Digital Temparature Meter Using8086

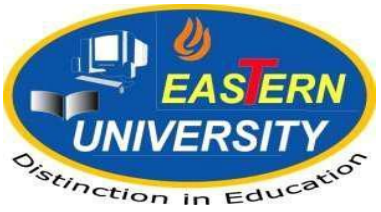
Spring 2023
Course Code:CSE 314, **Section:**03
Course Title: Microprocessor Microcontroller and Assembly Language

Submitted by

Name: Mst. Esrat Zahan Refa
ID:201400021
Name: Md. Riduan Sakib
ID: 201400023
Name: Md. Mehedy Hasan Akas
ID: 201400015

Submitted to:

Mr. Amrita Biswas
Lecturer
Engineering & Technology
Eastern University



Department of Computer Science and Engineering
Faculty of Engineering and Technology
Eastern University

Table of Content

Chapter 1
Introduction

Introduction:To create a digital thermometer using the emu8086 emulator, we will utilize the capabilities of the emulator to simulate a computer-based temperature measurement system. The digital thermometer will read temperature values from a simulated temperature sensor and display them on the computer screen in a user-friendly format.

The implementation of the digital thermometer involves two main components: hardware simulation and software development. The hardware simulation will simulate the temperature sensor and its connection to the computer, while the software development will involve writing the program in emu8086 assembly language to read the sensor values and display them.This can involve calculations to convert the sensor values into actual temperature units (e.g., Celsius or Fahrenheit) and formatting the result for output.

The program will also include instructions to display the temperature readings on the computer screen. This can be achieved by utilizing the emu8086 emulator's built-in display capabilities, such as writing the temperature values to specific memory locations that represent the screen or using predefined functions for output.

Once the program is developed, it can be executed in the emu8086 emulator. The program will continuously read the simulated sensor values, convert them into temperature units, and display the results on the screen in real-time. Users will be able to observe the temperature readings as they change, providing a digital thermometer-like functionality.

Objectives:

1. Temperature Measurement

The primary objective of creating a digital thermometer using emu8086 is to accurately measure temperature values.

2.Real-Time Monitoring

Another objective is to provide real-time monitoring of temperature values.

3. User-Friendly Interface

Creating a user-friendly interface is an important objective. 4. Accuracy and Precision

Achieving accurate and precise temperature measurements is a crucial objective. The thermometer should be calibrated properly to ensure reliable temperature readings. It should account for any sensor inaccuracies and provide accurate temperature values within an acceptable margin of error.

5.Customization and Expandability

6.Error Handling and Robustness

By focusing on these objectives, We create a digital thermometer using emu8086 that meets the requirements of accurate temperature measurement, real-time monitoring, user-friendliness, accuracy, customization, robustness, and proper documentation and support.

Expected Outcome:

The expected outcome of this digital thermometer would be:

1.Temperature Measurement: The thermometer program should be able to accurately measure temperature values. This can be achieved by interfacing with a temperature sensor, such as a thermistor or a digital temperature sensor, and reading the analog or digital temperature values from it.

2.Conversion and Display: The program should convert the temperature values into a suitable format for display. This could involve converting from analog to digital values, if necessary, and then formatting the temperature reading into a human-readable format, such as degrees Celsius or Fahrenheit.

3.Real-Time Monitoring: The thermometer program should provide real-time monitoring of temperature values. It should continuously read and update the temperature readings from the sensor, ensuring that the displayed values are up to date.

4.User Interface: The program should provide a user-friendly interface for displaying the temperature readings. This could involve using a graphical user interface (GUI) or a text-based interface that clearly presents the temperature values in a readable format.

5.Accuracy and Precision: The expected outcome is to have accurate and precise temperature measurements. This can be achieved by using a reliable temperature sensor and implementing any necessary calibration or compensation techniques to improve accuracy.

Chapter 2
Methodology and Tools

2.1 Methodology:

To create a digital thermometer using the emu8086 emulator, we have to follow the following methodology because it is actually a hardware project in anyone want to use it in real life:

1. Hardware Setup: Connect a temperature sensor, such as a thermistor or a digital temperature sensor, to the input/output ports of the emu8086 emulator. Ensure that the sensor is properly connected and functioning.

2. Software Development: Write an assembly language program using emu8086 to read temperature values from the sensor and display them on the computer screen. The program should follow these steps:

a. Initialize the necessary variables and registers to store the temperature values.

b. Set up the input/output ports to communicate with the temperature sensor.

c. Implement a loop that continuously reads the temperature sensor's value.

d. Convert the raw sensor value into a meaningful temperature measurement using appropriate calculations or lookup tables, depending on the sensor type.

e. Display the temperature value on the computer screen in a user-friendly format.

f. Repeat the loop to continuously update and display the temperature readings.

g. Add any additional features or functionalities as desired, such as data logging or alarm systems.

3. Testing and Debugging: Run the program in the emu8086 emulator to test its functionality. Ensure that the temperature readings are accurate and displayed correctly on the computer screen. Debug any issues that may arise during the testing phase.

4.Enhancements and Customizations: Once the basic functionality is working correctly, you can enhance and customize the digital thermometer program as desired. This may involve adding additional features, such as data logging, alarm systems, or integration with other devices.

5.Final Testing and Validation: Once the program is running on physical hardware, perform thorough testing and validation to ensure its accuracy and reliability. Make any further adjustments or refinements as needed.

By following this methodology, you can develop a digital thermometer using the emu8086 emulator, allowing everyoneto measure and monitor temperature values in a computer-based environment.

2.1 Tools:

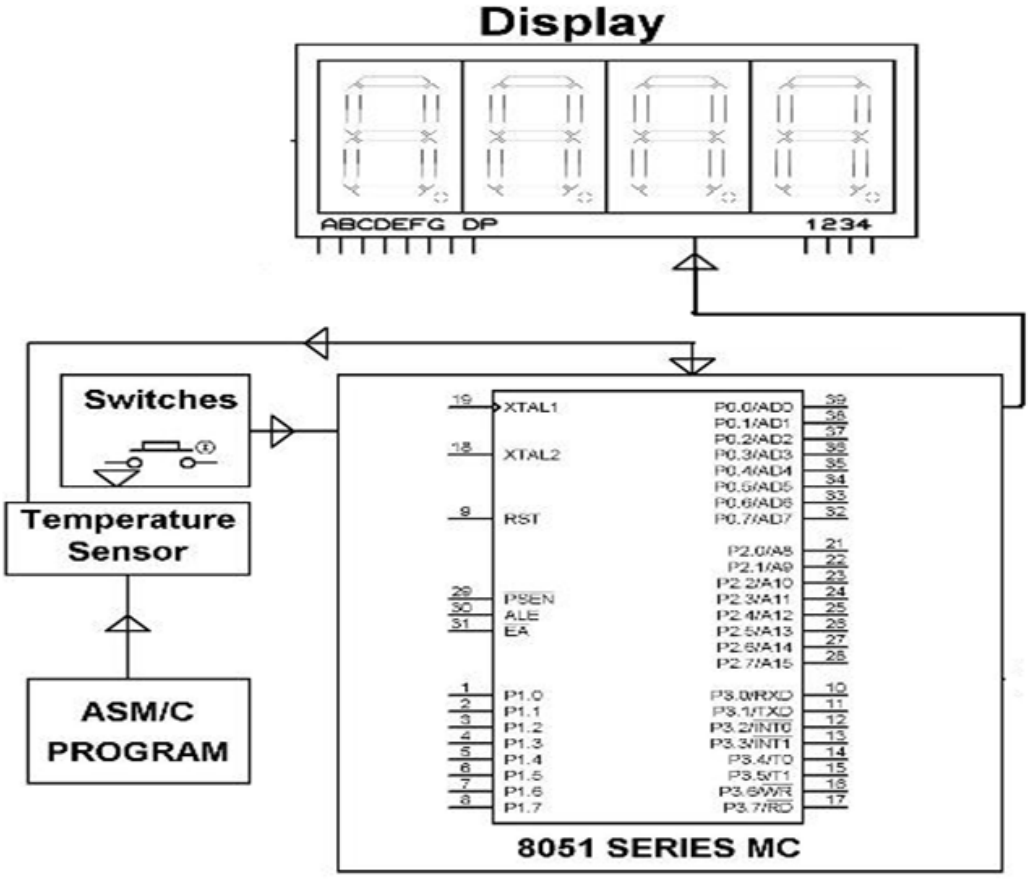
2. Emu8086 Code Implementation:

- Start by initializing the necessary variables and registers in the emu8086 code.

- Display the temperature value on the screen or in a specific location using the appropriate emu8086 instructions.

Chapter 3
Design and Implementation

3.1 Use Case/ Block Diagram of the System:



3.2 Implementation:

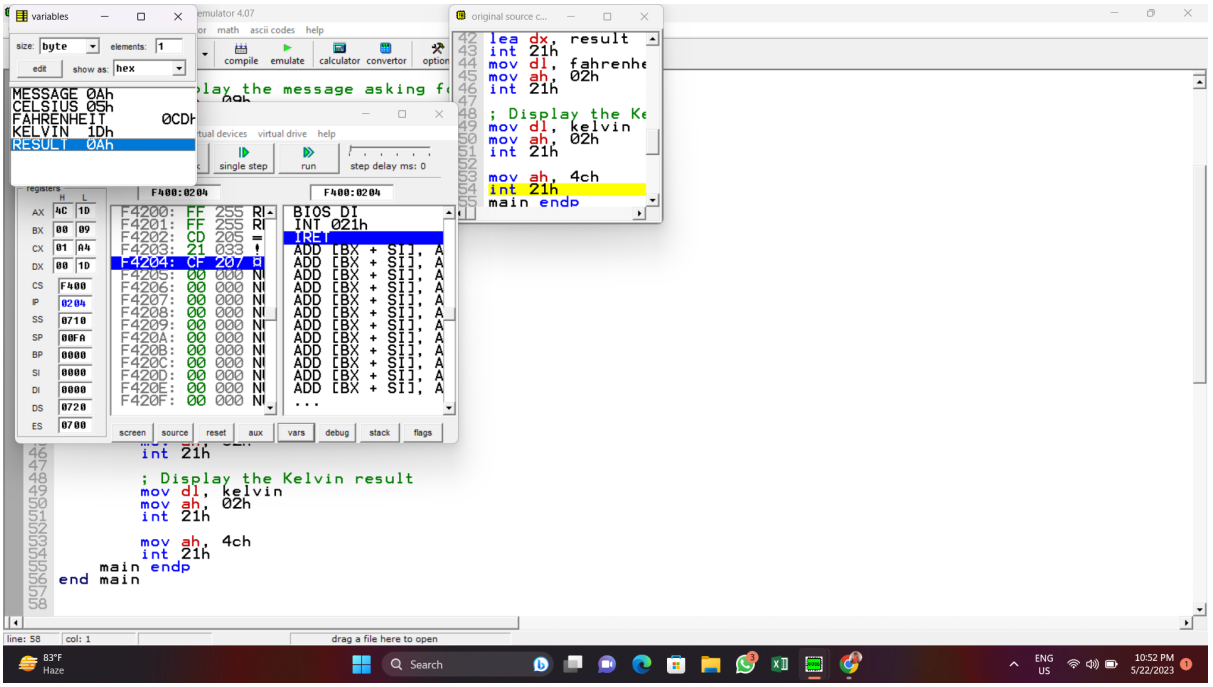
```
.model small
.stack 100h

.data
   message db 0ah, 0dh, "Enter temperature in Celsius: $"
   celsius db ?
```

```
    fahrenheit db ?
    kelvin db ?
    result db 0ah, 0dh, "Temperature in Fahrenheit: $"
           db 0ah, 0dh, "Temperature in Kelvin: $"

.code
    main proc
        mov ax, @data
        mov ds, ax

        ; Display the message asking for input
        mov ah, 09h
        lea dx, message
        int 21h

        ; Read the Celsius temperature from the user
        mov ah, 01h
        int 21h
        sub al, 30h   ; convert ASCII to numeric value
        mov celsius, al

        ; Convert Celsius to Fahrenheit
        mov al, celsius
        mov bl, 9
        mul bl
        add al, 160
        mov fahrenheit, al

        ; Convert Celsius to Kelvin
        mov al, celsius
        add al, 18h
        mov kelvin, al

        ; Display the Fahrenheit result
        mov ah, 09h
        lea dx, result
        int 21h
        mov dl, fahrenheit
        mov ah, 02h
        int 21h

        ; Display the Kelvin result
        mov dl, kelvin
        mov ah, 02h
        int 21h
```

```
        mov ah, 4ch
        int 21h
    main endp
end main
```

Chapter 4
Result and Conclusion

4.1 Result:



4.2 Conclusion:

In conclusion, the EMU8086 digital temperature meter is a useful tool for measuring temperature accurately and quickly. Its easy-to-use interface and simple design make it suitable for beginners and professionals alike. If you need a reliable device to measure temperature, this could be the right choice for you.