

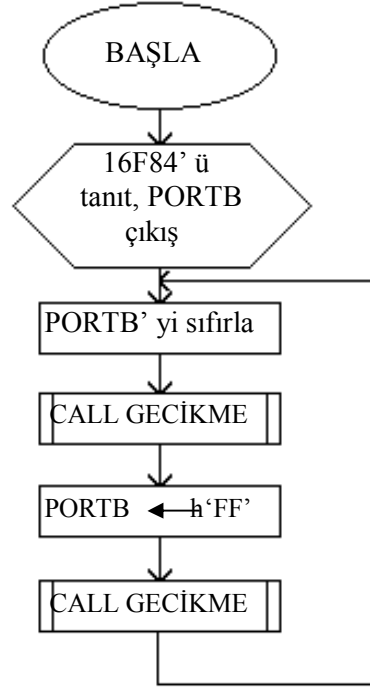
Program Örneği 9 : Gecikme altprogramı kullanarak Port B' ye bağlı tüm LED' leri yakıp söndüren bir program için akış diyagramı çizerek gerekli assembly programını PIC 16F84 için yapınız.

Cözüm: ; PROGRAM9 . ASM-----15 / 12 / 2003

```

LIST          P = 16F84
INCLUDE      " P16F84.INC"
SAYAC1 EQU   h'0C' ;16F84 de genel amaçlı RAM (Veri Hafızasında) ilk adres
SAYAC2 EQU   h'0D'
BSF         STATUS,5
CLRF        TRISB
BCF         STATUS,5
DEVAM
MOVLW      h'00'
MOVWF      PORTB
CALL       GECİKME
MOVLW      h'FF'
MOWF      PORTB
CALL       GECİKME
GOTO      DEVAM
GECİKME
MOVLW      h'FF'
MOWF      SAYAC1
TEKRAR1 MOWF SAYAC2
TEKRAR2 DECFSZ SAYAC2, F
GOTO      TEKRAR2
DECFSZ    SAYAC1, F
GOTO      TEKRAR1
RETURN
END

```



Program Örneği 10: PORTA' nın 0. bitine bağlı ve basıldığında (0) üreten bir butona 8 kere basıldıktan sonra yine PORTA' nın 1. bitine 2 kere basılırsa PORTB' ye bağlı bütün LED'lerin yanması isteniyor (Her tuşa basıldıktan sonra GECİKME alt programı kullanılacaktır). Gerekli akış diyagramını çizerek programı yazınız.

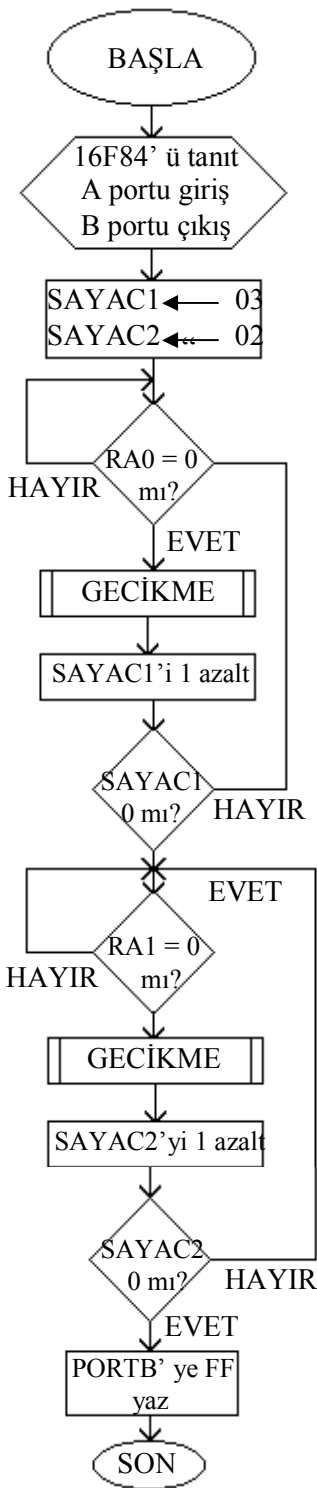
Cözüm:

; PROGRAM10.ASM-----22 / 12 / 2003

```

LIST          P = 16F84
INCLUDE      " P16F84.INC"
GECİK1 EQU   h'0C'
GECİK2 EQU   h'0D'
SAYAC1 EQU   h'0E'
SAYAC2 EQU   h'0F'
CLRF        PORTB ; PORTB yi sıfırla
BSF         STATUS,5 ;Bank1' e geç
CLRF        TRISB ;PORTB çıkış
BSF         TRISA, 0
BSF         TRISA, 1 ;RA0veRA1 Giriş
BCF         STATUS,5 ; Bank0' a geç
MOVLW      h'03'
MOVWF      SAYAC1 ; SAYAC1'e 03 yaz
MOVLW      h'02'

```



```

TEKRAR   MOVWF   SAYAC2   ;SAYAC2'ye 02 yaz
          BTFSZ   PORTA, 0   ; RA0 = 0 mı?
          GOTO    TEKRAR   ;Değilse TEKRAR'a
          CALL    GECİKME  ;Evetse GECİKMEye
          DECFSZ  SAYAC1   ;SAYAC1'i 1 azalt
          GOTO    TEKRAR   ; Sonuç 0 değilse
DEVAM     BTFSZ   PORTA, 1   ; RA1 = 0 mı?
          GOTO    DEVAM   ; Değilse DEVAM'a
          CALL    GECİKME  ;Evetse GECİKMEye
          DECFSZ  SAYAC2   ; SAYAC2'i 1 azalt
          GOTO    DEVAM   ;Sonuç 0 değilse
          MOVLW   h'FF'
          MOVWF   PORTB    ;PORTB'ye FF yükle
          ; GECİKME ALT PROGRAMI
GECİKME   MOVLW   h'FF'
          MOVWF   GECİK1   ;GECİK1'e FF yükle
DONGU1   MOVLW   h'FF'
          MOVWF   GECİK2   ;GECİK2'ye FF yükle
DONGU2   DECFSZ  GECİK2, F ; GECİK2'yi 1 azalt
          GOTO    DONGU2   ;Sonuç 0 değilse
          DECFSZ  GECİK1, F ;GECİK1'i 1 azalt
          GOTO    DONGU1   ; DONGU1 'e git.
          RETURN  ; Alt Programdan dön
          END    ; SON

```

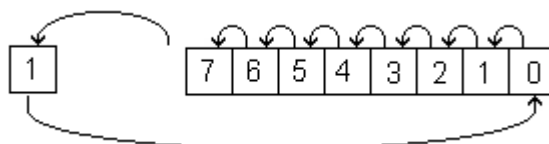
LOJİK İŞLEM KOMUTLARI

RLF Komutu (Bir bit Sola Kaydırma)

Bir file register içinde bulunan bitlerin birer bit sola kaydırılması işlemidir. Bu durumda en solda bulunan (7. bit) C (elde) bayrağına geçmekte, daha önce C' de bulunan bit ise en sağdaki (0.) bite geçmektedir. Komutun formatı;

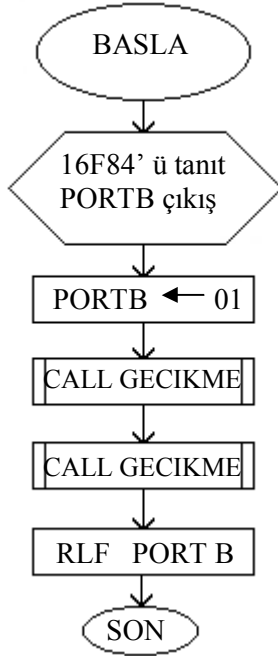
RFL **FILE REGISTER** , **d** ← W yada F (Sonucun Yeri)

Komutu şematik gösterirsek;



Program Örneği 11: Bir programla önce B portunun sadece 0. bitini (RB0) 1 yapın. Daha sonra 1 defa bu biti sola kaydırın. Bu işlemin daha iyi görülebilmesi için üst üste 2 kere GECIKME alt programı kullanın.

Cözüm:



```

; PROGRAM 11 . ASM_TARİH
LIST P = 16F84
INCLUDE "P16F84.INC"
GECİK1 EQU h'0C'
GECİK2 EQU h'0D'
BCF STATUS, 0 ; Elde yi sıfırla
BSF STATUS, 5 ; Bank1'e geç
CLRF TRISB ; PORTB Çıkış
BCF STATUS, 5 ; Bank0'a geç
MOVLW h'01'
MOVWF PORTB ;PORTB' ye 01 yaz
CALL GECIKME_ALTPROG
CALL GECIKME_ALTPROG
BEKLE GOTO BEKLE

GECIKME_ALTPROG ; Önceki Gecikme Alt Prog ile aynı
MOVLW h'FF'
MOVWF GECIK1
DONGU1 MOVLW h'FF'
MOVWF GECIK2
DONGU2 DECFSZ GECIK2, F
GOTO DONGU2
DECFSZ GECIK1, F
GOTO DONGU1
RETURN
END
  
```

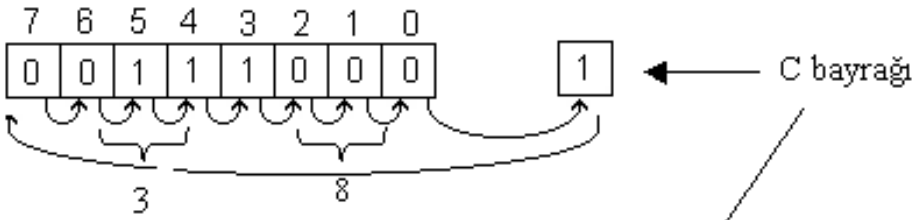
RRF Komutu (Bir Bit Sağa Kaydırma)

RRF komutu da RLF komutuna benzer olup fark bu defa kaydırma işleminin sağa olması dolayısıyla en sağdaki bit C (elde) bayrağına geçecektir. Daha önce C bayrağında bulunan bit ise bu sefer en soldaki (7.(bite) geçecektir. Komut formatı;

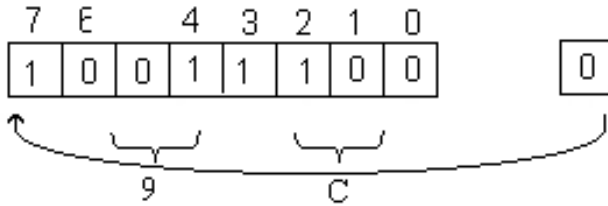
RRF **FİLE REGİSTER** , **d** ← Sonucun gideceği yer
W yada F

Mesela MEM adlı bir file registre hex 30 sayısını yazın. Aynı anda C bayrağı da (C= 1) ise RRF komutunun icrasından önce ve sonraki durum:

Önceki Durum



Sonraki Durum



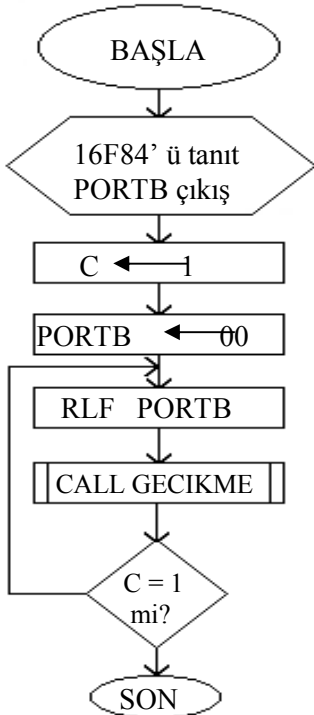
Bu işlem için kullanılacak program komutları :

```
MOVLW    h'49'    ; W Registerine (49)16 yükle
MOVWF    MEM      ; W ' yi MEM adresine sakla
RRF      MEM, F   ; MEM ' i Sağa bir bit ötele.
           şeklinde olacaktır.
```

Program Örneği 12: Önce C bayrağını (1) yapın ve başlangıçta sıfırlanmış PORT B üzerinden birer bit sola kaydırın. Her kaydırma arasında GECIKME alt programına gönderin. C bayrağı tekrar (1) olunca programı bitirin.

NOT: GECIKME_ALTPROG Gösterilmeyecektir.

Cözüm:



```
; PROGRAM 12.ASM-----Tarih
INCLUDE    "PIC16F84.INC"
BSF        STATUS, 5      ; Bank1'e geç
CLRF      TRISB          ; PORTB Çıkış
BCF        STATUS, 5      ; Bank0' a geç

KAYDIR
RLF        PORTB          ; PORTB yi sola kaydır
CALL      GECIKME_ALTPROG
BTFS      STATUS, 0      ; C bayrağı 1 mi?
GOTO      KAYDIR          ; Değilse KAYDIR'a

DONGU
GOTO      DONGU          ; Evetse burada bekle
END
```

COMF ve SWAPF Komutları

COMF komutu ile istenen bir file register içinde (0) lar (1) ve (1) ler (0) yapılabilir [1' e göre tümleyen işlemi !..] . Komut formatı;

COMF FILE REGISTER , d ← Sonucun gideceği yer
(destination) (W yada F yazılır)

şeklindedir.

Örnek: HAFIZA1 adlı registre (1F)₁₆ yüklendikten sonra bunun tersi olan (E0)₁₆ ' 1 HAFIZA2' ye saklayın. Bunun için gerekli rutini (Program Parçasını) yazın.

Cözüm:

		HAFIZA1
		0001 1111
MOVLW	h'0F'	
MOVWF	HAFIZA1	HAFIZA2
COMF	HAFIZA1, W	
MOVWF	HAFIZA2	1110 0000

SWAPF komutu ise bir file register içindeki ilk dört bit (Yüksek anlamlı Nibble) ile son dört bitlerin (Düşük anlamlı Nibble) yerlerini değiştirir. Komut formatı;

SWAPF FILE REGISTER , d şeklindedir.

Örnek: PORTB' ye (3F) yazdıktan sonra ilk ve son bitlerin yerini değiştiren ve sonucu W ye yazan Program parçası yazın.

```
MOVLW    h'3F'
MOVWF    PORTB
SWAPF    PORTB, W
```

ANDLW Komutu

W registerinin içeriğini sabit bir sayı ile Lojik AND (VE) işlemine tabi tutar. Sonucu tekrar W 'ye yazar. Bilindiği gibi işlemin sonunda her iki sayıda 1-1 olan bitler 1; diğerleri 0 olarak neticelenecektir. Komut formatı;

ANDLW Sabit sayı şeklindedir.

Bu komutla bir sayının (veri) istenen bitleri (0) yapılırken diğerleri olduğu gibi bırakılır. Bunun için sabit sayı (**maske**) seçilirken sıfır yapılmak istenenler (0), diğerleri (1) olarak seçilmelidir.

Örnek: W içinde (3C) sayısını olsun. Bu sayının 2.,3. ve 6. bitlerini (0) yapıp diğerlerini aynen bırakan komutu yazalım. Sonuçta sayı ne olur?

Önce W deki sayıya hiç bakmadan 2.,3.,6. bitleri 0, öteki bitleri 1 olan bir **maske** seçelim.

7 6 5 4 3 2 1 0
|

$$\begin{array}{|c|c|c|c|c|c|c|c|} \hline 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \hline \end{array} = (B3)_{16}$$

Bu maskeyle W deki sayıyı (3C yi) Lojik VE işlemine tabi tutmak için gerekli komut:

ANDLW h'B3' olacaktır.

Sonucu bulmak için ise, W deki sayı (3C)₁₆ verildiğine göre bu sayının 2.,3. ve 6. bitlerinin sıfırlanmış halini yazmamız gerekir. O da (30)₁₆ olarak elde edilecektir. [3C de 6. bit zaten 0 olduğuna göre, 3C yi binary yazın ve 2. ve 3. bitlerini sıfırlayın]

ANDWF Komutu

Bu komut ANDLW komutuna benzer olup bu defa W ile herhangi bir file register arasında Lojik AND (ve) işlemi gerçekleştirilir. Sonuç ise ya W 'ye ya da F 'e yazılır. Komut formatı:

ANDWF File Register, d ←(Sonucun yeri, W ya da F) şeklindedir.

Örnek: PORTB registerinin içinde b'00101100' sayısı olsun. W Registerine de b'11011111' değerini (maske) yerleştirelim. Bu iki sayıyı Lojik AND işlemine tabi tutarak neticeyi yine PORTB ye yazalım. Bunun için gerekli program parçası :

```
MOVLW    b'00101100' ; W Registerine 2C yükle.
ANDWF    PORTB, F    ; PORTB ile W yi AND (VE) işlemine tabi tut
                        ; ve sonucu yine PORTB ye yaz.
```

IORLW Komutu

Bu komut istenen bir biti 1 yapıp diğerlerini aynen bırakmak için kullanılır. Verilen sabit sayı ile W registeri Lojik OR (VEYA) işlemi yapar ve sonucu yine W registerine yazar. Komut Formatı:

IORLW Sabit sayı şeklindedir.

Bu amaçla seçilecek sabit sayıda (Maske'de), (1) yapılmak istenen bitler (1), değiştirilmesi istenmeyen bitler (0) seçilmelidir.

Örnek: W Registerinde bulunan sayının 5.,6.,7. bitleri 1 yapıp diğer bitler aynen bırakılmak istensin.

Önce Maskemizi seçelim;

7 6 5 4 3 2 1 0

$$\begin{array}{|c|c|c|c|c|c|c|c|} \hline 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline \end{array}$$

= E0 (Sayıda 1 yapılacak bitler 1)

Bu durumda gerekli program parçası (komut)

IORLW b'1110 0000' olacaktır.

NOT: Maske seçimi işlemi için W da bulunan sayının bilinmesine gerek yoktur. Hangi sıradaki bitlerin (burada 5,6,7 idi) 1 yapılacağıın bilinmesi yeterlidir. Soruda "sayı W dadır" denildiği için program parçasında W 'ya yükleme komutuna ihtiyaç yoktur.

Örnek: Yukarıdaki örnekte W daki sayımız (04)₁₆ olsaydı sonuç ne olurdu?

W ile Maske arasında OR işlemi: (sonuç E4 olacaktır)

Bit No:	7	6	5	4	3	2	1	0
W Reg.	0	0	0	0	0	1	0	0
Maske	1	1	1	0	0	0	0	0
Sonuç	1	1	1	0	0	1	0	0

IORWF Komutu

Bu komut da IORLW komutuna benzer olup fark, maskenin sabit sayı yerine bir file registerde bulunmuş olmasından ibarettir. Sonuç ise ister W 'ye , ister File Register 'a yazılabilir.

Komut Formatı:

IORWF File Register d ← (Sonucun yeri, W ya da F)

XORLW Komutu

Bu komut ise W Registeri ile verilen sabit sayı arasında EXOR (Özel VEYA) işlemi gerçekleştirerek sonucu W 'ye yazar. Komut Formatı:

XORLW Sabit Sayı şeklindedir.

Bu komut da W Registerde bulunan 8 bitlik sayının istenen bitlerinin tersini almak diğer bitleri aynen bırakmak için kullanılabilir. Bu maksatla belirlenecek Maske baytında tersi alınmak istenen bitler (1), diğerleri (0) seçilmelidir.

Örnek: Önce W 'ya $(3B)_{16}$ sayısını yükledikten sonra bu sayının sadece 1. , 3. , 5. , 7. bitlerinin tersini alıp neticeyi PORTA ya yazmak için gerekli komutları verin. Bu durumda sonuç ne olur?

```
MOVLW    h'3B'           ;W ye 3b yaz
XORLW    b'10101010'    ; maskede 7,5,3,1. bitler 1, yapıldı
MOVWF    PORTA          ; Sonuç PORTA ya yazıldı.
```

W ile sabit sayı arasında (EXOR işlemi)

Bit No:	7	6	5	4	3	2	1	0
W Reg.de(3B)	0	0	1	1	1	0	1	1
Maske (AA)	1	0	1	0	1	0	1	0
Sonuç (91)	1	0	0	1	0	0	0	1

XORWF Komutu

Bu komut da XORLW komutuna benzer olup burada asıl sayı W registerde, maske ise file registerde bulunur. İşlem sırası önemli olmadığı için maskeyi W'ya asıl sayıyı File Registerde yazmak da mümkündür.

NOT: XORLW ve XORWF iki tane 8 bitlik sayının aynı olup olmadıklarını test etmek için kullanılabilir. Sayılar aynı ise EXOR lanınca sonuç $(00)_{16}$ olacağından Z bayrağı (1) olacaktır. Aksi takdirde Z bayrağı (0) kalacaktır. [Z bayrağının STATUS 'un 2. biti olduğunu hatırlayın]

Örnek: PORTA da bulunan bir baytlık sayı $(09)_{16}$ dan farklı ise A Portu test edilmeye devam edilecek, $(09)_{16}$ 'a eşit ise bu sayı PORTB 'ye yazılms isteniyor. Gerekli Program Parçası:

```

MOVLW    h'09'
TEST
XORWF    PORTA,W    ; W ile PortA ya Lojik EXOR işlemi uygula
BTSS     STATUS,2   ; Z bayrağı 1 mi?
GOTO     TEST        ; Değilse TEST 'e git
MOVF     PORTA , W   ; Evetse PortA yı W ye aktar.
MOVWF    PORTB       ; W yi PortB ye yaz.

```

Mesela, PORTA ya gelen sayı (07)₁₆ olsa ve (09)₁₆ ile XORWF işlemi yapıldığında;

```

0000 0111    (07)
0000 1001    (09)
0000 1110    (0E)    Sonuç 0 'dan farklı olduğuna göre test etmeye devam et.

```

Benzer şekilde bir baytlık veriyi sıfır sayısı ile karşılaştırmak için IORLW ve IORWF komutları kullanılabilir.

Örnek: PORTA daki sayının (00)₁₆ olup olmadığını test eden, sıfır ise ILERI adresine atlayan aksi takdirde test etmeye devam eden bir program parçasını IOR türü bir komutla yazın.

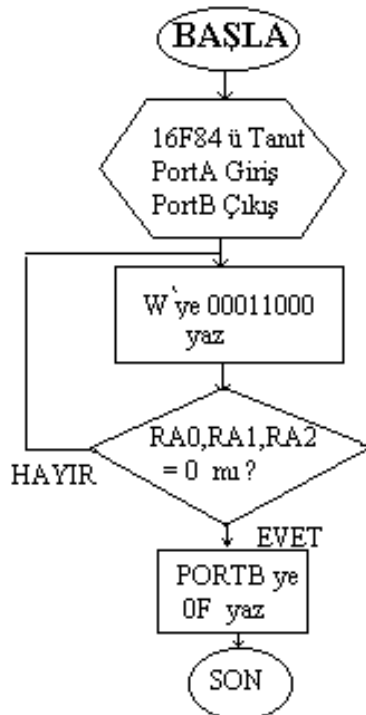
```

TEST
MOVF     PORTA , W   ; PortA yı W ye kopyala (yaz)
IORLW    h'00'       ; W ile 00 arasında Lojik VEYA uygula
BTSS     STATUS , 2 ; Sonuç=0 mı ( Z=1 mi?)
GOTO     TEST        ; Değilse TEST 'e git
GOTO     ILERI       ; Evetse ILERI adresine git.

```

Program Örneği 13 : PORTA nın bütün bitleri normalde (1) iken PORTA da 0,1,2. bitlerin (RA0,RA1,RA2) hepsine basılınca PORTB ye (0F)₁₆ yükleyen aksi halde test etmeye devam eden bir program için akış diyagramını çizerek assembly programı yazınız.

Maske: 0001 1000 [ilk 3 bit(0) PORTA da yok, son üç bit(0) test edilecek]



```

; PROGRAM13.ASM-----TARİH
LIST     P=16F84
INCLUDE  "P16F84.INC"
CLRF     PORTB
BSF      STATUS,5
MOVLW    h'FF'
MOVWF    TRISA    ;PortA Giriş
CLRF     TRISB    ;PortB Çıkış
BCF      STATUS,5 ;Bank0 a geç

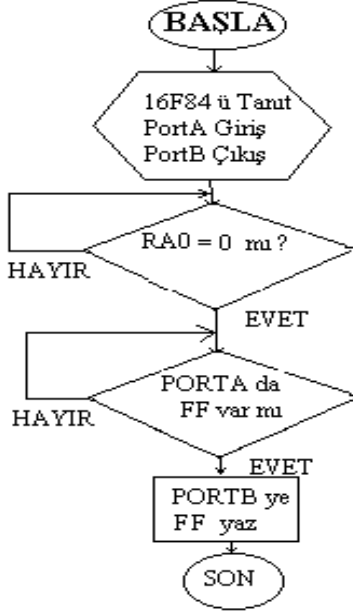
TEST
MOVLW    b'00011000' ;Maskeyi W ye
XORWF    PORTA,W    ;PortA ile EXOR
BTSS     STATUS,2   ; Z bayrağı 1 mi?
GOTO     TEST        ;Değilse TEST e
MOVLW    h'0F'      ;Evetse W ' ye 0F yaz
MOVWF    PORTB      ; PortB ye aktar
END

```

ÇEŞİTLİ ÖRNEKLER

1) PORTA 'nın 0. bitine (RA0) Lojik 0 uygulanarak enerji verilen bir PIC16F84'e daha sonra PORTA 'nın tamamına (FF)₁₆ uygulanırsa PORTB nin tüm bitleri (1) yapılacak aksi takdirde PORTA test edilmeye devam edilecektir. Akış diyagramını çizerek gerekli assembly programını yazınız.

; ORNEK1.ASM



```

LIST      P=16F84
INCLUDE   "P16F84.INC"
CLRF     PORTB
BSF      STATUS,5
MOVLW    h'FF'
MOVWF    TRISA
CLRF     TRISB
BCF      STATUS,5

TEST

BTFSK    PORTA,0 ; RA0=0 mı?
GOTO     TEST    ; Değilse TEST'e

BEKLE

XORWF    PORTA,W ; W ile PORTA aynı mı?
BTFSK    STATUS,2 ; Z bayrağı 1 mi?
GOTO     BEKLE  ; Değilse BEKLE 'ye

MOVLW    h'FF'
MOVWF    PORTB   ; Aynıysa PORTB ye FF yaz
END      ; Son
  
```

2) PORTA nın 1. bitine (RA1) 2 kere üstüste bir Lojik devreden (0) uygulandığı takdirde W Registerine (01)₁₆ yazıp son bulan bir program yazınız. Tuşa basılmalar arasında 10 tane NOP komutunun yaklaşık 255 defa icrası ile elde edilecektir. Gecikmeden hemen sonra ikinci (0) işareti gelmediği halde test işlemine baştan başlanacaktır. (Yani, yeniden iki (0) beklenecektir, Gecikme için Alt Program kullanılmayacaktır.) Gerekli assembly programını yazınız.

;ORNEK2.ASM

```

LIST      P=16F84
INCLUDE   "P16F84.INC"

SAYAC    EQU    h'0C'      ; SAYAC için 1 bayt yer ayır.
BSF      STATUS,5        ; Bank1 'e geç
MOVLW    h'FF'
MOVWF    TRISA          ; PORTA yı Giriş yap
BCF      STATUS,5        ; Bank0 'a geç

TEST

BTFSK    PORTA,1        ; RA1=0 mı?
GOTO     TEST          ; Değilse TEST 'e git
MOVLW    h'FF'          ; Evetse W 'ye FF yükle
MOVWF    SAYAC          ; Sayaca W deki FF sayısını yaz

GECIKME

NOP
NOP
NOP
  
```

```

NOP                ; 10 tane NOP komutu
DECFSZ    SAYAC,F  ; Sayacı 1 azalt
GOTO      GECIKME  ; SAYAC sıfırdan farklıysa GECIKME'ye
BTFSC     PORTA,1  ; SAYAC sıfırda, tekrar RA1=0 mı? bak
GOTO      TEST     ; Değilse TEST 'e git, yeniden iki 0 bekle
MOVLW    h'01'    ; İstenen (01) sayısını W 'ye yükle
END                ; Son

```

- 3) PORTA 'da bulunan 8 bitlik (1 baytlık) sayı 2 kere sola ötelendikten sonra ortaya çıkan sayı $(6C)_{16}$ ise PORTB 'ye (01) ; aksi takdirde yine PORTB 'ye (02) yazan bir programı assembly dilinde yazınız.

NOT: Başlangıçta C bayrağı (0) yapılacaktır.

;ORNEK3.ASM

```

LIST      P=16F84
INCLUDE   "P16F84.INC"
BSF       STATUS,5
MOVLW    h'FF'
MOVWF    TRISA      ; PortA Giriş
CLRF     TRISB      ; PortB Çıkış
BCF      STATUS,5   ; Bank0 'a geç
BCF      STATUS,0   ; C Bayrağı sıfır yapıldı.
RLF      PORTA,F    ; PortA 'yı 1 bit sola ötele
RLF      PORTA,F    ; PortA 'yı 1 bit (kere daha) sola ötele
MOVLW    h'FF'      ; W registerine FF yükle
XORWF    PORTA,W    ; W deki (FF) ile PortA daki sayı aynı mı?
BTFSS    STATUS,2   ; Z bayrağı 1 ise bir satır atla.
GOTO     GIT        ; Değilse GIT etiketine git.
MOVLW    h'01'
YAZ      MOVWF     PORTB ; PortB ye (01) yaz
GOTO     SON        ; SON etiketine git.
GIT      MOVLW     h'02'
MOVWF    PORTB      ; PortB ye (02) yaz
SON      END

```

KAYNAKLAR:

- 1- Mikroişlemciler Ders Notları 1 - 2 (6502) , Doç. Dr. Hakan ÜNDİL
- 2- Mikrodenetleyiciler ve PIC Programlama, Orhan ALTINBAŞ
- 3- Adım Adım PIC Programlama, Yaşar BODUR
- 4- PIC Microcontroller Uygulama Devreleri, Gökhan DİNÇER
- 5- PIC16F8X, Microchip PIC Data Sheet, (www.microchip.com)