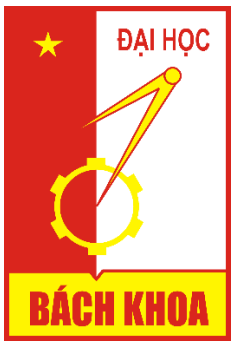
ĐẠI HỌC BÁCH KHOA HÀ NỘI

Trường Đại Học Công Nghệ Thông Tin và Truyền Thông

----- □ & □ -----



Embedded Systems

<u>Đề tài</u>: Bật/tắt đèn tự động sử dụng cảm biến ánh sáng và hẹn giờ

Giảng viên: **Ngô Lam Trung**

Mã lớp: 139400

Nhóm sinh viên thực hiện:

STT	Họ và tên	MSSV
1	Nguyễn Văn Tiến	20205195
2	Vũ Hồng Hải	20205150

Hà Nội, năm 2023

MỞ ĐẦU

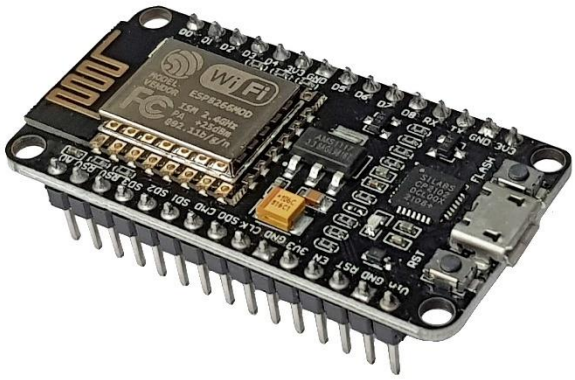
Tự động hóa và tích hợp công nghệ đã dẫn đến sự phát triển vượt bậc của các hệ thống nhúng trong thời gian gần đây. Những hệ thống nhúng này không chỉ tồn tại xung quanh chúng ta mà còn đóng một vai trò quan trọng trong nhiều khía cạnh của cuộc sống hàng ngày, từ các thiết bị điện tử tiêu dùng như điện thoại thông minh và đồng hồ thông minh, đến các ứng dụng y tế, công nghiệp, và giao thông. Báo cáo này nhằm tìm hiểu và trình bày về một sản phẩm hệ thống nhúng, đó là bật/tắt đèn tự động sử dụng cảm biến ánh sáng và hẹn giờ.

CHƯƠNG I. CÁC THIẾT BI, LINH KIỆN ĐƯỢC SỬ DỤNG

Bao gồm :

- NodeMCU ESP8266
- RTC DS1307
- LCD1602 I2C
- LED
- Light sensor
- Tụ
- Điện trở

I.1. ESP8266



ESP8266 là một hệ thống trên chip (SoC), do công ty Espressif của Trung Quốc sản xuất. Nó bao gồm bộ vi điều khiển Tensilica L106 32-bit (MCU) và bộ thu phát Wi-Fi. Nó có 11 chân GPIO (Chân đầu vào / đầu ra đa dụng) và một đầu vào analog, có nghĩa là bạn có thể lập trình nó giống như với Arduino hoặc vi điều khiển khác. Bản thân chip ESP8266 có 17 chân GPIO, nhưng 6 trong số các chân này (6-11) được sử dụng để giao tiếp với chip nhớ flash trên bo mạch. Ngoài ra nó có kết nối Wi-Fi, vì vậy có thể sử dụng nó để kết nối với mạng Wi-Fi, kết nối Internet, lưu trữ máy chủ web với các trang web thực, để điện thoại thông minh của mình kết nối với nó, ...

ESP8266 có thể được dùng làm module Wifi bên ngoài, sử dụng firmware tập lệnh AT tiêu chuẩn bằng cách kết nối nó với bất kỳ bộ vi điều khiển nào sử dụng UART nối tiếp hoặc trực tiếp làm bộ vi điều khiển hỗ trợ Wifi, bằng cách lập trình một chương trình cơ sở mới sử dụng SDK được cung cấp. Các chân GPIO cho phép IO Analog và Digital, cộng với PWM, SPI, I2C, v.v.

Bảng phát triển NodeMCU ESP8266 đi kèm với mô-đun ESP-12E chứa chip ESP8266 có bộ vi xử lý Tensilica Xtensa 32-bit LX106 RISC. Bộ vi xử lý này hỗ trợ RTOS và hoạt động ở tần số xung nhịp có thể điều chỉnh từ 80MHz đến 160 MHz.

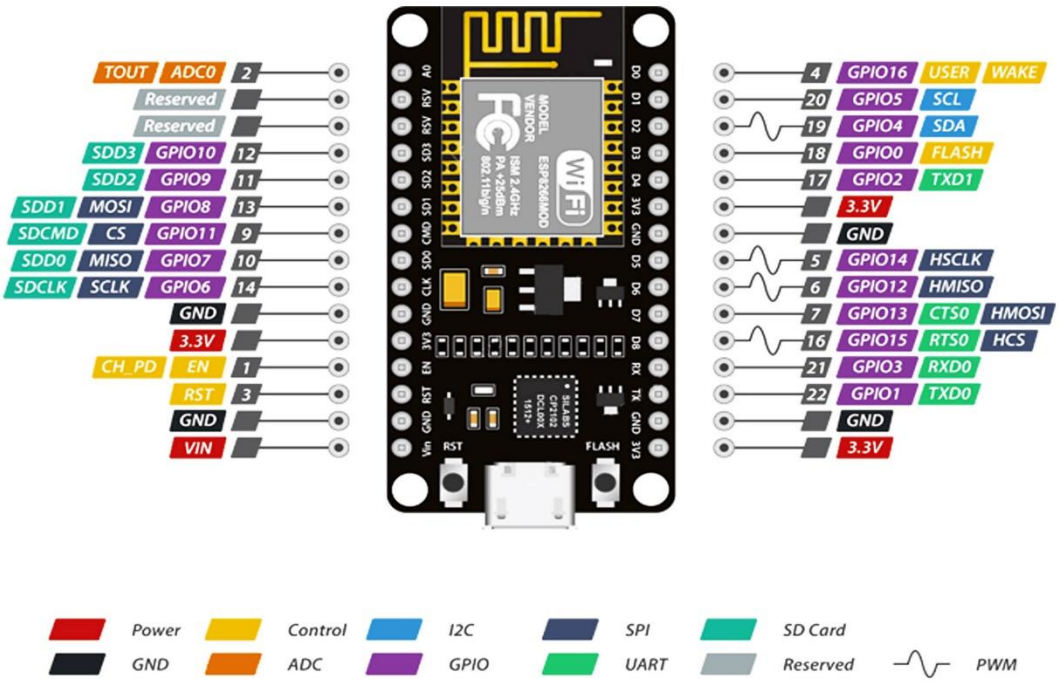
NodeMCU có 128 KB RAM và 4MB bộ nhớ Flash để lưu trữ dữ liệu và chương trình. Sức mạnh xử lý cao của nó với Wi-Fi / Bluetooth và các tính năng Điều hành Ngủ sâu tích hợp khiến nó trở nên lý tưởng cho các dự án IoT.

NodeMCU có thể được cấp nguồn bằng giắc cắm Micro USB và chân VIN (Chân nguồn cung cấp bên ngoài). Nó hỗ trợ giao diện UART, SPI và I2C.

Thông số kỹ thuật và tính năng NodeMCU ESP8266 :

- Bộ vi điều khiển: CPU RISC 32-bit Tensilica Xtensa LX106
- Điện áp hoạt động: 3.3V
- Điện áp đầu vào: 7-12V
- Chân I / O kỹ thuật số (DIO): 16
- Chân đầu vào tương tự (ADC): 1
- UARTs: 1
- SPI: 1
- I2Cs: 1
- Bộ nhớ Flash: 4 MB
- SRAM: 64 KB
- Tốc độ đồng hồ: 80 MHz
- USB-TTL dựa trên CP2102 được bao gồm trên bo mạch, cho phép Plug n Play
- Ăng-ten PCB

Thiết bị ngoại vi và chân I/O ESP8266 :



NodeMCU ESP8266 là một module IoT dựa trên vi điều khiển ESP8266. Nó được tích hợp sẵn các chân I/O (Input/Output) và hỗ trợ các thiết bị ngoại vi để kết nối và tương tác với các linh kiện và cảm biến khác. Dưới đây là một số thiết bị ngoại vi và các chân I/O quan trọng trên ESP8266 NodeMCU:

17 chân GPIO	Được sử dụng để đọc dữ liệu từ các cảm biến, điều khiển các thiết bị đầu ra, hoặc giao tiếp với các thiết bị khác như LED, động cơ, nút nhấn, v.v.
1 kênh ADC	1 kênh ADC có độ chính xác 10 bit theo công nghệ SAR ADC.
2 giao tiếp UART	2 giao tiếp UART hỗ trợ điều khiển dòng dữ liệu.
4 đầu ra PWM	4 chân PWM để điều khiển tốc độ động cơ hoặc độ sáng của đèn LED.
2 giao tiếp SPI và 1 giao tiếp I2C	2 giao tiếp SPI và một giao tiếp I2C để kết nối các cảm biến và thiết bị ngoại vi khác.
Giao tiếp I2S	Một giao tiếp I2S để thêm âm thanh vào dự án.

- **Chân GPIO :**

NodeMCU ESP8266 có tổng cộng 17 chân GPIO (General Purpose Input/Output) mà có thể sử dụng để đọc dữ liệu từ các cảm biến hoặc điều khiển các thiết bị khác. Mỗi GPIO có thể được cấu hình bên trong ở mức HIGH hoặc LOW.

- **Chân ADC :**

ESP8266 NodeMCU có một chân ADC (Analog-to-Digital Converter) duy nhất, được ký hiệu là A0. Chân ADC này cho phép đọc giá trị Analog từ các cảm biến hoặc linh kiện có đầu ra Analog.

- **Chân SPI :**

- ESP8266 có hai giao tiếp SPI, đó là SPI chính (SPI) và SPI phụ (HSPI). Cả hai giao thức SPI này hỗ trợ các tính năng và cấu hình sau:\
- Giao tiếp SPI cho phép bạn chọn từ 4 chế độ thời gian truyền dữ liệu khác nhau, cung cấp linh hoạt trong việc truyền và nhận dữ liệu qua SPI.
- ESP8266 hỗ trợ tốc độ truyền dữ liệu SPI lên đến 80 MHz, cho phép truyền dữ liệu nhanh chóng và hiệu quả.
- Xung clock SPI có thể được chia để tạo ra tần số hoạt động chính xác cho giao thức SPI.
- Cả SPI chính và SPI phụ đều hỗ trợ FIFO (First-In-First-Out) với bộ đệm 64 byte. Điều này giúp đảm bảo truyền dữ liệu liên tục và ổn định trong quá trình truyền và nhận thông tin qua SPI.

- **Chân I2C**

Phần cứng của ESP8266 không được tích hợp sẵn I2C, nhưng nó có thể được thực hiện bằng phương pháp ‘bitbanging’.

Theo mặc định, GPIO4 (SDA) và GPIO5 (SCL) được sử dụng làm chân giao tiếp I2C để giúp dễ dàng sử dụng các thư viện và code ví dụ Arduino.

- **Chân UART**

ESP8266 có hai giao tiếp UART, đó là UART0 và UART1, hỗ trợ giao tiếp không đồng bộ (RS232 và RS485) với tốc độ lên tới 4,5 Mbps.

Giao tiếp UART0 được sử dụng để truyền và nhận dữ liệu thông qua các chân TXD0 (Transmit Data 0), RXD0 (Receive Data 0), RST0 (Reset 0) và CTS0 (Clear To Send 0). Giao thức này thường được sử dụng để giao tiếp với các thiết bị ngoại vi hoặc kết nối với máy tính.

Giao tiếp UART1 chỉ có tín hiệu truyền dữ liệu thông qua chân TXD1 (Transmit Data 1). Thường được sử dụng để ‘printing logs’ hoặc gửi dữ liệu không đồng bộ.

- **Chân PWM**

Tất cả các chân GPIO từ GPIO0 đến GPIO15 trên ESP8266 đều có khả năng lập trình và sử dụng để điều chế độ rộng xung (PWM).

Trên ESP8266, tín hiệu PWM có độ phân giải 10 bit, tức là có thể có đến 1024 mức điều chỉnh khác nhau. Dải tần số của tín hiệu PWM có thể điều chỉnh từ khoảng 100 Hz đến 1 kHz, tương ứng với khoảng thời gian từ 1000 μs đến 10000 μs.

Điều này cho phép sử dụng các chân GPIO trên ESP8266 để điều khiển độ sáng của đèn LED, tốc độ động cơ và các ứng dụng khác sử dụng tín hiệu điều chế độ rộng xung.

- **Chân SDIO**

ESP8266 có một SDIO phụ để kết nối thẻ nhớ SD. Hỗ trợ SDIO v1.1 (4-bit 25 MHz) và SDIO v2.0 (4-bit 50 MHz).

- **Chân nguồn**

Chân VIN được sử dụng để cấp nguồn trực tiếp cho ESP8266 và các thiết bị ngoại vi.

Chân 3V3 là đầu ra được điều chỉnh từ IC ổn áp trên mạch.

GND là chân nối đất.

- **Chân ngắt (Interrupt)**

Tất cả các chân GPIO của NodeMCU ESP8266 có thể được cấu hình như ngắt, trừ GPIO16.

- **Chân điều khiển**

Chân EN (còn được gọi là CH_PD hoặc Chip Power Down) là chân kích hoạt cho ESP8266, mặc định được kéo lên mức cao. Khi kéo lên mức CAO, chip được kích hoạt; khi kéo xuống mức THẤP, chip bị vô hiệu hóa.

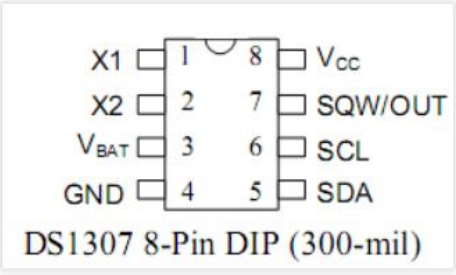
Chân RST là chân reset cho ESP8266, mặc định được kéo lên mức CAO. Khi kéo xuống mức THẤP trong một khoảng thời gian ngắn, nó sẽ khởi động lại ESP8266. Điều này giống như với việc nhấn nút RST trên bo mạch.

Chân FLASH được sử dụng để xác định khi nào khởi động vào chế độ nạp chương trình. Nếu chân này được giữ ở mức thấp trong quá trình khởi động, nó sẽ bắt đầu quá trình nạp chương trình! Điều này tương đương với việc nhấn nút FLASH trên bo mạch.

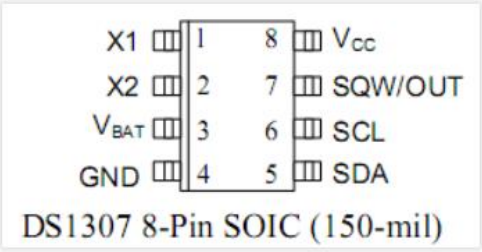
Chân WAKE được sử dụng để đánh thức ESP8266 từ chế độ ngủ sâu (deep sleep).

I.2. RTC DS1307

DS1307 là chip đồng hồ thời gian thực (RTC : Real-time clock), khái niệm thời gian thực ở đây được dùng với ý nghĩa thời gian tuyệt đối mà con người đang sử dụng, tình bằng giây, phút, giờ…DS1307 là một sản phẩm của Dallas Semiconductor (một công ty thuộc Maxim Integrated Products). Chip này có 7 thanh ghi 8-bit chứa thời gian là: giây, phút, giờ, thứ (trong tuần), ngày, tháng, năm. Ngoài ra DS1307 còn có 1 thanh ghi điều khiển ngõ ra phụ và 56 thanh ghi trống có thể dùng như RAM. DS1307 được đọc và ghi thông qua giao diện nối tiếp I2C (TWI của AVR) nên cấu tạo bên ngoài rất đơn giản. DS1307 xuất hiện ở 2 gói SOIC và DIP có 8 chân như trong hình



DS1307 8-Pin DIP (300-mil)



DS1307 8-Pin SOIC (150-mil)

Các chân của DS1307 được mô tả như sau:

- X1 và X2: là 2 ngõ kết nối với 1 thạch anh 32.768KHz làm nguồn tạo dao động cho chip.

- VBAT: cực dương của một nguồn pin 3V nuôi chip.

- GND: chân mass chung cho cả pin 3V và Vcc.

- Vcc: nguồn cho giao diện I2C, thường là 5V và dùng chung với vi điều khiển. Chú ý là nếu Vcc không được cấp nguồn nhưng VBAT được cấp thì DS1307 vẫn đang hoạt động (nhưng không ghi và đọc được).

- SQW/OUT: một ngõ phụ tạo xung vuông (Square Wave / Output Driver), tần số của xung được tạo có thể được lập trình. Như vậy chân này hầu như không liên quan đến chức năng của DS1307 là đồng hồ thời gian thực, chúng ta sẽ bỏ trống chân này khi nối mạch.

- SCL và SDA là 2 đường giao xung nhịp và dữ liệu của giao diện I2C mà chúng ta đã tìm hiểu trong bài TWI của AVR.

I.3. LCD 1602 I2C

I.3.1 LCD 1602

Ngày nay, thiết bị hiển thị LCD 1602 (Liquid Crystal Display) được sử dụng trong rất nhiều các ứng dụng của VĐK. LCD 1602 có rất nhiều ưu điểm so với các dạng hiển thị khác như: khả năng hiển thị kí tự đa dạng (chữ, số, kí tự đồ họa); dễ dàng đưa vào mạch ứng dụng theo nhiều giao thức giao tiếp khác nhau, tiêu tốn rất ít tài nguyên hệ thống, giá thành rẻ,…

Thông số kĩ thuật của sản phẩm LCD 1602:

- Điện áp MAX : 7V

- Điện áp MIN : - 0,3V

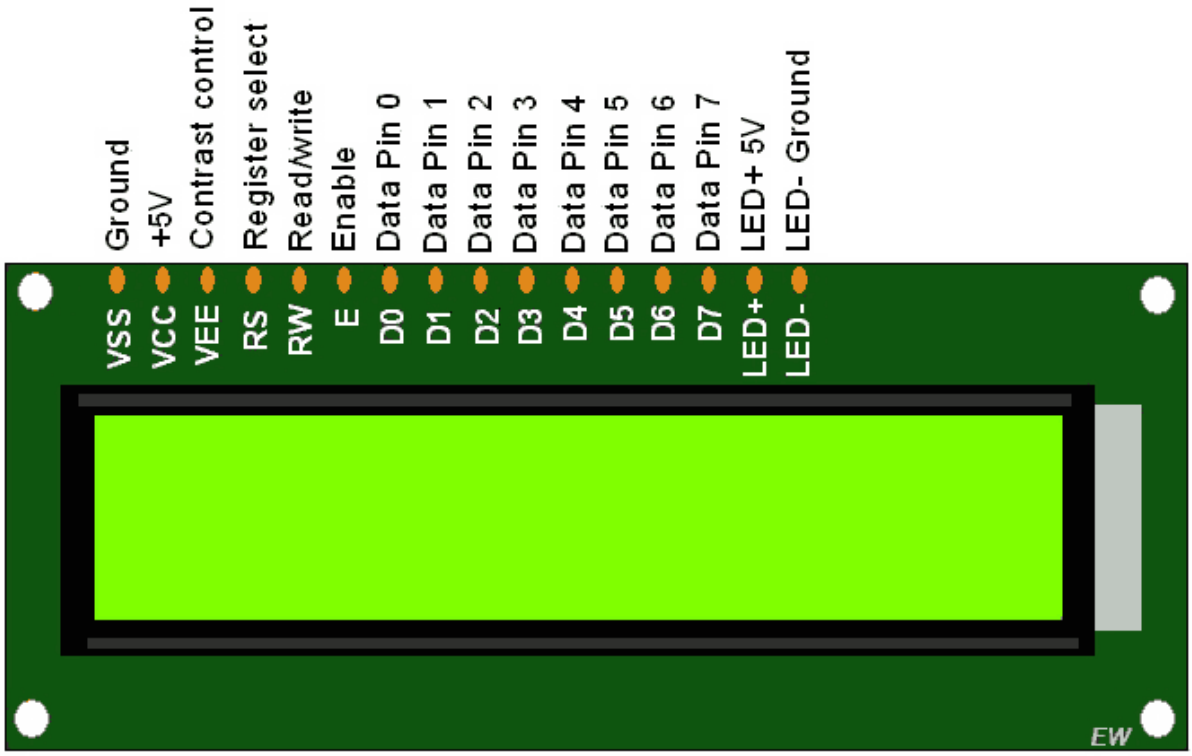
- Hoạt động ổn định : 2.7-5.5V

- Điện áp ra mức cao : > 2.4

- Điện áp ra mức thấp : <0.4V

- Dòng điện cấp nguồn : 350uA - 600uA

- Nhiệt độ hoạt động : - 30 - 75 độ C



Chức năng của từng chân LCD 1602:

- Chân số 1 - VSS : chân nối đất cho LCD được nối với GND của mạch điều khiển

- Chân số 2 - VDD : chân cấp nguồn cho LCD, được nối với VCC=5V của mạch điều khiển

- Chân số 3 - VE : điều chỉnh độ tương phản của LCD

- Chân số 4 - RS : chân chọn thanh ghi, được nối với logic "0" hoặc logic "1" :

+ Logic “0”: Bus DB0 - DB7 sẽ nối với thanh ghi lệnh IR của LCD (ở chế độ “ghi” - write) hoặc nối với bộ đếm địa chỉ của LCD (ở chế độ “đọc” - read)

+ Logic “1”: Bus DB0 - DB7 sẽ nối với thanh ghi dữ liệu DR bên trong LCD

- Chân số 5 - R/W : chân chọn chế độ đọc/ghi (Read/Write), được nối với logic “0” để ghi hoặc nối với logic “1” đọc

- Chân số 6 - E : chân cho phép (Enable). Sau khi các tín hiệu được đặt lên bus DB0-DB7, các lệnh chỉ được chấp nhận khi có 1 xung cho phép của chân này như sau:

+ Ở chế độ ghi: Dữ liệu ở bus sẽ được LCD chuyển vào thanh ghi bên trong khi phát hiện một xung (high-to-low transition) của tín hiệu chân E

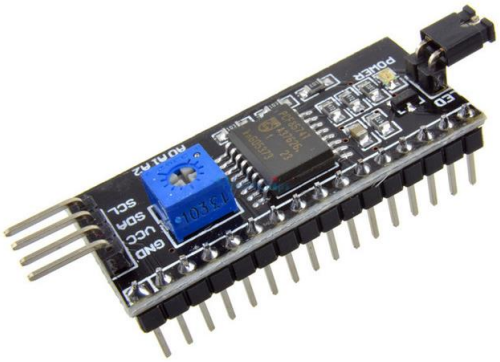
+ Ở chế độ đọc: Dữ liệu sẽ được LCD xuất ra DB0-DB7 khi phát hiện cạnh lên (low-to-high transition) ở chân E và được LCD giữ ở bus đến khi nào chân E xuống mức thấp

- Chân số 7 đến 14 - D0 đến D7: 8 đường của bus dữ liệu dùng để trao đổi thông tin với MPU. Có 2 chế độ sử dụng 8 đường bus này là: Chế độ 8 bit (dữ liệu được truyền trên cả 8 đường, với bit MSB là bit DB7) và Chế độ 4 bit (dữ liệu được truyền trên 4 đường từ DB4 tới DB7, bit MSB là DB7)

- Chân số 15 - A : nguồn dương cho đèn nền

- Chân số 16 - K : nguồn âm cho đèn nền

I.3.2 Module I2C Arduino



LCD có quá nhiều nhiều chân gây khó khăn trong quá trình đấu nối và chiếm dụng nhiều chân trên vi điều khiển.

Module I2C LCD ra đời và giải quyết vấn đề này.

Thay vì phải mất 6 chân vi điều khiển để kết nối với LCD 16×2 (RS, EN, D7, D6, D5 và D4) thì module IC2 bạn chỉ cần tốn 2 chân (SCL, SDA) để kết nối.

Module I2C hỗ trợ các loại LCD sử dụng driver HD44780(LCD 16×2, LCD 20×4, …) và tương thích với hầu hết các vi điều khiển hiện nay.

Ưu điểm

- Tiết kiệm chân cho vi điều khiển.
- Dễ dàng kết nối với LCD.

Thông số kĩ thuật

- Điện áp hoạt động: 2.5-6V DC.

- Hỗ trợ màn hình: LCD1602,1604,2004 (driver HD44780).
- Giao tiếp: I2C.
- Địa chỉ mặc định: 0X27 (có thể điều chỉnh bằng ngắn mạch chân A0/A1/A2).
- Tích hợp Jump chốt để cung cấp đèn cho LCD hoặc ngắt.
- Tích hợp biến trở xoay điều chỉnh độ tương phản cho LCD.

I.4. Light sensor

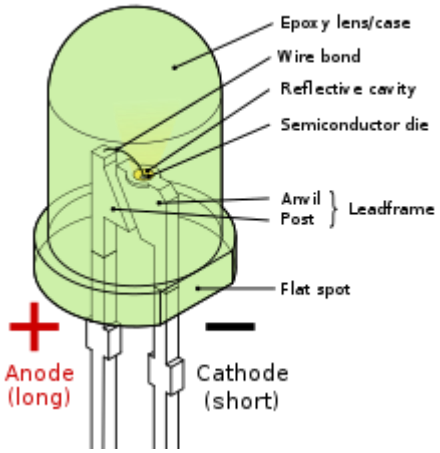
Quang trở còn được gọi là điện trở quang, photoresistor, photocell là một trong những linh kiện được tạo bằng một chất đặc biệt có thể thay đổi điện trở khi ánh sáng chiếu vào. Về cơ bản, bạn có thể hiểu nó là một tế bào quang điện được hoạt động dựa theo nguyên lý quang dẫn. Hay có thể hiểu nó là một điện trở có thể thay đổi được giá trị theo cường độ ánh sáng.

Quang trở được sử dụng nhiều trong các mạch cảm biến ánh sáng, đèn đường, báo động ánh sáng, đồng hồ ngoài trời,…

Cấu tạo của quang trở gồm 2 phần là phần trên và phần dưới là các màng kim loại được đấu nối với nhau thông qua các đầu cực. Linh kiện này được thiết kế theo cách cung cấp diện tích tiếp xúc tối đa nhất với 2 màng kim loại và được đặt trong một hộp nhựa có thể giúp tiếp xúc được với ánh sáng và có thể cảm nhận được sự thay đổi của cường độ ánh sáng.

Thành phần chính để tạo nên quang trở đó chính là Cadmium Sulphide (CdS) được sử dụng là chất quang dẫn, thường không chứa hoặc có rất ít các hạt electron khi không được ánh sáng chiếu vào.

I.5. LED



LED (viết tắt của light-emitting diode, có nghĩa là diode phát sáng hoặc diode phát quang) là các diode có khả năng phát ra ánh sáng hay tia hồng ngoại, tử ngoại. Cũng giống như diode, LED được cấu tạo từ một khối bán dẫn loại p ghép với một khối bán dẫn loại n.

I.6. Tụ

Tụ hóa là một loại linh kiện điện tử thụ động, có phân cực âm - dương, thường được sử dụng trong các mạch có tần số thấp hoặc dùng để lọc nguồn

Thông số kỹ thuật

- Điện dung: 100 uF
- Điện áp: 30V

- Nhiệt độ hoạt động: - 55°C -- 125°C
- Loại: Tụ phân cực

I.7. Điện trở

Điện trở hay Resistor là một linh kiện điện tử thụ động gồm 2 tiếp điểm kết nối, thường được dùng để hạn chế cường độ dòng điện chảy trong mạch, điều chỉnh mức độ tín hiệu, dùng để chia điện áp, kích hoạt các linh kiện điện tử chủ động như transistor, tiếp điểm cuối trong đường truyền điện và có trong rất nhiều ứng dụng khác. Điện trở công suất có thể tiêu tán một lượng lớn điện năng chuyển sang nhiệt năng có trong các bộ điều khiển động cơ, trong các hệ thống phân phối điện. Các điện trở thường có trở kháng cố định, ít bị thay đổi bởi nhiệt độ và điện áp hoạt động.

Thông số kỹ thuật:

- Model: 10K, 1K 1/4W
- Nhiệt độ hoạt động: -55oC – 155oC
- Linh kiện xuyên lỗ: 0.5mm
- Loại: Điện trở cố định

I.7. Nút bấm

Nút ấn là một loại công tắc đơn giản điều khiển hoạt động của máy hoặc một số loại quá trình. Hầu hết, các nút nhấn là nhựa hoặc kim loại. Hình dạng của nút ấn có thể phù hợp với ngón tay hoặc bàn tay để sử dụng dễ dàng. Tất cả phụ thuộc vào thiết kế cá nhân. Nút ấn có 2 loại chính là nút nhấn thường mở hoặc nút nhấn thường đóng.

Nguyên lí làm việc của nút nhấn :

Nút nhấn có ba phần: Bộ truyền động, các tiếp điểm cố định và các rãnh. Bộ truyền động sẽ đi qua toàn bộ công tắc và vào một xy lanh mỏng ở phía dưới. Bên trong là một tiếp điểm động và lò xo. Khi nhấn nút, nó chạm vào các tiếp điểm tĩnh làm thay đổi trạng thái của tiếp điểm. Trong một số trường hợp, người dùng cần giữ nút hoặc nhấn liên tục để thiết bị hoạt động. Với các nút nhấn khác, chốt sẽ giữ nút bật cho đến khi người dùng nhấn nút lần nữa.

CHƯƠNG II. CHƯƠNG TRÌNH VÀ THỰC THI

II.1. Sơ lược về UART và I2C

II.1.1. UART

Là chuẩn truyền nối tiếp không đồng bộ (Universal Asynchronous Receiver and Transmitter).

Thường được gọi nhanh là serial port trên máy tính/hệ nhúng.

Đóng khung dữ liệu và quy ước tốc độ truyền (thay cho tín hiệu clock).

Trong một sơ đồ giao tiếp UART:

1. Chân Tx (truyền) của một chip kết nối trực tiếp với chân Rx (nhận) của chip kia và ngược lại. Thông thường, quá trình truyền sẽ diễn ra ở 3.3V hoặc 5V. UART là một giao thức một master, một slave, trong đó một thiết bị được thiết lập để giao tiếp với duy nhất một thiết bị khác.

2. Dữ liệu truyền đến và đi từ UART song song với thiết bị điều khiển (ví dụ: CPU).

3. Khi gửi trên chân Tx, UART đầu tiên sẽ dịch thông tin song song này thành nối tiếp và truyền đến thiết bị nhận.

4. UART thứ hai nhận dữ liệu này trên chân Rx của nó và biến đổi nó trở lại thành song song để giao tiếp với thiết bị điều khiển của nó.

UART truyền dữ liệu nối tiếp, theo một trong ba chế độ:

- Full duplex: Giao tiếp đồng thời đến và đi từ mỗi master và slave
- Half duplex: Dữ liệu đi theo một hướng tại một thời điểm
- Simplex: Chỉ giao tiếp một chiều

Dữ liệu truyền qua UART được tổ chức thành các gói. Mỗi gói chứa 1 bit bắt đầu, 5 đến 9 bit dữ liệu (tùy thuộc vào UART), một bit chẵn lẻ tùy chọn và 1 hoặc 2 bit dừng.

UART là giao thức không đồng bộ, do đó không có đường clock nào điều chỉnh tốc độ truyền dữ liệu. Người dùng phải đặt cả hai thiết bị để giao tiếp ở cùng tốc độ. Tốc độ này được gọi là tốc độ truyền, được biểu thị bằng bit trên giây hoặc bps. Tốc độ

truyền thay đổi đáng kể, từ 9600 baud đến 115200 và hơn nữa. Tốc độ truyền giữa UART truyền và nhận chỉ có thể chênh lệch khoảng 10% trước khi thời gian của các bit bị lệch quá xa.

Mặc dù UART là giao thức cũ và chỉ có thể giao tiếp giữa một master và slave duy nhất, nhưng nó dễ thiết lập và cực kỳ linh hoạt.

Ưu và nhược điểm của UART

Không có giao thức truyền thông nào là hoàn hảo, nhưng UART thực hiện khá tốt công việc của nó. Dưới đây là một số ưu và nhược điểm để giúp bạn quyết định xem nó có phù hợp với nhu cầu của bạn hay không:

Ưu điểm :

- Chỉ sử dụng hai dây
- Không cần tín hiệu clock
- Có một bit chẵn lẻ để cho phép kiểm tra lỗi
- Cấu trúc của gói dữ liệu có thể được thay đổi miễn là cả hai bên đều được thiết lập cho nó
- Phương pháp có nhiều tài liệu và được sử dụng rộng rãi

Nhược điểm :

- Kích thước của khung dữ liệu được giới hạn tối đa là 9 bit
- Không hỗ trợ nhiều hệ thống slave hoặc nhiều hệ thống master
- Tốc độ truyền của mỗi UART phải nằm trong khoảng 10% của nhau

II.1.2. I2C

I2C kết hợp các tính năng tốt nhất của SPI và UART. Với I2C, có thể kết nối nhiều slave với một master duy nhất (như SPI) và có thể có nhiều master điều khiển một hoặc nhiều slave.

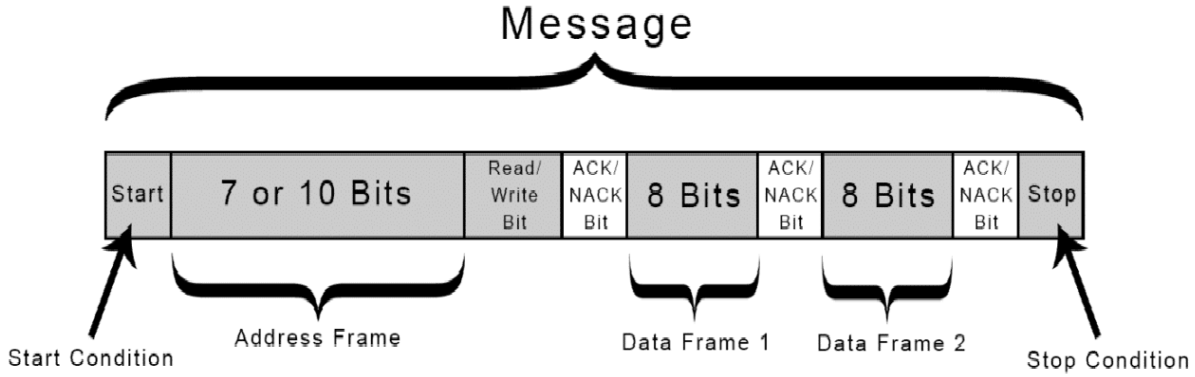
Giống như giao tiếp UART, I2C chỉ sử dụng hai dây để truyền dữ liệu giữa các thiết bị:

- SDA (Serial Data) - đường truyền cho master và slave để gửi và nhận dữ liệu.
- SCL (Serial Clock) - đường mang tín hiệu xung nhịp.
- I2C là một giao thức truyền thông nối tiếp, vì vậy dữ liệu được truyền từng bit dọc theo một đường duy nhất (đường SDA).

Giống như SPI, I2C là đồng bộ, do đó đầu ra của các bit được đồng bộ hóa với việc lấy mẫu các bit bởi một tín hiệu xung nhịp được chia sẻ giữa master và slave. Tín hiệu xung nhịp luôn được điều khiển bởi master.

Cách hoạt động của I2C

Với I2C, dữ liệu được truyền trong các tin nhắn. Tin nhắn được chia thành các khung dữ liệu. Mỗi tin nhắn có một khung địa chỉ chứa địa chỉ nhị phân của địa chỉ slave và một hoặc nhiều khung dữ liệu chứa dữ liệu đang được truyền. Thông điệp cũng bao gồm điều kiện khởi động và điều kiện dừng, các bit đọc / ghi và các bit ACK / NACK giữa mỗi khung dữ liệu:



- Điều kiện khởi động: Đường SDA chuyển từ mức điện áp cao xuống mức điện áp thấp trước khi đường SCL chuyển từ mức cao xuống mức thấp.
- Điều kiện dừng: Đường SDA chuyển từ mức điện áp thấp sang mức điện áp cao sau khi đường SCL chuyển từ mức thấp lên mức cao.
- Khung địa chỉ: Một chuỗi 7 hoặc 10 bit duy nhất cho mỗi slave để xác định slave khi master muốn giao tiếp với nó.
- Bit Đọc / Ghi: Một bit duy nhất chỉ định master đang gửi dữ liệu đến slave (mức điện áp thấp) hay yêu cầu dữ liệu từ nó (mức điện áp cao).
- Bit ACK / NACK: Mỗi khung trong một tin nhắn được theo sau bởi một bit xác nhận / không xác nhận. Nếu một khung địa chỉ hoặc khung dữ liệu được nhận thành công, một bit ACK sẽ được trả lại cho thiết bị gửi từ thiết bị nhận.

Địa chỉ

I2C không có các đường Slave Select như SPI, vì vậy cần một cách khác để cho slave biết rằng dữ liệu đang được gửi đến slave này chứ không phải slave khác. Nó thực hiện điều này bằng cách định địa chỉ. Khung địa chỉ luôn là khung đầu tiên sau bit khởi động trong một tin nhắn mới.

Master gửi địa chỉ của slave mà nó muốn giao tiếp với mọi slave được kết nối với nó. Sau đó, mỗi slave sẽ so sánh địa chỉ được gửi từ master với địa chỉ của chính nó. Nếu địa chỉ phù hợp, nó sẽ gửi lại một bit ACK điện áp thấp cho master. Nếu địa chỉ không khớp, slave không làm gì cả và đường SDA vẫn ở mức cao.

Bit đọc / ghi

Khung địa chỉ bao gồm một bit duy nhất ở cuối tin nhắn cho slave biết master muốn ghi dữ liệu vào nó hay nhận dữ liệu từ nó. Nếu master muốn gửi dữ liệu đến slave, bit đọc / ghi ở mức điện áp thấp. Nếu master đang yêu cầu dữ liệu từ slave, thì bit ở mức điện áp cao.

Khung dữ liệu

Sau khi master phát hiện bit ACK từ slave, khung dữ liệu đầu tiên đã sẵn sàng được gửi.

Khung dữ liệu luôn có độ dài 8 bit và được gửi với bit quan trọng nhất trước. Mỗi khung dữ liệu ngay sau đó là một bit ACK / NACK để xác minh rằng khung đã được nhận thành công. Bit ACK phải được nhận bởi master hoặc slave (tùy thuộc vào cái nào đang gửi dữ liệu) trước khi khung dữ liệu tiếp theo có thể được gửi.

Sau khi tất cả các khung dữ liệu đã được gửi, master có thể gửi một điều kiện dừng cho slave để tạm dừng quá trình truyền. Điều kiện dừng là sự chuyển đổi điện áp từ thấp lên cao trên đường SDA sau khi chuyển tiếp từ thấp lên cao trên đường SCL , với đường SCL vẫn ở mức cao.

Các bước truyền dữ liệu I2C :

- Master gửi điều kiện khởi động đến mọi slave được kết nối bằng cách chuyển đường SDA từ mức điện áp cao sang mức điện áp thấp trước khi chuyển đường SCL từ mức cao xuống mức thấp.
- Master gửi cho mỗi slave địa chỉ 7 hoặc 10 bit của slave mà nó muốn giao tiếp, cùng với bit đọc / ghi.
- Mỗi slave sẽ so sánh địa chỉ được gửi từ master với địa chỉ của chính nó. Nếu địa chỉ trùng khớp, slave sẽ trả về một bit ACK bằng cách kéo dòng SDA xuống thấp cho một bit. Nếu địa chỉ từ master không khớp với địa chỉ của slave, slave rời khỏi đường SDA cao.
- Master gửi hoặc nhận khung dữ liệu.
- Sau khi mỗi khung dữ liệu được chuyển, thiết bị nhận trả về một bit ACK khác cho thiết bị gửi để xác nhận đã nhận thành công khung.
- Để dừng truyền dữ liệu, master gửi điều kiện dừng đến slave bằng cách chuyển đổi mức cao SCL trước khi chuyển mức cao SDA.

Ưu điểm và nhược điểm của I2C

Có rất nhiều điều ở I2C có thể khiến nó nghe có vẻ phức tạp so với các giao thức khác, nhưng có một số lý do chính đáng khiến bạn có thể muốn hoặc không muốn sử dụng I2C để kết nối với một thiết bị cụ thể:

Ưu điểm

- Chỉ sử dụng hai dây
- Hỗ trợ nhiều master và nhiều slave
- Bit ACK / NACK xác nhận mỗi khung được chuyển thành công
- Phần cứng ít phức tạp hơn so với UART
- Giao thức nổi tiếng và được sử dụng rộng rãi

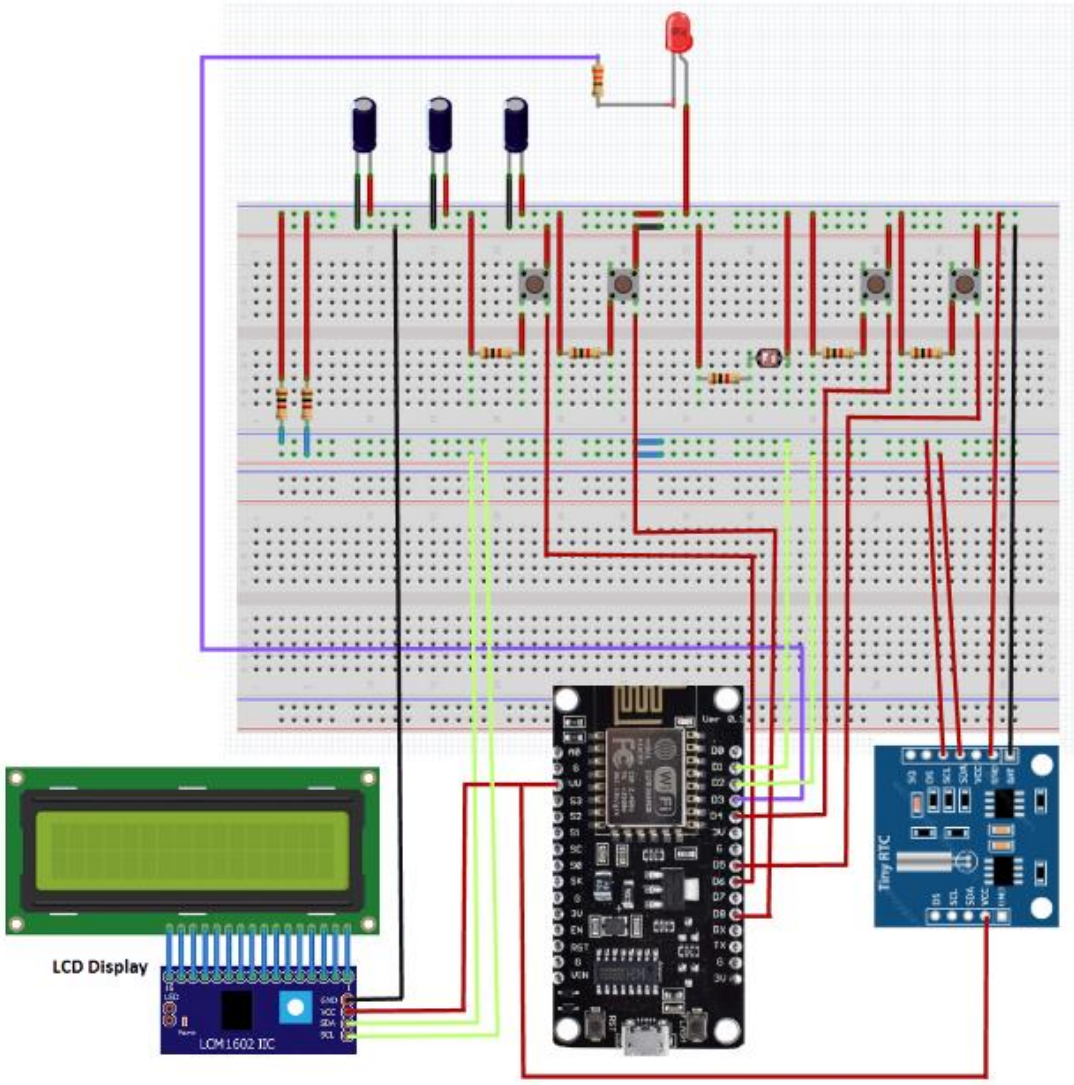
Nhược điểm

- Tốc độ truyền dữ liệu chậm hơn SPI
- Kích thước của khung dữ liệu bị giới hạn ở 8 bit
- Cần phần cứng phức tạp hơn để triển khai so với SPI

Chương III: Thiết kế mạch và Lập trình

III.1 Thiết kế mạch

Sơ đồ mạch



- ESP 8266 Node MCU: Điều khiển hoạt động mạch qua các cổng và cấp nguồn cho mạch hoạt động (nguồn từ USB máy tính kết nối với ESP 8266).
- LED1602 I2C: Hiển thị thời gian thực, màn hình tương tác giữ người dùng và hệ thống.
- Tiny RTC: Module thời gian thực.
- Nút bấm: điều chỉnh thời gian (giờ và phút) bật tắt (Sử dụng các chân D4, D5, D6, D8).
- Đèn led: Sử dụng tín hiệu từ cổng D3.
- Quang trở: Cảm biến ánh sáng (truyền về ESP 8266 qua cổng analog A0).
- Tụ và điện trở.

III.2 Lập trình

- **Khai báo thư viện, các chân vào các biến**

```
#include <LiquidCrystal_I2C.h>
#include "RTClib.h"
LiquidCrystal_I2C lcd(0x27, 16, 2);
const int analogInPin = A0; // ESP8266 Analog Pin ADC0 = A0

int sensorValue = 0; // value read from the pot

int ledPin = D3; //pin of led

int t = 0; // flag to check status on or off

//set pin and initial to time turn on led
int setOnHourPin = D6;
int setOnMinPin = D8;
int setOnHour = 0;
int setOnMin = 0;

//set pin and initial to time turn off led
int setOffHourPin = D4;
int setOffMinPin = D5;
int setOffHour = 0;
int setOffMin = 0;

RTC_DS1307 rtc;
```

Hàm : ***void setup()***

- **Khởi tạo LCD, in ra màn hình lời giới thiệu**

```
void setup()
{
    lcd.init();
    lcd.backlight();
    lcd.print("Embbeded Project");
    lcd.setCursor(0, 1);
    lcd.print("Group4: Tien Hai");
    delay(500);
    lcd.setCursor(0, 0);
    lcd.print("                ");
    lcd.setCursor(0, 1);
    lcd.print("                ");
```

- Khai báo, khởi tạo Serial và bật các chân nút bấm ở chế độ input, đèn LED ở chế độ output. Khởi tạo giá trị thời gian cho rtc

```
40      // initialize serial communication at 115200
41      Serial.begin(115200);
42
43      //set pin mode to timer turn on led
44      pinMode(setOnHourPin, INPUT);
45      pinMode(setOnMinPin, INPUT);
46
47      //set pin mode to timer turn off led
48      pinMode(setOffHourPin, INPUT);
49      pinMode(setOffMinPin, INPUT);
50
51      //set pin mode of led pid
52      pinMode(ledPin, OUTPUT);
53      if (!rtc.begin())
54      {
55          Serial.print("Couldn't find RTC");
56          while (1)
57              ;
58      }
59
60      if (!rtc.isrunning())
61      {
62          Serial.print("RTC is NOT running!");
63          Serial.println();
64      }
65      rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
66      // rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
67  }
```

Hàm : *void loop()*

- Trong loop() thì đọc giá trị thời gian thực từ rtc và đọc light sensor value được chuyền vào từ cổng analog A0

```
69  void loop()
70  {
71      // read the analog in value
72      sensorValue = analogRead(analogInPin);
73      DateTime now = rtc.now();
```

- In ra màn hình LCD thời gian thực ở thời điểm hiện tại

```
74      // Print to LCD current time
75      lcd.setCursor(0, 0);
76      lcd.print("Cur time: ");
77      if (now.hour() <= 9)
78      {
79          lcd.print("0");
80          lcd.print(now.hour());
81      }
82      else
83      {
84          lcd.print(now.hour());
85      }
86      lcd.print(':');
87      if (now.minute() <= 9)
88      {
89          lcd.print("0");
90          lcd.print(now.minute());
91      }
92      else
93      {
94          lcd.print(now.minute);
95      }
```

- In ra màn hình LCD thời gian hẹn giờ bật đèn, logic điều chỉnh thời gian bật

```
96      // print to LCD ON time
97      lcd.setCursor(0, 1);
98      lcd.print("ON");
99      if (setOnHour <= 9)
100     {
101         lcd.print("0");
102         lcd.print(setOnHour);
103     }
104     else
105     {
106         lcd.print(setOnHour);
107     }
108     lcd.print(':');
109     if (setOnMin <= 9)
110     {
111         lcd.print("0");
112         lcd.print(setOnMin);
113     }
114     else
115     {
116         lcd.print(setOnMin);
117     }
118     //Set ON time
119     if(digitalRead(setOnHourPin)==LOW){
120       setOnHour++;
121       if(setOnHour > 23) setOnHour = 0;
122     }
123
124     if(digitalRead(setOnMinPin)==LOW){
125       setOnMin ++;
126       if(setOnMin > 59) setOnMin = 0;
127     }
```

- In ra màn hình LCD thời gian hẹn giờ tắt, logic điều chỉnh thời gian tắt.

```
//Print to LCD OFF Time

    lcd.setCursor(8, 1);
    lcd.print("OFF");
    if (setOffHour <= 9)
    {
        lcd.print("0");
        lcd.print(setOffHour);
    }
    else
    {
        lcd.print(setOffHour);
    }
    lcd.print(':');
    if (setOffMin <= 9)
    {
        lcd.print("0");
        lcd.print(setOffMin);
    }
    else
    {
        lcd.print(setOffMin);
    }
    // Set OFF time
    if(digitalRead(setOffHourPin)==LOW){
      setOffHour++;
      if(setOffHour > 23) setOffHour = 0;
    }

    if(digitalRead(setOffMinPin)==LOW){
      setOffMin ++;
      if(setOffMin > 59) setOffMin = 0;
    }

```

- Logic bật, tắt đèn theo cảm biến và bộ hẹn giờ

```
// Logic
    // Turn ON/OFF depend on timer
    if(now.hour() == setOnHour && now.minute() == setOnMin){
      t=1;
    }

    }
    if(now.hour() == setOffHour && now.minute() == setOffMin){
      t=0;
    }

    if(t == 1){
      digitalWrite(ledPin, LOW);
    } else digitalWrite(ledPin, HIGH);

    // turn ON/OFF depend on sensor
    if(sensorValue > 700){
      digitalWrite(ledPin, HIGH);
    }
    if(sensorValue<=700){
      digitalWrite(ledPin, LOW);
    }
    delay(200);
}
```

KẾT LUẬN

Nhóm em đã hoàn thành được đề tài « Bật/tắt đèn sử dụng cảm biến ánh sáng và hẹn giờ », cũng như hệ thống lại và vận dụng được kiến thức về Hệ nhúng đã được học để ứng dụng thực tế.

Nghiên cứu và sử dụng kit ESP 8266 Node MCU, màn hình LCD1602 I2C và Module Tiny RTC Arduino, qua đó nắm thêm được nhiều kiến thức phục vụ cho việc học tập và nghiên cứu sâu hơn sau này.

Kết quả đạt được là sự cố gắng của cá nhân em và bạn cùng nhóm, giúp chúng em định hình được những công việc có thể ứng dụng Hệ nhúng trong thời điểm hiện tại và cách thức chúng hoạt động như thế nào. Chúng em rất cảm ơn sự giảng dạy và hướng dẫn của thầy trong suốt môn học. Điều này sẽ thúc đẩy đam mê học tập và nghiên cứu của chúng em về Hệ nhúng hiện tại và sau này hơn nữa.