University of Colorado Boulder

Faculty of Electrical, Computer & Energy Engineering

ECEN5833

Written Document: Final Report

Date	2023/05/10
Team Name	Embedded Wizards
Prepared by (Name)	Shuran Xu
	Pradyumna Gudluru

Table of Contents

1.Project Overview

The use of wearable technology has more than tripled in the last four years, in accordance with consumers' increased interest in monitoring their own health and vital signs. According to the research by Precedence Research, the market for global digital patient monitoring devices is predicted to reach US$451.54 billion by 2030. [1] Out of the numerous benefits offered by these gadgets, the biggest benefit is to allow doctors to keep an eye on their patients at all times so that the number of hospital visits can be reduced significantly.

The team "Embedded Wizards" would like to take this opportunity to express its interest in digital health monitoring applications by building a prototype of a portable low-power personal health monitoring system.

2. Problem Health Monitoring System Solves

The portable low-power health monitor system allows users to keep track of their physical conditions by checking the following health metrics at any given time:

- Body temperature
- Blood pressure
- Blood oxygen level
- Heat rate

Devices like health monitoring systems can be beneficial to a wide range of individuals such as the following:

- People suffering from the cardiovascular diseases
- People with high blood pressure
- Elder people
- Athletes

Given the possibility that the system can be commercialized, the mobile development on the iOS platform and the Android platform as well as the use of cloud data storage enables the application further to provide real-time data not only to the users themselves but also to their family doctors.

2.1. Product Specifications

The product is designed to meet the following specifications so as to be qualified as a portable IoT product:

Size	110 mm x 40 mm x 40 mm
Weight	150 gram
Connectivity	Low-Power Bluetooth
System Requirements	EFR Connect BLE Mobile APP is required for either iOS or Android platforms.
Power	mini-USB charging
	Energy Harvester - Solar Cells

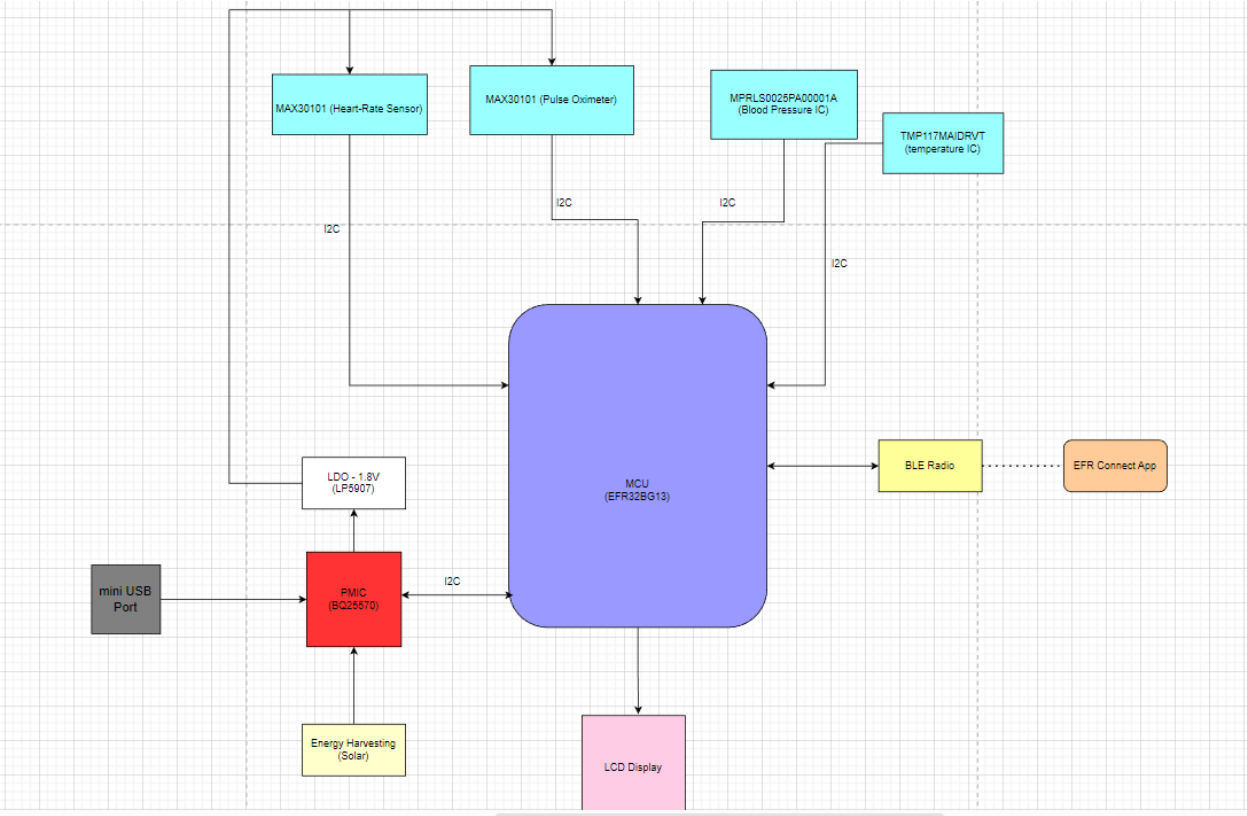
2.2 Product Features

The product offers the following key features:

- Instant measures of body skin temperature, heart rate, blood oxygen level and blood pressure
- Continuous power-up through super Capacitor or mini-USB
- Solar cells for renewable charging
- Bluetooth connectivity for sensor data available on mobile apps
- User-defined measurement schedule time
- Hospital mode and personal mode
- Offline sensor data display via micro-LCD display

3. Hardware block diagram

The following block diagrams show the key hardware components as well as the interactions required for the system to be functional as expected:



As can be seen above, all sensor ICs are communicated via the MCU via I2C protocol. All sensor data collected from the MCU is displayed on the attached LCD display and sent over to the EFR Connect App via BLE radio communication whenever they are available.

In addition, the solar cells and the mini USB port are used as energy sources for battery charging via BQ25570, which delivers 3.3V to all ICs. The LP5907 is used to deliver 1.8V exclusively to MAX30101.

4. Key Components

There are several key components required in order to build the product, this section covers them in sub-sections.

4.1 Processor Selection

Since the product is using BLE technology for communication, the Blue Gecko Bluetooth Low Energy SoC EFR32BG13 is used. EFR32BG13 offers the following key features that enable a rich set of applications:[2]

- 32-bit ARM® Cortex®-M4 core with 40 MHz maximum operating frequency
- 512 kB of flash and 64 kB of RAM
- Precision Low-Frequency Oscillator meets BLE Sleep Clock accuracy requirements
- Robust peripheral set and up to 31 GPIO
- 12-channel Peripheral Reflex System enabling autonomous interaction of MCU peripherals

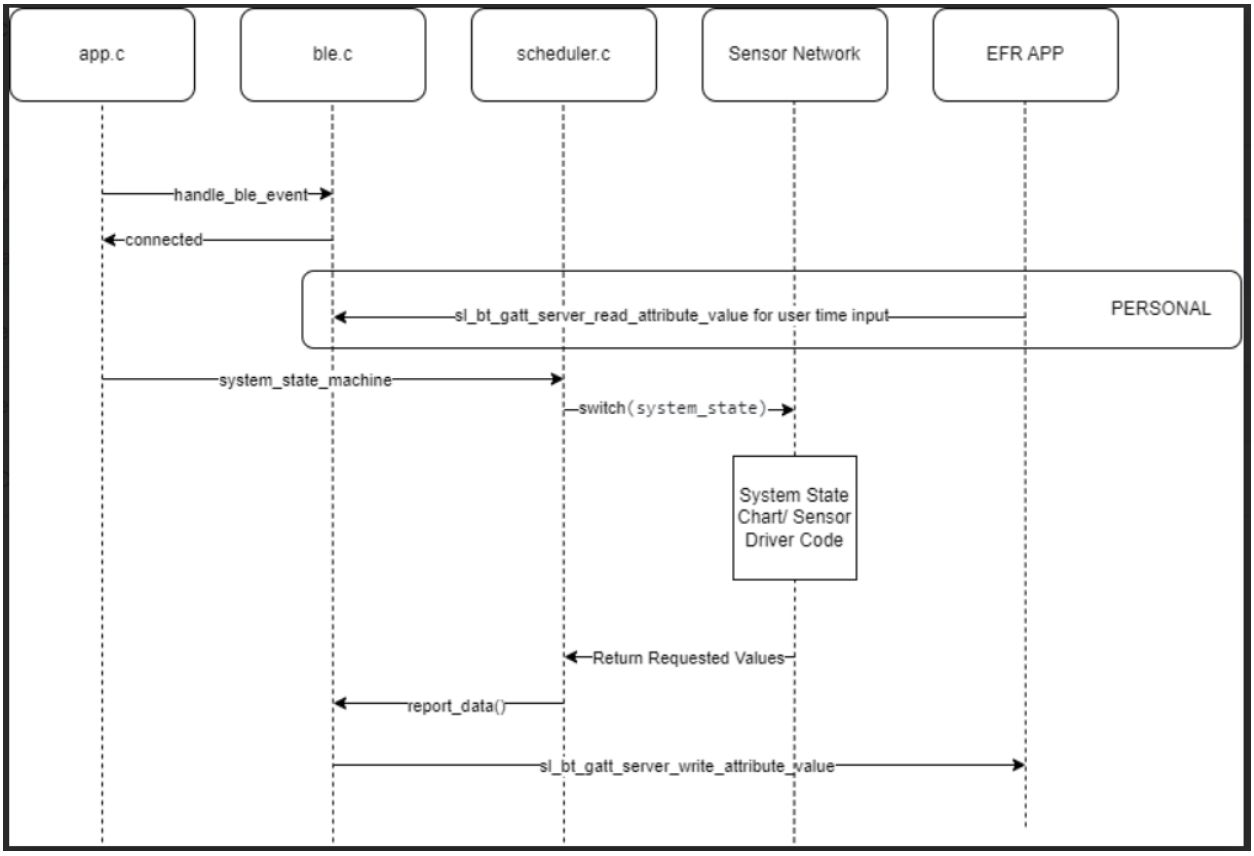
4.2 Sensor Selection

Sensor	Functionality	Power Performance
TMP117MAIDRVT	This is a high-precision digital temperature sensor. It is designed to meet ASTM E1112 and ISO 80601 requirements for electronic patient thermometers. [3] The TMP117 provides a 16-bit temperature result with a resolution of 0.0078 °C and an accuracy of up to ±0.1 °C across the temperature range of –20 °C to 50 °C with no calibration.[3]	TMP117 operates from 1.7 V to 5.5 V and typically consumes 3.5 μA. Additionally, it consumes only 150-nA in the shutdown mode. [3]
MPRLS0025PA00001A	The MPR Series is a very small piezoresistive silicon pressure sensor offering a digital output for reading pressure over the specified	The MPR Series consumes only 10 mW power typically, and it can consume as low as 0.01 mW. [4]

	full scale pressure span and temperature range.[4] It is designed for medical applications such as Non-invasive blood pressure monitoring, negative-pressure wound therapy, breast pumps and mobile oxygen concentrators.[4]	
MAX30101EFD+T	The MAX30101 is an integrated pulse oximetry and heart rate monitor module. The MAX30101 provides a complete system solution to ease the design-in process for mobile and wearable devices. [5]	The MAX30101 operates on a single 1.8V power supply and a separate minimum 3.3 V power supply for the internal LEDs. The module can be shut down through software with zero standby current and 0.7 uA shutdown current, allowing the power rails to remain powered at all times. [5]

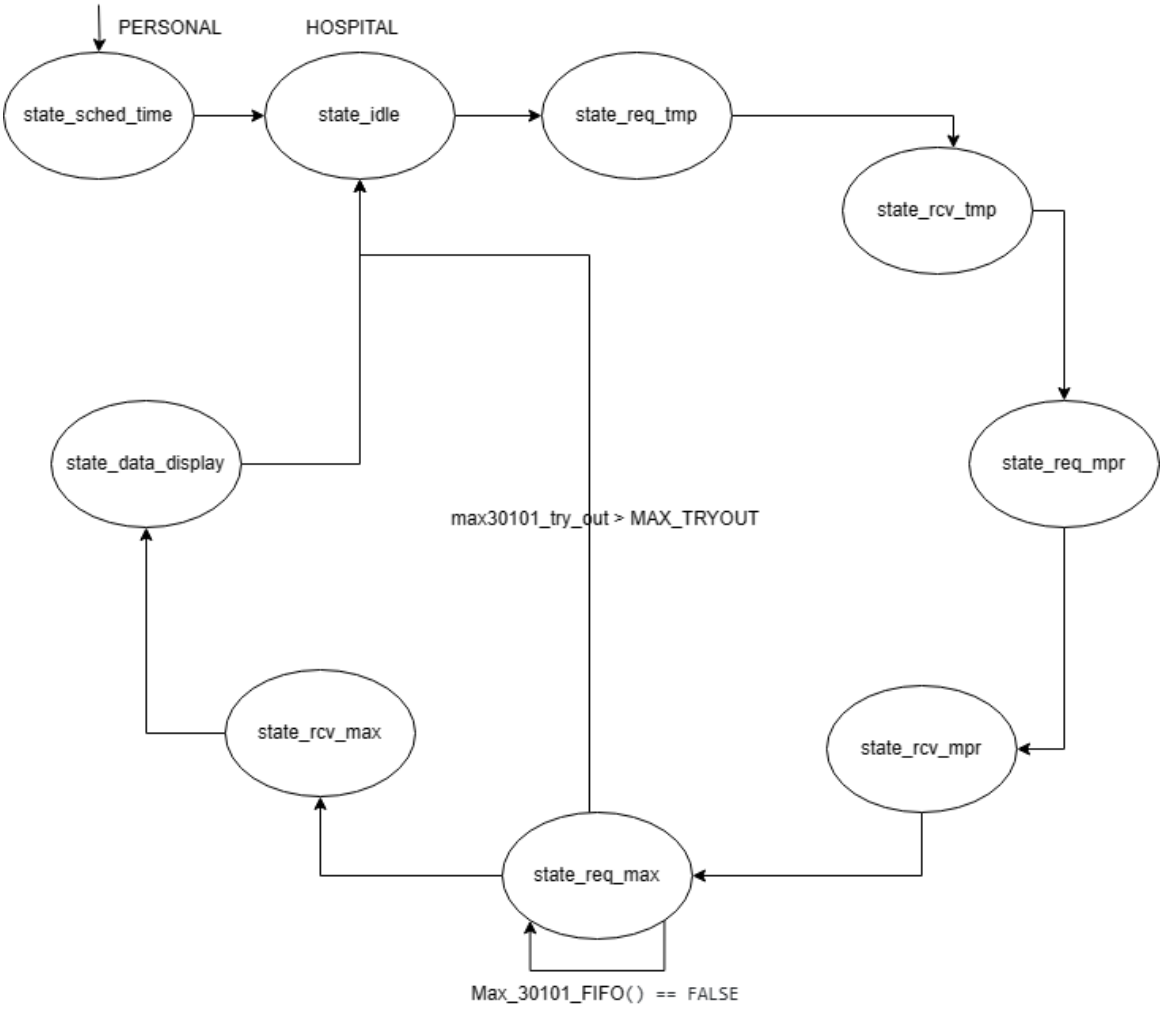
5. Software flow organizational or block diagram chart

The following diagram is the software data flow diagram in file level. The application function starts from the calling to handle_ble_evt() at app.c. handle_ble_evt( ) handles the smart bluetooth communication events such as opening and closing a connection. Provided that the BLE connection is established between the device and the user app, the system state machine starts to run to enable sensor data retrieval. All sensor data is obtained from the sensors through their respective driver code via I2C protocol, and all the sensor data is updated onto the EFR connect App through bluetooth.



In terms of the system state machine design, there are nine states in total, ranging from the idle state to the data display state. Specifically, the device enters the schedule-time state to receive the user-defined measurement schedule time, provided the device is running in personal mode. In the hospital mode, the device enters an idle state instead, since the device is preconfigured with the appropriate schedule time for sensor measurement.

The sensor measurement starts when a timer underflow interrupt occurs: the device transitions itself to temperature measurement state whenever a timer underflow event happens. The TMP117 is configured as one-shot mode during the temperature data requesting period to ensure the latest data is captured. As soon as the temperature value is received the sensor is set to the shutdown mode to reduce the power consumption. Moving forward, the blood pressure data is requested immediately after collecting the temperature value. Both the temperature sensor and the micro-pressure sensor share I2C0 bus and the MAX30101 uses I2C1 bus. Since it takes time and trials to obtain the instant heart rate and SpO2 data, a maximum tryout value of 200 is used to sample data. An internal FIFO is used to buffer raw data read from the MAX30101 and the device transitions to the data display state if no finger is detected;otherwise, the SpO2 and heart rate calculation algorithm is executed prior going to the data display state. Finally, the device enters the data display state and the collected data values are both displayed on the local LCD and sent to the mobile application via BLE.



The following table describes each BLE state used in detail:

Case	Description	Command	Description	Usage
sl_bt_evt_system_boot_id	Indicates that the device has started and the radio is ready. This event carries the firmware build number and other software and hardware identification codes.	sl_bt_system_get_identity_address	Read the Bluetooth identity address used by the device, which can be a public or random static device address.	Getting the identity address for further data transmission
		sl_bt_gatt_server_write_attribute_value	Write the value of an attribute in the local GATT database.	In system boot to write data on Bluetooth In reporting data to the Bluetooth characteristic IDs
		sl_bt_advertiser_create_set	Create an advertising set. The handle of the created advertising set is returned in response.	Creating an advertising handle
		sl_bt_advertiser_set_timing	Set the advertising timing parameters of the given advertising set. This setting will take effect next time that advertising is enabled.	advertiser timing set with different calibrations

		sl_bt_advertiser_start	Start advertising of a given advertising set with specified discoverable and connectable modes.	Initiate the process of BLE transmission
sl_bt_evt_connection_opened_id	Indicates that a new connection was opened.	sl_bt_advertiser_stop	Stop the advertising of the given advertising set.	Once the connection opened, disable the advertising handle
		sl_bt_connection_set_parameters	Request a change in the connection parameters of a Bluetooth connection.	New parameters set for further usage
sl_bt_evt_connection_closed_id	Indicates that a connection was closed.	sl_bt_advertiser_start	Start advertising of a given advertising set with specified discoverable and connectable modes.	Once the connection is closed, open the advertiser handle with the stored parameters.

Additionally, the following table shows the services as well as GATT characteristics used in the design:

Service Name	Temperature	Blood Pressure	Heart Rate	Pulse Oximeter	Read Time
Description	To display temperature of skin on app	To display blood pressure on	To display heart rate on app	To display Spo2 on app	To read input scheduled time from

		app			user
Service	health_thermometer	BloodPressure	Heart_Rate	Pulse_Oximeter	read_time
UUID	1809	a891d111-97f4-45e7-9dec-511e75273f91	1db1d7d9-4475-40bb-b1a0-063204aab1cc	76909632-a84a-417c-a67e-3f37b829ed1a	8ad36434-3bb5-4802-80db-7e8a37356772
Characteristic	Temperature Measurement	Blood Pressure Measurement	Heart Rate Measurement	SPO2	read_time_sec
UUID	2A1C	b00e43d4-7b5d-47db-b011-7910a0084f57	877d5939-5ba1-46d1-a3fc-2e323061eeba	bd5ff987-8d65-4b50-ba03-40309e736810	d4b1359b-04ae-4907-b47e-f8cd9b67642a
Value length	20	2	1	1	1
Properties	Read	Read	Read	Read	Write
Descriptor	Client Characteristic Configuration	Client Characteristic Configuration	Client Characteristic Configuration	Client Characteristic Configuration	Client Characteristic Configuration
UUID	2902	2902	2902	2902	2902

6.List of Commands

There are no user-defined commands for this product as the device is designed to interact with the users only when sensor data is available, and the device runs without user control for the rest

of the time. However, several BLE commands are used for BLE connectivity between the device and the EFR app. The following table shows all BLE commands used in the design:

BLE Command	Command Description
sl_bt_system_get_identity_address	Read the Bluetooth identity address used by the device, which can be a public or random static device address.
sl_bt_gatt_server_write_attribute_value	Write the value of an attribute in the local GATT database.
sl_bt_advertiser_create_set	Create an advertising set. The handle of the created advertising set is returned in response.
sl_bt_advertiser_set_timing	Set the advertising timing parameters of the given advertising set. This setting will take effect next time that advertising is enabled.
sl_bt_advertiser_start	Start advertising of a given advertising set with specified discoverable and connectable modes.
sl_bt_advertiser_stop	Stop the advertising of the given advertising set.
sl_bt_connection_set_parameters	Request a change in the connection parameters of a Bluetooth connection.

7. Planned development schedule and when tasks were completed

The team planned to develop the project incrementally by dividing the project tasks into multiple milestones. Within each milestone, a set of tasks are defined and assigned with given deadlines. Generally speaking, the team worked on both firmware and hardware in parallel to minimize the overall development time. The following table illustrates the development schedule and task completion status in a high-level manner, as the details of the task schedule can be checked from the team Gantt Chart.

Milestone	Software Tasks	Completion Time	Hardware Tasks	Completion Time

Project Planning	Evaluation board & design IDE setup	2/8/23	Hardware components selection	
Software Preparation & Hardware Component Design	Bring up the eval board	2/8/23	TMP117 schematic & footprint design	2/7/23
	Implement logic to display data on LCD	2/11/23	MPRLS0025PA00001A schematic & footprint design	2/15/23
	Implement the BLE handler to make the eval board communicate with the EFR Connect App	3/30/23	MAX30101 schematic & footprint design	2/11/23
	Read datasheet of TMP117	2/10/23	EFR32BG13P532F512GM48-D schematic & footprint design	2/19/23
	Read datasheet of MPRLS0025PA00001A	2/11/23	Passive component schematic & footprint design	2/20/23
	Read datasheet of MAX30101	2/12/23	BQ25570 schematic & footprint design	2/21/23
	Study reference manual for I2C protocol on EFR32BG13P532F512GM48-D	2/13/23	LP5907 schematic & footprint design	2/22/23
	Review BLE communication from IoT firmware course	3/28/23	LCD controller schematic & footprint design	2/20/23

	N/A	N/A	Component schematic & Footprint Review	2/27/23
Driver & Schematic Development	Implement I2C driver	2/23/23	Temperature sensor (TMP117) schematic design	2/23/23
	Test I2C driver	2/23/23	Micro-pressure sensor (MPRLS0025PA00001A) schematic design	2/23/23
	Implement temperature sensor driver on top of the I2C driver	2/24/23	MAX30101 schematic design	2/24/23
	Perform unit test on temperature sensor driver	2/24/23	Power management circuit schematic design	2/26/23
	Implement micro-pressure sensor driver on top of the I2C driver	2/25/23	EFR32BG13 radio circuit and power circuit schematic design	3/2/23
	Perform unit test on micro-pressure sensor driver	2/26/23	Inverter antenna schematic & footprint design	3/2/23
	Implement MAX30101 sensor driver on top of the I2C driver	3/30/23	Schematic design of EFR32BG13 I/O circuit, system reset circuit and debug port circuit	2/25/23
	Perform unit test on MAX30101 sensor driver for raw data check	3/31/23	LCD controller schematic design	2/26/23
	N/A	N/A	System Schematic Review	3/4/23

Low Power Design & PCB Layout Placement	Implement LCD controller driver	4/2/23	Preliminary PCB layout placement	3/8/23
	Implement logic to switch energy mode from EM0 to EM1/EM2/EM3	4/18/23	PCB layout review	3/10/23
BLE Design & PCB Layout Routing	Implement logic to support BLE communication with energy mode management	4/19/23	PCB layout routing design	3/18/23
	N/A	N/A	PCB layout routing review	3/19/23
System Architecture Design	System state machine design	4/2/23	N/A	N/A
	System state machine implementation	4/2/23		
	MAX30101 SpO2 & Heart Rate measurement algorithm	3/30/23		
	Unit testing of MAX30101	3/31/23		
Software Integration & Board Assembly	Software module integration	4/14/23	Develop verification plan	4/8/23
	System integration tests without BLE communication & Energy mode management	4/15/23	Component availability check placement	4/10/23
	System integration tests with BLE communication	4/16/23	Component placement on PCB board	4/11/23
	System integration tests with BLE communication &	4/26/23	Board assembly via electronic oven	4/11/23

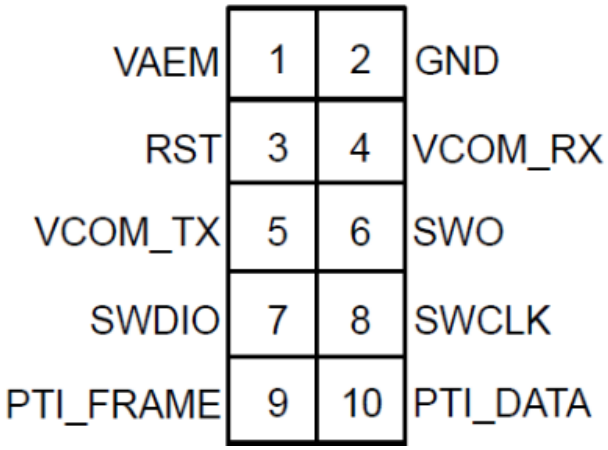
	Energy mode management			
Board Verification Test	N/A	N/A	Test the power delivery from mini USB port to supercapacitor via PMIC	4/15/23
			Test the voltage delivery from BQ25570 and LP5907 to ICs	4/15/23
			Solder solar cells and test supercapacitor charging from solar cells	4/27/23
System Integration Test	Test TMP117 sensor by running program with TMP117 driver alone	4/25/23	Trace I2C0 transactions to check potential abnormalities.	4/22/23
	Test MPR sensor by running program with MPR driver alone	4/25/23	Trace I2C0 transactions to check potential abnormalities.	4/22/23
	Test MAX301010 sensor by running program with MPR driver alone	4/25/23	Trace I2C1 transactions to check potential abnormalities.	4/22/23
	Test all sensors together by running the program without BLE communication	4/26/23	Trace both I2C0 and I2C1 transactions to check potential abnormalities.	4/22/23
	Test the entire program with BLE communication by running the code	4/26/23	N/A	

	on the device and using EFR Connect App to check received sensor data			
	Test the entire program with BLE communication and low-power management by running the code on the device and using EFR Connect App to check received sensor data.	5/1/23	N/A	

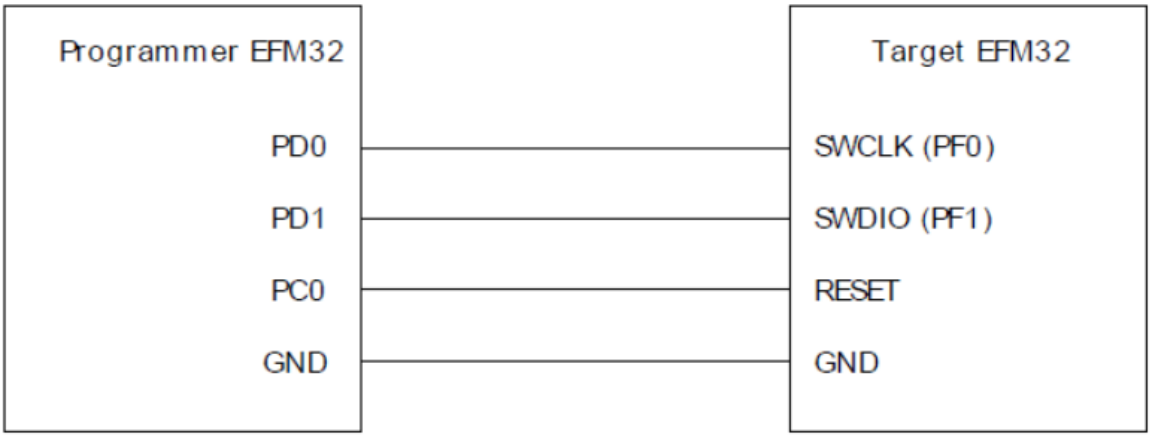
8. Programming Target Microcontroller

In our design, a Simplicity Debug Adapter Board is used to provide a subset of debug capabilities and features through a smaller form-factor connector interface, which is the Mini-Simplicity Connector, a 10-pin (2×5) small form-factor (1.27 mm pitch, 3.05 mm pin length) header connector, on the custom hardware design. Additionally, a standard 10-pin ribbon cable (Samtec part number FFSD-05-D-6.00-01-N) is required for the Mini Simplicity Interface Connector on the Simplicity Debug Adapter Board. The primary reason for using the Mini-Simplicity Connector is for space conservation as the device is space constrained.

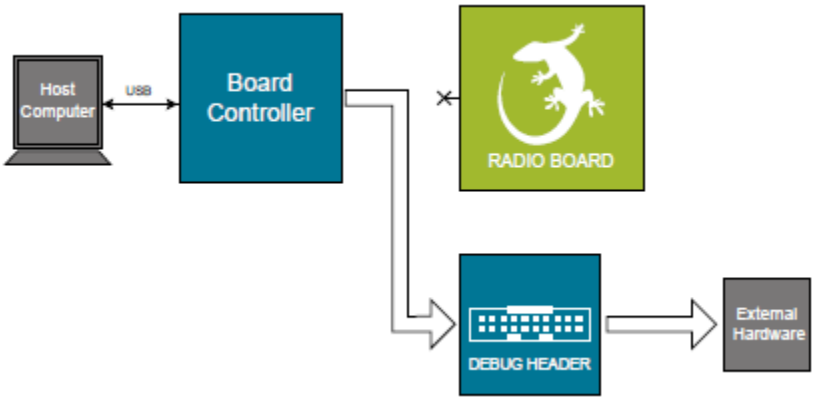
The connector pinout of the Mini-Simplicity Connector is shown below:



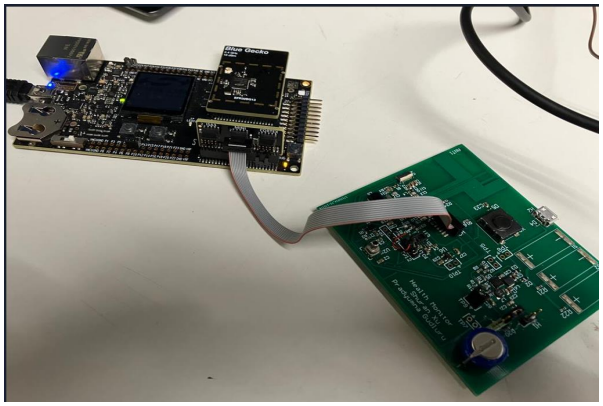
To program the code to the flash of the target board, the RESET, SWCLK, SWDIO and GND pins from the target must be connected to the configured pins on the programmer via the Mini-Simplicity Connector. Besides, the voltage level should be the same on both the host and the target. The following diagram shows the connection:



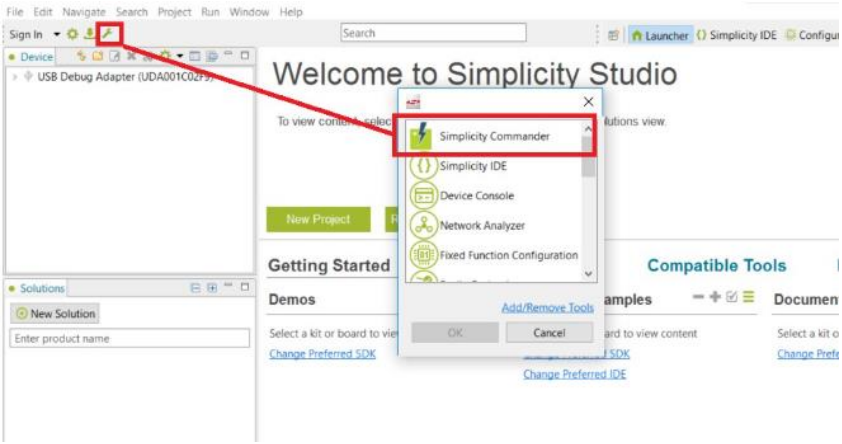
The evaluation board or the board controller is connected to the external hardware or custom-made board (in our case M-Wiz) through the debug header as per the below connections.



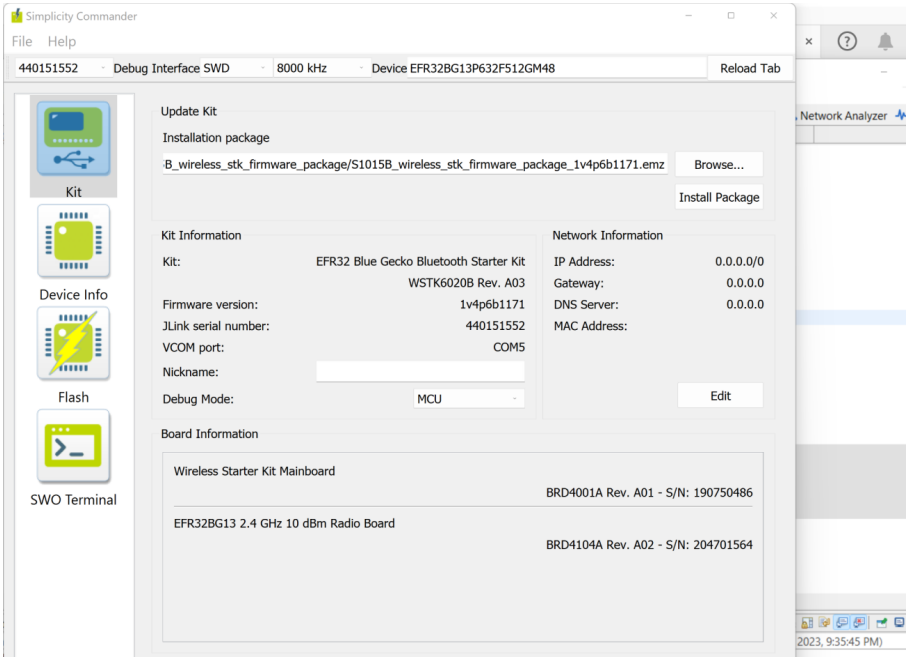
The above connection setup in team's case is shown as:



The simplicity commander is opened from the tools section of the Simplicity Studio:



The following display will be popped out and according to the user guide, for flashing the code and using the custom board on debug mode, the Debug mode has to be changed to OUT from MCU.



On the Flash widget, the memory on the radio chip present on the eval board can be erased and reset for the safe functioning of the custom board. Finally, the custom-made board is connected to the simplicity studio using the debug port on the eval board. This helps in flashing the code directly onto the custom board and checking for its energy profiling, logging, etc.

9. Current profile over time based on expected application usage

In this section, the current consumption of the system in different phases is displayed with the help of the energy profiler.

As for the demo, the hospital mode of operation is used so that the device resumes itself from the idle state for sensor data reading every 10 seconds. Since MAX30101 requires the finger pressing in order to provide the heart rate and the SpO2 rate, the device's current consumption differs significantly with and without the finger pressing. The following screenshot shows the average current consumption over one period when the user is not pressing the MAX sensor. As can be seen below, the average consumed current is 3.99 mA over one period.

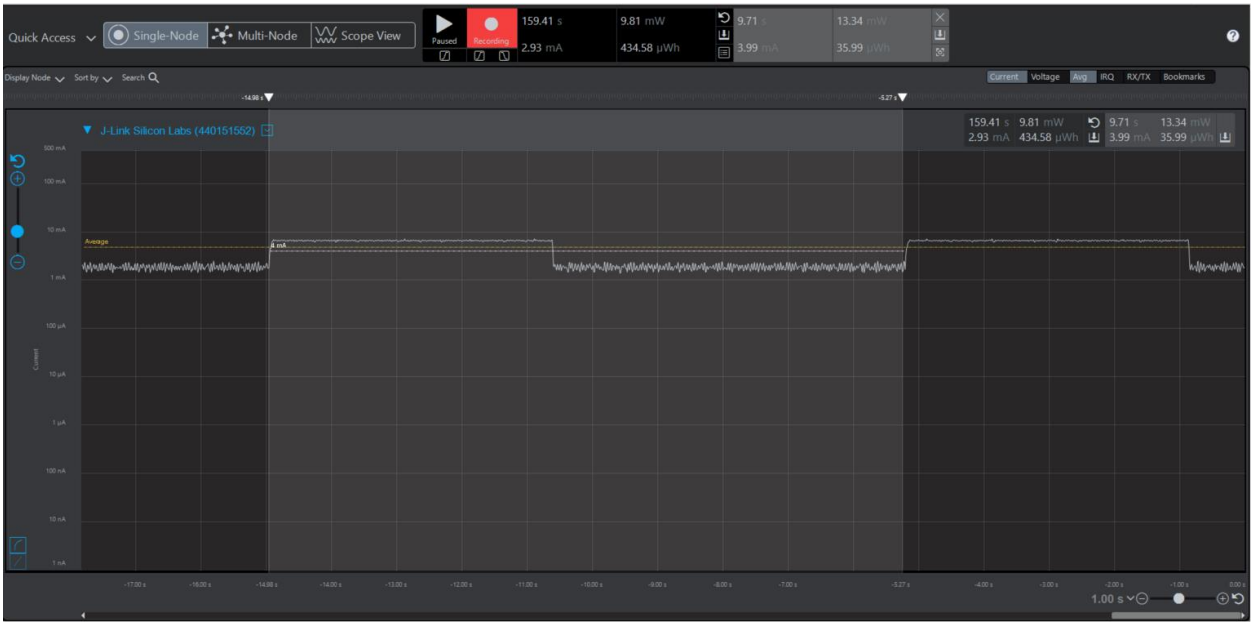


Fig: Average current consumption over the period without the finger pressing.

For the case where the user is pressing the MAX sensor, the time that the device spends in the EM1 mode is significantly less compared to the above case, and the average consumed current drops from 3.99 mA to 2.24 mA for one period of time.

Fig:Average current consumption over the period with the finger placed

Additionally, the following screenshots show the current consumption during the BLE advertising and BLE connection states. It is clearly that initiating a BLE connection consumes considerable amount of current, drawing the current from the device from 1.66 mA to 2.59 mA.

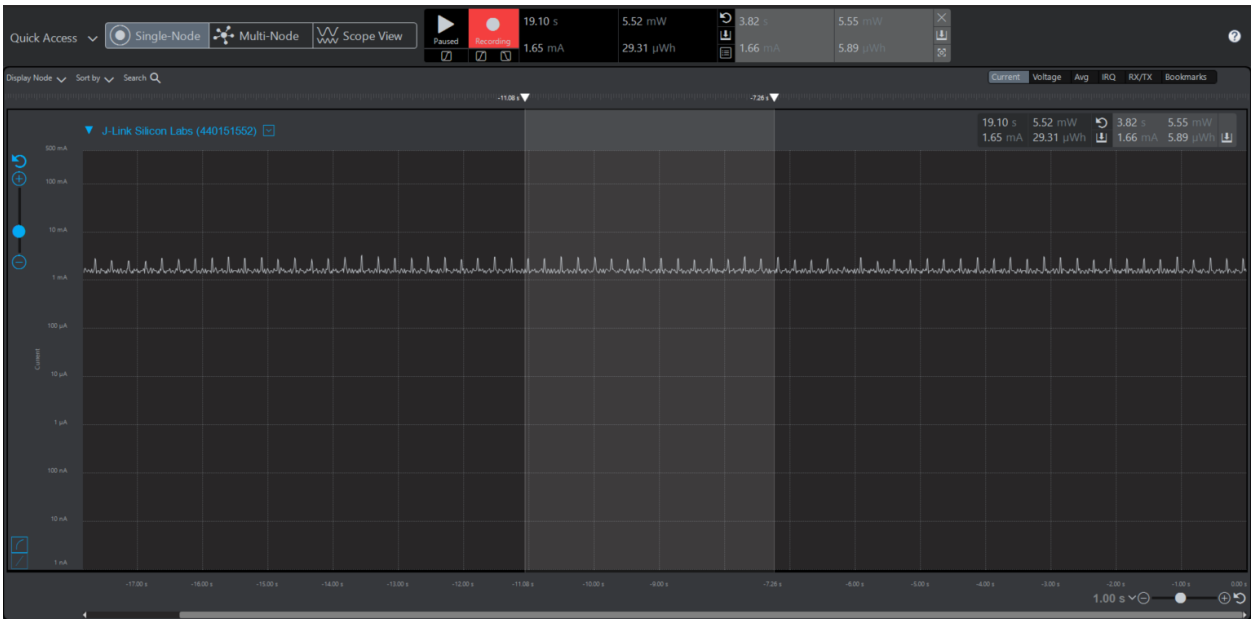


Fig: average current consumption during the BLE advertising state

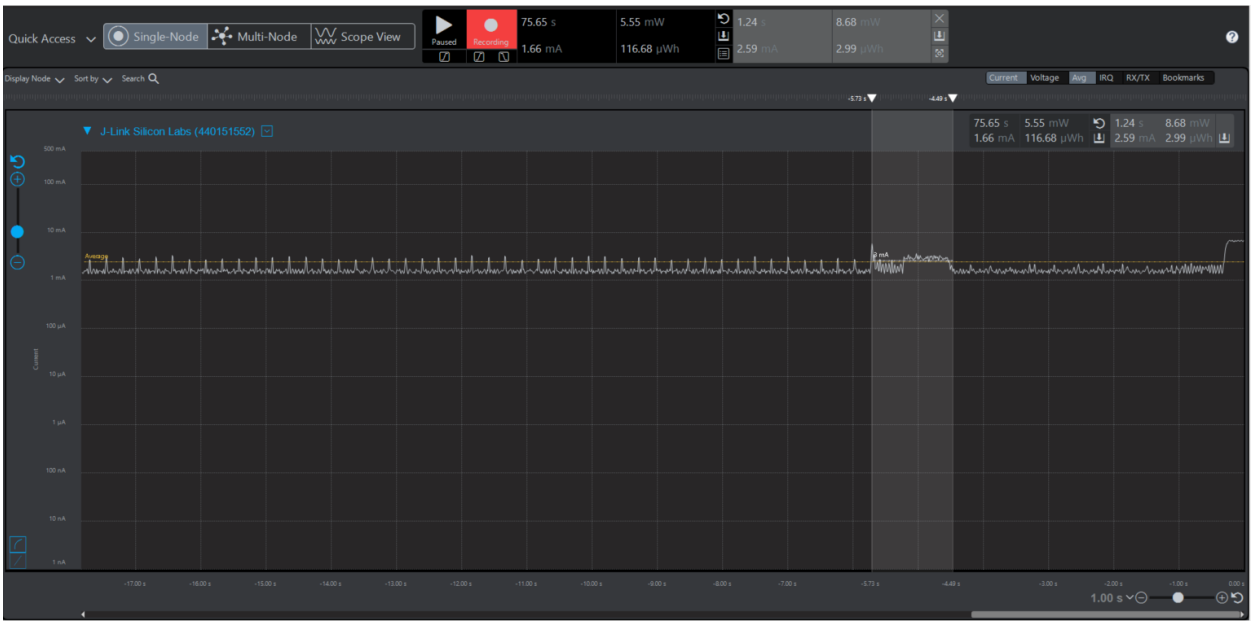


Fig: Average current consumption during the BLE connection

The following screenshot shows the instant consumed current during the sensor reading state, 5.07 mA is reasonable as all sensors are running and the MCU is running in the EM1 mode.

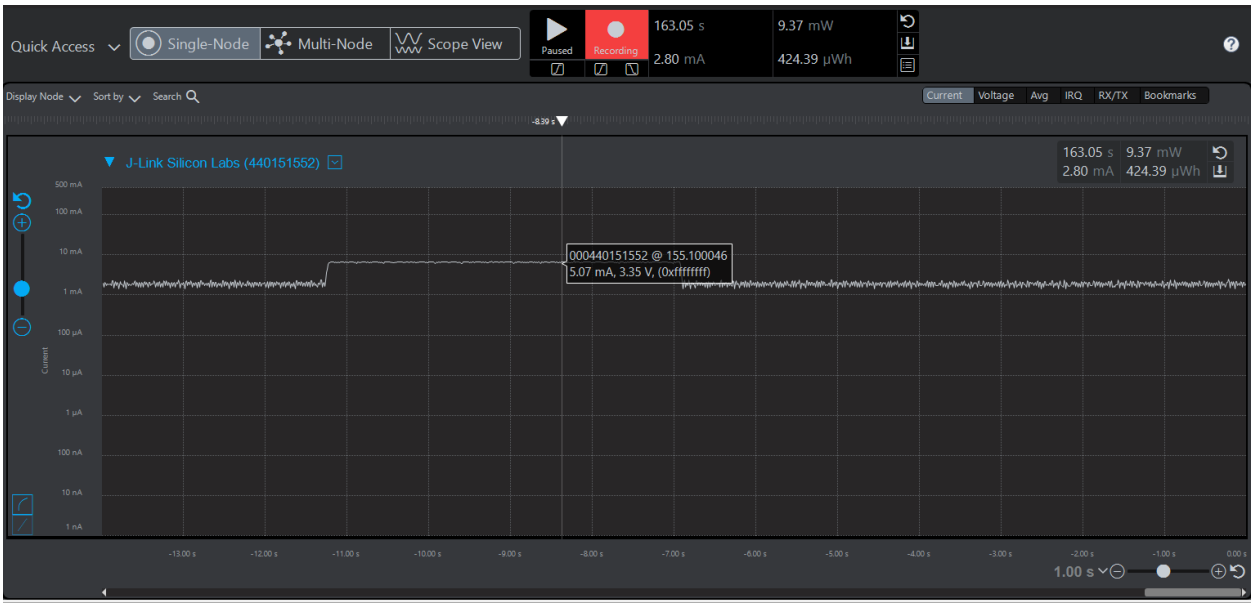


Fig: Instantaneous current during the sensor measurement state

The following screenshot shows the average consumed current when the device is in the idle state. It is clear that device is in the deep sleep mode as only 736 uA is consumed.

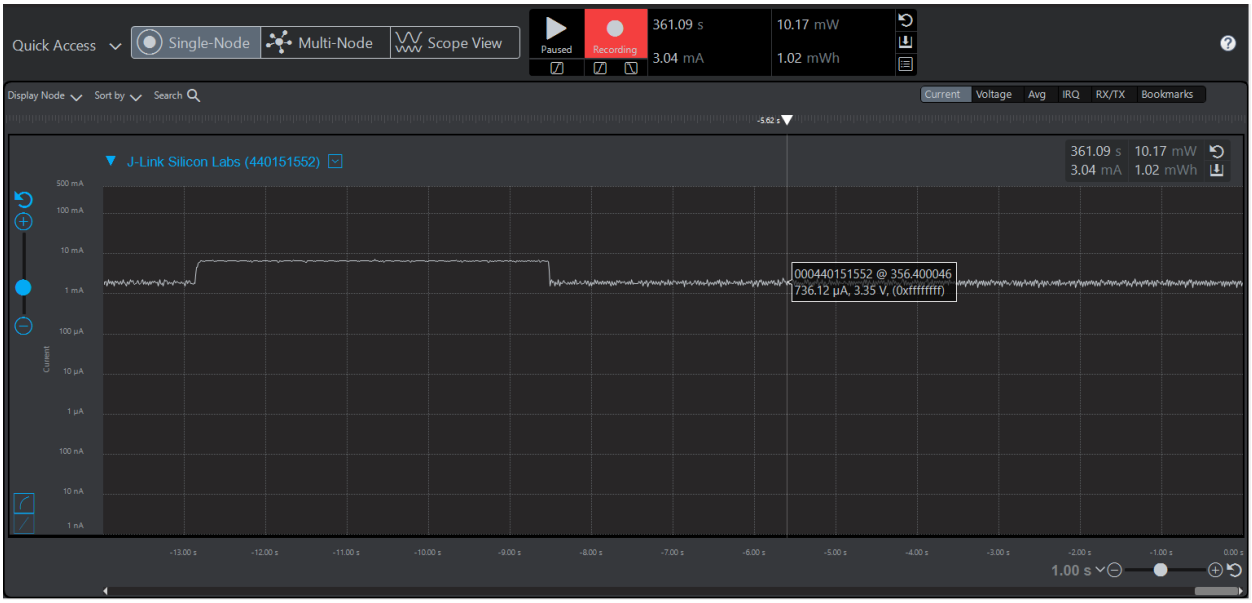


Fig: Instantaneous current at the device idle state

Apart from the current profiling, the current consumption analysis is performed to check against the initial proposed current consumption. It is clear to see that the device consumes less amount of current than the proposed amount overall, and this is due to the fact that the team was conservatively estimating the current consumption in the preliminary design phase.

Energy Mode	Current Consumption (mA) - Proposed	Current Consumption (mA) - Actual
EM0	EFR32BG13: 128 uA/MHz * 38.4 MHz / 1000 = 4.915 mA TMP117: 0.022 MPR: 0.000221 LCD: 0.01516 MAX30101: 0.0025 I2C:0 Total current = 4.915 + 0.022 + 0.000221 + 0.01516 + 0.0025 = 4.955	From the energy profiler current consumed = 5.23mA

EM1	EFR32BG13:10 I2C: 0.33 LCD: 0.01516 TMP117: 0.135 MPR: 0.000682 MAX30101: 1.1 Total current = 10 + 0.33 + 0.01516 + 0.135 + 0.000682 + 1.1 = 11.58	From the energy profiler current consumed = 5.07mA
EM3	EFR32BG13: 1.53 uA I2C: 0 TMP117: 0.022 MPR: 0.000221 LCD: 0.01516 MAX30101: 0.0025 Total current = 0.00153 + 0.022 + 0.000221 + 0.01516 + 0.0025 = 0.0415	From the energy profiler current consumed = 0.736 mA

10. Energy Storage Element selected and selection documentation

10.1. Energy Storage Element Selection

The team decided to use a Vishay, 15F, 3-cell supercapacitor as the energy storage element. The primary reason for such selection can be listed as follows:

- The device requires a high supply current (up to 50 mA) to the entire circuitry
- Low ESR
- The device requires high recharge cycles (up to 1092 cycles) for the supported warranty years (3 years)

Specifically, the Vishay 15F supercapacitor satisfies the project requirements according to the team's estimation. As the product is designed to support a warranty of 3 years of 7 days a week, so a total of 52 x 7 x 3 = 1092 recharge cycles are required. According to the datasheet of the Vishay supercapacitor, a maximum of 100,000 recharge cycles are supported. In addition, the usage energy of 38.48J from the Vishay 15F supercapacitor satisfies the estimated energy consumption calculated. Therefore, the team concluded that the Vishay 15F meets the system requirements from both energy usage and recharge cycle standpoints.

The following table shows the key Vishay 15F specifications that marks itself a matching candidate for the project:

Table: Specs of Vishay 15F supercapacitor [6]

Usable Energy	38.48 J
Peak discharge current	15 mA
Recharge cycles	Max 100,000 cycles
Max charge current	50 mA
self-discharge rate	0.042 mA
Tolerance	-20% to +80%
ESR	7.5 ohm
Capacitance	15 F
Voltage drop due to ESR @ max current	0.03 V
Operating Temperature	-20 to 85 Celsius
Mechanical dimensions	12 x 7.5 mm

10.2 Power Management IC Selection

In terms of the PMU selection, the team chose the BQ25570 as this IC supports energy extraction from solar cells and voltage conversion via an internal buck converter. In addition, the proprietary nano power management feature helps manage the energy budget of the system in a fine-grained manner.

Apart from the appealing features supported by the BQ25570, the functionality of the BQ25570 closely matches with the power requirement of the device. Specifically, since a 3.3v supply voltage is required to be supplied to the entire circuit with solar cells as well as miniUSB ports served as the energy sources, the extracted power from solar cells can firstly be stored in the Vishay 15F supercapacitor and then converts the battery voltage to 3.3V via the internal buck converter.

Additionally, the BQ25570 supports a low battery discharge cut-off voltage. Specifically, the PMU supports a low battery discharge cut-off voltage called 'VBAT_UV' threshold. As the team uses a supercapacitor as the energy storage element, the team planned to set the cut-off voltage to match the cutoff voltage of Vishay 15F, which is 2.4 V which can be configured in a programmable fashion.

10.3 Energy Source Selection

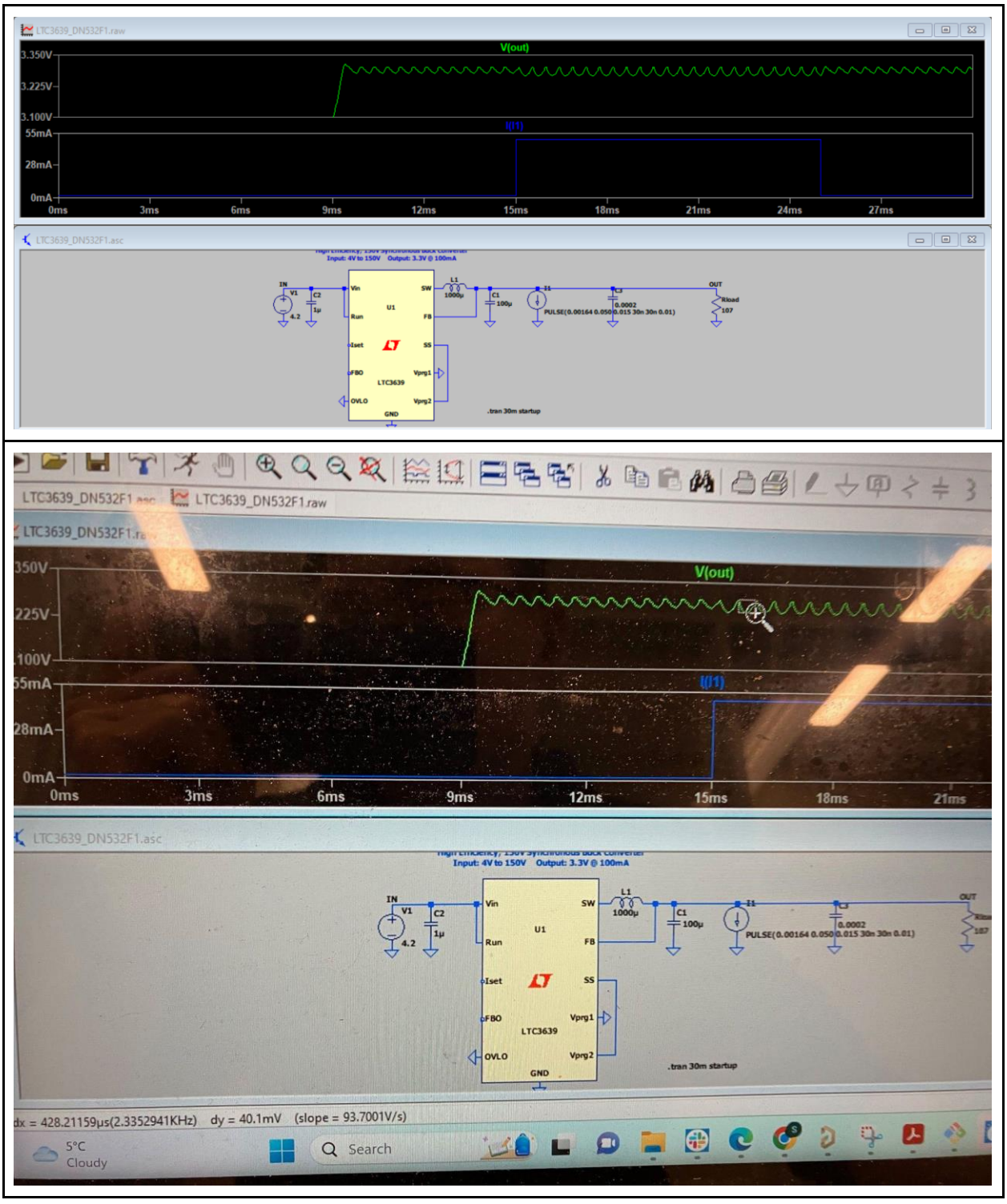
As the product is most likely being used at home or in a hospital, there is a good chance of having charging ports available. The major source of charging would be through a mini-USB port, and the alternative energy source is solar cells in case the device is used outdoors. The team decided to use the solar panel as the energy harvesting method for the following reasons:

- Renewable and economic-friendly energy
- Cost effective
- Availability of solar cells with a small form factor

11. PMU simulation results/summary

To enable the power circuit to handle step loads in power such as when the microcontroller wakes up or the radio turns on and off, the team performed PMU simulation with a 200uF bulk capacitor using the LTSpice. Specifically, with a 200uF bulk capacitor placed between the PMIC and the system load, the voltage ripples can be suppressed down to approximately 40mV (meaning +/- 20mV) which poses nearly no threat to the system loads:

Bulk Capacitance Simulation Screenshots



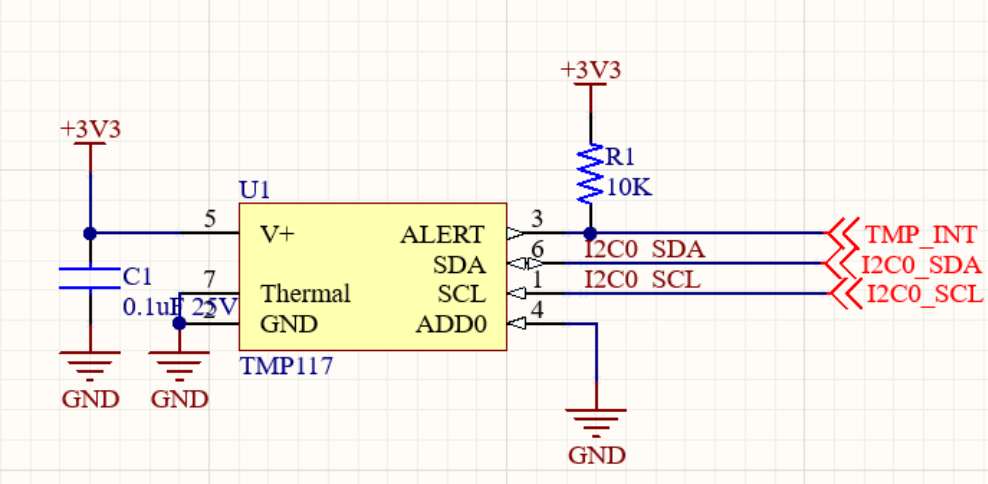
Since the team uses a supercapacitor as the battery, there is no need to use a bulk capacitor.

12.Bulk or sizeable decoupling capacitor selection and backup data

Bulk capacitors are only used in the sensor circuits and decoupling capacitors are used in all ICs to minimize switching noises.

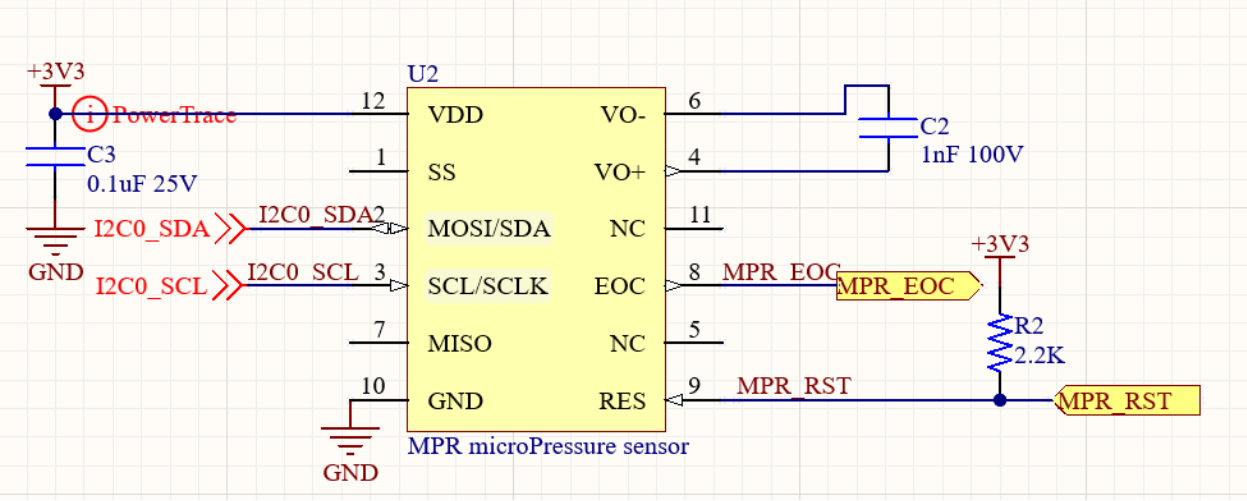
12.1 TMP117 Circuit

According to the TMP117 datasheet, a minimum of 100nF/0.1uF decoupling capacitor should be used near the VCC pin of the TMP117 IC to reject power supply noise. [3]



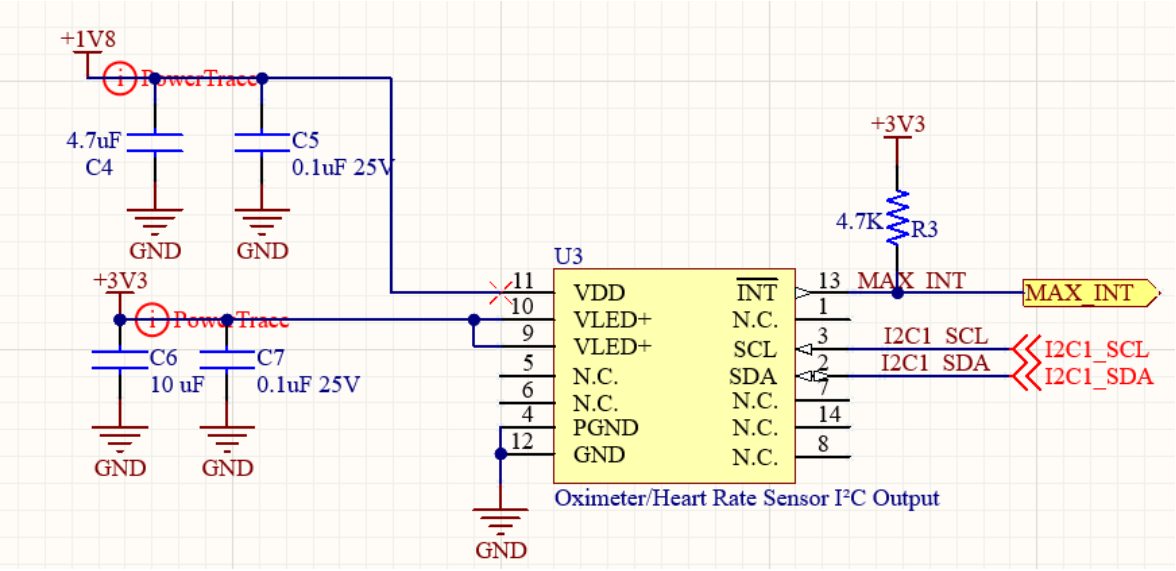
12.2 MPR Circuit

According to the datasheet of MPRLS0025PA00001A, a recommended 0.1uF bypass capacitor should be integrated into the end user design to ensure output noise suppression: [4]



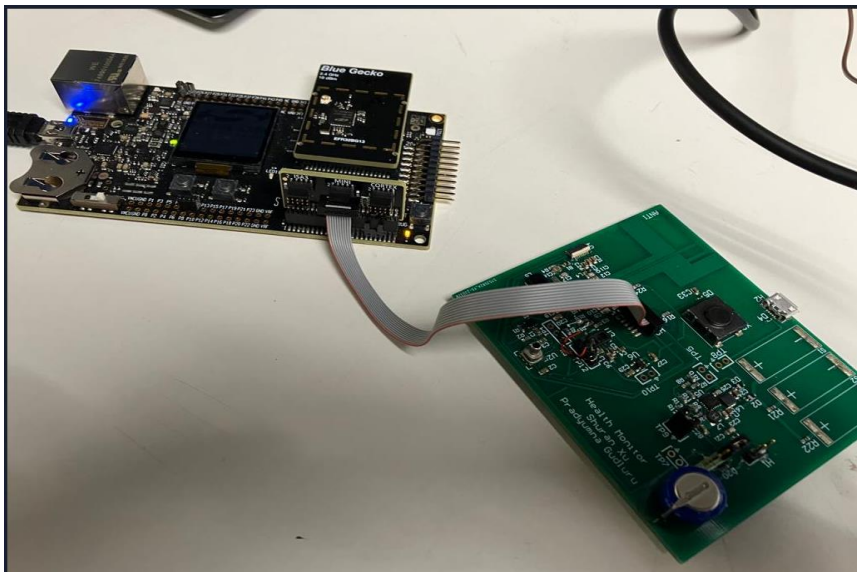
12.3 MAX30101 Circuit

The LEDs in the MAX30101 are pulsed with a low duty cycle for power savings, and the pulsed currents can cause ripples in the VLED+ power supply. To ensure these pulses do not translate into optical noise at the LED outputs, the power supply must be designed to handle these. To ensure that the resistance and inductance from the power supply to the pin is much smaller than 1Ω, at least 1µF of bypass capacitance to a good ground plane is needed and the capacitor should be located as close as physically possible to the IC. Thereby, the team used a 0.1 uF bypass capacitor followed with a 10 uF bulk capacitor the VLED line and a 0.1 uF bypass capacitor with a 4.7 uF bulk capacitor to minimize the power noise in both high frequencies and low frequencies. [5]

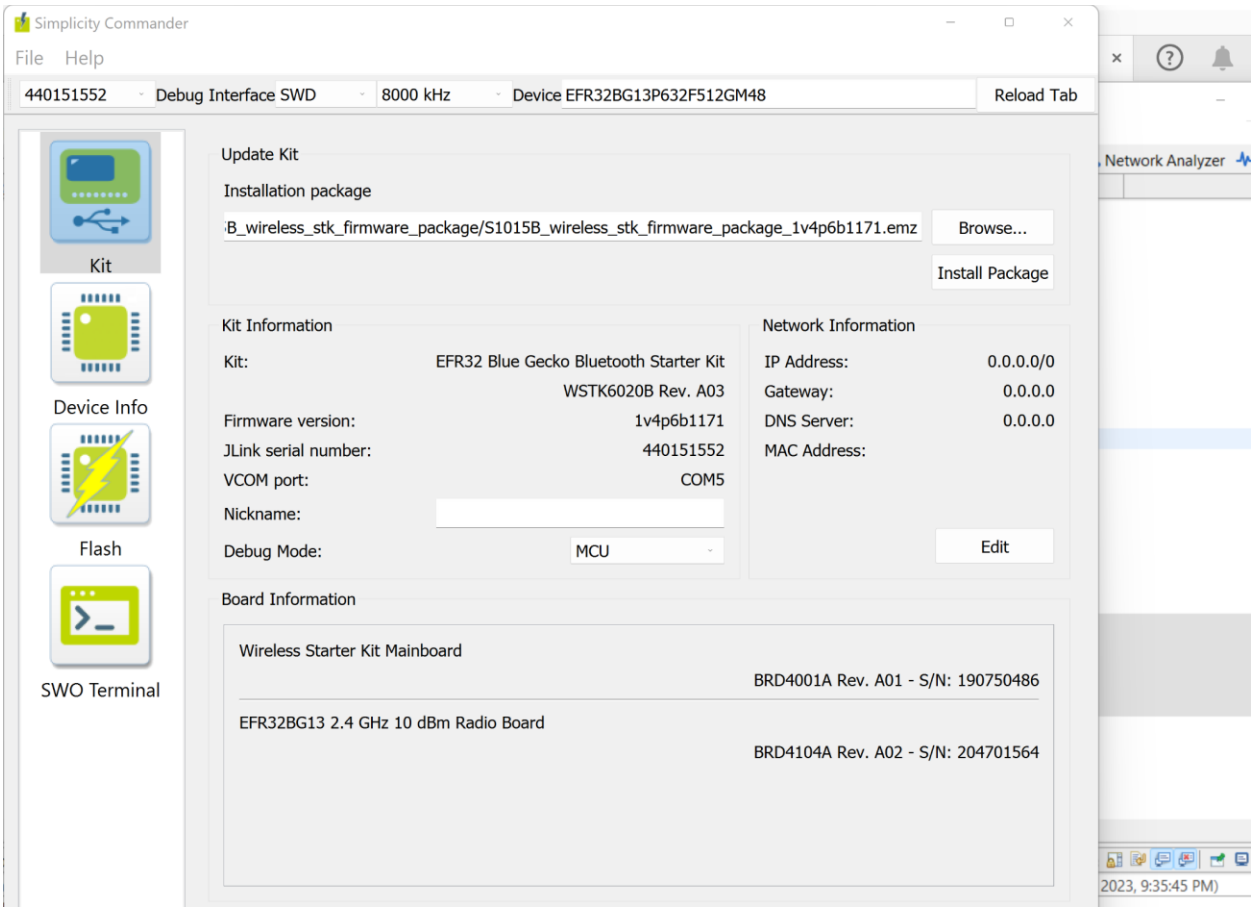


13. Will an external energy source be required to program the MCU

A Mini-Simplicity Connector and a BlueGecko evaluation board as well as the Simplicity IDE are required to program the MCU, and the evaluation board is used as the energy source for the MCU to be programmed. The evaluation board can be configured as the ‘OUT’ mode through the Simplicity Commander to deliver the code to the MCU via the Mini-Simplicity connector. The following setup shows that the simplicity mini connector is connected to the debug port of M-Wiz with a LED glowing on the eval board to show that the eval board is configured to OUT.



In the following screenshot, the debug mode is changed from OUT to MCU after the eval board has been used as an external board to program the MCU from the M-Wiz, which is the final product.



14. Planned test points

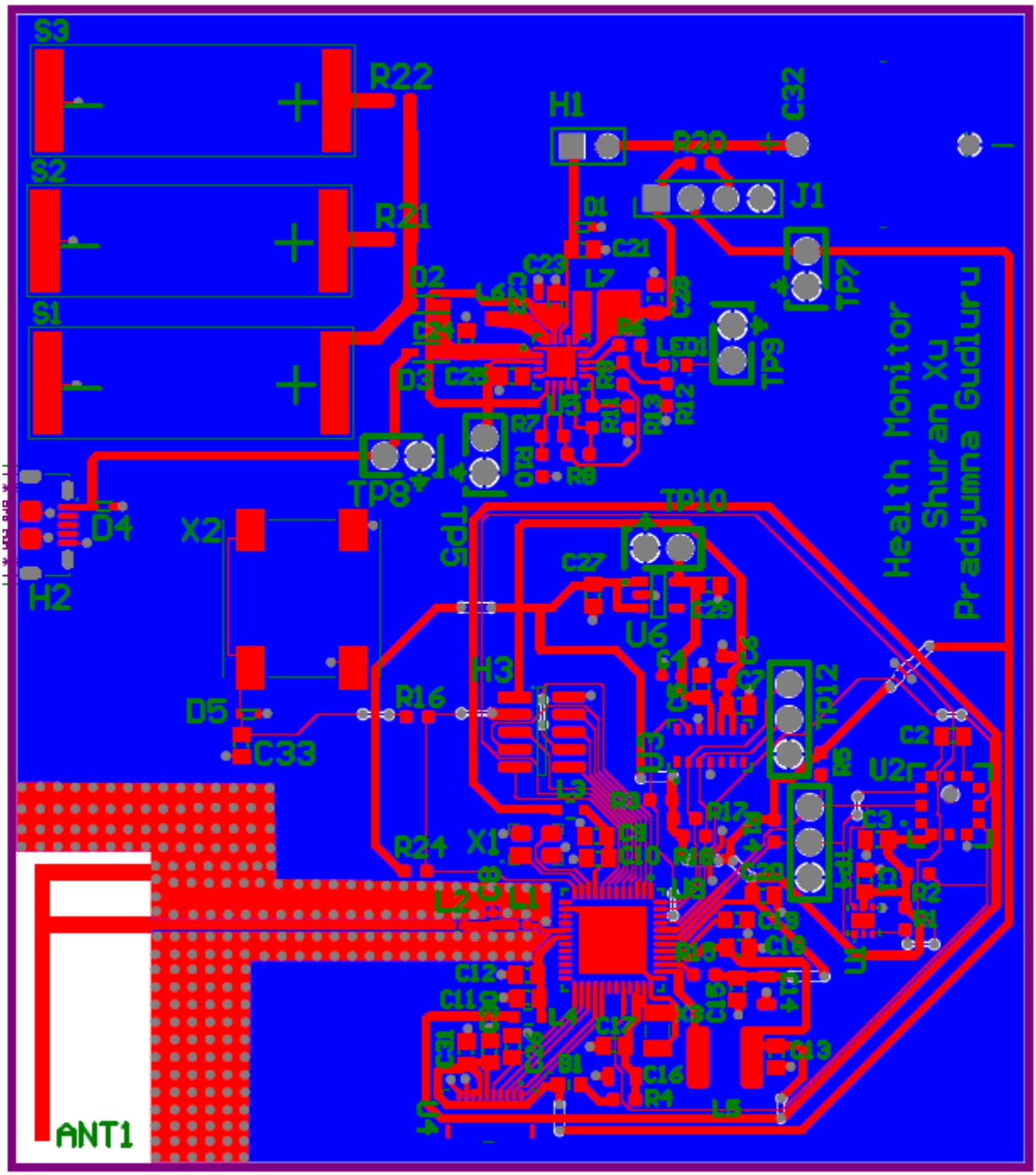
A set of test points are needed to assist board bring-up tests as well as firmware tests. The following table lists the key test points planned on the PCB board:

Signal Name	Test Pin Number	Description
I2C1_SDA	TP12	SDA line on I2C1 bus for probing data transfer between EFR32BG13 and MAX30101.
I2C1_SCL	TP12	SCL line on I2C1 bus for probing clock transfer between EFR32BG13 and MAX30101.
I2C0_SDA	TP4	SDA line on I2C0 bus for probing data transfer between EFR32BG13 and TMP117 and MPR.
I2C0_SCL	TP4	SCL line on I2C0 bus for probing data transfer between EFR32BG13 and TMP117 and MPR.
USB 5V	TP8	To check the 5V from USB
VSTOR	TP5	Check the voltage level supplied by the boost charger. It can be used to find the state of the BQ25570.
VBAT_OK	TP9	Check if the storage element is ready to supply power to the system load.
3.3V	TP7	To check 3.3 V to the digital plane
1.8V	TP10	To check 1.8V from LDO

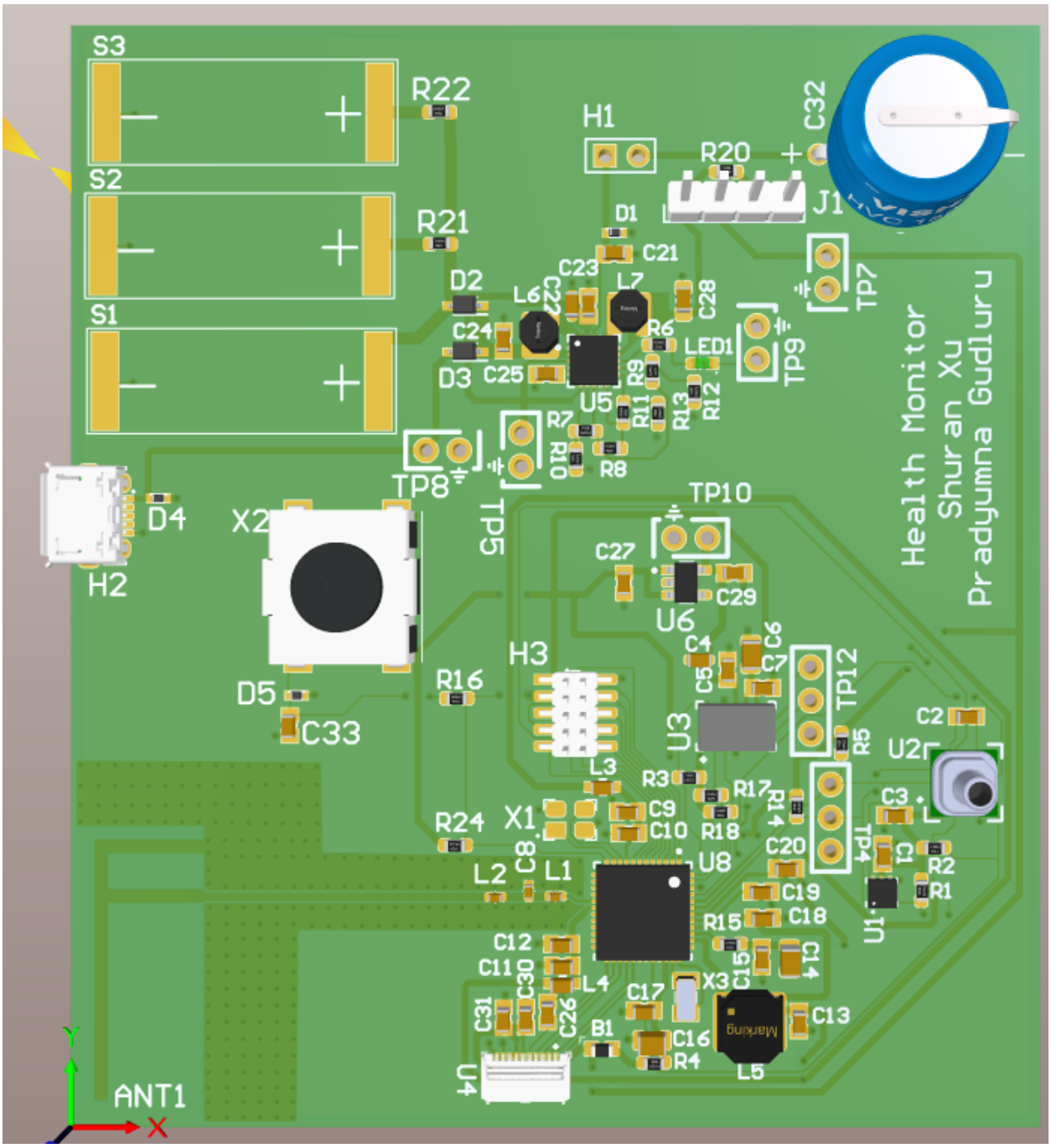
The planned test points helped the team tremendously during the board bring-up and firmware debugging phases. Specifically, TP7 and TP10 helped the team to verify that the PMIC successfully delivered 3.3V to all ICs. And TP9 as well as the connected LED indicator helped the team to check if BQ25570 detects the supercapacitor being ready to deliver voltage to the system loads. Furthermore, the I2C test points helped the team identify the I2C layout issue from the SoC to the MAX30101. Provided more development team is given, more test points should have been used to help detect the sensor status so as to ease the firmware development.

15. Photos of the assembled board

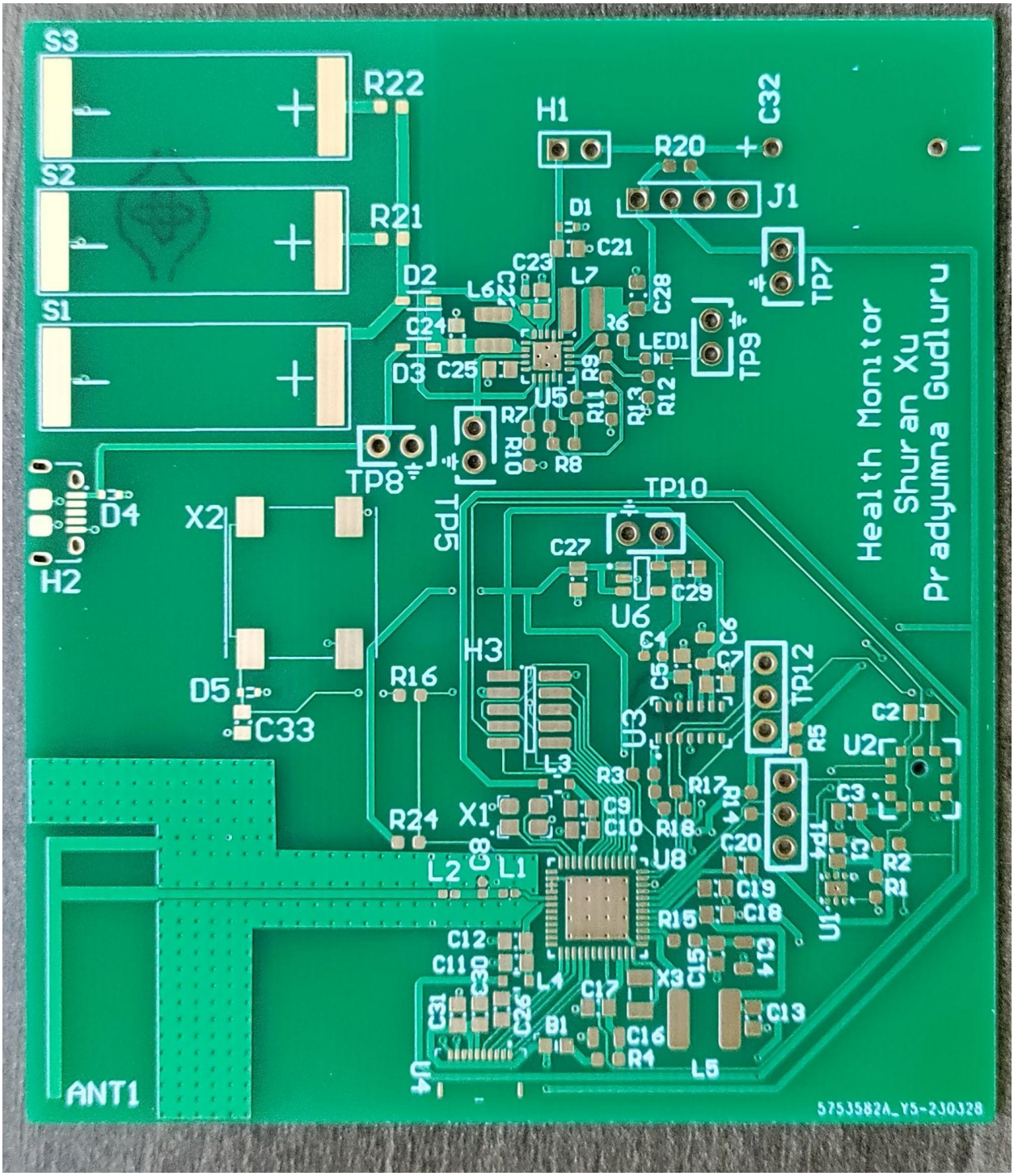
The connectivity of the bare board was verified as soon as the team received the board from JLCPCB. The basic verification steps are uploaded onto the verification plan. The PCB layout of the Health Monitoring System is shown as follows:



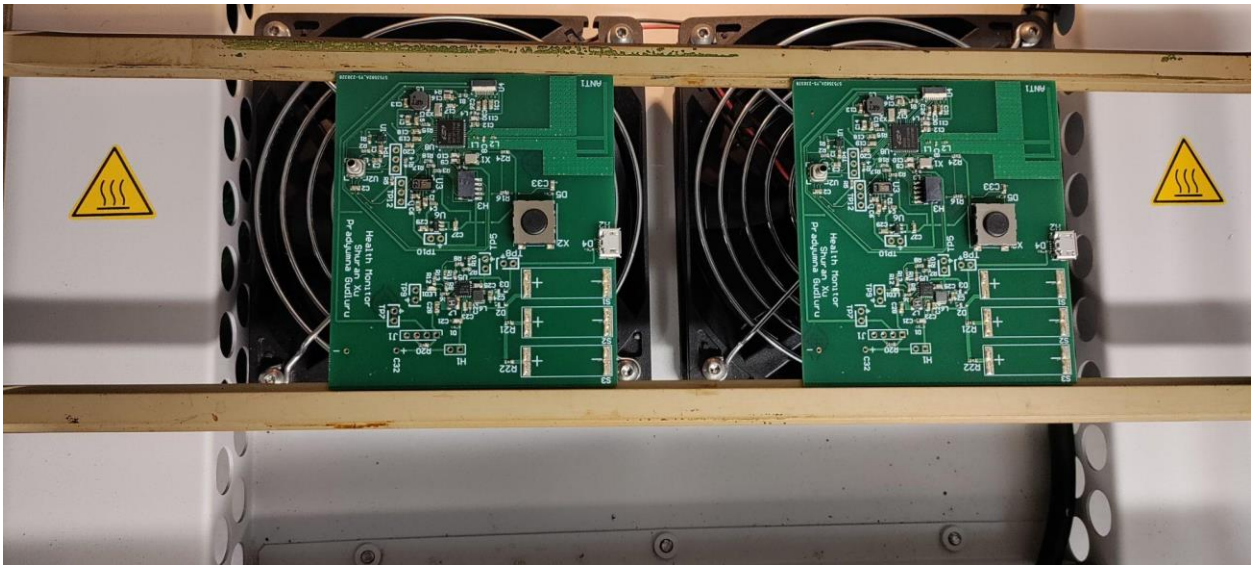
The corresponding PCB 3D layout is shown below:



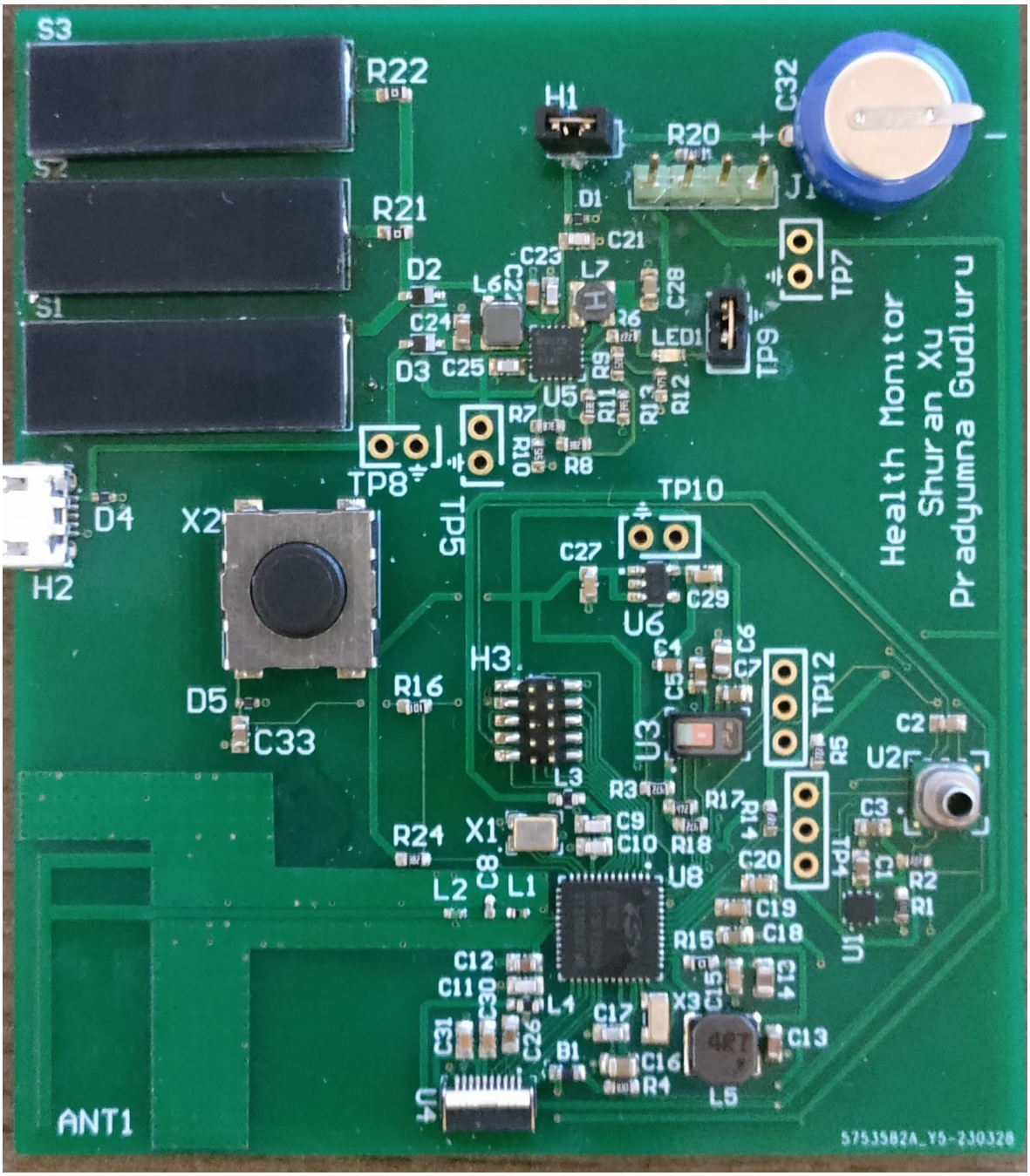
And the bare board can be seen below:


S3
S2
S1
R22
R21
H1
C32
R20
J1
D1
C21
C23
L7
C28
D2
L6
C24
R6
LED1
TP9
TP7
D3
C25
U5
R9
R11
R13
R12
R7
R10
R8
TP8
TP10
TP5
D4
X2
H2
C27
C29
U6
H3
C1
C6
C5
C7
TP12
R5
U2
C2
D5
R16
C33
U3
L3
R3
R17
R14
C3
R24
X1
C9
C10
R18
C20
TP4
C8
U8
L2
L1
C19
R2
R1
C18
R15
U1
C12
C11
C30
C15
C14
L4
C17
X3
C31
C26
B1
C16
C13
R4
L5
U4
ANT1
Health Monitor
Shuran Xu
Pradyumna Gudluru
5753582A_Y5-230328

Upon the component placement, the boards look as follows prior to sending to the electric oven:



The final assembled board is shown as follows:



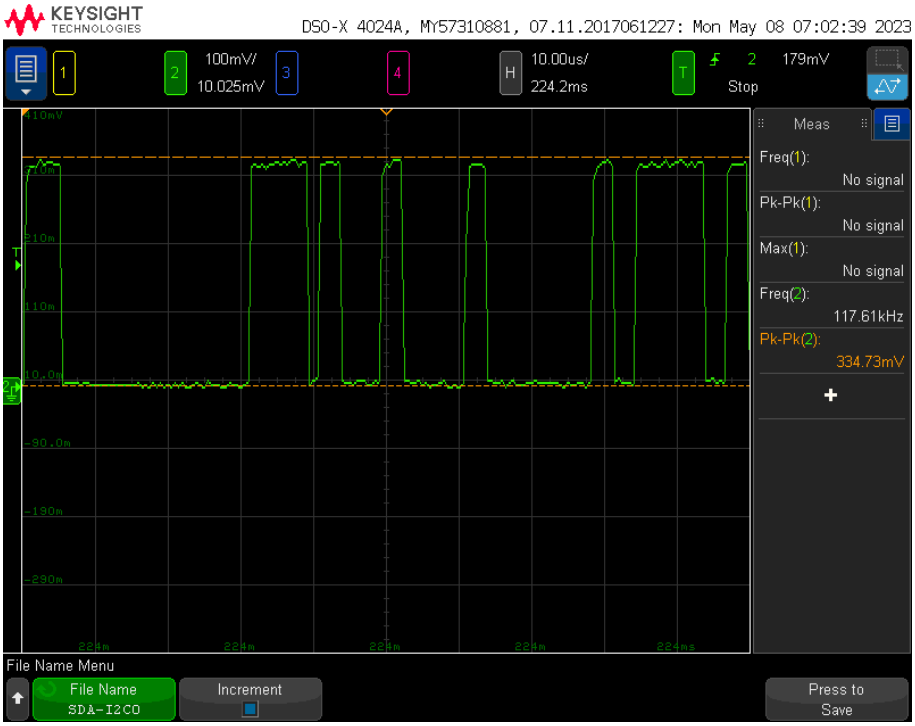
16. Complete detailed verification report

The detailed verification report is attached to the submission bundle and will be submitted along with the final report.

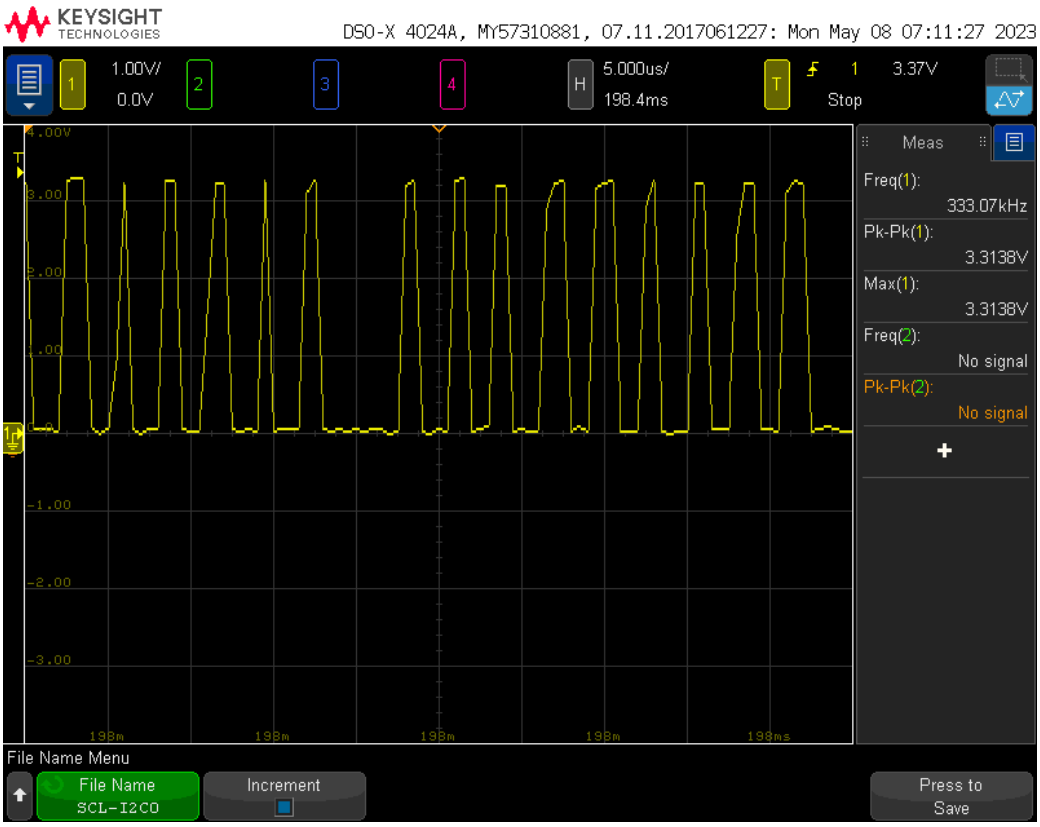
17. Signal quality analysis of key signals

The quality of key signals such as crystal frequencies and I2C clock and data lines are measured using various instrumentation tools such as oscilloscope and logic analyzer during the system integration test phase to ensure all signals can be delivered correctly among ICs.

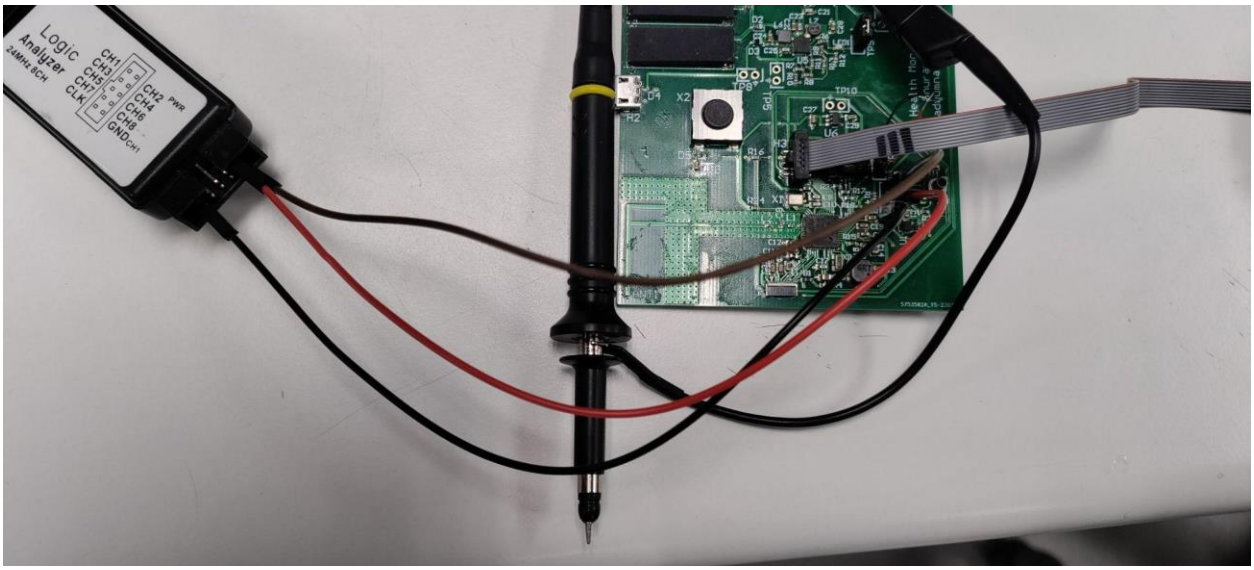
The following image shows the I2C transaction on the SDA line with a 1-x probe. There is a visible noise in the system, and this probably is due to the probes connected to the board while measuring the values.



The following screenshot shows the SCL line of I2C transactions. The non-ideal triangular waveform indicates that the RC time constant is bigger than the expected value. This might be due to the large pull-up resistors used when designing the I2C circuits. Despite the non-ideal waveform of the SCL line, the SCL frequency, which is shown as 333 KHz, matches the software configuration performed for the I2C0 controller.

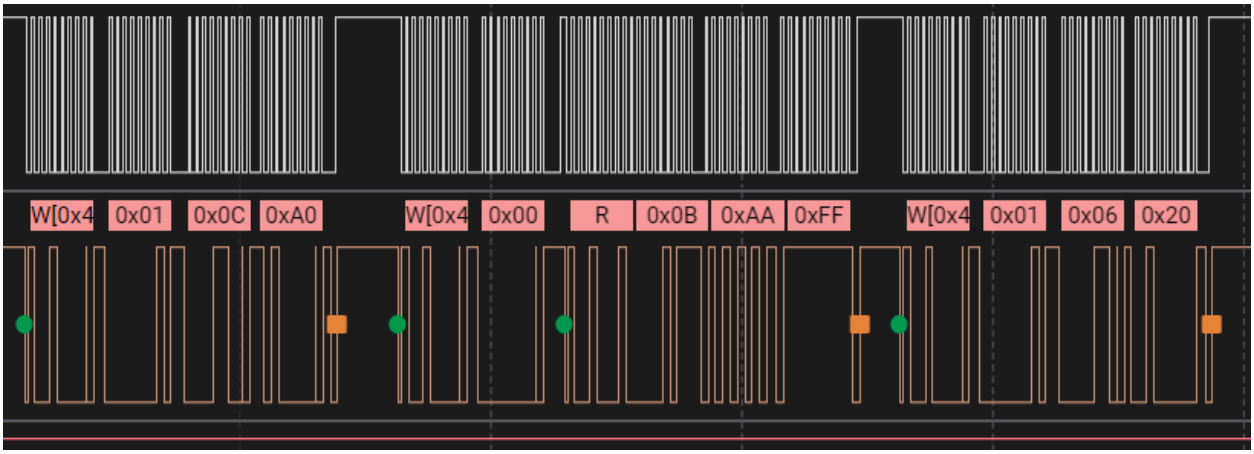


The following screenshot shows the hardware setup during the signal quality analysis phase. As can be seen below, a logic analyzer is probed onto the board to trace I2C line activities.



The following screenshot shows the I2C transactions when running the TMP117 driver alone. The brown wire is I2C_SDA, the red wire is the I2C_SCL and the black wire is the GND. The traced outputs for the I2C transfer matched the expectation and correct temperature values are

received by the MCU.



18. What were the difficulties encountered on the project

The team encountered many difficulties on the project and the following table shows the key ones as well as the respective solution:

Difficulties	Description
Unable to obtain raw data from the MAX30101	The team realized that the interpretation of the datasheet was wrong and a set of I2C driver functions were missing. The team resolved the issue by performing a I2C write followed by a burst of I2C read.
Unable to generate the heart rate and SpO2 values from the MAX30101 raw data	The team found that the LETimer period setting was too large to detect heart beat for the algorithm. The team resolved the issue by reducing the LETimer period to match the sampling rate of the MAX30101.
Occasional system halt due to I2C driver implementation issues found when running MAX30101 module	The team found bugs in the I2C write & read function where the functions were implemented in polling mode but the associated interrupts were enabled. The team resolved the issue by implementing I2C functions in the polling mode.

Unable to solder solar cells with the electric oven when all components have been soldered	The team used a wire as the medium to connect a solar cell and the PCB board. The connection was made by melting the wire so that both PCB board and the solar cell can be connected via the melted wire.
Unable to see the current reduction even when the system enters the EM2/EM3 mode	The team realized that the RF chip attached to the evaluation board was causing significant current draw during the energy profiling period, so the team resolved the issue by detaching the RF chip from the evaluation board and the current reduction was achieved as planned.

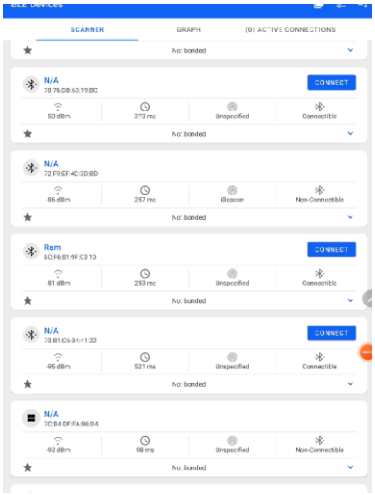
19. Summary of the functionality of the final project

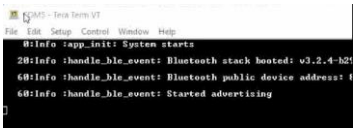
The product supports the following operation modes:

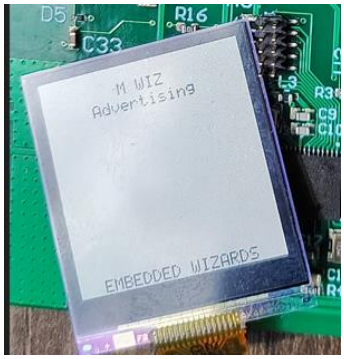
1. The Hospital mode where the scheduled time is selected to be 1 hr
2. The Personal mode where the scheduled time is user-defined input and has to be given as input every time the board is connected.

As for the demo and the functionality of the final project, the team configured the device to run in the hospital mode with 10 sec duration (since 1 hr would be a higher value for grabbing data and analysis). To demonstrate the functioning of the final project, a series of screenshots taken from EFR Connect App, terminal and LCD display show the correct functioning of the device phase by phase.

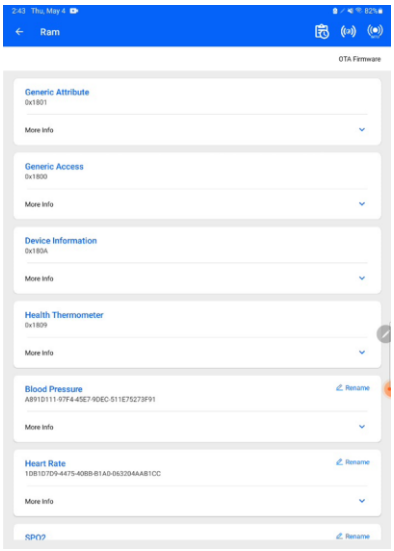
Initially, the system was initialized and started advertising for BLE connection:

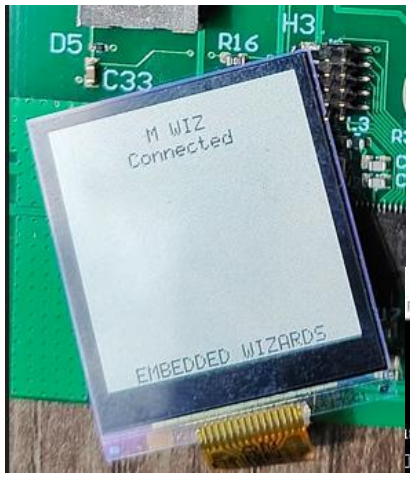


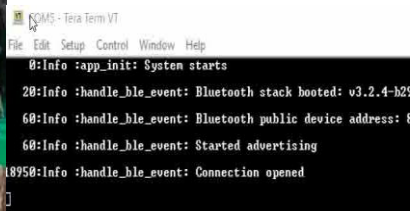




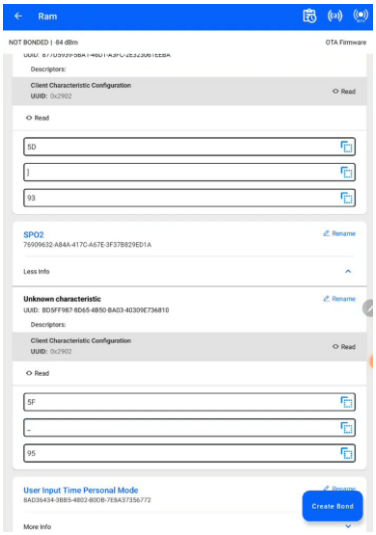
Once the Bluetooth is connected through the EFR app, the “connected” keyword appears on the LCD display and the GATT characteristics are shown on the EFR Connect app. In addition, the connection establishment log is also printed on the terminal.

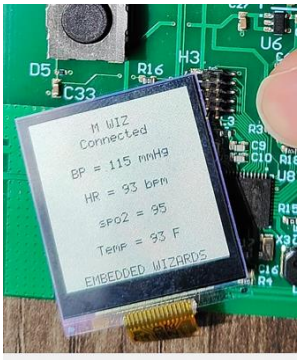


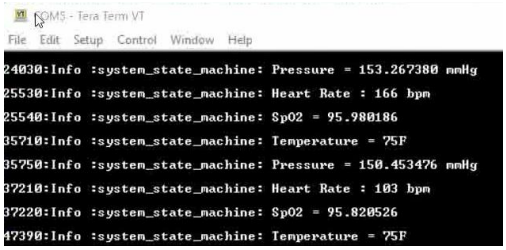




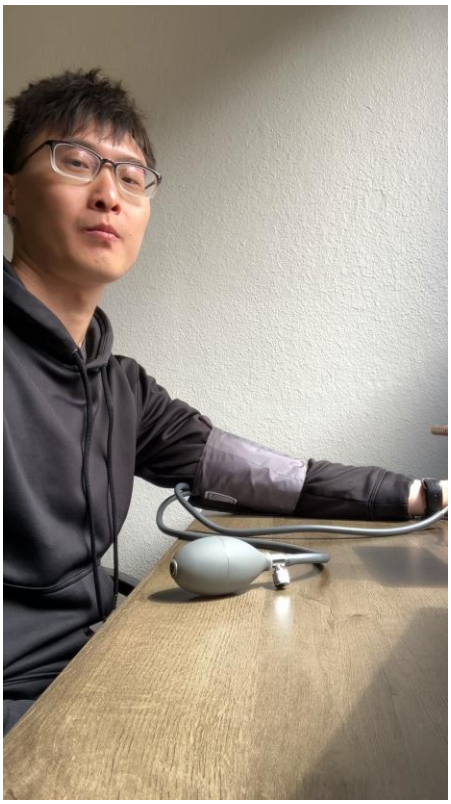
Once the device and the EFR App are connected, at every underflow interrupt the temperature and pressure data are fetched from respective sensors and the received data is displayed on LCD display and reported to the mobile application. As for the heart rate and SpO2 rate, if a finger is detected then both the heart rate and SpO2 rate are also displayed on LCD and sent over to the mobile application; otherwise no data transfers will happen for them. The following are the screenshots from EFR, LCD and LOG.

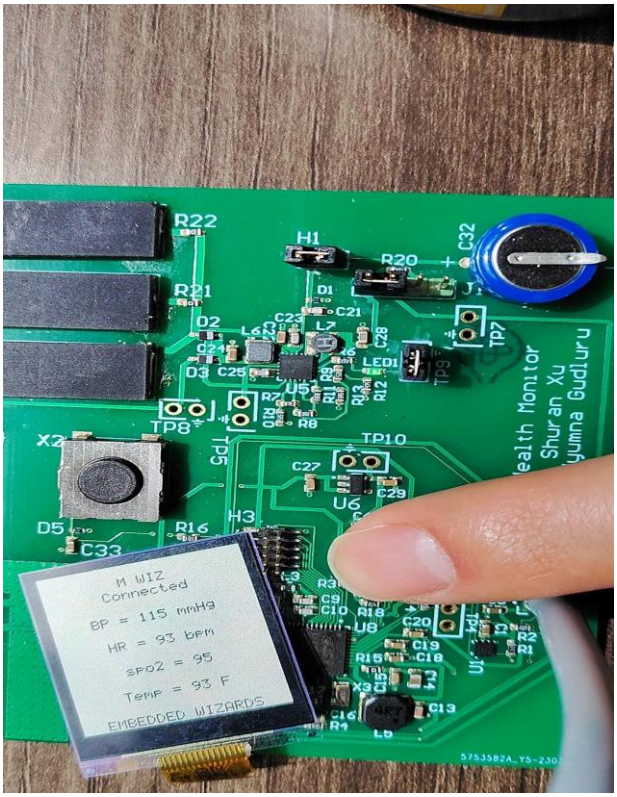






The following screenshot shows Shuran, being the user, experiencing the application with the medical setup. As can be seen below, the pressure cuff was worn around the left arm and the pressure tube was connected to the input side of the micro pressure sensor. Besides, a finger was placed over the MAX sensor to get the data from RED and IR LEDs. With such a setup, the device displayed all sensor data correctly on the LCD display.





The live demonstration is provided as a recorded video which will be submitted along with the final report.

20. Lessons learnt during the project development

Despite that the team learnt tremendously during the project development, there are three key lessons the team learnt out of class and the following table illustrates each of them:

Mistake Made	Lesson Learnt	Name of Student
MAX30101 footprint design error: The team placed the wrong I2C data pin and clock pin and that resulted in damage to the MAX30101 on one of the assembled boards.	The careful and thorough review of the component footprint before schematic design is crucial.	Shuran Xu
Missing push-button: The team forgot to add a push button to the circuit although the team planned to have a push button as the user-trigger element.	Periodic synchronization between the current design and the project plan is necessary since it eliminates any design faults or mistakes happened during the circuit design & implementation. Issues such as component missing could have been easily resolved during the schematic review session if the project plan was checked against the actual schematic design.	Pradyumna
MAX30101 Accuracy issue: the team was having hard time getting accurate heart rate and SpO2 rate accurately during the system integration phase, although accurate data was able to be captured when testing the MAX30101 alone.	I should consider the compatibility issue during the software architectural design to eliminate such issues. The different timing requirements from each sensor result in incompatibility which resulted in MAX30101 providing inaccurate data. So the team learnt that the software architectural design not only involves the logic design such as the system state machine but also module compatibility as well as system-wide configurations to	Shuran Xu

	accommodate all modules.	
Wrong component footprint design: the team was having trouble in designing the correct footprint of a component based on the corresponding datasheet even after going through the PCB footprint tutorial provided on Canvas.	I learnt to take existing footprint design available on the internet as reference design prior to the manual footprint design. In addition, asking for help such as consulting SAs to clarify misunderstanding of the mechanical information in the device datasheet helps prevent errors from happening in the first place.	Shuran Xu
Difficulty in developing a bare-metal driver: the team was having trouble in getting sensors to work with the team's driver implementation, even though the team studied each sensor very carefully prior to the driver implementation.	I learnt that having a reference driver design is tremendously helpful. So the team checked existing solutions implemented in different languages for the target sensor being used in different hardware. The knowledge gained from the reference design study helped the team identify errors made in their own implementations easily.	Pradyumna
Add LEDs for debugging and testing purposes on the system loads such as the SoC and the sensor ICs.	Adding more LED indicators for each system load such as the MCU and other sensors could have made the team much easier when bringing up the sensors.	Pradyumna
Eval Board was not working after soldering GPIO Pins, and the bootloader function was disturbed.	While working with the eval board do not try to solder without consultation to the seniors/peers . The team lost a significant amount of time in getting the eval board with soldered headers functional during the software integration phase.	Shuran Xu
Time management	While working on the project deadline, I missed on one of the updates and had to submit late. Had to focus more on timely reviews and worked on time to make it to an early demo.	Pradyumna

Validation on the referenced hardware platform prior to the validation of the custom board.	I learnt that verifying the code logic on Arduino prior to the validation on the eval board and on the custom board helped the team tremendously to create a benchmark for regular functioning prototype.	Shuran Xu
Could not probe all the voltages on the custom board	Adding more test points to ease probing and debugging would be helpful. Since it is a prototype level board developed, adding test points wouldn't affect a lot and would be helpful for analysis.	Pradyumna
More isolations required on the custom board	I learnt that having more isolation for different systems on the board would have been helpful to test and analyze current and voltages. The lines from the solar cells could have been isolated with a jumper and also the USB power line to PMIC.	Shuran Xu
Documentation & Gantt Chart	I learnt the importance of documentation to help in tracking the project progress and planning the tasks for the next week.	Pradyumna

21. References

1. Digital Patient Monitoring Devices Market to Hit US$ 451.54 (globenewswire.com)
2. EFR32BG13 Blue Gecko Bluetooth® Low Energy SoC Family Data Sheet (silabs.com)
3. TMP117MAIDRVT Texas Instruments | Sensors, Transducers | DigiKey
4. MPRLS0025PA00001A Honeywell Sensing and Productivity Solutions | Sensors, Transducers | DigiKey
5. MAX30101EFD+T Analog Devices Inc./Maxim Integrated | Sensors, Transducers | DigiKey
6. https://www.vishay.com/docs/28409/196hvc.pdf

22. Appendix

22.1 Video Demo

https://drive.google.com/file/d/1rcI-b21eZcvKNFUdQoLEeyXC7AHoNtPF/view?usp=sharing

22.2 Presentation Demo

https://drive.google.com/file/d/1802LGbb8Fxc2NXzH3BLibINGa75orQFY/view?usp=sharing

22.3 GitHub Repo

https://github.com/ShuranXu/LPEDT/tree/final_version