Mastering Embedded System Online Diploma

https://www.learn-in-depth.com/

Second Term (Final Project)

Eng. Mohamed Abd El-Naby Mohamed

My Profile:

https://www.learn-in-depth.com/online-diploma/mahameda.naby@gmail.com

Table of Contents

LIST OF FIGURES 4
Description 5
System Specifications 6
ECU1 6
ECU2 6
ECU3 6
System Assumptions: 6
Project Overview: 7
i- All System 7
ii- Calls 7
System Architecture 9
1- Case study 9
2- Method 9
3- Requirement 9
4- Space exploration/partitioning 10
5- System Analysis 11
i- Use Case Diagram 11
ii- Simple Activity Diagram 13
iii- Sequence Diagram (UML) 15
6- System Design 18
Boot Sequence of STM32 18
Map File 19
Memory dump 20
Some of Symbols 21
ELF image details 22
Hardware Simulation 23
Test Cases 23

- ECU1 23
- ECU2 25
- ECU1 27

LIST OF FIGURES

FIGURE 1:PROJECT OVERVIEW 7
FIGURE 2:ECU1 CALLS 7
FIGURE3 :ECU2 CALLS 8
FIGURE 4:ECU3 CALLS 8
FIGURE 5:SYSTEM ARCHITECTURE 9
FIGURE 6:SYSTEM REQUIREMENT 9
FIGURE 7:SYSTEM PARTITIONING 10
FIGURE 8:ECU2 USE CASE DIAGRAM 11
FIGURE 9: ECU1 & ECU3 USE CASE DIAGRAM 12
FIGURE 10:ECU2 ACTIVITY DIAGRAM 13
FIGURE 11:ECU1 ACTIVITY DIAGRAM 13
FIGURE 12:ECU3 ACTIVITY DIAGRAM 14
FIGURE 13:ECU1 UML DIAGRAM 15
FIGURE 14:ECU2 UML DIAGRAM 16
FIGURE 15:ECU3 UML DIAGRAM 17
FIGURE 16:SYSTEM DESIGN 18
FIGURE 17:SEQUENCE 19
FIGURE 18:VECTOR TABLE POSITION IN MAP FILE 19
FIGURE 19:MEMORY DUMP WITH DEBUG SECTION 20
FIGURE 20:SYMBOLS OF ELF IMAGE 21
FIGURE 21:ELF IMAGE DETAILS 22
FIGURE 22:SIMULATION TEST 23
FIGURE 23:ECU1 TEST CASES 23
FIGURE 24:ECU1 TEST CASES 24
FIGURE:25 ECU1 TEST CASES 24
FIGURE 26:ECU1 TEST CASES 24
FIGURE 27:ECU2 TEST CASES 25
FIGURE 28:ECU2 TEST CASES 25
FIGURE 29:ECU2 TEST CASES 26
FIGURE 30:ECU2 TEST CASES 26
FIGURE 31:ECU3 TEST CASES 27
FIGURE 32:ECU3 TEST CASES 27
FIGURE33 :ECU3 TEST CASES 27
FIGURE 34:ECU3 TEST CASES 28

Private Parking Garage

Description

This Project aims to make a design for a private parking garage area for people in a specific area or garage for a company. This system is split into three ECUs:

First ECU for the entrance gate

This ECU is responsible for the gate that is based on the servo motor, the RFID reader based on UART for the user interface, Buzzer for the beep sound when the driver enters an unauthorized ID, and Some LEDs like green and red for authorized and unauthorized ID.

Second ECU for admin dashboard

This ECU is responsible for the admin privileges to add, delete, and edit driver data. The system may have more than one admin, each one of them has its username and password.

The ECU has an LCD and keypad as an admin interface, a UART device for entering admin data, and a seven-segment to display the number of available slots in the garage.

Third ECU for the exit gate

This ECU is responsible for the gate that is based on the servo motor, the RFID reader based on UART for the user interface, Buzzer for the beep sound when the driver enters an unauthorized ID, and Some LEDs like green and red for authorized and unauthorized ID.

LCD will display some messages for the driver to determine what will do.

The Whole system is connected together. when a driver enters a valid ID the ECU1 sends data through SPI to ECU2, and the ECU2 starts checking if the ID is valid or not and checking if it is inside the garage and wants to exit from the entrance gate then the ECU2 return the result of checking to ECU1 to display the result of computing on the LCD is valid ID or invalid ID.

When the driver wants to exit the garage space and enter the ID the ECU3 sends the driver data to ECU2, and the ECU2 starts checking if the ID is valid or not and checking if it is outside the garage and wants to enter from the exit gate then the ECU2 return the result of checking to ECU1 to display the result of computing on the LCD is valid ID or invalid ID.

System Specifications

ECU1

1- Control The servo motor of the entrance gate.
2- Display the states on LCD

ECU2

1- Holds Predefined admins data.
2- Validate the driver data.
3- Display admin dashboard.
4- Display number of available slots in garage.

ECU3

1- Control The servo motor of the exit gate.
2- Display the states on LCD

System Assumptions:

1- The Distance between the ECU1 and ECU2 is shorter than 50 cm.
2- The Distance between the ECU3 and ECU2 is shorter than 50 cm.
3- Controller maintenance is not modeled.
4- Sensors never fail.
5- Communication wires are never damaged.

Project Overview:

i- All System

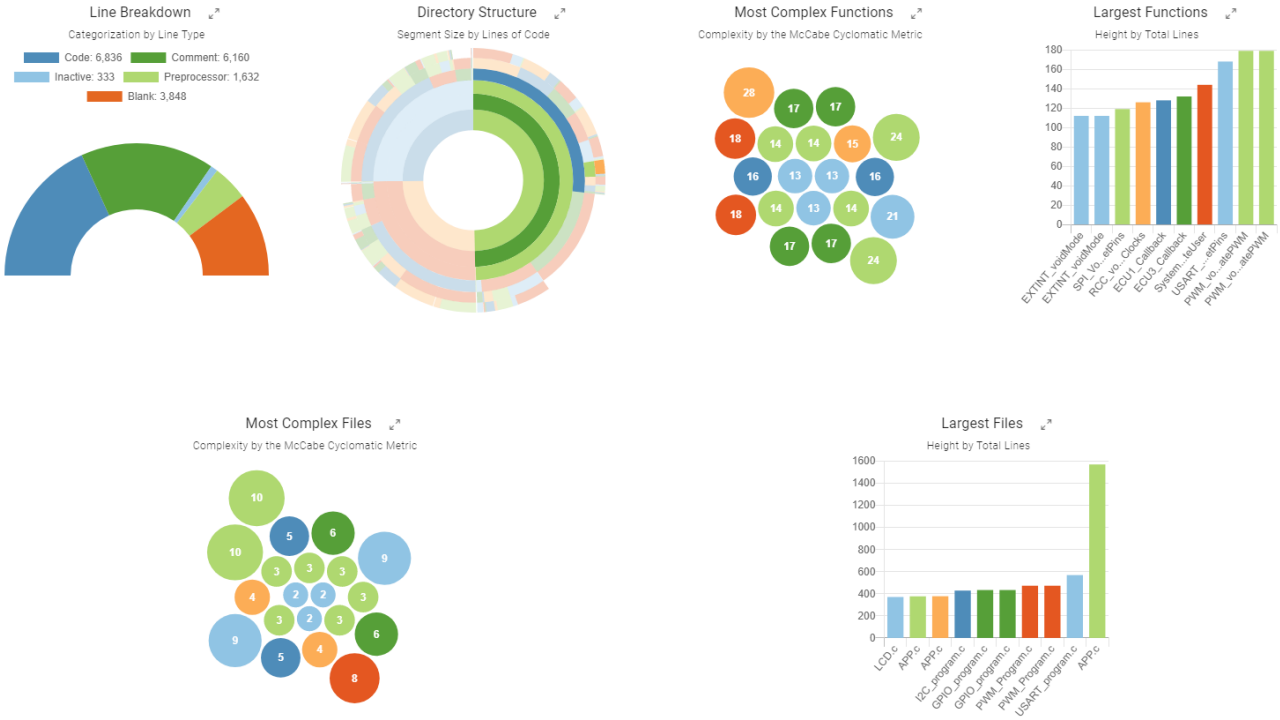


Figure 1:Project Overview

ii- Calls

- ECU1

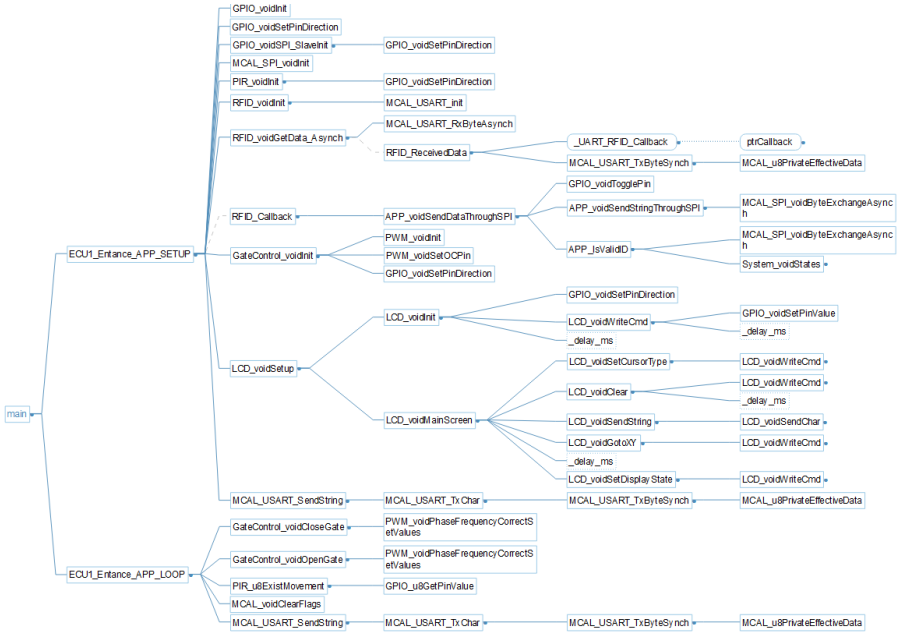


Figure 2:ECU1 Calls

- ECU2

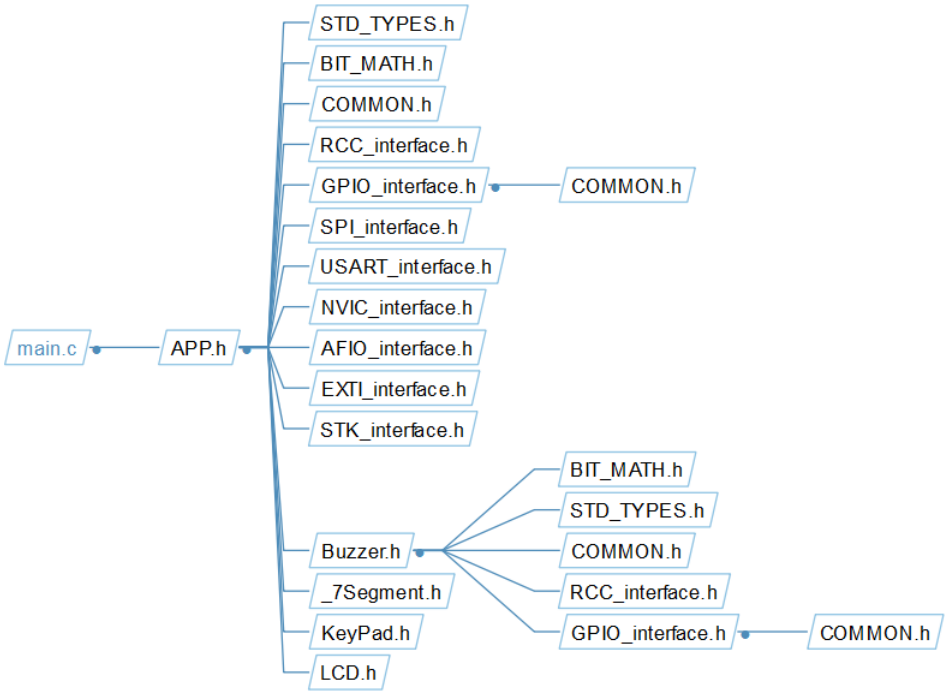


Figure3 :ECU2 Calls

- ECU3

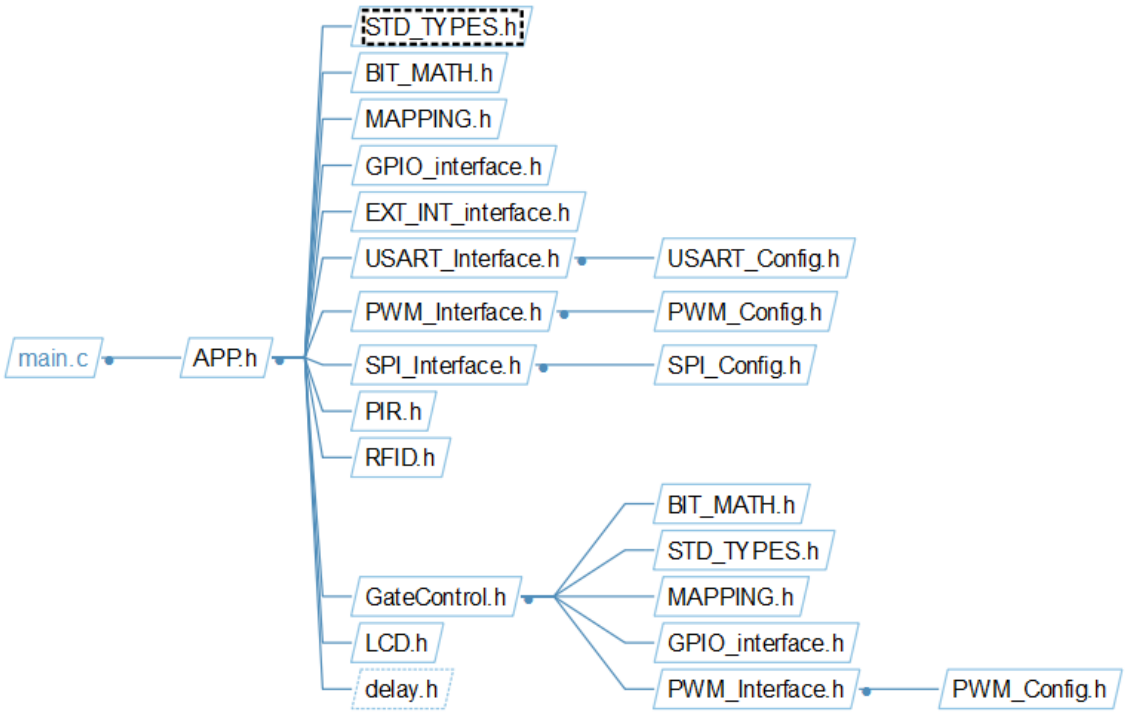


Figure 4:ECU3 Calls

System Architecture

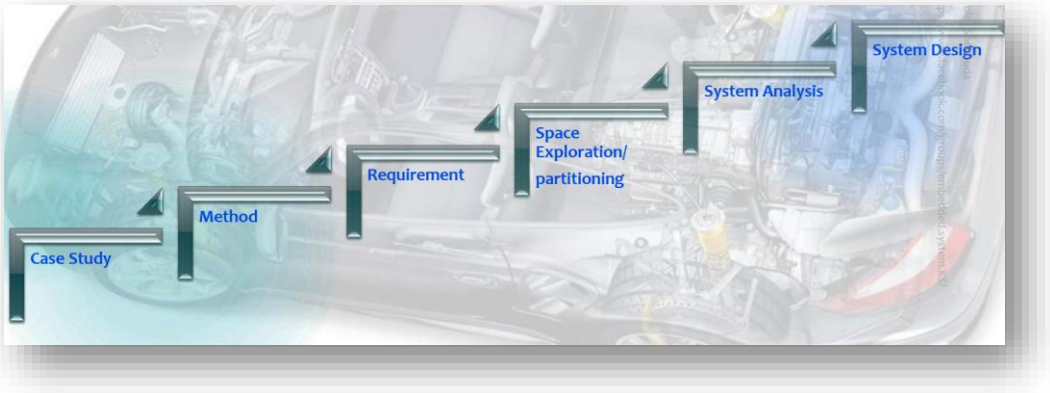


Figure 5:System Architecture

1- Case study

software that controls the private parking garage.

2- Method

Adaptive Technique: Agile Scrum Methodology

3- Requirement

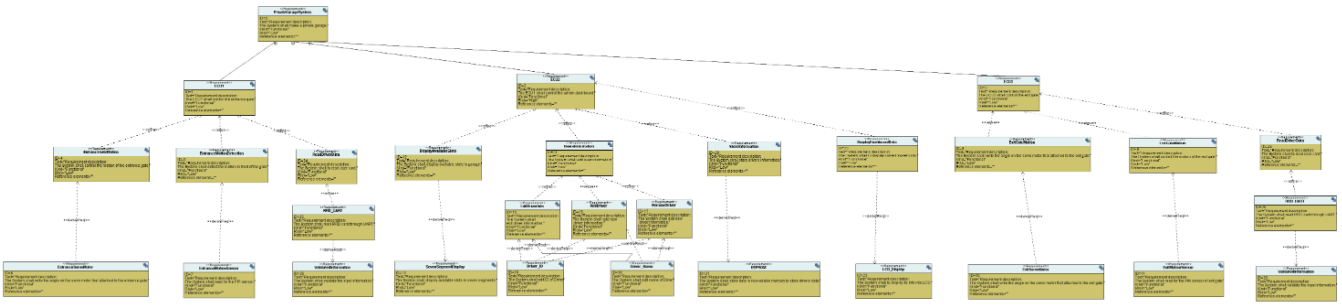


Figure 6:System Requirement

4- Space exploration/partitioning

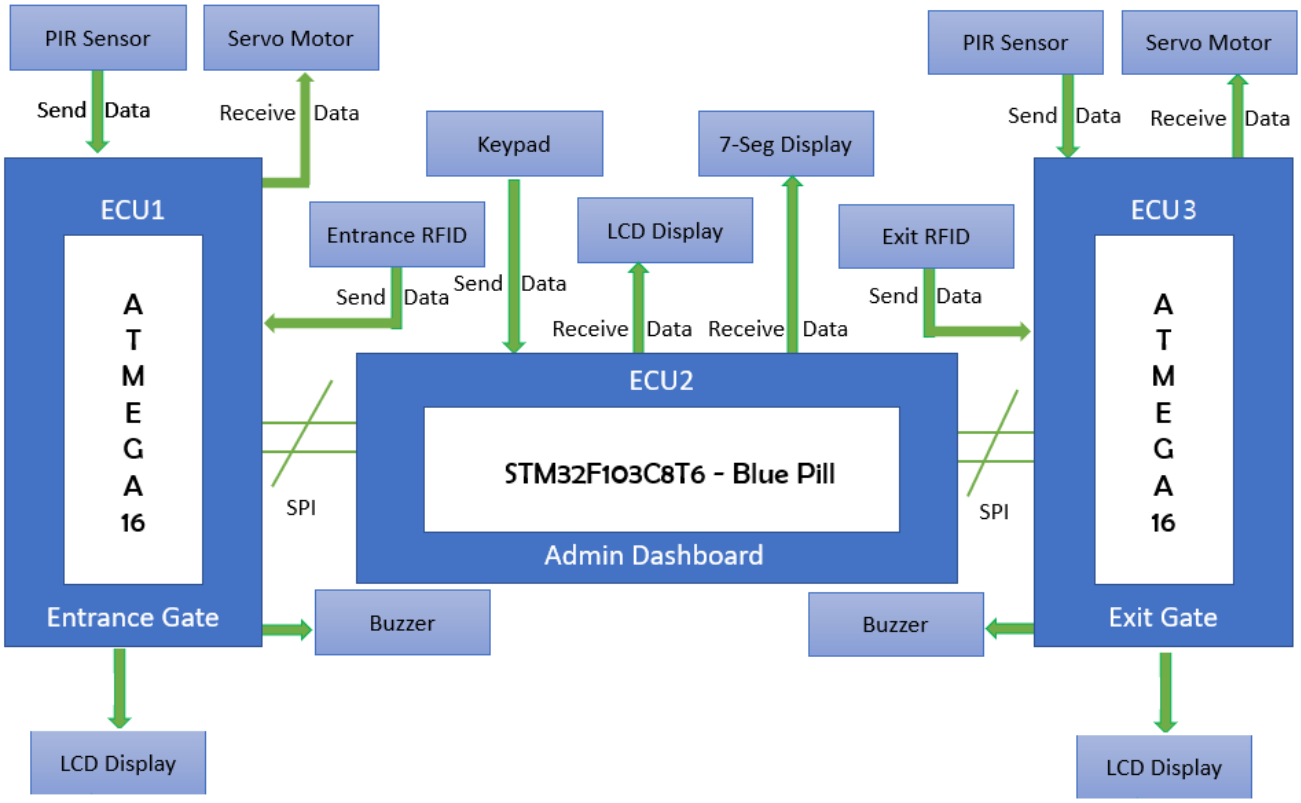


Figure 7:System Partitioning

microprocessor its specification

1- ARM 32-bit Cortex™-M3 CPU Core
   i) 72 MHz maximum frequency
   ii) Single-cycle multiplication and hardware division.
2- Memories
   i) 32 Kbytes of Flash memory
   ii) 10 Kbytes of SRAM
3- Clock, reset and supply management
   i) 2.0 to 3.6 V application supply and I/Os.
   ii) 4-to-16 MHz crystal oscillator.
   iii) 32 kHz oscillator for RTC with calibration

And used ATmega16 For ECU1 and ECU3

Program Memory Type	Flash
Program Memory Size (KB)	16
CPU Speed (MIPS/DMIPS)	16
Data EEPROM (bytes)	512

5- System Analysis

i- Use Case Diagram

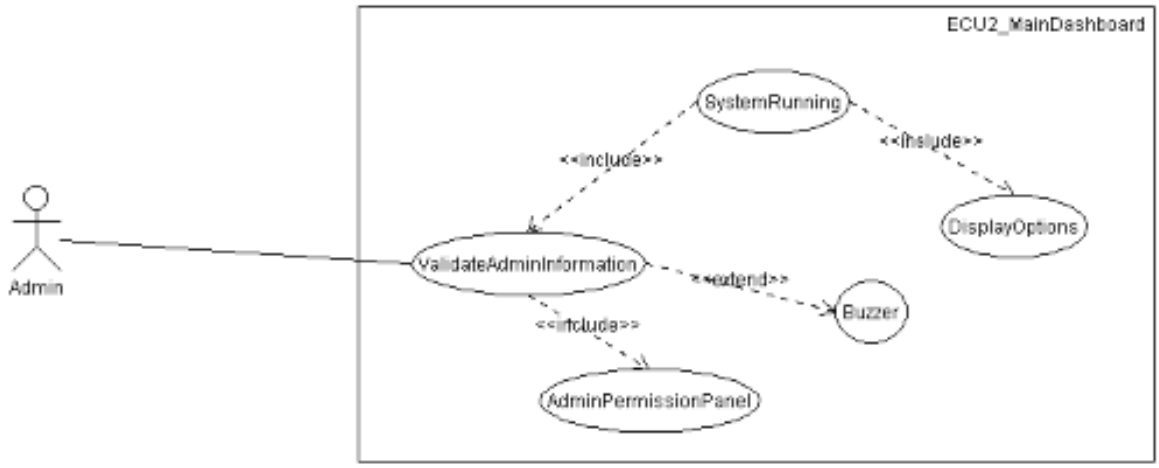


Figure 8:ECU2 Use Case Diagram

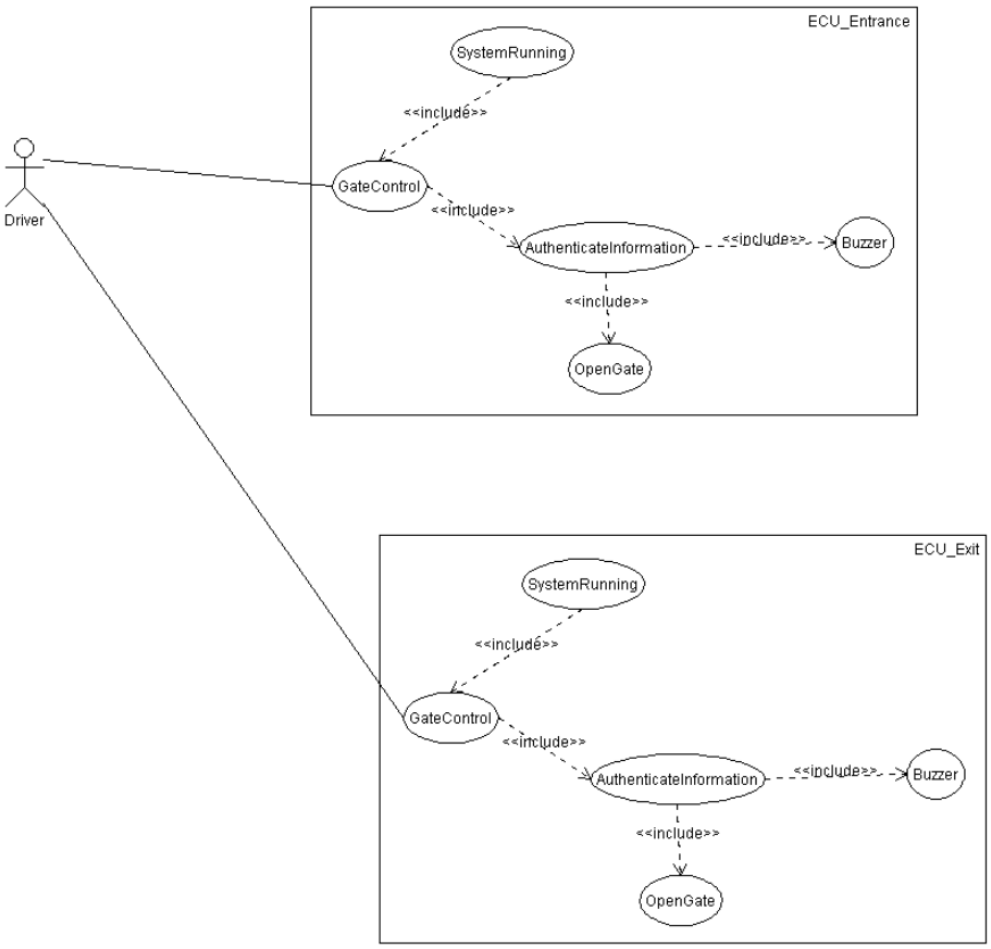


Figure 9: ECU1 & ECU3 Use Case Diagram

ii- Simple Activity Diagram

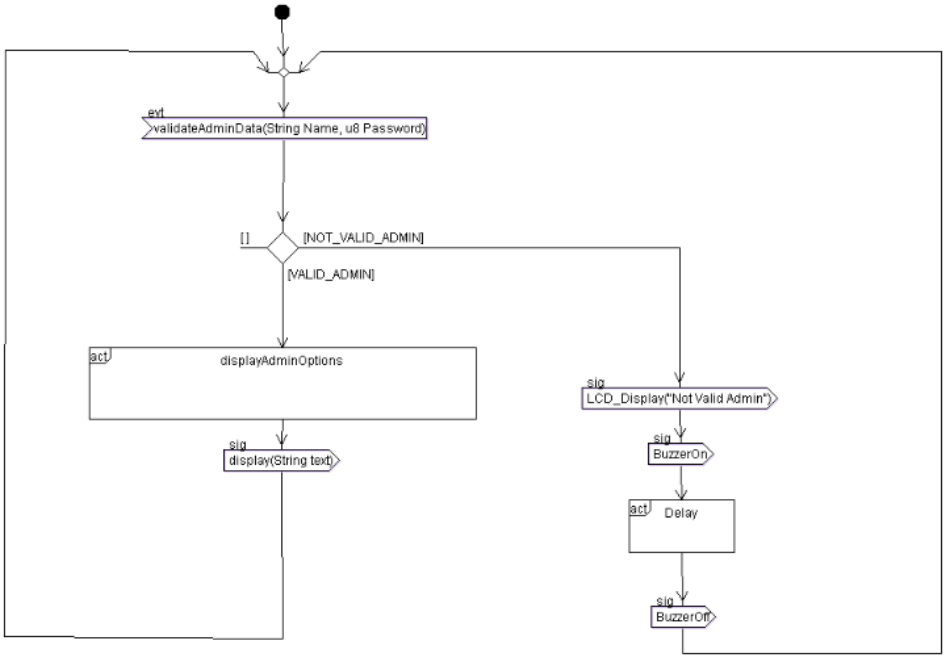


Figure 10:ECU2 Activity Diagram

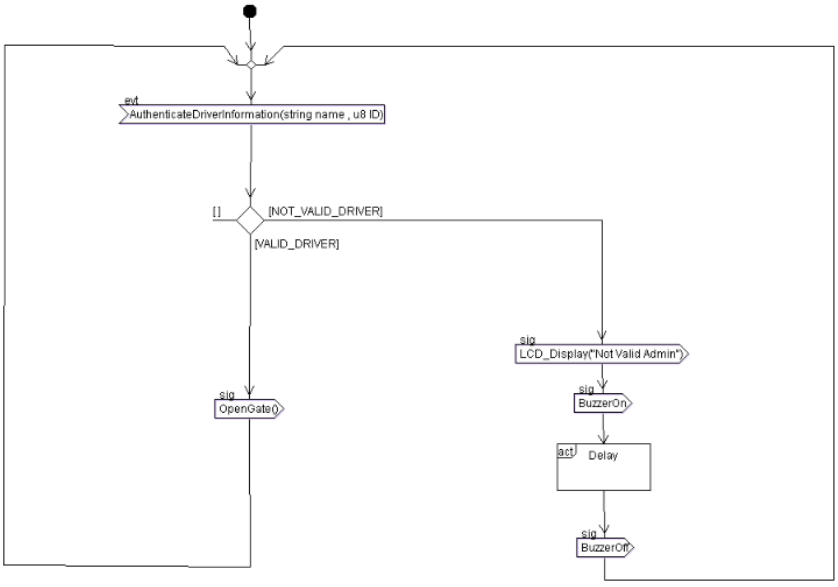


Figure 11:ECU1 Activity Diagram

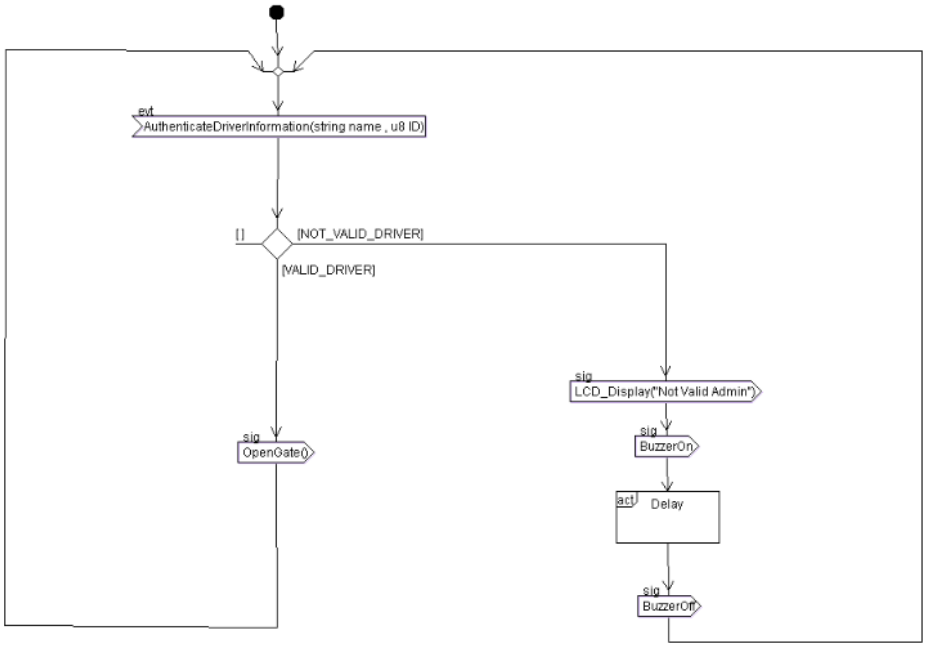


Figure 12:ECU3 Activity Diagram

iii- Sequence Diagram (UML)

- ECU1 UML

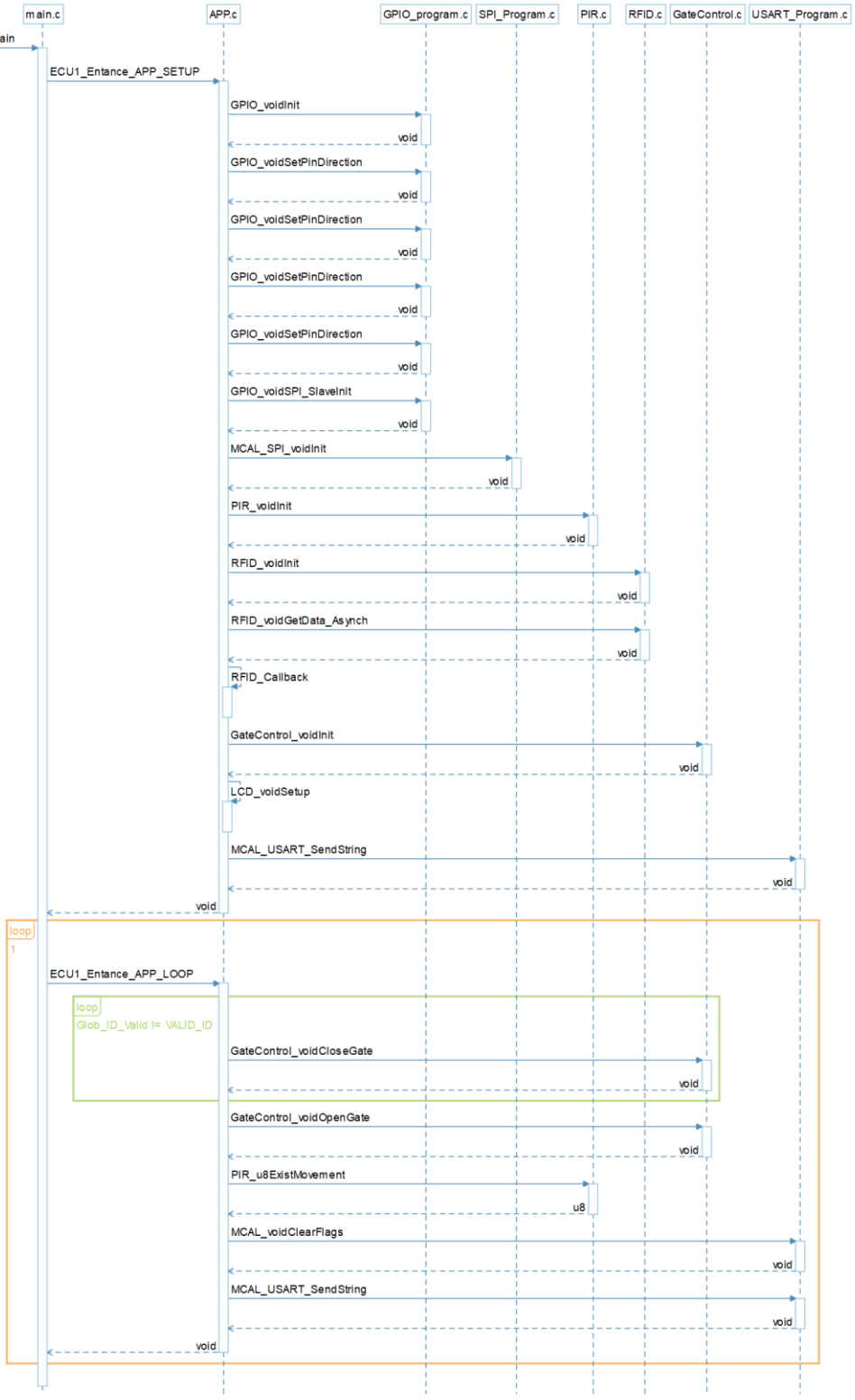


Figure 13:ECU1 UML Diagram

- ECU2 UML

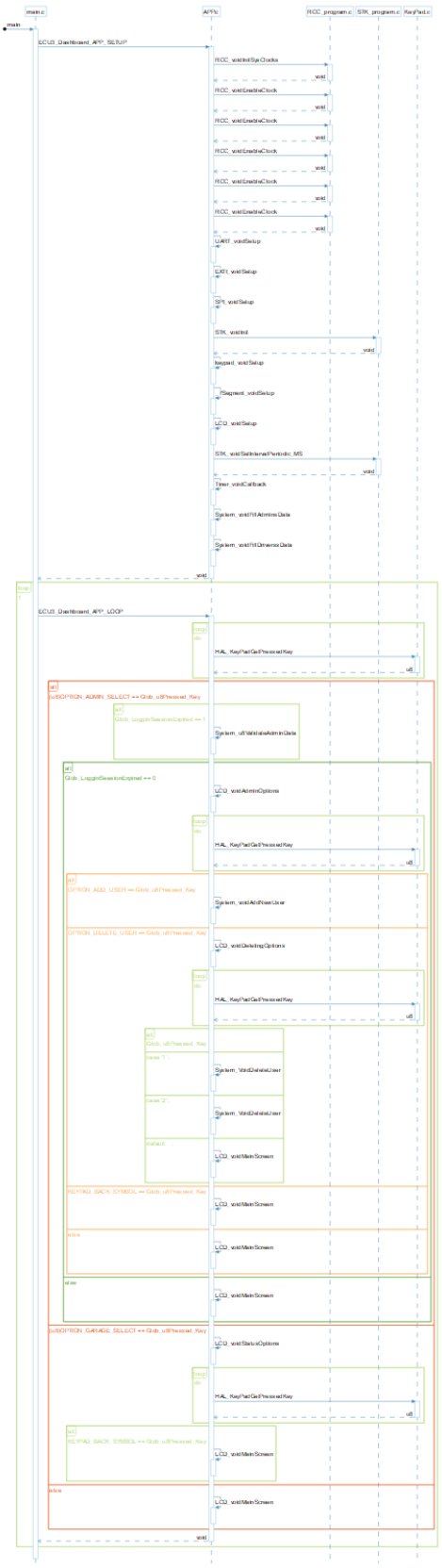


Figure 14:ECU2 UML Diagram

- ECU3 UML

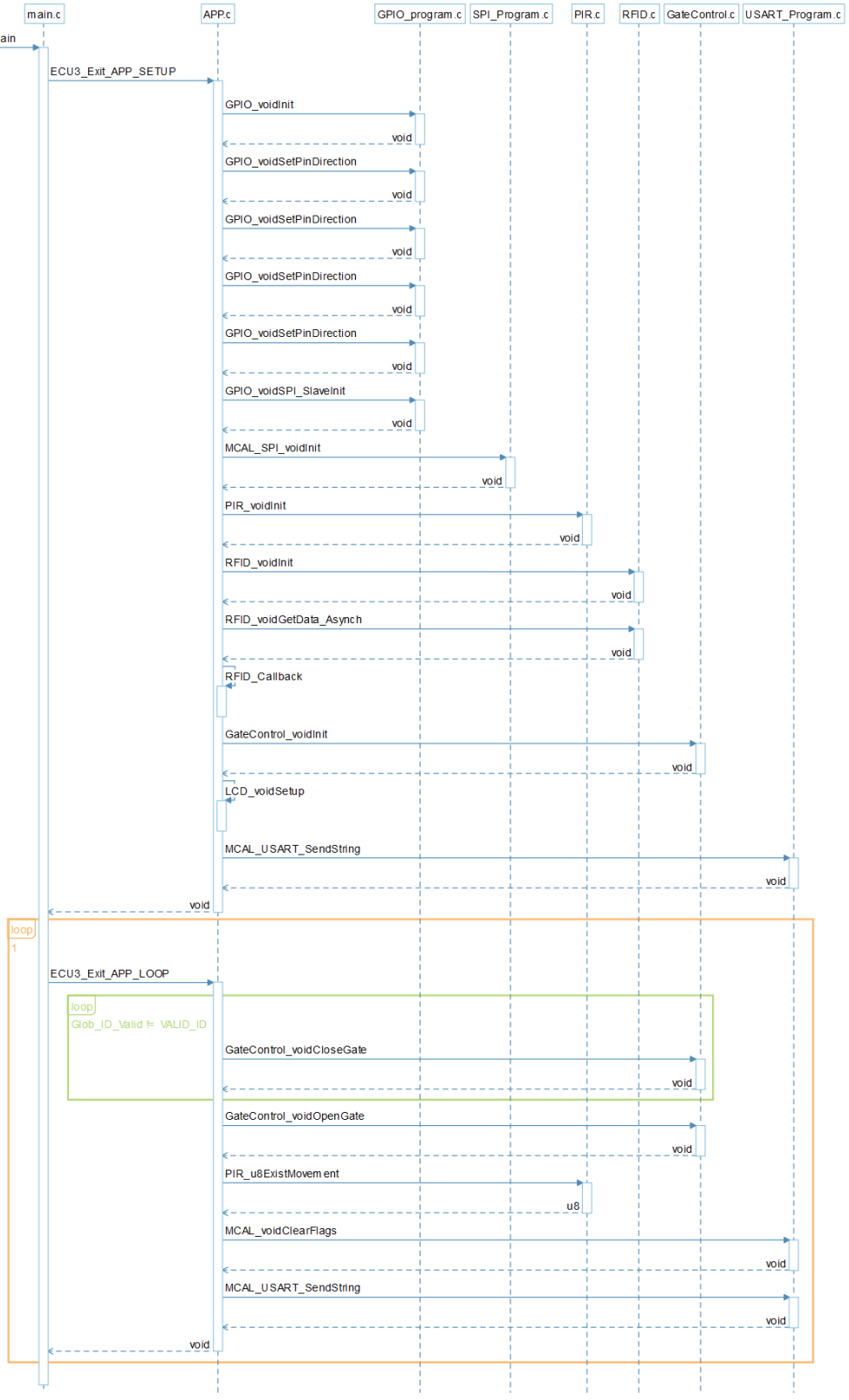


Figure 15:ECU3 UML Diagram

6- System Design

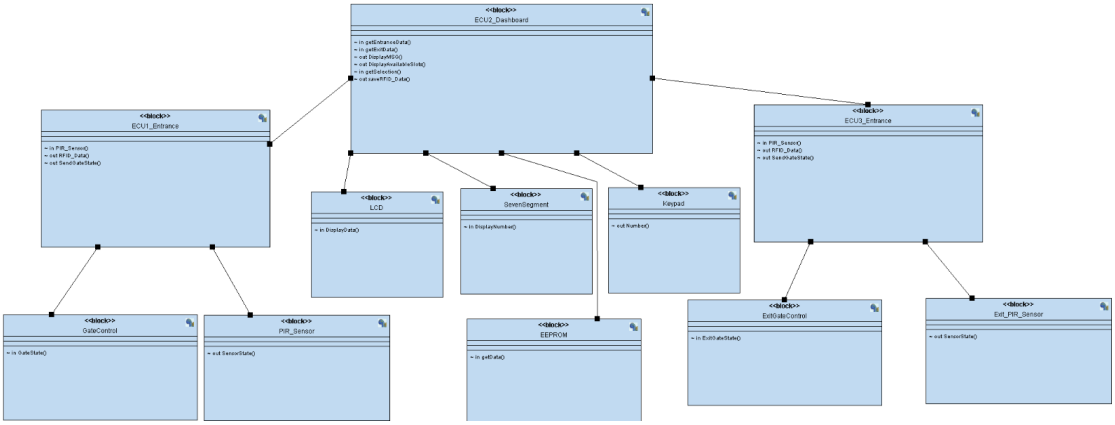
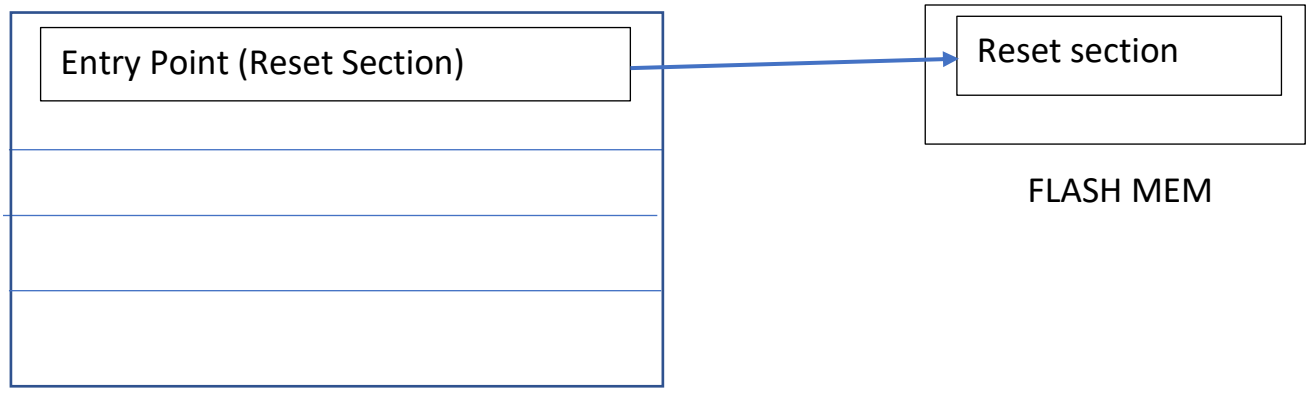


Figure 16:System Design

Boot Sequence of STM32



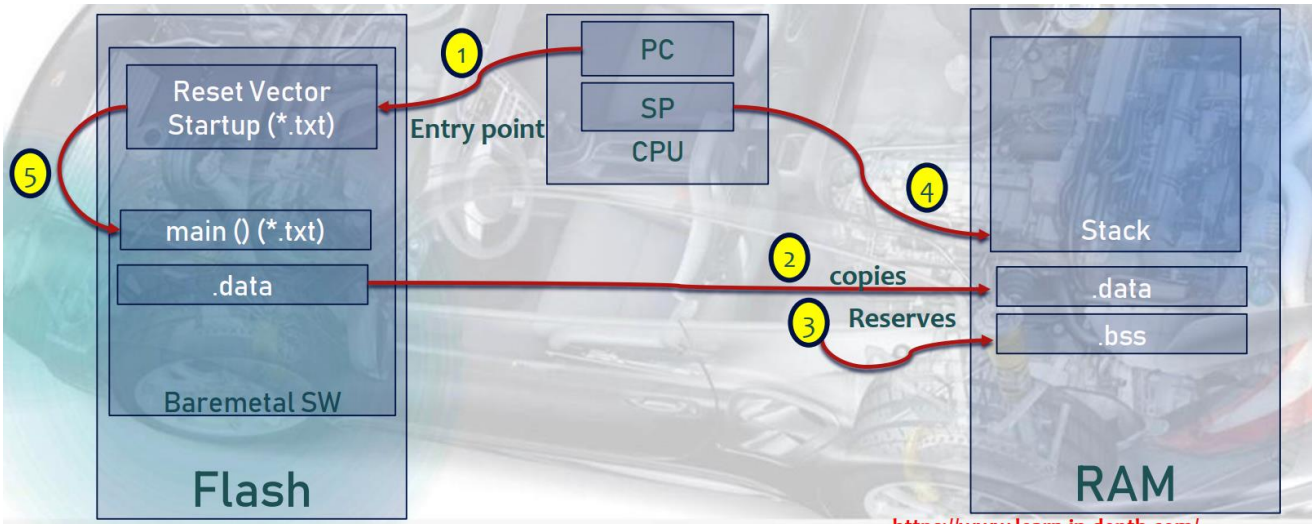


Figure 17:Sequence

Our entry point is reset handler that move .data from FLASH to SRAM and reserve .bss section in SRAM.

Map File

- Memory map

```
*(.isr_vector)
.isr_vector     0x0000000008000000      0x130 Startup/startup_stm32f103c6tx.o
                0x0000000008000000              g_pfnVectors
                0x0000000008000130              . = ALIGN (0x4)
```

Start address of flash memory

Figure 18:Vector table position in map file

Memory dump

```
Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .isr_vector   00000130  08000000  08000000  00010000  2**0
                  CONTENTS, ALLOC, LOAD, READONLY, DATA
  1 .text         000046ac  08000130  08000130  00010130  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 .rodata       0000040c  080047dc  080047dc  000147dc  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, DATA
  3 .ARM.extab    00000000  08004be8  08004be8  00020084  2**0
                  CONTENTS
  4 .ARM          00000000  08004be8  08004be8  00020084  2**0
                  CONTENTS
  5 .preinit_array 00000000  08004be8  08004be8  00020084  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  6 .init_array   00000004  08004be8  08004be8  00014be8  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  7 .fini_array   00000004  08004bec  08004bec  00014bec  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  8 .data         00000084  20000000  08004bf0  00020000  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  9 .bss          00000138  20000084  08004c74  00020084  2**2
                  ALLOC
 10 ._user_heap_stack 00000604  200001bc  08004c74  000201bc  2**0
                  ALLOC
 11 .ARM.attributes 00000029  00000000  00000000  00020084  2**0
                  CONTENTS, READONLY
 12 .debug_info   00003aa4  00000000  00000000  000200ad  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
 13 .debug_abbrev 00000f5e  00000000  00000000  00023b51  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
 14 .debug_loc    0000261c  00000000  00000000  00024aaf  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
 15 .debug_aranges 00000580  00000000  00000000  000270d0  2**3
                  CONTENTS, READONLY, DEBUGGING, OCTETS
 16 .debug_ranges 000004e0  00000000  00000000  00027650  2**3
                  CONTENTS, READONLY, DEBUGGING, OCTETS
 17 .debug_macro  00001719  00000000  00000000  00027b30  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
 18 .debug_line   000028f7  00000000  00000000  00029249  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
 19 .debug_str    000072fa  00000000  00000000  0002bb40  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
 20 .comment      0000007b  00000000  00000000  00032e3a  2**0
                  CONTENTS, READONLY
 21 .debug_frame  000014c0  00000000  00000000  00032eb8  2**2
                  CONTENTS, READONLY, DEBUGGING, OCTETS
```

Debug info

Figure 19:Memory Dump with debug section

Some of Symbols

```
080036d0 T GPIO_voidSetPinValue
08002730 T HAL_7SegmentInit
0800280c T HAL_7SegmentWriteNumber
08001ff4 T HAL_KeyPadGetPressedKey
08001f34 T HAL_KeyPadInit
080046c8 W HardFault_Handler
080046c8 W I2C1_ER_IRQHandler
080046c8 W I2C1_EV_IRQHandler
080046c8 W I2C2_ER_IRQHandler
080046c8 W I2C2_EV_IRQHandler
080046c8 t Infinite_Loop
08004744 T itoa
08001868 t keypad_voidSetup
20000040 D keys
08001778 t LCD_AddDriver
20000038 D LCD_Adding
20000030 D LCD_Deleting
2000001c D LCD_PortPin
20000028 D LCD_ProgrssBarChar
08001690 t LCD_voidAdminOptions
080024b2 T LCD_voidClear
080017b8 t LCD_voidDeleteDriver
080017f8 t LCD_voidDeletingOptions
08002448 T LCD_voidGotoXY
0800216a T LCD_voidInit
08001628 t LCD_voidMainScreen
08002236 T LCD_voidSendChar
0800253a T LCD_voidSendNumber
080023f0 T LCD_voidSendString
0800242a T LCD_voidSetCursorType
080013b8 t LCD_voidSetup
080016f8 t LCD_voidStatusOptions
080024d8 T LCD_voidStoreCustomChar
08002576 t LCD_voidWriteCmd
200000a0 B LOC_u8TimerCounter
08004690 t LoopCopyDataInit
080046a2 t LoopFillZerobss
080046ae t LoopForever
0800466a T main
080046c8 W MemManage_Handler
20000138 B myKeypad
200000a8 b myLCD
20000000 d mySegment
200000c4 b myUART
080046c8 W NMI_Handler
080038e0 T NVIC_voidEnableInterrupt
```

Figure 20:Symbols of ELF image

All symbols successfully resolved

For each symbol check [link]

ELF image details

```
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           ARM
  Version:                           0x1
  Entry point address:               0x8004679
  Start of program headers:          52 (bytes into file)
  Start of section headers:          226328 (bytes into file)
  Flags:                             0x5000200, Version5 EABI, soft-float ABI
  Size of this header:               52 (bytes)
  Size of program headers:           32 (bytes)
  Number of program headers:         3
  Size of section headers:           40 (bytes)
  Number of section headers:         26
  Section header string table index: 25
```

Figure 21:ELF image details

Hardware Simulation

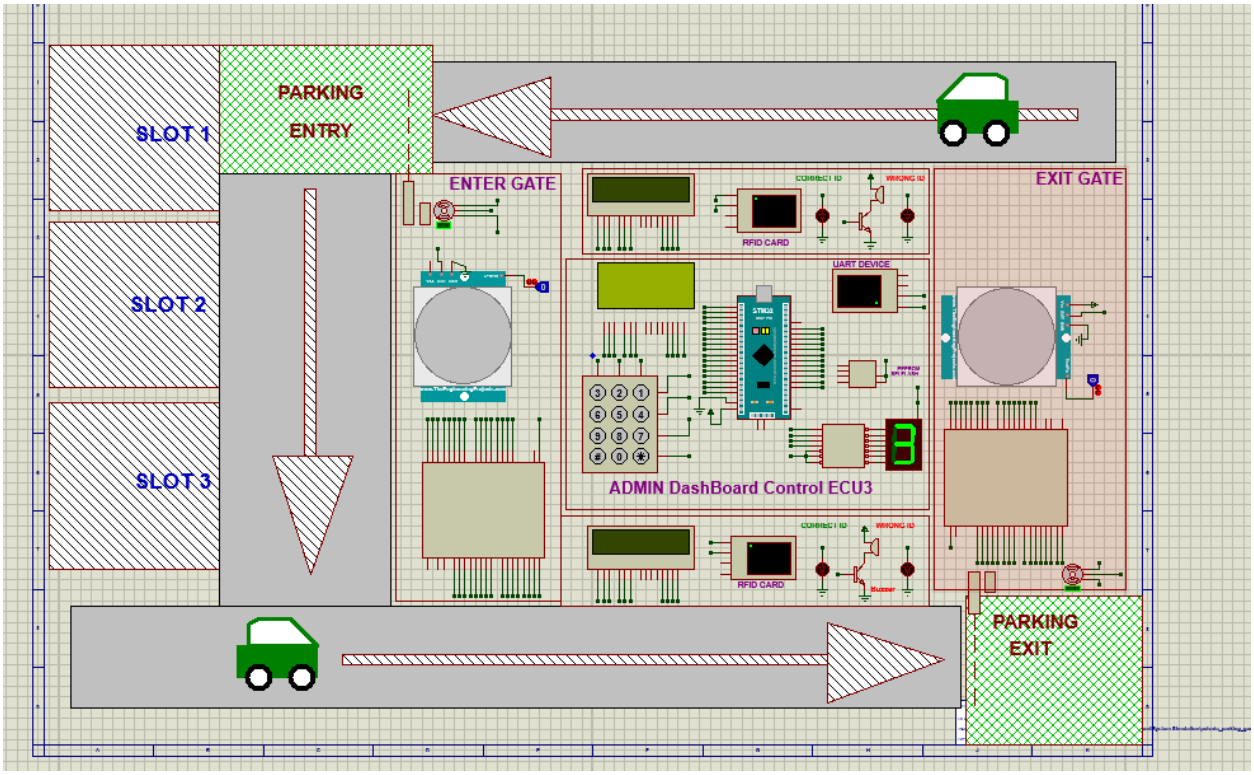


Figure 22:simulation test

Test Cases

- ECU1

Test scnario objective	Test ID	Test Title	Pre-conditions	Test Data	Expected Result	Actual Result	Status	Created By	Executed By	Test Type
	TC_RFID_01	Validate that RFID reader works well with a vaild data.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Mohamed" 2- ID= "0000001"	sys. Print "Driver Name : Mohamed Driver ID : 0000001"	Driver Name : Mohamed Driver ID : 0000001	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_RFID_02	Validate that behaviour of RFID Reader when enter username larger than expected	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Mohamed Abd El-Naby Mohamed" 2- ID= "0000001"	System will ignore any characters after the specified username length	Driver Name : Mohamedabd Driver ID : 0000001	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_RFID_03	Validate that behaviour of RFID Reader when enter ID larger than expected	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Mohamed" 2- ID= "00000001"	System will ignore any characters after the specified ID length	Driver Name : Mohamed Driver ID : 0000000	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_RFID_04	Validate that behaviour of RFID Reader when enter ID Smaller than expected	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Mohamed" 2- ID= "001"	The system will wait until the length of the ID be in a pre-specified length	Nothing	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
Validate functionality of RFID card reader (Entrnce Gate)	TC_RFID_05	Validate that behaviour of RFID Reader when enter Special Charcters in username or ID.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Moha_med" 2- ID= "00!!00001"	The system will ignore any special characters in usernames and passwords.	Driver Name : Mohamed Driver ID : 0000001	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test

Figure 23:ECU1 TEST CASES

Validate functionality of SPI Communication (Entrnce Gate)	TC_COMM_06	Validate that SPI works well with a vaild data.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- SPI Driver	Send "Hello!"	Master Debugger Write "Hello!" in form of hex in every ACK send.	"48656c6c6f21"	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_PIR_07	Validate that PIR Works when find motion.	1-Atmel Studio 2- Proteus Simulation 3- GPIO Driver 4- PIR Driver	Exist Motion	Turn On LED	LED is on.	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
Validate functionality of PIR Sensor (Entrnce Gate)	TC_PIR_08	Validate that PIR Works when there is no motion.	1-Atmel Studio 2- Proteus Simulation 3- GPIO Driver 4- PIR Driver	No Motion	Turn Off LED	LED is off.	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
Validate functionality of sending RFID data through SPI Communication (Entrnce Gate)	TC_RFID_SPI_09	Validate that the transmitted data of RFID	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 5- Enter Debug Mode	1- username = "Mohamed" 2- ID= "0000001"	sys. Print "Driver Name : Mohamed Driver ID : 0000001" Master Debugger Write The Data in form of hex in every ACK send	UART Driver Name : Mohamed Driver ID : 0000001 SPI Master "4d6f68616d65643030303 0303031"	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional test

Figure 24:ECU1 TEST CASES

	TC_Gate_10	Validate that the gate will open when ID of Driver is valid.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- PWM Driver	Enter Valid Driver 1- username = "Mohamed" 2- ID= "0000001" 3- PIR Reads Exist Motion	Gate Will Open (>= +90) and Never closed	Gate Will Open and Never Close	Pass			Functional test
	TC_Gate_11	Validate that the gate will open when ID of Driver is valid and close after that.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- PWM Driver 7- Servo Motor Driver	Enter Valid Driver 1- username = "Mohamed" 2- ID= "0000001" 3- PIR Reads no Motion	Gate Will Open (>= +90) till vechile fully entered the garage	Gate Will Open and will Close when vechile fully entered the garage	Pass			Functional test
	TC_Gate_12	Validate that the gate will not open when ID of Driver is invalid.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- PWM Driver 7- Servo Motor Driver	Enter invalid Driver 1- username = "M" 2- ID= "100000" 3- PIR Reads Exist Motion	The Gate Will Never Open	The Gate is closed	Pass			Functional test
Validate functionality of sending RFID data through SPI Communication (Entrnce Gate)	TC_Gate_13	Validate that the gate will not open when ID of Driver is invalid.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- PWM Driver 7- Servo Motor Driver	Enter invalid Driver 1- username = "M" 2- ID= "100000" 3- PIR Reads no Motion	The Gate Will Never Open	The Gate is closed	Pass			Functional test

Figure :25 ECU1 TEST CASES

Validate functionality of LCD and Buzzer (Entrnce Gate)	TC_Gate_14	Validate that the Valid ID massage	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- LCD Driver 7- Buzzer	Enter Valid Driver 1- username = "Mohamed" 2- ID= "0000001"	Your ID is Valid	Your ID is Valid	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional test
	TC_Gate_15	Validate that the invalid ID massage	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- LCD Driver 7- Buzzer	Enter invalid Driver 1- username = "Md" 2- ID= "0000001"	Invalid ID Buzzer Works 3 times	Invalid ID Buzzer Works 3 times	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional test

Figure 26:ECU1 TEST CASES

- ECU2

Test scnario objective	Test ID	Test Title	Pre-conditions	Test Data	Expected Result	Actual Result	Status	Created By	Executed By	Test Type
Validate functionality of GPIO and RCC (ECU3)	TC_RCC_1	Validate that RCC works with predefined clock	1-STM32CUBE IDE 2- KeiluVision 3- RCC Driver	1- Using PLL with HSE 2- Sys Clock = HSEx2 3- AHB clock = Sys Clock/1 4- APB1Clock = SysClock/1 5- APB2Clock = SysClock/1	1- Sys Clock = 16 MHZ 2- HCLK = 16 MHZ 3- PCLK1 = 16 MHZ 4- PCLK2 = 16 MHZ	1- Sys Clock = 16 MHZ 2- HCLK = 16 MHZ 3- PCLK1 = 16 MHZ 4- PCLK2 = 16 MHZ	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_GPIO_2	Validate that GPIO works as a output current source	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver	Connect LED with Microcontroller and ground	LED be on when MCU drive HIGH	LED turned on when MCU drive HIGH	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_GPIO_3	Validate that GPIO works as a output current sink	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver	Connect LED with Microcontroller and VCC	LED be on when MCU drive LOW	LED turned on when MCU drive LOW	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_GPIO_4	Validate that GPIO works as a input pull-up	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver	Connect LED with Microcontroller and VCC when I pressed on button LED turn ON	When Pressed LED turn on	LED turn ON	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_GPIO_5	Validate that GPIO works as a input pull-down	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver	Connect LED with Microcontroller and GND when I pressed on button LED turn OFF	When Pressed LED turn off	LED turn OFF	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
Validate functionality of Buzzer (ECU3)	TC_BUZZ_06	Validate that Buzzer Works when applay HIGH Signal	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- Buzzer Driver	Apply HIGH Signal	Buzzer turn on	Buzzer turn on	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_BUZZ_07	Validate that Buzzer stop when applay LOW Signal	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- Buzzer Driver	Apply LOW Signal	Buzzer turn off	Buzzer turn off	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test

Figure 27:ECU2 TEST CASES

Validate functionality of BCD seven Segment (ECU3)	TC_SevSeg_08	Validate that the Seven Segment works while sending numbers	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- _SevenSeqment Driver	Send Number 2	2	2	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
Validate functionality of keypad (ECU3)	TC_keypad_09	Validate that the keypad works well	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- _SevenSeqment Driver 6- Keypad	Send number 5 from keypad	MCU Read 5	MCU Read 5	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
Validate functionality of LCD (ECU3)	TC_LCD_10	Validate that the LCD can deal with strings	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver	"Hello World!"	"Hello World!"	"Hello World!"	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_LCD_11	Validate that the LCD can deal with numbers	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver	123123001	123123001	123123001	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_LCD_12	Validate that the LCD can deal with goto	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver	gotoLine(16,0) write("Ahleen") gotoLine(16,1) write("Ahleen") gotoLine(16,2) write("Ahleen") gotoLine(16,3) write("Ahleen")	Ahleen @x=16 y=0 Ahleen @x=16 y=1 Ahleen @x=16 y=2 Ahleen @x=16 y=3	Ahleen @x=16 y=0 Ahleen @x=16 y=1 Ahleen @x=16 y=2 Ahleen @x=16 y=4	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_LCD_13	Validate that the LCD can deal with special char	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver	print star ('|_|')	|_|	|_|	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_LCD_14	Validate that the LCD can deal with clear screen	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver	clearScreen()	noOutput	noOutput	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test

Figure 28:ECU2 TEST CASES

Validate functionality of SPI Communication (ECU3)	TC_SPI_15	Validate that the can exchange data by SPI	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- SPI	Exchange '+' and 'A' between master and slave	'a' at master '+' at slave	'a' at master '+' at slave	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test	
Validate functionality of EEPROM (ECU3)	TC_EEPROM_16	Validate that the can store data on eeprom	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver	store A @ 0x00FF	store A @ 0x00FF		Fail	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test	
	TC_EEPROM_17	Validate that the can get data from eeprom	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- SPI	Get data @ 0x00FF	Get data @ 0x00FF		Fail	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test	

Figure 29:ECU2 TEST CASES

Validate functionality of Admin (ECU3)	TC_ADMIN_18	Validate that Admin Can login	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver 6- Keypad Driver	username: "Mohamed" Pasword: "0000001"	Display admin privilege screen	Display admin privilege screen	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional Test
	TC_ADMIN_19	Validate that Admin Can't login if entered wrong username	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver 6- Keypad Driver	username: "Mohmed" Pasword: "0000001"	Display Wrong username	Display Wrong username	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional Test
	TC_ADMIN_20	Validate that Admin Can't login if entered wrong Password	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver 6- Keypad Driver	username: "Mohamed" Pasword: "000001"	Display Wrong password	Display Wrong password	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional Test
	TC_ADMIN_21	Validate that Admin Can Add New Driver	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver 6- Keypad Driver	Admin Data username: "Mohamed" Pasword: "000001" Driver Data "MoSalah" "1234567"	Display Adding screen	Display Adding screen	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional Test
	TC_ADMIN_22	Validate that Admin Can Delete Driver	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver 6- Keypad Driver	Admin Data username: "Mohamed" Pasword: "000001" No Exist Driver	Display Unsuccessful	Display Unsuccessful	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional Test
	TC_ADMIN_23	Validate that Admin Can Delete Driver	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver 6- Keypad Driver	Admin Data username: "Mohamed" Driver Data "MoSalh" "1234567"	Display Unsuccessful	Display Unsuccessful	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional Test
	TC_ADMIN_24	Validate that Admin Can Delete Driver	1-STM32CUBE IDE 2- Proteus Simulation 3- RCC Driver 4- GPIO Driver 5- LCD Driver 6- Keypad Driver	Admin Data username: "Mohamed" Driver Data "MoSalah" "1234567"	Display Successful	Display Successful	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional Test

Figure 30:ECU2 TEST CASES

- ECU1

Test scnario objective	Test ID	Test Title	Pre-conditions	Test Data	Expected Result	Actual Result	Status	Created By	Executed By	Test Type
	TC_RFID_01	Validate that RFID reader works well with a vaild data.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Mohamed" 2- ID= "0000001"	sys. Print "Driver Name : Mohamed Driver ID : 0000001"	Driver Name : Mohamed Driver ID : 0000001	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_RFID_02	Validate that behaviour of RFID Reader when enter username larger than expected	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Mohamed Abd El-Naby Mohamed" 2- ID= "0000001"	System will ignore any characters after the specified username length	Driver Name : Mohamedabd Driver ID : 0000001	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_RFID_03	Validate that behaviour of RFID Reader when enter ID larger than expected	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Mohamed" 2- ID= "00000001"	System will ignore any characters after the specified ID length	Driver Name : Mohamed Driver ID : 0000000	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_RFID_04	Validate that behaviour of RFID Reader when enter ID Smaller than expected	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Mohamed" 2- ID= "001"	The system will wait until the length of the ID be in a pre-specified length	Nothing	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
Validate functionality of RFID card reader (Entrnce Gate)	TC_RFID_05	Validate that behaviour of RFID Reader when enter Special Charcters in username or ID.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- Enter Debug Mode	1- username = "Moha_med" 2- ID= "00!!00001"	The system will ignore any special characters in usernames and passwords.	Driver Name : Mohamed Driver ID : 0000001	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test

Figure 31:ECU3 TEST CASES

Validate functionality of SPI Communication (Entrnce Gate)	TC_COMM_06	Validate that SPI works well with a vaild data.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- SPI Driver	Send "Hello!"	Master Debugger Write "Hello!" in form of hex in every ACK send.	"48656c6c6f21"	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
	TC_PIR_07	Validate that PIR Works when find motion.	1-Atmel Studio 2- Proteus Simulation 3- GPIO Driver 4- PIR Driver	Exist Motion	Turn On LED	LED is on.	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
Validate functionality of PIR Sensor (Entrnce Gate)	TC_PIR_08	Validate that PIR Works when there is no motion.	1-Atmel Studio 2- Proteus Simulation 3- GPIO Driver 4- PIR Driver	No Motion	Turn Off LED	LED is off.	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Unit test
Validate functionality of sending RFID data through SPI Communication (Entrnce Gate)	TC_RFID_SPI_09	Validate that the transmitted data of RFID	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 5- Enter Debug Mode	1- username = "Mohamed" 2- ID= "0000001"	sys. Print "Driver Name : Mohamed Driver ID : 0000001" Master Debugger Write The Data in form of hex in every ACK send	UART Driver Name : Mohamed Driver ID : 0000001 SPI Master "4d6f68616d65643030303030303031"	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional test

Figure 32:ECU3 TEST CASES

	TC_Gate_10	Validate that the gate will open when ID of Driver is valid.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- PWM Driver	Enter Valid Driver 1- username = "Mohamed" 2- ID= "0000001" 3- PIR Reads Exist Motion	Gate Will Open (>= +90) and Never closed	Gate Will Open and Never Close	Pass			Functional test
	TC_Gate_11	Validate that the gate will open when ID of Driver is valid and close after that.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- PWM Driver 7- Servo Motor Driver	Enter Valid Driver 1- username = "Mohamed" 2- ID= "0000001" 3- PIR Reads no Motion	Gate Will Open (>= +90) till vechile fully entered the garage	Gate Will Open and will Close when vechile fully entered the garage	Pass			Functional test
	TC_Gate_12	Validate that the gate will not open when ID of Driver is invalid.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- PWM Driver 7- Servo Motor Driver	Enter invalid Driver 1- username = "M" 2- ID= "100000" 3- PIR Reads Exist Motion	The Gate Will Never Open	The Gate is closed	Pass			Functional test
Validate functionality of sending RFID data through SPI Communication (Entrnce Gate)	TC_Gate_13	Validate that the gate will not open when ID of Driver is invalid.	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- PWM Driver 7- Servo Motor Driver	Enter invalid Driver 1- username = "M" 2- ID= "100000" 3- PIR Reads no Motion	The Gate Will Never Open	The Gate is closed	Pass			Functional test

:ECU3 TEST CASES33 Figure

Validate functionality of LCD and Buzzer (Entrnce Gate)	TC_Gate_14	Validate that the Valid ID massage	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- LCD Driver 7- Buzzer	Enter Valid Driver 1- username = "Mohamed" 2- ID= "0000001"	Your ID is Valid	Your ID is Valid	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional test
	TC_Gate_15	Validate that the invalid ID massage	1-Atmel Studio 2- Proteus Simulation 3- RFID Driver 4- UART Driver 5- SPI Driver 6- LCD Driver 7- Buzzer	Enter invalid Driver 1- username = "Md" 2- ID= "0000001"	Invalid ID Buzzer Works 3 times	Invalid ID Buzzer Works 3 times	Pass	Mohamed Abd El-Naby	Mohamed Abd El-Naby	Functional test

Figure 34:ECU3 TEST CASES