**REPORT**

**AESD: PROJECT -I**

**Project Partners**

Raj Lavingia

Yash Gupte

**INDEX**

*Topic Page No.*

1. Project Overview ……………………………………………………………………… 2
2. Thread Description …………………………………………………………………... 2
3. Block Diagram ………………………………………………………………………….. 3
4. BIST Description ………………………………………………………………………. 4
5. API Description ………………………………………………………………………… 5
6. References ………………………………………………………………………………. .

**Project Overview**

Design and Implement **“smart” environment** monitoring device using two sensors called temperature sensor and light sensor.

**Temperature sensor (TMP102)** and **light sensor(APDS-9301)** will be used for constantly monitoring the temperature of the device and for detection / sensing of the light in the area respectively.

The project will be implemented on **BeagleBone Green Board**.

The sensors will be interfaced with I2C communication method and continuously log data into a log file of both the sensors.

For this project, we have decided to compile 1 **Parent Thread** which is the **Main Task**.

The main task is spawned into 5 different child threads according to the requirement mentioned in the project.

Note :  There is only 1 bus where read and write of data takes place and since there is only 1 bus common for both the sensors a need arises for the forking of threads.

Note : **It is very important to exit the program gracefully**. In order to do that, the Main thread will continuously monitor all the following threads on a certain time period. If the threads are not able to respond( even 1 of them) in a given time interval then the Main thread will exit gracefully closing all the child threads.

For making such tasks, we have planned to use **pthreads.**

**Thread Description**

1. **Temperature Sensor Task** - This thread will be used for continuously for monitoring the Temperature in either Celsius or Fahrenheit through an I2C bus communication connected between a Microcontroller and the Temperature Sensor (TM102)
2. **Light Sensor Task** -    This thread will be used for continuously for monitoring the Light and getting the lux values through an I2C bus communication connected between a Microcontroller and the Light Sensor (APDS-9301)
3. **Socket Task** - In this thread, we will have a TCP server running continuously and which accepts connection from the client which can ask for various requests like sensor data.
4. **Logging task** - This thread is very much useful for logging the data with a timestamp into a file which has been acquired from both the sensors on a continuous basis.

**Block Diagram**

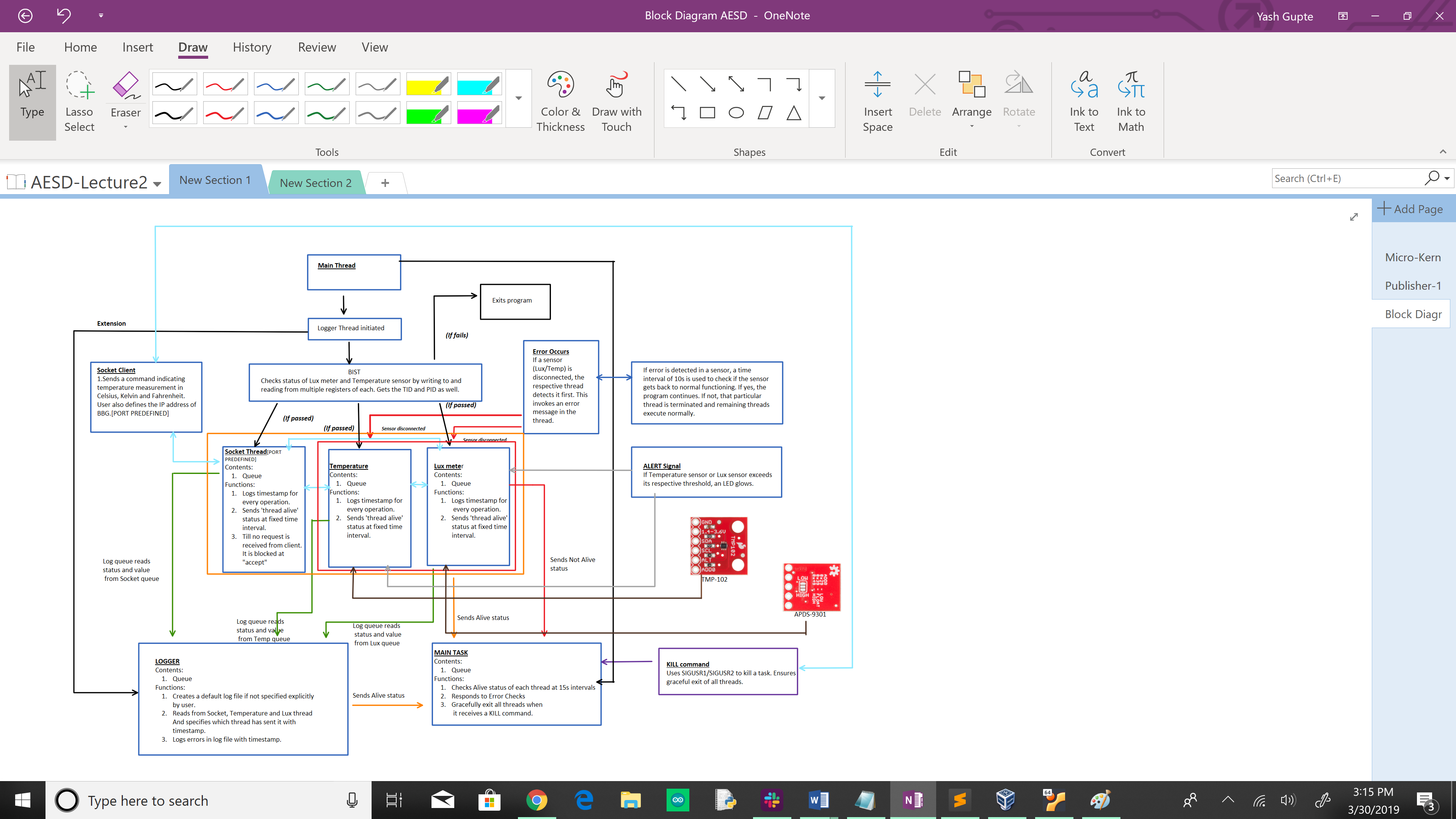


Fig. Block Diagram

**BIST Description**

Build In Self Test is used to check if Lux sensor and Temperature sensor are working. This is tested by using I2C communication read from register and write to register methods.

If BIST passes for both temperature and lux sensor, only then a separate thread for each lux sensor, temperature sensor and socket is created. If BIST fails for even one of the sensors, main thread exits gracefully without any retries.

BIST steps for Temperature sensor:

* 1. Temperature sensor is initialized.
  2. Base registers are set.
  3. All the remaining registers are written.
  4. All the registers are read to confirm written values.
  5. Base register is reset in order to measure temperature.
  6. Once, all these tests are done, “BIST passed message from temperature sensor” is put on the logger queue which prints to text file.

BIST steps for Lux sensor:

* 1. Lux sensor is initialized.
  2. Base registers are written.
  3. All the remaining registers are written.
  4. All the registers are read to confirm written values.
  5. Base register is set to 0X03. To measure light intensity.
  6. Once, all these tests are done, “BIST passed message from lux sensor” is put on the logger queue which prints to text file.

**API Description**

Temperature Thread:

1. uint8\_t base\_reg\_write(uint8\_t\* buffer\_value,int buffer\_bytes);// Common base register for writing to Thigh,Tlow and config.
2. void \* TempThread(void \* args);// Individual temp queue is made and communicated with all the remaning threads. Also checks for kill command from USR1 or USR2.If socket client asks for temperature in any unit, this thread returns that specific unit (C,F,K)
3. uint8\_t base\_reg\_read(uint8\_t \*buffer\_value,int buffer\_bytes);// Common base register for reading Thigh,Tlow and config.
4. uint8\_t main\_write\_register(uint8\_t register\_addr, uint16\_t desired\_val); //Register addr must be between(0x01-0x03).Common base funciton for all registers (writing purpose), sends a buffer to the register through file descriptor.
5. uint8\_t main\_read\_register(uint8\_t register\_addr, uint8\_t\* desired\_val); //Register addr must be between(0x01-0x03).Common base funciton for all registers (writing purpose), sends a buffer to the register through file descriptor. Reads whole buffer through file-descriptor and prints two 8 bit numbers.(LSB,MSB)
6. uint8\_t all\_registers\_check(void); // Thigh and Tlow are written and read according to threshold set.
7. uint8\_t config\_register\_temperature(void); // Bits of config reg are written and read for purposes like shutdown mode, fault bits - em mode and cr(conversion rate) mode
8. uint8\_t get\_temp(float \*t\_data); // Calculations are done in order to get a temp value in float in C,K or F from adc. Also checks id measured temo is above or below threshold.
9. uint8\_t temp\_initial\_sensor(void); // Starts temp sensor by openening I2C bus and checks slave addresses of I2C.
10. uint8\_t BIST\_Temp\_Check(void); // BIST is performed.

**References**

1. Queue: <https://www.geeksforgeeks.org/queue-set-1introduction-and-array-implementation/>
2. Socket: <https://www.geeksforgeeks.org/socket-programming-cc/>
3. Mutex: <https://www.geeksforgeeks.org/mutex-lock-for-linux-thread-synchronization/>
4. Errno: <http://man7.org/linux/man-pages/man3/errno.3.html>
5. TMP-102 datasheet: <http://www.ti.com/lit/ds/symlink/tmp102.pdf>
6. APDS-9301 datasheet: <http://www.farnell.com/datasheets/1816958.pdf>