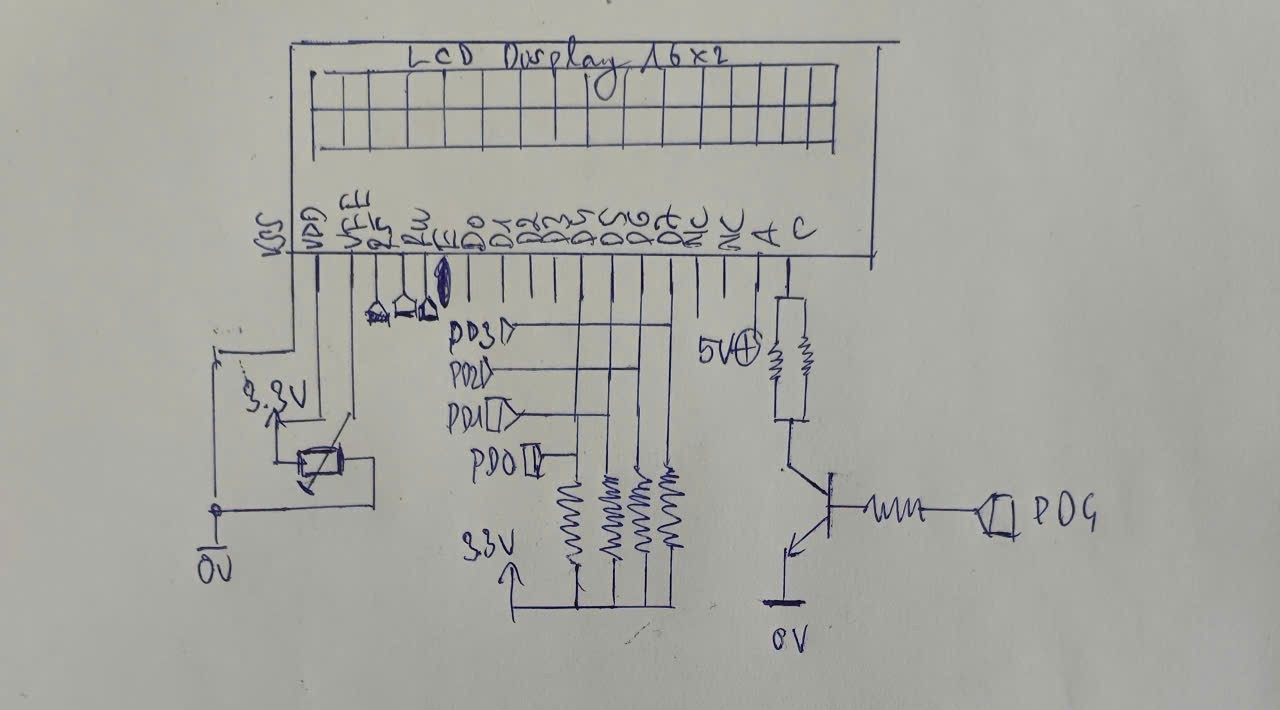
|  |  |
| --- | --- |
| TRƯỜNG ĐH SƯ PHẠM KỸ THUẬT TPHCM  **KHOA CƠ KHÍ CHẾ TẠO MÁY**  **BỘ MÔN CƠ ĐIỆN TỬ**  **Họ và tên: Võ Hồng Quân**  **Mã số SV: 22134012** | **BÁO CÁO**  **THỰC TẬP LẬP TRÌNH VI XỬ LÝ**  Ngày:  Tuần học: 04  Nội dung: **Giao tiếp LCD HD44780** |

1. **Khảo sát sơ đồ nguyên lý:**

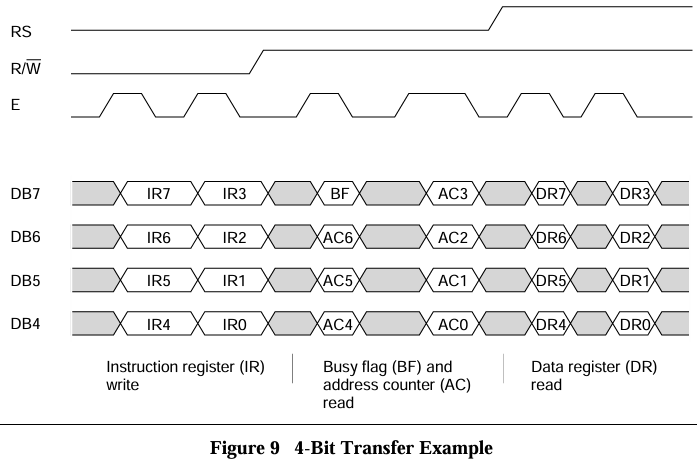
Sinh viên vẽ sơ đồ nguyên lý khối **LCD**



1. **Bảng kết nối input - output**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên phần cứng | Vị trí chân vi điều khiển | Mô tả chức năng |
| 1 | LCDRS | PD.7 | Chọn thanh ghi: 0 = Lệnh, 1 = Dữ liệu. |
| 2 | LCDRW | PD.6 | Chế độ Đọc/Ghi: 0 = Ghi, 1 = Đọc |
| 3 | LCDE | PD.5 | Tín hiệu bật/tắt: xung kích hoạt HIGH → LOW, giúp LCD nhận dữ liệu |
| 4 | LCDD4 | PD.0 | Dữ liệu Bit 4 |
| 5 | LCDD5 | PD.1 | Dữ liệu Bit 5 |
| 6 | LCDD6 | PD.2 | Dữ liệu Bit 6 |
| 7 | LCDD7 | PD.3 | Dữ liệu Bit 7 |
| 8 | LCDLED | PD.4 | Điều khiển đèn nền (Backlight) |

1. **Lập trình**
   1. **Giao thức giao tiếp 4 bit:**



Giải thích hoạt động của giao thức:

Bài làm:

1. Ghi lệnh (RS = 0, R/W = 0): Dữ liệu được ghi vào thanh ghi lệnh (IR), điều khiển LCD:

Gửi 4-bit cao (IR7 - IR4) → Kích xung E

Gửi 4-bit thấp (IR3 - IR0) → Kích xung E

1. Đọc Busy flag và Address counter (RS = 0, R/W = 1):

Busy flag (DB7) : 1 = LCD bận, 0 = LCD sẵn sàng.

Address Counter (DB6 - DB4): Địa chỉ hiện tại của con trỏ LCD từ AC6 – AC0

1. Đọc Dữ Liệu (RS = 1, R/W = 1):

Lấy 4-bit cao (IR7 - IR4) → Kích xung E

Lấy 4-bit thsap (IR3 – IR0) → Kích xung E

* 1. **Bảng mã lệnh LCD:**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Ý nghĩa lệnh** | **Mã lệnh** | |
| **Nhị phân** (D7.D6.D5.D4.D3.D2.D1.D0) | **HEX** |
| 1 | Clear display   |  | | --- | | Xóa toàn bộ màn hình và đặt địa chỉ DDRAM về 0. |  |  | | --- | |  | | 00000001 | 0x01 |
| 2 | Return home  Đưa con trỏ về địa chỉ 0 và đưa màn hình về vị trí ban đầu mà không làm thay đổi dữ liệu DDRAM. | 0000001- | 0x02-0x03 |
| 3 | Entry mode set  Xác định hướng di chuyển của con trỏ (I/D) và có dịch màn hình hay không (S). | 000001 I/D S | 0x04 - 0x07 (tùy I/D & S) |
| 4 | Display ON/OFF control  Bật/tắt hiển thị (D), bật/tắt con trỏ (C), và bật/tắt nhấp nháy con trỏ (B). | 00001 D C B | 0x08 - 0x0F (tùy D, C, B) |
| 5 | Cursor or display shift  Di chuyển con trỏ hoặc dịch màn hình theo hướng trái/phải. | 00001 S/C R/L | 0x10 - 0x1F (tùy S/C & R/L) |
| 6 | Function set  Chọn độ dài dữ liệu (4-bit hoặc 8-bit), số dòng hiển thị (1 hoặc 2 dòng), và font chữ. | 0011 DL N F -- -- | 0x20 - 0x3F (tùy DL, N, F) |
| 7 | Set CGRAM address  Chọn độ dài dữ liệu (4-bit hoặc 8-bit), số dòng hiển thị (1 hoặc 2 dòng), và font chữ. | 010 ADD ADD ADD ADD ADD ADD | 0x40 - 0x7F (tùy địa chỉ ADD) |
| 8 | Set DDRAM address  Thiết lập địa chỉ DDRAM để ghi dữ liệu hiển thị. | 100 ADD ADD ADD ADD ADD ADD ADD | 0x80 - 0xFF (tùy địa chỉ ADD) |
| 9 | Write data to CGRAM or DDRAM  Ghi dữ liệu vào DDRAM (bộ nhớ hiển thị) hoặc CGRAM (bộ nhớ tạo ký tự tùy chỉnh). | 1 0 DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 | Ghi dữ liệu cần ghi |
| 10 | Read data from CGRAM or DDRAM  Đọc dữ liệu từ DDRAM hoặc CGRAM | 1 1 DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0 | Đọc dữ liệu cần đọc |

* 1. **Tạo thư viện myLCD:**
* Viết thư viện ***myLCD.h*** và ***myLCD.c*** với các hàm và macro sau đây:
* Hàm **LCD\_PutChar**(uint8\_t **ch**) để ký tự ch lên hiển thị trên màn hình LCD ở vị trí con trỏ hiện tại.
* Hàm **LCD\_PutString**(uint8\_t \* s) để đẩy một chuỗi s hiển thị lên màn hình LCD ở vị trí con trỏ hiện tại.
* Hàm **LCD\_Init**(void) để khởi tạo LCD chế độ 4 bit.
* Hàm **LCD\_GotoXY**(uint8\_t **x**, uint8\_t **y**) để di chuyển con trỏ đến vị trí X và Y trên màn hình. Với X là vị trí cột từ 0 đến 15; Y là vị trí hàng từ 0 (hàng trên) đến 1 (hàng dưới).
* Hàm **LCD\_Command**(uint8\_t **cmd**) để đẩy 1 byte lệnh vào LCD theo bảng mã lệnh (HEX) bảng 3.2. Các Macro được định nghĩa sẵn (tự định nghĩa)

Bài làm:

* myLCD.h

|  |
| --- |
| #ifndef MYLCD\_H  #define MYLCD\_H  #include <stdint.h>  // LCD Commands  #define LCD\_CLEAR 0x01  #define LCD\_HOME 0x02  #define LCD\_ENTRY\_MODE 0x06  #define LCD\_DISPLAY\_ON 0x0C  #define LCD\_DISPLAY\_OFF 0x08  #define LCD\_CURSOR\_ON 0x0E  #define LCD\_CURSOR\_BLINK 0x0F  #define LCD\_SHIFT\_LEFT 0x18  #define LCD\_SHIFT\_RIGHT 0x1C  #define LCD\_FUNCTION\_SET 0x28  // Prototype các hàm  void LCD\_Init(void);  void LCD\_Command(uint8\_t cmd);  void LCD\_PutChar(uint8\_t ch);  void LCD\_PutString(uint8\_t \*s);  void LCD\_GotoXY(uint8\_t x, uint8\_t y);  #endif // MYLCD\_H |

* myLCD.c

|  |
| --- |
| #include "myLCD.h"  #include "main.h"  void LCD\_Enable(void) {  HAL\_GPIO\_WritePin(LCDE\_GPIO\_Port, LCDE\_Pin, GPIO\_PIN\_SET);  HAL\_Delay(1);  HAL\_GPIO\_WritePin(LCDE\_GPIO\_Port, LCDE\_Pin, GPIO\_PIN\_RESET);  HAL\_Delay(1);  }  void LCD\_Send4Bit(uint8\_t data) {  HAL\_GPIO\_WritePin(LCDD4\_GPIO\_Port, LCDD4\_Pin, (data >> 0) & 0x01);  HAL\_GPIO\_WritePin(LCDD5\_GPIO\_Port, LCDD5\_Pin, (data >> 1) & 0x01);  HAL\_GPIO\_WritePin(LCDD6\_GPIO\_Port, LCDD6\_Pin, (data >> 2) & 0x01);  HAL\_GPIO\_WritePin(LCDD7\_GPIO\_Port, LCDD7\_Pin, (data >> 3) & 0x01);  LCD\_Enable();  }  void LCD\_Command(uint8\_t cmd) {  HAL\_GPIO\_WritePin(LCDRS\_GPIO\_Port, LCDRS\_Pin, GPIO\_PIN\_RESET);  LCD\_Send4Bit(cmd >> 4); // Gửi 4-bit cao  LCD\_Send4Bit(cmd & 0x0F); // Gửi 4-bit thấp  }  // Hàm gửi ký tự lên LCD  void LCD\_PutChar(uint8\_t ch) {  HAL\_GPIO\_WritePin(LCDRS\_GPIO\_Port, LCDRS\_Pin, GPIO\_PIN\_SET);  LCD\_Send4Bit(ch >> 4);  LCD\_Send4Bit(ch & 0x0F);  }  void LCD\_PutString(uint8\_t \*s) {  while (\*s) {  LCD\_PutChar(\*s++);  }  }  void LCD\_GotoXY(uint8\_t x, uint8\_t y) {  uint8\_t address = (y == 0) ? (0x80 + x) : (0xC0 + x);  LCD\_Command(address);  }  void LCD\_Init(void) {  HAL\_Delay(40);  LCD\_Send4Bit(0x03);  HAL\_Delay(5);  LCD\_Send4Bit(0x03);  HAL\_Delay(1);  LCD\_Send4Bit(0x03);  HAL\_Delay(1);  LCD\_Send4Bit(0x02);  HAL\_Delay(1);  LCD\_Command(LCD\_FUNCTION\_SET);  LCD\_Command(LCD\_DISPLAY\_ON);  LCD\_Command(LCD\_ENTRY\_MODE);  LCD\_Command(LCD\_CLEAR);  HAL\_Delay(2);  } |

* 1. **Lập trình ứng dụng 1**
* Khi mới mở điện LCD hiển thị dòng trên là tên sinh viên (không dấu); dòng dưới là mã số sinh viên.

Bài làm:

|  |
| --- |
| int main(void)  {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();  LCD\_Init();  while (1)  {  LCD\_GotoXY(0, 0);  LCD\_PutString((uint8\_t \*)"Vo Hong Quan");  LCD\_GotoXY(0, 1);  LCD\_PutString((uint8\_t \*)"22134012");  }  } |

* 1. **Lập trình ứng dụng 2**

Sử dụng thư viện LCD đã viết, viết chương trình đồng hồ số theo yêu cầu sau:

* Khi mới mở điện, LCD hiển thị thời gian dạng: "CLOCK: HH:MM"SS" . Ví dụ "CLOCK: 14:03:05" (lúc 14 giờ 03 phút và 05 giây).
* Đồng hồ đếm lên liên tục sau mỗi 1 giây. Quy luật đếm giống như các đồng hồ số (khi giây tăng đến 59 thì phút tăng 1 đơn vị v.v...)

Bài làm:

|  |
| --- |
| uint8\_t hours = 14;  uint8\_t minutes = 3;  uint8\_t seconds = 5;  void DisplayTime(void) {  char timeString[16];  sprintf(timeString, "CLOCK: %02d:%02d:%02d", hours, minutes, seconds);  LCD\_GotoXY(0, 0);  LCD\_PutString((uint8\_t \*)timeString);  }  void UpdateTime(void) {  HAL\_Delay(1000);  seconds++;  if (seconds >= 60) {  seconds = 0;  minutes++;  if (minutes >= 60) {  minutes = 0;  hours++;  if (hours >= 24) {  hours = 0;  }  }  }  }  int main(void) {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();  LCD\_Init();  while (1) {  DisplayTime();  UpdateTime();  }  } |

* 1. **Lập trình ứng dụng 3**

Sử dụng thư viện LCD đã viết, viết chương trình theo yêu cầu sau:

* Khi mới mở điện, LCD hiển thị số lượng sản phẩm: "San pham: 0" ở dòng trên.
* Mỗi khi nhấn nút P1, LCD hiển thị tăng sản phẩm lên 1 đơn vị. (Ví dụ: "San pham: 1" ; "San pham: 2" ; v.v...)
* Mỗi khi nhấn nút P2, LCD hiển thị giảm sản phẩm xuống 1 đơn vị.
* Mỗi khi nhấn nút P3 (chỉ cần nhấn 1 lần rồi buông tay) , LCD hiển thị quá trình đếm lên tự động, mỗi giây tự tăng sản phẩm lên 1 đơn vị.
* Mỗi khi nhấn nút P4, nếu LCD đang đếm lên tự động thì ngừng đếm.

Bài làm:

|  |
| --- |
| uint8\_t product\_count = 0;  uint8\_t auto\_increment = 0;  uint32\_t last\_update\_time = 0;  void LCD\_UpdateProductCount() {  char lcd\_buffer[16];  sprintf(lcd\_buffer, "San pham: %02d", product\_count);  LCD\_GotoXY(0, 0);  LCD\_PutString((uint8\_t \*)lcd\_buffer);  }  int main(void) {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();  LCD\_Init();  LCD\_UpdateProductCount();  while (1) {  uint8\_t button\_P1 = HAL\_GPIO\_ReadPin(P1\_GPIO\_Port, P1\_Pin);  uint8\_t button\_P2 = HAL\_GPIO\_ReadPin(P2\_GPIO\_Port, P2\_Pin);  uint8\_t button\_P3 = HAL\_GPIO\_ReadPin(P3\_GPIO\_Port, P3\_Pin);  uint8\_t button\_P4 = HAL\_GPIO\_ReadPin(P4\_GPIO\_Port, P4\_Pin);  if (button\_P1 == GPIO\_PIN\_RESET) {  product\_count++;  LCD\_UpdateProductCount();  HAL\_Delay(200);  }    if (button\_P2 == GPIO\_PIN\_RESET) {  if (product\_count > 0) product\_count--;  LCD\_UpdateProductCount();  HAL\_Delay(200);  }  if (button\_P3 == GPIO\_PIN\_RESET) {  auto\_increment = 1;  HAL\_Delay(200);  }    if (button\_P4 == GPIO\_PIN\_RESET) {  auto\_increment = 0;  HAL\_Delay(200);  }  if (auto\_increment && (HAL\_GetTick() - last\_update\_time >= 1000)) {  last\_update\_time = HAL\_GetTick();  product\_count++;  LCD\_UpdateProductCount();  }  }  } |

* 1. **Lập trình ứng dụng 4**

Sử dụng thư viện LCD đã viết, viết chương trình theo yêu cầu sau:

- Khi mới mở điện, LCD hiển thị thời gian dạng: "HH:MM"SS" . Ví dụ "14:03:05" (lúc 14 giờ 03 phút và 05 giây).

- Mỗi khi nhấn nút P1 bật/tắt chế độ đếm lên của đồng hồ:

* LCD hiển thị đồng hồ đếm thời gian tăng dần sau mỗi 1 giây.
* Khi giây tăng đến 59 thì phút tăng lên 1 đơn vị;
* Khi phút tăng lên 59 thì giờ tăng lên 1 đơn vị.

- Mỗi khi nhấn nút P2, hiển thị giờ hiện tại tăng 1 đơn vị.

- Mỗi khi nhấn nút P3 hiển thị phút hiện tại tăng 1 đơn vị.

- Mỗi khi nhấn nút P4, hiển thị giây hiện tại tăng 1 đơn vị.

Bài làm:

|  |
| --- |
| uint8\_t hours = 14;  uint8\_t minutes = 3;  uint8\_t seconds = 5;  uint8\_t auto\_increment = GPIO\_PIN\_RESET;  uint32\_t last\_update\_time = 0;  void DisplayTime() {  char timeString[16];  sprintf(timeString, "CLOCK: %02d:%02d:%02d", hours, minutes, seconds);  LCD\_GotoXY(0, 0);  LCD\_PutString((uint8\_t \*)timeString);  }  void UpdateTime(void) {  if (HAL\_GetTick() - last\_update\_time >= 1000) {  last\_update\_time = HAL\_GetTick();  }    seconds++;  if (seconds >= 60) {  seconds = 0;  minutes++;  if (minutes >= 60) {  minutes = 0;  hours++;  if (hours >= 24) {  hours = 0;  }  }  }  DisplayTime();  }  int main(void) {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();  LCD\_Init();  while (1) {  uint8\_t button\_P1 = HAL\_GPIO\_ReadPin(P1\_GPIO\_Port, P1\_Pin);  uint8\_t button\_P2 = HAL\_GPIO\_ReadPin(P2\_GPIO\_Port, P2\_Pin);  uint8\_t button\_P3 = HAL\_GPIO\_ReadPin(P3\_GPIO\_Port, P3\_Pin);  uint8\_t button\_P4 = HAL\_GPIO\_ReadPin(P4\_GPIO\_Port, P4\_Pin);    DisplayTime();    if (button\_P1 == GPIO\_PIN\_RESET) {  if (auto\_increment == 0) {  auto\_increment = 1;  }  else auto\_increment = 0;  }  if (auto\_increment) {  UpdateTime();  }  if (button\_P2 == GPIO\_PIN\_RESET) {  hours++;  HAL\_Delay(200);  }  if (button\_P3 == GPIO\_PIN\_RESET) {  minutes++;  HAL\_Delay(200);  }  if (button\_P4 == GPIO\_PIN\_RESET) {  seconds++;  HAL\_Delay(200);  }  }  } |