GDD

for

Digital Elevator PO4\_DGELV

Version 1.2 proposed

Prepared by /   
- Ahmed Refaat

- Donia Mohamed

- Marcelle Samir

Jan 28, 2020

Document Status

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Document Status** | **Author** | **Date** |
| V\_1.2 | Proposed | Donia Mohamed  Ahmed Refaat  Marcelle Samir | Feb 28, 2020 |

Revision History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Version** | **Author** | **Date** | **Change Description** | **Document Status** |
| GDD\_DIGITAL\_  ELEVATOR | V\_1.0 | Donia Mohamed | Feb 26, 2020 | Initial creation of the GDD Document by adding  1)Naming convention  2)Document status  3)Revision history  4)table of signals  5)context diagram | Draft |
|  |  |  |  |  |  |
| GDD\_DIGITAL\_  ELEVATOR | V\_1.0 | Ahmed Refaat | Feb 26, 2020 | Adding the static architecture ( layered architecture) | Draft |
| GDD\_DIGITAL\_  ELEVATOR | V\_1.1 | Marcelle Samir | Feb 27, 2020 | Adding the static architecture ( components APIs) | Proposed |
| GDD\_DIGITAL\_  ELEVATOR | V\_1.2 | Donia  Mohamed | Feb 28,2020 | Adding some changes  1)changing the version of GDD  2)changing the Table of signals  3)changing the naming of context diagram  4)editing the layered architecture  - removing PORT\_driver and OS  5)modifying APIs according to the review session | Proposed |

Reference Table

|  |  |  |
| --- | --- | --- |
| **Ref. Document** | **Version** | **Document Status** |
| SRS\_DIGITAL\_ELEVATOR | V\_1.4 | Proposed |
| HSI\_DIGITAL\_ELEVATOR | V\_1.5 | Released |
|  |  |  |

Contents

[Revision History 3](#_Toc33967988)

[Reference Table 4](#_Toc33967989)

[1. Introduction 6](#_Toc33967990)

[1.1 Purpose 6](#_Toc33967991)

[1.2 Project Scope 6](#_Toc33967992)

[Table of Signals 1](#_Toc33967993)

[3. Software Context Diagram 2](#_Toc33967994)

[4. Static Architecture 3](#_Toc33967995)

[- Layered architecture 3](#_Toc33967996)

[- Component APIs 4](#_Toc33967997)

[**1)** **DIO\_Driver** 4](#_Toc33967998)

[**2)** **Switch\_Driver** 7](#_Toc33967999)

[**3)** **LCD\_Driver** 8](#_Toc33968000)

[**4)** **Keypad\_Driver** 1](#_Toc33968001)

[**5)** **Buzzer\_Driver** 1](#_Toc33968002)

[**6)** **Motor\_Driver** 1](#_Toc33968003)

[**7)** **Elevator\_Control** 1](#_Toc33968004)

[**8)** **User\_Registration\_and\_verification** 1](#_Toc33968005)

# 1. Introduction

## 1.1 Purpose

This project aims at developing a Digital Elevator with lock system to be more secure and have specific functionalities.

The purpose of this document is to present a detailed description of the Digital Elevator System. It will explain the purpose, scope and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate.

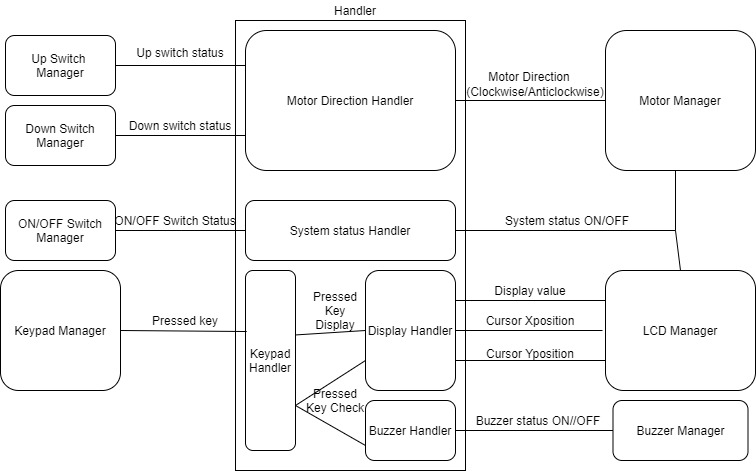
## 1.2 Project Scope

This software system will be an Embedded System for a digital elevator. This system will be designed to secure the usage of the elevator and handle the movement of it. By having a limited number of resigned users with unique ID and entered password, we can secure the usage of the elevator. Using some developed buttons, the users can easily control the movement of the elevator.

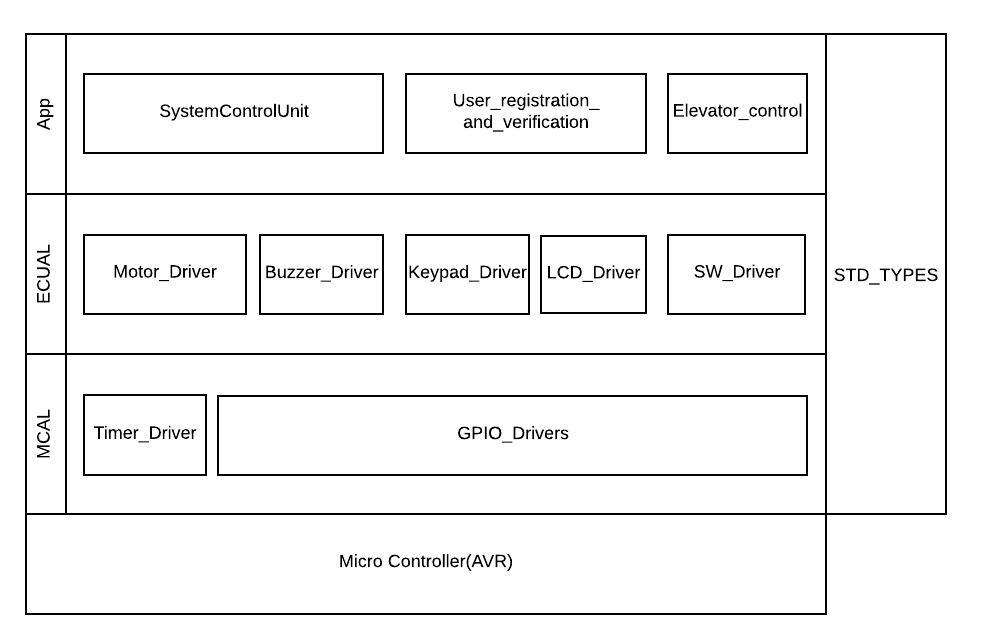
# Table of Signals

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Signal | Type | Initial Value | Range of Values | Input/Output | Unit |
| Up\_switch\_status | Non-Constant | 0 | [0 /1] | Input | Volt |
| Down\_switch\_status | Non-Constant | 0 | [0 /1] | Input | Volt |
| On/Off\_switch\_status | Non-Constant | 0 | [0 /1] | Input | Volt |
| Pressed Key | Non-Constant | 0 | [0 /1] | Input | Volt |
| Motor\_direction | Non-Constant | 0 | [0/1] | Input | Volt |
| System\_status | Non-Constant | 0 | [0/1] | Input | Volt |
| Display\_value | Non-Constant | 0 | [0-9] | Output | number |
| Curser\_XPosition | Non-Constant | 0 | [0-15] | Input | number |
| Curser\_yPosition | Non-Constant | 0 | [0/1] | Input | number |
| Buzzer\_State | Non-Constant | 0 | [0/1] | Input | volt |

# Software Context Diagram



# Static Architecture

* Layered architecture

## Component APIs

1. **DIO\_Driver**

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_001\_V1.0 |
| Component Name | DIO\_Driver |
| API Name | Error\_Status DIO\_SetPinValue(u32 Copy\_u32Port, u32 Copy\_u32Pin, u32 Copy\_u32Value); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u32 Copy\_u32Port   |  |  | | --- | --- | | PORTA | \*((u8\*)0x3B) | | PORTB | \*((u8\*)0x38) | | PORTC | \*((u8\*)0x35) | | PORTD | \*((u8\*)0x32) |   u32 Copy\_u32Pin   |  |  | | --- | --- | | PIN0 | 0 | | PIN1 | 1 | | PIN2 | 2 | | PIN3 | 3 | | PIN4 | 4 | | PIN5 | 5 | | PIN6 | 6 | | PIN7 | 7 |   u32 Copy\_u32Mode   |  |  | | --- | --- | | LOW | 0 | | HIGH | 1 | | PULLUP | 1 | |
| Description | The functionality of this API to select its value if it’s HIGH or LOW or PULLUP (in case of the pin is INPUT) of a PIN by sending the pin number and port number as arguments |

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_002\_V1.0 |
| Component Name | DIO\_Driver |
| API Name | Error\_Status DIO\_SetPinDIR(u32 Copy\_u32Port, u32 Copy\_u32Pin, u32 Copy\_u32Mode); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u32 Copy\_u32Port   |  |  | | --- | --- | | PORTA | \*((u8\*)0x3B) | | PORTB | \*((u8\*)0x38) | | PORTC | \*((u8\*)0x35) | | PORTD | \*((u8\*)0x32) |   u32 Copy\_u32Pin   |  |  | | --- | --- | | PIN0 | 0 | | PIN1 | 1 | | PIN2 | 2 | | PIN3 | 3 | | PIN4 | 4 | | PIN5 | 5 | | PIN6 | 6 | | PIN7 | 7 |   u32 Copy\_u32Mode   |  |  | | --- | --- | | INPUT\_PIN | 0 | | OUTPUT\_PIN | 1 | |
| Description | The functionality of this API to select the Mode if it’s INPUT or OUTPUT of a PIN by sending the pin number and port number as arguments |

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_003\_V1.0 |
| Component Name | DIO\_Driver |
| API Name | Error\_Status DIO\_GetPinValue(u32 Copy\_u32Port, u32 Copy\_u32Pin, u32 \*Copy\_u32Value); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u32 Copy\_u32Port   |  |  | | --- | --- | | PORTA | \*((u8\*)0x3B) | | PORTB | \*((u8\*)0x38) | | PORTC | \*((u8\*)0x35) | | PORTD | \*((u8\*)0x32) |   u32 Copy\_u32Pin   |  |  | | --- | --- | | PIN0 | 0 | | PIN1 | 1 | | PIN2 | 2 | | PIN3 | 3 | | PIN4 | 4 | | PIN5 | 5 | | PIN6 | 6 | | PIN7 | 7 |   u32 \*Copy\_u32Mode  it’s a pointer where the read value will be written in |
| Description | The functionality of this API to read the of a PIN by sending the pin number and port number as arguments and receiving the read value in a pointer. |

1. **Switch\_Driver**

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_004\_V1.1 |
| Component Name | Switch\_Driver |
| API Name | Error\_Status GetSwitchState(u8 Copy\_u8SwitchNum ,u8 \*Copy\_u8SwitchValue); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 Copy\_u8SwitchNum   |  |  | | --- | --- | | UP\_SWITCH | 0 | | DOWN\_SWITCH | 1 | | RESET\_SWITCH | 2 |   u8 \*Copy\_u8SwitchValue  it’s a pointer where the read value will be written in   |  |  | | --- | --- | | SW\_PRESSED | 1 | | SW\_RELEASED | 0 | |
| Description | The functionality of this API to read the switch value by receiving the read value in a pointer. |

1. **LCD\_Driver**

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_005\_V1.0 |
| Component Name | LCD\_Driver |
| API Name | Error\_Status CLCD\_voidInitialize(void); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | It doesn’t take any parameters |
| Description | The functionality of this API to make the hardware ready by applying standard sequence on it. |

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_006\_V1.1 |
| Component Name | LCD\_Driver |
| API Name | Error\_Status CLCD\_voidGoToXVPos(u8 Copy\_u8XPos, u8 Copy\_u8YPos); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 Copy\_u8XPos  this parameter takes the number of x position (range between 0 to 15)  u8 Copy\_u8YPos  this parameter takes the number of y position, it has 2 options  (CLCD\_U8\_LINE\_TWO or CLCD\_U8\_LINE\_ONE) |
| Description | The functionality of this API to move the cursor to the desired position |

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_007\_V1.1 |
| Component Name | LCD\_Driver |
| API Name | Error\_Status CLCD\_voidWriteString(const char\* Copy\_pchString); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | const char\* Copy\_pchString  this parameter takes the text you want to display on the LCD |
| Description | The functionality of this API to print a text on the LCD |

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_008\_V1.1 |
| Component Name | Keypad\_Driver |
| API Name | Error\_Status KEYPAD\_GetPressedKey( u8 \*Copy\_u8KeyValue); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 \*Copy\_u8KeyValue  u8 \*Copy\_u8SwitchValue  it’s a pointer where the read value will be written in   |  |  | | --- | --- | | KEY\_PRESSED | 1 | | KEY\_RELEASED | 0 | |
| Description | The functionality of this API to print a text on the LCD |

1. **Keypad\_Driver**
2. **Buzzer\_Driver**

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_009\_V1.1 |
| Component Name | Buzzer\_Driver |
| API Name | Error\_Status SetBuzzerOn(u8 Copy\_u8BuzzerNum); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 Copy\_u8BuzzerNum   |  |  | | --- | --- | | Buzzer\_ONE | 0 | |
| Description | The functionality of this API to set the buzzer ON |

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_010\_V1.1 |
| Component Name | Buzzer\_Driver |
| API Name | Error\_Status SetBuzzerOff(u8 Copy\_u8BuzzerNum); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 Copy\_u8BuzzerNum   |  |  | | --- | --- | | Buzzer\_ONE | 0 | |
| Description | The functionality of this API to set the buzzer OFF |

1. **Motor\_Driver**

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_011\_V1.1 |
| Component Name | Motor\_Driver |
| API Name | Error\_Status SetMotorOff(u8 Copy\_u8MotorNum); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 Copy\_u8MotorNum   |  |  | | --- | --- | | Motor\_ONE | 0 | |
| Description | The functionality of this API to set the Motor OFF |

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_012\_V1.1 |
| Component Name | Motor\_Driver |
| API Name | Error\_Status SetMotorUp(u8 Copy\_u8MotorNum); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 Copy\_u8MotorNum   |  |  | | --- | --- | | Motor\_ONE | 0 | |
| Description | The functionality of this API to set the Motor Up |

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_013\_V1.1 |
| Component Name | Motor\_Driver |
| API Name | Error\_Status SetMotorDown(u8 Copy\_u8MotorNum); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 Copy\_u8MotorNum   |  |  | | --- | --- | | Motor\_ONE | 0 | |
| Description | The functionality of this API to set the Motor Down |

1. **Elevator\_Control**

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_014\_V1.1 |
| Component Name | Elevator\_Control |
| API Name | Error\_Status ElevatorMotor\_Init(u8 Copy\_u8MotorNum); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 Copy\_u8MotorNum   |  |  | | --- | --- | | Motor\_ONE | 0 | |
| Description | The functionality of this API to call the Motor driver to initialize the predefined Motor pins. |

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_015\_V1.1 |
| Component Name | Elevator\_Control |
| API Name | Error\_Status ElevatorMotor\_Control(u8 \*Copy\_u8UpSwitchStatus, u8 \*Copy\_u8DownSwitchStatus, u8 \*Copy\_u8StopSwitchStatus) |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | u8 \*Copy\_u8UpSwitchStatus  u8 \*Copy\_u8DownSwitchStatus  u8 \*Copy\_u8StopSwitchStatus  there are pointers carries the switches values   |  |  | | --- | --- | | SWITCH\_PRESSED | 1 | | SWITCH\_RELEASED | 0 | |
| Description | The functionality of this API to call the Switch driver to read each switch (UP\_SWITCH and DOWN\_SWITCH and RESET\_SWITCH) status if it’s Pressed or Released |

1. **User\_Registration\_and\_verification**

|  |  |
| --- | --- |
| Req. ID | Req\_DIGELV\_GDD\_016\_V1.1 |
| Component Name | User\_Registration\_and\_verification |
| API Name | Error\_Status Keypad\_LCD\_Init(void); |
| Return type | It’s u8 Error\_Status, it returns E\_OK or E\_NO   |  |  | | --- | --- | | E\_OK | 0 | | E\_NOK | 1 | |
| Parameters type | It takes non parameters |
| Description | The functionality of this API to call the LCD driver and Keypad driver to initialize the predefined pins. |