Teknoloji Fakültesi Elektrik Elektronik Mühendisliği Bölümü

EE-302
Mikroişlemciler

Kesme (interrupt) işlemleri ve Kesme alt döngüleri

5. Hafta

Prof. Dr. Mehmet DEMİRTAŞ

Kesme (interrupt) işlemi mantığı

Mikrodenetleyicinin değişik kaynaklardan gelen sinyaller ile mevcut programının çalışmasını kesip, önceden tanımlanmış kesme programını (fonksiyonunu) icra edip, tekrar ana programa kaldığı yerden çalışmaya devam etmesi işlemini **kesme** olarak tanımlayabiliriz. Mikro işlemcili ve mikrodenetleyicili sistemlerde kesmeler olmazsa olmaz niteliktedir. Peki, neden kesmeler bu kadar önemlidir.

Tasarlanan sistemlerde yapılan rutin kontrol işlemlerinin yanında çoğunlukla bazı giriş/çıkış uçları da denetlenmektedir -taranmaktadır-, Bu denetleme işlemleri sadece program vasıtası ile yapıldığında programın çalışma süresinin ve hızının neredeyse büyük bir kısmını işgal etmektedir. Bu kontrol veya tarama işlemi programda icra edilmesi gereken diğer islerin yavaş yapılmasına veya kontrol edilmesi gereken diğer giriş/çıkış uçlarında veri kaybına yol açılmasına sebep olabilir.

Örneğin bir sistemde, normal rutin programını icra eden bir programın ayni anda sistemde bulunan klavyeden basılan tuş değerini taradığını ve de başka port girişlerine bağlı buton girişlerini de taradığını düşünelim. Bu isleri tümüyle program vasıtası ile yaptığımızda klavye girişleri taranırken, buton girişleri taranamamaktadır veya her iki tarama işlemi yapılırken program diğer komutlarını icra edememektedir. Yani program ile sadece bir iş kontrol edilebilmektedir. Program vasıtası ile yapılan, belirli zaman aralıklarında sürekli olarak kontrol edilen bu isleme **yoklama** denilir.

Kesme (interrupt) işlemi mantığı

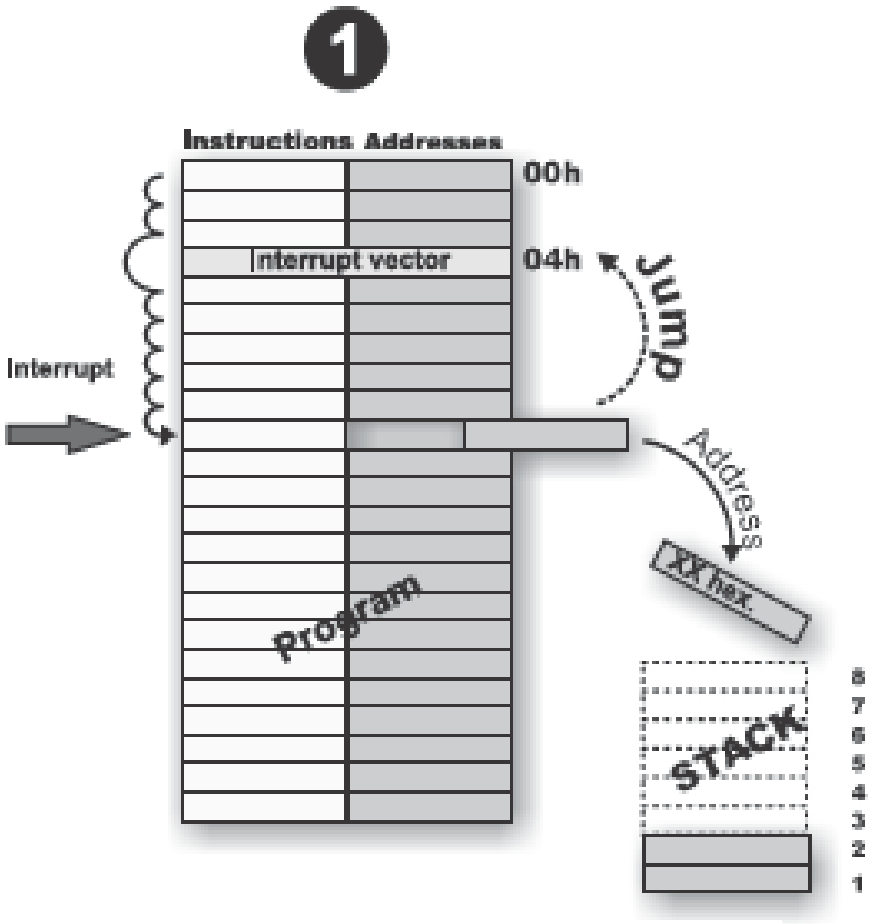
PIC denetleyicilere ilk enerji geldiğinde veya PIC sıfırlandığında (resetlendiğinde) program sayıcı (Program Counter-PC, çalıştırılan komutun adresini tutan kaydedici) 0000h adresini gösterir. Bu adrese **reset vektörü** denir. Bu 0000h adresine PIC'e yüklenen programın başlangıç adresi yüklenir. Böylece denetleyici ilk enerjilendiği anda ilk basta 0000h adresine bakar ve bu adresin içinde bulunan adres değerine giderek programını icra eder.

Yukarıda bahsedilen kesme durumlarından biri oluştuğunda da program sayıcı (PC) program hafızasında 0004h adresine gider. Bu adrese **kesme vektörü** denir. Bu adreste de kesme alt programının (fonksiyonunun) başlangıç adresi vardır. Bu nedenle program yazarken bir kesme olayını programımızda aktif etmişsek, 0004h adresine mutlaka kesme alt programımızın başlangıç adresini yazmamız gerekir.

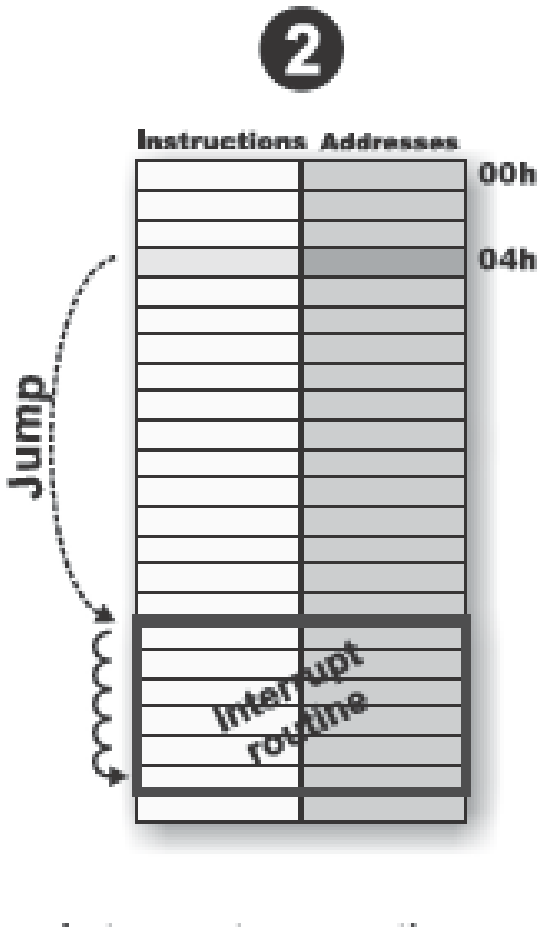
Bu sayede kesme durumu meydana geldiğinde PC, 0004h adresini gösterecek, program 0004h adresine gidecek ve bu adresteki kesme programı adresine dallanacak ve kesme programını icra edecektir. Anlatılan 0000h adresindeki reset vektörü ve 0004h adresindeki kesme vektörü tüm PIC denetleyicilerinde aynı adrestedir.

Kesme (interrupt) işlemi mantığı

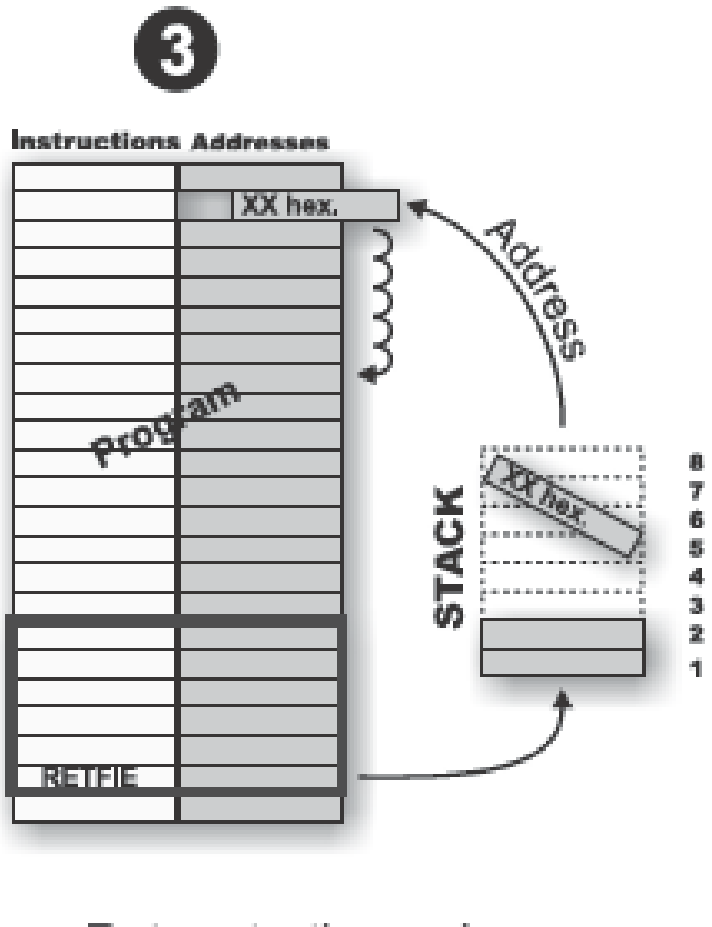
Herhangi bir kesme sinyali geldiğinde ana program çalışması kesiliyor, kesme alt programına gidiliyor, kesme alt programı icra ediliyor ve ana programa geri dönülüyor. **Peki denetleyici kesme sinyali geldiğinde icra etmeyi bıraktığı ana programda kaldığı yeri nasıl biliyor?** Kesme meydana geldiği anda denetleyici içindeki program sayıcı (PC) o anki değerini **yığın (stack)** kaydedicisine atar. Kesme programı dönüsünde yığından kaydedilen adres alınır ve bu adresteki komuttan itibaren ana program icra edilmeye devam eder.



Generation of interrupt



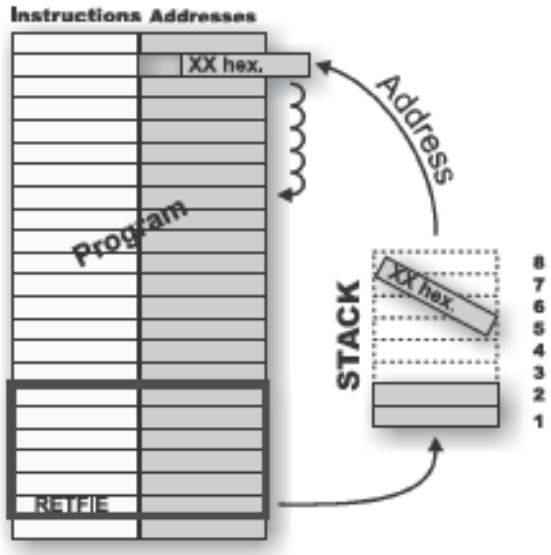
Interrupt execution



Return to the main program

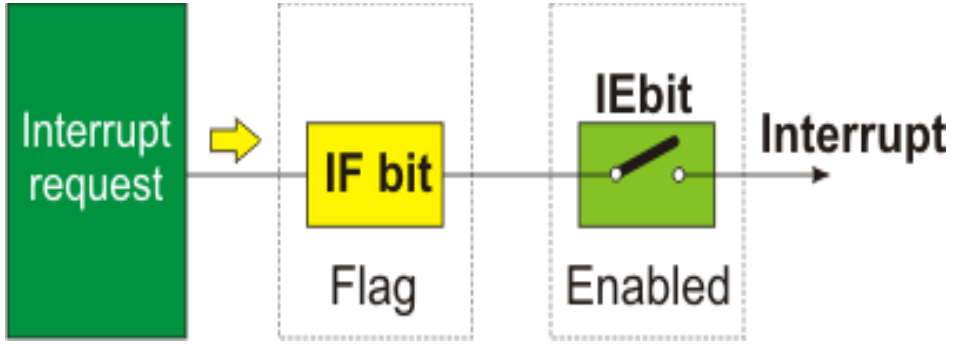
Kesme (interrupt) işlemi mantığı

- Yığın (stack) olarak kullanılan RAM'in bir bölümü sekiz 13 bitlik yazmaçtan oluşur. Mikrodenetleyici bir alt rutini (CALL komutu) yürütmeye başlamadan önce veya bir kesme meydana geldiğinde, çalıştırılacak ilk sonraki komutun adresi yığına, yani kayıtlarından birine itilir. Bu sayede mikro denetleyici, bir alt yordam veya bir kesinti yürütmesi üzerine düzenli program yürütmeye nereden devam edeceğini bilir. Bu adres, programa döndükten sonra silinir, çünkü artık onu kaydetmeye gerek yoktur ve yığının bir konumu daha sonra kullanılmak üzere otomatik olarak kullanılabilir hale gelir.
- Verilerin her zaman yığına dairesel olarak itildiğini unutmamak önemlidir. Bu, yığın sekiz kez itildikten sonra, dokuzuncu itmenin ilk basışta depolanan değerin üzerine yazacağı anlamına gelir. Onuncu itme, ikinci itmenin üzerine yazar ve böyle devam eder. Bu şekilde üzerine yazılan veriler kurtarılamaz.
- Ek olarak, programcı yazma veya okuma için bu kayıtlara erişemez ve yığın taşması veya yığın yetersizliği koşullarını belirtmek için Durum biti yoktur. Bu nedenle program yazımı sırasında özel dikkat gösterilmesi gerekmektedir.



Kesme (interrupt) işlemi mantığı

Bir kesme isteği geldiğinde, bir kesintinin otomatik olarak gerçekleşeceği anlamına gelmez, çünkü kullanıcı tarafından da etkinleştirilmesi gerekir (programın içinden). Bu nedenle, kesintileri etkinleştirmek veya devre dışı bırakmak için kullanılan özel bitler vardır. Adlarında bulunan IE harflerinden onları tanımak kolaydır (Kesme Etkinleştirmenin «interrup enable» kısaltmasıdır). Ayrıca, her kesme, etkinleştirilip etkinleştirilmediğine bakılmaksızın bir kesme isteğinin geldiğini belirten bayrak adı verilen başka bir bit ile ilişkilendirilir. Ayrıca adlarında bulunan son iki harften de kolayca tanınabilirler - IF (Interrupt Flag).



Kesme kaynakları

Denetleyicide bir kesme meydana geldiğinde o kesmeye ait bayrak (flag) "1" olur. Anlamı kolaylaştırmak açısından kesme meydana geldiğinde kesmenin bayrağı kalkar diyebiliriz. Biz hangi kesme bayrağı kalkmış yani "1" olmuşsa o kesmenin oluştuğunu anlarız. PIC 16f877A denetleyicilerde 14 adet kesme mevcuttur. Bu kesmeler şunlardır.

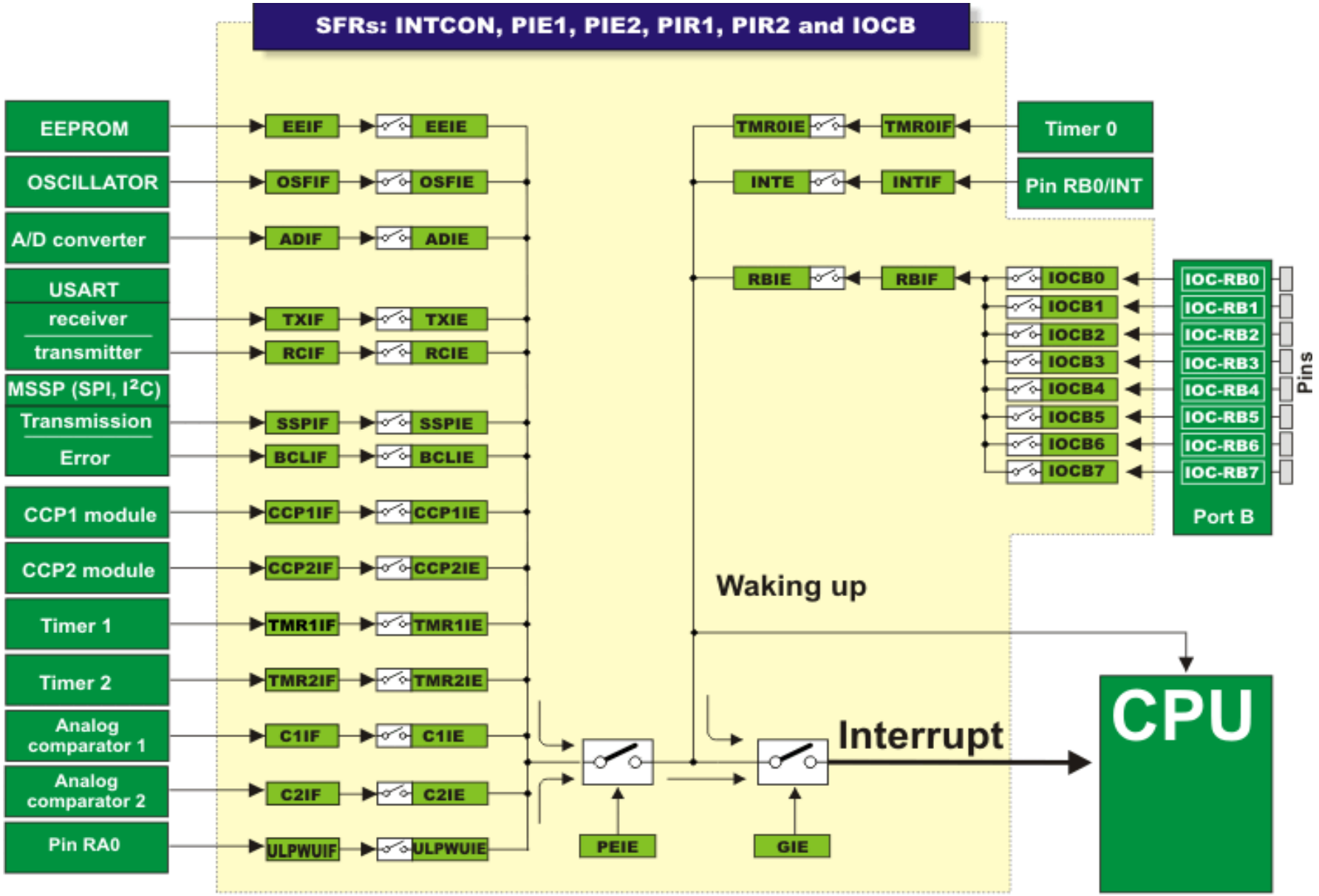
1. RB0 Harici Kesmesi.
2. RB4-RB7 Pin'lerindeki Değişiklik Kesmesi.
3. Timer0 Birimi Tasma Kesmesi.
4. Timer1 Birimi Tasma Kesmesi.
5. Timer2 Birimi Tasma Kesmesi.
6. A/D Çevrimi Yapıldığında Meydana Gelen Kesme.
7. CCP1 Modülü Kesmesi.
8. CCP2 Modülü Kesmesi.
9. Paralel Port'tan Veri Gelme Kesmesi.
10. Seri Port'tan Veri Geldiğinde Oluşan Kesme.
11. SPI veya I2C iletişimi Sırasında Veri Gelme Kesmesi.
12. EEPROM'a Veri Yazma İşlemi Sonlandığında Oluşan Kesme.
13. RS232 Seri İletişiminde Gönderilecek Veri Tamponunu Bos Olduğunda Meydana Gelen Kesme.
14. Bus Collision Kesmesi (MSSP modunda seri iletişimde hata oluştuğunda).

SFRs bank 0

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
00h	INDF	Indirect register							
01h	TMR0	Timer T0 Register							
02h	PCL	Least Significant Byte of Program Counter							
03h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C
04h	FSR	Indirect Data Memory Address Pointer							
05h	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
07h	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0
08h	PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0
09h	PORTE	-	-	-	-	RE3	RE2	RE1	RE0
0Ah	PCLATH	-	-	-	Upper 5 bits of Program Counter				
0Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
0Ch	PIR1	-	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
0Dh	PIR2	OSFIF	C2IF	C1IF	EEIF	BCLIF	ULPWUIF	-	CCP2IF
0Eh	TMR1L	Least Significant Byte of the 16-bit Timer TMR0							
0Fh	TMR1H	Most Significant Byte of the 16-bit Timer TMR0							
10h	T1CON	T1GINV	TMR1GE	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON
11h	TMR2	Timer T2 Register							
12h	T2CON	-	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register							
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
15h	CCPR1L	Capture/ComparePWM Register 1 Low Byte (LSB)							
16h	CCPR1H	Capture/ComparePWM Register 1 High Byte (LSB)							
17h	CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
19h	TXREG	EUSART Transmit Data Register							
1Ah	RCREG	EUSART Receive Data Register							
1Bh	CCPR2L	Capture/Compare PWM Register 1 Low Byte (LSB)							
1Ch	CCPR2H	Capture/Compare PWM Register 1 High Byte (LSB)							
1Dh	CCP2CON	-	-	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0
1Eh	ADRESH	A/D Result Register High Byte							
1Fh	ADCON0	ADCS1	ADCS0	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON

SFRs bank 1

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
80h	INDF	Indirect Register							
81h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
82h	PCL	Least Significant Byte of Program Counter							
83h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C
84h	FSR	Indirect Data Memory Address Pointer							
85h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
86h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
87h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
88h	TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0
89h	TRISE	-	-	-	-	TRISE3	TRISE2	TRISE1	TRISE0
8Ah	PCLATH	-	-	-	Upper 5 bits of the Ptogram Counter				
8Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
8Ch	PIE1	-	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
8Dh	PIE2	OSFIE	C2IE	C1IE	EEIE	BCLIE	ULPWUIE	-	CCP2IE
8Eh	PCON	-	-	ULPWUE	SBOREN	-	-	POR	BOR
8Fh	OSCCON	-	IRCF2	IRCF1	IRCF0	OSTS	HTS	LTS	SCS
90h	OSCTUNE	-	-	-	TUN4	TUN3	TUN2	TUN1	TUN0
91h	SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
92h	PR2	Timer T2 Period Register							
93h	SSPADD	Synchronous Serial Port (I^2C mode) Address Register							
93h	SSPMSK	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
94h	SSPSTAT	SMP	CKE	D/A	P	S	R/W	UA	BF
95h	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0
96h	IOCB	IOCB7	IOCB6	IOCB5	IOCB4	IOCB3	IOCB2	IOCB1	IOCB0
97h	VRCON	VREN	VROE	VRR	VRSS	VR3	VR2	VR1	VR0
98h	TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
99h	SPBRG	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0
9Ah	SPBRGH	BRG15	BRG14	BRG13	BRG12	BRG11	BRG10	BRG9	BRG8
9Bh	PWM1CON	PRSEN	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
9Ch	ECCPAS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0
9Dh	PSTRCON	-	-	-	STRSYNC	STRD	STRC	STRB	STRA
9Eh	ADRESL	A/D Result Register Low Byte							
9Fh	ADCON1	ADFM	-	VCFG1	VCFG0	-	-	-	-


SFRs: INTCON, PIE1, PIE2, PIR1, PIR2 and IOCB
EEPROM
OSCILLATOR
A/D converter
USART
receiver
transmitter
MSSP (SPI, I²C)
Transmission
Error
CCP1 module
CCP2 module
Timer 1
Timer 2
Analog comparator 1
Analog comparator 2
Pin RA0
EEIF
EEIE
OSFIF
OSFIE
ADIF
ADIE
TXIF
TXIE
RCIF
RCIE
SSPIF
SSPIE
BCLIF
BCLIE
CCP1IF
CCP1IE
CCP2IF
CCP2IE
TMR1IF
TMR1IE
TMR2IF
TMR2IE
C1IF
C1IE
C2IF
C2IE
ULPWUIF
ULPWUIE
TMR0IE
TMR0IF
Timer 0
INTE
INTF
Pin RB0/INT
RBIE
RBIF
IOCB0
IOCB1
IOCB2
IOCB3
IOCB4
IOCB5
IOCB6
IOCB7
IOC-RB0
IOC-RB1
IOC-RB2
IOC-RB3
IOC-RB4
IOC-RB5
IOC-RB6
IOC-RB7
Pins
Port B
Waking up
PEIE
GIE
Interrupt
CPU

Kesme (interrupt) işlemi mantığı

Bu kesmelerden ilk 3 kesme olan RB0 harici kesmesi, RB4-RB7 değişim kesmesi ve Timer0 taşma kesmeleri INTCON ve OPTION kaydedicileri ile kontrol edilir. Diğer 11 kesme ise çevresel kesme (Peripheral Interrupt) kaynaklarını oluşturur. çevresel kesme kaynaklarını kontrol eden kesme kaydedicileri ise PIE1, PIR1, PIE2 ve PIR2 kaydedicileridir. Bu kaydedicilerin içerikleri aşağıda sırayla açıklanmıştır. Bu çevresel kesme kaynakları kaydedicileri ile, kesmelerin aktif ve pasif yapılması kontrolü sağlanır ve kesme sinyali geldiğinde kesmenin hangi kesme olduğunu bize bildirir. PIE1 ve PIE2 kaydedicileri çevresel kesmeleri aktif etmek için, PIR1 ve PIR2 kaydedicileri ise kesme bayraklarını içerir. Yani oluşan kesmenin hangi kesme olduğunu bu bayrakları kontrol ederek anlarız.

INTCON Register_1

INTCON register TMR0 register taşması, PORTB değişikliği ve harici INT pin kesmeleri için çeşitli etkinleştirme ve bayrak bitleri içerir.

	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (x)	Features
INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Legend: R/W - Readable/Writable Bit, (0) After reset, bit is cleared, (X) After reset, bit is unknown

- **GIE - Global Interrupt Enable bit** - controls all possible interrupt sources simultaneously.
  - **1** - Enables all unmasked interrupts.
  - **0** - Disables all interrupts.
- **PEIE - Peripheral Interrupt Enable bit** acts similar to the GIE it, but controls interrupts enabled by peripherals.
  It means that it has no impact on interrupts triggered by the timer TMR0 or by changing the state of PORTB or the RB0/INT pin.
  - **1** - Enables all unmasked peripheral interrupts.
  - **0** - Disables all peripheral interrupts.
- **T0IE - TMR0 Overflow Interrupt Enable bit** controls interrupt enabled by TMR0 overflow.
  - **1** - Enables the TMR0 interrupt.
  - **0** - Disables the TMR0 interrupt.
- **INTE - RB0/INT External Interrupt Enable bit** controls interrupt caused by changing the logic state of the RB0/INT input pin (external interrupt).
  - **1** - Enables the INT external interrupt.
  - **0** - Disables the INT external interrupt.

INTCON Register_2

INTCON register TMR0 register taşması, PORTB değişikliği ve harici INT pin kesmeleri için çeşitli etkinleştirme ve bayrak bitleri içerir.

	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (x)	Features
INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

- **RBIE - RB Port Change Interrupt Enable bit**. When configured as inputs, PORTB pins may cause an interrupt by changing their logic state (no matter whether it is high-to-low transition or vice versa, the fact that something is changed only matters). This bit determines whether an interrupt is to occur or not.
  - **1** - Enables the port B change interrupt.
  - **0** - Disables the port B change interrupt.
- **T0IF - TMR0 Overflow Interrupt Flag bit** registers the timer TMR0 register overflow, when counting starts at zero.
  - **1** - TMR0 register has overflowed (bit must be cleared from within the software).
  - **0** - TMR0 register has not overflowed.
- **INTF - RB0/INT External Interrupt Flag bit** registers the change of the RB0/INT pin logic state.
  - **1** - The INT external interrupt has occurred (must be cleared from within the software).
  - **0** - The INT external interrupt has not occurred.
- **RBIF - RB Port Change Interrupt Flag bit** registers any change of logic state of some PORTB input pins.
  - **1** - At least one of the PORTB general purpose I/O pins has changed state. Upon reading PORTB, the RBIF bit must be cleared from within the software.
  - **0** - None of the PORTB general purpose I/O pins has changed the state.

PIE1 Register

The PIE1 register contains peripheral interrupt enable bits.

	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	Features
PIE1	-	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Legend: (-) Unimplemented bit, (R/W) - Readable/Writable Bit, (0) After reset, bit is cleared

- **ADIE - A/D Converter Interrupt Enable bit.**
  - **1** - Enables the ADC interrupt.
  - **0** - Disables the ADC interrupt.
- **RCIE - USART Receive Interrupt Enable bit.**
  - **1** - Enables the EUSART receive interrupt.
  - **0** - Disables the EUSART receive interrupt.
- **TXIE - USART Transmit Interrupt Enable bit.**
  - **1** - Enables the EUSART transmit interrupt.
  - **0** - Disables the EUSART transmit interrupt.
- **SSPIE - Master Synchronous Serial Port (MSSP) Interrupt Enable bit** - enables an interrupt request to be generated upon each data transmission via synchronous serial communication module (SPI or I2C mode).
  - **1** - Enables the MSSP interrupt.
  - **0** - Disables the MSSP interrupt.
- **CCP1IE - CCP1 Interrupt Enable bit** enables an interrupt request to be generated in CCP1 module used for PWM signal processing.
  - **1** - Enables the CCP1 interrupt.
  - **0** - Disables the CCP1 interrupt.
- **TMR2IE - TMR2 to PR2 Match Interrupt Enable bit**
  - **1** - Enables the TMR2 to PR2 match interrupt.
  - **0** - Disables the TMR2 to PR2 match interrupt.
- **TMR1IE - TMR1 Overflow Interrupt Enable bit** enables an interrupt request to be generated upon each timer TMR1 register overflow, i.e. when the counting starts from zer
  - **1** - Enables the TMR1 overflow interrupt.
  - **0** - Disables the TMR1 overflow interrupt.

PIE2 Register

The PIE2 Register also contains various interrupt enable bits.

	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)		R/W (0)	Features
PIE2	OSFIE	C2IE	C1IE	EEIE	BCLIE	ULPWUIE	-	CCP2IE	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Legend: (-) Unimplemented bit, (R/W) - Readable/Writable Bit, (0) After reset, bit is cleared

- **OSFIE - Oscillator Fail Interrupt Enable bit.**
  - **1** - Enables oscillator fail interrupt.
  - **0** - Disables oscillator fail interrupt.
- **C2IE - Comparator C2 Interrupt Enable bit.**
  - **1** - Enables Comparator C2 interrupt.
  - **0** - Disables Comparator C2 interrupt.
- **C1IE - Comparator C1 Interrupt Enable bit.**
  - **1** - Enables Comparator C1 interrupt.
  - **0** - Disables Comparator C1 interrupt.
- **EEIE - EEPROM Write Operation Interrupt Enable bit.**
  - **1** - Enables EEPROM write operation interrupt.
  - **0** - Disables EEPROM write operation interrupt.
- **BCLIE - Bus Collision Interrupt Enable bit.**
  - **1** - Enables bus collision interrupt.
  - **0** - Disables bus collision interrupt.
- **ULPWUIE - Ultra Low-Power Wake-up Interrupt Enable bit.**
  - **1** - Enables Ultra Low-Power Wake-up interrupt.
  - **0** - Disables Ultra Low-Power Wake-up interrupt.
- **CCP2IE - CCP2 Interrupt Enable bit.**
  - **1** - Enables CCP2 interrupt.
  - **0** - Disables CCP2 interrupt.

PIR1 Register

The PIR1 register contains the interrupt flag bits.

	R/W (0)	R (0)	R (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	Features
PIR1	-	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Legend: (-) Unimplemented bit, (R/W) - Readable/Writable Bit, (R) - Readable Bit, (0) After reset, bit is cleared

- **ADIF - A/D Converter Interrupt Flag bit.**
  - **1** - A/D conversion is completed (bit must be cleared from within the software).
  - **0** - A/D conversion is not completed or has not started.
- **RCIF - EUSART Receive Interrupt Flag bit.**
  - **1** - The EUSART receive buffer is full. Bit is cleared by reading the RCREG register.
  - **0** - The EUSART receive buffer is not full.
- **TXIF - EUSART Transmit Interrupt Flag bit.**
  - **1** - The EUSART transmit buffer is empty. The bit is cleared by any write to the TXREG register.
  - **0** - The EUSART transmit buffer is full.
- **SSPIF - Master Synchronous Serial Port (MSSP) Interrupt Flag bit.**
  - **1** - The MSSP interrupt conditions during data transmit/receive have occurred. They differ depending on MSSP operating mode (SPI or I^2C). This bit must be cleared from within the software before returning from the interrupt service routine.
  - **0** - No MSSP interrupt condition has occurred.
- **CCP1IF - CCP1 Interrupt Flag bit.**
  - **1** - CCP1 interrupt condition has occurred (CCP1 is unit for capturing, comparing and generating PWM signal). Depending on operating mode, capture or compare match has occurred. In both cases, bit must be cleared in software. This bit is not used in PWM mode.
  - **0** - No CCP1 interrupt condition has occurred.
- **TMR2IF - Timer2 to PR2 Interrupt Flag bit**
  - **1** - TMR2 (8-bit register) to PR2 match has occurred. This bit must be cleared from within the software prior to returning from the interrupt service routine.
  - **0** - No TMR2 to PR2 match has occurred.
- **TMR1IF - Timer1 Overflow Interrupt Flag bit**
  - **1** - The TMR1 register has overflowed. This bit must be cleared from within the software.
  - **0** - The TMR1 register has not overflowed.

PIR2 Register

The PIR2 register contains the interrupt flag bits.

	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)		R/W (0)	Features
PIR2	OSFIF	C2IF	C1IF	EEIF	BCLIF	ULPWUIF	-	CCP2IF	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Legend: (-) Unimplemented bit, (R/W) - Readable/Writable Bit, (0) After reset, bit is cleared

- **OSFIF - Oscillator Fail Interrupt Flag bit.**
  - **1** - System oscillator failed and clock input has changed to internal oscillator INTOSC. This bit must be cleared from within the software.
  - **0** - System oscillator operates normally.
- **C2IF - Comparator C2 Interrupt Flag bit.**
  - **1** - Comparator C2 output has changed (bit C2OUT). This bit must be cleared from within the software.
  - **0** - Comparator C2 output has not changed.
- **C1IF - Comparator C1 Interrupt Flag bit.**
  - **1** - Comparator C1 output has changed (bit C1OUT). This bit must be cleared from within the software.
  - **0** - Comparator C1 output has not changed.
- **EEIF - EE Write Operation Interrupt Flag bit.**
  - **1** - EEPROM write complete. This bit must be cleared from within the software.
  - **0** - EEPROM write is not complete or has not started yet.
- **BCLIF - Bus Collision Interrupt Flag bit.**
  - **1** - A bus collision has occurred in the MSSP when configured for I2C Master mode. This bit must be cleared from within the software.
  - **0** - No bus collision has occurred.
- **ULPWUIF - Ultra Low-power Wake-up Interrupt Flag bit.**
  - **1** - Wake-up condition has occurred. This bit must be cleared from within the software.
  - **0** - No Wake-up condition has occurred.
- **CCP2IF - CCP2 Interrupt Flag bit.**
  - **1** - CCP2 interrupt condition has occurred (unit for capturing, comparing and generating PWM signal). Depending on operating mode, capture or compare match has occurred. In both cases, the bit must be cleared from within the software. This bit is not used in PWM mode.
  - **0** - No CCP2 interrupt condition has occurred.

CCS C'DE KESME OLUSTURMA ISLEMLERI

CCS C'de bir kesme komutu kullanıldığında ilgili kesme meydana geldiğinde işlenecek komutları içinde barındıran bir kesme fonksiyonunun tanımlanması gereklidir. CCS C'de kullanılacak kesme fonksiyonları mutlaka ana fonksiyondan önce tanımlanmalıdır. Kesme fonksiyonu tanımlanırken aşağıda verilen yapı kullanılır.

int_xxx / / xxx yerine ilgili kesme ismi yazılır.

[fonksiyon geri dönüş değeri türü] [fonksiyon ismi] ( )

{

Komut veya komutlar;

}

Örnek bir kesme fonksiyonu;

```
#int_ext

void dis_kesme ()
{
output_low(pin_dl) ;
x++;
}
```

#int_xxx komutunda aktif edilen fonksiyon ismi yazılır. Bu kesme isimleri kullanılan denetleyicinin tanıtım dosyasında (örneğin 16f877.h dosyasının içinde) tanımlanmıştır. PIC16F877 için aşağıda verilen kesme komutları kullanılır.

#INT _EEPROM ---->EEPROM yazma işlemi bitince meydana gelen kesme.
#INT _EXT ---->Dış kesme.
#INT _I2C ---->I2C Kesmesi.
#INT _LOWVOLT ---->Düşük voltaj tespit kesmesi.
#INT _RB ---->B Portunun B4-B7 pin'lerinde herhangi bir değişiklik olduğunda meydana gelen kesme.
#INT _RC ---->C Port'unun C4-C7 pin'lerinde herhangi bir değişiklik olduğunda meydana gelen kesme.
#INT _RDA ---->RS232 data alma kesmesi.
#INT _RTCC ---->Timer0 kesmesi.
#INT _SSP ---->SPI veya I2C faaliyette kesmesi.
#INT _TBE RS232 ---->gönderme tamponu bos kesmesi.
#INT_TIMER0 ---->Timer O kesmesi.
#INT _TIMER1 ---->Timer 1 kesmesi.
#INT _TIMER2 ---->Timer 2 kesmesi.
#INT _ CCP1 ---->CCP1 kesmesi.
#INT _CCP2 ---->CCP2 kesmesi.

Kesmelerin aktif veya pasif yapılması için aşağıdaki komutlar kullanılır;

***#enable_interrupt(kesme ismi); //* Kesmeyi aktif yapmak için**
#disable_interrupt(kesme *ismi); //* Kesmeyi pasif yapmak için.

#enable_interrupt(int_ext); *//* RBO/INT dis kesmesi aktif.
#disable_interrupt(int_ext); *//* RBO/INT dis kesmesi pasif.

PIC denetleyicilerde kesmeleri aktif etmek yetmez. INTCON kaydedicisinin 7.bit'i olan GIE bitti aktif edilen tüm kesmelere izin vermek veya vermemek için kullanılır. CCS C'de aşağıdaki komutlarla tüm aktif edilmiş kesmelere izin verilir veya verilmez.

#enable_interrupt(GLOBAL); //Aktif edilen kesmelere izin verilir.
#disable_interrupt(GLOBAL); // Aktif edilen kesmelere izin verilmez.

Bir kesme meydana geldiğinde o kesme fonksiyonu icra edilirken başka bir kesme meydana gelse dahi aktif olamaz. İlk basta icra edilen kesme fonksiyonunun bitmesi gereklidir. Fakat aynı anda meydana gelen kesmeler de hangi kesmenin geçerli olacağını belirlemek için kesmelere öncelik verilmelidir. Öncelik vermek için aşağıdaki komut kullanılır. İlk başa yazılan kesme en fazla önceliğe sahiptir demektir.
#priority *kesme ismi, kesme ismi,...*
#priority ext, timer0 // dış kesme, timer0 kesmesinden daha öncelikli

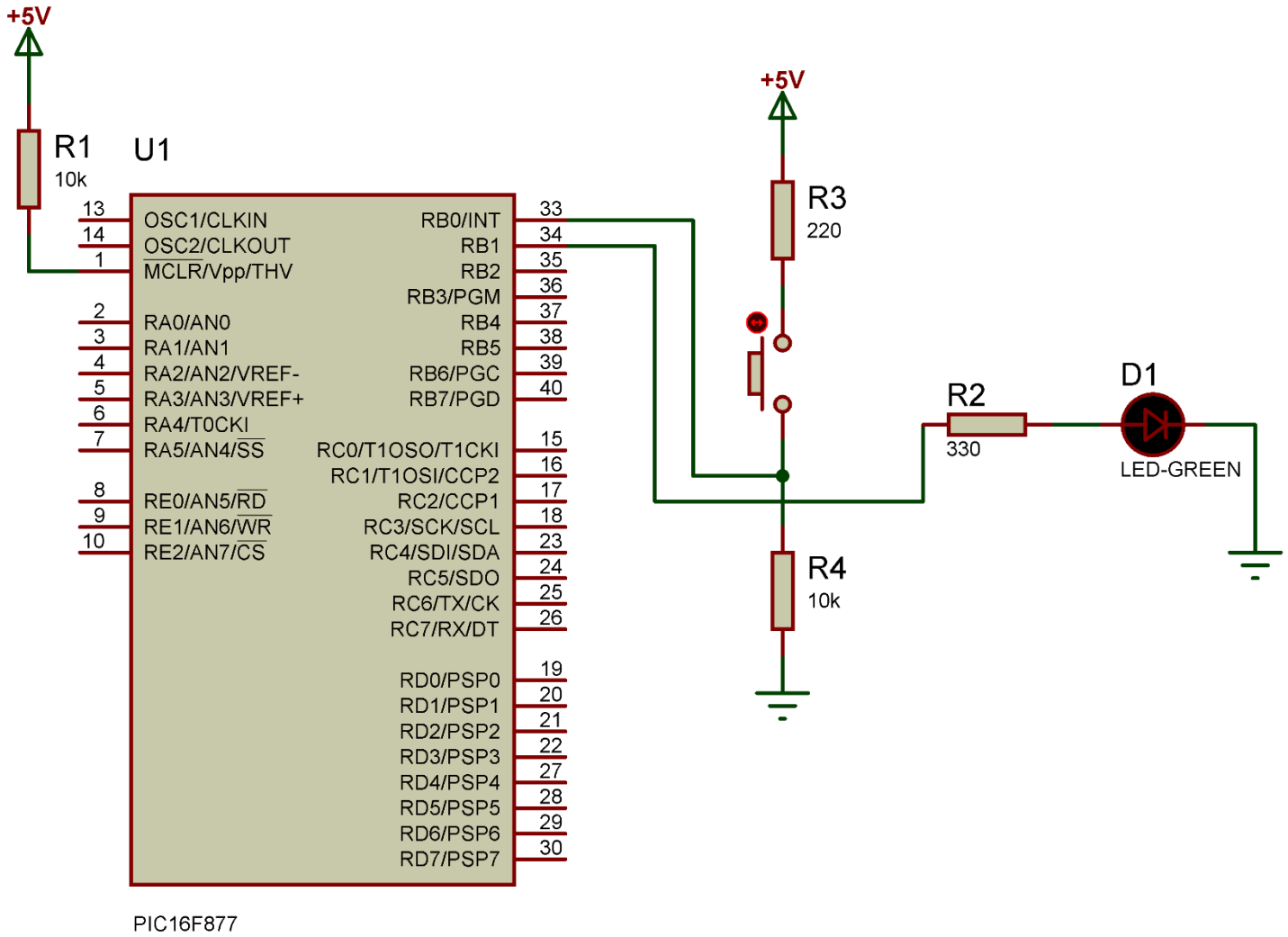
PIC denetleyicilerde bir kesme meydana geldiğinde o kesmeye ait kesme bayrağı bit'i lojik-1 olur. Bunun nedeni kesme komutları işlenirken yine aynı kesme meydana gelirse kesme komutları icra edilirken kesmeden çıkılmamasıdır. Kesme komutlarının sonunda bu bayrak bit'inin lojik-0 yapılması gerekir. Eğer bu bit lojik-0 yapılmazsa, ayni kesme şartı oluşsa da dahi ikinci kez kesme meydana gelmez.

CCS C derleyicisinde ise herhangi bir kesme meydana geldiğinde, CCS C programı kesme fonksiyonu çıkısında otomatik olarak kesme bayrağını siler. Kullanıcının bu işlemi komut ile yapmasına gerek kalmaz. Fakat yine de istendiğinde kesme bayrağını silmek için aşağıdaki komut kullanılır.

clear_interrupt(kesme ismi);

clear_interrupt(int_timer0); // Timer0 kesmesi bayrağı lojik-0 yapılır.

Örnek Program_1 (Dış kesme uygulaması)

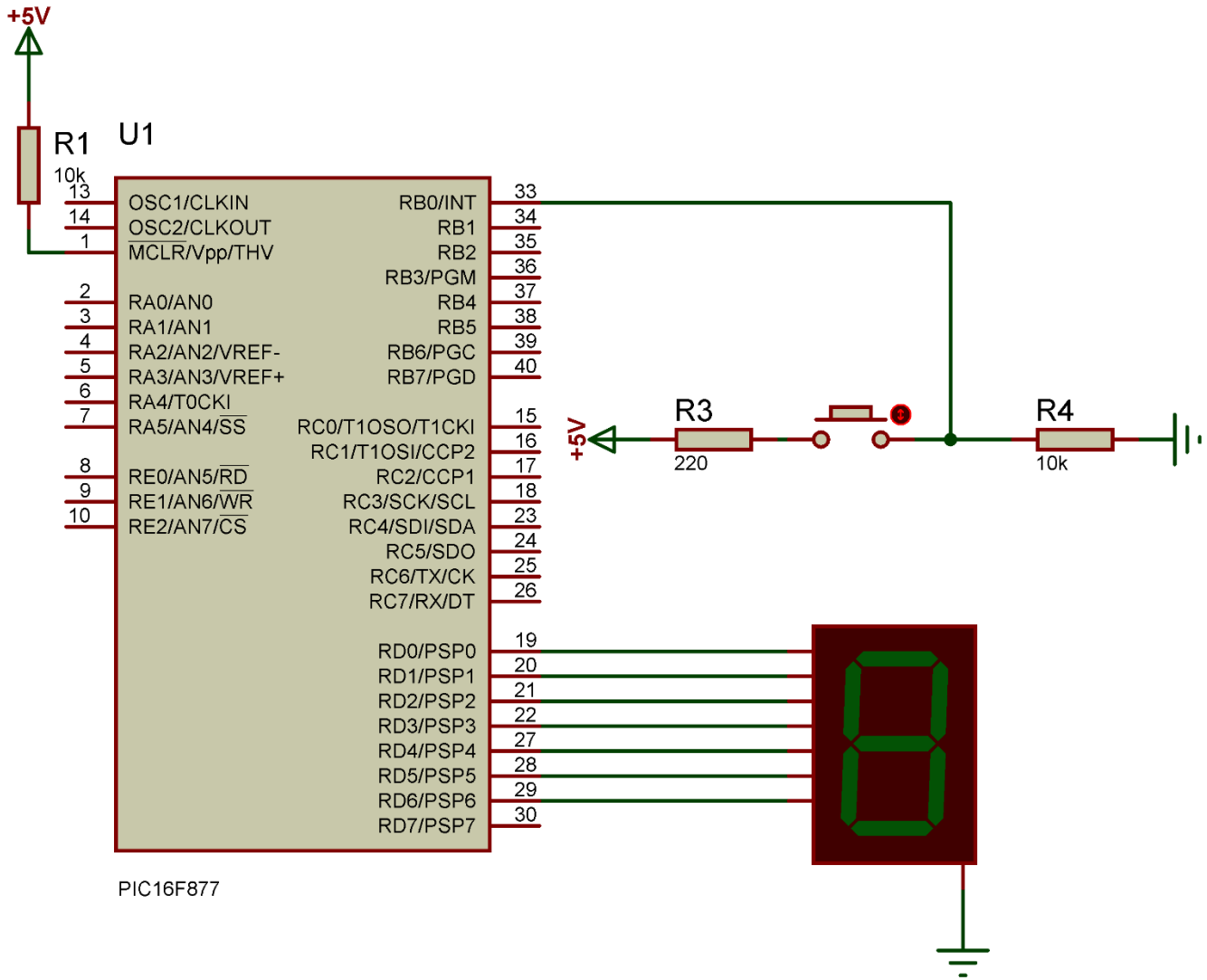


Örnek Program_1 (Dış kesme uygulaması)

```
#include <16f877.h>
#fuses XT,NOWDT,NOPROTECT
#use delay (clock=4000000)
#use fast_io(b)
int i; // Tamsayı tipinde değişken tanımlanıyor
//****** Dış Kesme Fonksiyonu *****************
#int_ext              // Dış(External)RB0/INTkesmesi
void  ext_kesmesi () // Dış kesme fonksiyonu
{
   output_high(pin_b1);
   delay_ms(1000);
   output_low(pin_b1);
   delay_ms(3000);

   for (i=0;i<10;i++)
   {
      output_high(pin_b1);
      delay_ms(500);
      output_low(pin_b1);
      delay_ms(500);
   }
}
```

```
/********* ANA PROGRAM FONKSİYONU********/
void main ( )
{
    set_tris_b(0x01);
    output_b(0x00);

   ext_int_edge(H_TO_L); // INT_EXT kesmesinin düşen kenarda
                         //aktif olacağını belirtir
   enable_interrupts(INT_EXT);  // INT_EXT kesmesini aktif yapar
   enable_interrupts(GLOBAL);   // Aktif edilen kesmelere izin ver

   while(1);
}
```

Örnek Program_2 (Dış kesme 7 segment display uygulaması)



Örnek Program_2 (Dış kesme 7 segment display uygulaması)

```
#include <16f877.h>
#fuses XT,NOWDT,NOPROTECT
#use delay (clock=4000000)
#use fast_io(b)
#use fast_io(d)
int i,z,y,x; // Tamsayı tipinde değişkenler tanımlanıyor
// Ortak katot display için veri değerleri
const int digit[16]={0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7C, 0x07, 0x7F, 0x6F, 0x77, 0x7C, 0x39, 0x5E, 0x79, 0x71};
const int digit2[7]={0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40};
//****** Dış Kesme Fonksiyonu *****************
#int_ext                // Dış (External) RB0/INT kesmesi
void  ext_kesmesi ()    // Dış kesme fonksiyonu
{
   for (z=0;z<5;z++)
   {
      output_d(0x6D);
      delay_ms(500);
      output_d(0x00);
      delay_ms(500);
   }
```

Örnek Program_2 (Dış kesme 7 segment display uygulaması)

```
21      for(y=0;y<3;y++)
22      {
23      for (x=0;x<6;x++)
24      {
25          output_d(digit2[x]); // digit[i] değerini B portuna gönder
26          delay_ms(50);       // 500 msn bekle
27      }
28      }
29  }
30  /********* ANA PROGRAM FONKSİYONU********/
31  void main ( )
32  {
33      set_tris_b(0x01);    // RB0 pini giriş,diğer uçlar çıkış olarak yönlendiriliyor
34      set_tris_d(0x00);
35      output_b(0x00);      // B portu çıkışı ilk anda sıfırlanıyor
36
37      ext_int_edge(H_TO_L);    // INT_EXT kesmesinin düşen kenarda aktif olacağını belirtir
38
39      enable_interrupts(INT_EXT);  // INT_EXT kesmesini aktif yapar
40      enable_interrupts(GLOBAL);   // Aktif edilen kesmelere izin ver
41
42      while(1)
```

Örnek Program_2 (Dış kesme 7 segment display uygulaması)

```
31  void main ( )
32  {
33     set_tris_b(0x01);    // RB0 pini giriş,diğer uçlar çıkış olarak yönlendiriliyor
34     set_tris_d(0x00);
35     output_b(0x00);      // B portu çıkışı ilk anda sıfırlanıyor
36
37     ext_int_edge(H_TO_L);    // INT_EXT kesmesinin düşen kenarda aktif olacağını belirtir
38
39     enable_interrupts(INT_EXT);  // INT_EXT kesmesini aktif yapar
40     enable_interrupts(GLOBAL);   // Aktif edilen kesmelere izin ver
41
42     while(1)
43     {
44     for(i=0;i<=15;i++)
45        {
46           output_d(digit[i]); // digit[i] değerini B portuna gönder
47           delay_ms(500);      // 500 msn bekle
48        }
49     }
50  }
```

Kaynaklar

- CCS C Programlama Kitabı, Serdar Çiçek, Altaş Yayıncılık
- Mikroelektronika C programlama e-kitabı «https://www.mikroe.com/ebooks/pic-microcontrollers-programming-in-c»
- Mikroişlemciler Lab. EE-304 dersi deney föyleri, Doç. Dr. Mehmet DEMİRTAŞ