**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐIỆN - ĐIỆN TỬ**

**BỘ MÔN KỸ THUẬT MÁY TÍNH – VIỄN THÔNG**

A logo of hands holding a book and a candle

Description automatically generated

**ĐỒ ÁN MÔN HỌC 1**

**THIẾT KẾ MẠCH ĐÈN GIAO THÔNG**

**SVTH: HOÀNG NGỌC VĨNH PHÚC**

**MSSV: 19161150**

**Khóa: 2019**

**Ngành: CÔNG NGHỆ KỸ THUẬT ĐIỆN TỬ - VIỄN THÔNG**

**GVHD: ThS. LÊ MINH THÀNH**

TP. Hồ Chí Minh, tháng 6 năm 2024

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐIỆN - ĐIỆN TỬ**

**BỘ MÔN KỸ THUẬT MÁY TÍNH – VIỄN THÔNG**

A logo of hands holding a book and a candle

Description automatically generated

**ĐỒ ÁN MÔN HỌC 1**

**THIẾT KẾ MẠCH ĐÈN GIAO THÔNG**

**SVTH: HOÀNG NGỌC VĨNH PHÚC**

**MSSV: 19161150**

**Khóa: 2019**

**Ngành: CÔNG NGHỆ KỸ THUẬT ĐIỆN TỬ - VIỄN THÔNG**

**GVHD: ThS. LÊ MINH THÀNH**

TP. Hồ Chí Minh, tháng 6 năm 2024

# **LỜI CẢM ƠN**

Để có thể thực hiện và hoàn thành đề tài này, tôi xin gửi lời cảm ơn chân thành nhất đến ThS. Lê Minh Thành đã luôn quan tâm, tận tình hướng dẫn, nhắc nhở để tránh những sai sót trong suốt quá trình thực hiện. Bên cạnh đó, tôi cũng cảm ơn gia đình, bạn bè đã hỗ trợ, đưa ra những lời góp ý, động viên để tôi có thể hoàn thành báo cáo một cách tốt nhất.

Tuy nhiên trong quá trình thực hiện đồ án, do kiến thức còn hạn chế nên tôi vẫn còn nhiều sai sót trong quá trình tìm hiểu, phân tích và hoàn thiện báo cáo. Rất mong nhận được nhận xét, ý kiến đóng góp, phê bình từ giáo viên hướng dẫn để tôi cải thiện bài báo cáo, cũng là kinh nghiệm để cho tôi hoàn thiện bản thân hơn trong học tập và công việc.

Tôi xin chân thành cảm ơn!

TP. Hồ Chí Minh, tháng 6 năm 2024

Sinh viên thực hiện

Hoàng Ngọc Vĩnh Phúc

# **TÓM TẮT**

Với sự gia tăng không ngừng của phương tiện giao thông và sự phức tạp của mạng lưới đường phố, việc quản lý và điều tiết giao thông trở nên vô cùng quan trọng. Nhằm giảm thiểu tai nạn, giảm tắc nghẽn và nâng cao hiệu quả sử dụng hạ tầng giao thông, đề tài “*Thiết kế mạch đèn giao thông*” được thực hiện. Mục đích tạo ra hệ thống đèn giao thông hoạt động hiệu quả, góp phần vào việc đảm bảo an toàn giao thông và tối ưu hóa luồng xe cộ lưu thông trên đường.

Đề tài này thiết kế một hệ thống mạch đèn giao thông ngã tư với tín hiệu cho phép rẽ trái, có ba nút nhấn cho phép chuyển chế độ hoạt động của mạch là chế độ bình thường, chế độ giờ cao điểm và chế độ ban đêm, có một nút nhấn cho phép chuyển đổi thủ công giữa các trạng thái đèn giao thông khi hoạt động ở chế độ giờ cao điểm. Hệ thống đã được thử nghiệm thành công theo các yêu cầu ban đầu, cho thấy khả năng hoạt động hiệu quả và nâng cao an toàn giao thông.

# **DANH MỤC CÁC CHỮ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| Viết tắt | Tiếng Anh | Tiếng Việt |
| DC | Direct Current | Dòng điện một chiều |
| I/O | Input/Output | Đầu vào/Đầu ra |
| IC | Integrated Circuit | Mạch tích hợp |
| IoT | Internet of Things | Internet vạn vật |
| LED | Light-Emitting Diode | Diode phát quang |
| PWM | Pulse Width Modulation | Điều chế độ rộng xung |
| UART | Universal Asynchronous Receiver-Transmitter | Truyền dữ liệu nối tiếp bất đồng bộ |
| USB | Universal Serial Bus | Chuẩn kết nối và truyền dữ liệu số tuần tự |

# **DANH MỤC CÁC BẢNG BIỂU**

[**Bảng 2.1.** Bảng mô tả các bit lựa chọn xung nhịp 10](#_Toc169730758)

[**Bảng 3.1.** Bảng thông số kỹ thuật chính của Arduino UNO 19](#_Toc169730759)

# **DANH MỤC CÁC HÌNH ẢNH**

[**Hình 2.1.** Hệ thống truyền bất đồng bộ 6](#_Toc169755299)

[**Hình 2.2.** Ví dụ về quá trình truyền – nhận dữ liệu của UART 7](#_Toc169755301)

[**Hình 2.3.** Khung dữ liệu UART 8](#_Toc169755303)

[**Hình 2.4.** 2 thanh ghi điều khiển TCCR1 của Timer 1 9](#_Toc169755307)

[**Hình 2.5.** 2 thanh ghi so sánh OCR1 của Timer 1 11](#_Toc169755308)

[**Hình 2.6.** Thanh ghi mặt nạ ngắt TIMSK của Timer 1 11](#_Toc169755309)

[**Hình 2.7.** Thanh ghi cờ nhớ TIFR của Timer 1 12](#_Toc169755310)

[**Hình 3.1.** Sơ đồ khối tổng quát của hệ thống 13](#_Toc169755312)

[**Hình 3.2.** Sơ đồ nguyên lý toàn hệ thống 15](#_Toc169755313)

[**Hình 3.3.**Khối nút nhấn 15](#_Toc169755314)

[**Hình 3.4.**Khối hiển thị 17](#_Toc169755315)

[**Hình 3.5.**Khối xử lý trung tâm 18](#_Toc169755316)

[**Hình 3.6.** Khối nguồn 22](#_Toc169755317)

[**Hình 3.7.** Lưu đồ giải thuật chính 22](#_Toc169755318)

[**Hình 3.8.** Lưu đồ giải thuật mô tả chương trình con “dem\_thoi\_gian()” 23](#_Toc169755319)

[**Hình 3.9.** Lưu đồ giải thuật mô tả chương trình con “hienthi()” 25](#_Toc169755320)

[**Hình 3.10.**  Lưu đồ giải thuật mô tả chương trình con “kt\_nut\_nhan()” 26](#_Toc169755321)

[**Hình 3.11.**  Lưu đồ giải thuật mô tả chương trình con “kt\_chuyen()” 27](#_Toc169755322)

[**Hình 4.1.** Sơ đồ mạch in lớp dưới 28](#_Toc169755323)

[**Hình 4.2.** Mạch in lớp dưới của hệ thống 29](#_Toc169755324)

[**Hình 4.3.** Mạch khi mới cấp nguồn hoặc khi nhấn nút nhấn xanh 30](#_Toc169755325)

[**Hình 4.4.** Mạch sau khi nhấn nút nhấn đỏ 30](#_Toc169755326)

[**Hình 4.5.** Mạch sau khi nhấn nút nhấn đỏ 3 giây 31](#_Toc169755327)

[**Hình 4.6.** Mạch sau khi nhấn nút nhấn đen 31](#_Toc169755328)

[**Hình 4.7.** Mạch sau khi nhấn nút nhấn đen 3 giây 32](#_Toc169755329)

[**Hình 4.8.** Mạch khi nhấn nút nhấn vàng 32](#_Toc169755330)

# **MỤC LỤC**

[**LỜI CẢM ƠN** i](#_Toc169757097)

[**TÓM TẮT** ii](#_Toc169757098)

[**DANH MỤC CÁC CHỮ VIẾT TẮT** iii](#_Toc169757099)

[**DANH MỤC CÁC BẢNG BIỂU** iv](#_Toc169757100)

[**DANH MỤC CÁC HÌNH ẢNH** v](#_Toc169757101)

[**MỤC LỤC** vi](#_Toc169757102)

[**CHƯƠNG 1. TỔNG QUAN** 1](#_Toc169757103)

[**1.1. GIỚI THIỆU** 1](#_Toc169757104)

[**1.2. MỤC TIÊU** 1](#_Toc169757105)

[**1.3. GIỚI HẠN** 2](#_Toc169757106)

[**1.4. BỐ CỤC ĐỒ ÁN** 2](#_Toc169757107)

[**CHƯƠNG 2. CƠ SỞ LÝ THUYẾT** 4](#_Toc169757108)

[**2.1. LÝ THUYẾT CƠ BẢN VỀ MẠCH ĐÈN GIAO THÔNG** 4](#_Toc169757109)

[**2.2. CHUẨN GIAO TIẾP UART** 6](#_Toc169757110)

[**2.2.1. Giới thiệu** 6](#_Toc169757111)

[**2.2.2. Nguyên lý hoạt động** 7](#_Toc169757112)

[**2.2.3. Khung dữ liệu** 8](#_Toc169757113)

[**2.3. TIMER 1** 9](#_Toc169757114)

[**2.3.1. Giới thiệu** 9](#_Toc169757115)

[**2.3.2. Khảo sát các thanh ghi của Timer 1** 9](#_Toc169757116)

[**2.3.3. Các chế độ hoạt động của Timer 1** 12](#_Toc169757117)

[**CHƯƠNG 3. THIẾT KẾ** 13](#_Toc169757118)

[**3.1. Yêu cầu của hệ thống** 13](#_Toc169757119)

[**3.2. Sơ đồ khối** 13](#_Toc169757120)

[**3.3. Thiết kế phần cứng** 14](#_Toc169757121)

[**3.3.1. Khối nút nhấn** 15](#_Toc169757122)

[**3.3.2. Khối hiển thị** 16](#_Toc169757123)

[**3.3.3. Khối xử lý trung tâm** 18](#_Toc169757124)

[**3.3.4. Khối nguồn** 21](#_Toc169757125)

[**3.4. Thiết kế phần mềm** 22](#_Toc169757126)

[**CHƯƠNG 4. KẾT QUẢ** 28](#_Toc169757127)

[**4.1. KẾT QUẢ MÔ HÌNH THI CÔNG** 28](#_Toc169757128)

[**4.2. HOẠT ĐỘNG CỦA HỆ THỐNG** 29](#_Toc169757129)

[**CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN** 33](#_Toc169757130)

[**5.1. KẾT LUẬN** 33](#_Toc169757131)

[**5.2. HƯỚNG PHÁT TRIỂN** 33](#_Toc169757132)

[**TÀI LIỆU THAM KHẢO** 34](#_Toc169757133)

[**PHỤ LỤC** 35](#_Toc169757134)

# **CHƯƠNG 1. TỔNG QUAN**

## **1.1. GIỚI THIỆU**

Trong những năm gần đây, xu hướng công nghệ, đặc biệt là trong lĩnh vực điện tử và tự động hóa, đã phát triển mạnh mẽ. Cách chúng ta đối mặt và giải quyết các vấn đề hàng ngày đang thay đổi do áp dụng những tiến bộ công nghệ mới như IoT, trí tuệ nhân tạo, tự động hóa vào đời sống hàng ngày. Công nghệ hiện đại không chỉ cải thiện hiệu suất và sự hiệu quả mà còn đóng một vai trò quan trọng trong việc nâng cao chất lượng cuộc sống và đảm bảo an toàn cho con người. Hệ thống điều khiển giao thông là một ứng dụng quan trọng của công nghệ điện tử, nơi các giải pháp tự động hóa đang trở thành tiêu chuẩn để quản lý luồng giao thông trong các thành phố lớn.

Để quản lý tín hiệu đèn giao thông một cách hiệu quả và đáng tin cậy, nghiên cứu này tập trung vào thiết kế mạch đèn giao thông ngã tư, có thể điều khiển bằng tay. Sự phát triển của vi điều khiển và các linh kiện điện tử tiên tiến đã khiến việc xây dựng một hệ thống đèn giao thông hoạt động hiệu quả, tiết kiệm năng lượng và dễ dàng triển khai trở nên đơn giản hơn bao giờ hết. Đề tài này không chỉ nhằm mục đích cải thiện quản lý giao thông mà còn nhằm mục đích giảm thiểu tai nạn và tối ưu hóa thời gian di chuyển của các phương tiện.

Mục tiêu tạo ra một hệ thống đèn giao thông linh hoạt và hiệu quả hơn, tôi sử dụng vi điều khiển Arduino UNO R3 cùng với các linh kiện điện tử khác. Hệ thống sẽ được tối ưu hóa đặc biệt để hoạt động hiệu quả và dễ dàng bảo trì. Kết quả dự kiến là một hệ thống đèn giao thông hiệu quả có khả năng hiển thị đếm ngược, cho phép rẽ trái, cho phép lực lượng chức năng có thể điều khiển bằng tay khi lưu lượng người tham gia giao thông ở mật độ cao vào giờ cao điểm. Điều này sẽ góp phần giảm thiểu tai nạn và tối ưu hóa thời gian di chuyển cho các phương tiện tham gia giao thông.

## **1.2. MỤC TIÊU**

Mục tiêu của đề tài là tạo ra một hệ thống đèn giao thông hoạt động hiệu quả và linh hoạt, có khả năng cải thiện luồng giao thông và giảm thiểu tai nạn giao thông tại các giao lộ. Hệ thống đèn giao thông sẽ giảm tắc nghẽn, nâng cao hiệu quả vận hành giao thông và tối ưu hóa thời gian chờ cho các phương tiện với các LED 7 đoạn đếm ngược thời gian chờ, tín hiệu cho phép rẽ trái, nút nhấn cho phép điều khiển bằng tay khi lưu lượng người tham gia giao thông trên đường đông vào giờ cao điểm. Hệ thống cũng sẽ được thiết kế với tính năng dễ dàng bảo trì, đảm bảo hoạt động bền bỉ và liên tục trong điều kiện môi trường khắc nghiệt.

Một mục tiêu quan trọng khác là phát triển một hệ thống có thể mở rộng và tích hợp với các hệ thống giao thông thông minh khác để hỗ trợ xây dựng các thành phố thông minh và bền vững bằng cách thêm các cảm biến. Hệ thống này không chỉ cải thiện hiệu suất giao thông hiện tại mà còn được sử dụng rộng rãi trong các đô thị hiện đại, nâng cao chất lượng cuộc sống và an toàn cho người tham gia giao thông.

## **1.3. GIỚI HẠN**

Đề tài tập trung vào việc xây dựng một hệ thống đèn giao thông hoạt động tại ngã tư, sử dụng vi điều khiển Arduino UNO R3 để điều khiển các tín hiệu đèn giao thông theo một chu kỳ định sẵn. Mục tiêu chính là thiết kế và triển khai một mạch điện tử có khả năng thay đổi trạng thái của các đèn LED (đỏ, vàng, xanh) theo một trình tự logic và thời gian cố định, giúp điều tiết giao thông tại các giao lộ một cách hiệu quả. Đề tài sẽ tập trung vào các khía cạnh sau:

- Xây dựng mạch điện tử với vi điều khiển Arduino UNO R3 làm trung tâm, điều khiển các đèn LED giao thông.

- Phát triển chương trình điều khiển cho vi điều khiển để quản lý trình tự và thời gian của các tín hiệu đèn giao thông, đảm bảo thời gian chờ đợi được tối ưu hóa, giảm thiểu tắc nghẽn.

- Có tín hiệu cho phép rẽ trái, có nút nhấn cho phép điều khiển bằng tay khi lưu lượng giao thông trên đường đông, tránh ùn tắc giao thông.

Tóm lại, đề tài sẽ đặc biệt chú trọng vào việc tạo ra mạch điện tử hoàn chỉnh với Arduino UNO R3 làm trung tâm để điều khiển các đèn LED giao thông theo chu kỳ đã lập trình. Hệ thống có tín hiệu cho phép rẽ trái và các nút nhấn cho phép lực lượng chức năng điều khiển bằng tay khi ở giờ cao điểm.

## **1.4. BỐ CỤC ĐỒ ÁN**

Chương 1. ***Tổng quan:*** Giới thiệu về tình hình nghiên cứu, tên đề tài, mục tiêu, giới hạn đề tài và bố cục quyển báo cáo.

Chương 2. ***Cơ sở lý thuyết:*** Trình bày các lý thuyết liên quan đến mạch đèn giao thông, chuẩn giao tiếp UART, Timer 1.

Chương 3. ***Thiết kế:*** Chương này tập trung vào việc xây dựng sơ đồ khối của hệ thống, cũng như thiết kế chi tiết khối và thiết kế phần mềm của hệ thống.

Chương 4. ***Kết quả:*** Chương này chủ yếu đưa ra các kết quả mạch in, mạch điện sau khi thi công, kết quả đạt được sau khi thiết kế mạch.

Chương 5. ***Kết luận và hướng phát triển:*** Tổng kết đề tài, đưa ra nhận định và hướng phát triển của đề tài trong tương lai.

# **CHƯƠNG 2. CƠ SỞ LÝ THUYẾT**

## **2.1. LÝ THUYẾT CƠ BẢN VỀ MẠCH ĐÈN GIAO THÔNG**

Đèn giao thông là phần quan trọng của hệ thống điện tử tự động điều khiển các tín hiệu giao thông để điều chỉnh luồng xe và người đi bộ tại các giao lộ và các địa điểm khác trên đường. Đèn giao thông rất quan trọng để duy trì trật tự an toàn giao thông, giảm thiểu tai nạn và đảm bảo hiệu quả di chuyển khi mật độ người tham gia giao thông ngày càng tăng, đặc biệt là các giờ cao điểm gây nên ùn tắc giao thông. Mạch đèn giao thông là một phần quan trọng của hệ thống này, sử dụng các thành phần điện tử và vi điều khiển để tự động hóa quá trình điều khiển các tín hiệu đèn giao thông. Nó được lắp đặt ở trung tâm giao lộ hoặc ngay vỉa hè, có thể hoạt động tự động hoặc được cảnh sát giao thông điều khiển bằng tay.

Mạch đèn giao thông hoạt động theo một chu kỳ định trước, trong đó mỗi màu của đèn được bật trong một khoảng thời gian xác định. Đèn xanh cho phép các phương tiện di chuyển. Đèn vàng cảnh báo các phương tiện chuẩn bị dừng lại. Đèn đỏ yêu cầu các phương tiện dừng lại. Chu kỳ được lặp đi lặp lại liên tục để đảm bảo các luồng giao thông được điều tiết một các hợp lý. Trong các hệ thống phức tạp hơn, chu kỳ này có thể được điều chỉnh dựa trên các tín hiệu từ các cảm biến giao thông để tối ưu hóa luồng xe và giảm thời gian chờ đèn chuyển tín hiệu. Tại các giao lộ phức tạp với nhiều hướng xe cộ và phân làn cho người đi bộ, mạch đèn giao thông có thể được thiết kế để điều khiển riêng biệt từng luồng giao thông. Điều này đòi hỏi mạch phải có khả năng quản lý nhiều tín hiệu đầu ra và các chu kỳ đèn phức tạp.

Mạch đèn giao thông có cấu trúc không quá phức tạp, bao gồm nhiều thành phần điện tử và cơ học được sử dụng cùng nhau để tạo ra các tín hiệu giao thông rõ ràng và chính xác. Trong mạch đèn giao thông, vi điều khiển đóng vai trò là bộ phận trung tâm cũng như là bộ não của mạch điện tử. Vi điều khiển được sử dụng ở đây có nhiều loại, như Arduino, PIC hoặc AVR, hay bất kỳ vi điều khiển nào có đủ khả năng và có cổng I/O. Nó có thể quản lý các tín hiệu đầu vào và điều khiển các đèn LED bật/tắt theo một chu kỳ và theo một thứ tự cụ thể hay điều khiển các LED 7 đoạn đếm ngược được lập trình để kiểm soát thời gian chờ của người tham gia giao thông. Vi điều khiển có thể nhận tín hiệu từ các cảm biến hoặc sử dụng các chu kỳ đã lập trình sẵn để điều chỉnh đèn. Nó cũng có khả năng thực hiện các lệnh được lập trình, sử dụng các chân I/O để điều khiển các đèn.

Các đèn LED cũng được sử dụng trong mạch đèn giao thông, bao gồm cả LED đơn và LED 7 đoạn, giúp mô phỏng tín hiệu của các đèn giao thông và đếm ngược thời gian chờ. Ba đèn LED đỏ, xanh, vàng và LED 7 đoạn được sử dụng trong mỗi cột đèn giao thông và được điều khiển bởi các chân I/O của vi điều khiển. Để bảo vệ đèn LED khỏi quá tải điện, mạch điều khiển sẽ cần có các điện trở hạn dòng, có thể dùng điện trở thông thường hoặc điện trở thanh để làm giảm diện tích và tăng tính hiệu quả, đảm bảo các đèn LED hoạt động tốt với độ sáng ổn định và lâu dài. Bên cạnh đó, cũng có thể sử dụng thêm IC 74HC595 để giảm số lượng chân I/O cần thiết trên vi điều khiển, cho phép dịch dữ liệu số từ vi điều khiển đến các LED 7 đoạn một cách hiệu quả và chính xác, dễ dàng quản lý việc hiển thị số trên LED 7 đoạn.

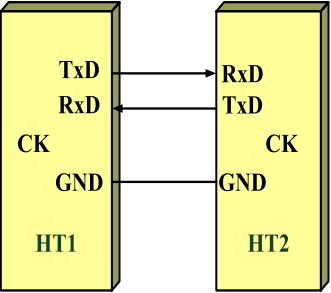
Việc thiết kế mạch in và lập trình điều khiển mạch cũng rất quan trọng. Các công cụ thiết kế mạch in phổ biến bao gồm Proteus, Multisim, Eagle, Altium Designer và Orcad. Nhìn chung, chúng đều hỗ trợ thiết kế mạch in dễ dàng và chính xác, cho phép người dùng tạo ra các thiết bị và hệ thống điện tử hoạt động hiệu quả và có chất lượng cao. Lập trình mạch điện tử cũng là quá trình viết và phát triển các mã lệnh để điều khiển các thành phần điện tử được sử dụng trong mạch. Mục đích của việc này là điều khiển các thiết bị điện tử đáp ứng các yêu cầu của người dùng hoặc thực hiện các chức năng cụ thể. Có khả năng lập trình mạch điện tử với các ngôn ngữ lập trình như C/C++ hoặc Python. Các môi trường lập trình phù hợp phải được cài đặt trước khi có thể sử dụng chúng. Tùy thuộc vào loại mạch điện hoặc vi điều khiển, phần mềm Arduino IDE, PlatformIO hoặc MATLAB X IDE phải được cài đặt. Ngoài ra, cần cài đặt trình biên dịch và công cụ lập trình, chẳng hạn như GCC Compiler cho C/C++ hoặc Python interpreter cho Python. Sau đó, cần phải viết mã cho mạch điện tử bằng ngôn ngữ lập trình đã chọn. Người dùng sẽ có thể tương tác với các linh kiện và chức năng của mạch bằng cách sử dụng các hàm và thư viện có sẵn. Sau đó, người dùng phải biên dịch mã lệnh bằng cách sử dụng trình biên dịch phù hợp. Sau đó, họ phải kiểm tra xem chương trình đã được tải vào mạch điện tử có hoạt động hay không. Có thể sử dụng các công cụ debug và các phương pháp để tìm và sửa lỗi trong chương trình nếu gặp lỗi trong quá trình chạy chương trình.

Bằng cách kết hợp các hệ thống và giải pháp kỹ thuật này, có thể hướng đến thiết kế và triển khai một mạch đèn giao thông hoạt động ổn định, hiệu quả và dễ dàng bảo trì, đáp ứng tốt các yêu cầu cơ bản của hệ thống giao thông.

## **2.2. CHUẨN GIAO TIẾP UART**

### **2.2.1. Giới thiệu**

Chuẩn giao tiếp UART, còn được gọi là Serial, là một chuẩn giao tiếp đơn giản và phổ biến trong các ứng dụng của hệ thống nhúng, bao gồm hai đường truyền dữ liệu độc lập là TX (truyền) và RX (nhận). Đây là một giao thức truyền thông phần cứng dùng giao tiếp nối tiếp bất đồng bộ và có thể cấu hình được tốc độ, là một chuẩn giao tiếp phần cứng được thiết kế để truyền dữ liệu giữa các thiết bị điện tử mà không cần sự đồng bộ hóa bằng xung CK (clock), không còn phân biệt chủ và tớ - các hệ thống là ngang cấp [1], được minh họa như hình 2.1. Thay vào đó nó sử dụng các bit đặc biệt để xác định điểm bắt đầu và kết thúc của một chuỗi dữ liệu, đồng thời yêu cầu cả hai thiết bị truyền và nhận phải hoạt động ở cùng một tốc độ baud để đảm bảo thông tin được truyền đạt một cách chính xác.



**Hình 2.1.** Hệ thống truyền bất đồng bộ

Chuẩn giao tiếp UART cho phép truyền và nhận dữ liệu nối tiếp một cách không đồng bộ. Ở kiểu truyền này, có một đường phát dữ liệu và một đường nhận dữ liệu, không còn tín hiệu xung nhịp nên gọi là bất đồng bộ. Để truyền được dữ liệu thì cả bên phát và bên nhận phải tự tạo xung nhịp có cùng tần số và thường gọi là tốc độ truyền dữ liệu (baud), ví dụ 2400 baud, 4800 baud,…, 2400 baud có nghĩa là truyền 2400 bit trên 1 giây [1]. Khi sử dụng bo Arduino UNO để giao tiếp với máy tính, chỉ cần cắm cáp USB vào bo Arduino để sử dụng của sổ Serial Monitor trên phần mềm Arduino IDE, không cần phải nối thêm gì cả vì trên bo Arduino đã thực hiện sẵn việc đó. Để khai báo sử dụng UART, ta sử dụng lệnh “Serial.begin(x);” với x là tốc độ baud, có các giá trị là 4800, 9600, 57600, 115200 [2].

### **2.2.2. Nguyên lý hoạt động**

Các thành phần chính của UART bao gồm:

- Bộ truyền (Transmitter): thu thập dữ liệu từ nguồn rồi định dạng dữ liệu thành các bit nối tiếp và gửi qua chân TX.

- Bộ nhận (Receiver): nhận dữ liệu qua chân RX, xử lý dữ liệu sau đó chuyển đổi thành dữ liệu song song cho máy chủ.

UART hoạt động bằng cách truyền dữ liệu dưới dạng các bit nối tiếp. Dữ liệu được truyền và nhận qua các đường truyền TX và RX dưới dạng các khung dữ liệu (data frame) có cấu trúc chuẩn, với một bit bắt đầu (start bit), các bit dữ liệu (data bits), một bit kiểm tra chẵn lẻ (parity bit) và một hoặc nhiều bit dừng (stop bit). Không giống như truyền song song, nơi nhiều bit được truyền cùng một lúc, UART truyền từng bit một. Chính điều này làm giảm số lượng dây dẫn cần thiết nhưng cũng yêu cầu cơ chế đồng bộ hóa dữ liệu phức tạp hơn.

Hình 2.2 ví dụ về quá trình truyền – nhận dữ liệu của UART, bus dữ liệu trước khi truyền là “0111\_0010”. Khi UART truyền dữ liệu, nó bắt đầu nhận dữ liệu dưới dạng song song, do đó bộ truyền nhận được “0111\_0010”. Sau khi nhận dữ liệu từ bus dữ liệu, UART thêm vào các bit bắt đầu (0), bit chẵn lẻ (chọn là 1) và bit dừng (0). Toàn bộ gói dữ liệu (bao gồm dữ liệu từ bộ truyền và ba bit được thêm vào được truyền từ bộ truyền đến bộ nhận theo tuần tự. Quá trình này diễn ra theo một tốc độ baud định sẵn, đảm bảo rằng dữ liệu đưuọc truyền tại một tốc độ thống nhất giữa các bộ. Bộ nhận sau đó nhận dữ liệu tại tốc độ baud đã định sẵn để đồng bộ hóa với bộ truyền. Sau khi nhận đủ các bit trong khung dữ liệu, bộ nhận sẽ loại bỏ các bit điều khiển để lấy ra dữ liệu gốc là “0111\_0010”. Sau đó, bộ nhận chuyển đổi dữ liệu từ dạng tuần tự về dạng song song và gửi dữ liệu này lên bus dữ liệu tại nơi nhận.



**Hình 2.2.** Ví dụ về quá trình truyền – nhận dữ liệu của UART

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

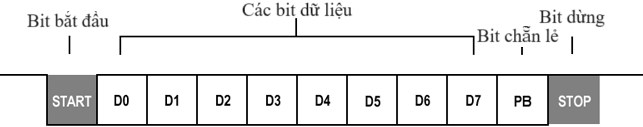
- Đơn công: chỉ tiếp hành giao tiếp một chiều.

- Bán song công: dữ liệu sẽ đi theo một hướng tại một thời điểm.

- Song công: thực hiện giao tiếp đồng thời đến và đi từ mỗi chủ và tớ.

### **2.2.3. Khung dữ liệu**

Khung dữ liệu UART là một thành phần quan trọng trong giao tiếp nối tiếp, bao gồm các thành phần sau và được minh họa như hình 2.3.



**Hình 2.3.** Khung dữ liệu UART

- Bit bắt đầu (start bit): Bit bắt đầu luôn ở mức logic thấp (0). Khi bộ nhận phát hiện một bit bắt đầu, nó biết rằng một khung dữ liệu mới đang bắt đầu và chuẩn bị để nhận các bit sắp tới [2].

- Các bit dữ liệu (data bits): Là thành phần chính trong UART, mang thông tin cần truyền. Số lượng bit dữ liệu có thể thay đổi tùy theo cấu hình, thường là 8 bit, cho phép truyền một byte dữ liệu, có thể đại diện cho nhiều giá trị khác nhau, bao gồm các ký tự ASCII, giá trị số, … Tuy nhiên, UART cũng hỗ trợ các số lượng bit trong bit dữ liệu khác nhau, bao gồm 7 bit và 6 bit, tùy thuộc vào yêu cầu ứng dụng cụ thể. Các bit dữ liệu đại diện cho các ký tự hoặc dữ liệu sử dụng mã nhị phân, mỗi bit tương ứng với một lũy thừa của 2. Mỗi bit trong byte dữ liệu có vị trí và trọng số cụ thể trong biểu diễn nhị phân. Việc mã hóa này cho phép truyền tải một phạm vi thông tin rộng, từ các ký tự văn bản đơn giản đến dữ liệu nhị phân phức tạp [2].

- Bit chẵn lẻ (parity bit): Là cơ chế kiểm tra lỗi, giúp phát hiện các lỗi truyền dữ liệu. Bit chẵn lẻ có thể được đặt là chẵn hoặc lẻ, đảm bảo cho tổng số bit 1 trong một ký là chẵn hoặc lẻ, tùy thuộc vào loại chẵn lẻ được chọn. Nếu số lượng bit 1 không khớp với loại chẵn lẽ dự kiến, một lỗi sẽ được phát hiện [2].

- Bit dừng (stop bit): Một hoặc nhiều bit dừng ở mức logic cao (1) được gửi báo hiệu kết thúc của khung dữ liệu. Cấu hình phổ biến nhất là sử dụng một bit dừng, nhưng trong những trường hợp cần yêu cầu độ tin cậy cao thì hai bit dừng được sử dụng [2].

Độ chính xác của việc truyền dữ liệu trong giao tiếp UART phụ thuộc vào việc cấu hình đúng của các bit dữ liệu. Nếu cấu hình sai lệch, dữ liệu có thể bị lỗi. Ví dụ, nếu bộ truyền gửi dữ liệu 8 bit nhưng bộ nhận lại được cấu hình để nhận dữ liệu 7 bit thì dữ liệu có thể bị giải mã sai.

## **2.3. TIMER 1**

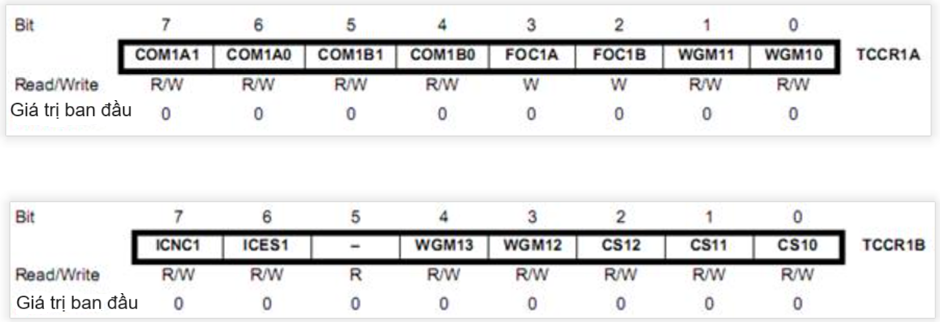
### **2.3.1. Giới thiệu**

Arduino UNO R3 được trang bị một chip Atmega328P với 3 bộ Timer/Counter là Timer/Counter 0, Timer/Counter 1 và Timer/Counter 2. Trong đó, bộ Timer/Counter 1 là một bộ Timer/Counter 16 bit, có khả năng định thời chính xác hơn so với các Timer/Counter 8 bit khác (Timer/Counter 0 và 2). Là một thanh ghi 16 bit, có độ phân giải cao, Timer/Counter 1 cho phép đếm từ 0 đến 65535. Nó hỗ trợ các chức năng như tạo xung PWM, đo thời gian chính xác, tạo ngắt dựa trên thời gian. Với các tính năng như thanh ghi so sánh, thanh ghi dữ liệu và các chế độ đếm phong phú, Timer/Counter 1 rất hữu tích trong các ứng dụng điều khiển và đo đạc thời gian chính xác.

### **2.3.2. Khảo sát các thanh ghi của Timer 1**

*2.3.2.1. Thanh ghi điều khiển*

Hình 2.4 dưới đây minh họa cấu trúc hai thanh ghi điều khiển của Timer 1.



**Hình 2.4.** 2 thanh ghi điều khiển TCCR1 của Timer 1

Hai thanh ghi TCCR1A và TCCR1B được dùng để điều khiển hoạt động của Timer 1, các bit trong hai thanh ghi này cho phép xác định tất cả các trạng thái hoạt động của Timer 1. Trong đó, thanh ghi TCCR1A chịu trách nhiệm cho PWM và TCCR1B được sử dụng để thiết lập giá trị chia trước. Các bit trong hai thanh ghi này bao gồm các bit chọn dạng sóng WGM (Waveform Generating Mode), các bit quy định ngõ ra COM (Compare Output Match), các bit chọn giá trị chia trước cho xung nhịp CS (Chip Select). Ta đặt tất cả các bit trong thanh ghi TCCR1A về 0 vì không sử dụng nó. Tuy nhiên, đối với thanh ghi TCCR1B, ba bit đầu tiên được sử dụng để thiết lập các giá trị chia trước. Các bit này được gọi là CS10, CS11, CS12 [3]. Các giá trị chia trước cho các bit này được thiết lập như bảng 2.1 dưới đây.

**Bảng 2.1.** Bảng mô tả các bit lựa chọn xung nhịp

|  |  |  |  |
| --- | --- | --- | --- |
| CS12 | CS11 | CS10 | Mô tả |
| 0 | 0 | 0 | Dừng Timer |
| 0 | 0 | 1 | clkI/O/1 |
| 0 | 1 | 0 | clkI/O/8 |
| 0 | 1 | 1 | clkI/O/64 |
| 1 | 0 | 0 | clkI/O/256 |
| 1 | 0 | 1 | clkI/O/1024 |
| 1 | 1 | 0 | Timer 1 đếm xung trên chân T1 với xung cạnh xuống |
| 1 | 1 | 1 | Timer 1 đếm xung trên chân T1 với xung cạnh lên |

*2.3.2.2. Thanh ghi so sánh*

Hai thanh ghi OCR1A và OCR1B là hai thanh ghi so sánh, lưu trữ giá trị so sánh được sử dụng khi Timer 1 hoạt động ở chế độ CTC (Clear Timer on Compare Match) hoặc chế độ PWM. Khi giá trị của TCNT1 bằng với giá trị của một trong hai thanh ghi này thì một hành động (như ngắt hoặc thay đổi trạng thái chân) sẽ xảy ra. Hình 2.5 dưới đây minh họa cấu trúc hai thanh ghi so sánh của Timer 1.

A black and white text on a white background

Description automatically generated

**Hình 2.5.** 2 thanh ghi so sánh OCR1 của Timer 1

*2.3.2.3. Thanh ghi mặt nạ ngắt*

TIMSK1 là thanh ghi mặt nạ ngắt của Timer 1, cho phép hoặc vô hiệu hóa các ngắt của Timer 1. Hình 2.6 dưới đây minh họa cấu trúc của thanh ghi mặt nạ ngắt của Timer 1.

A black and white text

Description automatically generated

**Hình 2.6.** Thanh ghi mặt nạ ngắt TIMSK của Timer 1

Bit 0 TOIE0: cho phép ngắt khi xảy ra tràn trên Timer 1.

Bit 2 TOIE1: bit quy định ngắt tràn cho Timer 1.

Bit 3 OCIE1B: bit cho phép ngắt khi có 1 “Match” xảy ra trong việc so sánh TCNT1 với OCR1B.

Bit 4 OCIE1A: bit cho phép ngắt khi có 1 “Match” xảy ra trong việc so sánh TCNT1 và OCR1A.

Bit 5 TICIE1: bit cho phép ngắt trong trường hợp Input Capture được dùng [4].

*2.3.2.4. Thanh ghi cờ nhớ*

TIFR (Timer/Counter Interrupt Flag Register) là thanh ghi cờ nhớ cho tất cả các bộ Timer/Counter, được minh họa như hình 2.7. Các bit từ 2 đến 5 trong thanh ghi này là các cờ trạng thái của Timer 1 [3].

A black and white text

Description automatically generated

**Hình 2.7.** Thanh ghi cờ nhớ TIFR của Timer 1

### **2.3.3. Các chế độ hoạt động của Timer 1**

- Chế độ thường: Đây là chế độ hoạt động đơn giản nhất của Timer 1. Trong chế độ này, giá trị của thanh ghi đếm TCNT1 tăng từ 0 lên 65535 hoặc từ 0xffff về 0. Để thiết lập Timer 1 hoạt động ở chế độ thường, ta cần đặt 4 bit WGM (Waveform Generation Mode) về 0 vì 0 là giá trị mặc định của các thanh ghi nên trên thực tế không cần tác động đến các bit này. Việc quan trọng cần làm là đặt các bit Clock Select (CS12, CS11, CS10) trong thanh ghi TCCR1B [3].

- Chế độ CTC (Clear Timer on Compare Match): Đây là chế độ xóa timer nếu xảy ra bằng trong so sánh. Có 2 mode CTC trên Timer 1 là mode 4 và mode 12. Ví dụ, khi đặt các bit Waveform Generation Mode WGM13 = 0, WGM12 = 1, WGM11 = 0, WGM10 = 0 thì mode 4 được chọn. Trong mode này, thanh ghi OCR1A chứa giá trị so sánh do người dùng đặt, thanh ghi đếm TCNT1 tằng từ 0, khi TCNT1 bằng với giá trị chứa trong OCR1A thì một “Compare Match” xảy ra. Khi đó một ngắt có thể xảy ra nếu ta cho phép ngắt [3].

# **CHƯƠNG 3. THIẾT KẾ**

Trong đề tài này, để thuận tiện cho việc quan sát, tác giả chọn thời gian đèn xanh sáng là 12 giây, thời gian đèn vàng sáng là 3 giây, thời gian đèn đỏ sáng là 15 giây. Ban đầu, mạch hoạt động ở chế độ làm việc bình thường.

## **3.1. Yêu cầu của hệ thống**

Mạch có các chức năng điều khiển các phương tiện lưu thông trên ngã tư, hoạt động ở 3 chế độ:

- Chế độ làm việc bình thường: Được sử dụng khi lưu lượng người tham gia giao thông trên đường với mật độ bình thường. Khi nhấn nút nhấn xanh thì đèn ở 2 cột đối diện sáng cùng màu với nhau.

- Chế độ làm việc vào giờ cao điểm: Hoạt động như một chế độ điều khiển bằng tay. Chế độ này được sử dụng vào giờ cao điểm khi lưu lượng người tham gia giao thông trên đường lớn, nếu để làm việc tự động thì có thể gây ùn tắc giao thông. Khi nhấn nút nhấn đỏ thì đèn vàng ở 4 cột sáng 3 giây để người tham gia giao thông kịp thời phản ứng với sự thay đổi đột ngột này. Sau đó, đèn ở 2 hàng ngang chuyển sang màu xanh, cho phép các phương tiện theo tuyến đường ngang lưu thông, đồng thời đèn cho phép rẽ trái ở 2 cột này bật, cho phép các phương tiện trên tuyến đường ngang được rẽ trái, đèn ở 2 cột dọc chuyển sang màu đỏ. Nhấn nút nhấn đen thì đảo ngược lại, cho phép chuyển đổi thủ công giữa các trạng thái đèn giao thông.

- Chế độ làm việc vào ban đêm: Được sử dụng khi lưu lượng người tham gia giao thông trên đường ít, vào ban đêm. Nhấn nút nhấn vàng thì các đèn vàng ở 4 cột sáng nhấp nháy.

## **3.2. Sơ đồ khối**

Hệ thống gồm 4 khối kết nối với nhau, được trình bày như hình 3.1 bên dưới.

**A diagram of a diagram

Description automatically generated with medium confidence**

**Hình 3.1.** Sơ đồ khối tổng quát của hệ thống

Chức năng của từng khối trong sơ đồ khối được trình bày như sau:

- Khối nguồn: Tạo ra điện áp ổn định 5V từ nguồn xoay chiều 220V để cung cấp điện năng cho toàn bộ hệ thống hoạt động, đảm bảo các thành phần của hệ thống hoạt động ổn định.

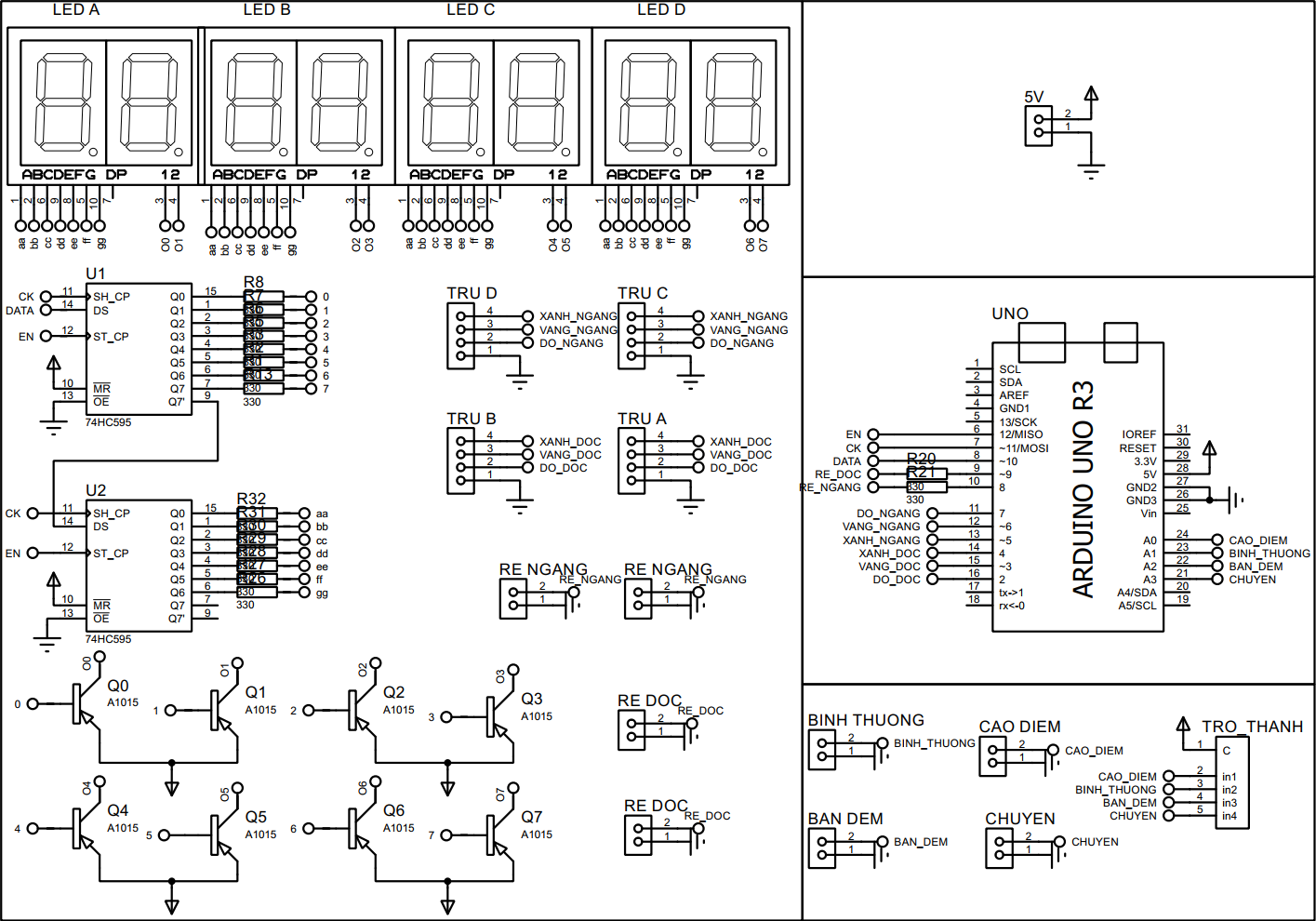
- Khối xử lý trung tâm: Điều khiển toàn bộ hệ thống dựa trên các tín hiệu đầu vào từ các nút nhấn và điều khiển các đèn hiển thị. Arduino UNO R3 nhận tín hiệu từ các nút nhấn, thực hiện xử lý logic để chuyển đổi giữa các chế độ hoạt động và điều khiển các đèn giao thông hiển thị theo trạng thái được chỉ định.

- Khối nút nhấn: Cho phép người dùng chuyển đổi giữa các chế độ hoạt động của hệ thống, gồm 4 nút nhấn, mỗi nút nhấn tương ứng với một chế độ hoạt động. Khi một nút nhấn được nhấn, tín hiệu điện từ nút nhấn đó được gửi tới Arduino để thay đổi trạng thái hoạt động của hệ thống.

- Khối hiển thị: Hiển thị tín hiệu giao thông cho các phương tiện theo hướng dẫn của hệ thống, gồm các đèn LED giao thông (xanh, vàng, đỏ) và các LED 7 đoạn được điều khiển bởi Arduino. Tùy theo chế độ hoạt động được chọn, Arduino sẽ điều khiển các đèn này bật hoặc tắt theo trình tự phù hợp.

## **3.3. Thiết kế phần cứng**

Theo sơ đồ khối được thiết kế như hình 3.1, tác giả sẽ thiết kế sơ đồ nguyên lý gồm gồm 4 khối như trong sơ đồ khối. Hình 3.2 thể hiện sơ đồ nguyên lý toàn hệ thống với các khối được mô tả chi tiết như các phần dưới đây.



**Hình 3.2.** Sơ đồ nguyên lý toàn hệ thống

### **3.3.1. Khối nút nhấn**

Hình 3.3 mô phỏng khối nút nhấn sử dụng phần mềm Proteus. Khối này dùng để điều chỉnh các chế độ hoạt động của hệ thống đèn giao thông trong ngày (chế độ bình thường, chế độ giờ cao điểm, chế độ ban đêm).

**A diagram of a machine

Description automatically generated with medium confidence**

**Hình 3.3.**Khối nút nhấn

Để phù hợp với kích thước cũng như yêu cầu đơn giản của mạch, ở đây tôi chọn sử dụng nút nhấn nhả DS-318 12mm. Nút nhấn này có 2 chân, trong đó 1 chân của mỗi nút nhấn được nối với các chân A0, A1, A2, A3 của Arduino UNO R3 để nhận tín hiệu từ nút nhấn rồi điều khiển các mạch hoạt động theo các chế độ mong muốn. Trong khi đó, chân còn lại được nối đất.

### **3.3.2. Khối hiển thị**

Khối hiển thị của mạch được mô phỏng như hình 3.4, cho phép hiển thị tín hiệu giao thông cho các phương tiện theo các chế độ được điều khiển bằng nút nhấn. Ở đây tôi sử dụng 4 LED 7 đoạn 2 số loại anode chung để hiển thị thời gian đếm ngược, 4 mạch LED giao thông 5V để hiển thị đèn giao thông và 4 LED đơn xanh cho phép rẽ trái.

Có nhiều phương pháp điều khiển LED 7 đoạn như kết nối trực tiếp các chân của LED 7 đoạn với các PORT của vi điều khiển, quét LED dùng IC chốt. Tuy nhiên, trong đồ án này, tác giả điều khiển 4 LED 7 đoạn 2 số bằng phương pháp quét dùng transistor. Ở phương pháp quét thì transistor đóng vai trò như một công tắc khi tắt thì không cho dòng qua LED nên LED tắt, khi mở thì cho phép dòng qua LED để LED sáng đúng con số mong muốn. Để lựa chọn transistor thì ta giả sử cho tất cả 7 đoạn và dấu chấm thập phân sáng và dòng qua mỗi đoạn là 15mA, tổng dòng là 15mA × 8 bằng 90mA, nên chọn transistor có dòng làm việc IC lớn hơn hay bằng 90mA là được. Các transistor có dòng từ 100mA đến 150mA khá phổ biến là A564, A1015 [1]. Ở phạm vi đồ án này, tôi chọn sử dụng transistor A1015. Đây là trasistor loại pnp, phân cực ngược vì có cấp tín hiệu vào cực Base. Các chân dương chung của các LED 7 đoạn 2 số được kết nối với các chân Collector của transistor để nhận dòng hoạt động.

**A diagram of a circuit board

Description automatically generated**

**Hình 3.4.**Khối hiển thị

Bên cạnh đó, khối hiển thị có sử dụng 2 IC 74HC595 để mở rộng số lượng chân đầu ra từ Arduino UNO R3 để điều khiển nhiều LED 7 đoạn. IC 74HC59 là một thanh ghi dịch với đầu ra song song, cho phép chuyển đổi dữ liệu nối tiếp từ vi điều khiển gửi tới thành tín hiệu song song để điều khiển các LED 7 đoạn. Sử dụng 2 IC 74HC595 trong chuỗi nối tiếp giúp tăng số lượng chân điều khiển từ Arduino mà không cần phải tăng số lượng chân I/O sử dụng. Việc này cho phép điều khiển nhiều đèn LED hơn mà không cần sử dụng nhiều chân I/O của Arduino, giúp giảm thiểu sự phức tạp của hệ thống dây làm tối ưu hóa việc thiết kế và đảm bảo hệ thống hoạt động hiệu quả, ổn định. Trong mạch, bằng cách kết nối chân Q7’ của IC 74HC595 thứ nhất với chân DS của IC 74HC595 thứ hai, ta được thanh ghi dịch 16 bit. Các chân a, b, c, d, e, f, g của các LED 7 đoạn 2 số được nối với các chân Q0 đến Q7 của IC 74HC595 thứ hai để nhận tín hiệu đầu ra song song. Chân DS (chân đầu vào nối tiếp) của IC 74HC595 thứ nhất được nối với chân 10 của Arduino, các chân SH\_CP (chân đầu vào xung nhịp) và ST\_CP (chân chốt) được kết nối với lần lượt với chân 11 và chân 12.

### **3.3.3. Khối xử lý trung tâm**

Khối xử lý trung tâm trong hệ thống điều khiển đèn giao thông được thiết kế nhằm đảm bảo sự hoạt động ổn định, hiệu quả của các tín hiệu đèn giao thông, đảm bảo an toàn giao thông và tối ưu hóa luồng xe qua các ngã tư. Hình 3.5 biểu diễn khối xử lý trung tâm trong mạch.

**A circuit board with many wires

Description automatically generated**

**Hình 3.5.**Khối xử lý trung tâm

Khối này sử dụng vi điều khiển Arduino UNO R3 như bộ não của hệ thống để làm trung tâm điều khiển, kết hợp với 2 IC 74HC595 để mở rộng số lượng chân điều khiển, từ đó điều khiển các đèn LED giao thông và hiển thị đếm ngược thời gian trên các LED 7 đoạn.

Arduino UNO R3 là trung tâm xử lý của mạch, thực hiện các chức năng điều khiển chính của hệ thống. Nó là một bo mạch vi điều khiển nguồn mở dựa trên vi điều khiển Microchip Atmega328P. Được phát triển bởi Arduino.cc, Arduino UNO đã trở thành một công cụ quan trọng trong việc giáo dục, nghiên cứu và phát triển các dự án điện tử. Bo mạch này được trang bị các chân I/O digital và analog có thể giao tiếp với các bo mạch mở rộng khác nhau. Nó có 14 chân digital I/O (trong đó có 6 chân có thể sử dụng làm đầu ra PWM), 6 chân analog. Arduino UNO có thể được cấp nguồn 5V thông qua cổng USB hoặc cấp nguồn ngoài thông qua jack cắm với điện áp khuyên dùng là 7-12V DC và giới hạn là 6-20V, nếu cấp nguồn vượt quá ngưỡng giới hạn trên sẽ làm hỏng Arduino. Bảng 3.1 dưới đây trình bày các thông số kỹ thuật chính của nó [4].

**Bảng 3.1.** Bảng thông số kỹ thuật chính của Arduino UNO

|  |  |
| --- | --- |
| Vi điều khiển | ATmega328P (họ 8 bit) |
| Điện áp hoạt động | 5V DC (được cung cấp qua cổng USB) |
| Tần số hoạt động | 16MHz |
| Dòng tiêu thụ | 30mA |
| Điện áp vào khuyên dùng | 7 – 12V DC |
| Điện áp vào giới hạn | 6 – 20V DC |
| Số chân digital I/O | 14 (6 chân PWM) |
| Số chân analog | 6 (độ phân giải 10 bit) |
| Dòng tối đa trên mỗi chân I/O | 30mA |
| Dòng ra tối đa (5V) | 500mA |
| Dòng ra tối đa (3.3V) | 50mA |
| Bộ nhớ Flash | 32KB với 0.5KB được dùng cho bootloader |
| SRAM | 2KB |
| EEPROM | 1KB |

Các chân của Arduino được kết nối như sau:

- Chân D2 kết nối với tín hiệu đèn đỏ của mạch LED giao thông ở cột dọc DO\_DOC.

- Chân D3 kết nối với tín hiệu đèn vàng của mạch LED giao thông ở cột dọc VANG\_DOC.

- Chân D4 kết nối với tín hiệu đèn xanh của mạch LED giao thông ở cột dọc XANH\_DOC.

- Chân D5 kết nối với tín hiệu đèn xanh của mạch LED giao thông ở hàng ngang XANH\_NGANG.

- Chân D6 kết nối với tín hiệu đèn vàng của mạch LED giao thông ở hàng ngang VANG\_NGANG.

- Chân D7 kết nối với tín hiệu đèn đỏ của mạch LED giao thông ở hàng ngang DO\_NGANG.

- Chân D8 kết nối với tín hiệu cho phép rẽ trái ở hàng ngang RE\_NGANG thông qua 1 điện trở hạn dòng 330Ω.

- Chân D9 kết nối với tín hiệu cho phép rẽ trái ở cột dọc RE\_DOC thông qua 1 điện trở hạn dòng 330Ω.

- Chân D10, kỳ hiệu là DATA, kết nối với tín hiệu DS của IC 74HC595 thứ nhất, là chân đầu vào dữ liệu nối tiếp của IC. Chân này được sử dụng để gửi dữ liệu tuần tự vào IC 74HC595. Mỗi bit dữ liệu được gửi vào chân này theo trình tự thời gian, bit cao nhất được gửi trước, sau đó là các bit tiếp theo cho đến bit thấp nhất.

- Chân D11, ký hiệu là CK, kết nối với tín hiệu SH\_CP của cả 2 IC 74HC595, là chân clock của IC. Chân này được sử dụng để cung cấp tín hiệu xung clock cho cả 2 IC. Mỗi cạnh lên của xung clock sẽ dịch chuyển dữ liệu trong thanh ghi dịch của IC 74HC595. Mỗi lần có một xung clock, một bit dữ liệu từ chân DATA sẽ được dịch vào thanh ghi dịch.

- Chân D12, ký hiệu là EN, kết nối với tín hiệu ST\_CP của cả 2 IC 74HC595, là chân latch của IC. Chân này được sử dụng để cập nhật giá trị từ thanh ghi dịch vào các chân ngõ ra của cả 2 IC. Khi chân EN được đưa lên mức cao, dữ liệu từ thanh ghi dịch sẽ được ghi vào các chân ngõ ra của 2 IC. Khi chân EN được đưa xuống mức thấp, các đầu ra giữ nguyên trạng thái hiện tại. Theo đó, sau khi dữ liệu đã được dịch vào thanh ghi dịch, sử dụng chân EN để chuyển dữ liệu này từ thanh ghi dịch vào thanh ghi lưu trữ, khóa dữ liệu vào các chân đầu ra. Đưa chân EN lên mức cao để sao chép dữ liệu từ thanh ghi dịch vào thanh ghi lưu trữ và cập nhật giá trị vào các đầu ra. Đưa chân EN xuống mức thấp để hoàn tất quá trình khóa, dữ liệu đầu ra giữ nguyên trạng thái hiện tại cho đến khi có tín hiệu khóa mới nhằm đảm bảo tính ổn định của dữ liệu đầu ra.

- Chân A0 được kết nối với chân của nút nhấn điều khiển CAO\_DIEM cho phép hoạt động ở chế độ cao điểm.

- Chân A1 được kết nối với chân của nút nhấn điều khiển BINH\_THUONG cho phép hoạt động ở chế độ bình thường.

- Chân A2 được kết nối với chân của nút nhấn điều khiển BAN\_DEM cho phép hoạt động ở chế độ ban đêm.

- Chân A3 được kết nối với chân của nút nhấn điều khiển CHUYEN cho phép chuyển đổi thủ công các trạng thái của đèn giao thông khi mạch đang ở chế độ cao điểm. Ở các chế độ khác, nút nhấn này không hoạt động.

- Chân 5V được kết nối với nguồn thông qua adapter 5V – 3A.

- 2 chân GND2 và GND3 được nối đất.

### **3.3.4. Khối nguồn**

Các linh kiện trong mạch đòi hỏi sử dụng nguồn một chiều, do đó có nhiều phương án để có thể lựa chọn như sử dụng nguồn từ pin, ắc quy, nguồn từ cổng USB, nguồn từ mạch nguồn thiết kế riêng,… Tuy nhiên, để cho hệ thống đơn giản, dễ lắp đặt và thiết kế, tác giả chọn sử dụng nguồn từ một adapter 5V – 3A để cấp nguồn cho hệ thống. Đây là thiết bị có thể cung cấp một nguồn điện ổn định và liên tục mà không lo bị gián đoạn do sụt áp và cũng đơn giản hơn so với việc thiết kế một mạch nguồn riêng. Không giống như pin, adapter không cần phải thay thế thường xuyên, giảm thiểu thời gian và chi phí bảo trì. Adapter có khả năng cung cấp nguồn điện với công suất cao hơn pin và ắc quy, adapter cũng cho phép chuyển đổi điện áp từ lưới điện xoay chiều sang một chiều với hiệu suất cao và ít tổn hao năng lượng. Ngoài ra, nguồn từ cổng USB không thể cung cấp đủ công suất, cổng USB không được thiết kế để cung cấp nguồn liên tục. Khi sử dụng cổng USB, nếu thiết bị nguồn bị tắt hoặc ngắt kết nối, hệ thống đèn giao thông sẽ bị gián đoạn, do đó sử dụng cổng USB là không phù hợp cho ứng dụng này. Trong mạch, khối nguồn được thiết kế đơn giản như hình 3.6 dưới đây.

**A diagram of a circuit

Description automatically generated**

**Hình 3.6.** Khối nguồn

## **3.4. Thiết kế phần mềm**

Với mô hình và sơ đồ nguyên lý đã nghiên cứu trong các phần trên, tôi xây dựng và đưa ra lưu đồ giải thuật tóm gọn các bước xử lý như hình 3.7. Hình 3.7 là lưu đồ thể hiện chương trình chính của đồ án.

A diagram of a diagram

Description automatically generated

**Hình 3.7.** Lưu đồ giải thuật chính

Đầu tiên, tác giả thiết lập các chân I/O cho Arduino để điều khiển các module ngoại vi như IC 74HC595 và các LED, khởi tạo các biến sử dụng trong chương trình, khởi tạo mã 7 đoạn, khởi động Serial để giao tiếp với máy tính qua cổng UART, thiết lập các giá trị ban đầu, thiết lập Timer 1 tạo ra ngắt với chu kỳ 100ms để đếm ngược và cập nhật các biến liên quan. Việc thực thi chương trình sẽ được giải quyết thông qua 3 chương trình con sau:

- Chương trình con “dem\_thoi\_gian()” dùng để quản lý và điều khiển thời gian hoạt động của các chế độ đèn giao thông.

- Chương trình con “hienthi()” dùng để hiển thị thông tin đếm ngược trên các LED 7 đoạn,

- Chương trình con “kt\_nut\_nhan()” dùng để kiểm tra trạng thái của các nút nhấn và thiết lập chế độ hoạt động của hệ thống đèn giao thông.

Công việc của chương trình chính là khởi tạo phần cứng và các biến, truyền các giá trị tham số xuống các hàm chức năng để sử dụng cho việc thực thi mạch. Dưới đây là lưu đồ giải thuật cho cả bốn chương trình con “dem\_thoi\_gian()” (hình 3.8), “hienthi()” (hình 3.9), “kt\_nut\_nhan()” (hình 3.10) và “kt\_chuyen()” (hình 3.11).

Lưu đồ bên dưới (hình 3.8) mô tả chương trình con “dem\_thoi\_gian()”, thể hiện quá trình quản lý thời gian cho các trạng thái của đèn giao thông.

A diagram of a flowchart

Description automatically generated

**Hình 3.8.** Lưu đồ giải thuật mô tả chương trình con “dem\_thoi\_gian()”

Khi “tt\_ kich == 0” tức là mạch ở chế độ bình thường thì chương trình đếm số lần ngắt và chuyển đổi trạng thái đèn giao thông khi thời gian đạt đến ngưỡng xác định. “ngat” là một biến đếm, tăng lên mỗi lần hàm được gọi. Hàm này được gọi mỗi 100ms do Timer 1 được cấu hình với khoảng thời gian 100000µs, tức là 100ms. Khi “ngat” đạt đến giá trị là 10, tức là đã qua 1s (vì 100ms × 10 = 1000ms = 1s) thì tiến hành các bước sau:

- Gọi hàm “dem\_nguoc\_ngang()” để đếm ngược và điều chỉnh trạng thái cho đèn hàng ngang.

- Gọi hàm “dem\_nguoc\_doc()” để đếm ngược và điều chỉnh trạng thái cho đèn cột dọc.

- Đặt lại giá trị của biến “ngat” về 0 để bắt đầu đếm lại.

Khi “tt\_kich == 2” tức là mạch ở chế độ ban đêm thì đèn giao thông màu vàng sẽ sáng nhấp nháy để báo hiệu, thể hiện mạch đang hoạt động theo chế độ cảnh báo thay vì điều khiển giao thông. Ở đây, biến “ngat” vẫn tăng mỗi lần hàm được gọi, tương tự như trong chế độ bình thường. Tiếp theo sẽ kiểm tra giá trị của biến “ngat”, nếu:

- “ngat” nhỏ hơn 5 (tương đương 500ms) thì tất cả các đèn đều tắt.

- “ngat” nằm trong khoảng từ 5 đến 9 (tương đương 400ms) thì đèn vàng sẽ bật để tạo hiệu ứng nhấp nháy.

- “ngat” đạt đến giá trị 9 (tương đương 900ms) thì đặt giá trị “ngat” về 0 để bắt đầu chu kỳ nhấp nháy mới.

Có thể thấy, tổng thời gian của một chu kỳ nhấp nháy là 500ms (tắt) + 400ms (bật) = 900ms. Khoảng thời gian này tạo ra một nhịp nhấp nháy rõ ràng, giúp người giao thông nhận biết được tín hiệu cảnh báo.

Sau lưu đồ của chương trình con “dem\_thoi\_gian()”, tôi tiếp tục trình bài lưu đồ giải thuật của chương trình con “hienthi()”, được thể hiện trong hình 3.9 bên dưới. Chương trình này có nhiệm vụ hiển thị các LED 7 đoạn bằng cách điều khiển từng LED thông qua các byte dữ liệu và hàm “delay()”. Chương trình sẽ khởi tạo vòng lặp for với biến đếm “dem”chạy từ 0 đến “lap - 1”, có nghĩa là lặp “lap” lần, mỗi lần lặp là một chu kỳ hiển thị tất cả các LED. Sau đó, chương trình sẽ xuất dữ liệu để điều khiển các LED bằng cách gọi hàm “xuat2so()” để gửi byte bật LED (ví dụ “0xfe” cho LED0), dùng lệnh “delay(1);” để đợi 1ms nhằm duy trì trạng thái bật rồi gọi hàm “xuat2so()” để tắt LED bằng cách gửi byte “0xff”. Sau đó, chương trình lặp lại các dòng lệnh này với sự thay đổi byte bật LED tương ứng cho các LED để điều khiển cho tất cả các LED. Nếu số lần đếm “dem” lớn hơn hoặc bằng số lần lặp “lap” thì kết thúc chương trình.

A diagram of a computer program

Description automatically generated

**Hình 3.9.** Lưu đồ giải thuật mô tả chương trình con “hienthi()”

Lưu đồ giải thuật của chương trình con “kt\_nut\_nhan()” được mô tả như hình 3.10 bên dưới. Chương trình này dùng để kiểm tra trạng thái của các nút nhấn và thực hiện các hành động tương ứng với từng trường hợp. Ban đầu, chương trình sẽ kiểm tra nút nhấn chế độ bình thường BINH\_THUONG có được nhấn hay không. Nếu được nhấn thì thiết lập các biến “tt\_kich”, “tt\_ngang”, “tt\_doc” về 0, gán các giá trị cho các biến “gh\_xanh\_ngang”, “gh\_re\_ngang” là 12s, “gh\_vang\_ngang” là 3s, “gh\_do\_ngang” là 15s, “gh\_xanh\_doc”, “gh\_re\_doc” là 12s, “gh\_vang\_doc” là 3s, “gh\_do\_doc” là 15s, gán các giá trị này vào các biến tương ứng như “xanh\_ngang”, “re\_ngang”, “vang\_ngang”, “do\_ngang”, “xanh\_doc”, “re\_doc”, “vang\_doc”, “do\_doc”. Nếu nút nhấn này không được nhấn thì kiểm tra nút chế độ cao điểm CAO\_DIEM được nhấn hay không. Nếu được nhấn thì thiết lập các chân “tt\_kich” về 1 và “tt\_chuyen” về 0, tắt tất cả các LED 7 đoạn, cho đèn vàng ở cả 4 cột đèn giao thông sáng trong 3s rồi cho đèn xanh, đèn rẽ trái ở hàng ngang và đèn đỏ hướng cột dọc bật. Nếu nút nhấn này không được nhấn thì chuyển sang kiểm tra nút nhấn chế độ ban đêm BAN\_DEM. Nếu nút nhấn này được nhấn thì thiết lập “tt\_kich” về 2, tắt tất cả LED 7 đoạn. Nếu nút nhấn này cũng không được nhấn nữa thì kết thúc chương trình. Ở trong chương trình này, biến “tt\_kich” được sử dụng để xác định chế độ hoạt động hiện tại của hệ thống đèn giao thông. Giá trị “tt\_kich” được gán bằng 0 khi nút nhấn BINH\_THUONG được nhấn. Giá trị “tt\_kich” được gán bằng 1 khi nút nhấn CAO\_DIEM được nhấn. Giá trị “tt\_kich” được gán bằng 2 khi nút nhấn BAN\_DEM được nhấn.

A diagram of a diagram

Description automatically generated

**Hình 3.10.**  Lưu đồ giải thuật mô tả chương trình con “kt\_nut\_nhan()”

Hình 3.11 là lưu đồ giải thuật biểu diễn chương trình con “kt\_chuyen()”. Hàm “kt\_chuyen()” sẽ kiểm tra trạng thái của nút nhấn CHUYEN. Khi nút này được nhấn, hàm sẽ thay đổi trạng thái của hệ thống đèn giao thông giữa hai chế độ khác nhau. Trong chương trình này, khi chạy thực tế thì do tốc độ của vi điều khiển quá nhanh, khi nhấn nút nhấn CHUYEN thì do thời gian nhấn phím dài nên vi điều khiển thực hiện thao tác đảo LED liên tục, ta sẽ nhìn thấy LED trong các mạch LED giao thông luôn sáng cho đến khi ta buông phím, hoặc ta nhấn nhanh thì trạng thái đảo của LED không thể xác định rõ ràng. Để xác định rõ ràng thì phải chống dội bằng cách kiểm tra phím nhấn, nếu có thì delay, rồi xử lý và kiểm tra buông phím [2]. Tiếp theo tăng giá trị “tt\_chuyen” lên 1. Nếu “tt\_chuyen” lớn hơn 1 thì đặt lại về 0. Điều này tạo ra hai trạng thái chuyển xen kẽ nhau là 0 và 1. Khi chuyển thì cho đèn vàng ở các cột sáng trong 3 giây để báo hiệu thay đổi trạng thái, giúp người đi đường kịp thời xử lý tình huống để không gây ra tai nạn. Nếu “tt\_chuyen” là 0 thì đèn xanh, đèn cho phép rẽ trái hàng ngang và đèn đỏ cột dọc sáng. Ngược lại, nếu “tt\_chuyen” là 1 thì đèn đỏ hàng ngang, đèn xanh và đèn cho phép rẽ trái cột dọc sáng.

**A diagram of a flowchart

Description automatically generated**

**Hình 3.11.**  Lưu đồ giải thuật mô tả chương trình con “kt\_chuyen()”

# **CHƯƠNG 4. KẾT QUẢ**

## **4.1. KẾT QUẢ MÔ HÌNH THI CÔNG**

Sau khi vẽ sơ đồ nguyên lý của hệ thống, mô phỏng và thiết kế mạch in sử dụng phần mềm Protues, tác giả thu được sơ đồ mạch in lớp dưới được thể hiện trong hình 4.1.

A computer circuit board with many different colored lines

Description automatically generated

**Hình 4.1.** Sơ đồ mạch in lớp dưới

Sau khi thực hiện in và rữa mạch thì tác giả được mạch in của hệ thống với kích thước là 142.875 × 121.92mm. Hình 4.2 là hình ảnh mạch in lớp dưới của hệ thống.

A close up of a circuit board

Description automatically generated

**Hình 4.2.** Mạch in lớp dưới của hệ thống

## **4.2. HOẠT ĐỘNG CỦA HỆ THỐNG**

Ở đây, để thuận tiện cho việc quan sát các hình trong phần này, quy ước hàng ngang là hàng mà ta có thể thấy rõ đếm ngược của LED 7 đoạn 2 số hay mạch khi nhìn chính diện, cột dọc là hướng vuông góc với hàng ngang. Ban đầu, khi cắm adapter vào ổ điện hoặc khi nhấn nút nhấn xanh, mạch hoạt động ở chế độ làm việc bình thường. Đèn xanh ở các hàng ngang sáng, đồng thời đèn đỏ ở các cột dọc cũng sáng. Hình 4.3 thể hiện hoạt động của mạch khi mới có dòng điện chạy qua.

A group of traffic lights on a cardboard box

Description automatically generated

**Hình 4.3.** Mạch khi mới cấp nguồn hoặc khi nhấn nút nhấn xanh

Khi nhấn nút nhấn đỏ, mạch hoạt động ở chế độ cao điểm. Khi mới nhấn nút nhấn đỏ thì các đèn vàng sáng trong 3 giây, giúp các phương tiện tham gia giao thông có thời gian để thay đổi tốc độ. Sau đó đèn xanh và đèn cho phép rẽ trái ở hàng ngang sáng, cho phép các phương tiện trên tuyến đường hàng ngang lưu thông, đồng thời đèn đỏ ở cột dọc sáng, dừng tất cả phương tiện trên tuyến đường cột dọc. Hình 4.4 và 4.5 thể hiện mạch khi nhấn nút nhấn đỏ và 3 giây sau đó.

A group of white post with lights on it

Description automatically generated

**Hình 4.4.** Mạch sau khi nhấn nút nhấn đỏ

A group of white post with lights on it

Description automatically generated

**Hình 4.5.** Mạch sau khi nhấn nút nhấn đỏ 3 giây

Khi nhấn nút đen thì thay đổi trạng thái của 2 cột đèn giao thông đối diện. Nhấn lần đầu thì đèn vàng ở 4 cột sáng 3 giây. Sau đó, đèn xanh và đèn cho phép rẽ trái ở hàng ngang tắt, sáng đèn đỏ, đồng thời đèn đỏ ở cột dọc tắt, sáng đèn xanh và đèn cho phép rẽ trái. Nếu nhấn lần nữa thì đảo trạng thái. Cứ thế lặp đi lặp lại nếu tiếp tục nhấn nút nhấn đen. Hình 4.6 và 4.7 thể hiện cho nút nhấn này khi nhấn lần đầu.

A group of traffic lights on a cardboard box

Description automatically generated

**Hình 4.6.** Mạch sau khi nhấn nút nhấn đen

A group of traffic lights on a cardboard box

Description automatically generated

**Hình 4.7.** Mạch sau khi nhấn nút nhấn đen 3 giây

Khi nhấn nút nhấn vàng, mạch chuyển sang trạng thái hoạt động vào ban đêm, các đèn vàng ở 4 cột chớp tắt liên tục. Hình 4.8 thể hiện cho điều này.

A group of traffic lights on a cardboard box

Description automatically generated

**Hình 4.8.** Mạch khi nhấn nút nhấn vàng

# **CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

## **5.1. KẾT LUẬN**

Trong đề tài “*Thiết kế mạch đèn giao thông*”, tác giả đã thiết kế được một hệ thống đếm ngược thời gian chờ, có tín hiệu cho phép rẽ trái, có các nút nhấn cho phép điều khiển các chế độ hoạt động của mạch, điều khiển luồng giao thông vào giờ cao điểm. Bên cạnh đó, hệ thống cũng góp phần cải thiện luồng giao thông và giảm thiểu tình trạng ùn tắc giao thông. Tuy đã thiết kế thành công, nhưng hệ thống vẫn còn tồn tại một số nhược điểm như chưa đạt được độ thông minh, chưa tích hợp thêm các tính năng mới và được thiết kế không được đẹp mắt, sơ sài.

## **5.2. HƯỚNG PHÁT TRIỂN**

Đề tài vẫn tồn tại nhiều hạn chế mà tác giả vẫn chưa giải quyết được. Nếu có thời gian, tác giả sẽ thiết kế mô hình đẹp mắt hơn và thân thiện hơn. Bên cạnh đó, để cải thiện những hạn chế và phù hợp hơn với thời đại chuyển đổi số, hệ thống có thể có những hướng phát triển như sau:

- Tích hợp hoặc sử dụng cảm biến như radar, phát hiện tốc độ hay cảm biến thời tiết để thu thập dữ liệu giao thông hay điều chỉnh tín hiệu đèn dựa trên tình trạng thực tế của giao thông. Sử dụng camera thông minh để phát hiện và ghi nhận các sự kiện trong thời gian thực từ đó có thể đưa ra những phương án kịp thời khi có sự cố xảy ra.

- Hệ thống sẽ được kết nối hoặc tích hợp với các hệ thống giao thông thông minh, tích hợp các thiết bị thu thập thông tin.

- Hệ thống sẽ có tính năng điều khiển từ xa qua mạng hoặc bất kỳ kết nối không dây nào, có thể điều khiển tín hiệu giao thông theo thời gian thực.

# **TÀI LIỆU THAM KHẢO**

1. Nguyễn Đình Phú, Phan Vân Hoàn, Trương Ngọc Anh, “Vi điều khiển PIC 16F887: Giao tiếp LED, LCD, phím đơn, ma trận phím”, *Giáo trình Vi điều khiển PIC*, NXB Đại học Quốc gia TP. HCM, 2017, trang 119-220.
2. Arduino Docs. (2023, Nov. 11). *Universal Asynchronous Receiver-Transmitter (UART)* [Online]. Đường dẫn: [Universal Asynchronous Receiver-Transmitter (UART) | Arduino Documentation](https://docs.arduino.cc/learn/communication/uart/).
3. Ho Tam. (2018, Jun. 8). *Bài 4 – Timer – Counter* [Online]. Đường dẫn: [Bài 4 - Timer - Counter | Cùng học AVR (AVR tutorial) - Tài liệu AVR tiếng Việt (hocavr.com)](https://www.hocavr.com/2018/06/bai-4-timer-counter.html).
4. Datasheet Arduino UNO R3 Datasheet, Mar. 2024.

# **PHỤ LỤC**

#include <TimerOne.h>

int DATA  = 10;

int CK    = 11;

int EN    = 12;

int XANH\_NGANG  = 5;

int VANG\_NGANG  = 6;

int DO\_NGANG    = 7;

int XANH\_DOC  = 4;

int VANG\_DOC  = 3;

int DO\_DOC    = 2;

int RE\_NGANG  = 8;

int RE\_DOC    = 9;

int BINH\_THUONG = A1;

int CAO\_DIEM    = A0;

int BAN\_DEM     = A2;

int CHUYEN      = A3;

int ma7doan[] = { 0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92, 0x82, 0xf8, 0x80, 0x90 };

int led0, led1, led2, led3, led4, led5, led6, led7;

int tt\_ngang, tt\_doc;

int xanh\_ngang, re\_ngang, vang\_ngang, do\_ngang, xanh\_doc, re\_doc, vang\_doc, do\_doc;

int gh\_xanh\_ngang, gh\_re\_ngang, gh\_vang\_ngang, gh\_do\_ngang;

int gh\_xanh\_doc, gh\_re\_doc, gh\_vang\_doc,  gh\_do\_doc;

int ngat, tt\_kich, tt\_chuyen;

void xuat1byte(int x) {

  int d, i;

  d = x;

  for (i = 0; i < 8; i++) {

    if ((d & 0x80) == 0x80) digitalWrite(DATA, HIGH);

    else                    digitalWrite(DATA, LOW);

    digitalWrite(CK, HIGH);   delayMicroseconds(5);

    digitalWrite(CK, LOW);    delayMicroseconds(5);

    d = d << 1;

  }

}

void xuat2so(int l1, int l0) {

  xuat1byte(l1);

  xuat1byte(l0);

  digitalWrite(EN, HIGH);   delayMicroseconds(5);

  digitalWrite(EN, LOW);    delayMicroseconds(5);

}

void hienthi(int lap) {

  int dem;

  for (dem = 0; dem < lap; dem++) {

    xuat2so(led0, 0xfe);  delay(1);   xuat2so(led0, 0xff);

    xuat2so(led1, 0xfd);  delay(1);   xuat2so(led1, 0xff);

    xuat2so(led2, 0xfb);  delay(1);   xuat2so(led2, 0xff);

    xuat2so(led3, 0xf7);  delay(1);   xuat2so(led3, 0xff);

    xuat2so(led4, 0xef);  delay(1);   xuat2so(led4, 0xff);

    xuat2so(led5, 0xdf);  delay(1);   xuat2so(led5, 0xff);

    xuat2so(led6, 0xbf);  delay(1);   xuat2so(led6, 0xff);

    xuat2so(led7, 0x7f);  delay(1);   xuat2so(led7, 0xff);

  }

}

void kt\_nut\_nhan() {

  if (digitalRead(BINH\_THUONG) == 0) {

    tt\_kich = 0;

    tt\_ngang = 0;

    tt\_doc = 0;

    gh\_xanh\_ngang = 12;      gh\_re\_ngang = 12;    gh\_vang\_ngang = 3;    gh\_do\_ngang = 15;

    gh\_xanh\_doc   = 12;      gh\_re\_doc   = 12;    gh\_vang\_doc   = 3;    gh\_do\_doc   = 15;

    xanh\_ngang = gh\_xanh\_ngang;   re\_ngang = gh\_re\_ngang;   vang\_ngang = gh\_vang\_ngang;   do\_ngang = gh\_do\_ngang;

    xanh\_doc   = gh\_xanh\_doc;     re\_doc   = gh\_re\_doc;     vang\_doc   = gh\_vang\_doc;     do\_doc   = gh\_do\_doc;

  }

  else if (digitalRead(CAO\_DIEM) == 0) {

    tt\_kich = 1;

    tt\_chuyen = 0;

    led0 = 0xff;

    led1 = 0xff;

    led2 = 0xff;

    led3 = 0xff;

    led4 = 0xff;

    led5 = 0xff;

    led6 = 0xff;

    led7 = 0xff;

    digitalWrite(XANH\_NGANG,  LOW);

    digitalWrite(RE\_NGANG,    LOW);

    digitalWrite(VANG\_NGANG,  HIGH);

    digitalWrite(DO\_NGANG,    LOW);

    digitalWrite(XANH\_DOC,    LOW);

    digitalWrite(RE\_DOC,      LOW);

    digitalWrite(VANG\_DOC,    HIGH);

    digitalWrite(DO\_DOC,      LOW);

    delay(3000);

    digitalWrite(XANH\_NGANG,  HIGH);

    digitalWrite(RE\_NGANG,    HIGH);

    digitalWrite(VANG\_NGANG,  LOW);

    digitalWrite(DO\_NGANG,    LOW);

    digitalWrite(XANH\_DOC,    LOW);

    digitalWrite(RE\_DOC,      LOW);

    digitalWrite(VANG\_DOC,    LOW);

    digitalWrite(DO\_DOC,      HIGH);

    while (digitalRead(CAO\_DIEM) == 0);

  }

  else if (digitalRead(BAN\_DEM) == 0) {

    tt\_kich = 2;

    led0 = 0xff;

    led1 = 0xff;

    led2 = 0xff;

    led3 = 0xff;

    led4 = 0xff;

    led5 = 0xff;

    led6 = 0xff;

    led7 = 0xff;

  }

}

void kt\_chuyen() {

  if (digitalRead(CHUYEN) == 0) {

    delay(20);

    if (digitalRead(CHUYEN) == 0) {

      tt\_chuyen++;

      if (tt\_chuyen > 1) tt\_chuyen = 0;

      digitalWrite(XANH\_NGANG,  LOW);

      digitalWrite(RE\_NGANG,    LOW);

      digitalWrite(VANG\_NGANG,  HIGH);

      digitalWrite(DO\_NGANG,    LOW);

      digitalWrite(XANH\_DOC,    LOW);

      digitalWrite(RE\_DOC,      LOW);

      digitalWrite(VANG\_DOC,    HIGH);

      digitalWrite(DO\_DOC,      LOW);

      delay(3000);

      if (tt\_chuyen == 0) {

        digitalWrite(XANH\_NGANG,  HIGH);

        digitalWrite(RE\_NGANG,    HIGH);

        digitalWrite(VANG\_NGANG,  LOW);

        digitalWrite(DO\_NGANG,    LOW);

        digitalWrite(XANH\_DOC,    LOW);

        digitalWrite(RE\_DOC,      LOW);

        digitalWrite(VANG\_DOC,    LOW);

        digitalWrite(DO\_DOC,      HIGH);

      } else {

        digitalWrite(XANH\_NGANG,  LOW);

        digitalWrite(RE\_NGANG,    LOW);

        digitalWrite(VANG\_NGANG,  LOW);

        digitalWrite(DO\_NGANG,    HIGH);

        digitalWrite(XANH\_DOC,    HIGH);

        digitalWrite(RE\_DOC,      HIGH);

        digitalWrite(VANG\_DOC,    LOW);

        digitalWrite(DO\_DOC,      LOW);

      }

    }

  }

}

void dem\_nguoc\_ngang() {

  if (tt\_ngang == 0) {

    if (xanh\_ngang > 0) {

      xanh\_ngang--;

      led0 = ma7doan[xanh\_ngang / 10];

      led1 = ma7doan[xanh\_ngang % 10];

      led2 = ma7doan[xanh\_ngang / 10];

      led3 = ma7doan[xanh\_ngang % 10];

      digitalWrite(XANH\_NGANG,  HIGH);

      digitalWrite(RE\_NGANG,    HIGH);

      digitalWrite(VANG\_NGANG,  LOW);

      digitalWrite(DO\_NGANG,    LOW);

    } else {

      xanh\_ngang = gh\_xanh\_ngang;

      vang\_ngang = gh\_vang\_ngang;

      digitalWrite(XANH\_NGANG,  LOW);

      digitalWrite(RE\_NGANG,    LOW);

      digitalWrite(VANG\_NGANG,  HIGH);

      digitalWrite(DO\_NGANG,    LOW);

      tt\_ngang = 1;

    }

  } else if (tt\_ngang == 1) {

    if (vang\_ngang > 0) {

      vang\_ngang--;

      led0 = ma7doan[vang\_ngang / 10];

      led1 = ma7doan[vang\_ngang % 10];

      led2 = ma7doan[vang\_ngang / 10];

      led3 = ma7doan[vang\_ngang % 10];

      digitalWrite(XANH\_NGANG,  LOW);

      digitalWrite(RE\_NGANG,    LOW);

      digitalWrite(VANG\_NGANG,  HIGH);

      digitalWrite(DO\_NGANG,    LOW);

    } else {

      do\_ngang = gh\_do\_ngang;

      vang\_ngang = gh\_vang\_ngang;

      digitalWrite(XANH\_NGANG,  LOW);

      digitalWrite(RE\_NGANG,    LOW);

      digitalWrite(VANG\_NGANG,  LOW);

      digitalWrite(DO\_NGANG,    HIGH);

      tt\_ngang = 2;

    }

  } else if (tt\_ngang == 2) {

    if (do\_ngang > 0) {

      do\_ngang--;

      led0 = ma7doan[do\_ngang / 10];

      led1 = ma7doan[do\_ngang % 10];

      led2 = ma7doan[do\_ngang / 10];

      led3 = ma7doan[do\_ngang % 10];

      digitalWrite(XANH\_NGANG,  LOW);

      digitalWrite(RE\_NGANG,    LOW);

      digitalWrite(VANG\_NGANG,  LOW);

      digitalWrite(DO\_NGANG,    HIGH);

    } else {

      do\_ngang = gh\_do\_ngang;

      re\_ngang = gh\_re\_ngang;

      digitalWrite(XANH\_NGANG,  HIGH);

      digitalWrite(RE\_NGANG,    HIGH);

      digitalWrite(VANG\_NGANG,  LOW);

      digitalWrite(DO\_NGANG,    LOW);

      tt\_ngang = 0;

    }

  }

}

void dem\_nguoc\_doc() {

  if (tt\_doc == 0) {

    if (do\_doc > 0) {

      do\_doc--;

      led4 = ma7doan[do\_doc / 10];

      led5 = ma7doan[do\_doc % 10];

      led6 = ma7doan[do\_doc / 10];

      led7 = ma7doan[do\_doc % 10];

      digitalWrite(XANH\_DOC,  LOW);

      digitalWrite(RE\_DOC,    LOW);

      digitalWrite(VANG\_DOC,  LOW);

      digitalWrite(DO\_DOC,    HIGH);

    } else {

      do\_doc = gh\_do\_doc;

      re\_doc = gh\_re\_doc;

      led4 = ma7doan[xanh\_doc / 10];

      led5 = ma7doan[xanh\_doc % 10];

      led6 = ma7doan[xanh\_doc / 10];

      led7 = ma7doan[xanh\_doc % 10];

      digitalWrite(XANH\_DOC,  HIGH);

      digitalWrite(RE\_DOC,    HIGH);

      digitalWrite(VANG\_DOC,  LOW);

      digitalWrite(DO\_DOC,    LOW);

      tt\_doc = 1;

    }

  } else if (tt\_doc == 1) {

    if (xanh\_doc > 0) {

      xanh\_doc--;

      led4 = ma7doan[xanh\_doc / 10];

      led5 = ma7doan[xanh\_doc % 10];

      led6 = ma7doan[xanh\_doc / 10];

      led7 = ma7doan[xanh\_doc % 10];

      digitalWrite(XANH\_DOC,  HIGH);

      digitalWrite(RE\_DOC,    HIGH);

      digitalWrite(VANG\_DOC,  LOW);

      digitalWrite(DO\_DOC,    LOW);

    } else {

      vang\_doc = gh\_vang\_doc;

      xanh\_doc = gh\_xanh\_doc;

      led4 = ma7doan[vang\_doc / 10];

      led5 = ma7doan[vang\_doc % 10];

      led6 = ma7doan[vang\_doc / 10];

      led7 = ma7doan[vang\_doc % 10];

      digitalWrite(XANH\_DOC,  LOW);

      digitalWrite(RE\_DOC,    LOW);

      digitalWrite(VANG\_DOC,  HIGH);

      digitalWrite(DO\_DOC,    LOW);

      tt\_doc = 2;

    }

  } else if (tt\_doc == 2) {

    if (vang\_doc > 0) {

      vang\_doc--;

      led4 = ma7doan[vang\_doc / 10];

      led5 = ma7doan[vang\_doc % 10];

      led6 = ma7doan[vang\_doc / 10];

      led7 = ma7doan[vang\_doc % 10];

      digitalWrite(XANH\_DOC,  LOW);

      digitalWrite(RE\_DOC,    LOW);

      digitalWrite(VANG\_DOC,  HIGH);

      digitalWrite(DO\_DOC,    LOW);

    } else {

      vang\_doc = gh\_vang\_doc;

      do\_doc = gh\_do\_doc;

      led4 = ma7doan[do\_doc / 10];

      led5 = ma7doan[do\_doc % 10];

      led6 = ma7doan[do\_doc / 10];

      led7 = ma7doan[do\_doc % 10];

      digitalWrite(XANH\_DOC,  LOW);

      digitalWrite(RE\_DOC,    LOW);

      digitalWrite(VANG\_DOC,  LOW);

      digitalWrite(DO\_DOC,    HIGH);

      tt\_doc = 0;

    }

  }

}

void dem\_thoi\_gian() {

  if (tt\_kich == 0) {

    ngat++;

    if (ngat >= 10) {

      dem\_nguoc\_ngang();

      dem\_nguoc\_doc();

      ngat = 0;

    }

  }

  else if (tt\_kich == 2) {

    ngat++;

    if (ngat < 5) {

      digitalWrite(XANH\_NGANG,  LOW);

      digitalWrite(RE\_NGANG,    LOW);

      digitalWrite(VANG\_NGANG,  LOW);

      digitalWrite(DO\_NGANG,    LOW);

      digitalWrite(XANH\_DOC,    LOW);

      digitalWrite(RE\_DOC,      LOW);

      digitalWrite(VANG\_DOC,    LOW);

      digitalWrite(DO\_DOC,      LOW);

    } else if (ngat < 9) {

      digitalWrite(XANH\_NGANG,  LOW);

      digitalWrite(RE\_NGANG,    LOW);

      digitalWrite(VANG\_NGANG,  HIGH);

      digitalWrite(DO\_NGANG,    LOW);

      digitalWrite(XANH\_DOC,    LOW);

      digitalWrite(RE\_DOC,      LOW);

      digitalWrite(VANG\_DOC,    HIGH);

      digitalWrite(DO\_DOC,      LOW);

    } else {

      ngat = 0;

    }

  }

}

void setup() {

  Serial.begin(9600);

  pinMode(DATA, OUTPUT);

  pinMode(CK,   OUTPUT);

  pinMode(EN,   OUTPUT);

  pinMode(XANH\_NGANG, OUTPUT);

  pinMode(VANG\_NGANG, OUTPUT);

  pinMode(DO\_NGANG,   OUTPUT);

  pinMode(XANH\_DOC, OUTPUT);

  pinMode(VANG\_DOC, OUTPUT);

  pinMode(DO\_DOC,   OUTPUT);

  pinMode(RE\_NGANG, OUTPUT);

  pinMode(RE\_DOC,   OUTPUT);

  pinMode(BINH\_THUONG,  INPUT);

  pinMode(CAO\_DIEM,     INPUT);

  pinMode(BAN\_DEM,      INPUT);

  pinMode(CHUYEN,       INPUT);

  ngat = 0;

  tt\_kich = 0;

  tt\_ngang = 0;

  tt\_doc = 0;

  gh\_xanh\_ngang = 12;      gh\_re\_ngang = 12;    gh\_vang\_ngang = 3;    gh\_do\_ngang = 15;

  gh\_xanh\_doc   = 12;      gh\_re\_doc   = 12;    gh\_vang\_doc   = 3;    gh\_do\_doc   = 15;

  xanh\_ngang = gh\_xanh\_ngang;   re\_ngang = gh\_re\_ngang;   vang\_ngang = gh\_vang\_ngang;   do\_ngang = gh\_do\_ngang;

  xanh\_doc   = gh\_xanh\_doc;     re\_doc   = gh\_re\_doc;     vang\_doc   = gh\_vang\_doc;     do\_doc   = gh\_do\_doc;

  Timer1.initialize(100000);

  Timer1.attachInterrupt(dem\_thoi\_gian);

  digitalWrite(XANH\_NGANG,  LOW);

  digitalWrite(RE\_NGANG,    LOW);

  digitalWrite(VANG\_NGANG,  LOW);

  digitalWrite(DO\_NGANG,    LOW);

  digitalWrite(XANH\_DOC,    LOW);

  digitalWrite(RE\_DOC,      LOW);

  digitalWrite(VANG\_DOC,    LOW);

  digitalWrite(DO\_DOC,      LOW);

  led0 = 0xff;

  led1 = 0xff;

  led2 = 0xff;

  led3 = 0xff;

  led4 = 0xff;

  led5 = 0xff;

  led6 = 0xff;

  led7 = 0xff;

  hienthi(1);

}

void loop() {

  hienthi(2);

  kt\_nut\_nhan();

  if (tt\_kich == 1) {

    kt\_chuyen();

  }

}