**Firefighter Dashboard Application**



Prashant Gandhi -013712361

Computer Engineering Department, College of Engineering San Jose State University, San Jose, CA 94303

E-mail: prashantshushilbhai.gandhi@sjsu.edu

Table of Contents

[**Introduction** 3](#_Toc16524712)

[**The Innovation Design** 4](#_Toc16524713)

[**Technical Discussion and R&D:** 5](#_Toc16524714)

[**Results, Experiments and Evaluation:** 7](#_Toc16524715)

[**Conclusion** 8](#_Toc16524716)

[**Readme** 8](#_Toc16524717)

[**References:** 9](#_Toc16524718)

# **Introduction**

The proposed product - Firefighter dashboard shall be used for the firefighter’s safety and precautions against dangerous hazards. The present invention relates to sensor data monitoring which provides an automated alarm system for monitoring multiple parameters during firefighting activities and providing appropriate instructions or indications to a firefighter to inform him of a dangerous situation. The aim of this project is to get all the telemetry data of the firefighter team members and send it to the team lead either using serial communication/network protocol. The product being developed for the firefighters and their team members, keeping their requirements in mind the major market segment of the product aims to be Fire Department, but it can be extended to various communities, military and sport adventurous groups, by adding or subtracting some sensor controllers as per their requirements. In the current scenario, firefighters are not embedded with any sensor device and most of the time, firefighters must restore to manual methods of raising the alarm when they are exposed to dangerous environmental situations. These firefighters are covered with thick insulating uniforms which gives them little to no indication about the temperatures rising above the dangerous limits, the heat may get accumulated in these uniforms without any warning to the firefighters, adding further risk to their lives. Hence it becomes difficult to keep track of all the firefighters by the team lead or the firefighter marshal, so with the help of this product, the team lead, or the firefighter marshal shall be able to monitor the firefighter’s health, position and their surrounding areas. As this product is initially targeted for local firefighter department, so after completion of the product, this prototype will be experimented to monitor the product’s efficiency, we can expand the domain to medical applications where blood pressure sensor, sensor to measure glucose, etc can be integrated and monitored by the doctors.

*“Our product promises to deliver safety to our safety providers!!”*

The product provides the following key differentiative features and benefits to the potential customer:

● Firefighter’s location and health monitoring

● Push-to-talk service in extreme situations

● Easy maintenance and low cost

● Interactive Dashboard

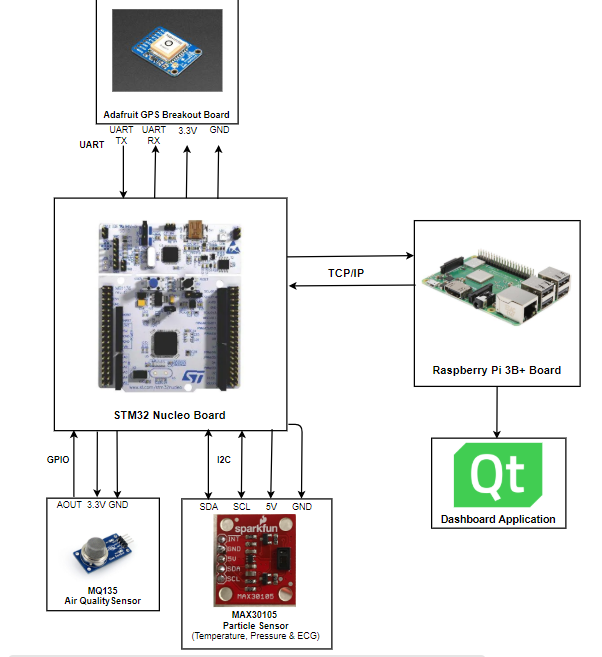
Our product shall consist of the air quality sensor to measure the gas concentration around the firefighter, where STM32 controller shall take the concentration value and warn about the hazards to the firefighter marshal, if the data exceeds certain limits. Temperature and ECG values are obtained from particle sensor to measure temperature around firefighter and heartbeat of the firefighter. The controller shall notify the firefighter marshal about the temperature exceeding above the dangerous limits. These data will be monitored by the marshal with the help of QT application, which gets the data over a network. The push-to-talk service shall be implemented directly using a microphone and a speaker.

Figure 1 System Block Diagram

**Related Work:**

The existing competitive technology allows firefighters to carry the wearable device and an additional walkie-talkie. Sometimes carrying this additional device may seem difficult to manage during the extreme fire situations. To avoid this shortcoming, our proposed product shall implement a built-in push-to-talk feature, which will prevent the firefighters to carry an additional device. This feature will help firefighters to talk to each other and firefighter marshal at ease. Another technological hurdle that these current states of the art products have includes lack of providing the position of the firefighters. Our proposed product will help the firefighter marshal to monitor the GPS coordinates of the team and accordingly provide the instructions to those who are stuck in dangerous surroundings using push-to-talk service. The GPS coordinates can also be monitored over dashboard in real time by the marshal, who guides other firefighters about the whereabouts of each other. In the worst-case scenario, if the firefighter stops responding or acknowledging, the firefighter marshal can summon a rescue team for the firefighter in danger based on their GPS coordinates in real time scenario. Our proposed product involved interfacing STM32 board with Raspberry Pi 3 B+, implementing a push-to-talk service using Raspberry Pi and designing interactive GUI considering QT framework as a platform. Due to such a low scale implementation, our total product cost shall be less than the existing market products. In case of any pitfall in the device, the faulty controllers and sensor modules are easily replaceable aiding low and easy product maintenance. After successful completion of this product, the firefighting team lead shall monitor the telemetry data and accordingly provide instructions to its team members using push-to-talk service preventing any life casualty/injury.

# **The Innovation Design**

Innovation in the design is to implement in-built push-to-talk service and get heartbeat of each firefighter. To design in-built push-to-talk service, I have used b-l475e-iot01a discovery board, who has in-built microphone. I have also worked on MAX30105 particle sensor to get heartbeat. To get heartbeat data from particle sensor, necessary calculation and C code is available online. So, my focus was to implement code for b-l475e-iot01a discovery board to get microphone data and interface particle sensor with b-l475e-iot01a discovery board.

* **b-l475e-iot01a discovery board microphone:**

b-l475e-iot01a discovery board comes with two digital omnidirectional MEMS microphones (MP34DT01). It uses DFSDM (Delta Filter for Sigma-Delta Modulation), which is new type of peripheral available in stm32 boards to convert analog signals to digital signals. Major challenge is, there is no tutorial or example code available, which can describe how to configure MEMS microphone available in discovery board. So, it took few days to figure out how to configure DFSDM peripheral and get data from microphone. After doing some research, I found tutorial for STM32, which describes how to configure DFSDM peripheral and how to get data from MEMS microphone. Second challenge was to send microphone’s data over Wi-Fi as data come from MEMS is 32-bit signed integer and we can only send 8-bit unsigned integer. To overcome this problem, member who was working on W-Fi type casted data to char and tried to send. We have successfully developed push-to-talk service in raspberry pi.

* **MAX30105 particle sensor interfacing with b-l475e-iot01a discovery board**

To get heartbeat form MAX30105 particle sensor, we must implement Peripheral Beat Amplitude Algorithm on IR data. MAX30105 particle sensor uses I2C communication protocol. Key challenge was to implement code for I2C communication between sensor and board. I follow the datasheet and first tried to read part ID of the sensor, which I successfully got. Then I configured sensor register to get IR data only. But data which was coming from the sensor was garbage. I followed online tutorials and C code, which is freely available on sparkfun for Arduino, but still I did not get sensible data. To overcome this problem, I run C code on Arduino, as library is available for MAX30105 in Arduino and I got data perfectly. So, I interfaced MAX30105 particle sensor with Arduino and sent data from Arduino to b-l475e-iot01a discovery board using UART. This data is sent to raspberry pi over Wi-Fi. We have implemented dashboard using raspberry pi and connected LCD to display dashboard. Raspberry pi successfully receiving data and shows data on dashboard.

# **Technical Discussion and R&D:**

* **DFSDM (Delta Filter for Sigma-Delta Modulation):**

DFSDM is new digital peripheral available in STM32 boards. It is same as standard ADC, but the difference is analog part is outside microcontroller. Here, DFSDM is inside the microcontroller and it represents digital part. As analog part is outside the microcontroller DFSDM and analog part is connected by fast serial interface [[3]](https://www.st.com/content/ccc/resource/training/technical/product_training/96/b6/2b/ea/72/3f/4e/d5/STM32L4_System_DFSDM.pdf/files/STM32L4_System_DFSDM.pdf/jcr:content/translations/en.STM32L4_System_DFSDM.pdf).

Here, DFSDM peripheral will collect PDM data and convert them into PCM data. b-l475e-iot01a discovery board is equipped with two omnidirectional MEMS microphones (MP34DT01). It requires external clock

in the range of 1MHz to 3.5 MHz to operate proper operation. Microphone is connected to the following pins of board.

* PE7 – DFSDM1\_DATIN2
* PE9 – DFSDM1\_CKOUT

The digital audio output from the microphone is coded in PDM and is connected PE7. Parameters for our system is as follow:

1. DFSDM Clock: 80 MHz (System Clock)
2. Microphone input clock: 2 MHz
3. Output Sampling Frequency: 8KHz
4. Resolution of output Signal: 24 bits (Signed)

I have used STM32CubeMX to configure and generate code for Truestudio. Below are necessary steps to configure DFSDM peripheral.

1. Search for STM32l475VG and select discovery board.
2. Go to **Computing** and select **DFSDM1**. This will open tab with configuration options.
3. Disable channel 1 and select “PDM/SPI input from cha2 and intern clock” in Channel 2.
4. In configuration tab:

Filter 0 -> Regular Channel Selection -> Channel 2

DMA setting -> Add -> DFSDM\_FLT0 -> Mode: circular

Output clock -> Divider=40

Filter 0 -> Filter parameter -> Fosr = 250

Filter 0 -> Regular Channel Selection -> fast mode = DMA mode = enable

This will automatically initialize necessary parameters in code. Once code is generated, we need to create buffer of size 1024. In this buffer, we will collect PDM data coming from microphone. Now to we need to convert this PDM data to PCM, for that we will create another buffer of same size. We will first convert first half of the buffer and let fill another half buffer and vice versa. To convert PDM to PCM we need to shift data 8 position right side.

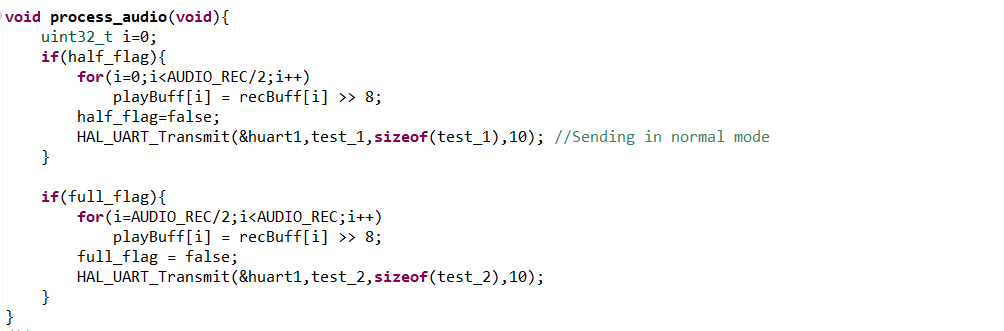


Figure 2 Code for PDM to PCM conversion

* **MAX30105 particle sensor:**

We will begin our communication by first sending sensor address which is 0x57. Then will try to read part-ID to ensure our communication is established. Then we setup our sensor’s registers based on our requirements. Steps to setup particle sensors:

1. Sample Average = 4
2. Mode = MultiLED
3. ADC Range = 16384 (62.5pA per LSB)
4. Sample rate = 50

Set mode to read on IR data. After that we will apply PBA algorithm in continuous loop. The theory behind this is, when we put over finger over particle sensor, infrared signals get reflected from our blood cells and captured by senor. PBA algorithm this then apply to that data to calculate heartrate. PBA algorithm code readily available on internet.

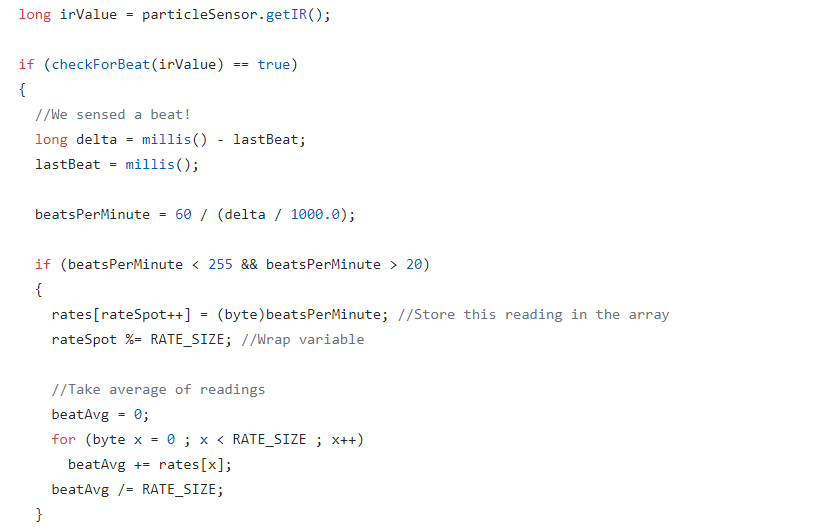


Figure 3 PBA Algorithm

* **Benefits**

As we are using on board microphone, firefighter does not need to carry devices like Walkie-Talkie. So, our project will reduce the components. Also, we are measuring heartbeats of all firefighters and sending those data to their leader. So, this will help leader to monitor firefighter’s health and in emergency He/she can talk necessary steps before it is too late.

# **Results, Experiments and Evaluation:**

* **DFSDM buffer output:**

Figure 2 is simulation results of PDM data and PCM data. In the figure recBuff represents PDM data. We shift this data 8 position right to convert it into PCM. playBuff is converted PCM data of mirophone.

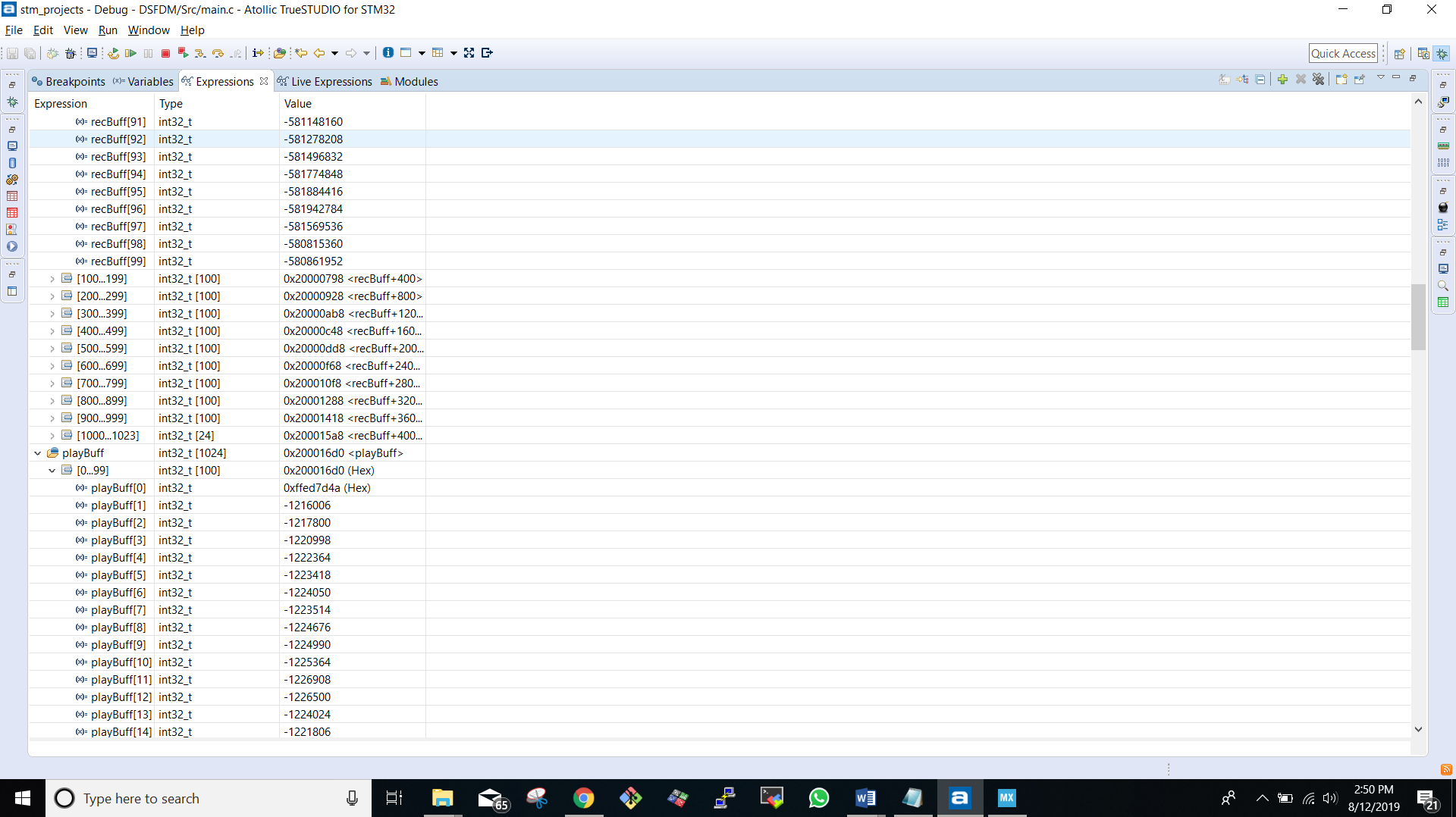


Figure 4 PDM and PCM data

* **Particle sensor data:**

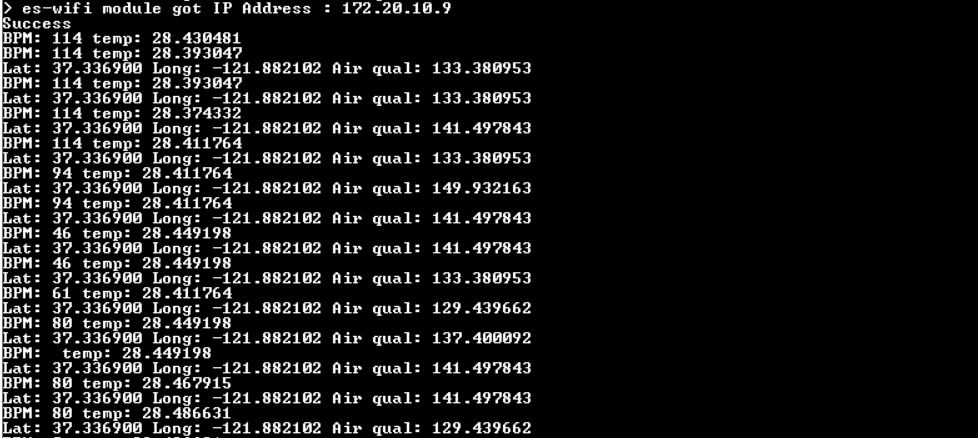


Figure 5 BPM - Calculated Heartrate

* **Firefighter Dashboard Screen:**

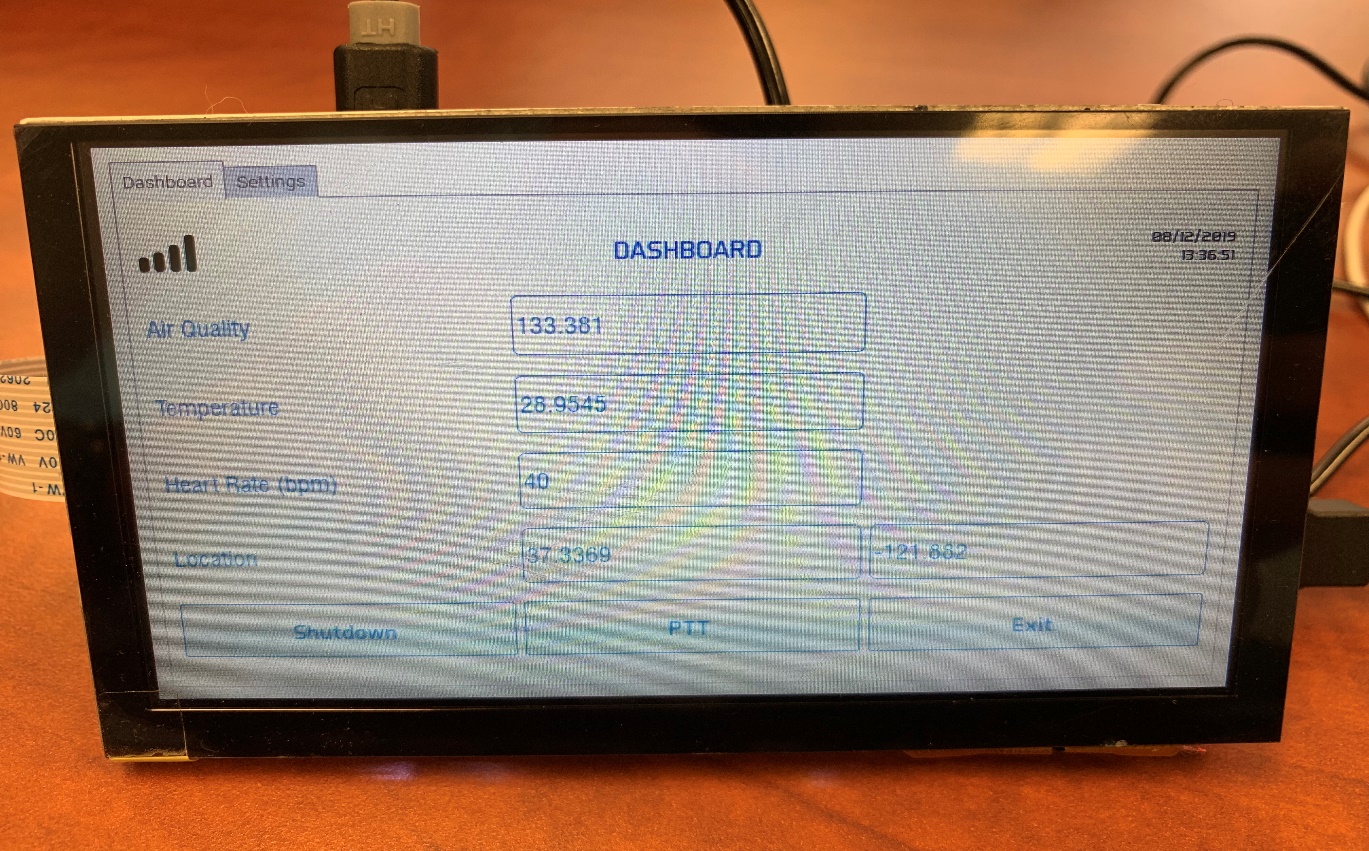
****

Figure 6 Dashboard Screen

# **Conclusion**

Safety of firefighters is our priority. This can be done by equipped firefighters with necessary electronics. But at the same time this should have less weight. We have developed the product for the firefighters and their team members, keeping their requirements in mind the major market segment of the product aims to be Fire Department, but it can be extended to various communities, military and sport adventurous groups, by adding or subtracting some sensor controllers as per their requirements. Also, weight of this project is very less. This product consists real-time location facility, because of that firefighter marshal will be able to monitor all its team members’ location. Also, this product is embedded with sensors like temperature, heartbeat sensor and air quality sensor, so if situation near any firefighter becomes harsh, firefighter marshal will come to know and guide its team members, evacuate the area. we can expand the domain to medical applications where blood pressure sensor, sensor to measure glucose, etc can be integrated and monitored by the doctors.

# **Readme**

* Link for code: <https://drive.google.com/drive/u/1/folders/1czfnS6Wtnl_XwculLWopqk-ULxiiHj2Y>
* Tutorial for DFSDM: <https://www.youtube.com/watch?v=MdDqVeIGhec>
* Heartrate PAB calculation code: <https://github.com/sparkfun/SparkFun_MAX3010x_Sensor_Library/blob/master/src/heartRate.cpp>
* MAX30105 Datasheet: <https://cdn.sparkfun.com/assets/learn_tutorials/5/7/7/MAX30105_3.pdf>
* Discovery Board Datasheet: <https://www.st.com/content/ccc/resource/technical/document/user_manual/group0/b1/b8/7a/f2/f7/8d/4b/6b/DM00347848/files/DM00347848.pdf/jcr:content/translations/en.DM00347848.pdf>

# **References:**

1. <https://learn.sparkfun.com/tutorials/max30105-particle-and-pulse-ox-sensor-hookup-guide/example-5---heartrate>
2. <https://www.youtube.com/watch?v=MdDqVeIGhec>
3. <https://www.st.com/content/ccc/resource/training/technical/product_training/96/b6/2b/ea/72/3f/4e/d5/STM32L4_System_DFSDM.pdf/files/STM32L4_System_DFSDM.pdf/jcr:content/translations/en.STM32L4_System_DFSDM.pdf>