**Microcontrollers**

-Room thermostat-

Student: Bogdan Meseșan

Coordinating teacher: Andreea Ignat



Contents

[1. The temperature sensor 3](#_Toc9949414)

[2 The ADC stage 4](#_Toc9949415)

[3. LCD configuration 6](#_Toc9949416)

[4. The relay stage 11](#_Toc9949417)

[5. The microcontroller 13](#_Toc9949418)

[6. Software implementation 14](#_Toc9949419)

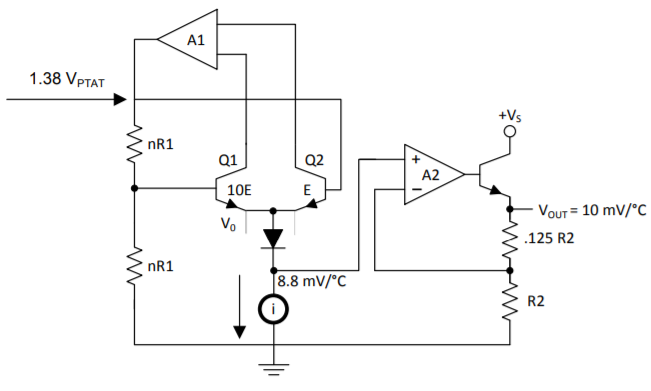
[6.1. The ASM program 15](#_Toc9949420)

[The program in C 21](#_Toc9949421)

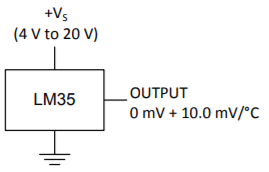
[References: 26](#_Toc9949422)

# 1. The temperature sensor

The temperature sensor I have chose is the LM35, which is a semiconductor sensor which comes in the form of an integrated circuit. It is a voltage output temperature sensor, which has two identical diodes with temperature-sensitive voltage vs. current characteristics that can be used to monitor changes in temperature. A linear response and a low impedance are offered at the output and it draw only 60μA from the supply, which limits its power dissipation. The operating range od the sensor is rated between −55°C and 150°C. The temperature-sensing element is comprised of a delta-V BE architecture. Its accuracy is of ±0.5°C at 25°C and of ±1° for a higher temperature range.



The temperature-sensing element is the buffered by a class A amplifier, then the voltage is provided to the output pin, with a 0.5Ω output impedance. The transfer function of the sensor is provided by the following equation: VOUT = 10 mv/°C × T.



In the following application, the supply voltage for the sensor will be +5Vdc, and the output will then be amplified and sent to an Analog to Digital converter, having the same +5V reference voltage. Finally, the information from the sensor will be processed by the microprocessor.

# 2 The ADC stage

For the current application, I have chosen to use the ADC 0808, produced by Texas Instruments. The ADC uses successive approximation as the conversion technique, which means that is has a 256R voltage divider and a SAR register. It has 8 channels which can be accessed via an 8-channel multiplexer.

Key Specifications:

Resolution: 8 Bits; Total Error: ±1/2 LSB; Conversion Time: 100 µ;

Because the ADC is on 8bits, then VLSB is: .

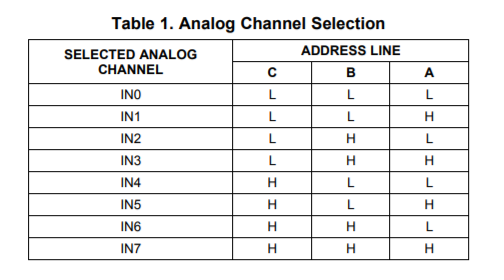
The application will measure temperatures between 0 and 50°C, which means that at 0°C I expect an output of logic 0000 0000b, and at 50°C I expect 1111 1111b. The error will be of ±1 LSB.

ADC pins:

The explanation of the pins:

-IN0, … , IN7 – represent the 8 different input channels of the ADC; only one channel is active at a given time. The selection of the channel is made via the address line decoder.

-ADD\_A, ADD\_B, ADD\_C – address lines used to select a particular channel.



-ALE – Address Latch Enable is used enable the selected channel

-Vref(+) ,Vref(-) – Reference voltage levels; for our system: Vref(+) = 5V, Vref(-) = GND;

-Clock – the typical clock frequency for this ADC is 640kHz, for minimum conversion time

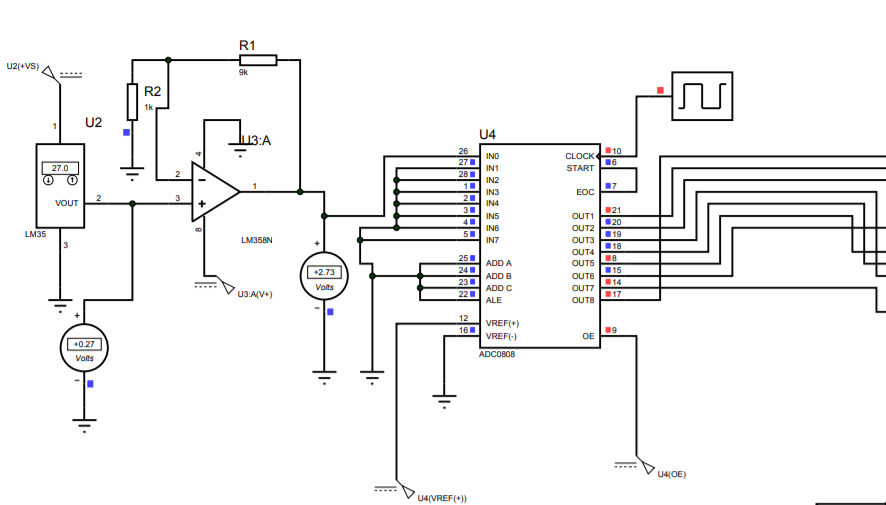
-Start, EOC – The ADC’s SAR is reset on the positive edge of the Start Conversion pulse. A conversion in progress can be interrupted by the pulse received from a new conversion. In order to have a continuous conversion, the End of Conversion pin will be connected to the Start pin.

-Out1, … , Out8 – The digital output of the ADC

-OE – Output enable, must be powered to 5V

I will use a noninverting amplifier, with amplification A = 10 to amplify the output of the analog sensor: . I have chosen the LM358N operational amplifier, with a positive supply of 5V and a negative supply of 0V.

The circuit of the ADC stage:



In the software part of the project, a lookup table will be designed in order to process the output of the ADC in a correct manner. For every temperature level, the processor will interpret the information at its input via a filtering routine.

# 3. LCD configuration

For the current application I have chosen to use the LM016L LCD, which has a 16-character x 2-line display and a 5x8 dot cursor. Pins:

VDD – Power supply pin with a typical power supply of 5V

VSS - Ground

VEE – Contrast adjustment

RS – 0: Instruction code input; 1: Data Input

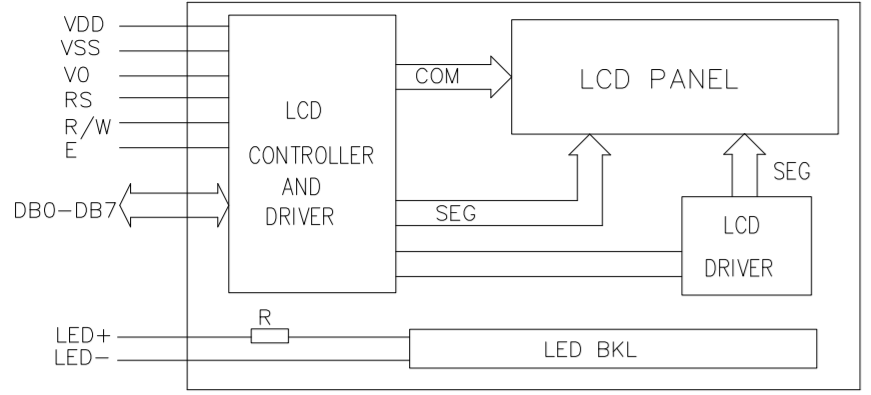
R/W – 0: Data Write; 1: Data Read

E – Operation Enable Signal

DB0 – DB3 – low order bi-directional data bus lines, not used during 4-bit operation

DB4 – DB7 – high order data bus lines, which are used during 4-bit operation

Block diagram:



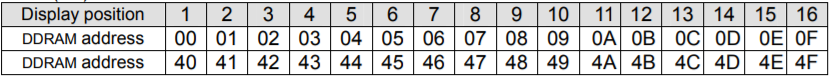
The LCD module has two types of interfaces: 4-bit interface and 8-bit interface, depending on the microprocessor have for our application.

Address counter (AC):

Stores the current address of the DDRAM/CGRAM. After writing from DDRAM/CGRAM, the AC is automatically increased by 1; when a read operation is done from the DDRAM/CGRAM the AC is decreased by 1.

Display Data RAM (DDRAM):

DDRAM stores display data of maximum 80x8 bits. The position of the next character to be written is set via DDRAM.



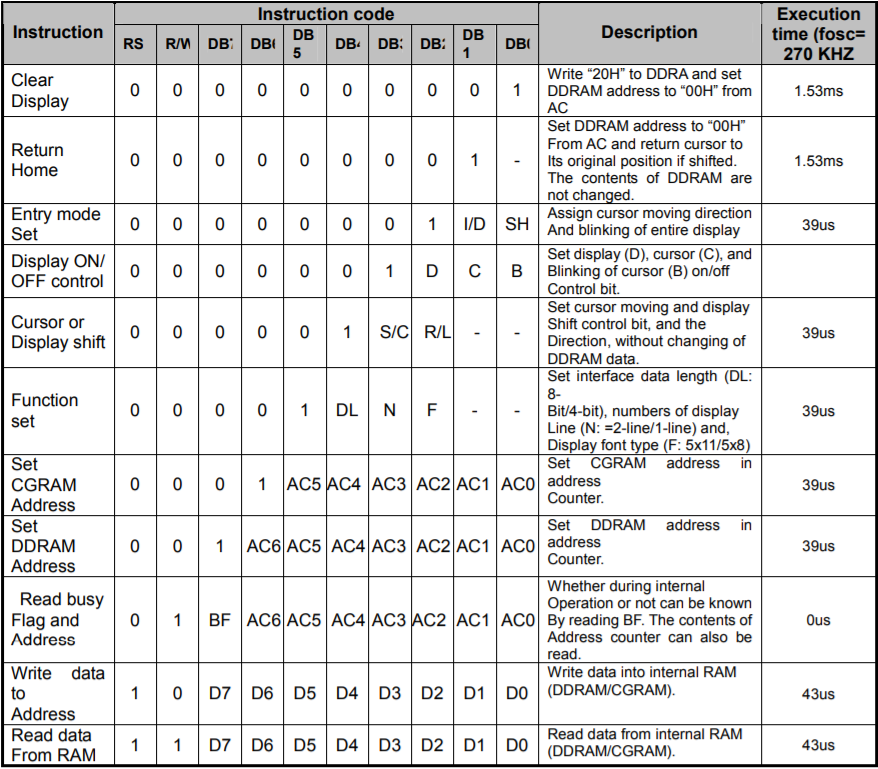
Character Generator RAM (CGRAM):

By writing certain data to the CGRAM, user defined characters can be used.

Character Generator ROM (CGROM):

CGROM has 204 characters with 5 x 8 dot patterns and 32 characters with 5 x 10 dot patterns.

Instructions:



Content:

**Clear Display**-writes 20h to al DDRAM addresses and resets the AC to 00h

**Return home** – resets the cursor to its initial position and sets the DDRAM address to 00h

**Entry mode set** – sets the moving directions of the cursor and display; I/D = 1 => DDRAM address incremented; I/D = 0 => DDRAM address decremented; SH = 1 => shift of entire display;

**Display On/Off** – D -> 0: display turned off, but data remains in DDRAM; 1: display turned on;

C-> 0: cursor turned off, but I/D preserves data; 1: cursor turned on; B-> 0: cursor blink off; 1: cursor blink on

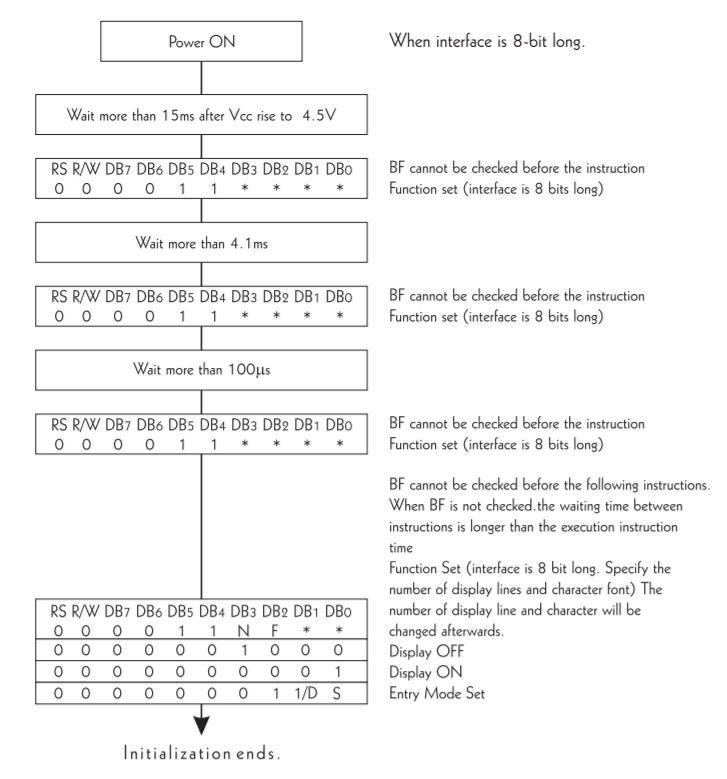
**Function set** – DL -> 0: 4-bit interface; 1: 8-bit interface; N-> 0: use only 1 line; 1: use line 1 and line 2; F-> 0: 5x8 dots display format; 1: 5x11 display format

Standard character pattern:



8-bit interface:

Initialization:



The sequence I use to initialize the Display:

-instruction1: function set

-instruction2: display off

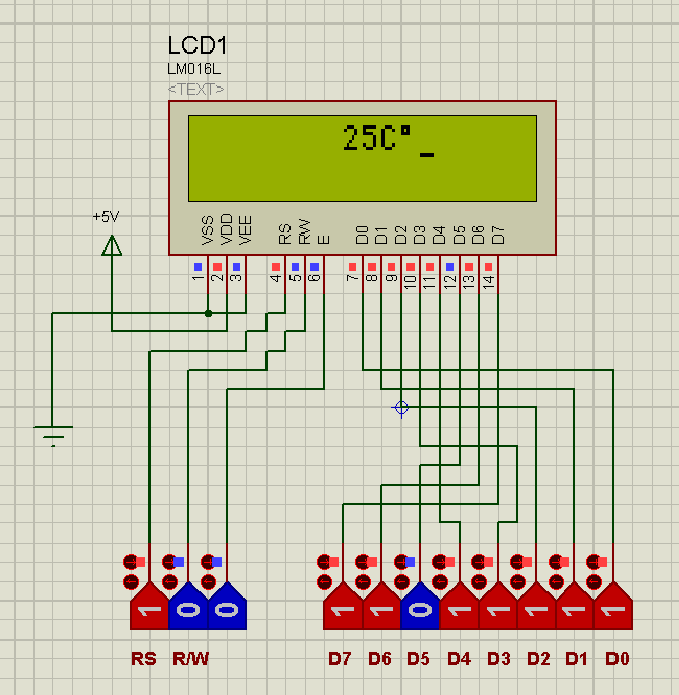
-instruction3: display on

-instruction4: display on/off control

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| RS | R/W | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

Display:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| RS | R/W | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |



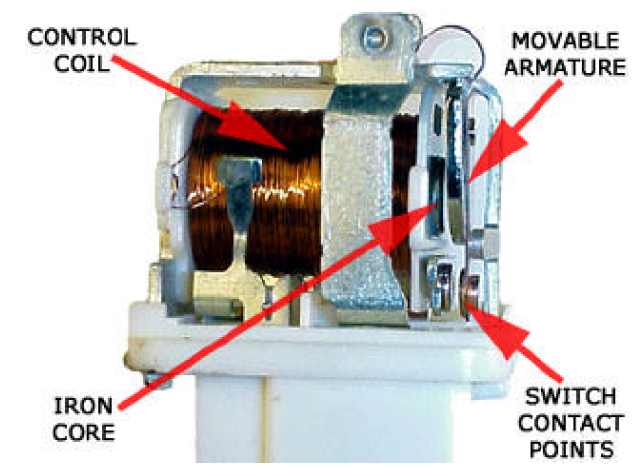
# 4. The relay stage

The operating principle of a relay

A relay is an electromagnetic switch that is used to turn on and off via a low power signal. It is operated both electrically and mechanically because is uses an electromagnet and a set of contacts is the switching mechanism. One common application of the relay is in systems where one signal is used to control multiple circuits.

A typical relay is constituted of:

* Electromagnet – an electric coil, which acts as the sensing unit and can be powered either by AC or DC current
* Movable Armature - operates either to close the open contacts or to open the closed contacts
* Switch point contacts - connected to the load and controlled by the coil in the control circuit
* Spring - used so as to produce an air gap in the circuit when the relay becomes de-energized

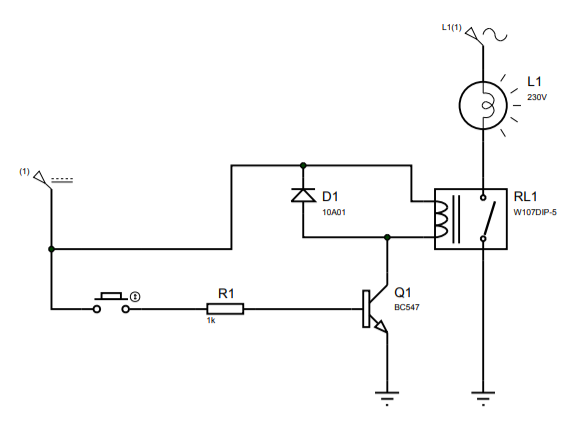


How does a relay work? Current starts flowing through the coil until it exceeds a certain threshold value. The electromagnet starts energizing and intensifies the magnetic field. The upper contact arm starts to be attracted to the lower part until they form a short circuit between the power and the load. When the current is shut off, the movable armature is returned by a force to its initial position. This force is provided by two factors: the spring and gravity.

A relay can be used either in low-voltage applications, where they reduce the noise of the whole circuit, or in high-voltage applications, where they provide protection. In our temperature control system, we will use the relay in On/Off operation.

The command circuit:

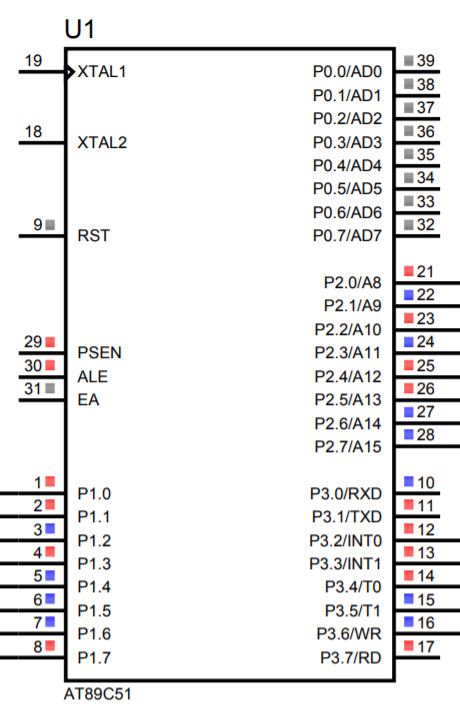
The main problem with relay coils is that they have highly inductive loads. When current flows, a self-induced magnetic field is generated around the coil. When the current goes off, a large electromotive force voltage is produced as the magnetic flux collapses within the coil. This voltage may very high in comparison to the switching voltage and may damage any semiconductor device, like a transistor.



To prevent this effect, connect a reverse biased diode across the relay coil. The EMF voltage will turn the diode on, which in turn, will dissipate the energy stored in the relay. This diode is called Fly-Back diode. When the relay is on, the bipolar transistor will start to conduct, effectively linking the relay to ground. The 1k resistor in the base of the transistor is used to limit the base current. The relay will be commanded from the microcontroller, when the measured temperature is lower than the reference temperature.

# 5. The microcontroller

The component that I have chosen for my application is the AT89C51, which does not have an internal ADC. Therefore, I will use the ADC0808 as an external ADC.



The microcontroller has 4 ports:

-Port 0 -> Open-drain, 8-bit bi-directional I/O port

-Port 1 -> 8-bit bi-directional I/O port with internal pull-ups; I will use this port to read data sent from the ADC

-Port 2 ->8-bit bi-directional I/O port with internal pull-ups;

I will use P2.5, P2.6 and P2.7 to control the RS, R/W and EN pins of the LCD

-Port 3-> 8-bit bi-directional I/O port with internal pull-ups; I will use Port 3 to send instructions to the LCD controller

Utility of the Ports:

-Port 1: Input port, connected to the output of the ADC, tasked with receiving temperature information in digital form.

-Port 2: Output port, connected to the Data lines of the LCD, tasked with sending instructions to the Display

-Port 3: Output port:

P3.0-Relay control;

P3.2 – “Button+” external interrupt 0;

P3.3 - “Button-” external interrupt 1;

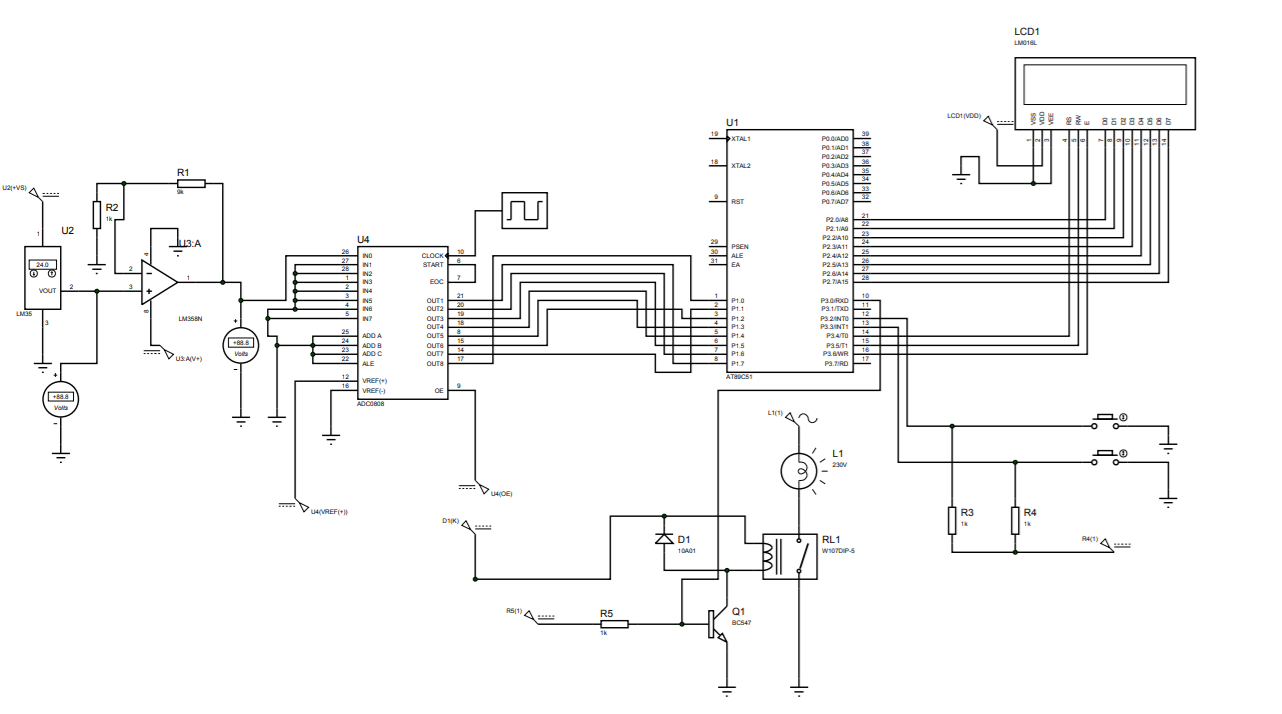
P3.4 – LCD RS;

P3.5 – LCD R/W;

P3.6 – LCD Enable;

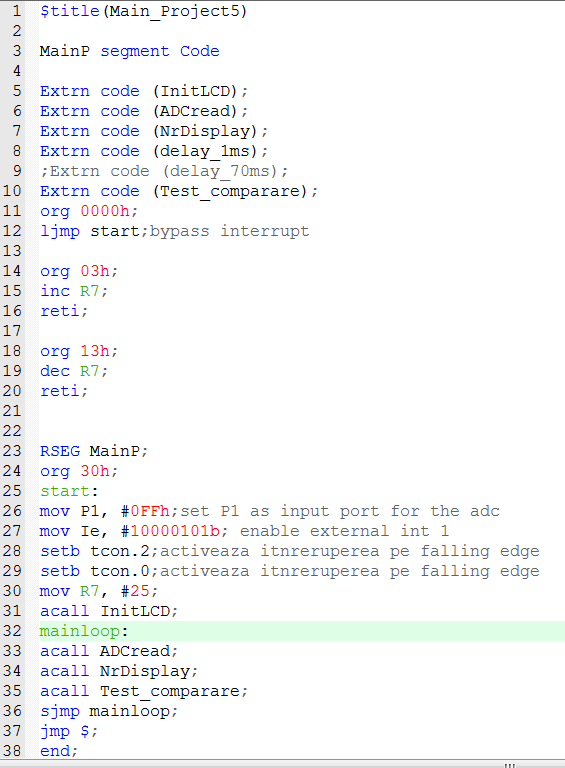
# 6. Software implementation

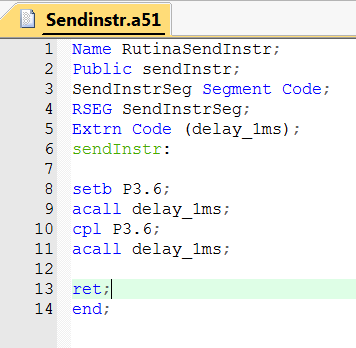
The final circuit:



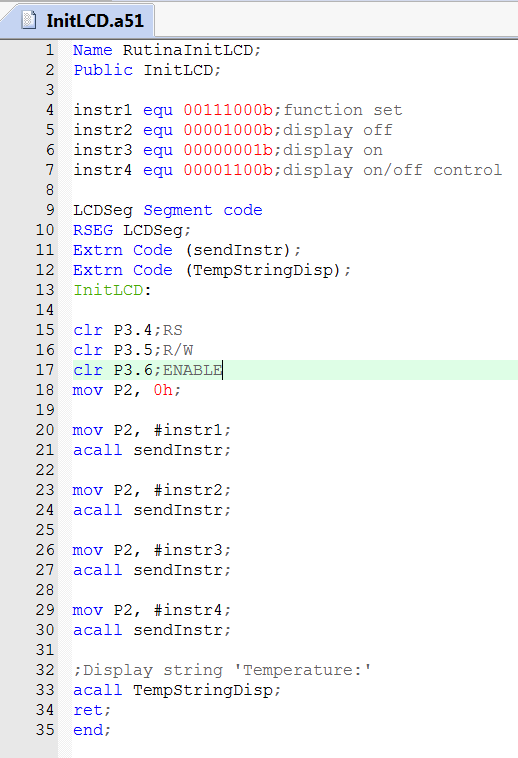
## 6.1. The ASM program

Main:

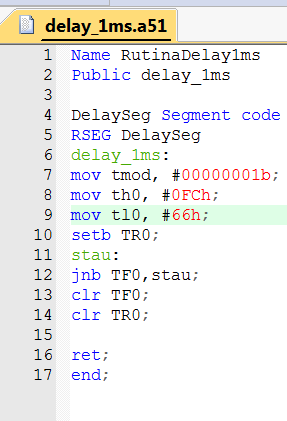


Send instruction:

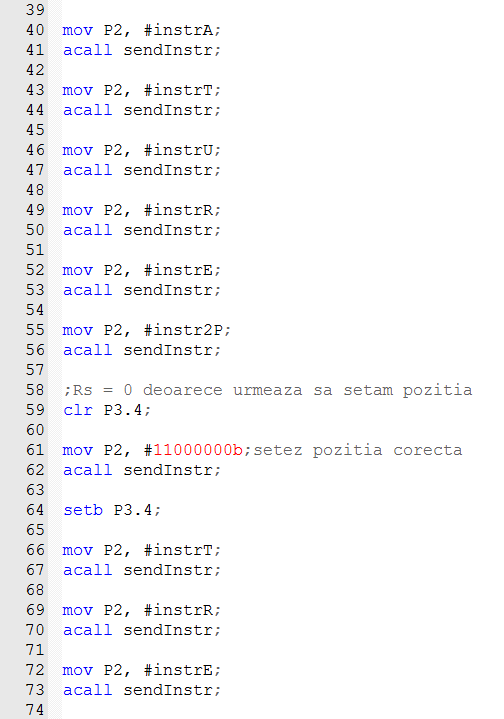
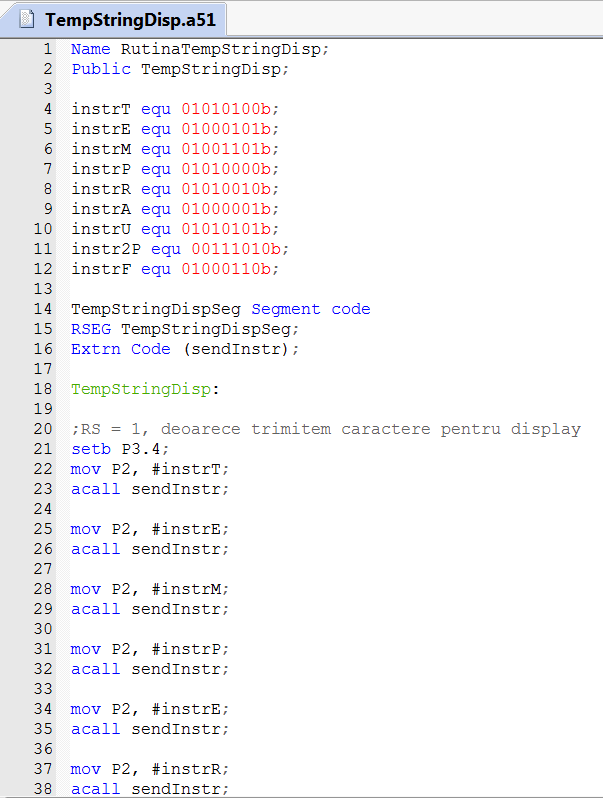
Init Lcd:

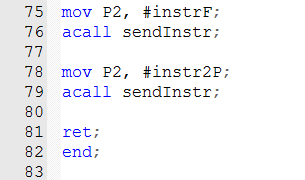


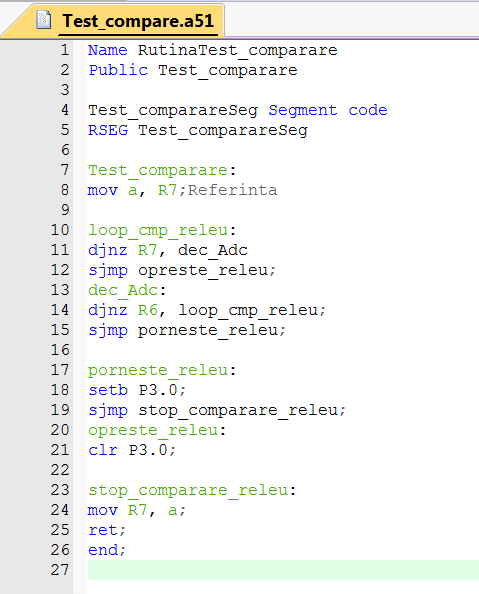
Delay\_1ms:

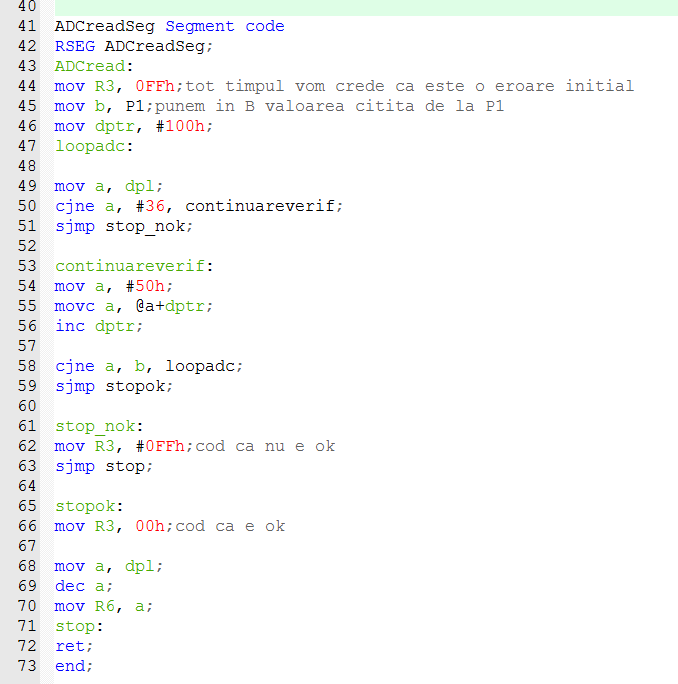
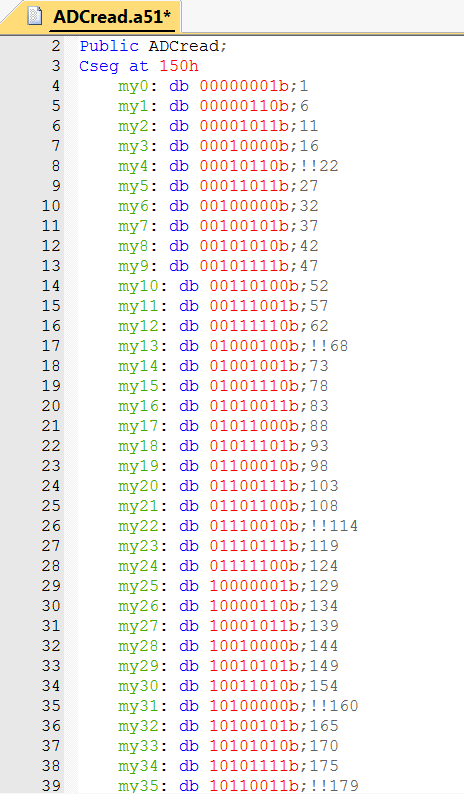


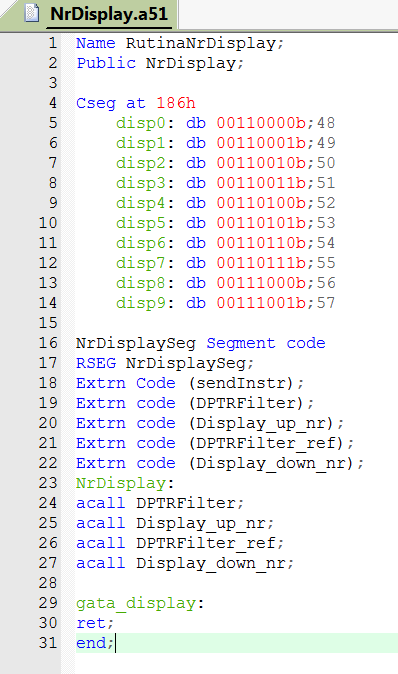
Temperature\_string display:



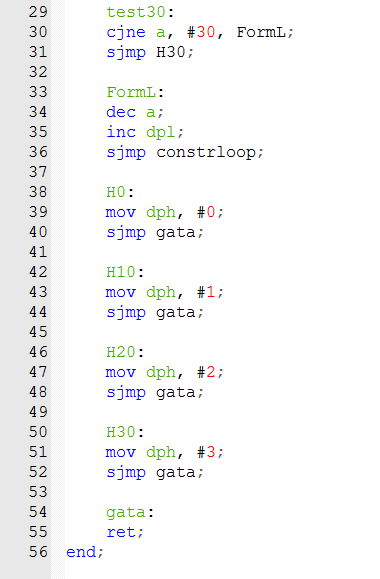
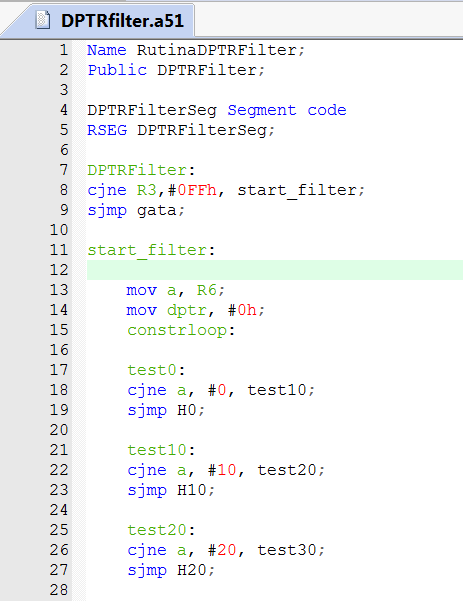


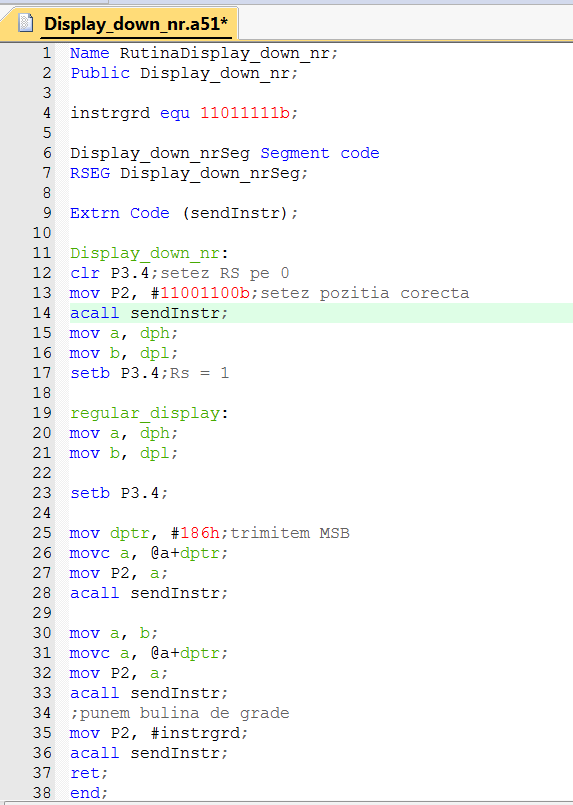
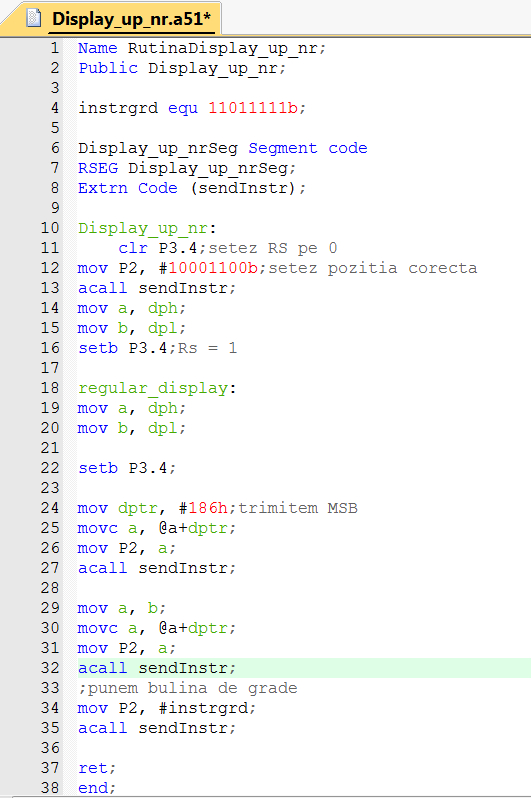






I consider in R6-the value which was read from the ADC and in R7-the reference temperature value.

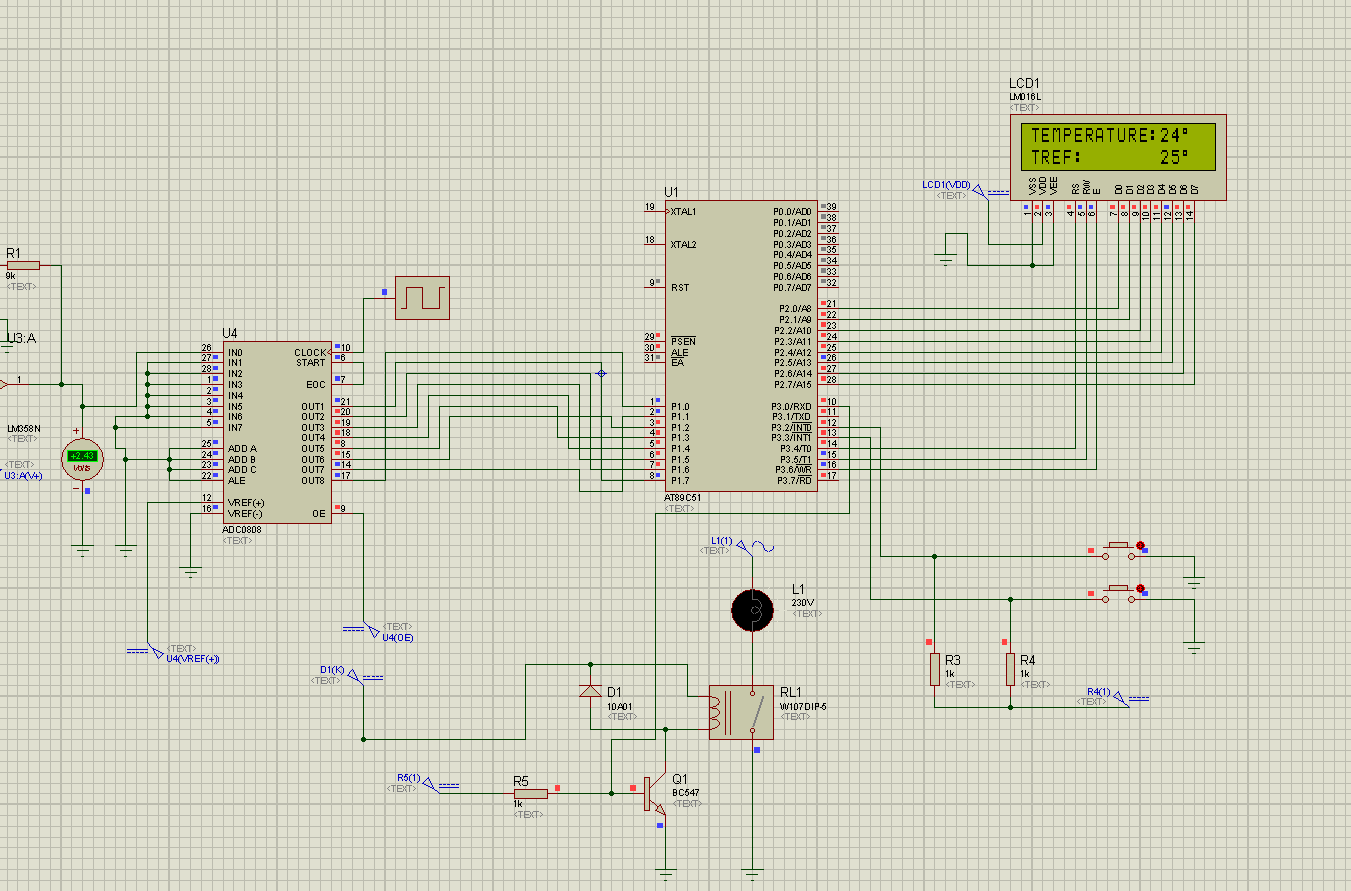




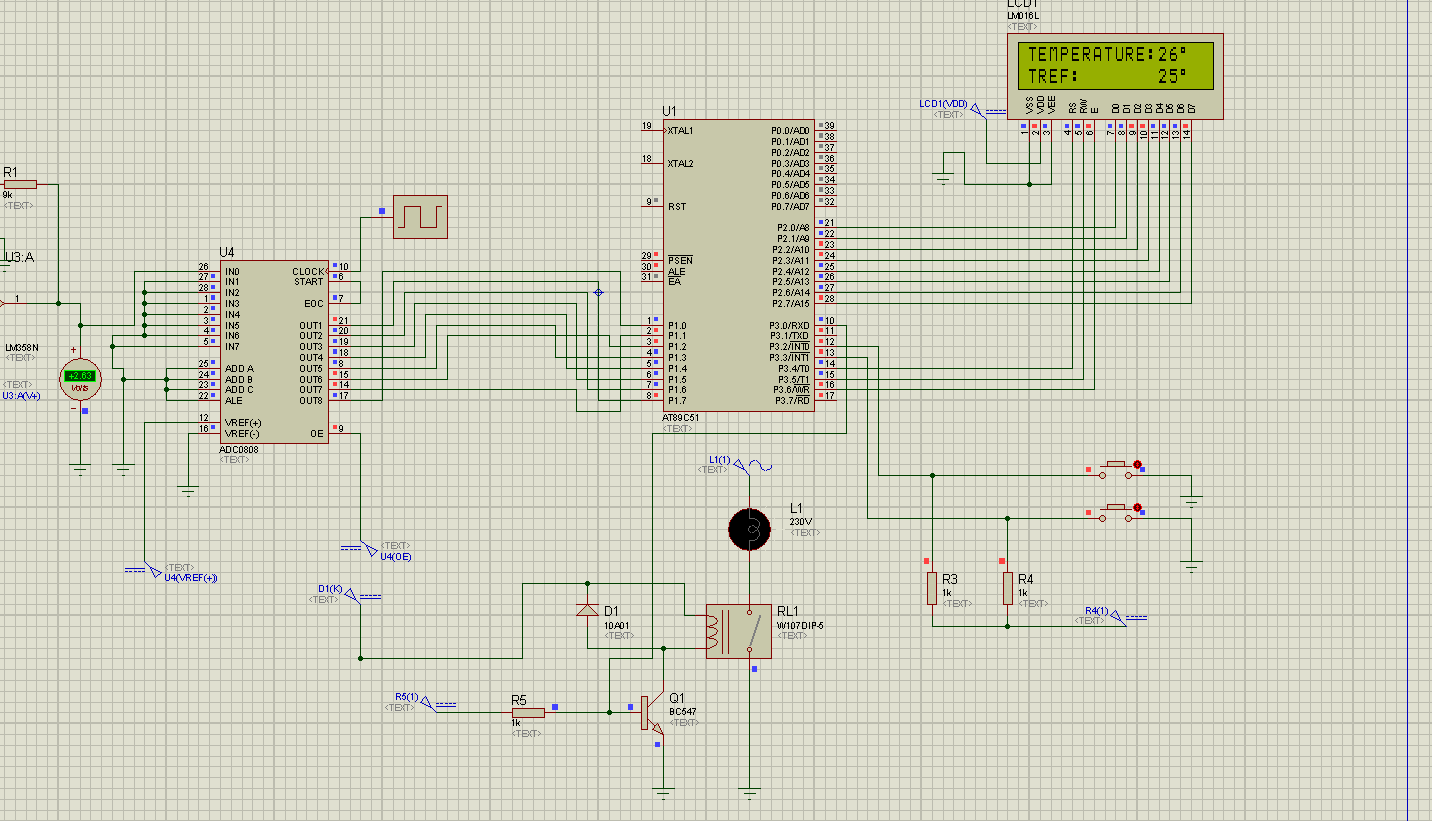
The second DPTR filter is similar to the first one, but it filters the value in R7-Vref

Simulation of the ASM program:

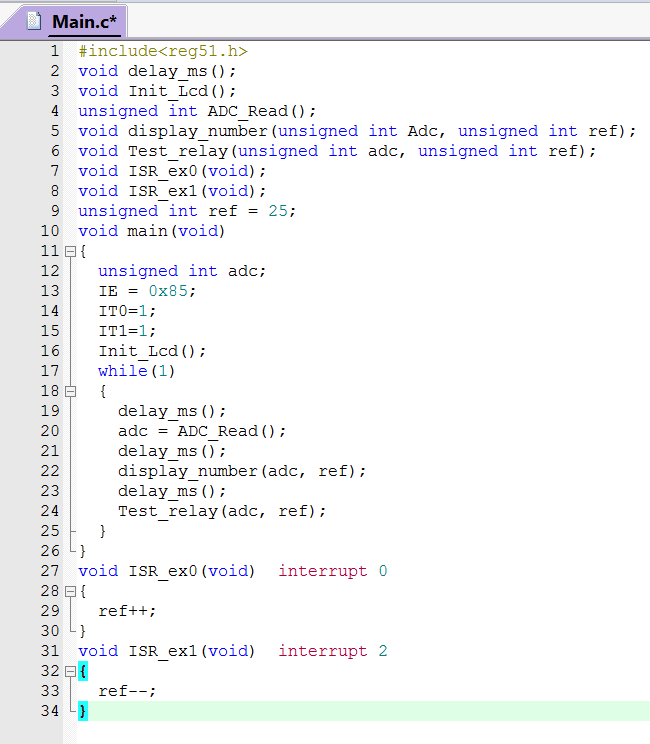
In this case, the reference temperature is higher than the reading from the ADC, therefore P3.0 will be set HIGH and the relay will be turned on.

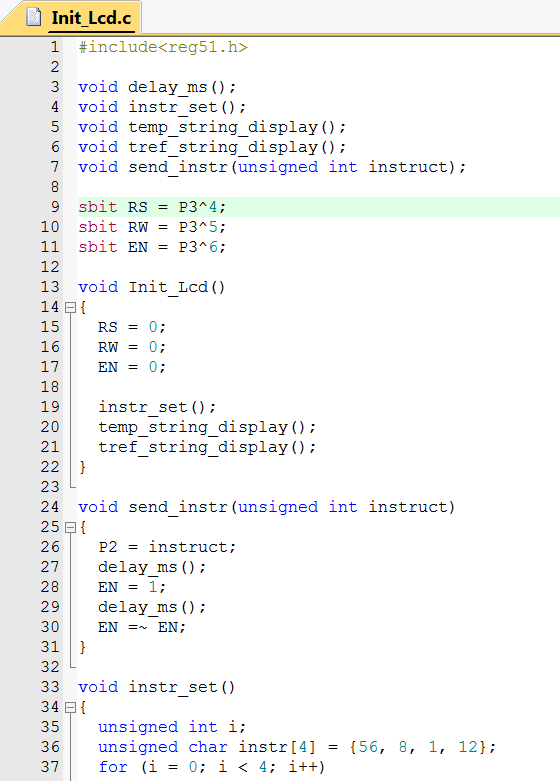


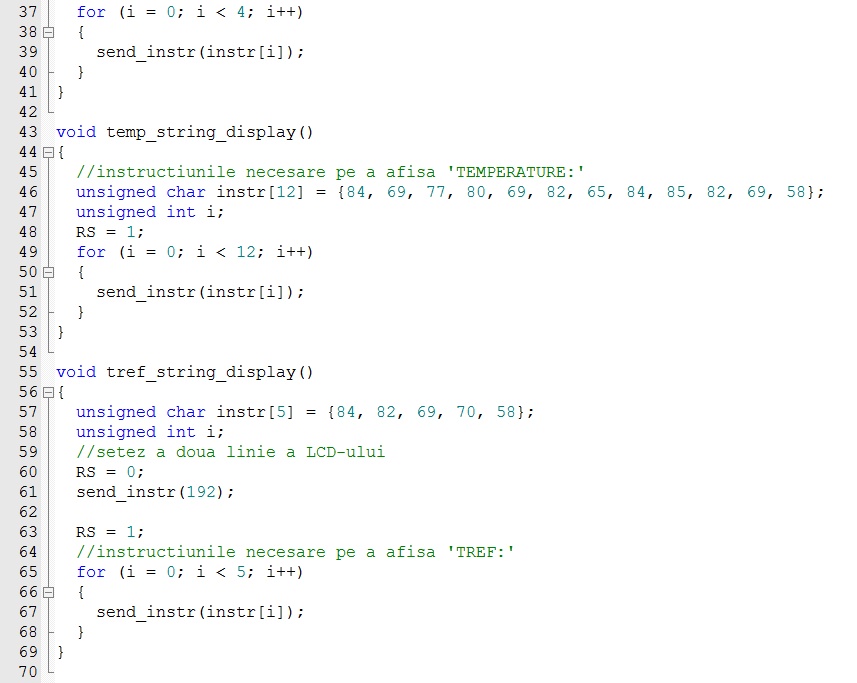
The ‘-‘ button was pressed, the reference temperature decreased, being equal to the reading temperature, thus switching the relay off.

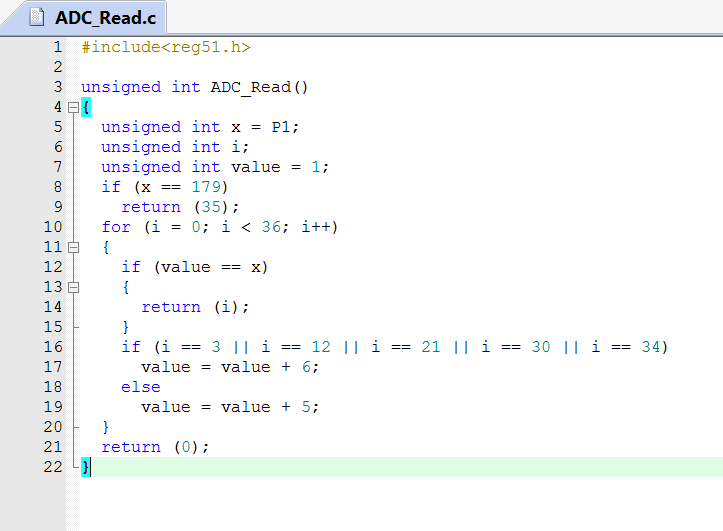


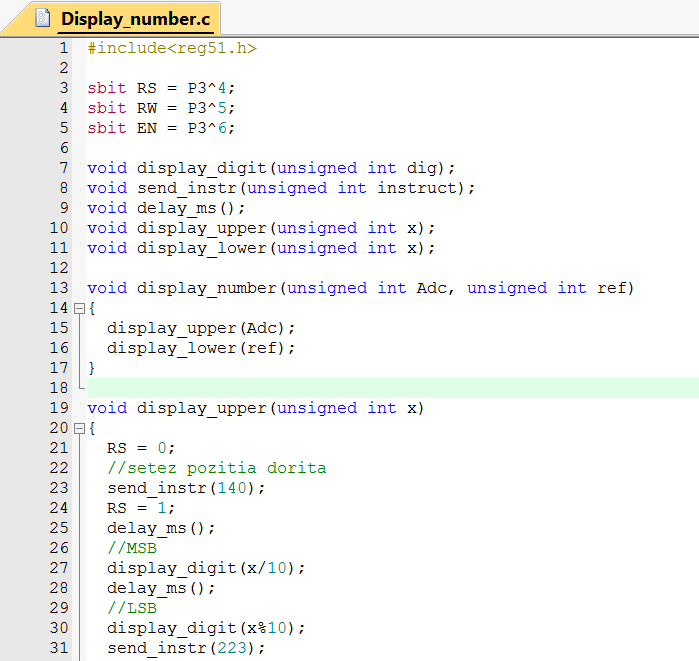
## The program in C

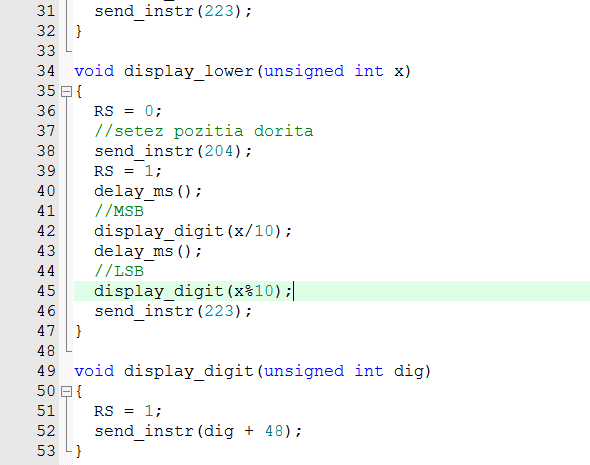


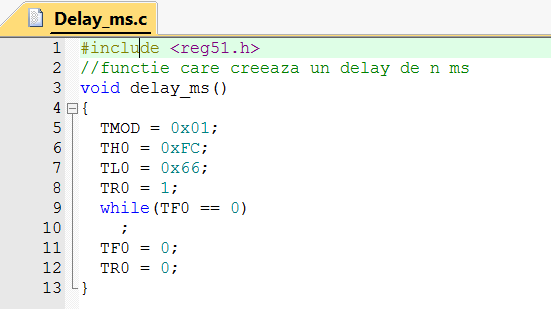


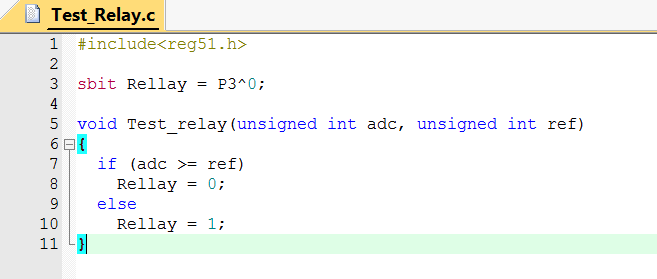






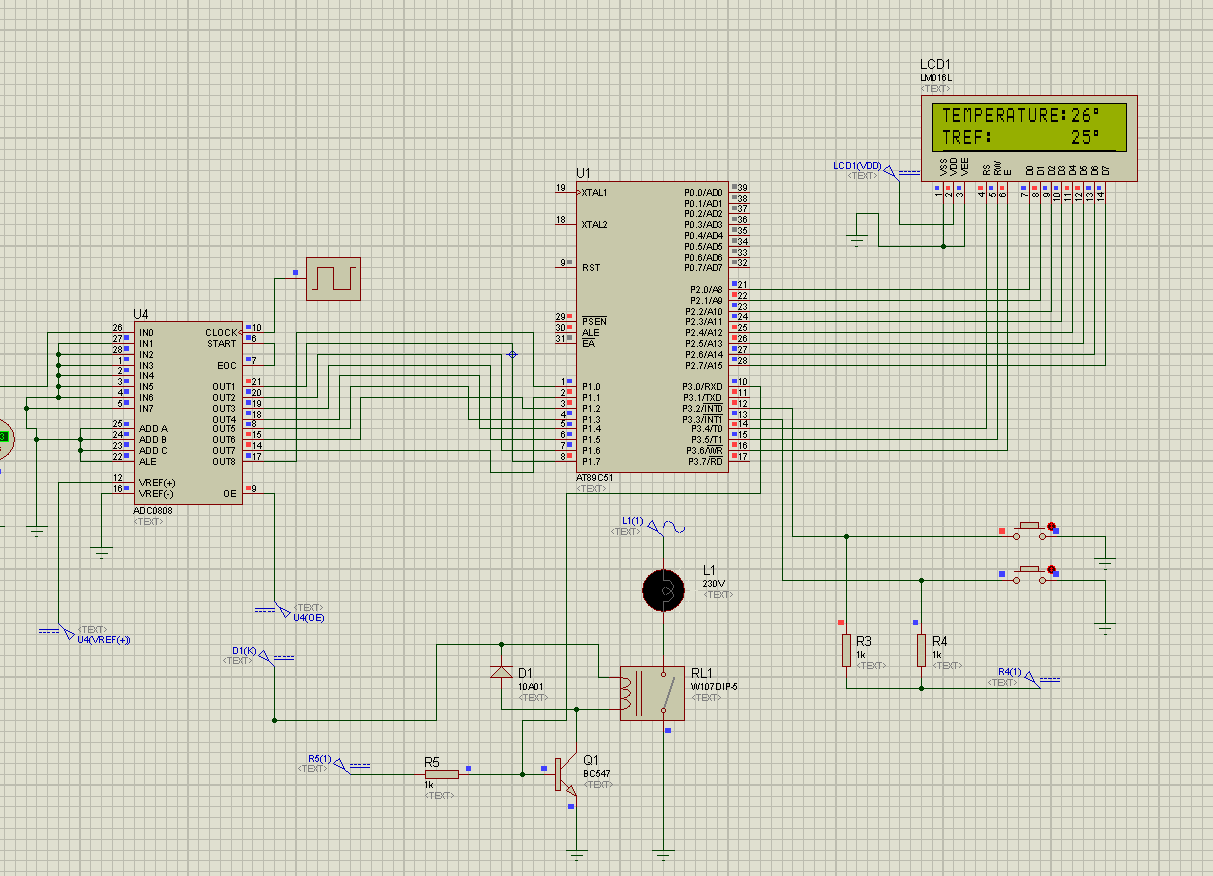




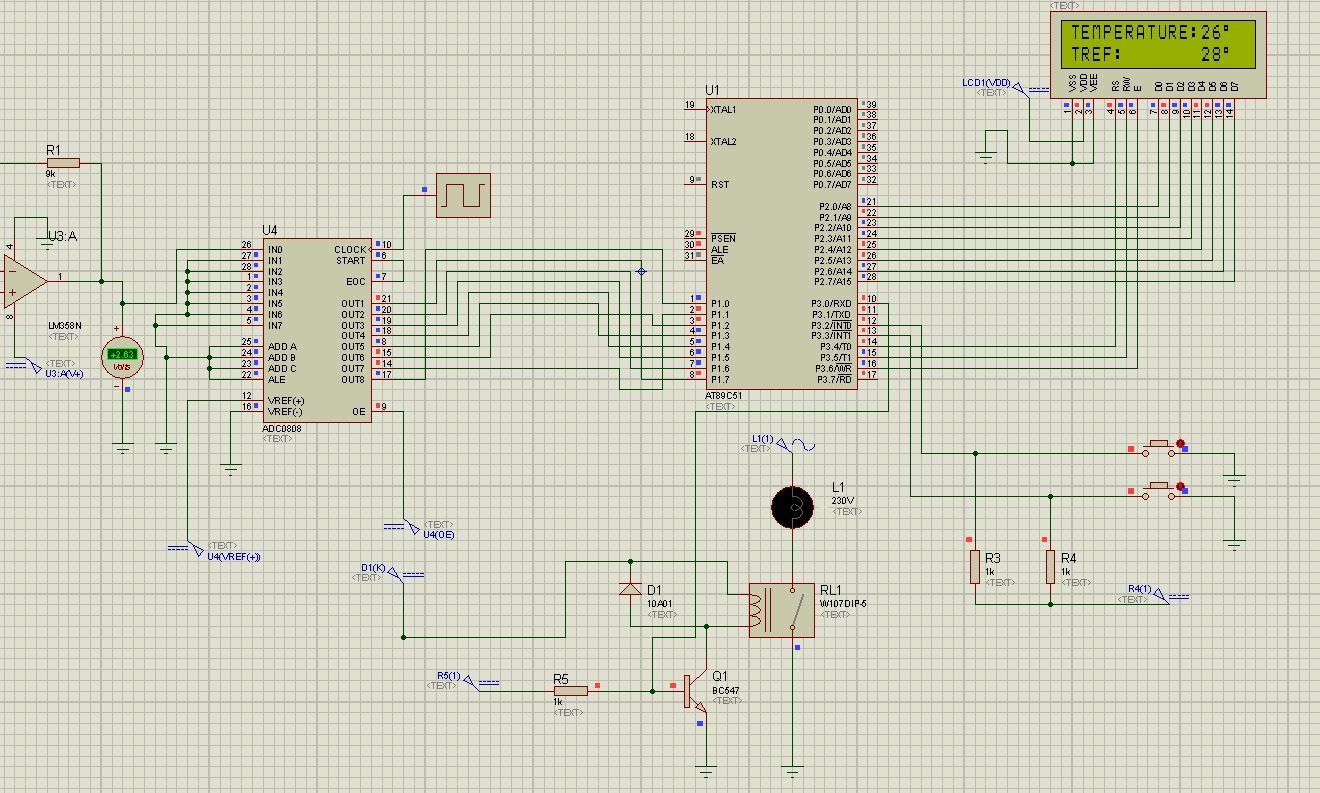


Simulation of the C program:

The read temp is higher than the reference=>relay off:



The relay turns on in this case:



# References:

<http://www.ti.com/lit/ds/symlink/lm35.pdf>

<http://www.ti.com/lit/ds/symlink/adc0804-n.pdf>

<http://www.circuitstoday.com/op-amp-circuits-in-proteus>

<http://www.circuitstoday.com/proteus-tutorial-switches-and-relays>

<http://www.ti.com/lit/ds/symlink/adc0808-n.pdf>

<https://www.youtube.com/watch?v=iNvtgEB7q6U>

<https://www.youtube.com/watch?v=TZuXfapExic&fbclid=IwAR0bdc5HNcUX-dmrg3ojPR8QmZzLVZXNkvM8RzKCjaBRwToiRhBWWAiBBfI>

<http://www.picaxe.com/docs/led008.pdf>

<https://www.sparkfun.com/datasheets/LCD/ADM1602K-NSW-FBS-3.3v.pdf>

<https://www.techshopbd.com/uploads/product_document/lm016l%20lcd.pdf>

<http://www.keil.com/dd/docs/datashts/philips/8xc752_ds.pdf>

<https://circuitglobe.com/relay.html>

<https://www.electroschematics.com/8648/run-stop-relay-circuit/>

<https://www.pc-control.co.uk/relays.htm>

<http://www.circuitstoday.com/working-of-relays>

<https://www.explainthatstuff.com/howrelayswork.html>

<https://microcontrollerslab.com/interrupts-8051-microcontroller/>

<http://www.keil.com/support/man/docs/armcc/armcc_chr1359124246903.htm>