BÀI 3. NGẮT NGOÀI

Mục tiêu bài học:

Sinh viên nhớ và hiểu:

- Khái quát về ngắt

  + Khái niệm về ngắt

  + Lý do cần sử dụng ngắt

- Ngắt ngoài trên PIC18F4520

  + Các vector

  + Các nguồn ngắt

- Quá trình thực hiện ngắt của vi điều khiển
- Các bit điều khiển ngắt ngoài;
- Các bước lập trình sử dụng ngắt ngoài.

Sinh viên vận dụng các kiến thức đã học để lập trình, mô phỏng hoạt động của ngắt ngoài, bao gồm:

- 01 nguồn ngắt với mức ưu tiên cao;
- 01 nguồn ngắt với mức ưu tiên thấp;*
- 02 nguồn ngắt với mức ưu tiên khác nhau.*

**: Học trực tiếp (trên lớp).*

3.1. Khái quát về ngắt

Khái niệm chung: Ngắt là tạm thời dừng công việc hiện tại và chuyển sang thực hiện một nhiệm vụ khác (thường là cấp thiết, quan trọng hơn) sau đó lại quay lại thực hiện tiếp công việc cũ (đang thực hiện dở).

Khái niệm ngắt ở vi điều khiển: Khi vi điều khiển đang thực hiện các lệnh ở chương trình chính (CTC), nếu có tín hiệu ngắt (giả thiết là trước đó đã cho phép ngắt), vi điều khiển sẽ thực hiện nốt lệnh đang thực hiện dở (giả sử là lệnh thứ n) và tạm dừng CTC, chuyển sang thực hiện chương trình con phục vụ ngắt (CTCPVN). Sau khi thực hiện xong CTCPVN, vi điều khiển lại quay trở lại thực hiện tiếp các lệnh ở CTC (từ lệnh thứ n+1).

Vai trò của khối xử lý ngắt: Giúp cho vi điều khiển trở nên "đa nhiệm" hơn, ví dụ: Vừa có thể thực hiện điều khiển tuần tự, vừa có thể trao đổi dữ liệu.

3.2. Ngắt ngoài trên PIC18F4520

- PIC18F4520 có 02 vector ngắt khác nhau, bao gồm:

    + 0x0008: vector ngắt ưu tiên cao;

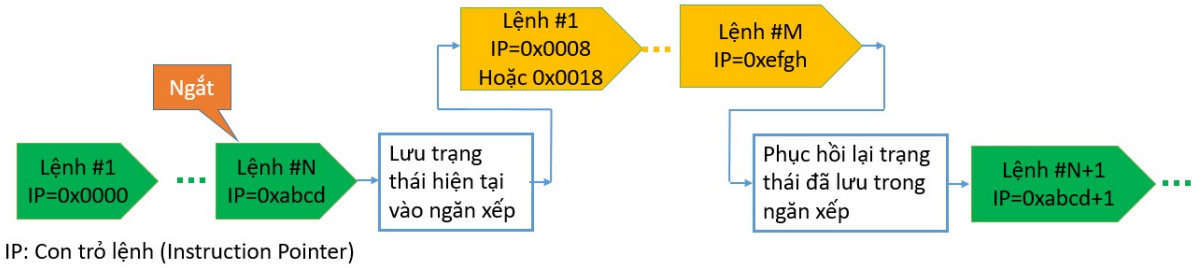
    + 0x0018: vector ngắt ưu tiên thấp. (xem thêm mục 4.2 trong giáo trình hoặc mục 9.0 trong tài liệu PIC18F4520 datasheet)

- PIC18F4520 có 15 nguồn ngắt khác nhau (xem thêm mục 4.2 trong giáo trình hoặc mục 9.0 trong tài liệu PIC18F4520 datasheet.

- Ngắt ngoài (External Interrupt): Là ngắt gây ra bởi nguyên nhân đến từ bên bên ngoài vi điều khiển. PIC184520 có 3 nguồn ngắt ngoài, gọi tắt là INT0, INT1 và INT2. Để gây ra ngắt cần có mạch điện bên ngoài tạo ra sự chuyển mức logic từ "0" sang "1" (thường được gọi là sườn dương) hoặc sự chuyển mức logic từ "1" sang "0" (thường được gọi là sườn âm) đưa đến chân RB0/INT0 hoặc RB1/INT1 hoặc RB2/INT2

3.3. Quá trình thực hiện ngắt của vi điều khiển

Khái niệm về con trỏ lệnh: Là một thanh ghi đặc biệt, luôn trỏ và lệnh tiếp theo sẽ được thực hiện. Khi hoạt động, vi điều khiển đọc (lấy) mã lệnh ở ô nhớ trong bộ nhớ chương trình có địa chỉ tương ứng với giá trị của IP để giải mã và thực hiện lệnh.



Quá trình thực hiện ngắt của vi điều khiển:

Giả thiết tín hiệu ngắt đến khi vi điều khiển đang thực hiện ở lệnh thứ N trong CTC, khi đó vi điều khiển sẽ thực hiện theo trình tự sau:

- Thực hiện nốt lệnh thứ N; lưu giá trị của các thanh ghi vào ngăn xếp, trong đó có IP=0xabcd;

- Chuyển sang thực hiện CTCPVN bằng cách gán cho IP giá trị 0x0008 (nếu là ngắt có mức ưu tiên cao) hoặc 0x0018 (nếu là ngắt có mức ưu tiên thấp).

- Sau khi thực hiện xong các lệnh của CTCPVN, vi điều khiển quay lại thực hiện tiếp các lệnh của CTC bằng cách gán cho IP giá trị 0xabcd+1 (nếu lệnh thứ N được mã hóa bằng 1 byte) đồng thời phục hồi lại giá trị của các thanh khi khác.

3.4. Các bit điều khiển ngắt ngoài

Các bit điều khiển ngắt ngoài nằm trong các thanh ghi sau (xem thêm mục 4.4 trong giáo trình hoặc mục 9.1-9.6 trong tài liệu PIC18F4520 datasheet):

- Thanh ghi ADCON1:

Thiết lập các chân AN0-AN12 là vào/ra số hoặc vào tương tự.

—	—	VCFG1	VCFG0	**PCFG3**	**PCFG2**	**PCFG1**	**PCFG0**
bit 7							bit 0

- Thanh ghi TRISB:

Thiết lập các chân RB0-RB7 có chiều vào hoặc ra (số).

Khi sử dụng các nguồn ngắt ngoài INT0, INT1 hoặc INT2, cần thiết lập chiều vào cho các chân RB0/INT0, RB1/INT1 hoặc RB2/INT2.

Ví dụ: Khi sử dụng nguồn ngắt INT1 cần viết:

TRISB=0bxxxxxx1x;

Trong đó: x có thể là 0 hoặc 1 nếu các chân RB0, RB2-RB7 không được sử dụng.

- Thanh ghi INTCON:

GIE/GIEH	**PEIE/GIEL**	TMR0IE	**INT0IE**	RBIE	TMR0IF	**INT0IF**	RBIF(1)
bit 7							bit 0

- Thanh ghi INTCON2

RBPU	**INTEDG0**	**INTEDG1**	**INTEDG2**	—	TMR0IP	—	RBIP
bit 7							bit 0

- Thanh ghi INTCON3

INT2IP	**INT1IP**	—	**INT2IE**	**INT1IE**	—	**INT2IF**	**INT1IF**
bit 7							bit 0

- Thanh ghi RCON

IPEN	SBOREN	—	/RI	/TO	/PD	/POR	/BOR
bit 7							bit 0

3.5. Các bước lập trình sử dụng ngắt ngoài

3.5.1. Khung chương trình có sử dụng ngắt và cách viết

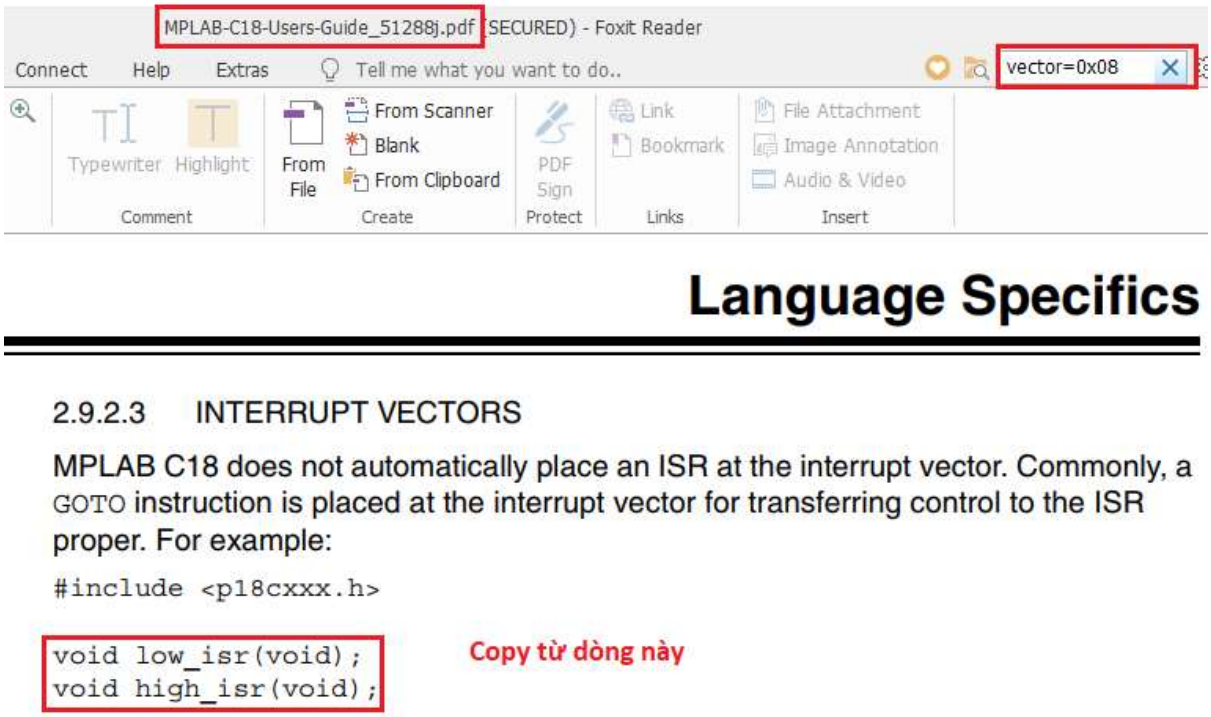
Ngoài các phần tương tự như một chương trình không sử dụng ngắt (khai báo thư viện, biến/hằng số, cấu hình, chương trình chính), khung chương trình có sử dụng ngắt có thêm 02 điểm để viết các CTCPVN (phần tô vàng bên dưới) tương ứng với ngắt có mức ưu tiên cao (vector 0x0008) và ngắt có mức ưu tiên thấp (vector 0x0018), cụ thể như sau:

```
//khai báo thu viện
//Cấu hình hệ thống
void low_isr(void);
void high_isr(void);
#pragma code low_vector=0x18
void interrupt_at_low_vector(void)
{_asm GOTO low_isr _endasm}
#pragma code
#pragma interruptlow low_isr
void low_isr (void)
{
// Các lệnh của CTCPVN với mức ưu tiên thấp
}
#pragma code high_vector=0x08
void interrupt_at_high_vector(void)
{_asm GOTO high_isr _endasm}
#pragma code
#pragma interrupt high_isr
void high_isr (void)
{
//Các lệnh của CTCPVN với mức ưu tiên cao
}
```

```
void main()
{
// Các lệnh khởi tạo ngắt
// Các lệnh khác thuộc CTC
}
```

TIPs: Sử dụng "Tài liệu tra cứu PIC" hoặc tài liệu "MPLAB® C18 C COMPILER USER'S GUIDE" để viết khung chương trình:

- Nhập từ khóa vector=0x08, copy phần khung CTCPVN ở mục 2.9.2.3:



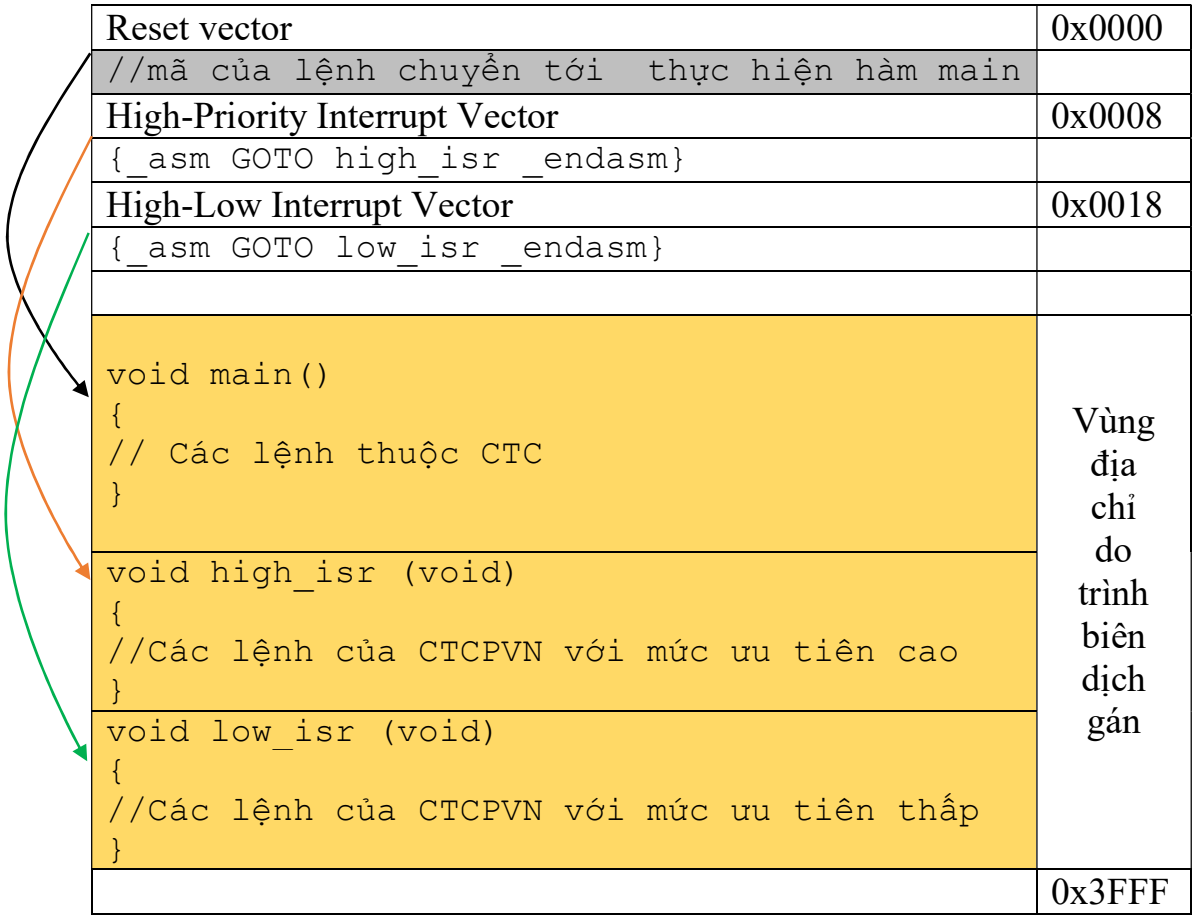
- Dán (paste) vào sau các câu lệnh cấu hình (#pragma):

```
#pragma config  OSC=HS
#pragma config  WDT=OFF
#pragma config  MCLRE=ON
#pragma config  PBADEN=OFF
#define LED5 PORTBbits.RB7
```

Đặt con trỏ và dán vào điểm này

```
void low_isr(void);
void high_isr(void);
```

Khi biên dịch, các phần khác nhau trong khung chương trình trên sẽ được trình biên dịch bố trí vào các vị trí khác nhau trong bộ nhớ chương trình, cụ thể như sau:



Khi reset, vi điều khiển sẽ đọc mã lênh ở địa chỉ 0x0000. Do vùng nhớ dành cho reset vector chỉ gồm 08 byte có địa chỉ từ 0x0000 đến 0x0007 nên không đủ để chứa mã lệnh của chương trình chính. Bởi vậy ở đây chỉ chứa mã của lệnh gọi tới hàm main. Tương tự như vậy, vùng nhớ dành cho High-Priority Interrupt Vector và Low-Priority Interrupt Vector thường cũng không đủ để viết các lệnh của CTCPVN có mức ưu tiên cao và CTCPVN có mức ưu tiên thấp nên ở đây chi chứa mã của lệnh gọi tới các hàm high_isr và low_isr.

3.5.2. Viết các lệnh của chương trình chính.

a. Khởi tạo ngắt

Ngoài viết các lệnh khởi tạo giá trị thích hợp cho ADCON1 và TRISB, để khởi tạo ngắt cần tác động đến các bit GIE, INTxIE và INTEDGx. Khi xảy ra ngắt, bit cờ ngắt tương ứng (INTxIF) sẽ được đặt bằng 1. Để biết các bit GIE, INTxIE,

INTEDGx và INTxIF thuộc thanh ghi nào cần nhớ các từ khóa GIE, INTxIE, INTEDGx và INTxIF (x=0, 1 hoặc 2) và tra cứu trong "Tài liệu tra cứu PIC" hoặc tài liệu "PIC18F4520 Data Sheet". Hình dưới minh họa cách sử dụng tài liệu "PIC18F4520 Data Sheet" để tra cứu bit INT0IE.

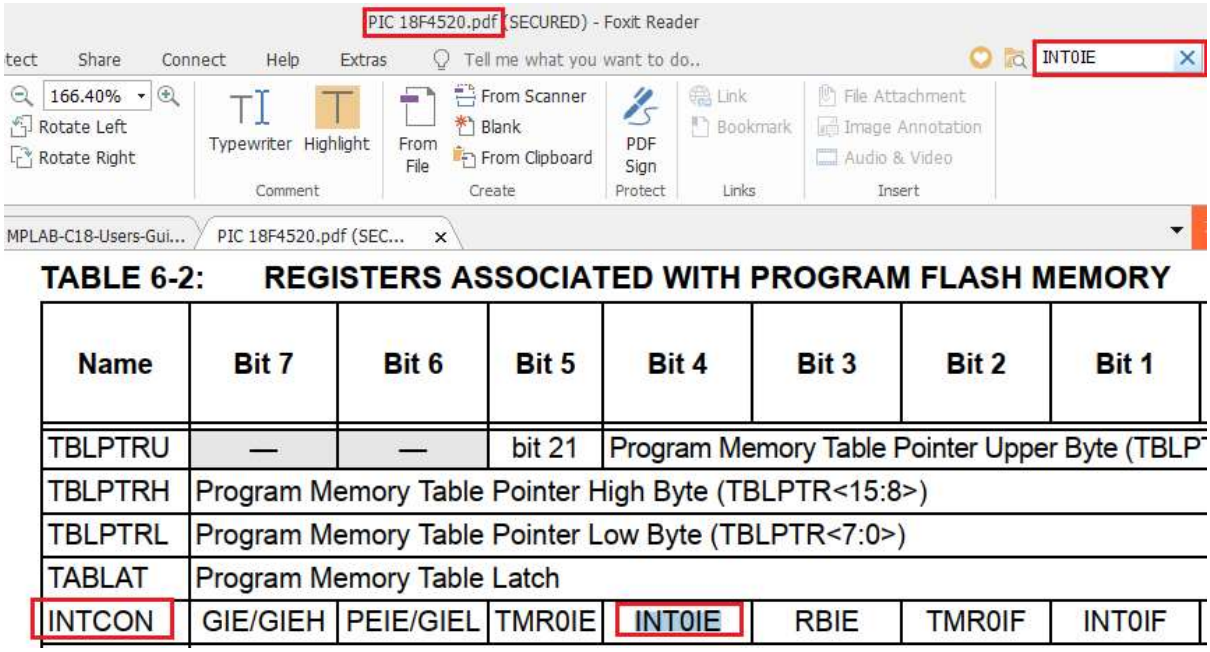


TABLE 6-2: REGISTERS ASSOCIATED WITH PROGRAM FLASH MEMORY

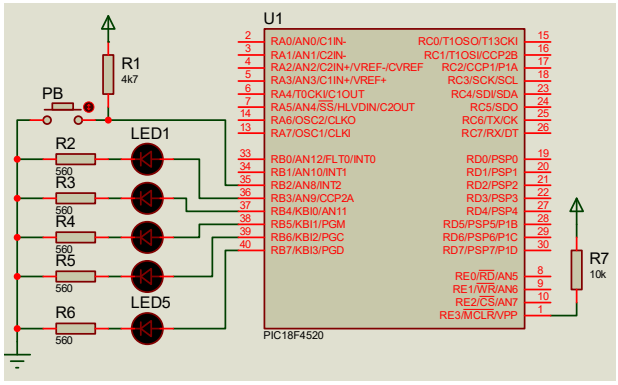
Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
TBLPTRU	—	—	bit 21	Program Memory Table Pointer Upper Byte (TBLP			
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)						
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)						
TABLAT	Program Memory Table Latch						
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF

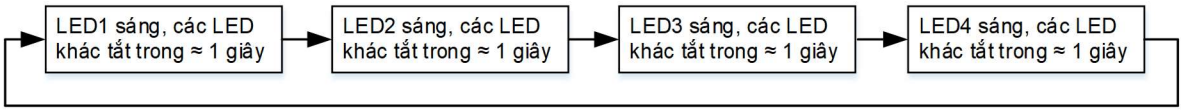
b. Các lệnh theo đề bài

Các lệnh được viết ở chương trình chính sẽ là "các công việc tuần tự, lặp lại". Ví dụ với yêu bài toán sau:

Giả thiết có mạch điện mô phỏng như hình bên. Viết chương trình điều khiển:

- Các LED1-4 sáng tuần tự theo chu trình như hình dưới;
- LED5 sáng trong thời gian ~1s khi nhấn PB.





Các lệnh được viết ở chương trình chính sẽ là "điều khiển các LED1-4 sáng tuần tự theo chu trình"

3.5.3. Viết các lệnh của CTCPVN.

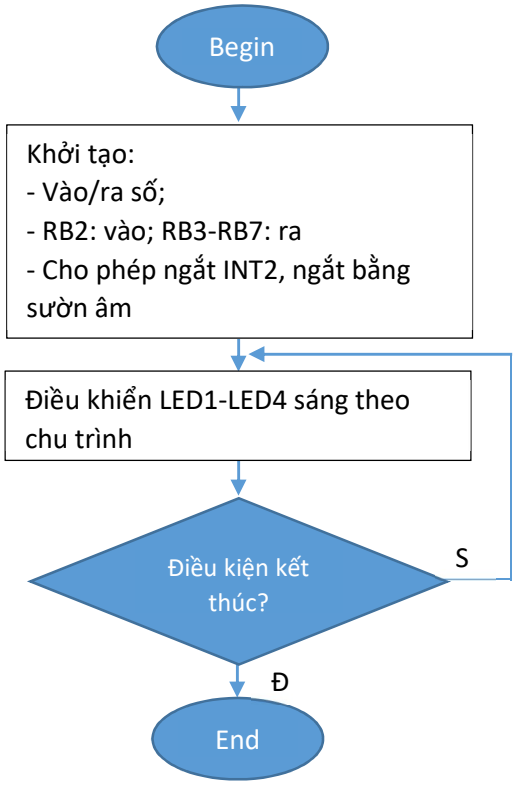
Các lệnh được viết ở CTCPVN sẽ là "các công việc được thực hiện khi xảy ra ngắt".

Ví dụ với bài toán trên, nếu ngắt INT2 được khởi tạo, khi nhấn PB sẽ gây ra ngắt. Vì vậy, "điều khiển LED5 sáng trong thời gian ~1s" sẽ được đặt ở CTCPVN.

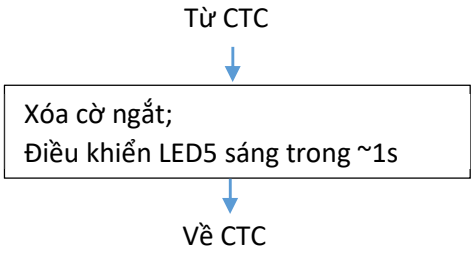
3.6. Lập trình, mô phỏng hoạt động của ngắt ngoài

Từ các phân tích nêu trong mục 3.5, thuật toán của bài toán được trình bày như sau:

CTC



CTCPVN



Thể hiện bằng câu lệnh:

Viết ở CTC:

Khởi tạo:
- Vào/ra số;
- RB2: vào; RB3-RB7: ra
- Cho phép ngắt INT2, ngắt bằng sườn âm

```
ADCON1=0x0F;
TRISB=0b00000111;
INTCONbits. GIE=1;
INTCON3bits.INT2IE=1;
INTCON2bits.INTEDG2=0;
```

Điều khiển LED1-LED4 sáng theo chu trình

```
while(1)
{
PORTB=0b0000100; //LED1 sáng
DelayKTCYx(100); // trễ
PORTB=0b0001000; //LED2 sáng
DelayKTCYx(100); // trễ
PORTB=0b0010000; //LED3 sáng
DelayKTCYx(100); // trễ
PORTB=0b0100000; //LED4 sáng
DelayKTCYx(100); // trễ
}
```

Viết ở CTCPVN:

Xóa cờ ngắt;
Điều khiển LED5 sáng trong ~1s

```
INTCON3bits.INT2IF=0;
LED5=1;
Delay10KTCYx(100);
LED5=0;
```