***NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY***

***School of Electrical Engineering and Computer Sciences***

***Microprocessor Systems (EE-222)***
***ASSIGNMENT No.*** 2

***ASSIGNMENT- TITLE****:* *TinyML on AVR*

***SUBMITTED TO:*** *Dr. Abid Rafique*

***CLASS + SECTION:*** *BEE-13A*

***SUBMISSION DATE:*** *27/04/2023*

***SUBMITTED BY:***


**1.** Muhammad Sami Ullah - 369138

**2.** Zohaib Irfan - 372692

# TinyML on AVR

## Introduction:

This report presents the design and implementation of a microcontroller-based temperature sensing system that utilizes various concepts learned in the Microprocessor Systems course. The main objective of the project was to design a system that could acquire temperature data from a sensor, process it, display it on a Seven Segment Display, and communicate it with a PC using serial communication.

## Implementation Procedure:

The procedure and implementation process for this project can be outlined as follows:

**1. Sensor Data Acquisition:** The first step is to configure a timer to acquire data from the sensor after every second. Upon receiving an interrupt from the timer, data is acquired from the ADC and displayed on a Seven Segment Display.

**2. Pre-Processing:** In the second step, a 4-tap moving average filter is implemented to pre-process the acquired data. The filtered output (voltage) will be displayed on a LCD.

**3. Min/Max Block:** The pre-processed output is passed on to a min/max computation block. Based on fixed settings within the code, either min/max or nominal value (voltage) will be displayed on the LCD.

**4. Temperature Calibration:** The pre-processed voltage is calibrated using formulas. A temperature value is obtained, passed on to the min/max computation module, and displayed on a LCD.

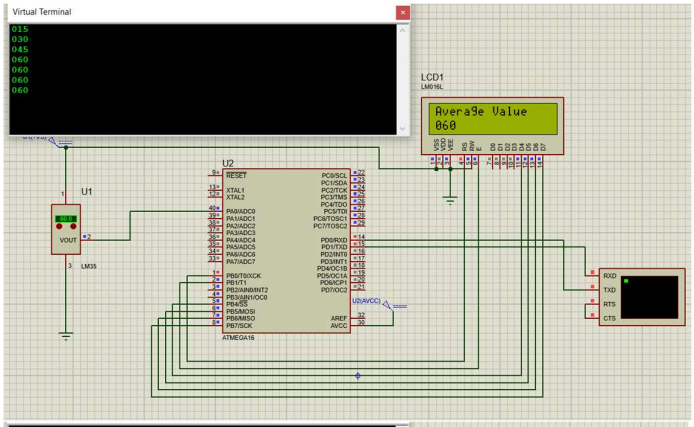
**5. Serial Port (AVR Side):** The serial port on the AVR microcontroller is configured, and data will be sent to the PC, where it will be displayed on the teraterm software within the PC.

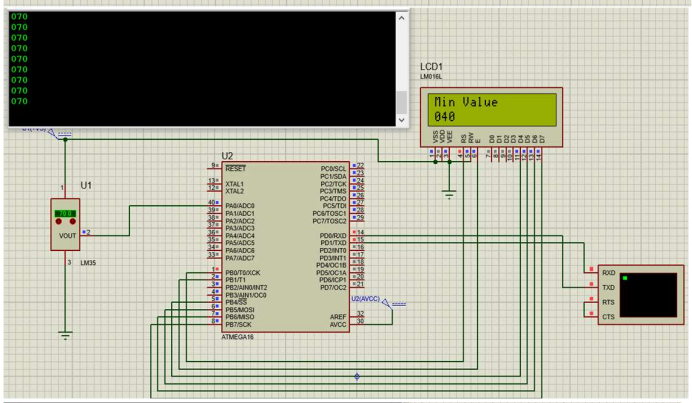
**6. Serial Port (AVR + PC):** The serial port on the PC is configured using Python, and data coming from the serial port will be displayed on the console and in a graph using matplotlib.

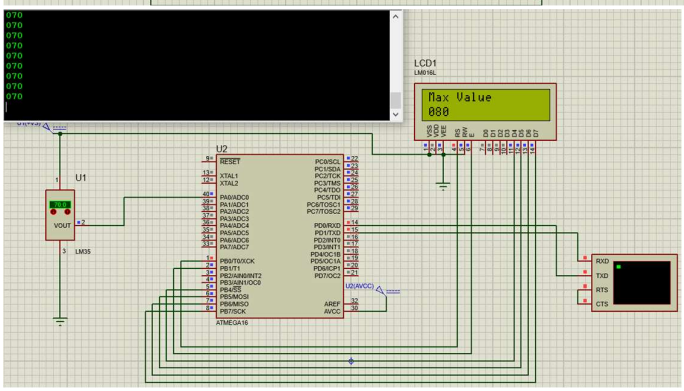
**7. Commands:** Commands will be sent from the PC to the AVR over the serial port for displaying either min/max/nominal value of temperature on the LCD.

**8. Training ML Model:** A number of readings of different temperature settings of the room will be noted down, along with readings of voltage on the LCD and temperature value from a digital thermometer. At least 20 readings of x (sensor data voltage) and y (temperature reading of the digital thermometer) will be noted down. The ScikitLearn library in Python will be used for linear regression to estimate the parameters. These parameters will be used to implement the equation to obtain temperature using the machine learning model.

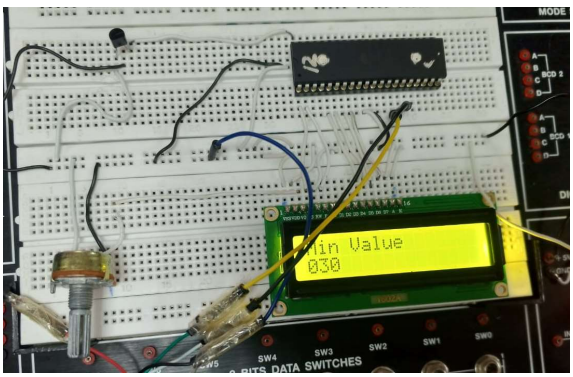
## Proteus Simulation:

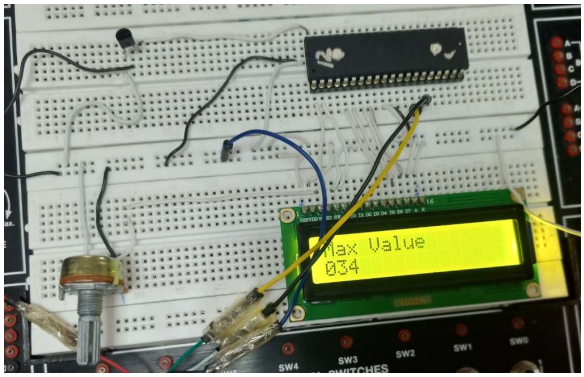


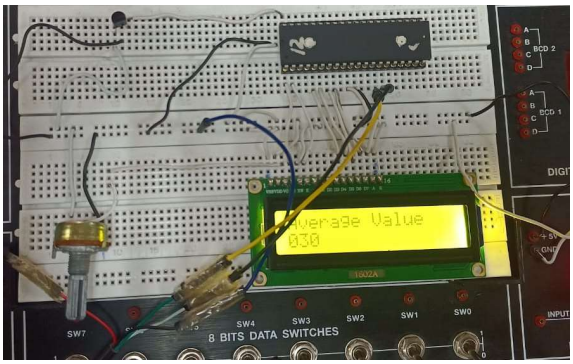




## Hardware:






## TeraTerm Display:


## ML Algorithm:

Following is our data set:

| | Temperature | LM35 Readings |
|---|---|---|
| 1 | 24.0 | 19.0 |
| 2 | 22.5 | 23.0 |
| 3 | 23.0 | 23.0 |
| 4 | 24.6 | 23.0 |
| 5 | 25.0 | 24.0 |
| 6 | 25.0 | 25.0 |
| 7 | 22.0 | 21.0 |
| 8 | 23.6 | 23.0 |
| 9 | 26.0 | 25.0 |
| 10 | 25.5 | 25.0 |
| 11 | 32.0 | 31.0 |
| 12 | 33.0 | 33.0 |
| 13 | 34.6 | 33.0 |
| 14 | 35.0 | 35.0 |
| 15 | 31.0 | 31.0 |
| 16 | 32.4 | 32.0 |
| 17 | 33.5 | 33.0 |
| 18 | 34.5 | 34.0 |
| 19 | 35.0 | 34.5 |
| 20 | 34.0 | 34.0 |

This is the ML Algorithm

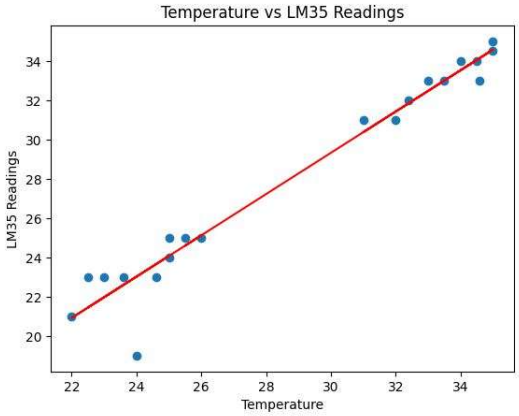
```
import pandas as pd
from sklearn.linear_model import LinearRegression

# Create the dataframe from the given data
data = {'Temperature': [24.0, 22.5, 23.0, 24.6, 25.0, 25.0, 22.0, 23.6, 26.0, 25.5,
                        32.0, 33.0, 34.6, 35.0, 31.0, 32.4, 33.5, 34.5, 35.0, 34.0],
        'LM35 Readings': [19.0, 23.0, 23.0, 23.0, 24.0, 25.0, 21.0, 23.0, 25.0, 25.0,
                          31.0, 33.0, 33.0, 35.0, 31.0, 32.0, 33.0, 34.0, 34.5, 34.0]}
df = pd.DataFrame(data)

# Create the linear regression model and fit it on the data
X = df[['Temperature']]
y = df['LM35 Readings']
model = LinearRegression().fit(X, y)

# Print the intercept and coefficients of the model
print('Intercept: ', model.intercept_)
print('Coefficient: ', model.coef_[0])
```

```
Intercept:  -2.1773954027559768
Coefficient:  1.0500657897520296
```

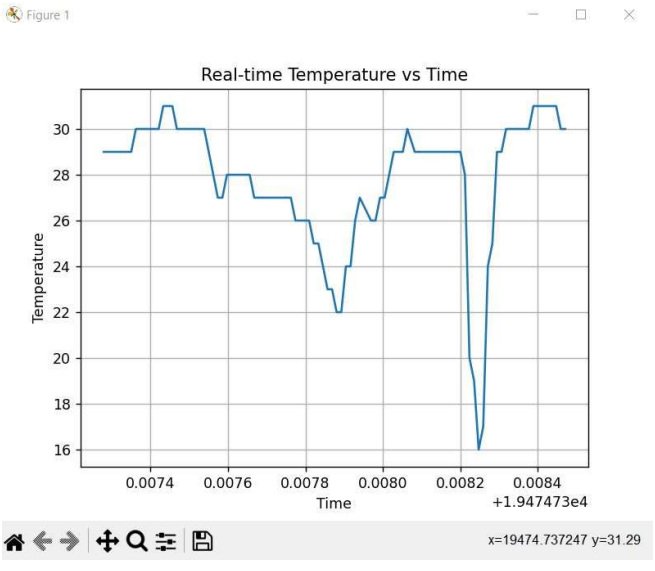
We have obtained following equation:

```
Actual Value = 1.05*(LM35) -2.1774
```

Following is the scatter plot of data set and the hypothesis line.




## Real Time Plot



## Challenges Faced:

- **Configuring ADC:** The Analog-to-Digital Converter (ADC) used in the project posed a challenge during the implementation process. Configuring the ADC involved setting the correct prescaler and reference voltage, as well as understanding the conversion process. This required a good understanding of the datasheet and thorough testing to ensure that the ADC was working correctly.

- **Configuring LCD Display:** Another challenge encountered during the implementation process was configuring the LCD display. This involved setting up the correct pins, defining the LCD commands, and understanding the timing requirements for each command. Debugging this process involved using an oscilloscope to measure the signal timing and comparing it to the expected values.

- **Configuring PC to plot real-time graph:** The final challenge faced during the implementation process was configuring the PC to plot real-time graphs using Python. This involved setting up the correct libraries, understanding the data transfer protocol, and implementing the correct data processing algorithms.

## Conclusion:

In conclusion, this project aimed to apply the concepts learned in Microprocessor Systems, specifically related to microcontroller programming, such as sensor data acquisition, timers, interrupts, serial port communication, and display technologies. Through the implementation process, we were able to successfully develop a system that acquired data from a sensor, pre-processed the data, computed temperature values, and displayed the results on a seven-segment display. Additionally, we were able to establish serial communication between the microcontroller and a PC, allowing us to send and receive data for further analysis and plotting real-time graphs.

However, during the implementation, we encountered some challenges related to configuring the ADC, LCD display, and plotting real-time graphs on the PC. These challenges required extra effort and research to overcome.

Overall, this project provided a valuable learning experience and an opportunity to apply the concepts learned in class to a real-world problem. The project also demonstrated the potential for microcontrollers to be used in a wide range of embedded systems applications, such as temperature monitoring and control systems.