Detailed Design (DD)

Smart home

Authors: Burners Team (Ahmed Salah, Basem Moufreh, Hassan El Gabass , Hazem El Morshedi , Mohamed Safwat, Mohamed Abd El Azeem Sallam).

Customer: NTI

Instructor: Mahmoud Ali, Ahmed Abd El Raheim.

APIs

1- Initializing LCD
2 -Initialize the Ultrasonic API
3- Initializing ADC API
4- Set Pin Direction API
5- Navigate the screen API
6-Display String API
7-Integer to String API
8- Clear Screen API
9- water level API
10- ADC reading API
11- calculate the temperature API
12- get number API
13- save new password API
14- login API
15- port initialization API
16- UART initialization API
17- timer initialization API
18- I2C initialization API
19- Keypad API
20- EEPROM write API

1- Initializing LCD

Name	Initialization LCD function
Prototype	STD HAL_LCD_init(void);
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	STD E_OK= 0 STD E_NOT_OK=1
Description	This Function is responsible of initializing the LCD: 1. Setup the LCD pins directions by use the GPIO driver. 2. Setup the LCD Data Mode 4-bits or 8-bits.
Covered Requirement	[SRS_S_H_200], [SRS_S_H_205], [SRS_S_H_209], [SRS_S_H_303]

2 -Initialize the Ultrasonic API

Name	Initialize the Ultrasonic function
Prototype	STD HAL_Ultrasonic_init(void);
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	STD E_OK= 0 STD E_NOT_OK=1
Description	1. Initialize the Trigger pin direction 2. Initialize the ICU driver and Its Call back function.
Covered Requirement	[SRS_S_H_205], [SRS_S_H_300]

3- Initializing ADC API

Name	Initializing ADC peripheral function	
Prototype	STD HAL_ADC_voidInit (void);	
Parameter in	-	
Parameter out	-	
Parameter in-out	-	
Return Type	STD E_OK= 0 STD E_NOT_OK=1	
Description	This Function is responsible of initializing the ADC peripheral	
Covered Requirement	[SRS_S_H_203], [SRS_S_H_206], [SRS_S_H_207], [SRS_S_H_208]	

4- Set Pin Direction API

Name	Set pin direction function	
Prototype	STD MCAL_GPIO_setupPinDirection(pin_t * pin);	
Parameter in	const pin_t * pin	Pointer to Struct of pin_t data type
Parameter out	-	
Parameter in-out	-	
Return Type	STD E_OK= 0 STD E_NOT_OK=1	
Description	This Function is responsible of: Set pin direction of pin_t instance	
Covered Requirement	[SRS_S_H_207], [SRS_S_H_302]	

5- Navigate the screen API

Name	Move Cursor Function	
Prototype	STD HAL_LCD_moveCursor(uint8 row,uint8 col);	
Parameter in	uint8 row,uint8 col	Row for row index and col for column index
Parameter out	-	
Parameter in-out	-	
Return Type	STD E_OK= 0 STD E_NOT_OK=1	
Description	This Function is responsible of moving cursor to a specific position	
Covered Requirement	[SRS_S_H_200], [SRS_S_H_205]	

6-Display String API

Name	Display String in specific position Function
Prototype	STD HAL_LCD_displayStringColRow(uint8 data,uint8 row,uint8 col);
Parameter in	uint8 data,uint8 row,uint8 col
Parameter out	-
Parameter in-out	-
Return Type	STD E_OK =0 STD E_NOT_OK=1
Description	This Function is responsible of: Display the required string in a specified row and column index on the screen.
Covered Requirement	[SRS_S_H_200], [SRS_S_H_205], [SRS_S_H_209]

7-Integer to String API

Name	Integer to String Function	
Prototype	STD HAL_LCD_integerToString(u16 data);	
Parameter in	U16 data	Integer value that would be converted and displayed to LCD
Parameter out	-	
Parameter in-out	-	
Return Type	STD E_OK =0 STD E_NOT_OK=1	
Description	This Function is responsible of: Converting the integer to string and display on screen.	
Covered Requirement	[SRS_S_H_200], [SRS_S_H_205], [SRS_S_H_209]	

8- Clear Screen API

Name	Clear Screen Function
Prototype	STD HAL_LCD_clearLcd(void);
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	STD E_OK =0 STD E_NOT_OK=1
Description	This Function is responsible of clearing the screen.
Covered Requirement	[SRS_S_H_200], [SRS_S_H_205], [SRS_S_H_209]

9- water level API

Name	Measure the water level	
Prototype	STD HAL_Ultrasonic_readDistance(uint16 * _distance)	
Parameter in	-	
Parameter out	uint16 * _distance	Pointer to variable to store the distance
Parameter in-out	-	
Return Type	STD E_OK =0 STD E_NOT_OK=1	
Description	This Function is responsible of measure the distance	
Covered Requirement	[SRS_S_H_200], [SRS_S_H_204]	

10- ADC reading API

Name	Get ADC reading	
Prototype	STD HAL_ADC_u16GetChannelReading(ADC_Channel_SingleEnded_t Copy_u8Channel,uint16* reading)	
Parameter in	ADC_Channel_SingleEnded_t Copy_u8Channel	Enum to the channel of ADC
Parameter out	Uint16 * reading	Pointer to variable to store the reading of ADC
Parameter in-out	-	
Return Type	STD E_OK =0 STD E_NOT_OK=1	
Description	This Function is responsible of get readings from ADC	
Covered Requirement	[SRS_S_H_204], [SRS_S_H_205]	

11- calculate the temperature API

Name	calculate the temperature	
Prototype	STD HAL_LM35_u8GetTemp(uint16 reading,uint8 *temp)	
Parameter in	uint16 reading	Value of the ADC reading
Parameter out	Uint8 * temp	Pointer to variable to store the temperature value after calculation
Parameter in-out	-	
Return Type	STD E_OK =0 STD E_NOT_OK=1	
Description	This Function is responsible of calculate the temperature value	
Covered Requirement	[SRS_S_H_205], [SRS_S_H_206]	

12- get number API

Name	calculate the temperature
Prototype	u32 GetNumber(void);
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	U32 value of the number
Description	This Function is responsible of take the number from user via keypad
Covered Requirement	[SRS_S_H_200], [SRS_S_H_202]

13- save new password API

Name	Save new password
Prototype	u32 MakeNewPassword(void);
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	U32 return the passcode
Description	This Function is responsible of save the new password
Covered Requirement	[SRS_S_H_204], [SRS_S_H_206]

14- login API

Name	Login function
Prototype	void login(void);
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	-
Description	This Function is responsible of login the user
Covered Requirement	[SRS_S_H_200], [SRS_S_H_300]

15- port initialization API

Name	Port initializing
Prototype	STD PORT_voidInit(void);
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	STD E_OK =0 STD E_NOT_OK=1
Description	This Function is responsible of initializing the port
Covered Requirement	[SRS_S_H_205], [SRS_S_H_302] , [SRS_S_H_206], [SRS_S_H_207], [SRS_S_H_208],

16- UART initialization API

Name	initializing uart protocol
Prototype	STD USART_voidInit();
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	STD E_OK =0 STD E_NOT_OK=1
Description	This Function is responsible of initializing the UART module
Covered Requirement	[SRS_S_H_201], [SRS_S_H_202] , [SRS_S_H_203],

17- timer initialization API

Name	initializing timer
Prototype	STD TIMER0_voidInit();
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	STD E_OK =0 STD E_NOT_OK=1
Description	This Function is responsible of initializing the timer module
Covered Requirement	[SRS_S_H_201], [SRS_S_H_202] , [SRS_S_H_203],

18- I2C initialization API

Name	initializing I2C protocol
Prototype	STD I2C_Init();
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	STD E_OK =0 STD E_NOT_OK=1
Description	This Function is responsible of initializing the I2C module
Covered Requirement	[SRS_S_H_200], [SRS_S_H_204]

19- Keypad API

Name	Get key pressed
Prototype	U8 KPD_u8GetPressedKey();
Parameter in	-
Parameter out	-
Parameter in-out	-
Return Type	U8 key_preesed
Description	This Function is responsible of getting the pressed key from keypad
Covered Requirement	[SRS_S_H_200], [SRS_S_H_209]

20- EEPROM write API

Name	Write in EEPROM	
Prototype	STD EEPROM_WriteByte(u8* address,u8 value);	
Parameter in	u8* address	Location the data will store in
	u8 value	The data that will be store in EEPROM
Parameter out	-	
Parameter in-out	-	
Return Type	STD E_OK =0 STD E_NOT_OK=1	
Description	This Function is responsible of write data to EEPROM	
Covered Requirement	[SRS_S_H_201], [SRS_S_H_202]	