

PİC BASIC PROFESSIONAL

Farklı Mikroişlemcilerin farklı Assembler kodları olması genellikle sorun olmuştur. Bu dezavantajdan kurtulmak için compiler lar geliştirilmiştir. C++ ve Basic dillerinde yazılanlar bunların en çok kullanılanlarındandır. Microchip in kendi işlemcileri için geliştirdiği assembler dilde 34 adet komut bulunuyor. Bu komutların işlevlerini ve daha çoğunu yapan PicBasicPro dili bu yazıda tanıtılacaktır.

PicBasic Editörünü [creative contraptions](http://www.creative-contraptions.com) sitesinden ücretsiz indirebilirsiniz. Assembler ile PicBasic Komutlarının Karşılaştırılması:

Microchip in assembler komut seti aşağıda verilmiştir.

ADDLW k	ADDWF f,d	ANDLW k	ANDWF f,d
BCF f,b	BSF f,b	BTFSC f,b	BTFSS f,b
CALL k	CLRF f	CLRW	CLRWDT
COMF f,d	DECF f,d	DECFSZ f,d	GOTO k
INCF f,d	INCFSZ f,d	IORLW k	IORWF f,d
MOVF f,d	MOVLW k	MOVWF f	NOP
RETFIE	RETLW k	RETURN	RLF f,d
RRF f,d	SLEEP	SUBLW k	SUBWF f,d
SWAPF f,d	XORLW k	XORWF f,d	

b - bit address

d - destination; 0 = w, 1 = f

f - register file address

k - literal constant

Görüldüğü gibi bu komutları hafızada tutmak ve özellikle uzun programları yazmak sorun olmaktadır. Bunun yerine if,then,else,high,low,pause... gibi akılda kalması daha kolay ve assembler a göre kat kat daha yetenekli komutlarla program yapmak daha kolaydır.

Aşağıda PicBasic in komut seti verilmiştir.

@ Insert one line of assembly language code. ADCIN Read on-chip analog to digital converter. ASM..ENDASM Insert assembly language code section.
BRANCH Computed GOTO (equiv. to ON..GOTO). BRANCHL BRANCH out of page (long BRANCH). BUTTON Debounce and auto-repeat input on specified pin.
CALL Call assembly language subroutine. CLEAR Zero all variables. CLEARWDT Clear (tickle) Watchdog Timer. COUNT Count number of pulses on a pin. DATA Define initial contents of on-chip EEPROM. DEBUG Asynchronous serial output to fixed pin and baud. DEBUGIN Asynchronous serial input from fixed pin and baud.

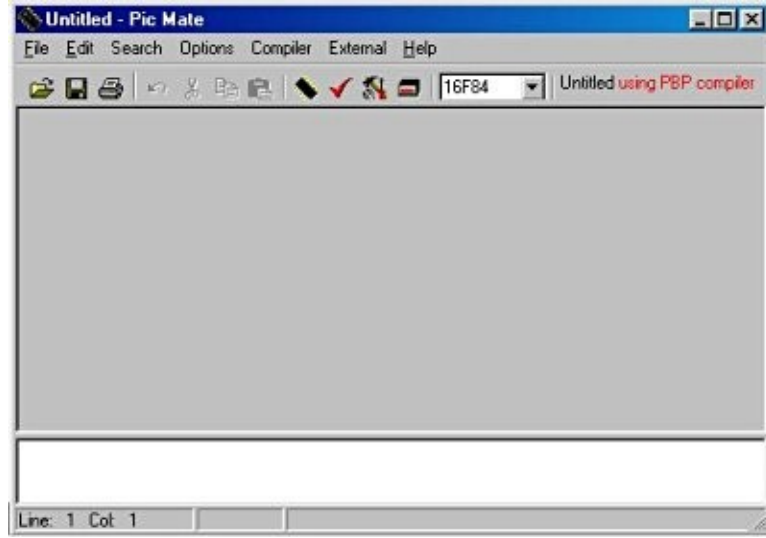
DISABLE Disable ON DEBUG and ON INTERRUPT processing.
DISABLE DEBUG Disable ON DEBUG processing.
DISABLE INTERRUPT Disable ON INTERRUPT processing.
DTMFOUT Produce touch-tones on a pin.
EEPROM Define initial contents of on-chip EEPROM.
ENABLE Enable ON DEBUG and ON INTERRUPT processing.
ENABLE DEBUG Enable ON DEBUG processing.
ENABLE INTERRUPT Enable ON INTERRUPT processing.
END Stop execution and enter low power mode.
FOR..NEXT Repeatedly execute statements.
FREQOUT Produce up to 2 frequencies on a pin.

GOSUB Call BASIC subroutine at specified label.
GOTO Continue execution at specified label.
HIGH Make pin output high.
HSERIN Hardware asynchronous serial input.
HSEROUT Hardware asynchronous serial output.
I2CREAD Read bytes from I2C device.
I2CWRITE Write bytes to I2C device.
IF..THEN..ELSE..ENDIF Conditionally execute statements.
INPUT Make pin an input.
LCDIN Read from LCD RAM.
LCDOUT Display characters on LCD.
{LET} Assign result of an expression to a variable.
LOOKDOWN Search constant table for value.
LOOKDOWN2 Search constant / variable table for value.
LOOKUP Fetch constant value from table.
LOOKUP2 Fetch constant / variable value from table.
LOW Make pin output low.
NAP Power down processor for short period of time.
ON DEBUG Execute BASIC debug monitor.
ON INTERRUPT Execute BASIC subroutine on an interrupt.
OUTPUT Make pin an output.
PAUSE Delay (1mSec resolution).
PAUSEUS Delay (1uSec resolution).
PEEK Read byte from register. (Do not use.)
POKE Write byte to register. (Do not use.)
POT Read potentiometer on specified pin.
PULSIN Measure pulse width on a pin.
PULSOUT Generate pulse to a pin.
PWM Output pulse width modulated pulse train to pin.
RANDOM Generate pseudo-random number.
RCTIME Measure pulse width on a pin.

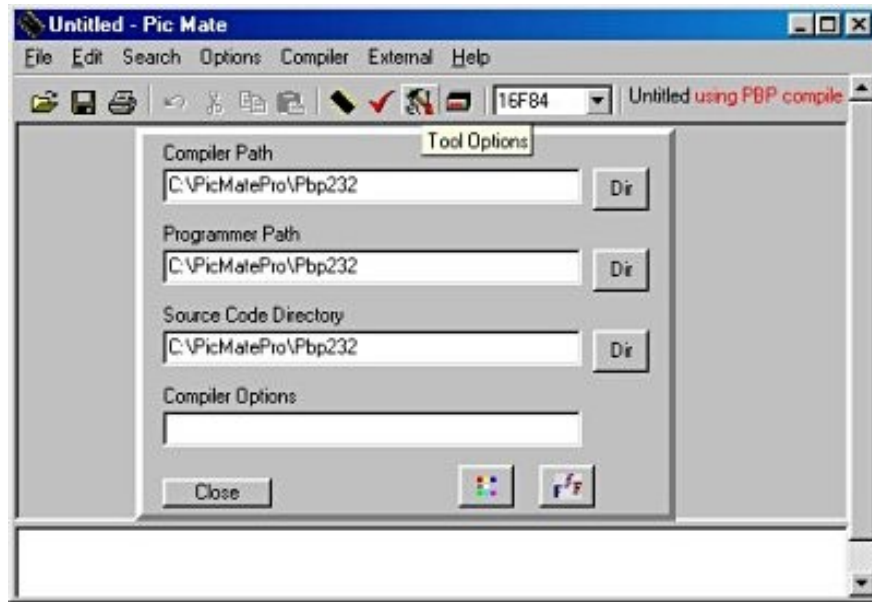
READ Read byte from on-chip EEPROM.
READCODE Read word from code memory.
RESUME Continue execution after interrupt handling.
RETURN Continue at statement following last GOSUB.
REVERSE Make output pin an input or an input pin an output.
SERIN Asynchronous serial input (BS1 style).
SERIN2 Asynchronous serial input (BS2 style).
SEROUT Asynchronous serial output (BS1 style).
SEROUT2 Asynchronous serial output (BS2 style).
SHIFTIN Synchronous serial input.
SHIFTOUT Synchronous serial output.
SLEEP Power down processor for a period of time.
SOUND Generate tone or white-noise on specified pin.
STOP Stop program execution.
SWAP Exchange the values of two variables.
TOGGLE Make pin output and toggle state.
WHILE..WEND Execute statements while condition is true.
WRITE Write byte to on-chip EEPROM.
WRITECODE Write word to code memory.
XIN X-10 input.
XOUT X-10 output.

PicBasic İle Çalışmak:

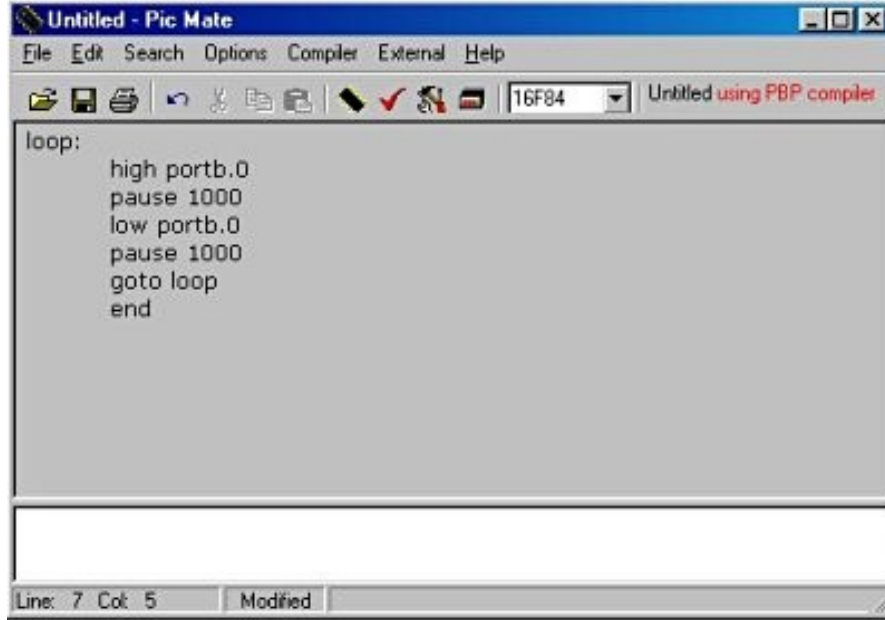
Programı download edip çalıştırdığınızda aşağıdaki menü ekrana çıkar.



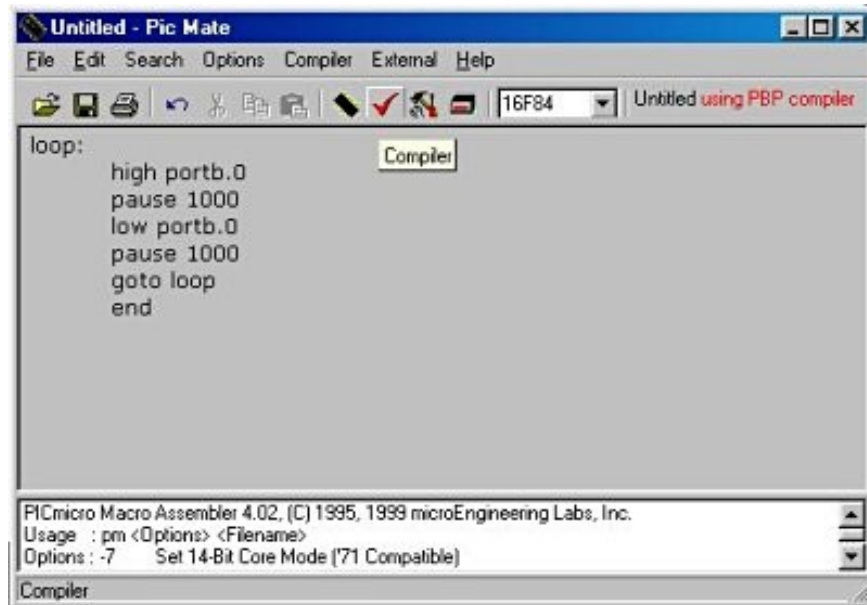
Bu yazılımda da tüm windows programlarında olan standart dosya aç,kapa,kaydet menüleri vardır. Bunlara ilaveten Programmer,Compiler,Tools Option ve çalışılacak mikroişlemciyi seçmemize yarayan combo box vardır. İlk kodlarımızı yazmadan önce Tools Option dan bazı dosyaların yerini göstermek gerekiyor.



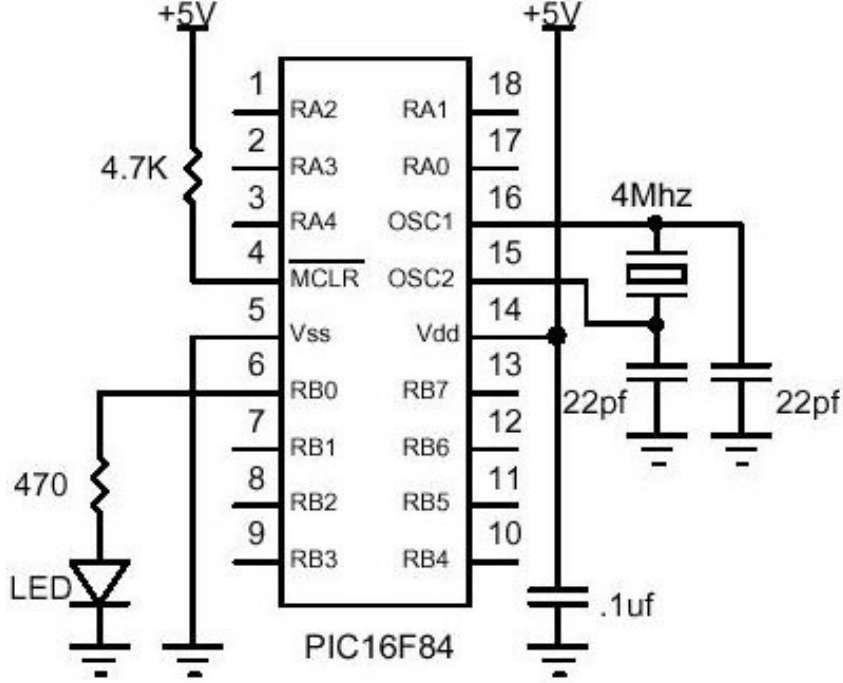
Bunlardan sonra artık ilk programımızı oluşturabiliriz. Aşağıdaki şekilde portb. 0 dan 1 saniyelik pulsler üreten bir kod verilmiştir.



Bu kodu yazdıktan sonra dosyayı kaydedip compile etmemiz gerekiyor. C:\PicMatePro\Pbp232 klasörünün içinde bizim yeni oluşturduğumuz file ler .ASM ve .HEX uzantılı olarak bulabiliriz.



Eğer bir hatamız yoksa (ki yukardaki yazılım denenmiştir) HEX uzantılı dosyayı mikroişlemcinin içine yükleyip devreyi çalıştırabiliriz.



PIC KOMUTLARI

PIC meraklılarını uzunca bir süre beklettim. Yoğun bir çalışma temposunun ardından (henüz bitmemiş olsa da) bir şeyler hazırlamak için vakit bulabildim.

Artık pic programlamaya hazırlanacağız. Pic programlamak için assembly, basic, C dilleri geliştirilmiştir. Biz ücretsiz olması nedeniyle pic assembly dili üzerinde duracağız ve MPLab yazılım geliştirme ortamını tanıyacağız. Daha önce de belirttiğimiz gibi PIC ailesinden PIC16F84 mikro denetleyicisinin 35 adet assembly komutu vardır. Şimdi bu komutlara, biraz detaylı olarak göz atalım.

ADDLW Literal ile W'yu topla

Yazılış : ADDLW k

İşleçler: $0 \leq k \leq 255$

İşlem: $(W) + k \rightarrow (W)$

Etkilenen Yazmaçlar : C, DC, Z

Kodlama : 11 111x kkkk kkkk

Tanım : W yazmacının içeriği sekiz bitlik k literali ile toplanır, sonuc W yazmacına yerleştirilir.

Word: 1

Saat Çevrimi (Cycle) : 1

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"k" literalini oku	İşlemi yap	W'ye yaz

Örnek : ADDLW 0x15

İşlemden önce $W = 0x10$

İşlemden sonra $W = 0x25$

ADDWF W ve f'i Topla

Yazılış : ADDWF f,d

İşleçler: $0 \leq f \leq 127$, $d \in [0,1]$

İşlem: $(W) + (f) \rightarrow (\text{destination})$

Etkilenen Yazmaçlar : C, DC, Z

Kodlama : 00 0111 dfff ffff

Tanım : W yazmacının içeriğiyle "f" yazmacının içeriğini topla. Eğer "d" 0 ise sonucu W 'ye, 1 ise "f" yazmacına yaz.

Word: 1

Saat Çevrimi (Cycle) : 1

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"f" yazmacını oku	İşlemi yap	Sanucu hedefe yaz

Örnek : ADDWF FSR, 0

İşlemden önce W = 0x17 FSR = 0xC2

İşlemden sonra W = 0xD9 FSR = 0xC2

ANDLW Literal ile W'yi mantıksal VE

Yazılış : ANDLW k

İşlemler: $0 \leq k \leq 255$

İşlem: (W) .AND. (k) $\rightarrow$ (W)

Etkilenen Yazmaçlar : Z

Kodlama : 11 1001 kkkk kkkk

Tanım : W yazmacı ile sekiz bitlik "k" literaline VE işlemi uygulanır.

Sonuç W'ye yazılır.

Word: 1

Saat Çevrimi (Cycle) : 1

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"k" literalini oku	İşlemi yap	W'ye yaz

Örnek : ANDLW 0x5F

İşlemden önce W = 0xA3

İşlemden sonra W = 0x03

ANDWF W ile "f"e mantıksal VE işlemi uygula

Yazılış : ANDWF f,d

İşleçler: $0 \leq f \leq 127$, $d \in [0,1]$

İşlem: (W) .AND. (f) $\rightarrow$ (destination)

Etkilenen Yazmaçlar : Z

Kodlama : 00 0101 dfff ffff

Tanım : W yazmacı ile "f" yazmacına VE işlemi uygula. Eğer "d" 0 ise sonucu W 'ye, 1 ise "f" yazmacına yaz.

Word: 1

Saat Çevrimi (Cycle) : 1

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"f" yazmacını oku	İşlemi yap	Sonucu yaz

Örnek : ANDWF FSR, 1

İşlemden önce W = 0x17, FSR = 0xC2

İşlemden sonra W = 0x17, FSR = 0x02

BCF f'in bir bitini sil (sıfırla)

Yazılış : BCF f,b

İşleçler: $0 \leq f \leq 127$, $0 \leq b \leq 7$

İşlem: $0 \rightarrow (f[b])$

Etkilenen Yazmaçlar : yok

Kodlama : 01 00bb bfff ffff

Tanım : "f" yazmacının "b" biti sıfırlanır.

Word: 1

Saat Çevrimi (Cycle) : 1

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"f" yazmacını oku	İşlemi yap	"f" yazmacına yaz

Örnek : BCF FLAG_REG, 7

İşlemden önce FLAG_REG = 0xC7

İşlemden sonra FLAG_REG = 0x47

BSF f'in bir bitini kur (1 yap)

Yazılış : BSF f,b

İşleçler: 0 ≤ f ≤ 127, 0 ≤ b ≤ 7

İşlem: 1 -rarr; (f[b])

Etkilenen Yazmaçlar : Yok

Kodlama : 01 01bb bfff ffff

Tanım : "f" yazmacının "b" biti 1 yapılır.

Word: 1

Saat Çevrimi (Cycle) : 1

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"f" yazmacını oku	İşlemi yap	"f" yazmacına yaz

Örnek : BSF FLAG_REG, 7

İşlemden önce FLAG_REG = 0x0A

İşlemden sonra FLAG_REG = 0x8A

BTFSC "f" yazmacının "b" bitini kontrol et, 0 ise atla

Yazılış : BTFSC f,b

İşleçler: 0 ≤ f ≤ 127, 0 ≤ b ≤ 7

İşlem: eğer (f[b]) = 0 ise bir sonraki komutu atla

Etkilenen Yazmaçlar : Yok

Kodlama : 01 10bb bfff ffff

Tanım : Eğer "f" yazmacının "b" biti 1 ise sıradaki komut işletilir, 0 ise bir sonraki komut atlanır ve yerine NOP işletilir ve 2 çevrimli komut olur.

Word: 1

Saat Çevrimi (Cycle) : 1 (2)

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"f" yazmacını oku	İşlemi yap	İşlem yapılmaz

Eğer atlama yapılmışsa 2. çevrimde ;

Q1	Q2	Q3	Q4
İşlem yapılmaz	İşlem yapılmaz	İşlem yapılmaz	İşlem yapılmaz

BTFSS "f" yazmacının "b" bitini kontrol et, 1 ise atla

Yazılış : BTFSS f,b

İşleçler: 0 ≤ f ≤ 127, 0 ≤ b ≤ 7

İşlem: eğer (f[b]) = 1 ise bir sonraki komutu atla

Etkilenen Yazmaçlar : Yok

Kodlama : 01 11bb bfff ffff

Tanım : Eğer "f" yazmacının "b" biti 0 ise sıradaki komut işletilir, 1 ise bir sonraki komut atlanır ve yerine NOP işletilir ve 2 çevrimli komut olur.

Word: 1

Saat Çevrimi (Cycle) : 1 (2)

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"f" yazmacını oku	İşlemi yap	İşlem yapılmaz

Eğer atlama yapılmışsa 2. çevrimde ;

Q1	Q2	Q3	Q4
İşlem yapılmaz	İşlem yapılmaz	İşlem yapılmaz	İşlem yapılmaz

CALL Alrutin çağır

Yazılış : CALL k

İşleçler: 0 □ k □ 2047

İşlem: $(PC)+1 \rightarrow TOS$, $k \rightarrow PC<10:0>$, $(PCLATH<4:3>) \square$

$PC<12:11>$

Etkilenen Yazmaçlar : Yok

Kodlama : 10 0kkk kkkk kkkk

Tanım : Bir altrutin çağırır. Önce dönüş adresi $(PC+1)$ yığın (stack) hafızaya kaydedilir. 11 bitlik adres $PC<10:0>$ 'a yüklenir. PC'nin üst bitleri PCLATCH'den yüklenir. CALL 2 çevrimlik bir komuttur.

Word: 1

Saat Çevrimi (Cycle) : 2

Q çevrimi (Q Cycle Activity) :

1. çevrim;

Q1	Q2	Q3	Q4
----	----	----	----

Kod çöz	"k" literalini oku PC'yi yığına kaydet	Veriyi işle	PC'ye yaz

2. çevrim;

Q1	Q2	Q3	Q4
İşlem yapılmaz	İşlem yapılmaz	İşlem yapılmaz	İşlem yapılmaz

CLRF f'in içeriğini temizle (0 yap)

Yazılış : CLRF f

İşleçler: 0 f 127

İşlem: 00h → (f), 1 → Z

Etkilenen Yazmaçlar : Z

Kodlama : 00 0001 1fff ffff

Tanım : "f" yazmacının içeriği temizlenir(0) ve "Z" biti kurulur(1).

Word: 1

Saat Çevrimi (Cycle) : 1

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"f" yazmacını oku	Veriyi işle	"f" yazmacına yaz

Örnek : CLRWD

İşlemden önce WDT counter = ?

İşlemden sonra WDT counter = 0x00,

WDT prescaler= 0, TO =1, PD =1

COMF "f" in tümleyenini al

Yazılış : COMF f,d

İşleçler: 0 ≤ f ≤ 127, d ∈ [0,1]

İşlem: (f) → (destination)

Etkilenen Yazmaçlar : Z

Kodlama : 00 1001 dfff ffff

Tanım : "f" yazmacının içeriğinin tümleyeni alınır. Eğer d 0 ise, sonuc "w", 1 ise "f" yazmacına yazılır.

Word: 1

Saat Çevrimi (Cycle) : 1

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"f" yazmacını oku	Veriyi işle	Hedefeye yaz

Örnek : COMF REG1,0

İşlemden önce REG1 = 0x13

İşlemden sonra REG1 = 0x13

DECF "f" yazmacının içeriğini 1 azalt

Yazılış : DECF f,d

İşleçler: 0 ≤ f ≤ 127, d ∈ [0,1]

İşlem: (f) - 1 → (destination)

Etkilenen Yazmaçlar : Z

Kodlama : 00 0011 dfff ffff

Tanım : "f" yazmacının içeriğini bir(1) azalt. Eğer d 0 ise, sonuc "w", 1 ise "f" yazmacına yazılır.

Word: 1

Saat Çevrimi (Cycle) : 1

Q çevrimi (Q Cycle Activity) :

Q1	Q2	Q3	Q4
Kod çöz	"f" yazmacını oku	Veriyi işle	Hedefeye yaz

Örnek : DECF CNT, 1

İşlemden önce CNT = 0x01, Z =0

İşlemden sonra CNT = 0x00, Z =1