

**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG ĐIỆN – ĐIỆN TỬ**

BÁO CÁO DỰ ÁN

VI XỬ LÝ

**ĐỀ TÀI: Thiết kế hệ thống khóa mở cửa bằng mật khẩu sử dụng vi điều khiển MCS-51 với giao tiếp máy tính**

**GVHD: PGS.TS Nguyễn Quốc Cường**

**Nhóm: 13**

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **MSSV** | **Lớp** |
| **Phạm Minh Tuấn** | **20222191** | **EE–TN** |

**Ngày nộp báo cáo: 31/12/2024**

**Hà Nội, 2024**

**MỤC LỤC**

[Chương I: Giới thiệu dự án 5](#_Toc186195540)

[1. Yêu cầu dự án 5](#_Toc186195541)

[2. Mục tiêu 5](#_Toc186195542)

[3. Bảng kế hoạch thực hiện 5](#_Toc186195543)

[Chương II: Cơ sở lý thuyết 5](#_Toc186195544)

[1. Mcs-51 8051 6](#_Toc186195545)

[1.1 Đặc điểm 6](#_Toc186195546)

[1.2 Sơ đồ cấu trúc 6](#_Toc186195547)

[1.3 Sơ đồ chân 7](#_Toc186195548)

[1.4 Tổ chức bộ nhớ 7](#_Toc186195549)

[1.5 Thanh ghi chức năng đặc biệt (SFR) 9](#_Toc186195550)

[1.6 Tạo tốc độ truyền thông tin (Baud Rates) 15](#_Toc186195551)

[2. LED 7 thanh 16](#_Toc186195552)

[3. Truyền tin nối tiếp UART 17](#_Toc186195553)

[3.1 Truyền dữ liệu 17](#_Toc186195554)

[3.2 Truyền dữ liệu ở 8051 17](#_Toc186195555)

[Chương III: Thiết kế hệ thống 19](#_Toc186195556)

[1. Sơ đồ khối chức năng: 19](#_Toc186195557)

[2. Chức năng từng khối phần cứng: 19](#_Toc186195558)

[2.1. Vi điều khiển AT89S52: 19](#_Toc186195559)

[2.3 Mạch Reset: 19](#_Toc186195560)

[2.4 LED báo hiệu 19](#_Toc186195561)

[2.5 Loa: 19](#_Toc186195562)

[2.6 LED 7 thanh: 19](#_Toc186195563)

[2.7 Khối nút bấm: 20](#_Toc186195564)

[2.8 Cổng UART: 20](#_Toc186195565)

[3. Thiết kế chi tiết từng khối: 20](#_Toc186195566)

[3.1. Mạch phát xung: 20](#_Toc186195567)

[3.2. Mạch Reset: 20](#_Toc186195568)

[3.4. Loa: 21](#_Toc186195569)

[3.5: Led 7 thanh: 22](#_Toc186195570)

[3.6. Khối nút bấm: 22](#_Toc186195571)

[3.7. Khối UART: 23](#_Toc186195572)

[4. Thiết kế chương trình 23](#_Toc186195573)

[1. Giới thiệu Tổng quan 23](#_Toc186195574)

[2. Phân tích Chương trình 23](#_Toc186195575)

[3. Quản lý ID và Lựa chọn Chế độ 26](#_Toc186195576)

[4. Quản lý LED 7 Thanh và Hiển thị Thông tin 27](#_Toc186195577)

[5. Chức năng Bảo mật và Kiểm tra Mật khẩu 28](#_Toc186195578)

[6. Xử lý Sự kiện và Điều khiển Ngoài 29](#_Toc186195579)

[7. Điều khiển Thời gian 30](#_Toc186195580)

[8. Tổng kết và Cải tiến 30](#_Toc186195581)

[Chương IV: Kết quả thực nghiệm 30](#_Toc186195582)

[1. Kịch bản đánh giá 30](#_Toc186195583)

[1.1 Kịch bản 30](#_Toc186195584)

[1.2 Kết quả dự kiến 31](#_Toc186195585)

[2. Thử nghiệm 31](#_Toc186195586)

[3. Đánh giá 31](#_Toc186195587)

[Chương V: Kết luận và Hướng phát triển 31](#_Toc186195588)

[1. Kết luận 31](#_Toc186195589)

[2. Hướng phát triển 31](#_Toc186195590)

[1. Sơ đồ cấu trúc 6](#_Toc186155476)

[2. Sơ đồ chân của chip 8051 6](#_Toc186155477)

[**3**. Bộ nhớ chương trình (ROM) 8](#_Toc186155478)

[4. Bộ nhớ dữ liệu 8](#_Toc186155479)

[5. Không gian Lower 128 8](#_Toc186155480)

[6.Không gian thanh ghi đặc biệt 9](#_Toc186155481)

[7. Bản đồ bộ nhớ thanh ghi đặc biệt 9](#_Toc186155482)

[8. Seven segment common cathode display 16](#_Toc186155483)

[9. Seven segment common anode display 16](#_Toc186155484)

[10. Truyền song song 17](#_Toc186155485)

[11. Truyền nối tiếp 17](#_Toc186155486)

[12. Sơ đồ cổng khối nối tiếp 18](#_Toc186155487)

[13. Sơ đồ khối chức năng 19](#_Toc186155488)

[14. Thiết kế mạch phát xung 20](#_Toc186155489)

[15. Thiết kế mạch Reset 20](#_Toc186155490)

[16. Thiết kế mạch Led báo hiệu 21](#_Toc186155491)

[17. Thiết kế mạch loa 21](#_Toc186155492)

[18. Thiết kế mạch led 7 thanh 22](#_Toc186155493)

[19. Thiết kế mạch nút bấm 22](#_Toc186155494)

# Chương I: Giới thiệu dự án

## 1. Yêu cầu dự án

**-** Cho phép người dùng nhập mật khẩu và xác nhận mật khẩu.

Hiển thị quá trình nhập mật khẩu trên màn hình LED.

**-** Phát âm thanh báo hiệu trạng thái đúng/sai của mật khẩu qua còi.

**-** Thiết lập mật khẩu và ID khách hàng trên máy tính và truyền thông qua UART.

**-** Ghi nhận và lưu trữ thông tin khách hàng và thời gian mở cửa.

**-** Yêu cầu nâng cao: Hệ thống có khả năng mã hóa thông tin giao tiếp để đảm bảo an toàn.

## 2. Mục tiêu

**-** Hiểu rõ cấu trúc, nguyên lý hoạt động của bộ vi điều khiển MCS-51.

**-**  Mạch hoạt động ổn định.

## 3. Bảng kế hoạch thực hiện

|  |  |  |  |
| --- | --- | --- | --- |
| **Tuần** | **Nội dung** | **Kết quả** | **Ghi chú** |
|
| 8/9/10 | Tìm hiểu về vi điều khiển MCS-51, nắm được cách hoạt động của vi điều khiển MCS-51 | Nắm được cách hoạt động, các mã lệnh cơ bản |  |
| 11 | Tìm hiểu sơ đồ khối chức năng, vẽ sơ đồ khối chức năng | Vẽ sơ đồ khối | Sơ đồ khối chức năng |
| 12 | Vẽ mạch mô phỏng | Mô phỏng trên Proteus, code trên Keil uVision 5 | code mô phỏng chức năng bằng C, sau đó dịch qua Assembly |
| 13 | Hoàn thành các khối chức năng, hoàn thành phần code cho LED 7 thanh, đặt linh kiện | Mạch mô phỏng hoàn chỉnh |  |
| 14 | Lắp mạch, test mạch | Mạch hoàn chỉnh |  |
| 15 | Hoàn thiện báo cáo, test các chức năng |  |  |

# Chương II: Cơ sở lý thuyết

## 1. Mcs-51 8051

### 1.1 Đặc điểm

* 8-bit CPU được tối ưu hóa cho ứng dụng điều khiển
* Có khả năng xử lý 1-bit logic
* 64K Program Memory address space
* 64K Data Memory address space
* Program Memory có 4K bytes trên chip
* Data Ram có 128 bytes trên chip
* 32 đường I/O hai chiều và có thể định địa chỉ riêng biệt
* 2 bộ timer/counters 16-bit
* Cổng truyền tin nối tiếp UART
* Cấu trúc ngắt gồm 6-nguồn/5-vecto với 2 mức độ ưu tiên
* Bộ phát xung trên chip

### 1.2 Sơ đồ cấu trúc

A diagram of a computer program

Description automatically generated

1. Sơ đồ cấu trúc

### 1.3 Sơ đồ chân

A white rectangular object with numbers and letters

Description automatically generated

2. Sơ đồ chân của chip 8051

### 1.4 Tổ chức bộ nhớ

- Kiến trúc bộ nhớ của 8051 là kiến trúc Harvard

- Không gian của Program Memory và Data Memory là tách rời.

- 8051 có các bộ nhớ trong

* Bộ nhớ chứa code lệnh: ROM
* Bộ nhớ chứa dự liệu đọc ghi: RAM

- Có khả năng quản lý bộ nhớ code và data ngoài qua

* Bus data 8 bit AD0-AD7 (~ P0.0-P0.7)
* Bus địa chỉ tới 16 bit
  + AD0-AD7
  + A8-A15 (~ P2.0-P2.7)
* Bus điều khiển PSEN, ALE, RD (P3.7), WR (P3.6)

- Chỉ sử dụng 1 trong 2 bộ nhớ code lệnh

- Có thể sử dụng đồng thời bộ nhớ dữ liệu trong và ngoài

#### 1.4.1. Program Memory

|  |  |
| --- | --- |
| A diagram of a house  Description automatically generated  **3**. Bộ nhớ chương trình (ROM) | Các ngắt được định sẵn 1 vị trí cố định trong Program Memory.  Vị trí của chương trình phục vụ ngắt được đặt cách nhau 8 byte: Nếu ISR đủ ngắn có thể vừa trong khoảng 8 byte đó, các ISR dài có thể sử dụng lệnh JUMP để bỏ qua các vị trí ngắt đã đước sắp xếp sẵn.  4K byte thấp nhất của Program Memory có thể ở trong on-chip ROM hoặc External ROM. Việc lựa chọn này phụ thuộc vào chân EA được nối với VCC hay VSS:  + EA = VCC: địa chỉ từ 0000H-0FFFH sẽ được truy cập ở internal ROM, địa chỉ 1000H-FFFFH sẽ ở external ROM.  + EA = VSS: toàn bộ địa chỉ sẽ được truy cập ở external ROM. |

#### 1.4.2 Data Memory

|  |  |
| --- | --- |
| A diagram of a computer component  Description automatically generated with medium confidence  4. Bộ nhớ dữ liệu | Không gian bộ nhớ internal Ram được chia làm 3 khối: Lower 128, Upper 128 và không gian SFR.  Với 8051, không gian Upper 128 không tồn tại |

**a) Lower 128 (00H-7FH)**

|  |  |
| --- | --- |
| 32 byte (00H-1FH) thấp nhất được nhóm thành 4 băng thanh ghi, mỗi bang gồm 8 thanh ghi từ R0-R7. Hai bit của PSW sẽ lựa chọn bang thanh ghi nào được sử dụng.  16 byte kế tiếp hình thành 1 khối không gian bộ nhớ có thể định địa chỉ theo bit.  Tất cả các byte có thể được truy cập bằng **địa chỉ trực tiếp hoặc gián tiếp.** | **A diagram of a block diagram  Description automatically generated**  5. Không gian Lower 128 |

**b) SFR (80H-FFH)**

|  |  |
| --- | --- |
| Các thanh ghi đặc biệt (SFR) gồm các chốt cổng, các bộ định thời gian, điều khiển ngoại vị, ...  SFR chỉ có thể được truy cập bằng **địa chỉ trực tiếp.**  Có 16 địa chỉ trong không gian SFR vừa được đánh địa chỉ theo byte và đánh địa chỉ theo bit. Các thanh ghi có địa chỉ kết thúc với 000B được đánh địa chỉ theo bit. | **A diagram of a port  Description automatically generated**  6.Không gian thanh ghi đặc biệt |

### 1.5 Thanh ghi chức năng đặc biệt (SFR)

A map with text and numbers

Description automatically generated with medium confidence

7. Bản đồ bộ nhớ thanh ghi đặc biệt

#### 1.5.1 PSW: Program Status Word



|  |  |  |
| --- | --- | --- |
| CY | PSW.7 | Cờ nhớ (Carry Flag) |
| AC | PSW.6 | Cờ nhớ phụ (Auxiliary Carry Flag) |
| F0 | PSW.5 | Flag 0 dùng cho mục đích chung |
| RS1 | PSW.4 | Bit chọn băng thanh ghi 1 |
| RS0 | PSW.3 | Bit chọn bang thanh ghi 0 |
| OV | PSW.2 | Cờ tràn số (Overflow Flag) |
| - | PSW.1 | Cờ người dung tự định nghĩa |
| P | PSW.0 | Cờ kiểm tra tính chẵn lẻ (Parity Flag) |

\* **Ghi chú lựa chọn băng thanh ghi của bit RS0,RS1**

|  |  |  |  |
| --- | --- | --- | --- |
| RS1 | RS0 | Băng thanh ghi | Địa chỉ |
| 0 | 0 | 0 | 00H-07H |
| 0 | 1 | 1 | 08H-0FH |
| 1 | 0 | 2 | 10H-17H |
| 1 | 1 | 3 | 18H-1FH |

#### 1.5.2 PCON: Power control register – Not bit addressable



|  |  |
| --- | --- |
| SMOD | Bit gấp đôi baud rate (tốc độ truyền dữ liệu). Nếu Timer 1 được sử dụng để tạo baud rate và SMOD =1, baud rate sẽ được gấp đôi khi Serial Port (Cổng truyền tin nối tiếp hoạt động ở mode 1, 2, 3. |
| - | Chưa được định nghĩa, dùng cho hoạt động trong tương lai |
| - | Chưa được định nghĩa, dùng cho hoạt động trong tương lai |
| - | Chưa được định nghĩa, dùng cho hoạt động trong tương lai |
| GF1 | Bit cờ mục đích chung |
| GF2 | Bit cờ mục đích chung |
| PD | Bit Power Down Mode. PD = 1, kích hoạt hoạt động Power Down (chỉ có ở CHMOS) |
| IDL | Bit Idle Mode. IDL =1, kich hoạt động Idle Mode (chỉ có ở CHMOS) |

\* Chế độ Power Down có mức ưu tiên cao hơn chế độ Idle.

#### 1.5.3 IE: Interrupt enable register – Bit addressable



|  |  |  |
| --- | --- | --- |
| EA | IE.7 | Bit cho phép tổng (Enable all). Nếu EA=0, không ngắt nào được thừa nhận. Nếu EA=1, với mỗi nguồn ngắt sẽ được cho phép/ ko cho phép bởi từng bit cho phép của chúng. |
| - | IE.6 | Chưa được định nghĩa, dùng cho hoạt động trong tương lai |
| ET2 | IE.5 | Bit cho phép ngắt Timer 2 (chỉ có ở 8052) |
| ES | IE.4 | Bit cho phép ngắt công truyền tin nối tiếp |
| ET1 | IE.3 | Bit cho phép ngắt Timer 1 |
| EX1 | IE.2 | Bit cho phép ngắt ngoại vi 1 |
| ET0 | IE.1 | Bit cho phép ngắt Timer 0 |
| EX0 | IE.0 | Bit cho phép ngắt ngoại vi 0 |

\*Bit =0, không cho phép ngắt. Bit =1, cho phép ngắt

\* **NGẮT**

Các bước để sử đụng một ngắt:

1. Set EA=1
2. Set bit cho phép ngắt của ngắt tương ứng lên 1
3. Bắt đầu ISR tại địa chỉ vector của ngắt tương ứng (bảng dưới)

|  |  |
| --- | --- |
| Nguồn ngắt | Địa chỉ vector |
| IE0 | 0003H |
| TF0 | 000BH |
| IE1 | 0013H |
| TF1 | 001BH |
| RI & TI | 0023H |
| TF2 & EXF2 | 002BH |

Với các ngắt ngoài, các chân và phải =1

#### 1.5.4 IP: Interrupt priority register – Bit addressable



|  |  |  |
| --- | --- | --- |
| - | IP.7 | Chưa được định nghĩa, dùng cho hoạt động trong tương lai |
| - | IP.6 | Chưa được định nghĩa, dùng cho hoạt động trong tương lai |
| PT2 | IP.5 | Bit định nghĩa mức độ ưu tiên cho ngắt Timer 2 (8052 only) |
| PS | IP.4 | Bit định nghĩa mức độ ưu tiên cho ngắt Serial Port |
| PT1 | IP.3 | Bit định nghĩa mức độ ưu tiên cho ngắt Timer 1 |
| PX1 | IP.2 | Bit định nghĩa mức độ ưu tiên cho ngắt ngoại vi 1 |
| PT0 | IP.1 | Bit định nghĩa mức độ ưu tiên cho ngắt Timer 0 |
| PX0 | IP.0 | Bit định nghĩa mức độ ưu tiên cho ngắt ngoại vi 2 |

\* Bit =0, ngắt có mức độ ưu tiên thấp hơn. Bit =1, ngắt có mức độ ưu tiên cao hơn

\* Khi một ISR được thực hiện, nó không thể bị ngắt bởi ngắt có cùng mức độ ưu tiếp hoặc thấp hơn.

\* Mức độ ưu tiên thực hiện của các ngắt khi có các ngắt cùng mức độ ưu tiên xuất hiện cùng lúc: (Từ cao xuống thấp): IE0🡪TF0🡪IE1🡪TF1🡪RI &TI🡪TF2&EXF2

#### 1.5.5 TCON: Timer/Counter control register – Bit addressable



|  |  |  |
| --- | --- | --- |
| TF1 | TCON.7 | Cờ tràn Timer 1. TF1=1 khi Timer/Counter 1 tràn. |
| TR1 | TCON.6 | Bit điều khiển Timer 1. TR1 = 1/0 thì ON/OFF Timer/Counter 1 (Set by software) |
| TF0 | TCON.5 | Cờ tràn Timer 0. TF0=1 khi Timer/Counter 0 tràn. |
| TR0 | TCON.4 | Bit điều khiển Timer 0. TR0 = 1/0 thì ON/OFF Timer/Counter 0 (Set by software) |
| IE1 | TCON.3 | Cờ sườn ngắt ngoài 1. IE1=1 khi phát hiện sườn ngắt ngoài, IE1=0 khi ngắt được thực hiện. (Set by hardware) |
| IT1 | TCON.2 | Cờ điều khiển loại ngắt 1. IT1=1/0 tương ứng ngắt ngoài là ngắt theo sườn xuống/ mức thấp (Set by software) |
| IE0 | TCON.1 | Cờ sườn ngắt ngoài 0. IE0=1 khi phát hiện sườn ngắt ngoài, IE0=0 khi ngắt được thực hiện. (Set by hardware) |
| IT0 | TCON.0 | Cờ điều khiển loại ngắt 0. IT0=1/0 tương ứng ngắt ngoài là ngắt theo sườn xuống/ mức thấp (Set by software) |

#### 1.5.6 TMOD: Timer/Counter Mode control register – Not bit addressable

A black and white rectangular object with black text

Description automatically generated

|  |  |
| --- | --- |
| GATE | Khi TRx (trong TCON)=1 và GATE=1, Timer/Counterx chỉ hoạt động khi chân INTx đang ở mức cao (Hardware control)  Khi GATE=0, Timer/Counter chỉ hoạt động khi TRx=1 (Software control) |
| C/ | Lựa chọn Timer/Counter. C/=0, Timer hoạt động, C/=1, Counter hoạt động |
| M1 | Bit chọn chế độ |
| M0 | Bit chọn chế độ |

\* **Bảng ghi chú chế độ hoạt động của Timer/Counter**

|  |  |  |  |
| --- | --- | --- | --- |
| M1 | M0 | Chế độ hoạt động | |
| 0 | 0 | 0 | 13-bit Timer |
| 0 | 1 | 1 | 16-bit Timer/Counter |
| 1 | 0 | 2 | 8-bit Auto-Reload Timer/Counter |
| 1 | 1 | 3 | (Timer 0) TL0 là 1 bộ 8-bit Timer/Counter được điều khiển bởi các bit điều khiển Timer 0, TH0 là 1 bộ 8-bit Timer/Counter được điều khiển bởi các bit điều khiển Timer 1 |
| 1 | 1 | 3 | (Timer 1) Timer/Counter 1 dừng hoạt động |

\* **Khởi tạo Timer**

**a) Timer/ Counter 0**

**Timer**

|  |  |  |  |
| --- | --- | --- | --- |
| **MODE** | **TIMER 0**  **FUNCTION** | **TMOD** | |
| **INTERNAL**  **CONTROL** | **EXTERNAL  CONTROL** |
| 0 | 13-bit Timer | 00H | 08H |
| 1 | 16-bit Timer | 01H | 09H |
| 2 | 8-bit Auto-Reload | 02H | 0AH |
| 3 | Two 8-bit Timers | 03H | 0BH |

**Counter**

|  |  |  |  |
| --- | --- | --- | --- |
| **MODE** | **Counter 0**  **FUNCTION** | **TMOD** | |
| **INTERNAL**  **CONTROL** | **EXTERNAL  CONTROL** |
| 0 | 13-bit Timer | 04H | 0CH |
| 1 | 16-bit Timer | 05H | 0DH |
| 2 | 8-bit Auto-Reload | 06H | 0EH |
| 3 | One 8-bit Counter | 07H | 0FH |

**b) Timer/ Counter 0**

**Timer**

|  |  |  |  |
| --- | --- | --- | --- |
| **MODE** | **TIMER 1**  **FUNCTION** | **TMOD** | |
| **INTERNAL**  **CONTROL** | **EXTERNAL  CONTROL** |
| 0 | 13-bit Timer | 00H | 80H |
| 1 | 16-bit Timer | 10H | 90H |
| 2 | 8-bit Auto-Reload | 20H | A0H |
| 3 | Does not run | 30H | B0H |

**Counter**

|  |  |  |  |
| --- | --- | --- | --- |
| **MODE** | **Counter 1**  **FUNCTION** | **TMOD** | |
| **INTERNAL**  **CONTROL** | **EXTERNAL  CONTROL** |
| 0 | 13-bit Timer | 40H | C0H |
| 1 | 16-bit Timer | 50H | D0H |
| 2 | 8-bit Auto-Reload | 60H | E0H |
| 3 | Not available | - | - |

#### 1.5.7 SCON: Serial port control register – Bit addressable



|  |  |  |
| --- | --- | --- |
| SM0 | SCON.7 | Xác định chế độ Serial Port |
| SM1 | SCON.6 | Xác định chế độ Serial Port |
| SM2 | SCON.5 | Cho phép tính năng giao tiếp với nhiều bộ xử lý ở mode 2&3.  Ở mode 2&3, nếu SM2=1 thì bit RI sẽ không được kích hoạt nếu bit dữ liệu thứ 9 được nhận (RB8) là 0.  Ở mode 1, nếu SM2=1 thì bit RI sẽ không được kích hoạt nếu bit dừng hợp lệ không được nhận .  Ở mode 0, SM2=0. |
| REN | SCON.4 | EA=1/0, cho phép/ không cho phép việc nhận thông tin (Set by software) |
| TB8 | SCON.3 | Lưu giữ bit thứ 9 sẽ được truyền tải trong mode 2&3. (Set by software) |
| RB8 | SCON.2 | Lưu giữ bit thứ 9 được nhận trong mode 2&3.  Trong mode 1, nếu SM2=0, RB8 là bit dừng đã được nhận. Không hoạt động trong mode 0. |
| TI | SCON.1 | Cờ ngắt truyền. |
| RI | SCON.0 | Cờ ngắt nhận. |

\* **Bảng ghi chú các chế độ làm việc của Serial Port**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SM0 | SM1 | Mode | Mô tả | Baud Rate |
| 0 | 0 | 0 | SHIFT REGISTER | Fosc./12 |
| 0 | 1 | 1 | 8-Bit UART | Variable |
| 1 | 0 | 2 | 9-Bit UART | Fosc./64  Fosc./32 |
| 1 | 1 | 3 | 9-Bit UART | Variable |

\* **Serial Port SET-UP:**

|  |  |  |
| --- | --- | --- |
| **MODE** | **SCON** | **SM2** |
| 0 | 10H | Môi trường 1 bộ xử lý  (SM2 = 0) |
| 1 | 50H |
| 2 | 90H |
| 3 | D0H |
| 0 | NA | Môi trường đa bộ xử lý  (SM2 = 1) |
| 1 | 70H |
| 2 | B0H |
| 3 | F0H |

### 1.6 Tạo tốc độ truyền thông tin (Baud Rates)

#### 1.6.1 Serial Port ở Mode 0:

Để Serial Port hoạt động ở chế độ này không cần phải khởi tạo bất kì bộ Timer/Counters nào. Chỉ cần thanh ghi SCON được định nghĩa.

Tốc độ truyền thông tin:

#### 1.6.2 Serial Port ở Mode 1:

Ở Mode 1, Baud Rate có thể thay đổi được, có thể được tạo bởi Timer1/Timer2.

\* **Sử dụng Timer/Counter 1 để tạo Baud Rate**

Timer 1 phải hoạt động ở Mode 2 (Auto-Reload).

Tốc độ truyền thông tin:

* Nếu SMOD = 0, K = 1
* Nếu SMOD = 1, K = 2

Thông thường người dùng sẽ biết trước giá trị Baud Rate và cần khởi tạo giá trị cho TH1, ta có công thức tính TH1:

\* **Sử dụng Timer/Counter 2 để tạo Baud Rate (8052 only)**

#### 1.6.3 Serial Port ở Mode 2

Ở Mode 2, Baud Rate đã được định sẵn giá trị là hoặc Osc Fred, phụ thuộc vào giá trị của bit SMOD.

Trong Mode này, không cần sử dụng bộ Timer, xung đến từ

* Nếu SMOD = 1, Baud Rate = Osc Fred.
* Nếu SMOD = 0, Baud Rate = Osc Fred.

Cách để set bit SMOD: ORL PCON,#80H.

#### 1.6.4 Serial Port ở Mode 3

Ở Mode 3, Baud Rate có thể thay đổi được và cách khởi tạo tương tự ở Mode 1.

## 2. LED 7 thanh

Có 2 loại LED 7 thanh: Common Anode (CA) & Common Cathode (CC)

|  |  |
| --- | --- |
| A circuit board with numbers and wires  Description automatically generated  8. Seven segment common cathode display | A digital display with numbers and symbols  Description automatically generated with medium confidence  9. Seven segment common anode display |

* Common Anode: Các LED sẽ sẽ khi chân tương ứng của LED có mức logic thấp.
* Common Cathode: Các LED sẽ sáng khi chân tương ứng của LED có mức logic cao

\* **Bảng ghi chú các giá trị hiển thị của LED 7 thanh**

**a) Common Anode**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Giá trị hiển thị** | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **DP** | **Giá trị Hexa**  **DP GFEDCBA** |
| Số 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | C0H |
| Số 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | F9H |
| Số 2 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | A4H |
| Số 3 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | B0H |
| Số 4 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 99H |
| Số 5 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 92H |
| Số 6 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 82H |
| Số 7 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | F8H |
| Số 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 80H |
| Số 9 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 90H |

## 3. Truyền tin nối tiếp UART

### 3.1 Truyền dữ liệu

Khi bộ vi xử lý làm việc với các thiết bị khác bên ngoài cần phải trao đổi truyền dữ liệu đến các thiết bị. Có 2 cách truyền dữ liệu:

* Truyền dữ liệu song song: Truyền dữ liệu liệu dưới dạng từng byte (8 bit) 1🡪 sử dụng 8 đường bus dữ liệu. Phương pháp này được sử dụng khi đường cáp không quá dài.
* Truyền dữ liệu nối tiếp: Truyền dữ liệu dưới dạng từng bit một 🡪 chỉ sử dụng 1 đường bus dữ liệu. Phương pháp này sử dụng khi hai hệ thống cách xa nhau.

Khi truyền dữ liệu nối tiếp, trước hết byte dữ liệu sẽ được chuyển thành các bit nối tiếp nhờ thanh ghi dịch vào-song song-ra-nối tiếp (PISO). Tiếp theo, dữ liệu được truyền qua một đường dữ liệu đơn. Như vậy ở đầu thu cũng phải có một thanh ghi dịch vào-nối tiếp-ra-song song (SIPO).

|  |  |
| --- | --- |
| A diagram of a parallel interface  Description automatically generated  10. Truyền song song | A diagram of a computer  Description automatically generated with medium confidence  11. Truyền nối tiếp |

\* **Truyền dữ liệu đơn công, bán song công và song công (song công hoàn toàn)**

* Truyền dữ liệu đơn công (Simplex): Quá trình truyền dữ liệu chỉ có thể là phát hoặc thu dữ liệu. VD: Máy in chỉ có thể nhận dữ liệu từ máy tính.
* Truyền dữ liệu bán song công (Half – Duplex): Quá trình truyền dữ liệu vừa có thể là phát hoặc là nhận dữ liêu. Nhưng tại một thời điểm dữ liệu chỉ có được truyền 1 kiểu phát/thu.
* Truyền dữ liệu song công hay song công hoàn toàn (Full-Duplex): Quá trình truyền dữ liệu vừa có thể là phát hoặc thu dữ liệu. Tại một thời điểm dữ liệu truyền được truyền cả hai chiều phát/thu. Ở chế độ này muốn truyền cả 2 chiều động thời phải sử dụng 2 đường dữ liêu, 1 đường phát, 1 đường thu.

### 3.2 Truyền dữ liệu ở 8051

Vi xử lý 8051 có cơ chế truyền dữ liệu kiểu nối tiếp thông qua 2 chân RXD (P3.0) và chân TXD (P3.1). Chân nối tiếp có chức năng cơ bản là thực hiện việc chuyển đổi dữ liệu song song sang nối tiếp khi phát và chuyển đổi dữ liệu từ nối tiếp sang song song khi thu.

Ngoài ra truyền dữ liệu ở 8051 là truyền dữ dữ liệu song công, tức là có khả năng phát và thu dữ liệu đồng thời.

Phần mềm sử dụng 2 thanh ghi chức năng là SBUF và SCON để truy xuất cổng nối tiếp. Bộ đệm thanh ghi SBUF có địa chỉ byte là 99H, trên thực tế bao gồm 2 bộ đệm. Việc ghi lên thanh ghi SBUF sẽ nạp dữ liệu để phát và việc đọc SBUF sẽ truy xuất dữ liệu đã nhận được. Điều này có nghĩa là ta có hai thanh ghi riêng rẽ và phân biệt: thanh ghi chỉ phát (chỉ ghi) hay bộ đẹm phát và thanh ghi thu (chỉ đọc) hay bộ đệm thu.

A diagram of a bus

Description automatically generated

12. Sơ đồ cổng khối nối tiếp

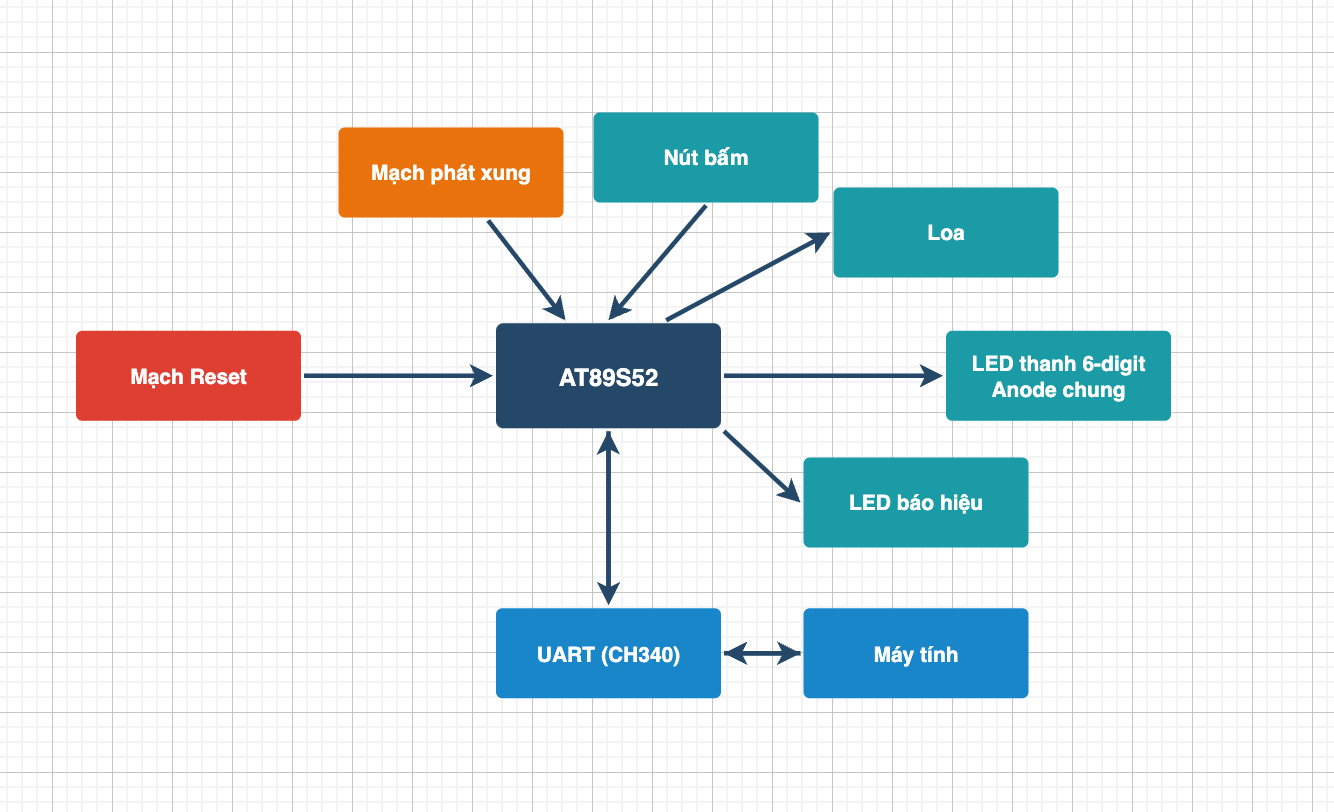
**Hình 2.12 Sơ đồ khối cổng nối tiếp**

Thanh ghi SCON có địa chỉ byte là 98H chứa các bit trạng thái và bit điều khiển. Các bit điều khiển sẽ thiết lập chế độ hoạt động của cổng nối tiếp còn các bt trạng thái chri ra sự kết thúc việc thu hoặc phát 1 ký tự. Các bit trạng thái được kiểm tra bởi phần mềm hoặc được lập trình để tạo ra ngắt.

Việc thiết lập các chế độ truyền dữ liệu và tốc độ truyền dữ liệu (Baud Rates) đã được nói đến ở chương II.1

# Chương III: Thiết kế hệ thống

## 1. Sơ đồ khối chức năng:



13. Sơ đồ khối chức năng

## 2. Chức năng từng khối phần cứng:

### 2.1. Vi điều khiển AT89S52:

* Vi điều khiển thuộc họ MCS-51, có tác dụng điều khiển hoạt động của các phần tử trong mạch

2.2 Mạch phát xung:

* Sử dụng thạch anh kết hợp cùng tụ để phát xung đồng bộ cho các hoạt động của vi xử lý.

### 2.3 Mạch Reset:

* Khởi động lại chương trình của vi điều khiển khi mới cấp nguồn hoặc khi ấn nút

### 2.4 LED báo hiệu

* Báo hiệu mạch đang hoạt động/ tắt. Khi mạch hoạt động, 3 Led tương ứng với các nút bấm đều sáng. Khi bấm nút nào, led tương ứng nút đấy sẽ sáng

### 2.5 Loa:

Loa báo hiệu các hoạt động của mạch:

* Kêu nhiều lần trong 5 giây khi nhập sai mật khẩu
* Kêu 1 lần trong 500ms khi nhập đúng mật khẩu

### 2.6 LED 7 thanh:

* Hiển thị giá trị số từ 0 đến 9 khi nhập mật khẩu.
* Led bên trái hiển thị ID, Led bên phải hiển thị các chữ số trong mật khẩu

### 2.7 Khối nút bấm:

Gồm 3 nút bấm:

* Nút bấm 1
  + Khi ấn thì giá trị số trên LED 7 thanh tăng lên 1.
  + Khi LED hiển thị số 9, nếu ấn nút 1 thì giá trị LED trở về số 0.
* Nút bấm 2
  + Nút ấn xác nhận các chữ số của mật khẩu là giá trị đang hiển thị của LED
  + Sau khi ấn, giá trị số trên LED trở về 0. Đến lần nhấn thứ 6, kiểm tra mật khẩu để quyết định việc mở cửa, sau đó reset mạch để chuẩn bị cho lần nhập mật khẩu tiếp theo.

### 2.8 Cổng UART:

* Chuyển dữ liệu từ vi xử lý lên máy tính.
* Phục vụ nhận dữ liệu như ID, mật khẩu từ máy tính.

## 3. Thiết kế chi tiết từng khối:

### 3.1. Mạch phát xung:

A diagram of a circuit

Description automatically generated

14. Thiết kế mạch phát xung

* Mạch gồm thạch anh 11,0592MHz cùng với 2 tụ 22pF làm mạch phát xung. Để có Baud rate là 9600 bps, áp dụng công thức

BaudRate = (2^SMOD1 \* fosc) / (32 \* 12 \* [256 - TH1])

* Với SMOD1 = 0, fosc=11,0592MHz => TH1 = 253 = FDH
* Trong mạch thực tế, mạch phát xung được tích hợp vào trong đế nạp ISP/AVR

### 3.2. Mạch Reset:

A diagram of a circuit

Description automatically generated

15. Thiết kế mạch Reset

Chân RST của AT89S52 cần được nối với một điện trở 4,7-10kOhm nhằm đưa chân này về mức thấp khi mạch hoạt động bình thường. Tụ được mắc nhằm khi ấn BT5, tụ xả kéo chân RST lên mức cao trong một khoảng thời gian, kích hoạt chân RST, khi nút không được nhấn, VCC nạp cho tụ, dòng không đi vào RST, vi điều khiển hoạt động bình thường.

Trong mạch thực tế, mạch reset được tích hợp vào trong đế nạp ISP/AVR

3.3. LED báo hiệu:

A diagram of a circuit

Description automatically generated

16. Thiết kế mạch Led báo hiệu

Cả 3 Led sẽ sáng khi mạch hoạt động bình thường, Led sẽ nháy tương ứng với nút được bấm. Sử dụng điện trở hạn dòng 270 Ohm, do led siêu sáng có Imax = 30mA. I = 5V/270Ohm = ~ 18mA

### 3.4. Loa:

A diagram of a circuit

Description automatically generated

17. Thiết kế mạch loa

Khi chân P1.5 hoặc P1.6 được đặt lên mức cao, transistor NPN cho phép dòng chảy từ cực C sang E, khiến chân dương của BUZZER được cấp nguồn, làm cho BUZZER kêu.

### 3.5: Led 7 thanh:

A screenshot of a computer

Description automatically generated

18. Thiết kế mạch led 7 thanh

Gồm 6 Led 7 thanh Anode chung nối nhau, do dòng tối đa qua Led là 20mA, ta cần một điện trở nhỏ nhất là R = 5V/20mA = 250Ohm để hạn dòng => Chọn điện trở R = 270 Ohm. Mạch cũng bao gồm một điện trở thanh Respack-8 để kéo các chân P0 của vi điều khiển lên cao, tắt Led khi không sử dụng đến.

### 3.6. Khối nút bấm:

A diagram of a circuit

Description automatically generated

19. Thiết kế mạch nút bấm

Khối nút bấm bao gồm 3 nút bấm và 3 LED sẽ nhấp nháy khi nút được bấm. Cổng P1 được mặc định ở mức cao, khi bấm nút, nút sẽ kéo chân tương tự về mức thấp, nên ta dùng lệnh JNB để kiểm tra nút bấm. Khi bấm nút, đầu Anode của Led cũng đồng thời bị nút bấm kéo xuống mức thấp, làm Led tắt.

### 3.7. Khối UART:

Cài đặt SCON = 50H = 01010000B

* Bit 7 (SM0): 0
* Bit 6 (SM1): 1
* Bit 5 (SM2): 0
* Bit 4 (REN): 1
* Bit 3 (TB8): 0
* Bit 2 (RB8): 0
* Bit 1 (TI): 0
* Bit 0 (RI): 0

Với SM0 = 0 và SM1 = 1, cổng nối tiếp được đặt ở chế độ 1 (8-bit UART, tốc độ baud thay đổi được.

Bit REN = 1, cho phép bộ nhận của cổng nối tiếp hoạt động.

TB8/RB8 = 0: Bit thứ 9 để truyền/nhận dữ liệu được đặt giá trị 0.

TI/RI = 0: Cờ ngắt truyền/nhận không được kích hoạt.

## 4. Thiết kế chương trình

### 1. Giới thiệu Tổng quan

Hệ thống được thiết kế nhằm mục đích quản lý mật khẩu cho một thiết bị sử dụng vi điều khiển 8051. Người dùng có thể nhập mật khẩu và xác thực mật khẩu qua giao diện UART. Hệ thống sử dụng LED 7 thanh để hiển thị thông tin và tương tác với người dùng. Ngoài ra, người dùng có thể thay đổi mật khẩu và quản lý các chế độ truy cập.

#### 1.1 Mục tiêu và Yêu cầu

• Thiết lập hệ thống bảo mật bằng mật khẩu.  
• Quản lý và xác thực mật khẩu qua giao tiếp UART.  
• Hiển thị thông tin và mã hóa các giá trị số trên LED 7 thanh.  
• Phát triển chức năng thay đổi mật khẩu và kiểm tra mật khẩu nhập vào.

#### 1.2 Chức năng chính

• Nhập và xác thực mật khẩu.  
• Thay đổi mật khẩu cho người dùng.  
• Hiển thị thông tin mật khẩu và ID người dùng qua LED 7 thanh.  
• Chế độ bảo mật và thông báo sai mật khẩu.

### 2. Phân tích Chương trình

#### 2.1 Khởi tạo Hệ thống

Chương trình bắt đầu bằng việc cấu hình UART và các chân điều khiển.

|  |  |
| --- | --- |
| ```asm  ; Define LED pins  LED1 BIT P2.0  LED2 BIT P2.1  LED3 BIT P2.2  LED4 BIT P2.3  LED5 BIT P2.4  LED6 BIT P2.5  PIN\_ID EQU 35H  ID EQU 32H  TEMP EQU 33H  CNT EQU 34H  SU MACRO STR  MOV R1,#0  MOV DPTR, #STR  ACALL TRANSMIT\_STRING  ENDM UART\_SETUP:  MOV SCON, #50H   MOV TMOD, #20H   MOV TH1, #0FDH   SETB TR1   RET ``` | ;Định nghĩa các chân Led  ;Khới tạo các thanh ghi để lưu trữ các giá trị đặc biệt trong ROM  ;Đây là đoạn code để gửi một string đã được định nghĩa sang máy tính qua giao thức truyền thông uart.  ; Các thông số đã được giải thích ở mục III.3.7 |

#### 2.2 Nhập và Xác minh Mật khẩu cho vài trò ADMIN

* Hệ thống cho phép người dùng nhập mật khẩu qua UART. Mật khẩu nhập vào được so sánh với mật khẩu đã lưu trong bộ nhớ. Chức năng ADMIN\_VERIFY thực hiện việc này.
* Nếu mật khẩu đúng, hệ thống hiển thị "MAT KHAU DUNG" qua UART và chuyển sang chế độ đổi mật khẩu. Nếu sai, hệ thống hiển thị "MAT KHAU SAI".

|  |  |
| --- | --- |
| ADMIN\_VERIFY:  ACALL ENTER  SU SCREEN\_PASSWORD  MOV TEMP, #1 | ;In ra string được định nghĩa ở biến SCREEN\_PASSWORD |
| ADMIN\_VERIFY\_NEXT:  JNB RI,$  MOV A, SBUF  CLR RI  MOV B, A | ;Nhận ký tự từ máy tính qua UART, ký tự này được lưu trữ ở B |
| MOV DPTR, #PIN  MOV A, CNT  MOVC A, @A+DPTR  CJNE A, B, ADMIN\_VERIFY\_FAIL | ;Trỏ DPTR đến vị trí ký tự đầu tiên trong chuỗi PIN  ; Sau đó gán giá trị của CNT vào A, DPTR sẽ trỏ đến ký tự có index = CNT. Ký tự này được ghi vào A.  ;Tiến hành so sánh A và B, nếu sai, chạy ADMIN\_VERIFY\_FAIL để thông báo cho chương trình đã có ký tự sai |
| INC CNT  MOV R0, CNT  CJNE R0, #6, ADMIN\_VERIFY\_NEXT  SJMP ADMIN\_CORRECT | ;Khi CNT tăng đến 6 chương trình chuyển sang trạng thái kiểm tra mật khẩu |

#### 2.3 Cài đặt và Thay đổi Mật khẩu

* Hệ thống cho phép người dùng thay đổi mật khẩu thông qua chế độ cài đặt. Các mật khẩu được nhận từ UART và lưu vào bộ nhớ.
* Mật khẩu mới được lưu trữ tại các vị trí bộ nhớ được chỉ định, và có thể được sử dụng cho các lần đăng nhập sau.

|  |  |
| --- | --- |
| MODE\_SETUP\_PASSWORD:  ACALL ENTER  SU SCREEN\_MODE\_SETUP  ACALL ENTER  SU SCREEN\_ID  ACALL SELECT\_ID  ;ACALL ENTER  SU SCREEN\_PASSWORD  ACALL SETUP\_PASSWORD  ;SU SCREEN\_ENTER\_PASSWORD  AJMP MAIN  ;RET  SETUP\_PASSWORD:  MOV CNT,#0  CONTINUE\_RECEIVE:  JNB RI,$  MOV A,SBUF  CLR RI  CJNE A,#0DH,CONT\_SETUP\_PASSWORD  SJMP OUT\_SETUP\_PASSWORD  CONT\_SETUP\_PASSWORD:  MOV TEMP,A  MOV B,ID  MOV A,#6  MUL AB  ADD A,CNT  ADD A,#46H  MOV R0, A  MOV @R0, TEMP  INC CNT  SJMP CONTINUE\_RECEIVE  OUT\_SETUP\_PASSWORD:  RET | ; Chọn ID để nhập mật khẩu  ;Hàm đặt mật khẩu cho ID N. Hàm này sẽ ghi lại mật khẩu của người dùng đồng thời xuất ra ký tự mà admin nhập để người dùng đối chiếu  ;Mỗi ID sẽ có 6 thanh ghi để lưu trữ mật khẩu trong RAM. Ví dụ: đối với ID = N thanh ghi lưu trữ ký tự đầu tiên của ID sẽ là INDEX = 46H + N\*6H |

### 3. Quản lý ID và Lựa chọn Chế độ

#### 3.1 Lựa chọn Chế độ

Khi hệ thống khởi động, người dùng có thể chọn giữa hai chế độ:  
1. Nhập mật khẩu.  
2. Cài đặt mật khẩu mới.  
Hệ thống sẽ hiển thị thông báo yêu cầu người dùng chọn chế độ thông qua UART.

```asm  
START\_1:

JNB P1.2, MAIN ;Nhấn nút 1, chuyển đến chế độ nhập mật khẩu

JNB P1.3, ADMIN\_VERIFY ;Nhấn nút 2, chuyển đến cài đặt mật khẩu

Acall delay\_100ms

SJMP START\_1 ;Nếu chưa nhấn nút, lặp lại START\_1  
```

#### 3.2 Quản lý ID Người Dùng

ID người dùng được nhập qua UART và lưu vào bộ nhớ. Người dùng có thể thay đổi ID trong chế độ cài đặt.

```asm  
;enter id for entrance

id\_enter:

Acall check

MOV CNT, #0

JNB P1.3, pre\_main

sjmp id\_enter

check:

MOV DPTR, #LED\_7DOAN

ACALL chinh\_so ;Gọi hàm hiển thị số được chỉnh

ACALL clear\_all\_leds

MOV A, R0

MOV TEMP, R0

MOVC A, @A+DPTR

SETB LED1

MOV P0, A

MOV R4, #0

Ret  
```

### 4. Quản lý LED 7 Thanh và Hiển thị Thông tin

#### 4.1 Hiển thị Thông tin trên LED 7 Thanh

Thông qua mảng LED\_7DOAN, các ký tự số được mã hóa và hiển thị lên LED 7 thanh. Đoạn mã dưới đây cho thấy cách hiển thị số trên LED:

```asm  
MOV DPTR, #LED\_7DOAN  
MOV A, R0  
MOVC A, @A + DPTR  
MOV P0, A  
```

#### 4.2 Điều khiển và Xử lý LED

Các LED sẽ được bật hoặc tắt tùy theo các trạng thái của hệ thống, ví dụ như khi người dùng nhập mật khẩu đúng/sai. Các hàm clear\_all\_leds và chinh\_so giúp reset và điều chỉnh trạng thái của LED.

Ví dụ với hàm check1:

```asm

check\_1:

MOV DPTR, #LED\_7DOAN

CJNE R7, #1, check\_2

ACALL chinh\_so

ACALL clear\_all\_leds

MOV A, R0

MOVC A, @A+DPTR

SETB LED2

MOV P0, A

SJMP main\_loop

Hàm này sẽ trỏ DPTR đến địa chỉ của giá trị đầu tiên trong bảng led 7 thanh, tức giá trị tương ứng với số 0. Sau đó, hàm này kiểm tra xem R7 bằng bao nhiêu để bật Led tương ứng, ví dụ R7 = 6 => bật Led 6. Sau đó, gọi hàm chỉnh số, để người dùng có thể điều chỉnh số hiển thị lên Led, giá trị tương ứng được lưu vào R0, rồi chuyển vào A, lúc này giá trị của A sẽ bằng index của số cần hiển thị trong bảng Led 7 thanh, ta dùng lệnh movc và mov để đưa giá trị đấy ra P0 để hiển thị.

```

```asm

;Adjust number function==============================

chinh\_so:

JNB P1.2, increment ; Ki?m tra tr?ng thái P1.2

ACALL delay\_100ms

SJMP chinh\_so\_exit

increment:

JNB P1.2,$

ACALL delay\_100ms

INC R0 ; Tang giá tr? R0

CJNE R0, #10, chinh\_so\_exit ; N?u R0 không ph?i 10 thì thoát

MOV R0, #0 ; N?u R0 là 10 thì reset v? 0

chinh\_so\_exit:

NOP

RET

```

Hàm chỉnh số hoạt động bằng cách kiểm tra xem P1.2 có được nhấn hay không, nếu có, tăng R0, đến khi R0 bằng 10 thì lại trở về 0. Hàm có sử dụng một delay 100ms để tránh hiện tượng đè nút

### 5. Chức năng Bảo mật và Kiểm tra Mật khẩu

#### 5.1 Kiểm tra Mật khẩu

Mật khẩu nhập vào được so sánh với mật khẩu đã cài đặt trong bộ nhớ. Hệ thống sử dụng các chức năng như check\_pin để thực hiện việc xác minh này.

|  |  |
| --- | --- |
| check\_pin:  MOV A, R0    MOV TEMP, A  MOV B, ID  MOV A, #6  MUL AB  add A, R4  add A, #46H  mov R1, A  MOV A, TEMP    ADD A, #30H  MOV B, A  MOV A, @R1  CJNE A, B, sai\_pin  Acall r4\_inc  sjmp out\_check\_pin  sai\_pin:  CLR P2.7  out\_check\_pin:  RET  r4\_inc:  inc r4  cjne r4, #6, chinh\_so\_exit  mov r4, #0  ret | Chi tiết từng dòng  Sao chép giá trị trong thanh ghi R0 vào thanh ghi tích lũy A. Giá trị này thường là chỉ số của ký tự mật khẩu cần kiểm tra.  Lưu tạm giá trị trong A (chỉ số) vào thanh ghi TEMP để sử dụng lại sau.  Lấy giá trị định danh (ID), có thể là địa chỉ cơ bản của mảng mật khẩu lưu trữ trong bộ nhớ, và đưa vào thanh ghi B.  Đưa hằng số 6 vào thanh ghi A. Đây có thể là độ dài của mật khẩu hoặc bước nhảy giữa các phần tử trong bộ nhớ.  Thực hiện phép nhân A \* B và lưu kết quả vào A. Đây là phép tính để xác định địa chỉ cơ bản của mảng cần truy cập (mỗi phần tử có kích thước bằng 6 đơn vị).  Cộng giá trị trong R4 (có thể là offset của phần tử cần kiểm tra) vào kết quả trong A.  Cộng thêm giá trị 46H (địa chỉ cơ sở trong bộ nhớ lưu trữ) vào A. Dòng này thực hiện việc tính toán địa chỉ chính xác của ký tự cần kiểm tra trong bộ nhớ.  Lưu địa chỉ vừa tính được (trong A) vào thanh ghi R1 để truy cập dữ liệu tại địa chỉ đó sau.  Phục hồi giá trị chỉ số (từ TEMP) vào A để sử dụng trong bước tiếp theo.  Chuyển đổi ký tự nhập vào từ dạng số sang mã ASCII bằng cách cộng thêm giá trị 30H.  Lưu giá trị ASCII của ký tự nhập vào (từ A) vào thanh ghi B.  Lấy giá trị từ địa chỉ tính được (điểm đến của R1) và đưa vào thanh ghi A. Đây là ký tự của mật khẩu đúng để so sánh.  So sánh ký tự nhập vào (B) với ký tự đúng (A). Nếu không khớp, nhảy đến nhãn sai\_pin.  Gọi hàm con r4\_inc để tăng con trỏ kiểm tra R4 lên 1.  Nhảy qua nhãn sai\_pin và kết thúc kiểm tra hiện tại.  Nếu sai đặt R7 về 0 để làm tham số đối chiếu |

Nếu mật khẩu đúng, hệ thống sẽ chuyển sang chế độ tiếp theo. Nếu sai, hệ thống sẽ yêu cầu nhập lại mật khẩu hoặc thông báo sai mật khẩu.

### 6. Xử lý Sự kiện và Điều khiển Ngoài

#### 6.1 Xử lý Sự kiện Đầu vào

Các nút nhấn được kiểm tra liên tục để xác định chế độ hoạt động của hệ thống. Nếu người dùng nhấn nút, hệ thống sẽ thực hiện hành động tương ứng.

#### 6.2 Động tác Báo động và Reset Hệ thống

Khi mật khẩu sai, hệ thống sẽ kích hoạt một động tác báo động và reset. Chức năng bell\_sai\_pin sẽ kích hoạt báo động.

```asm  
bell\_sai\_pin:  
 setb P1.5

acall delay\_500ms  
 ajmp START  
```

### 7. Điều khiển Thời gian

#### 7.1 Cài đặt và Sử dụng Delay

Các hàm delay, như delay\_100ms, được sử dụng để tạo ra độ trễ trong hệ thống, giúp đồng bộ các hoạt động.

```asm  
delay\_100ms:

MOV R2, #250 ; Chu k? l?p ngoài d? t?o 0.1s

outer\_loop:

MOV R1, #250 ; Chu k? l?p bên trong

inner\_loop:

NOP ; No Operation (1 chu k? máy)

DJNZ R1, inner\_loop

DJNZ R2, outer\_loop

RET

; Hàm delay 500ms

DELAY\_500MS:

MOV R0, #25 ; 25 vòng l?p (m?i vòng ~20ms)

DELAY\_LOOP1:

MOV R1, #250 ; 250 vòng l?p

MOV R2, #250 ; 250 chu k? máy (tuong duong ~1ms)

INNER\_LOOP1:

DJNZ R2, INNER\_LOOP1 ; Gi?m R2, l?p d?n khi R2 = 0

DJNZ R1, DELAY\_LOOP1 ; Gi?m R1, l?p d?n khi R1 = 0

DJNZ R0, DELAY\_LOOP1 ; Gi?m R0, l?p 25 l?n

RET  
```

### 8. Tổng kết và Cải tiến

#### 8.1 Đánh giá Hệ thống

Hệ thống đã hoạt động ổn định và có thể thực hiện đầy đủ chức năng bảo mật như yêu cầu. Tuy nhiên, vẫn có một số điểm có thể cải tiến như:  
- Hỗ trợ mật khẩu phức tạp hơn.  
- Lưu mật khẩu vào EEPROM để tránh mất dữ liệu khi tắt nguồn.

#### **8**.2 Cải tiến và Phát triển

Trong tương lai, hệ thống có thể được mở rộng với các tính năng như:  
- Thêm các chế độ bảo mật khác (ví dụ: xác thực vân tay).  
- Cải tiến giao diện người dùng để dễ sử dụng hơn.

## 5. Danh mục linh kiện:

|  |  |  |
| --- | --- | --- |
| Tên linh kiện | Hình ảnh | Số lượng |
| AT89CS52 | Combo Vi Điều Khiển AT89S52 Và Đế IC 40P | 1 |
| CH340 | Module Chuyển Đổi USB To COM CH340 | 1 |
| Đế nạp ISP/AVR | Đế nạp T49 AVR ATMEGA16 - Nshop | 1 |
| BUZZER | Buzzer 12v | 2 |
| Transistor NPN | Transitor NPN 400 V TO-220 giá rẻ của Trung Quốc - Báo giá - THÀNH PHẦN GNS | 2 |
| LED 7 thanh Anode chung 2 số | LED 7 THANH 0.56 INCH 2 SỐ ANODE CHUNG Điện Tử 360(E360) | 1 |
| Button | Nút Nhấn 2 Chân Vuông | 3 |
| Led báo hiệu | LED 5MM SIÊU SÁNG MÀU XANH LÁ (10 CHIẾC) Điện Tử 360(E360) | 3 |
| Điện trở 270Ohm | Điện trở 270 Ohm 1/4W 5% 4 Vòng màu - gói 50 | 13 |

# Chương IV: Kết quả thực nghiệm

## 1. Kịch bản đánh giá

### 1.1 Kịch bản

- Sử dụng nguồn phát từ máy tính.

- Thử nghiệm với các hành động khác nhau, thiết lập mật khẩu cho ID từ 0 đến 9

- Thử nhập sai và đúng mật khẩu

### 1.2 Kết quả dự kiến

- Mạch hoạt động ổn định, có khả năng tái lập sau các lần nhấn

## 2. Đánh giá

- Mạch hoạt động ổn định, tuy vẫn còn nhiễu do dùng mạch breadboard. Mạch đảm bảo tốt các chức năng đề xuất ban đầu

# Chương V: Kết luận và Hướng phát triển

## 1. Kết luận

Dự án mạch khóa mở cửa dùng mật khẩu với vi điều khiển AT89S52 đã hoàn thành thành công và đạt được các mục tiêu đề ra. Hệ thống có khả năng:

* Nhận và xử lý mật khẩu người dùng một cách chính xác.
* Điều khiển cơ chế mở khóa cửa một cách ổn định và đáng tin cậy.
* Đảm bảo bảo mật thông tin người dùng thông qua cơ chế kiểm tra mật khẩu.

Ngoài ra, hệ thống được thiết kế với phần cứng tối ưu và phần mềm đơn giản nhưng hiệu quả, đáp ứng yêu cầu về chi phí thấp, dễ sử dụng và dễ tích hợp vào các ứng dụng thực tiễn. Dự án đã chứng minh được tiềm năng ứng dụng trong các hệ thống bảo mật thông minh và mang lại sự tiện lợi cho người dùng.

## 2. Hướng phát triển

Trong tương lai, dự án có thể được mở rộng và nâng cấp theo các hướng sau:

1. **Tích hợp giao tiếp không dây:** Sử dụng công nghệ Bluetooth hoặc Wi-Fi để cho phép điều khiển và thay đổi mật khẩu từ xa thông qua điện thoại hoặc máy tính.
2. **Cải thiện bảo mật:**
   * Sử dụng các thuật toán mã hóa để lưu trữ và xử lý mật khẩu an toàn hơn.
   * Bổ sung cơ chế khóa khẩn cấp khi có hành vi xâm nhập trái phép, như gửi cảnh báo qua SMS hoặc email.
3. **Thêm tính năng sinh trắc học:** Kết hợp với các công nghệ như nhận diện vân tay, nhận diện khuôn mặt hoặc quét thẻ RFID để tăng cường bảo mật.
4. **Nâng cấp giao diện:** Sử dụng màn hình LCD lớn hơn hoặc màn hình cảm ứng để hiển thị thông tin và hỗ trợ thao tác thân thiện hơn.
5. **Tối ưu hóa nguồn năng lượng:** Sử dụng các giải pháp tiết kiệm năng lượng hoặc tích hợp pin dự phòng để đảm bảo hệ thống hoạt động ngay cả khi mất điện.
6. **Ứng dụng IoT:** Kết nối hệ thống vào mạng IoT (Internet of Things) để quản lý từ xa và giám sát thông qua các nền tảng đám mây.

Những cải tiến trên không chỉ nâng cao tính năng của hệ thống mà còn mở rộng phạm vi ứng dụng, đưa giải pháp này trở nên phổ biến và thiết thực hơn trong cuộc sống hiện đại.