12/2021

EE-366 Project Report

Ensure Sterilization

Section: IA

Instructor: Dr. Wassim Zouch

No	Member Name	ID
1	**Mohammed Mousa Alshamrani**	1847991
2	**Ammar Slahden Hobi**	1768459
3	**Asaad Waleed Dadoush**	1766585

Table of Content

List of Figures

List of Tables

Introduction:

What is the increase in the number of people infected with COVID-19 recently, and the fact that some people in closed places do not follow safety procedures from contracting COVID-19 In this report, we will explain the EE 366 project, which is based on reminding people of the need to sterilize hands? this report will contain the whole process that use to implement the project starting from the project flow chart, later the implementation section will start to describe the whole process the program that has been written and the use of the simulation tools to correct and make sure the program working correctly later the final implementation in the using the electric and electronic components on the hardware design, so the project in simple words is it will simulate a real-life application on the breadboard so the project can be used on buildings or hospitals, etc, so whenever a person enters the building there is a push-button before entering the door so it will allow the person to open the door and enter meanwhile there is a counter will start countdown and the person must sterilize his/her hands before the count gose to zero it will be about 10 secounds if the counter gose to zero the alarm will start working and the person must sterilize his/her hands other wise he/her will not be allowed to enter the building the process will be expliend in deatil in the next sections.

Implementation:

In this section of the report, everything about our project will be explained, and it contains the Flowchart, codes, and simulation of the way the project works, as well as the pieces used in the project, the presentation of the outputs, and finally the shape of the project in reality. We used the MPLAB program to write codes using the C programming language, and we used the Proteus 8 Professional program to design the shape of the electrical circuit and simulate the outputs, and the lucid website to design the Flowchart. The work was distributed among the team members and the project was built in Asaad house.

Flowchart:

This section will show the flowchart that was made for this project and describe each stage and how this flowchart help to make the code and implement the project.

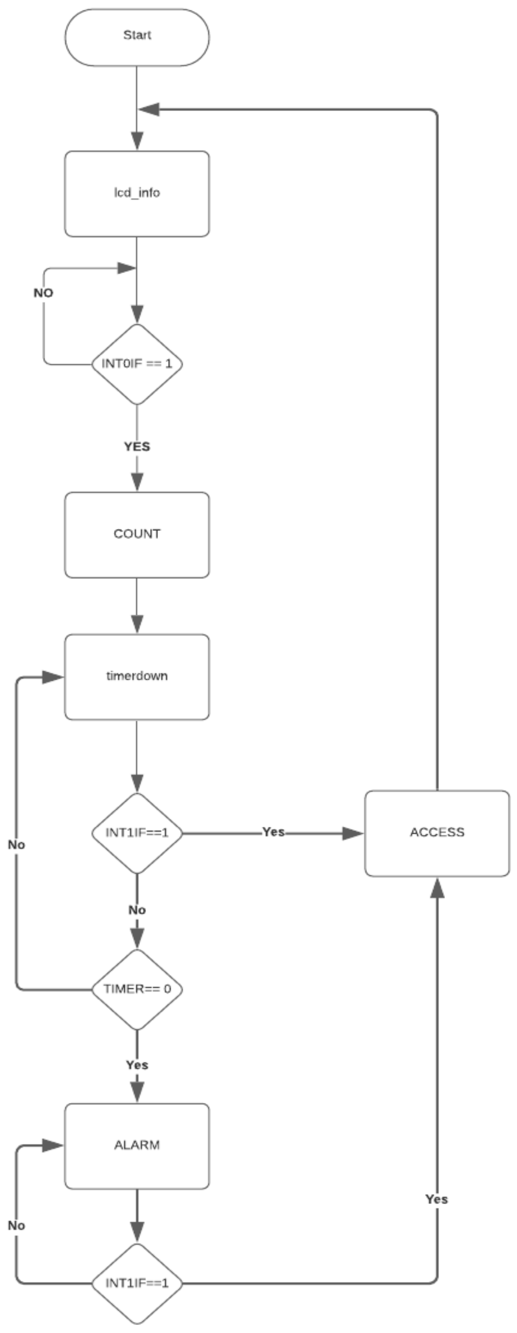


Figure 1: Flowchart for the program

As shown in figure 1 when the program is executed first thing is will start to load the screen then the program will wait for the INT0IF to occur if yes it will go to the interrupt method if nothing happened it will keep waiting for INT0IF to be true, next the method count will be executed therefore timer down will start also, in timer down process there is a flag to check the INT1IF during the method timer down is running if the flag is 1 it will break the loop and execute the method access if no the method will keep running until the timer goes to 0 and break the method and then If the timer is zero the method alarm will be executed inside the alarm method it will keep check for the flag INT1IF if it occurs it will break alarm and executed access method if no it will keep in alarm method forever.

Code:

This section will show the stage of coding and the logic of the program will go into each method and explain each one and how it will affect the behavior of this project.

```
# include <xc.h>
# include <stdlib.h>
# include <stdio.h>
# include <string.h>
# include "config.h"
# include "lcd.h"
```

First, include all the header classes that are needed for this project so including LCD.h header class because will use some instruction from that class later in the main class project to display the information on the LCD.

```
void main(void){
    loading();
    setup();
    Lcd_Clear();
    while (1){

        lcd_info();

    }

    return;
}
```

Here is the main class it very simply made it like this to make the code more readable and easier to implement will go into each method of those in detail and explain them.

```
void loading(){
    TRISD = 0;
    TRISE = 0;
    Lcd_Init();
    Lcd_Set_Cursor(1, 1);
    Lcd_Write_String(idle);
    unsigned i = 5;
    while (i > 0){
        Lcd_Write_String(".");
        __delay_ms(500);
        i--;
    }
    Lcd_Clear();
}
```

So this is the loading method stating to configure the LCD ports and make port d and port c as an output starting to initialize the LCD and set the cursor in the first line starting to load the program the string will appear loading……. For 2.5 seconds only displaying the string then I will load the whole program to make sure everything will work correctly.

```
void setup(void){
    LATC = 0;
    ANCON0 = 0; // only digital pins
    ANCON1 = 0;
    OSCCON = 0x60; // 8 MHz
    // initializeports I / O

    TRISC = 0;
    TRISB0 = 1;
    TRISB1 = 1;
    // enable interrupts
    INT0IF = 0;
    INT0IE = 1;
    INT1IF = 0;
    INT1IE = 1;
    GIE = 1;

}
```

Here in the setup method is used to configure the i/o ports and the frequency then enabling all the interrupts configuration for the interrupts INT0 and INT1 finally enable the GIE bit so here all the interrupts are needed are enabled now.

```
void lcd_info(){

    Lcd_Set_Cursor(1, 1);
    Lcd_Write_String("Enter: ");
    Lcd_Set_Cursor(1, 7);
    sprintf(myStr, "%d", enter_count);
    Lcd_Write_String(myStr);
    Lcd_Set_Cursor(2, 1);
    Lcd_Write_String("Access: ");
    Lcd_Set_Cursor(2, 8);
    sprintf(myStg, "%d", access_count);
    Lcd_Write_String(myStg);
    __delay_ms(500);
}
```

The use of the lcd_info method to keep updating the LCD with information about the count under and the access number and display it on the LCD stating to put the cursor in the first line for the entry number then because the count is an integer it cant be print out on the LCD so creating an array of char to store that int on the array and convert it into a char so later it can be displayed on the LCD same for access number but its location on the second line on the LCD.

```
void__interrupt() intrr(){
    if (INT0IF & & INT0IE){
        timer = 10;
        count();
        timerdown();
        if (LATB1 == 0){
            alarm();
        }
    INT0IF = 0;
    }
    if (INT1IF & & INT1IE){
        access();
        __delay_ms(500);
        Lcd_Clear();
        lcd_info();

        INT1IF =0;
    }
    return;
}
```

Here are the interrupts that are used in this program starting with checking the INT0 status if is used it will start to initiate the timer value which is 10 seconds then it will call the count method after that the time down method so if LATB1 never use the alarm method will execute, meanwhile it will keep checking the INT1 status if it occurs it will execute the access method and break form alarm method and top it.

```
void count(){
    enter_count++;
    Lcd_Set_Cursor(1, 7);
    sprintf(myStr, "%d", enter_count);
    Lcd_Write_String(myStr);
    return;
}
```

The method count is used for increasing the number of entries whenever the LATB0 is high then print the number of the count beside the Enter on the LCD.

```
void timerdown(){

    while (1){
        if (INT1IF == 1){
        break;
        }
        Lcd_Set_Cursor(1, 9);
        Lcd_Write_String(" Counter");
        __delay_ms(1000);
        Lcd_Set_Cursor(2, 12);
        sprintf(mycount, "%d", timer);
        Lcd_Write_String(mycount);
        if (timer == 9){
            Lcd_Set_Cursor(2, 13);
            Lcd_Write_String(" ");
        }
        --timer;
        if (timer == 0){
            break;
        }
    }
    if (INT1IF == 0){
        __delay_ms(1000);
        Lcd_Set_Cursor(1, 9);
        Lcd_Write_String("        ");
        Lcd_Set_Cursor(2, 12);
        Lcd_Write_String("   ");
    }
    return;
}
```

The method timer down used to start counting down the value of the timer meanwhile it will keep checking the value of the INT1IF if it occurs it will break this method and active the access method if not it will keep decreasing the value of timer until it goes zero the method alarm will start working.

```
void alarm(){
    Lcd_Clear();
    while (INT1IF == 0){
        __delay_ms(300);
        LATC2 = !LATC2;
        LATC5 =  !LATC5;
        Lcd_Set_Cursor(1, 1);
        Lcd_Write_String("Please sanitize");
        Lcd_Set_Cursor(2, 1);
        Lcd_Write_String("   your hands");

    }
    return;
}
```

The method alarm will keep working after the method timer down executed until the value of INT1IF goes 1 if it goes 1 it will break the method alarm if not the method will keep toggling the LATC2 & LATC5 and print out a string on the LCD it's like a warning message.

```
void access(){
    LATC2 = 0; // RED
    Lcd_Clear();
    Lcd_Set_Cursor(1, 1);
    Lcd_Write_String("======Done======");
    unsigned int L = 4;
    while (L > 0){
        __delay_ms(80);
        LATC3 = !LATC3; // GREEN
        LATC5 = !LATC5; // BUZ
        L--;
    }
    if (access_count < enter_count){
        access_count++;
    }
    LATC3 = 0; // GREEN
    LATC5 = 0;

    return;
}
```

The method access when it is called it will clear the LATC2 then it will print out a string on the LCD then it will toggle LATC3 & LATC5 two times then it will increase the access count to be printed later on the LCD after that it will clear LATC3 & LATC5.

Simulation

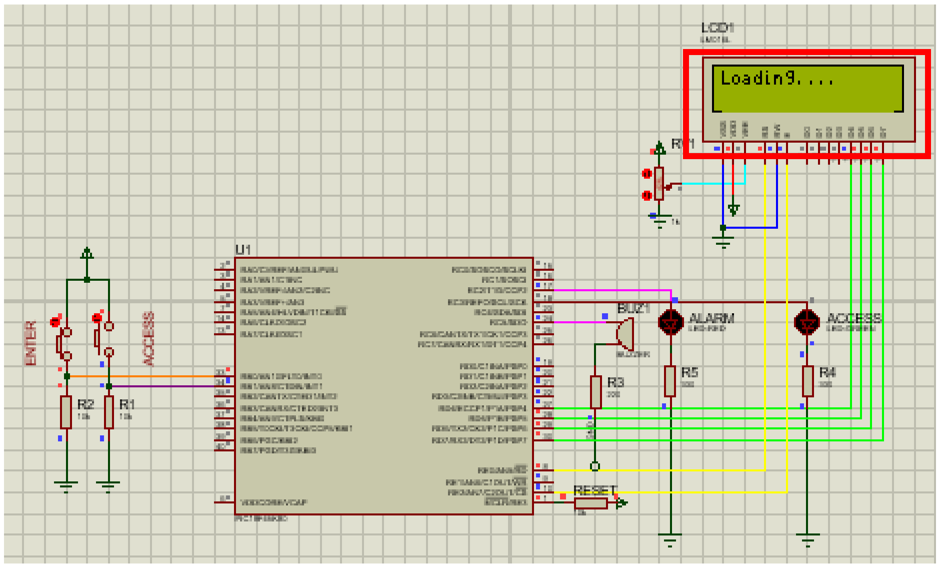


Figure 2: Simulation Results 1

The First of all, whence the device On, the screen start loading.

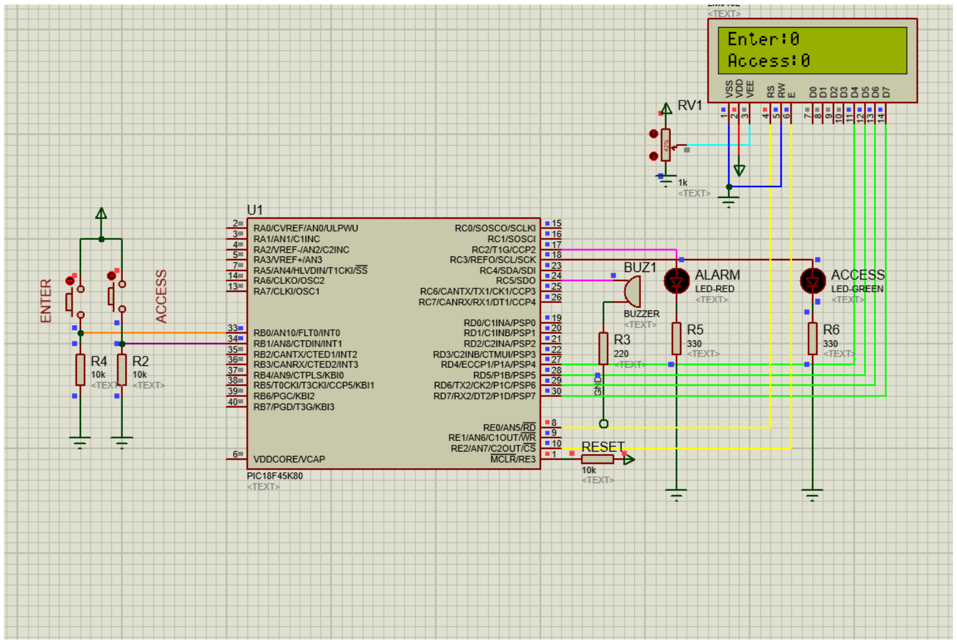
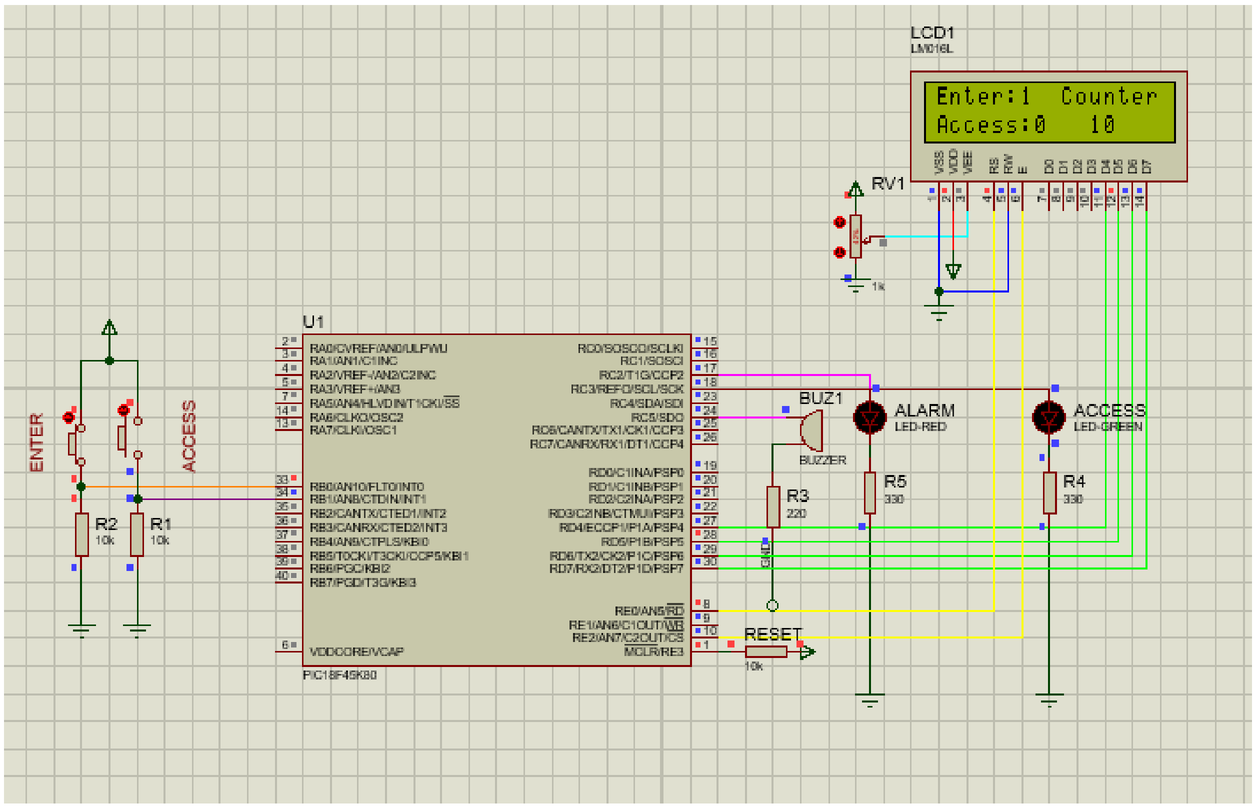


Figure 3: Simulation Results 2

Then Screen will provide the Number of People who enter the Area (One by one) and the number of who used the sterilizer bottle.



Enter:1 Counter
Access:0   10

Figure 4: Simulation Results 3

When the Person enters the hall, stimulated (by PB 1)the number of entering will increment and start decrement counter (delay (10 s) des).

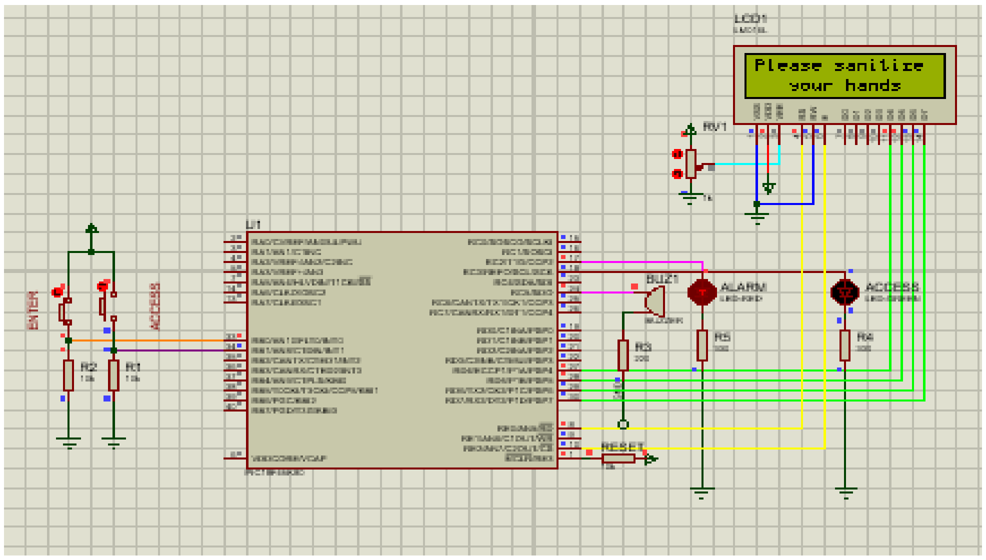


Figure 5: Simulation Results 4

In case the person has not used a bottle of sterilization (PB2 no Pressed)after finishing counter decrement until 0, the LCD screen will display “please sterilization your hand” at the same time the buzzer start buzz and Blinking red LED as warming.

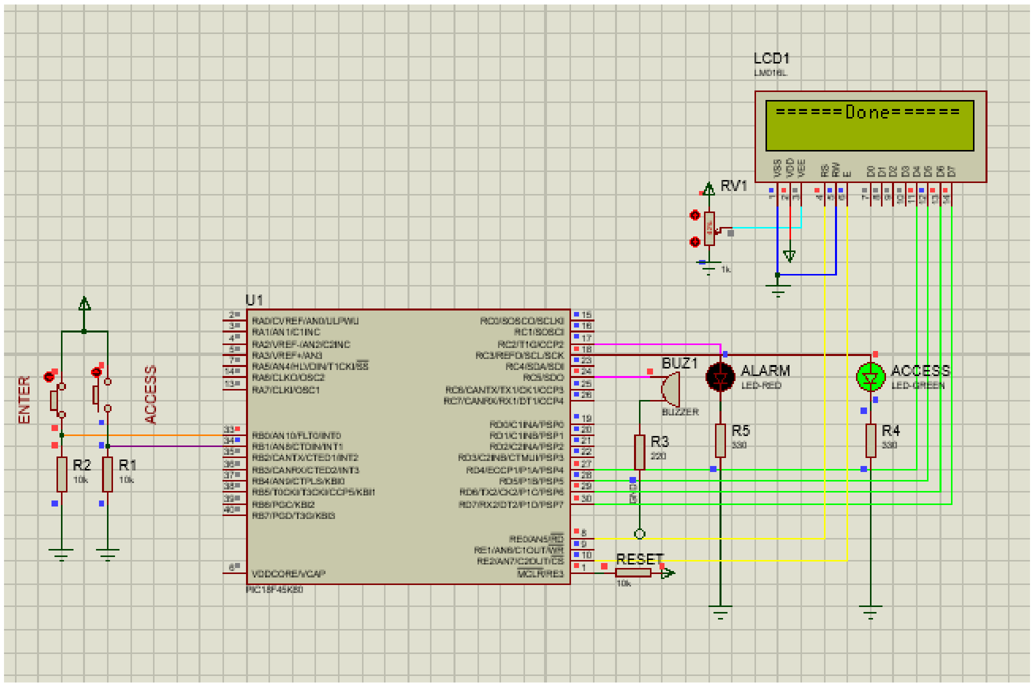


Figure 6: Simulation Results 5

Once the person used Sterilization bottle (PB2 pressed)then LCD screenprint ==Done==

At the same time, Buzzer will be cleared and Blinking one-second green LED.

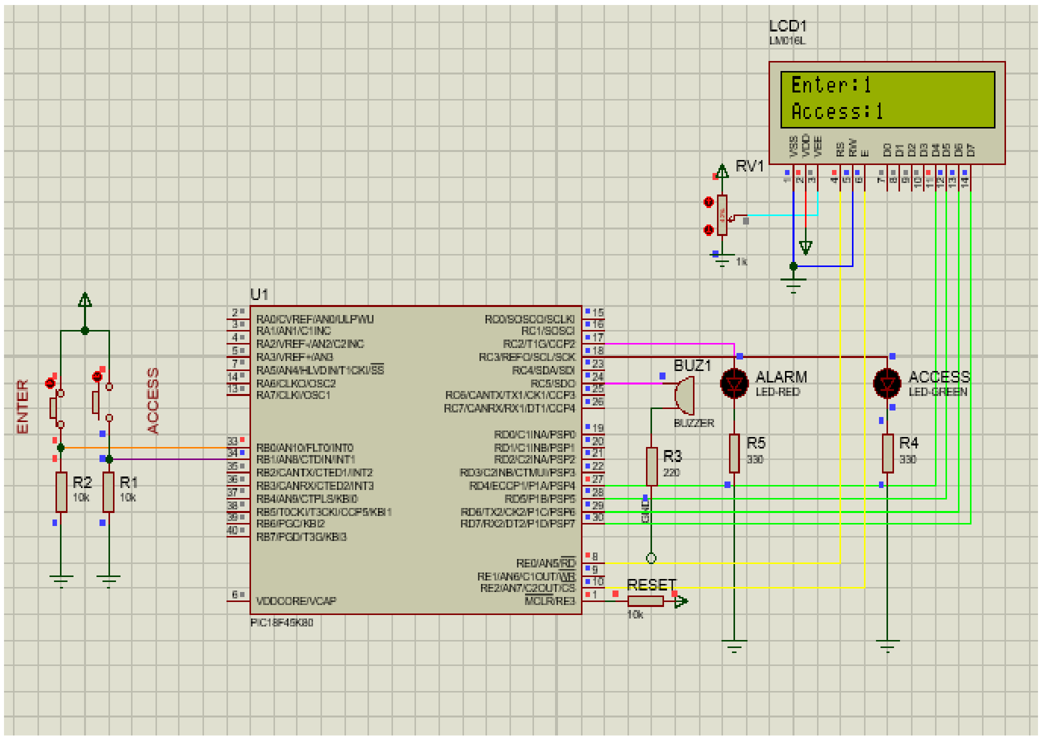
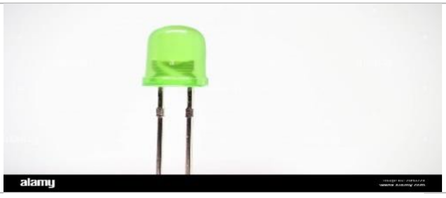
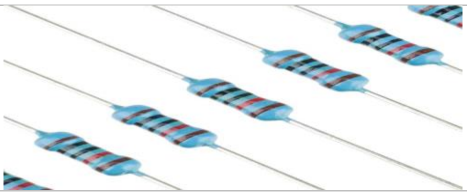
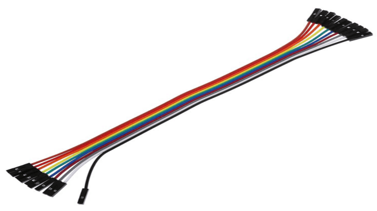
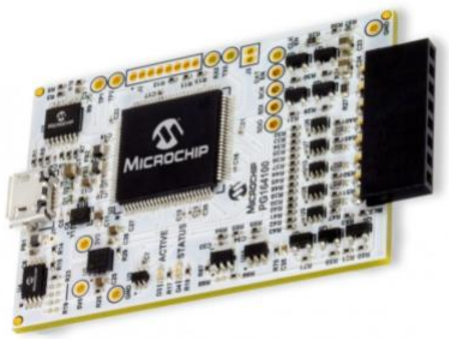


Figure 7: Simulation Results 6

After that LCD static data about several enter persons and who used sterilization bottle, and keep working like that.

Table 1: Components of the Artifact

Name	Function	Picture
PIC18f45K80	**microcontroller device that needs it, to execute the instructions that performed before to specific function.**	
Push Button (1)	**Used as a sensor for a person when entering**	
Push Button (2)	**Used as a sensor to sterilizer bottle cover**	
BUYER	**Need for Warning(Sound) system**	
LED(1)	**Blinking a red color as Warming.**	
LED(2)	**Blinking a green color as a Natural case**	
LCD Screen	**to Display all results and monitor case**	
Power Supply	**two power supplies Connected with PIC and LCD to take power (9 V)**	
Resistances 10KΩ	**Connected with two bush buttons and with the power supply to maintain the current**	

Bread Board	**To connect all components in one circuit**	
Voltage regulator	**To maintain the amount of voltage to desire state (5 V)**	
Wires	**To connect the components each other**	
Variable resistance	**It change the value of resistance (it connect with screen)until reach to desired state**	
M-Snap	**For programming PIC18 ,which take instructions from MPLAB program**	

Hardware

Here is the final stage of implementing the project after testing the code and simulating it's time to implement it on the board to give the final check is everything is working as expected.

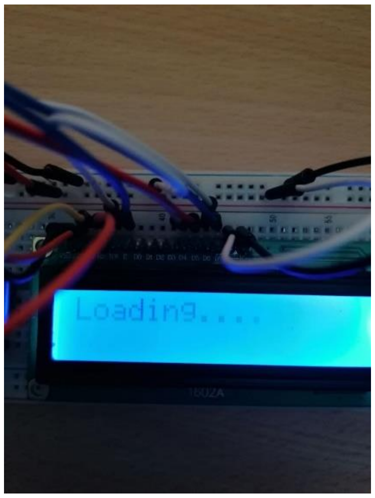


Figure 8: LCD output 1

As shown in the figure above this is the initial stage of starting to display the loading on the LCD

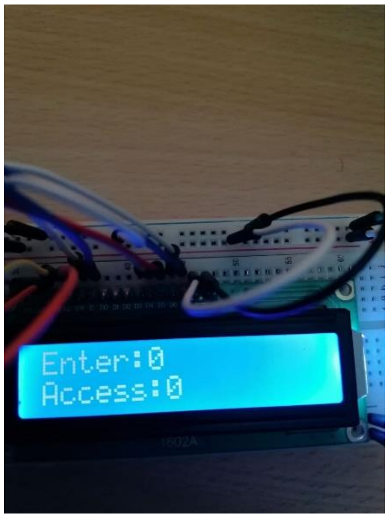


Figure 9: LCD output 2

So this figure shows the idle output on the screen it will show the entry and access number.

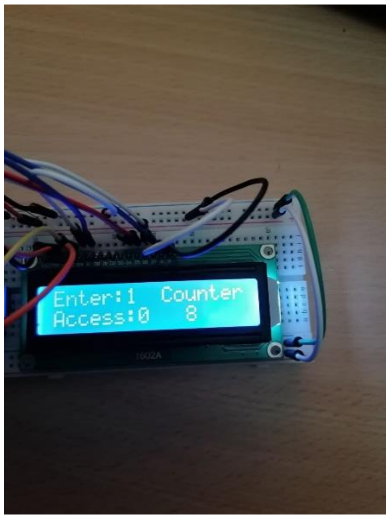


Figure 10: LCD output 3

When pressing the pushbutton 1 the counter will start to count down until it reaches zero.

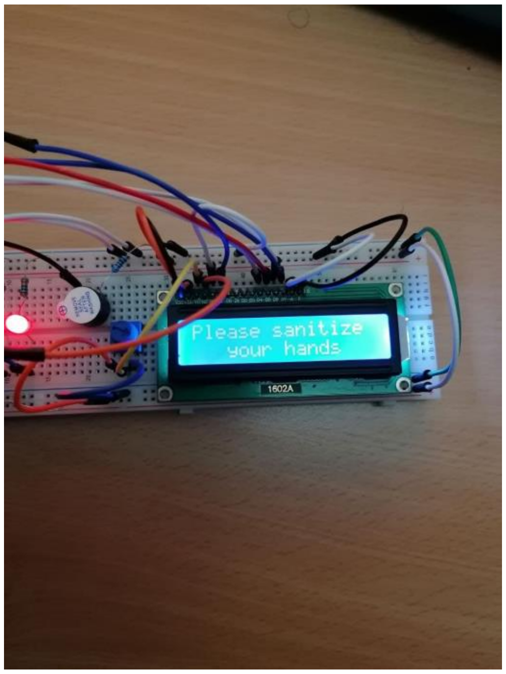


Figure 11: LCD output 4

Whenever it each zero it will print out this message and the red led will start to toggle also the buzzer.

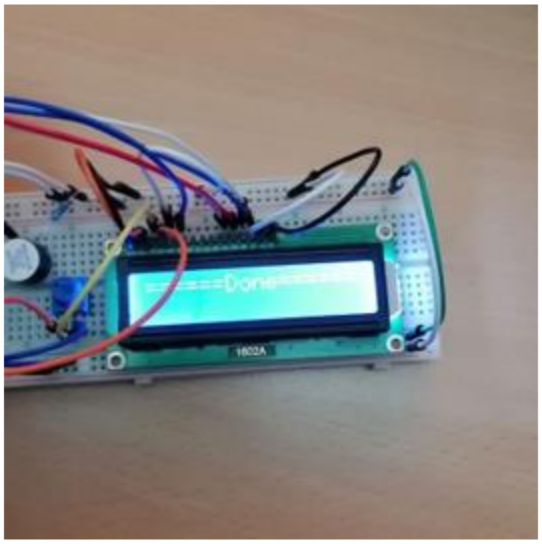


Figure 12: LCD output 5

Here is this figure is shown if pushbutton 2 is pressed the green time will start to toggle 2 times also the buzzer.

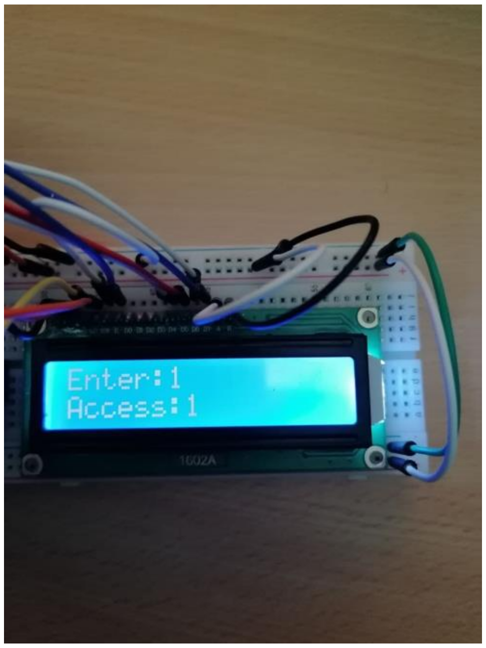


Figure 13: LCD output 6

Finally after doing all of those this the output of the LCD shows the count and access number.

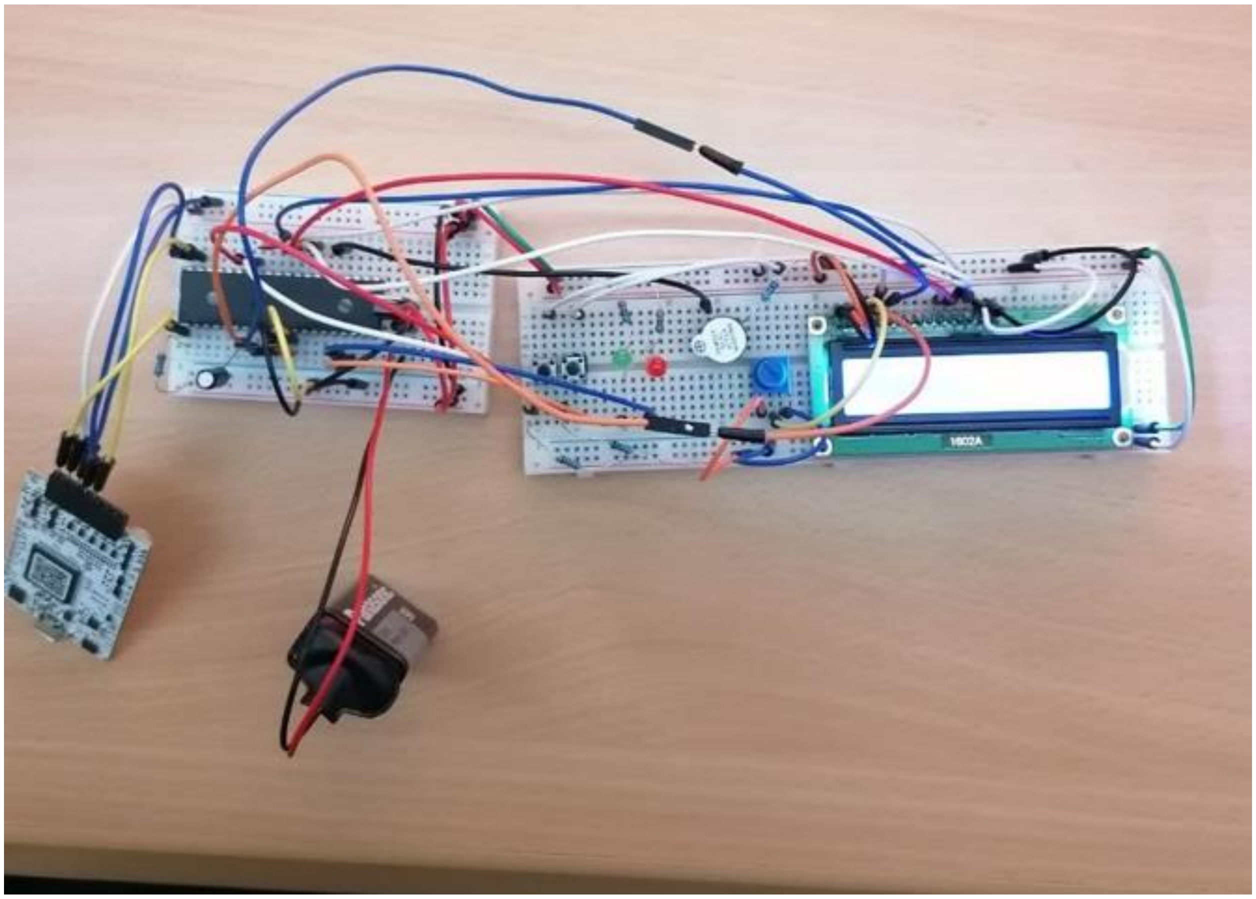


Figure 14:Project hardware

Here is the final implementation for the project after assembling it as shown in the figure above.

Discussion :

After completing the project, we learned how to use PIC18F45K80, connect it to an electrical circuit, and use MP-SNAP to transfer code from the computer to the electrical circuit parts to perform the required work. We also found that our project should be implemented at present, especially in closed places, to combat the spread of COVID-19. Project performance has been checked more than 50 times with all capabilities and outputs. This project is working well, but one of the problems that we faced in implementing our project was the lack of time, especially during the testing period and the high cost, as we hoped that the project would be developed in the coming months and used officially at home to limit and control the spread of COVID-19.

References

[1] "Github," [Online]. Available: https://github.com/.

[2] "mirochip," [Online]. Available: https://www.microchip.com/.

[3] "lucid," [Online]. Available: https://www.lucidchart.com/.

Appendices:

Appendix i

Work distribution

#	Task	Load	Distribution		
			Mohammed	Ammar	Asaad
1	Style & Formatting	12			12
2	Introduction	8	8		
3	implementation report	8	8		
4	Flowchart	12	4	3	5
5	Flowchart report	8			8
6	Code	20			20
7	Code report	12			12
8	Simulation	12			12
9	Simulation report	8		8	
10	Hardware design	20			20
11	Hardware design report	8			8
12	Discission	8	8		
13	Appendices	5			5
14	Power point	12		7	5
15	Meetings	20	6.6	6.6	6.6
16	Revision the report	15			15

Name	Member Key	Points	Percentage %
Mohammed	1	34.6	18.40425532
Ammar	2	24.6	13.08510638
Asaad	3	128.6	68.40425532
Total Points		188	

Amount of work Scale				
Easy	Moderate	Much	Hard	Extreme
5	8	12	15	20

Appendix ii

Code:

Main class part 1

```
/ *
*File: main.c
*Author: AsaadDadoush
*
*Created on November 26, 2021, 1: 27 AM
* /

# include <xc.h>
# include <stdlib.h>
# include <stdio.h>
# include <string.h>
# include "config.h"
# include "lcd.h"
void setup(void);
void loading(void);
# define idle "Loading"
int access_count = 0;
char myStg[10];
int enter_count = 0;
char myStr[10];
int timer;
char mycount[10];
void lcd_info(){

    Lcd_Set_Cursor(1, 1);
    Lcd_Write_String("Enter: ");
    Lcd_Set_Cursor(1, 7);
    sprintf(myStr, "%d", enter_count);
    Lcd_Write_String(myStr);
    Lcd_Set_Cursor(2, 1);
    Lcd_Write_String("Access: ");
    Lcd_Set_Cursor(2, 8);
    sprintf(myStg, "%d", access_count);
    Lcd_Write_String(myStg);
    __delay_ms(500);
}
void count(){
    enter_count++;
    Lcd_Set_Cursor(1, 7);
    sprintf(myStr, "%d", enter_count);
    Lcd_Write_String(myStr);
    return;
}
```

Main class part 2

```
void timerdown(){

    while (1){
        if (INT1IF == 1){
        break;
        }
        Lcd_Set_Cursor(1, 9);
        Lcd_Write_String(" Counter");
        __delay_ms(1000);
        Lcd_Set_Cursor(2, 12);
        sprintf(mycount, "%d", timer);
        Lcd_Write_String(mycount);
        if (timer == 9){
            Lcd_Set_Cursor(2, 13);
            Lcd_Write_String(" ");
        }
        --timer;
        if (timer == 0){
            break;
        }
    }
    if (INT1IF == 0){
        __delay_ms(1000);
        Lcd_Set_Cursor(1, 9);
        Lcd_Write_String("        ");
        Lcd_Set_Cursor(2, 12);
        Lcd_Write_String("   ");
    }
    return;
}

void alarm(){
    Lcd_Clear();
    while (INT1IF == 0){
        __delay_ms(300);
        LATC2 = !LATC2;
        LATC5 =  !LATC5;
        Lcd_Set_Cursor(1, 1);
        Lcd_Write_String("Please sanitize");
        Lcd_Set_Cursor(2, 1);
        Lcd_Write_String("   your hands");

    }
    return;
}
```

Main class part 3

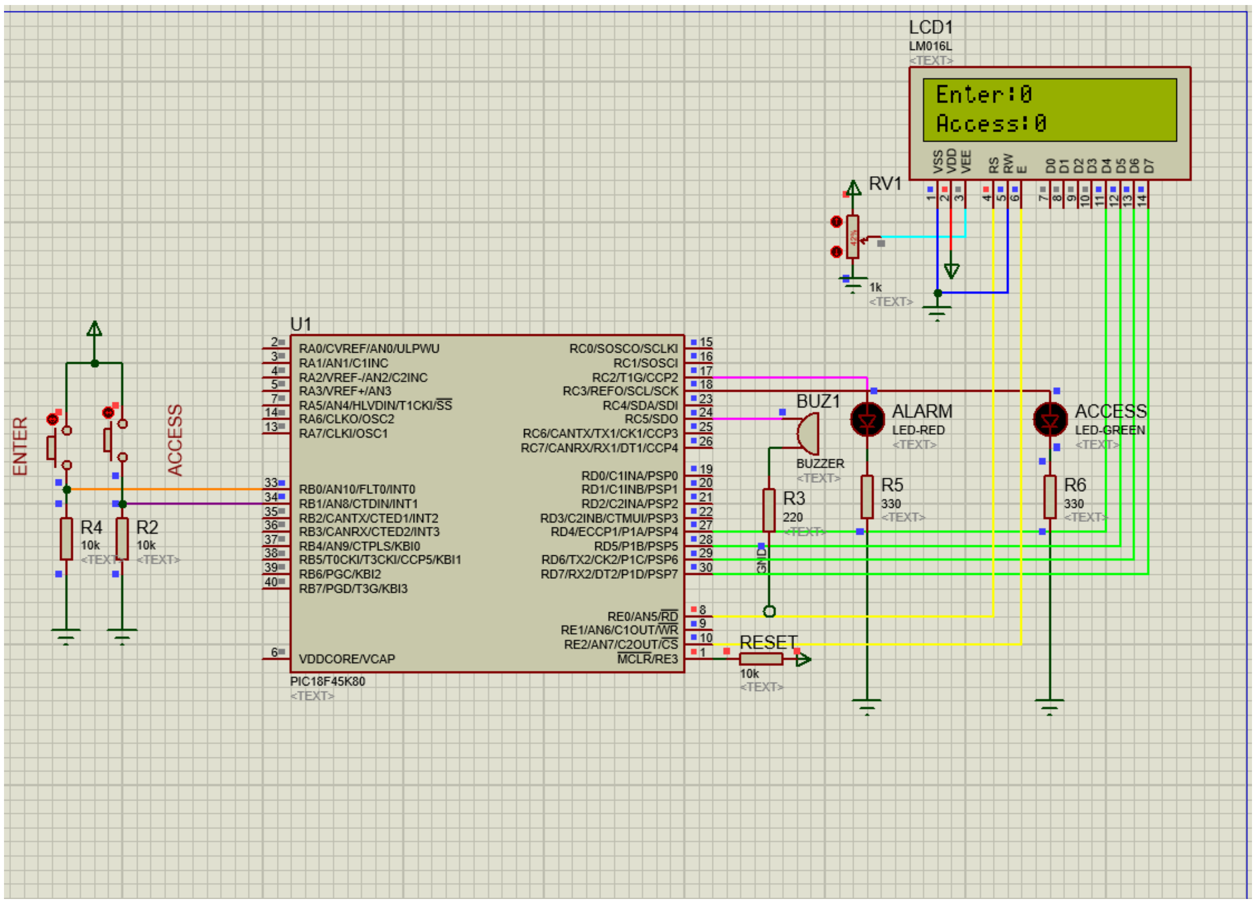
```
void access(){
    LATC2 = 0; // RED
    Lcd_Clear();
    Lcd_Set_Cursor(1, 1);
    Lcd_Write_String("======Done======");
    unsigned int L = 4;
    while (L > 0){
        __delay_ms(80);
        LATC3 = !LATC3; // GREEN
        LATC5 = !LATC5; // BUZ
        L--;
    }
    if (access_count < enter_count){
        access_count++;
    }
    LATC3 = 0; // GREEN
    LATC5 = 0;

    return;
}

void__interrupt() intrr(){
    if (INT0IF & & INT0IE){
        timer = 10;
        count();
        timerdown();
        if (LATB1 == 0){
            alarm();
        }
    INT0IF = 0;
    }
    if (INT1IF & & INT1IE){
        access();
        __delay_ms(500);
        Lcd_Clear();
        lcd_info();

        INT1IF =0;
    }
    return;
}

void main(void){
    loading();
    setup();
    Lcd_Clear();
    while (1){

        lcd_info();

    }

    return;
}
```

Main class part 4

```
void setup(void){
    LATC = 0;
    ANCON0 = 0; // only digital pins
    ANCON1 = 0;
    OSCCON = 0x60; // 8 MHz
    // initializeports I / O

    TRISC = 0;
    TRISB0 = 1;
    TRISB1 = 1;
    // enable interrupts
    INT0IF = 0;
    INT0IE = 1;
    INT1IF = 0;
    INT1IE = 1;
    GIE = 1;

}

// LCD configuration
void loading(){
    TRISD = 0;
    TRISE = 0;
    Lcd_Init();
    Lcd_Set_Cursor(1, 1);
    Lcd_Write_String(idle);
    unsigned i = 5;
    while (i > 0){
        Lcd_Write_String(".");
        __delay_ms(500);
        i--;
    }
    Lcd_Clear();
}
```

Appendix iii

Simulation


LCD1
LM016L
Enter:0
Access:0
RV1
1k
U1
PIC18F45K80
ENTER
ACCESS
R4
10k
R2
10k
BUZ1
BUZZER
R3
220
ALARM
LED-RED
R5
330
ACCESS
LED-GREEN
R6
330
RESET
10k