Buyruklar

Prof. Dr. Eşref ADALI

Doç. Dr. Şule Gündüz Öğüdücü

Sürüm-B

Konular

- Buyruklar
- Veri aktarma buyrukları
- Aritmetik işlem buyrukları
- Mantıksal işlem buyrukları
- İşlem buyrukları
- Öteleme buyrukları
- Karar Verme ve Dallanma buyrukları
- Giriş-Çıkış buyrukları

Buyruklar

- Bir bilgisayar gerçeklendiği an sadece, buyruk kümesini bilmektedir. Kullanıcı, bu buyruk kümesini kullanarak, bilgisayara isteklerini anlatabilir; başka bir deyişle, bilgisayarı programlayabilir. Bilgisayarın buyruk kümesini doğrudan kullanarak program yazmaya Makine Dilinde Program Yazma denilmektedir.
- MİB'in yapısı ve buyruk türleri konusunda sürekli araştırmalar yapılmaktadır. Bazı MİB'ler çok çeşitli buyruk kümeleri ile sunulurken, bazıları az, ancak etkin buyruk kümesiyle üretilmektedir. Buyruk çeşidi fazla olan bilgisayarlara CISC, buyruk çeşidi az olanlara da RISC mimarinde bilgisayar denilmektedir.
- Buyruk kümesi MİB'in yapısına bağlı olarak hazırlanmaktadır. Bu nedenle her mikroişlemci ailesinin buyruk kümesi diğer mikroişlemci ailelerininkinden farklıdır. Bu özelik makine dilinde program yazabilmeyi zorlaştırmaktadır.
- Her MİB'in buyruk kümesinin farklı olmasına ek olarak, komutlara karşılık kullanılan kısaltmalar üzerinde de anlaşma sağlanamamıştır. Bu nedenle bu bölümde buyruklar genelleştirilerek ele alınacak ve kısaltmalar Türkçe verilecektir.

Veri Aktarma Buyrukları

- Merkezi İşlem Birimi içindeki bir kütüğün içeriğinin, diğer bir kütüğe aktarılması ve bir bellek gözünün içeriğinin bir kütüğe aktarılması veya bunun tersi işlemler veri aktarma işlemi olarak kabul edilirler. MİB içindeki aktarma işlemleri eşit boydaki kütükler arasında yapılabilir.

  - Aktarma
  - Yükleme
  - Yazma
  - Takas
  - Değiş

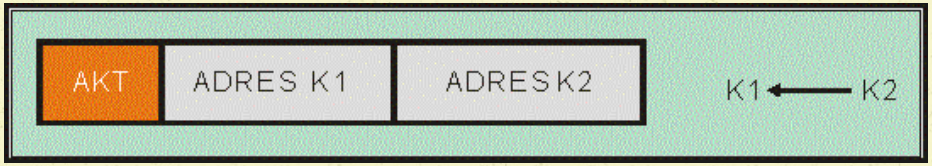
Aktarma

- Aktarma buyruğu MİB içindeki kütüklerin içeriklerinin birbirine aktarılması için kullanılır. MİB içindeki 8 bitlik kütükler Ki ve Kj biçiminde gösterilecektir. Benzer şekilde 16 bitlik kütükler Kii ve Kjj olarak gösterilecektir.

  AKT Ki, Kj Ki ⟵ Kj

- Buyruğun yanında verilen açıklamadan da anlaşılacağı gibi, bu yazımda, Kj' nin içeriği Ki ye aktarılmaktadır. Bu yazım biçiminde kural, sağdaki kütük içeriğinin soldaki kütüğe aktarılmasıdır. Bu nedenle Ki' nin içeriği Kj' ye aktarılmak istenirse, buyruk şöyle yazılmalıdır:

  AKT Kj, Ki Kj ⟵ Ki



  AKT A, B ACC A ⟵ ACC B

  AKT A, C ACC A ⟵ C

  AKT B, DK ACC B ⟵ DK

  AKT SK, YG SK ⟵ YG

  AKT AB, CD AB ⟵ CD

Yükleme

- Yükleme buyruğu, bir bellek gözünde bulunan verinin akümülatörlere veya kütüklere aktarılması işlemidir. Yükleme işleminin yönü, her zaman bellekten Merkezi İşlem Birimine doğru olacaktır.

YÜK Ki,<BELLEK> Ki ⟵ <BELLEK>

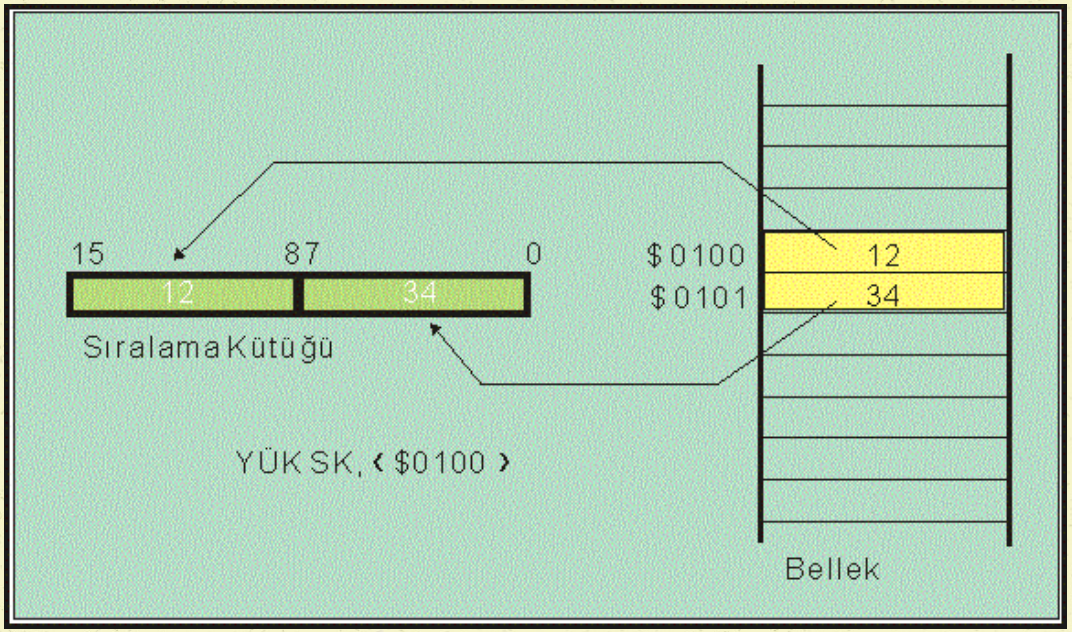
- Merkezi İşlem Birimi içindeki, SK ve DK kütüklerine, yükleme işlemi ivedi adresleme biçiminde de yapılır. Yani bir veri kütüğe doğrudan doğruya yüklenebilir. İvedi yüklemede, yüklenen verinin boyu, yüklendiği kütüğün boyuna denk olmalıdır. Yani 8 bitlik kütükler 8 bitlik veri ile ve 16 bitlik kütükler 16 bitlik veri ile yüklenmelidir.

YÜK A,<$1000>	ACC A ⟵ <$1000>
YÜK C,<$2000>	C ⟵ <$2000>
YÜK B, $25	ACC B ⟵ $25
YÜK SK,$1234	SK ⟵ $1234

Yükleme

- 8bitlik mikroişlemcilerde akümülatör ve yardımcı kütükler 8 bit uzunluğundadır. Aynı şekilde bir bellek gözü de 8 bit uzunluktadır. Yükleme sırasında, bir kütüğe boyu kadar veri sığdırılabileceği için 8 bitlik bir kütüğe en çok 8 bit uzunluğundaki bir bellek gözünün içeriği yüklenebilir. 16 bitlik kütüklerin yüklenmesinde ise, peşpeşe gelen iki bellek gözünün içeriği işleme katılır. Sözgelimi;

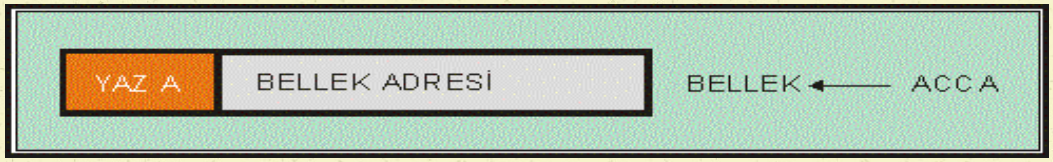
YÜK SK, <$0100> SK ⟵ <$0100> + <$0101>



Yazma

- Yazma işlemi yükleme işlemiyle aynı biçimde. Ancak verinin aktarılma yönü terstir. Yani, MİB içindeki kütüklerin içerikleri bellek gözüne aktarılır.

YAZ Ki,<BELLEK> BELLEK ⟵ Ki



Kütüğün boyu ile yazılacağı bellek boyunun denk olması yine gerek koşuldur. Bu nedenle 8 bitlik bir kütüğün içeriği 8 bitlik bir bellek gözüne yazılabilir.

YAZ A, $1000 $1000 ⟵ ACC A
YAZ C, $E000 $E000 ⟵ C

Belleğe ivedi yazma işleminde, 8 bitlik veri, 8 bitlik bir bellek gözüne yazılır.

YAZ $25,$1000 $1000 ⟵ $25

Yazma

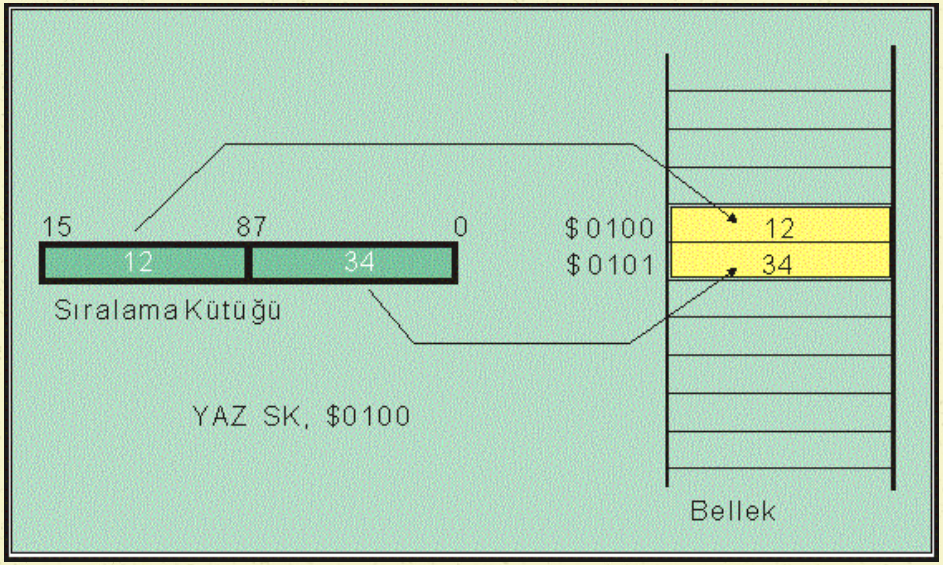
16 bitlik bir kütüğün içeriği ise, peşpeşe gelen iki bellek gözüne yazılabilir.

YAZ SK,$0100 $0100+ $0101 ⟵ SK
YAZ AB,$C000 $C000+ $C001 ⟵ AB

Belleğe 16 bitlik veri ivedi olarak Yazılabilir.

YAZ $1234,$1000

$1000+ $1001 ⟵ $1234



Takas

Takas buyruğu, MİB içindeki bazı kütüklerin, içeriklerini birbiri ile takas etmelerini sağlar. Takas buyruğunun genel yazım biçimi şöyledir:

TKS Ki, Kj	Ki $\longleftrightarrow$ Kj
TKS Kii, Kjj	Kii $\longleftrightarrow$ Kjj

Takas işleminde Ki kütüğünün içeriği Kj kütüğüne ve Kj kütüğünün içeriği Ki kütüğüne aktarılır. Takas işlemine örnek olarak aşağıdaki buyruklar verilebilir:

TKS A, B	A $\longleftrightarrow$ B
TKS C, D	C $\longleftrightarrow$ D
TKS SK, YG	SK $\longleftrightarrow$ YG

Değiştirme

- Değiş buyruğu, MİB içindeki 8 bitlik kütüklerin, ilk dört biti ile ikinci dört bitinin yer değiştirmesini sağlar. Değiş buyruğunun genel yazım biçimi şöyledir:

DGS Ki	Buyruk işletilmeden önce	Ki [D7,D6,D5,D4,D3,D2,D1,D0]
	İşletildikten sonra	Ki [D3,D2,D1,D0 ,D7,D6,D5,D4]

- Değiş işlemine örnek olarak aşağıdaki buyruklar verilebilir:

DGS A

DGS C

Aritmetik Buyruklar

Aritmetik işlemlerin yerine getirilmesi için kullanılan buyruklara aritmetik işlem buyrukları denilmektedir. Aritmetik işlem buyruklarında, birinci işlenen kesinlikle akümülatör olmak zorundadır. Aritmetik işlemin sonunda elde edilen sonuç da yine akümülatörde yer alır.

- Toplama
- Eldeli toplama
- Çıkarma
- Borçlu çıkarma veya eldeli çıkarma
- Çarpma
- Bölme

Toplama

Toplama buyruğu ile, iki akümülatörün içeriği, bir akümülatör ile bir yardımcı kütüğün içeriği, bir akümülatör ile bir bellek gözünün içeriği veya bir akümülatör ile bir veri toplanabilir. Sonuç buyrukta birinci işlenen durumunda bulunan akümülatöre yazılır. Bu işlemlere ilişkin genel yazım kalıpları şöyledir: (V, 8 bitlik bir veriyi temsil etmektedir.)

TOP A, B	ACC A ⟵ ACC A + ACC B
TOP A, Ki	ACC A ⟵ ACC A + Ki
TOP A, V	ACC A ⟵ ACC A + V
TOP A, <BELLEK>	ACC A ⟵ ACC A + <BELLEK>

Eldeli Toplama

Bir toplama işlemi sonunda, sonuç akümülatörün boyunu aşabilir. Bu durum elde bayrağı ile gösterilir. Bir sonraki toplama işleminde bu elde biti toplama katılabilir. Elde bitinin toplama katılması istenirse eldeli toplama komutu kullanılır. Eldeli toplama örneği aşağıdadır:

TOPE A, B	ACC A ⟵ ACC A + ACC B + E
TOPE A, C	ACC A ⟵ ACC A + C + E
TOPE A, V	ACC A ⟵ ACC A + V + E
TOPE A, <BELLEK>	ACC A ⟵ ACC A + <BELLEK> + E

Toplama Örneği

X sayısı 10 ve 11 sayılı bellek gözlerine yazılmıştır. 10 sayılı bellek gözünde sayının yüksek anlamlı ve 11 sayılı bellek gözünde düşük anlamlı kısmı yazılıdır. Benzer şekilde Y sayısı da 12 ve 13 sayılı bellek gözlerine yazılmıştır. İki sayının toplanmasından sonra ortaya çıkacak sayı benzer şekilde 14 ve 15 sayılı bellek gözlerine yazılacaktır. Bu işlem sonunda üçüncü bir basamağın oluşmayacağı varsayılmıştır.

```
    < 10 > < 11 >    X
    < 12 > < 13 >    Y
+ ------------------------
    < 14 > < 15 >    sonuç
```

```
YÜK A, <10>
YÜK B, <11>
TOP B, <13>
TOPE A, <12>
YAZ A, 14
YAZ  B, 15
```

Örnek sayılar

X = 0100 0000 1010 0000
Y = 0001 1011 1100 0001

<10> = 0100 0000 <11> = 1010 0000 <12> = 0001 1011 <13> = 1100 0001

```
    <11> =      1010 0000
    <13> =      1100 0001
+ ------------------------
    <15> =  1   0110 0001
```

sonucu elde edilir. Dikkat edilirse elde biti 1 dir. Yüksek anlamlı kısımların toplanmasında bu elde toplama katılır.

```
    elde   =          1
    <10>   = 0100 0000
    <12>   = 0001 1011
+ ------------------------
    <14>   = 0101 1100
```

Toplama sonunda taşma oluşur ise bu durum taşma bayrağı ile belirtilir.

Çıkarma

Çıkarma buyruğu ile, toplama buyruğuna benzer şekilde, bir akümülatörden diğer akümülatörün içeriği, bir akümülatörden bir yardımcı kütüğün içeriği, bir akümülatörden bir bellek gözünün içeriği veya bir akümülatörden bir veri çıkartılabilir. Sonuç, buyrukta birinci işlenen durumunda bulunan akümülatöre yazılır. Bu işlemlere ilişkin genel yazım kalıpları şöyledir:

ÇIK A, B	ACC A ⟵ ACC A - ACC B
ÇIK A, Ki	ACC A ⟵ ACC A - Ki
ÇIK A, V	ACC A ⟵ ACC A - V
ÇIK A, <BELLEK>	ACC A ⟵ ACC A - <BELLEK>

Borçlu veya Eldeli Çıkarma

Bir çıkarma işlemi sonunda, borçlu kalınabilir. Bu durum borç bayrağı ile gösterilir. Bir sonraki çıkarma işleminde bu borç biti çıkarma işlemine katılabilir. Borç bitinin çıkarma işlemine katılması istenirse borçlu çıkarma komutu kullanılır. Borçlu çıkarmalara ilişkin genel yazım kalıpları aşağıda verilmiştir: (Durum kütüğü içinde elde ve borç durumları aynı bayrakla belirtilir.) Borçlu çıkarma işleminin kısaltılması ÇIKE olarak kullanılmıştır.

ÇIKE A, B	ACC A ⟵ ACC A - ACC B - E
ÇIKE A, C	ACC A ⟵ ACC A - C - E
ÇIKE A, V	ACC A ⟵ ACC A - V - E
ÇIKE A, <BELLEK>	ACC A ⟵ ACC A - <BELLEK> - E

Çıkarma Örneği

X sayısı 10 ve 11 sayılı bellek gözlerine yazılmıştır. 10 sayılı bellek gözünde sayının yüksek anlamlı ve 11 sayılı bellek gözünde düşük anlamlı kısmı yazılıdır. Benzer şekilde Y sayısı da 12 ve 13 sayılı bellek gözlerine yazılmıştır. İX sayısından Y sayısının çıkarılmasından sonra ortaya çıkacak sayı benzer şekilde 14 ve 15 sayılı bellek gözlerine yazılacaktır. Bu işlem sonunda üçüncü bir basamağın oluşmayacağı varsayılmıştır.

```
        < 10 >  < 11 >    X
        < 12 >  < 13 >    Y
+ ------------------------------
        < 14 >  < 15 >    sonuç
```

```
YÜK A, <10>
YÜK B, <11>
ÇIK  B, <13>
ÇIKE A, <12>
YAZ  A,  $14
YAZ  B,  $15
```

Çarpma

Çarpma işleminde, bilindiği gibi iki işlenen ve bir sonuç olacaktır. Sonucun boyu, iki işlenenin boylarının toplamı kadar olabilir. Örneğin, 8 bitlik bir işlemcide çarpan ve çarpılan sekizer bitlik ise, sonuç 16 bit uzunlukta olabilir. Bu nedenlerle çarpma işlemi kalıbı şöyle yazılabilir:

ÇAR A,B	ACC A + ACC B ⟵ ACC A * ACC B
ÇAR A,Ki	ACC A + ACC B ⟵ ACC A * Ki
ÇAR A,V	ACC A + ACC B ⟵ ACC A * V
ÇAR A, <BELLEK>	ACC A + ACC B ⟵ ACC A * <BELLEK>

Çarpma işleminde, birinci sayı sadece A akümülatörü içinde olmak zorundadır. İkinci sayı, B akümülatöründe, bir yardımcı kütükte veya bir bellek gözünde olabilir. Bu arada ikinci sayı ivedi bir veri de olabilir. Sonuç A ve B akümülatör çiftinde yer alacağıdır.

Çarpma işleminde, çarpan ve çarpılan sayıların işaretsiz olması gerekir.

Örnek

YÜK A, 50
YÜK B, 10
ÇAR A,B

Çarpma işleminin sonunda A ve B akümülatörlerinin içerikleri 0000 0001 1111 0100
A akümülatörü, sonucun yüksek anlamlı kısmını,
B akümülatörü düşük anlamlı kısmını içerir.

Bölme

Bölme sonucunda bir sonuç ve bir de kalan oluşur. Sonuç ve kalan sayıların boyları, bölünen ve bölen sayının boyları ile ilişkilidir. 8 bitlik işlemciler için örnek verirsek, bölünen sayı 16 ve bölen sayı 8 bit ise, sonuç 15 bit , kalan 7 bit uzunlukta olabilir.
Bölünen sayı A ve B akümülatörlerine yerleştirilir. 8 bitlik bölen sayı, bir yardımcı kütük veya bir bellek gözü içeriği ya da salt bir veri olabilir. Sonuç A ve B akümülatörlerinde oluşur. Artan kısım C yardımcı kütükte yer alır.

BÖL AB,V	ACC A + ACC B ⟵ < ACC A + ACC B > / V
BÖL AB,Ki	ACC A + ACC B ⟵ < ACC A + ACC B > / Ki
BÖL AB,<BELLEK>	ACC A + ACC B ⟵ < ACC A + ACCB > / <BELLEK>

Her durumda, kalan C kütüğüne yazılacaktır.
Bölme işleminde, ana sayı ve bölen sayı işaretsiz sayı olarak değerlendirilir. Dolayısıyla, sonuç da işaretsiz sayı biçiminde oluşur.

Örnek

```
YÜK AB, $01ED   ; onluk 1005
BÖL AB,$0A      ; onluk 10
```

AB akümülatör çiftinde oluşan sayı $00C4 (onluk 100)
C yardımcı kütükte oluşan sayı $05 olacaktır.

Mantıksal İşlem Buyrukları

Bilgisayarda VE, VEYA ve YADA gibi mantıksal işlemler gerçekleştirilebilir. Buyruklarda birinci işlenen akümülatör olmak zorundadır. Bu nedenle, bir akümülatörün içeriği, bir veri, diğer bir akümülatör, yardımcı kütük veya bir bellek gözünün içeriği ile işleme sokulabilir.

- VE
- VEYA
- YADA

VE İşlemi

VE buyruklarının genel yazım kuralı şöyledir:

VE A, V	ACC A ⟵ ACC A .VE. V
VE A, B	ACC A ⟵ ACC A .VE. ACCB
VE A, Ki	ACC A ⟵ ACC A .VE. Ki
VE A, <BELLEK>	ACC A ⟵ ACC A .VE. <BELLEK>

Örnek :

A akümülatörünün içeriği % 1011 0011 ve
C yardımcı kütüğünün içeriği % 1010 0101 ise,
VE A, C buyruğunun işlenmesi sonunda,

YÜK A, % 1011 0011
YÜK C, % 1010 0101
VE A, C

ACC A nın içeriği = % 1010 0001 olur

VEYA İşlemi

VEYA buyruklarının genel yazım kuralı şöyledir:

VEYA A, V	ACC A ⟵ ACCA .VEYA. V
VEYA A, B	ACC A ⟵ ACC A .VEYA. ACCB
VEYA A, Ki	ACC A ⟵ ACC A .VEYA. Ki
VEYA A,<BELLEK>	ACC A ⟵ ACC A .VEYA. <BELLEK>

Örnek :

A akümülatörünün içeriği % 1011 0011 ve
C yardımcı kütüğünün içeriği % 1010 0101 ise,
VEYA A, C buyruğunun işlenmesi sonunda,

YÜK A, % 1011 0011
YÜK C, % 1010 0101
VEYA A, C

ACC A nın içeriği = % 1011 0111 olur

YADA İşlemi

YADA buyruklarının genel yazım kuralı şöyledir:

YADA A,V	ACC A ⟵ ACC A .YADA. V
YADA A, B	ACC A ⟵ ACC A .YADA. ACCB
YADA A, Ki	ACC A ⟵ ACC A .YADA. Ki
YADA A,<BELLEK>	ACC A ⟵ ACC A .YADA. <BELLEK>

Örnek :

A akümülatörünün içeriği % 1011 0011 ve
C yardımcı kütüğünün içeriği % 1010 0101 ise,
YADA A, C buyruğunun işlenmesi sonunda,

YÜK A, % 1011 0011
YÜK C, % 1010 0101
YADA A,C

ACC A nın içeriği = % 0001 0110 olur

İşlem Buyrukları

Buyruk kümesi içindeki bazı buyruklar, aritmetik veya buyruk kümesine katılmak yerine özelikleri gereği işlem buyrukları içinde sayılırlar.

- Silme
- Kurma
- Artırma
- Azaltma
- Tümleme
- Eksileme
- Onluk ayarı
- Yığma
- Çekme
- Kesme izni
- Boş geç

Silme

Silme buyruğunun çeşitli kullanım biçimi vardır. Silme buyruğu ile, durum kütüğü içindeki bayraklar silinebilir, bellek gözü içinde bir göze silinebilir. Bu yeteneğine ek olarak, bir akümülatörün, bir yardımcı kütüğün veya bir belleğin içeriğini siler. Bir başka deyişle sıfırlar.

Bayrakları silme

SİL E	E ⟵ 0 (E, elde bayrağı)
SİL N	N ⟵ 0 (N, eksi bayrağı)
SİL T	T ⟵ 0 (T, taşma bayrağı)
SİL S	S ⟵ 0 (S, sıfır bayrağı)

Kütük veya bellek gözü içinde bir gözeyi silme

SİL 3, Ki	Belirtilen kütük içinde 3. gözeyi sıfırlar
SİL 5, <BELLEK>	Adresi verilen belleğin 5. gözesini sıfırlar

Kütük ya da bellek gözünü silme

SİL A	ACC A ⟵ % 0000 0000
SİL Ki	Ki ⟵ % 0000 0000
SİL <BELLEK>	BELLEK ⟵ % 0000 0000

Kurma

Kurma buyruğu, Silme buyruğuna benzer şekilde, Durum Kütüğü içindeki bayrakları kurar veya bellek gözü içindeki bir gözeyi 1 yapar.

Bayrakları kurma

KUR E	E ⟵ 1 (E, elde bayrağı)
KUR N	N ⟵ 1 (N, eksi bayrağı)
KUR T	T ⟵ 1 (T, taşma bayrağı)
KUR S	S ⟵ 1 (S, sıfır bayrağı)

Kütük veya bellek gözü içinde bir gözeyi kurma

KUR 3, Ki	Belirtilen kütük içinde üçüncü gözeyi 1 yapar
KUR 5, <BELLEK>	Adresi verilen belleğin beşinci gözesini 1 yapar

Artırma

Artırma buyruğu ile, akümülatör, yardımcı kütük, sıralama kütüğü, yığın göstergesi veya bir bellek gözünün içeriğine bir eklenir.

ART A	ACC A ← ACC A + 1
ART Ki	Ki ← Ki + 1
ART Kii	Kii ← Kii + 1
ART <BELLEK>	BELLEK ← <BELLEK> + 1

Azaltma

Azaltma buyruğu ile, akümülatör, yardımcı kütük, sıralama kütüğü, yığın göstergesi veya bir bellek gözünün içeriği bir azaltılır.

AZT A	ACC A ⟵ ACC A - 1
AZT Ki	Ki ⟵ Ki - 1
AZT Kii	Kii ⟵ Kii - 1
AZT <BELLEK>	BELLEK ⟵ <BELLEK> - 1

Tümleme

Tümleme buyruğu, bir akümülatör, bir yardımcı kütük veya bir bellek gözünün içeriğini 1'e tümler, yani sıfırları bir ve birleri de sıfır yapar.

TÜM A	ACC A ⟵ ACCA' nın 1'e tümlenmişi
TÜM Ki	Ki ⟵ Ki' nin 1'e tümlenmişi
TÜM <BELLEK>	BELLEK ⟵ <BELLEK> ‘ ğin 1'e tümlenmişi

A akümülatörünün içeriği % 1011 0011 iken TÜM A buyruğu kullanılırsa
A akümülatörünün içeriği % 0100 1100 olur.

Eksileme

Eksileme buyruğu ile bir akümülatör, bir yardımcı kütük veya bir bellek gözünün içeriği 2'ye tümlenebilir, yani sayı eksilenebilir.

EKS A	ACC A ⟵ ACC A'nın 2'ye tümlenmişi
EKS Ki	Ki ⟵ Ki' nin 2'ye tümlenmişi
EKS BELLEK	BELLEK ⟵ <BELLEK> in 2'ye tümlenmişi

Bir bellek gözünün içeriği % 0011 1100 olduğu varsayılır ve EKS komutu işlenir ise bu bellek gözünün içeriği % 1100 0100 olur.

Onluk Ayarı

Bu buyruk akümülatörde bulunan, ikilik düzendeki veriyi ikilionluk sayı biçimine dönüştürür. 8bitlik akümülatör içinde yazılabilecek en büyük ikilionluk sayının 99 olduğu bilindiğine göre, onluk ayarı yapılacak ikilik sayının 99'u aşmaması gerekir. Onluk ayarı buyruğu A ve B akümülatörü için şöyle yazılır:

ONA A
ONA B

Sözgelimi A akümülatöründe bulunan sayı $ 0001 1001 ise ONA A buyruğunun işlenmesi sonunda akümülatör A nın içeriği $ 0010 0101 yani !25 haline dönüşür.

Yığma ve Çekme

Bir akümülatörün içeriğini, yığına atmaya veya bir başka deyişle yığmaya yarayan buyruğa Yığma Buyruğu diyoruz. Yığma buyruğu örneği şöyledir:

YIĞ A	A akümülatörünün içeriğini yığına at
YIĞ B	B akümülatörünün içeriğini yığına at

Yığın göstergesinin gösterdiği verinin akümülatöre alınmasına yarayan buyruğa Çekme Buyruğu diyoruz. Yığın Göstergesi, yığının en üst adresini gösterir. Adresi azalan yönde oluşturulan yığında, yığının üst noktasının en altta olduğu unutulmamalıdır. Çekme buyruğu örneği aşağıda verilmiştir:

ÇEK A	Yığının tepesindeki veriyi akümülatör A ya al
ÇEK B	Yığının tepesindeki veriyi akümülatör B ye al

Kesme İzni

Kesme izni girişini denetlemede kullanılan bayrağı 1 ya da 0 yapmak için kullanılan özel bir buyruktur. Kesme bayrağı olarak da anılan bu bayrak, genelde, Durum Kütüğü içinde yer alır ve K bayrağı olarak anılır. Kesmeye izin veren ve kesmeyi engelleyen buyruklar şunlardır:

KIZ	Kesmeye izin ver
KEN	Kesmeyi engelle

Boş Geç

Boş geç buyruğu, hiçbir işlem yapmadan, bir adım artıran buyruktur. Bu buyruk iki durumda kullanılır: Yazılmış bir programda, silinmesi gereken bazı adımlar yerine Boş Geç buyruğu yazılabilir. Asıl yararlı kullanımı ise, işlemlerin zaman ayarlarının yapılması sırasında kullanılışıdır. Boş Geç buyruğu;

```
GEÇ
```

biçiminde yazılır. Boş Geç buyruğunun zamanlama işlerinde kullanımı için bir örnek şöyle verilebilir:

```
YAZ A, <BELLEK>
GEÇ
GEÇ
GEÇ
YAZ A, <BELLEK>
```

İşlemcinin saat hızına bağlı olarak her buyruk belli bir sürede işlenir. En kısa sürede işlenen buyruk GEÇ buyruğudur. Bu nedenle, yukarıda verilen basit örnekte olduğu gibi, zaman geçirme işlerinde en hassas ayarlama GEÇ buyruğu ile yapılabilir.

Öteleme ve Döndürme

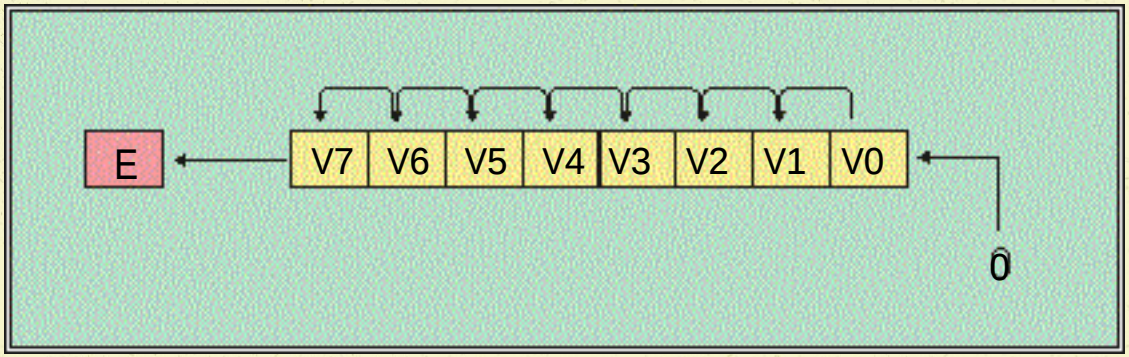
Öteleme işlemi ile akümülatör, yardımcı kütük veya bir bellek gözünün içeriği bir bit sağa veya sola ötelenir. Bazı işlemcilerde, birden fazla adım öteleme yapılabilmektedir. Bu tür buyruklarda adım sayısının belirtilmesi gerekir.

- Öteleme Buyrukları
  - Sola öteleme
  - Sağa öteleme
  - Sağa işaretli öteleme
- Döndürme Buyrukları
  - Sola döndürme
  - Sağa döndürme

Sola Öteleme

Bu buyruk ile ötelenecek yerdeki veri içindeki her bir bit bir adım sola ötelenir. Sola doğru ötelemede YAB'ın içeriği elde bitine yerleşir ve DAB'a ise sıfır ile yüklenir.

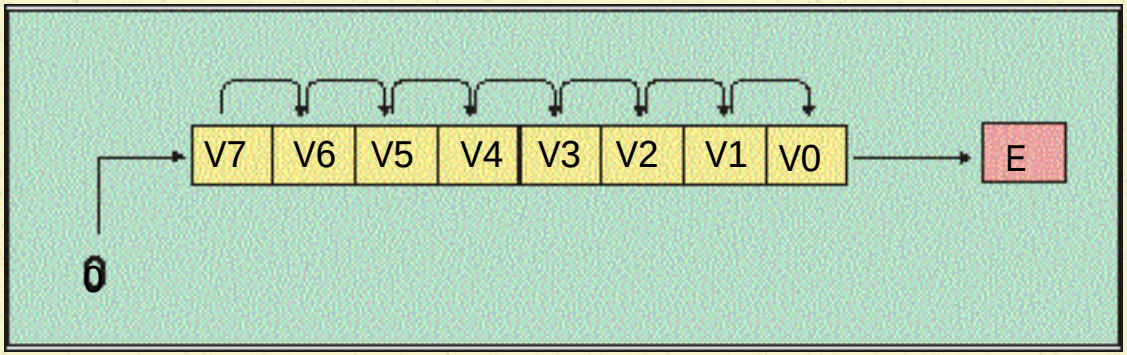
SOL A
SOL Ki
SOL <BELLEK>



Sağa Öteleme

Ötelenecek yerdeki veri içindeki her bir bit bir adım sağa ötelenir. Sağa doğru ötelemede DAB elde bitine yerleşir. YAB sıfır ile yüklenir

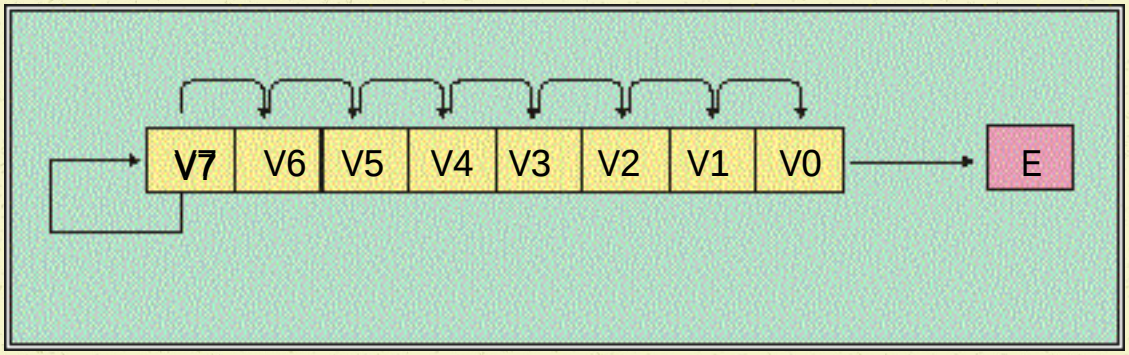
SAĞ A
SAĞ Ki
SAĞ <BELLEK>



Sağa İşaretli Öteleme

Bu buyruk ile, ötelenecek yerdeki veri içindeki her bir bit bir adım sağa ötelenir. Sağa doğru ötelemede DAB elde bitine yerleşir. YAB eski değerini korur. YAB işaretli sayılarda, işaret biti olarak kabul edildiğinden bu öteleme türüne İşaretli veya Aritmetik Sağa Öteleme denir.

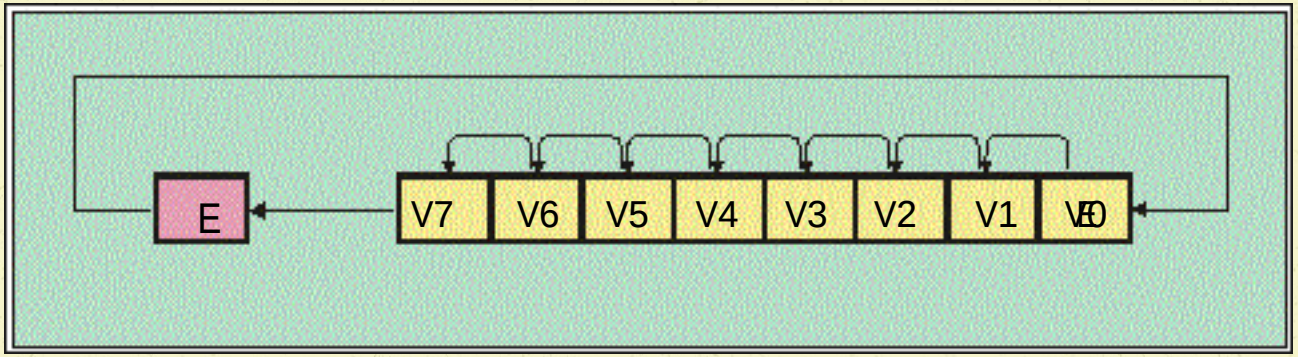
SAĞİ A
SAĞİ Ki
SAĞİ <BELLEK>



Sola Döndürme

Bu buyruk ile döndürülecek yerdeki veri içindeki her bir bit bir adım sola ötelenir. Sola doğru döndürmede YAB elde bitine yerleşir ve elde bitinin içeriği DAB'a kaydırılır.

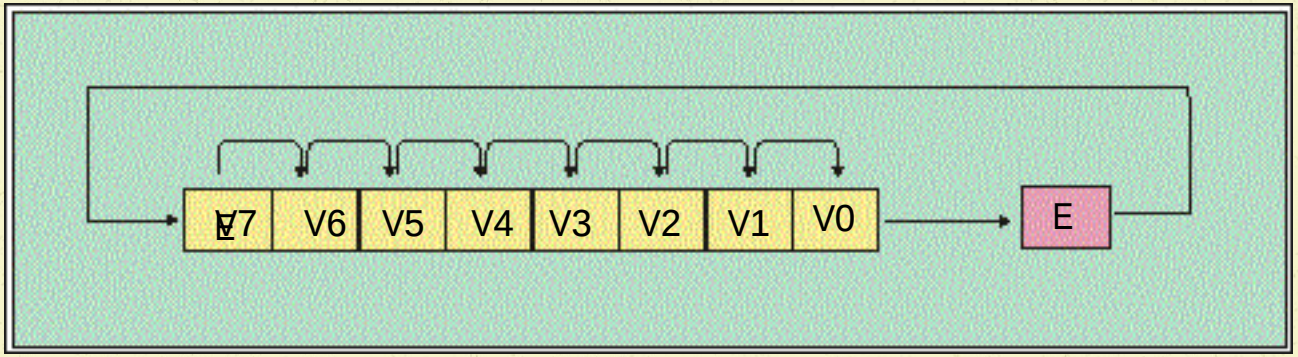
SOLD A
SOLD Ki
SOLD <BELLEK>



Sağa Döndürme

Bu buyruk ile, döndürülecek yerdeki veri içindeki her bir bit bir adım sağa ötelenir. Sağa doğru döndürmede DAB elde bitine yerleşir. YAB'a ise elde bitinin içeriği yerleşir.

SAĞD A
SAĞD Ki
SAĞD <BELLEK>



Karşılaştırma, Karar Verme, Dallanma

Bilgisayarın temel özeliklerinden biri, iki değeri ya da durumu karşılaştırma ve ortaya çıkan sonuca göre karar vermektir. Verilen karara uygun olarak belli bir iş yapılır; yani bu işi yapacak program parçasına dallanılır. Program akışı içinde, karara bağlı olarak bir başka program parçasına dallanma, program akışını değiştireceği için, bu başlık altında toplanan buyruklara Program Akışını Değiştiren Buyruklar adı da verilmektedir.

- Karşılaştırma
- Sınama
- Karar Verme
- Dallanma
- Bağlanma

Karşılaştırma

Karşılaştırma işlemi aslında bir çıkarma işlemidir. Tek farkla, çıkarma işlemi sonunda bulunan sonuç hiçbir yere yazılmaz. Karşılaştırma işlemi sonunda, doğal olarak durum kütüğünün bayrakları etkilenir ve bayrakların durumuna göre karar verilir. Sıralama kütüğü ve yığın göstergesi üzerinde yapılan karşılaştırmalarda, durum kütüğünün tüm bayrakları anlamlı biçimde etkilenmeyebilir. Bu nedenle, karar verme sırasında dikkat edilmesi gerekir. Bu tür geniş kütüklerin karşılaştırılmasında, genellikle eşit olup olmadıkları karşılaştırılabilir, yani, sadece sıfır bayrağına bakılarak karar verilir.

KAR A,V	ACC A - V
KAR Ki,V	Ki - V
KAR A, B	ACC A - ACC B
KAR A, Ki	ACCA - Ki
KAR Ki, Kj	Ki - Kj
KAR A, <BELLEK>	ACC A - <BELLEK>
KAR Kii, Kjj	Kii - Kjj
KAR Kii, VV	Kii - VV
KAR Kii,<BELLEK>	Kii - [<BELLEK>+<BELLEK+1>]

Buyrukların içinde görülen VV, 16 bitlik veriyi temsil etmektedir.

Sınama

Sınama işlemi, akümülatör veya yardımcı kütük içeriği ile bir veri ya da bellek gözü içeriğinin VE'lenmesi işlemidir. Bilinen VE'leme işleminden tek farkı, VE'leme işleminin sonucunda ortaya çıkan sonuç hiçbir yere yazılmaz; sadece Durum Kütüğünün bayrakları etkilenir. Sınama işlemi, genellikle veri içindeki bir bit ya da birden çok bitin durumunu anlamak için kullanılır.

SIN A,V	ACC A · V
SIN Ki,V	Ki · V
SIN A, B	ACC A · ACC B
SIN A, Ki	A · Ki
SIN Ki, Kj	Ki · Kj
SIN A, <BELLEK>	ACC A · <BELLEK>

Dallanma ve Bağlanma

- Dallanma ve bağlanma buyrukları, program akışını koşullu ya da koşulsuz olarak değiştirmeye neden olur. Bağlanma işleminde, gidilecek programın başlangıç adresi belirtilir. Buna karşın dallanma işleminde, gidilecek programın, halen bulunulan yerden ne kadar uzakta olduğu belirtilir. Bir başka deyişle, bağlanmada doğrudan adresleme, dallanmada bağıl adresleme yöntemi kullanılır.
- Dallanma ve bağlanma işleminde, Program Sayacının değeri değiştirilir. PS'nin değeri, gidilecek program parçasının başlangıç adresidir. PS'nin yeni değeri, bağlanma işleminde açıkça belirtilir; buna karşın dallanma işleminde, halen bulunulan yerin adresine uzaklık değeri eklenerek hesaplanır.
- Doğrudan adresleme yöntemi kullanılıyor ise bu buyruklara bağlanma ve bağıl adresleme yöntemi kullanılıyor ise bu buyruklara dallanma buyruğu denecektir. Dallanma ve bağlanma buyruklarının, bir önemli kullanış alanı da altprogramlara dallanmaktır. Altprogramlara dallanmalar da koşullu ve koşulsuz yapılabilir.

  - Koşulsuz dallanma ve bağlanma
  - Koşullu dallanma ve bağlanma
  - Altprograma dallanma ve bağlanma
    - Altprograma koşulsuz dallanma
    - Altprograma koşullu dallanma

Koşulsuz Bağlanma

Bu buyruklar Program Sayacının içeriğini, bir koşula bakmaksızın değiştirirler ve PS'ye programın bundan sonraki kısmının başlangıç adresini yüklerler.

BAĞ (Koşulsuz bağlan)

BAĞ $1000

Bağlanma buyruğunun işlenen kısmında görülen değer, gidilecek programın başlangıç adresidir. Bu değer PS'ye yüklenir ve program bundan sonra bu adresten başlayarak çalışmaya devam eder. Verilen örnekte, programın bundan sonra $1000 adresinden çalışmaya devam edeceği görülmektedir.

Koşulsuz Dallanma I

Koşulsuz dallanma anlamı taşıyan bu buyrukta, işlenen yerinde görülen adım değeri, DAL komutunun bulunduğu adrese eklenir ve böylece yeni bir adres elde edilir. Program DAL buyruğundan sonra bu yeni adresten başlayarak işlemeye devam eder.

DAL (Koşulsuz dallan)
DAL ADIM

PS	etiket	buyruk
$1050		YÜK A, <$B000>
$1053	GERİ	YAZ A, <$C0C0>
$1056		DAL GERİ
$1058		ART A
$1059		DAL İLERİ
$105B		ART A
$105C		YAZ A, <$1000>
$105F	İLERİ	YÜK B, <$D000>
$1062		YAZ B, <$C0C1>

Bu örnekte iki tane dallanma buyruğu görülmektedir. İlk dallanma geriye doğru, ikinci dallanma ileriye doğru yapılmaktadır. Bağıl dallanma buyruğunda yer alan adım miktarı tümleyen aritmetiğine göre verilmiş bir sayıdır. 8 bitlik işlemcilerde adım değeri, $80...$00...$7F arasında değişebilir. $80$00 arasındaki değerler geriye doğru dallanmaya $00$7F arasındaki değerler ileri doğru dallanmaya neden olur.

Koşulsuz Dallanma II

- DAL İLERİ buyruğunun bulunduğu adres $1059 dur. Dallanılacak adres $105E adresidir. Dolayısıyla atılacak adım sayısı 4 tür. Bu değer dallanma buyruğunun yanına DAL $04 biçiminde yazılır. Atılacak adım sayısını hesaplarken, ileri doğru ilk adımın 00 olarak değerlendirildiğine dikkat edilmelidir.

- Geri doğru dallanmada, dallanma buyruğunun adım alanına yazılacak değer, tümleyen aritmetik sayı çizelgesindeki eksi sayıdır. Geriye doğru dallanmada, adım değeri şöyle bulunabilir: Dallanma buyruğundan sonra ileri doğru ilk adımın sayısal karşılığı $00 ise, adım değerinin bulunduğu yerin değeri $FF dir. Dolayısıyla geriye doğru gidildiğinde adım değeri $FE, $FD, $FC biçiminde azalacaktır. Bu hesaplama yöntemiyle, DAL GERİ buyruğunda, GERİ etiketi yerine $FB yazmak gerekir.

PS	etiket	buyruk	adres hesabı			buyruk boyu
$1050		YÜK A, <$B000>	F8	F9	FA	3
$1053	GERİ	YAZ A, <$C0C0>	FB	FC	FD	3
$1056		DAL GERİ	FE	FF		2
$1058		ART A	00			1
$1059		DAL İLERİ	FE	FF		2
$105B		ART A	00			1
$105C		YAZ A, <$1000>	01	02	03	3
$105F	İLERİ	YÜK B, <$D000>	04	05	06	3
$1062		YAZ B, <$C0C1>	07	08	09	3

Koşullu Bağlanma

Koşullu dallanma buyruklarında, koşullar Durum Kütüğüne bakılarak değerlendirilir. Eğer koşul yerine gelmiş ise işlenenin belirttiği adrese gidilir. Eğer koşul yerine gelmiyor ise, program bir alt buyruktan devam eder. Dallanma buyrukları doğrudan ve bağıl adresleme kabul ederler. Daha önce değinildiği gibi, doğrudan adresleme kullanıldığında bu buyruklara Bağlanma Buyruğu adını veriyoruz.

Bazı mikroişlemcilerde DK'nın bayraklarına tek tek bakarak karar verilebilir; bazı mikroişlemcilerde ise DK bayraklarının birlikte yorumlanmasını sağlayan buyruklar bulunmaktadır.
DK bayraklarının tek tek değerlendirildiği buyruklara ilişkin örnekler aşağıda verilmiştir:

BAĞK S, $1000	sıfır bayrağı 1 ise $1000 adresine bağlan
BAĞK N, $1000	sonuç negatif ise $1000 adresine bağlan
BAĞK E, $1000	elde var ise $1000 adresine bağlan
BAĞK T, $1000	taşma var ise $1000 adresine bağlan

Koşullu Dallanma - I

DK bayraklarının topluca değerlendirildiği mikroişlemcilerde ise dallanma buyrukları tek bir durum bayrağını sorgulama biçiminde çalışmamaktadır. Bayraklar arasında VE, VEYA, YADA gibi işlemler gerçeklenerek, daha yetkin karşılaştırma ve karar verme olanakları elde edilmektedir.

DEE ADIM Dallan Eğer Eşit ise: Yani bir önceki adımda yapılan karşılaştırma işlemindeki iki sayı birbirine eşit ise veya son adımda bulunan sonuç sıfır ise dallanılır.

DED ADIM Dallan Eğer Eşit Değil ise: Yani bir önceki adımda yapılan karşılaştırma işlemindeki iki sayı birbirine eşit değil ise veya bir önceki adımdaki işlem sonucu sıfır değil ise dallanılır.

DEB ADIM Dallan Eğer Büyük ise: Bir önceki adımda yapılan karşılaştırma işleminde birinci işlenen ikinci işlenenden büyük ise dallan. Son işlemde elde edilen sayı sıfırdan büyük ise de dallanacaktır.

DBE ADIM Dallan Eğer Büyük veya Eşit ise: Bir önceki adımda yapılan karşılaştırma işleminde birinci işlenen ikinci işlenenden büyük veya eşit ise dallanılır. Son işlemde elde edilen sayı sıfır veya sıfırdan büyük ise de dallanacaktır.

DEK ADIM Dallan Eğer Küçük ise: Bir önceki adımda yapılan karşılaştırma işleminde birinci işlenen ikinci işlenenden küçük ise dallanılır. Son işlemde elde edilen sayı sıfırdan küçük ise de dallanacaktır.

DEI ADIM Dallan Eğer İri ise: Bir önceki adımda yapılan karşılaştırma işleminde birinci işlenen ikinci işlenenden iri ise, yani salt değer olarak büyük ise dallanılır.

DIE ADIM Dallan Eğer İri veya Eşit ise: Bir önceki adımda yapılan karşılaştırma işleminde birinci işlenen ikinci işlenenden iri, yani salt değer olarak büyük veya eşit ise dallanılır.

DEU ADIM Dallan Eğer Ufak ise: Bir önceki adımda yapılan karşılaştırma işleminde birinci işlenen ikinci işlenenden ufak ise, yani salt değer olarak küçük ise dallanılır.

Koşullu Dallanma - II

DTV ADIM Dallan Eğer Taşma varsa: Son işlemde taşma bayrağı 1 olmuş ise dallanılır.

DEV ADIM Dallan Eğer Elde varsa: Son işlemde elde bayrağı 1 olmuş ise dallanılır.

DYV ADIM Dallan Eğer Yarım Elde varsa: Son işlemde yarım elde bayrağı 1 olmuş ise dallanılır.

DTY ADIM Dallan Eğer Taşma yoksa: Son işlemde taşma bayrağı 0 olmuş ise dallanılır.

DEY ADIM Dallan Eğer Elde yoksa: Son işlemde elde bayrağı 0 olmuş ise dallanılır.

DYY ADIM Dallan Eğer Yarım Elde yoksa: Son işlemde yarım elde bayrağı 0 olmuş ise dallanılır.

Koşullu dallanma buyruklarının azaltma buyrukları ile birleştirilmiş olanları da bulunmaktadır. Bu tür buyruklar içinde en çok kullanılan bir örnek; azalt, eğer sıfır değilse dallan buyruğudur. Bu buyrukta azaltılan yer bir kütük ya da bellek içeriği olabilir.

ADED Ki,ADIM

ADED <BELLEK>,ADIM

Bu dallanma işleminin ilk aşamasında, işlenenin değeri 1 azaltılır ve sıfır olup olmadığı sınanır. Sıfır olmadı ise, belirtilen adrese dallanır; sıfır ise bir sonraki buyruktan devam edilir.

Altprograma Koşullu Dallanma

ALTK S, ADIM Bir önceki adımdaki sonuç sıfır ise adım miktarı kadar ötedeki altprograma dallan.

ALTK N, ADIM Bir önceki adımdaki sonuç negatif ise adım miktarı kadar ötedeki altprograma dallan.

ALTK E, ADIM Bir önceki adımda elde bayrağı 1 olmuş ise adım miktarı kadar ötedeki altprograma dallan.

ALTK T, ADIM Bir önceki adımda taşma bayrağı 1 olmuş ise adım miktarı kadar ötedeki altprograma dallan.

ALDK S, ADRES Bir önceki adımdaki sonuç sıfır ise adresi verilen altprograma dallan.

ALDK N, ADRES Bir önceki adımda negatif bayrağı 1 olmuş ise adresi verilen altprograma dallan.

ALDK E, ADRES Bir önceki adımda elde bayrağı 1 olmuş ise adresi verilen altprograma dallan.

ALDK T, ADRES Bir önceki adımda taşma bayrağı 1 olmuş ise adresi verilen altprograma dallan.

Altprograma Koşulsuz Dallanma

Altprogramlara dallanma işlemleri de normal dallanma ve bağlanma işlemleri gibi yerine getirilir. Altprogramın tamamlanması durumunda, tekrar dallanma noktasına, daha doğrusu dallanılan noktanın bir sonrasına dönülmesi gerektiği için, dallanma sırasında dönüş adresinin bir yere yazılması gerekir. Bu yer de yığındır.

ALTD ADRES	Verilen adresteki altprograma dallan.
ALT ADIM	Adım miktarı kadar ötedeki altprograma dallan.
ALTD $1000	Belli bir adreste bulunan altprograma bağlanma
ALT $25	$25 adım ötedeki altprograma dallan

Altprogramdan dönüşü belirtmek için DÖN buyruğu kullanılacaktır. Bu buyruk sayesinde, program hangi adrese geri döneceğini öğrenecektir.

Giriş-Çıkış Buyrukları - I

- Giriş-Çıkış buyrukları, bilgisayarın yapısına bağlı olarak değişmektedir. Bazı bilgisayarlarda giriş-çıkış arabirimleri bellek haritası içinde yer alırken bazı bilgisayarlarda giriş-çıkış arabirimleri bellek haritası dışında yer alırlar.
- Giriş-çıkış arabirimini bellek haritası içine yerleştiren bilgisayarlarda, bu arabirimlerle ilgili özel buyruklar yoktur. Arabirim içinde bulunan bir kütüğün içeriğinin okunması için YÜK ve bir kütüğe yazmak için YAZ buyrukları kullanılır.

YÜK A, <İSKELE>

- Giriş-çıkış arabiriminin giriş iskelesinin içeriği A akümülatörüne yüklenir. İskele adresi bellek içinde bir adrestir. Bu nedenle, bellek içinde aynı adreste bir bellek gözü olamaz.

YAZ A, İSKELE

- A akümülatörünün içeriğini çıkış iskelesine aktarır. Giriş iskelesine benzer şekilde, çıkış iskelesinin adresi de bellek içindedir ve aynı adreste bellek olamaz.

Giriş-Çıkış Buyrukları - II

- Giriş-çıkış arabirimini bellek haritası dışında bulunduran bilgisayarlarda giriş ve çıkış işlemleri için özel buyruklar bulunur.

  GİR A, <İSKELE>

- Giriş/çıkış arabiriminin giriş iskelesi içeriğini akümülatör A aktarır. Böylece bilgisayar dışındaki bir bilgi akümülatöre alınmış olur.

  ÇIK A, İSKELE

- Akümülatörün içeriğini, giriş-çıkış arabirimi, çıkış iskelesine aktarır. Böylece akümülatördeki veri bilgisayar dışına çıkarılmış olur.

  GİR A, <DURUM>

- Giriş-çıkış arabirimi durum kütüğünün içeriği, incelenmek üzere akümülatöre aktarılmış olur.

  ÇIK A, DENETİM

- Akümülatörün içeriği giriş-çıkış arabirimi denetim kütüğüne aktarılarak uygun denetim olanağı sağlanır.
- Giriş-çıkış arabirimini bellek haritası dışında tutan bilgisayarlarda, giriş-çıkış arabirimlerinin adresleri, bellek haritası dışında olduğu için aynı adreste bellek ve giriş-çıkış arabirimi olabilir.