



Đại Học Công Nghiệp Thành Phố Hồ Chí Minh

Hướng dẫn Lập Trình LCD

(Liquid Crystal display)

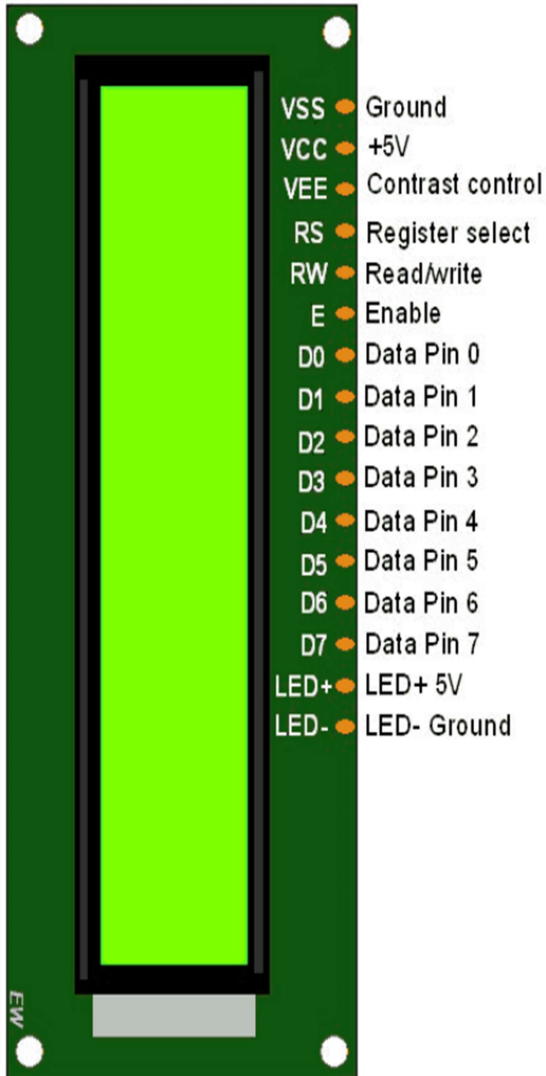
Bộ môn: Điện Tử Máy Tính
GV: Vũ Thị Hồng Nga
Mail: hongngavuh@yahoo.com

1. Giới thiệu LCD - HD44780



- LCD 16x2 có thể hiển thị tất cả các ký tự trong bảng mã ASCII và tám ký tự tự do do người dùng tạo ra
- LCD 16x2 có khả năng hiển thị hai dòng, với mỗi dòng là 16 ký tự, mỗi ký tự có độ phân giải 8x5 pixel trên nền sáng phát ra từ led

2. Sơ đồ chân



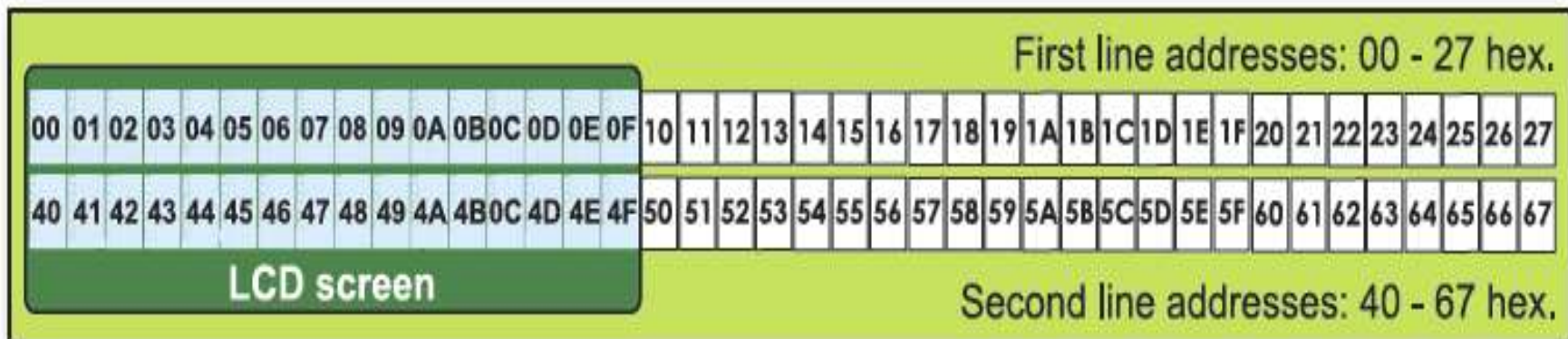
Thứ tự chân	Tên chân	Chức năng
1	VSS	GND của LCD
2	VCC	Nguồn cấp cho LCD
3	VEE	Điều chỉnh độ tương phản (cần được gắn với biến trở)
4	RS	Chọn thanh ghi RS=0: Đưa LCD vào chế độ ghi lệnh RS=1: Đưa LCD vào chế độ ghi dữ liệu (dữ liệu xuất lên màn hình)
5	RW	Chọn chế độ đọc/ghi LCD RW=0: Vi điều khiển truyền dữ liệu vào LCD RW=1: Vi điều khiển đọc dữ liệu từ LCD
6	E	E=0: Vô hiệu hóa đọc/ghi E=1: Cho phép LCD đọc/ghi E chuyển từ mức 1 về 0: bắt đầu đọc/ghi LCD
7	D0	Dữ liệu bit thứ 0
8	D1	Dữ liệu bit thứ 1
9	D2	Dữ liệu bit thứ 2
10	D3	Dữ liệu bit thứ 3
11	D4	Dữ liệu bit thứ 4
12	D5	Dữ liệu bit thứ 5
13	D6	Dữ liệu bit thứ 6
14	D7	Dữ liệu bit thứ 7
15	LED+	Nguồn dương cấp cho LED nền
16	LED-	Nguồn âm cấp cho LED nền

3. Bộ nhớ trong của LCD

a. DDRAM

DDRAM là bộ nhớ tạm chứa các ký tự cần hiển thị lên LCD, bộ nhớ này gồm có 80 ô được chia thành hai hàng, mỗi ô có độ rộng 8 bit và được đánh số từ 0 đến 39 cho dòng 1, từ 64 đến 103 cho dòng 2. Mỗi ô nhớ tương ứng với một ô trên màn hình LCD

DDRAM memory



b. CGROM

CGROM là vùng nhớ cố định chứa định nghĩa font cho các ký tự , mỗi ký tự trong vùng nhớ CGROM chính là mã ASCII của ký tự đó

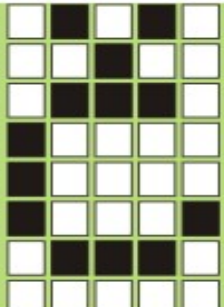
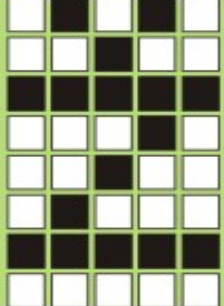
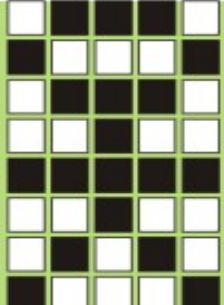
Lower 4 Bits	Upper 4 Bits	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
		CG RAM (1)															
xxxx0000				0	a	P	`	P					-	タ	ミ	α	p
xxxx0001	(2)			!	1	A	Q	a	q			。	ア	チ	△	ä	q
xxxx0010	(3)			"	2	B	R	b	r			「	イ	ツ	×	β	θ
xxxx0011	(4)			#	3	C	S	c	s			」	ウ	テ	モ	ε	∞
xxxx0100	(5)			\$	4	D	T	d	t			、	エ	ト	ヤ	μ	Ω
xxxx0101	(6)			%	5	E	U	e	u			・	オ	ナ	1	℃	Ü
xxxx0110	(7)			&	6	F	V	f	v			ヲ	カ	ニ	ヨ	ρ	Σ
xxxx0111	(8)			'	7	G	W	g	w			ア	キ	ヌ	ラ	g	π
xxxx1000	(1)			(8	H	X	h	x			イ	ク	ネ	リ	ℓ	×
xxxx1001	(2))	9	I	Y	i	y			ウ	ケ	ル	ル	'	γ
xxxx1010	(3)			*	:	J	Z	j	z			エ	コ	ハ	レ	j	〒
xxxx1011	(4)			+	;	K	L	k	l			オ	サ	ヒ	ロ	*	⌘
xxxx1100	(5)			,	<	L	¥	l	l			ハ	シ	フ	ワ	⊕	⌘
xxxx1101	(6)			-	=	M	J	m	}			ユ	ズ	ハ	ン	も	÷
xxxx1110	(7)			.	>	N	^	n	÷			ヨ	セ	ホ	°	ℓ	
xxxx1111	(8)			/	?	O	_	o	+			ッ	ソ	マ	°	ö	

Vùng nhớ CGROM.

c. CGRAM

CGRAM là vùng nhớ chứa các ký tự do người dùng tự định nghĩa, mỗi ký tự có kích thước 5x8 và được dành cho 8 ô nhớ 8 bit. Các ký tự thường được định nghĩa trước và được gọi hiển thị khi cần thiết. Vùng này có tất cả 64 ô nhớ nên có tối đa 8 ký tự được định nghĩa

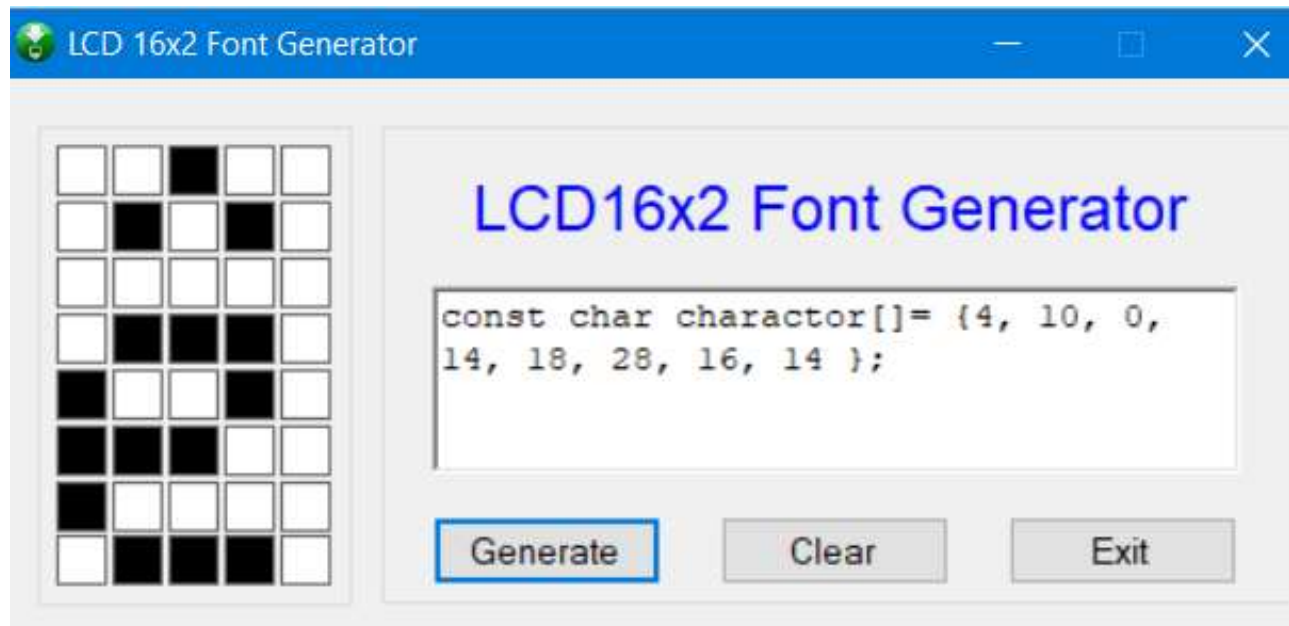
Register
Address hex.

	CGRAM Memory Registers								LCD Display	
00				0	1	0	1	0	 <p>First symbol in CGRAM memory (latin letter <i>c</i> in lowercase)</p> <p>Symbol Address: 0000 0000</p>	
01				0	0	1	0	0		
02				0	1	1	1	0		
03				1	0	0	0	0		
04				1	0	0	0	0		
05				1	0	0	0	1		
06				0	1	1	1	0		
07				0	0	0	0	0		
08				0	1	0	1	0	 <p>Second symbol in CGRAM memory (latin letter <i>z</i> in lowercase)</p> <p>Symbol Address: 0000 0001</p>	
09				0	0	1	0	0		
0A				1	1	1	1	1		
0B				0	0	0	1	0		
0C				0	0	1	0	0		
0D				0	1	0	0	0		
0E				1	1	1	1	1		
0F				0	0	0	0	0		
10										
11										
12										
14										
38				0	1	1	1	0	 <p>Eight symbol in CGRAM memory (figure)</p> <p>Symbol Address: 0000 0111</p>	
39				1	0	0	0	1		
3A				0	1	1	1	0		
3B				0	0	1	0	0		
3C				1	1	1	1	1		
3D				0	0	1	0	0		
3E				0	1	0	1	0		
3F				1	0	0	0	1		

4. Cách tạo ký tự tự do(do người dùng định nghĩa):
vẽ ký tự cần tạo trên ma trận 5x8, các vị trí cần hiển thị sẽ
điền số 1, không hiển thị điền số 0



Ví dụ: tạo chữ ê



- Mã tạo ra theo số thập phân:
`const char charactor[]= {4, 10, 0, 14, 18, 28, 16, 14 };`
- Mã tạo ra theo số thập lục phân:
`const char charactor[]= {0X04, 0X0A, 0X00, 0X0E, 0X12, 0X1C, 0X10, 0X0E };`

5. Địa chỉ ký tự tự do:

8 ký tự tự do sẽ được lưu vào địa chỉ từ 0x00 → 0x07, theo thứ tự tạo ra của các ký tự

Ví dụ:

```
const char cgram_dat[] = { 0X0E, 0X09, 0X09, 0X1D, 0X09, 0X09, 0X0E, 0X00,  
                           0X04, 0X0A, 0X00, 0X0E, 0X12, 0X1C, 0X10, 0X0E,  
                           0x99 };
```

Mã chữ đ
địa chỉ 0x00

Mã chữ ê
địa chỉ 0x01

6. Cách đọc địa chỉ ký tự trong bảng mã ASCII

Mỗi kí tự trong bảng mã ASCII được lưu trong địa chỉ 8 bit, đọc địa chỉ 4 bit đầu theo theo cột, đọc địa chỉ 4 bit sau theo hàng tương ứng với vị trí của ký tự đó

- Khi lập trình ta có thể nhập trực tiếp ký tự từ bàn phím hoặc lấy theo địa chỉ trong bảng mã ASCII

Vd: đọc địa chỉ của ký tự 0

Ký tự 0 có địa chỉ là : 0x30

4 bit đầu

4 bit sau

Lower 4 Bits \ Upper 4 Bits	0000	0001	0010	0011	0100
xxxx0000	CG RAM (1)			0	a
xxxx0001	(2)		!	1	A

Cách đọc vị trí các kí tự trên LCD 16x2

LCD 16x2

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0			A	B	C	D										
1			E	F	G	H										

Notes: A blue speech bubble labeled "Cột" points to the column headers. Another blue speech bubble labeled "hàng" points to the row headers.

Ví dụ : đọc vị trí của chữ A, F
chữ A có vị trí là : hàng 0, cột 2
chữ F có vị trí là : hàng 1, cột 3

Các hàm được hỗ trợ trong thư viện lcd.h

```
void lcd_init(void); // Khởi động LCD.
void lcd_gotoxy(unsigned char col, unsigned char row); // Định tọa độ trên LCD.
void lcd_putc(char c); // Ghi một ký tự lên LCD.
void lcd_puts(const char* s); // Ghi một chuỗi lên LCD.
void lcd_ShiftLeft(void); // Dịch chuyển nội dung sang trái.
void lcd_ShiftRight(void); // Dịch chuyển nội dung sang phải.
void lcd_MoveRight(unsigned char p); // Di chuyển sang phải p vị trí.
void lcd_MoveLeft(unsigned char p); // Di chuyển sang trái p vị trí.
unsigned char lcd_busy(void); // Thông báo LCD bạn xử lý thông tin.
// Đọc một byte dữ liệu từ LCD.
unsigned char lcd_get_byte(unsigned char rs);
// Ghi một byte dữ liệu/lệnh lên LCD.
void lcd_put_byte(unsigned char a, unsigned char b);
// Hiện thị từng ký tự trong chuỗi theo thời gian.
void lcd_String_Delay(unsigned char*s, unsigned int dly);
```

```

void lcd_init()
{
// Khai bao bien.
unsigned char i;

// Dinh nghia ham.
    LCD_EN_TRIS = 0;           // Cau hinh cac chan ket noi LCD.
    LCD_RS_TRIS = 0;
    LCD_RW_TRIS = 0;
    LCD_DATA4_TRIS = 0;
    LCD_DATA5_TRIS = 0;
    LCD_DATA6_TRIS = 0;
    LCD_DATA7_TRIS = 0;

    __delay_ms(15);           // Tao tre cho cap nguon LCD on dinh.

    LCD_RS = 0;
    LCD_RW = 0;
    LCD_EN = 0;

    lcd_put_nibble(3);        // Reset LCD
    __delay_ms(10);
    lcd_put_nibble(3);
    __delay_ms(10);
    lcd_put_nibble(3);
    __delay_ms(10);

    lcd_put_nibble(2);
    while(lcd_busy());

    lcd_put_byte(0, FOUR_BIT & LINES_5X7);        // Loai LCD.
    while(lcd_busy());

    lcd_put_byte(0, DON & CURSOR_OFF & BLINK_OFF);    // Bat
hien thi, tat con tro, tat chop tat con tro.
    while(lcd_busy());

    lcd_put_byte(0, 0x01);        // Xoa man
hinh va dua con tro ve dau dong.
    while(lcd_busy());

    lcd_put_byte(0, SHIFT_DISP_LEFT);        // Che do dich
chuyen con tro.
    while(lcd_busy());
}

```



```
void lcd_gotoxy(unsigned char col, unsigned char row)
{
    // Khai bao bien.
    unsigned char address;

    // Dinh nghia ham.
    if(row!=0)        // Xac dinh ma qui dinh dia chi dong.
        address=0x40;    // Dong duoi.
    else
        address=0;      // Dong tren.

    address += col;

    lcd_put_byte(0,0x80|address);    // Gui lenh dieu khien sang
LCD.
    while(lcd_busy());
}
```

```
void lcd_putc(char c)
{
    // Khai bao bien.

    // Dinh nghia ham.
    switch(c)
    {
        case '\f':                // Xoa man hinh.
            lcd_put_byte(0,1);
            while(lcd_busy());
            break;
        case '\n':                // Xuong dong va dua con
tro ve dau dong.
            lcd_gotoxy(0,1);
            break;
        case '\b':
            lcd_put_byte(0,0x10);
            break;
        default:                  // Hien thi thong tin len
LCD.
            if(isprint(c))
            {
                lcd_put_byte(1,c);
                while(lcd_busy());
            }
            break;
    }
}
```

```
void lcd_puts(const char* s)
{
    // Khai bao bien.

    // Dinh nghia ham.
    while(*s)
    {
        lcd_putc(*s++);
    }
}
```

```
}

void lcd_String_Delay(unsigned char*s, unsigned int dly)
{
    // Khai bao bien.

    // Dinh nghia ham.
    while(*s!=0)
    {
        lcd_putc(*s++);
        //      __delay_ms(dly);
    }
}
```

```
unsigned char lcd_busy()
{
    // Khai bao bien.
    unsigned char busy;

    // Dinh nghia ham.
    LCD_DATA4_TRIS = 1;           // Cau hinh cac chan ket noi
LCD.
    LCD_DATA5_TRIS = 1;
    LCD_DATA6_TRIS = 1;
    LCD_DATA7_TRIS = 1;

    LCD_RW = 1;
    LCD_RS = 0;
    __delay_us(20);
    LCD_EN = 1;
    __delay_us(20);

    busy = LCD_DATA7;           // Doc bit bao ban.

    LCD_EN = 0;
    __delay_us(20);
    LCD_EN = 1;
    __delay_us(20);
    LCD_EN = 0;

    return busy;           // Tra ve ket qua.
}
```

```
void lcd_put_nibble(unsigned char b)
{
    // Khai bao bien.
    BYTE_VAL temp;

    // Dinh nghia ham.
    LCD_DATA4_TRIS = 0;           // Cau hinh cac chan ket noi
LCD.
    LCD_DATA5_TRIS = 0;
    LCD_DATA6_TRIS = 0;
    LCD_DATA7_TRIS = 0;

    temp.Val = b;

    LCD_DATA4 = temp.bits.b0;    // Gui 4 bit thap.
    LCD_DATA5 = temp.bits.b1;
    LCD_DATA6 = temp.bits.b2;
    LCD_DATA7 = temp.bits.b3;

    __delay_us(20);
    LCD_EN = 1;
    __delay_us(20);
    LCD_EN = 0;
}
```

```
unsigned char lcd_get_byte(unsigned char rs)
{
    // Khai bao bien.
    BYTE_VAL b;

    // Dinh nghia ham.
    LCD_DATA4_TRIS = 1;           // Cau hinh cac chan ket noi
LCD:
    LCD_DATA5_TRIS = 1;
    LCD_DATA6_TRIS = 1;
    LCD_DATA7_TRIS = 1;

    LCD_RW = 1;

    if(rs)
        LCD_RS = 1;
    else
        LCD_RS = 0;
    __delay_us(20);

    LCD_EN = 1;
    __delay_us(20);

    b.bits.b7 = LCD_DATA7;        // Doc 4 bit cao.
    b.bits.b6 = LCD_DATA6;
    b.bits.b5 = LCD_DATA5;
    b.bits.b4 = LCD_DATA4;

    LCD_EN = 0;
    __delay_us(20);

    LCD_EN = 1;
    __delay_us(20);

    b.bits.b3 = LCD_DATA7;        // Doc 4 bit thap.
    b.bits.b2 = LCD_DATA6;
    b.bits.b1 = LCD_DATA5;
    b.bits.b0 = LCD_DATA4;

    LCD_EN = 0;
    __delay_us(20);

    return b.Val;    // Tra ve ket qua doc duoc.
}
```



```
void lcd_put_byte(unsigned char rs, unsigned char b)
{
    // Khai bao bien.

    // Dinh nghia ham.
    LCD_DATA4_TRIS = 0;           // Cau hinh cac chan ket noi
LCD.
    LCD_DATA5_TRIS = 0;
    LCD_DATA6_TRIS = 0;
    LCD_DATA7_TRIS = 0;

    if(rs)
        LCD_RS = 1;
    else
        LCD_RS = 0;
    __delay_us(20);

    LCD_RW = 0;
    __delay_us(20);

    LCD_EN = 0;

    lcd_put_nibble(b >> 4);      // Gui 4 bit cao.

    lcd_put_nibble(b & 0xf);     // Gui 4 bit thap.
}
```

```
void lcd_ShiftLeft(void)
{
// Khai bao bien.

// Dinh nghia ham.
    lcd_put_byte(0,0x18);
}
```

```
void lcd_MoveRight(unsigned char p)
{
// Khai bao bien.
unsigned char i;

// Dinh nghia ham.
    for(i=0;i<p;i++)
    {
        lcd_ShiftRight();
        __delay_ms(100);
    }
}
```

```
void lcd_ShiftRight(void)
{
// Khai bao bien.

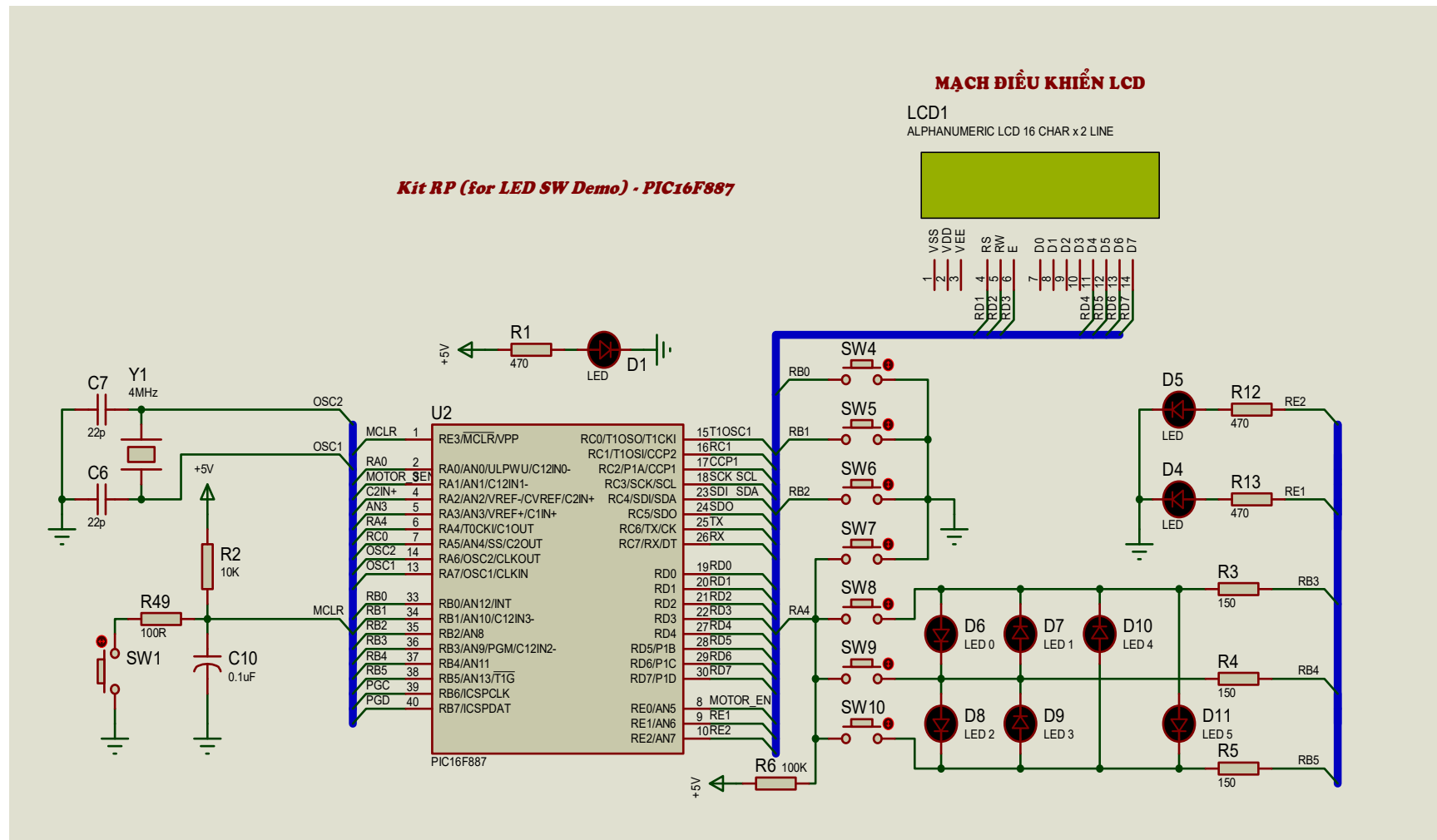
// Dinh nghia ham.
    lcd_put_byte(0,0x1C);
}
```

```
void lcd_MoveLeft(unsigned char p)
{
// Khai bao bien.
unsigned char i;

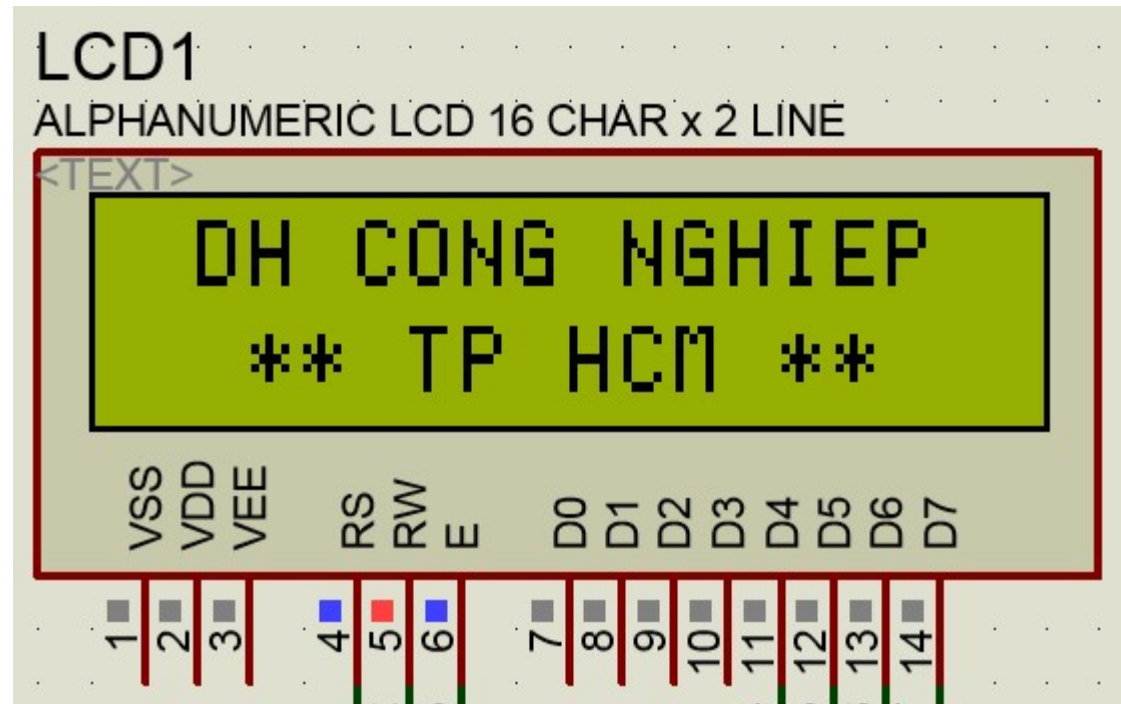
// Dinh nghia ham.
    for(i=0;i<p;i++)
    {
        lcd_ShiftLeft();
        __delay_ms(100);
    }
}
```

Bài Giảng Vi xử lý

Lập trình LCD 16x2 trên kit RP



Ví dụ 1: hiển thị chữ không dấu như màn hình sau



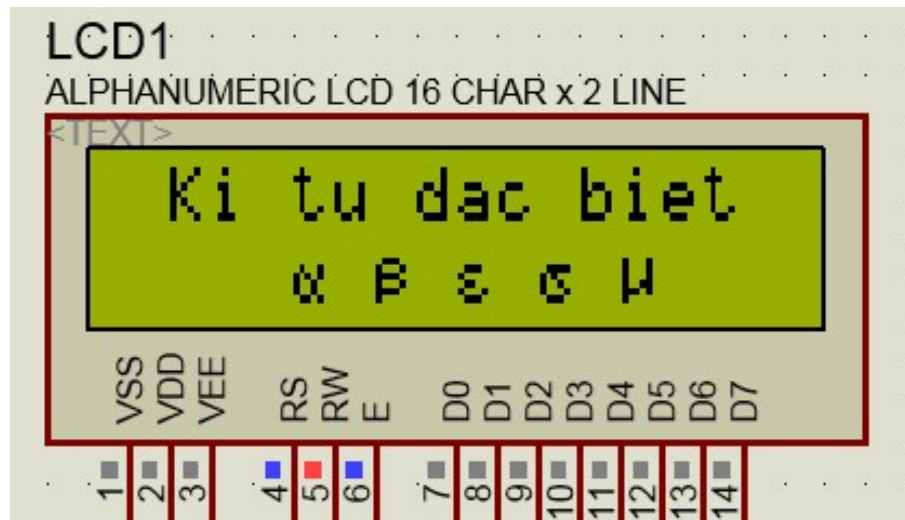
```
#include <htc.h>
#include <math.h>
#include <stdlib.h>
#include <stdio.h>
#include "lcd.h"    // khai báo thư viện LCD

__CONFIG (FOSC_HS & WDTE_OFF & PWRTE_OFF & MCLRE_ON & CP_OFF & CPD_OFF &
BOREN_OFF & IESO_OFF & FCMEN_OFF & LVP_OFF & DEBUG_ON);
//__CONFIG(FOSC_HS & WDTE_OFF & PWRTE_ON & CP_OFF & BOREN_OFF & LVP_OFF);

#define _XTAL_FREQ 4000000

void main()
{
    lcd_init();
    lcd_puts("\f DH CONG NGHIEP\n ** TP HCM ** ");
    while(1);
}
```

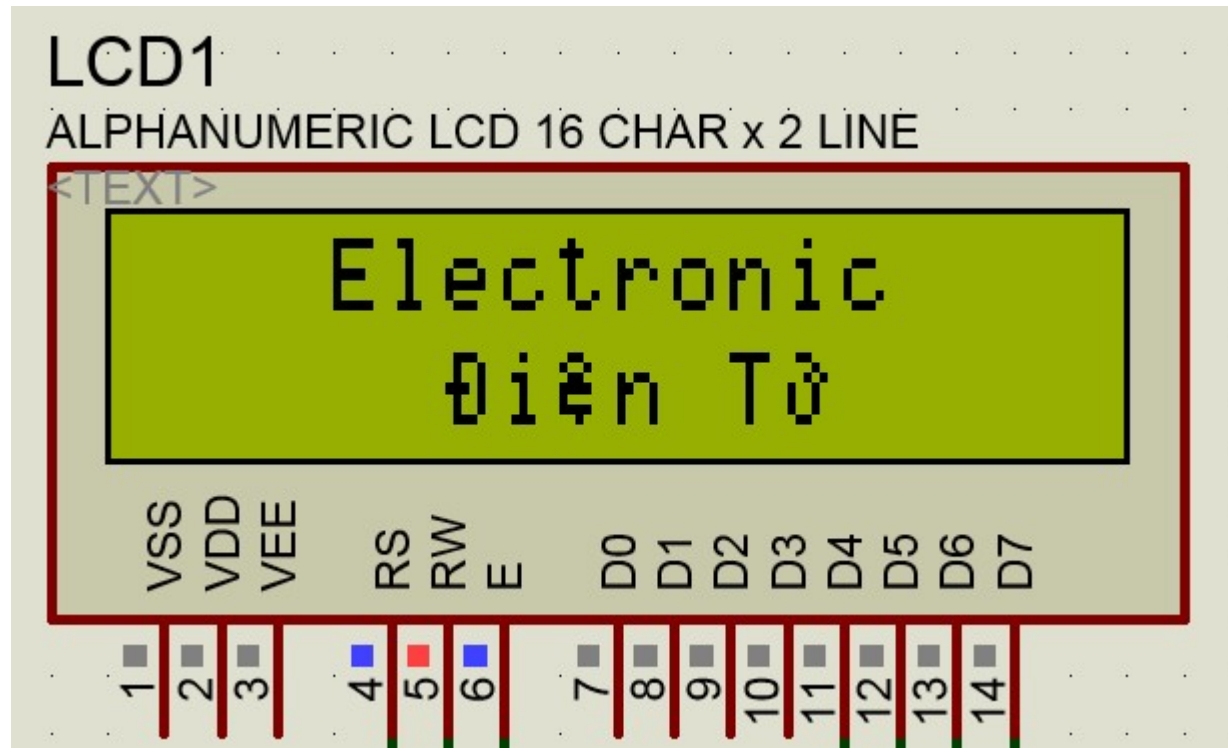
Ví dụ 2: hiển thị kí tự đặc biệt như màn hình sau



Ví dụ 2: hiển thị kí tự đặc biệt

```
void main()
{
    unsigned int i;
    lcd_init();
    lcd_put_byte(0,0x01);           // Xoa man hinh LCD.
    while(lcd_busy());             // Kiem tra LCD bao ban.
    lcd_gotoxy(0,0);               // dua con tro den hang 0,cot 0
    lcd_puts(" Ki tu dac biet "); // hien thi chuoi
    lcd_gotoxy(4,1);              // dua con tro den hang 1,cot 4
    lcd_putc(0b11100000);         // hien thi ki tu
    lcd_gotoxy(6,1);
    lcd_putc(0b11100010);
    lcd_gotoxy(8,1);
    lcd_putc(0b11100011);
    lcd_gotoxy(10,1);
    lcd_putc(0b11100101);
    lcd_gotoxy(12,1);
    lcd_putc(0b11100100);
    while (1);
}
```

Ví dụ 3: hiển thị chữ có dấu như màn hình sau



```

const char dat_line1[] = "  Electronic  ";           // Noi dung hang 1.
const char dat_line2[] = {0x00,0X69,0X01,0X6E,' ',0X54,0X02}; // Noi dung hang 2.
const char cgram_dat[] = {0X0E,0X09,0X09,0X1D,0X09,0X09,0X0E,0X00, // Đ - địa chỉ là 00h
                          0x0C,0x12,0x0C,0x1E,0X10,0x1E,0x04,0x00, // ệ - địa chỉ là 01h
                          0X0C,0X02,0X05,0X12,0X12,0X12,0X0C,0X00, // ử - địa chỉ là 02h
                          0x99 };                      // Ma ket thuc chuoai la 99H

void main()
{
  unsigned char i;
  lcd_init();
  i = 0;
  lcd_put_byte(0,0x40);
  while(lcd_busy());
  while(cgram_dat[i]!=0x99) // Kiem tra đã nạp xong dữ liệu cho các ký tự tự do hay chưa
  {
    lcd_put_byte(1,cgram_dat[i]);
    while(lcd_busy());
    i++;
  }
  lcd_gotoxy(0,0);
  lcd_putc('\f');
  for(i=0;i<=16;i++)
    lcd_putc(dat_line1[i]);
  lcd_gotoxy(5,1);
  for(i=0;i<=6;i++)
    lcd_putc(dat_line2[i]);
  while(1);
}

```



**Thank you for
listening**

