GENESIS - Learning Outcome & Mini-project Summary Report

Details

Ver. Rel. No.	Release Date	Prepared By	Reviewed By	To Be Approved	Remarks/Revision Details
1.0	17/02/2022	Sanapala Usha Rani 40021045			

Contents

List of Figures

Mini project – 1: Rock Paper Scissor [Individual]

Modules:

1. C Programming
2. Git

Requirements:

Description: Rock paper scissors is a hand game usually played between two people, in which each player simultaneously forms one of three shapes with an outstretched hand. These shapes are "rock", "paper", and "scissors". This C++ Program [Mini Project] is aimed at automating one of the players called as Computerized Player and the taking rock, paper or scissor as an input from the used.

Introduction:

Rock paper scissors (also known by other orderings of the three items, with "rock" sometimes being called "stone", roshambo or ro-sham-bo) is a hand game usually played between two people, in which each player simultaneously forms one of three shapes with an outstretched hand. These shapes are "rock" (a closed fist) "paper" (a flat hand), and "scissors" (a fist with the index finger and middle finger extended, forming a V). "Scissors" is identical to the two-fingered V sign (also indicating "victory" or "peace") except that it is pointed horizontally instead of being held upright in the air.

A simultaneous, zero-sum game, it has only two possible outcomes: a draw, or a win for one player and a loss for the other. A player who decides to play rock will beat another player who has chosen scissors ("rock crushes scissors" or sometimes "blunts scissors"), but will lose to one who has played paper ("paper covers rock"); a play of paper will lose to a play of scissors ("scissors cuts paper"). If both players choose the same shape, the game is tied and is usually immediately replayed to break the tie.

Game Play and Rules:

Partners say, “Roshambo” or “Rock Paper Scissors” together with their hands in a fist. On “bo” or “paper,” players pick one of three things to show with their hand: Rock which is demonstrated by a fist. Scissors with two fingers spread out to represent scissors. (This looks like a sideways peace sign.) Paper by holding out a palm down, flat hand. If players show the same things, they go again. If one player picks rock and one scissor, the player who showed rock wins the dispute. To explain this, say rock crushes scissors (no need to actually crush). If one player picks scissors and the other paper, the player who showed scissors succeeds. Scissors cuts paper. If a player shows paper while the other shows rock, the player who picked paper succeeds. Paper covers rock. Once game is taught, ask players to use in many different games and situations to solve minor disputes, such as is the ball in or out or who arrived first in line.

Cost Features and Timeline:

In ancient times this game is played in person. In our digital era this game has an app and only some of the apps are paid.

Defining Our System:

64 Bit Windows / Open-Source Linux & its derivatives. Open-Source Programming Tools like G++/GCC, C IDE (Visual Studio Code or any other).

SWOT Analysis

Strength:

Develops knowledge Improves Math Skills Improves concentration

Weakness:

Works only with keyboard

Opportunities:

We can increase the number of things used in the game like lizard, snake. Etc if updated. Our game is portable i.e., it can be used in both Linux and windows

Threats:

There are so many improvements with the latest technology but our game is fundamental.

4W's and 1 H's:

What:

Playing Rock paper scissor.

Where:

Our Game has to be played in PC/Laptop.

Who:

Anyone can play this game. But it is very interesting for young kids.

When:

This game can be played whenever you are feeling bored.

How:

Requires one player to play against computer by giving required input.

High Level Requirements (Software)

1.Windows OS

2.Github

3.Visual Studio Code

4.Windows Subsystem for Linux (WSL)

5.gcc Compiler

Low Level Requirements

1.User should put the valid requirements.

Design:

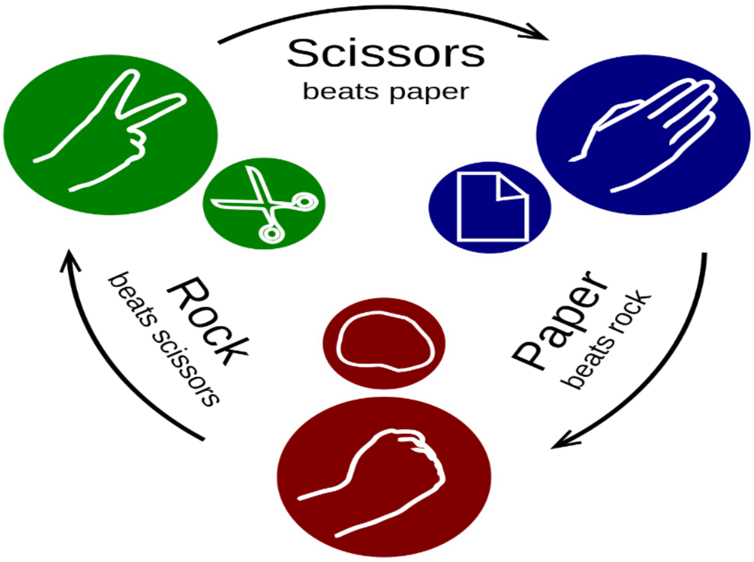


Fig: Activity Diagram

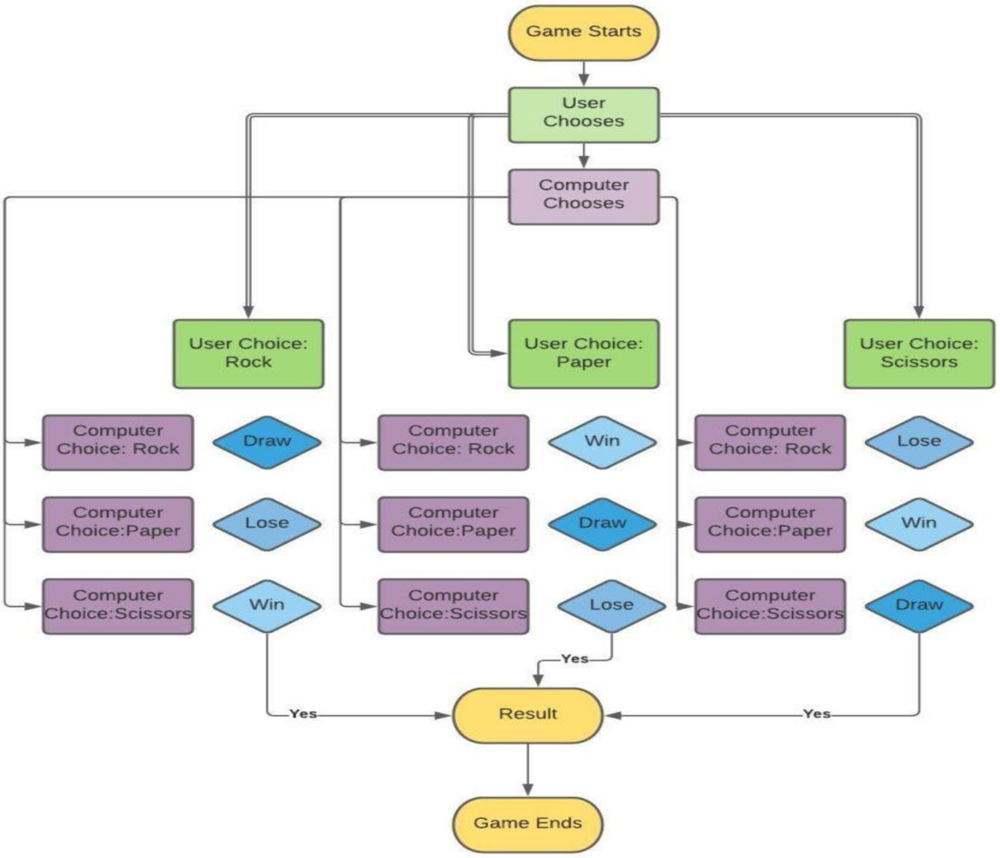
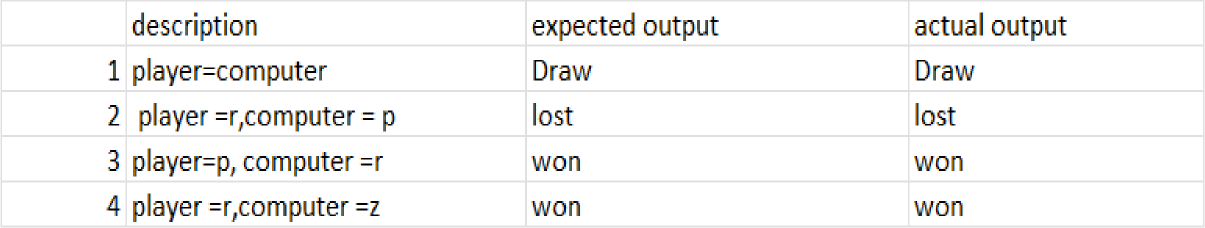


Fig: sequential diagram

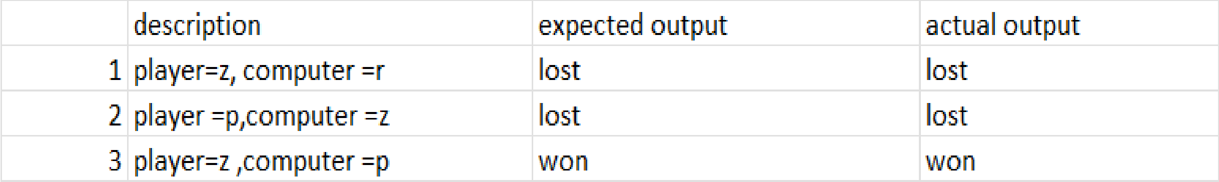
Test Plan:

High level Test Plan:



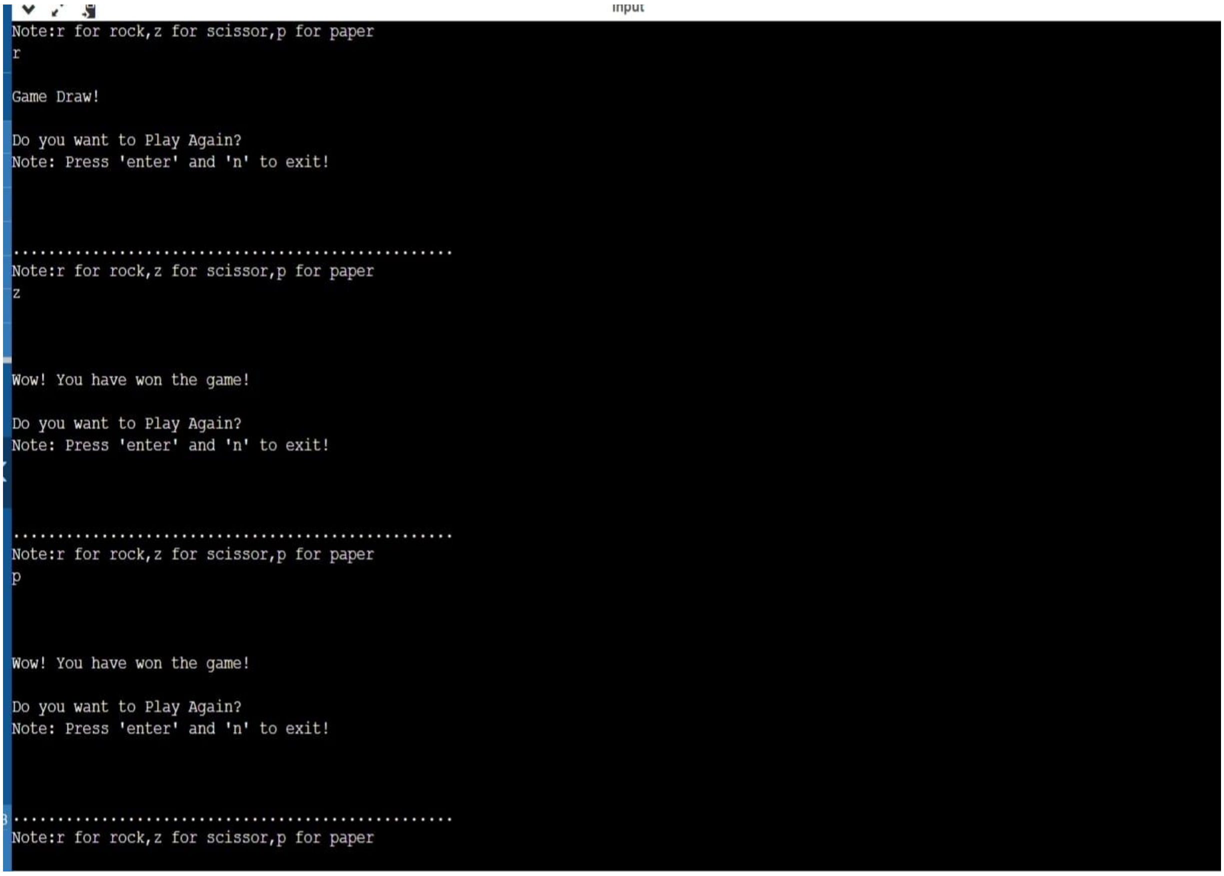
	description	expected output	actual output
1	player=computer	Draw	Draw
2	player =r,computer = p	lost	lost
3	player=p, computer =r	won	won
4	player =r,computer =z	won	won

Low level Test Plan:



	description	expected output	actual output
1	player=z, computer =r	lost	lost
2	player =p,computer =z	lost	lost
3	player=z ,computer =p	won	won

Output:



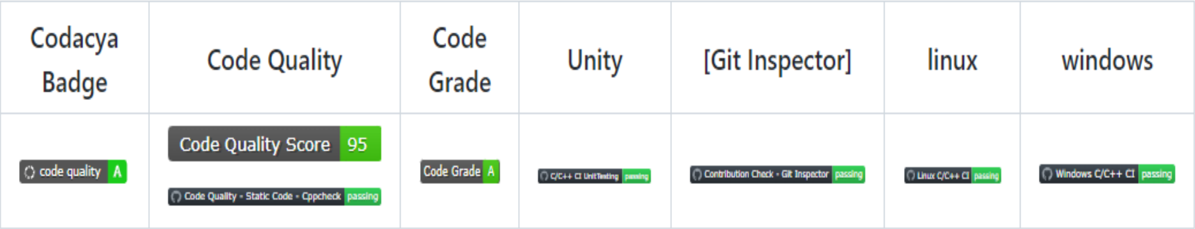
```
Note:r for rock,z for scissor,p for paper
r

Game Draw!

Do you want to Play Again?
Note: Press 'enter' and 'n' to exit!



..................................................
Note:r for rock,z for scissor,p for paper
z



Wow! You have won the game!

Do you want to Play Again?
Note: Press 'enter' and 'n' to exit!



..................................................
Note:r for rock,z for scissor,p for paper
p



Wow! You have won the game!

Do you want to Play Again?
Note: Press 'enter' and 'n' to exit!



..................................................
Note:r for rock,z for scissor,p for paper
```

Implementation and Summary:

Git link:

Link: https://github.com/Usharani8/M1_game_rock-paper-scissors.git

Git Dashboard:



Codacya Badge	Code Quality	Code Grade	Unity	[Git Inspector]	linux	windows
code quality A	Code Quality Score 95 Code Quality - Static Code - Cppcheck passing	Code Grade A	C/C++ CI UnitTesting passing	Contribution Check - Git Inspector passing	Linux C/C++ CI passing	Windows C/C++ CI passing

Miniproject2: Car Seat Heat Control System [Individual]

Introduction:

This project aims to explain the task of keeping to control the temperature of the seat .To produce a robust-designed of heat control system. Gives full functional statements to the management of the User. The purpose of this project is to present a general approach to result of controlling the heat of the seats.

Research:

OBJECTIVES: The aim of the project is to design a CAR SEAT HEAT CONTROL SYSTEM, Climate control is a more sophisticated form of air-conditioning, which allows the temperature of a cars Seats to be accurately controlled. Users can set the required temperature and the system automatically adjusts the speed and amount of cold air introduced into the Seats & cabin

Modules:

1. C Programming
2. Embedded System
3. Simulink IDE
4. Git

SWOT Analysis:

Strength:

- It aims to simplify the task of Showing a accurate result

- Simple & Easy to Use

Weakness:

- It is not completely encrypted

Opportunities:

In future, more Data such as temperature and seat comfortability can be added apart from the ones which are used here in this project.

Threats:

Company now send and retrieve data from databases to help better manage inventory and the like, but some companies don’t have the right data encryption practices in place. This can often lead to lost data and sometimes, the data easily obtainable by criminals.

Requirements

4W's and 1 H's

Where:

This problem is surfaced in all the user and manufacture in the world.

Who:

•All the Car Users & Manufactures who has lots of System requirement in their different units and their different services

When:

• As the business of the organisation increases number of System users

• With the increases of user end

How:

•This project takes in all the input values and yields out the management parameters.

Design:

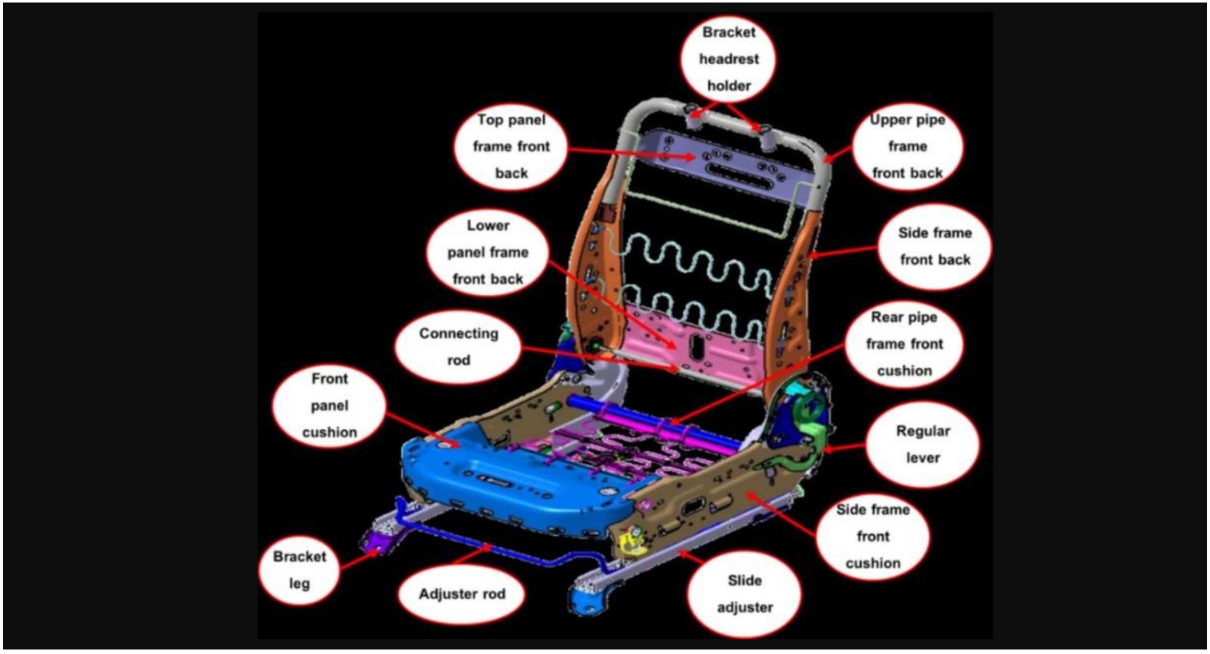
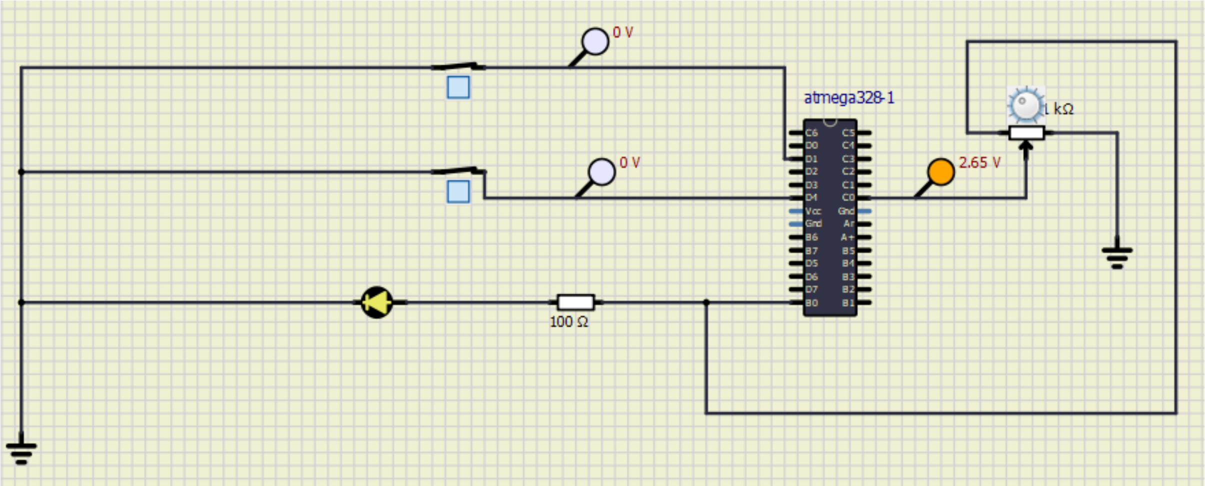
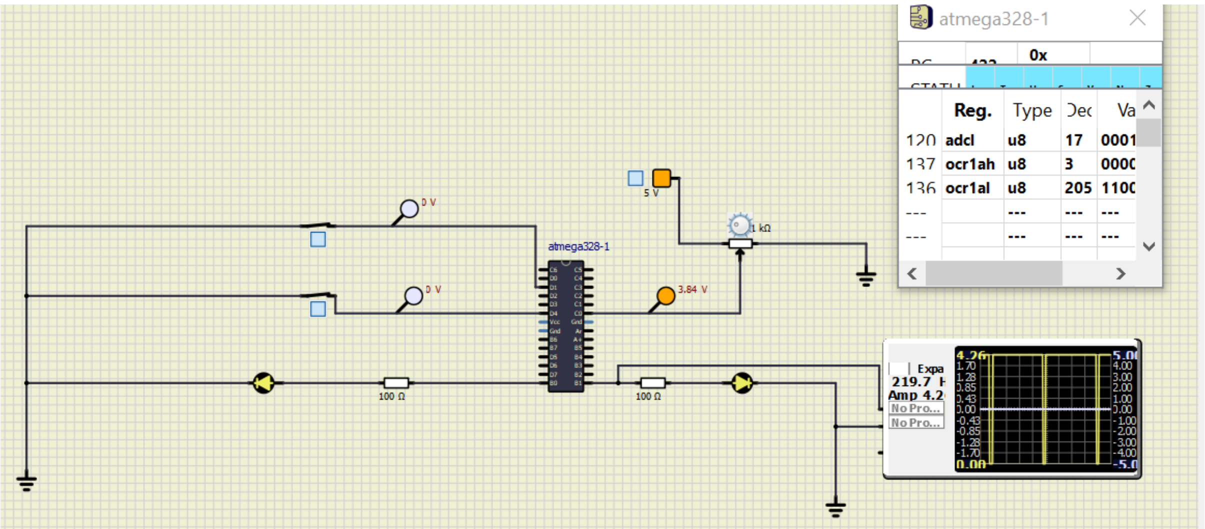


Fig: Behavioural diagram





	Reg.	Type	Dec	Va
120	adcl	u8	17	0001
137	ocr1ah	u8	3	0000
136	ocr1al	u8	205	1100
---		---	---	---
---		---	---	---

Fig: simulation

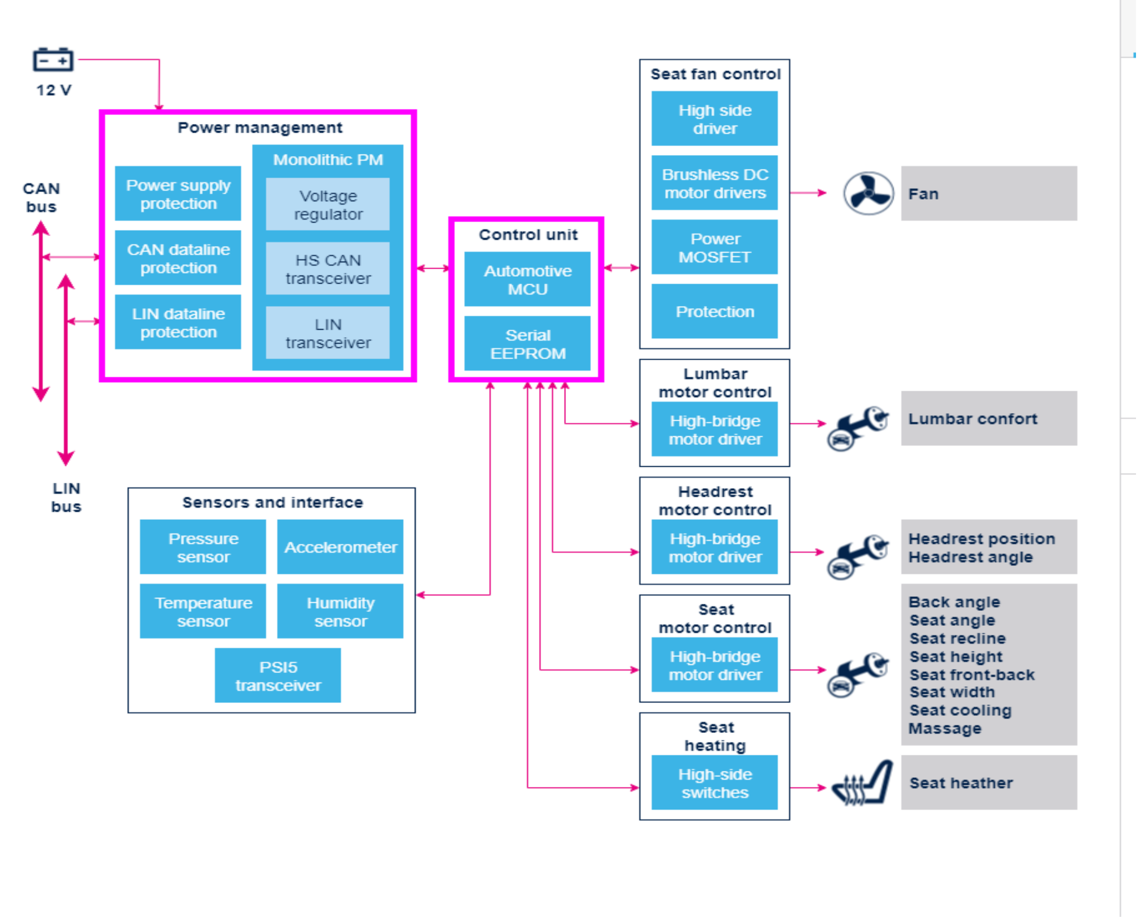


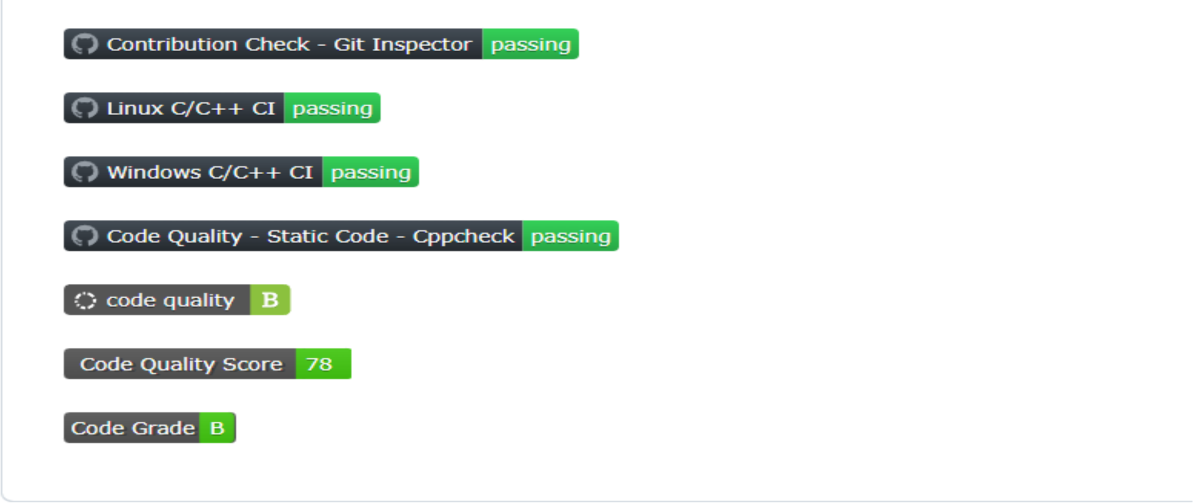
Fig:structural diagram

Implementation and Summary:

Git Link:

Link: https://github.com/Usharani8/M2-Embedded_BellControl.git

Git Dashboard:



Contribution Check - Git Inspector passing

Linux C/C++ CI passing

Windows C/C++ CI passing

Code Quality - Static Code - Cppcheck passing

code quality B

Code Quality Score 78

Code Grade B

Miniproject3- Scientific Calculator [Team]

Requirements:

Introduction:

It is an advanced calculator that will allow users to perform operations in mathematics (Arithmetic, Trigonometric, Matrices, Conversion etc). However, the input has to be provided by user. The input values can be from any integer to even a number with decimals. Moreover, this calculator is smart enough to operate all the operations.

Modules:

1. SDLC
2. Git

Features:

- It can do all operations (Addition, Subtraction, Multiplication, Division).
- Logarithmic operations, Exponential operations are also available.
- Power functions, Factorial and Conversions which are helpful for students are added.
- Basic Trigonometric Operations are also available.
- Basic Matrix Operations are also available.
- It has double precision.

Cost v/s Features:

COST

Casio FX (Programmable)	Casio FX (Non-color Programmable)	Casio FX (color Programmable)
Casio FX (240 functions)	Casio FX (300 functions)	Casio FX (522 functions)
10 Digit Casio	12 digit Casio	12 digit robust Casio

FEATURES

SWOT Analysis:

Strengths

- User Friendly
- All basic operations
- Double Precision
- Trigonometric operations
- Matrix Operations

Weakness

- Limited Operations
- Memory Wastage

Opportunities

- It can be expanded by adding additional features like Inverse Trigonometric operations, Equations etc.

Threats

- There are other programmable calculators which may affect our product marketing.

4W's and 1H's:

What: Scientific Calculator.

Where: Used in Provisional Stores and Students.

When: For Complex Calculations.

Why: For Easy and faster Calculations.

How: Operates by using User Inputs.

Detail Requirements:

High level Requirements:

High level Requirement	Description
HLR1	Basic Operations
HLR2	Trigonometric Operations

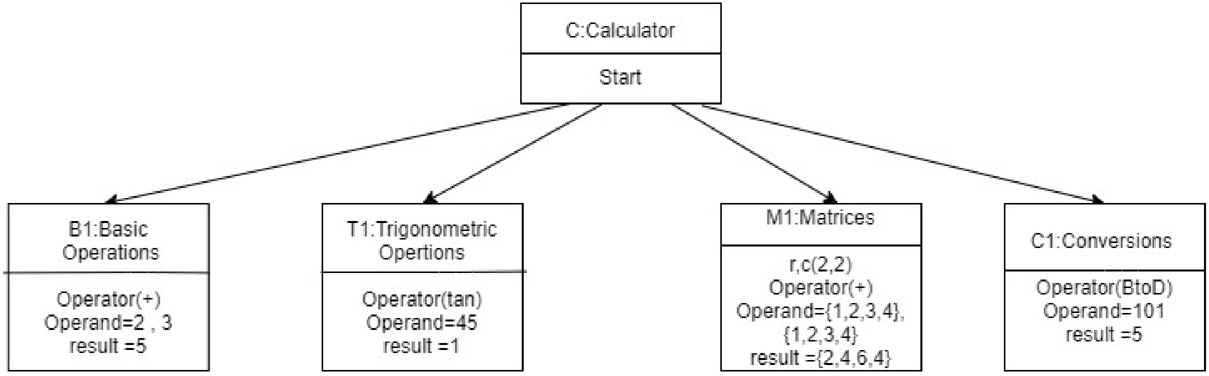
High level Requirement	Description
HLR3	Matrix Operations
HLR4	Conversions
HRL5	Advanced operations

Low level Requirements:

Low Level Requirement	Description	Related HLR
LLR1	Addition	HLR1
LLR2	Subtraction	HLR1
LLR3	Multiplication	HLR1
LLR4	Division	HLR1
LLR5	Sine	HLR2
LLR6	Cos	HLR2
LLR7	Tan	HLR2
LLR8	Cosec	HLR2
LLR9	sec	HLR2
LLR10	Cot	HLR2
LLR11	matrix addition	HLR3
LLR12	matrix subtraction	HLR3
LLR13	matrix multiplication	HLR3
LLR14	Binary - Decimal Conversion	HLR4
LLR15	Decimal - Binary Conversion	HLR4
LLR16	Decimal - Octal Conversion	HLR4
LLR17	Octal - Decimal Conversion	HLR4
LLR18	Length unit Conversion	HLR4
LLR19	Temperature unit Conversion	HLR4
LLR20	Current AC-DC Conversion	HLR4

Low Level Requirement	Description	Related HLR
LLR21	Log	HLR5
LLR22	Exponential	HLR5
LLR23	Modulus (remainder)	HLR5
LLR24	Factorial	HLR5
LLR25	Square root	HLR5
LLR26	Cube root	HLR5
LLR27	LCM	HLR5
LLR28	GCD	HLR5
LLR29	Permutation	HLR5
LLR30	Combination	HLR5

Design:



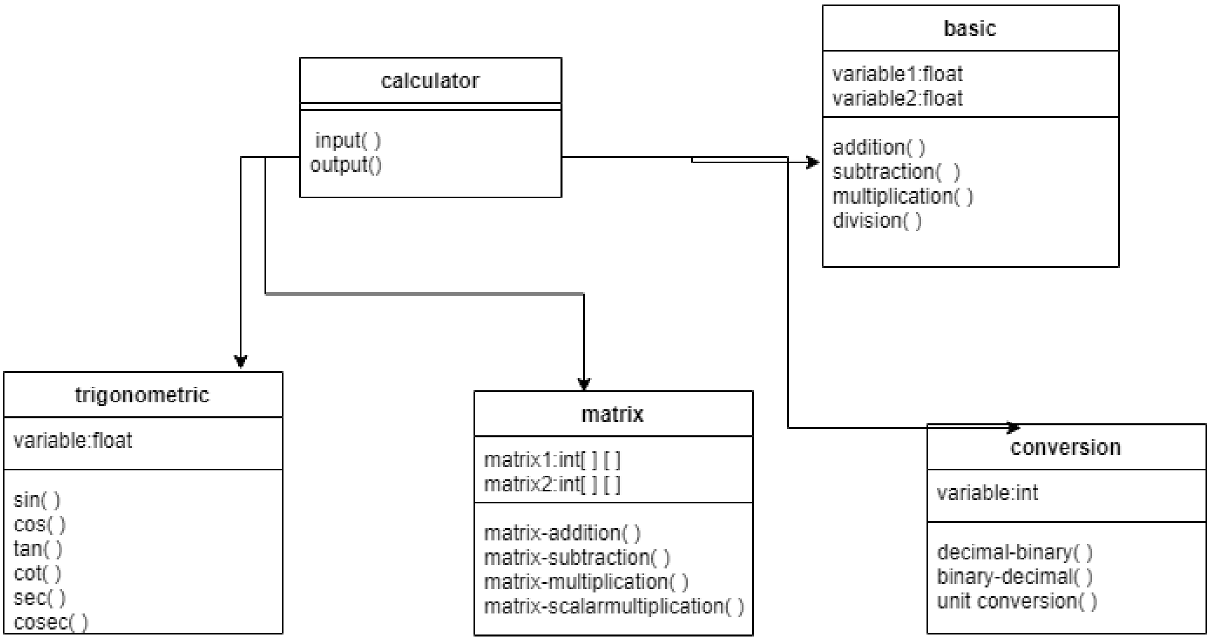


Fig:Hlr digram

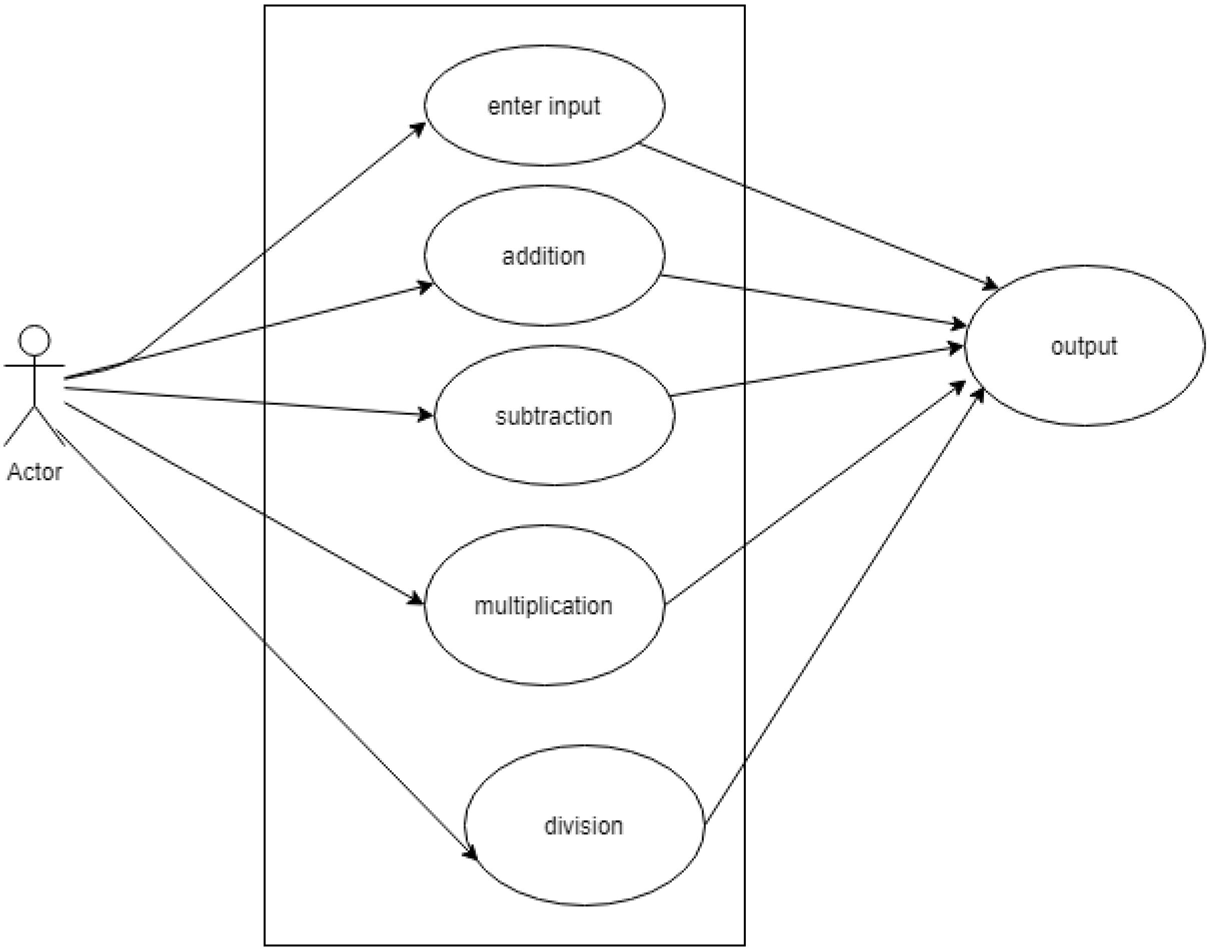


Fig:Llr diagram

Test cases:

Test Plan for Matrix Operation:

Test ID	Requirement	Expected Output	Actual output
LLR8	R1! =R2 &C1! =C2	Invalid	Invalid
LLR8	R1=R2 &C1=C2	Do Matrix Addition	Do Matrix Addition
LLR9	R1! =R2 &C1! =C2	Invalid	Invalid
LLR9	R1=R2 &C1=C2	Do Matrix Subtraction	Do Matrix Addition
LLR10	R2! =C1	Invalid	Invalid
LLR10	R2=C1	Do Matrix Multiplication	Do Matrix Addition

Test Plan for Trigonometric Operation:

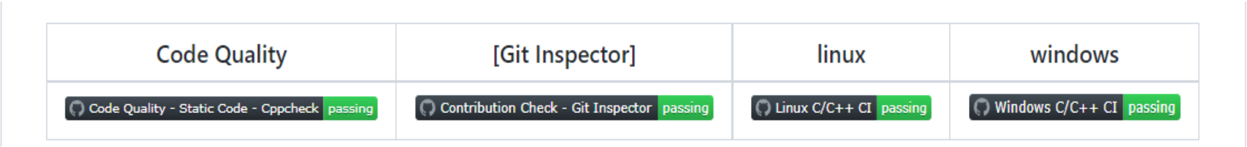
Test ID	Requirement	Expected Output	Type of test case	Actual output
LLR1	sin (30)	-0.988032	Positive Test case	-0.988032
LLR2	cos (30)	0.1542	Positive Test case	0.1542
LLR3	tan (30)	-6.405331	Positive Test case	-6.405331
LLR4	cot (30)	-0.15612	Positive Test case	-0.15612
LLR5	sec (30)	6.4829	Positive Test case	6.4829

Test ID	Requirement	Expected Output	Type of test case	Actual output
LLR6	cosec (30)	-1.012113	Positive Test case	-1.012113
LLR7	sin (60)	-0.304811	Positive Test case	-0.304811
LLR8	cos (60)	-0.9524	Positive Test case	-0.9524
LLR9	tan (60)	0.32004	Positive Test case	0.32004
LLR10	cot (60)	3.124606	Positive Test case	3.124606
LLR11	sec (60)	-1.04996	Positive Test case	-1.04996
LLR12	cosec (60)	-3.280726	Positive Test case	-3.280726

Implementation and Summary:

Github Link: https://github.com/GENESIS2021Q1/Applied_SDLC-Dec_Team_48.git

Git Dashboard:



Code Quality	[Git Inspector]	linux	windows
Code Quality - Static Code - Cppcheck passing	Contribution Check - Git Inspector passing	Linux C/C++ CI passing	Windows C/C++ CI passing

Individual Contribution and Highlights

- Requirements
- Implementation (Add Order in System)
- Created Unity File
- Test cases.

Miniproject4- Calendar Automation [Team]

Module: OOPS with Python

Requirements:

High level Requirements:

ID	Feature	MATLAB v0 Status	Python v0 Status
HR01	GUI	Implemented	Implemented
HR02	Master calendar	Implemented	Implemented
HR03	Faculty calendar	Implemented	Implemented
HR04	Faculty load sheet	Implemented	Implemented
HR05	Showing Available Open Slots based on faculty and modules	Not Available	Not Available
HR06	Output file generated across different computers (windows + Linux)	Not Available	Implemented
HR07	Visualizing data to create Meaningful Insights	Not Available	Not Available
HR08	Calculate Individual Faculty Load	Implemented	Implemented

Low level Requirements:

ID	Feature	High Level ID	MATLAB v0 Status	Python v0 Status
LR01	GUI should allow user to login using credentials	HR01	Not Available	Not Available
LR02	Input Files Based on Different Initiatives and Timelines	HR01	Implemented	Not Available

ID	Feature	High Level ID	MATLAB v0 Status	Python v0 Status
LR03	GUI should get Base Calendar as Input	HR01	Implemented	Implemented
LR04	GUI should get Month and Initiative as Input	HR01	Implemented	Implemented
LR05	GUI should be able to show Conflicts/Warnings	HR01	Implemented	Not Implemented
LR06	Master Calendar: display Month wise	HR02	Implemented	Implemented
LR07	Master Calendar: display Initiative wise	HR02	Implemented	Not Available
LR08	Master Calendar: Differentiate Initiatives (Color Codes/Numbers)	HR02	Implemented	Implemented
LR09	Master Calendar: Appending	HR02	Implemented	Not Available
LR10	Master Calendar: Course code correction	HR02	Implemented	Not Available

Link for standard input template:

https://docs.google.com/spreadsheets/d/1EWYp_1iyK2wLMfKGJOiTJAk5WexZusCP/edit?usp=sharing&ouid=113003694561146884677&rtpof=true&sd=true

- Using the template above, training schedule can be added month wise and initiatives wise
- The name of the input excel sheet MUST be named as "Test vector"(as shown in template)
- Along with the Test vector sheet, "Key" sheet MUST be present under the columns assigned as in the template
- The "Key" sheet must contain all times the 6 fixed initiatives with their respective codes and total list of course code and course title in order to refer for corrections while writing to output files

- Appending additional slots for existing courses is possible by adding just the additional slots in the input file for the same course

Requirements for updating Master calendar using Master calendar as input:

Link for template

2 Slots format - M/A
: https://docs.google.com/spreadsheets/d/1jtKnXV12VE1fH20CGDo4B3uNWRTAhQCWz-hHUDWUe3I/edit?usp=sharing

4 Slots format - M1/M2/A1/A2
: https://docs.google.com/spreadsheets/d/1jVheSPZkOtfNKRNoc_858nwk2UaHCe0gExTNZfZ8vxA/edit?usp=sharing

- Any of the two templates can be used for updating Master calendar month wise on to the drive
- The blocked slots must have the corresponding initiative code in the cell according to the key as shown in the sample data in the template
- The name of the sheet must be the name of the month to be updated
- The "Key" sheet must be present with the fixed list of initiatives and initiative code

App deployment:

- The app is deployed on heroku servers.
- To add/modify new features, you will be required to install HEROKU CLI link
- After installation, open terminal in working directory and enter the following commands:
  - "Heroku git:clone -a Gea calendar"
  - login using Heroku credentials
- After pulling and making changes, enter the following commands to push app and deploy on server
  - Git add.
  - git commit -m "commit message"
  - git push heroku master

Additional features for V1 to do:

- Update key sheet by appending new initiatives/courses list
- Check for duplicate course entries in input file
- Using built in libraries to identify number of days in month, current year and highlight weekend and holidays
- Function to remove a course schedule
- Read multiple months data in one sheet as input file (currently takes data one by one month)

- Calculate individual faculty load

Github link:

https://github.com/Usharani8/Oopswithpython_Calendar_Automation_Team-47.git

Individual contribution and Highlights:

1. Improved implementation of Python Programming
2. Source code management using Git Hub.

Mini project 5 – Hyundai Project [Team]

Module: Applied Model Based Design Module

Contributors:

S. No	NAME	PS Number	Feature
1	Y. Lakshman Swami	40021033	Door locking
2	P. Haritha	40021034	Sunroof control
3	V. V K Mallikarjun	40021038	Sunroof control
4	V. Sai Kumar	40021040	Window control
5	S. Usha Rani	40021045	Power Window
6	N. Kesava Kumar	40021058	Security Systems
7	T. Lakshmi Narayana	40021060	Wiper Control

Requirements:

Door Locking System:

Power door locks (also known as electric door locks or central locking) allow the driver or front passenger to simultaneously lock or unlock all the doors of an automobile or truck, by pressing a button or flipping a switch. Nearly every car model today offers this feature as at least optional equipment.

Sunroof control:

A sunroof is a movable panel that opens to uncover a window in an automobile roof, allowing light and fresh air to enter the passenger compartment. Sunroofs can be manually operated, or motor driven, and are available in many shapes, sizes and styles. While the term sunroof is now used generically to describe any glass panel in the roof, the term "moonroof" was historically used to describe stationary glass panes rigidly mounted in the roof panel over the passenger compartment.

Power Windows:

Power windows or electric windows are automobile windows which can be raised and lowered by pressing a button or switch, as opposed to using a crank handle. Power windows are usually inoperable when the car is not running. This is primarily a security feature. It would be a simple thing to allow electric power windows to be operable when the ignition is turned off, however it would also make the car much easier to steal. Some systems offer the compromise of leaving power applied to the windows until a passenger door is opened at which time the window power is removed.

Security System:

A car alarm is an electronic device installed in a vehicle to discourage theft of the vehicle itself, its contents, or both. Car alarms work by emitting high-volume sound (often a vehicle-mounted siren, klaxon, pre-recorded verbal warning, the vehicle's own horn, or a combination of these) when the conditions necessary for triggering it are met. Such alarms may also cause the vehicle's headlights to flash, may notify the car's owner of the incident via a paging system, and may interrupt one or more electrical circuits necessary for the car to start. Although inexpensive to acquire and install, the effectiveness of such devices in deterring vehicle burglary or theft when their only effect is to emit sound appears to be negligible. He individual triggers for a car alarm vary widely, depending on the make and model of the vehicle, and the brand and model of the alarm itself (for aftermarket alarms). Since aftermarket alarms are designed to be universal. Although car alarms of some kind have been available since the beginning of the automobile era, the dramatic increase in their installation in the 1980s and 1990s coupled with the fact that nearly all types of car alarms are easily triggered accidentally (frequently because of high sensitivity settings) means that people who hear them often ignore them.

Wiper Control:

A windscreen wiper, windshield wiper or wiper blade (American English) is a device used to remove rain, snow, ice, washer fluid, water, and/or debris from a vehicle's front window so the vehicle's operator can better see what's ahead of them. Almost all motor vehicles, including cars, trucks, buses, train locomotives, and watercraft with a cabin—and some aircraft—are equipped with one or more such wipers, which are usually a legal requirement. On some vehicles, a windscreen washer system is also used to improve and expand the function of the wiper(s) to dry or icy conditions. This system sprays water, or an antifreeze window washer fluid, at the windscreen using several well-positioned nozzles. Most wipers are of the pivot (or radial) type: they are attached to a single arm, which in turn is attached to the motor. These are commonly found on many cars, trucks, trains, boats, airplanes, etc. Wipers may be powered by a variety of means, although most in use today are powered by an electric motor through a series of mechanical components, typically two 4-bar linkages in series or parallel.

References:

https://mechvibesblog.com/control-modules/

https://youtu.be/7zzpTH9Hl-s

https://youtu.be/4KS8jUCCbkQ

https://youtu.be/g6MgZY6Gbc8

https://youtu.be/PdH792tFV7M

https://youtu.be/r6gbQTt1Blc

Design:

This project was implemented using MATLAB.

Individual contribution:

Design the MATLAB scripting for Placement Data Full Class data sheet from google information.

Mini project 6 – Wiper Control [Team]

Module: - Mastering Microcontrollers with Embedded Driver Development Module

WIPER CONTROL SYSTEM

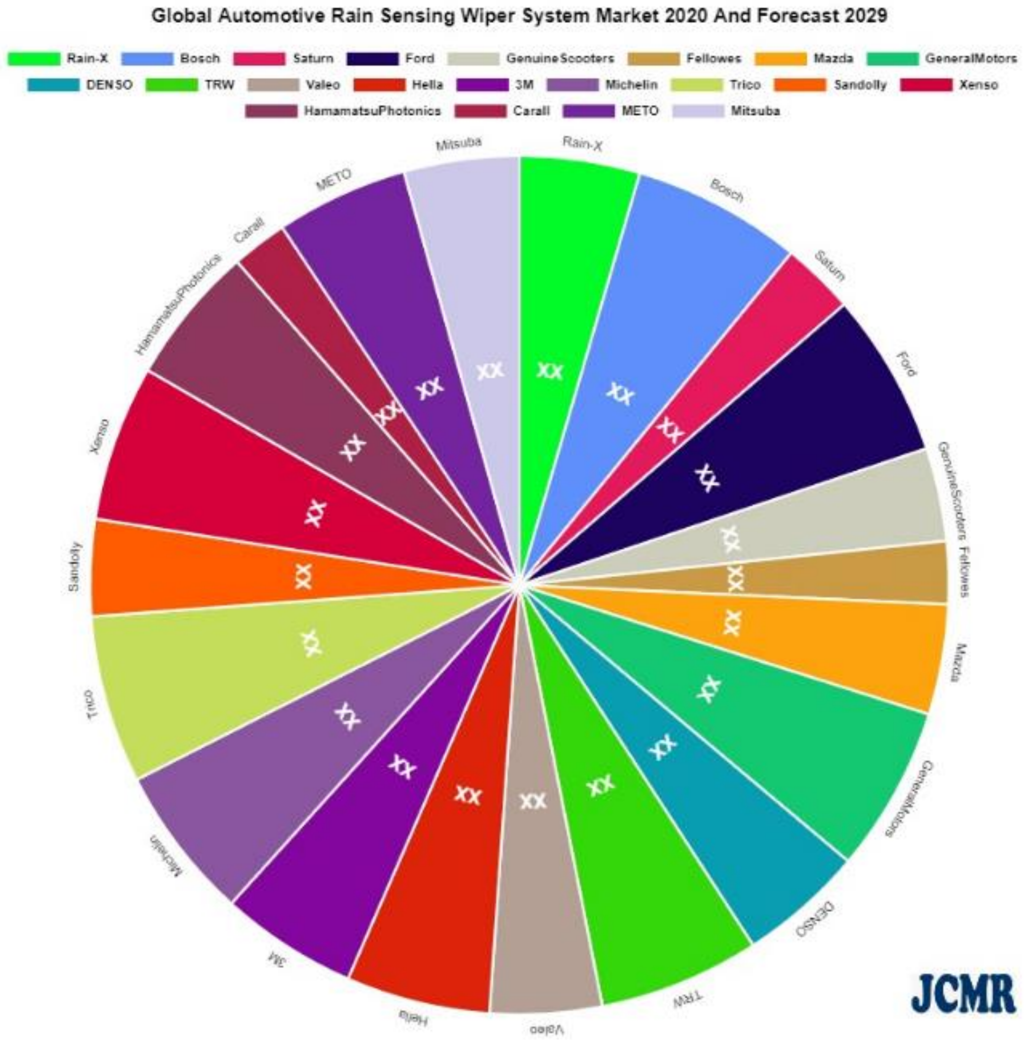
Introduction:

Automotive wipers form an essential part for any vehicle. They perform to remove water, ice, snow, and dust from a windshield of a vehicle. An automotive wiper is either powered by an electric motor or pneumatic power. Almost all motor vehicles including cars, trucks, buses, train locomotives and watercraft with a cabin are equipped with one or more such wipers. The automotive wiper market is multiplying as there is an exponentially increased production of automobiles globally.

Features:

- To achieve high safety
- To reduce manpower
- To increase the efficiency of the vehicle
- To reduce the workload
- To reduce the vehicle accident
- To reduce the fatigue of workers
- To high responsibility
- Less Maintenance cost

State of Art



SWOT Analysis:

Strength:

- It is possible to operate Manually/automatically by proving on/Off switch
- Improve Visibility of car in rain. Can drive easily in any climatic situation.

Weakness:

- This system applied in the case of water falling on the class only.
- Addition cost is required to install this system to four-wheeler.

Opportunities:

- To increase automation in vehicle driving system

- To dispense with troublesome wiper operation needed when rainfall condition change or driving condition change, including the car speed and entry to or exit from tunnels.
- To operate the wiper with response to changing rainfall or driving conditions, thus keeping the driver’s windshield clear.

Threats:

- Dust particles and non-conductive particles accumulated on the surface of sensors cannot be detected by conductive sensors.

4W’s & 1H:

Who:

- A wiper speed control system for an automotive wiper controls the operational speed of a wiper in accordance with rain conditions.

What:

- Vehicles are now available with driver-programmable intelligent (automatic) windscreen wipers that detect the presence and amount of rain using a rain sensor.

Where:

- It is located underneath the dashboard, above the brake and accelerator pedal, and is responsible for the complete operation of the windshield wiper system.

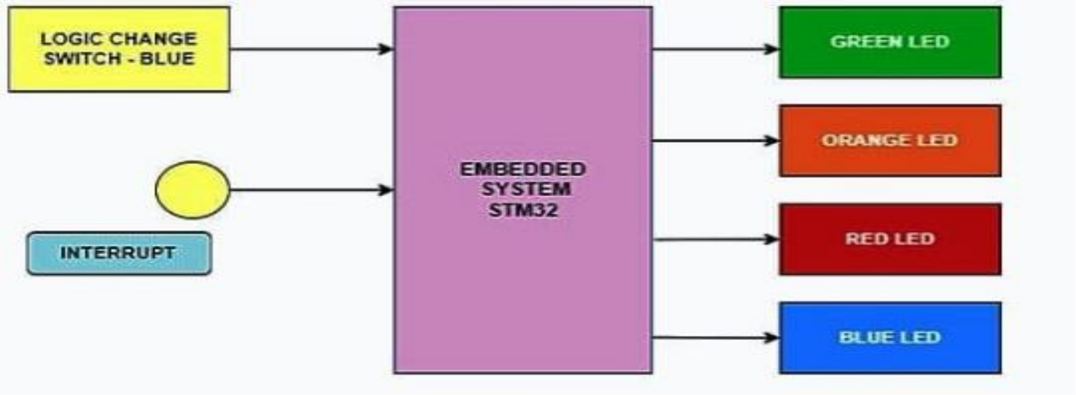
When:

- Whenever the water hit a dedicated sensor that located on windscreen, it will send a signal to move on the wiper motor. Once water is not detected by sensor, the wiper will automatically stop. This will help the driver to give more concentration and reduce the car accident probability.

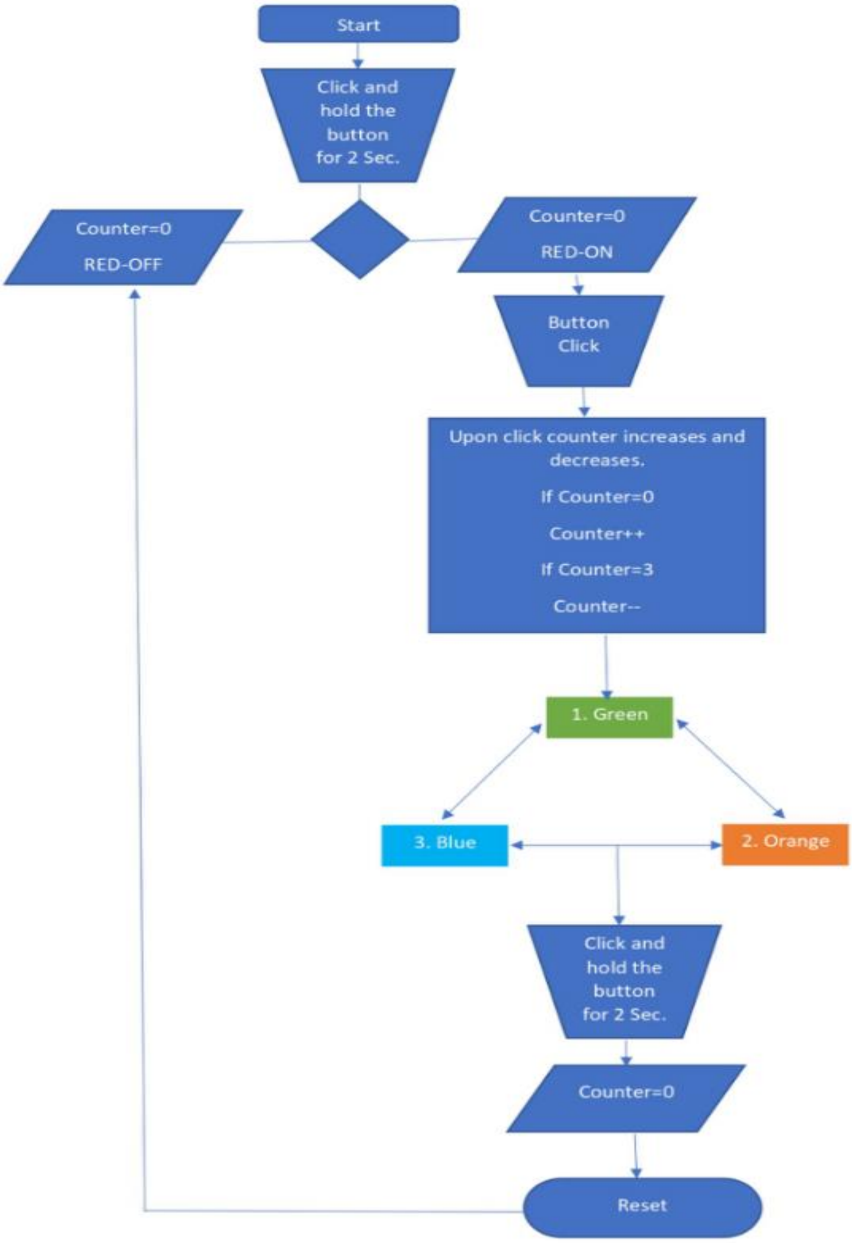
How:

Windshield wipers are controlled by the stalk on the right side of your steering wheel. Simply moving the stalk down will turn your windshield wipers on. Moving the stalk down will turn your wipers on.

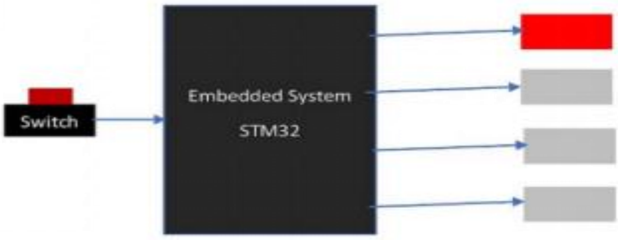
BLOCK DIAGRAM:



FLOW CHART:



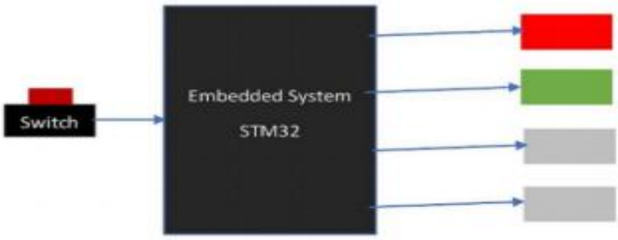
CASE STUDY



User holds button for 2 seconds.

Turns – Red LED ON.

Stage - 0



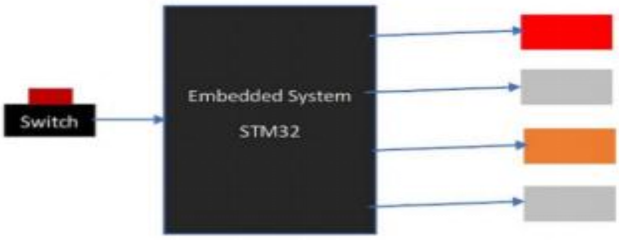
User click's button 1 time(s).

Turns – Green LED ON

Wiper is in Level – 1.

Frequency - 1 Hz

Stage - 1



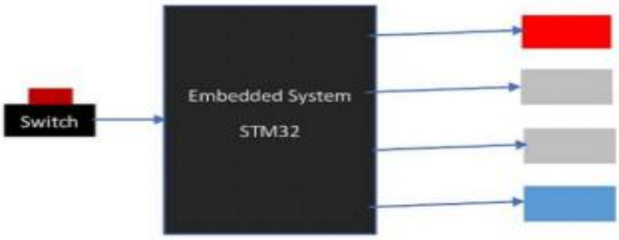
User click's button 1 time(s).

Turns – Orange LED ON

Wiper is in Level – 2.

Frequency - 4 Hz.

Stage - 2



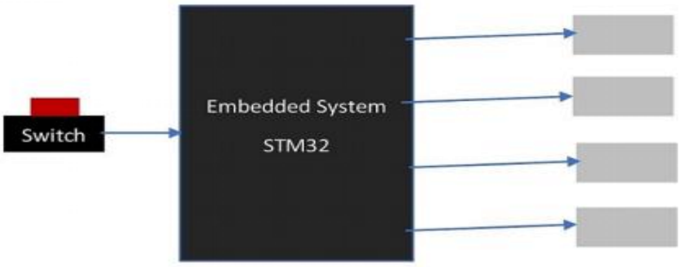
User click's button 1 time(s).

Turns – Blue LED ON

Wiper is in Level – 3.

Frequency – 8Hz.

Stage - 3



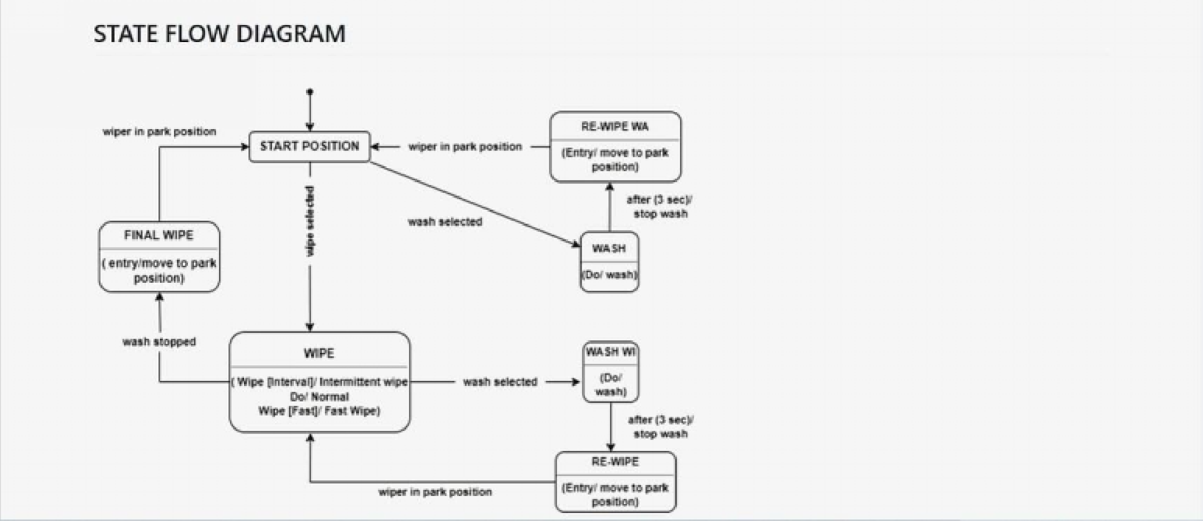
User holds button for 2 seconds.

Turns – All LED OFF.

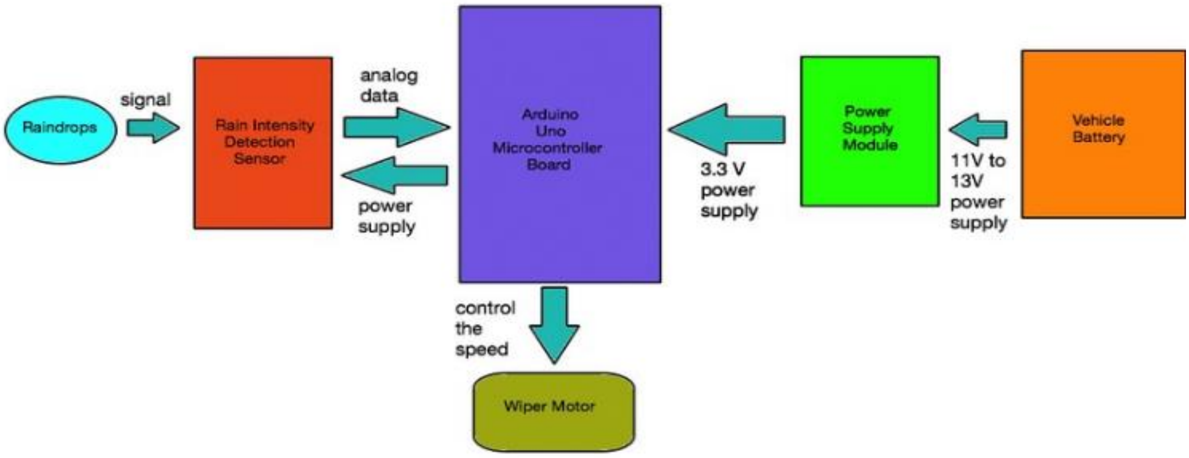
Stage - 0

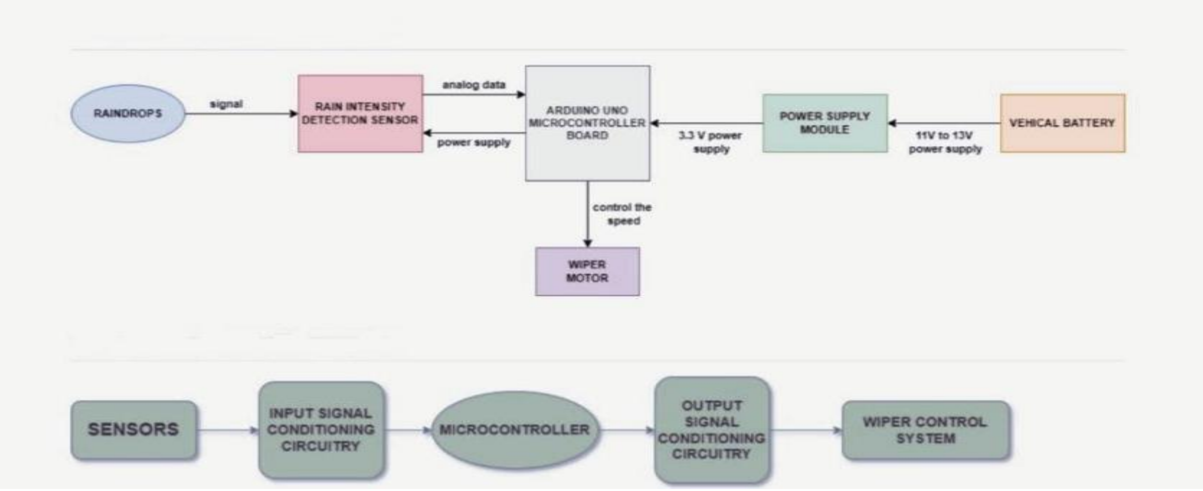
According to the requirement user can change the wiper levels, this process is shown in forward direction, and the three LEDs other than red LED can be changed from level to level.

STATE FLOW:



SYSTEM DESIGNS:





High Level Requirements:

ID	Description
HLR1	These systems detect droplets of rain on the windshield and automatically turn on and adjust the wiper system in accordance with the level of precipitation.
HLR2	A windscreen wiper or windshield wiper is a device used to remove rain, snow, ice and debris from a windscreen or windshield.
HLR3	Quality and reliability wiper systems meet the highest technical requirements and are the basis for vehicles with sophisticated features.
HLR5	Almost all motor vehicle, including trains, aircraft, and watercraft, are equipped with such wipers, which are usually an essential requirement.
HLR6	Our project brings forward this system to automate the wiper system having no need for manual intervention.

Low Level Requirements:

ID	Description
LLR1	A new mechatronic reversing system can now be used to clean the windshield with two wiper arms, whereby one wiper arm is powered directly and the other indirectly using a connection link.
LLR2	Wiper motor is automatically ON during the time of rainfall.
LLR3	Existing system manually used control stalk to activate wiper and the process of pulling up wiper is difficult to be handled.
LLR4	Lower-level parsing. Under the hood, the Requirement class does most of the heavy lifting. class requirements.

LLR5	These systems detect droplets of rain on the windshield and automatically turn on and adjust the wiper system.

Implementation and Summary:

Git Link: https://github.com/GENESIS-2022/MasteringMCU-Team36.git

Individual Contribution and Highlights:

1. Wiper System using C Programming
2. Source code management using GitHub
3. Requirements & Implementation (start-up's, STM32F407XX.H)

Role in Project Team:

1. Programmer: Done Programming for Wiper System
2. Integrator: Integrated all the codes
3. Tester: Writing Test cases and testing the integrated code.

Mini project 7 – LAMBHORGINI[Team]

Module: Automotive Systems

Requirements:

In this project Lamborghini we have taken following features and I have contributed to Parking System Feature

1. Wiper Control System
2. Sunroof control system
3. Power window system
4. Seat belt control

Research:

Automation is a necessity in our day-to-day life because it not only seeks to improve the quality of life for humans at both home and work.

It allows the distribution of both quality products and services to be made available at faster rates, reduces time and human error.

These days the technology is developing at very high-speed innovation and advancements is there in all the sectors.

SWOT- Strengths, and Weakness, Opportunities Threats:

Strengths:

Comfortable, Easy to use, Fast and accuracy operation, Easy transportation

Weakness:

High cost

Opportunities:

it's a modern way of approach, High use in modern days

Threat:

if sensor is damages, operation of the system will stop

4W's and 1'H:

WHO:

Any user who is travelling in the car can use this system.

WHAT:

It’s a Body control model of a car which describes different features

WHEN:

There is a requirement of vehicle these systems are used

WHERE:

Used in automatic cars

HOW:

By developing an embedded system which is user friendly and can be implemented without difficulty.

Detail requirements:

High Level Requirements:

Number	Requirements	Description	Features
HLR1	Rain and Humidity sensor	checks the intensity of rain and humidity	Wiper Control
HLR2	Semi-Automatic control	Operated automatically or manually	Wiper Control
HLR3	Speed control	High, Medium, Low	Wiper Control
HLR4	Dry mode	front and back wiper system	Wiper Control
HLR5	Obstacle Sensor	when obstacle detected the operation is delayed for 10seconds.	Power window
HLR6	Both driver and passenger control	operated with the first received command (either by driver or passenger)	Power window
HLR7	Dust detected sensor	window will be closed when the dust is detected	Power window
HLR8	Temperature and Rain sensor	Which detects intensity of rain and temperature	Sunroof Control
HLR9	Obstacle sensors	when obstacle is detected operation of window will be stopped	Sunroof Control
HLR10	Pressure control alarm	alarm buzzes when seat belt is not buckled up even the pressure is applied	Seatbelt Control

Number	Requirements	Description	Features
HLR11	Open airbag	depending upon the tension airbag will be opened	Seatbelt Control
HLR12	buckles and its types	lap belts, diagonal belts, the harness, combination of lap and diagonal belts.... etc.	Seatbelt Control
HLR13	Height adjustment of seat belt	Height adjustment of seat belt	Seatbelt Control

Low Level Requirements:

Number	Requirements	Description	Features
LLR1	Length of wipers	Depends on the model of the car	Wiper Control
LLR2	Direction of wipers	Same and Opposite Directions (180 degrees)	Wiper Control
LLR3	Semi controlled	manually/automatic	Power window and Sunroof
LLR4	Length of the belt	Depends on the model of the car	Seat Belt control
LLR5	Engine On	Can be operated manually or automatically	wiper control, power window, sunroof control
LLR6	Engine Off	Can be operated manually	Power window and Sunroof control

Design:

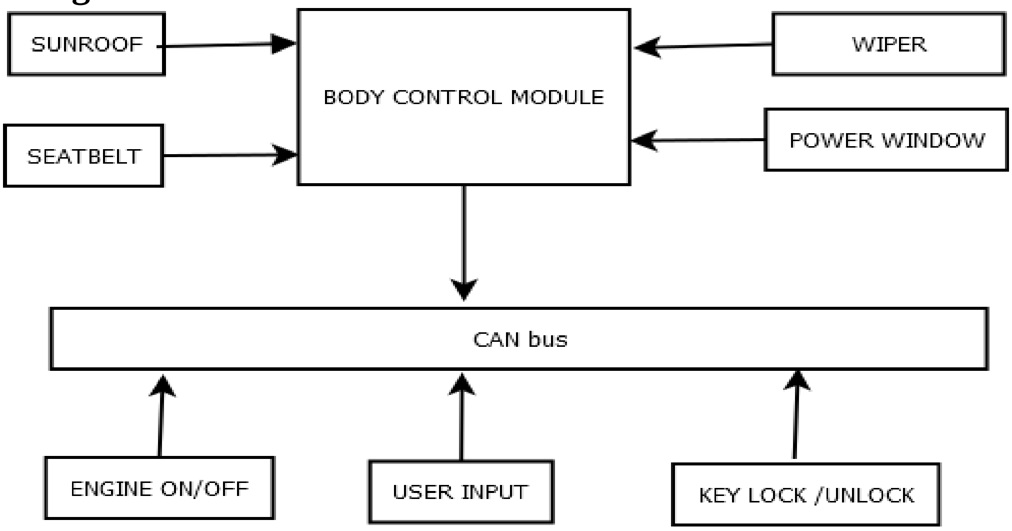


Figure 14 Structure Diagram

Implementation and Summary:

Git Link:

Link: https://github.com/pasumarthiharitha/Automotive_Lamborghini.git

Individual Contribution and Highlights:

1. Power Window Study
2. Source code management using GitHub

Role in Project Team:

1. Designer: Done Designing for Project
2. Researcher: Done case study for Power window

Mini project 8 – EV Bike [Team]

Modules:

1. MATLAB
2. MATLAB Script

Requirements

Motor Performance:

1. Our Arrow M1 has a Mid Drive IPM motor which can produce 7.2 kW power and 40 Nm torque. We find these figures to be a nice balance of drivability and efficiency.
2. Arrow M1 has an acceleration time from 0 to 60 km/hr of 6.5 seconds.
3. Top speed of our Arrow M1 is 100 km/hr

Battery Performance:

1. We are using a Lithium polymer battery to reduce weight and thereby increase fuel efficiency, performance and handling.
2. A range of 220 km is class leading due to our battery being the biggest at 4.6 kWh.
3. Charging times of our Arrow M1 is higher than the competition at 7.15 hours but we make up for it in the range section.
4. We also offer fast charging.

Braking Performance:

1. Our Arrow M1 also uses combi braking system and use disc brakes for both front and back wheels.
2. Braking performance is on par with the competition.

Wheel Performance:

1. Our Arrow M1 uses Alloy wheels at 12 inches diameter.
2. We use a 90 section, 90 profile tire for a balance between grip, efficiency and ride quality.

Suspension Performance:

1. We use Mono shocks for rear and single fork for front.

Dimensions:

1. Our kerb weight is 110 kg which is just 2 kg heavier than the Ather 450X while having a substantially bigger battery and more powerful motor.
2. Length, Height and Weight are all comparable to the competition.
3. Wheelbase is 1370 mm is the longest in the segment.
4. With a seat height of 782 mm it is accessible for a wide range of people in terms of height.

Implementation and Summary

Submission: Submitted in GEA Learn

Individual Contribution and Highlights

1. Done in MATLAB Script

Role in Project Team

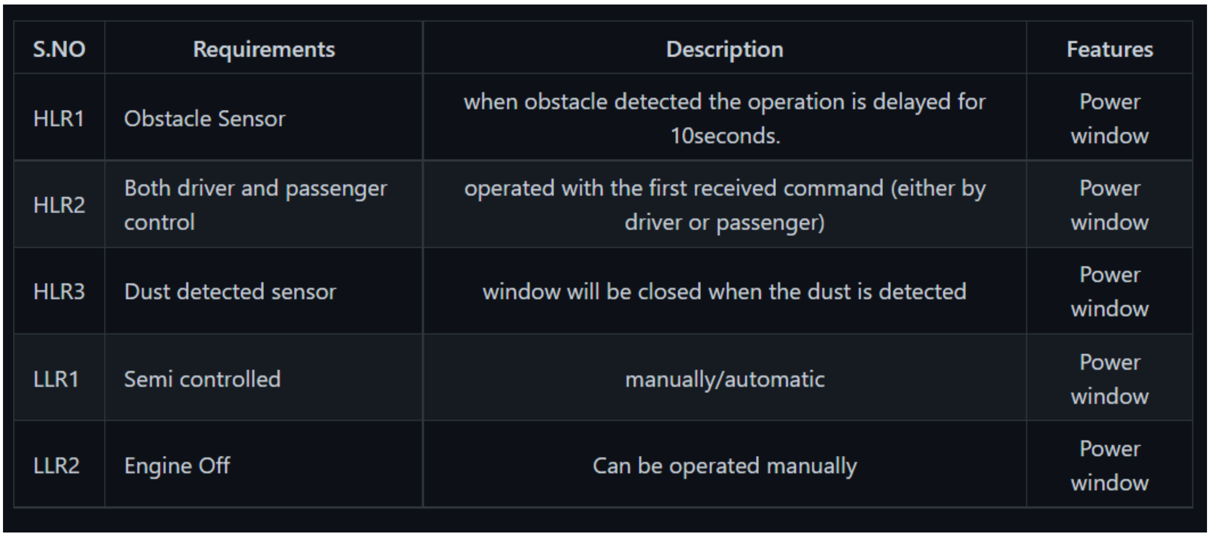
1. Done MATLAB scripting for EV Bike
2. Researcher: Done case study for EV Bike

Mini project 9 – Power Window [Individual]

Modules

1. AUTOSAR
2. Git

Requirements



S.NO	Requirements	Description	Features
HLR1	Obstacle Sensor	when obstacle detected the operation is delayed for 10seconds.	Power window
HLR2	Both driver and passenger control	operated with the first received command (either by driver or passenger)	Power window
HLR3	Dust detected sensor	window will be closed when the dust is detected	Power window
LLR1	Semi controlled	manually/automatic	Power window
LLR2	Engine Off	Can be operated manually	Power window

Design

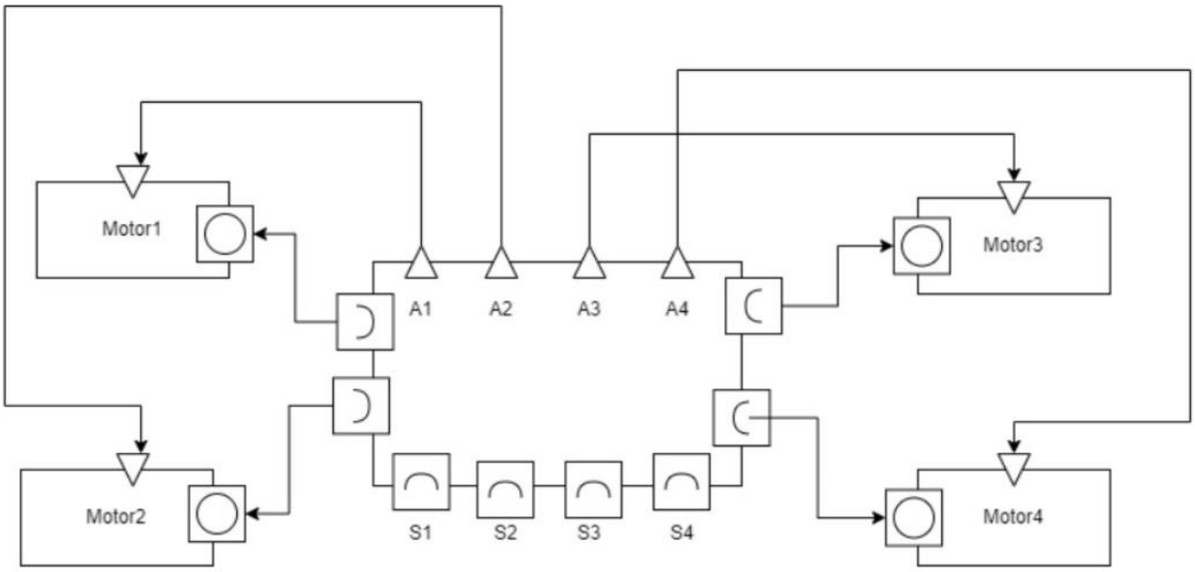


Figure 15 VFB Diagram

Implementation and Summary

Git Link:

Link: https://github.com/pasumarthiharitha/Automotive_Lamborghini.git

Individual Contribution and Highlights

1. Power Window System Case Study
2. AtomicSwComponent
3. SWCInternalBehavior
4. SWCImplementation

Learning of Essential of python

Outcomes:

Lesson 1: print()

It’s the tradition. Print **“Hello World!”**

Lesson 2: Variables

Variables are probably the most fundamental building blocks in high-level programming. Learn and Practice variables with this Python course.

Lesson 3: Data Types

Learn Python data types: int, float and str. They have different functions to store, process and represent different types of data.

Lesson 4: Type Conversion

Sometimes it makes sense to convert Python data types between each other (when possible). int, float and str are also functions for converting data types. And when you’re not sure of a variable’s type, you can use type function!

Lesson 5: Data Structures

In this Python course data gets a bit more structured. Python lists, dictionaries and tuples are famous sequences that can contain various type of data. You will learn most common Python data structures along with functions to create them.

Lesson 6: Lists

A closer look at Python lists and some of their built-in methods and functions. This lesson introduces a lot of fundamental Python topics but it’s so worth it. Make sure you take your time and get comfortable with Python lists as you will be using them a lot.

Lesson 7: Tuples

Python tuple concept, difference between tuple and list along with some tuple examples and built-in tuple methods in Python.

Lesson 8: Dictionaries

Yet another cool Python data structure: dictionaries will be unraveled in detail. Python Dictionaries new perspectives to data such as usage of key and unindexed structure.

Lesson 9: Strings

Good ol’ strings revisited. More string methods, more built-in functions and more string examples. When you think about string, it’s everywhere. Web data, reports, news, social media, books, descriptive text, user input, survey answers, gui and many more. So it deserved a revisit.

Lesson 10: len()

A practical Python function to get length of different types of data in Python.

Lesson 11: .sort()

This list method can be very useful to sort data in a list. Later in intermediate lessons its cousin **sorted** function will be introduced along with slightly more advanced concepts. sort is a list method while sorted is a built-in function.

Lesson 12: .pop() method

In this course an interesting dictionary method, pop, will be introduced. It's also an opportunity to polish our Python dictionary knowledge.

Lesson 13: input()

One of the most exciting Python function for many beginners, input allows interacting with users. You can ask questions or share messages with users and harvest their answers to use them in your computer program. Input also provides opportunities to practice Python data types.

Lesson 14: range() |

Range function is practical and it can be used to create range objects in Python. Range objects are very useful when used with for loops and they can also be used to create lists of numbers (int or float) with different steps (default 1).

Lesson 15: Error Handling

Errors are a way of computers to say something's not right or they don't understand something when we write computer programs. It can be very useful to know what different errors mean in Python and how to handle them. As you get familiar with more errors they will stress you less when coding!

Lesson 16: Defining Functions

In computer world you are not limited to built-in functions only. Often it makes sense to construct a user-defined function and tuck your pieces of code in there. Re-usability, structure, sophisticated Python operations… You are advancing in Python programming. Congrats!

Lesson 17: Slicing

Slicing notation is a must know. It may seem a bit weird at first but Python slicing notations are actually addictively cool. Slice many different types of data, from the beginning to the end, end to beginning and with steps too!

Lesson18:PythonOperators

You have probably been using Python Operators all along. In this course you will be officially introduced to different Python operators

Learnings of Electrical vehicles

Domain Knowledge Videos

1. Understanding Hill Start Assist!- https://youtu.be/aXEPnWgRnjk?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
2. Differential | How does it work?- https://youtu.be/nC6fsNXdcMQ?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
3. Seatbelt | How does it work?- https://youtu.be/uRaU1HMJyCo?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
4. Understanding Wheel Alignment !- https://youtu.be/7d2K_mKgsZ0?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
5. How does the Steering Wheel automatically returns to its center?- https://youtu.be/wLbs8kBXgrw?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
6. Understanding your Car's Steering & Power Steering !- https://youtu.be/em1O8mz7sF0?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
7. Understanding Anti-lock Braking System (ABS) !- https://youtu.be/98DXe3uKwfc?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
8. Torque Converter, How does it work ?- https://youtu.be/bRcDvCj_JPs?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
9. Why you should not PARTIALLY press the Clutch ?- https://youtu.be/_hKvS6xTC0E?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
10. Clutch, How does it work ?- https://youtu.be/devo3kdSPQY?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
11. Electric cars vs Petrol cars- https://youtu.be/ewcWN-rHQ6Q?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
12. How does an Electric Car work ? | Tesla Model S- https://youtu.be/3SAxXUIre28?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
13. Understanding PLANETARY GEAR set !- https://youtu.be/ARd-Om2VyiE?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
14. Automatic vs Manual Transmission- https://youtu.be/auQgOtveQi0?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
15. Working of Dual Clutch Transmission (DSG)- https://youtu.be/lFAtc-zOKZs?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
16. Manual Transmission, How it works ?- https://youtu.be/wCu9W9xNwtI?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
17. Automatic Transmission, How it works ?- https://youtu.be/u_y1S8C0Hmc?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
18. Petrol (Gasoline) Engine vs Diesel Engine- https://youtu.be/bZUoLo5t7kg?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
19. Diesel Engine, How it works ?- https://youtu.be/DZt5xU44IfQ?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
20. How a Differential works ?- https://youtu.be/SOgoejxzF8c?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg

Electrical vehicle basics:

1. EPT trainings learning content

   Video link:EPT Trainings Learning Content | Microsoft Stream (mcas.ms)

   1. EV Architecture and components

   2. Inverter Hardware and software -part 1

   3. Inverter Hardware and software -part 2

   4.EV Lab and testing training

   5.Worst case analysis Tolerance analysis

   6. Design calculations -Inverter losses & thermal design

   7.Hardware Simulation and control simulation of DC-DC converter topologies

   8.software closed loop control DC -DC Converter topologies

2.BMS (Battery management system)

   Video link EV Learning Content | Microsoft Stream (mcas.ms)

   1.system requirements, specification feature and DFMEA

   2.BMS -software application and Algorithm

   3.FUSA-1

   4.FUSA-2

   5.Wireless BMS

   6.BMS testing and BI HIL

   7.EV lab Demo and Amaze BMS

   8.Overall BMS architecture and platform

3.System level -conventional/EV

   Video link: System Level - Conventional/EV | Microsoft Stream (mcas.ms)

   1. Inviting for Battery Management System
   2. Introduction to Functional Safety
   3. Function Safety Session 2
   4. Overview of Engine After treatment System, Engine Sensors and transmission System
   5. Inviting for Battery Management System – Session 2
   6. Overview of different Vehicle architectures
   7. DC – DC converter