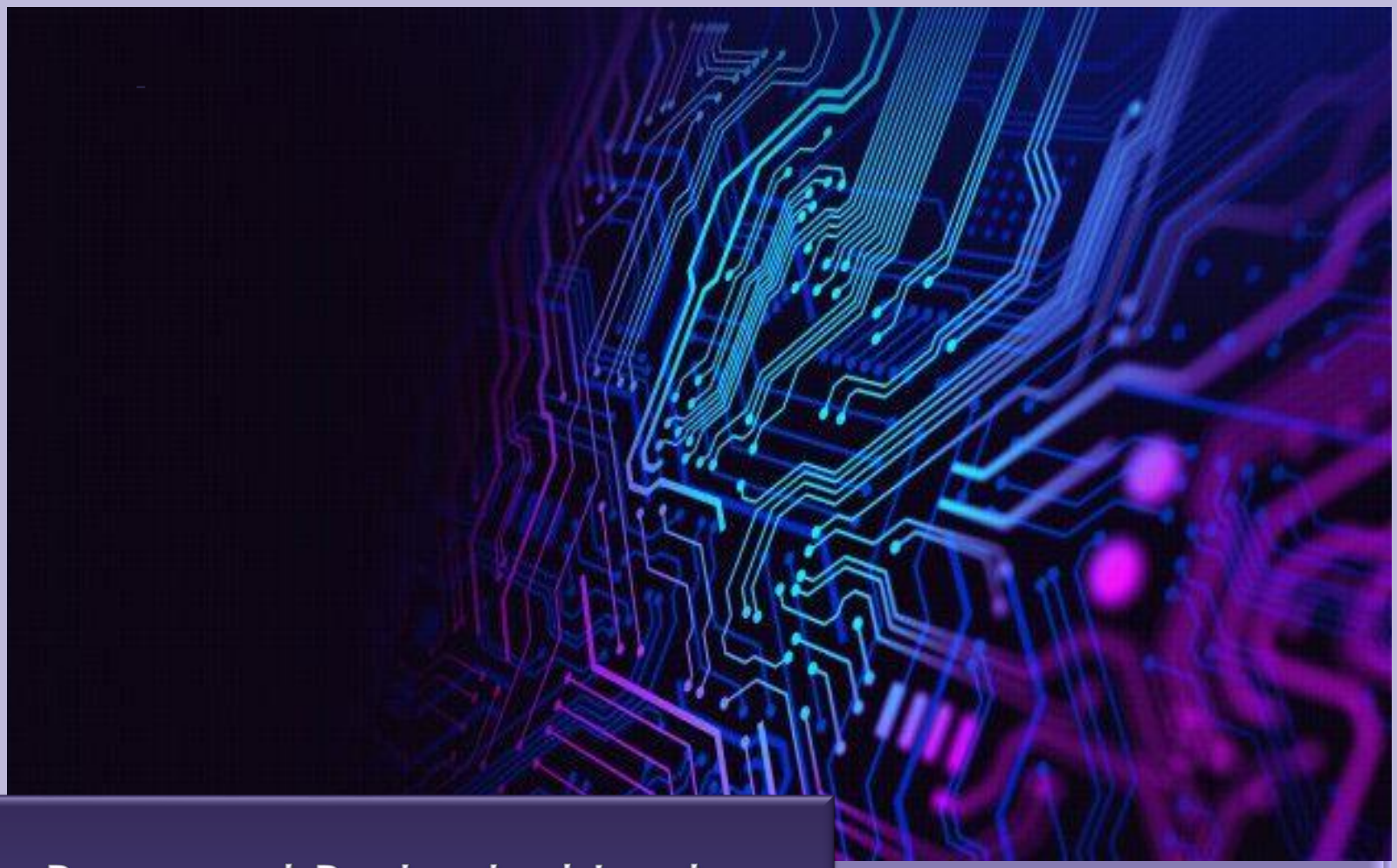


Password Protected Locker

M2M Communication

Group Members:

Javeria Adil (CS-19301)

Zobia Khan (CS-19063)

Wajiha Khan (CS-19064)

Uzma Aijaz (CS-19052)

Muqadas Ashraf (CS-19053)

Khadija (CS-19075)

Umm e Hani (CS-19054)

Ms. Syeda Ramish Fatima

TABLE OF CONTENTS

ABSTRACT

Security is a major focus these days for secure communication. For secure machine-to-machine (M2M) communication, a microcontroller is implemented which operates a password-protected locker for the safe transfer of messages.

Password Protected Locker with machine-to-machine communication is incorporated in the microcontroller design where a secure password will act as a door unlocking system.

DESIGN PHILOSOPHY:

A microcontroller-based application is made in which a password is required to provide through a keypad for the successful transfer of messages from the Transmitter to the Receiver. The Receiver microcontroller matches the received password from Transmitter with its original password, a servo-motor is placed in the interface of the receiver microcontroller to latch or unlatch the locker door through the dip switch/push button. So when the password is matched servo-motor opens the locker door and else locked.

We have created a function named **GetKeyPressed()**, this function returns an index value of keypad to main function, if the key is pressed. In main function we check if index is returned or not, if it is returned then we check returned index data to our password character and increment a pointer for next index of password if it matches.

We run our loop 4 times as our password's maximum length is 4. After entering 4 digits the password will be checked at receiver end if password matches we display O to seven-segment otherwise it display C. Displaying O means door in unlocked and C means door is locked and receiver entered wrong password. In coding we use USART1 for both receiver and transmitter and 4x4 keypad for entering password.

HARDWARE COMPONENTS:

The description of the hardware components is a follows:

1. **Microcontroller:**

   An AVR AT Mega 162 microcontroller is used which is an 8-bit microcontroller. It has an advanced RISC architecture and ensure high performance and low power design.

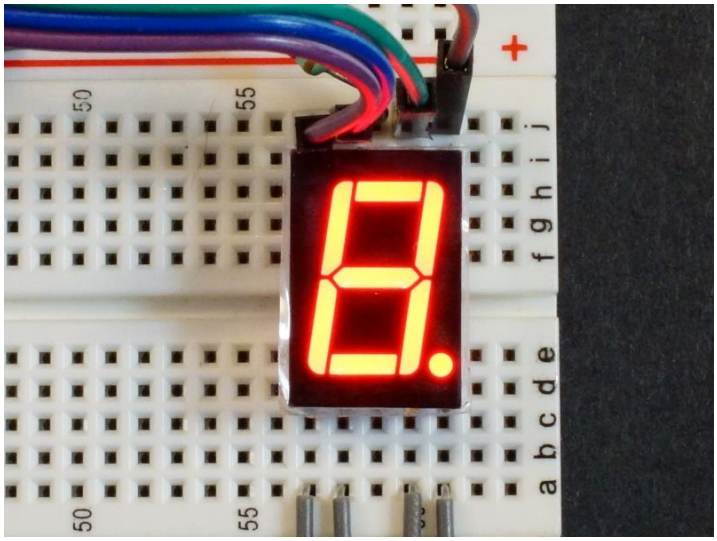
   Two microcontrollers are used for the given project. One acts as a Transmitter microcontroller which is linked with a locker. The other is the Receiver microcontroller which matches the password and latch or unlatch the locker door through a servo-motor.

2. Servo Motor:

Servo motors are self-contained mechanical devices that are used to control the machines with great precision. It is attached with receiver microcontroller.

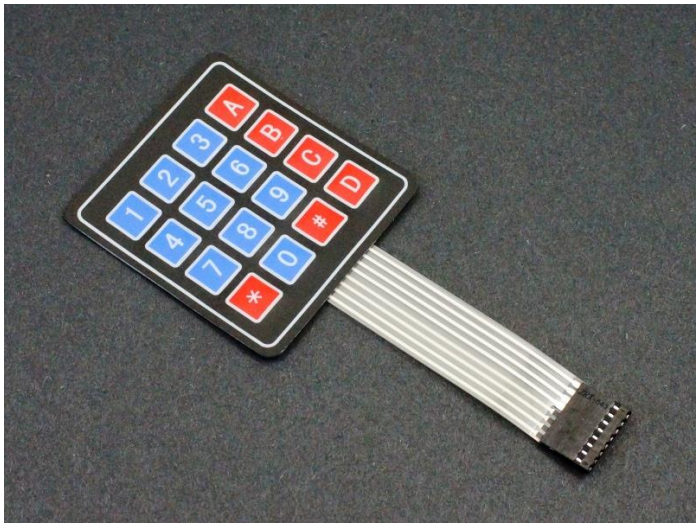
3. Seven Segment Display:

A seven-segment display is a form of electronic display device for displaying decimal numerals that is an alternative to the more complex dot matrix displays. It consists of seven LEDs (hence its name) arranged in a rectangular fashion. Each of the seven LEDs is called a segment because when illuminated the segment forms part of a numerical digit (both Decimal and Hex) to be displayed.



4. Key Pad:

A matrix keypad consists of arrangement of switches in matrix format in rows and columns with the microcontroller I/O pins connected to the rows and columns of the matrix such that switches in each row are connected to one pin and switches in each column are connected to another pin.



CODE OF TRANSMITTER:

```c
/*
 * try_trans.c
 *
 * Created: 7/27/2022 10:39:21 PM
 * Author : Administrator
 */

#include <avr/io.h>
#define F_CPU 1000000UL
#include <util/delay.h>
#include <string.h>


int GetKeyPressed(void);
int main(void)

{
    int key;
    char digit[20]={'7', '8', '9', '/', '4', '5', '6', '*', '1', '2', '3','-', 'A', '0', '=','+','\0'};
    DDRB|=1<<PB3;
    DDRB&=~(1<<PB2);
    PORTB|=1<<PB2;
```

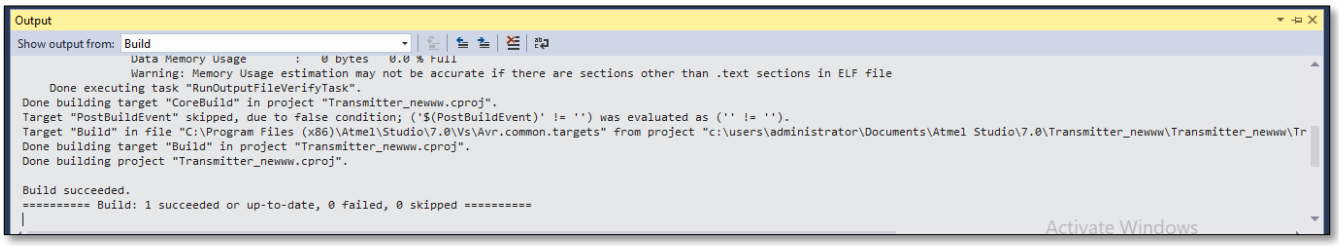
```c
    DDRC=0x00;
    DDRD=0xF0;
    DDRB=(1<<PINB1);
    DDRA=0xFF;

    UBRR1H=0x00;
    UBRR1L=0x05;
    UCSR1B =(1<<TXEN1);
    /*UCSR1B |=(1<<RXEN1);*/
    UCSR1C =(1<<USBS1)|(3<<UCSZ10);


    int i=0;
    char password[4]={};

    int length_of_array;
    unsigned char receiveData;
    /* Replace with your application code */
    while (1)

    {/*key = GetKeyPressed();
        if(key!=16){
            while(!(UCSR1A&(1<<UDRE1)));
            UDR1=digit[key];*/
```

```c
    if(key!=16){
        while(!(UCSR1A&(1<<UDRE1)));
        UDR1=digit[key];*/

    key = GetKeyPressed();
    if(key!=16){
            while(!(UCSR1A&(1<<UDRE1)));
            UDR1=digit[key];

            i=i+1;
            /*password[i]+=digit[key];
            i=i+1;
            }*/

    }

    /* length_of_array=(int)( sizeof(password) / sizeof(password[0]) );
     if(length_of_array==4){
            for (i = 1; i < 5;i++)
            {
               while(!(UCSR1A&(1<<UDRE1)));
               UDR1=password[i];
```

```c
                UDR1=password[i];



                _delay_ms(1000);}

           }*/
        if (i==1){
        UCSR1B =(1<<RXEN1);

        PORTC=0;
        receiveData=0;
        while(!(UCSR1A&(1<<RXC1)));
        receiveData=UDR1;
        PORTC=receiveData;
        if(receiveData==0b00000001){
            PORTA=0b11000000;
            _delay_ms(1000); }
        else if(receiveData!=0b00000001){
            PORTA=0b11000110;
            _delay_ms(1000); }}
```

```c
        }
    }
    int GetKeyPressed(void){
        char x;
        PORTB=0x00;
        int data;
        x=PINC;
        if(x==0x01)
        {
            data=(PIND&0x0F);

            return data;
        }
        return 16;
    }
```

OUTPUT:



CODE OF THE RECEIVER:

```
*
* try_rec.c
*
* Created: 7/27/2022 10:44:42 PM
* Author : Administrator
*/

include <avr/io.h>
define F_CPU 1000000UL
include <util/delay.h>
include <string.h>


nt main(void)


   DDRA=0x01;

   DDRB|=1<<PB3;
   DDRB&=~(1<<PB2);

   DDRC=0xFF;
   int j;
```

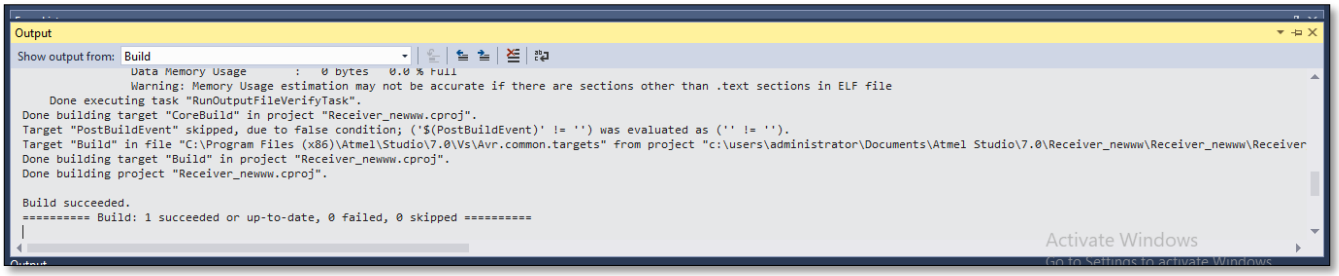
```
    DDRC=0xFF;
    int j;

    //DDRC=0x00;//
    //DDRB=(1<<PINB1);//
    UBRR1H=0x00;
    UBRR1L=0x05;

    UCSR1B =(1<<RXEN1);
    UCSR1C =(1<<USBS1)|(3<<UCSZ10);
    unsigned char receiveData;
    int index;
    char Original_Password='4';

    char push_button;

    /* Replace with your application code */
    while (1){

        PORTC=0;
        receiveData=0;
        while(!(UCSR1A&(1<<RXC1)));
        receiveData=UDR1;
```

```
    /* Replace with your application code */
    while (1){

        PORTC=0;
        receiveData=0;
        while(!(UCSR1A&(1<<RXC1)));
        receiveData=UDR1;

        PORTC=receiveData;
        if (receiveData==Original_Password){
        //Rotate motor to 0'//
        PORTA=0x01;
        _delay_ms(10);
        PORTA=0x00;
        _delay_ms(10);

        //Rotate to 90//
        PORTA=0x01;
        _delay_ms(100);
        PORTA=0x00;
        _delay_ms(100);

        PORTB|=1<<PB2;
```

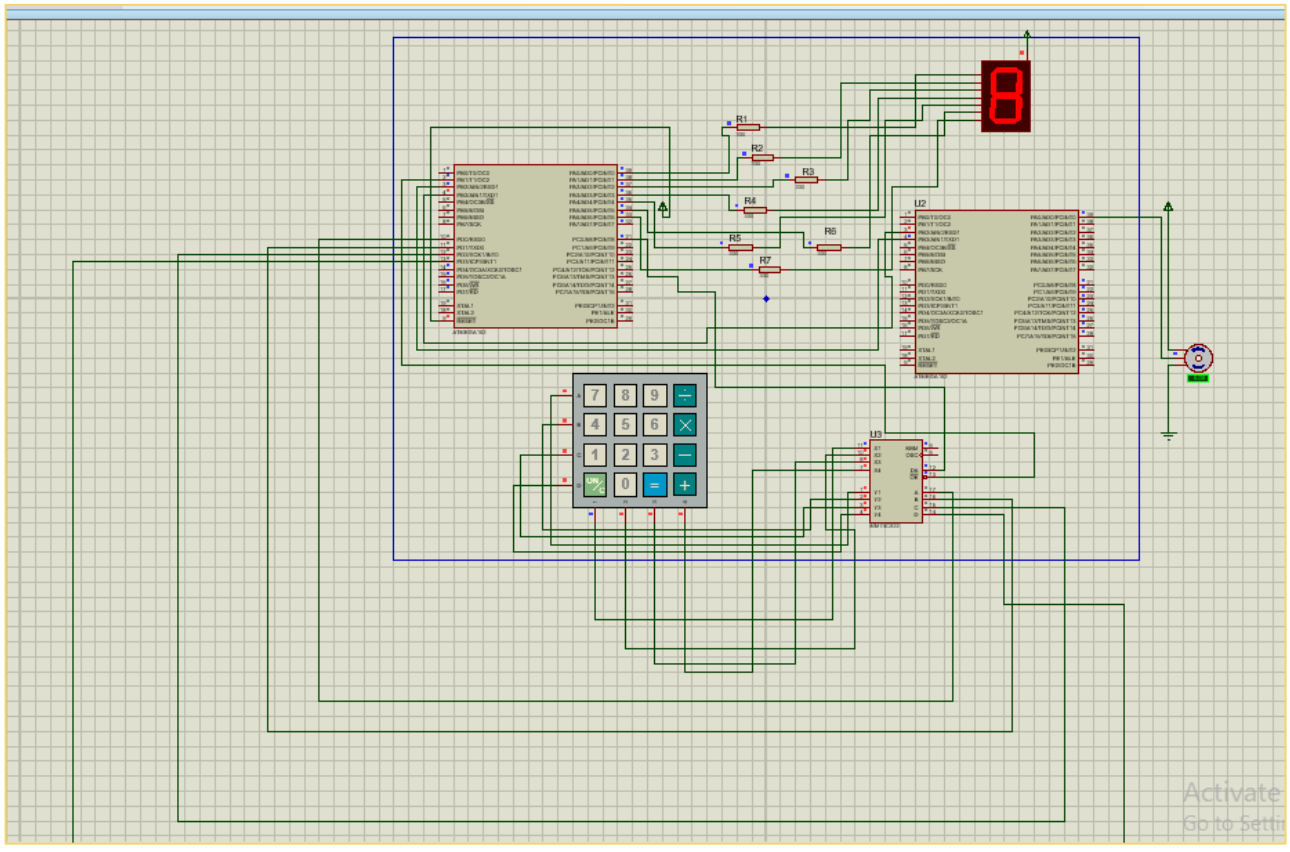
```
    _delay_ms(10);

    //Rotate to 90//
    PORTA=0x01;
    _delay_ms(100);
    PORTA=0x00;
    _delay_ms(100);

    PORTB|=1<<PB2;
    UCSR1B =(1<<TXEN1);
    /*UCSR1B |=(1<<RXEN1);*/


    while(!(UCSR1A&(1<<UDRE1)));
    UDR1=0b00000001;
    /*PORTD=0b01000000;
    _delay_ms(1000);*/
}
else if(receiveData!=Original_Password){
    PORTB|=1<<PB2;
    UCSR1B =(1<<TXEN1);
    /*UCSR1B |=(1<<RXEN1);*/
```

```
    else if(receiveData!=Original_Password){
        PORTB|=1<<PB2;
        UCSR1B =(1<<TXEN1);
        /*UCSR1B |=(1<<RXEN1);*/


        while(!(UCSR1A&(1<<UDRE1)));
        UDR1=0b00000010;
        PORTA=0x00;
        /*PORTD=0b11000110;
        _delay_ms(1000);*/
    };


    }
}
```

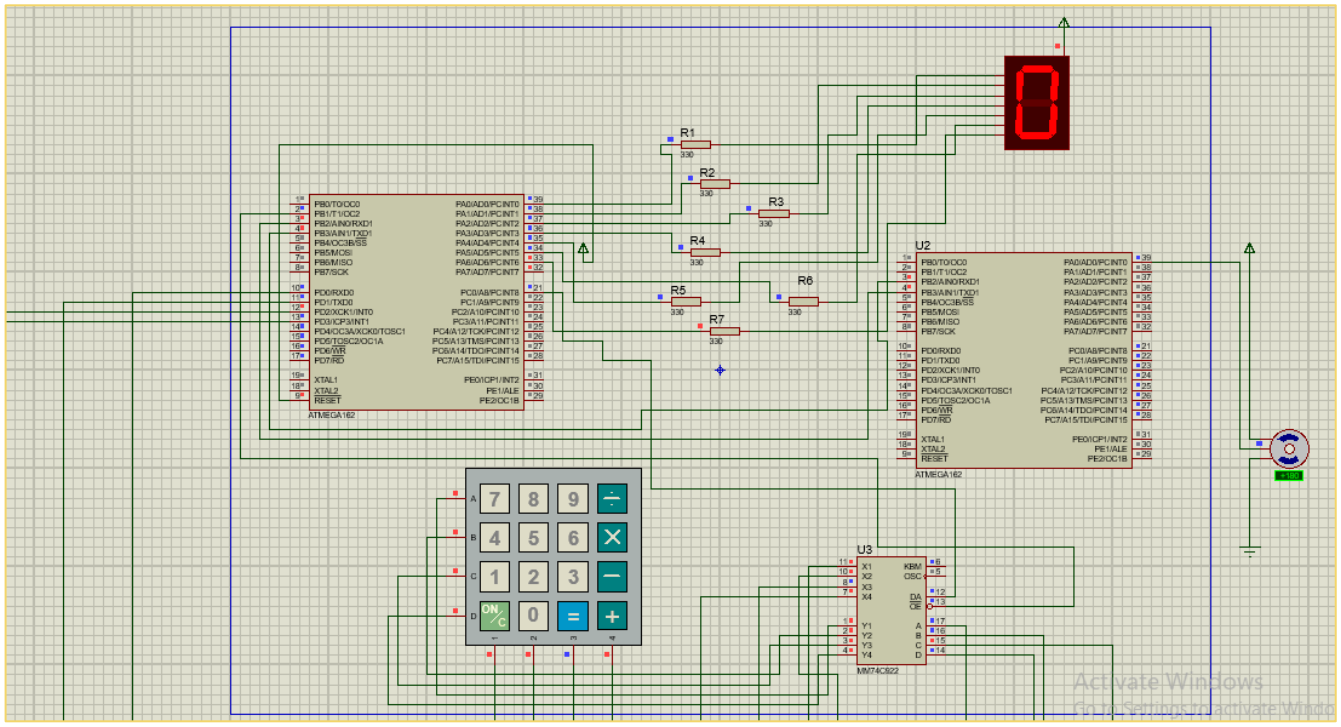
OUTPUT:



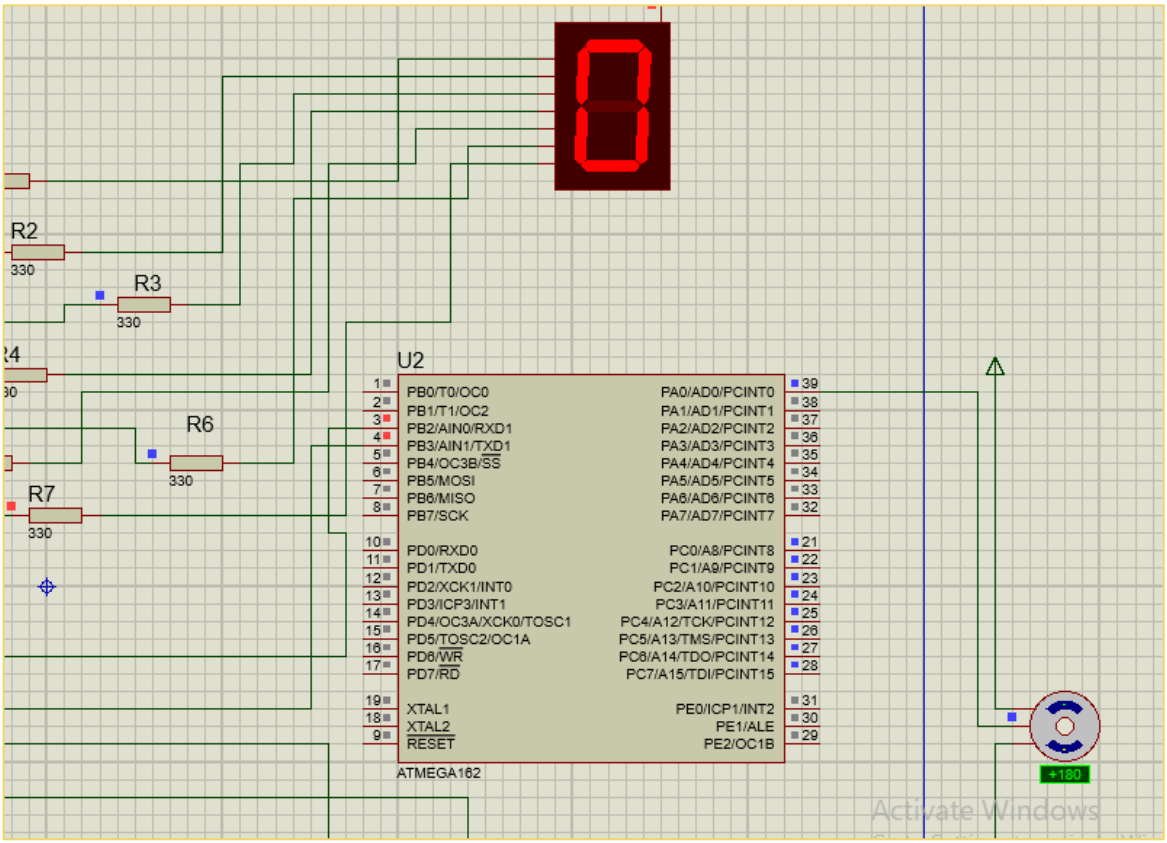
SIMULATOR:



When Password is correct:

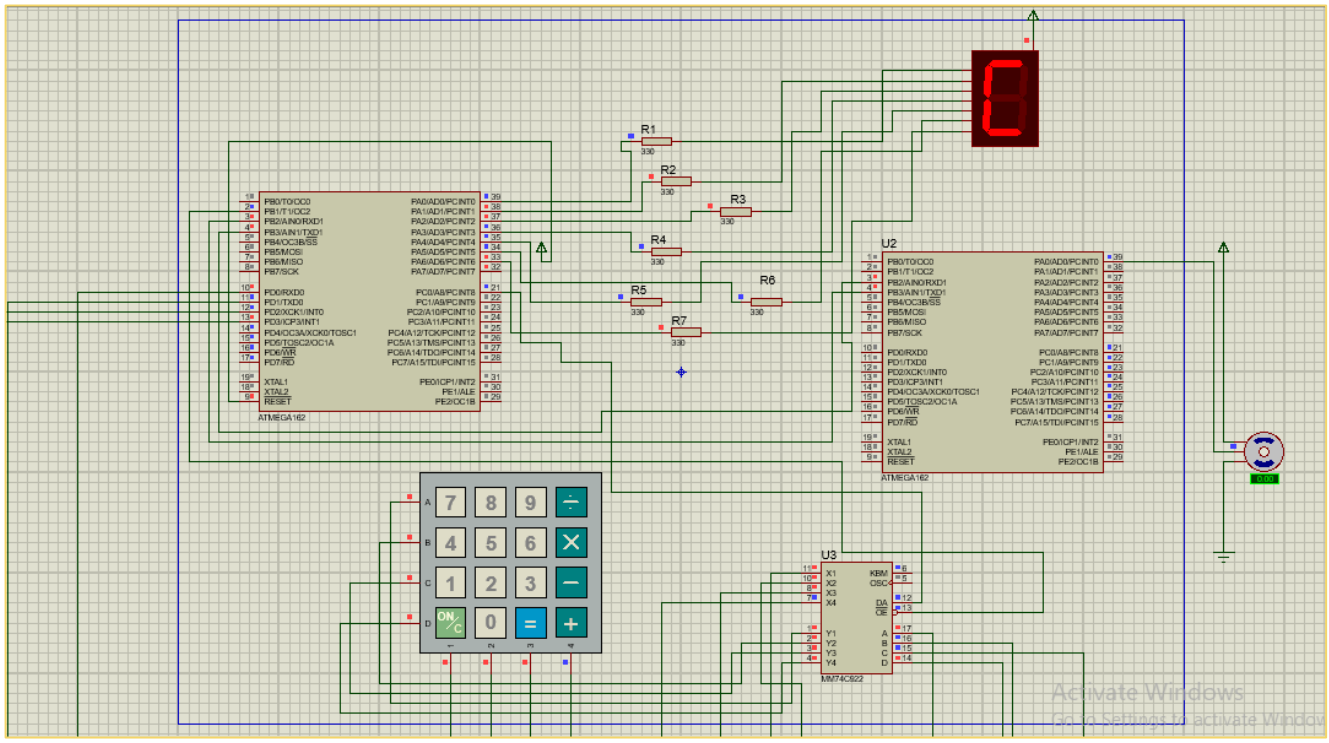


The 7-segment shows ‘O’, that represents the lock is unlatch, and servo motor moves to 180’.



When Password is correct:

The 7-segment shows ‘C’ as the door closed and the motor does not move, it remains in 0’.



Conclusion:

We are successfully solving the problem. And our systems work according to the problem statement. We have 1 digit or character password to unlock the door. If the password is correct the door is latched otherwise it remains closed.