EE322: Embedded Systems Design - Project

Elevator System

Project Progress Report 3

20/08/2021

E/17/034: Chamandika H.W.T
E/17/211: Mihiranga M.V.R
E/17/279: Rasdeen R.S

Page Index

List of Figures

List of Tables

Elevator System
Progress from <30/7/2021> **to** <20/8/2021>

Overall percentage progress

0	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100

Introduction

In this project we are going to design and build a prototype of a horizontal elevator like cable car system in a city. Which mainly move along horizontal direction. From this we mainly focused on reducing traffic congestions in cities.

Brief of past progress (up to from date of this progress report)

As we mentioned in the progress report 2 the project was divided into 3 main parts and simulations were implemented. According to the part we became familiar with the Hardware specifications of each component, microcontroller features and assembly language. The simulations were implemented using proteus simulation software and mplab X IDE was used to program the microcontroller with assembly language.

Simulation of Unipolar Stepper Motor

For the simulation a unipolar stepper motor was used because it is the most likely to be used in the hardware implementation of the project. It will provide sufficient torque to the load but not as like a bipolar stepper motor. Therefore, the ULN2003A motor driver was used which is consist of transistor array.

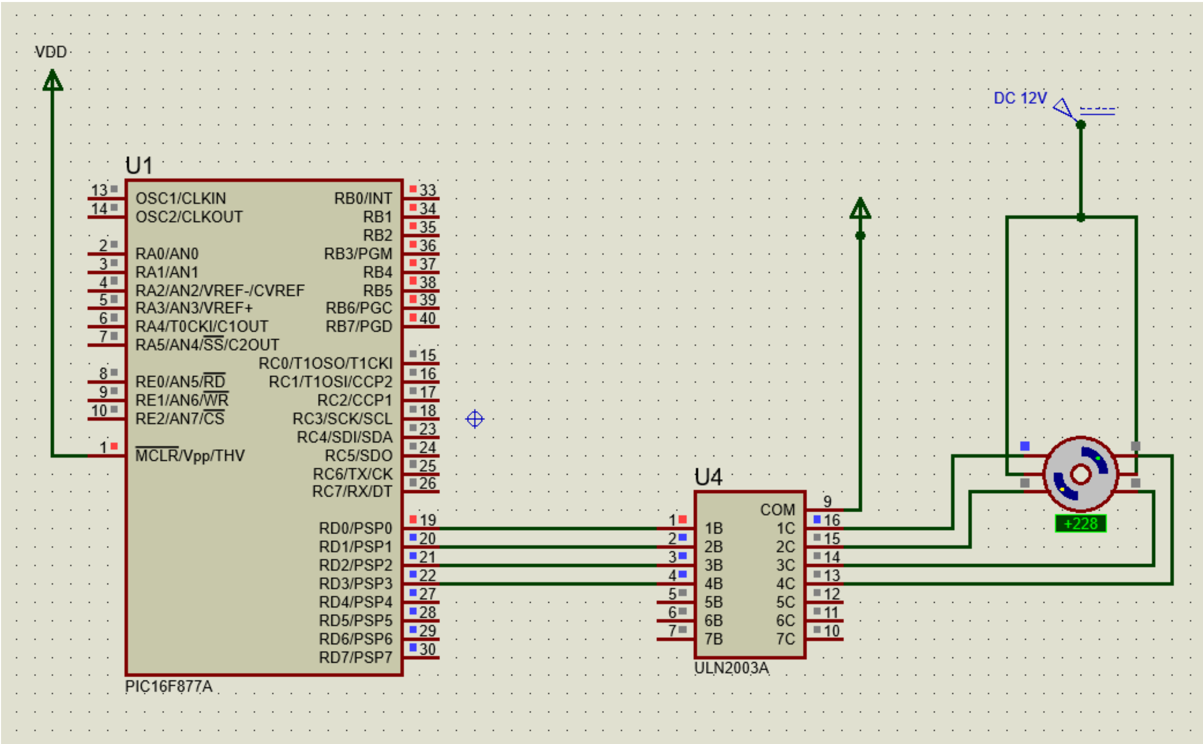

Figure 1: Circuit diagram in proteus

For the motor a 12V power supply was used. On the basis of the way the coils are energized a unipolar stepper motor can be classified into three categories.

1. Wave drive Mode
2. Full drive mode
3. Half drive mode

In wave drive mode only one coil is energized at a time and each coil is energized one after another in sequence. In this mode the least torque is produced compared to others but this is the highest power saving mode.

Steps	A	B	C	D
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

Table 1: Stator coils energizing sequence in wave drive mode

‘1’ refers to HIGH and ‘0’ refers to LOW states. ABCD are stator coils.

For speed control a delay was used. The delay subroutine was implemented using an 8-bit timer0 module in pic16f877a microcontroller. For this simulation 100ms delay was used.

In Full drive mode two coils are energized at a time this results in highest torque in 3 categories and also the power consumption.

Steps	A	B	C	D
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1
4	1	0	0	1

Table 2: Stator coils energizing sequence in full drive mode

In half drive mode at one moment only one coil is energized but in the next step two coils are energized and again back to one coil in next step. This sequence is repeated. This mode is used to gain high torque while power efficient.

Steps	A	B	C	D
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

Table 3: Stator coils energizing sequence in Half drive mode

All these 3 modes were simulated and tested using proteus simulation software using assembly language.

```
list p=16F877A
#include "p16f877a.inc"

     org              0x00

  ; setup ports

 bcf            STATUS,5        ; clear status reg
 bsf            STATUS,5
 movlw          b'00000111'
 movwf          OPTION_REG


 movlw     b'00000000'         ; move 0 to w reg
 movwf          TRISD     ; set port D as output


 bcf       STATUS,5       ; select memory bank 0



FULL_STEP_MOTOR_CONTROL_CCW

 movlw          b'00000011'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000110'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00001100'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00001001'
 movwf          PORTD
 call           TIMER_DELAY
 goto      FULL_STEP_MOTOR_CONTROL_CCW



FULL_STEP_MOTOR_CONTROL_CW

 movlw          b'00001100'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000110'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000011'
 movwf          PORTD
 call           TIMER_DELAY
```

```
 movlw          b'00001001'
 movwf          PORTD
 call           TIMER_DELAY
 goto      FULL_STEP_MOTOR_CONTROL_CW



HALF_STEP_MOTOR_CONTROL_CCW

 movlw          b'00000001'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000011'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000010'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000110'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000100'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00001100'
 movwf          PORTD
 call      TIMER_DELAY
 movlw          b'00001000'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00001001'
 movwf          PORTD
 call           TIMER_DELAY
 goto      HALF_STEP_MOTOR_CONTROL_CCW


HALF_STEP_MOTOR_CONTROL_CW

 movlw          b'00001000'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00001100'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000100'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000110'
 movwf          PORTD
 call           TIMER_DELAY
 movlw          b'00000010'
```

```
movwf          PORTD
call           TIMER_DELAY
movlw          b'00000011'
movwf          PORTD
call           TIMER_DELAY
movlw          b'00000001'
movwf          PORTD
call           TIMER_DELAY
movlw          b'00001001'
movwf          PORTD
call           TIMER_DELAY
goto      HALF_STEP_MOTOR_CONTROL_CW



WAVE_DRIVE_MOTOR_CONTROL_CCW

movlw          b'00000001'
movwf          PORTD
call           TIMER_DELAY
movlw          b'00000010'
movwf          PORTD
call           TIMER_DELAY
movlw          b'00000100'
movwf          PORTD
call           TIMER_DELAY
movlw          b'00001000'
movwf          PORTD
call           TIMER_DELAY
goto      WAVE_DRIVE_MOTOR_CONTROL_CCW

WAVE_DRIVE_MOTOR_CONTROL_CW

movlw          b'00001000'
movwf          PORTD
call           TIMER_DELAY
movlw          b'00000100'
movwf          PORTD
call           TIMER_DELAY
movlw          b'00000010'
movwf          PORTD
call           TIMER_DELAY
movlw          b'00000001'
movwf          PORTD
call           TIMER_DELAY
goto      WAVE_DRIVE_MOTOR_CONTROL_CW
```

```
TIMER_DELAY

    movlw        d'8'
    movwf        0x0C
L2  movlw        d'11'
    movwf        TMR0
    bcf               INTCON,2
L1  btfss        INTCON,2
    goto              L1
    decfsz       0x0C,1
    goto              L2
return

end
```

Simulation for The IR Sensor

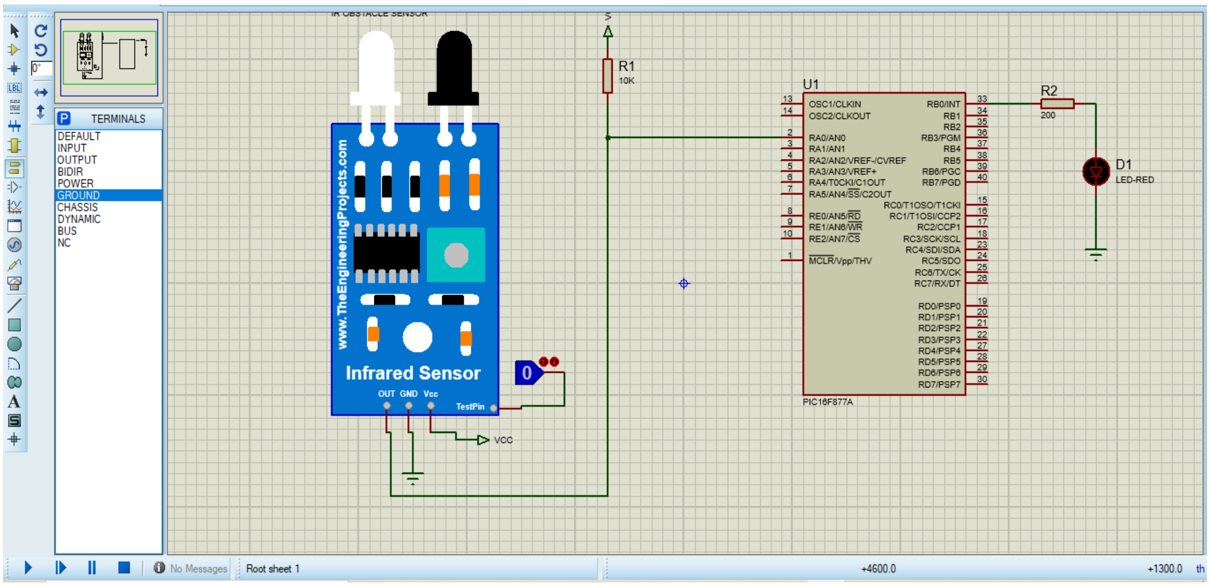

Figure 2: Circuit diagram in proteus

When making the schematic diagram IR sensor has to be imported to the proteus simulator. PIC16F877A is used as the microcontroller as that is the one we choose for our project. The LED is connected to the output port to see the output.

```
list p=16F877A
#include "p16f877a.inc"

org 0

main

; setting portB as the output port
    bsf STATUS,5
    clrf TRISB
    movlw b'00000001'
    movwf TRISA
    bcf STATUS,5

start

    clrf   PORTB
    btfss  PORTA,0
    goto   start
    bsf    PORTB,0
    goto   start

end
```

In the assembly code, IR sensor output data which is connected to PORTA0 is taken as input for the microcontroller and according to that value the LED which is connected to output PORTB0 should be on or off.

Simulation of LCD Display

We were planned to use an LCD display to some details to passengers who used to travel using our cable car system. Stopped station, status of the door (open / close), bill for the trip are the examples for the details we were planned to display.

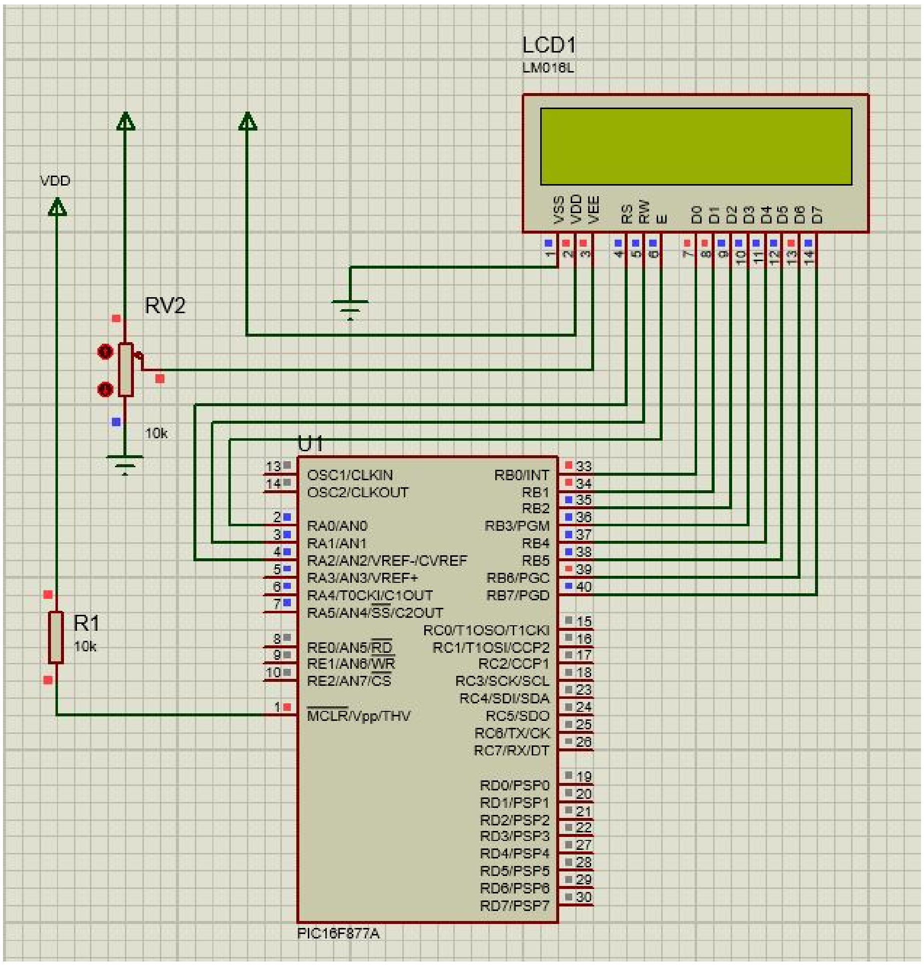

Figure 3: Circuit diagram of LCD display

The code is following shows the assembly code for the LCD display

Here when coding for LCD display first 2line mode and 5*8 dot method was selected under Function set. For that we have to set value zero to the E, R/W, and RS pins. Also, we can choose 1line mode and 5*11 dot method by giving value (0 or 1) to DB3 and DB2 bits of the LCD.

Likewise, we have to set display controls, clear the display, set entry mode (which side we need to display upcoming letter, left to right or right to left). here we select right to left.

The usage of DRAM ADDRESS is to select the starting point.
Then we should be able to write a message that we need to display on the LCD screen.

We hope to use a lookup table because cable car motion is linear and continuous and it will be easier.

```
list p=16F877A
#include "p16f877a.inc"


COUNTER1 EQU 20H         ; compiler directives for prepare a
delay
COUNTER2 EQU 21H

    ORG 0

    ;====INITIALIZATION====
    BSF      STATUS,5
    BCF      STATUS,6    ; select bank 1

    CLRF     TRISA       ; port A output
    CLRF     TRISB       ; port B output

    BCF      STATUS,5    ; back to bank 0
    CALL     DELAY1

    ;====FUNCTION SET====
    CLRF     PORTA       ; (E = 0 , R/W = 0 , RS = 0)
    MOVLW    b'00111000' ; set 2line mode
    MOVWF    PORTB
    CALL     PULSE

    ;====DISPLAY CONTROL====
    MOVLW    b'00001100';set display on / cursor off / blink
off
    MOVWF    PORTB
    CALL     PULSE

    ;====CLEAR DISPLAY====
    MOVLW    b'00000001' ; clear the display
    MOVWF    PORTB
    CALL     PULSE
```

```
    CALL     DELAY2

    ;====SET ENTRY MODE====
    MOVLW    b'00000110' ; set the increment mode
    MOVWF    PORTB
    CALL     PULSE

    ;====SET DRAM ADDRESS====
    MOVLW    b'10000000'
    MOVWF    PORTB
    CALL     PULSE

    ;====DISPLAY ON LCD====
    BSF      PORTA,2       ; RS = 1
    MOVLW    'C'            ; write a letter in W Reg.
    MOVWF    PORTB         ; write the letter from W to F
    CALL     PULSE

LOOP GOTO LOOP

    MOVLW    b'11111111'
    MOVWF    COUNTER1      ; given 255 on COUNTER1

DELAY1
    DECFSZ   COUNTER1,1
    GOTO     DELAY1
    RETURN

MOVLW    b'11111111'
MOVWF    COUNTER1          ; given 255 on COUNTER1
MOVWF    COUNTER2          ; given 255 on COUNTER2

DELAY2
    DECFSZ   COUNTER1,1
    GOTO     DELAY2
    DECFSZ   COUNTER2,1
    GOTO     DELAY2
    RETURN

    ;====ENABLE PULSE====
PULSE
    BSF      PORTA,0       ; E = 1
    CALL     DELAY1
    BCF      PORTA,0       ; E = 0
    CALL     DELAY1
    RETURN

    END
```

Progress for the period from <30/7/2021> to <20/8/2021>

16/06/2021 - Decided the project title as Elevator system.

20/06/2021 - Discussed how to design a useful elevator system as normal elevators are common in Sri lanka.

23/06/2021 - Decided to build a cable car system using the theories of an elevator system as it will be a new thing to Sri lanka. And also, it gives a good solution to the traffic problem.

24/06/2021 - Discussed how to implement the system. Mainly focused on the simulation side.

25/06/2021 - Submitted the project proposal

27/06/2021 - Drew the UML user case diagram for the system

30/06/2021 - Discussed about the components that we are going to need. Looked for suitable components to order.

06/07/2021 - Drew the UML class diagram for the system

08/07/2021 - Ordered some components.

13/07/2021 - Drew the sequence diagram for the system

15/07/2021 - Discussed about how we are going to implement the system.

21/07/2021 - Draw the state machine diagram for the system.

25/07/2021 - Discussed about the process and assigned each member to look into some particular part of the project.

26/07/2021 - Ordered some components.

28/07/2021 - Studied about how to program this system using assembly

01/08/2021 - 15/08/2021 - Studied our own simulation parts

05/08/2021 - Ordered some components

20/08/2021 - Discussed the problems that we faced

Cost Analysis

Task	**Budgeted cost (Rs)**	**Expenses up to** <16/6/2021>	**Expenses** <16/6/2021> **to** <30/7/2021>	**comments**
Microcontroller and pickit 3	3150.00	-	3750.00	pickit 3 cost is not included in the project proposal
12 MHz Crystal Oscillator	15.00 × 2	-	30.00	
20 MHz Crystal Oscillator THT	15.00 × 2	-	30.00	
16V Electrolytic Capacitor THT	7.00 × 2	-	14.00	
LCD Display (16×2)	320.00 × 1	-	320.00	
IR sensor	110.00 × 3	-	330.00	
Breadboard	300.00 x 2		600.00	
Resistors and Jumper wires, push button, buzzer	100.00		100.00	
Unipolar stepper motor and controller	350.00		300.00	

Table 4: Cost Analysis

Timeline (Gantt Chart)

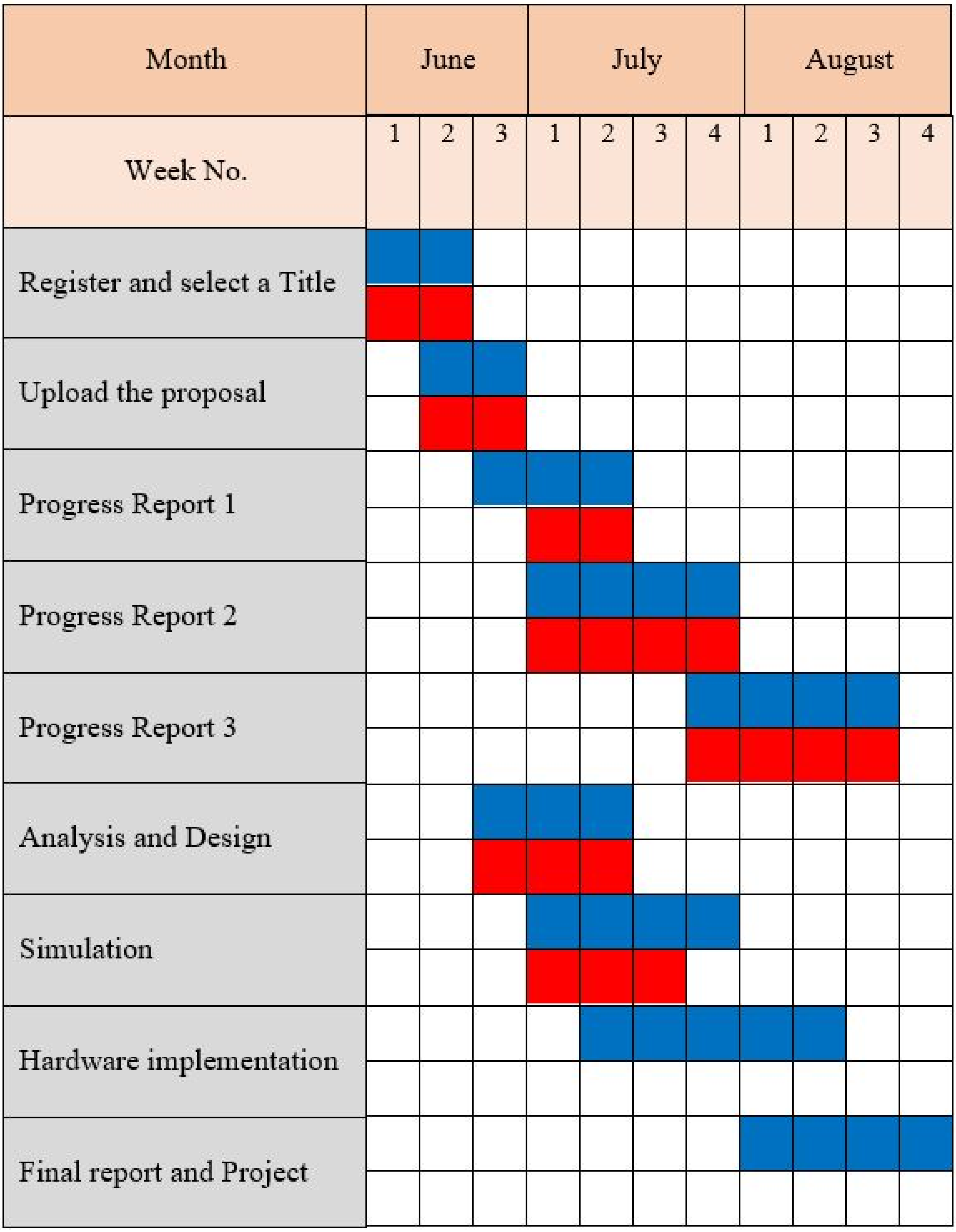

Figure 4: timeline