IoT Implementation for a Harsh Temperature Monitoring System

Advisor

Wilfredo Moreno
Mia Naeini
Kevin D. Wolf

Team

Brandon Collins
Steve Lambropoulos
Paul Polgar
Ashley Porter

Problem Statement

The IoT Implementation for Harsh Temperature Monitoring System is an innovative system that will allow companies to maintain and improve the longevity of their operational equipment, such as concrete crushers. Our product will monitor and alert onsite personnel if components on equipment is operating outside of their desired temperature range by using a temperature sensor to monitor the temperature of the equipment, send this data back to a server to be analyzed and provide real time measurements for operators. In addition, our system:

- Maintains the longevity of a company's large operational equipment by ensuring the equipment is operating at a safe temperature
- Sends alerts back to a server if operational temperature fluctuates out of range
- Alerts onsite personnel of unsafe temperature conditions of operating equipment
- Ultimately serves to eliminate a potentially hazardous situation, while protecting the quality and durability of a company's equipment

Design Approach

- ✓ Initial configuration of device using Bluetooth from the project manager's phone/laptop.
- ✓ Batteries will have a minimum 6-month lifetime
- ✓ The device will be contained within a durable box to withstand sustained exterior temperatures of 150F and mechanical shocks of 3g.
- ✓ A LED will be placed within the device that will be visible for on-site personnel to alert them of dangerous conditions with respect to temperatures being out of range.
- ✓ The temperature sensor will read the internal temperature of the concrete crusher and relay this data back to the server at a user defined interval, with a minimum time of at least 30 seconds.
- ✓ The device requires to be surface mounted to the surface of interest and will remain there for the duration of the job or at least 6 months. A low battery alert will be given via LED.
- ✓ The data collected by the device will be sent to a server to be analyzed.

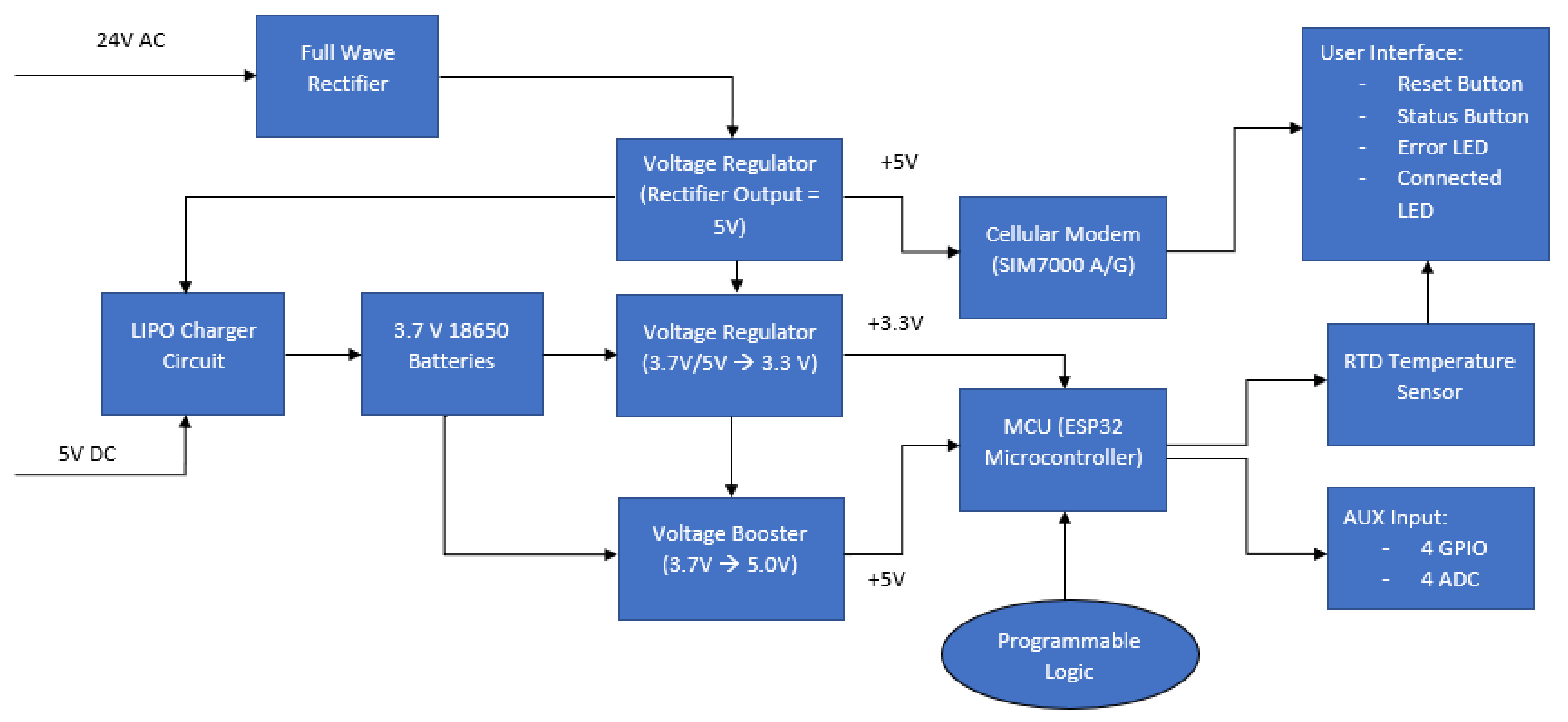
Design Approach (continued)

- ✓ Pre-configured ranges for acceptable temperatures within the concrete crusher will be programmed into the device

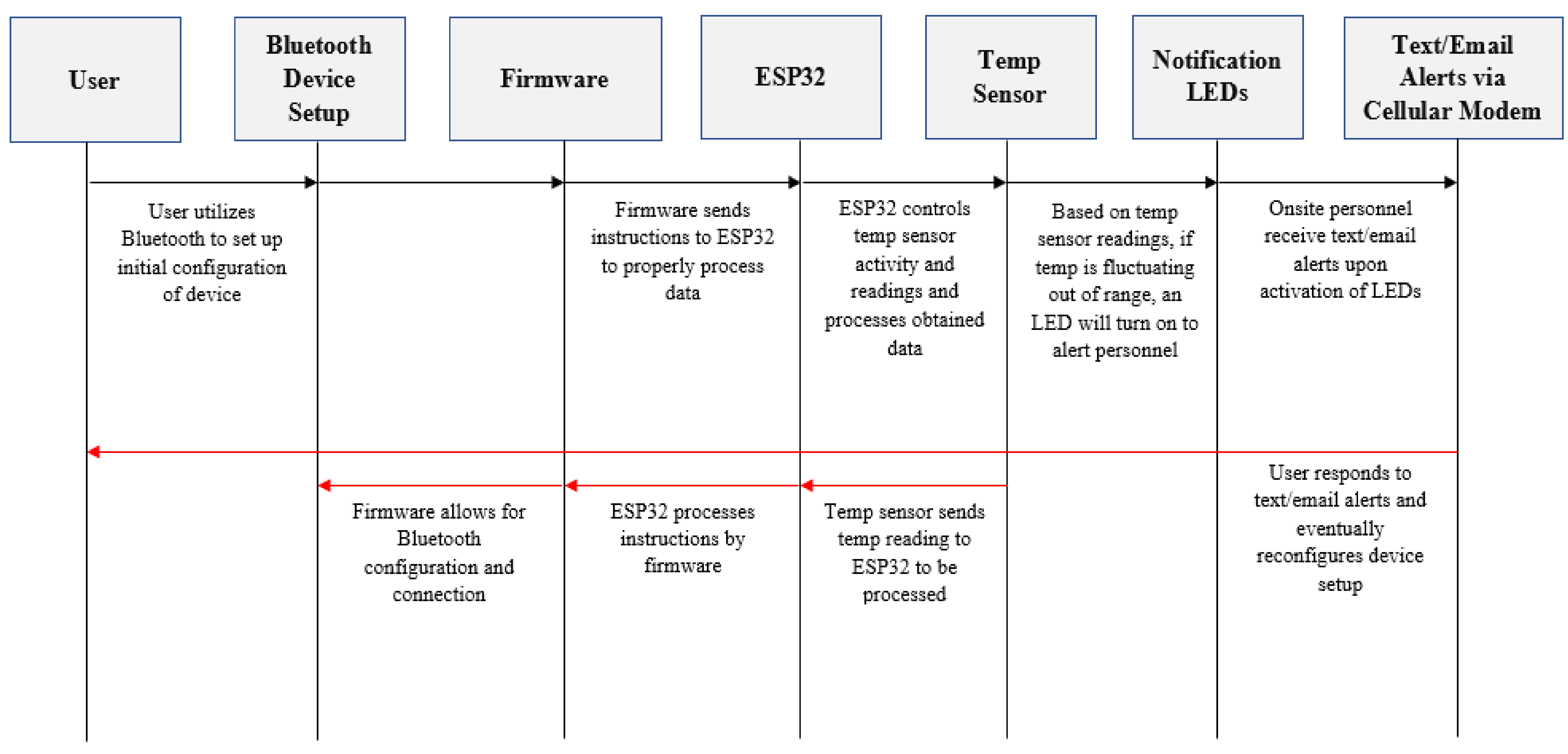
  - If the temperature fluctuates out of this range, the sensor will sense this change, relay the data back to the server, and an alert will be sent out. the accuracy of the measurement would need to be within +/- 2 degrees.

- ✓ A text/email alert will be sent to relevant personnel to notify them of the condition.
- ✓ An additional LED visual alert on the device will also notify relevant personnel of the condition, in case of a missed text/email.

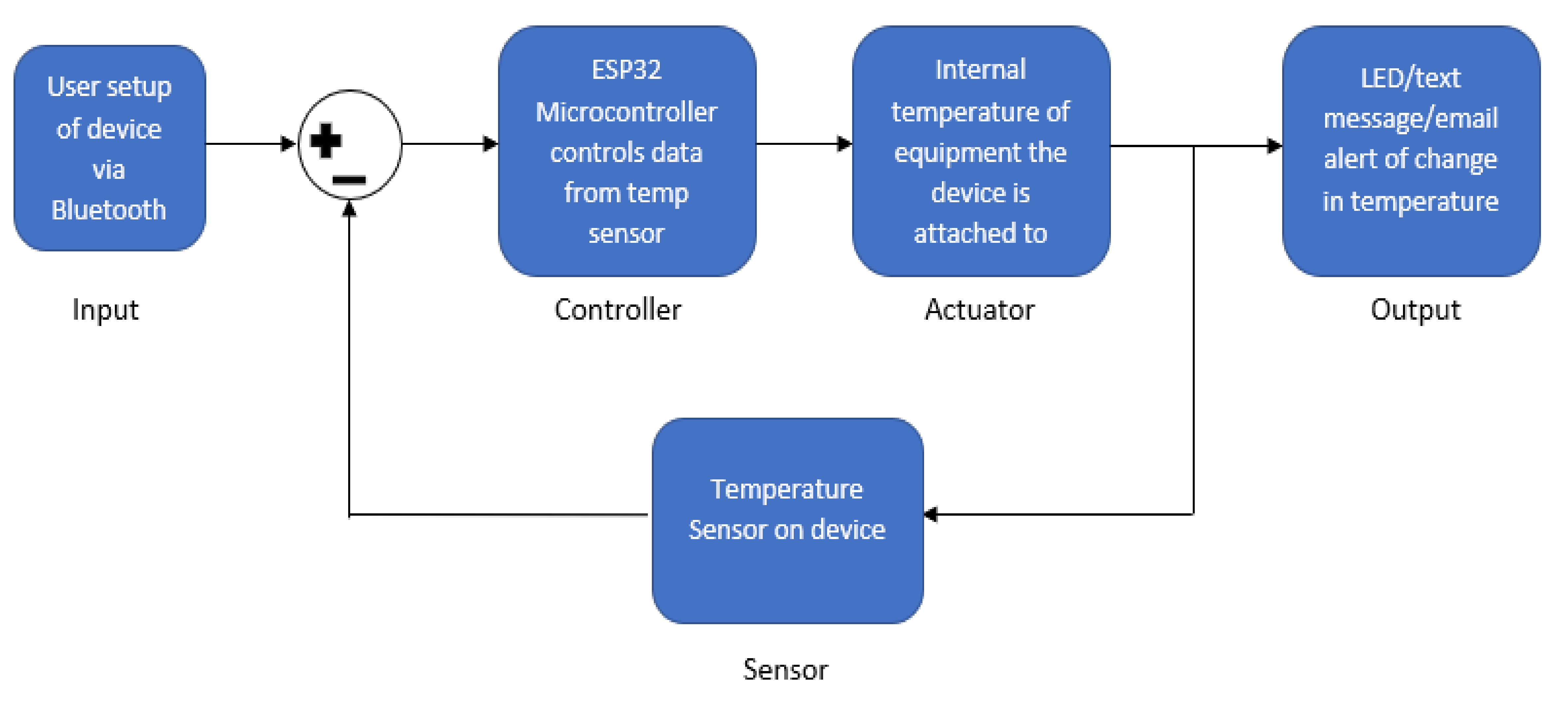
System Block Diagram

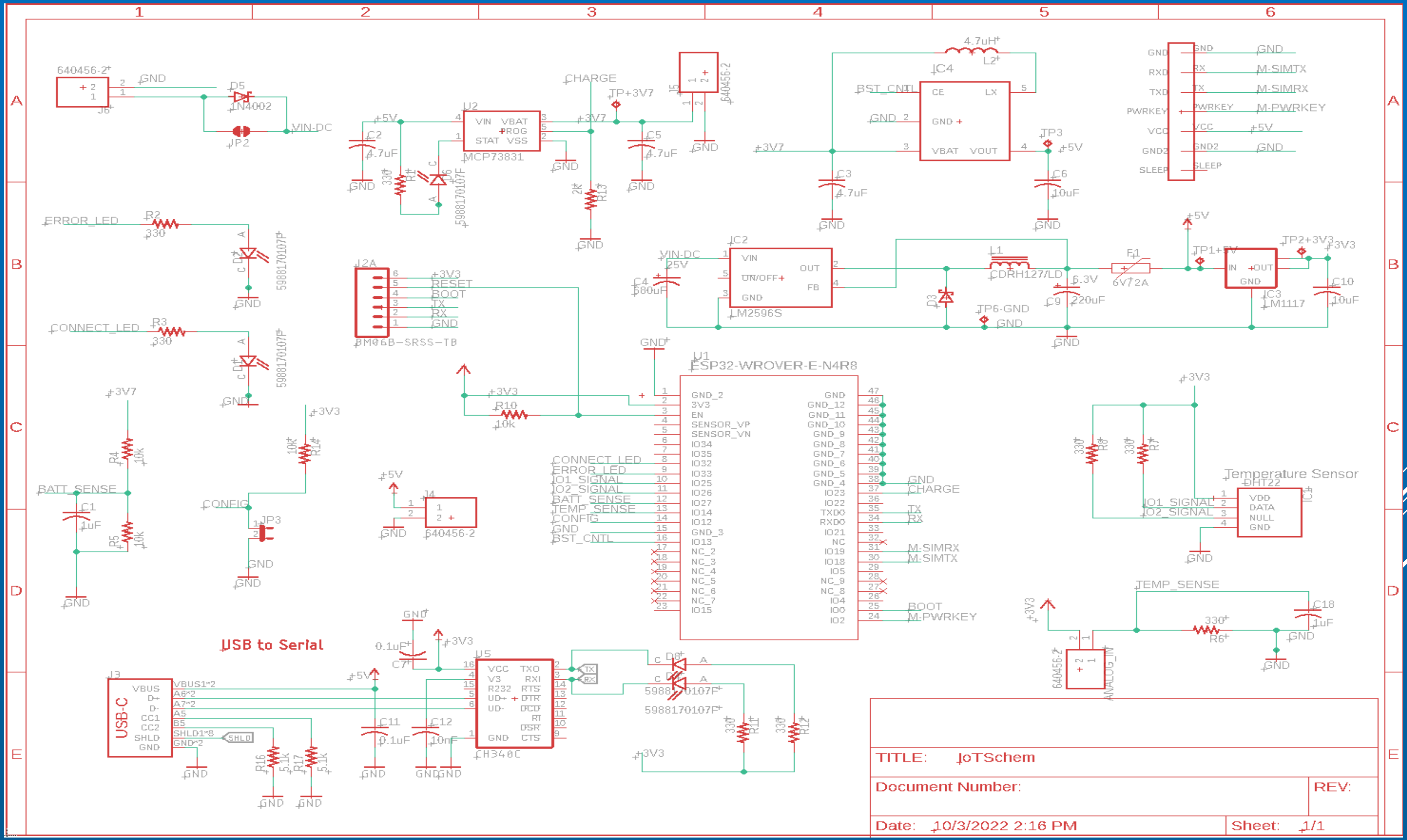


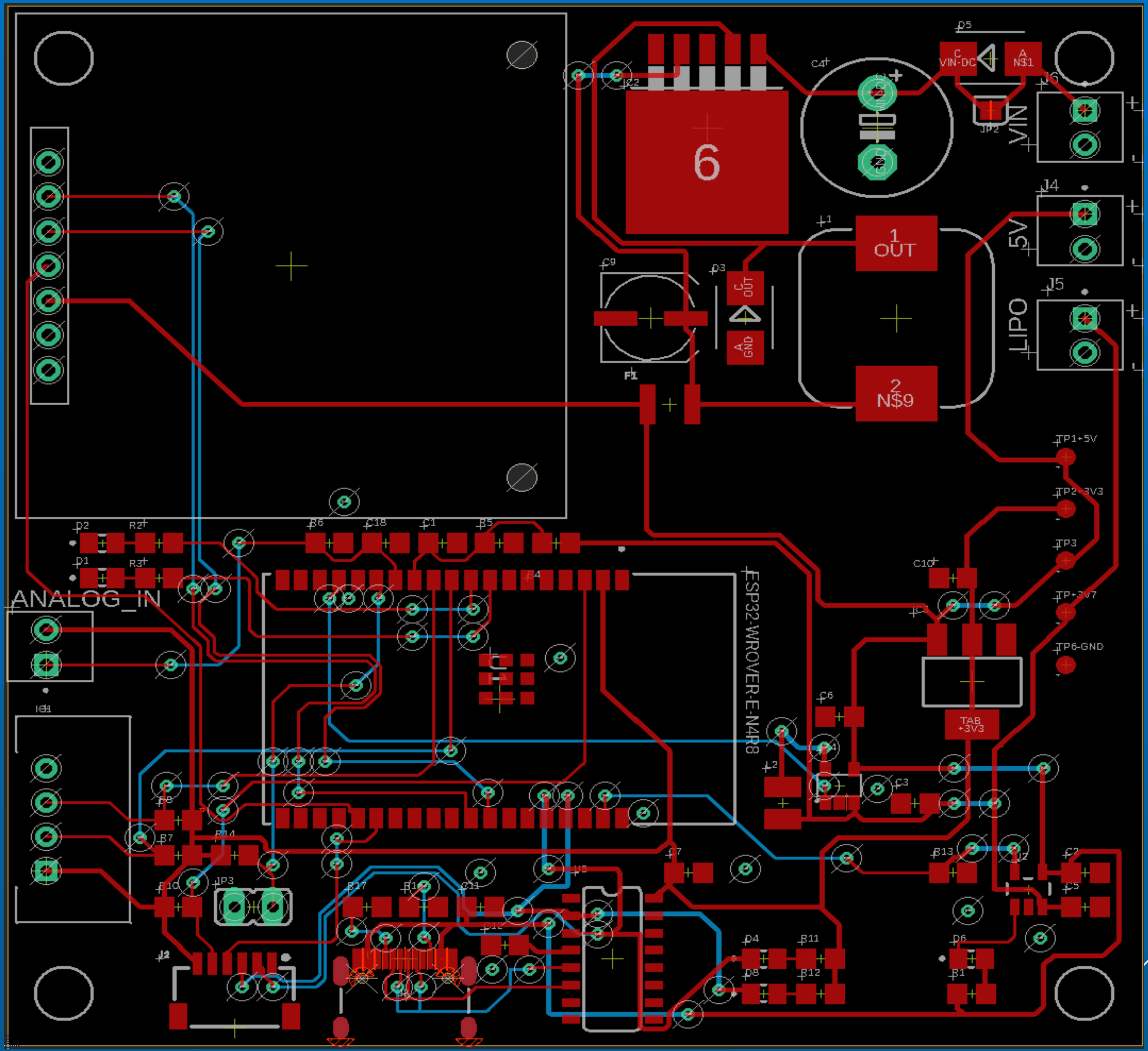
Sequence Diagram



System Block Diagram




USB to Serial
ESP32-WROVER-E-N4R8
MCP73831
LM2596S
LM1117
CH340C
USB-C
BM06B-SRSS-TB
Temperature Sensor
DHT22
VIN-DC
CHARGE
ERROR_LED
CONNECT_LED
BATT_SENSE
CONFIG
TEMP_SENSE
BST_CNTL
M-SIMTX
M-SIMRX
M-PWRKEY
ANALOG_IN
TITLE: IoTSchem
Document Number:
REV:
Date: 10/3/2022 2:16 PM
Sheet: 1/1


ANALOG_IN
ESP32-WROVER-E-N4R8
VIN
5V
LIPO
1 OUT
2 N$9
TP1-5V
TP6-GND
TAB +3V3
VIN-DC

Part	Value	Device	Description	DIGI-KEY_PART_NUMBER
ANALOG_IN	640456-2	640456-2	220P MTA100 HDR ASSY SQ SPCL	A1921-ND
C1	1uF	C-USC0805	CAPACITOR, American symbol	311-1365-1-ND
C2	4.7uF	C-USC0805	CAPACITOR, American symbol	1276-6722-1-ND
C3	4.7uF	C-USC0805	CAPACITOR, American symbol	1276-6722-1-ND
C4	680uF	CPOL-USE5-10.5	POLARIZED CAPACITOR, American symbol	
C5	4.7uF	C-USC0805	CAPACITOR, American symbol	1276-6722-1-ND
C6	10uF	C-USC0805	CAPACITOR, American symbol	
C7	0.1uF	C-USC0805	CAPACITOR, American symbol	311-1140-1-ND
C9	220uF	CPOL-USUD-6,3X5,8	POLARIZED CAPACITOR, American symbol	
C10	10uF	C-USC0805	CAPACITOR, American symbol	399-C0805C106K8PAC7800CT-
C11	0.1uF	C-USC0805	CAPACITOR, American symbol	311-1140-1-ND
C12	10nF	C-USC0805	CAPACITOR, American symbol	1276-1015-1-ND
C18	1uF	C-USC0805	CAPACITOR, American symbol	311-1365-1-ND
D1	5988170107F	5988170107F	Standard LEDs - SMD Green Water Clr 40mcd 570nm	350-2044-1-ND
D2	5988170107F	5988170107F	Standard LEDs - SMD Green Water Clr 40mcd 570nm	350-2044-1-ND
D3		SCHOTTKY-DIODESMD	Schottky Diode	
D4	5988170107F	5988170107F	Standard LEDs - SMD Green Water Clr 40mcd 570nm	350-2044-1-ND
D5	1N4002	SCHOTTKY-DIODESMD	Schottky Diode	
D6	5988170107F	5988170107F	Standard LEDs - SMD Green Water Clr 40mcd 570nm	350-2044-1-ND
D8	5988170107F	5988170107F	Standard LEDs - SMD Green Water Clr 40mcd 570nm	350-2044-1-ND
F1	6V/2A	PPTC_6V2A	Resettable Fuse PPTC	
IC1	DHT22	DHT22	Digital-output relative humidity & temperature sensor/module DHT22	1528-1504-ND
IC2	LM2596S	LM2596S	SIMPLE SWITCHER® Power Converter 150 kHz 3A Step-Down Voltage Regulator	2156-LM2596SX-5.0/NOPB-ND
IC3		LM1117MPX-3.3		LM1117IMP-3.3/NOPBCT-ND
IC4		XC9140		893-1181-1-ND
J1	SIM7000A	SIM7000A		
J2	BM06B-SRSS-TB	JST_6_PIN_HORIZONTAL	JST 6 pin horizontal connector	
J3		USB_C16PIN	USB Type C 16Pin Connector	
J4	640456-2	640456-2	220P MTA100 HDR ASSY SQ SPCL	A1921-ND
J5	640456-2	640456-2	220P MTA100 HDR ASSY SQ SPCL	A1921-ND
J6	640456-2	640456-2	220P MTA100 HDR ASSY SQ SPCL	A1921-ND
JP2	JUMPER-SMT_2_NO_SILK	JUMPER-SMT_2_NO_SILK	Normally open jumper	
JP3		JP1E	JUMPER	
L1	CDRH127/LD	CDRH127/LD	Power Inductor	
L2	4.7uH	L-USL2825P	INDUCTOR, American symbol	
R1	330	R-US_R0805	RESISTOR, American symbol	
R2	330	R-US_R0805	RESISTOR, American symbol	RMCF0805JT330RCT-ND
R3	330	R-US_R0805	RESISTOR, American symbol	RMCF0805JT330RCT-ND
R4	10k	R-US_R0805	RESISTOR, American symbol	RMCF0805FT2K00CT-ND
R5	10k	R-US_R0805	RESISTOR, American symbol	RMCF0805FT2K00CT-ND
R6	330	R-US_R0805	RESISTOR, American symbol	RMCF0805JT330RCT-ND
R7	330	R-US_R0805	RESISTOR, American symbol	RMCF0805JT330RCT-ND
R8	330	R-US_R0805	RESISTOR, American symbol	RMCF0805JT330RCT-ND
R10	10k	R-US_R0805	RESISTOR, American symbol	RMCF0805FT2K00CT-ND
R11	330	R-US_R0805	RESISTOR, American symbol	RMCF0805JT330RCT-ND
R12	330	R-US_R0805	RESISTOR, American symbol	RMCF0805JT330RCT-ND
R13	2k	R-US_R0805	RESISTOR, American symbol	RMCF0805FT2K00CT-ND
R14	10k	R-US_R0805	RESISTOR, American symbol	RMCF0805FT2K00CT-ND
R16	5.1k	R-US_R0805	RESISTOR, American symbol	RMCF0805JT5K10CT-ND
R17	5.1k	R-US_R0805	RESISTOR, American symbol	RMCF0805JT5K10CT-ND
TP+3V7	TPB1,27	TPB1,27	Test pad	
TP1+5V	TPB1,27	TPB1,27	Test pad	
TP2+3V3	TPB1,27	TPB1,27	Test pad	
TP3	TPB1,27	TPB1,27	Test pad	
TP6-GND	TPB1,27	TPB1,27	Test pad	
U1	ESP32-WROVER-E-N4R8	ESP32-WROVER-E-N4R8		
U2	MCP73831	MCP73831	Miniature single cell, fully integrated Li-Ion, Li-polymer charge management contro	MCP73831T-2DCI/OTCT-ND
U5	CH340C	CH340C		

Simulations and Calculations Performed

	Component	Currrent (A)	Voltage (V)	Power (W)	Size (mm)	Weight (g)
1	Standard LEDs - SMD Green Water	0.03	5	0.015	2x1.25x0.7	1
2	DHT22 Temperature Sensor	0.021	5	0.1155	15.1x7.7x25.1	1
3	Power Converter 150 kHz 3A Step-Down Voltage Regulator	7.5	45	337.5	10.17x8.69x4.55	100
4	Low-Dropout Linear Regulator	0.8	20	16	6.5x3.5	≈ 28
5	Step-UpSynchronous PFM DC/DC Converter	0.295	5	1.475	40x40	≈ 28
6	SIM7000A Module	0.011	4.3	0.0473	24 X 24 X 2.6mm	3
7	USB Type C 16Pin Connector	3	20	60	8.9x6.5x2.56	≈ 28
8	ESP32 WROVER Microcontroller	1.1	3.6	3.96	31.4x18x3.3	7.74
9	Power Supply	5	24	120		
10	Charger IC Lithium Ion/Polymer	0.5	6	3	3.1x3x1.3	≈ 28
			TOTAL:	542.1128		112.74

- Power was calculated using data obtained from datasheets of components
- Size and weight was obtained from datasheet

Hardware Development to Date

- Ordered PCB Board on 10/3
- Ordered components on 10/3
- In process of prototyping the enclosure for device using CAD (will be 3D printed)

Once all components and PCB board come in, we will begin developing the board and attaching components.

The device enclosure will be designed to meet exact measurements of the board will be developed to accommodate any special features.

Software Development to Date

```cpp
#define DHT_DEBUG

#define TEMP_SNSR_BOARD_V3

#define WIFIPW "password goes here..."

#include <TemperatureProbes.h>
#include <Arduino.h>
#include <NuvIoT.h>

#define TEMP_SNSR_SKU "RSB-01"
#define FIRMWARE_VERSION "0.7.3"
#define HARDWARE_REVISION "3.0"

#define BATT_SENSE_PIN 27
#define TEMP_SENSE_PIN 13

#define IO1_PIN 25
#define IO2_PIN 26

bool hasDHT22 = false;
bool hasProbe1 = false;
bool hasProbe2 = false;

NuvIoT_DHT *dht;
DallasTemperature *probe1;
DallasTemperature *probe2;

byte buffer[8];
bool running = true;

void determineSensorConfiguration()
{
  dht = new NuvIoT_DHT(IO1_PIN, DHT22, 6, &console);
  dht->begin();
  uint8_t retryCount = 0;
  while (retryCount++ < 5 && !hasDHT22)
  {
    float temp = dht->readTemperature(true,true);
    if (!isnan(temp)){
      hasDHT22 = true;
      console.println("Found DHT22");
    }
    else {
      console.println("Attempt " + String(retryCount));
      delay(1000);
    }
  }

  if (!hasDHT22){
    console.println("Did Not Find DHT22");
    delete dht;
    dht = NULL;
  }
  else{
    ioConfig.GPIO1Config = GPIO_CONFIG_DHT22;
    ioConfig.GPIO1Name = "Digital Temperature";
    ioConfig.GPIO1Scaler = 1;
    ioConfig.GPIO1Zero = 0;
    ioConfig.GPIO1Calibration = 1;

    ioConfig.GPIO2Config = GPIO_CONFIG_DHT22_HUMIDITY;
    ioConfig.GPIO2Name = "Digital Humidity";
    ioConfig.GPIO2Scaler = 1;
    ioConfig.GPIO2Zero = 0;
    ioConfig.GPIO2Calibration = 1;
  }

  if (dht == NULL){
    probe1 = new DallasTemperature(new OneWire(IO1_PIN));
    retryCount = 0;
    while (retryCount++ < 5 && !hasProbe1)
    {
      float temp = probe1->getTempFByIndex(0);
      if (!isnan(temp) && temp > -50.60f)
      {
```

- This code allows temperature sensor to relay data (i.e. temperature and humidity) to ESP32 microcontroller, which then interprets data and allows user to see results
- In addition, ESP32 is programmed to connect to WIFI so that it may be used to interface with user app

Software Development (continued)

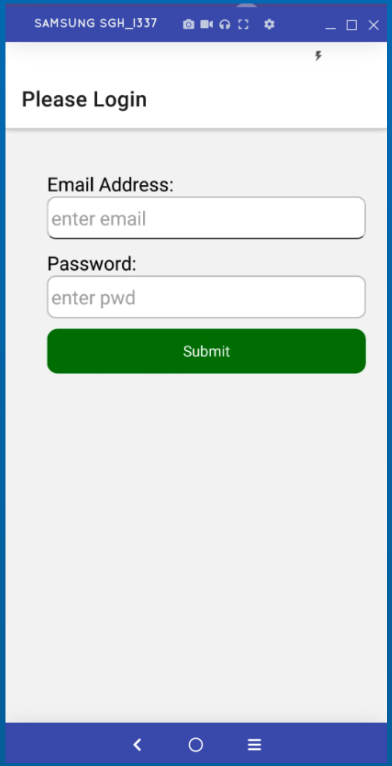
```
            console.println("actual probe 2 response " + String(temp));
            hasProbe1 = true;
        }
    }

    if (!hasProbe1){
        probe1 = NULL;
        console.println("Does not have DS18B Probe 1");
    }
    else{
        console.println("Has DS18B Probe 1");
        ioConfig.GPIO1Config = GPIO_CONFIG_DBS18;
        if (ioConfig.GPIO1Name == "")
            ioConfig.GPIO1Name = "Digital Temperature - Port 1";
        ioConfig.GPIO1Scaler = 1;
        ioConfig.GPIO1Zero = 0;
        ioConfig.GPIO1Calibration = 1;
    }

    probe2 = new DallasTemperature(new OneWire(IO2_PIN));
    retryCount = 0;
    while (retryCount++ < 5 && !hasProbe2)
    {
        float temp = probe2->getTempFByIndex(0);
        if (!isnan(temp) && temp > -50.60f)
```

This part of code is configuring probes on temp sensor to display temperature.

```
    configureConsole();
    writeConfigPins();
    determineSensorConfiguration();

    console.registerCallback(handleConsoleCommand);
    welcome(TEMP_SNSR_SKU, FIRMWARE_VERSION);

    String btName = "NuvIoT - " + (sysConfig.DeviceId == "" ? "Temp Sensor" : sysConfig.DeviceId);

    BT.begin(btName.c_str(), TEMP_SNSR_SKU);

    sysConfig.WiFiSSID = "Collins";
    sysConfig.WiFiPWD = WIFIPW;

    wifiMgr.setup();

    ledManager.setup(&ioConfig);
    ledManager.setOnlineFlashRate(1);
    ledManager.setErrFlashRate(0);

    probes.configure(&ioConfig);
}
```

This part of code allows the ESP to connect to WIFI using the WIFI ID and password which will allow for the interfacing of the microcontroller with user app.

Software Development (android app)



launch.json | auth.page.tsx

pages > auth.page.tsx > ...

```
const login = async (email: string, password: string) => {
    let request = {
        GrantType: 'password',
        AppInstanceId: 'ABC123',
        AppId: 'ABC1234',
        DeviceId: 'ABC123',
        ClientType: 'mobileapp',
        Email: email,
        Password: password,
        UserName: email
    }

    setIsBusy(true);

    fetch('https://api.nuviot.com/api/v1/auth',
        {
            method: 'POST',
            headers: {
                Accept: 'application/json',
                'Content-Type': 'application/json'
            },
            body: JSON.stringify(request)
        }).then(result => result.json())
        .then(async result => {
            setIsBusy(false);
            if(result.successful){
                await AsyncStorage.setItem("isLoggedIn", "true");

                await AsyncStorage.setItem("jwt", result.result.accessToken);
                await AsyncStorage.setItem("refreshtoken", result.result.refreshToken);
                await AsyncStorage.setItem("refreshtokenExpires", result.result.refreshTokenExpiresUTC);
                await AsyncStorage.setItem("jwtExpires", result.result.accessTokenExpiresUTC);
                navigation.replace('homePage')
            }
            else {
```

The react-native code as well as the c code/firmware was developed by the company we are working with. our involvement is to make it work with our design and adjust/add as needed. we are currently testing against a test setup that we have using an esp32 microcontroller and temperature sensor.

- App on your android and eventually ios device
- Initial configuration of device
- Developed in react-native

Testing Plan

- Battery Charging Requirements:
  - After a complete charge cycle, perform 3 type of discharges to gauge battery life:
    - Constant use/Constant upload = 1 upload / second
    - Realistic Upload rate of 1 upload / 10 mins
    - Idle
  - Time to failure will be measured by the time from initiation of test until failure to broadcast data or ESP32 shutdown.
- Temperature Test:
  - Device will be placed into a controlled temperature environment for 30 minutes and afterward tested to ensure integrity. A pass is defined as retaining complete capability with no material deformation.
    - 100F, 150F, 200F, 250F
    - 50F, 20F, 0F, -10F if possible
- Harsh environment
  - Test device with regards to abrasion, impact resistance, vibrations, and dust.

Work Division

Roles	Responsible Individual	TASK DESCRIPTION - Design I	TASK DESCRIPTION - Design II
MANAGEMENT			
Technical Leader	Collins	Understands the interconnections of the systems and works to ensure that each subsystem is compatible and fulfills objectives.	
Team Coordination Leader	Polgar	Ensures utilization of all team members, and aids in coordination between design aspects.	
Cost & Schedule leader	Porter	Responsible for coordinating between design aspects to ensure overall project remains in budget. Sets expected due dates for objectives based on class requirements.	
SYSTEM ENGINEERING			
SYST ENGINEERING	Polgar	PCB design expert. Will work to research PCB design and assemble all libraries to create a final PCB design.	
SYST ENGINEERING	Brandon	Coding and firmware system expert. Works to code the device from sensor to the communications module.	
SYST ENGINEERING	Steve	Communications expert, works to establish communications from the system to destination.	
SYST ENGINEERING	Porter	Sensor and Power expert. Will work to provide power for each subsystem and research and order an appropiate sensor for the design.	

TEST ENGINEERING	Steve	Runs analytics on final design, comparing final parameters to intentended.
PROD ENGINEERING	Brandon	Expert on final assembly and creating mounts for modules to PCB, PCB to enclosure, and enclosure to mounting location.
CONFIG MGMT	Brandon	Works to analyze design layout to minimize space requirements to minimize overall enclosure size.
RELIABILITY	Steve	Ensures that all chosen components meet requirements for required product lifetime.
QUALITY	Porter	Works to ensure components meet all required standards.
PARTS MANAGEMENT	Porter	Ensures that parts are selected and ordered in preparation for final assembly. Maintains documentation of all ordered parts.
EMI / EMC	Polgar	Analyzes the design and suggests improvements to minimize the effects of EMI.
SAFETY	Polgar	Ensures that during assembly all
Hardware		
ELECT. DESIGN		
DESIGN	Brandon	
MECH. DESIGN		
PSNA DESIGN	Steve	
SOFTWARE		
SW DEVELOPM	Brandon	
TEST Engineering		
Test	Steve	
TEST SOFTWARE		
DESIGN	Steve	

Hardware Demo