**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA ĐIỆN – ĐIỆN TỬ  
**BỘ MÔN ĐIỆN TỬ**

------------O0O----------



**BÀI TẬP LỚN MÔN HỌC**

**THIẾT KẾ HỆ THỐNG NHÚNG**

**GVHD: Bùi Quốc Bảo**

**SINH VIÊN THỰC HIỆN**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **STT** | **MSSV** | **HỌ** | **TÊN** | **% ĐIỂM BTL** | **ĐIỂM BTL** | **GHI CHÚ** |
| 1 | 2210198 | Đặng Dương Gia | Bảo | 100% |  |  |
| 2 | 2211109 | Phan Đỗ Minh | Hoàng | 100% |  |  |
| 3 | 2213799 | Phạm Đức | Tuấn | 100% |  |  |

**Tp. Hồ Chí Minh, năm 2025**

**MỤC LỤC**

**[CHƯƠNG I – GIỚI THIỆU THIẾT BỊ](#_Toc184935311)** [3](#_Toc184935311)

**[I.Đặc tả hệ thống:](#_Toc184935312)** [3](#_Toc184935312)

**[II.Đặc tả kỹ thuật:](#_Toc184935313)** [5](#_Toc184935313)

**[III.Các vấn đề cơ bản của hệ thống:](#_Toc184935314)** [12](#_Toc184935314)

**[CHƯƠNG II – THIẾT KẾ](#_Toc184935320)** [28](#_Toc184935320)

**[I.Phần cứng:](#_Toc184935321)** [28](#_Toc184935321)

**[II.Phần mềm:](#_Toc184935323)** [30](#_Toc184935323)

**[CHƯƠNG III – THI CÔNG VÀ KẾT QUẢ](#_Toc184935324)** [32](#_Toc184935324)

**[I.Kết quả thi công mô hình:](#_Toc184935325)** [32](#_Toc184935325)

**[II.Kết quả thi công hệ thống:](#_Toc184935326)** [36](#_Toc184935326)

# **CHƯƠNG I – GIỚI THIỆU THIẾT BỊ**

## **I.Đặc tả hệ thống:**

1. Tên: Product Counter - Bộ đếm sản phẩm.

2. Mục đích:

* *Quản lý sản xuất chính xác***:** Giúp theo dõi số lượng sản phẩm đã hoàn thành trong từng ca làm việc hoặc theo ngày, hạn chế sai sót do đếm thủ công.
* *Kiểm soát chất lượng*: Nếu có lỗi sản phẩm, có thể truy vết theo mốc thời gian và số lượng đã đếm.
* *Tối ưu hóa hiệu suất làm việc*: Giúp đánh giá hiệu suất của máy móc, công nhân.
* *Hiển thị kết quả đếm*: Hiển thị số lượng đếm trên LCD.

3. Ngõ vào/ra:

* Ngõ vào:
* Nguồn 9V
* Ngõ ra:
* LCD 16x2: Hiển thị số sản phẩm đã được đếm.

4. Hiệu năng:

* Có sử dụng LCD để hiển thị kết quả một cách rõ ràng và dễ đọc, giúp người dùng dễ dàng theo dõi và phân tích số liệu.
* Có thể đếm và hiển thị đồng thời số lượng sản phẩm

5. Công suất:

* Dùng nguồn pin 9V

6. Kích thước/cân nặng:

* Kích thước khoảng 165x85x10, cân nặng khoảng 100g.

7. Lưu ý khác:

* Để nơi khô ráo thoáng mát, tránh ánh sáng trực tiếp.
* Không để gần lửa tránh cháy nổ.

## **II. Đặc tả kỹ thuật:**

1. Nguyên lý hoạt động:

* Một cảm biến quang được đặt sao cho mỗi khi sản phẩm đi qua, tia hồng ngoại bị chặn lại.
* Khi tín hiệu đầu ra của cảm biến chuyển từ HIGHsang LOW, điều này được hiểu là có một sản phẩm vừa đi qua.
* Tín hiệu này được đưa vào chân ngắt ngoài (INT0 hoặc INT1) của ATmega328P để xử lý tức thời, tránh bỏ sót do trễ thời gian.
* Khi xảy ra ngắt, hàm xử lý ngắt (ISR - Interrupt Service Routine) sẽ được gọi.
* Trong ISR, một biến đếm toàn cục (counter) sẽ được tăng lên 1 đơn vị.
* Để tránh đếm sai do nhiễu hoặc dao động cơ học, chương trình sẽ:
* Thêm trễ nhỏ (debounce delay) hoặc so sánh với thời gian đếm lần trước để loại bỏ xung sai.
* Cập nhật giá trị của biến counter lên màn hình LCD.
* Kiểm tra nút nhấn RESET, nếu có thì đặt counter = 0.

1. Cấu tạo:

* Cảm biến quang: Sử dụng nguyên lý phát hiện phản xạ hồng ngoại. Gồm một LED phát tia hồng ngoại và một phototransistor thu ánh sáng phản xạ từ vật thể. Khi có vật cản trong phạm vi từ 3 đến 80 cm, ánh sáng phản xạ được thu lại và mạch bên trong sẽ kích hoạt tín hiệu đầu ra mức thấp (LOW). Đầu ra dạng NPN hở cực cần được kéo lên mức logic bằng điện trở pull-up để tương thích với vi điều khiển như ATmega328P. Cảm biến có chiết áp để điều chỉnh khoảng cách phát hiện và hoạt động tốt với nguồn 5V hoặc 12V tuỳ model.
* ATmega328P: Là vi điều khiển 8-bit của hãng Microchip (trước đây là Atmel), thuộc họ AVR. Có 32KB bộ nhớ Flash, 2KB SRAM, 1KB EEPROM và hoạt động tối đa ở xung nhịp 20 MHz (thường sử dụng 16 MHz trong các ứng dụng Arduino). Cung cấp 23 chân I/O kỹ thuật số (trong đó có 6 chân hỗ trợ PWM), 6 kênh ADC 10-bit, và 3 kiểu ngắt ngoài. ATmega328P hỗ trợ nhiều giao tiếp ngoại vi như UART, SPI và I2C.
* Màn hình hiển thị hoặc thiết bị giao tiếp: Kết nối với Atmega328P để hiển thị giá trị đếm được (màn hình LCD).

1. Quy trình hoạt động:

* Khởi động hệ thống: Vi điều khiển (ATmega328P) khởi tạo các ngoại vi: ADC, ngắt ngoài, I/O và màn hình hiển thị.
* Giám sát cảm biến: Hệ thống liên tục theo dõi tín hiệu từ cảm biến quang.
* Phát hiện vật thể: Khi sản phẩm đi qua, cảm biến phát hiện vật thể thông qua ánh sáng hồng ngoại phản xạ. Tín hiệu từ cảm biến chuyển trạng thái (thường từ HIGH → LOW). Tín hiệu này được đưa vào chân ngắt ngoài hoặc được kiểm tra trong vòng lặp chính.
* Tăng bộ đếm: Khi phát hiện có sản phẩm, biến đếm (counter) được tăng thêm 1.
* Cập nhật hiển thị: Sau mỗi lần tăng, giá trị của biến đếm được cập nhật lên màn hình LCD.
* Xử lý nút điều khiển: Nếu người dùng nhấn nút RESET, bộ đếm sẽ quay về 0.

1. Môi trường hoạt động:
2. Điện áp và nguồn cấp

* Nguồn vào hệ thống: 5V DC (ổn định) cho ATmega328P và các thiết bị ngoại vi.
* Nguồn cho cảm biến: 5V–12V DC tuỳ loại (ví dụ E18-D80NK dùng 5V hoặc 12V).
* Cần nguồn ổn định, chống nhiễu để tránh sai lệch tín hiệu đếm.

1. Nhiệt độ môi trường

* Dải nhiệt độ hoạt động: 0°C đến 50°C (thích hợp với hầu hết môi trường nhà xưởng).
* Tránh lắp đặt ở khu vực có nhiệt độ quá cao, hơi nước hoặc bụi dầu gây ảnh hưởng đến cảm biến IR.

1. Độ ẩm

* Hoạt động ổn định trong độ ẩm tương đối 20% – 80%.
* Nên tránh các môi trường ẩm ướt cao hoặc có khả năng ngưng tụ sương.

1. Ánh sáng và nhiễu quang học

* Cảm biến hồng ngoại có thể bị ảnh hưởng bởi ánh sáng mạnh trực tiếp (như ánh sáng mặt trời).
* Khuyến cáo tránh đặt cảm biến hướng ra cửa sổ hoặc nguồn sáng IR khác.

1. Rung động và va đập

* Hệ thống nên được lắp cố định, tránh va đập cơ học hoặc rung động mạnh làm sai lệch vị trí cảm biến hoặc chập chờn dây nối.

1. Vị trí lắp đặt

* Cảm biến nên được gắn ngang tầm chuyển động của sản phẩm.
* Khoảng cách tối ưu giữa cảm biến và vật thể: 5–60 cm, tùy theo loại và cấu hình.

1. Nhiễu điện từ (EMI)

* Tránh đặt gần các thiết bị có công suất cao (motor, biến tần, hàn điện), vì có thể gây nhiễu cho vi điều khiển hoặc cảm biến.
* Có thể bổ sung tụ lọc nguồn và chống nhiễu đường tín hiệu nếu cần thiết.

1. Sơ đồ khối hệ thống:

LCD 16X2

9V Supply

dsad

Resistor

Buzzer

UART

USB-B

Atmega328P

ADC

IR Sensor

Reset

1. Mô tả các khối chính:

|  |  |
| --- | --- |
| **Tên** | **Chức năng** |
| Pin 9V | Cung cấp năng lượng cho hệ thống hoạt động |
| Atmega328P | Xử lí tín hiệu được đưa vào từ bộ ADC và trả giá trị ra LCD |
| LCD 16x2 | Hiển thị giá trị đếm được |
| Nút nhấn | Reset bộ đếm về 0 |
| Còi | Phát ra âm thanh khi vật đi qua cảm biến và khi nhấn nút Reset |
| Điện trở | Điện trở hạn dòng bảo vệ LCD |
| Biến trở | Điều chỉnh độ sáng của màn LCD |
| USB-B | Nạp dữ liệu từ máy tính bằng giao thức UART |

1. Phân chia phần cứng, phần mềm:
   1. Phần cứng:

* Pin 9V
* LCD 16x2
* Atmega328p trên Arduino Uno
* Nút nhấn
* Còi
* Điện trở và biến trở
* USB-B
  1. Software:

Để lập trình và triển khai các bộ đếm sử dụng Atmega328P, bạn có thể sử dụng các phần mềm phát triển tích hợp (IDE) và công cụ hỗ trợ sau:

* *Arduino IDE*

Arduino IDE là một môi trường phát triển phổ biến và dễ sử dụng cho Atmega328P. Nó cung cấp các thư viện và công cụ cần thiết để lập trình và nạp mã vào Atmega328P.

Ưu điểm: Giao diện thân thiện, hỗ trợ nhiều thư viện, cộng đồng người dùng lớn.

* *Thư viện hỗ trợ*

Thư viện cảm biến: Các thư viện như EEPROM, LiquidCrystal giúp dễ dàng tích hợp và sử dụng các cảm biến.

Thư viện giao tiếp: Các thư viện giao tiếp giúp Atmega328P kết nối và truyền dữ liệu.

**III. Các vấn đề của hệ thống**

1. **Bộ đếm**

* Constraints:

Bộ đếm thường có một số thông số ràng buộc (constraints) để đảm bảo tính chính xác, an toàn và hiệu quả khi sử dụng:

* Giới hạn bộ đếm: Cần chọn đúng kiểu dữ liệu để tránh tràn số nếu đếm số lượng lớn.
* Tốc độ đếm: Nếu sản phẩm đi qua quá nhanh, có thể gây bỏ sót đếm nếu không xử lý ngắt đúng cách hoặc không có debounce tốt, hệ thống cần xử lý được các sản phẩm đi qua liên tục cách nhau vài chục mili-giây.
* Khoảng cách cảm biến: Giới hạn khoảng cách phát hiện: 3–80 cm, nếu sản phẩm nhỏ hoặc màu tối, độ phản xạ kém → có thể không được phát hiện.
* Công suất tiêu thụ (Power Consumption): Vì điện trở trong rất lớn mà dòng điện tiêu thụ lại rất nhỏ, dẫn đến công suất tiêu thụ cũng cực kì khiêm tốn, từ đó việc sử dụng bộ đo không tốn quá nhiều điện năng ( <10mW).
* Môi trường hoạt động: Ánh sáng mạnh, bụi, nhiệt độ cao hoặc nhiễu từ môi trường có thể ảnh hưởng đến cảm biến hồng ngoại, không nên đặt cảm biến gần cửa sổ, đèn chiếu mạnh hoặc quạt công nghiệp.
* Tương tác người dung: Giới hạn số lượng nút điều khiển nếu không bố trí giao diện rõ rang, nhấn nút liên tục hoặc bị dính nút có thể gây lỗi đếm hoặc reset sai thời điểm.
* Functional issue:
* Đếm sai số:

a) **Nguyên nhân**:

* Cảm biến không phát hiện được sản phẩm do vật thể quá nhỏ hoặc phản xạ kém.
* Tốc độ sản phẩm đi quá nhanh, hệ thống không kịp xử lý.
* Rung cơ khí hoặc nhiễu điện từ gây tín hiệu ảo.

b) **Giải pháp**:

Tối ưu vị trí lắp cảm biến, điều chỉnh độ nhạy.

Dùng ngắt ngoài để đảm bảo phản hồi nhanh.

Bổ sung delay chống rung (debounce) hoặc bộ lọc tín hiệu.

* + Không reset được bộ đếm:

a) **Nguyên nhân**:

 Nút nhấn bị lỗi phần cứng hoặc thiếu xử lý debounce.

 Không có logic kiểm tra đúng điều kiện reset trong code.

b) **Giải pháp**:

 Kiểm tra nút và wiring.

 Bổ sung delay phần mềm để xử lý chống dính nút.

 Xác nhận thao tác reset qua âm thanh hoặc hiển thị.

* + **Buzzer không hoạt động / kêu liên tục**

**a)Nguyên nhân**:

* Lỗi chân I/O hoặc transistor điều khiển.
* Logic điều khiển buzzer không đúng.
* Điều kiện cảnh báo bị kích sai (đếm sai hoặc tràn số).

**b)Giải pháp**:

* Kiểm tra mạch điều khiển buzzer và thay buzzer thử.
* Kiểm tra điều kiện kích buzzer trong chương trình.
* **Màn hình hiển thị sai hoặc không lên**

1. **Nguyên nhân**:
   * Sai chân kết nối LCD.
   * Mất nguồn màn hình.
   * Code không khởi tạo đúng thư viện màn hình.
2. **Giải pháp**:
   * Rà soát wiring.
   * Kiểm tra điện áp.
   * Thử màn hình với code đơn giản để xác nhận.

* Real-time system:

Hệ thống product counter được xem là một hệ thống thời gian thực (real-time system), cụ thể là soft real-time system. Product counter cần xử lý kịp thời khi sản phẩm đi qua cảm biến, nếu không sẽ mất sự kiện đếm. Tuy nhiên, không đến mức nguy hiểm nếu trễ vài mili-giây, vì chỉ ảnh hưởng đến độ chính xác, không gây hậu quả nghiêm trọng như trong hệ thống điều khiển máy bay hay y tế.

* Concurrent system
  + Concurrent system là hệ thống có thể xử lý nhiều tác vụ gần như đồng thời, dù vi điều khiển thực hiện tuần tự, nhưng thông qua: ngắt và đa nhiệm giả lập. Trong product counter có những tác vụ đồng thời như: đếm sản phẩm khi cảm biến phát hiện, hiển thị số lên LCD, phát âm thanh buzzer khi đếm, kiểm tra nút nhấn reset.
* Đặc điểm của hệ thống đồng thời:
* **Xử Lý Dữ Liệu Song Song**: Dữ liệu từ các cảm biến hông ngoại có thể được xử lý đồng thời để đảm bảo phản hồi nhanh chóng và chính xác.
* **Giao Tiếp Liên Tục**: Hệ thống có khả năng gửi và nhận dữ liệu qua mạng trong khi vẫn tiếp tục thực hiện các phép đo, giúp duy trì kết nối và cập nhật trạng thái.
* **Lợi ích của hệ thống đồng thời:**
* **Tăng Tốc Độ Phản Hồi**: Hệ thống có thể phản hồi ngay lập tức khi phát hiện vật thể đi qua, từ đó giúp đảm bảo số lượng sản phẩm và ngăn ngừa sự cố.
* **Giảm Thiểu Thời Gian Ngừng Hoạt Động**: Hệ thống có thể hoạt động liên tục mà không cần phải dừng lại để xử lý từng tác vụ, điều này rất quan trọng trong các ứng dụng công nghiệp.
* **Cải Thiện Hiệu Suất**: Hệ thống có thể thực hiện nhiều nhiệm vụ đồng thời, từ việc cảm biến và ghi lại dữ liệu, giúp tối ưu hóa hiệu suất tổng thể.
* **Kiến Trúc Hệ Thống:**
* **Vi Điều Khiển**: Sử dụng vi điều khiển mạnh để xử lý đồng thời nhiều tín hiệu từ cảm biến dòng điện.
* **Giao Thức Giao Tiếp**: Sử dụng các giao thức như MQTT, HTTP/REST để truyền tải dữ liệu giữa các cảm biến và máy chủ hoặc ứng dụng.
* Các ví dụ ứng dụng:
* Hệ thống dây chuyền đóng gói tự động: Đếm số lượng sản phẩm đã đóng gói (chai nước, hộp bánh, linh kiện...). Đảm bảo mỗi thùng/hộp chứa đúng số lượng quy định trước khi niêm phong.
* Hệ thống dây chuyền sản xuất công nghiệp: Theo dõi số lượng sản phẩm được sản xuất ra trong ca/kíp. Giám sát hiệu suất máy móc và lao động.
* Hệ thống thống kê kho: Đếm sản phẩm ra/vào kho theo thời gian thực. Tự động cập nhật số tồn kho mà không cần kiểm tra thủ công.
* **Công Nghệ và Nguyên Tắc Hoạt Động:**
* **Đa Tầng Kiến Trúc**: Hệ thống có thể được chia thành nhiều tầng, bao gồm cảm biến, vi điều khiển, và giao diện người dùng, cho phép xử lý đồng thời ở từng tầng.
* **Lập Trình Song Song**: Sử dụng các thư viện lập trình hỗ trợ đa luồng (như FreeRTOS cho Arduino) để thực hiện các tác vụ đồng thời mà không làm ảnh hưởng đến hiệu suất.
* **Thách Thức:**
* **Quản Lý Tài Nguyên**: Cần phải đảm bảo rằng các tài nguyên hệ thống (bộ nhớ, băng thông) được quản lý hiệu quả để tránh tình trạng tắc nghẽn.
* **Đồng Bộ Hóa Dữ Liệu**: Cần có các cơ chế đồng bộ hóa để đảm bảo rằng dữ liệu từ các kênh khác nhau được xử lý chính xác và không bị xung đột.

# **CHƯƠNG II – THIẾT KẾ**

**I. PHẦN CỨNG**

**1. Sờ đồ mạch:**

**A diagram of a circuit

AI-generated content may be incorrect.**

**II.PHẦN MỀM:**

1.Lưu đồ giải thuật:

A diagram of a product

AI-generated content may be incorrect.

*Lưu đồ giải thuật thiết kế*

2. Giải thích giải thuật:

- Xác định tên công tắc, nút nhấn(button).

- Viết chương trình con cho vòng lặp chính khi bấm nút RESET.

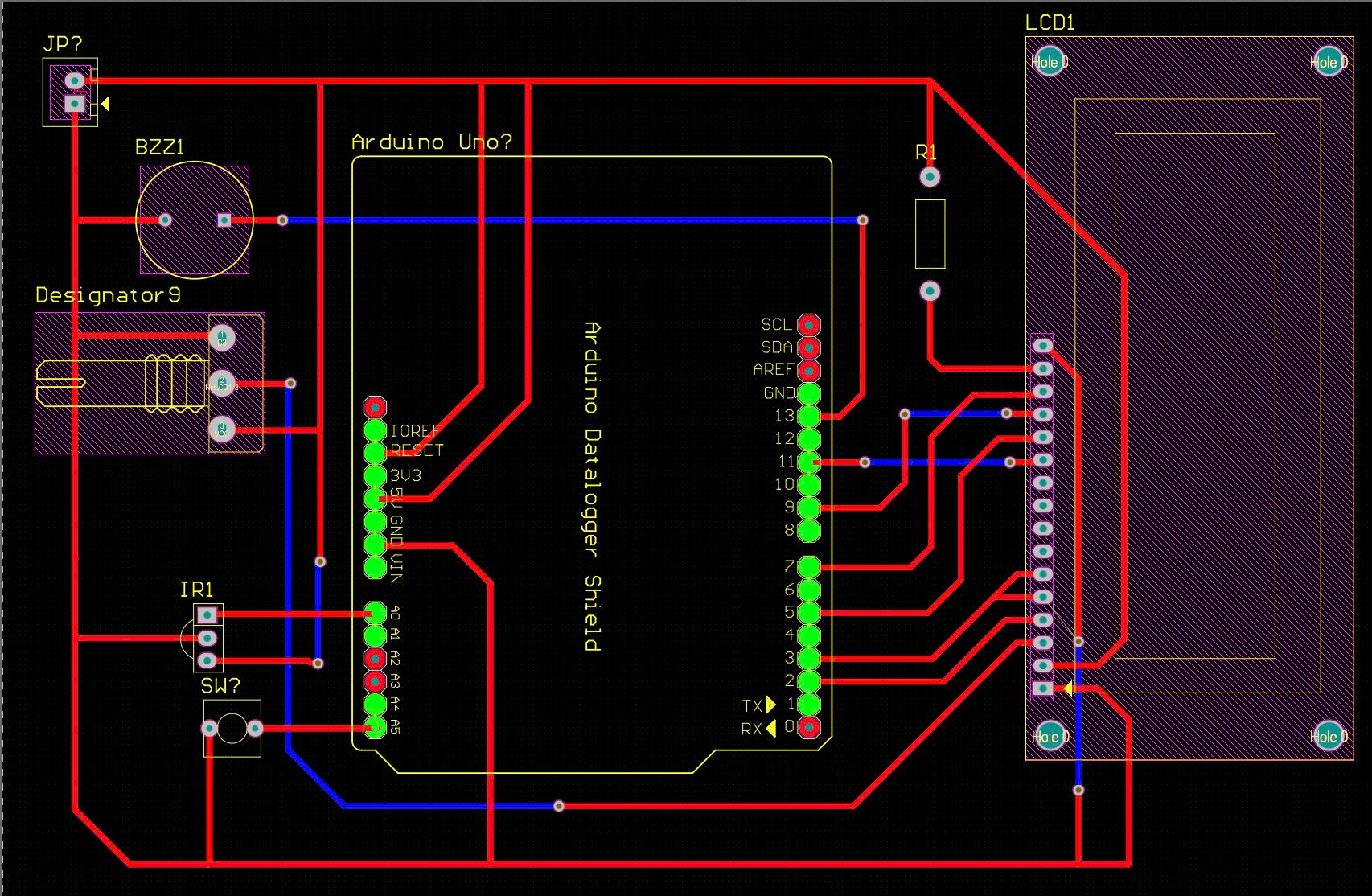
a. Khi màn hình mở:

* Xác định trạng thái hoạt động:
* Khi BUTTON chưa được nhấn: COUNT=0, thiết bị trạng thái ở chế độ 1, màn hình thể hiện giá trị cơ bản. Thiết bị chờ nhấn nút.
* Khi BUTTON được nhấn: tín hiệu reset sẽ được gửi về Atmega328P và truyền về buzzer và màn hình LCD.

# **CHƯƠNG III – THI CÔNG VÀ KẾT QUẢ**

**I.Kết quả thi công mô hình:**

1. Mô phỏng mạch:



1. Chương trình đếm sản phẩm, lập trình trên Arduino IDE:

#include <EEPROM.h>

#include <LiquidCrystal.h>

LiquidCrystal lcd(2,3,5,7,9,11);

#define sensor\_pin A0

#define reset\_pin A5

#define buzzer 13

int d5=1, d4=2, d3=3, d2=4, d1=5;

int flag=0, timer=0;

byte bar1[8] = {0x1C, 0x3E, 0x3E, 0x3E, 0x3E, 0x3E, 0x3E, 0x1C};

byte bar2[8] = {0x07, 0x1F, 0x1F, 0x1F, 0x1F, 0x1F, 0x07, 0x00};

byte bar3[8] = {0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF};

byte bar4[8] = {0xF0, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x30, 0x3C};

byte bar5[8] = {0x1F, 0x0F, 0x00, 0x00, 0x00, 0x00, 0x03, 0x07};

byte bar6[8] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF};

byte bar7[8] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1C, 0x3E};

byte bar8[8] = {0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};

void setup(){

pinMode(sensor\_pin, INPUT\_PULLUP);

pinMode(reset\_pin, INPUT\_PULLUP);

pinMode(buzzer, OUTPUT);

lcd.createChar(1,bar1);

lcd.createChar(2,bar2);

lcd.createChar(3,bar3);

lcd.createChar(4,bar4);

lcd.createChar(5,bar5);

lcd.createChar(6,bar6);

lcd.createChar(7,bar7);

lcd.createChar(8,bar8);

lcd.begin(16, 2);

delay(100);

lcd.clear();

lcd.setCursor (0,0);

lcd.print(" BTL ");

lcd.setCursor (0,1);

lcd.print(" Product Counter ");

delay(1000);

lcd.clear();

if(EEPROM.read(0)==0){

}else{

WriteEeprom();

EEPROM.write(0,0);

}

ReadEeprom();

}

void loop(){

if(digitalRead (sensor\_pin) == 1){

if(flag==0){flag=1;

digitalWrite(buzzer, HIGH);

d1=d1+1;

if(d1>9){d1=0; d2=d2+1;}

if(d2>9){d2=0; d3=d3+1;}

if(d3>9){d3=0; d4=d4+1;}

if(d4>9){d4=0; d5=d5+1;}

if(d5>9){d5=0;}

WriteEeprom();

}

}else{flag=0;}

if(digitalRead (reset\_pin) == 0){

digitalWrite(buzzer, HIGH);

if(timer<50){timer=timer+1;}

if(timer==50){

d1=0, d2=0, d3=0, d4=0, d5=0;

WriteEeprom();

}

}else{digitalWrite(buzzer, LOW); timer=0;}

lcd.setCursor(0,0);

lcd.print(flag);

printNumber(d5, 1);

printNumber(d4, 4);

printNumber(d3, 7);

printNumber(d2, 10);

printNumber(d1, 13);

delay(10);

}

void printNumber(int value, int col) {

if (value == 0) {

custom0(col);

} if (value == 1) {

custom1(col);

} if (value == 2) {

custom2(col);

} if (value == 3) {

custom3(col);

} if (value == 4) {

custom4(col);

} if (value == 5) {

custom5(col);

} if (value == 6) {

custom6(col);

} if (value == 7) {

custom7(col);

} if (value == 8) {

custom8(col);

} if (value == 9) {

custom9(col);

}

}

void custom0(int col)

{ lcd.setCursor(col, 0);

lcd.write(2);

lcd.write(8);

lcd.write(1);

lcd.setCursor(col, 1);

lcd.write(2);

lcd.write(6);

lcd.write(1);

}

void custom1(int col)

{

lcd.setCursor(col,0);

lcd.write(32);

lcd.write(32);

lcd.write(1);

lcd.setCursor(col,1);

lcd.write(32);

lcd.write(32);

lcd.write(1);

}

void custom2(int col)

{

lcd.setCursor(col,0);

lcd.write(5);

lcd.write(3);

lcd.write(1);

lcd.setCursor(col, 1);

lcd.write(2);

lcd.write(6);

lcd.write(6);

}

void custom3(int col)

{

lcd.setCursor(col,0);

lcd.write(5);

lcd.write(3);

lcd.write(1);

lcd.setCursor(col, 1);

lcd.write(7);

lcd.write(6);

lcd.write(1);

}

void custom4(int col)

{

lcd.setCursor(col,0);

lcd.write(2);

lcd.write(6);

lcd.write(1);

lcd.setCursor(col, 1);

lcd.write(32);

lcd.write(32);

lcd.write(1);

}

void custom5(int col)

{

lcd.setCursor(col,0);

lcd.write(2);

lcd.write(3);

lcd.write(4);

lcd.setCursor(col, 1);

lcd.write(7);

lcd.write(6);

lcd.write(1);

}

void custom6(int col)

{

lcd.setCursor(col,0);

lcd.write(2);

lcd.write(3);

lcd.write(4);

lcd.setCursor(col, 1);

lcd.write(2);

lcd.write(6);

lcd.write(1);

}

void custom7(int col)

{

lcd.setCursor(col,0);

lcd.write(2);

lcd.write(8);

lcd.write(1);

lcd.setCursor(col, 1);

lcd.write(32);

lcd.write(32);

lcd.write(1);

}

void custom8(int col)

{

lcd.setCursor(col, 0);

lcd.write(2);

lcd.write(3);

lcd.write(1);

lcd.setCursor(col, 1);

lcd.write(2);

lcd.write(6);

lcd.write(1);

}

void custom9(int col)

{

lcd.setCursor(col, 0);

lcd.write(2);

lcd.write(3);

lcd.write(1);

lcd.setCursor(col, 1);

lcd.write(7);

lcd.write(6);

lcd.write(1);

}

void ReadEeprom(){

d1=EEPROM.read(1);

d2=EEPROM.read(2);

d3=EEPROM.read(3);

d4=EEPROM.read(4);

d5=EEPROM.read(5);

}

void WriteEeprom(){

EEPROM.write(1, d1);

EEPROM.write(2, d2);

EEPROM.write(3, d3);

EEPROM.write(4, d4);

EEPROM.write(5, d5);

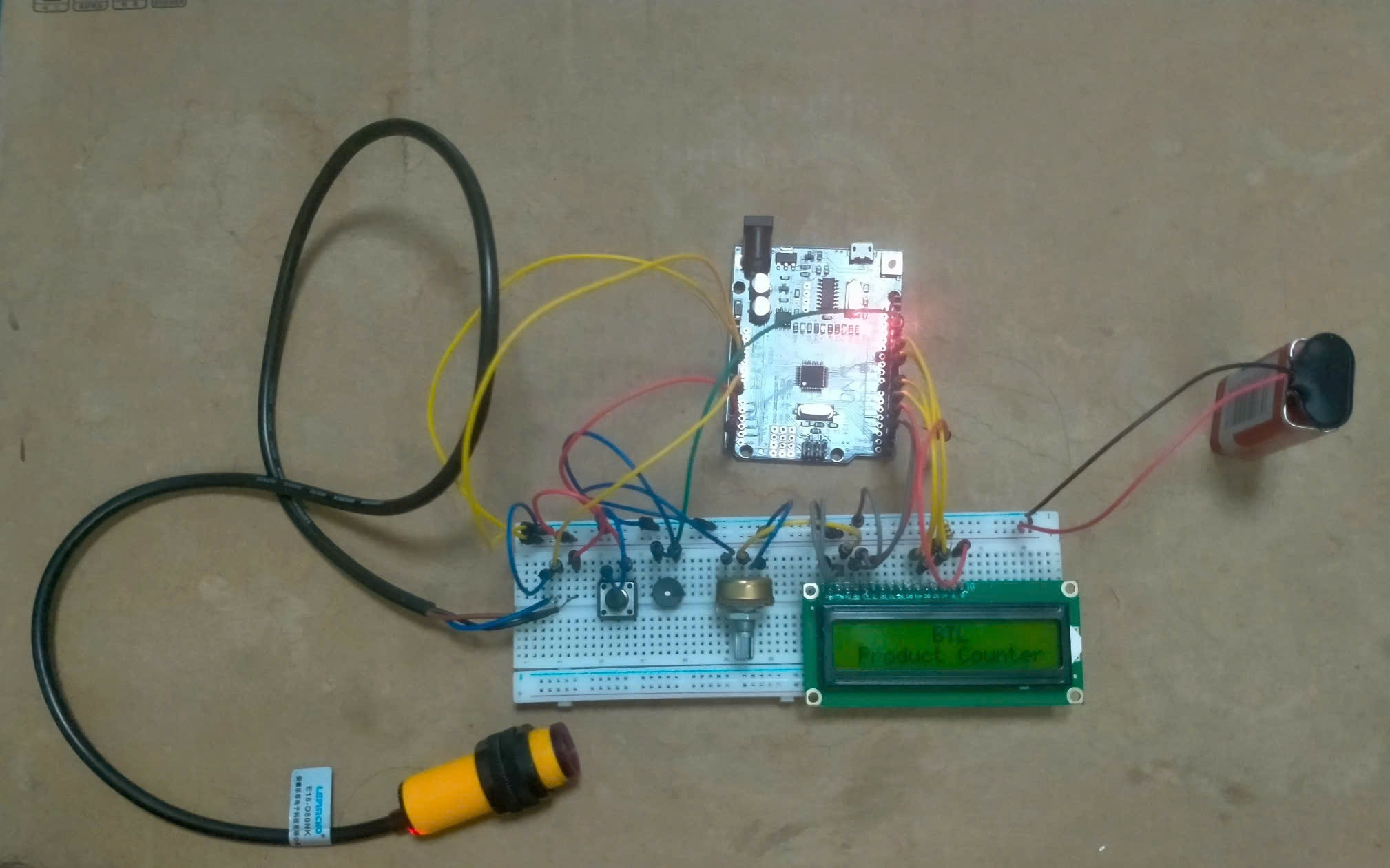
}

3. Hạn chế:

* Độ chính xác bị giới hạn: Nếu sản phẩm đi qua quá nhanh hoặc quá sát nhau, cảm biến IR có thể không phân biệt được .
* Có thể bị đếm trùng hoặc bỏ sót nếu không có cơ chế lọc nhiễu tốt.
* Không thể hoạt động trong những môi trường khắc nghiệt như nhiệt độ cao, độ ẩm lớn,…
* Giá thành cao hơn so với thị trường hiện tại.
* Hệ thống đơn giản chỉ đếm số lượng, không thể phân biệt kích thước, màu sắc hay loại sản phẩm khác nhau.

**II. Kết quả thi công**

Sản phẩm thực tế:



**TÀI LIỆU THAM KHẢO**

1.Atmega328p datasheet. Truy cập từ: <https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf>

2. Elec2PCB Tech-Solutions. Hướng dẫn vẽ mạch PCB. Truy cập từ: <https://www.youtube.com/channel/UC7b10JGIt5MlQKRLBMHR3eA>

3.Complete Guide to ATMEGA328P Microcontroller: https://www.allelcoelec.com/blog/Complete-Guide-to-ATMEGA328P-Microcontroller.html?srsltid=AfmBOoo7jb2HPJc-3Nfxv\_2RwpNyZnZ9o8kdy6mSlvgwEJTWair1Koh1