Bilgisayar Mimarisi

Bölüm 6

Temel Bilgisayarın Programlanması

Dr. Emre Ünsal

Cumhuriyet Üniversitesi

Bilgisayar Mühendisliği Bölümü

İçerik

- Giriş
- Makine Dili
- Birleştirici Dil
- Derleyici
- Program Döngüleri
- Aritmetik ve Mantıksal İşlemlerin Programlanması
- Alt Programlar
- Giriş/Çıkış Programlama

Giriş

- Bilgisayar Mimarisi ile ilgilenenler hem donanım hem de yazılım bilgisine sahip almalıdırlar. Çünkü her iki dalda birbirini tamamlamaktadır.

Sembol	*Hex Kod*	*Tanım*
AND	0 veya 8	M yi AC ile VE le
ADD	1 veya 9	M yi AC ekle, eldeyi E ye aktar
LDA	2 veya A	M yi AC ye yükle
STA	3 veya B	AC yi belleğe sakla
BUN	4 veya C	m ye şartsız dallan
BSA	5 veya D	geri dönüş adresini bellekte sakla ve m+1 dallan
ISZ	6 veya E	M yi 1 arttır eğer sıfı ise atla
CLA	7800	AC yi sil
CLE	7400	E sil
CMA	7200	AC yi tümle
CME	7100	E yi tümle
CIR	7080	AC ve E yi dairesel sağa kaydır
CIL	7040	AC ve E yi dairesel sola kaydır
INC	7020	AC yi 1 arttır
SPA	7010	AC pozitif ise sonraki buyruğu atla
SNA	7008	AC negatif ise sonraki buyruğu atla
SZA	7004	AC 0 ise sonraki buyruğu atla
SZE	7002	E 0 ise sonraki buyruğu atla
HLT	7001	Bilgisayarı durdur
INP	F800	Bilgiyi al ve bayrağı temizle
OUT	F400	Bilgiyi yolla ve bayrağı temizle
SKI	F200	Eğer giriş bayrağı varsa sonraki buyruğu atla
SKO	F100	Eğer çıkış bayrağı varsa sonraki buyruğu atla
ION	F080	Kesmeyi çalıştır
IOF	F040	Kesmeyi kapat

Temel bilgisayarı oluşturan 25 buyruk

m: efektif adres
M: efektif adresteki Bellek değeri

Makine Dili

- Program:

  İstenilen bir veriyi işlemek için bilgisayar tarafından icra edilen buyruklardır.

- Bilgisayar için yazılan programlar aşağıdaki sınıflardan birinde olabilir
  - Makine Kodu
    - İkili kod
    - Sekizlik veya Onaltılık kod
  - Sembolik kod (Assembly languages)
  - Yüksek seviyeli programlama dilleri (Derleyiciler)
    - C, C++, Java vs.

Programlama Dillerinin Karşılaştırılması

İki sayının toplamını veren ikili program

Adres	Buyruk Kodu
0	0010 0000 0000 0100
1	0001 0000 0000 0101
10	0011 0000 0000 0110
11	0111 0000 0000 0001
100	0000 0000 0101 0011
101	1111 1111 1110 1001
110	0000 0000 0000 0000

Onaltılık Program

Adres	Buyruk Kodu
000	2004
001	1005
002	3006
003	7001
004	0053
005	FFE9
006	0000

Sembolik İşlem Kodlarıyla Program

Adres	Buyruk		Açıklama
000	LDA	004	Birinci veriyi AC al
001	ADD	005	AC ile ikinci veriyi topla
002	STA	006	Toplamı 006 adresine depola
003	HLT		İşlemi durdur
004	0053		Birinci veri
005	FFE9		İkinci veri (negatif)
006	0000		Toplamın saklanacağı yer

Birleştirici Dil (Assembly) Programı

```
      ORG   0     /Origin of program is location 0
      LDA   A     /Load  operand from location A
      ADD   B     /Add  operand from location B
      STA   C     /Store sum in location C
      HLT         /Halt computer
A,    DEC   83    /Decimal operand
B,    DEC   -23   /Decimal operand
C,    DEC   0     /Sum stored in location C
      END         /End of symbolic program
```

Fortran Programı

```
INTEGER  A, B, C
DATA  A,83 / B,-23
C = A + B
END
```

Birleştirici Dil

- Birleştirici dil programlarının her satırı 3 sütundan oluşur:
  1. **Başlık alanı:** Boş olabilir veya sembolik adres belirler.
     - Sembolik adres en fazla 3 karakterden oluşabilir.
     - Sembolik Adres virgül ile sonlanır.
  2. **Buyruk alanı:** Bir makine buyruğu yada açıklama vardır.
     - Bellek adreslemeli buyruk (Memory Reference Instruction) (MRI)
       - Doğrudan yada dolaylı adreslemeli bir buyruk olabilir.

```
ADD   OPR       (direkt adreslemeli MRI)
ADD   PTR   I    (Dolaylı adreslemeli MRI)
```

     - Yazaç adreslemeli buyruk (non-MRI):

```
CLA
```

     - Sözde Buyruk: Sözde buyruk bir makine buyruğu değildir. Daha çok derleyiciye bilgi vermek için kullanılır.
  3. **Açıklama alanı:** Boştur veya açıklama vardır
     - ‘/ ’ ifadesinden sonra açıklama yazılır.

Sözde Buyruk Tanımları

Buyruk	*Açıklama*
ORG N	Buyruk veya Veri Listesinin bellekteki Başlangıç yeri N onaltılık sayıdır
END	Sembolik programın bittiğini belirtir.
DEC N	İkiliye çevrilecek işaretli ondalık N sayısı
HEX N	İkiliye çevrilecek işaretli onaltılık N sayısı

Örnek Program

- İki sayının çıkarılması için yazılan birleştirici dili programı

```
        ORG  100      /Program 100 adresinden başlar
        LDA  SUB      /Çıkarılanı AC ye yükle
        CMA           /AC nin tümleyenini al
        INC           /AC yi 1 arttır
        ADD  MIN      /Çıkanı AC ile topla
        STA  DIF      /Sonucu sakla
        HLT           /Programı durdur
MIN,    DEC  83       /Çıkan
SUB,    DEC  -23      /Çıkarılan
DIF,    HEX  0        /Sonuç
        END           /Program sonu
```

Örnek Makine Programı

- İki sayının çıkarılması için yazılan birleştirici dili programı

Onaltılık Kod		*Birleştirici Program*	
Adres	*Buyruk*		
			ORG 100
100	**2107**		**LDA SUB**
101	**7200**		**CMA**
102	**7020**		**INC**
103	**1106**		**ADD MIN**
104	**3108**		**STA DIF**
105	**7001**		**HLT**
106	**0053**	**MIN,**	**DEC 83**
107	**FFE9**	**SUB,**	**DEC -23**
108	**0000**	**DIF,**	**HEX 0**
			END

Derleyici

- Derleyici sembolik programı alır ve bunu birleştirici dil ikili eşdeğerine çevirir.
  - Girilen sembolik programa kaynak program.
  - Sonuçta elde edilen ikili programa amaç program denir.
- Derleyici için giriş, kullanıcının sembolik programıdır.
  - Programın dönüştürülmesi işlemi derleyicinin programı iki kez taraması ile elde edilir.

Program Döngüleri

- **Döngü:** ardışık olarak birçok kez icra edilen buyruklar sırasıdır. Fortranda 100 sayının toplamı ile ilgili bir örnek:

```
DIMENSION  A(100)
INTEGER  SUM,  A
SUM = 0
DO  3  J = 1,  100
3 SUM = SUM + A(J)
```

```
        ORG  100          /Program bellekte 100 üncü adrestedir
        LDA  ADS          /Verinin ilk adresini AC yükle
        STA  PTR          /Adresi Göstericiye aktar
        LDA  NBR          /Veri sayacını AC yükle
        STA  CTR          /Sayacı kur
        CLA               /AC temizle
LOP,    ADD  PTR  I       /Birinci veriyi AC ile topla
        ISZ  PTR          /Veri göstericiyi 1 arttır
        ISZ  CTR          /Sayacı 1 arttır Sayaç 0 ise sonraki buyruğa geç
        BUN  LOP          /LOP'a şartsız dallan
        STA  SUM          /Toplamı Bellekte SUM olduğu yere kaydet
        HLT               /Programı durdur
ADS,    HEX  150          /Verinin ilk adresi
PTR,    HEX  0            /Veri adres göstericisi
NBR,    DEC  -100         /Sayaç için ilk değer
CTR,    HEX  0            /Sayaç bellek alanı
SUM,    HEX  0            /Toplamın yükleneceği bellek alanı
        ORG  150
        DEC  75           /İlk veri
        .
        .
        DEC  23           /Son veri
        END               /Sembolik programın sonu
```

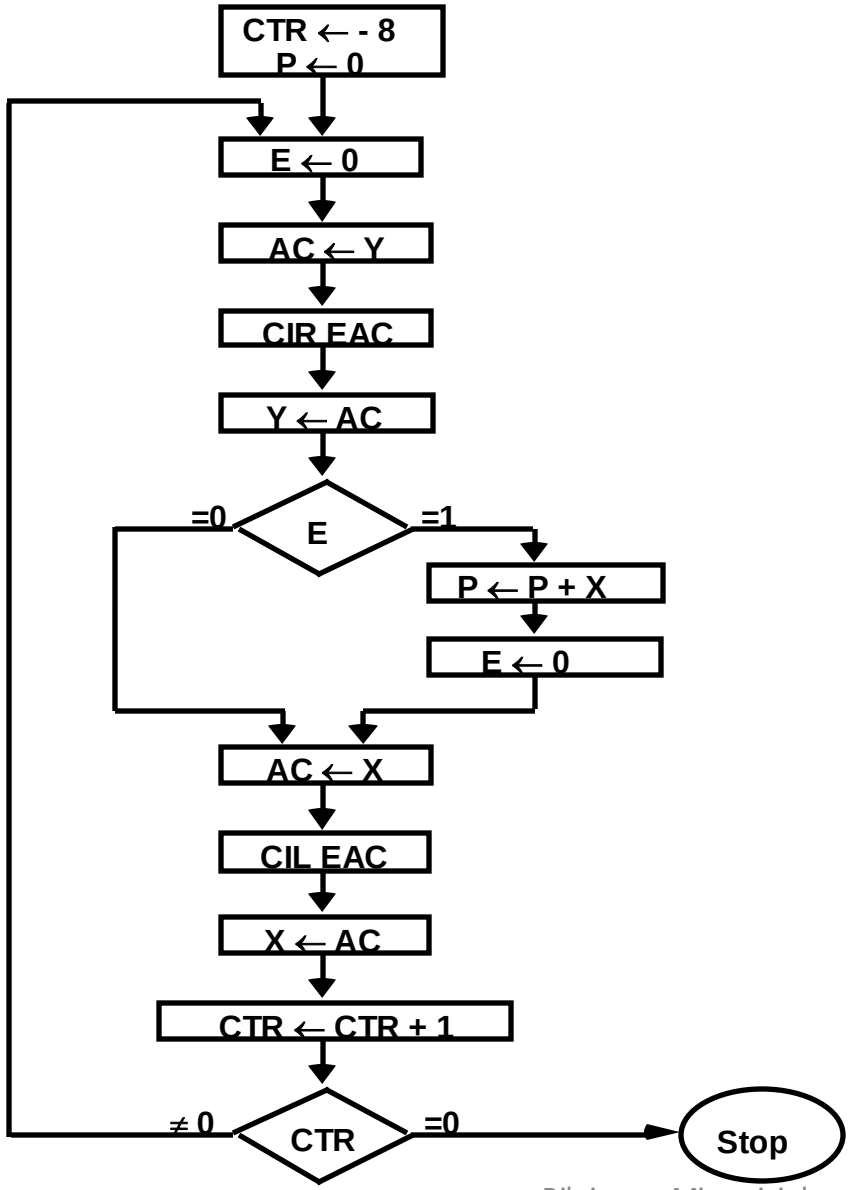
Aritmetik ve Mantıksal İşlemlerin Programlanması

- Bilgisayarlarda Aritmetik ve Mantıksal İşlemlerin Gerçekleştirilmesi iki şekilde gerçekleşebilir.
- **Yazılım Tarafından Gerçekleştirim**
  - Bir buyruk kümesi yardımı ile yapılan işlemlere verilen isimdir.
  - Daha küçük bir buyruk kümesi ile işlemler yazılım yardımıyla gerçekleştirilir.
- **Donanım Tarafından Gerçekleştirim**
  - Bilgisayar içinde tek bir makine buyruğu ile gerçekleştirilen işlemlerdir.
  - Donanım ile işlemlerin yapılması daha maliyetlidir. Çünkü çok sayıda elektronik devreye ihtiyaç vardır.

***Yazılım gerçekleştirim Örneği**

- Çarpma Programı
  - 8 bitlik iki işaretsiz sayının çarpımı -> Sonuç 16 bitlik işaretsiz sayı olacaktır.

Çarpma Programı Akış Şeması



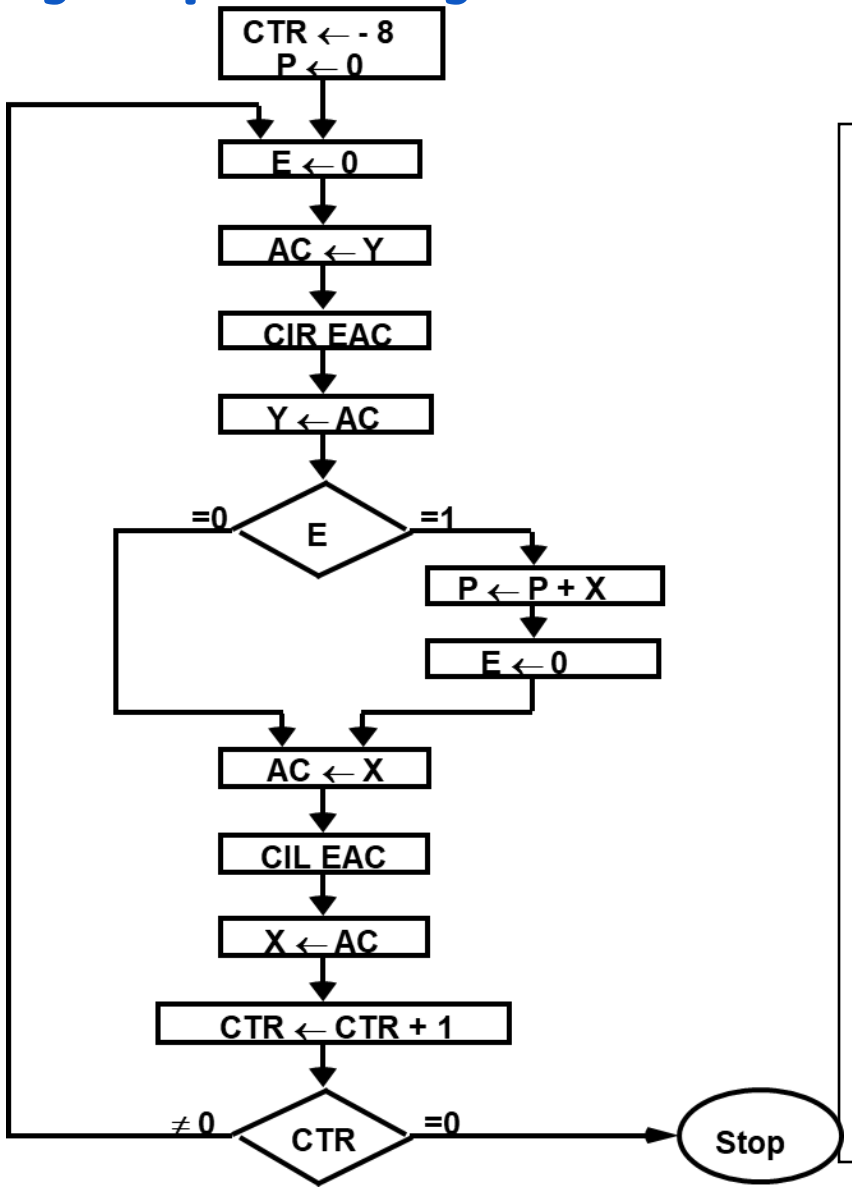
X Çarpılanı tutar
Y Çarpanı tutar
P Sonucu tutar

- ✓ Çarpan (Y) nin 1 olan her biti için toplama yapılır
- ✓ Çarpılan (X) değeri her Y biti için sola kaydırılır.
- ✓ Y nin 1 olan her biti için X in değeri Çarpım (P) ile toplanır.

4 bit için çarpma işlemi örneği

		P
X =	0000 1111	
Y =	0000 1011	0000 0000
	0000 1111	0000 1111
	0001 1110	0010 1101
	0000 0000	0010 1101
	0111 1000	1010 0101
	1010 0101	

Çarpma İşleminin Assembly Programı




```
        ORG  100
LOP,    CLE          / E yi temizle
        LDA  Y       / Çarpanı yükle
        CIR          / Çarpan bitini E ye aktar
        STA  Y       / Kaydırılmış Çarpanı sakla
        SZE          / E=0 mı kontrol et
        BUN  ONE     / E=1 ise ONE etiketine git
        BUN  ZRO     / E=0 ise ZRO etiketine git
ONE,    LDA  X       / Çarpılanı yükle
        ADD  P       / Kısmi çarpıma ekle
        STA  P       / Kısmi çarpımı sakla
        CLE          / E yi temizle
ZRO,    LDA  X       / Çarpılanı yükle
        CIL          / Sola kaydır
        STA  X       / Kaymış Çarpılanı sakla
        ISZ  CTR     / Sayacı 1 arttır
        BUN  LOP     / Sayaç=0 değilse LOOP devam et
        HLT          / Sayaç=0 ise; Programı durdur
CTR,    DEC  -8      / Sayacın ilk değeri 8 bit için
X,      HEX  000F    / Çarpılan değer
Y,      HEX  000B    / Çarpanın değeri
P,      HEX  0       / Çarpımın saklanacağı bellek alanı
        END
```

Çift Duyarlılıklı iki Sayının Toplanması Örneği

- Çift duyarlılıklı sayılardan birisi ardışık iki bellek adresine yerleştirilsin.
- Bunlar AL ve AH olup, AL düşük seviyeli 16 biti içerir.
- Diğer Rakam BL ve BH tadır.
- İki düşük seviyeli kısım toplanır ve elde E ye konur.
- Sonra AC temizlenir ve E nin değeri AC nin en önemsiz bitine aktarılır.
- Sonra sayıların önermli kısımları AC ye eklenir.
- Sonuç CL ve CH ta saklanır.

```
LDA   AL      / A nın düşük seviyeli bit AC ye yükle
ADD   BL      / AC yi B ile topla, eldeyi E ye aktar
STA   CL      / Toplamı CL de sakla
CLA           / AC yi sil
CIL           / E değerini AC(0) a aktar
ADD   AH      / A nın yüksek seviyeli bitlerini Ac ie topla
ADD   BH      / AC ye B yi ekle
STA   CH      / Toplamı CH a kaydet
HLT           / Programı sonlandır
```

Mantıksal İşlemler

- Temel Bilgisayarımızda 3 adet mantıksal işlem tanımlıdır
  - AND, CMA, CLA
  - LDA ile buyruğu AC taşıyan mantıksal buyruk olarak düşünülebilir.
- Örnek Temel Mano Bilgisayarında VEYA işlemi

  x+y=(x'y')' olarak ifade edilebilir.

```
LDA   A      / A nın değerini AC yükle
CMA          / A nın tümleyenini al A'
STA   TMP    / AC yi TMP alanına aktar
LDA   B      / B nin değerini AC yükle
CMA          / B nin tümleyenini al B'
AND   TMP    / A' ile AC yi VE'le (A' AND B')
CMA          / tekrar tümleyen alarak A VEYA B bul
```

Kaydırma İşlemleri

- Temel Bilgisayarımızda sadece dairesel kaydırma işlem tanımlıdır.
- Diğer kaydırma işlemleri az sayıda buyruk ile tanımlanabilir.

- Mantıksal Sağa-kaydırma

```
CLE
CIR
```

- Mantıksal Sola-kaydırma

```
CLE
CIL
```

- Arithmetic Sağa-kaydırma

```
CLE     / Clear E to 0
SPA     / Skip if AC is positive
CME     / AC is negative
CIR     / Circulate E and AC
```

Alt Programlar

- Bazen bir program parçasının bir program içerisinde birçok kez kullanılması gerekir.
- Bunun için program parçası bir defa yazılarak bir fonksiyon şeklinde program içerisinde birçok kez çağrılabilir. Buna alt program denilir.
- Temel bilgisayarda ana program ile alt program arasındaki bağlantı BSA ile sağlanır.

Alt Program Örneği

Adres			
100		ORG 100	/ Ana program
101		LDA X	/ X i AC ye yükle
102		BSA SH4	/ Alt programa git
103		STA X	/ AC yi X e aktar
104		LDA Y	/ Y yi AC ye yükle
105		BSA SH4	/ Alt programa git
106		STA Y	/ AC yi Y ye aktar
107		HLT	/ Programı durdur
108	X,	HEX 1234	
	Y,	HEX 4321	
			/ 4 kez sola kaydıran alt program
109	SH4,	HEX 0	/ Geri dönüş adresi burada saklanır
10A		CIL	/ Bir defa sola kaydır
10B		CIL	
10C		CIL	
10D		CIL	/ 4. kez sola kaydır
10E		AND MSK	/ AC(13-16) bitlerini sıfırla
10F		BUN SH4 I	/ Ana programa dön
110	MSK,	HEX FFF0	/ Maskeleme değeri
		END	

Alt Program Parametreleri ve Veri Bağlantıları

Mantıksal OR işlemi gerçekleştiren alt program

Adres			
200		ORG 200	
200		LDA X	/ İlk veriyi AC ye yükle
201		BSA OR	/ OR alt programına git
202		HEX 3AF6	/ İkinci veri burada saklanmakta
203		STA Y	/ Altprogramın döndüğü yer OR sonucu sakla
204		HLT	/ Programı durdur
205	X,	HEX 7B95	/ İlk Veri burada saklanmakta
206	Y,	HEX 0	/ Sonuç burada saklanacak
207	OR,	HEX 0	/ OR alt programı
208		CMA	/ İlk verinin tümleyenini al
209		STA TMP	/ AC yi geçici TMP ye yaz
20A		LDA OR I	/ AC ye ikinci veriyi yükle
20B		CMA	/ İkinci verinin tümleyenini al
20C		AND TMP	/ İlk veri ile VE le
20D		CMA	/ VEYA işlemini elde etmek için tümleyenini al
20E		ISZ OR	/ Geri dönüş değerini 1 arttır
20F		BUN OR I	/ Ana programa dön
210	TMP,	HEX 0	/ Geçici sonuç değişkeni
		END	

Veri Bloğu Taşınması Alt Program Örneği

```
                    / Ana Program
        BSA  MVE    / MVE bulunan Alt programa dallan
        HEX  100    / Kaynak verinin başlangıç adresi
        HEX  200    / Hedef verinin başlangıç adresi
        DEC  -16    / Toplam aktarılacak veri sayısı
        HLT
MVE,    HEX  0      / MVE adlı alt program
        LDA  MVE I  / Kaynak verinin başlangıç adresini al
        STA  PT1    / PT1 kaynak göstergesini kur
        ISZ  MVE    / Dönüş adresini arttır
        LDA  MVE I  / Hedefin başlangıç adresini getir.
        STA  PT2    / PT2 hedef göstergesini kur
        ISZ  MVE    / Dönüş adresini 1 arttır
        LDA  MVE I  / Aktarılacak olan veri sayısını al
        STA  CTR    / Sayacı kaydet
        ISZ  MVE    / Dönüş adresini 1 arttır (Dönüş değeri)
LOP,    LDA  PT1 I  / PT1 gösterdiği değeri AC ye aktar
        STA  PT2 I  / AC nin değerini PT2 nin gösterdiği adrese kaydet
        ISZ  PT1    / PT1 i 1 arttır
        ISZ  PT2    / PT2 yi 1 arttır
        ISZ  CTR    / Sayacı 1 arttır
        BUN  LOP    / Döngüyü 16 kez tekrarla
        BUN  MVE I  / Ana Programa dön
PT1,    --
PT2,    --
CTR,    --
```

- **Fortran Örneği**

```
      SUBROUTINE  MVE (SOURCE, DEST, N)
      DIMENSION  SOURCE(N), DEST(N)
      DO  20  I = 1, N
20    DEST(I) = SOURCE(I)
      RETURN
      END
```

Giriş / Çıkışın Programlanması

a) Bir karakter alma

```
CIF,    SKI         / Giriş Bayrağını kontrol et Eğer 1 ise sonraki buyruğa atla
        BUN  CIF    / Bayrak=0, CIF e git
        INP         / Bayrak=1 ise giriş karakterini al
        OUT         / Karakteri ekrana bas
        STA  CHR    / AC dekini belleğe aktar
        HLT
CHR,    --          / Karakterin saklandığı bellek alanı
```

b) Bir Karakterin Yazılması

```
        LDA  CHR    / CHR yi AC ye yükle
COF,    SKO         / Çıkış Bayrağını kontrol et Eğer 1 ise sonraki buyruğa atla
        BUN  COF    / Bayrak=0, COF a git
        OUT         / Bayrak=1, karakteri çıkışa gönder
        HLT
CHR,    HEX  0057   / Karakter "W"
```

Program Kesmeleri

- Temel bir bilgisayar aynı anda sadece bir programı çalıştırabilir.
- Kesme G/Ç birimlerinin bayrak kaldırması ile devreye girer.
- Çalışan program içerisinde bir kesme geldiğinde bilgisayar buyruğun çalışmasını tamamlar sonra işlemci yazmaçların içeriğini saklayarak kesme programına dallanır.
- Kesme yordamı içinde bulunması gerekenler:

1. İşlemci yazaçlarının içeriklerinin saklanması.
2. Hangi bayrağın kaldırıldığının bulunması.
3. Bayrağı kaldıran birimin işleminin yapılması.
4. İşlemci yazaç içeriklerinin tekrar yerine yazılması.
5. Kesme olayının tekrar açılması.
6. Çalışan programa geri dönüş.

Program Kesme Örneği

```
Adres
   0        ZRO,    -            / Dönüş adresi burda saklanıyor
   1                BUN  SRV     / Servis yordamına dallanıyor
 100                CLA          / Portion of running program
 101                ION          / Kesme servisi açıldı
 102                LDA  X
 103                ADD  Y       / Kesme burada oluştu
 104                STA  Z       / Kesmeden sonra program buraya döner
   .                  .
   .                  .         / Kesme Servis Yordamı
 200        SRV,    STA  SAC     / AC nin içeriğini sakla
                    CIR          / E yi AC(15) e aktar
                    STA  SE      / E değerini sakla
                    SKI          / Giriş bayrağını kontrol et
                    BUN  NXT     / Giriş Bayrak=0 ise atla
                    INP          / Bayrak=1 ise karakteri oku
                    OUT          / Karakteri ekrana yaz
                    STA  PT1  I  / Girilen karakteri giriş bufferında depola
                    ISZ  PT1     / Giriş Göstergesini 1 arttır
        NXT,        SKO          / Çıkış bayrağını kontrol et
                    BUN  EXT     / Bayrak=0 ise çıkışa dallan
                    LDA  PT2  I  / Çıkış bufferından karakteri AC ye yükle
                    OUT          / Karakteri çıkış birimine gönder
                    ISZ  PT2     / Increment output pointer
        EXT,        LDA  SE      / E nin eski değerini AC ye geri yükle
                    CIL          / Değeri E ye kaydır
                    LDA  SAC     / AC nin değerini geri yükle
                    ION          / Kesmeyi Aç
                    BUN  ZRO  I  / Daha önce Çalıştırılan programa geri dön
        SAC,        -            / AC is stored here
        SE,         -            / E is stored here
        PT1,        -            / Pointer of input buffer
        PT2,        -            / Pointer of output buffer
```

Teşekkürler…

- Sorularınız?