

# PROJECT

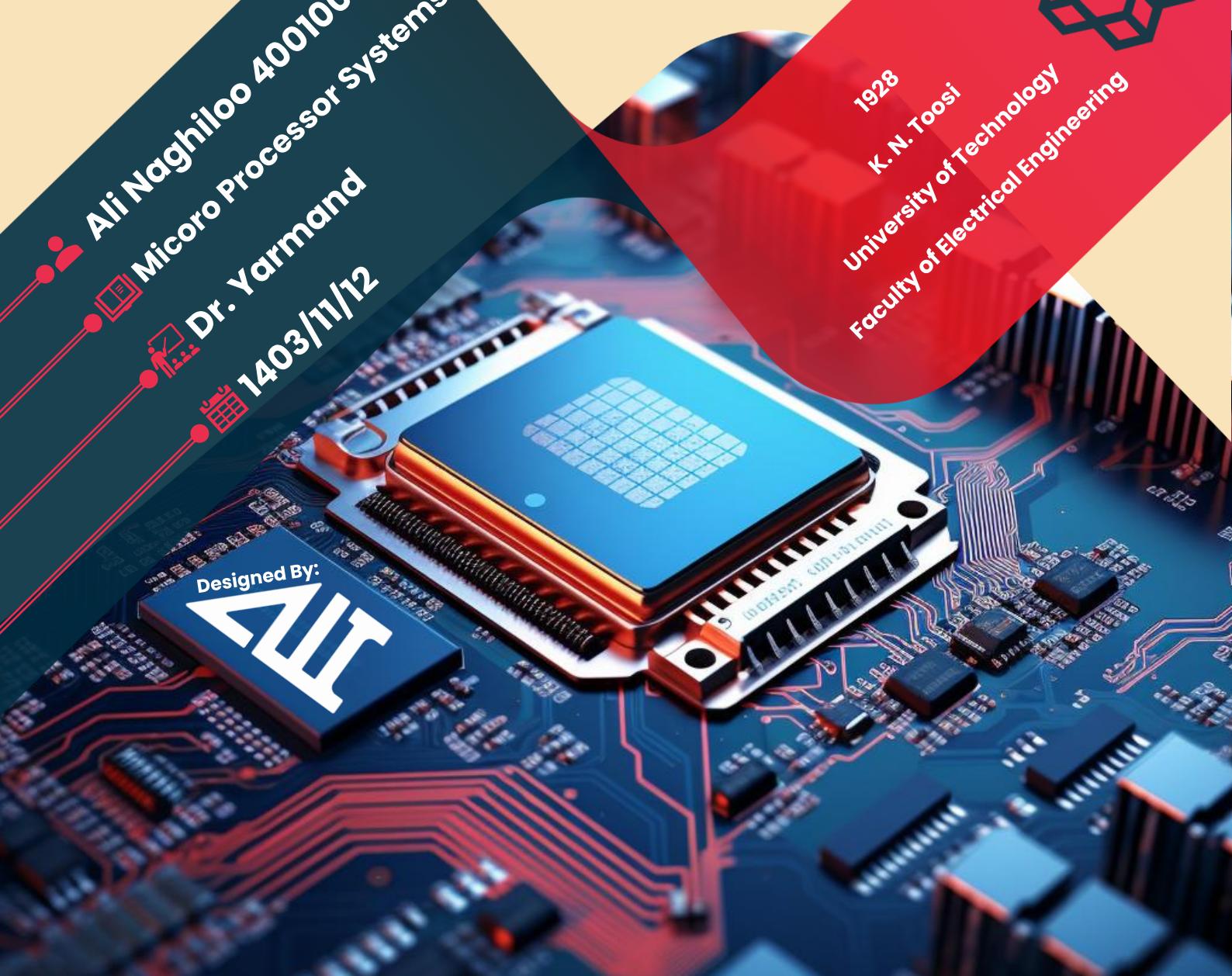
## Micro Processor Systems

Ali Naghilo 40010093  
Micro Processor Systems  
Dr. Yarmand  
1403/11/12

Designed By:



1928  
K. N. Toosi  
University of Technology  
Faculty of Electrical Engineering



## ۱ مقدمه

در دنیای امروزی، نظارت هوشمند بر محیط‌های مختلف از جمله گلخانه‌ها، انبارها و منازل از اهمیت ویژه‌ای برخوردار است. استفاده از سیستم‌های هوشمند برای پایش و کنترل عوامل محیطی نظیر دما، رطوبت و نور، نقش مهمی در بهینه‌سازی مصرف انرژی، افزایش بهره‌وری و کاهش هزینه‌های نگهداری دارد.

پروژه حاضر یک سیستم نظارت و کنترل هوشمند را معرفی می‌کند که برپایه میکروکنترلر STM32 طراحی شده است. این سیستم قابلیت اندازه‌گیری پارامترهای محیطی از جمله دما، رطوبت و شدت نور را دارد و در صورت خروج این مقادیر از محدوده مجاز، اقدامات کنترلی مانند روشن/خاموش کردن فن، پمپ آب و بخاری را انجام می‌دهد. همچنین، در صورت بروز شرایط خطرناک، سیستم از طریق مایزول GSM پیام هشدار ارسال می‌کند. علاوه بر این، داده‌های اندازه‌گیری شده در حافظه EEPROM ذخیره شده و بر روی نمایشگر LCD 2004 نمایش داده می‌شوند.

هدف از این پروژه، ارائه یک راهکار کارآمد و کم‌هزینه برای پایش شرایط محیطی و افزایش ایمنی و بهره‌وری در محیط‌های مختلف است. این سیستم می‌تواند در کاربردهای کشاورزی، صنعتی و خانگی مورد استفاده قرار گیرد و با ارتقا‌های نرم‌افزاری و سخت‌افزاری بیشتر، به یک سامانه جامع‌تر برای مدیریت هوشمند محیط تبدیل شود.

## ۲ سخت‌افزار مورد استفاده

برای پیاده‌سازی این پروژه، از سخت‌افزارهای مختلفی استفاده شده است که در ادامه معرفی می‌شوند:

**میکروکنترلر STM32 :** این پروژه برپایه میکروکنترلر سری STM32F1 پیاده‌سازی شده است که به دلیل توان پردازشی مناسب و امکانات جانی گسترده انتخاب شده است.

سنسورهای اندازه‌گیری:

**ADC داخلی STM32 :** برای اندازه‌گیری مقدار آنالوگ خروجی سنسورهای نور، رطوبت و دما استفاده شده است.

**سنسور نور (LDR) :** شدت نور محیط را اندازه‌گیری کرده و مقدار آن به درصد تبدیل می‌شود.

**سنسور رطوبت خاک :** مقدار رطوبت خاک را به صورت درصدی مشخص می‌کند.

**سنسور دما :** مقدار دمای محیط را اندازه‌گیری کرده و مقدار آن بر حسب درجه سانتی‌گراد نمایش داده می‌شود.

**Nمایشگر 4x20 LCD :** برای نمایش اطلاعات اندازه‌گیری شده و وضعیت سیستم.

**حافظه EEPROM (25LC512) :** جهت ذخیره‌سازی داده‌های مربوط به سنسورها و وضعیت سیستم.

**مایزول GSM (SIM800) :** برای ارسال پیامک هشدار در شرایط بحرانی نظیر باز بودن درب به مدت طولانی، قرار گرفتن میزان نور، دما یا رطوبت در محدوده خطرناک.

عملکرها:

فن: برای کنترل دما و تهویه هوا.

پمپ آب: برای تنظیم رطوبت محیط.

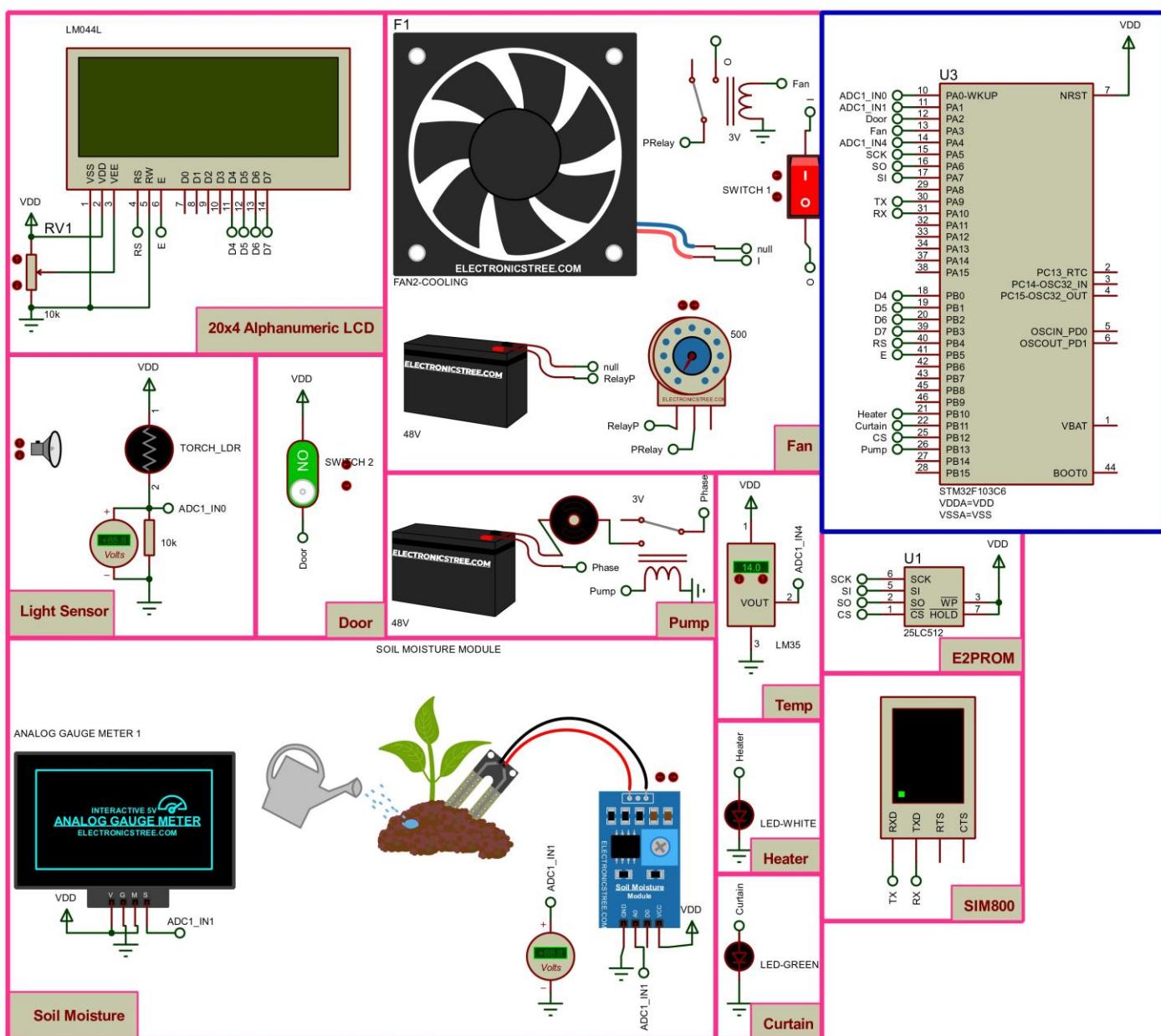
هیتر: برای افزایش دمای محیط.

پرده هوشمند: برای تنظیم میزان نور ورودی به محیط.

مدارهای راه انداز: شامل ترانزیستورها، رله ها و درایورهای مناسب برای کنترل عملکرها از طریق میکروکنترلر.

این سخت افزارها با استفاده از پروتکل های ارتباطی مانند SPI، UART و GPIO به یکدیگر متصل شده و عملیات لازم را انجام می دهند.

## شمایتیک گلی پروژه در PROTEUS



## [ ] طراحی نرم افزار

### ۱. هدف کلی کد:

این کد برای مدیریت هوشمند شرایط محیطی گلخانه طراحی شده است. با استفاده از سنسورهای دما، نور، رطوبت خاک و وضعیت درب، داده‌ها جمع‌آوری شده و عملگرهایی مانند فن، بخاری، پمپ آب و پرده به صورت خودکار کنترل می‌شوند. داده‌هاروی LCD نمایش داده شده و در حافظه ذخیره می‌شوند. در شرایط خطر (مثلًاً دما یا رطوبت غیرمجاز)، هشدار از طریق پیامک ارسال می‌گردد.

### ۲. بخش‌های اصلی کد:

#### الف) سنسورها و جمع‌آوری داده:

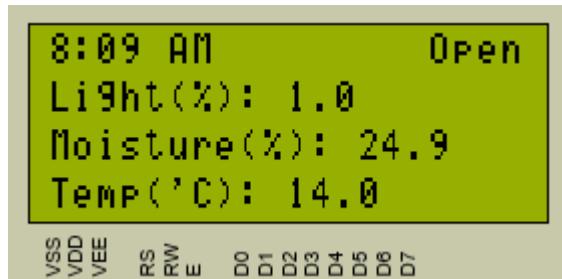
- سنسور دما (LM35): مقدار دمای محیط را اندازه می‌گیرد. داده خام ADC به درجه سانتیگراد تبدیل می‌شود.
- سنسور نور (LDR): سطح نور محیط را تشخیص می‌دهد. داده ADC به درصد تبدیل می‌شود.
- سنسور رطوبت خاک: رطوبت خاک را اندازه می‌گیرد (با شبیه‌سازی مقاومت متغیر). داده ADC به درصد تبدیل می‌شود.
- سنسور درب: وضعیت باز یا بسته بودن درب را با یک کلید (Push Button) تشخیص می‌دهد.

#### ب) عملگرها و کنترل:

- فن: اگر دما از حد مجاز (مثلًاً  $35^{\circ}\text{C}$ ) بیشتر شود، روشن می‌شود و در دمای پایین‌تر (مثلًاً  $25^{\circ}\text{C}$ ) خاموش می‌شود.
- بخاری: با PWM کنترل می‌شود. اگر دما از حد مجاز (مثلًاً  $25^{\circ}\text{C}$ ) کمتر شود، قدرت بخاری افزایش می‌یابد.
- پمپ آب: اگر رطوبت خاک از حد مجاز (مثلًاً ۳۳٪) کمتر شود، روشن می‌شود و در رطوبت بالا (مثلًاً ۷۵٪) خاموش می‌شود.
- پرده: با PWM کنترل می‌شود. اگر نور از حد مجاز (مثلًاً ۶۰٪) بیشتر یا کمتر شود، پرده باز یا بسته می‌شود.

#### ج) نمایش اطلاعات:

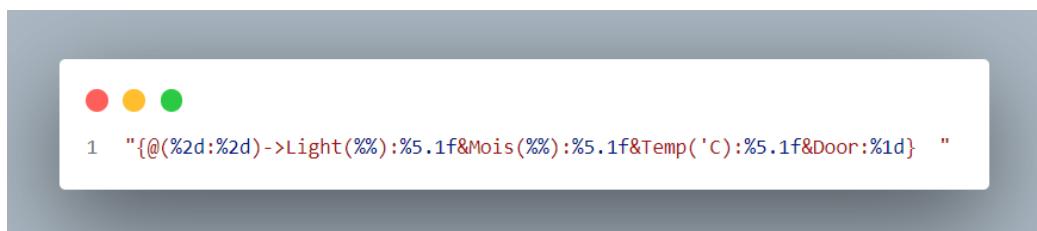
- LCD 20x4: زمان (ساعت و دقیقه)، (تاریخ که به علت جانشدن کنسل کردم این ویژگی رو ولی در کد هست)، مقادیر سنسورها (دما، نور، رطوبت خاک) و وضعیت درب به صورت لحظه‌ای نمایش داده می‌شوند. (علامت ° پشتیبانی نمی‌شود و مجبور شدم بجاش از 'استفاده کنم.)



د) ذخیره‌سازی داده:

- EEPROM 25LC512: داده‌های محیطی (دما، نور، رطوبت، وضعیت درب) همراه با زمان و تاریخ هر ثانیه در حافظه خارجی ذخیره می‌شوند.

- قالب ذخیره‌سازی: داده‌ها به صورت متن (مثل تصویر زیر) ذخیره می‌شوند. فقط اینکه متناسفانه علامت ° ساپورت نشده توسط حافظه در پروتئوس و بجاиш نقطه نوشته شده است.



SPI Memory\Internal Memory - U1

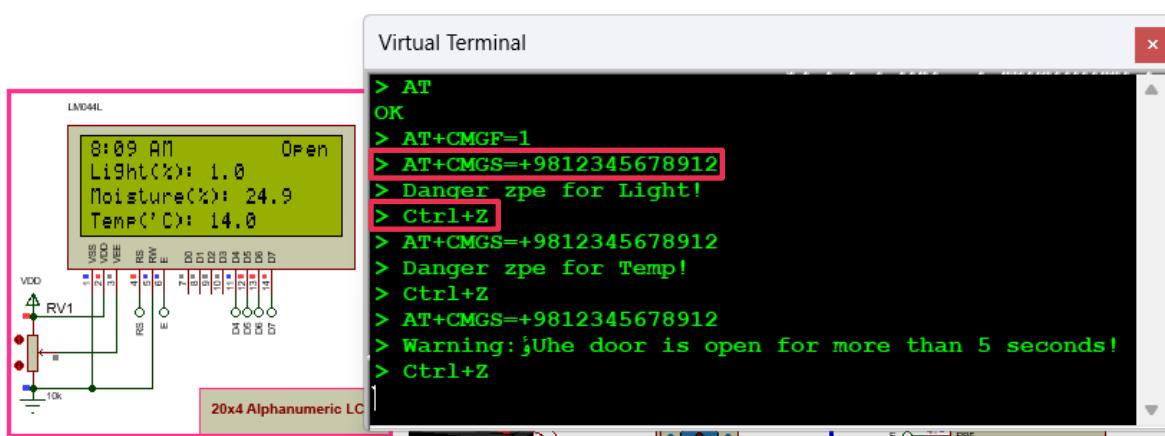
0000	7B	40	28	20	38	3A	20	39	29	2D	3E	4C	69	67	68	74	28	25	29	3A	20	33	33	2E	33	26	4D	6F	69	73	28	25
0020	29	3A	20	37	35	2E	31	26	54	65	6D	70	28	27	43	29	3A	20	33	30	2E	39	26	44	6F	6F	72	3A	30	7D	20	20
0040	7B	40	28	20	38	3A	20	39	29	2D	3E	4C	69	67	68	74	28	25	29	3A	20	33	33	2E	33	26	4D	6F	69	73	28	25
0060	29	3A	20	37	35	2E	31	26	54	65	6D	70	28	27	43	29	3A	20	33	30	2E	39	26	44	6F	6F	72	3A	30	7D	20	20
0080	7B	40	28	20	38	3A	20	39	29	2D	3E	4C	69	67	68	74	28	25	29	3A	20	33	33	2E	33	26	4D	6F	69	73	28	25
00A0	29	3A	20	37	35	2E	31	26	54	65	6D	70	28	27	43	29	3A	20	33	30	2E	39	26	44	6F	6F	72	3A	30	7D	20	20
00C0	7B	40	28	20	38	3A	20	39	29	2D	3E	4C	69	67	68	74	28	25	29	3A	20	31	36	2E	37	26	4D	6F	69	73	28	25
00E0	29	3A	20	37	35	2E	31	26	54	65	6D	70	28	27	43	29	3A	20	33	30	2E	31	26	44	6F	6F	72	3A	31	7D	20	20
0100	7B	40	28	20	38	3A	20	39	29	2D	3E	4C	69	67	68	74	28	25	29	3A	20	31	36	2E	37	26	4D	6F	69	73	28	25
0120	29	3A	20	37	35	2E	31	26	54	65	6D	70	28	27	43	29	3A	20	33	30	2E	30	26	44	6F	6F	72	3A	31	7D	20	20
0140	7B	40	28	20	38	3A	20	39	29	2D	3E	4C	69	67	68	74	28	25	29	3A	20	31	36	2E	37	26	4D	6F	69	73	28	25
0160	29	3A	20	37	35	2E	31	26	54	65	6D	70	28	27	43	29	3A	20	33	30	2E	30	26	44	6F	6F	72	3A	31	7D	20	20
0180	7B	40	28	20	38	3A	20	39	29	2D	3E	4C	69	67	68	74	28	25	29	3A	20	31	36	2E	37	26	4D	6F	69	73	28	25
01A0	29	3A	20	37	35	2E	31	26	54	65	6D	70	28	27	43	29	3A	20	33	30	2E	30	26	44	6F	6F	72	3A	31	7D	20	20
01C0	7B	40	28	20	38	3A	20	39	29	2D	3E	4C	69	67	68	74	28	25	29	3A	20	31	36	2E	37	26	4D	6F	69	73	28	25
01E0	29	3A	20	37	35	2E	31	26	54	65	6D	70	28	27	43	29	3A	20	32	39	2E	30	26	44	6F	6F	72	3A	31	7D	20	20
0200	7B	40	28	20	38	3A	20	39	29	2D	3E	4C	69	67	68	74	28	25	29	3A	20	31	36	2E	37	26	4D	6F	69	73	28	25
0220	29	3A	20	36	35	2E	31	26	54	65	6D	70	28	27	43	29	3A	20	32	38	2E	30	26	44	6F	6F	72	3A	31	7D	20	20

## ۵) ارتباطات و هشدار:

- مژول GSM (SIM800): در شرایط خطر (مثلاً دمای بالای ۴۰°C یا رطوبت خاک زیر ۱۵٪)، پیامک به شماره تعیین شده ارسال می‌شود.

- هشدار درب: اگر درب بیش از ۵ ثانیه باز بماند، پیامک هشدار ارسال می‌شود.

توجه: به دلیل اینکه مژول وجود خارجی ندارد، ما خودمان باید نقش آن را بازی کنیم. مثلاً خودمان باید تایپ کنیم OK تا میکرو کنترلر آن را دریافت و ادامه کد ران شود. پس جاهایی که ما (یعنی SIM800) فرمانی به میکرو ارسال میکند بدون > و جاهایی که میکرو به SIM800 فرمانی صادر میکند با > در ترمینال مشخص شده اند. ضمناً دستورات AT به کار رفته اغلب مشابه تمرین ۳ و فقط ارسال اس ام اس دستور جدید دارد که با **کادر قرمز** مشخص شده است. یکی قبل از ارسال پیام برای تعیین شماره و دیگری (CTRL+Z) به معنا اتمام پیام است.



## ۳. نحوه اجرا:

- راه اندازی اولیه: میکرو کنترلر (STM32F103) مژول های UART، SPI، RTC، ADC، PWM را پیکربندی می کند.

```

● ● ●
1 HAL_Init();
2 SystemClock_Config();
3
4 MX_GPIO_Init();
5 MX_DMA_Init();
6 MX_ADC1_Init();
7 MX_USART1_UART_Init();
8 MX_RTC_Init();
9 MX_TIM1_Init();
10 MX_TIM2_Init();
11 MX_SPI1_Init();
12
13 HAL_stuffs();
14 LCD_Init();
15 GSM_Init();

```

```

while (1)
{
    E2PROM_Init();

    sprintf(printer0 ,"%d:%02d %s\0 %s,%s \0", hours[SensorData.theTime.Hours % 12],
    /* \0 is intentional to truncate for LCD */ SensorData.theTime.Minutes,
    meridiem[SensorData.theTime.Hours / 12],
    days[SensorData.theDate.WeekDay],
    months[SensorData.theDate.Month-1],
    SensorData.theDate.Date);

    LCD(0, 0, printer0);
    SensorData.DoorStatus = (HAL_GPIO_ReadPin(Door_Port, Door_Pin) == GPIO_PIN_SET);
    LCD(0, 15, SensorData.DoorStatus ? " Open": "Close");
    if (SensorData.DoorStatus && !doorWasOpen)
    {
        doorOpenTime = SensorData.theTime.Seconds;
        doorWasOpen = 1;
    }
    if (SensorData.DoorStatus)
    {
        uint32_t elapsed = (SensorData.theTime.Seconds >= doorOpenTime) ?
            (SensorData.theTime.Seconds - doorOpenTime) :
            (60 - doorOpenTime +SensorData.theTime.Seconds);

        if (elapsed > 5)
        {
            GSM_Send_Message("Warning: The door is open for more than 5 seconds!", "+9812345678912");
            doorOpenTime = SensorData.theTime.Seconds + 5;
        }
    }
    else
    {
        doorWasOpen = 0;
    }

    SensorData.Light = ADCtoPercentage(AChannels[0]);
    sprintf(printer0, "Light(%): %-4.1f", SensorData.Light);
    LCD(1, 0, printer0);
    currentDanger.light = (SensorData.Light <= 5.0 || SensorData.Light >= 60.0);
    if (currentDanger.light != previousDanger.light)
    {
        if (currentDanger.light)
        {
            GSM_Send_Message("Danger zone for Light!", "+9812345678912");
        }
        previousDanger.light = currentDanger.light;
    }

    SensorData.Moisture = ADCtoPercentage(AChannels[1]);
    sprintf(printer0, "Moisture(%): %-4.1f", SensorData.Moisture);
    LCD(2, 0, printer0);
    currentDanger.moisture = (SensorData.Moisture <= 15.0 || SensorData.Moisture >= 90.0);
    if (currentDanger.moisture != previousDanger.moisture)
    {
        if (currentDanger.moisture)
        {
            GSM_Send_Message("Danger zone for Moisture!", "+9812345678912");
        }
        previousDanger.moisture = currentDanger.moisture;
    }

    SensorData.Temp = ADCtoCentigrad(AChannels[2]);
    sprintf(printer0, "Temp('C): %-4.1f", SensorData.Temp);
    LCD(3, 0, printer0);
    currentDanger.temp = (SensorData.Temp <= 15.0 || SensorData.Temp >= 40.0);
    if (currentDanger.temp != previousDanger.temp)
    {
        if (currentDanger.temp)
        {
            GSM_Send_Message("Danger zone for Temp!", "+9812345678912");
        }
        previousDanger.temp = currentDanger.temp;
    }

    FanController(SensorData.Temp, FanOn, FanOff);

    PumpController(SensorData.Moisture, PumpOff, PumpOn);

    PWMCurtain(SensorData.Light, 50.0, 10.0);

    PWMHeater(SensorData.Temp, 25.0, 20.0);

    if (write == yes)
    {
        write = no;
        EEPROM_Write(E2PROM_WriteAddress+last_adr, printer1);
        last_adr += strlen(printer1);
    }
}
}

```

۱. داده‌های سنسورها خوانده و پردازش می‌شوند.

۲. وضعیت خطر (مثلاً نور کم/زیاد) بررسی و در صورت تغییر، هشدار ارسال می‌شود.

۳. عملکرها (فن، پمپ، بخاری، پرده) براساس داده‌ها کنترل می‌شوند.

۴. داده‌ها روی LCD نمایش و در EEPROM ذخیره می‌شوند.

#### ۴. نکات فنی مهم:

- تبدیل ADC:

× نور و رطوبت خاک: از محدوده ۰-۴۰۹۵ به درصد تبدیل می‌شوند.

× دما: با فرمول خاصی به درجه سانتیگراد تبدیل می‌شود (با تنظیم آفست).

- کنترل PWM: برای تنظیم نرم قدرت بخاری و پرده استفاده می‌شود.

- استفاده از DMA: برای خواندن مقادیر ADC بدون دخالت CPU.

موارد بالا در صفحه بعد به صورت کد قرار داده شده‌اند. ☺

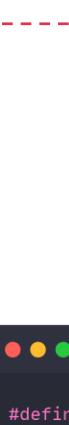
- مدیریت زمان: از RTC برای زمان‌بندی ذخیره‌سازی داده و هشدارها استفاده می‌شود که در پایین کال بک آله

 مشاهده می‌شود



```

1 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
2 {
3     if (htim->Instance == TIM1)
4     {
5         HAL_RTC_GetTime(&hrtc, &SensorData.theTime, RTC_FORMAT_BIN);
6         HAL_RTC_GetDate(&hrtc, &SensorData.theDate, RTC_FORMAT_BIN);
7
8         sprintf(prINTER1, "@(%2d;%2d)->Light(%%):%5.1f&Mois(%%):%5.1f&Temp('C):%5.1f&Door:%1d} ", SensorData.theTime.Hours,
9
10
11
12
13
14
15     write = yes;
16 }
17 }
```



```

#define HAL_stuffs() HAL_TIM_Base_Start_IT(&htim1); \
    HAL_ADC_Start_DMA(&hadc1, AChannels, sizeof(AChannels)); \
    HAL_TIM_PWM_Start(&htim2, Heater_CH); \
    HAL_TIM_PWM_Start(&htim2, Curtain_CH); \
    HAL_RTC_GetTime(&hrtc, &SensorData.theTime, RTC_FORMAT_BIN); \
    HAL_RTC_GetDate(&hrtc, &SensorData.theDate, RTC_FORMAT_BIN)

#define LCD_Init() Lcd_PortType D_ports[] = {portD4, portD5, portD6, portD7}; \
    Lcd_PinType D_pins[] = {pinD4, pinD5, pinD6, pinD7}; \
    SensorData.hlcd1 = Lcd_create(D_ports, D_pins, portRS, pinRS, portE, pinE, LCD_4_BIT_MODE)

#define EEPROM_Init() EEPROM_Init(&SensorData.E2PROM, &hspi1, CS_GPIO_Port, CS_Pin)

#define EEPROM_Write(writeAddress, data) EEPROM_Write(&SensorData.E2PROM, writeAddress, (uint8_t *)data, strlen(data));

#define LCD(row, column, text) Lcd_cursor(&SensorData.hlcd1, row, column); \
    Lcd_string(&SensorData.hlcd1, text)

#define ADCtoPercentage(Channel) (float)Channel * (100.0 / 4095.0)

#define ADCtoCentigrad(Channel) ((float)Channel * (5.0 / 4095.0) / 0.01) - tune_offset

#define FanController(temp, m, M) (temp >= m) \
    ? HAL_GPIO_WritePin(Fan_GPIO_Port, Fan_Pin, GPIO_PIN_SET) \
    : ((temp <= M) \
        ? HAL_GPIO_WritePin(Fan_GPIO_Port, Fan_Pin, GPIO_PIN_RESET) \
        : __NOP())

#define PumpController(moisture, m, M) (moisture <= M) \
    ? HAL_GPIO_WritePin(Pump_GPIO_Port, Pump_Pin, GPIO_PIN_SET) \
    : ((moisture >= m) \
        ? HAL_GPIO_WritePin(Pump_GPIO_Port, Pump_Pin, GPIO_PIN_RESET) \
        : __NOP())

#define PWMCurtain(light, m, M) pwm_value = (uint32_t)((light <= M) \
    ? ((M - light) * 0.01 * 65535) \
    : ((light >= m) \
        ? 0 \
        : (10.0 * 65535.0) * 0.01)); \
    __HAL_TIM_SET_COMPARE(&htim2, Curtain_CH, pwm_value)

#define PWMHeater(temp, m, M) pwm_value = (uint32_t)((temp < M) \
    ? ((M - temp) * 10.0 * 0.01 * 65535.0) \
    : ((temp >= m) \
        ? 0 \
        : (5.0 * 65535.0) * 0.01)); \
    __HAL_TIM_SET_COMPARE(&htim2, Heater_CH, pwm_value)

#define GSM_Init() HAL_UART_Transmit(&huart1, "> AT\r\n", strlen("> AT\r\n"), 1000); \
    uint8_t condition; \
    do { \
        condition = HAL_UART_Receive(&huart1, OK, sizeof(OK), 1000); \
    } while (condition); \
    HAL_UART_Transmit(&huart1, "OK\r\n", strlen("OK\r\n"), 1000); \
    HAL_UART_Transmit(&huart1, "> AT+CMGF=1\r\n", strlen("> AT+CMGF=1\r\n"), 1000)

#define GSM_Send_Message(text, number) sprintf((char *)Message, "> AT+CMGS=%s\r\n%$s\r\nCtrl+Z\r\n", number, text); \
    HAL_UART_Transmit(&huart1, Message, strlen((char const *)Message), 1000)

```

## failel هدري يعني main.h

```

1 #ifndef __MAIN_H
2 #define __MAIN_H
3
4 #ifdef __cplusplus
5 extern "C" {
6 #endif
7
8 #include "stm32f1xx_hal.h"
9
10 void Error_Handler(void);
11
12 #define Fan_Pin GPIO_PIN_3
13 #define Fan_GPIO_Port GPIOA
14 #define CS_Pin GPIO_PIN_12
15 #define CS_GPIO_Port GPIOB
16 #define Pump_Pin GPIO_PIN_13
17 #define Pump_GPIO_Port GPIOB
18
19 #define Door_Pin GPIO_PIN_2
20 #define Door_Port GPIOA
21
22 #define portD4 GPIOB
23 #define portD5 GPIOB
24 #define portD6 GPIOB
25 #define portD7 GPIOB
26 #define portRS GPIOB
27 #define porte GPIOB
28
29 #define pinD4 GPIO_PIN_0
30 #define pinD5 GPIO_PIN_1
31 #define pinD6 GPIO_PIN_2
32 #define pinD7 GPIO_PIN_3
33 #define pinRS GPIO_PIN_4
34 #define pinE GPIO_PIN_5
35
36 #define Heater_CH TIM_CHANNEL_3
37 #define Curtain_CH TIM_CHANNEL_4
38
39 #define E2PROM_writeAddress 0x0000
40
41 #define FanOn 30.0
42 #define FanOff 25.0
43
44 #define PumpOff 75.0
45 #define PumpOn 35.0
46
47 #define tune_offset 0.2
48
49 #ifdef __cplusplus
50 }
51 #endif
52
53 #endif /* __MAIN_H */
54

```

## تعاريف متغيرها و تايپ های مورد نياز در کد

```

1 typedef struct {
2     EEPROM_HandleTypeDef E2PROM;
3     Lcd_HandleTypeDef lcd1;
4     RTC_TimeTypeDef theTime;
5     RTC_DateTypeDef theDate;
6     float Temp;
7     float Light;
8     float Moisture;
9     uint8_t DoorStatus;
10 } SensorDataType;
11 typedef struct {
12     uint8_t light;
13     uint8_t moisture;
14     uint8_t temp;
15 } CurrentDangerType;
16 typedef struct {
17     uint8_t light;
18     uint8_t moisture;
19     uint8_t temp;
20 } PreviousDangerType;

```

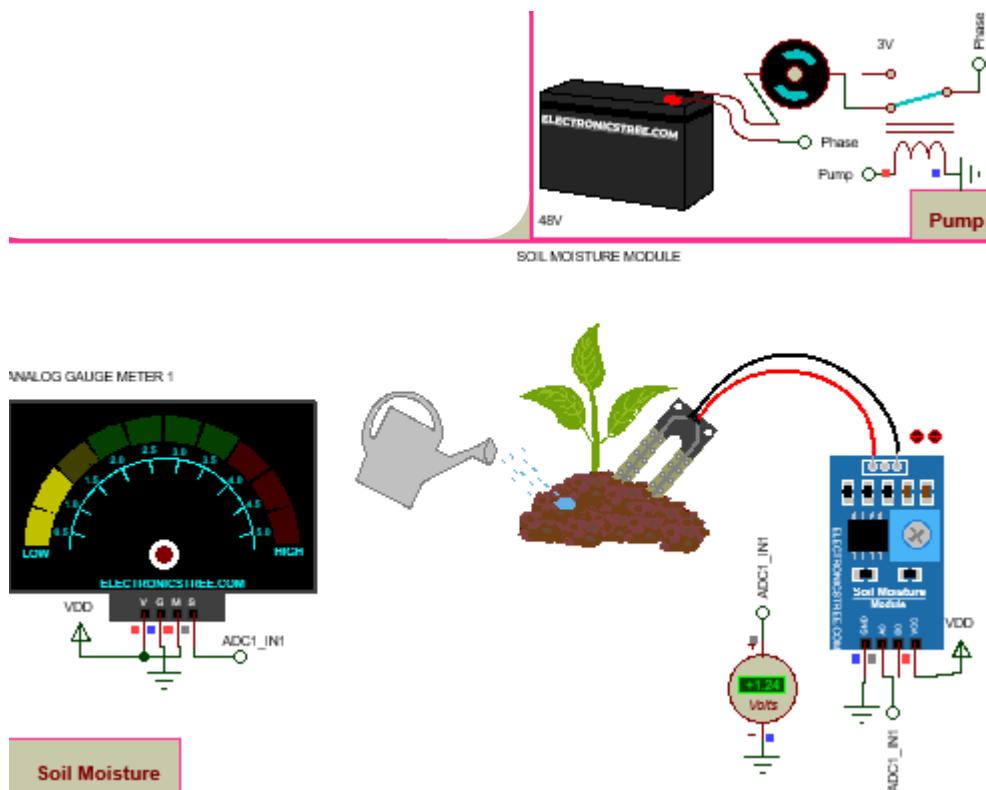
```

1 SensorDataType SensorData;
2
3 char* months[] = {"Jan", "Feb", "Mar", "Apr", "May", "Jun",
4                     "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};
5
6 char* days[] = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"};
7
8 int hours[] = {12, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11};
9
10 char* meridiem[] = {"AM", "PM"};
11
12 char printer0 [21];
13 char printer1 [100];
14
15 uint32_t AChannels[3];
16
17 uint32_t pwm_value;
18
19 uint16_t last_addr = 0;
20
21 static PreviousDangerType previousDanger = {0, 0, 0};
22 static CurrentDangerType currentDanger = {0, 0, 0};
23
24 uint8_t OK[] = {"OK\r\n"};
25 uint8_t Message[] = {"> AT+CMGS=%s\r\n%s\r\n> Ctrl+Z\r\n"};
26
27 uint32_t doorOpenTime = 0;
28 uint8_t doorWasOpen = 0;
29
30 enum {yes, no}write;

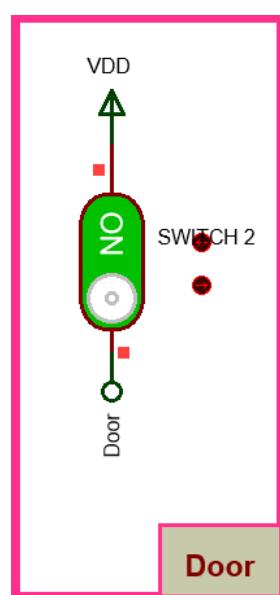
```

بررسی عملکرد برخی مازول هایی که قبل تر گفته نشدند:

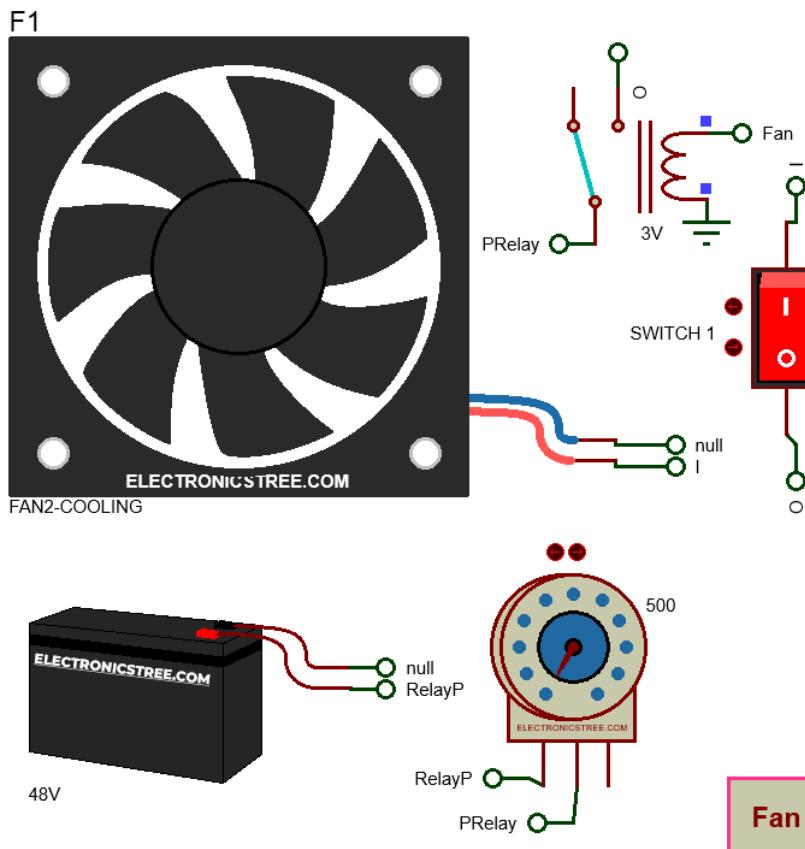
پمپ و روطوبت خاک که همانطور که می بینید، زمانی که رطوبت در ناحیه زرد رنگ نوار باشد، پمپ وصل و اگر در ناحیه قرمز باشد، پمپ قطع می شود.



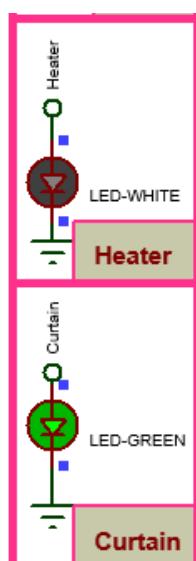
در مورد درب نیز، هر چهار کلید روشن/خاموش باشد، روی نمایشگر باز یا بسته نوشته می شود.



درمورد فن نیز، اغلب نکات گفته شد ولی بهتر است این مورد هم بیان شود که بنده یه کلید جداگانه برای خاموش/روشن کردن در نظر گرفته ام که دستی نیز بتوان تصمیم گرفت. همچنین یک پتانسیو متر قرار گرفته است که میتوان با آن سرعت چرخش را تنظیم کرد.



درمورد قطعاتی که ولتاژ آبالوگ را با PWM برایشان تداعی میکردیم هم باید گفت که به درستی عمل میکنند ولی چون پروتئوس هنگ است، خیلی اغراقانه نور ال ای دی شان، چشمک میزند یا به عبارتی سو سو میکنند که این در واقعیت به شکل نورکم پا زیاد هست و این شکلی نباید باشد. (ال ای دی سفید و سبز)



## ۷ نتیجه‌گیری

پروژه سیستم مانیتورینگ و کنترل گلخانه با موفقیت توانست شرایط محیطی (دما، نور، رطوبت خاک و وضعیت درب) را به صورت لحظه‌ای اندازه‌گیری، کنترل و نمایش دهد. عملکرد سیستم در کنترل خودکار عملکرها (فن، بخاری، پمپ و پرده) مبتنی بر آستانه‌های تعیین شده، پایدار و قابل اعتماد بود. ذخیره‌سازی داده‌های DHT22 و ارسال هشدار از طریق GSM در شرایط خطر، ارزش افزوده قابل توجهی به سیستم بخشید.

۷ به عنوان آخرین گام، نظر خودم درمورد این پروژه این بود که تجربه خوبی بود ولی تست آن در پروتئوس زجر آوررر بود و همچنین برای انجام دادن عملی این پروژه باید از سنسورهای دقیق‌تر دما و رطوبت مثل DHT22 استفاده کرد.