Temperature system

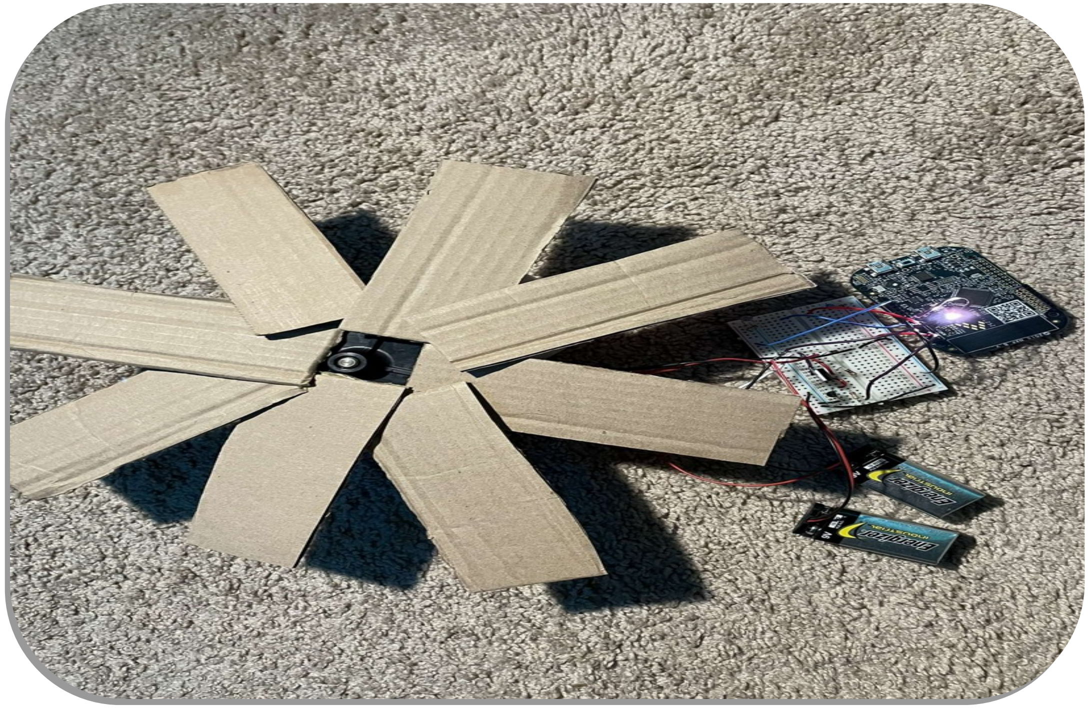
Javier Mendez

Hamad Alzaabi

Kailen King

Marwan Basaham

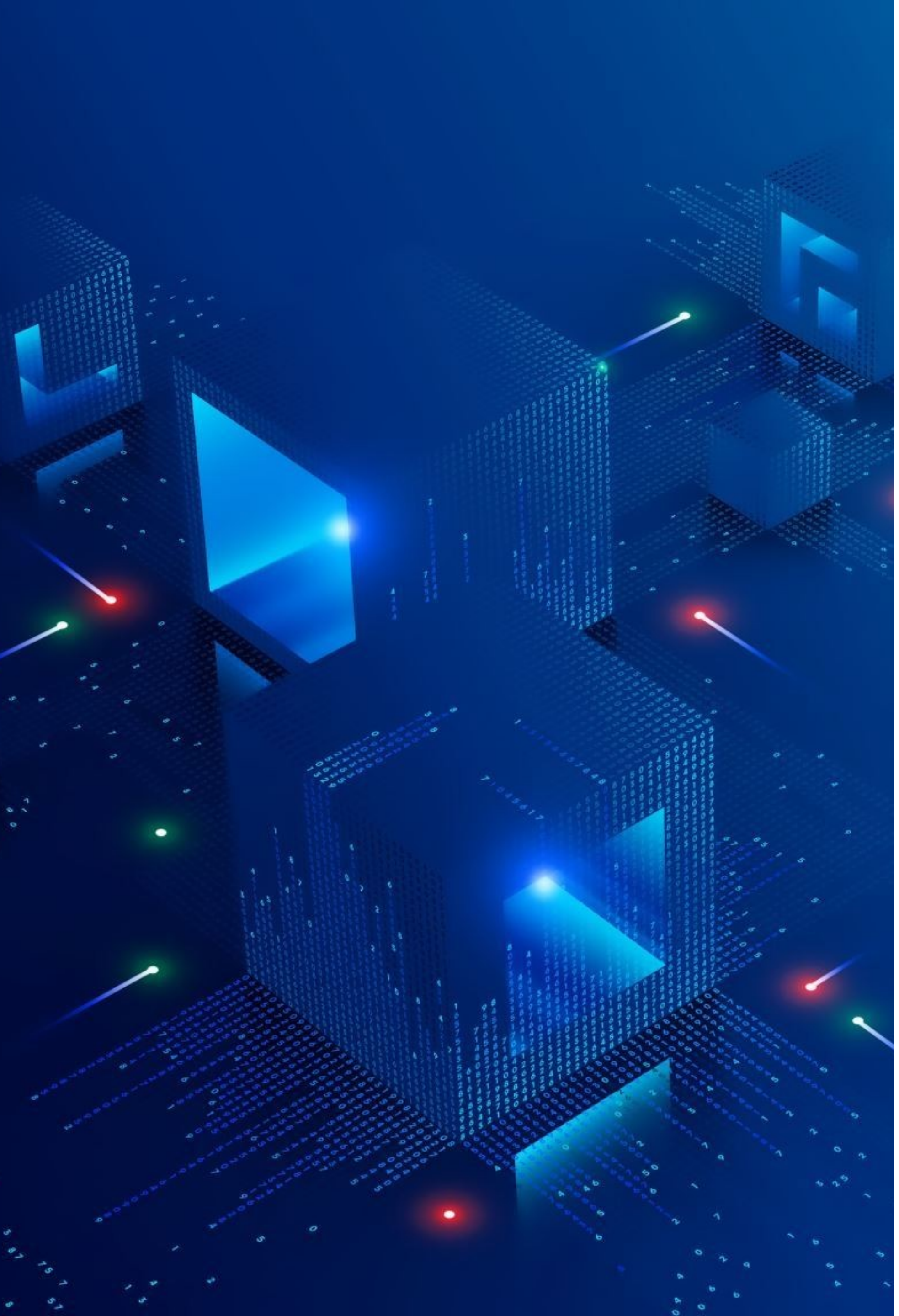
Introduction



- System connected to a fan that measure the temperature and display different colors depending on the measured temperature.

- Above 30°C fan turns on.

-10°C - 0°C	0°C - 10°C	10°C - 20°C	20°C - 30°C	30°C - 40°C
Purple	Blue	White	Yellow	Red



Previous Projects

- Project 2
- ✓ Light sensor
- ✓ Shaded : Nighttime - System On
- ✓ Unshaded : Daytime – System Off
- Project 4
- ✓ Thermistor
- Project 7
- ✓ Code
- ✓ Microcontroller

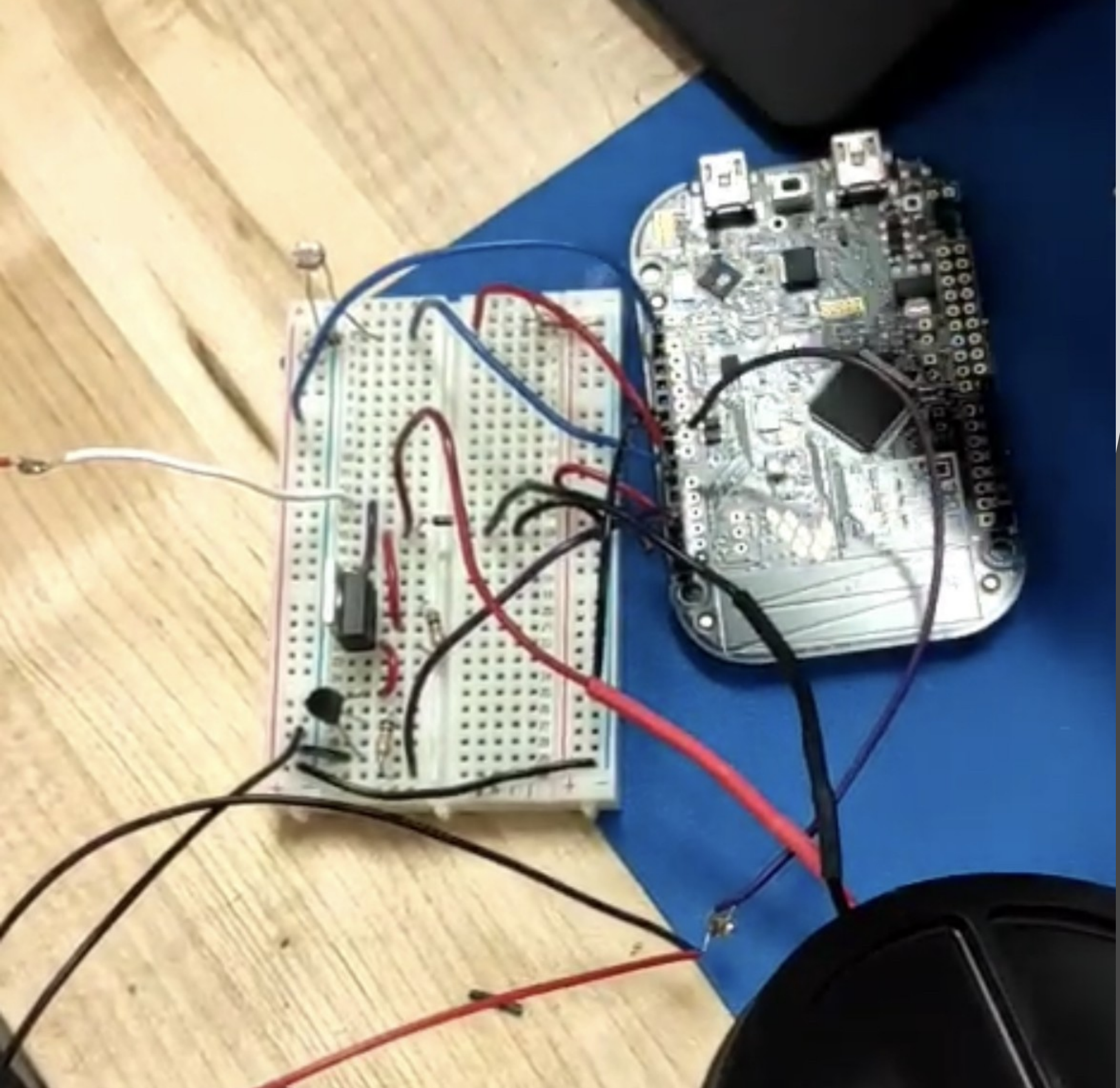
Objectives

BUILDING A SYSTEM THAT CAN READ THE TEMPERATURE AND DISPLAY DIFFERENT COLORS AND DO AN ACTION DEPENDING ON THE TEMPERATURE.

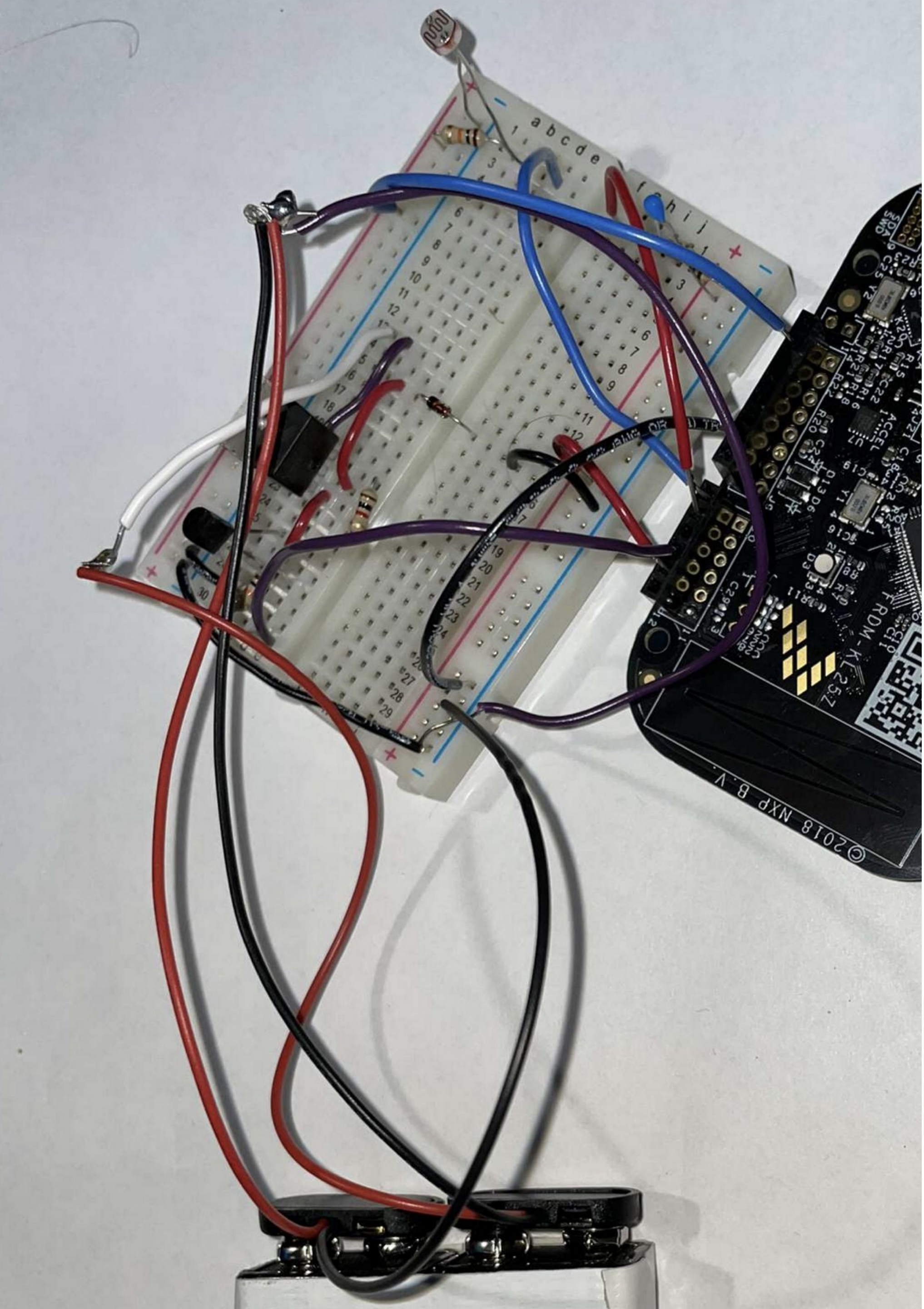
PRACTICING WHAT WE HAVE LEARNED THROUGHOUT THE SEMESTER AND APPLYING IT IN REAL LIFE IDEAS.

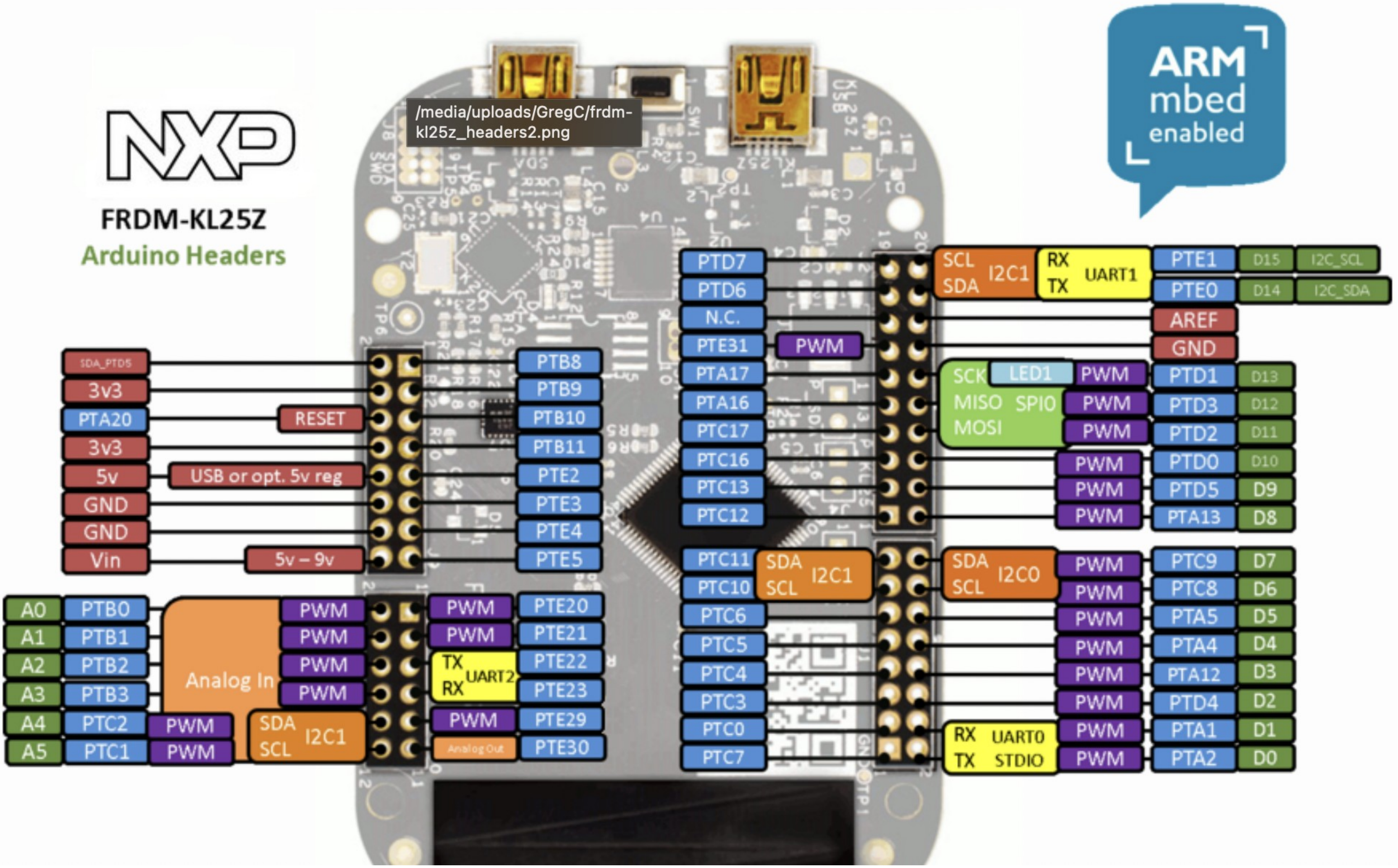
Idea's progress

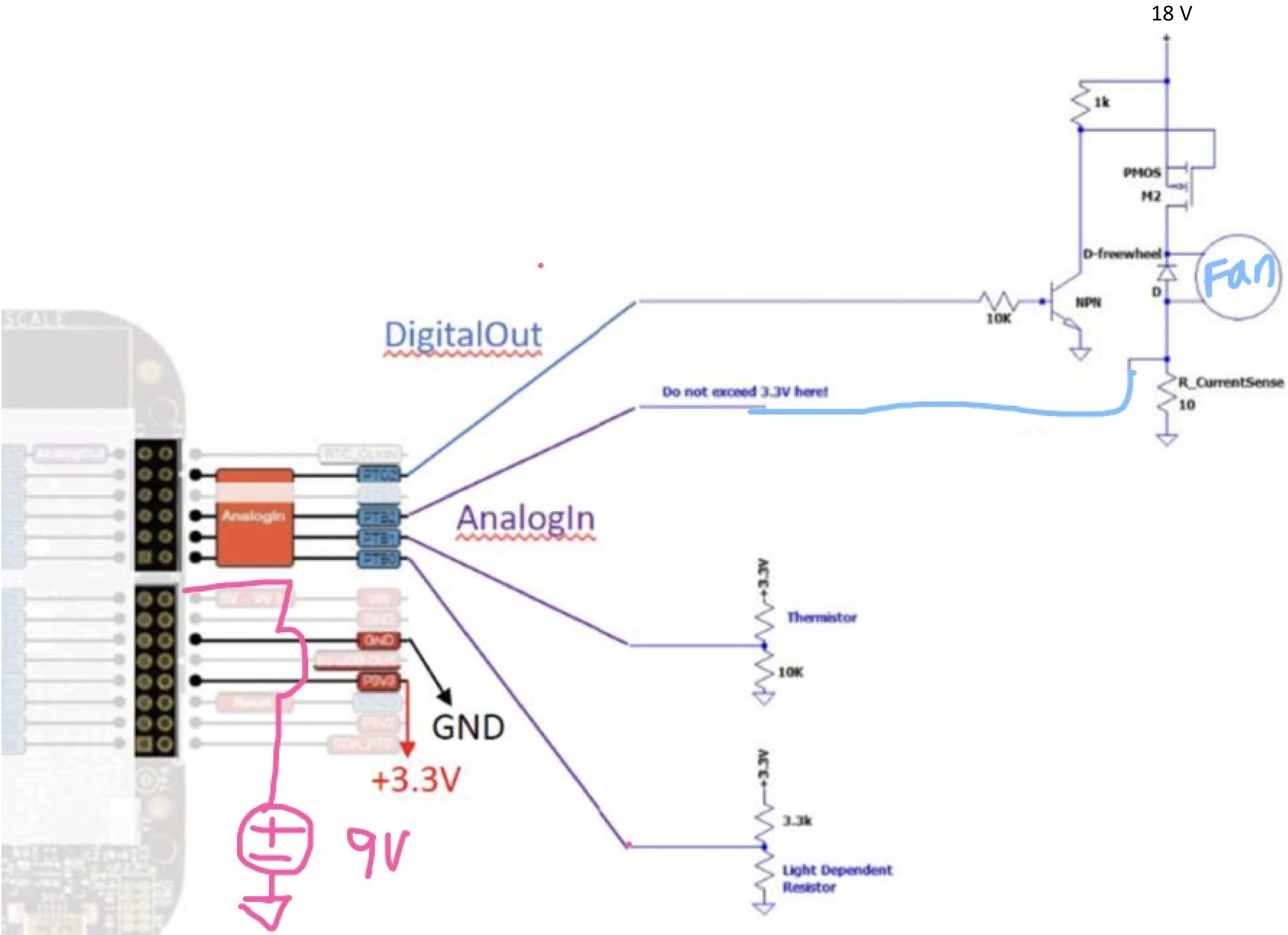
1. Finding the Idea
2. Designing the circuit
3. Building the circuit
4. Adding the Fan
5. Writing the Code



Circuit

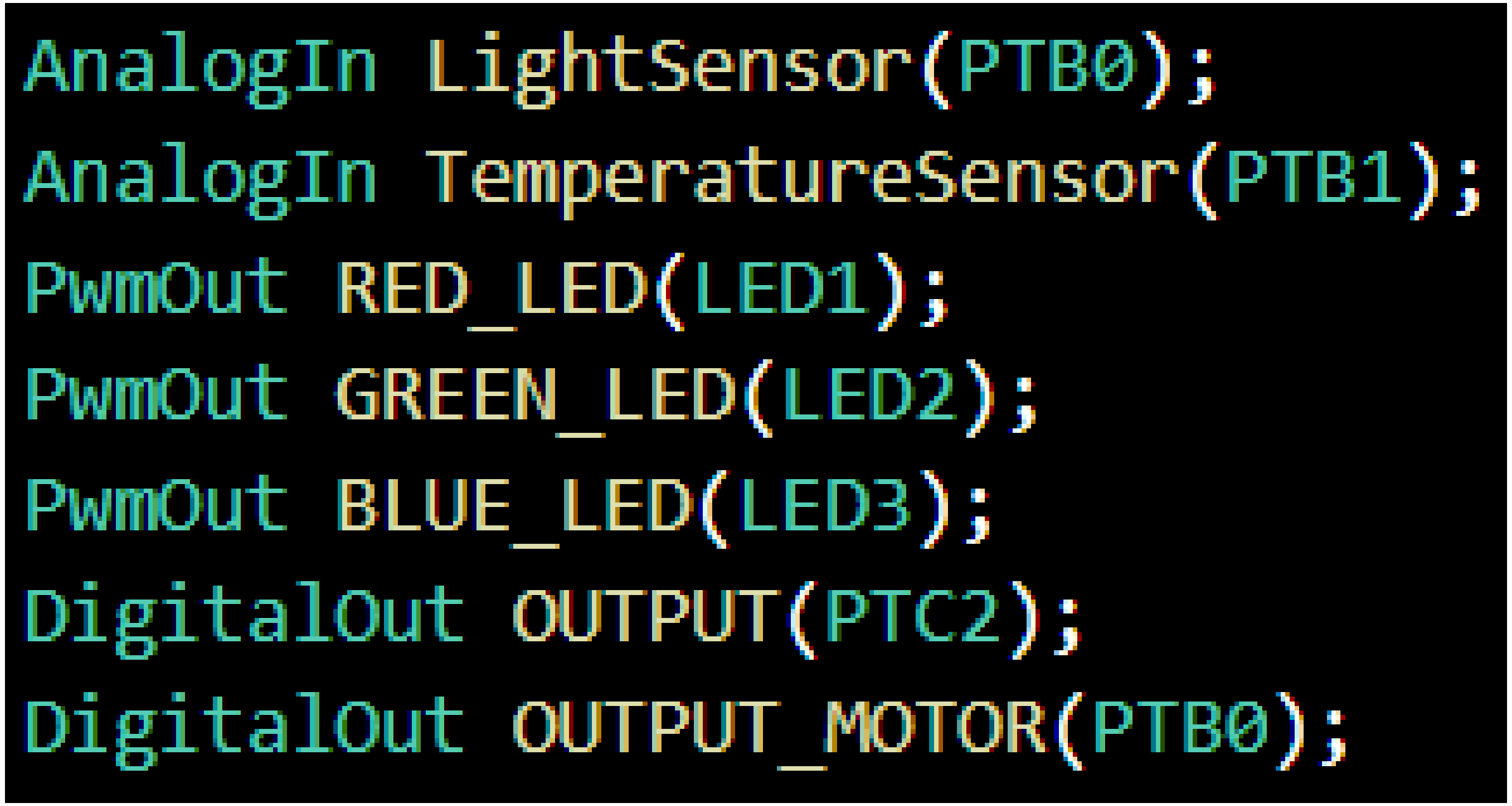



NXP
FRDM-KL25Z
Arduino Headers
ARM mbed enabled
/media/uploads/GregC/frdm-kl25z_headers2.png
SDA_PTD5
3v3
PTA20
RESET
3v3
5v
USB or opt. 5v reg
GND
GND
Vin
5v – 9v
PTB8
PTB9
PTB10
PTB11
PTE2
PTE3
PTE4
PTE5
A0 PTB0 PWM
A1 PTB1 PWM
A2 PTB2 PWM
A3 PTB3 PWM
A4 PTC2 PWM
A5 PTC1 PWM
Analog In
SDA SCL I2C1
PWM PTE20
PWM PTE21
TX RX UART2
PTE22
PTE23
PWM PTE29
Analog Out PTE30
PTD7
PTD6
N.C.
PTE31 PWM
PTA17
PTA16
PTC17
PTC16
PTC13
PTC12
PTC11
PTC10
SDA SCL I2C1
PTC6
PTC5
PTC4
PTC3
PTC0
PTC7
SCL SDA I2C1
RX TX UART1
PTE1 D15 I2C_SCL
PTE0 D14 I2C_SDA
AREF
GND
SCK LED1 PWM PTD1 D13
MISO SPI0 PWM PTD3 D12
MOSI PWM PTD2 D11
PWM PTD0 D10
PWM PTD5 D9
PWM PTA13 D8
SDA SCL I2C0
PWM PTC9 D7
PWM PTC8 D6
PWM PTA5 D5
PWM PTA4 D4
PWM PTA12 D3
PWM PTD4 D2
RX UART0 PWM PTA1 D1
TX STDIO PWM PTA2 D0


18 V
1k
PMOS
M2
D-freewheel
Fan
D
NPN
10K
R_CurrentSense
10
DigitalOut
Do not exceed 3.3V here!
AnalogIn
Analogin
PTC2
PTB2
PTB1
PTB0
GND
P3V3
GND
+3.3V
9V
+3.3V
Thermistor
10K
+3.3V
3.3k
Light Dependent Resistor

Code

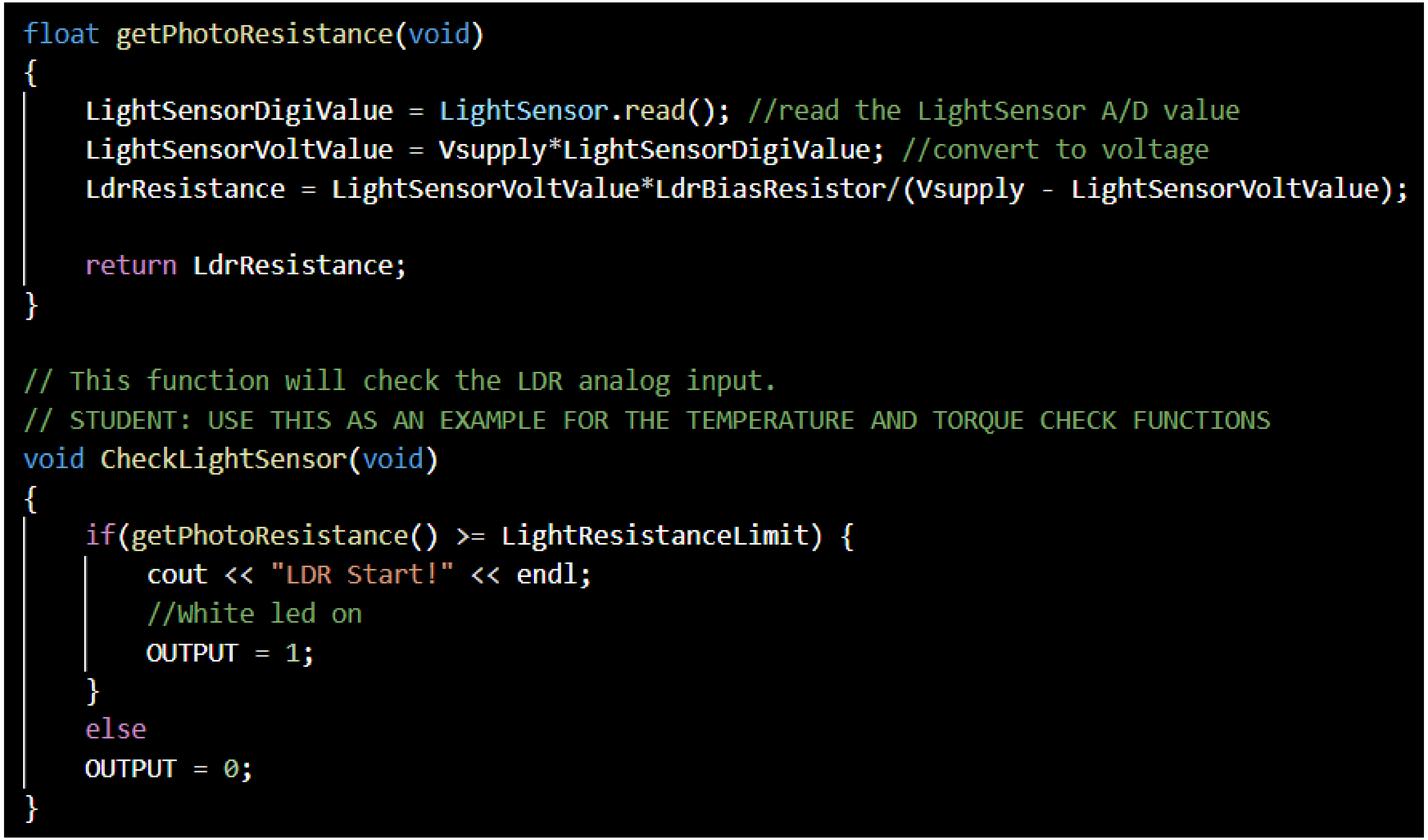
Inputs/Outputs



```
AnalogIn LightSensor(PTB0);
AnalogIn TemperatureSensor(PTB1);
PwmOut RED_LED(LED1);
PwmOut GREEN_LED(LED2);
PwmOut BLUE_LED(LED3);
DigitalOut OUTPUT(PTC2);
DigitalOut OUTPUT_MOTOR(PTB0);
```

Code

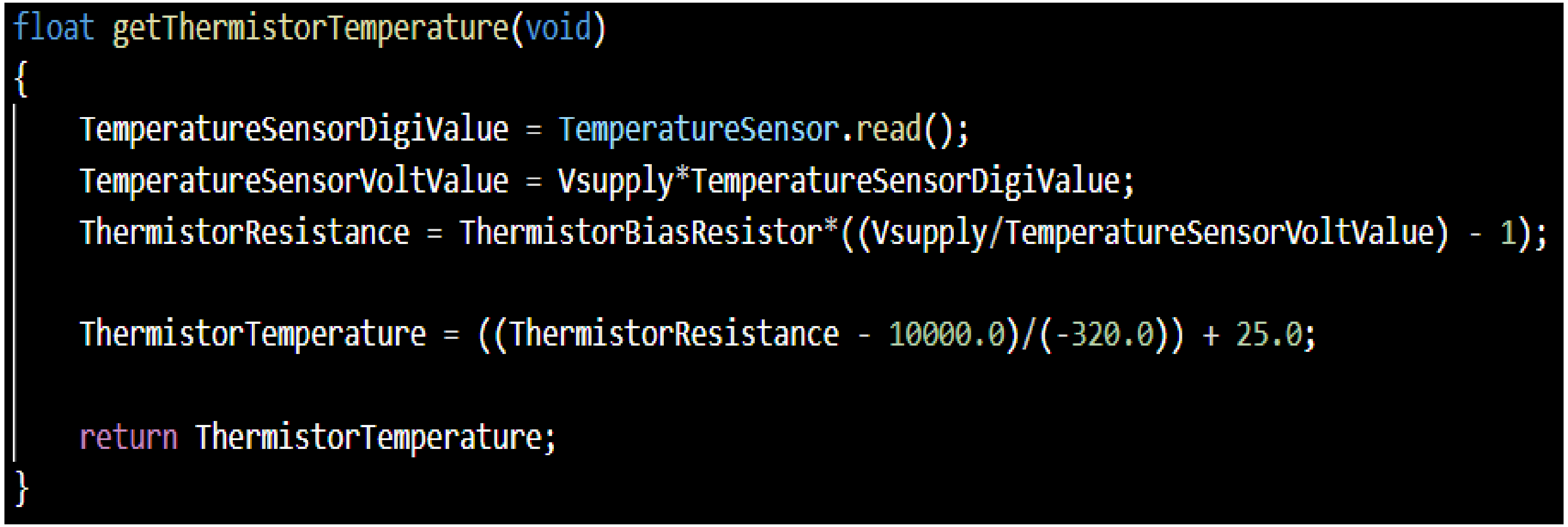
Photo Resistance function



```
float getPhotoResistance(void)
{
    LightSensorDigiValue = LightSensor.read(); //read the LightSensor A/D value
    LightSensorVoltValue = Vsupply*LightSensorDigiValue; //convert to voltage
    LdrResistance = LightSensorVoltValue*LdrBiasResistor/(Vsupply - LightSensorVoltValue);

    return LdrResistance;
}

// This function will check the LDR analog input.
// STUDENT: USE THIS AS AN EXAMPLE FOR THE TEMPERATURE AND TORQUE CHECK FUNCTIONS
void CheckLightSensor(void)
{
    if(getPhotoResistance() >= LightResistanceLimit) {
        cout << "LDR Start!" << endl;
        //White led on
        OUTPUT = 1;
    }
    else
    OUTPUT = 0;
}
```

Thermistor temperature function

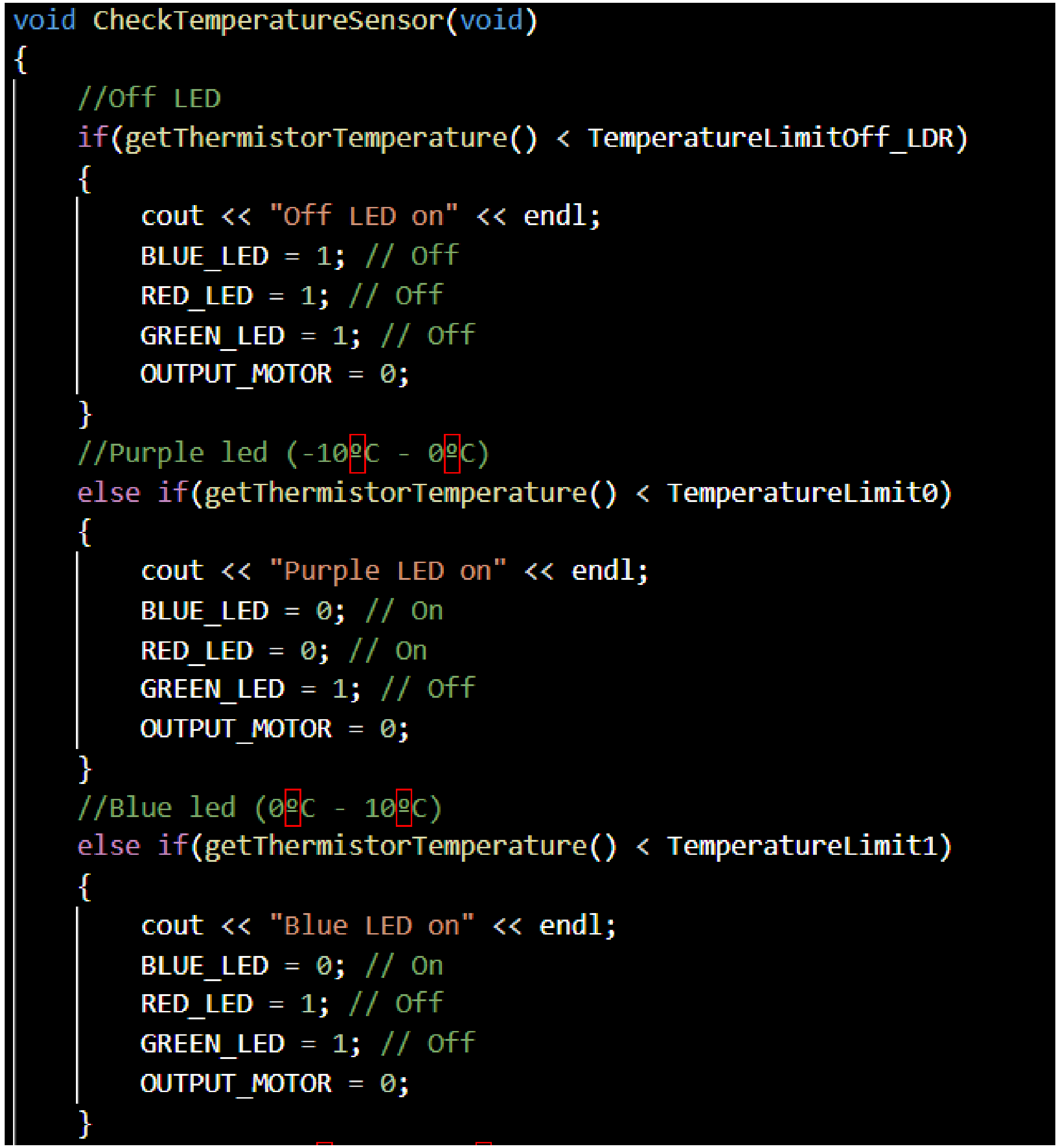
Code



```
float getThermistorTemperature(void)
{
    TemperatureSensorDigiValue = TemperatureSensor.read();
    TemperatureSensorVoltValue = Vsupply*TemperatureSensorDigiValue;
    ThermistorResistance = ThermistorBiasResistor*((Vsupply/TemperatureSensorVoltValue) - 1);

    ThermistorTemperature = ((ThermistorResistance - 10000.0)/(-320.0)) + 25.0;

    return ThermistorTemperature;
}
```

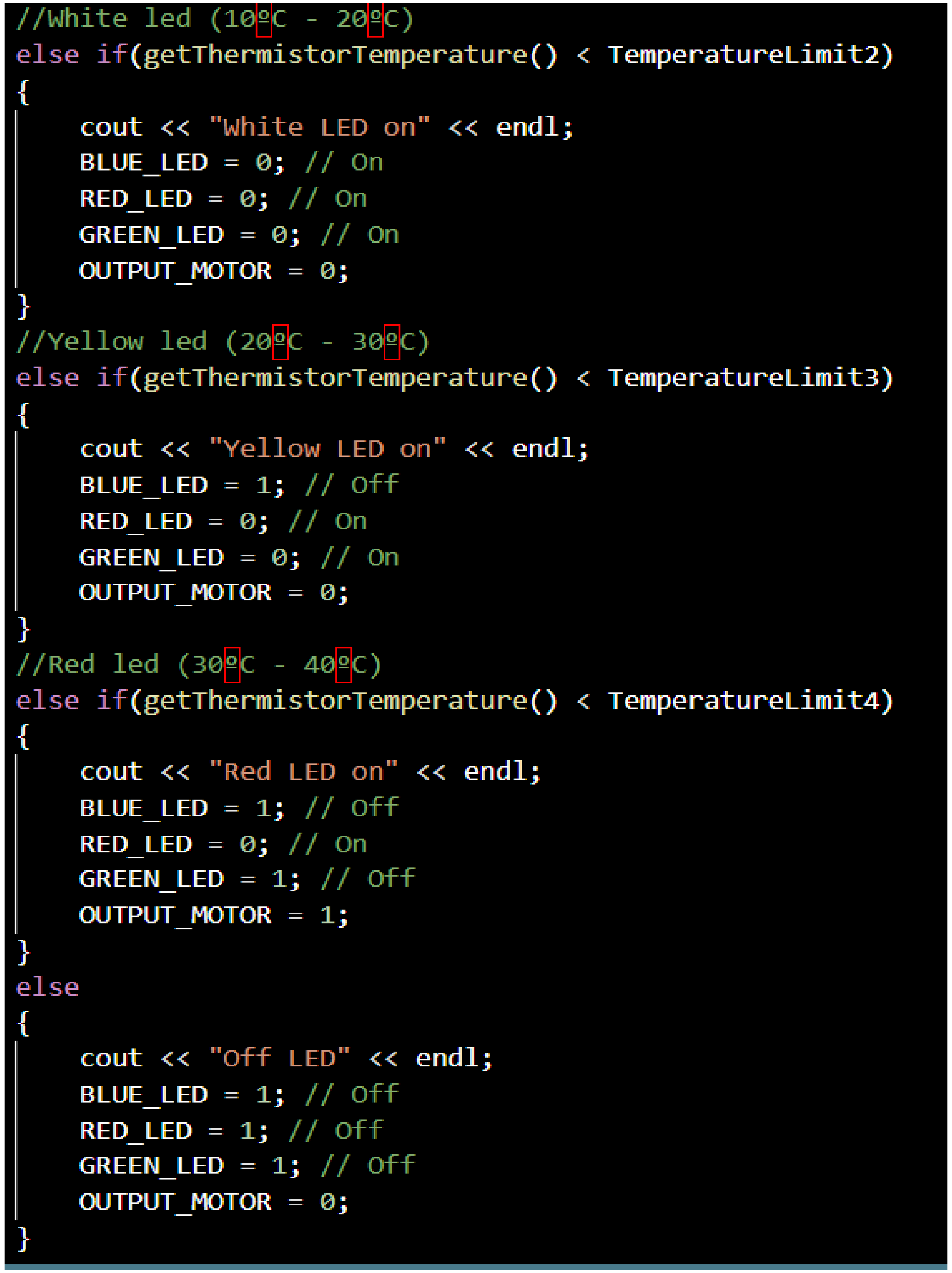
Code

Temperature threshold



```
void CheckTemperatureSensor(void)
{
    //Off LED
    if(getThermistorTemperature() < TemperatureLimitOff_LDR)
    {
        cout << "Off LED on" << endl;
        BLUE_LED = 1; // Off
        RED_LED = 1; // Off
        GREEN_LED = 1; // Off
        OUTPUT_MOTOR = 0;
    }
    //Purple led (-10ºC - 0ºC)
    else if(getThermistorTemperature() < TemperatureLimit0)
    {
        cout << "Purple LED on" << endl;
        BLUE_LED = 0; // On
        RED_LED = 0; // On
        GREEN_LED = 1; // Off
        OUTPUT_MOTOR = 0;
    }
    //Blue led (0ºC - 10ºC)
    else if(getThermistorTemperature() < TemperatureLimit1)
    {
        cout << "Blue LED on" << endl;
        BLUE_LED = 0; // On
        RED_LED = 1; // Off
        GREEN_LED = 1; // Off
        OUTPUT_MOTOR = 0;
    }
```

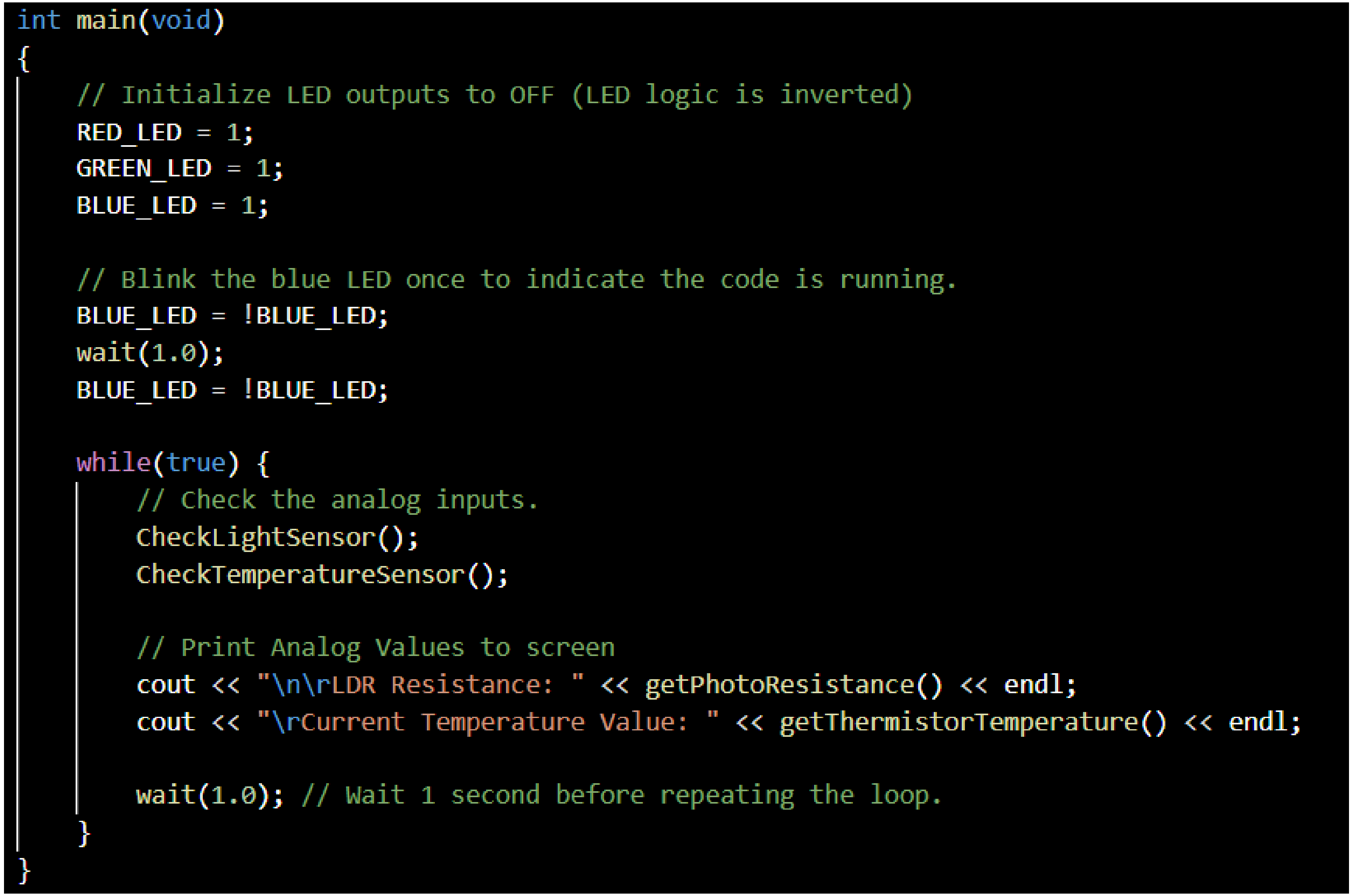
Code



```
//White led (10ºC - 20ºC)
else if(getThermistorTemperature() < TemperatureLimit2)
{
    cout << "White LED on" << endl;
    BLUE_LED = 0; // On
    RED_LED = 0; // On
    GREEN_LED = 0; // On
    OUTPUT_MOTOR = 0;
}
//Yellow led (20ºC - 30ºC)
else if(getThermistorTemperature() < TemperatureLimit3)
{
    cout << "Yellow LED on" << endl;
    BLUE_LED = 1; // Off
    RED_LED = 0; // On
    GREEN_LED = 0; // On
    OUTPUT_MOTOR = 0;
}
//Red led (30ºC - 40ºC)
else if(getThermistorTemperature() < TemperatureLimit4)
{
    cout << "Red LED on" << endl;
    BLUE_LED = 1; // Off
    RED_LED = 0; // On
    GREEN_LED = 1; // Off
    OUTPUT_MOTOR = 1;
}
else
{
    cout << "Off LED" << endl;
    BLUE_LED = 1; // Off
    RED_LED = 1; // Off
    GREEN_LED = 1; // Off
    OUTPUT_MOTOR = 0;
}
```

Code

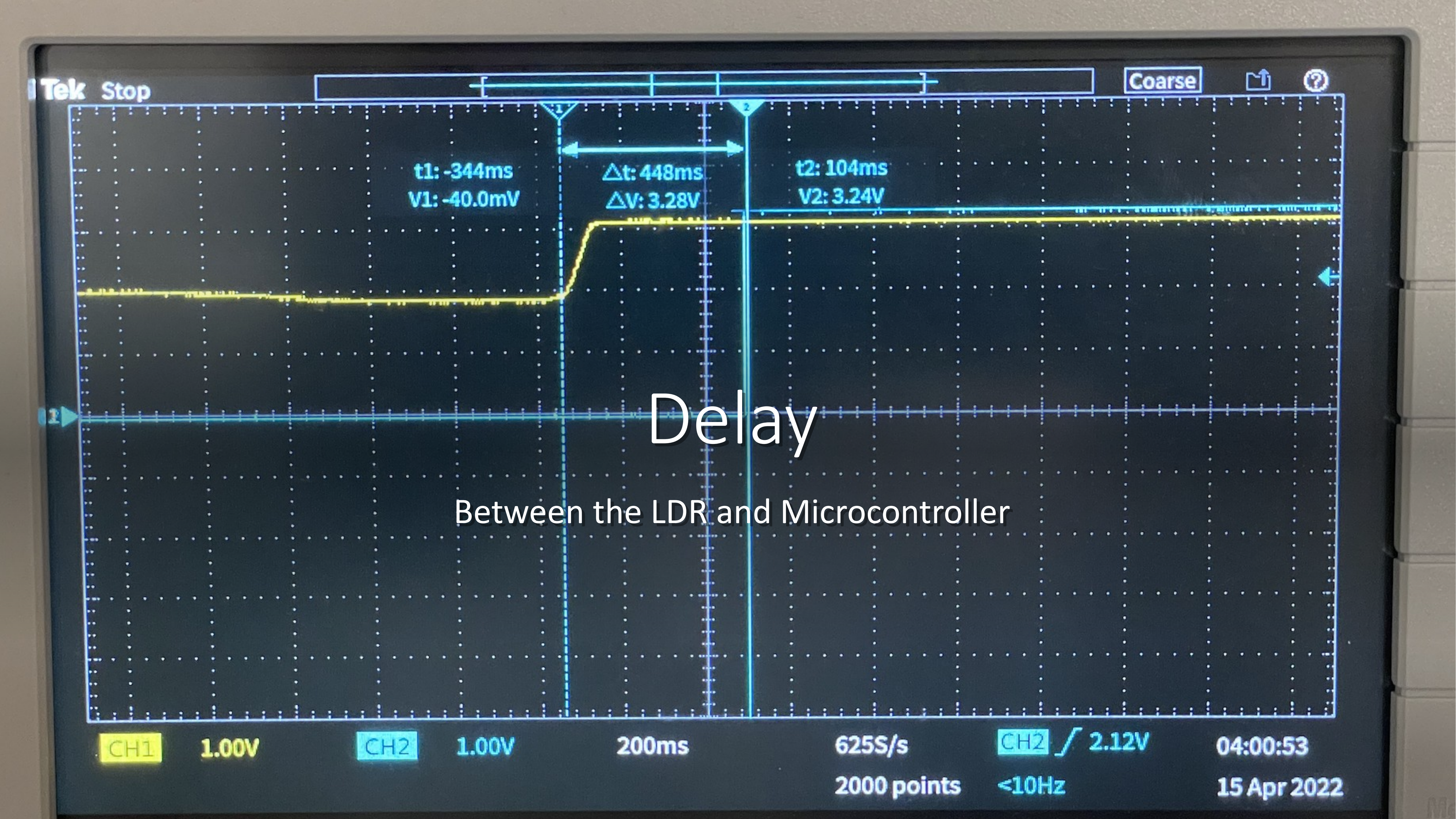
Main



```cpp
int main(void)
{
    // Initialize LED outputs to OFF (LED logic is inverted)
    RED_LED = 1;
    GREEN_LED = 1;
    BLUE_LED = 1;

    // Blink the blue LED once to indicate the code is running.
    BLUE_LED = !BLUE_LED;
    wait(1.0);
    BLUE_LED = !BLUE_LED;

    while(true) {
        // Check the analog inputs.
        CheckLightSensor();
        CheckTemperatureSensor();

        // Print Analog Values to screen
        cout << "\n\rLDR Resistance: " << getPhotoResistance() << endl;
        cout << "\rCurrent Temperature Value: " << getThermistorTemperature() << endl;

        wait(1.0); // Wait 1 second before repeating the loop.
    }
}
```

Delay

Between the LDR and Microcontroller



Uses

In your yard or deck

Green Houses and farms

Warehouses

Cost

- One: $30
- Over 20: $26
- Over 50: $22.50

Problems encountered

Adjusting the values of the temperature thresholds to easily show all the light functions (-10ºC to 40ºC).

We could not make the orange color by combining colors and we had to pick white instead.

Problems with the PTC1 output for the motor, it would not work properly so we had to choose a different pin (PTB0).

Two 9 V batteries in series (18V), and we needed to only use from 5-9 V to feed the microcontroller, but 18 Volts for the fan.

ANY
QUESTIONS
?