Dallanma ifadelerinde Büyüktür/Büyüktür veya Eşittir ifadeleri:

3. Büyük ise Dallan İfadesi

JG (JUMP GREATER) /JNLE (JUMP NOT LESS OR EQUAL) // JG=JNLE (özdeş ifadeler)

İSİM BİÇİM ANLAM

JG JG ETİKET Büyük ise ETİKET'e dallan

JNLE JNLE ETİKET Küçük değil veya eşit değil ise ETİKET'e dallan

4. Büyük veya Eşit ise Dallan İfadesi

JGE (JUMP GREATER OR EQUAL) /JNL (JUMP NOT LESS ) // JG=JNL (özdeş ifadeler)

İSİM BİÇİM ANLAM

JGE JGE ETİKET Büyük veya eşit ise ETİKET'e dallan

JNL JNL ETİKET Küçük değil ise ETİKET'e dallan

ÖRNEK: 10000H adresinden başlamak üzere "F0H" değerinden büyük veya eşit olan ilk 100 adet 8 bitlik sayı değerlerinin kaç adet olduğunu bulup, bunu AL yazmacına yazdıran 8086 komut kümesini yazınız.

MOV DX,1000H

MOV DS,DX

MOV AL,00H

MOV SI, 0000H // [SI]=10000H başlangıç için; ayrıca SI sayaç olarak kullanılabilir.

DEVAM

MOV BL, [SI]

CMP BL,F0H

JGE ARTTIR

INC SI // ofset bir arttır.

CMP SI, 64H

JE BITIS

JMP DEVAM

ARTTIR: INC AL

INC SI

CMP SI,64H

JNE DEVAM

BITIS HLT

ÖRNEK: 20000H adresinden başlamak üzere toplam 100 adet 8 bitlik sayı denetlenecektir. Buna göre kaç adet "0" dğeri varsa BL yazmacına yazılacak. Ayrıca sıfır olan değerler 10H ile toplanacaktır.

MOV DX, 2000H

MOV DS,DX

MOV DI,0000H

MOV BL,00H

DEVAM

MOV CL, [DI] // 20000H adresinden ilk değeri denetim için CL'ye yüklendi.

CMP CL, 00H

JE ARTTIR

INC DI

CMP DI,64H

JE BITIS

JMP DEVAM

ARTTIR INC BL

ADD [DI],10H

INC DI

CMP DI,64H

JNE DEVAM

BITIS HLT

5. Elde var ise Dallan İfadesi

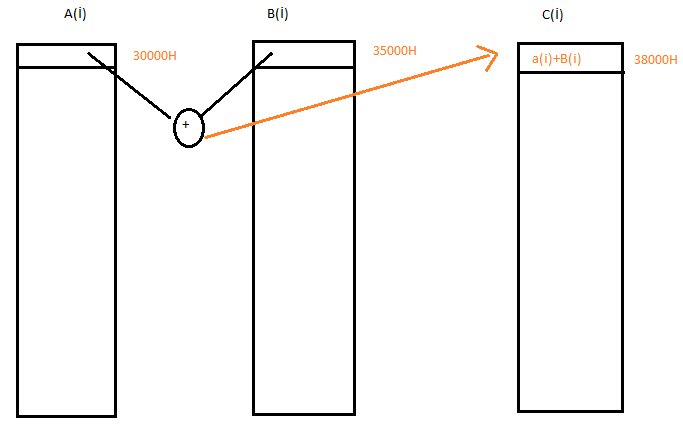
Bu ifade bir işlem sonucunda elde durumuna göre dallanma belirtiilen etikete dallanma yapmaktadır.

İSİM BİÇİM ANLAM

JC (Jump if Carry) JC ETIKET Elde var ise belirtilen etikete dallanma yapılır.

JNC (Jump if Not Carry) JNC ETIKET ELde yok ise dallan

ÖRENK: A(i) ve B(i) olmak üzere iki farklı dizi bulunmaktadır. Her bir dizide toplam 100 adet 8 bitlik değerler yüklenmiştir. Bu sayı değerlerini karşılıklı olarak sırası ile toplayıp, C(i) dizisine sırası ile yazdıran 8086 komut kümelerini yazınız. Eğer herhangi bir toplama sonucunda elde durumu olursa, program terk edilecektir. Buna göre A(i) dizisi 30000H adresinden B(i) dizisi 35000H adresinden ve C(i) dizisi ise 38000H adresinden başlamaktadır.



MOV DX, 3000H

MOV DS,DX

MOV SI,0000H // A(İ) için ofset, aynı zamanda 100 adet sayı sayacı olarak kull.

MOV DI,5000H // B(İ) için

MOV BX, 8000H // C(İ) için

DEVAM

MOV CL, [SI]

ADD CL, [DI]

JC BITIS

MOV [BX],CL

INC DI

INC BX

INC SI

CMP SI,64H // Aynı zamanda sayaç dı, 100 mü??

JE BITIS

JMP DEVAM

BITIS HLT

ÖRNEK: A(İ) dizisinde toplam 100 adet 8 bitlik sayı bulunmakatadır. Bu sayıları tarayıp ilk "0" değerine sahip hafıza lokasyonunun ofset adresini AL yazmacına yükleyen 8086 komut kümelerini yazınız. Eğer sıfır değeri 100 adet sayı değerleri arasında yok ise, BL yazmacına FFH değeri yüklenecektir.

NOT: A(İ) dizisi FD300H adresinden başlamaktadır.

MOV DX, F000H

MOV DS,DX // Taban şu an F0000H

MOV SI, D300H

DEVAM MOV DL, [SI]

CMP DL,00H // Sıfır olma durumu kontrol ediliyor.

JE YUKLE

INC SI

CMP SI, D364H

JE SON

JMP DEVAM

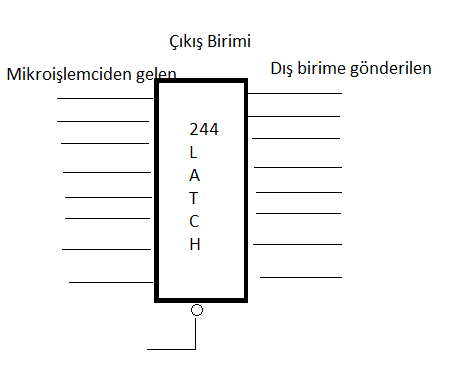
YUKLE AL,SI

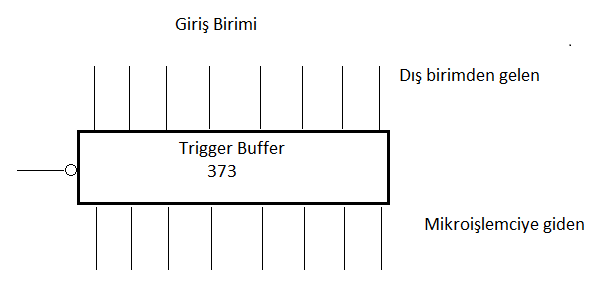
JMP BITIS

SON BL,FFH

BITIS HLT

8086 Komut Kümelerinde Giriş/Çıkış ifadeleri





Giriş İfadeleri

|  |  |  |
| --- | --- | --- |
| İsim | Biçim | Anlam |
| IN AL, p8  p8: 8 bitlik port demektir. | ör: IN AL,FEH | 8 bitlik fiziksel adres değerine sahip port üzerinden 8 bit değer alma.  NOT: Giriş/ Çıkış birimleri ile sadece akümülatör üzerinden haberleşilir( değer gönderilir yada alınır). Eğer 8 bit değer alınacaksa AL, kullanılmalıdır.  Eğer 16 bit değer alınacaksa AX, kullanılmalıdır. |
| IN AL, DX | Ör: MOV DX, FF0EH  IN AL,DX | 16 bitlik fiziksel adres değerine sahip port üzerinden 8 bit değer alma |
| IN AX,p8 | Ör: IN AX,E4H | 8 bitlik giriş portu üzerinden 16bit değer alma. |
| IN AX,DX | ÖR: MOV DX, 1234H  IN AX,DX | 16 bitlik fiziksel adres değerine sahip port üzerinden 16 bit değer alma.  NOT: 16 bit değer alma olduğu için AX kullanılmalıdır. |

ÖRNEK: FEH adresli giriş biriminden, 8 bitlik değer alıp, bunu 2 ile çarptıktan sonra 20000H adresine yükleyen 8086 komut kümelerini yazınız.

MOV DX, 2000H

MOV DS,DX // 20000H tabanındayız.

IN AL, FEH

MOV BL, 02H

MUL BL // AX=AL\*BL

MOV [0000H], AX // Sonucun 20000H adresine yüklenmesi.

ÖRNEK: FFEEH adresli giriş portu üzerinden 16 bitlik sayı değerini alıp, 2 eksilttikten sonra 34000H adresli hafıza lokasyonuna kayıt eden 8086 komut kümelerini yazınız.

MOV DX, 3400H

MOV DS,DX // 34000H tabanındayız.

MOV DX, FFEEH // IN AX, FFEEH yazılırsa derleme hatası verir.

IN AX, DX

SUB AX,02H

MOV [0000H], AX

ÖDEV: 16 bitlik bir adres değerine sahip giriş biriminden 15 adet 8 bitlik değer tarayıcıdan okunacaktır. Bu değerlerin sırasıyla tersinin iki katı alındıktan sonra, 78000H adresinden başlamak üzer sırası ile ilgili hafıza lokasyonlarına yerleştiren 8086 komut kümelerini yazınız.

NOT: Giriş birimin değeri FF02H alınmıştır.

MOV DX, 7800H

MOV DS,DX // 78000H tabanındayız.

MOV SI,0000H

MOV CL,0FH // 0Fh---- 15 dir.

MOV DX,FF02H

DEVAM: IN AX,DX // Değerlerin okunması

NOT AX

MOV CX,AX

ADD AX,CX // Tersinin iki katı alınmış oluyor. AX=AX+CX

MOV [SI],AX // sonucun hafızaya yazılması.

INC SI // Bir sonraki hafıza lokasyonuna geçmek için SI arttırıldı.

DEC CL

JZ BITIS

JMP DEVAM

BITIS: HLT

ÖDEV: FE0EH adresli giriş birimine bağılı klavye üzerinden tek tek 100 adet 8 bitlik sayısal değerleri girilmektedir. Buna göre girilen çift sayı değerlerini 80000H adresinden, tek sayı değerlerini ise 88000H adresli hafıza lokasyonuna (çift yada tek olma durumuna bağılı) yükleyen 8086 komut kümesini yazınız.

Sayı tek ise; en düşük seviyeli biti “1” değerine sahiptir.

Sayı çift ise; en düşük seviyeli biti “0” değerine sahiptir.

IN AX,DX

MOV CX,AX

AND CX,01H // Eğer AX deki sayı tek ise AX 01H’ a eşit olur, çift ise 00H eşit olur.

………..

CX:03H

0000 0011B

0000 0001B

0000 0001B=01H // Sayı Tekdir.

CX:08H

0000 1000B

0000 0001B (01H)

0000 0000B = 00H // Sayı Çiftdir.

MOV DX, 8000H

MOV DS,DX // 80000H tabanı oluştu.

MOV CL,64H // Genel sayaç, 100 sayıyı saymak için.

MOV SI,0000H // Çift sayıların yüklenecek hafıza bölümü için ofset adres tanımlama.

MOV DI, 8000H // Tek sayıların yüklenecek hafıza bölümü için ofset adres tanımlama.

MOV DX,FE0EH // Giriş birimin 16 bitlik adres değerinin yüklenmesi.

DEVAM: IN AL, DX // Sıradaki 100 adet sayı değerlerinden birinin AL’ye yüklenmesi.

MOV BL,AL // Orijinal sayı değerinin BL’ye yedeklenmesi.

AND AL, 01H // Tek mi, Çift mi test etmek için 01H değeri ile AND’lenmesi.

CMP AL,00H // Eşit ise sayı çift dir.

JE CIFT

MOV [DI],BL // Teksayının ilgili hafıza lokasyonuna yüklenmesi.

INC DI // Bir sonraki teksayı hafıza lokasyonu için hazır hale getirme.

DEC CL

JNZ DEVAM

JMP BITIS

CIFT: MOV [SI],BL

INC SI

DEC CL

JNZ DEVAM

BITIS: HLT

ÖRNEK: Giriş birimin adresi FFEEH dır. Bu giriş birimine bir adet klavye bağlanmıştır. Giriş biriminden girilen 8 bitlik sayıların, kaç adedi çift kaç adedi tek olduğunu bulan ve bunu ilgili hafıza lokasyonuna yükleyen 8086 komut kümesini yazınız. Toplam çift sayı adedi 30000H tek sayı adedi ise 30001H fiziksel adresine yüklenecektir. Toplam 100 adet sayı girişi yapıldıktan sonra program sonlanacaktır.

MOV DX, 3000H

MOV DS,DX

MOV CL,64H

MOV SI,0000H // Çiftsayı sayacı

MOV DI,0000H // Teksayı sayacı

MOV DX, FFEEH

DEVAM: IN AL,DX // sayı değerin klavyeden alınması

AND AL,01H

CMP AL, 01H

JE TEK

INC SI

DEC CL

JNZ DEVAM

JMP BITIS

TEK: INC DI

DEC CL

JNZ DEVAM

BITIS: MOV [0000H],SI //30000H adresine çift sayı adedi yüklendi

MOV [0001H],DI // 30001H adresine tek sayı adedi yüklendi.

HLT

8086 ‘da çıkış ifadeleri:

Giriş ifadelerine benzer şekilde, 8 bitlik adres değerleri doğrudan yazılırken, 16 bitlik adres değerleri DX yazmacı üzerinden ifade edilmektedir.

|  |  |  |
| --- | --- | --- |
| İsim | Biçim | Anlam |
| OUT p8,AL  p8: 8 bitlik port demektir. | ör: OUT FEH,AL | 8 bitlik fiziksel adres değerine sahip port üzerinden 8 bit değer gönderme.  NOT: Giriş/ Çıkış birimleri ile sadece akümülatör üzerinden haberleşilir( değer gönderilir yada alınır). Eğer 8 bit değer alınıp/gönderilecekse AL, kullanılmalıdır. Eğer 16 bit değer alınıp/gönderilecekse AX, yazmacı kullanılmalıdır. |
| OUT DX,AL | Ör: MOV DX, FF0EH  OUT DX,AL | 16 bitlik fiziksel adres değerine sahip port üzerinden 8 bit değer gönderilmektedir. |
| OUT p8,AX | Ör: OUT E4H,AX | 8 bitlik giriş portu üzerinden 16 bit değer gönderme. |
| OUT DX,AX | ÖR: MOV DX, 1234H  OUT DX,AX | 16 bitlik fiziksel adres değerine sahip port üzerinden 16 bit değer gönderme.  NOT: 16 bit değer gönderme olduğu için AX kullanılmalıdır. |

ÖRNEK: 30000H adresinden başlamak üzere her biri 8 bitlik toplam 15 adet sayı değerini E0FFH adresli çıkış birimine bağılı yazıcıya gönderen 8086 komut kümesini yazınız.

MOV DX,3000H

MOV DS,DX

MOV DX,E0FFH

MOV CL,0FH // Genel sayı sayacı

MOV SI,0000H // Hafızadan sayıları okuyabilme için ofset adresi tanımlama.

DEVAM:MOV AL,[SI]

OUT DX,AL // Sayı değeri yazıcıya gönderildi.

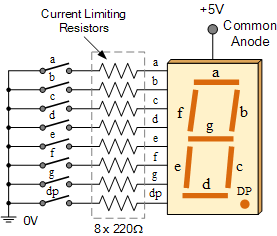
INC SI // Bir sonraki sayı değeri okuma için ofset adresi arttırma.

DEC CL // Genel sayaç değeri eksiltme.

JNZ DEVAM

BITIS: HLT

ÖRNEK: 8086 mikroişlemcili bir sistemde 40000H adresinden başlamak üzere toplam 100 adet 8 bitlik sayı yüklenmiştir. Buna göre her okunan sayının çift yada tek olma durumuna bağılı olarak FEH adresli çıkış birimine bağılı 7\_SEG\_LED üzerine okunan son sayının çift yada tek olduğunu gösteren uygulamayı, 8086 komut kümelerini kullanarak yazınız. Eğer sayı Tek ise 7\_Seg\_Led üzerinde “O” harfi (ODD), eğer sayı çift ise “E” harfi (EVEN) yazılacaktır. (İnsan gözünün son okunan değeri LED üzerinde algılaya bilmesi için geciktirme kodu kullanınız.) Led’ler genel Anoda göre bağlanmıştır (‘0’ yakar,’1’ yakmaz).



a b c d e f g dp

0 0 0 0 0 0 1 1 03H (“O”)

0 1 1 0 0 0 0 1 61H (“E”)

MOV DX, 4000H

MOV DS,DX // Taban 40000H da.

MOV BL,64H // Genel sayı sayacı

MOV DI,0000H // Ofset tanımlama

TEKRAR: MOV AL, [DI]

AND AL,01H

CMP AL, 00H // Sayının tek olup olmaması denetleniyor.

JE CIFT\_YAZ

MOV AL, 03H // 0 0 0 0 0 0 1 1 =03H “O” harfi yazdırma kodu.

OUT FEH,AL // “O” harfi 7\_LED üzerine yazıldı. //

INC DI // Sonraki hafıza lokasyonundaki değeri okuyabilmek için.

MOV CX,FFFFH // LED’ e zaman kazandırmak için.

DEVAM: DEC CX

JNZ DEVAM

DEC BL

JNZ TEKRAR

JMP BITIS

CIFT\_YAZ: MOV AL,61H

OUT FEH, AL // 7\_Seg\_Led’e “E” harfi yazacaktır.

INC DI

MOV CX,FFFFH

DEVAM: DEC CX

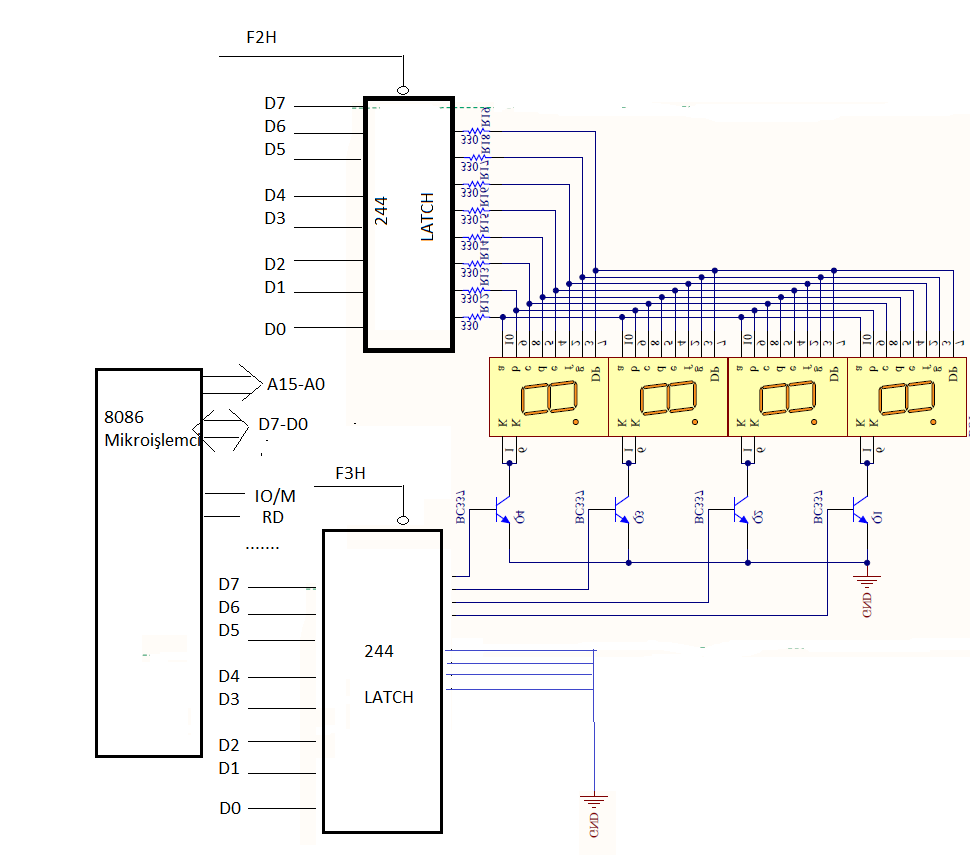
JNZ DEVAM

DEC BL

JNZ TEKRAR

BITIS: HLT

ÖRNEK: 8086 mikroişlemcili bir sistemde 4 adet 7\_Seg\_Led şekildeki gibi bağlanmıştır. Buna göre sırası ile 7 Seg\_Led üzerine “7,5,3,2” rakamlarını yazdıran 8086 komut kümesini yazınız. (led’ler genel Anoda göre bağlanmıştır.)



1. LED için:

1.adım F2H adresli çıkış biriminden F8H değeri gönderilir.

D7 D6 D5 D4 D3 D2 D1 D0 dp g f e d c b a

1 1 1 1 1 0 0 0 F8H= “7” rakamı yazdırır.

1. Adım F3H adresli çıkış biriminden 80H değeri gönderilir.

D7 D6 D5 D4 D3 D2 D1 D0

1. 0 0 0 0 0 0 0 80H // LED 1 açık

2. LED için:

1.adım F2H adresli çıkış biriminden 49H değeri gönderilir.

D7 D6 D5 D4 D3 D2 D1 D0

0 1 0 0 1 0 0 1 49H= “5” rakamı yazdırır.

1. Adım F3H adresli çıkış biriminen 40H değeri gönderilir.

D7 D6 D5 D4 D3 D2 D1 D0

0 1 0 0 0 0 0 0 40H // LED 2 açık

3. LED için:

1.adım F2H adresli çıkış biriminden ODH değeri gönderilir.

D7 D6 D5 D4 D3 D2 D1 D0

1. 0 0 0 1 1 0 1 0DH= “3” rakamı yazdırır.

2.Adım F3H adresli çıkış biriminen 20H değeri gönderilir.

D7 D6 D5 D4 D3 D2 D1 D0

0 0 1 0 0 0 0 0 20H // LED 3 açık

4. LED için:

1.adım F2H adresli çıkış biriminden 25H değeri gönderilir.

D7 D6 D5 D4 D3 D2 D1 D0

0 0 1 0 0 1 0 1 25H= “2” rakamı yazdırır.

2.Adım F3H adresli çıkış biriminen 40H değeri gönderilir.

D7 D6 D5 D4 D3 D2 D1 D0

0 0 0 1 0 0 0 0 10H // LED 4 açık

MOV AL, F8H

OUT F2H, AL

MOV AL,80H

OUT F3H, AL

------------------

MOV AL, 49H

OUT F2H, AL

MOV AL,40H

OUT F3H,AL

----------------

MOV AL,0DH

MOV F2H, AL

MOV AL,20H

OUT F3H,AL

----------------

MOV AL,25H

OUT F2H,AL

MOV AL, 10H

OUT F3H,AL

ÖDEV: 8086 mikroişlemcili bir sistemde toplam 7 adet 8 bitlik sayı 45FFH adresine bağılı klavyeden alınıp sırası ile 7 adet 7 Segmenlı Ledler üzerine yazdırılacaktır. Buna göre LED üzerine 0 dan 9 kadar olan sayı rakamlarını yazdırma kodları 30000H ile 30009H arasında yüklü bulunmaktadır. Buna göre klavyeden girilen her sayının rakamsal değerini sırası ile “1” led’den başlamak üzere yazdıran 8086 komut kümesini yazınız. NOT: led’lere değer göndermeyi sağlayan LATCH’lerin fiziksel adresleri sırası ile EEH ve EFH dır.

### 1) ROL : Rotate Left (Sola Döndür)

The ROL instruction is an abbreviation for ‘Rotate Left’. This instruction rotates the mentioned bits in the register to the left side one by one such that leftmost bit that is being rotated is again stored as the rightmost bit in the register, and it is also stored in the Carry Flag (CF).

Syntax: ROL Register, Bits to be shifted

Example:

MOV CL,04H

ROL AH, CL

Working:



ÖRNEK:

MOV CL, 05H

MOV DX,FF03H

ROL DX,CL

DX=? CF=?

Baş; DX= 1111 1111 0000 0011B CF=?

1Adım: DX= 1111 1110 0000 0111B CF=01H

2.Adım: DX= 1111 1100 0000 1111B CF=01H

3.Adım: DX= 1111 1000 0001 1111B CF=01H

4.Adım: DX= 1111 0000 0011 1111B CF=01H

5. Adım DX=1110 0000 0111 1111B CF=01H

CVP:

DX: E07FH CF=01H

### 2) ROR : Rotate Right (Sağ Döndür)

The ROR instruction stands for ‘Rotate Right’. This instruction rotates the mentioned bits in the register to the right side one by one such that rightmost bit that is being rotated is again stored as the MSB in the register, and it is also stored in the Carry Flag (CF).

Syntax: ROR Register, Bits to be shifted

MOV CL, 04H

Example: ROR AH,CL

Working:



ÖRNEK: CF=01H

MOV CL,02H

MOV BX, 300FH

ROR BX,CL

BX=? CF=?

Baş: BX= 0011 0000 0000 1111H CF=01H

1. Adım: BX= 1001 1000 0000 0111 CF=01H
2. Adım: BX= 1100 1100 0000 0011 CF=01H

BX= CC03H CF=01H

### 3) RCL : Rotate Carry Left (Eldeden geçirerek sola döndür)

This instruction rotates the mentioned bits in the register to the left side one by one such that leftmost bit that is being rotated it is stored in the Carry Flag (CF), and the bit in the CF moved as the LSB in the register.

Syntax: RCL Register, Bits to be shifted

Example: RCL CH, 1

Working:



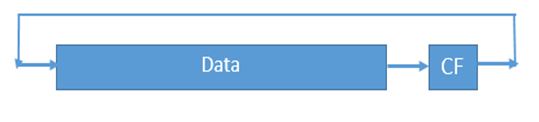
### 4) RCR : Rotate Carry Right (Eldeden geçirerek sağ döndür)

This instruction rotates the mentioned bits in the register to the right side such that rightmost bit that is being rotated it is stored in the Carry Flag (CF), and the bit in the CF moved as the MSB in the register.

Syntax: RCR Register, Bits to be shifted

Example: RCR BH, 6

Working:



ÖRNEK: CF başlangıç değeri 01H

MOV CL,04H // CF=01H

MOV AX, 1234H

RCL AX, CL

RCR AX,1

CF=? AX=?

Baş: AX=0001 0010 0011 0100B CF=01H

1. Adım: AX= 0010 0100 0110 1001B CF=00H

2Adım: AX= 0100 1000 1101 0010B CF=00H

3.Adım: AX= 1001 0001 1010 0100 CF=00H

4.Adım AX= 0010 0011 0100 1000B CF=01H

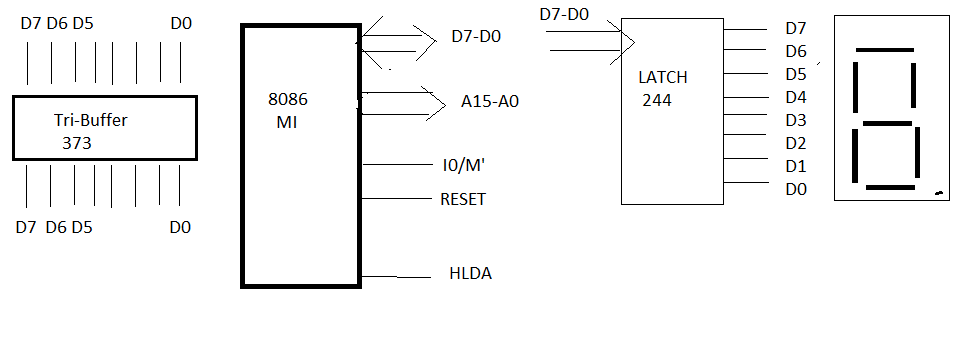
AX= 2348H CF=01H

ÖRNEK: Klavyeden girilen toplam 100 adet sayı değerini okuyup sırası ile sayı değerlerini 20000H adresinden başlamak üzere yüklemektedir. Eğer sayı çift sayı ise, çıkış birimine bağılı 7 Seg\_Led üzerine “E” eğer yüklenen sayı değeri tek sayı ise 7 Seg\_Led üzerine “O” harfini yazdıran 8086 komut kümesini yazınız.

Not: Sayıların herbiri 8 bittir. 7Seg\_Led’in bağılı bulunduğu çıkış adresi FFDCH dır. Giriş birimi adresi ise E3H dır.

NOT: Bir sayının en düşük seviyeli biti ‘0’ ise O sayı çift sayıdır, 1 ise sayı tek sayıdır.

NOT: 7 Seg\_Led genel Anoda göre tasarlanmıştır, ’0’ yakar ‘1’ yakmaz.



a b c d e f g dl

0 1 10 0 0 1 1B = 63H….. ‘E’ 0 0 0 0 0 0 1 1B=03H…….’O’

MOV SI,0000H //ofset adres yüklendi, ayrıca sayaç olarak kull.

MOV DX,2000H

MOV DS,DX // 20000H tabanı oluştu.

MOV DX,FFDCH // Çıkış birimi adresi 16bit olduğu için DX’e yüklendi.

DEVAM IN AL, F3H // Klavyden değer alınıyor

MOV[SI], AL // Değer hafızaya yüklendi.

INC SI // hem ofset hem sayaç arttırma

CMP SI, 0064H // 100’ü geçti mi? denetleniyor.

JE BITIS

ROR AL, 1 // Çift yada tek mi? Test!!!

JNC CIFTYAZ // JC yazılsaydı, elde olma durumu denetlenecekti.

MOV AL, 03H

OUT DX, AL // / Seg\_Led’de artık ‘O’ yazıyor.

JMP DEVAM

CIFTYAZ MOV AL, 63H

OUT DX, AL

JMP DEVAM

BITIS HLT

ÖRNEK: 8086 mikroişlemcili bir sistemde 7 Segmanlı Led üzerine sayı değeri yazdırma uygulaması yapılacaktır. Buna göre ‘0’ dan başlamak üzere ‘A’ya kadar 7 Seg\_Led üzerine yazılacak sayı kodları 10000H ile 10009H arasında yüklenmiştir. Buna göre Klavyeden girilen 10 adet sayılardan, kaçıncı sayının girildiğini 7 Seg\_Led üzerinde gösteren 8086 komut kümesini yazınız.

Not: Sayıların herbiri 8 bittir. 7Seg\_Led’in bağılı bulunduğu çıkış adresi FFDCH dır. Giriş birimi adresi ise E3H dır.

NOT: 7 Seg\_Led genel Anoda göre tasarlanmıştır, ’0’ yakar ‘1’ yakmaz.

MOV DX, 1000H

MOV DS,DX // 10000H tabanı yüklendi.

MOV DI, 0000H // Hem sayaç hem ofset adresi

MOV DX,FFDCH

DEVAM IN AL,E3H

MOV AL, [DI] // AL ye 7 Seg\_Led’e ilgili rakam yazdırma kodu alındı.

OUT DX, AL // Kaçıncı sayı girişini gösteren rakam 7 Seg\_Led gönderildi.

INC DI

CMP DI, 0AH

JE BITIS

JMP DEVAM

BITIS HLT

ÖRNEK: Klavyeden girilen son iki adet sayı değerlerini toplayıp, sonuçlarını 20000H adresinden başlamak üzere yükleyen ve sonucun çift yada tek sayı olduğunu F2H adresli çıkış birimine bağılı 7\_Seg\_Led üzerinde gösteren 8086 sistemi için TUS\_TOPLA\_2 adında bir procedür programı yazınız. Led’ler genel anoda göre bağlanmıştır. Klavye EE02H adresli giriş biriminden değer göndermektedir. Toplanacak sayıların her biri 8 bittir. Toplam 100 adet toplama işlemi yapıldıktan sonra Procedür Programı sonlanacaktır.

PROC NEAR TUS\_TOPLA\_2

MOV DX, 2000H // taban

MOV DS,DX

MOV SI,0000H

MOV BL,00H // Birinci sayı girişi için temizleme.

MOV CL, 00H // İkinici sayı giriş için temizleme.

DEVAM IN AL,F2H // ilk sayı alındı

MOV BL, AL

IN AL, F2H // İkinci sayı değeri alındı.

MOV CL,AL

ADD BL, CL // BL=BL+CL, toplam sonucu BL de dir.

MOV [SI], BL // Toplam sonucu hafızaya yüklendi.

ROR BL,1 // Çift mit tek mi testi???

JNC CIFTYAZ

MOV AL,03H // Tek olduğunu gösterecek ‘O’ harfi AL yüklendi.

OUT F2H,AL // 7\_Seg\_Led’e ‘O’ harfi yazıldı.

JMP ARTTIR

CIFTYAZ MOV AL,63H

OUT F2H,AL // 7 SEG\_Led’e ‘E’ harfi yazıyor.

ARTTIR INC SI

CMP SI,64H // 100’ü geçtiysek döngüden çıkılır.

JNE DEVAM

ENDP

Alt Program Çağırma İfadesi CALL

CALL ifadesi kullanılarak herhangi bir alt program, ilgili programın adı ile çağrılır.

ÖR: CALL alt\_program\_LED

İlgili alt programın son satırında RET ifadesi ile ana programdaki çağrılmadan sonraki satıra ( ana programın ana satırın bir sonrasına) geri dönülür.

PROC NEAR TUS\_TOPLA\_2

MOV DX, 2000H // taban

MOV DS,DX

MOV SI,0000H

MOV BL,00H // Birinci sayı girişi için temizleme.

MOV CL, 00H // İkinici sayı giriş için temizleme.

DEVAM IN AL,F2H // ilk sayı alındı

MOV BL, AL

IN AL, F2H // İkinci sayı değeri alındı.

MOV CL,AL

ADD BL, CL // BL=BL+CL, toplam sonucu BL de dir.

MOV [SI], BL // Toplam sonucu hafızaya yüklendi.

CALL ALT\_PRO\_LED\_YAZ

INC SI

CMP SI,64H // 100’ü geçtiysek döngüden çıkılır.

JNE DEVAM

ENDP

ALT\_PRO\_LED\_YAZ

ROR BL,1 // Çift mit tek mi testi???

JNC CIFTYAZ

MOV AL,03H // Tek olduğunu gösterecek ‘O’ harfi AL yüklendi.

OUT F2H,AL // 7\_Seg\_Led’e ‘O’ harfi yazıldı.

JMP

CIFTYAZ MOV AL,63H

OUT F2H,AL // 7 SEG\_Led’e ‘E’ harfi yazıyor.

SON HLT

RET

ÖDEV: Klaveyden 0-9 sayı değerleri arasında girilen toplam 100 adet sayıların herbirinin hangi sayı değerine sahip olduğunu F3H adresli çıkış birimine bağılı 7\_Seg\_Led üzerine yazdıran porcedür programını yazınız. Ayrıca girilen her sayı 34000H adresinden başlamak üzere hafızaya kayıt edilecektir. Klavyenin bağılı bulunduğu giriş birimi adresi ise FF00H dır. Buna göre hafızaya kayıt etme işlemi bir kayit\_et adında bir alt program tarafından yapılacaktır.

NOT: 7\_Seg\_Led üzerine yazılacak sayı kodları 35000H dan başlamak üzere 35009H kadar yüklenmiştir. Sayı değeri girilen sayı ile eşlenip, hafızadan çekilip 7\_seg\_led üzerine yazılacaktır.