



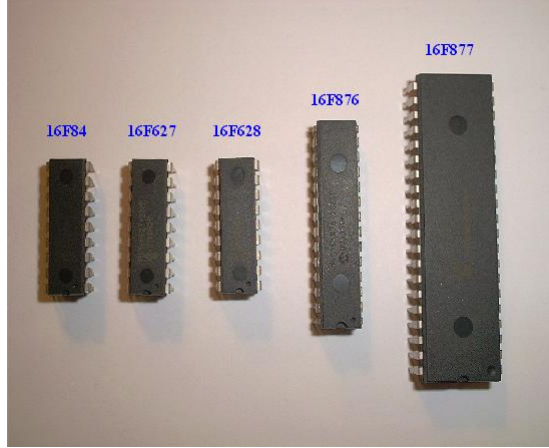
Microchip PIC Mikrodenetleyicileri

EEM 332 Mikroişlemciler

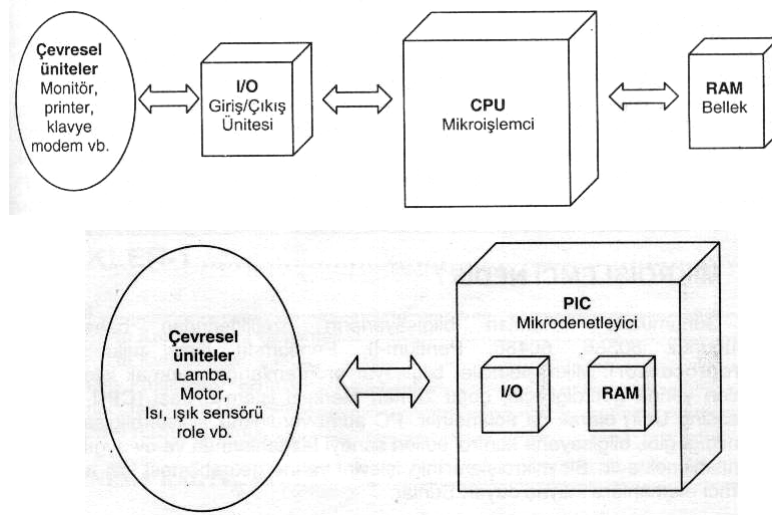


PIC Nedir?

- PIC:Peripheral Interface Controller
- Çevresel Arayüz Birimi Denetleyicisi
- Amacı: Çok fonksiyonlu logic uygulamalarının hızlı ve ucuz bir mikroişlemci ile yazılım yoluyla karşılanmasıdır
- İlk PIC: PIC16C54 (1994)
- En Popüler PIC: PIC16F(C)84



Neden PIC?



Neden PIC?

- Lojik uygulamalarının hızlı olması
- Fiyatının oldukça ucuz olması
- 8 bitlik mikrodeneleyici olması, bellek ve veri için ayrı yerleşik bus' ların kullanılması
- Veri ve belleğe hızlı olarak erişimin sağlanması
- PIC'e göre diğer mikrodeneleyicilerde veri ve programı taşıyan bir tek bus bulunması, dolayısıyla PIC' in bu özelliği ile diğer mikrodeneleyicilerden iki kat daha hızlı olması
- Yüksek frekanslarda çalışabilme özelliği
- Standby durumunda çok düşük akım çekmesi
- Azaltılmış kod yapısı (RISC yapısı) ile işlemleri daha hızlı gerçekleştirebilmesi

Neden PIC?

- Herhangi bir ek bellek veya giriş/çıkış elemanı gerektirmeden sadece 2 kondansatör, bir kristal ve bir direnç ile çalışabilmeleri
- Program belleğinin Eeprom olması
- Seri olarak dört adet kabloyla bulunduğu devreden sökülmeden programlanabilmesi

RISC Programlama

- RISC: *reduced instruction set computer*
- Avantajları:
 - Komutlar daha hızlı çalıştırılır
 - RISC tabanlı çipler daha az transistör'e ihtiyaç duyar. Tasarlanması kolay ve maliyeti ucuzdur.

Komutlar - I

- Assembly dili ile PIC'e istediklerinizi yaptırmak için 35 komut öğrenmek yeterlidir
- Yazılım Pascal, C/C++, Basic, Assembler ve hatta Binary olarak yazılabilir
- Her komut 14 bit uzunluğundadır ve istisnai bir kaç komut dışında (CALL, GOTO ve bit test eden BTFSS, INCFSZ gibi) her komut tek çevrimde çalışır

Komutlar - II

Mnemonic, Operands		Description	Cycles	14-Bit Opcode		Status Affected	Notes
				MSb	LSb		
BYTE-ORIENTED FILE REGISTER OPERATIONS							
ADDWF	f, d	Add W and f	1	00	0111 dfff ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00	0101 dfff ffff	Z	1,2
CLRF	f	Clear f	1	00	0001 1fff ffff	Z	2
CLRWF	-	Clear W	1	00	0001 0xxx xxxx	Z	
COMF	f, d	Complement f	1	00	1001 dfff ffff	Z	1,2
DECf	f, d	Decrement f	1	00	0011 dfff ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011 dfff ffff		1,2,3
INCF	f, d	Increment f	1	00	1010 dfff ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111 dfff ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00	0100 dfff ffff	Z	1,2
MOVF	f, d	Move f	1	00	1000 dfff ffff	Z	1,2
MOVWF	f	Move W to f	1	00	0000 1fff ffff		
NOP	-	No Operation	1	00	0000 0xx0 0000		
RLF	f, d	Rotate Left f through Carry	1	00	1101 dfff ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00	1100 dfff ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00	0010 dfff ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00	1110 dfff ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00	0110 dfff ffff	Z	1,2

Komutlar - III

BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF f, b	Bit Clear f	1	01	00bb bfff ffff		1,2
BSF f, b	Bit Set f	1	01	01bb bfff ffff		1,2
BTFSC f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb bfff ffff		3
BTFSS f, b	Bit Test f, Skip if Set	1 (2)	01	11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW k	Add literal and W	1	11	111x kkkk kkkk	C,DC,Z	
ANDLW k	AND literal with W	1	11	1001 kkkk kkkk	Z	
CALL k	Call subroutine	2	10	0kkk kkkk kkkk		
CLRWDt -	Clear Watchdog Timer	1	00	0000 0110 0100	TO,PD	
GOTO k	Go to address	2	10	1kkk kkkk kkkk		
IORLW k	Inclusive OR literal with W	1	11	1000 kkkk kkkk	Z	
MOVLW k	Move literal to W	1	11	00xx kkkk kkkk		
RETFIE -	Return from interrupt	2	00	0000 0000 1001		
RETLW k	Return with literal in W	2	11	01xx kkkk kkkk		
RETURN -	Return from Subroutine	2	00	0000 0000 1000		
SLEEP -	Go into standby mode	1	00	0000 0110 0011	TO,PD	
SUBLW k	Subtract W from literal	1	11	110x kkkk kkkk	C,DC,Z	
XORLW k	Exclusive OR literal with W	1	11	1010 kkkk kkkk	Z	

Accumulator

- 8086'daki kullanılan yazmaçlar (AX,BX,CX,DX) yerine PIC yalnızca W Yazmacı'nı kullanır
- 8086'dakinin aksine, bir değeri bir değişkene atmak istediğimizde mutlaka W yazmacını kullanmak zorundayız

W Yazmacı (W Register)

- Bu register sanal bir saklayıcıdır. İçeriğine direk ulaşmak mümkün değildir. Ancak bütün yükleme işlemleri bu register yardımıyla yapılmaktadır.
- Bir değişkene, bir registre yada bir porta bilgi göndermek için, önce bu bilgiyi W registerine, daha sonra W registerini ilgili porta ya da değişkene yüklemek yolunu kullanmak şarttır.
- Örnek: Bir değişkene bilgi yükleme:
`MOVLW D'15' ;W=15`
`MOVWF SAYAC ;sayac=15`

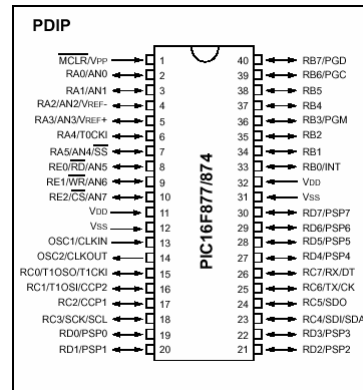
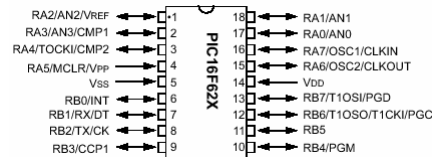
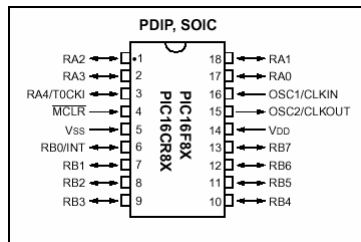
W Yazmacı (W Register)

- Örnek: Bir değişken içeriğini başka bir değişkene yükleme.

```
MOVF SAYAC1,W ;W=SAYAC1
MOVWF SAYAC2 ;SAYAC1=SAYAC2
```

!!! Bir değişken ya da register içeriğini W'e yüklemek için MOVF (ya da MOVWF) komutunu kullanmak şarttır. MOWLW komutunu kullanarak W'e sadece sabit bir değer yüklenebilir.

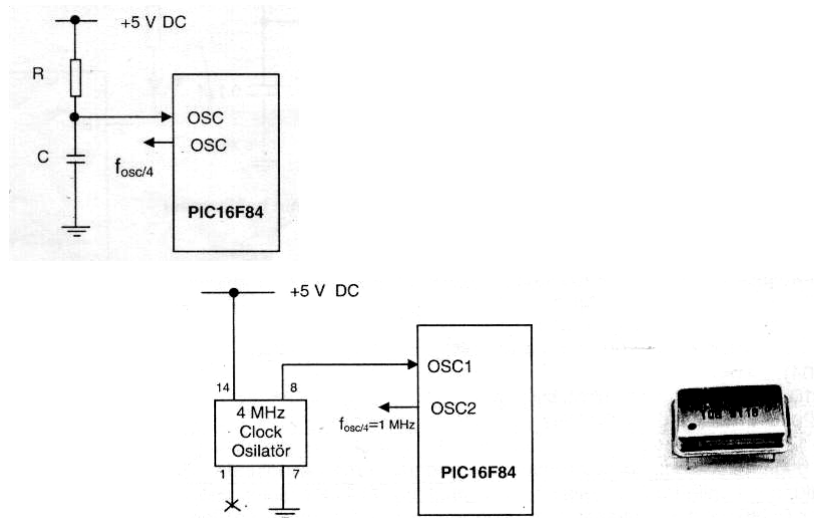
Giriş Çıkış Bacakları



Clock Uçları ve Bağlantıları

- PIC'in clock sinyal girişi olarak kullanılan 2 ucu vardır.
- Bu uçlara farklı tipte osilatörlerden elde edilen sinyaller uygulanabilir.
 - RC-Direnç/Kondansatör
 - XT-Kristal veya seramik resonatör (**XTAL**)
 - HS-Yüksek hızlı kristal veya resonatör (**High Speed**)
 - LP-Düşük frekanslı kristal (**Low Power**)
- PIC'e bağlanan clock'un tipi programlama sırasında konfigürasyon bitlerine yazılır.

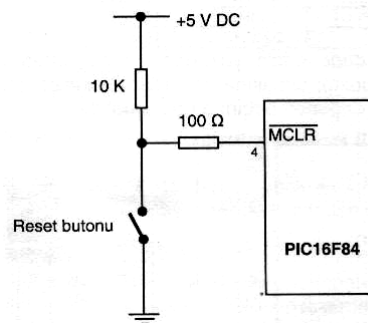
Clock Uçları ve Bağlantıları



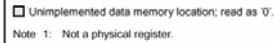
Clock Uçları ve Bağlantıları



Reset Devresi



- 8086'daki Segment'ler yerine PIC'te Bank'lar ve bankların içinde bulunan özel ve genel amaçlı Register'lar kullanılır.



Bank (16F877)

File Address	File Address	File Address	File Address
Indirect addr. ^(*) 00h	Indirect addr. ^(*) 80h	Indirect addr. ^(*) 100h	Indirect addr. ^(*) 180h
TMR0 01h	OPTION_REG 81h	TMR0 101h	OPTION_REG 181h
PCL 02h	PCL 82h	PCL 102h	PCL 182h
STATUS 03h	STATUS 83h	STATUS 103h	STATUS 183h
FSR 04h	FSR 84h	FSR 104h	FSR 184h
PORTA 05h	TRISA 85h	PORTA 105h	TRISA 185h
PORTB 06h	TRISB 86h	PORTB 106h	TRISB 186h
PORTC 07h	TRISC 87h	PORTC 107h	TRISC 187h
PORTD ^(*) 08h	TRISD ^(*) 88h	PORTD 108h	TRISD 188h
PORTE ^(*) 09h	TRISE ^(*) 89h	PORTE 109h	TRISE 189h
PCLATH 0Ah	PCLATH 8Ah	PCLATH 10Ah	PCLATH 18Ah
INTCON 0Bh	INTCON 8Bh	INTCON 10Bh	INTCON 18Bh
PIR1 0Ch	PIE1 8Ch	EEDATA 10Ch	EEDATA 18Ch
PIR2 0Dh	PIE2 8Dh	EEDR 10Dh	EEDR 18Dh
TMR1L 0Eh	PCON 8Eh	EEADRH 10Eh	Reserved ⁽²⁾ 18Eh
TMR1H 0Fh	8Fh	EEADRH 10Fh	Reserved ⁽²⁾ 18Fh
TTCN 10h	90h	110h	Reserved ⁽²⁾ 190h
TMR2 11h	SSPCON2 91h	111h	Reserved ⁽²⁾ 191h
T2CON 12h	PR2 92h	112h	Reserved ⁽²⁾ 192h
SSPBUF 13h	SSPMDR 93h	113h	Reserved ⁽²⁾ 193h
SSPCON 14h	SSPSTAT 94h	114h	Reserved ⁽²⁾ 194h
CCP1RL 15h	95h	115h	Reserved ⁽²⁾ 195h
CCP1RH 16h	96h	116h	Reserved ⁽²⁾ 196h
CCP1CON 17h	97h	117h	Reserved ⁽²⁾ 197h
RCSTA 18h	TXSTA 98h	118h	Reserved ⁽²⁾ 198h
TXREG 19h	SRSG 99h	119h	Reserved ⁽²⁾ 199h
RCREG 1Ah	9Ah	11Ah	Reserved ⁽²⁾ 19Ah
CCP2RL 1Bh	9Bh	11Bh	Reserved ⁽²⁾ 19Bh
CCP2RH 1Ch	9Ch	11Ch	Reserved ⁽²⁾ 19Ch
CCP2CON 1Dh	9Dh	11Dh	Reserved ⁽²⁾ 19Dh
ADRESH 1Eh	ADRESL 9Eh	11Eh	Reserved ⁽²⁾ 19Eh
ADCON0 1Fh	ADCON1 9Fh	11Fh	Reserved ⁽²⁾ 19Fh
20h	A0h	120h	Reserved ⁽²⁾ 1A0h
General Purpose Register 96 Bytes	General Purpose Register 80 Bytes	General Purpose Register 80 Bytes	General Purpose Register 80 Bytes
accesses 70h-7Fh	accesses 70h-7Fh	accesses 70h-7Fh	accesses 70h-7Fh
Bank 0 7Fh	Bank 1 FFh	Bank 2 17Fh	Bank 3 1FFh

^(*) Unimplemented data memory locations, read as '0'.

^(*) Not a physical register.

Note 1: These registers are not implemented on the PIC16F876.
2: These registers are reserved, maintain these registers clear.

PIC Nasıl Programlanır ?

■ PIC Programlamak İçin Gerekenler

- ☐ IBM uyumlu bilgisayar
- ☐ Metin editörü
- ☐ Assembler programı (MPLAB)
- ☐ PIC programlayıcı donanım ve yazılım
- ☐ PIC

Assembler programı (MPLAB)

- Ücretsiz bir yazılım
- Simulasyon yapılabilir
- www.microchip.com ve www.baskent.edu.tr/~eemekdas/dersler/ee_m332l.htm adreslerinden edinilebilir.

Status Yazmacı-1

- Status Yazmacı: Bank'lar arasındaki geçişi sağlar.

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	\overline{TO}	\overline{PD}	Z	DC	C
bit 7							bit 0

bit 6-5 **RP1:RP0:** Register Bank Select bits (used for direct addressing)
11 = Bank 3 (180h - 1FFh)
10 = Bank 2 (100h - 17Fh)
01 = Bank 1 (80h - FFh)
00 = Bank 0 (00h - 7Fh)
Each bank is 128 bytes

Status Yazmacı-2

- **BSF** **STATUS,RP0** ;Bit Set f
; (Status'da RP0)
- **BSF** **STATUS,5** ;Bit Set f
; (Status'da Bit 5)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	\overline{TO}	\overline{PD}	Z	DC	C
bit 7							bit 0

Program Bölümleri

<pre>LIST P=16F877 INCLUDE P16F877.INC</pre>		Başlık Bloğu
<pre>#DEFINE DEGISKEN_1 d'100' #DEFINE DEGISKEN_2 b'10010110' ;=d'150'=0x96 DEGIS_3 EQU 0x20 ;HAFIZADA SEÇİLEN BİR YERE DEGIS_4 EQU 0x21 ;ATANAN DEĞİŞKEN</pre>		Adres ve Değişken Atama Bloğu
<pre>ORG 0 ;PROGRAMI 0X00'DAN BAŞLAT</pre>		
BASLA	<pre>BSF STATUS,RP0 MOVLW b'00000111' MOVWF TRISA CLRF TRISB BCF STATUS,RP0 MOVLW b'00001010' MOVWF PORTB CLRF PORTB</pre>	<pre>;BANK1'E GEÇİŞ ;İSTENEN I/O KONFIGURASYONU SEÇİLİR ;TRISA REGISTER'INA ATANIR ;TRISB ÇIKIŞ MODUNA ATANIR (BÖYLECE ;PORTB'YE ATANACAK BİLGİLER ÇIKIŞ OLARAK ;GÖRÜLÜR) ;BANK0'A GEÇİŞ ;B PORTUNA 1010 BİLGİSİ GONDER ; ;DİYEREK ÇIKIŞI TAMAMEN SIFIRLAYABİLİRİZ ;SONSUZ LOOP</pre>
	<pre>DONGU GOTO DONGU END</pre>	Program Bloğu
Etiket Bloğu		Sonlandırma Bloğu

Büyük ve Küçük Harf Kullanımı

- PIC Komutlarının büyük veya küçük harfle yazılması önemli değildir:

`MoVlW=movlw=MOVLW`

- Ancak etiketler harf büyüklüğüne duyarlıdır

`Dongu≠DONGU`

Başlık Bloğu

- `LIST P=16F877`

Kullanılacak PIC'i tanımlar

- `INCLUDE P16F877.INC`

Kullanılan PIC'teki Yazmaç tanımlamaları bu dosyada saklanır. Böylece 'Default' olan yazmaçların teker teker tanımlanması gerekmez.

Kullanılan Sayı/Karakter Tipleri

■ Hexadecimal

STATUS	EQU	0x03	
		3	
		03	Adres ve Değişken
		03h	Atama Bloğu
		h'03'	
MOVLW		0x03	
		h'03'	
		03	Program Bloğu
		03h	

■ Binary

MOVLW		b'00010110'
-------	--	-------------

■ Decimal

MOVLW		d'15'
-------	--	-------

■ ASCII Character

RETLW		'A'
-------	--	-----

CONFIG Satırı

- Konfigurasyon bit'leri, PIC'e gerilim uygulandığı anda PIC'in uyması gereken koşulları belirler.

Koşul	Yazılacak Tanım Kodu
Kod koruma var	_CP_ON
Kod koruma yok	_CP_OFF
Power-on-reset var	_PWRTE_ON
Power-on-reset yok	_PWRTE_OFF
Watchdog timer devrede	_WDT_ON
Watchdog timer devrede yok	_WDT_OFF
Low Power Osilatör	_LP_OSC
Kristal Osilatör	_XT_OSC
High Speed Osilatör	_HS_OSC
RC Osilatör	_RC_OSC

Örnek Program - 1

```
LIST          P=16F877
INCLUDE       P16F877.INC

ORG          0                ;PROGRAMI 0X00'DAN BAŞLAT

BSF          STATUS,RP0       ;BANK1'E GEÇİŞ
CLRF         TRISD             ;TRISD ÇIKIŞ MODUNA ATANIR
BCF          STATUS,RP0       ;BANK0'A GEÇİŞ

MOVLW        b'01100110'      ;D PORTUNA 01100110 BİLGİSİ GONDER
MOVWF        PORTD

DONGU
GOTO DONGU                ;SONSUZ LOOP
END
```

Örnek Program - 2

```
LIST          P=16F877
INCLUDE       P16F877.INC
__CONFIG_CP_OFF & _WDT_OFF & _FWRTE_ON &
_XT_OSC
;-----
#define        DSAYI          0xFF
CNTRH         EQU            0x25
CNTRL         EQU            0x26
CNTRM         EQU            0x27
;-----
ORG          0x00
BASLA
    CLRF     PORTA
    CLRF     PORTB
    CLRF     PORTC
    CLRF     PORTD
    CLRF     PORTE

    BSF      STATUS,RP0
    CLRF     TRISB
    CLRF     TRISC
    CLRF     TRISD
    CLRF     TRISE
    BCF      STATUS,RP0
;-----
    CALL     BEK
    MOVLW    b'01010101'
    MOVWF    PORTB
    CALL     BEK
    MOVLW    b'11001100'
    MOVWF    PORTC

    CALL     BEK
    MOVLW    DSAYI
    MOVWF    PORTD
    CALL     BEK
    MOVLW    0x05
    MOVWF    CNTRM
    CALL     BEK
    MOVLW    0xFF
    MOVWF    CNTRL
    GOTO     DONGU

    BEK
    MOVLW    0x05
    MOVWF    CNTRM
    TEKM
    MOVLW    0xF0
    MOVWF    CNTRH
    TEK1
    MOVLW    0xFF
    MOVWF    CNTRL
    TEK2
    DECFSZ   CNTRL,F
    GOTO     TEK2
    DECFSZ   CNTRH,F
    GOTO     TEK1
    DECFSZ   CNTRM,F
    GOTO     TEKM
    RETURN

DONGU
    GOTO     BASLA
END
```