ECEN 5013-FINAL PROJECT REPORT

TEAM MEMBERS

GAURAV GANDHI

SANJANA KALYANAPPAGOL

1. INTRODUCTION

The main objective of the project was to use various on-board peripherals of FFRDM-KL25Zand deliver an application.

In line with that we used peripherals such as ADC, I2C, UART, SPI, RTC, DAC as well as interfacing external devices such as Capacitive touch input, Accelerometer, Temperature sensor, SD card to produce a desired output on motor and RGB led.

Low power design considerations are be implemented in this project to save energy. In addition, message structure is also incorporated for implementing user commands.

We provided Commands for getting sensor data, changing speed of motor, starting particular peripheral in our application.

Binary data logging to the SD card via SPI and optional uart logging is provided for monitoring status of all the peripherals. User can start or stop UART data logging through command structure to reduce processor load.

RTC is used as a time stamp for logged data as well as it is used as scheduler for handling events such as sd card logging , uart logging, data acquisition.

Git Link of the code:

https://github.com/gauravgandhi70/Gaurav_repo/tree/master/projet4

2. SYSTEM BLOCK DIAGRAM

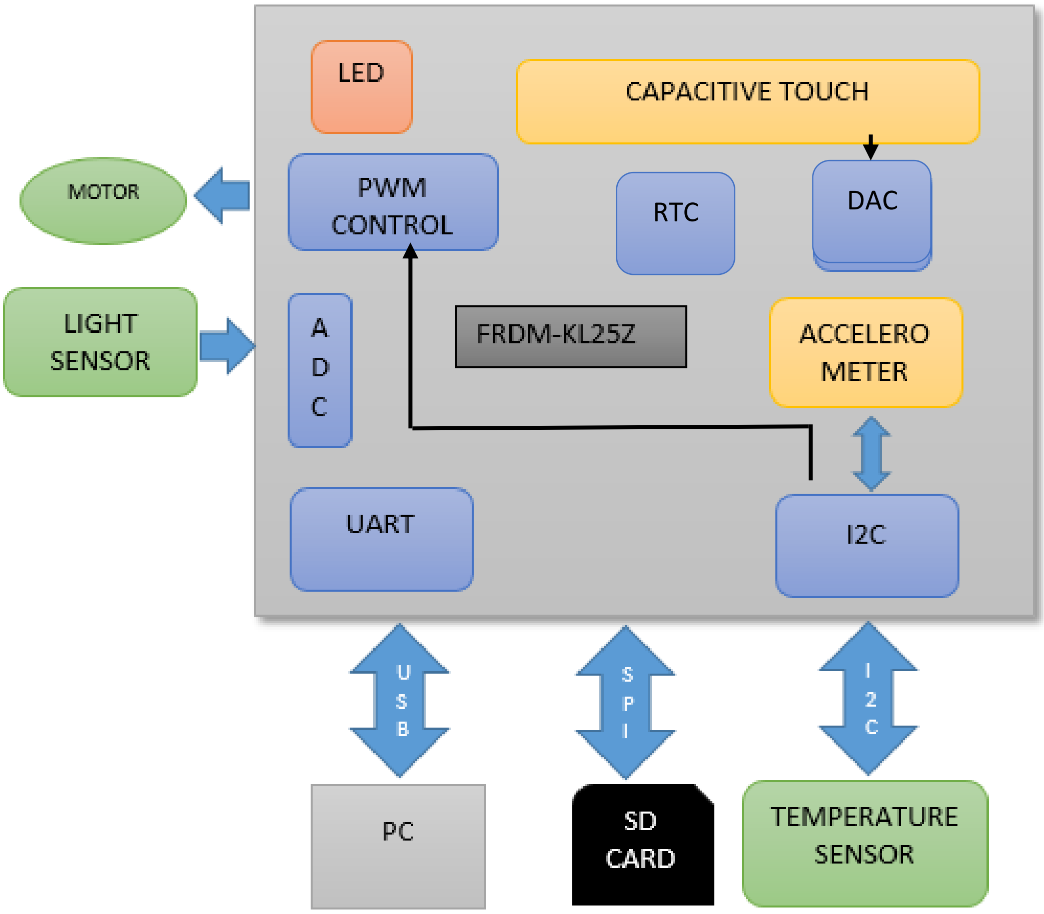


Fig 1

HARDWAR ELEMENTS:

1. SD CARD
2. Digital Temperature Sensor
3. Analog Light Sensor
4. Motor
5. Accelerometer
6. Capacitive touch

Software Elements:

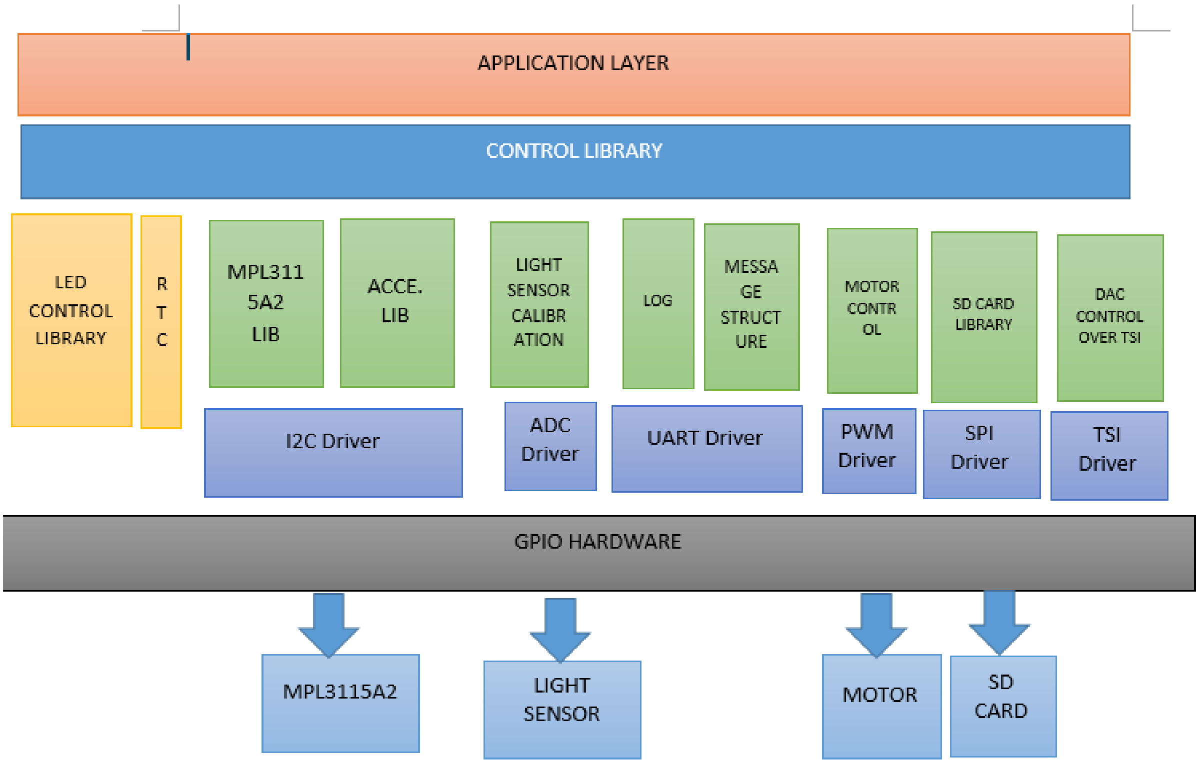
1. SPI Driver
2. I2C Driver
3. UART Driver
4. SD card library
5. Sensor calibration
6. TSI Library
7. DAC Control
8. ADC library
9. Accelerometer lib
10. RTC Control
11. Motor Control library
12. Timing profiling using PIT timer

Pin Map:

Pin	Function
PTC8	I2c(Accelerometer, Temp Sensor)
PTC9	I2c(Accelerometer, Temp Sensor)
PTB0	ADC_CH8(Light Sensor)
PTC4	CS(SD Card)
PTC5	SCK(SD Card)
PTC6	MOSI(SD Card)
PTC7	MISO(SD Card)
PTC1	DAC output
PTE20	PWM out

3. Software Architecture

The software design is done in a modular way and using software abstraction method. The layered software diagram is as following:



The various software modules of the project are:

1. Temperature Controlled Motor speed
2. Light Sensing
3. Accelerometer controlled LED
4. Touch controlled DAC & LED
5. Logging (Binary to SD card and UART Logging)
6. Command Structure
7. Scheduling using RTC Interrupt

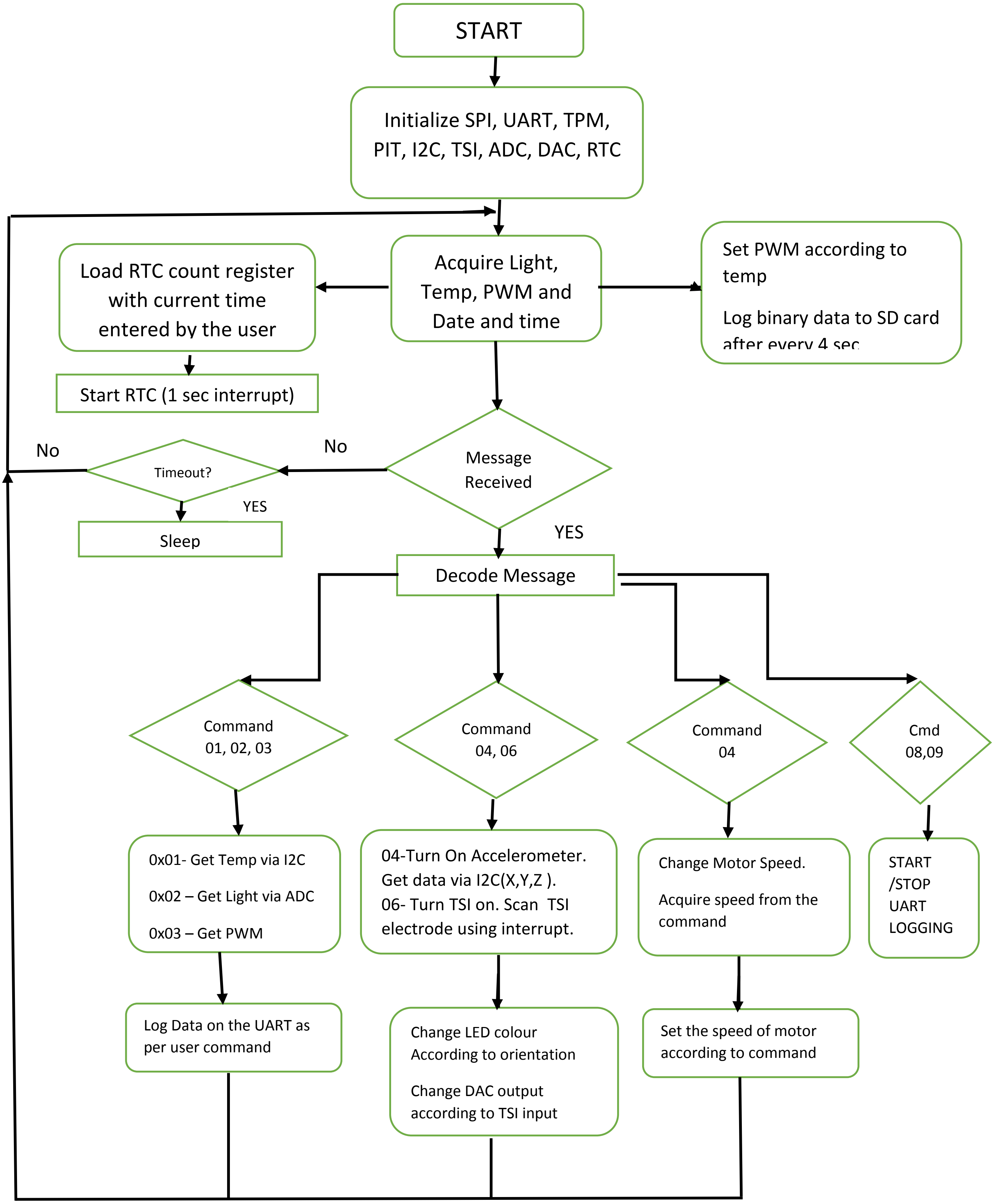


Fig 3. FLOW CHART

Software flow:

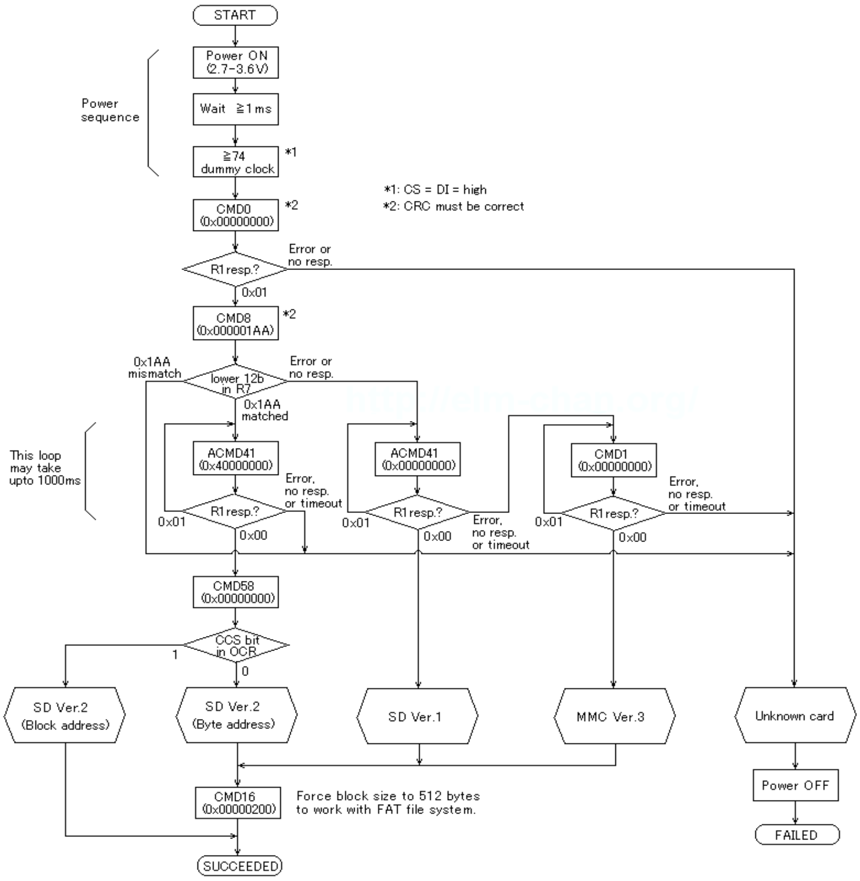
1. As soon in the flow chart: As soon as the system starts all the peripherals are initialized
2. All the data from the sensors is acquired and calibrated and stored.
3. Wait for command from the user. At the same time keep acquiring latest state of all peripherals and sensors. After 4 seconds Log all this data to the SD card. Repeat this process until user sends any command or timeout occurs and processor goes in sleep mode.
4. If user sends any command, act on that command and go to step 2.

Firmware Design for SD card and Accelerometer:

1. SD Card: SD card is used for binary logging. Library was written for the SD card write, Read, Initialization, While initializing we followed initialization sequence to put the SD card in SPI mode. Write is performed in the blocks of 512 bytes. Also multiple block write is implemented for increasing the efficiency of the sd card write.

2. Accelerometer: This on-board sensor uses I2C as a serial communication. The sensor is configured via I2C and also calibrated to give appropriate values. The application of the sensor is to change to different colours on the LED. The accelerometer is configured to indicate X positive axis [1], X negative axis [-1], Y positive axis [1], Y negative axis [-1], Z positive axis [1], Z negative axis [-1].

   In the similar manner temperature sensor is configured via I2C to get required resolution and accuracy. And TPM in PWM mode was linked to this rading to control the speed of the motor.
   ADC was configured to get 10 bit resolution for light sensor.



SD Card initialization : Image source: lm-chan.org/docs/mmc/mmc_e.html

On Board Temperature sensor:

ADC is configured to operate on-board temperature sensor which uses the 26th channel. The ADC is configured for 10 bit resolution and long sample time. The clock is Pre-scaled of 8 for the ADC peripheral. The obtained ADC value is calculated using a transfer function.

$$\mathrm{Temp} = 25 - \left(\left(V_{TEMP} - V_{TEMP25}\right) \div m\right)$$

where:

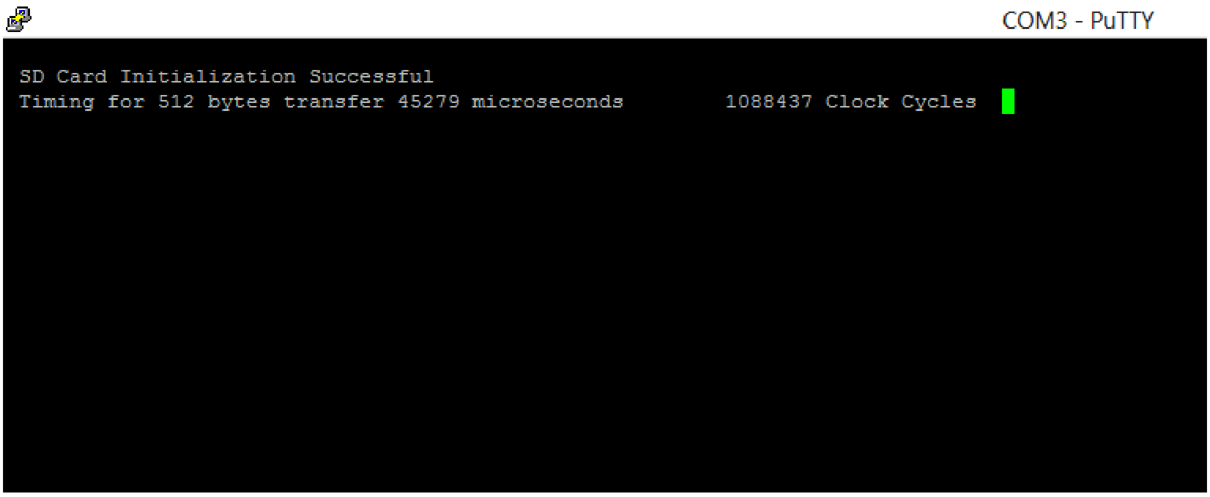
• VTEMP is the voltage of the temperature sensor channel at the ambient temperature.
• VTEMP25 is the voltage of the temperature sensor channel at 25 °C.
• m is referred as temperature sensor slope in the device data sheet. It is the hot or cold voltage versus temperature slope in V/°C.

To handle various events, interrupts were used. They are listed in most important to least (priority based):

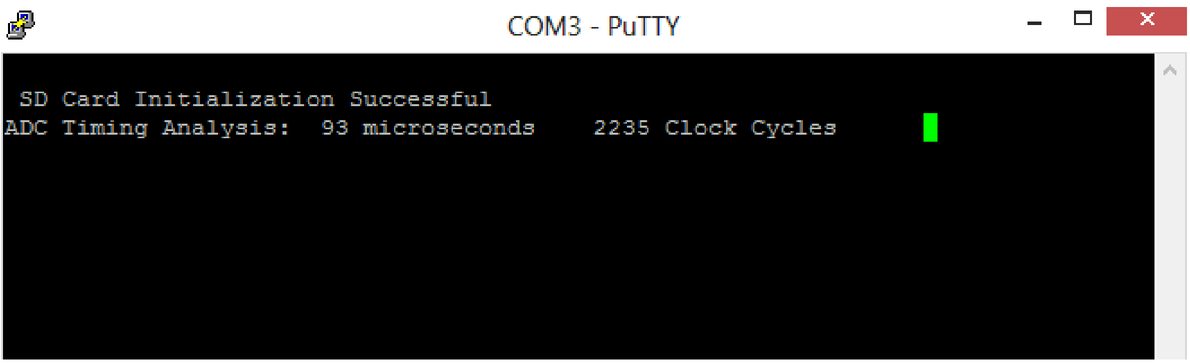
1. RTC seconds interrupt: For providing 1 sec reference interval for the events such as UART logging and SD card logging.
2. TSI Interrupt: To get the reading of the capacitive electrode as soon as the scanning ends
3. UART interrupt
4. DMA Interrupt: To now that dma transfer is complete
5. PIT Interrupt: this interrupt is used for timing profiling. Interrupt occurs after every 1 microseconds. Hence it provides 1 microseconds resolution for profiling

4. Timing analysis and Results

SPI Block write timing analysis:

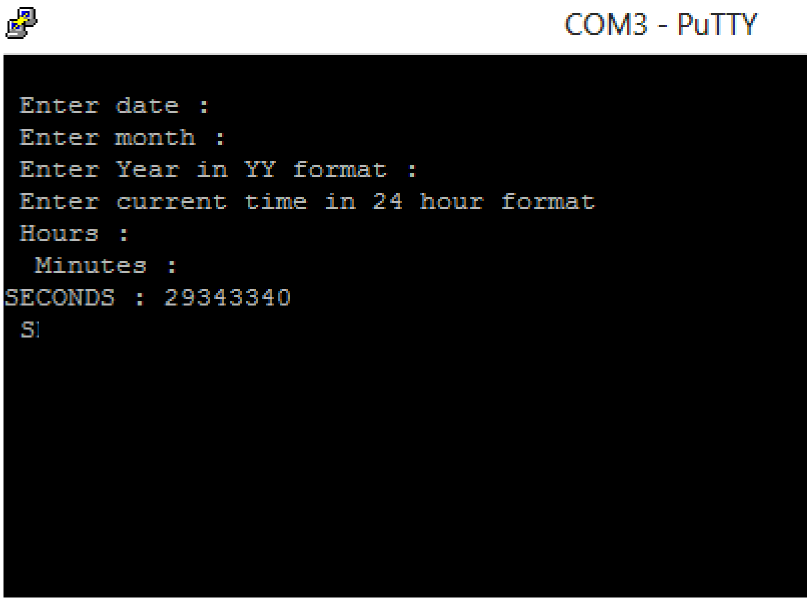


ADC Timing analysis :



Results:

1. RTC Initialization from the user:

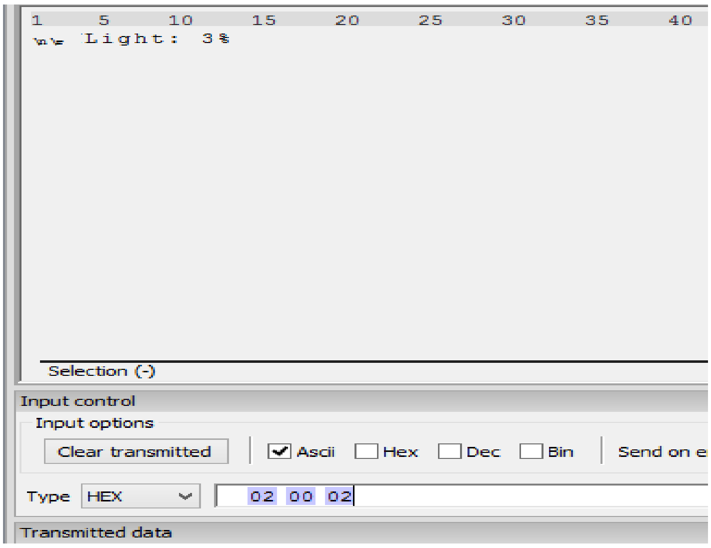


Converting user input into seconds by comparing with 1 jan 2016.

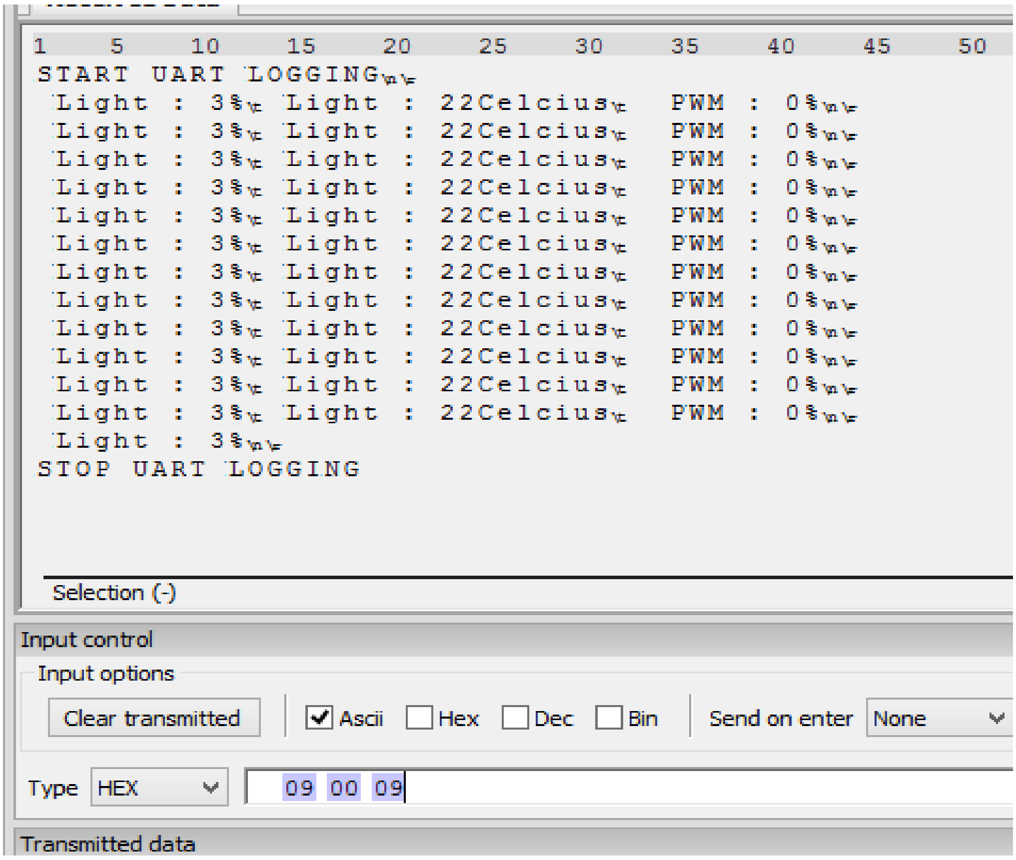
User input is not echoed to the terminal

2. COMMAND STRUCTURE:

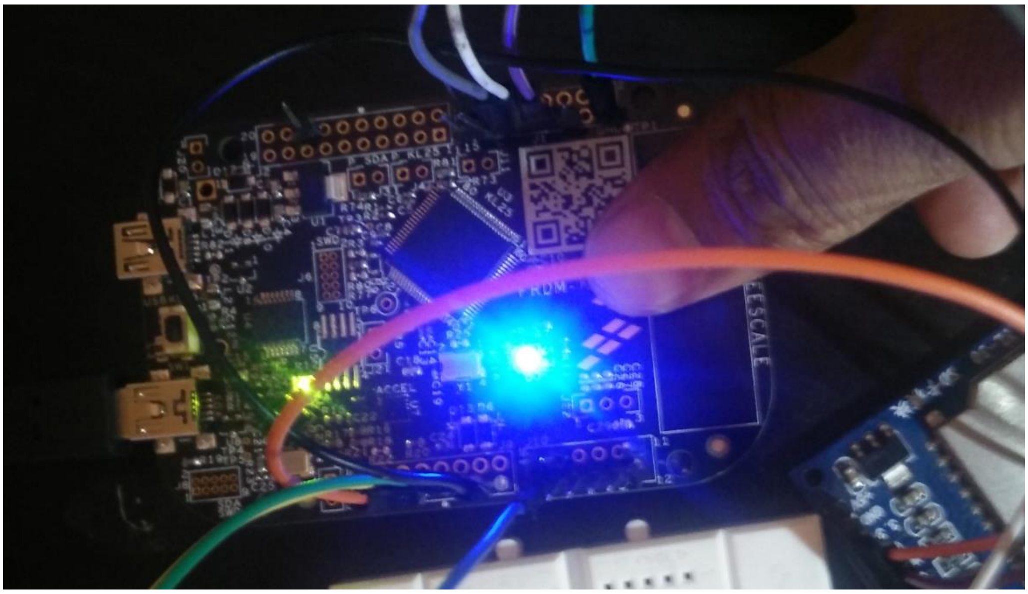
After sending command 0x02 , Light sensor output is displayed on the terminal.

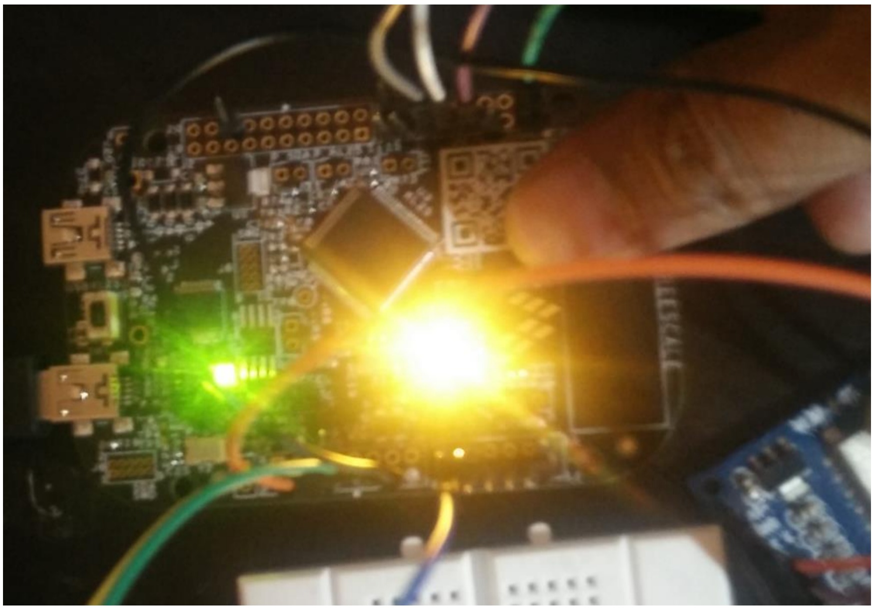


3. UART Logging:



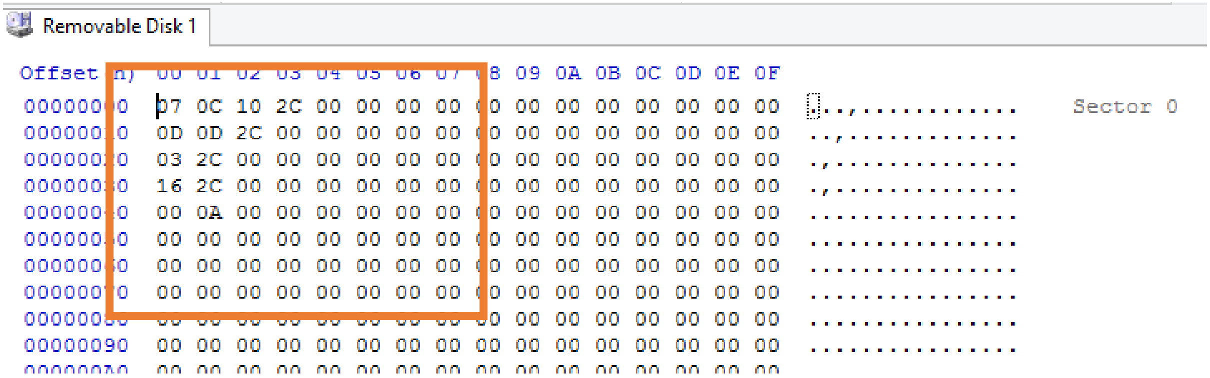
4. TSI Output on LED:





Different colour for different position of the finger on capacitive touch

5. SD Card Data Logging:



Data is written on the sd card in a particular format, as following

Date,

Time,

Light,

Temp

PWM \n

Each new line contains data of the different sensor. Raw data is logged (binary logging). Hence overhead for logging is reduced.

Data is written with time stamp.

5. Difficulties faced during project:

Obtaining Accurate values for accelerometer reading. Calibration ha to be done to obtain the accurate values

Validating the accelerometer data with appropriate orientation. This needed various trial and error methods.

Testing and verifying I2C's SDA and SCL line for accelerometer via logic analyser as the GPIO pins PE24 and PE25 are internally connected to accelerometer.

Obtaining the right transfer function to calculate the temperature after obtaining the values from the ADC.

Finding the correct sequence for the initialization of the SD card.

Writing to the sd card without file system was challenge because behaviour of the card can not be predicted after formatting the card. So aligning our code accordingly was a challenge.

5. CONCLUSION:

I2C was designed to configure accelerometer and external temperature sensor and the colours on the RGB LED changes according to the direction of the accelerometer.

ADC was designed and configured to obtain the temperature from the internal temperature sensor.

SD card logging was successfully implemented and performance of writing can be improved by writing multiple blocks to the sd card.

Use of commands provided user interface to this application.

Also it can be used in low power mode.

Using all aspects of this project, many applications can be developed such as:

Weather monitoring, Temperature controlled fan and many more.

It will be more useful because user interface such as Capacitive touch pad and Command control is included in the project.