2022

Smart Home Project

The Team

16- Esraa Samir Ezat Sadek

32- Nourhan Gamal Abdeen Mostafa Saad Elden

36- Reem Ahmed Mohmmed

Abstract

We aim to growth the field because it is the future industry, and we need to take the place in it. The project is a home that we control its vans, LEDs, and door. The door is opened by password, vans turn on by the current temperature, the LEDs by the current light, smart clock showing the time and the temperature degree.

Table of Contents

Introduction

Automatic systems are being preferred over manual system in our now-world. In our project -Smart Home- plays an important role for humans. So, we talk about basic needs to make the project well and for its future advancements.

First, we need sensors to check the temperature, light and we used

- LDR -light- sensor
- LM35 -temp- senso

Second, we need microcontroller to burn in it the code and control the system, so we used ATmega32.

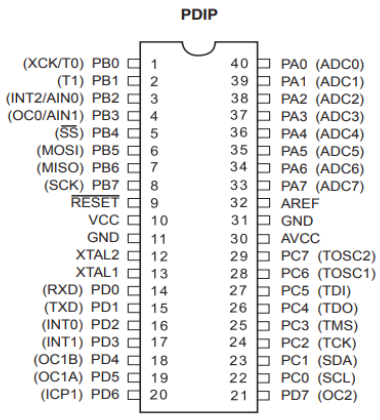
Third, we used another component to create the H.W implementation like:

- Capacitors
- Resistors
- LEDs
- Vans
- Servo and Stepper motors
- Keypad
- Lcd

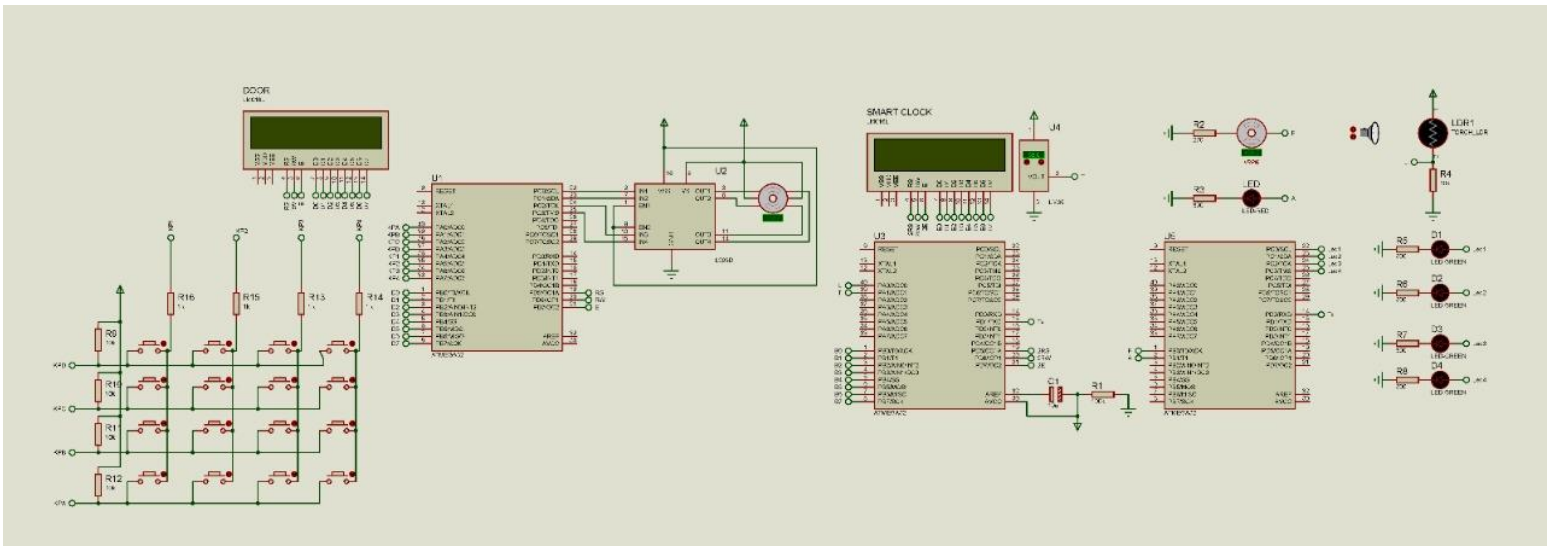
MICROCONTROLLER

small and low-cost microcomputer, which is designed to perform the specific tasks of embedded systems like displaying microwave's information, receiving remote signals, etc. And consists of the processor, the memory, Serial ports, peripherals (timers, counters), etc.

As we notice before we used ATmega32 in our project.

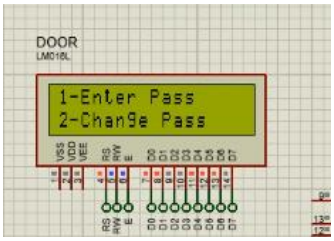


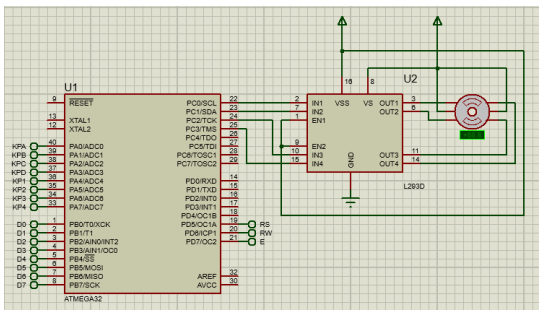
The Project's Design

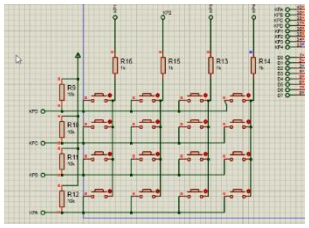


The Door

We designed door opened by write the password on **KEYPAD** attached by **LCD** that showing two options = "1-Enter Pass", "2-Change Pass".







Door circuit

The code:

This code to print on the lcd initial welcome to user **"Welcome Home"** and delay half second and clear then print **"chose a mode"** then clear and print two options **= "1-Enter Pass", "2-Change Pass".**

```
LCD_Init ();
Display_Str_Row_Col(0,2,"Welcome Home");
_delay_ms(500);
while(1)
{
    LCD_CLEAR();
    Display_Str_Row_Col(0,2,"Chose a Mode");
    _delay_ms(500);
    LCD_CLEAR();
    Display_Str("1-Enter Pass");
    Display_Str_Row_Col(1,0,"2-Change Pass");
    mode = Keypad_GetPressedKey();
```

```
// Code to get the Password From the User
if ( mode == 1)
{
    LCD_CLEAR();
    Display_Str("Password : ");
    for (u8 i = 0; i<4 ;i++)
    {
        Send_CMD(CURSOR_ON);
        _delay_ms(100);
        Send_CMD(CURSOR_OFF);

        key = Keypad_GetPressedKey();
        if ((key>=0) && (key <=9))
        {
            Integer_TO_String (key);
            _delay_ms(200);
            Display_Str_Row_Col(0,11+i,"*");
        }
        else
        {
            LCD_CLEAR();
            Display_Str_Row_Col(0,1,"Invalid Input");
            _delay_ms(500);
            break;
        }
        Comp_Pass += key;
        Comp_Pass = Comp_Pass * 10;
    }
    Comp_Pass = Comp_Pass / 10;
    _delay_ms(500);
    LCD_CLEAR();
```

Take the pass from the user

Test the pass if right and print * instead of the numbers as a safe

Print invalid input if the user writes invalid char // numbers only

Enter the pass as digits 1,10,100,1000

```
    if (Saved_Pass == Comp_Pass)
    {
        Display_Str("Welcome Back");
        PORTC = (PORTC & 0xF0) | (motor_steps[1] & 0x0F);  // 90
        _delay_ms(3000);
        PORTC = (PORTC & 0xF0) | (motor_steps[0] & 0x0F);  //   0
        LCD_CLEAR();
        Comp_Pass = 0;
    }
    else if (Reversed_Pass == Comp_Pass)
    {
        Display_Str_Row_Col(0,1,"Calling  911");
        _delay_ms(1000);
        LCD_CLEAR();
        Comp_Pass = 0;
    }
    else
    {
        Display_Str_Row_Col(0,1,"Wrong Password");
        _delay_ms(500);
        LCD_CLEAR();
        Comp_Pass = 0;
    }
}
```

Check if the user enters the right pass

If the user enters the inverse pass, call the

If the user enters wrong pass print it

```
// Code to Change the Password
else if (mode == 2)
{
    LCD_CLEAR();
    Display_Str("Old Pass : ");
    for (u8 i = 0; i<4 ;i++)
    {
        Send_CMD(CURSOR_ON);
        _delay_ms(200);
        Send_CMD(CURSOR_OFF);

        key = Keypad_GetPressedKey();
        if ((key>=0) && (key <=9))
        {
            Integer_TO_String (key);
            _delay_ms(200);
            Display_Str_Row_Col(0,11+i,"*");
        }
        else
        {
            LCD_CLEAR();
            Display_Str_Row_Col(0,1,"Invalid Input");
            _delay_ms(500);
            break;
        }
        Comp_Pass += key;
        Comp_Pass = Comp_Pass * 10;
    }
    Comp_Pass = Comp_Pass / 10;
    _delay_ms(500);
    LCD_CLEAR();
    if (Saved_Pass == Comp_Pass)
    {

        Comp_Pass = 0;
        Saved_Pass = 0;
        Reversed_Pass = 0;
        Get_Reversed_Pass = 0;
        Display_Str("New Pass : ");
        Display_Str_Row_Col(1,0,"Just Numbers");
        for (u8 i = 0; i<4 ;i++)
        {
            Send_CMD(CURSOR_ON);
            _delay_ms(200);
            Send_CMD(CURSOR_OFF);

            key = Keypad_GetPressedKey();
            if ((key>=0) && (key <=9))
            {
                Go_To_Row_Col(0,11+i);
                Integer_TO_String (key);
                _delay_ms(200);
                Display_Str_Row_Col(0,11+i,"*");
            }

            Saved_Pass += key;
            Saved_Pass = Saved_Pass * 10;
        }
        Saved_Pass = Saved_Pass / 10;
        Temp_Pass = Saved_Pass;
        while (Temp_Pass > 0)
        {
            Get_Reversed_Pass = Temp_Pass % 10;
            Reversed_Pass = (Reversed_Pass * 10) + Get_Reversed_Pass;
            Temp_Pass = Temp_Pass / 10;
        }
        _delay_ms(500);
        LCD_CLEAR();
        Display_Str("Password changed");
        _delay_ms(500);
        LCD_CLEAR();
    }
    else
    {
        Display_Str_Row_Col(0,1,"Wrong Password");
        _delay_ms(500);
        LCD_CLEAR();
        Comp_Pass = 0;
    }
}
else
{
    LCD_CLEAR();
    Display_Str_Row_Col(0,1,"Invalid Input");
    _delay_ms(500);
}
```

Take the old pass about 4 numbers and check if them right or invalid or wring

Enter the pass as digits 1,10,100,1000

If the user entered right pass, we clear the past data, The pass should be 4 numbers

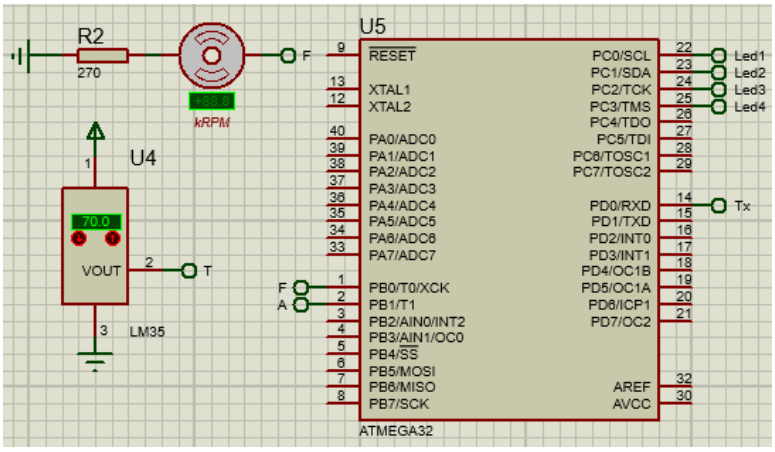
take the new pass and save it

Get the inverse of the pass to save it in data.

Then the process is done, and pass is changed

The van

We design it to turn on by itself. It takes the signal from the lm35 sensor and if the temp degree > 50 turn on LED as flash alert. If the temp degree > 30 the van turns on.




```
USART_Init();
Set_PinDir(PORT_B,Pin_0,OUTPUT);
Set_PinDir(PORT_B,Pin_1,OUTPUT);
Set_GroupDir(PORT_C,0x0F);
while(1)
{

    if(USART_receiveByte() == 'H')          // Alarm On
    {
        Set_PinLevel(PORT_B,Pin_0,HIGH);
        Set_PinLevel(PORT_B,Pin_1,HIGH);
    }
    else if(USART_receiveByte() == 'F')     // Fan On
    {
        Set_PinLevel(PORT_B,Pin_0,HIGH);
        Set_PinLevel(PORT_B,Pin_1,LOW);
    }
    else
    {
        Set_PinLevel(PORT_B,Pin_0,LOW);
        Set_PinLevel(PORT_B,Pin_1,LOW);
    }
```

Set pins as input / output

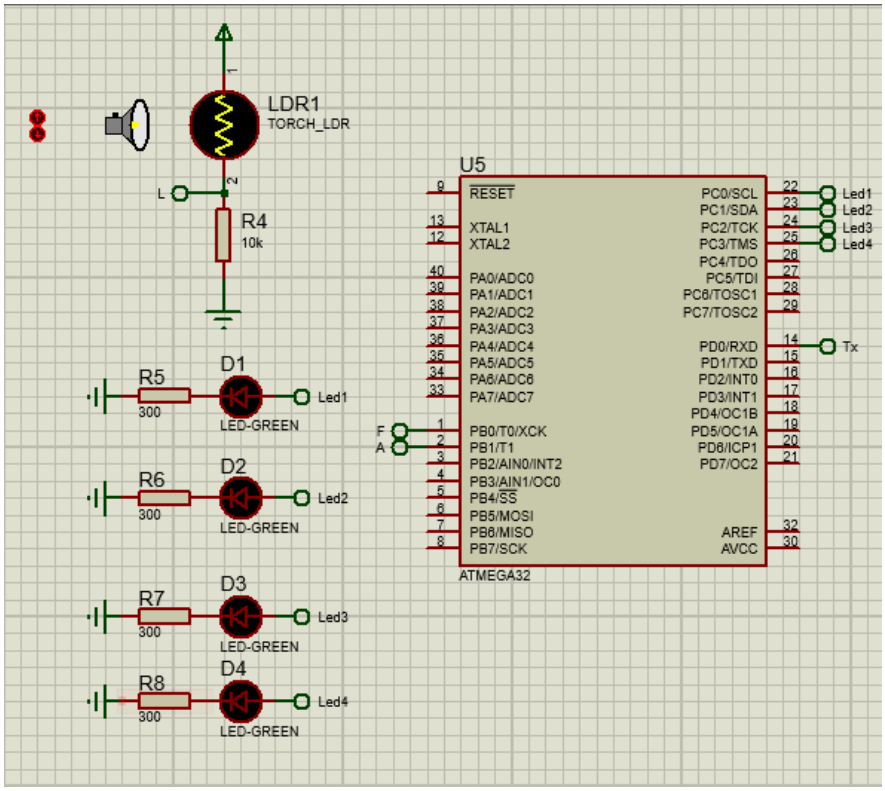
Check the temp degree If > 50 turn the alert on

Check the temp degree If > 30 turn the van on

Check the temp degree If <30 let the flash and the van off

The LED

We design it to sense the light by the LDR sensor if there is a light turn the LEDs off, and if not, turn the LEDs on.

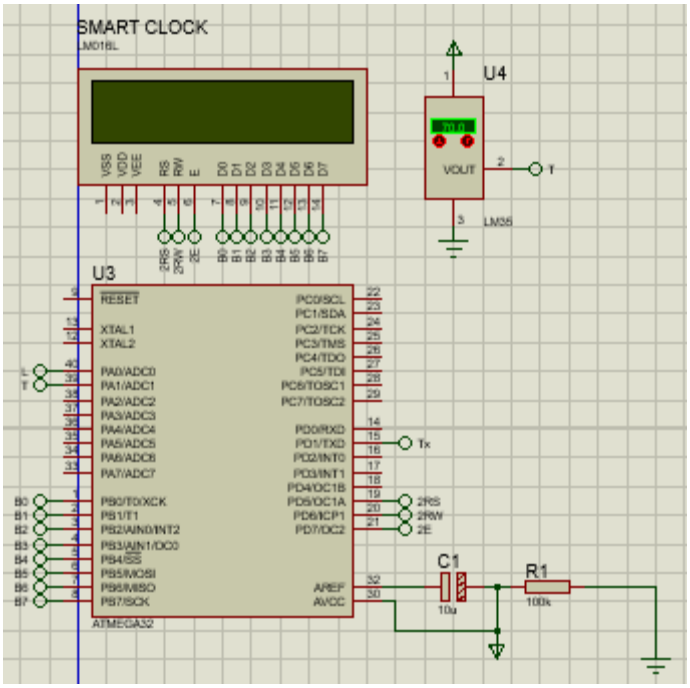


The code

```
if(USART_receiveByte() == 'L')          // Led On
{
    Set_GroupLevel(PORT_C,0x0F);
}
else
{
    Set_GroupLevel(PORT_C,0x00);
}
```

Smart Clock

We design it to sense the time and the temperature degree and showing it on lcd.



The code

```
// Clock Code
if (seconds == 60)
{
    LCD_CLEAR();
    seconds = 0;
    minuts++;
    if (minuts == 60)
    {
        LCD_CLEAR();
        minuts = 0;
        hours++;
        if (hours == 24)
        {
            hours = 0;
        }
    }
}
Go_To_Row_Col(0,0);
Display_Str("Time:  ");
Integer_TO_String(hours);
Display_Str(":");
Integer_TO_String(minuts);
Display_Str(":");
Integer_TO_String(seconds);
adc_value = ADC_readChannel(1);      // read adc value at PA0
Temperature = adc_value / 2.01;      // finding the temperature
Go_To_Row_Col(1,0);
Display_Str("Temp:  ");              //Display temperature
Integer_TO_String(Temperature);
Display_Str(" C     ");
```

```
if(Temperature > 30 && Temperature > 50)
{
    USART_sendByte('H');    // Alarm On
}
else if(Temperature > 30)
{
    USART_sendByte('F');    // Fan On
}
else
{
    USART_sendByte('X');
}

// LDR Code
ADC_Reading = ADC_readChannel(0);
mv_Value=(ADC_Reading * 5000)/256;
LDR_Value = mv_Value * 3.8;

if(LDR_Value < 400)
{
    USART_sendByte('L');    // Led On
}
else
{
    USART_sendByte('X');
}
```

```
void Timer0_init(void)
{
    TCNT0 = 7;              // Initialize Timer/Counter register with value 7;
    SET_BIT(TIMSK,TOIE0);   // Enable Overflow mode interrupt
    SET_BIT(TCCR0,CS02);    // PreScaler 1024
    SET_BIT(TCCR0,CS00);
    SET_BIT(TCCR0,FOC0);    //  Set Timer0 with normal mode & Start Counting
}
```

Drives

-> refers to a special kind of software program or a specific type of software application that controls a specific hardware device that enables different hardware devices to communicate with the computer's Operating System.

-> communicates with the computer hardware by computer subsystem or computer bus connected to the hardware.

So, we decide to use the drives to build our code and the drivers we used are:

- ✓ MCAL (DIO, ADC, UART)
- ✓ HAL (LCD, KEYPAD)
- ✓ UTLIS (Common_Macros, Std_Types)

DIO DRIVE

Digital input output device (prototype function)

```
typedef enum
{
    LOW,
    HIGH,
}PinLevel_t;

typedef enum
{
    INPUT,
    OUTPUT,
}PinDir_t;

typedef enum
{
    PORT_A,
    PORT_B,
    PORT_C,
    PORT_D,
}GroupName_t;
```

```
typedef enum
{
    LOW_VALUE,
    HIGH_VALUE,
}RET_VALUE_t;

typedef enum
{
    Pin_0,
    Pin_1,
    Pin_2,
    Pin_3,
    Pin_4,
    Pin_5,
    Pin_6,
    Pin_7,
}PIN_NO_t;
```

```
/*************************************************************************************************/
/*************************************Function Prototypes*****************************************/

//  For Certain Bit
void Set_PinDir (GroupName_t group, PIN_NO_t number, PinDir_t direction);   // Function to set direction of certain bit
void Set_PinLevel (GroupName_t group, PIN_NO_t number, PinLevel_t level );  // Function to set level of certain bit
RET_VALUE_t Read_PinLevel (GroupName_t group, PIN_NO_t number);             // Function to read level of certain bit

//   For Group Of Pins
void Set_GroupDir (GroupName_t group, u8 value);                            // Function to set direction of group of pins
void Set_GroupLevel (GroupName_t group, u8 value);                          // Function to set level of group of pins
```

USART

Universal **S**ynchronous and **A**synchronous serial **R**eceiver and **T**ransmitter.

It is a serial communication protocol between two microcontrollers

```
/*****************************************************************************************************/
/*************************************Preprocessor Macros*********************************************/

#define UART_BAUDRATE 9600

/*****************************************************************************************************/
/*************************************Function Prototypes*********************************************/

void USART_Init (void);                 // Function to initialize the USART
void USART_sendByte (u8 data);          // Function to send data byte
u8 USART_receiveByte (void);            // Function to receive data byte
void USART_sendString (const u8 *str);  // Function to send string
void USART_receiveString (u8 *str);     // Function to receive string
```

ADC

Analog Digital Converter translation unit that can convert any signal from its original analog form to a digital form that can be processed by the processor.

```
#include "../../UTLIS/Common_Macros.h"
#include "../../UTLIS/Std_Types.h"
#include <avr/io.h>
#include <avr/delay.h>
#include <avr/interrupt.h>
#include "../../MCAL/DIO/DIO.h"

/******************FUNCTION DEF***********/

void ADC_Init(void); //function to intialize ADC

u16 ADC_readChannel (u8 channel); //function to read data from specific channel
```

LCD

```
/*************************************************************************************************************/
/*****************************************Define Data Types*****************************************************/

// LCD CTRL Pins
#define LCD_CTRL_PORT_DIR DDRD
#define LCD_CTRL_PORT PORTD
#define RS PD0
#define RW PD1
#define EN PD2

// LCD Data Pins
#define LCD_DATA_PORT_DIR DDRC
#define LCD_DATA_PORT PORTC

// LCD Commands
#define CLEAR_COMMAND 0x01
#define TWO_LINE_LCD_EIGHT_BIT_MODE 0x38
#define CURSOR_OFF 0x0C
#define CURSOR_ON 0x0E
#define SET_CURSOR_LOCATION 0x80

/*************************************************************************************************************/
/*****************************************Function Prototypes*****************************************************/

void LCD_Init (void);                                       // Function to initialize the LCD
void Send_CMD (u8 cmd);                                     // Function to send commands to LCD
void Send_DATA (u8 data);                                   // Function to send data to LCD
void Display_Str (const u8 *str);                           // Function to send data to LCD
void Go_To_Row_Col (u8 row, u8 col);                        // Function to move to certain column and row on LCD
void LCD_CLEAR (void);                                      // Function to clear LCD
void Display_Str_Row_Col (u8 row, u8 col, const u8 *str);   //Display string at specified row and column
void Integer_TO_String (u8 value);                          // Display certain integer value on screen
```

KEYPAD

```
/*************************************************************************************************************/
/*****************************************Function Prototypes*****************************************************/

u8 Keypad_GetPressedKey (void);

/*************************************************************************************************************/
/*****************************************Define Data Types*****************************************************/

//  Keypad port Configuration
#define KEYPAD_PORT_DIR     DDRA
#define KEYPAD_PORT_OUT     PORTA
#define KEYPAD_PORT_IN      PINA

//  Keypad Configuration for Rows & ColumnS
#define N_ROW 4
#define N_COL 4
```