**Ngắt ngoài**

1. **Lý thuyết về ngắt.**
   1. **Ngắt (Interrupt) là gì?**
      1. Như tên của nó, là một số sự kiện khẩn cấp bên trong hoặc bên ngoài bộ vi điều khiển xảy ra, buộc vi điều khiển tạm dừng thực hiện chương trình hiện tại, phục vụ ngay lập tức nhiệm vụ mà ngắt yêu cầu – nhiệm vụ này gọi là trình phục vụ ngắt (ISR: Interrupt Service Routine).
      2. Lấy ví dụ**:**

Bộ vi điều khiển đóng vai trò như một vị bác sĩ, các thiết bị kiểm soát bởi vi điều khiển được coi như các bệnh nhân cần được bác sĩ phục vụ.Bình thường, vị bác sĩ sẽ hỏi thăm lần lượt từng bệnh nhân, đến lượt bệnh nhân nào được hỏi thăm nếu có bệnh thì sẽ được bác sĩ phục vụ, xong lại đến lượt bệnh nhân khác, và tiếp tục đến hết. Điều này tương đương với phương pháp thăm dò - hỏi vòng(Polling) trong vi điều khiển.Cứ như thế, nếu chúng ta có 10 bệnh nhân, thì bệnh nhân thứ 10 dù muốn hay không cũng phải xếp hàng chờ đợi 09 bệnh nhân trước đó. Giả sử trường hợp bệnh nhân thứ 10 cần cấp cứu thì sao? Anh ta sẽ gặp nguy cấp trước khi đến lượt hỏi thăm của bác sĩ mất! L Nhưng, nếu anh ta sử dụng phương pháp “ngắt” thì mọi chuyện sẽ ổn ngay. Lúc đó vị bác sĩ sẽ ngừng mọi công việc hiện tại của mình, và tiến hành phục vụ trường hợp khẩn cấp này ngay lập tức, xong việc bác sĩ lại trở về tiếp tục công việc đang dở. Điều này tương đương với ngắt (Interrupts) trong vi điều khiển.

* 1. **Điểm mạnh của phương pháp ngắt là:**

 Bộ vi điều khiển có thể phục vụ được rất nhiều thiết bị (tất nhiên là không tại cùng một thời điểm). Mỗi thiết bị có thể nhận được sự chú ý của bộ vi điều khiển dựa trên mức ưu tiên được gán cho nó. Đối với phương pháp thăm dò thì không thể gán mức ưu tiên cho các thiết bị vì nó kiểm tra tất cả mọi thiết bị theo kiểu hỏi vòng.

Quan trọng hơn, trong phương pháp ngắt thì bộ vi điều khiển còn có thể che (làm lơ) một yêu cầu phục vụ của thiết bị. Điều này lại một lần nữa không thể thực hiện được trong phương pháp thăm dò.

Lý do quan trọng nhất mà phương pháp ngắt được ưu chuộng là vì nó không lãng phí thời gian cho các thiết bị không cần phục vụ. Còn phương pháp thăm dò làm lãng phí thời gian của bộ vi điều khiển bằng cách hỏi dò từng thiết bị kể cả khi chúng không cần phục vụ.

* 1. **Sáu ngắt trong 8051**
     1. Sáu ngắt của 8051 được phân bố như sau:
* RESET: Khi chân RESET được kích hoạt từ 8051, bộ đếm chương trình nhảy về địa chỉ 0000H.  Đây là địa chỉ bật lại nguồn.
* 2 ngắt dành cho các bộ định thời: 1 cho Timer0 và 1 cho Timer1. Địa chỉ tương ứng của các ngắt này là 000BH và 001BH.
* 2 ngắt dành cho các ngắt phần cứng bên ngoài: chân 12 (P3.2) và 13 (P3.3) của cổng P3 là các ngắt phần cứng bên ngoài INT0 và INT1 tương ứng. Địa chỉ tương ứng của các ngắt ngoài này là 0003H và 0013H.
* Truyền thông nối tiếp: có 1 ngắt chung cho cả nhận và truyền dữ liệu nối tiếp. Địa chỉ của ngắt này trong bảng vector ngắt là 0023H.
  1. **Trình phục vụ ngắt**

Đối với mỗi ngắt thì phải có một **trình phục vụ ngắt** (**ISR**) hay trình quản lý ngắt để đưa ra nhiệm vụ cho bộ vi điều khiển khi được gọi ngắt. Khi một ngắt được gọi thì bộ vi điều khiển sẽ chạy trình phục vụ ngắt. Đối với mỗi ngắt thì có một vị trí cố định trong bộ nhớ để giữ địa chỉ **ISR** của nó. Nhóm vị trí bộ nhớ được dành riêng để lưu giữ địa chỉ của các **ISR** được gọi là **bảng** **vector ngắt**.



* 1. **Quy trình khi thực hiện một ngắt**
* Khi kích hoạt một ngắt bộ vi điều khiển thực hiện các bước sau:
* Nó hoàn thành nốt lệnh đang thực hiện và lưu địa chỉ của lệnh kế tiếp vào ngăn xếp.
* Nó cũng lưu tình trạng hiện tạ**i** của tất cả các ngắt.
* Nó nhảy đến một vị trí cố định trong bộ nhớ được gọi là bảng vector ngắt, nơi lưu giữ địa chỉ của một trình phục vụ ngắt.
* Bộ vi điều khiển nhận địa chỉ ISR từ bảng vector ngắt và nhảy tới đó. Nó bắt đầu thực hiện trình phục vụ ngắt cho đến lệnh cuối cùng của ISR và trở về chương trình chính từ ngắt.
* Khi bộ vi điều khiển quay trở về nơi nó đã bị ngắt. Trước hết nó nhận địa chỉ của bộ đếm chương trình PC từ ngăn xếp bằng cách kéo 02 byte trên đỉnh của ngăn xếp vào PC. Sau đó bắt đầu thực hiện tiếp các lệnh từ địa chỉ đó.
  1. **Các bước cho phép và cấm ngắt**

Khi bật lại nguồn thì tất cả mọi ngắt đều bị cấm (bị che), có nghĩa là không có ngắt nào được bộ vi điều khiển đáp ứng trừ khi chúng được kích hoạt.

Các ngắt phải được kích hoạt bằng phần mềm để bộ vi điều khiển đáp ứng chúng. Có một thanh ghi được gọi là thanh ghi cho phép ngắt **IE** (Interrupt Enable) – ở địa chỉ A8H chịu trách nhiệm về việc cho phép và cấm các ngắt.  trình bày chi tiết về thanh ghi **IE**.



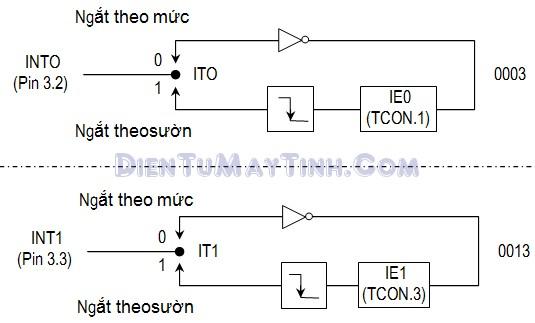
*Thanh ghi cho phép ngắt IE.*

Để cho phép một ngắt ta phải thực hiện các bước sau:

* Nếu EA = 0 thì không có ngắt nào được đáp ứng cho dù bit tương ứng của nó trong IEcó giá trị cao. Bit D7 - EA của thanh ghi IE phải được bật lên cao để cho phép các bit còn lại của thanh ghi hoạt động được.
* Nếu EA = 1 thì tất cả mọi ngắt đều được phép và sẽ được đáp ứng nếu các bit tương ứng của chúng trong IE có mức cao.
  1. **Lập trình các ngắt phần cứng bên ngoài**

         Bộ vi điều khiển 8051 có 2 ngắt phần cứng bên ngoài ở chân 12 (P3.2) và chân 13 (P3.3) gọi là ngắt INT0 và INT1.

          Như đã nói ở trên thì chúng được phép và bị cấm bằng việc sử dụng thanh ghi IE. Nhưng cấu hình cho ngắt ngoài có phần phức tạp hơn.Có hai mức kích hoạt cho các ngắt phần cứng ngoài: Ngắt theo mức và ngắt theo sườn.



*Ngắt ngoài****INT0****và****INT1***

**Ngắt theo mức**

Ở chế độ ngắt theo mức thì các chân INT0 và INT1 bình thường ở mức cao và nếu một tín hiệu ở mức thấp được cấp tới thì chúng ghi nhãn ngắt. Sau đó bộ vi điều khiển dừng tất cả mọi công việc nó đang thực hiện và nhảy đến bảng vector ngắt để phục vụ ngắt. Đây là chế độ ngắt mặc định khi cấp nguồn cho 8051.Tín hiệu mức thấp tại chân INTx phải được lấy đi trước khi thực hiện lệnh cuối cùng của trình phục vụ ngắt, nếu không một ngắt khác sẽ lại được tạo ra, và vi điều khiển sẽ thực hiện ngắt liên tục.

**Ví dụ :**

    Giả sử chân INT1 được nối đến công tắc bình thường ở mức cao. Mỗi khi nó  ấn xuống thấp phải bật một đèn LED ở chân P1.3 (bình thường Led tắt), khi nó được bật lên nó phải sáng vài giây. Chừng nào công tắc được giữ ở trạng thái thấp đèn LED phải sáng liên tục.

#include<at89x51.h> //Khai báo thư viện cho VÐK 89x51

main**()** //Chương trình chính

**{**

IE **=** 0x84**;** //cho phép ngắt ngoài 1

**while** **(**1**)** //vòng lặp vô hạn

**{**

//không làm gì

**}**

**}**

void nutan**(**void**)** interrupt 2 //Khai báo trình phục vụ ngắt ngoài 1

**{** //(mặc định là ngắt theo mức)

int a **=** 50000**;** //Biến đếm trễ

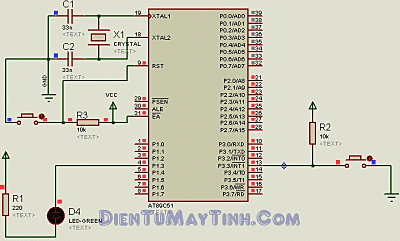
P1\_3 **=** 0**;** //Cho Led sáng

**while** **(**a**--)** **{}** //Trễ cho Led sáng vài giây

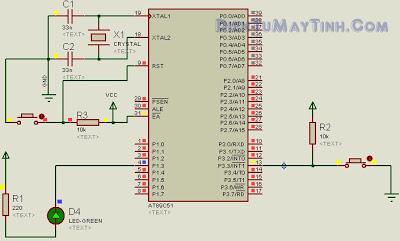
P1\_3 **=** 1**;** //Tắt Led

//Không cần xóa cờ ngắt

**}**

}

*Ấn công tắc xuống sẽ làm cho đèn LED sáng một thời gian.*



*Công tắc được giữ ở trạng thái ấn thì đèn LED sáng liên tục.*

**Lưu ý:**

Trong chương trình trên bộ vi điều khiển quay vòng liên tục trong vòng lặp while(1)của chương trình chính. Mỗi khi công tắc trên chân P3.3 (INT1) được kích hoạt thì bộ vi điều khiển thoát khỏi vòng lặp và nhảy đến bảng vector ngắt tại địa chỉ 0013H. Trình ISR cho INT1 bật đèn LED lên giữ nó một lúc và tắt nó trước khi trở về. Nếu trong lúc nó thực hiện lệnh cuối cùng để quay trở về từ ISR mà chân INT1 vẫn còn ở mức thấp thì bộ vi điều khiển khởi tạo lại ngắt, ngắt lại xảy ra 1 lần nữa.

Do vậy, để giải quyết vấn đề này thì chân INT1 phải được đưa lên cao trước thời điểm lệnh cuối cùng của ngắt được thực hiện.

Có một cách khác để giải quyết triệt để vấn đề trên: đó là sử dụng ngắt theo sườn.Khi đó với mỗi 1 lần ấn phím, dù thế nào ngắt cũng chỉ thực hiện 1 lần.

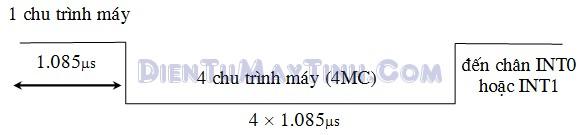
Trước khi tìm hiểu ngắt theo sườn là gì? Ta hãy xem qua ngắt theo mức hoạt động như thế nào.

**Trích mẫu ngắt theo mức:**

        Các chân P3.2 và P3.3 bình thường được dùng cho vào/ra nếu các Bit INT0 vàINT1 trong thanh ghi IE không được kích hoạt. Sau khi các ngắt phần cứng trong thanh giIE được kích hoạt thì bộ vi điều khiển duy trì trích mẫu trên chân INTx đối với tín hiệu mức thấp 1 lần trong 1 chu trình máy.

Theo bảng dữ liệu từ nhà sản xuất của bộ vi điều khiển thì “chân ngắt phải được giữ ở mức thấp cho đến khi bắt đầu thực hiện trình phục vụ ngắt ISR. Nếu chân INTx được đưa trở lại mức cao trước khi bắt đầu thực hiện ISR thì sẽ chẳng có ngắt nào xảy ra”. Do vậy, để bảo đảm việc kích hoạt ngắt phần cứng tại các chân INTx phải đảm bảo rằng thời gian tồn tại tín hiệu mức thấp là khoảng 4 chu trình máy và không được bé hơn, nếu không đủ lâu thì ngắt không được thực hiện.

Tuy nhiên trong quá trình kích hoạt ngắt theo mức thấp nên nó lại phải đưa lên mức cao trước khi ISR thực hiện lệnh cuối cùng và lại theo bảng dữ liệu từ nhà sản xuất thì “nếu chân INTx vẫn ở mức thấp sau lệnh cuối cùng của trình phục vụ ngắt thì một ngắt khác lại sẽ được kích hoạt”. Điều này do một thực tế là ngắt theo mức không được chốt.



*Thời gian tối thiểu của xung ngắt theo mức thấp (XTAL = 11.0592MHz)*

* 1. **Ngắt theo sườn**

Ngắt theo sườn là ngắt sẽ xảy ra khi có một sườn âm xuất hiện trên các chân ngắt của vi điều khiển. Điều này làm cho ngắt theo sườn khắc phục được nhược điểm của ngắt theo mức như ta đã thấy ở trên.Để kích hoạt chế độ ngắt theo sườn thì chúng ta phải viết chương trình cài đặt cho các bit của thanh ghi TCON:



*Thanh ghi TCON.*

**Các Bit IT0 và IT1:**

       Các bit TCON.0 và TCON.2 được coi như là các bit IT0 và IT1 tương ứng. Đây là các bit xác định kiểu ngắt theo sườn xung hay theo mức xung của các ngắt phần cứng trên chân INT0 và INT1 tương ứng. Khi bật lại nguồn cả 2 bit này đều có mức 0 để biến chúng thành ngắt theo tín hiệu mức thấp. Lập trình viên có thể điều khiển một trong số chúng lên cao để chuyển ngắt phần cứng bên ngoài thành ngắt theo sườn.

**Các Bit IE0 và IE1:**

       Các bit TCON.1 và TCON.3 còn được gọi là IE0 và IE1 tương ứng. Các bit này được 8051 dùng để bám kiểu ngắt theo sườn xung, nếu các bit IT0 và IT1 bằng 0 thì có nghĩa là các ngắt phần cứng là ngắt theo mức thấp và các bit IE0 và IE1 sẽ không dùng đến. Các Bit IE0 và IE1 chỉ được 8051 dùng để chốt sườn xung từ cao xuống thấp trên các chân INT0 và INT1. Khi có chuyển dịch sườn xung trên chân INT0 (hay INT1) thì 8051 đánh dấu (bật lên cao) các bit IEx trên thanh ghi TCON và nhảy đến bảng vector ngắt và bắt đầu thực hiện trình phục vụ ngắt ISR. Trong khi 8051 thực hiện ISR thì không có một sườn xung nào được ghi nhận trên chân INT0 (hay INT1) để ngăn mọi ngắt trong ngắt. Chỉ trong khi thực hiện lệnh cuối của trình phục vụ ngắt ISR thì các bit IEx mới được 8051 tự động xóa, và các chân ngắt lại hoạt động bình thường.

Ta thấy rằng các bit  IE0 và IE1 được 8051 sử dụng bên trong để báo có một ngắt đang được xử lý hay không. Hay nói cách khác là lập trình viên không phải quan tâm đến các bit này.

**Các Bit TR0 và TR1:**

       Đây là những bit D4 và D6 (hay TCON.4 và TCON.6) của thanh ghi TCON. Các bit này đã được giới thiệu ở các bài trước, chúng được dùng để khởi động và dừng các bộ định thời Timer0 và Timer1 tương ứng.

**Các Bit TF0 và TF1:**

       Các bit này là D5 (TCON.5) và D7 (TCON.7) của thanh ghi TCON mà đã được giới thiệu ở các bài trước. Chúng được sử dụng bởi các bộ Timer0 và Timer1 tương ứng để báo rằng các bộ định thời bị tràn hay quay về không.

Để hiểu rõ sự khác biệt của ngắt theo sườn âm, ta xét ví dụ 5. Chú ý rằng sự khác nhau duy nhất giữa ví dụ 5 và ví dụ 4 là ở lệnh chuyển ngắt INT1 về kiểu ngắt theo sườn. Khi sườn âm của tín hiệu được cấp đến chân INT1 thì đèn LED sẽ bật lên một lúc. Đèn LED có thời gian sáng phụ thuộc vào độ trễ bên trong ISR của INT1. Trong ví dụ 4 do bản chất ngắt theo mức của ngắt thì đèn LED còn sáng chừng nào tín hiệu ở chân INT1 vẫn còn ở mức thấp. Nhưng trong ví dụ 5 này để bật lại đèn LED thì xung ở chân INT1phải được đưa lên cao rồi sau đó bị hạ xuống thấp để tạo ra một sườn âm làm kích hoạt ngắt.

**Ví dụ:**

#include<regx52.h > //Khai báo thư viện cho VĐK 89x51

main**()** //Chương trình chính

**{**

IE **=** 0x84**;** //cho phép ngắt ngoài 1

IT1 **=** 1**;** //Thiết lập ngắt ngoài 1 theo sườn âm

**while** **(**1**)** //vòng lặp vô hạn

**{**

//không làm gì

**}**

**}**

void nutan**(**void**)** interrupt 2 //Khai báo trình phục vụ ngắt ngoài 1

**{** //(mặc định là ngắt theo mức)

int a **=** 50000**;** //Biến đếm trễ

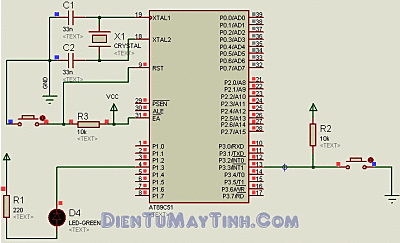
P1\_3 **=** 0**;** //Cho Led sáng

**while** **(**a**--)** **{}** //Trễ cho Led sáng vài giây

P1\_3 **=** 1**;** //Tắt Led

//Không cần xóa cờ ngắt

**}**

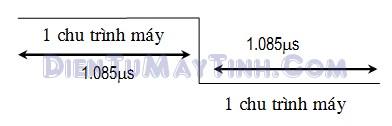


*Mô phỏng ngắt ngoài 1 theo sườn âm*

Dù công tắc được giữ, cũng chỉ có 1 ngắt xảy ra.

**Trình mẫu ngắt theo sườn**

       Trước khi kết thúc phần này ta cần trả lời câu hỏi: vậy thì ngắt theo sườn được trích mẫu thường xuyên như thế nào? Trong các ngắt theo sườn, nguồn xung phải giữ ở mức cao tối thiểu là 1 chu kỳ máy, và xung thấp cũng phải kéo dài 1 chu kỳ máy nữađể đảm bảo bộ vi điều khiển nhìn thấy được sự chuyển dịch từ cao xuống thấp của sườn âm.



*Thời hạn xung tối thiểu để phát hiện ra các ngắt theo sườn âm với tần số XTAL = 11.0592MHz*

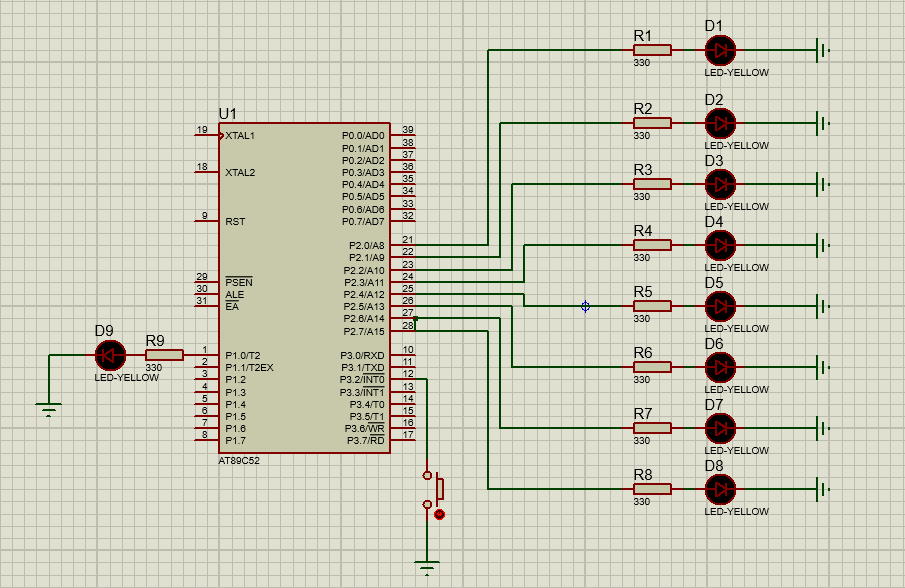
Sườn âm của xung được chốt bởi 8051 và được giữ bởi thanh ghi TCON. Các bitTCON.1 (IE0) và TCON.3 (IE1) giữ các sườn được chốt của chân INT0 và INT1 tương ứng như chỉ ra trên hình 11. Chúng hoạt động như các cờ “ngắt đang được phục vụ” (Interrupt-in-server). Khi một cờ “ngắt đang được phục vụ” bật lên thì nó báo rằng ngắt hiện nay đang được xử lý và trên chân INTx này sẽ không có ngắt nào được đáp ứng chừng nào ngắt này chưa được phục vụ xong. Đây giống như tín hiệu báo bận ở máy điện thoại.

Ngoài ra cần phải nhấn mạnh 2 điểm dưới đây khi quan tâm đến các bit IE0 và IE1của thanh ghi TCON:

·        Khi các trình phục vụ ngắt ISR kết thúc: Các Bit IE0 và IE1 được tự động xoá để báo rằng ngắt được hoàn tất và 8051 sẵn sàng đáp ứng ngắt khác trên chân đó. Để ngắt khác được nhận và thì tín hiệu trên chân đó phải trở lại mức cao và sau đó nhảy xuống thấp để được phát hiện như một ngắt theo sườn âm.

·        Trong thời gian trình phục vụ ngắt đang được thực hiện thì chân INTx bị làm ngơ, 8051 không quan tâm đến nó có bao nhiêu lần chuyển dịch từ cao xuống thấp. Trong thực tế điều này có được là do các bit IEx. Vì lý do này mà các bit IEx được gọi là các cờ báo “ngắt đang được phục vụ”, cờ này sẽ lên cao khi 1 sườn âm được phát hiện trên chân INTx và giữ ở mức cao trong toàn bộ quá trình thực hiện ISR. Nó chỉ bị xoá sau lệnh cuối cùng của ISR. Do vậy, ta cũng sẽ không bao giờ cần đến các lệnh xoá cờ này trong trình phục vụ ngắt đối với các ngắt cứng INT0 và INT1.

1. **Lập trình ngắt ngoài với nút nhấn điều khiển LED:**
   1. **Sơ đồ nguyên lý:**



**Nguyên lí hoạt đông**:Chương trình đang chạy hàm **Sangdan8led** thì bấm nút ở chân P3\_2 => khi đó chương trình chạy chương trình ngắt.Sau khi hoàn thành chương trình ngắt thì tiếp tục hoàn thành chương trình **Sangdan8led**

* 1. **Code:**

#include<main.h>

#include"delay.h"

#define LED P2

sbit LED1 **=** P1**^**0**;**

void Sangdan8led**()** // hàm sáng dần 8 LED

unsigned char i**;**

LED **=** 0x00**;** // LED = 0000 0000 => tắt tất cả các LED

**for(**i**=**0**;**i**<**8**;**i**++)**

**{**

LED **=** **(**LED**<<**1**)|**0x01**;** //thực hiện dịch trái sáng dần LED

Delay\_ms**(**500**);**

**}**

**for(**i**=**0**;**i**<**8**;**i**++)**

**{**

LED **=** LED **>>** 1**;** //thực hiện phép dịch phải tắt dần LED

Delay\_ms**(**500**);**d

**}**

**}**

void main**()**

**{** LED1 **=** 0**;** // thiết lập mức 0 cho P1.0

EX0 **=** 1**;** // cho phép ngắt ngoài 0

EA **=** 1**;** // cho phép ngắt ngoài toàn cục

**while(**1**)**

**{**

Sangdan8led**();** // gọi hàm sáng dần 8 LED

**}**

**}**

void ngat**()** interrupt 0 // gọi hàm ngắt ngoài 0

**{**

unsigned char i**;**

**for(**i**=**0**;**i**<**5**;**i**++)**

**{**

LED1**=**1**;** // LED1 sáng

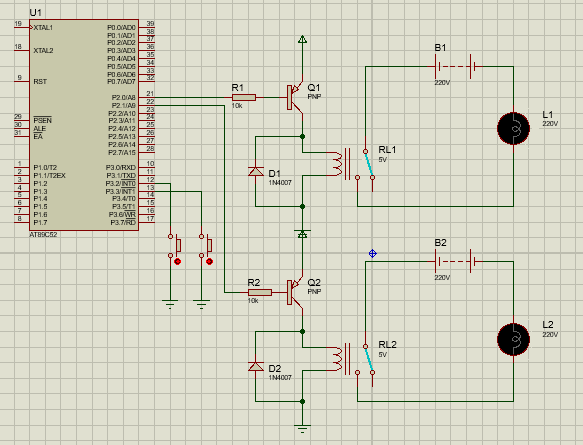
Delay\_ms**(**500**);**

LED1**=**0**;** // LED1 tắt

Delay\_ms**(**500**);**

**}**

1. **Lập trình nhấn nút sử dụng ngắt ngoài bật, tắt 2 bóng đèn 220V**
   1. **Sơ đồ nguyên lý**



**Nguyên lý hoạt động**: nếu nút bấm ở chân P3.2 bấm thì chương trình phục vụ ngắt 0 thực hiện và khi đó L1 sáng và nếu bấm lần thứ 2 thì chương trình phục vụ ngắt 0 lại thực hiện và khi đó L1 tắt.Tương tự đối với nút bấm thứ 2 ở chân P3.3 ,L2 và chương trình phục vụ ngắt 1.

* 1. **Code**

include "main.h" //Khai báo các thư viện cho vi điều khiển

sbit LED1 **=** P2 **^** 0**;** // gắn chân LED1 là chân P2.0

sbit LED2 **=** P2 **^** 1**;** // gắn chân LED2 là chân P2.1

void main**(**void**)**

**{** //chương trình chính

EX0 **=** 1**;** //cho phép ngắt ngoài 0

EX1 **=** 1**;** //cho phép ngắt ngoài 1

IT0 **=** 1**;** //Thiết lập ngắt ngoài 0 theo sườn âm

IT1 **=** 1**;** //Thiết lập ngắt ngoài 1 theo sườn âm

EA **=** 1**;** //cho phép ngắt toàn cục

LED1 **=** 0**;** //ban đầu LED1 tắt

LED2 **=** 0**;** //ban đầu LED2 tắt

**while** **(**1**)** //vòng lặp vô hạn

**{**

**}**

**}**

void ISR\_EX0**(**void**)** interrupt 0 //Khai báo trình phục vụ ngắt ngoài 0

**{**

LED1 **=** **~**LED1**;** //đảo LED1

**}**

void ISP\_EX1**(**void**)** interrupt 2 //Khai báo trình phục vụ ngắt ngoài 1

**{**

LED2 **=** **~**LED2**;** //đảo LED2

**}**