UNIVERSITY OF TEHRAN

Instrumentation

Homework 4

Spring 2025

Amir Shahang

810101448

Q1
CAN:

Applications: Initially, CAN protocol was designed to target the communication issue that occurs within the vehicles. But later on, due to the features it offers, it is used in various other fields. The following are the applications of CAN protocol:

- **Automotive Systems**: Widely used in vehicles for communication between Electronic Control Units (ECUs), such as engine control, airbags, ABS, and infotainment systems.
- **Industrial Automation**: Supports real-time control in machinery, robotics, and manufacturing systems.
- **Medical Equipment**: Used in devices like imaging machines and diagnostic tools for reliable internal communication.
- **Railway and Aerospace**: Provides robust and noise-resistant communication in high-vibration and EMI-prone environments.
- **Building Automation**: Employed in HVAC, lighting, and access control systems.

Speed:

- **Standard CAN (Classical CAN)**:
  - Speeds up to **1 Mbps** over short distances.
  - The maximum bus length **decreases** as speed increases:
    - ~40 meters at 1 Mbps
    - ~1000 meters at 50 kbps
- **CAN FD (Flexible Data-rate)**:
  - Supports **up to 5 Mbps or more** for data phase (with the arbitration phase still at 1 Mbps).

Data frame: The CAN data frame is composed of seven fields: Start of frame (SOF) arbitration, control, data, cyclical redundancy check (CRC), acknowledge (ACK) and end of frame (EOF). CAN message bits are referred to as “dominant” (0) or “recessive” (1). The SOF field consists of one dominant bit.

Required Hardware & Circuits: Requires a CAN controller and transceiver on each node, along with a twisted-pair cable for communication. Terminating resistors are used at both ends of the bus to minimize reflections. Each of these communication protocols has its strengths and weaknesses, making them suitable for different applications based on factors like speed, reliability, and complexity of the communication network.

Accuracy and verification methods: Known for its robustness and error handling capabilities, offering high accuracy and precisioneven in noisy environments or over long distances. Integrates built-in error detection and correction mechanisms such as CRC, ACK, and frame retransmission, ensuring high accuracy and fault tolerance.

ModBus:

Applications:

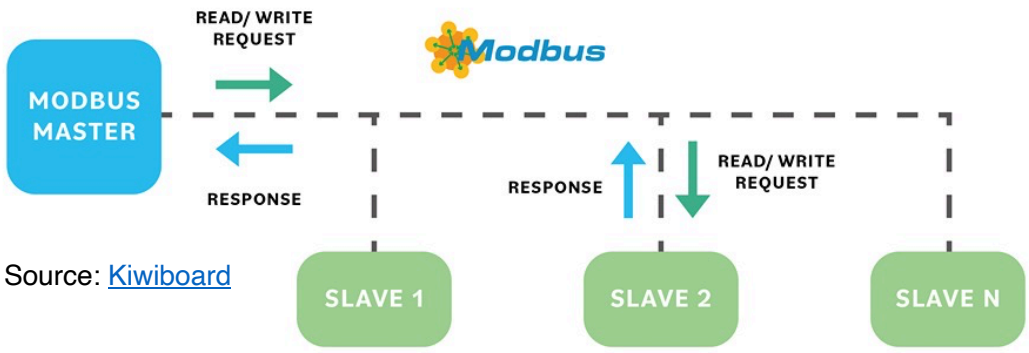
- **Industrial Automation**: Modbus is widely used for communication between PLCs, sensors, actuators, and other industrial devices.
- **SCADA Systems**: Facilitates data acquisition and supervisory control in distributed environments.
- **Building Management Systems (BMS)**: Integrates HVAC, lighting, and other building infrastructure systems.
- **Energy Monitoring**: Used in smart meters, power distribution systems, and energy monitoring devices.
- **Remote Monitoring**: Enables communication between control centers and remote field instruments.

Speed: Typically from **300 bps to 115200 bps**, with common speeds like 9600, 19200, or 38400 bps.

Data frame: A Modbus data frame is a message transmitted over Modbus network. There are Request and Response frames. A request is a message from the master to a slave. A response is a message from the slave back to the master. Modbus messages are framed (separated) by idle (silent) periods. Every 1 byte (8 bit) data/message of Modbus RTU needs to have 3 more bits to form the total 11 bits: 1 start bit. 8 bit data/message, LSB bit is first sent.

Required Hardware & Circuits: Implementation can vary but usually involves a master device (e.g., PLC) and slave devices (e.g., sensors, actuators) connected via serial communication interfaces (RS-232, RS-485). Additional hardware such as transceivers and termination resistors may be needed for RS-485 communication.

Accuracy and verification methods: Utilizes CRC or checksums for error detection, ensuring data integrity and reliability.



Source: Kiwiboard

Ethernet :

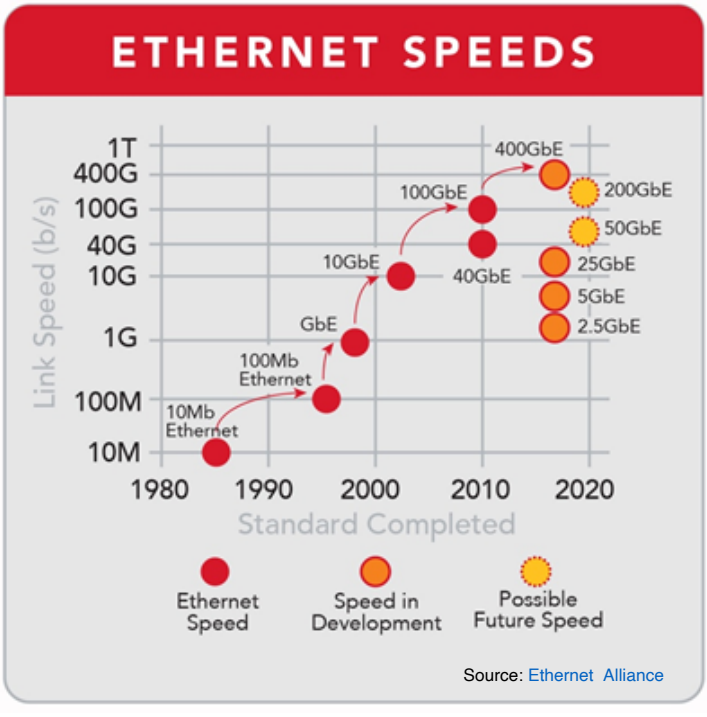
Applications: Ethernet is a widely adopted networking technology used in various environments, including:

- **Industrial Automation**: Facilitates real-time communication between programmable logic controllers (PLCs), sensors, and actuators.
- **Office and Home Networking**: Provides reliable and high-speed connections for computers, printers, and other devices.
- **Data Centers**: Supports high-throughput connections between servers and storage systems.
- **Telecommunications**: Forms the backbone for many internet service providers and cellular networks

Speed: Ethernet supports a range of data transfer rates, evolving over time to meet increasing bandwidth demands:

- **10 Mbps**: Original Ethernet standard.
- **100 Mbps (Fast Ethernet)**: Introduced to support faster data transmission.
- **1 Gbps (Gigabit Ethernet)**: Common in modern networks, balancing speed and cost.
- **10 Gbps and above**: Used in data centers and high-performance computing environments.

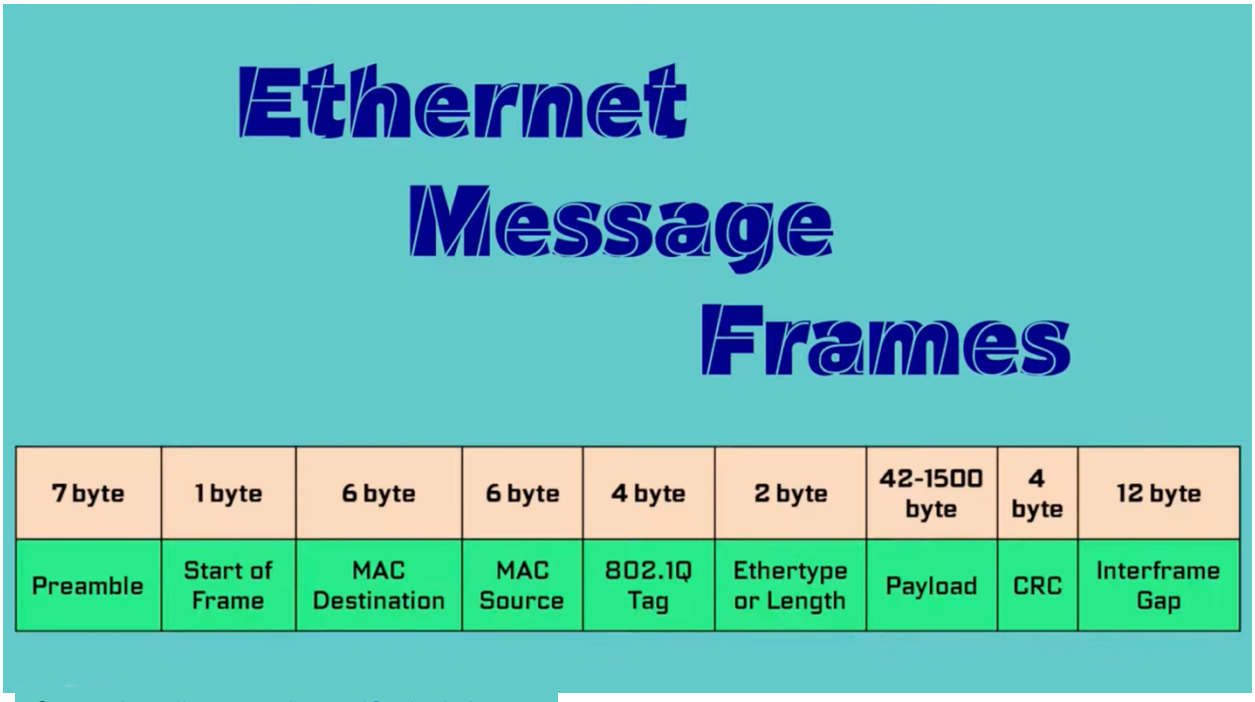
The choice of Ethernet cable (e.g., Cat5e, Cat6, Cat6a) influences the achievable speed and distance.



Data frame: An Ethernet frame is the basic unit of data transmission, consisting of several fields:

- **Preamble (7 bytes)**: Used to synchronize communication between devices.
- **Start Frame Delimiter (1 byte)**: Indicates the beginning of the frame.
- **Destination MAC Address (6 bytes)**: Specifies the recipient's hardware address.
- **Source MAC Address (6 bytes)**: Specifies the sender's hardware address.
- **Type/Length (2 bytes)**: Indicates the protocol type or payload length.
- **Payload (46–1500 bytes)**: Contains the actual data being transmitted.
- **Frame Check Sequence (4 bytes)**: Used for error checking via CRC.

The total frame size ranges from 64 to 1518 bytes, excluding the preamble and start frame delimiter.



Source : https://www.youtube.com/@eclecticclasses

Required Hardware & Circuits: Implementing Ethernet connectivity typically involves:

- **Network Interface Cards (NICs)**: Hardware components that connect a computer to a network.
- **Ethernet Cables**: Twisted pair cables (e.g., Cat5e, Cat6) used for physical connections.
- **Switches and Routers**: Devices that manage and direct network traffic.
- **RJ45 Connectors**: Standard connectors for Ethernet cables.
- **PHY Chips**: Physical layer transceivers that handle the analog and digital signal processing.
- **Magnetics (Transformers)**: Provide signal isolation and noise reduction.

Proper circuit design and PCB layout are crucial to ensure signal integrity and compliance with Ethernet standards.

Accuracy and verification methods: Ethernet employs several mechanisms to ensure data integrity:

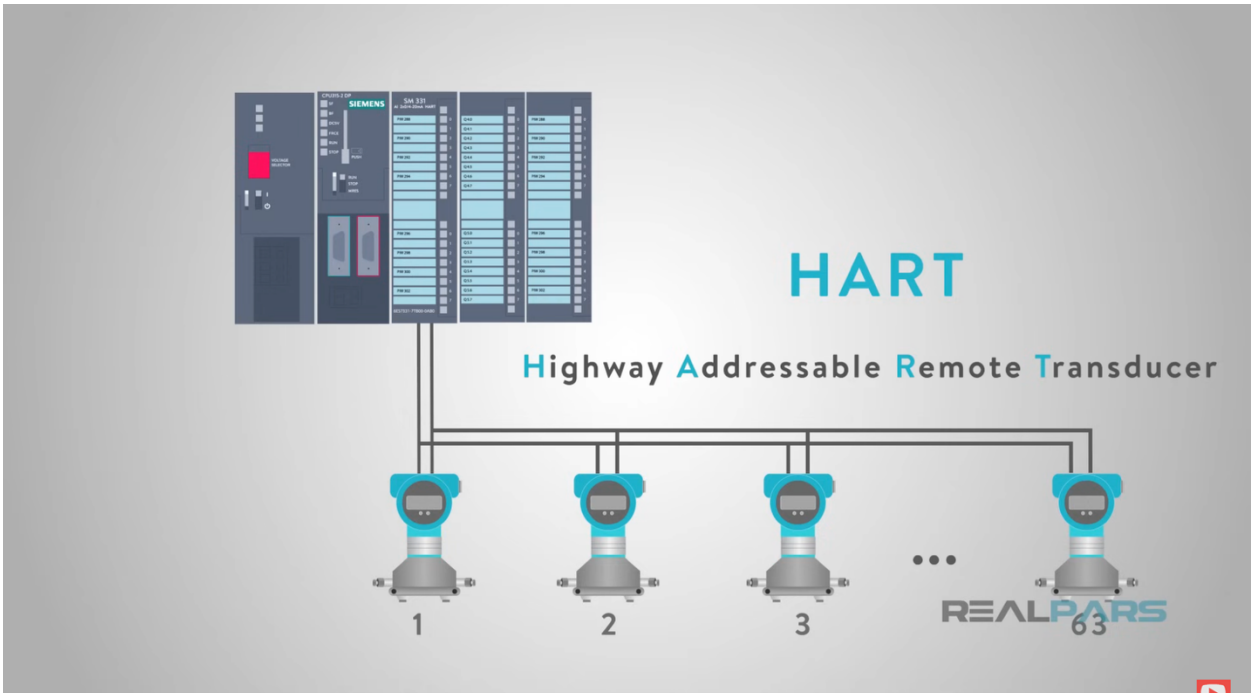
- **Cyclic Redundancy Check (CRC)**: A 32-bit checksum used to detect errors in the transmitted frames.
- **Error Detection**: Frames with mismatched CRC values are discarded to prevent corrupted data from propagating.
- **Collision Detection**: In half-duplex networks, Ethernet uses Carrier Sense Multiple Access with Collision Detection (CSMA/CD) to manage data collisions.

These methods collectively maintain the reliability and accuracy of data transmission over Ethernet networks.

Hart :

Applications: HART is extensively used in process industries for

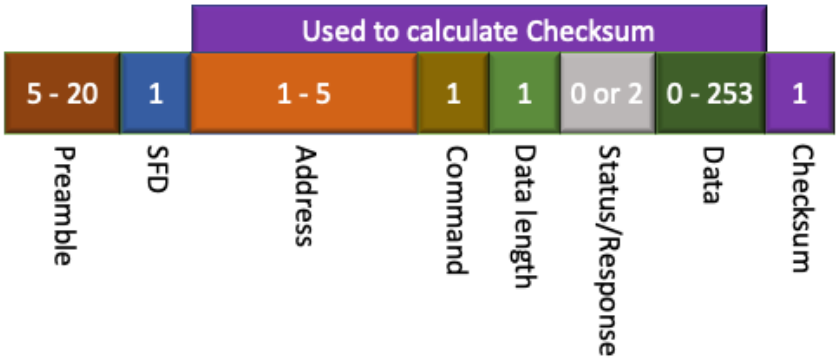
- **Instrumentation Communication**: Facilitating communication between smart field devices (like transmitters and actuators) and control systems (such as PLCs, DCSs, or SCADA systems
- **Device Diagnostics**: Providing access to device diagnostics and health information, enabling predictive maintenance strategies.
- **Configuration and Calibration**: Allowing remote configuration, calibration, and monitoring of field devices without disrupting the analog signal.
- **Hybrid Communication**: Combining analog (4–20 mA) and digital communication over the same wiring, preserving existing infrastructure while adding digital capabilities.



Speed: HART employs Frequency Shift Keying (FSK) modulation at a baud rate of **1200 bits per second**. This allows for approximately 2–3 data updates per second in request-response mode and up to 3–4 updates per second in burst mode.

Data frame: A typical HART data frame consists of the following fields:

- **Preamble (5–20 bytes)**: Synchronization bytes (0xFF) to establish communication.
- **Start Delimiter (1 byte)**: Indicates the start of the frame and contains information about address type and frame type.
- **Address (1 or 5 bytes)**: Specifies the target device's address.
- **Expansion (0–3 bytes)**: Reserved for future enhancements.
- **Command (1 byte)**: Indicates the specific command or request.
- **Byte Count (1 byte)**: Specifies the number of bytes in the data field.
- **Status (2 bytes)**: Provides information about the device's status and any errors.
- **Data (0–253 bytes)**: Contains the actual data being transmitted.
- **Checksum (1 byte)**: Used for error detection by performing an XOR of all bytes from the start delimiter to the last data byte.



Required Hardware & Circuits: Implementing HART communication typically involves:

- **HART-Enabled Field Devices**: Such as smart transmitters and actuators capable of digital communication.
- **HART Modem**: Interfaces that modulate and demodulate the FSK signals for communication between devices and host systems.
- **250 Ω Resistor**: Inserted in the loop for proper voltage development, essential for communication.
- **Host Systems**: Control systems like PLCs, DCSs, or PCs equipped with appropriate software to communicate with HART devices.
- **Handheld Communicators**: Portable devices used for on-site configuration and diagnostics of HART instruments.

Accuracy and verification methods: HART ensures data integrity and device accuracy through:

- **Checksum Verification**: Each message includes a checksum for error detection, ensuring reliable communication.
- **Device Calibration**: Regular calibration procedures are performed to maintain measurement accuracy, often facilitated by HART's digital capabilities.
- **Diagnostic Information**: HART devices provide diagnostic data, allowing for early detection of issues and reducing downtime.
- **Compliance with Standards**: Adherence to industry standards and protocols ensures consistent and accurate performance across devices.

Q2

1-

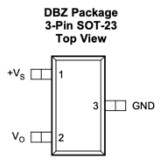
Operating temperature range: -40 °C to +150 °C

Output Voltage Range: 100 mV at –40°C to 1.75 V at +125°C

Precision: At 25°C: ±2°C typical, ±3°C maximum

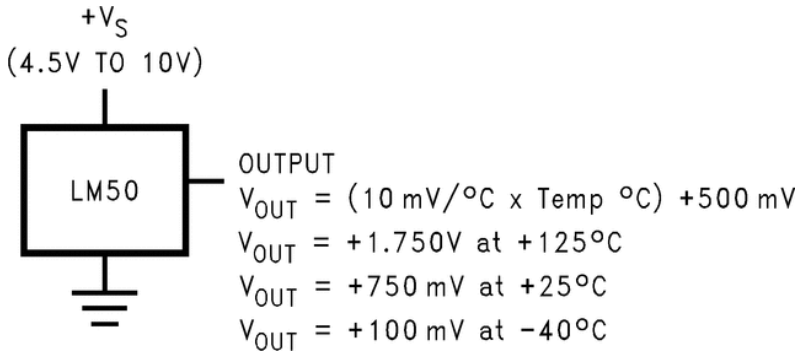
Sensitivity: 10 mV/°C

Pin Configuration and Functions:



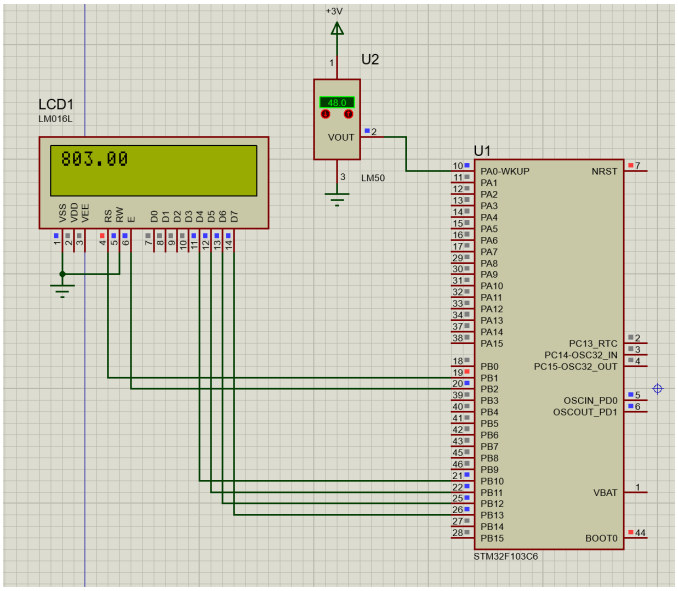
Pin Functions

PIN		TYPE	DESCRIPTION
NO.	NAME		
1	+VS	Power	Positive power supply pin.
2	VOUT	Output	Temperature sensor analog output.
3	GND	Ground	Device ground pin, connected to power supply negative terminal.



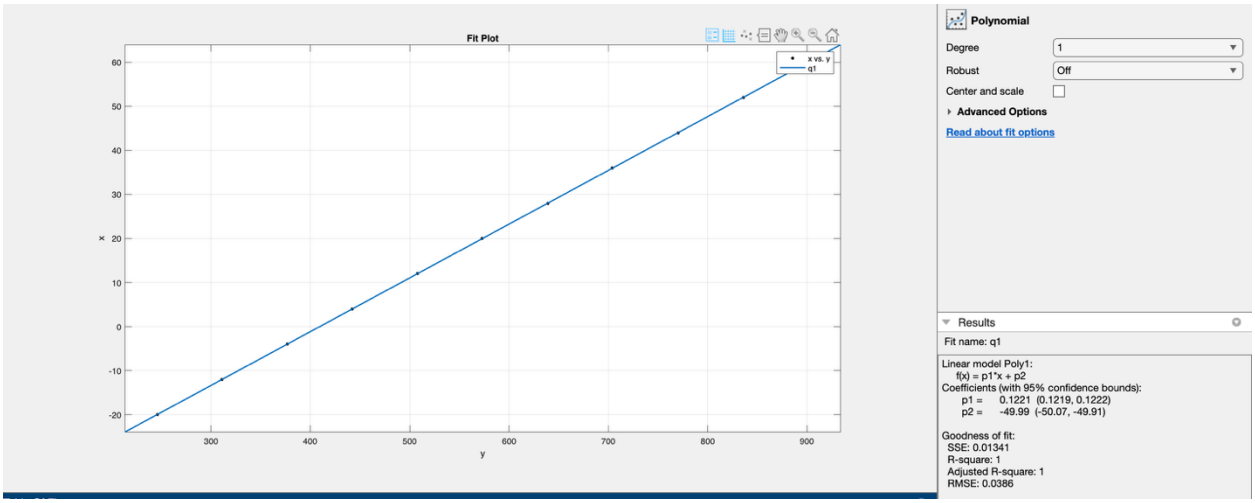
2-

تنظیمات ADC را انجام داده و آن را به میکروکنترلر وصل میکنیم و سپس خروجی دیجیتال آن را روی LCD نمایش میدهیم. برای اینکه رابطه ی بین دما و مقدار نمایش داده شده روی LCD رو متوجه شویم باید چند مقدار را جمع آوری کرده و Curve Fitting انجام دهیم
مقادیر جمع آوری شده به صورت زیر می باشد :

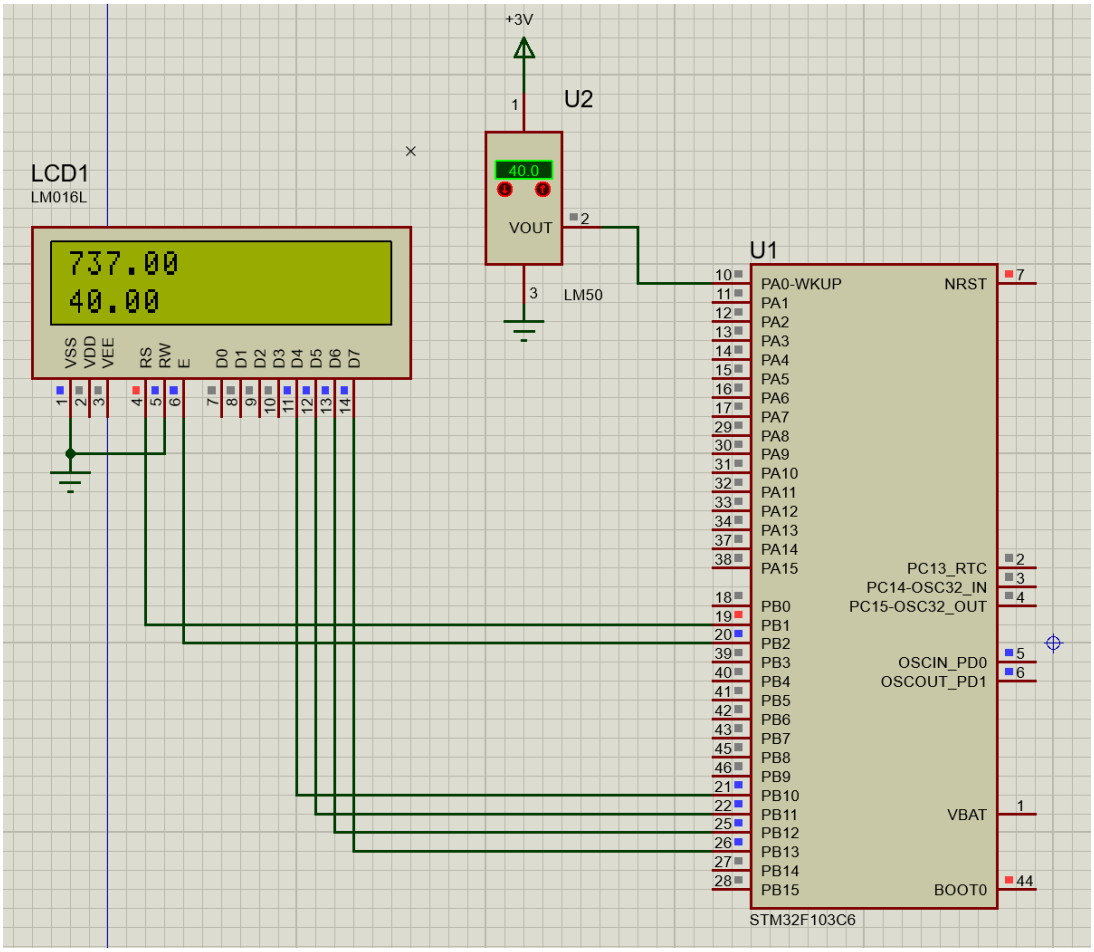


-20	-->	246
-12	-->	311
-4	-->	377
4	-->	442
12	-->	508
20	-->	573
28	-->	639
36	-->	704
44	-->	770
52	-->	836
60	-->	901

سپس در متلب بهترین خطی که ممکن را فیت میکنیم :



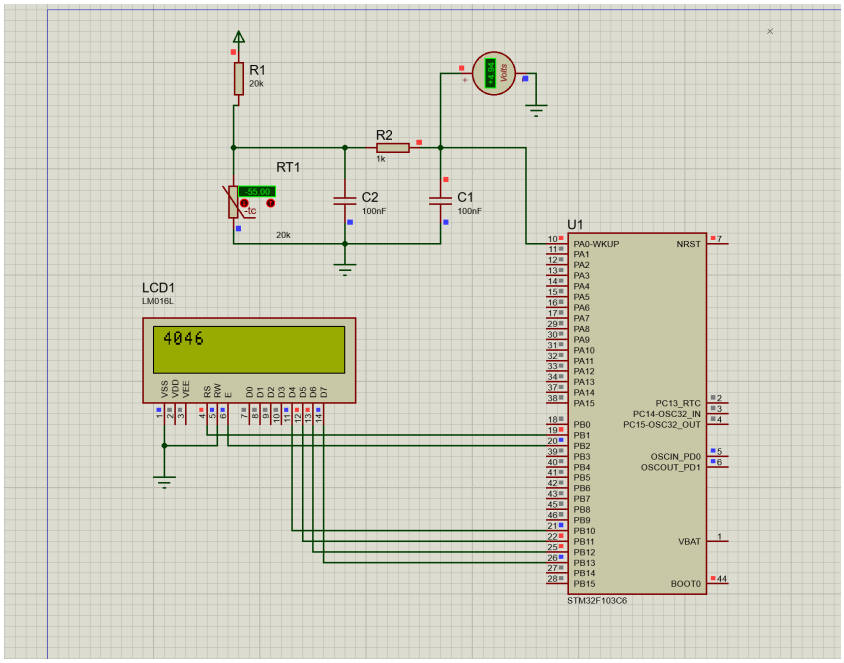
با توجه به خروجی متلب به رابطه ی $temperature = 0.1221 \times adcvalue - 49.99$ میرسیم. با قرار دادن این رابطه در Main.c میتوان مقدار دمای اندازه گیری شده توسط حسگر را به خوبی روی LCD نمایش داد.



با توجه به خروجی رزولوشن سیستم 0.01C و دقت آن حدود 0.05 C است . از آنجایی که دقت سیستم توسط حسگر و رزولوشن توسط میکرو کنترلر میشود با توجه به اینکه مقدار دقت از رزولوشن بیشتر است متوجه میشویم این دو پارامتر به حسگر محدود شده اند.

Q3

برای پیاده سازی این قسمت با استفاده از مدار بهسازی NTC , مدار مورد نظر را به صورت زیر پیاده سازی میکنیم :



خروجی ADC را با استفاده از فرمول زیر به ولتاژ تبدیل کرده و lookup table میسازیم:

$$V = \frac{x * Vref}{2^n - 1}$$

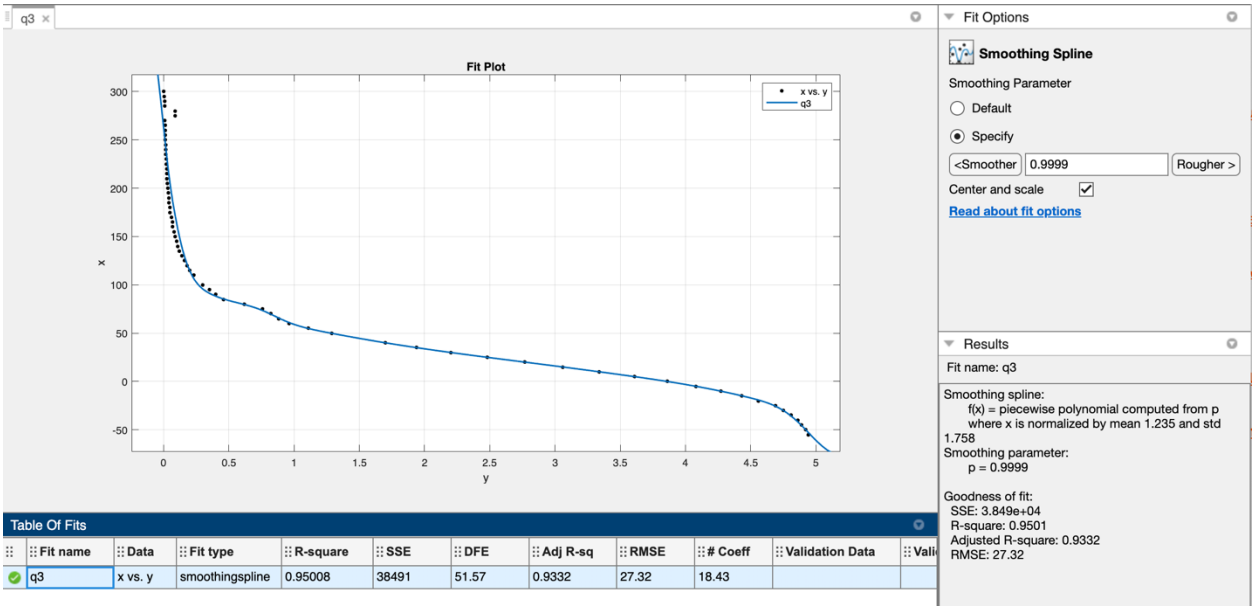
-55 --> 4046 --> 4.94 v
-50 --> 4032 --> 4.92 v
-45 --> 4012 --> 4.89 v
-40 --> 3984 --> 4.86 v
-35 --> 3945 --> 4.81 v
-30 --> 3894 --> 4.75 v
-25 --> 3841 --> 4.69 v
-20 --> 3740 --> 4.56 v
-15 --> 3632 --> 4.43 v
-10 --> 3500 --> 4.27 v
-5 --> 3342 --> 4.08 v
0 --> 3161 --> 3.86 v
5 --> 2958 --> 3.61 v
10 --> 2738 --> 3.34 v
15 --> 2507 --> 3.06 v
20 --> 2270 --> 2.77 v
25 --> 2035 --> 2.48 v

30 --> 1808 --> 2.20 v
35 --> 1594 --> 1.94 v
40 --> 1396 --> 1.70 v
50 --> 1056 --> 1.29 v
55 --> 914 --> 1.11 v
60 --> 790 --> 0.96 v
75 --> 682 --> 0.83 v
80 --> 508 --> 0.62 v
85 --> 381 --> 0.46 v
90 --> 330 --> 0.40 v
95 --> 287 --> 0.35 v
100 --> 250 --> 0.30 v
110 --> 192 --> 0.23 v
115 --> 168 --> 0.20 v
120 --> 148 --> 0.18 v
125 --> 131 --> 0.16 v
130 --> 116 --> 0.14 v

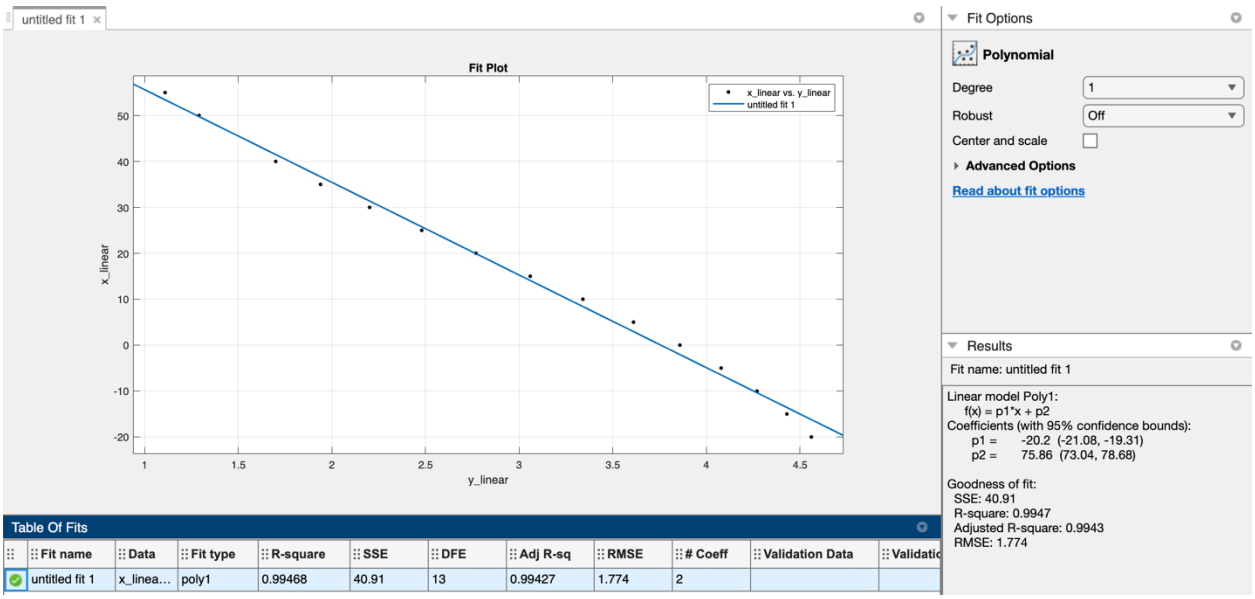
135 --> 103 --> 0.12 v
140 --> 91 --> 0.11 v
145 --> 81 --> 0.1 v
150 --> 73 --> 0.09 v
155 --> 65 --> 0.08 v
160 --> 59 --> 0.07 v
165 --> 53 --> 0.07 v
170 --> 48 --> 0.06 v
175 --> 43 --> 0.05 v
180 --> 39 --> 0.047 v
185 --> 35 --> 0.042 v
190 --> 32 --> 0.039 v
195 --> 29 --> 0.035 v
200 --> 27 --> 0.033 v
205 --> 24 --> 0.029 v
210 --> 22 --> 0.026 v
215 --> 21 --> 0.025 v

220 --> 19 --> 0.023 v
225 --> 17 --> 0.020 v
230 --> 16 --> 0.019 v
235 --> 15 --> 0.018 v
240 --> 14 --> 0.017v
245 --> 13 --> 0.015 v
250 --> 12 --> 0.014 v
255 --> 11 --> 0.013 v
260 --> 10 --> 0.012 v
265 --> 10 --> 0.012 v
270 --> 9 --> 0.010 v
275 --> 8 --> 0.09 v
280 --> 8 --> 0.09 v
285 --> 7 --> 0.008 v
290 --> 6 --> 0.007 v
295 --> 4 --> 0.004 v
300 --> 2 --> 0.002 v

در متلب با استفاده از cftool منحنی adc_out بر حسب temp رو رسم میکنیم تا قسمت خطی را پیدا کنیم :

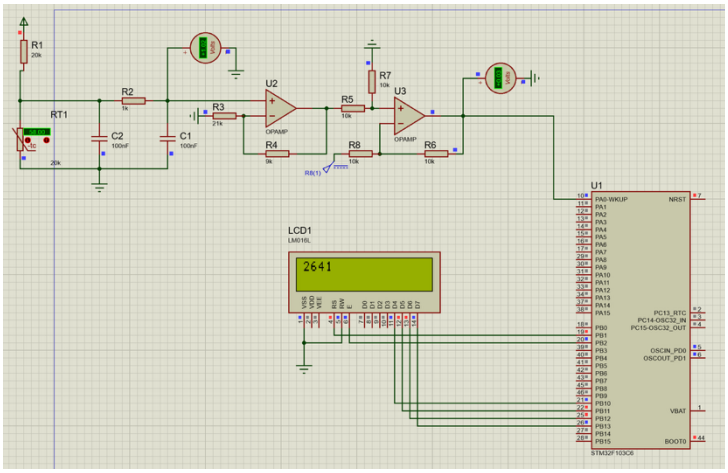


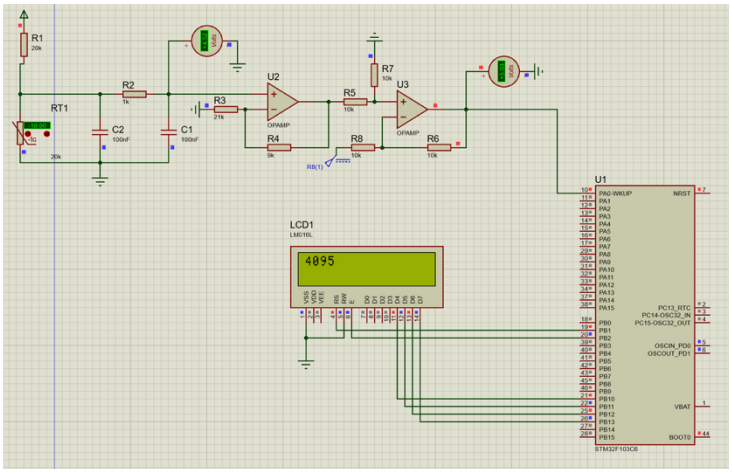
با توجه به منحنی فیت شده ی بالا مشخص میشود که سنسور مورد نظر در بازه ی بین 20- تا 50 درجه به صورت تقریبا خطی عمل کرده و اگر در این بازه cftool رسم کنیم به صورت زیر میشود:



بازه ی کاری سنسور را این بازه انتخاب کرده (1v - 4.5v) و برای نگاشت این ناحیه به 0v – 5v از یک نگاشت خطی با معادله ی $\frac{10}{7}x - \frac{10}{7}$ استفاده خواهیم کرد.

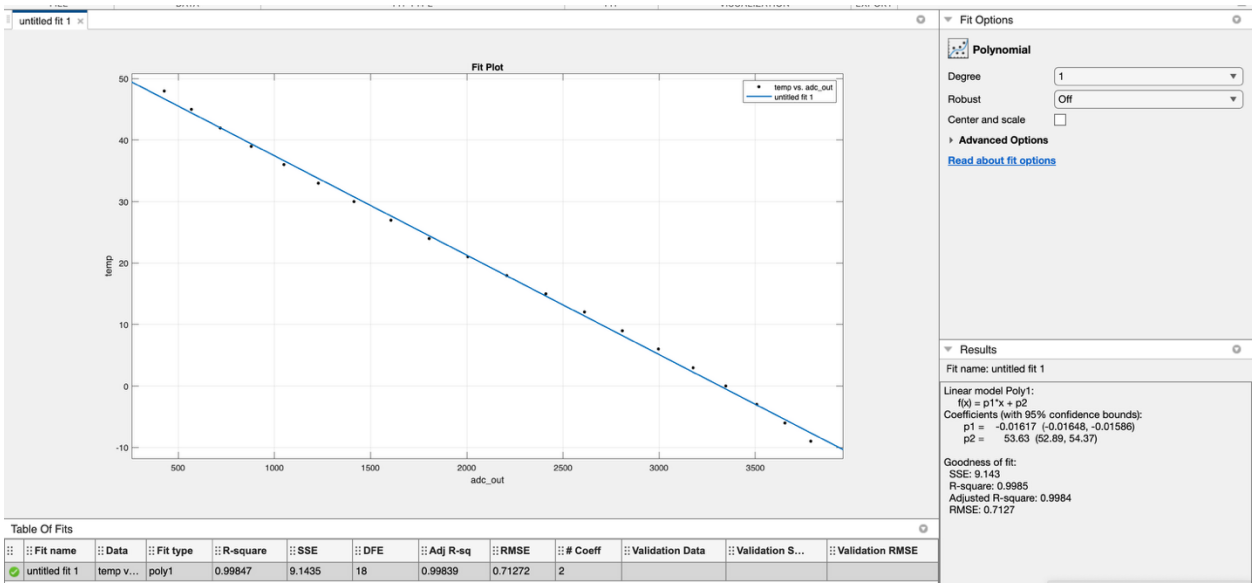
برای پیاده سازی این نگاشت با مدار های بهسازی از یک مدار تقویت کننده با بهره $\frac{10}{7}$ و سپس از یک مدار تفریق کننده استفاده خواهیم کرد. در نهایت مدار به صورت زیر میشود:





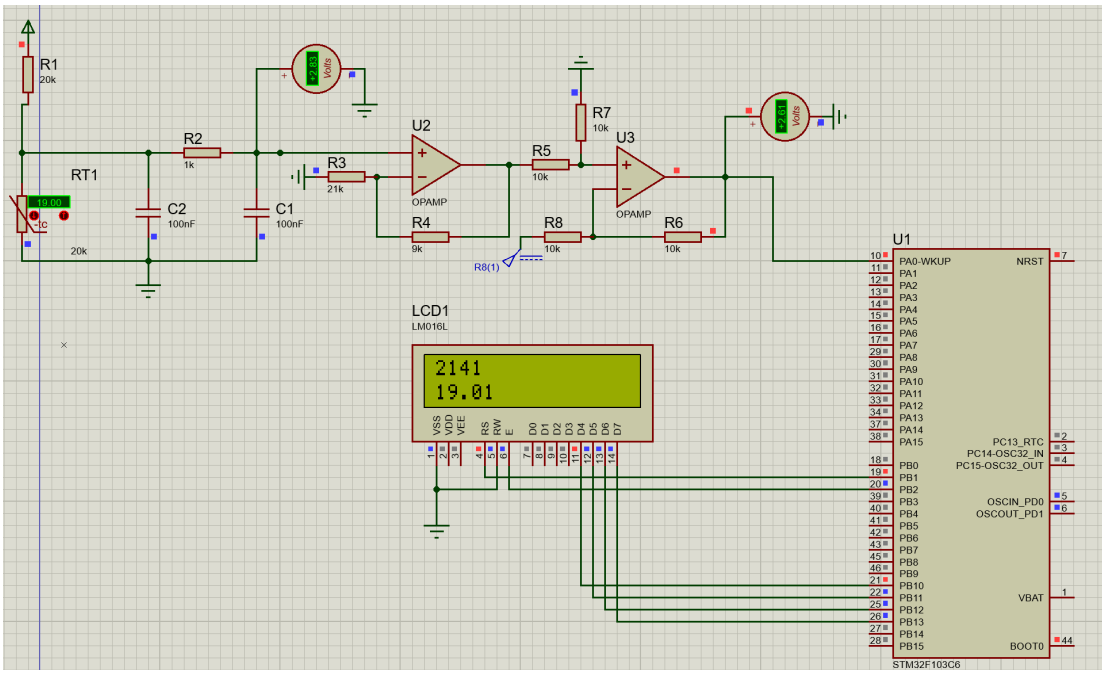
سپس در متلب curve fitting انجام داده و رابطه ی خطی بین خروجی adc و ولتاژ رو پیدا میکنیم :

-18 --> 4095 --> 5.03 v
-15 --> 4020 --> 4.91 v
-12 --> 3911 --> 4.78 v
-9 --> 3789 --> 4.63 v
-6 --> 3654 --> 4.46 v
-3 --> 3507 --> 4.28 v
0 --> 3348 --> 4.09 v
3 --> 3177 --> 3.88 v
6 --> 2997 --> 3.66 v
9 --> 2808 --> 3.43 v
12 --> 2612 --> 3.19 v
15 --> 2412 --> 2.94 v
18 --> 2209 --> 2.70 v
21 --> 2006 --> 2.45 v
24 --> 1805 --> 2.20 v
27 --> 1607 --> 1.96 v
30 --> 1414 --> 1.73 v
33 --> 1228 --> 1.50 v
36 --> 1049 --> 1.28 v
39 --> 880 --> 1.07 v
42 --> 719 --> 0.88 v
45 --> 568 --> 0.69v
48 --> 427 --> 0.52 v
51 --> 296 --> 0.36 v
54 --> 174 --> 0.21 v
57 --> 62 --> 0.08 v
58 --> 26 --> 0.03 v
60 --> 0 --> 0.00 v



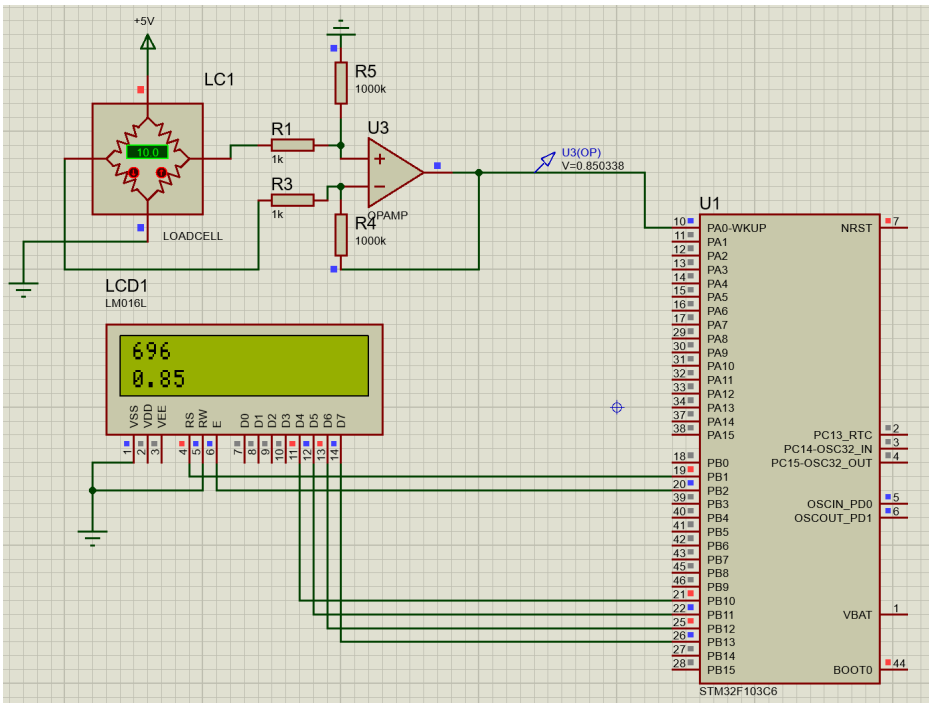
اکنون رابطه ی خطی بدست آمده را درstm32 قرار داده تا خروجی دمای اندازه گیری شده ی سنسور را به 2 رقم اعشار ببینیم:

Temperature = -0.01617 * adc_out + 53.63



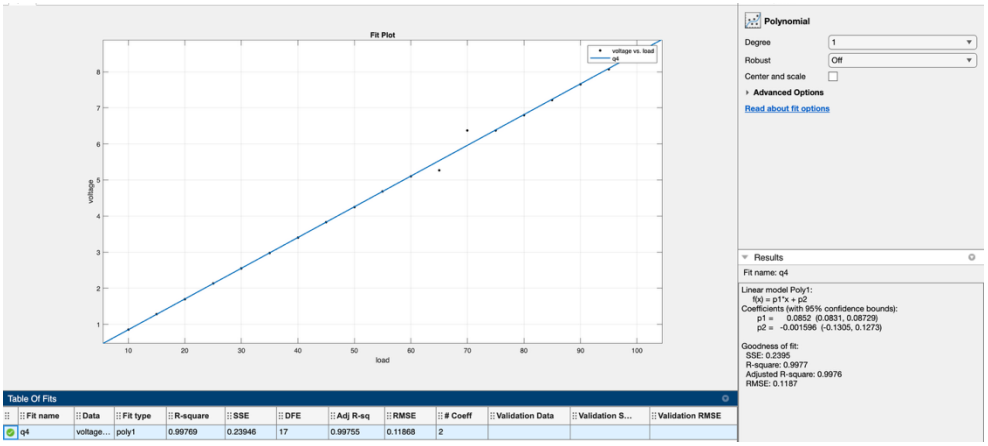
Q4

با استفاده از یک تقویت کننده تفاضلی با بهره 1000 خروجی load cell را به میکرو متصل کرده و خروجی آن را روی LCD نمایش میدهیم :



برای محاسبه ی zero drift و sensivity drift باید چندین نمونه از بار روی load cell و خروجی ولتاژ میکرو بگیریم و یک خط به آن فیت کنیم :

10 ---> 0.85 v
15 ---> 1.28 v
20 ---> 1.70 v
25 ---> 2.13 v
30 ---> 2.55 v
35 ---> 2.98 v
40 ---> 3.40 v
45 ---> 3.83 v
50 ---> 4.25 v
55 ---> 4.68 v
60 ---> 5.10 v
65 ---> 5.27 v
70 ---> 6.37 v
75 ---> 6.37 v
80 ---> 6.80 v
85 ---> 7.22 v
90 ---> 7.65 v
95 ---> 8.07 v
100 ---> 8.5 v



Voltage = 0.0852 * load – 0.001596

Sensivity drift = 0.0852 Zero drift = -0.001596

با مقدار لودسل برابر شود باید از رابطه ی بدست آمده LCDبرای اینکه خروجی نمایش داده شده در آن را باز نویسی کنیم :Loadcellاستفاده کنیم و برحسب

$$\text{loadcell value} = \frac{voltage + 0.001596}{0.0852}$$

با وارد کرد این رابطه در main.c میتوان ورودی loadcell را روی LCD نشون داد.
به دلیل اینکه به میکروکنتلر نمیتوان ولتاژ منفی داد برای اینکه بتوان قسمت منفی رو پوشش داد باید از یک offset استفاده کنیم تا ورودی میکرو مثبت داده شود سپس در کد این offset رو کم میکنیم تا مقدار ولتاژ درست شود.(offset = 2.6v) :

$$\text{loadcell value} = \frac{voltage - 2.6 + 0.001596}{0.0852}$$