BÀI 9. THIẾT KẾ ỨNG DỤNG SỬ DỤNG PIC18F4520

Mục tiêu bài học:

Sinh viên nhớ và hiểu:

- Quy trình thiết kế một ứng dụng sử dụng PIC18F4520;
- Nguyên tắc hiển thị trên LED 7 đoạn bằng phương pháp quét.

Sinh viên vận dụng các kiến thức đã học để thiết kế:

- Mạch hiển thị số nguyên trên 04 LED 7 đoạn;
- Bộ đếm sản phẩm hiển thị trên 04 LED 7 đoạn sử dụng ngắt ngoài và ngắt timer, số sản phẩm hiển thị trên 04 LED 7 đoạn. *

**: Học trực tiếp (trên lớp)*

9.1. Quy trình thiết kế một ứng dụng sử dụng PIC18F4520.

Bước 1: Phân tích bài toán và tính chọn thiết bị.

Phân tích bài toán nhằm xác định:

+ Các yêu cầu kỹ thuật đặt ra cho bài toán (các chỉ tiêu kỹ thuật);
+ Các tài nguyên của vi điều khiển cần sử dụng (số đầu vào/ra số; số đầu vào/ra tương tự; số lượng bộ đếm/định thời; ước lượng dung lượng bộ nhớ chương trình, bộ nhớ dữ liệu cần thiết …);

+ Các linh kiện, thiết bị ngoại vi cần phối hợp với vi điều khiển để giải quyết yêu cầu của bài toán.

Bước 2: Thiết kế sơ đồ nguyên lý.

Sơ đồ nguyên lý nên thiết kế trên phần mềm Proteus để mô phỏng. Sau đó có thể vẽ lại trên các phần mềm thiết kế mạch điện tử chuyên dụng (ORCAD, Altium…)

Bước 3: Lập trình và mô phỏng

Căn cứ vào yêu cầu của bài toán tiến hành lập lưu đồ thuật toán và viết chương trình điều khiển. Sau khi biên dịch có thể dùng file *.hex chứa mã chương trình đưa vào vi điều khiển trong bản thiết kế trên phần mềm hỗ trợ thiết kế ứng dụng có linh kiện lập trình (Proteus).

Trong bước này nếu không đạt do lỗi thiết kê nguyên lý, có thể quay lại bước 2.

Bước 4: Thiết kế, làm mạch in và lắp ráp linh kiện trên mạch in

Một số lưu ý khi thiết kế mạch in:

- Đường mạch nguồn cần có động rộng đủ lớn;
- Sắp xếp các linh kiện, đi dây tín hiệu để chống nhiễu;
- Cách tiết kiệm không gian sao cho mạch in nhỏ nhất có thể…

Bước 5: Nạp chương trình vào vi điều khiển

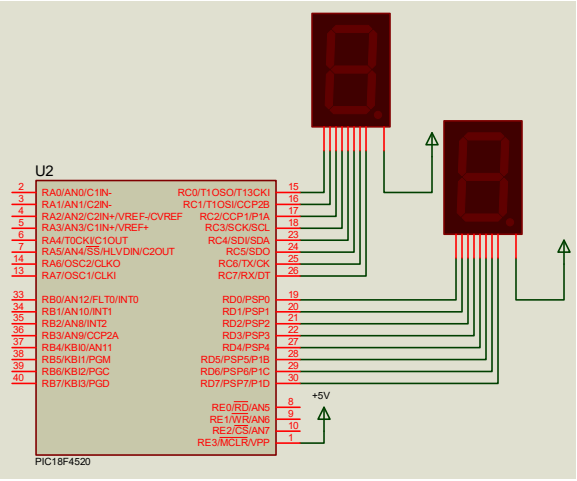
Để nạp chương trình cho PIC18F4520 cần sử dụng một số bộ nạp thông dụng như PICkit2, PICkit3, IDC3 của Microchip hoặc Burn-E của Pduytech. PICkit3 và IDC3 ngoài chức năng nạp chương trình còn có chức năng gỡ rối (debug).

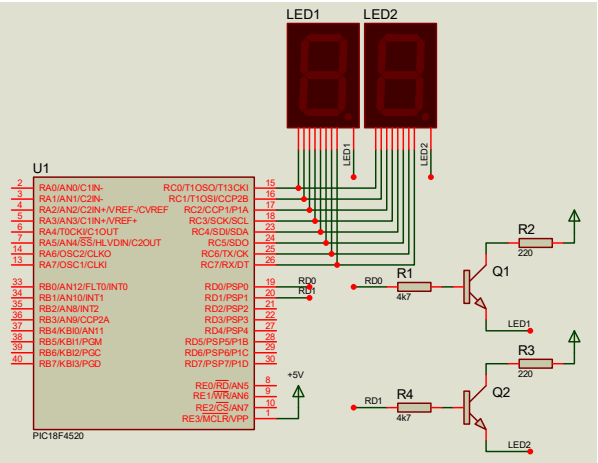
Bước 6: Chạy thử nghiệm hệ thống.

Trong bước này, ngoài việc đánh giá hoạt động của hệ thống theo các chức năng bằng cách quan sát còn có thể sử dụng các thiết bị đo để kiểm tra. Nếu chưa đạt có thể quay lại các bước trên để hiệu chỉnh.

9.2. Nguyên tắc hiển thị trên LED 7 đoạn bằng phương pháp quét

Mục đích của hiển thị trên LED 7 đoạn bằng phương pháp quét: Tiết kiệm tài nguyên GPIO của vi điều khiển. Ví dụ nếu không sử dụng phương pháp quét (hình dưới, bên trái) sẽ cần 16 chân của vi điều khiển để điều khiển 2 LED. Trong khi nếu sử dụng phương pháp quét (hình dưới, bên phải) chỉ cần 10 chân của vi điều khiển để điều khiển 2 LED.

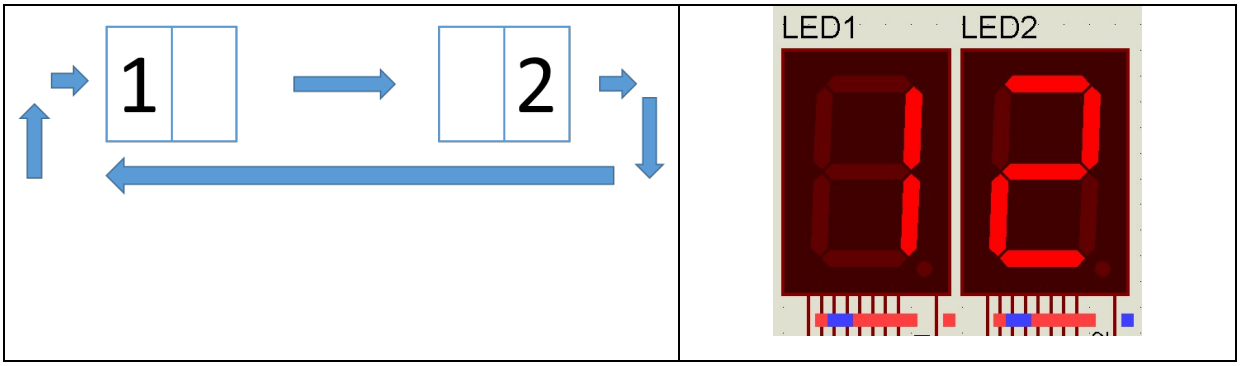




Nguyên tắc hiển thị trên LED 7 đoạn bằng phương pháp quét:

Phương pháp quét dựa trên nguyên lý lưu ảnh trên võng mạc. Tại 1 thời điểm chỉ có 1 LED 7 đoạn sáng, các LED khác tắt. Nói cách khác, các LED 7 đoạn luân phiên nhau sáng với các số theo yêu cầu (hình dưới, bên trái)

Thời gian sáng của mỗi LED <1/24 giây nên quan sát bằng mắt ta không nhận ra các LED sáng lần lượt mà có cảm giác như các LED sang đồng thời (hình dưới, bên phải).

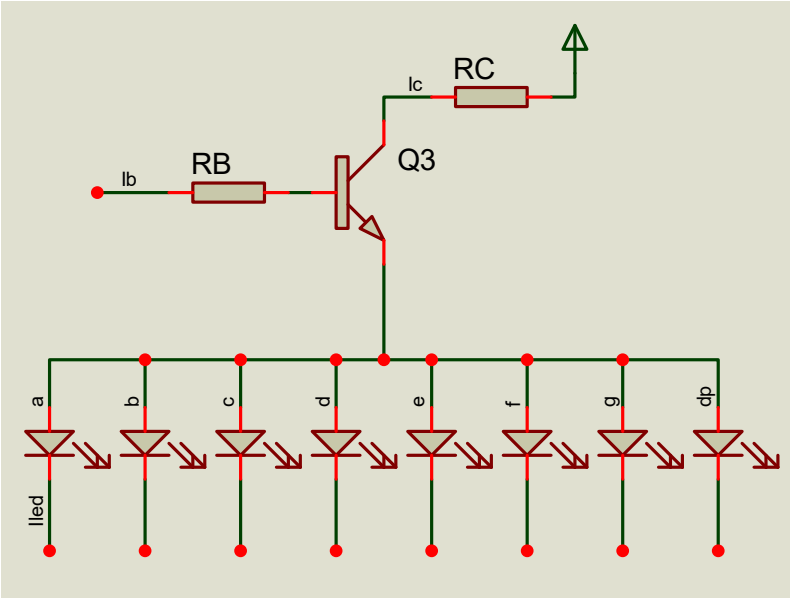


9.3. Thiết kế mạch hiển thị trên 04 LED 7 đoạn.

Bước 1: Phân tích bài toán và tính chọn thiết bị.

Giả thiết loại LED cần sử dụng có điện áp hoạt động là 3V, dòng điện định mức là 20mA/đoạn. Khi đó cần tính, chọn các linh kiện khác để có thể điều khiển hiển thị trên các LED này, bao gồm:

- Tính, chọn transitor, Rb, Rc; tính Ib (xem lại học phần điện tử tương tự).



- Kiểm tra dòng ra/vào (current sink/source) trên các chân của vi điều khiển. Nếu Ib>dòng ra và Iled> dòng vào thì cần sử dụng thêm vi mạch đệm dòng hoặc transitor. Trong trường hợp cụ thể này, Iled=20mA và Ib≈0,1mA nên có thể nối trực tiếp vào chân của PIC18F4520 do vi điều khiển này có dòng ra/vào

là 25mA (trang 3, tài liệu "PIC18F4520 Data Sheet") và tổng dòng ra/vào trên các chân Vss (Max current out of Vss pin), V_{DD} (Max current into V_{DD} pin) lần lượt là 300 mA và 250 mA (trang 323, tài liệu "PIC18F4520 Data Sheet").

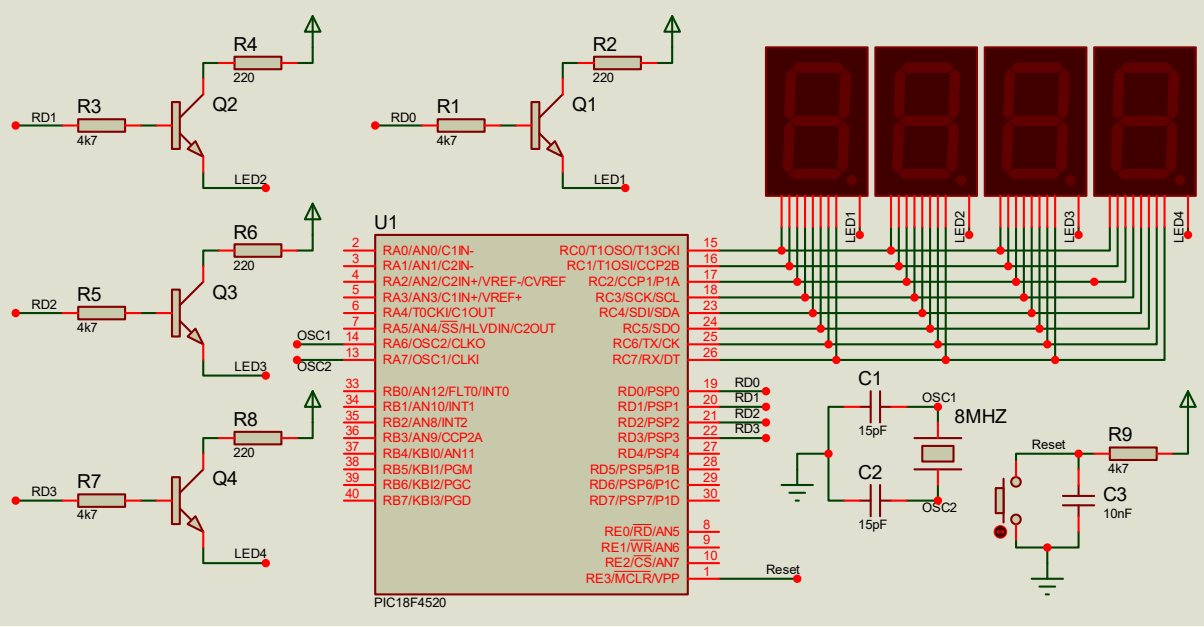
Maximum current out of Vss pin .....

Peripheral Highlights:

- Chọn giá trị của bộ dao động thạch anh: Với ứng dụng này không yêu tần số hoạt động của vi điều khiển nên giá trị của bộ dao động thạch anh có thể chọn từ 1Mhz đến 40Mhz.

Bước 2: Thiết kế sơ đồ nguyên lý

Từ các phân tích, tính chọn linh kiện ở bước 1, sơ đồ nguyên lý được xây dựng như sau:



Bước 3: Lập trình và mô phỏng

- Lập trình:

+ Tính mã của các số 0-9 với LED loại A chung, không sáng dấu chấm (dp) và lưu vào mảng:

```
unsigned char ma_led[]=
{0b11000000, //mã số 0
```

0b11111001, //mã số 1

…

0b10010000}; //mã số 9

+ Tính và chọn tham số, hàm tạo trễ để tạo thời gian trễ giữa các lần hiển thị:

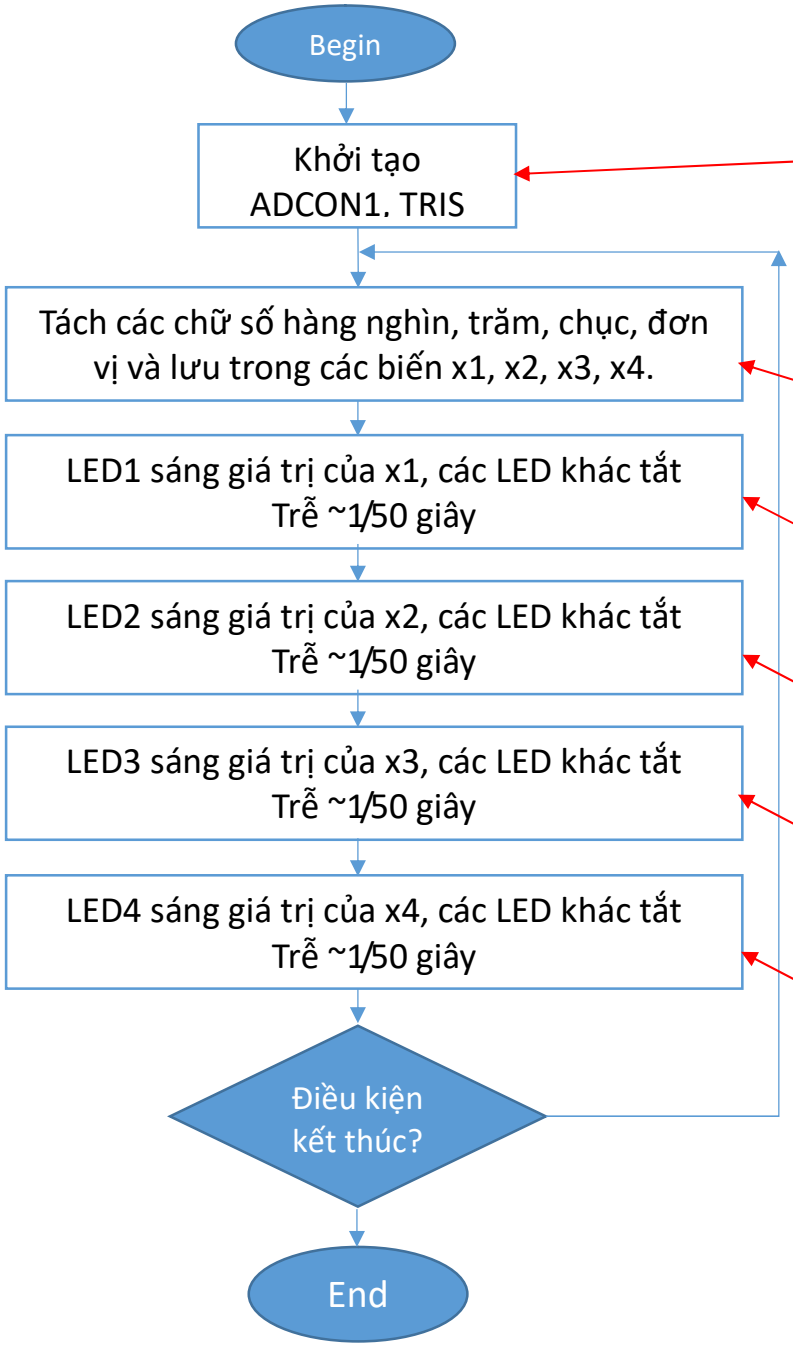
Tần số thạch anh: Fosc=8Mhz => Tosc=0,125 µs

=> Chu kỳ lệnh là Tins=0,5 µs

Nếu cần tạo thời gian trễ là 5 ms thì cần trễ 10.000 chu kỳ lệnh

=> chọn hàm Delay1KTCYx với tham số là 10 hoặc chọn hàm Delay10KTCYx với tham số là 1. Cụ thể là lệnh Delay1KTCYx(10) hoặc lệnh Delay10KTCYx(1)

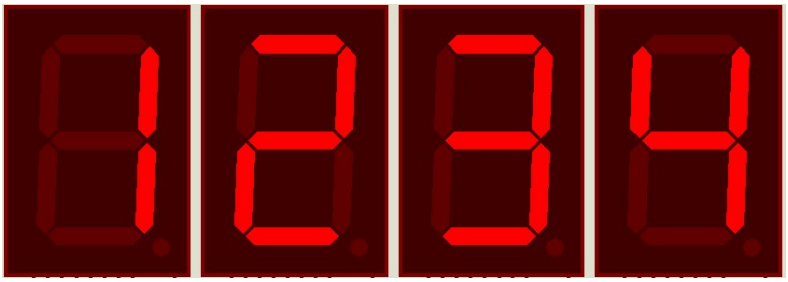
+ Xây dựng lưu đồ thuật toán và thể hiện bằng các câu lệnh:



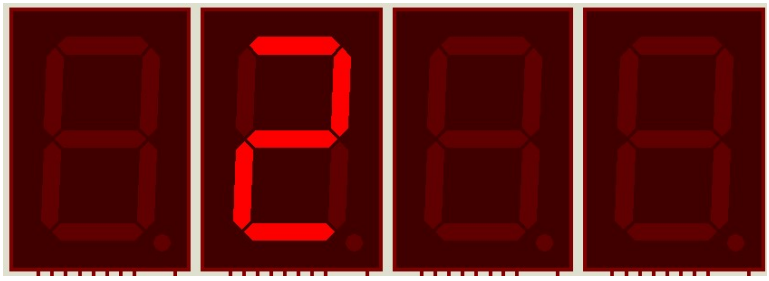

```
#include<p18f4520.h>
#include<delays.h>

#pragma config OSC = HS
#pragma config MCLRE = ON
#pragma config WDT = OFF
#pragma config PBADEN = OFF
#pragma config PWRT=ON
#pragma config BOREN=OFF
#pragma config LVP=OFF

unsigned char ma_led[]=
{0b11000000,0b11111001,
0b10100100,0b10110000,
0b10011001,0b10010010,
0b10000010,0b11111000,
0b10000000,0b10010000};
unsigned int x,x1,x2,x3,x4;

void main (void)
{
ADCON1=0x0F;
TRISC=0b00000000;
TRISD=0b00000000;
x=1234;
PORTD=0b00000000;
while(1)
        {
        x1=x/1000;
        x2=(x%1000)/100;
        x3=((x%1000)%100)/10;
        x4=((x%1000)%100)%10;

        PORTD=0b00000001;
        PORTC=ma_led[x1];
        Delay1KTCYx(10);

        PORTD=0b00000010;
        PORTC=ma_led[x2];
        Delay1KTCYx(10);

        PORTD=0b00000100;
        PORTC=ma_led[x3];
        Delay1KTCYx(10);

        PORTD=0b00001000;
        PORTC=ma_led[x4];
        Delay1KTCYx(10);
        }
}
```

- Mô phỏng:

+ Khi sử dụng lệnh Delay10KTCYx(1), thời gian quét (trễ giữa các lần hiển thị các số hàng nghìn, trăm, chục và đơn vị) phù hợp, LED hiển thị đúng yêu cầu:



+ Khi sử dụng lệnh Delay10KTCYx(10), thời gian quét tương ứng là 50ms, LED hiển thị lần lượt từng chữ số (sai yêu cầu):



- Tối ưu hóa chương trình:

Các dòng lệnh điều khiển LED1, 2, 3 và 4 sáng các giá trị của các biến x1, x2, x3 và x4 có tính chất lặp lại. Vì vậy có thể sử dụng mảng và vòng lặp như sau:

```
//Khai báo mảng gồm 4 phần tử là các giá trị xuất ra PORTD làm cho các
//LED1, 2, 3, 4 sáng:
unsigned char led[4]={0b00000001,0b00000010,0b00000100, 0b00001000};
// Khai báo mảng 4 phần tử lưu các giá trị của x1, x2, x2, x4 tách ra từ x
unsigned int so[4];
unsigned int i; //biến dùng cho vòng lặp for
…
        so[0]=x/1000;       //tách số hàng nghìn từ biến x, lưu vào phần tử có chỉ
                            // số  bằng 0 trong mảng so
        so[1]=(x%1000)/100;
        so[2]=((x%1000)%100)/10;
        so[3]=((x%1000)%100)%10;
```

```
for (i=0; i<=3; i++)
        {
        PORTD=led[i];
        PORTC=ma_led[so[i]];
        Delay10KTCYx(1);
        }
```