 ./

GENESIS - Learning Outcome & Mini-project Summary Report



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **To be Approved** | **Remarks/Revision Details** |
| 1. | 15/04/2021 | Neha Tabassum |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Details**

Contents

[Contents 3](#_Toc69565375)

[Miniproject -1 [Team/Individual] 5](#_Toc69565376)

[Module/s 5](#_Toc69565377)

[Topic and Subtopics 5](#_Toc69565378)

[Objectives & Requirements 7](#_Toc69565379)

[Design 9](#_Toc69565380)

[Test Plan 14](#_Toc69565381)

[Implementation Summary 20](#_Toc69565382)

[Video Summary 21](#_Toc69565383)

[Git Link 21](#_Toc69565384)

[Git Dashboard 21](#_Toc69565385)

[Summary 22](#_Toc69565386)

[Individual Contribution & Highlights 30](#_Toc69565387)

[Summary 30](#_Toc69565388)

[Challenges faced and how were they overcome 30](#_Toc69565389)

[Future Scope (If applicable): NA 30](#_Toc69565390)

[Miniproject -2 [Individual]: Reading Excel Sheet and summarizing the data in the mastersheet 31](#_Toc69565391)

[Module/s: 31](#_Toc69565392)

[Topic and Subtopics 31](#_Toc69565393)

[Objectives & Requirements 31](#_Toc69565394)

[Design 32](#_Toc69565395)

[Test Plan 34](#_Toc69565396)

[Implementation Summary: 35](#_Toc69565397)

[Git Link 36](#_Toc69565398)

[Git Dashboard 36](#_Toc69565399)

[Summary 36](#_Toc69565400)

[Individual Contribution & Highlights: 37](#_Toc69565401)

[Challenges faced and how were they overcome 37](#_Toc69565402)

[Miniproject -3 [Team] – Embedded C 38](#_Toc69565403)

[Module/s 38](#_Toc69565404)

[Topic and Subtopics 38](#_Toc69565405)

[Objectives & Requirements 38](#_Toc69565406)

[Design 40](#_Toc69565407)

[Test Plan 42](#_Toc69565408)

[Implementation Summary 43](#_Toc69565409)

[Video Summary 44](#_Toc69565410)

[Git Link 44](#_Toc69565411)

[Git Dashboard 44](#_Toc69565412)

[Summary 45](#_Toc69565413)

[Challenges faced and how were they overcome 45](#_Toc69565414)

[Future Scope (If applicable): NA 45](#_Toc69565415)

[Miniproject -4 [Individual]: Embedded linux and Kernel Programming 46](#_Toc69565416)

[Module/s 46](#_Toc69565417)

[Topic and Subtopics 46](#_Toc69565418)

[Objectives & Requirements 46](#_Toc69565419)

[Test Plan 48](#_Toc69565420)

[Video Summary 49](#_Toc69565421)

[Git Link 49](#_Toc69565422)

[Git Dashboard 49](#_Toc69565423)

[Individual Contribution & Highlights 49](#_Toc69565424)

[Summary 49](#_Toc69565425)

[Challenges faced and how were they overcome 50](#_Toc69565426)

[Future Scope (If applicable):- NA 50](#_Toc69565427)

# Miniproject -1 [Team/Individual]

## Module/s

Module linked to the miniproject is SDLC.

### Topic and Subtopics

Below is the list of core topics and subtopics being implemented:

**SDLC (Software Development Life Cycle) :-**

SDLC is a process followed by a software project, within a software organization. Contains a detailed description of how you can improve, maintain, modify and modify or improve a particular software. The life cycle describes how to improve software quality and the overall development process.

Phase 1: Requirement and Planning Analysis

Analysis is the most important and fundamental phase in the SDLC. It is done by senior team members with input from customers, industry experts, market surveys and sales department. This information is also used to plan the basic course of the project and to conduct feasibility studies in the technical, conservation and operational areas.

Phase 2: Requirement Definition

Once the requirements has been taken the next step is to clearly define and document the product requirements and is approved for customers or market analysts. This is done through an SRS (Software Requirement Specification) document that contains all the product requirements that will be created and developed during the life cycle of the project.

Phase 3: Product Design

SRS is the trust of product manufacturers to come up with the best technology for product development. In accordance with the requirements set out in SRS, usually more than one method of product design is proposed and documented in the DDS - Design Document Specification. An excellent design approach is chosen that clearly defines all product building modules and the representation of communication and data flow through external and third party modules.

Step 4: Building or developing a product

At this stage of the SDLC, real development begins and the product is built. Application code is generated per DDS. If construction is done in a detailed and systematic way, coding can be accomplished without much hassle. Various programming languages ​​such as C, C ++, PHP, Java and Pascal are used for encoding the design.

Step 5: Product testing

This section is usually the basis of all categories as modern models of SDLC, testing activities are very involved in all sections of SDLC.

Section 6: Market Delivery and Care

Once the product has been tested and ready for shipment it is officially released from the relevant markets.

SWOT Analysis:

SWOT analysis is used to assess the organization’s current position before going to any new strategy.

SWOT stands for Strengths, Weaknesses, Opportunities, and Threats, so SWOT Analysis is a way to evaluate four business aspects. SWOT analysis is used to make the most of what a person has gained, benefiting their organization. It can also reduce the likelihood of failure, and eliminating the risks that would otherwise be unknown.

Strength

It is the thing that an organization does best, or in a way that separates the organization from its competitors.

Weaknesses

Now is the time to look at organization weaknesses. Be honest! SWOT analysis will be useful only if one collects all the necessary information. Therefore, it is better to be realistic, and to deal with unpleasant facts very quickly.

Opportunities

Opportunities to open up or opportunities for something good to happen, but one needs to seek them out through the organization.

Threats

Threats include anything that could adversely affect the business from the outside, such as procurement problems, changes in market demand, or a shortage of employees.

Unified Modelling Language (UML) Diagrams:

UML is a common language for interpreting, visualizing, constructing and transcribing software archeology. UML drawings are designed not only for engineers but also for business users, the general public, and anyone with an interest in understanding the system. There are two broad categories of it and they are further sub-divided as follows:

Structural Diagrams:

* Class Diagram
* Object Diagram
* Component Diagram
* Deployment Diagram
* Profile Diagram
* Package Diagram
* Composite Structure Diagram

Behavioral Diagrams:

* Use Case Diagram
* State Machine Diagram
* Activity Diagram
* Communication Diagram
* Sequence Diagram
* Interaction Overview Diagram
* Timing Diagram

## Objectives & Requirements

The designed product has all the necessary features required by the undergraduates and postgraduates students including scholars. The High level requirements include arithmetic operations, decimals, trigonometric functions, nth root, power of n, fractions, percentage, logarithms, exponentials, number conversions etc. The low level requirements of the product includes addition, subtraction, multiplication, division under arithmetic operation, decimal upto 8 digits, trigonometric functions with square root and radians. Under number conversions there are the features like Binary to decimal conversion, Decimal to binary conversion, Decimal to octal conversion. Calculators also includes the functions of database management, higher accuracy, wider and has smart touch, solar cell operations, battery charging and is waterproof.

Requirement:-

**High Level Requirement (HLR):**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirements | Description | Status |
| HLR\_01 | Arithmetic operations | Addition, Subtraction, Multiplication, Division, Remainder, GCD | Implemented |
| HLR\_02 | Percentage and power | Calculates percentage of one number wrt to another and raised to power of a number to the number provided. | Implemented |
| HLR\_03 | Trigonometric functions | Sine, cosine, tangent functions | Implemented |
| HLR\_04 | Number Conversions | Converts Binary to Decimal, Decimal to Binary and Decimal to Octal. | Implemented |
| HLR\_05 | Area Calculation | Calculates area of Square, Rectangle, Triangle and circle. | Implemented |

**Low Level Requirement (LLR):-**

1. Arithmetic Operations:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirements | Description | Status |
| LLR\_01 | Addition | Takes two numbers as an input and adds them | Implemented |
| LLR\_02 | Subtraction | Takes two numbers as an input and subtracts them | Implemented |
| LLR\_03 | Multiply | Takes two numbers as an input and multiplies them | Implemented |
| LLR\_04 | Divide | Taking two numbers as an input and divides them | Implemented |
| LLR\_05 | Percentage | Takes two numbers as an input and calculate percentage of one number with the other | Implemented |
| LLR\_06 | Greatest Common Divisor(GCD) | Takes two numbers and calculate their GCD | Implemented |

2. Percentage and Power:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirements | Description | Status |
| LLR\_01 | Percentage | Calculates percentage of two numbers | Implemented |
| LLR\_02 | Power | Calculates power of one number raised to the other | Implemented |

3. Trigonometric Functions:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirements | Description | Status |
| LLR\_01 | Sine Function | Calculate sine function of a number in both degree and radian | Implemented |
| LLR\_02 | Cosine Function | Calculate cosine function of a number in both degree and radian | Implemented |
| LLR\_03 | Tangent Function | Calculate tan function of a number in both degree and radian | Implemented |

4. Number conversions:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirements | Description | Status |
| LLR\_01 | Binary Function | Convert binary number to decimal number | Implemented |
| LLR\_02 | Decimal Function | Convert Decimal number to binary number | Implemented |
| LLR\_03 | Octal Function | Convert Decimal to octal number | Implemented |

5. Area Calculator:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirements | Description | Status |
| LLR\_01 | Square Function | Calculate area of a square | Implemented |
| LLR\_02 | Rectangle Function | Calculate area of a rectangle | Implemented |
| LLR\_03 | Circle Function | Calculate area of a Circle | Implemented |
| LLR\_04 | Triangle Function | Calculate area of a Triangle | Implemented |

## Design

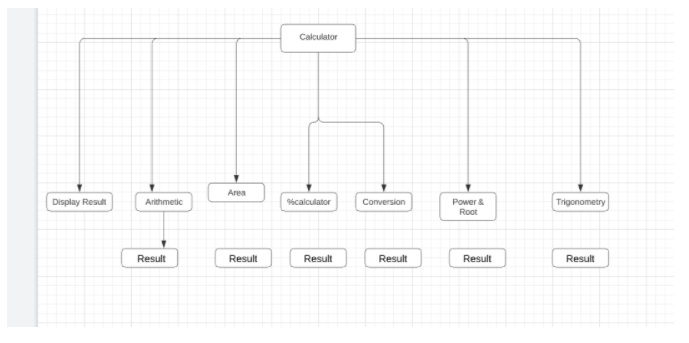


Figure : High Level Diagram of Calculator

Structural Diagram:-

Arithmetic Operation:

System Level:

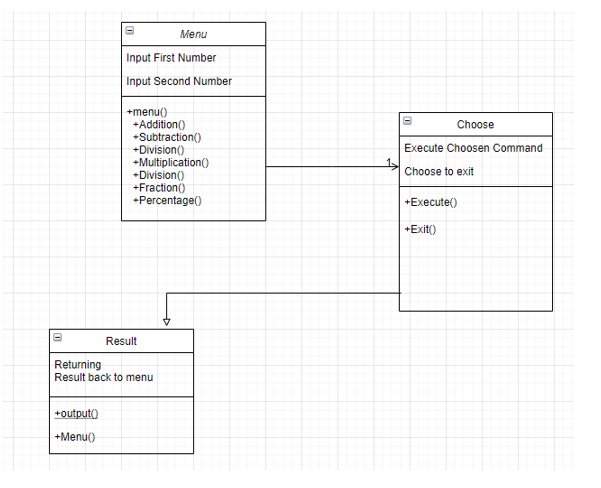


Figure : System Level Structural Diagram of Arithmetic Operation

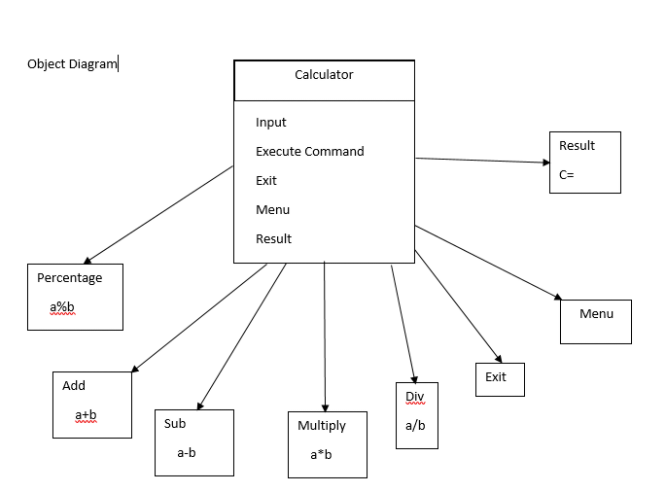


Figure : System Level Object Diagram of Arithmetic operation

Subsystem Level:

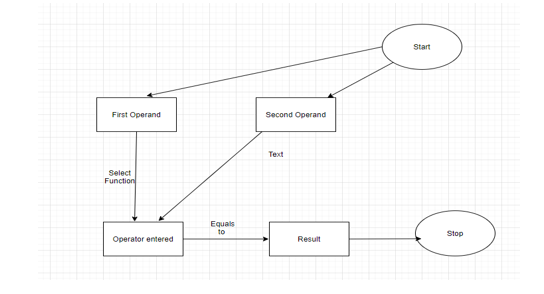


Figure : Subsystem level Structural Diagram of Arithmetic Operation

Trigonometry Function:

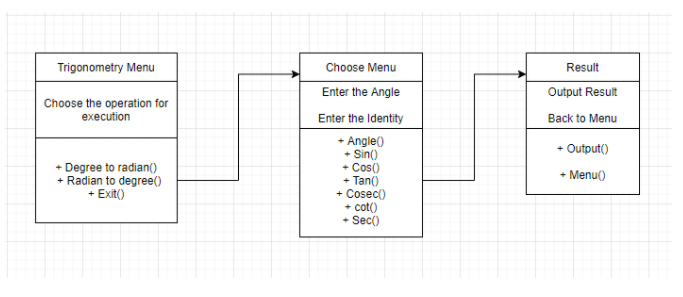


Figure : Structural Diagram of Trigonometric Function

Number Conversion:

System Level:

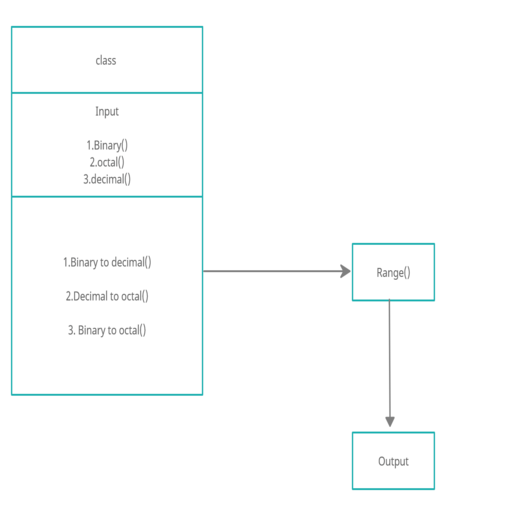


Figure : System Level Structural Diagram of Number conversion

Subsystem-level

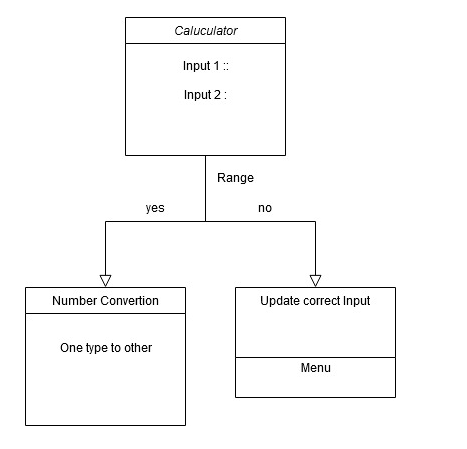


Figure : Subsystem Level Structural Diagram of Number Conversion

Area Calculator:

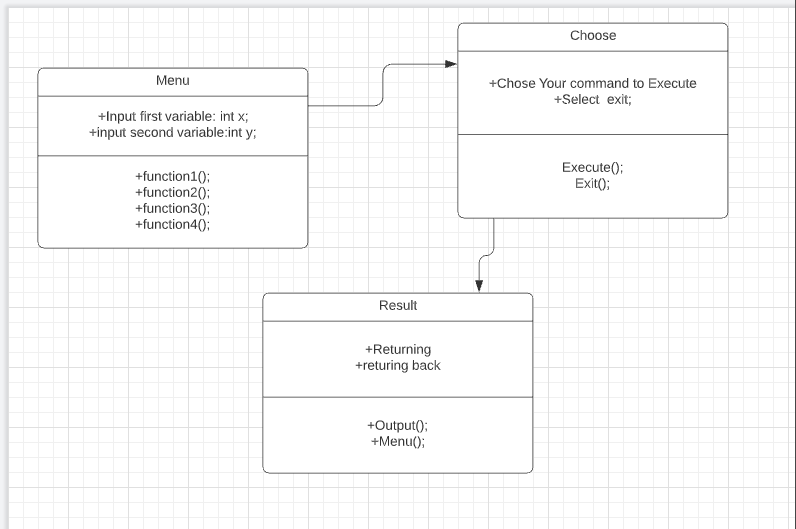


Figure : Structural Diagram of Area Calculator

Behavioral Diagram:

Arithmetic Operation:

System Level:

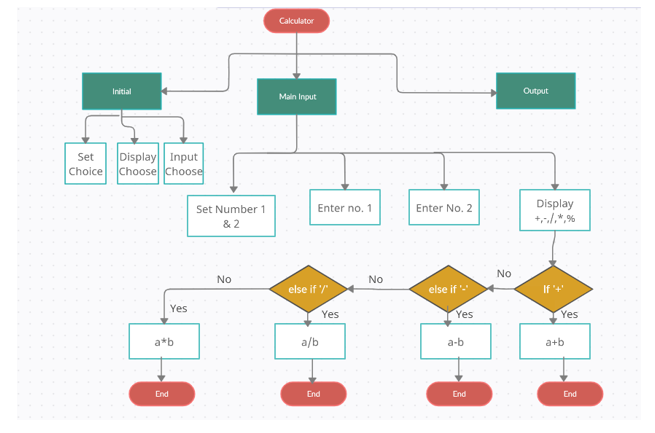


Figure : System Level Behavioural Diagram of Arithmetic Operation

Subsystem-Level:

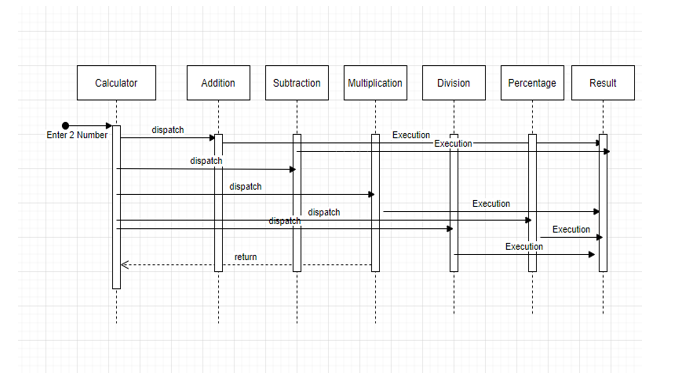


Figure : Subsystem Level Behavioral Diagram of Arithmetic Operation

Number Conversion:

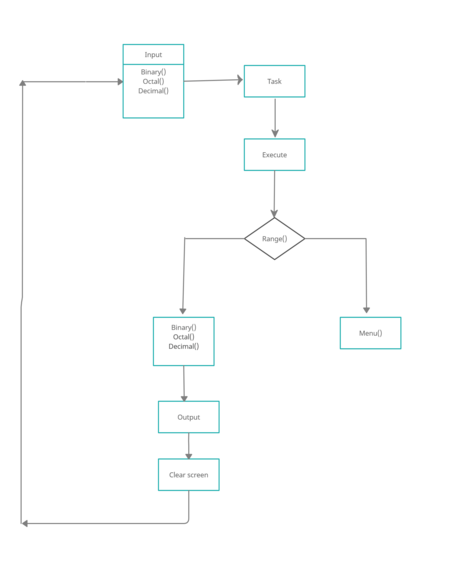


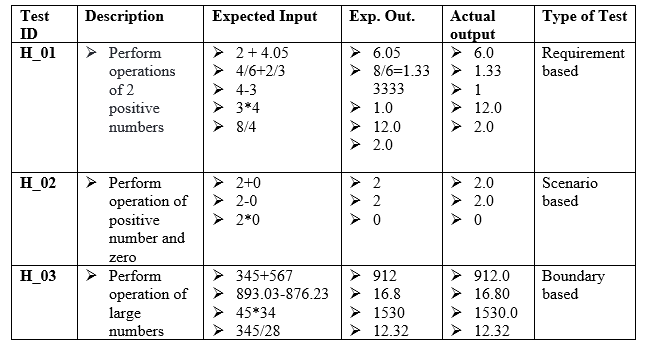
Figure : Behavioural Diagram of Number Conversion.

## Test Plan

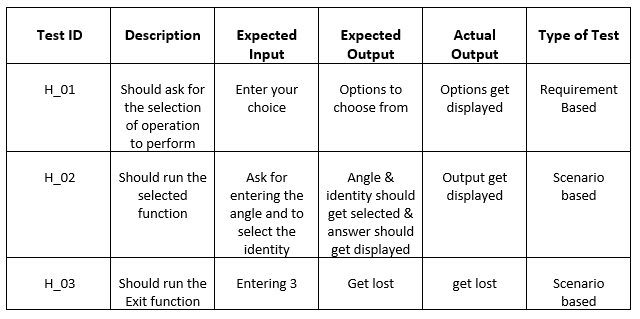
“Integration level and unit level in the template”

High Level Test Plan:

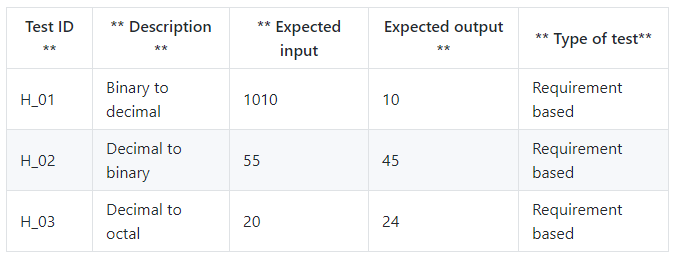
Arithmetic Operation



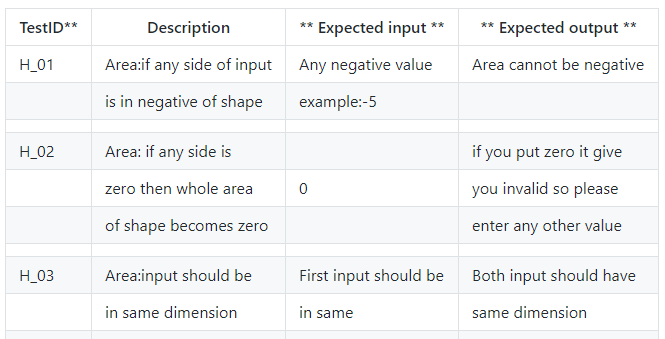
Trigonometric Operation



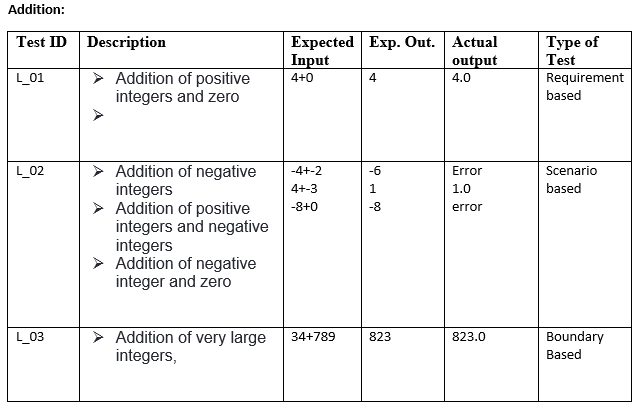
Number Conversion

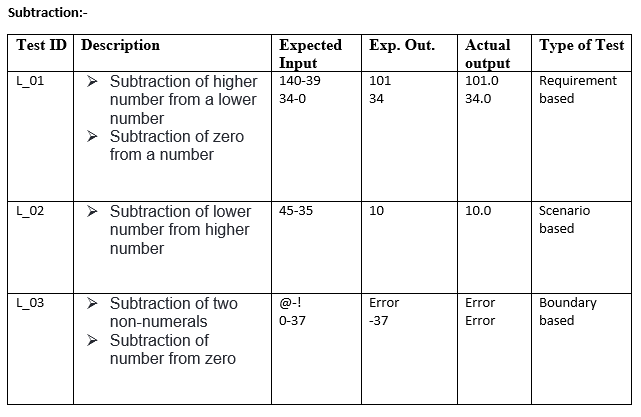


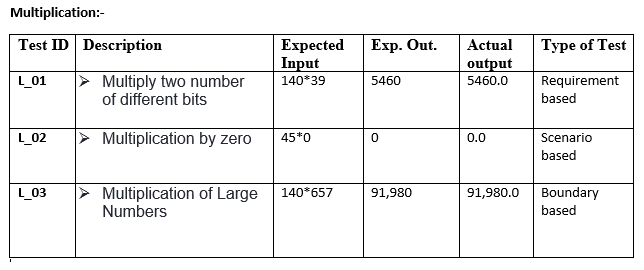
Area Calculation

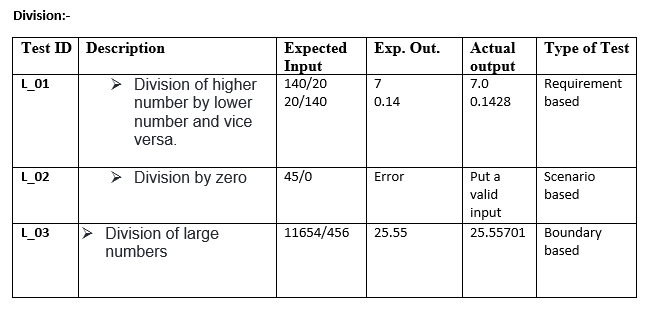


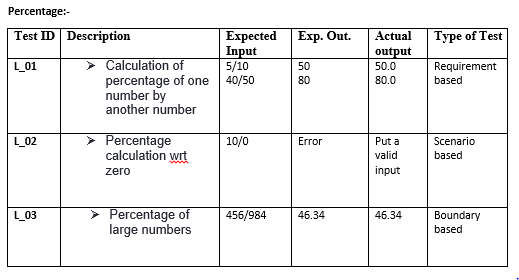
Low Level Test Plan:

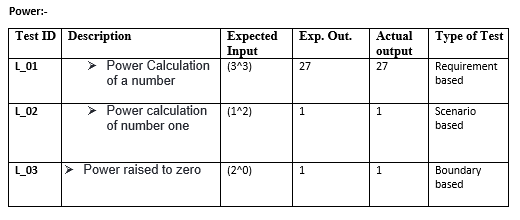


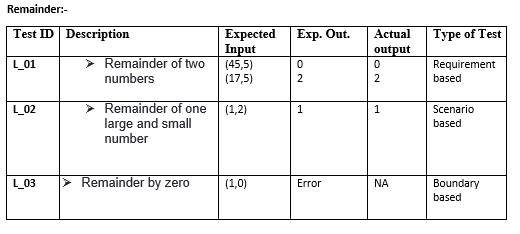


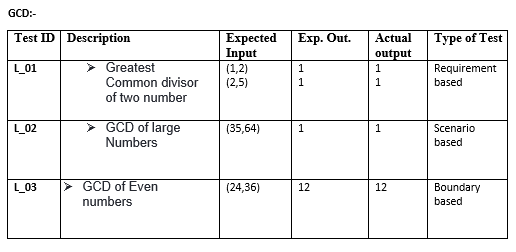




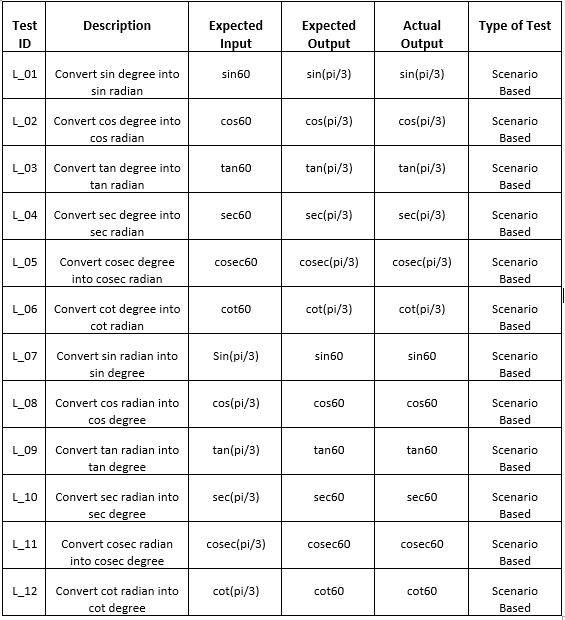




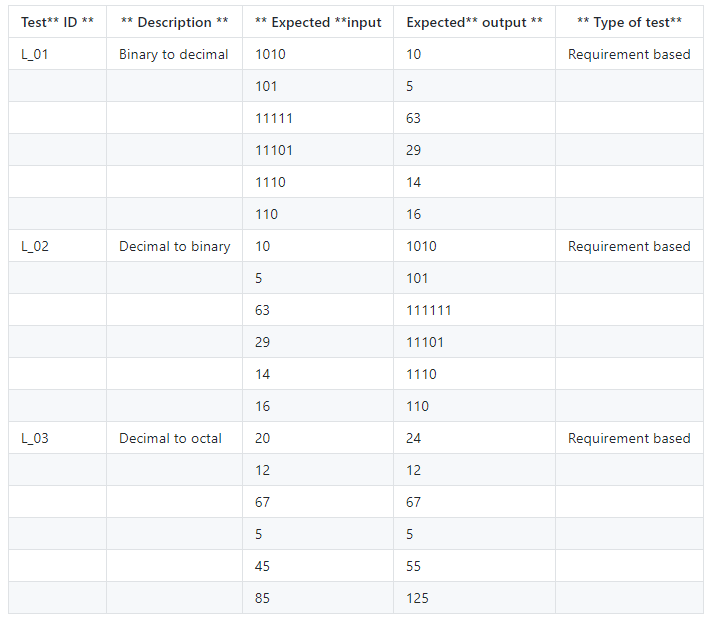




**Trigonometry:-**



**Number Conversion:-**



## Implementation Summary

The implementation part consist of software implementation of the designed product known as calculator as per the specification mentioned in requirements and design. The folder consist of inc folder which consist of various header files (.h) of various section of designed calculator. It also contains src folder which is having the source files (i.e .c file). The test folder contains the test case implementation of the designed product. It also contains the Makefile which is used to build, run and clean all the multiple files and check for the test cases that are formulated according to the requirement specified.

The source file contains the implementation of the following functions:

* Arithmetic operation (Addition, Subtraction, Multiplication, Division, Remainder, GCD)
* Percentage and power operation
* Trigonometric functions (Radian and Degree)
* Number conversion function (Binary, Decimal, Octal)
* Area Calculator operation (Rectangle, Square, Triangle, Circle)

Note: The GitHub private repo should be documented (Readme.md files at each folder level)

Ensure code quality and clean code and description practices

Mandatory: To add the GitHub user - **stepin654321** as a contributor to the repo”

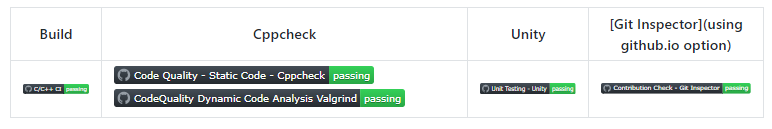
### Video Summary

“Please upload a short video on the repo for the walkthrough of the project (Team/Individual) less than 7min and less than 30MB File Size. Start is the Standard opening slide with title of miniproject + Team members followed by the walkthrough ”

### Git Link

[https:/github.com/99003781/N8-Calculator.git](https://github.com/99003781/N8-Calculator.git)

### Git Dashboard





### Summary

The whole project begins with the research on different calculators The Research has been divided on the basis of cost and features of different calculators. After the research the requirements for the customized calculator was documented in the form high level and low level requirements. Once the requirements were decided the same was implemented using UML diagrams. Once the UML diagrams were made for visual understanding of the design then the software implementaion was started using c code. Once the header files and source files were written then the same was tested using the test cases. Different files were combined using a single Makefile which build and run the code as a single code. After software implementation of codes CPP and unity check was done inn order to generate various badges for the correctness of the code.

#### Git inspector summary

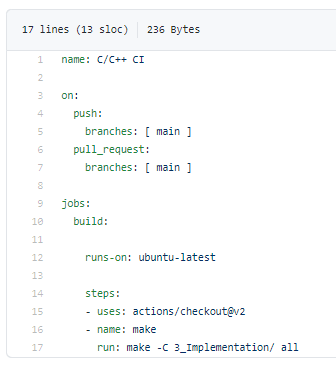
“In linux install gitinspector and Run the command –

gitinspector -H -l -m -T -w -r --grading --format=html > gitinsp.html

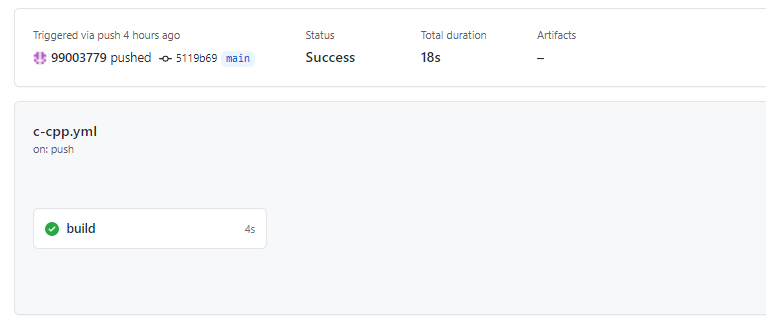
and upload the same to your repo and paste the snapshot in the report”

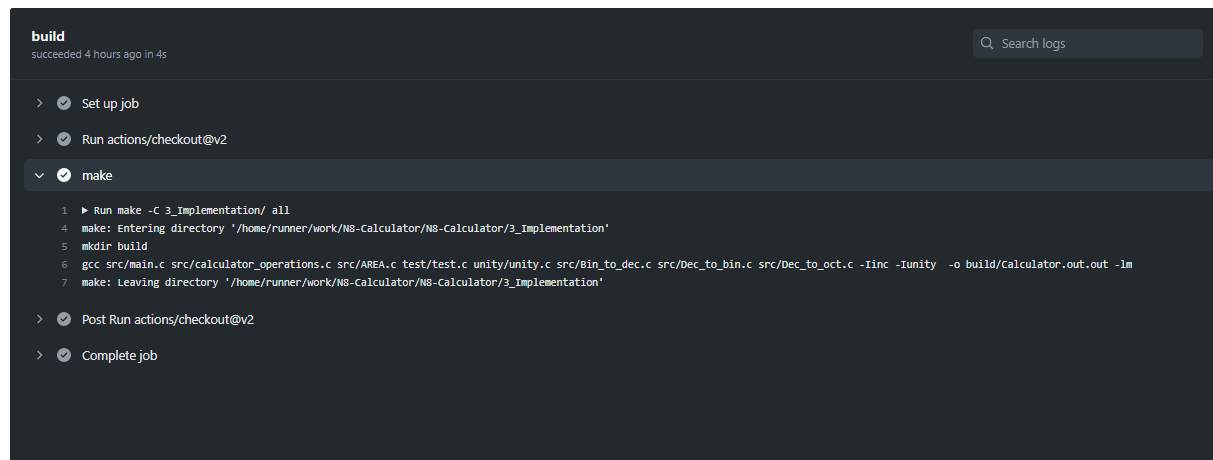
#### Build

Setup Done:



Outcome:





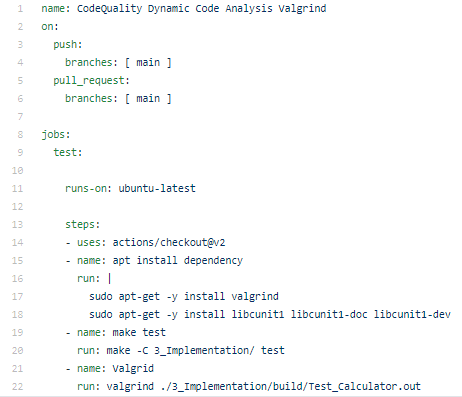
#### Code quality and Issues or Bug Tracking

Setup:-

Static Analysis:

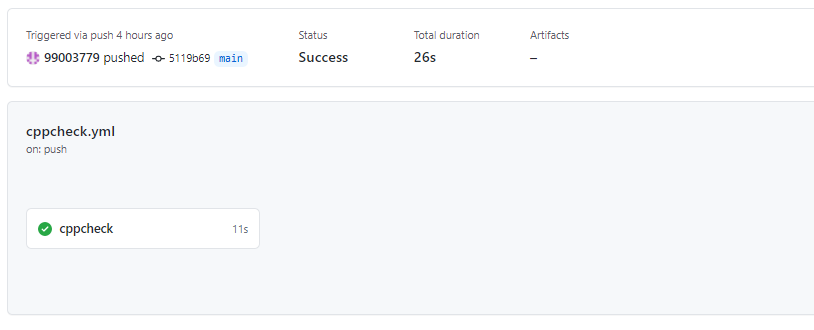


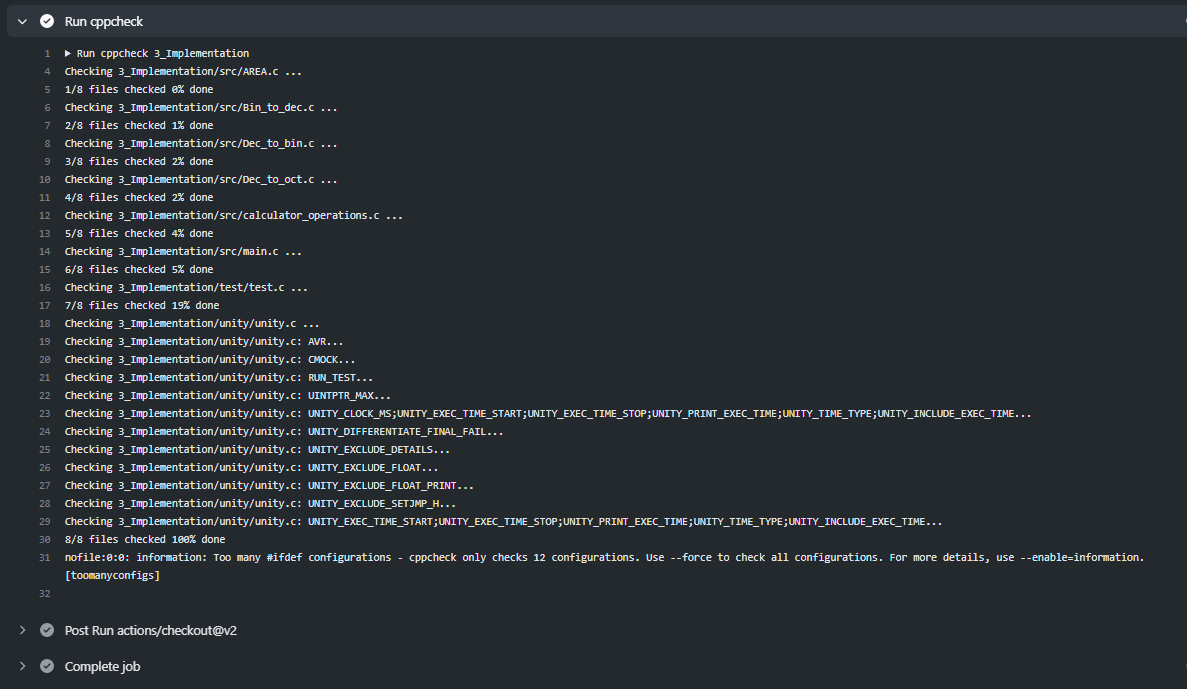
Dynamic Analysis:



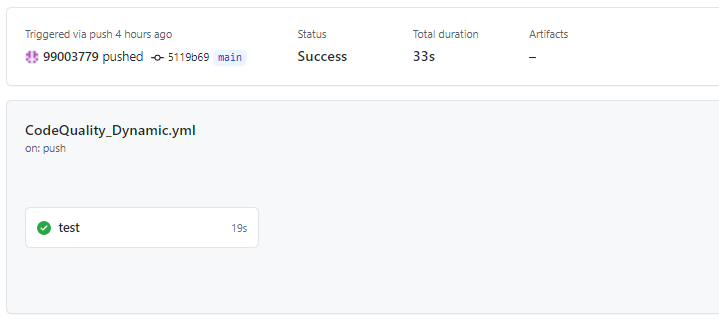
Outcome:

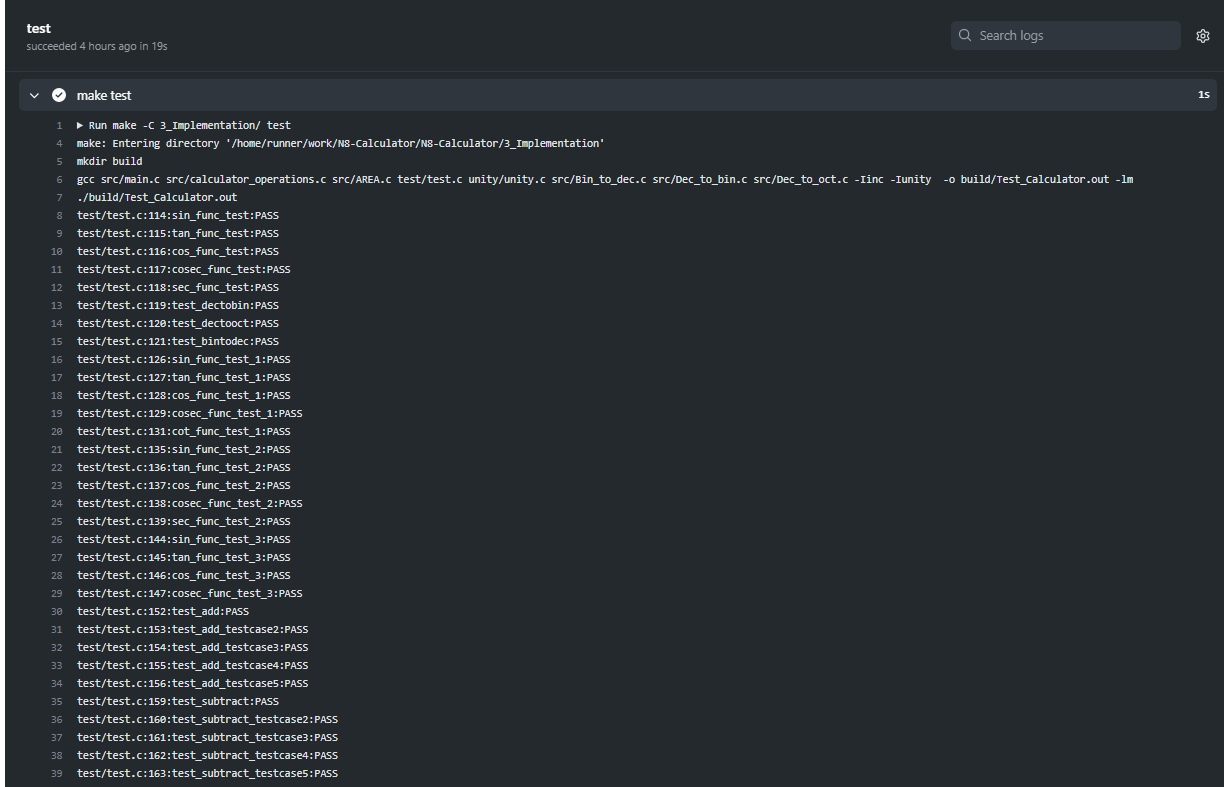
Static Analysis:

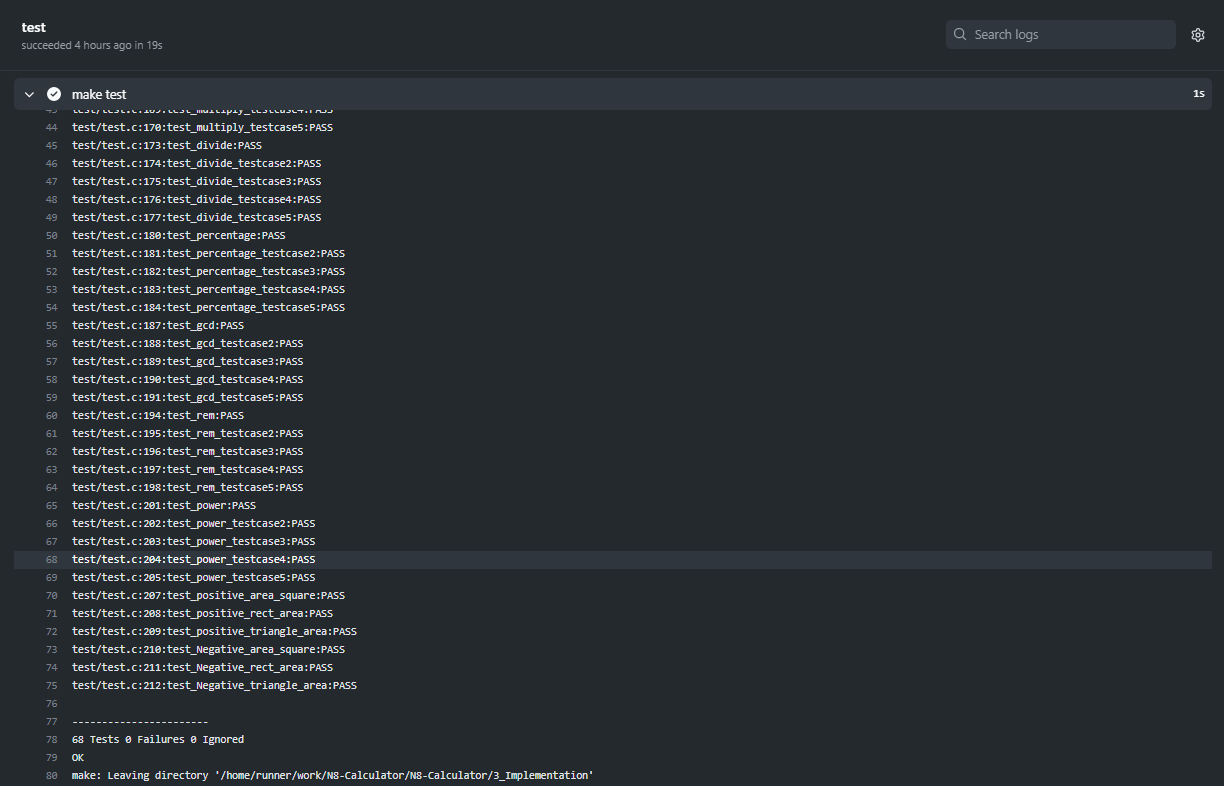




Dynamic Analysis:

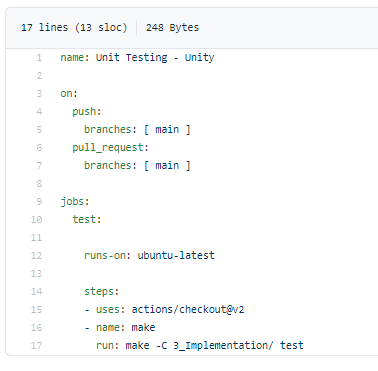




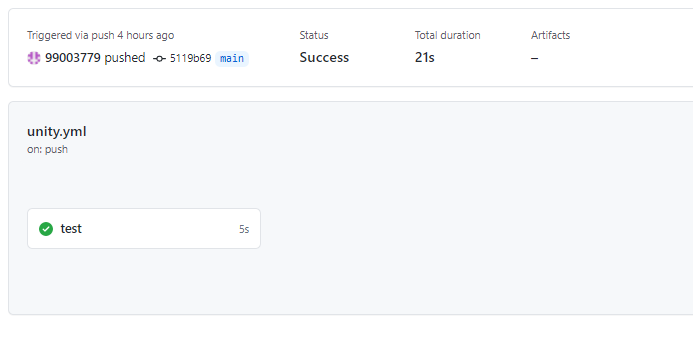


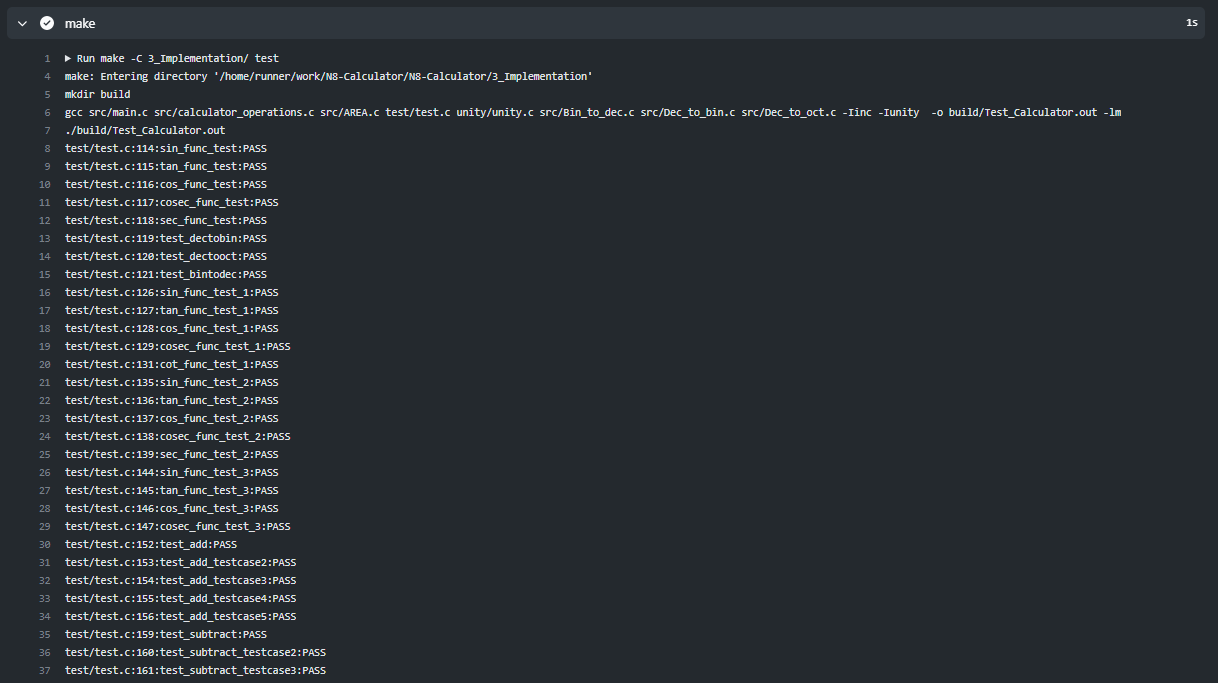
#### Unit Testing

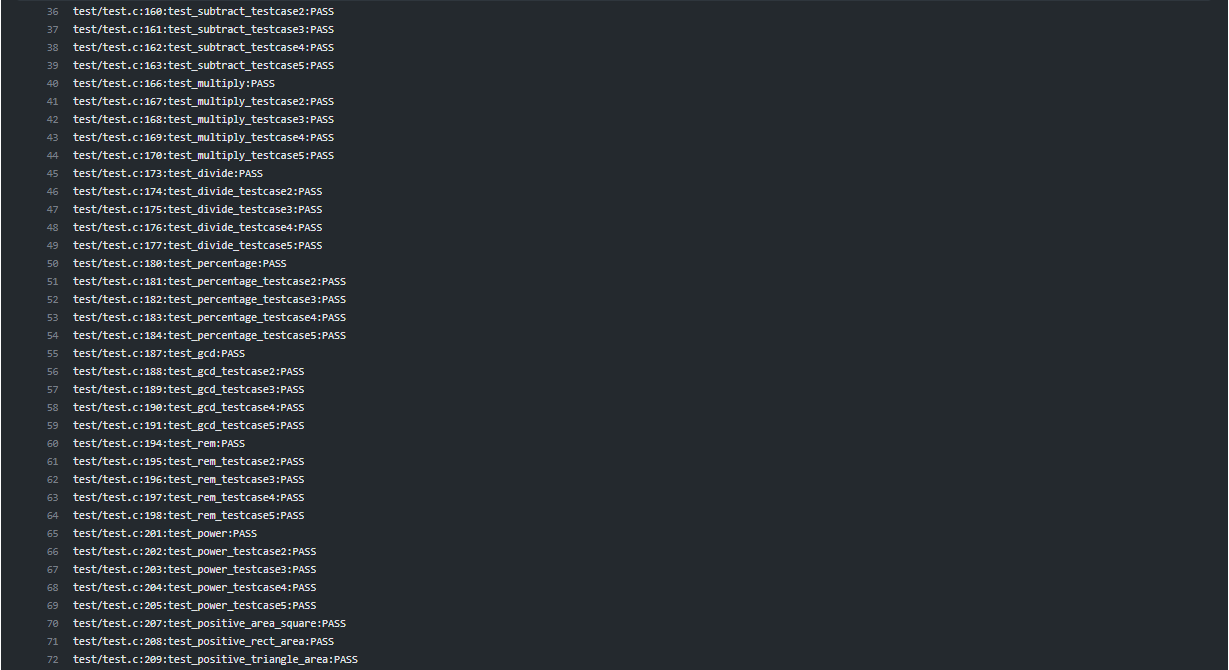
Setup:

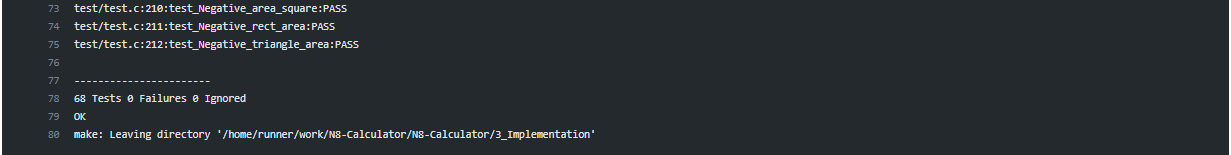


Outcome:









## Individual Contribution & Highlights

Contributions by me for the Team:

Arithmetic Operations:

* Implements various arithmetic operations which includes addition, subtraction, multiplication, division, remainder and GCD
* Implements low-level and high-level designs
* Contributed for low-level and high-level requirements in arithmetic operations
* Write various test cases for the same.

Percentage and Power:

* Implements percentage and power operation functions.
* Implements low-level and high-level designs
* Wrote various test cases for the same.

### Summary

Hardware is not implemented yet. Only software portion has been implemented.

### Challenges faced and how were they overcome

1. Found difficulty in writing the make file but with the help of collegues we are able to make our own make file
2. Initially it was difficult to understand the functions in test cases, but with self and group study we resolved the problems
3. Code debugging was a bit time consuming, but with brainstorming we excel it.
4. Integration of individual codes was difficult to understand, but with peer support, we did it.

### Future Scope (If applicable):

NA

### 

# Miniproject -2 [Individual]: Reading Excel Sheet and summarizing the data in the mastersheet

## Module/s:

Modules linked to this mini-project is Python and SDLC

### Topic and Subtopics

## Objectives & Requirements

Objective:

The main objective of the designed code is to read and write data. The code implemented, has considered the basic requirement of reading and writing data with user friendly environment.

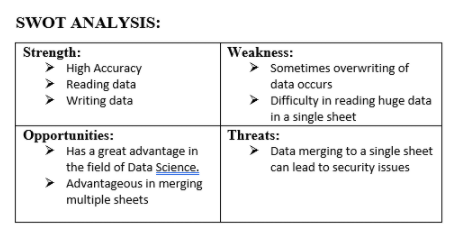
An excel sheet has been made manually which consists of 5 sheets, 1 master sheet and 1 summary sheet. Here we are searching details of an individual in all the 5 sub-sheets corresponding to :

Name:

Registration number:

Email ID:

Once the data has been fetched from the sub sheets then it will be printed to the master sheet. The excel sheet also consists of a summary sheet which indicates the count number of data fetched from each sheet. The whole implementation is used to read a file for better searching and writing. The code makes the study easier in the field of data science where lots and lots of data needs extraction.



4W’s and 1H:-

Who:- Basically used in research field and data science fields where large data needs manipulation and extraction.

What:- Xlsx file or CSV file with python code to read and write data

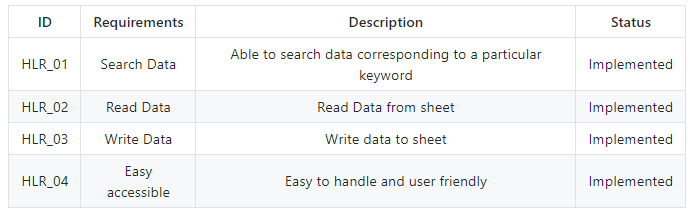
When:- Used when large data needs to manipulated and extracted.

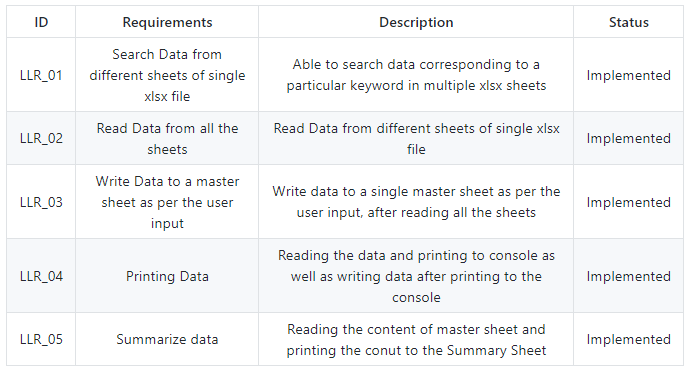
Where:- In Research and technical fields

How:- User friendly and easily accessible.

Requirements:

High-Level:



Low-Level: 

## Design

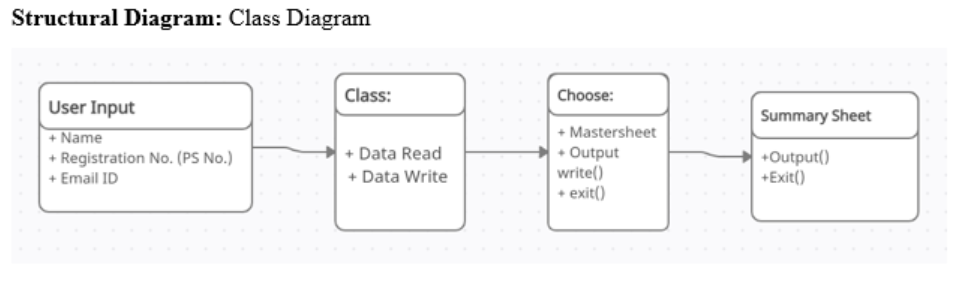


Figure : Structural Diagram of data read/write

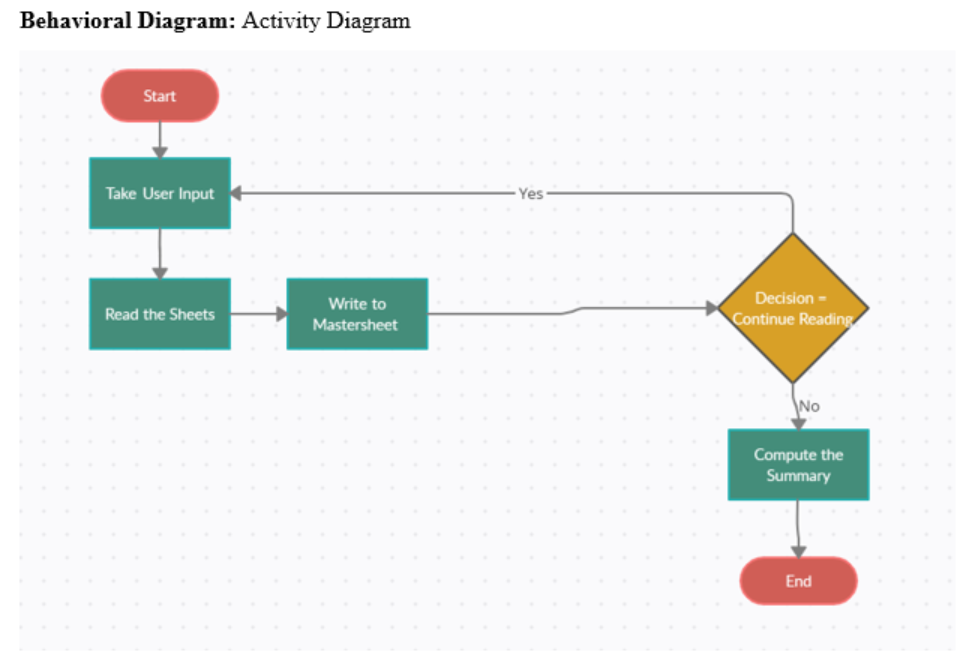


Figure : Behavioural Diagram of Data read/write

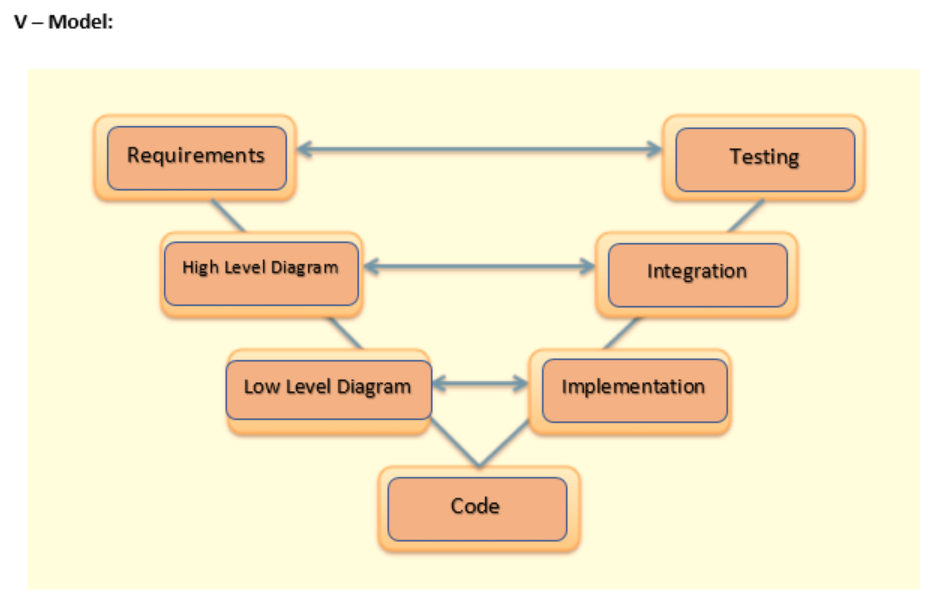


Figure : V-model Implementation of data read/write

## Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test\_ID | Description | Expected Input | Expected Output | Actual Output |
| TP\_01 | User enters the registration number, name and Email ID of the person being searched | 99673798  Ali Adibi  Ali.adibi@ece.gatech.edu | The data corresponding to the given input will be searched in all the sub-sheets and printed to mastersheet. | The data is printed to mastersheet and summary count has been incremented. |
| TP\_02 | User enters the registration number, name and Email ID of the person being searched. Now it takes multiple input of data by selecting Yes/No in the terminal window. | 99673798  Ali Adibi  Ali.adibi@ece.gatech.edu  Continue (y/n) ? y  99673786  Professor Peter Y. K. Cheung  p.cheung@imperial.ac.uk | The data corresponding to the given multiple inputs will be searched in all the sub-sheets and printed to mastersheet. | The data is printed to mastersheet and summary count has been incremented. |
| TP\_03 | User enters the registration number, name and Email ID of the person being searched. Now it takes multiple input of data by selecting Yes/No in the terminal window. Now user select no for further data intake. | 99673798  Ali Adibi  Ali.adibi@ece.gatech.edu  Continue (y/n) ? y  99673786  Professor Peter Y. K. Cheung  p.cheung@imperial.ac.uk  Continue (y/n) ? n | The data corresponding to the given multiple inputs will be searched in all the sub-sheets and printed to mastersheet.    Data has been printed to Mastersheet | The data is printed to mastersheet and summary count has been incremented. |

## Implementation Summary:

The user provides data of the person being searched which includes:

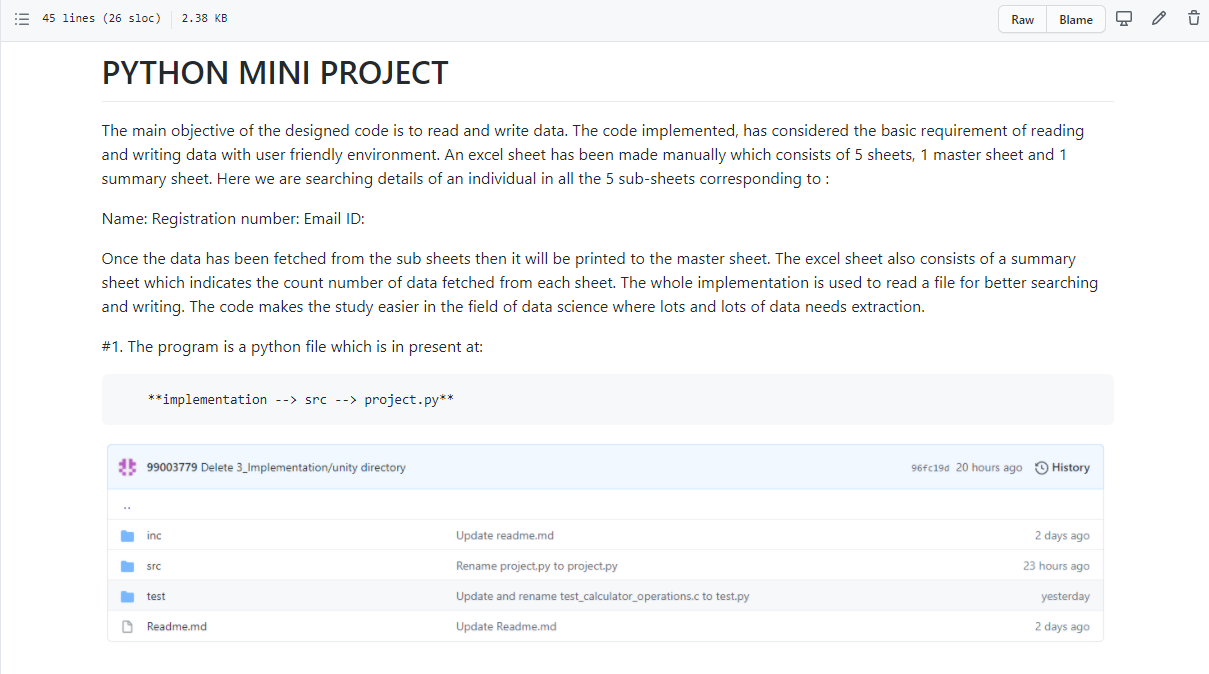
* Registration number:
* Name:
* Email ID:

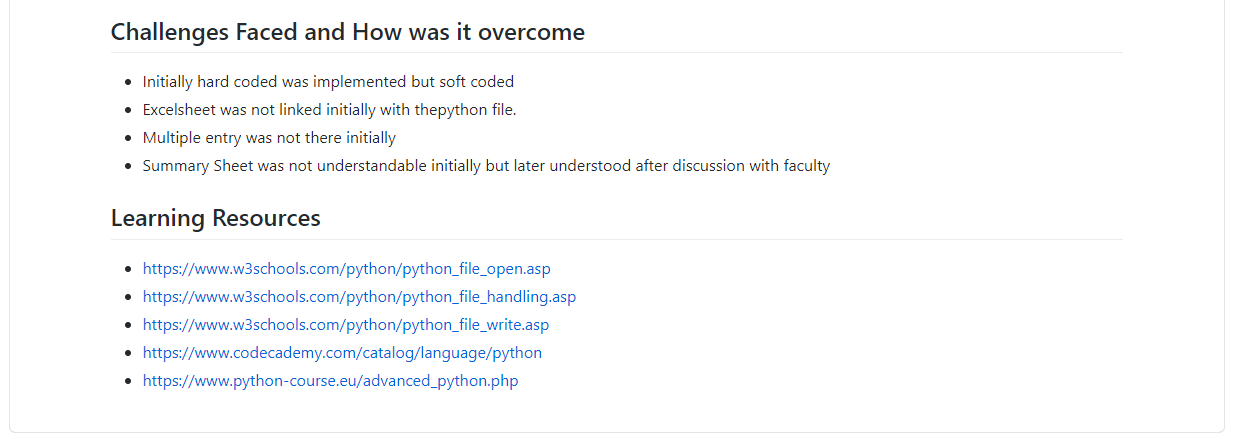
The designed python code will search for the provided data in all the sheets of a single excel file and summarize the complete data of that person in a single excel Mastersheet. Once all the data of a single person or multiple person has been printed to master sheet then the summary sheet will summarize the total number count of the mastersheet. The detailed steps are on Git Link shown below.

### Git Link

<https://github.com/99003779/Python_Mini_Project.git>

### Git Dashboard





### Summary

In this python mini-project, 5 sheets were created in a single excel file which contains data of a delegates. It sheet contains 40 rows with 10 columns. The user provides data of the person being searched which includes:

* Registration number:
* Name:
* Email ID:

The designed python code will search for the provided data in all the sheets of a single excel file and summarize the complete data of that person in a single excel Mastersheet. Once all the data of a single person or multiple person has been printed to master sheet then the summary sheet will summarize the total number count of the mastersheet.

## Individual Contribution & Highlights:

The whole python mini project is a single and individual project .

### Challenges faced and how were they overcome

* Initially hard coded was implemented but soft coded
* Excel sheet was not linked initially with the python file.
* Multiple entry was not there initially
* Summary Sheet was not understandable initially but later understood after discussion with faculty

# Miniproject -3 [Team] – Embedded C

## Module/s

Modules linked to this miniproject is Embedded C

### Topic and Subtopics

Driver Designing

GPIO Programming

I2C, SPI, UART and ADC Protocols

Hardware Abstraction Layer (HAL) Application interface

## Objectives & Requirements

Objective:

The main objective of the project is design a Body Control Module (BCM) with some features using STM32F407VG microcontroller with 32-bit ARM cortex-M4 FPU core processor.

A control module is a computer component in a car that monitors, controls and uses electrical devices throughout the vehicle. When electric devices began to be used in cars and trucks, each tool was controlled by a different electrical component. There was a cooling module, an interior lighting module, a door lock module, and so on. The body control module combines all these different modules under a single system to work together instead of partitioning, a simple format for both production and problem solving. While there are many different types of body control modules, they are generally the same, the sensors for connecting, switching and automatic switching together in a single computer system. These components are subdivided into inputs, such as sensor data about temperature or speed, and the effects, or the way a computer control system responds to control motor performance. Inputs and outputs are further divided into analog and digital information types - analog signals used by modular continuous modules (such as oil pressure) and digital signals used for modules that can either turn on or off (such as headlights or oil indicator light).

Requirement:

In our project we have implemented six features on a microcontroller STM32f4 discovery board. In the project we integrated IR Sensor, thermal sensor, Gas sensor (mq7), LDR sensor, Moisture sensor,  ignition sensor with microcontroller STM32f4 using GPIO protocol.

* IR Sensor:-An IR sensor is an electronic device, which illuminates light to detect the surrounding environment. The IR sensor can detect movement. Usually, in the infrared spectrum, all substances emit a certain type of radiation. These types of radiation are invisible to our eyes, but the infrared sensor can detect these rays.
* The gas sensor receives the attention of the liquor gas in the air and the analog voltage reading output. The sensor can operate at temperatures ranging from -10 to 50 ° C with an electrical power of less than 150 Ma to 5V. Sensitivity range ranges from 0.04 mg / L to 4 mg / L, suitable for breathalyzers.
* LDR or light-resistant light resistor is also known as photo resistor, photocell, photoconductor. One type of opposition its resistance varies depending on the amount of light falling on its surface. When light falls on an opponent, the resistance changes.
* Moisture sensors measure the amount of water in the soil. Since direct gravimetric measurement of free soil moisture requires removal, drying, and weight of the sample, soil moisture sensors measure water content indirectly using other earth materials, such as electricity resistance, dielectric constant, or neutron contact, as representative of moisture content.
* The ignition system produces sparks or burns the electrode at high temperatures to heat a mixture of gasoline gas in combustion engines fire engines, oil-fired boilers and gas, rocket engines, etc.
* Thermal sensors detect a change in a physical parameter such as resistance or output voltage that corresponds to a temperature change.

High Level Requirements:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirements | Description | Status |
| HLR\_01 | Enable control Module | Control Module should be enabled in order to detect different sensors. | Implemented |
| HLR\_02 | Temperature sensor | Able to detect temperature | Implemented |
| HLR\_03 | Gas Sensor | Able to detect gas leakage | Implemented |
| HLR\_04 | LDR (light dependent resistor) Sensor | Able to detect higher intensity of light | Implemented |
| HLR\_05 | Moisture Sensor | Able to detect moisture | Implemented |
| HLR\_06 | IR Sensor | Able to detect obstacles | Implemented |

Low Level Requirements:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirements | Description | Status |
| LLR\_01 | Implementation of air Conditioning Control module | Thermal sensor detects the high temperature and AC switches ON. | Implemented |
| LLR\_02 | Implementation of gas leakage control module | Gas sensor detects gas leakage and glows LED which prevents spontaneous car explosion. | Implemented |
| LLR\_03 | Implementation of shed controller module | LDR sensor enables the intensity of sunlight and the shed is enabled when the sunlight is too bright | Implemented |
| LLR\_04 | Implementation of Headlight control module | LDR sensor senses darkness and headlight turns ON. | Implemented |
| LLR\_05 | Implementation of Wiper control module | Moisture sensor senses water in wiper and LED switches ON | Implemented |
| LLR\_06 | Implementation of obstacle detector control module | LED glows when some obstacle is detected | Implemented |

## Design

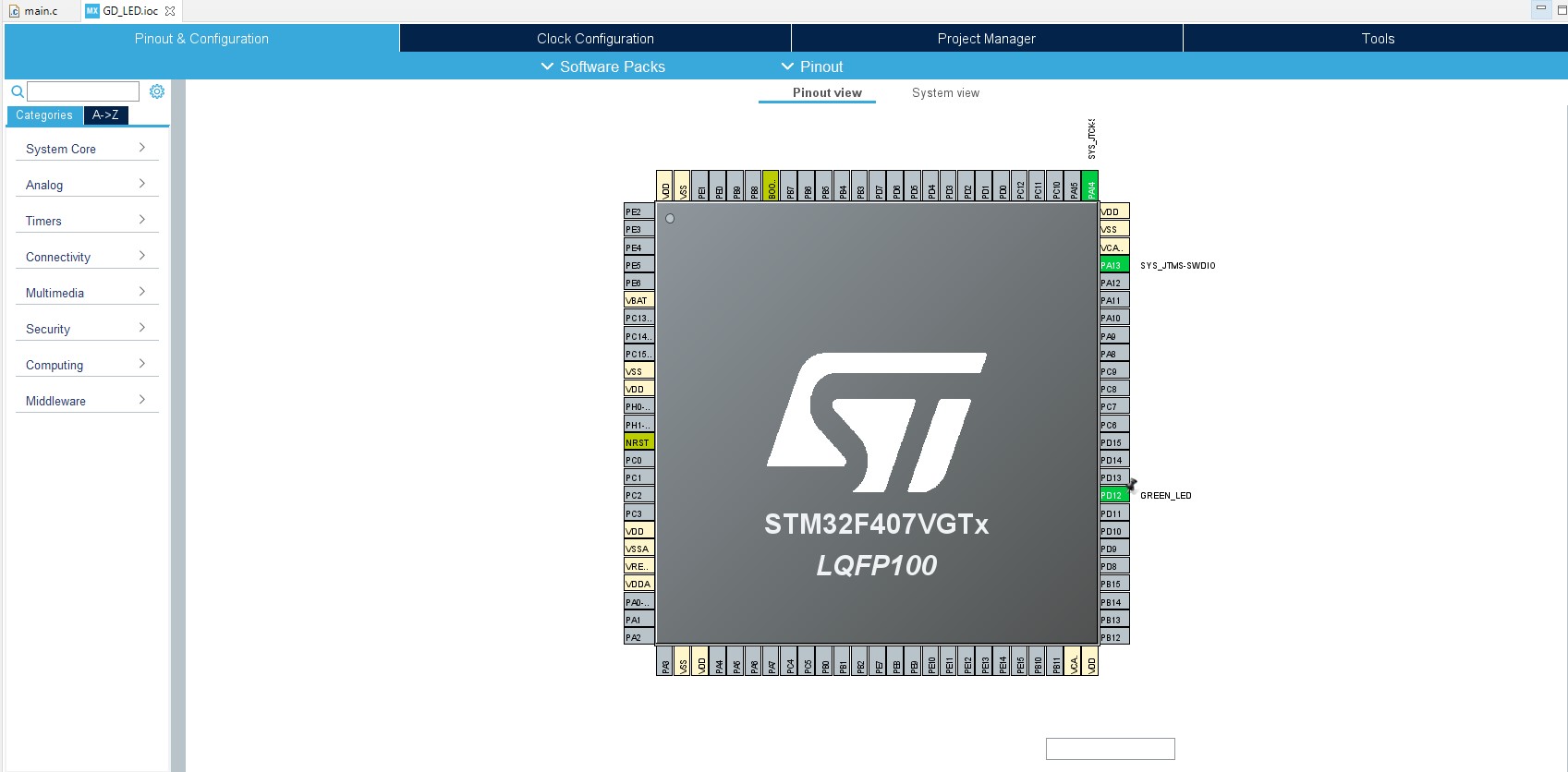


Figure : High Level Design of hardware

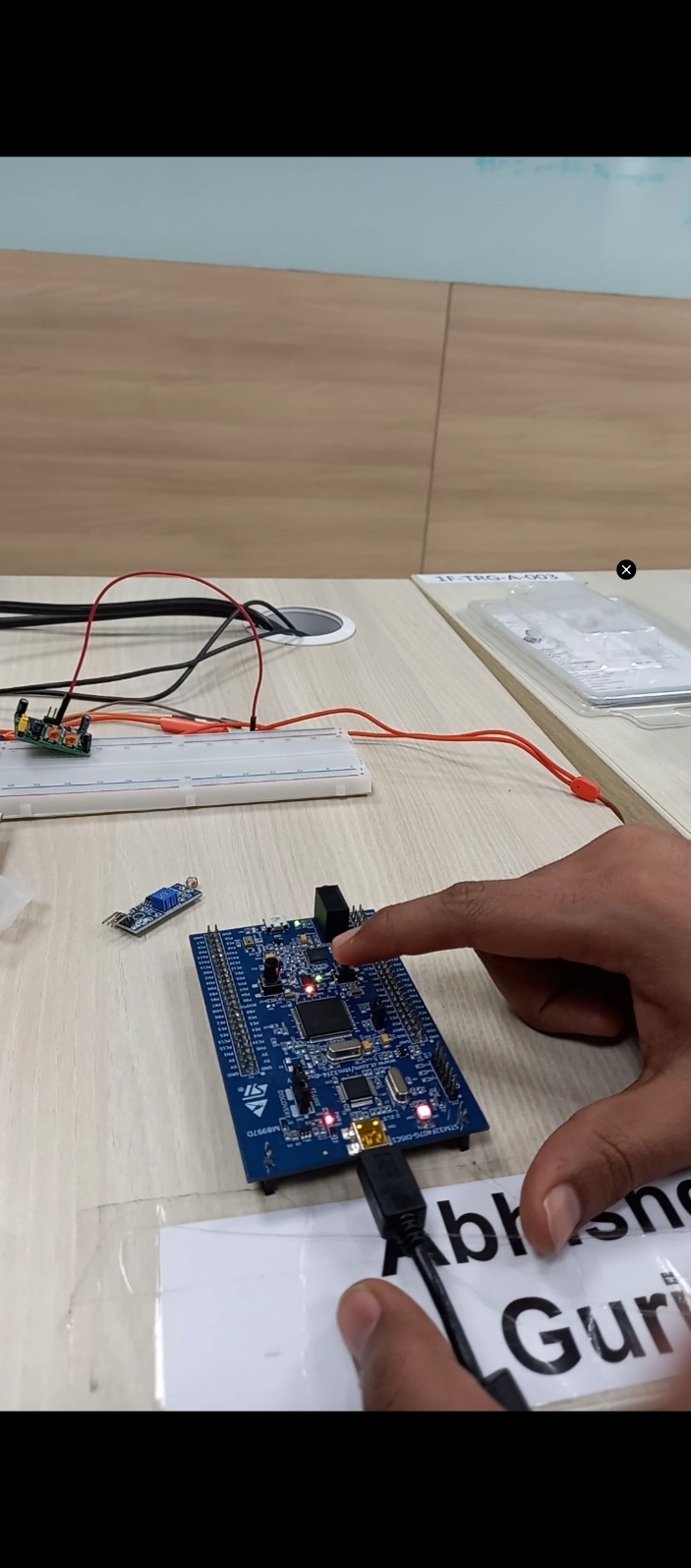
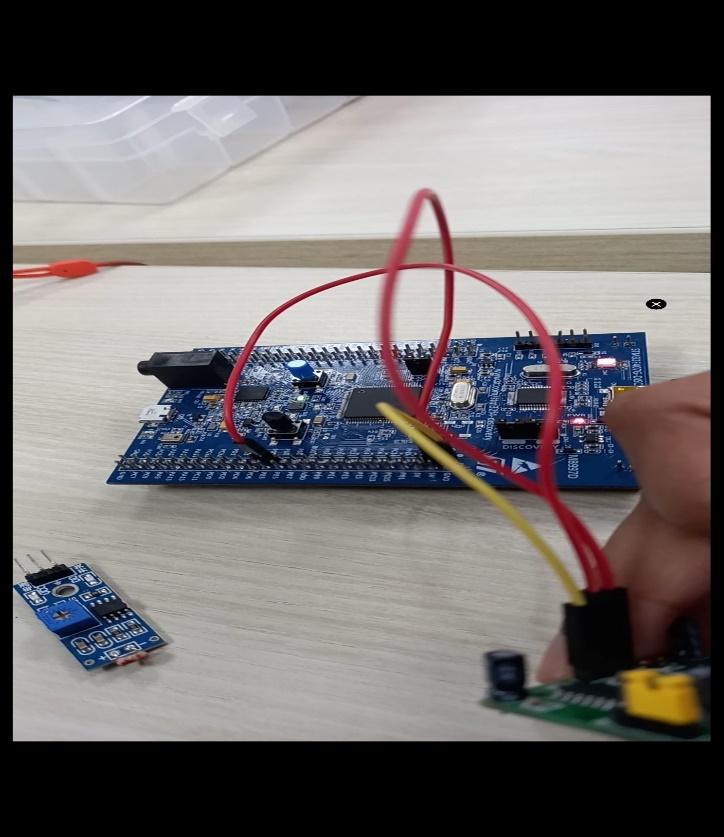


Figure :1. Air Conditioner Control Module

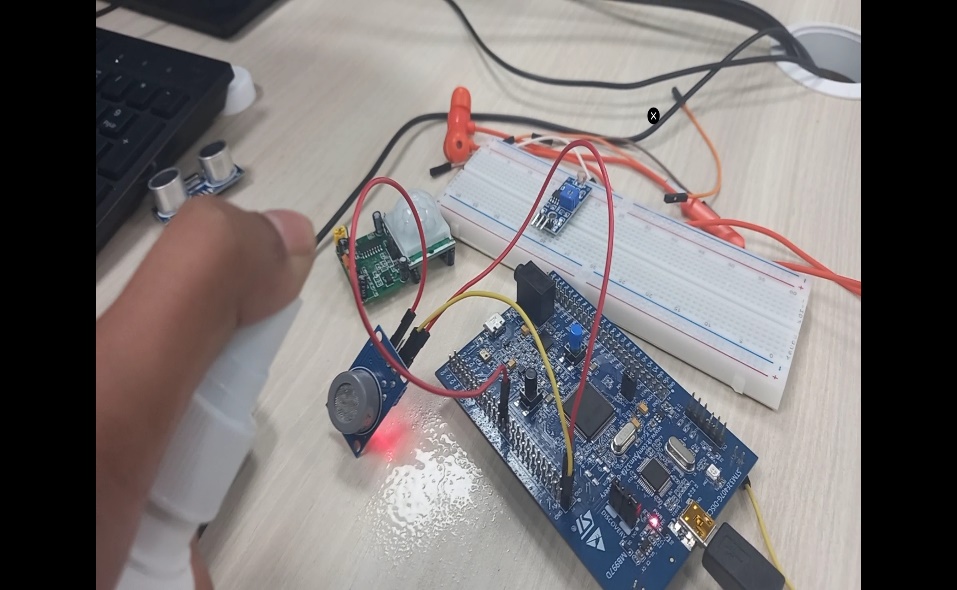
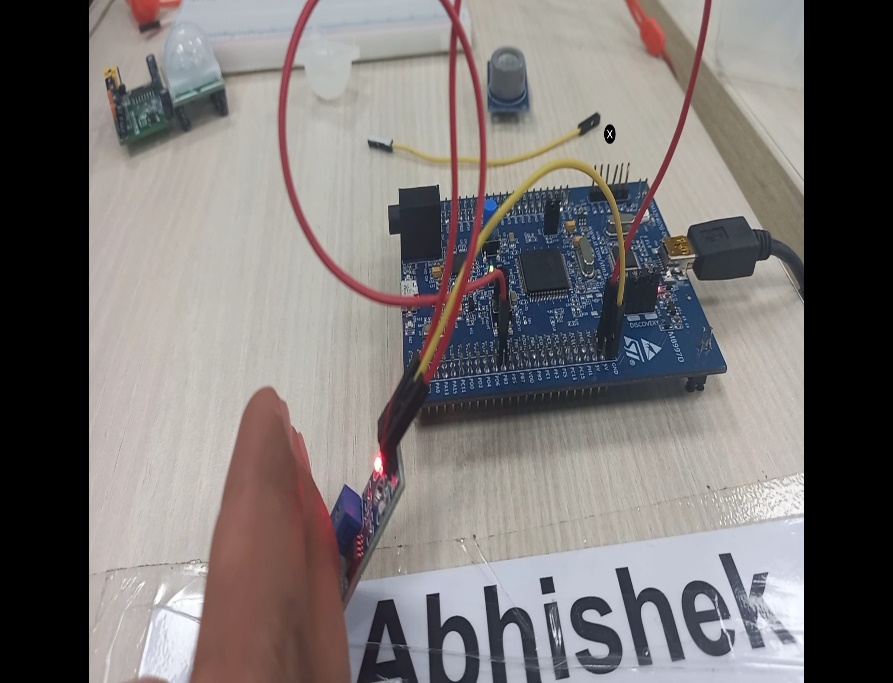
 

Figure : Gas Leakage Control Module

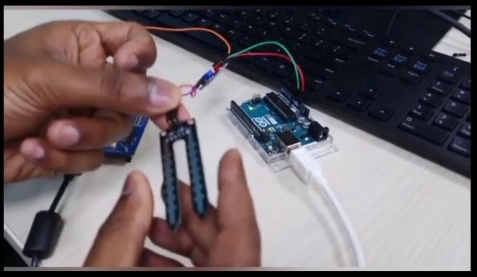
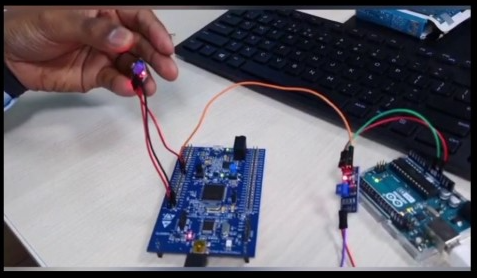
 

Figure :Wiper Control Module

## Test Plan

High Level Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Expected Input | Expected Output | Actual Output |
| HLP\_01 | Enable control Module should be enabled in order to detect different sensors. | Module is turned on | Module gets enabled | Module enabled |
| HLP\_02 | Temperature sensor should be able to detect temperature | High Temperature is provided | LED glows | LED glows indicating AC switches ON. |
| HLP\_03 | Gas Sensor should be able to detect gas leakage | Excess gas leakage | LED glows and system shut down | LED glows and car shut down |
| HLP\_04 | LDR Sensor should be able to detect higher intensity of light | Excess/low sunlight/light | Sun shed enabled or headlight switches ON | Sun shed enabled or headlight switched ON |
| HLP\_05 | Moisture sensor should be able to detect moisture | Water droplets | Moisture sensor senses water droplets and wiper starts moving | Moisture sensor senses water droplets and wiper starts moving |
| HLP\_06 | IR sensor should be able to detect obstacles | Sensor occupied by any obstacle | Red LED blinks | Red LED blinks |

Low Level Plan:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Expected Input | Expected Output | Actual Output |
| LLP\_01 | Thermal sensor detects the high temperature and AC switches ON. | High Temperature provided at the input of thermal sensor | AC switches ON | AC Switched ON |
| LLP\_02 | Gas sensor detects gas leakage and glows LED which prevents spontaneous car explosion. | Hand sanitizer containing alcohol | LED is switched ON and Buzzer starts beeping. | LED is ON |
| LLP\_03 | LDR sensor senses the intensity of sunlight and the shed is enabled when the sunlight is too bright | Torch Light (at small scale) or sunlight | Red LED turns ON | Red LED turns ON indicating Shed has been open |
| LLP\_04 | LDR sensor senses darkness and headlight turns ON. | Push button is pressed indicating seat shifts front or back | Green LED Glows. | Green LED Glows indicating headlight is ON |
| LLP\_05 | Moisture sensor senses water in wiper and LED switches ON | Water droplets | LED turned ON | LED turned ON indicating wiper is active |
| LLP\_06 | LED glows when some obstacle is detected | IR sensor detects obstacle | LED glows | Green LED Glows. |

## Implementation Summary

In this project, six functions of a Body Control Module has been implemented on a STM microcontroller using different sensors and Arduino board.

Following six features has been implemented:

1. **Air Conditioner Control Module:** In this module, temperature sensor has been used. Thermal sensor detects the high temperature and AC switches ON. Thermal sensors detect a change in a physical parameter such as resistance or output voltage that corresponds to a temperature change.
2. **Gas Leakage control Module/ Car shut down control Module:** In this module, Gas sensor has been used. Gas sensor detects gas leakage and glows LED which shuts down the car and prevents spontaneous car explosion. The gas sensor receives the attention of the liquor gas in the air and the analog voltage reading output.
3. **Sun Shed Control Module:** In this module, LDR sensor has been used. LDR sensor senses the intensity of sunlight and the shed is enabled when the sunlight is too bright. When light falls on it, the resistance changes and hence it works.
4. **Headlight Control Module:** In this module, again LDR sensor has been used. LDR sensor senses darkness and headlight turns ON.
5. **Wiper Control Module:** In this module, moisture sensor has been used. Moisture sensor senses water in wiper and LED switches ON indicating that wiper is working. Moisture sensor measures water content indirectly with the help of electricity resistance, dielectric constant, or neutron contact, as representative of moisture content.
6. **Obstacle Detector Control Module:** In this module, IR sensor has been used. LED glows when some obstacle is detected by the sensor. An IR sensor is an electronic device, which illuminates light to detect the surrounding environment. The IR sensor can detect movement.

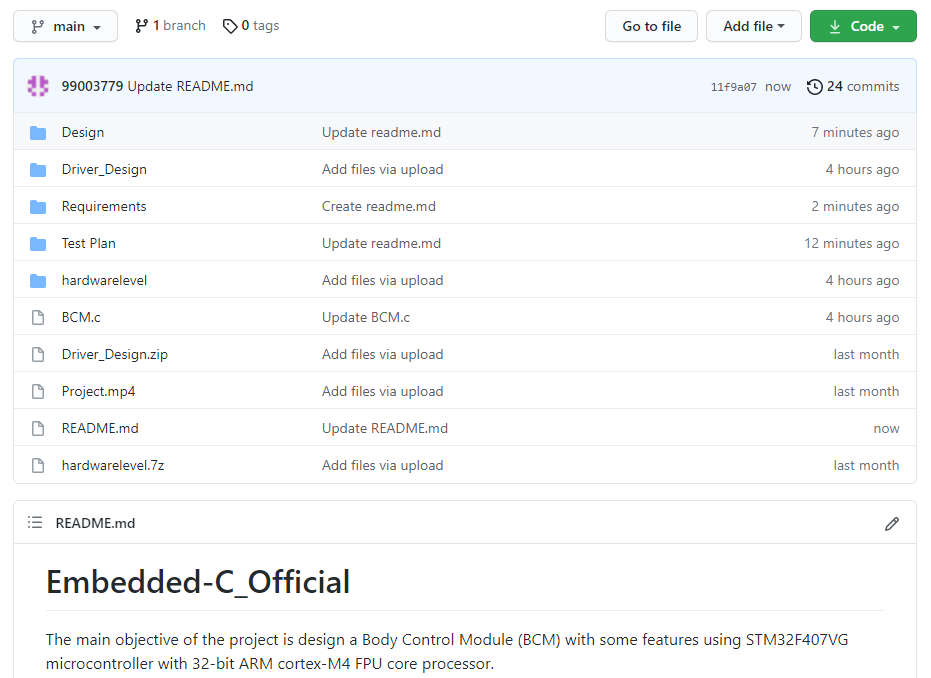
### Video Summary

“Please upload a short video on the repo for the walkthrough of the project (Team/Individual) less than 7min and less than 30MB File Size. Start is the Standard opening slide with title of miniproject + Team members followed by the walkthrough ”

### Git Link

<https://github.com/99003779/Embedded-C_Official.git>

### Git Dashboard



### Summary

In this project, we have implemented a body control module showing six features on a STM32 microcontroller board having a core processor of 32-bit ARM cortex-M4 FPU. Following six features has been implemented:

* Air Conditioner Control Module
* Gas Leakage Control Module/ Car shut down control Module
* Sun Shed Control Module
* Headlight Control Module
* Wiper Control Module
* Obstacle Detector Control Module

### Challenges faced and how were they overcome

* Integration of all the features on a single STM32 microcontroller board.
* Hardware implementation on STM32 IDE platform.

### Future Scope (If applicable): NA

# Miniproject -4 [Individual]: Embedded linux and Kernel Programming

## Module/s

Modules linked to this miniproject is Ex – Linux, Embedded C, OS programming and Kernel Device Drivers.

### Topic and Subtopics

1. Working with QEMU:-

* Setting up Qemu
* Building Custom Kernel
* Cross Compilation Techniques
* Booting Techniques

1. Kernel Programming and Device Drivers

* Introduction to Kernel
* Static and Dynamic Modules
* Kconfig entries
* Adding System Calls
* Simple Drivers
* File Operations
* Kernel Data structures
* Concurrency and Inter Process Communication (IPC)
* IOCTL Operations

## Objectives & Requirements

Objective:

The objective of this project is to implement the learning of the module in the following activities:

* Making a System call to echo back the given string
* Making a System Call to pass the Arguments
* Making a System call to traverse process list and print PID and PPID of a process and making a System call to retrieve the attributes of calling process
* Implementing IOCTL to echo back the string, implement two operations such that the string passed by one operation is retrieved by the other operation
* Applying mutual exclusion between first two threads by implementing three kernel threads. First two threads will write N nodes into a kernel list. The third thread waits for the first two threads and traverses the list while mutual exclusion is applied between the first two threads. Exit method cleanup the list nodes

Requirements:

High Level Requirement:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requirements** | **Description** | **Status** |
| HLR\_01 | System Call to echo back string | The system call echoes back the given string on target from user space | Implemented |
| HLR\_02 | System Call to pass the argument | The System Call passes the argument from user space to Kernel Space | Implemented |
| HLR\_03 | System Call to traverse process list and print PID and PPID and retrieve attributes of the calling process | The System Call traverse the process list, prints PID and PPID and retrieve attributes of the calling process on the target window | Implemented |
| HLR\_04 | IOCTL to echo back string using two operations | Use IOCTL to echo back string by implementing 2 operations, one for passing string and other of for retrieving it. | Implemented |
| HLR\_05 | Threads implementation of 3 threads where 3rd third retrieves data for first 2 threads with list of N nodes and its traversal | Applying mutual exclusion to implement 3 threads where 2 write N nodes to threads and 3rd traverses the list. Exit module cleans up the node. | Implemented |

Low Level Requirement:

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requirements** | **Description** | **Status** |
| LLR\_01 | The string is echoed back to Target or the kernel space by system call | The string “abcdef” gets echoed to Qemu by creating a system call invoked by the user space. | Implemented |
| LLR\_02 | The Argument is passed back to the target | The Argument is passed back to the kernel space from user space | Implemented |
| LLR\_03 | The System Call traverse the process list, prints PID and PPID and retrieve attributes of the calling process on the target window | User invokes a system call to get PID, PPID and attributes of the calling process | Implemented |
| LLR\_04 | IOCTL used to echo back string with 2 operations | IOCTL to echo back string using 2 operations one to pass the argument and other to retrieve data and display it on Qemu. | Implemented |
| LLR\_05 | Mutual exclusion of three threads to traverse, list and clean up nodes | First two threads will write N nodes into a kernel list. The third thread waits for it and traverses the list. Mutual exclusion applied between the first two threads. Exit method cleanup the list nodes | Implemented |

## Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Expected Input** | **Expected Output** | **Actual Output** |
| TP\_01 | The string passed by the user is echoed to target space when invoked | “abcdef” as string passed in userspace | “abcdef” is printed in target space | “abcdef” is printed in target space using command dmesg |
| TP\_02 | The argument passed by user is printed to the kernel space | 1,2,3 as argument is passed by user | 1,2,3 is passed to the kernel space | 1,2,3 is printed in kernel space using command dmesg in kernel space only |
| TP\_03 | The PID, PPID and the attributes of the process is retrieved in target space when user space code is invoked | User space invocation for the system call | PID, PPID, Process State, Priority, RT Priority, Static Priority and Normal priority are printed in kernel space. | PID, PPID, process State, Priority, RT Priority, Static Priority and Normal priority are printed in kernel space. |
| TP\_04 | 3 threads are implemented. 1st and 2nd thread creates a list of N nodes, 3rd traverses the list after waiting for 2 threads to complete. | Invocation of the code | 3rd thread waits for 1st 2 threads to create a list through N nodes. 3rd thread then traverses the list and adds in Kfifo. | 3rd thread waits for 1st 2 threads to create a list through N nodes. 3rd thread then traverses the list and adds in Kfifo. |
| TP\_05 | 3 threads are implemented. 1st and 2nd thread creates a list of N nodes, 3rd traverses the list after waiting for 2 threads to complete by mutual exclusion. | Invocation of the code | 3rd thread waits for 1st 2 threads to create a list through N nodes by mutual exclusion. 3rd thread then traverses the list. Exit method cleans up all the nodes. | 3rd thread waits for 1st 2 threads to create a list through N nodes by mutual exclusion. 3rd thread then traverses the list. Exit method cleans up all the nodes. |

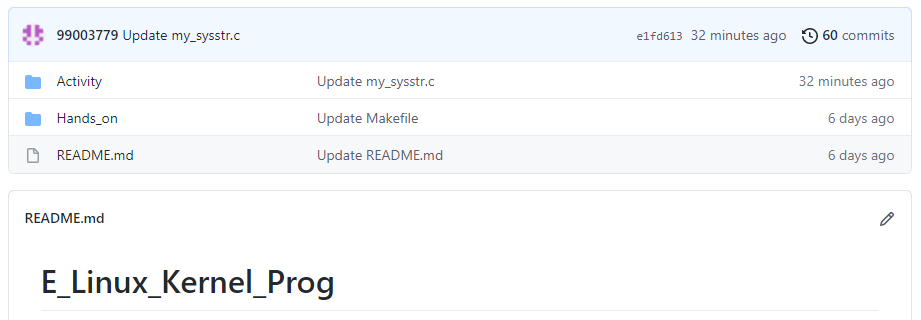
### Video Summary

“Please upload a short video on the repo for the walkthrough of the project (Team/Individual) less than 7min and less than 30MB File Size. Start is the Standard opening slide with title of miniproject + Team members followed by the walkthrough ”

### Git Link

<https://github.com/99003779/Embedded_linux_and_kernel_programming.git>

### Git Dashboard



## Individual Contribution & Highlights

“Brief on Contributions by you for Team”

### Summary

The whole project is based on kernel programming with understanding of embedded linux, device drivers, concept of system call and IOCTL i.e input output control.

The Linux Kernel Driver model is a combination of all the different types of drivers that were previously used in the kernel. It aims to add bus-specific drivers to bridges and devices by integrating data collection and operating in data that are globally accessible.

System calling is a system configuration program that asks for applications from the kernel, and strace is a powerful tool that allows you to follow a small layer between user processes and Linux kernel.

Ioctl, which means "input output control" is a type of system call to a specific device. There are only a few system calls on Linux (300-400), which are not enough to display all the functions of unique devices. The driver can then specify ioctl which allows the user app to send orders.

### Challenges faced and how were they overcome

* System call implementation.
* Depth understanding of user space and kernel space

### Future Scope (If applicable):- NA