Temperature Monitoring System

Name: Hrutam Sabale
Class: TE-A
Roll No: TETA05

SYNOPSIS

This project focuses on the design and implementation of a temperature sensing system using an LM35 temperature sensor interfaced with an STM32F407 microcontroller board. The LM35 sensor provides an analog output proportional to the ambient temperature, which is processed by the STM32F407's Analog-to-Digital Converter (ADC) to convert it into a digital value. The system will then display the temperature on two multiplexed 7-segment displays for monitoring purposes.

Key features of the project include accurate temperature readings, low power consumption, and real-time data acquisition. The project involves configuring the ADC of the STM32F407 to read the analog input from the LM35, performing necessary calibration, and developing firmware for data processing and display.

This project has potential applications in temperature monitoring systems, home automation, and industrial controls where precise temperature measurement is crucial.

- # Components

1) LM35 Temperature Sensor
2) STM32F407 Discovery Board
3) 7-Segment Displays (Common Anode)
4) NPN Transistors
5) 2k2 Resistors
6) Jumper Wires
7) Breadboard

- # Software

1) STM32CubeIDE

- # Connections

1) Connect the o/p pin of the LM35 sensor to PORT A pin 0
2) Connect the common anode pins of the first (one's digit) and second (ten’s digit) 7-segment display to PORT E pin 0 and pin 1 respectively via an NPN transistor for each display.
3) Connect the data pins of both displays parallelly to pins 0 - 7 of PORT D. (a to PD.0, b to PD.1, and so on)

CODE

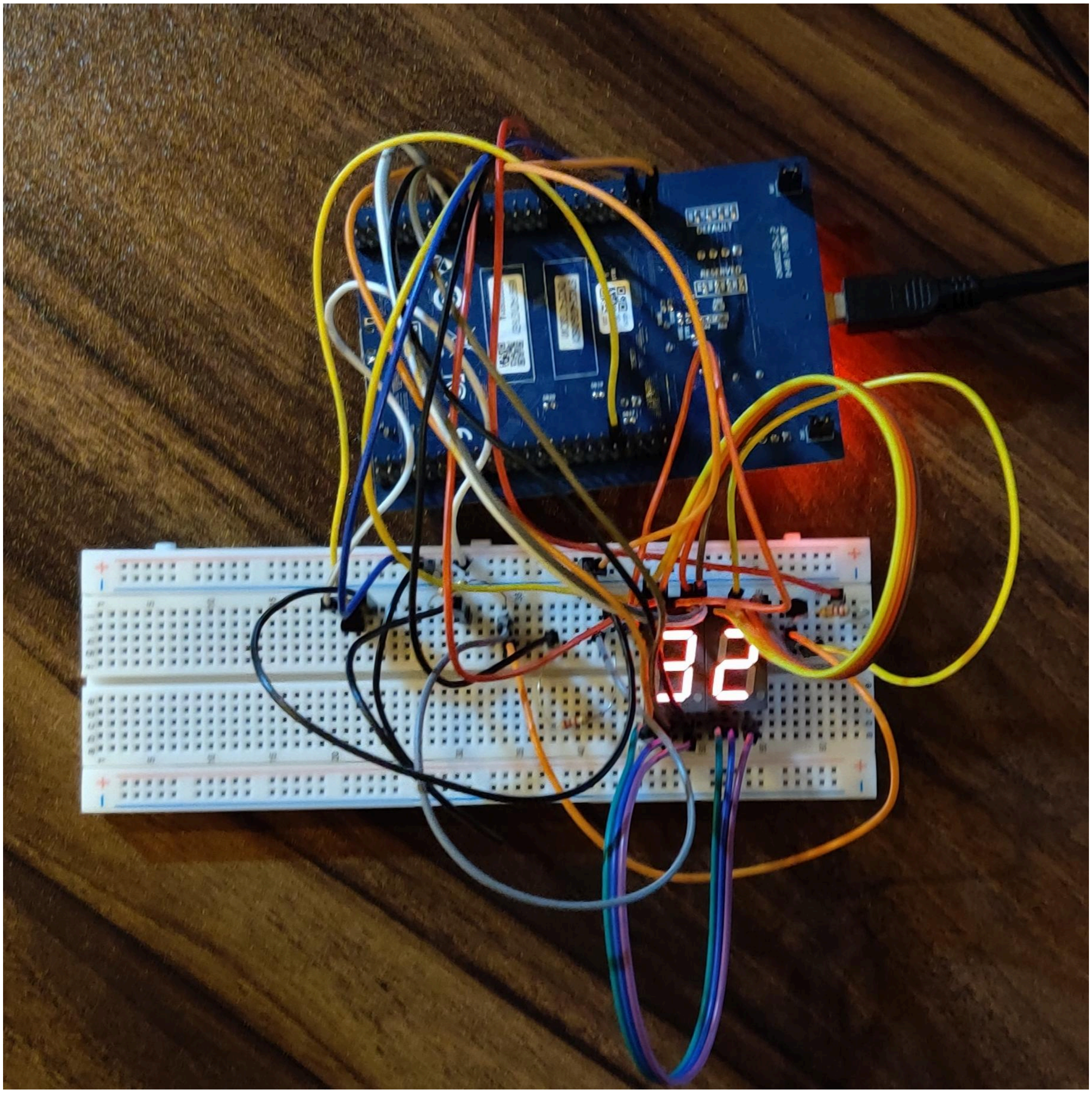
```c
#include <stdint.h>
#include <stdio.h>
#include <math.h>
// Base addresses for the peripherals
#define RCC_BASE        0x40023800
#define GPIOA_BASE      0x40020000
#define GPIOD_BASE      0x40020C00
#define GPIOE_BASE      0x40021000
#define ADC1_BASE       0x40012000

// Offsets for the specific registers
#define RCC_AHB1ENR     *(volatile uint32_t *)(RCC_BASE + 0x30)
#define RCC_APB2ENR     *(volatile uint32_t *)(RCC_BASE + 0x44)

#define GPIOA_MODER     *(volatile uint32_t *)(GPIOA_BASE + 0x00)
#define GPIOD_MODER     *(volatile uint32_t *)(GPIOD_BASE + 0x00)
#define GPIOE_MODER     *(volatile uint32_t *)(GPIOE_BASE + 0x00)
#define GPIOD_OUTPUT     *(volatile uint32_t *)(GPIOD_BASE + 0x14)
#define GPIOE_OUTPUT     *(volatile uint32_t *)(GPIOE_BASE + 0x14)

#define ADC1_SR         *(volatile uint32_t *)(ADC1_BASE + 0x00)
#define ADC1_CR1        *(volatile uint32_t *)(ADC1_BASE + 0x04)
#define ADC1_CR2        *(volatile uint32_t *)(ADC1_BASE + 0x08)
#define ADC1_SMPR2      *(volatile uint32_t *)(ADC1_BASE + 0x10)
#define ADC1_SQR1       *(volatile uint32_t *)(ADC1_BASE + 0x2C)
#define ADC1_SQR3       *(volatile uint32_t *)(ADC1_BASE + 0x34)
#define ADC1_DR         *(volatile uint32_t *)(ADC1_BASE + 0x4C)


void GPIO_Config(void);
void ADC_Config(void);
uint32_t ADC_Read(void);
void delay();
```

```c
uint8_t segment_hex[10] = {
                0xC0,
                0xF9,
                0xA4,
                0xB0,
                0x99,
                0x92,
                0x82,
                0xF8,
                0x80,
                0x90
};

int main(void) {
    uint32_t adc_value;
    int temperature;
    int msb, lsb;

    GPIO_Config();
    ADC_Config();

    while (1) {

        adc_value = ADC_Read();
        temperature = (adc_value * 3.3 / 4096.0) / 0.01;
        msb = temperature / 10;
        lsb = temperature % 10;
        GPIOD_OUTPUT = segment_hex[msb];
        GPIOE_OUTPUT = 0x01;
        delay();
        GPIOD_OUTPUT = segment_hex[lsb];
        GPIOE_OUTPUT = 0x02;
        delay();


    }
}
```

```
void GPIO_Config(void) {

    RCC_AHB1ENR |= 0x19;


    GPIOA_MODER |= (3 << (0 * 2));
    GPIOD_MODER |= 0x5555;
    GPIOE_MODER |= 0x05;
}

void ADC_Config(void) {

    RCC_APB2ENR |= (1 << 8);
    ADC1_CR2 = 0;
    ADC1_SQR1 = 0;
    ADC1_SQR3 |= (0 << 0);
    ADC1_SMPR2 |= (7 << 0);
    ADC1_CR2 |= (1 << 0);
}

uint32_t ADC_Read(void) {

    ADC1_CR2 |= (1 << 30);
    while (!(ADC1_SR & (1 << 1)));
    return ADC1_DR;
}

void delay()
{
        for(int i = 0; i<10000; i++);
}
```

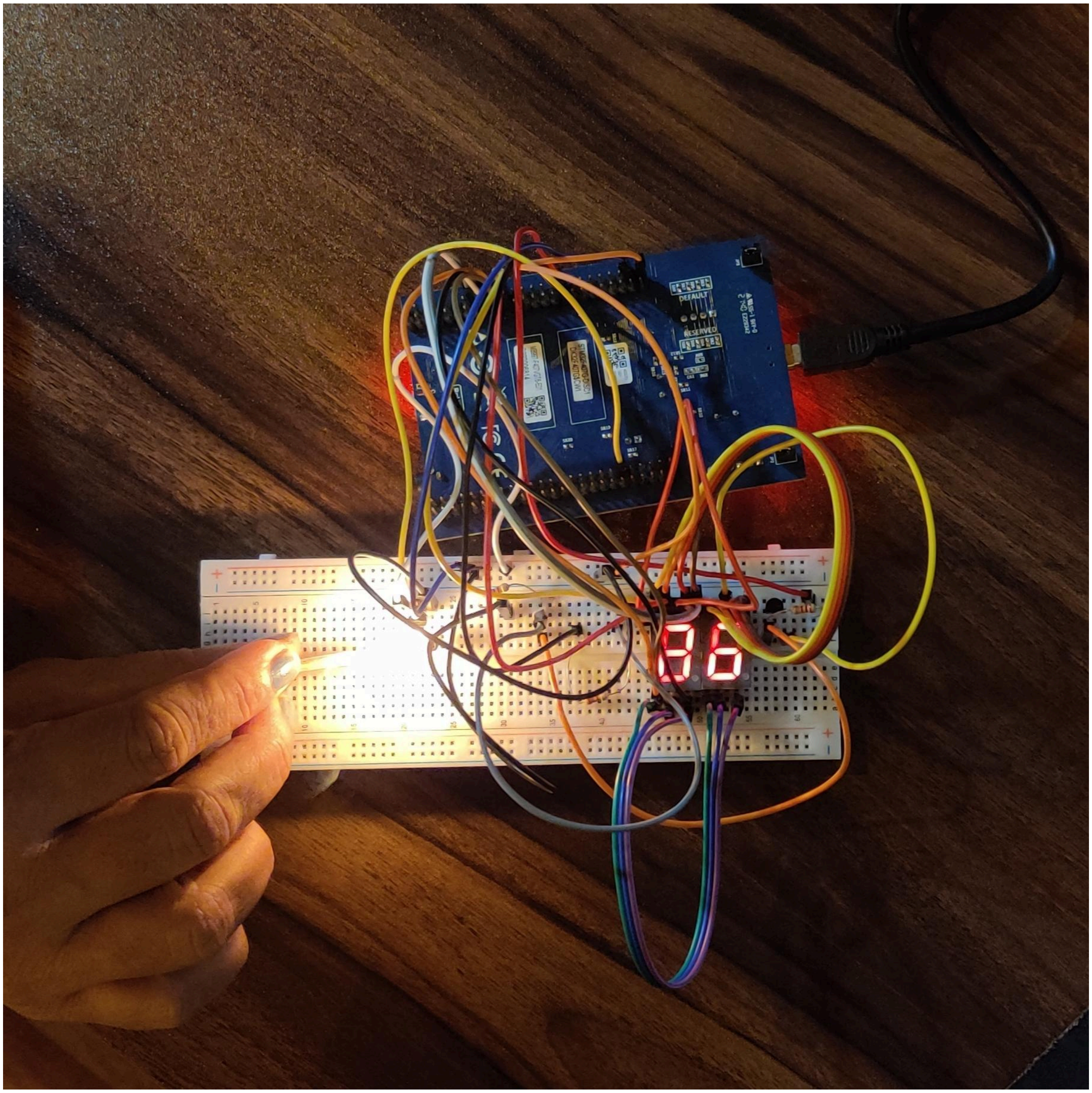
RESULT

At Room Temperature



***display reads 32 degrees Celsius**

In Presence of a Lit Match



display reads 86 degrees Celsius

Conclusion

The implementation of a temperature sensing system using the LM35 sensor and STM32F407 board successfully demonstrates the integration of an analog temperature sensor with a modern microcontroller for real-time monitoring. The system accurately measures ambient temperature and converts it to a digital value using the STM32F407's ADC, with the output displayed.

This project highlights the efficiency and precision of the LM35 sensor, combined with the robust capabilities of the STM32F407 microcontroller. The system is scalable and can be adapted for various applications, from home automation to industrial temperature monitoring. Overall, this project serves as a practical foundation for understanding embedded system design involving sensor interfacing, signal conversion, and data handling, providing room for further enhancements such as wireless communication or data logging capabilities.