw d'28'             ;S
movwf Digit5
movlw d'46'             ;Display OFF
movwf Digit4
movlw d'22'             ;M
movwf Digit3
movlw d'18'             ;I
movwf Digit2
movlw d'51'             ;invert N
movwf Digit1
call LongDelay          ;Long Delay (1 s) with Display Refresh
call SetMin
BEEP 0xAF, 0x8          ;Mode Button's Sound
movlw d'28'             ;S
movwf Digit5
movlw d'46'             ;Display OFF
movwf Digit4
movlw d'22'             ;M
movwf Digit3
movlw d'10'             ;A
movwf Digit2
movlw d'52'             ;invert X
movwf Digit1
call LongDelay          ;Long Delay (1 s) with Display Refresh
call SetMax
BEEP 0xAF, 0x8          ;Mode Button's Sound
movlw d'18'             ;I
movwf Digit5
movlw d'23'             ;N
movwf Digit4
movlw d'46'             ;Display OFF
movwf Digit3
movlw d'29'             ;T
movwf Digit2
movlw d'50'             ;invert M
movwf Digit1
call LongDelay          ;Long Delay (1 s) with Display Refresh
call InitRelaySwitching
BEEP 0xAF, 0x8          ;Mode Button's Sound
call LongDelay          ;Long Delay (1 s) with Display Refresh
call InitRelaySwitching
BEEP 0xAF, 0x8          ;Mode Button's Sound
goto Main_Loop          ;End of Mode routine
```

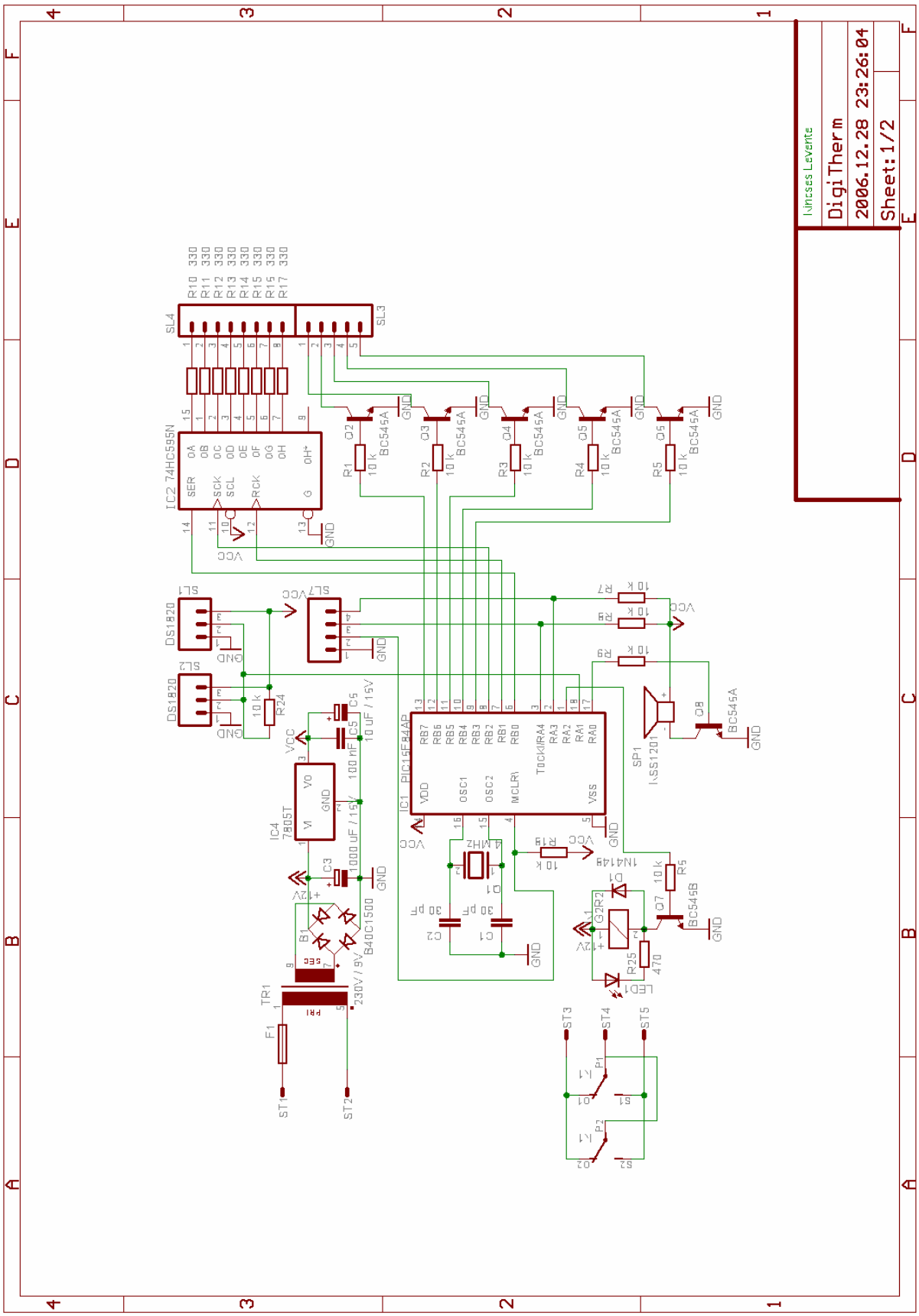
6.Store operation values for switching (EEPROM)

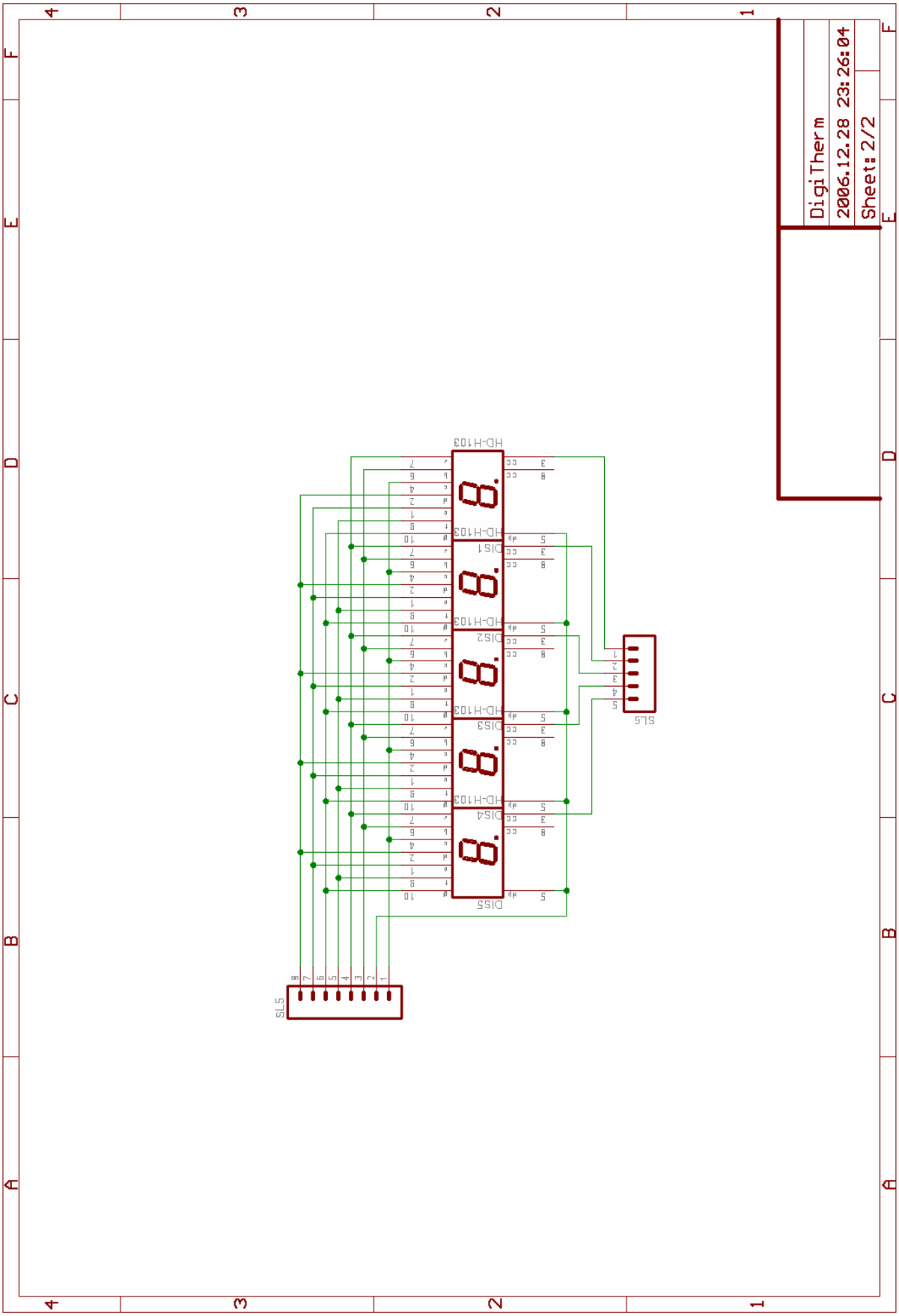
DigiTherm has the feature to store the preset temperature values even if no power supplied.The MCU's EEPROM makes it possible, that isn't requires any power to hold the data.

```
SetMin
banksel EEADR                ;Bank 1
MOVLW d'0'                   ;MIN H Value from 00 Position of the EEPROM
MOVWF EEADR                  ;Address to read
BSF EECON1, RD               ;EE Read
MOVF EEDATA, W               ;W = EEDATA
banksel PORTA
movwf TemperatureMSB
banksel EEADR
MOVLW d'1'                   ;MIN L Value from 01 Position of the EEPROM
MOVWF EEADR                  ;Address to read
BSF EECON1, RD               ;EE Read
MOVFW EEDATA                 ;W = EEDATA
banksel PORTA                ;Bank 0
movwf TemperatureLSB
...
The preset value changer subroutine
...
movfw TemperatureMSB
BSF STATUS, RP0              ;Bank 1
MOVWF EEDATA                 ;W = EEDATA
MOVLW d'0'                   ;MIN L Value from 00 Position of the EEPROM
MOVWF EEADR                  ;Address to read
BSF EECON1, WREN             ;Enable write
BCF INTCON, GIE              ;Disable INTs.
MOVLW 0x55
MOVWF EECON2                 ;Write 55h
MOVLW 0xAA
MOVWF EECON2                 ;Write AAh
BSF EECON1,WR                ;Set WR bit
BCF STATUS, RP0              ;begin write
                             ;Write NEW MIN Value
movlw d'31'                  ;V
movwf Digit5
movlw d'10'                  ;A
movwf Digit4
movlw d'18'                  ;I
movwf Digit3
movlw d'29'                  ;T
movwf Digit2
movlw d'46'                  ;Disp. OFF
movwf Digit1
call LongDelay               ;Long Delay (1 s) with Display Refresh
movfw TemperatureLSB
```

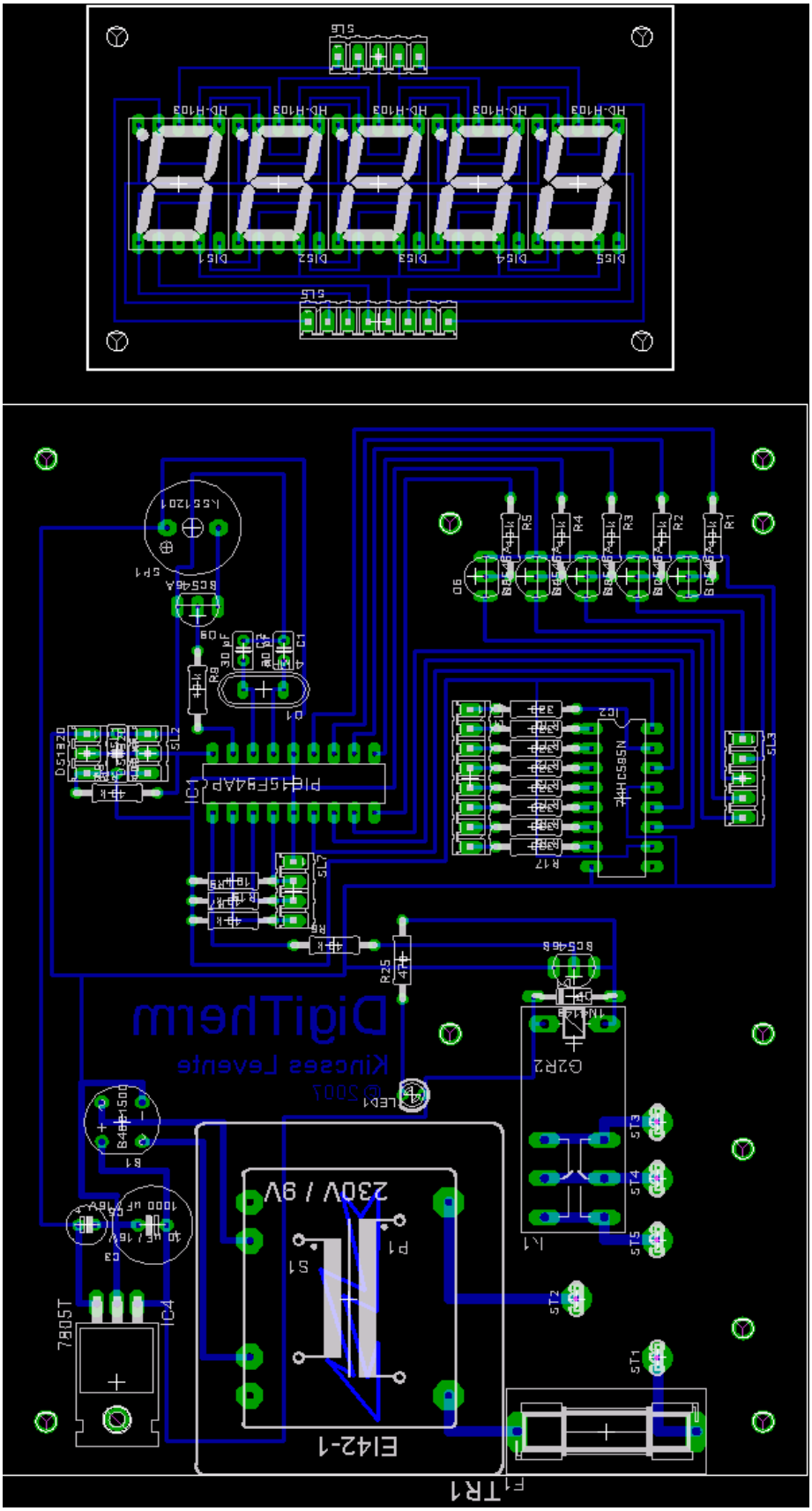
```
BSF STATUS, RP0
MOVWF EEDATA            ;W = EEDATA
MOVLW d'1'              ;MIN L Value from 00 Position of the EEPROM
MOVWF EEADR             ;Address to read
BSF EECON1, WREN        ;Enable write
BCF INTCON, GIE         ;Disable INTs.
MOVLW 0x55
MOVWF EECON2            ;Write 55h
MOVLW 0xAA
MOVWF EECON2            ;Write AAh
BSF EECON1,WR           ;Set WR bit
                        ;begin write
BCF STATUS, RP0         ;Bank 1
movlw d'31'             ;V
movwf Digit5
movlw d'10'             ;A
movwf Digit4
movlw d'18'             ;I
movwf Digit3
movlw d'29'             ;T
movwf Digit2
movlw d'46'             ;Disp. OFF
movwf Digit1
call LongDelay          ;Long Delay (1 s) with Display Refresh
return
```

DigiTherm schematics




DigiTherm
2006.12.28 23:26:04
Sheet: 2/2
HD-H103
DIS1
DIS2
DIS3
DIS4
DIS5
SL5
SL5

DigiTherm PCB



ASM Source Code

```
;-------------------------------------------------------------------

errorlevel -302, -305, -205, -203, -206, -204, -205     ;Disable showing warninngs
PROCESSOR 16f627                                        ;Processor type
#include "p16f627.inc"                                  ;Include processor's library

;-------------------------------------------------------------------
                                                        ;These constants are standard DS 1-Wire ROM commands
constant SRCHROM=0xF0                                   ;Search ROM
constant RDROM=0x33                                     ;Read ROM
constant MTCHROM=0x55                                   ;Match ROM
constant SKPROM=0xCC                                    ;Skip ROM
constant ConvertT=0x44                                  ;Start Temperature Convertion
constant RDScrPad=0xBE                                  ;Read ScratchPad

constant DQ=1                                           ;DS Sensor data pin on PortA bit no. 1

#define BEEPport PORTA,0                                ;Beeper's Port
#define BEEPtris TRISA,0                                ;Beeper's TRIS Reg.

#define RelayPORT PORTA,2                               ;Relay's Control Port

#define SSDspl1En PORTB,6                               ;1. Seven Segment Display Enable Port
#define SSDspl2En PORTB,7                               ;2. Seven Segment Display Enable Port
#define SSDspl3En PORTB,5                               ;3. Seven Segment Display Enable Port
#define SSDspl4En PORTB,4                               ;4. Seven Segment Display Enable Port
#define SSDspl5En PORTB,3                               ;5. Seven Segment Display Enable Port
#define SSDsplSIN PORTB,0                               ;Seven Segment Display SerialData Port
#define SSDsplSCLK PORTB,2                              ;Seven Segment Display SerialClock Port
#define SSDsplSEN PORTB,1                               ;Seven Segment Display SerialData Valid Port

;-------------------------------------------------------------------

cblock 0x20                                             ;Address of RAM
;.........                                              ;Display Driver's Variables
Digit1                                                  ;1. Digit's value
Digit2                                                  ;2. Digit's value
Digit3                                                  ;3. Digit's value
Digit4                                                  ;4. Digit's value
Digit5                                                  ;5. Digit's value
D2Send                                                  ;SHIFT Reg's Data to Send
BitCnt                                                  ;Bit Counter count's the SHIFTed bits
DDlyCnt                                                 ;Display Driver's DELAY value
SDDlyCnt                                                ;Display Driver's DELAY2 value
;.........                                              ;Byte 2 Digits Converter's Variables
DigitTemp                                               ;Temporary Value for Calculating Digits
DigitHundreds
;.........                                              ;Beeper's Variables
HIcnt
LOcnt
LOOPcnt
PRESCwait
Beep_TEMP1
Beep_TEMP2
Beep_TEMP3
;.........                                              ;DS 18S20's Variables
IOBYTE
TMP0                                                    ; Address 0x23
COUNT                                                   ; Keep track of bits
PICMSB                                                  ; Store the MSB
PICLSB                                                  ; Store the LSB
PDBYTE                                                  ; Presence Detect Pulse
TemperatureLSB                                          ;0. byte of ScratchPad
TemperatureMSB                                          ;1. byte of ScratchPad
TH                                                      ;2. byte of ScratchPad
TL                                                      ;3. byte of ScratchPad
Res1                                                    ;4. byte of ScratchPad
Res2                                                    ;5. byte of ScratchPad
CountRemain                                             ;6. byte of ScratchPad
CountPerC                                               ;7. byte of ScratchPad
CRC                                                     ;8. byte of ScratchPad
```

```
Temperature                             ;Value of the Real Temperature
;.........                              ;Mode Variables
LDlyCnt
;.........
TempLSB
;.........
MinMSB
MinLSB
MaxMSB
MaxLSB
WMinMSB
WMinLSB
WMaxMSB
WMaxLSB
;.........
CntTemp
endc                                    ;End of the Variables declaration

;-----------------------------------------------------------------

org         0x0000                      ;On RESET
 goto Start                             ;GoTo Start
org         0x0004                      ;On INTERRUPT
 goto ISR                               ;GoTo ISR

;-----------------------------------------------------------------
#include "1W_16F6X.inc"                 ;Dallas 1-Wire Bus Driver
#include "Beeper.inc"                   ;Beeper's Driver
;-----------------------------------------------------------------

DataTable                               ;Table of Data, which represents the masked value of the characters
addwf PCL                               ;Jump to the proper value, through the Program Counter Latch
retlw b'10111011'                       ;mask of 0
retlw b'10100000'                       ;mask of 1
retlw b'00110111'                       ;mask of 2
retlw b'10110101'                       ;mask of 3
retlw b'10101100'                       ;mask of 4
retlw b'10011101'                       ;mask of 5
retlw b'10011111'                       ;mask of 6
retlw b'10110000'                       ;mask of 7
retlw b'10111111'                       ;mask of 8
retlw b'10111101'                       ;mask of 9
retlw b'10111110'                       ;mask of A                      10
retlw b'10001111'                       ;mask of B
retlw b'00000111'                       ;mask of C
retlw b'10100111'                       ;mask of D
retlw b'00011111'                       ;mask of E
retlw b'00011110'                       ;mask of F
retlw b'10111100'                       ;mask of G
retlw b'10001110'                       ;mask of H
retlw b'10000000'                       ;mask of I
retlw b'10100001'                       ;mask of J
retlw b'10001010'                       ;mask of K                      20
retlw b'00001011'                       ;mask of L
retlw b'10111010'                       ;mask of M
retlw b'10000110'                       ;mask of N
retlw b'10000111'                       ;mask of O
retlw b'00111110'                       ;mask of P
retlw b'10111100'                       ;mask of Q
retlw b'00000110'                       ;mask of R
retlw b'10011101'                       ;mask of S
retlw b'00001111'                       ;mask of T
retlw b'10000011'                       ;mask of U                      30
retlw b'10101011'                       ;mask of V
retlw b'10101110'                       ;mask of X
retlw b'10101101'                       ;mask of Y
retlw b'00110111'                       ;mask of Z
retlw b'00011111'                       ;mask of C degree
retlw b'11111011'                       ;mask of 0.
retlw b'11100000'                       ;mask of 1.
retlw b'01110111'                       ;mask of 2.
retlw b'11110101'                       ;mask of 3.
retlw b'11101100'                       ;mask of 4.                     40
retlw b'11011101'                       ;mask of 5.
retlw b'11011111'                       ;mask of 6.
retlw b'11110000'                       ;mask of 7.
```

```
retlw b'11111111'                       ;mask of 8.
retlw b'11111101'                       ;mask of 9.
retlw b'00000000'                       ;Display OFF
retlw b'01000000'                       ;Dot Point
retlw b'00000100'                       ;mask of -
retlw b'01011010'                       ;mask of invert F
retlw b'10111010'                       ;mask of invert M
retlw b'11000010'                       ;mask of invert N
retlw b'11101010'                       ;mask of invert X
;--------------------------------------------------------------------

Start                                   ;Start-Up Initialising

banksel CMCON                           ;Select CMCON's bank (0)
movlw b'00000111'                       ;Turn off Comparators
movwf CMCON

banksel TRISA                           ;Select TRISA's bank (1)
movlw b'00111000'                       ;RA0-RA2 OUTPUTS, RA3-RA5 INPUTS, RA6-RA7 OSCILLATOR
movwf TRISA
movlw b'00000000'                       ;RB0-RB7 OUTPUTS
movwf TRISB

banksel PORTA                           ;Select PORTA's bank (0)
clrf PORTA                              ;CLEAR all bits of PORTA
clrf PORTB                              ;CLEAR all bits of PORTB

clrf Digit1                             ;CLEAR Digit1
clrf Digit2                             ;CLEAR Digit2
clrf Digit3                             ;CLEAR Digit3
clrf Digit4                             ;CLEAR Digit4
clrf Digit5                             ;CLEAR Digit5
clrf Temperature                        ;CLEAR Temperature

BEEPinit                                ;Initializing the Beeper
BEEP 0xFF, 0x8                          ;Generate Initial BEEP Sound
call InitRelaySwitching

;--------------------------------------------------------------------

Main_Loop                               ;Main Loop Program

call IssueTempConv                      ;Start Temperature Conversion
call GetTemperature                     ;Get Value of Temperature
call TempHandling                       ;Convert and Calculate Incoming Data
call RelaySwitching                     ;Routine for comparing current temp. and stored
call TempORDemo                         ;Switch displaying between temp. and relay's status
call DisplayRefresh                     ;Redraw Characters on Display

btfss PORTA,5                           ;Enter MODE
goto Mode                               ;when Mode button is pressed

goto Main_Loop                          ;GoTo the start of the Main Loop Program

;--------------------------------------------------------------------

ISR                                     ;INTERRUPT SERVICE ROUTINE

return                                  ;RETURN from INTERRUPT

;--------------------------------------------------------------------

Mode                                    ;Mode routine

BEEP 0xAF, 0x8                          ;Mode Button's Sound

movlw d'28'                             ;S
movwf Digit5
movlw d'46'                             ;Display OFF
movwf Digit4
movlw d'22'                             ;M
movwf Digit3
movlw d'18'                             ;I
movwf Digit2
movlw d'51'                             ;invert N
movwf Digit1
```

```
call LongDelay                  ;Long Delay (1 s) with Display Refresh
call SetMin
BEEP 0xAF, 0x8                  ;Mode Button's Sound
movlw d'28'                     ;S
movwf Digit5
movlw d'46'                     ;Display OFF
movwf Digit4
movlw d'22'                     ;M
movwf Digit3
movlw d'10'                     ;A
movwf Digit2
movlw d'52'                     ;invert X
movwf Digit1
call LongDelay                  ;Long Delay (1 s) with Display Refresh
call SetMax
BEEP 0xAF, 0x8                  ;Mode Button's Sound
movlw d'18'                     ;I
movwf Digit5movlw d'23'         ;N
movwf Digit4
movlw d'46'                     ;Display OFF
movwf Digit3
movlw d'29'                     ;T
movwf Digit2
movlw d'50'                     ;invert M
movwf Digit1
call LongDelay                  ;Long Delay (1 s) with Display Refresh
call InitRelaySwitching
BEEP 0xAF, 0x8                  ;Mode Button's Sound

goto Main_Loop                  ;End of Mode routine

;-----------------------------------------------------------------

SetMin

banksel EEADR                   ;Bank 1
MOVLW d'0'                      ;MIN H Value from 00 Position of the EEPROM
MOVWF EEADR                     ;Address to read
BSF EECON1, RD                  ;EE Read
MOVF EEDATA, W                  ;W = EEDATA
banksel PORTA
movwf TemperatureMSB
banksel EEADR
MOVLW d'1'                      ;MIN L Value from 01 Position of the EEPROM
MOVWF EEADR                     ;Address to read
BSF EECON1, RD                  ;EE Read
MOVFW EEDATA                    ;W = EEDATA
banksel PORTA                   ;Bank 0
movwf TemperatureLSB

SecondTime

call TempHandling
call Byte2Digit
call DisplayRefresh             ;Redraw Characters on Display

btfss PORTA,3                   ;UP button
call UPButton
call DisplayRefresh             ;Redraw Characters on Display
btfss PORTA,4                   ;Down button
call DownButton

btfsc PORTA,5                   ;Enter MODE
goto SecondTime                 ;when Mode button is pressed

movfw TemperatureMSB
BSF STATUS, RP0                 ;Bank 1
MOVWF EEDATA                    ;W = EEDATA
MOVLW d'0'                      ;MIN L Value from 00 Position of the EEPROM
MOVWF EEADR                     ;Address to read
BSF EECON1, WREN                ;Enable write
BCF INTCON, GIE                 ;Disable INTs.
MOVLW 0x55                      ;
MOVWF EECON2                    ;Write 55h
MOVLW 0xAA                      ;
MOVWF EECON2                    ;Write AAh
```

```
BSF EECON1,WR                   ;Set WR bit
BCF STATUS, RP0                 ;begin write
;BSF INTCON, GIE                ;Enable INTs
                                ;Write NEW MIN Value
movlw d'31'                     ;V
movwf Digit5
movlw d'10'                     ;A
movwf Digit4
movlw d'18'                     ;I
movwf Digit3
movlw d'29'                     ;T
movwf Digit2
movlw d'46'                     ;Disp. OFF
movwf Digit1
call LongDelay                  ;Long Delay (1 s) with Display Refresh
movfw TemperatureLSB
BSF STATUS, RP0
MOVWF EEDATA                    ;W = EEDATA
MOVLW d'1'                      ;MIN L Value from 00 Position of the EEPROM
MOVWF EEADR                     ;Address to read
BSF EECON1, WREN                ;Enable write
BCF INTCON, GIE                 ;Disable INTs.
MOVLW 0x55                      ;
MOVWF EECON2                    ;Write 55h
MOVLW 0xAA                      ;
MOVWF EECON2                    ;Write AAh
BSF EECON1,WR                   ;Set WR bit
                                ;begin write
;BSF INTCON, GIE                ;Enable INTs
BCF STATUS, RP0                 ;Bank 1
movlw d'31'                     ;V
movwf Digit5
movlw d'10'                     ;A
movwf Digit4
movlw d'18'                     ;I
movwf Digit3
movlw d'29'                     ;T
movwf Digit2
movlw d'46'                     ;Disp. OFF
movwf Digit1
call LongDelay                  ;Long Delay (1 s) with Display Refresh
return

SetMax

banksel EEADR                   ;Bank 1
MOVLW d'2'                      ;MAX H Value from 02 Position of the EEPROM
MOVWF EEADR                     ;Address to read
BSF EECON1, RD                  ;EE Read
MOVF EEDATA, W                  ;W = EEDATA
banksel PORTA
movwf TemperatureMSB
banksel EEADR
MOVLW d'3'                      ;MAX L Value from 03 Position of the EEPROM
MOVWF EEADR                     ;Address to read
BSF EECON1, RD                  ;EE Read
MOVFW EEDATA                    ;W = EEDATA
banksel PORTA                   ;Bank 0
movwf TemperatureLSB

ThirdTime

call TempHandling
call Byte2Digit
call DisplayRefresh             ;Redraw Characters on Display

btfss PORTA,3                   ;UP button
call UPButton
call DisplayRefresh             ;Redraw Characters on Display
btfss PORTA,4                   ;Down button
call DownButton

btfsc PORTA,5                   ;Enter MODE
goto ThirdTime                  ;when Mode button is pressed
                                ;Write NEW MAX Value
movfw TemperatureMSB
```

```
BSF STATUS, RP0                 ;Bank 1
MOVWF EEDATA                    ;W = EEDATA
MOVLW d'2'                      ;MIN L Value from 00 Position of the EEPROM
MOVWF EEADR                     ;Address to read
BSF EECON1, WREN                ;Enable write
BCF INTCON, GIE                 ;Disable INTs.
MOVLW 0x55                      ;
MOVWF EECON2                    ;Write 55h
MOVLW 0xAA                      ;
MOVWF EECON2                    ;Write AAh
BSF EECON1,WR                   ;Set WR bit
BCF STATUS, RP0                 ;Bank 1
                                ;begin write
;BSF INTCON, GIE                ;Enable INTs
                                ;Write NEW MIN Value
movlw d'31'                     ;V
movwf Digit5
movlw d'10'                     ;A
movwf Digit4
movlw d'18'                     ;I
movwf Digit3
movlw d'29'                     ;T
movwf Digit2
movlw d'46'                     ;Disp. OFF
movwf Digit1
call LongDelay                  ;Long Delay (1 s) with Display Refresh
movfw TemperatureLSB
BSF STATUS, RP0                 ;Bank 1
MOVWF EEDATA                    ;W = EEDATA
MOVLW d'3'                      ;MIN L Value from 00 Position of the EEPROM
MOVWF EEADR                     ;Address to read
BSF EECON1, WREN                ;Enable write
BCF INTCON, GIE                 ;Disable INTs.
MOVLW 0x55                      ;
MOVWF EECON2                    ;Write 55h
MOVLW 0xAA                      ;
MOVWF EECON2                    ;Write AAh
BSF EECON1,WR                   ;Set WR bit
                                ;begin write
;BSF INTCON, GIE                ;Enable INTs
BCF STATUS, RP0                 ;Bank 1
movlw d'31'                     ;V
movwf Digit5
movlw d'10'                     ;A
movwf Digit4
movlw d'18'                     ;I
movwf Digit3
movlw d'29'                     ;T
movwf Digit2
movlw d'46'                     ;Disp. OFF
movwf Digit1
call LongDelay                  ;Long Delay (1 s) with Display Refresh
return

UPButton
BEEP 0x7F, 0x02                 ;Mode Button's Sound
btfsc TemperatureMSB,0
goto NegTem
movfw TemperatureLSB
sublw d'249'
btfss STATUS,C
return
NegTem
incfsz TemperatureLSB
return
comf TemperatureMSB
return

DownButton
BEEP 0x7F, 0x02                 ;Mode Button's Sound
btfss TemperatureMSB,0
goto PosTem
movfw TemperatureLSB
sublw d'146'
btfsc STATUS,C
return
```

```
PosTem
decf TemperatureLSB
movfw TemperatureLSB
movwf TempLSB
movlw 0xFF
xorwf TempLSB
btfsc STATUS,Z
comf TemperatureMSB
return

;-------------------------------------------------------------------

DisplayRefresh                  ;Display Refresh Routine for driving 5 MUX and SHIFTed 7Segment Displays

movf Digit1,W                   ;Digit1 to display on DISPLAY 1
call DataTable                  ;Find in the DATATABLE the masked value of CHARACTER
movwf D2Send                    ;Send masked value to the SHIFT register
call DataSender                 ;Call Data Sender routine
bsf SSDspl1En                   ;Enable DISP 1
call DisplayDelay               ;Time to display the character

movf Digit2,W                   ;Digit2 to display on DISPLAY 2
call DataTable                  ;Find in the DATATABLE the masked value of CHARACTER
bcf SSDspl1En                   ;Disable DISP 1
movwf D2Send                    ;Send masked value to the SHIFT register
call DataSender                 ;Call Data Sender routine
bsf SSDspl2En                   ;Enable DISP 2
call DisplayDelay               ;Time to display the character

movf Digit3,W                   ;Digit3 to display on DISPLAY 3
call DataTable                  ;Find in the DATATABLE the masked value of CHARACTER
bcf SSDspl2En                   ;Disable DISP 2
movwf D2Send                    ;Send masked value to the SHIFT register
call DataSender                 ;Call Data Sender routine
bsf SSDspl3En                   ;Enable DISP 3
call DisplayDelay               ;Time to display the character

movf Digit4,W                   ;Digit4 to display on DISPLAY 4
call DataTable                  ;Find in the DATATABLE the masked value of CHARACTER
bcf SSDspl3En                   ;Disable DISP 3
movwf D2Send                    ;Send masked value to the SHIFT register
call DataSender                 ;Call Data Sender routine
bsf SSDspl4En                   ;Enable DISP 4
call DisplayDelay               ;Time to display the character

movf Digit5,W                   ;Digit5 to display on DISPLAY 5
call DataTable                  ;Find in the DATATABLE the masked value of CHARACTER
bcf SSDspl4En                   ;Disable DISP 4
movwf D2Send                    ;Send masked value to the SHIFT register
call DataSender                 ;Call Data Sender routine
bsf SSDspl5En                   ;Enable DISP 5
call DisplayDelay               ;Time to display the character

bcf SSDspl5En                   ;Disable DISP 5

return

;-------------------------------------------------------------------

DisplayDelay                    ;Delay routine to DISPLAY
clrf DDlyCnt
movlw d'2'
movwf SDDlyCnt
NotYet                          ;Not Yet Elapsed the time
decfsz DDlyCnt                  ;Decrement the Display Delay Counter
goto NotYet                     ;If still not ZERO, then go back and do again
decfsz SDDlyCnt                 ;Decrement the Display Delay Counter
goto NotYet                     ;If still not ZERO, then go back and do again
return                          ;Return from Display Delay Routine

;-------------------------------------------------------------------

DataSender                      ;Routine for SENDING 8 BITS to the SHIFT register
movlw d'8'                      ;8 bits will be sent
movwf BitCnt                    ;Bit Counter variable for counting the current bit
```

```
SendNext                                ;Send Next Bit
btfsc D2Send,0                          ;If the 0.BIT is SET then
bsf SSDsplSIN                           ;SET the PORTB's 0.bit (SHIFT reg.'s SERIAL INPUT)
btfss D2Send,0                          ;If the 0.BIT is CLEAR then
bcf SSDsplSIN                           ;CLEAR the PORTB's 0.bit (SHIFT reg.'s SERIAL INPUT)
bsf SSDsplSCLK                          ;Generate clock on PORTB's 2.bit (SHIFT reg.'s SERIAL CLOCK)
nop                                     ;Delay to SHIFTT Reg. to realize the clock
bcf SSDsplSCLK                          ;The clock's 2. half period
rrf D2Send                              ;Shift right the content of D2Send, and lose the 0.bit
decfsz BitCnt                           ;Decrement Bit Counter,
goto SendNext                           ;In case that not all bits sent, continue the sending process

bcf SSDsplSIN                           ;Clear SHIFT Reg's SERIAL INPUT
bsf SSDsplSEN                           ;Enable SHIFT Reg's STORAGE Reg. to appear the data on DISPLAY
nop                                     ;Delay to SHIFTT Reg. to realize the STORAGE Enable
bcf SSDsplSEN                           ;Enable SHIFT Reg's STORAGE Reg. to fix data on DISPLAY
return                                  ;Return from data sending routine

;-----------------------------------------------------------------

IssueTempConv                           ;Routine for Starting Temperature Conversion
call OW_RESET                           ;Send Reset Pulse and read for Presence Detect Pulse
btfss PDBYTE,0                          ;1 = Presence Detect Detected
goto NOPDPULSE
movlw SKPROM
call DSTXBYTE                           ;Send Skip ROM Command (0xCC)
movlw ConvertT
call DSTXBYTE                           ;Issue Temperature Converting
return

;-----------------------------------------------------------------

GetTemperature                          ;Temperature Measuring

call OW_RESET                           ;Send Reset Pulse and read for Presence Detect Pulse
btfss PDBYTE,0                          ;1 = Presence Detect Detected
goto NOPDPULSE
movlw SKPROM
call DSTXBYTE                           ;Send Skip ROM Command (0xCC)
movlw RDScrPad
call DSTXBYTE                           ;Issue Read ScratchPad
call DSRXBYTE                           ;Get TemperatureLSB
movfw IOBYTE
movwf TemperatureLSB
call DSRXBYTE                           ;Get TemperatureMSB
movfw IOBYTE
movwf TemperatureMSB
call DSRXBYTE                           ;Get Temperature High
movfw IOBYTE
movwf TH
call DSRXBYTE                           ;Get Temperature Low
movfw IOBYTE
movwf TL
;call DSRXBYTE                          ;Get Reserved 1
;movfw IOBYTE
;movwf Res1
;call DSRXBYTE                          ;Get Reserved 2
;movfw IOBYTE
;movwf Res2
;call DSRXBYTE                          ;Get CountRemain
;movfw IOBYTE
;movwf CountRemain
;call DSRXBYTE                          ;Get CountPerC
;movfw IOBYTE
;movwf CountPerC
;call DSRXBYTE                          ;Get Cyclic Redundancy Checks
;movfw IOBYTE
;movwf CRC
call OW_RESET                           ;Send Reset Pulse and read for Presence Detect Pulse
btfsc PDBYTE,0                          ;1 = Presence Detect Detected
return

NOPDPULSE                               ;Add some error processing here!

return                                  ;Return from Temperature measuring
```

```
;-------------------------------------------------------------------

TempHandling                            ;Determine + or - Temperature and handle half Degrees

movlw d'35'                             ;mask of Celsius Degree
movwf Digit1
movfw TemperatureLSB
movwf TempLSB
btfss TemperatureMSB,0                  ;when MSB = 1, Temperature is negative
goto Positive
comf TempLSB                            ;Complement of Temperature
movlw d'1'
addwf TempLSB
Positive
movlw d'0'
movwf Digit2
btfss TempLSB,0                         ;Detect half degree
goto NoHalf
movlw d'5'
movwf Digit2
NoHalf
movfw TempLSB
movwf Temperature
rrf Temperature                         ;Convert to integer
bcf Temperature,7
return                                  ;Return from Temperature Handling

;-------------------------------------------------------------------

Byte2Digit
        clrf Digit3
        clrf Digit4
        clrf Digit5

        movfw Temperature
        movwf DigitTemp


        movlw d'100'
DoAgain0
        incf Digit5
        subwf DigitTemp
        btfsc STATUS,C
        goto DoAgain0
        decf Digit5
        addwf DigitTemp
        movfw Temperature
        movwf DigitHundreds
        movlw d'100'
        subwf DigitHundreds
        btfsc STATUS,C
        goto LeaveDsply4ON
        movlw d'46'                     ;Display OFF
        movwf Digit5
        btfss TemperatureMSB,0
        goto LeaveDsply4ON
        movlw d'48'                     ;mask of minus
        movwf Digit5
LeaveDsply4ON
        movlw d'10'
DoAgain1
        incf Digit4
        subwf DigitTemp
        btfsc STATUS,C
        goto DoAgain1
        decf Digit4
        addwf DigitTemp
        movfw Temperature
        movwf DigitHundreds
        movlw d'100'
        subwf DigitHundreds
        btfsc STATUS,C
        goto LeaveDsply3ON
        movfw Digit4
        xorwf d'0'
        btfss STATUS,Z
```

```
            goto LeaveDsply3ON
            movlw d'46'                 ;Display OFF
            movwf Digit4
LeaveDsply3ON
            movlw d'1'
DoAgain2
            incf Digit3
            subwf DigitTemp
            btfsc STATUS,C
            goto DoAgain2
            decf Digit3
            addwf DigitTemp
            movlw d'36'                 ;Offset between 0 and 0.
            addwf Digit3

return

;-------------------------------------------------------------------

LongDelay                               ;Long Delay Routine ( 2 s )
movlw d'203'
movwf LDlyCnt

LDWait                                  ;Not Yet Elapsed the time
call DisplayRefresh
decfsz LDlyCnt                          ;Decrement the Delay Counter
goto LDWait                             ;If still not ZERO, then go back and do again

return                                  ;End of Long Delay Routine

;-------------------------------------------------------------------

InitRelaySwitching

banksel EEADR                           ;Bank 1
MOVLW d'0'                              ;MIN H Value from 00 Position of the EEPROM
MOVWF EEADR                             ;Address to read
BSF EECON1, RD                          ;EE Read
MOVF EEDATA, W                          ;W = EEDATA
banksel PORTA
movwf MinMSB
banksel EEADR
MOVLW d'1'                              ;MIN L Value from 01 Position of the EEPROM
MOVWF EEADR                             ;Address to read
BSF EECON1, RD                          ;EE Read
MOVFW EEDATA                            ;W = EEDATA
banksel PORTA                           ;Bank 0
movwf MinLSB
banksel EEADR                           ;Bank 1
MOVLW d'2'                              ;MIN H Value from 00 Position of the EEPROM
MOVWF EEADR                             ;Address to read
BSF EECON1, RD                          ;EE Read
MOVF EEDATA, W                          ;W = EEDATA
banksel PORTA
movwf MaxMSB
banksel EEADR
MOVLW d'3'                              ;MIN L Value from 01 Position of the EEPROM
MOVWF EEADR                             ;Address to read
BSF EECON1, RD                          ;EE Read
MOVFW EEDATA                            ;W = EEDATA
banksel PORTA                           ;Bank 0
movwf MaxLSB
return

;-------------------------------------------------------------------

RelaySwitching
movfw MinMSB
movwf WMinMSB
movfw MaxMSB
movwf WMaxMSB
movfw MinLSB
movwf WMinLSB
movfw MaxLSB
movwf WMaxLSB
```

```
movfw TemperatureMSB
xorwf WMinMSB
btfsc STATUS,Z
call CheckMinLSB
movfw TemperatureMSB
xorwf WMaxMSB
btfsc STATUS,Z
call CheckMaxLSB
return

CheckMinLSB
movfw TemperatureLSB
subwf WMinLSB
btfsc STATUS,C
bsf RelayPORT
return

CheckMaxLSB
decf WMaxLSB
movfw TemperatureLSB
subwf WMaxLSB
btfss STATUS,C
bcf RelayPORT
return

;-------------------------------------------------------------------

Demo
btfss RelayPORT
goto OFF
movlw d'25'                             ;P
movwf Digit5
movlw d'46'                             ;Display OFF
movwf Digit4
movlw d'24'                             ;O
movwf Digit3
movlw d'23'                             ;N
movwf Digit2
movlw d'46'                             ;Display OFF
movwf Digit1
return
OFF
movlw d'25'                             ;P
movwf Digit5
movlw d'46'                             ;Display OFF
movwf Digit4
movlw d'24'                             ;O
movwf Digit3
movlw d'15'                             ;F
movwf Digit2
movlw d'49'                             ;invert F
movwf Digit1
return

;-------------------------------------------------------------------

TempORDemo
incf CntTemp
btfsc CntTemp,7
goto Test2
call Byte2Digit                         ;Convert Byte Value to Digits
return
Test2
call Byte2Digit                         ;Convert Byte Value to Digits
btfss CntTemp,6
return
call Demo
return

;-------------------------------------------------------------------

end                                     ;The END of ASM file

;-------------------------------------------------------------------
```

1W_16F6X.inc

```
; *********************************************************
;
; Dallas 1-Wire Support for PIC16F628
;
; Processor has 4MHz clock and 1µs per instruction cycle.
;
; *********************************************************
; *********************************************************
; Dallas Semiconductor 1-Wire MACROS
; *********************************************************
OW_HIZ:MACRO
BSF STATUS,RP0                          ; Select Bank 1 of data memory
BSF TRISA, DQ                           ; Make DQ pin High Z
BCF STATUS,RP0                          ;Select Bank 0 of data memory
ENDM
; ---------------------------------------------------------
OW_LO:MACRO
BCF STATUS,RP0                          ; Select Bank 0 of data memory
BCF PORTA, DQ                           ; Clear the DQ bit
BSF STATUS,RP0                          ; Select Bank 1 of data memory
BCF TRISA, DQ                           ; Make DQ pin an output
BCF STATUS,RP0                          ; Select Bank 0 of data memory
ENDM
; ---------------------------------------------------------
WAIT:MACRO TIME
                                        ;Delay for TIME µs.
                                        ;Variable time must be in multiples of 5µs.
MOVLW (TIME/5)-1                        ;1µs
MOVWF TMP0                              ;1µs
CALL WAIT5U                             ;2µs
ENDM
; *********************************************************
; Dallas Semiconductor 1-Wire ROUTINES
; *********************************************************
WAIT5U:
                                        ;This takes 5µS to complete
NOP                                     ;1µs
NOP                                     ;1µs
DECFSZ TMP0,F                           ;1µs or 2µs
GOTO WAIT5U                             ;2µs
RETLW 0                                 ;2µs
; ---------------------------------------------------------
OW_RESET:
OW_HIZ                                  ; Start with the line high
CLRF PDBYTE                             ; Clear the PD byte
OW_LO
WAIT .500                               ; Drive Low for 500µs
OW_HIZ
WAIT .70                                ; Release line and wait 70µs for PD Pulse
BTFSS PORTA,DQ                          ; Read for a PD Pulse
INCF PDBYTE,F                           ; Set PDBYTE to 1 if get a PD Pulse
WAIT .400                               ; Wait 400µs after PD Pulse
RETLW 0
; ---------------------------------------------------------
DSRXBYTE:                               ; Byte read is stored in IOBYTE
MOVLW .8
MOVWF COUNT                             ; Set COUNT equal to 8 to count the bits
DSRXLP:
OW_LO
NOP
NOP
NOP
NOP
NOP
NOP                                     ; Bring DQ low for 6µs
OW_HIZ
NOP
NOP
NOP
NOP                                      ; Change to HiZ and Wait 4µs
MOVF PORTA,W                            ; Read DQ
```

```
ANDLW 1<<DQ                             ; Mask off the DQ bit
ADDLW .255                              ; C=1 if DQ=1: C=0 if DQ=0
RRF IOBYTE,F                            ; Shift C into IOBYTE
WAIT .50                                ; Wait 50µs to end of time slot
DECFSZ COUNT,F                          ; Decrement the bit counter
GOTO DSRXLP

RETLW 0
; ------------------------------------------------------------
DSTXBYTE:                               ; Byte to send starts in W
MOVWF IOBYTE                            ; We send it from IOBYTE
MOVLW .8
MOVWF COUNT                             ; Set COUNT equal to 8 to count the bits
DSTXLP:
OW_LO
NOP
NOP
NOP                                     ; Drive the line low for 3µs
RRF IOBYTE,F
BSF STATUS,RP0                          ; Select Bank 1 of data memory
BTFSC STATUS,C                          ; Check the LSB of IOBYTE for 1 or 0
BSF TRISA,DQ                            ; HiZ the line if LSB is 1
BCF STATUS,RP0                          ; Select Bank 0 of data memory
WAIT .60                                ; Continue driving line for 60µs
OW_HIZ                                  ; Release the line for pullup
NOP
NOP                                     ; Recovery time of 2µs
DECFSZ COUNT,F                          ; Decrement the bit counter
GOTO DSTXLP
RETLW 0
; ------------------------------------------------------------
```

Beeper.inc

```
;***** Deklaracija konstanti *****

        CONSTANT PRESCbeep = b'00000110'    ; 32,7 ms po Ciklusu

;***** Makroi *****

BEEP    macro   freq,duration
        movlw   freq
        movwf   Beep_TEMP1
        movlw   duration
        call    BEEPsub
        endm

BEEPinit macro
        bcf     BEEPport
        bsf STATUS,RP0
        bcf     BEEPtris
        bcf STATUS,RP0
        endm

;***** Podprogrami *****

BEEPsub
        movwf Beep_TEMP2                ; postavi vrednost trajanja zvuka
        clrf    TMR0                    ; inicijalizuj brojac
        bcf     BEEPport
        bsf STATUS,RP0
        bcf     BEEPport
        movlw   PRESCbeep               ; postavi preskaler za TMR0
        movwf   OPTION_REG              ; OPTION <- W

        bcf STATUS,RP0
BEEPa
        bcf     INTCON,T0IF             ; izbrisi TMR0 Overflow Fleg
BEEPb
        bsf     BEEPport
        call    B_Wait                  ; trajanje logicke "1"
        bcf     BEEPport
        call    B_Wait                  ; trajanje logicke "0"
        btfss   INTCON,T0IF             ; proveri TMR0 Overflow Fleg,
        goto    BEEPb                   ; preskoci ako je setovan
        decfsz  Beep_TEMP2,1            ; Da li je Beep_TEMP2 = 0 ?
        goto    BEEPa                   ; Ako nije skoci ponovo na BEEPa
        RETURN

B_Wait
        movfw   Beep_TEMP1
        movwf   Beep_TEMP3
B_Waita
        decfsz  Beep_TEMP3,1
        goto    B_Waita
RETURN
```