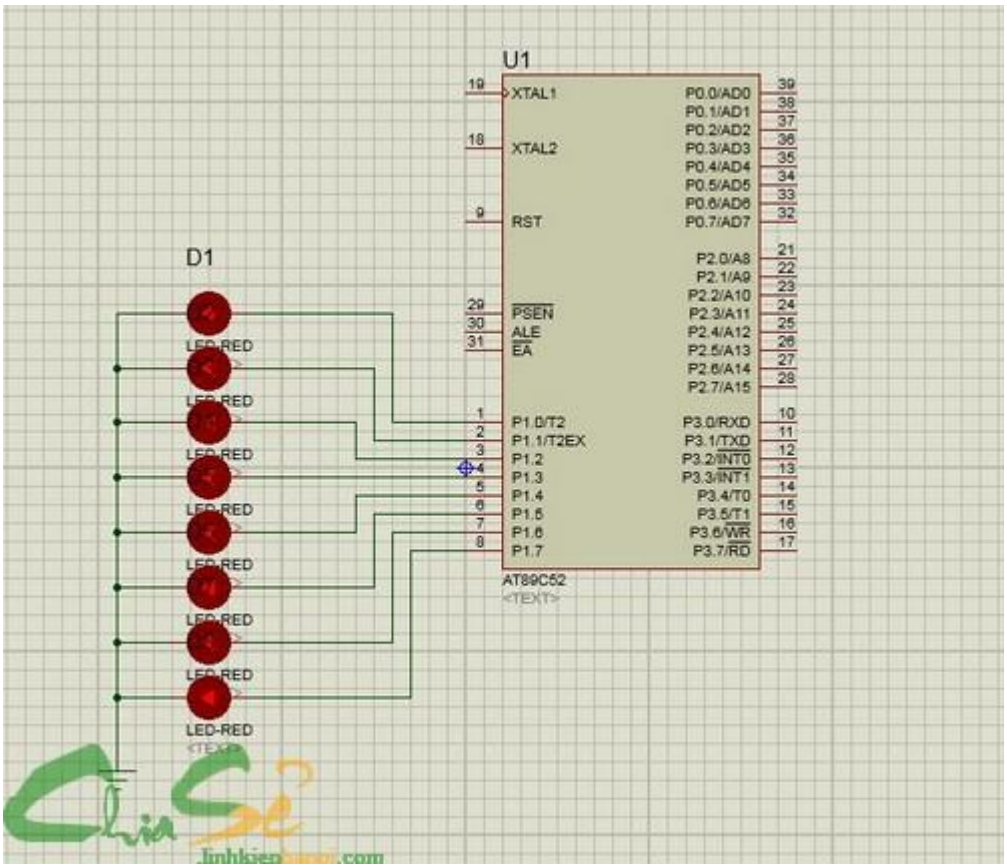
[BÀI 4]:
Lập trình nháy LED đơn cơ bản với 8051

Ở bài này mình sẽ hướng dẫn các bạn lập trình điều khiển Output cơ bản thực hành với LED đơn. Các bạn thực hiện vẽ mạch sau trong Proteus.



Các bạn chú ý: do là mô phỏng thử nghiệm chương trình nên mình vẽ mạch ở mức độ cơ bản tuy nhiên ở mạch thật các bạn cần bổ sung nguồn, thạch anh và mạch reset cho mạch để chương trình có thể chạy được. Với mạch như trên chúng ta đã có thể thực hiện mô phỏng chương trình cơ bản với LED đơn.

a.Lập trình điều khiển từng Pin của vi điều khiển.

8051 cho phép chúng ta tác động tới từng chân IO của vi điều khiển. Để sử dụng tính năng này chúng ta có thể sử dụng khai báo **sbit** để định nghĩa tên cho chân chúng ta muốn sử dụng. Khai báo sbit được thực hiện theo mẫu sau:

```
sbit Tên_Chân_Định_Nghĩa = Địa_Chỉ_Pin;
```

Ví dụ cụ thể như sau:

> *sbit LED = P1^0;*

Ở đây, chân P1.0 được định nghĩa có tên là LED.

Sau khi khai báo sbit như trên chúng ta có thể sử dụng Tên chân mà chúng ta mới định nghĩa để có thể điều khiển độc lập từng chân. Khi các bạn gán chân đó bằng giá trị 0 thì đầu ra sẽ ở mức 0V còn khi các bạn gán cho chân đó ở mức 1 thì sẽ có mức điện áp ra là 5V.

Ví dụ ở chương trình sau chúng ta sẽ sử dụng định nghĩa sbit để điều khiển chân P1.0 của 8051 thực hiện chức năng nhấp nháy một con LED. Các bạn xem code theo mẫu sau.

```
 1  /* Khai bao su dung thu vien cho 89c51 */
 2  #include <REGX51.H>
 3
 4  /* Su dung sbit de dinh nghia cho tung chan cua Vi dieu khien*/
 5  sbit LED = P1^0;
 6
 7  /* Trien khai ham tao do tre */
 8  void Fn_Delay (unsigned int _vrui_Time)
 9  {
10          while(_vrui_Time--);
11 }
12
13 /* Chuong trinh chinh */
14 void main (void)
15 {
16          /* Vong lap vo han */
17          while(1)
18          {
19                  LED = 0;            // Xuat 0 ra chan LED
20                  Fn_Delay(5000);     // Tre 1 thoi gian
21                  LED = 1;            // Xuat 1 ra chan LED
22                  Fn_Delay(5000);     // Tre 1 thoi gian
23          }
24 }
```

Với việc lập trình như trên các bạn sẽ LED nối với chân P1.0 sẽ được thay đổi giữa sáng và tắt. Ở đây có hàm *Fn_Delay()* có chức năng thay đổi tốc độ nhấp nháy của

con LED. Khi tham số truyền vào hàm này càng lớn thì độ trễ càng lâu tức tần số nháy nháy các chậm. Các bạn có thể thay đổi thông số cho hàm này để thấy rõ hơn. Sau khi lập xong Project theo các bước đã hướng dẫn ở các bài trước các bạn gõ các dòng code trên vào và thực hiện *Build* để tạo ra file hex sau đó các bạn chuyển file hex đó vào IC 89C52 trên Proteus và chạy chế độ mô phỏng. Các bạn sẽ thấy kết quả. Như vậy là các bạn đã có thể điều khiển được một chân của 8051 ở chế độ Output.
Sau khi thực hiện cho chân P1.0 các bạn có thể thực hiện trên các chân khác hoặc thực hiện nhiều hơn một chân nếu các bạn muốn.

b.Lập trình điều khiển theo Port của vi điều khiển.

Ngoài điều khiển theo từng chân chúng ta có thể xuất dữ liệu ouput theo cả Port trên 8051. 8051 là một vi điều khiển 8 bit và một Port của nó cũng chỉ có 8 pin. Vì thế bạn có thể xuất data cho 8 pin một lúc với 1 dòng lệnh với điều kiện các Pin đó chung một Port. Để can thiệp đến một Port các bạn chỉ cần gán dữ liệu các bạn mong muốn cho tên Port đó. Ví dụ như sau:

```
1  P2 = 0x55;
```

Với câu lệnh trên các bạn đã xuất dữ liệu trên Port 2 như sau: 0-1-0-1-0-1-0-1 tương ứng với chiều từ P2.7 tới P2.0.

Giới thiệu về toán tử | và &:

Toán tử | là toán tử OR bit còn toán tử & là toán tử AND bit. Các toán tử trên thực hiện theo công thức sau:

> *0 | 0 = 0*
> *0 | 1 = 1*
> *1 | 1 = 1*
> *0 & 0 = 0*
> *0 & 1 = 0*
> *1 & 1 = 1*

Đây là 2 toán tử rất hay được sử dụng trong việc khi bạn muốn tác động đến một vài bit trong một Port mà các bạn muốn xuất dữ liệu. Toán tử | thường được dùng để đưa một Pin lên mức 1 còn toán tử & thường được sử dụng để đưa một Pin về mức 0.

Ví dụ như sau:

```
P1 = P1 | 0x01; // Đưa pin P1.0 lên mức 1
P1 = P1 & 0xFE; // Đưa pin P1.0 về mức 0
```

Ngoài ra mình giới thiệu với các bạn 2 toán tử << và >>. Đây là 2 toán tử dùng để dịch bít. Các bạn quan sát các ví dụ sau để hiểu rõ về 2 toán tử này:

```
0x01<<1 = 0x02; //00000001 -> 00000010
0x10>>1 = 0x08; //00010000 -> 00001000
```

Các bạn chú ý 4 toán tử này vì có thể các bạn sẽ gặp rất nhiều trong các bài toán sau này của các bạn.

Sau đây là một ví dụ mình sử dụng việc xuất dữ liệu trên một Port tạo hiệu ứng LED chạy từ trên xuống và lặp lại. Các bạn có thể tham khảo.

```
1  /* Khai bao su dung thu vien cho 89c51 */
2
3  #include <REGX51.H>
4
5  /* Trien khai ham tao do tre */
6  void Fn_Delay (unsigned int _vrui_Time)
7  {
8          while(_vrui_Time--);
9  }
10
11 /* Chuong trinh chinh */
12 void main (void)
13 {
14         /* Khai bao bien i */
15         unsigned char i;
16         /* Vong lap vo han */
17         while(1)
18         {
19                 for (i = 0; i < 8; i++)
20                 {
21                         P1 = 0x01<<i;
22                         Fn_Delay(5000);
23                 }
```

```
24          }
25 }
```

Các bạn thực hiện Build chương trình trên và đưa vào mạch trên Proteus để thử nghiệm chương trình. Chương trình có hiệu ứng có một LED sẽ chạy từ trên xuống dưới sau khi đến vị trí cuối cùng nó lại xuất phát từ vị trí đầu tiên.

3.Bài tập:

Bài 1: Lập trình điều khiển 2 chân LED luôn phiên sáng (Tức là khi 1 chân sáng thì chân còn lại sẽ tắt và 2 chân thực hiện liên tiếp xen kẽ nhau).

Bài 2: Lập trình trên một Port của 8051 thực hiện chức năng theo trò chơi xếp hình.

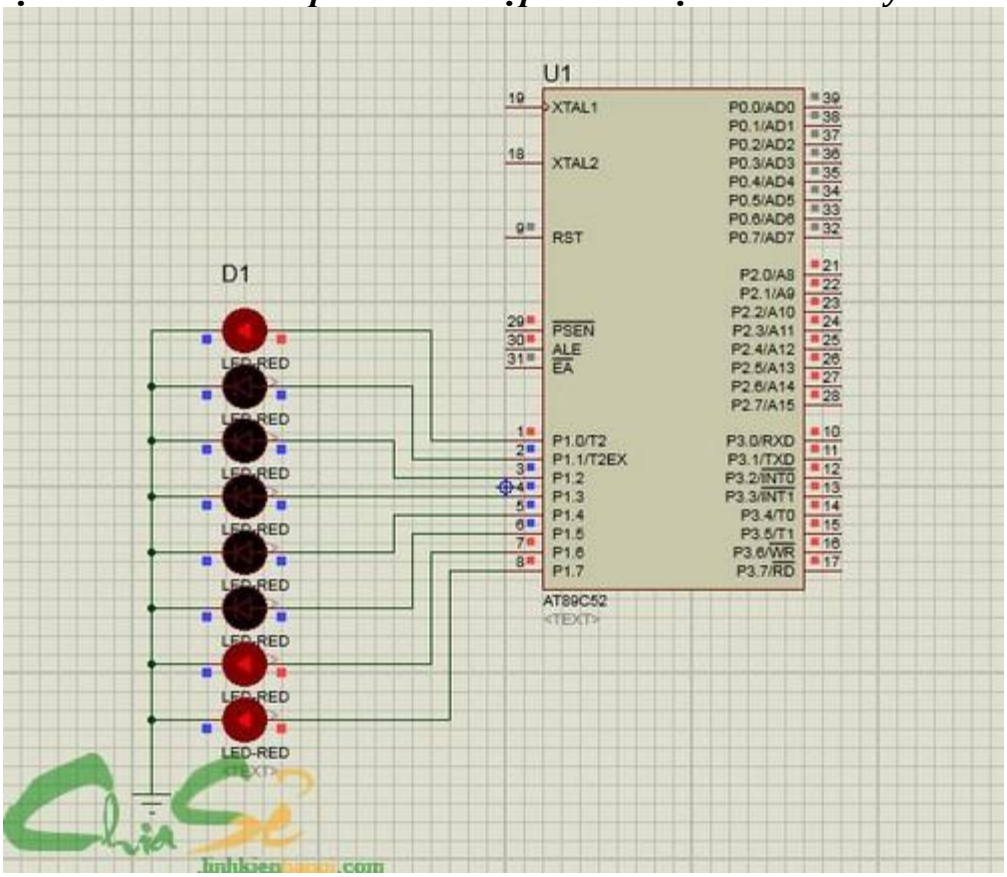
Cụ thể như sau:

– Đầu tiên trên cùng có 1 LED sáng sau đó LED rơi dần xuống dưới và sẽ dừng lại ở vị trí cuối cùng.

– Khi LED số 1 chạm đến vị trí dưới cùng thì xuất hiện 1 LED sáng ở vị trí ở trên cùng và lặp lại việc rơi xuống dưới và dừng lại khi đến vị trí ngay trên con LED cũ.

– Lặp lại như vậy cho đến khi số LED đầy trên cột và sẽ bắt đầu lại từ đầu.

Đây là một hình ảnh về kết quả của bài tập để các bạn hiểu rõ về yêu cầu.



Chúc các bạn thành công !