

**PO3\_DGW\_Digital watch**

**(GDD)**

Table 1 Status Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document Name** | **Author** | **Version** | **Update Date** | **Status** |
| GDD | Mostafa Nader | 1.4 | 12/03/2020 | Released |

**DOCUMENT HISTORY**

Table 2 Document History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Description of Change** | **Author** | **Date** | **Document Status** |
| 1 | Initial creation of GDD Document | Norhan Nassar | 27/01/2020 | Draft |
| 1.1 | Adding Software input output,  Updating Software feature diagram and updating static layered architecture | Norhan Nassar | 1/03/2020 | Proposed |
| 1.2 | Adding Requirement at LCD Component  Req\_PO3\_DGW\_GDD\_11\_V01.0 and shifting other Requirements ids | Norhan Nassar | 4/03/2020 | Proposed |
| 1.3 | Updating Feature Diagram, Return type of all Requirements, Deleting some Requirements at LCD Component | Norhan Nassar | 12/3/2020 | Proposed |
| 1.4 | Updated the Descriptions of the return parameters in all requirements  Added Table of contents/List of Figures | Mostafa Nader | 14/3/2020 | Released |

Table 3 Reference Table

|  |  |  |
| --- | --- | --- |
| **Reference Input Documents** | **Version** | **Status** |
| **SRS** | 1.7 | Proposed |

Table of Contents

[Table of Contents 3](#_Toc35078917)

[1 Project Description 4](#_Toc35078918)

[2 Software Context Diagram 4](#_Toc35078919)

[3 Input Output Signals 5](#_Toc35078920)

[4 Software Feature: 7](#_Toc35078921)

[5 Static Architecture: 7](#_Toc35078922)

[6 Requirements 8](#_Toc35078923)

[6.1 MCAL Layer: 8](#_Toc35078924)

[6.1.1 DIO Component: 8](#_Toc35078925)

[6.1.2 Port Component: 9](#_Toc35078926)

[6.1.3 Timer Component: 10](#_Toc35078927)

[6.2 HAL Layer: 11](#_Toc35078928)

[6.2.1 Switch Component: 11](#_Toc35078929)

[6.2.2 Buzzer Component: 12](#_Toc35078930)

[6.3 APP Layer 14](#_Toc35078931)

[6.3.1 Time-Display Component 14](#_Toc35078932)

[6.3.2 Alarm mode requirements 16](#_Toc35078933)

[6.3.3 Stop-Watch Component 17](#_Toc35078934)

[6.3.4 DGW Requirements 19](#_Toc35078935)

List of Figures

[Figure 1 Software Context Diagram 4](file:///C:\Users\MOSTAFA\Downloads\GDD%20(1).odt#_Toc35079056)

[Figure 2 Software Input/output Diagram 5](file:///C:\Users\MOSTAFA\Downloads\GDD%20(1).odt#_Toc35079057)

[Figure 3 Software Features Diagram 7](#_Toc35079058)

[Figure 4 Static Layered Architecture 7](file:///C:\Users\MOSTAFA\Downloads\GDD%20(1).odt#_Toc35079059)

# Project Description

Digital Watch System that has three different modes.

1. Time Display Mode that displays the time as a normal clock and increments its time every 1 second.
2. Alarm Mode that it turns a buzzer on when time is on.
3. Stopwatch Mode.

The user can select an option of these three modes.

# Software Context Diagram

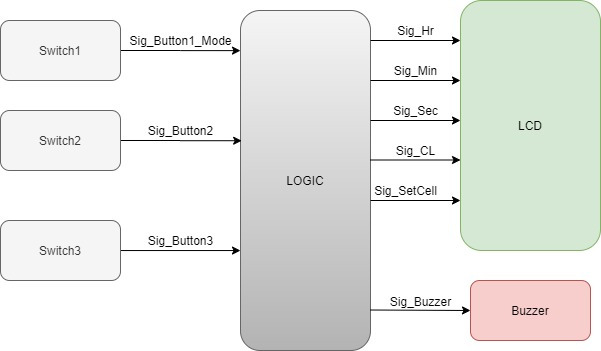


Figure 1 Software Context Diagram

# Input Output Signals

**3.1 Input Output Diagram**

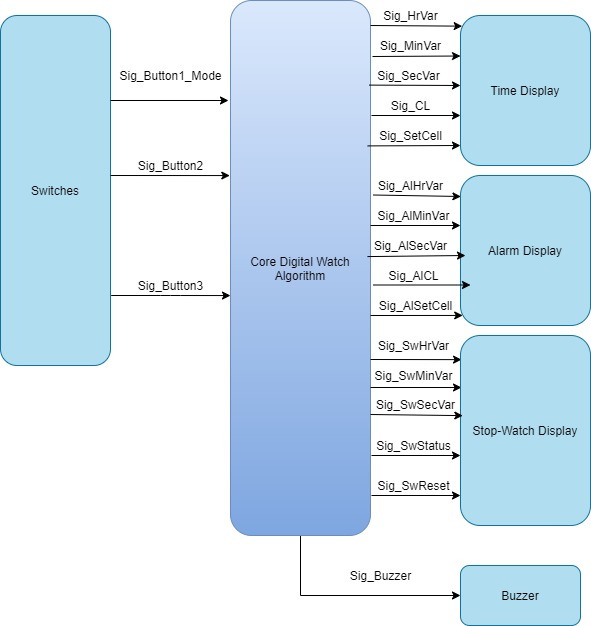


Figure 2 Software Input/output Diagram

**3.2 Input output Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Inputs** | **Name** | **Range** |  | **Unit** |
| Sig\_Button1\_Mode | 0:2 | NA | |
| Sig\_Button2 | 0:1 | NA | |
|  | Sig\_Button3 | 0:1 | NA | |
|  | Sig\_HrVar | 0:12 | Time-Hours | |
|  | Sig\_MinVar | 0:60 | Time-Minutes | |
|  | Sig\_SecVar | 0:60 | Time\_Seconds | |
|  | Sig\_CL | 0:1 | Am/Pm | |
|  | Sig\_SetCell | 0:16 | NA | |
|  | Sig\_AlHrVar | 0:12 | Time-Hours | |
|  | Sig\_AlMinVar | 0:60 | Time-Minutes | |
|  | Sig\_AlSecVar | 0:60 | Time-Seconds | |
|  | Sig\_AlCL | 0:1 | Am/Pm | |
|  | Sig\_AlSetCell | 0:16 | NA | |
|  | Sig\_SwHrVar | 0:12 | Time-Hours | |
|  | Sig\_SwMinVar | 0:60 | Time-Minutes | |
|  | Sig\_SwSecVar | 0:60 | Time-Seconds | |
|  | Sig\_Status | 0:1 | NA | |
|  | Sig\_Reset | 0:1 | NA | |
|  | Sig\_SecVar | 0:60 | Time-Seconds | |
| **Outputs** | **Name** | **Range** | **Unit** | |
|  | Sig\_Buzzer | 0:1 | NA | |

# Software Feature:

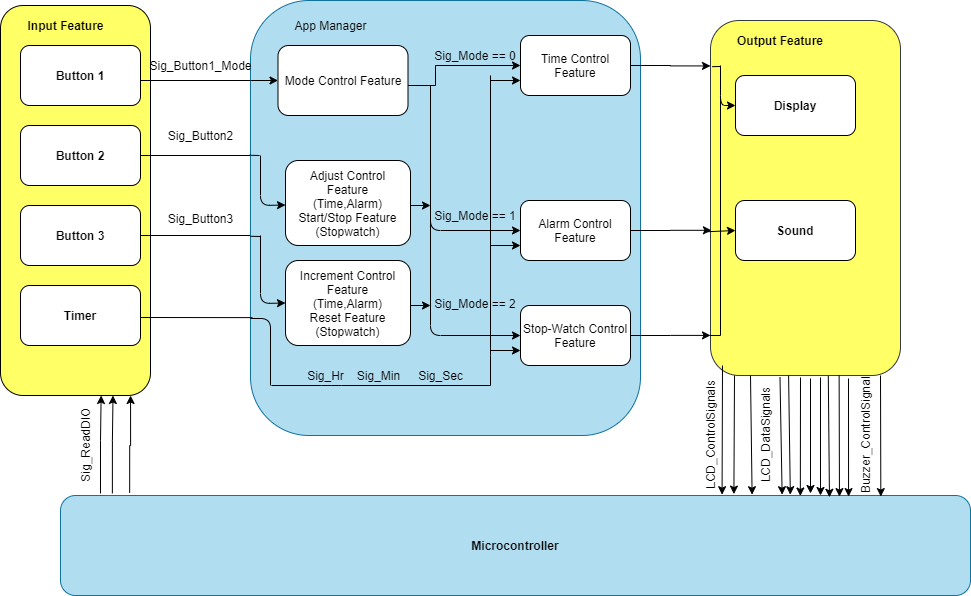


Figure 3 Software Features Diagram

# Static Architecture:

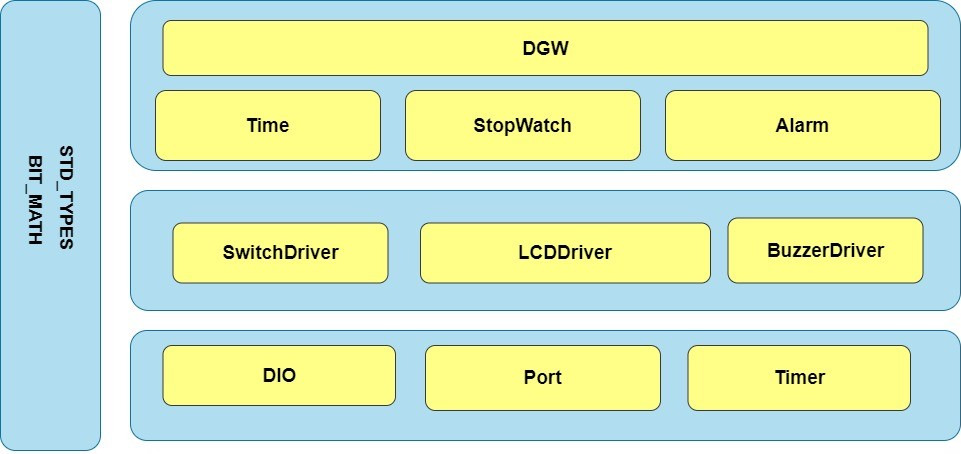


Figure 4 Static Layered Architecture

# 6 Requirements

## **6.1 MCAL Layer:**

## **6.1.1 DIO Component:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_01\_V01.0 | | **Covers** |  | \_\_\_\_\_\_\_\_\_\_\_\_ |
| **Author** | Norhan Nassar | | **Date** | 1/03/2020 | |
| **Description:** | Set Pin Direction for certain Port | | | | |
| **API** | STD\_ERORR DIO\_SetPinDirection(u8 Copy\_u8Port,u8 Copy\_u8Pin, u8 Copy\_u8Direction) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8  u8  u8 | Copy\_u8Port  Copy\_u8Pin  Copy\_u8Direction | | | Port from ‘A’ to ‘D’ as Ascii code  Pin from 0 to 7  Output -> 1 or input -> 0 |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If sent to the function its valid ranges of port, pin and direction  NOT\_OK -> If the ranges sent to the function is out of scope |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_02\_V01.0 | | **Covers** |  | \_\_\_\_\_\_\_\_\_\_ |
| **Author** | Norhan Nassar | | **Date** | 1/03/2020 | |
| **Description:** | Set or Reset Pin at certain port | | | | |
| **API** | STD\_ERORR DIO\_SetPinValue(u8 Copy\_u8Port,u8 Copy\_u8Pin, u8 Copy\_u8Value); | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8  u8  u8 | Copy\_u8Port  Copy\_u8Pin  Copy\_u8Value | | | Port from ‘A’ to ‘D’ as Ascii code  Pin from 0 to 7  Value -> 1 to set Pin high  Value -> 0 to set Pin low |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If sent to the function its valid ranges of port, pin and direction  NOT\_OK -> If the ranges sent to the function is out of scope |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_03\_V01.0 | | **Covers** |  | \_\_\_\_\_\_\_\_\_\_\_\_ |
| **Author** | Norhan Nassar | | **Date** | 1/03/2020 | |
| **Description:** | Get Pin value at certain Port  1 -> High  0 -> Low | | | | |
| **API** | STD\_ERORR DIO\_GetPinVal(u8 Copy\_u8Port,u8 Copy\_u8Pin,u8\* Copy\_PtrVal) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8  u8  u8\* | Copy\_u8Port  Copy\_u8Pin  Copy\_PtrVal | | | Port from ‘A’ to ‘D’ as Ascii code  Pin from 0 to 7  Pointer to return the status of the pin in it |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If sent to the function its valid ranges of port, pin and direction  NOT\_OK -> If the ranges sent to the function is out of scope |

### **6.1.2 Port Component:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_04\_V01.0 | | **Covers** |  | \_\_\_\_\_\_\_\_\_\_\_ |
| **Author** | Norhan Nassar | | **Date** | 1/03/2020 | |
| **Description:** | It initialize each Port and Pins to be output or input according to Port\_config file  Output -> 1 Input -> 0 | | | | |
| **API** | STD\_ERORR Port\_Initialize(void); | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If the Pins are Initialized Successfully  NOT\_OK -> If a problem occurred while initializing any pin |

### **6.1.3 Timer Component:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_05\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_01\_V01.2  Req\_PO3\_DGW\_SRS\_02\_V01.2  Req\_PO3\_DGW\_SRS\_20\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It makes Timer start to increment. | | | | |
| **API** | STD\_ERORR Timer\_Init(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If the timer initialization is done successfully  NOT\_OK -> If any problem occurred while initializing the timer |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_06\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_01\_V01.2  Req\_PO3\_DGW\_SRS\_02\_V01.2  Req\_PO3\_DGW\_SRS\_20\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It will display HrVar, MinVar, Secvar and CycleLength  And it will take which cell to display onto the LCD the it called LCD  to write these data onto this location. | | | | |
| **API** | STD\_ERORR Timer\_SetCallBack(void(\*Copy\_pvCallBackPtr)(void)) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void (\*) (void) | Copy\_pvCallBackPtr | | | Pointer to function that it will be implemented when Interrupt has been happened. |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If a function is sent to the callback  NOT\_OK -> If anything except a function is sent to the callback |

## **6.2 HAL Layer:**

# 6.2.1 Switch Component:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_07\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_09\_V01.2  Req\_PO3\_DGW\_SRS\_10\_V01.2  Req\_PO3\_DGW\_SRS\_14\_V01.2  Req\_PO3\_DGW\_SRS\_21\_V01.2  Req\_PO3\_DGW\_SRS\_23\_V01.2  Req\_PO3\_DGW\_SRS\_24\_V01.2  Req\_PO3\_DGW\_SRS\_25\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | Get Switch state according to its config file that it configures its Port and pin to get its status | | | | |
| **API** | STD\_ERORR Switch\_GetState (u8 Copy\_u8SwNum,u8\* Copy\_ptrState) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8  u8\* | Copy\_u8SwNum  Copy\_ptrState | | | Switch Number to get state of a certain switch  Pointer to state of the switch according to Switch\_config file  Pressed will represent 0 or 1  If RELEASE\_STATE defined as 1  So pressed will represent 0  If RELEASE\_STATE defined as 0  So pressed will represent 1 |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If the SwNum sent is in the scope  NOT\_OK -> If SwNum sent is out of the scope |

# 6.2.2 Buzzer Component:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_08\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_03\_V01.2  Req\_PO3\_DGW\_SRS\_04\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It sets buzzer state according to its config file that it congiures its port and pin. | | | | |
| **API** | STD\_ERORR Buzzer\_SetState(u8 State) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8 | State | | | Buzzer state to set its pin to this state  High -> 1  Low -> 0 |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If State Sent is 0 or 1  NOT\_OK -> If State send is not 0 nor 1 |

# 6.2.3 LCD Component:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_09\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_06\_V01.2  Req\_PO3\_DGW\_SRS\_08\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It clears the screen. | | | | |
| **API** | STD\_ERORR CLCD\_Clear(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If the function cleared the LCD successfully  NOT\_OK -> If any problem occurred while clearing the LCD |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_10\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_06\_V01.2  Req\_PO3\_DGW\_SRS\_08\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It takes Pointer to char of the string that it will be written as whole onto the LCD. | | | | |
| **API** | STD\_ERORR CLCD\_WriteString(const char\* Copy\_pchString); | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| const char\* | Copy\_pchString | | | Pointer to char of the string to be written onto LCD |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If sent character pattern valid in the CGROM  NOT\_OK -> If a pattern sent outside the scope of the CGROM |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_11\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_10\_V01  Req\_PO3\_DGW\_SRS\_11\_V01.2  Req\_PO3\_DGW\_SRS\_12\_V01.2  Req\_PO3\_DGW\_SRS\_13\_V01.2  Req\_PO3\_DGW\_SRS\_15\_V01.2  Req\_PO3\_DGW\_SRS\_16\_V01.2  Req\_PO3\_DGW\_SRS\_17\_V01.2  Req\_PO3\_DGW\_SRS\_18\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It sets new position for the LCD to start write from.  XPos -> which column  YPos -> which Row  To select which cell. | | | | |
| **API** | STD\_ERORR CLCD\_GoToXYPos(u8 Copy\_u8XPos,u8 Copy\_u8YPos); | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8  u8 | Copy\_u8XPos  Copy\_u8YPos | | | XPos -> which column  YPos -> which row |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If XPos is sent between 0-15 and YPos is sent between 0-1  NOT\_OK -> If any other range is sent either in XPos or YPos |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_12\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_06\_V01.2  Req\_PO3\_DGW\_SRS\_08\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It writes number onto the LCD. | | | | |
| **API** | STD\_ERORR CLCD\_WriteNum (u32 Copy\_u32Num); | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u32 | Copy\_u32Num | | | Number to be written onto the LCD |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If Numbers between 0-9 are sent  NOT\_OK -> If anything except 0-9 is sent to the function |

## **6.3** **APP Layer**

## **6.3.1 Time-Display Component**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_13\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_02\_V01.2  Req\_PO3\_DGW\_SRS\_10\_V01.2  Req\_PO3\_DGW\_SRS\_14\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It displays HrVar, MinVar, Secvar and CycleLength  And it takes which cell to display onto the LCD. | | | | |
| **API** | STD\_ERORR Time\_Display(u8\* Copy\_Pu8SetCell) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8\* | Copy\_Pu8SetCell | | | Certain Position to start write from onto the LCD |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If a valid SetCell between 0-15 is sent  NOT\_OK -> If anything is sent in SetCell except 0-15 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_14\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_11\_V01.2  Req\_PO3\_DGW\_SRS\_12\_V01.2  Req\_PO3\_DGW\_SRS\_13\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It adjusts the cursor position at LCD. | | | | |
| **API** | STD\_ERORR Time\_AdjustCursor (u8\* Copy\_Pu8SetCell) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8\* | Copy\_Pu8SetCell | | | Pointer to Certain Position |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If a valid SetCell between 0-15 is sent  NOT\_OK -> If anything is sent in SetCell except 0-15 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_15\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_14\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It increments Hrvar, SecVar and MinVar according to which cell LCD on at this time. | | | | |
| **API** | STD\_ERORR Time\_Incr(u8\* Copy\_Pu8SetCell) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8\* | Copy\_Pu8SetCell | | | Pointer to Certain Position |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If a valid SetCell between 0-15 is sent  NOT\_OK -> If anything is sent in SetCell except 0-15 |

## **6.3.2 Alarm mode requirements**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_16\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_15\_V01.2  Req\_PO3\_DGW\_SRS\_16\_V01.2  Req\_PO3\_DGW\_SRS\_17\_V01.2  Req\_PO3\_DGW\_SRS\_18\_V01.2  Req\_PO3\_DGW\_SRS\_19\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It displays AlHrVar, AlMinVar, AlSecvar and AlCycleLength  And it takes which cell to display onto the LCD. | | | | |
| **API** | STD\_ERORR Alarm\_Display(u8\* Copy\_Pu8SetCell) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8\* | Copy\_Pu8SetCell | | | Certain Position to start write from onto the LCD |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If a valid SetCell between 0-15 is sent  NOT\_OK -> If anything is sent in SetCell except 0-15 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_17\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_15\_V01.2  Req\_PO3\_DGW\_SRS\_16\_V01.2  Req\_PO3\_DGW\_SRS\_17\_V01.2  Req\_PO3\_DGW\_SRS\_18\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It adjusts the cursor position at LCD. | | | | |
| **API** | STD\_ERORR Alarm\_AdjustCursor (u8\* Copy\_Pu8SetCell) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8\* | Copy\_Pu8SetCell | | | Pointer to Certain Position |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If a valid SetCell between 0-15 is sent  NOT\_OK -> If anything is sent in SetCell except 0-15 |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_18\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_19\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It increments AlHrvar, AlSecVar and AlMinVar according to which cell LCD on at this time. | | | | |
| **API** | STD\_ERORR Alarm\_Incr(u8\* Copy\_Pu8SetCell) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| u8\* | Copy\_Pu8SetCell | | | Pointer to Certain Position |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If a valid SetCell between 0-15 is sent  NOT\_OK -> If anything is sent in SetCell except 0-15 |

### 6.3.3 Stop-Watch Component

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_19\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_22\_V01.2  Req\_PO3\_DGW\_SRS\_23\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It displays StHrVar, StMinVar, StSecvar.  And it takes which cell to display onto the LCD. | | | | |
| **API** | STD\_ERORR SW\_Display(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If variables of StopWatch are displayed successfully on the LCD  NOT\_OK -> If a problem occurred while displaying the StopWath Variables on the LCD |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_20\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_21\_V01.2  Req\_PO3\_DGW\_SRS\_22\_V01.2  Req\_PO3\_DGW\_SRS\_23\_V01.2  Req\_PO3\_DGW\_SRS\_24\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It checks which press at this time  First press -> start SW -> Set SW\_flag = 1 to start increment  Second press -> Stop SW -> Stopwatch\_flag = 1 to reset  Thirs press -> Resume SW\_flag = 0 | | | | |
| **API** | STD\_ERORR SW\_CheckStatus(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If SW\_Flag is within its range  NOT\_OK -> If SW\_Flag overflowed its range |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_20\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_25\_V01.2  Req\_PO3\_DGW\_SRS\_26\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It Resets SW\_flag and StopWatch\_flag | | | | |
| **API** | STD\_ERORR SW\_Reset(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If StopWatch Variables are set to zeros Successfully  NOT\_OK -> If a problem occurred while setting the StopWatch Variables to Zeros |

## **6.3.4 DGW Requirements**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_21\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_06\_V01.2  Req\_PO3\_DGW\_SRS\_09\_V01.2  Req\_PO3\_DGW\_SRS\_10\_V01.2  Req\_PO3\_DGW\_SRS\_21\_V01.2  Req\_PO3\_DGW\_SRS\_23\_V01.2  Req\_PO3\_DGW\_SRS\_25\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It reads Button 1 switch state and modifies Current\_Mode(Global Variable) signal to detect which mode is running know (Time or Alarm or Stop Watch) | | | | |
| **API** | STD\_ERORR ReadMode(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If current mode is within its range 0-2  NOT\_OK -> If current mode overflowed its range |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_22\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_02\_V01.2  Req\_PO3\_DGW\_SRS\_06\_V01.2  Req\_PO3\_DGW\_SRS\_07\_V01.2  Req\_PO3\_DGW\_SRS\_10\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It checks Current\_Mode signal to choice which to display -> Time/Alarm/Stopwatch  Current\_Mode is a global variable. | | | | |
| **API** | STD\_ERORR Display(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If current mode is within its range 0-2  NOT\_OK -> If current mode overflowed its range |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_23\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_11\_V01.2  Req\_PO3\_DGW\_SRS\_12\_V01.2  Req\_PO3\_DGW\_SRS\_13\_V01.2  Req\_PO3\_DGW\_SRS\_15\_V01.2  Req\_PO3\_DGW\_SRS\_16\_V01.2  Req\_PO3\_DGW\_SRS\_17\_V01.2  Req\_PO3\_DGW\_SRS\_18\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It reads Button 2 switch state  If (Pressed) -> Check Current\_Mode signal to detect which mode is running know.  If (Current\_Mode == 0) -> call Time\_voidAdjustCursor()  If (Current\_Mode == 1) -> call Alarm\_voidAdjustCursor()  If (Current\_Mode == 2) -> call SW\_voidCheckstatus()  Current\_Mode is a global variable. | | | | |
| **API** | STD\_ERORR ReadPos(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If current mode is within its range 0-2  NOT\_OK -> If current mode overflowed its range |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_24\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_14\_V01.2  Req\_PO3\_DGW\_SRS\_19\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It reads Button 3 switch state  If (Pressed) -> Check Current\_Mode signal to detect which mode is running know.  If (Current\_Mode == 0) -> call Time\_voidIncr()  If (Current\_Mode == 1) -> call Time\_voidIncr()  If (Current\_Mode == 2) -> call SW\_voidReset()  Current\_Mode is a global variable. | | | | |
| **API** | STD\_ERORR ReadInc(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If current mode is within its range 0-2  NOT\_OK -> If current mode overflowed its range |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_25\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_20\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It checks Buzzer\_signal  If (Buzzer\_signal ) -> turn Buzzer on  If (Buzzer\_signal) -> turn Buzzer off  Buzzer\_signal is a global variable. | | | | |
| **API** | STD\_ERORR CheckBuzzerState(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If buzzer set at 1 and reset at 0  NOT\_OK -> If undefined behavior occurs regarding the buzzer |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req\_ID** | Req\_PO3\_DGW\_GDD\_26\_V01.0 | | **Covers** |  | Req\_PO3\_DGW\_SRS\_01\_V01.2  Req\_PO3\_DGW\_SRS\_02\_V01.2  Req\_PO3\_DGW\_SRS\_03\_V01.2 |
| **Author** | Norhan Nassar | | **Date** | 27/2/2020 | |
| **Description:** | It increments HrVar, MinVar and SecVar every 1 sec by passing this function to the ISR of the timer.  And it checks Current\_Mode signal  If(Current\_Mode == 1) -> Alarm Mode  So if (HrVar,MinVar and SecVar == AlHrVar, AlMinVar and AlSecVar )  Buzzer\_signal = 1 | | | | |
| **API** | STD\_ERORR TimeInc(void) | | | | |
| **Parameters** | **Type** | **Name** | | | **Description** |
| void | \_\_\_\_\_ | | | \_\_\_\_\_ |
| **Return** | **Type** | **Name** | | | **Description** |
| STD\_ERORR | \_\_\_\_\_ | | | STD\_ERORR  OK -> If current mode is within its range 0-2  NOT\_OK -> If current mode overflowed its range |