

MİKROİŞLEMCİ SİSTEMLERİ

Yrd. Doç. Dr. Şule Gündüz Öğüdücü
www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

Adresleme Yöntemleri

- İşlenenin nerde olacağını belirtmek için kullanılır.
- Buyruk çözme aşamasında adresleme yöntemi belirlenir ve işlenenin nerede bulunacağı hesaplanır.
- Genel olarak 6 temel adresleme yöntemi vardır.
- Bir mikroişlemcide bulunan adresleme yöntemlerinin sayısının çokluğu yüksek düzeyli dillerdeki karmaşık işlemleri daha kolay yerine getirmesini sağlar.

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

2

Örnek MİB ile Adresleme

- Bir mikroişlemcinin adresleme yeteneği adres yollarını sayısı ile sınırlıdır.
 - 16 bit $\rightarrow 2^{16}$ farklı adres $\rightarrow 65.536$ byte
- Bellek adresleri, doğrudan buyrukta verilebilir ya da bir işaretçiden (başlangıç adresi) kaç adım ileride veya geride olduğu belirtilir.

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

3

Adresleme Yöntemleri

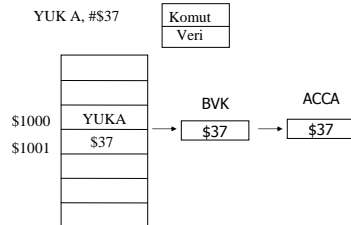
- Temel adresleme yöntemleri
 - İvedi Adresleme
 - Doğal Adresleme
 - Doğrudan Adresleme
 - Dolaylı Adresleme
 - Sıralı Adresleme
 - Bağlı Adresleme

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

4

İvedi Adresleme

- İşlenen yerine yazılan bilgi bir adres değil, işlenenin kendisidir.
- Bellek okuma yazma işlemi yapılmaz.
- Hızlı
- Örnek:
 - YUK B, \$41
 - YUK SK, #1000

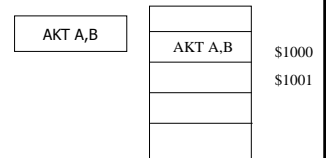


www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

5

Doğal Adresleme

- Buyrukta bellek adresi kullanılmaz.
- İşlenen alanında bulunan bilgi, işlenenin bulunduğu kütüğün adıdır.
- Sınırlı sayıda kütük adresleme
- Hızlı işlem
- Örnek:
 - AKT SK, CD SK \leftarrow CD
 - AKT B, C A \leftarrow B
 - AKT CD, AB C \leftarrow A; D \leftarrow B

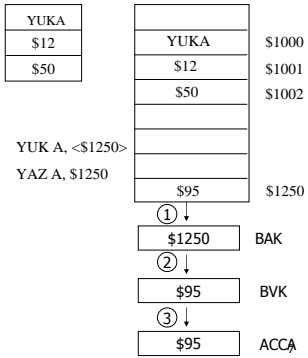


www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

6

Doğrudan Adresleme

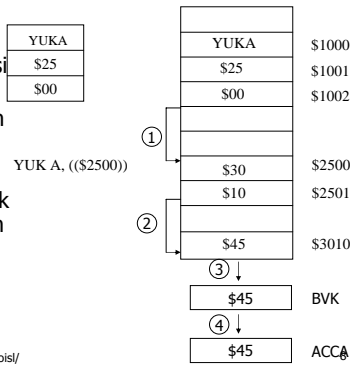
- İşlenen yerinde, işlenecek verinin bulunduğu ya da bulunacağı bellek gözünün adresi yazılıdır.
- Veriye ulaşmak için sadece bir bellek işlemi
- Etkin adres hesaplama yapılmaz



www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

Dolaylı Adresleme

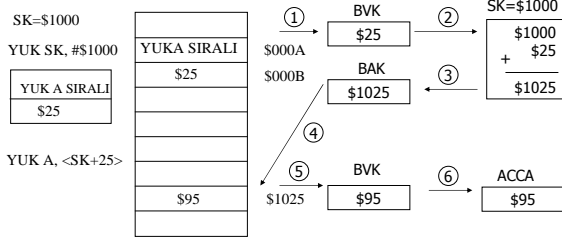
- İşlenen yerinde verinin yazılı olduğu bellek gözünün adresi bulunmaktadır.
- İşlenene ulaşmak için bellek işlemi fazla
- Yavaş
- Bağlantılı liste, kuyruk gibi işaretçi kullanılan veri yapılarını yaratmak için kullanılır.



www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

Sıralı Adresleme

- MİB içinde bulunan sıralama kütüğüne dizinin ilk elemanının adresi yazılır. Veriler üzerinde işlem yaparken verinin sıra numarasını belirtmek yeterli olur.

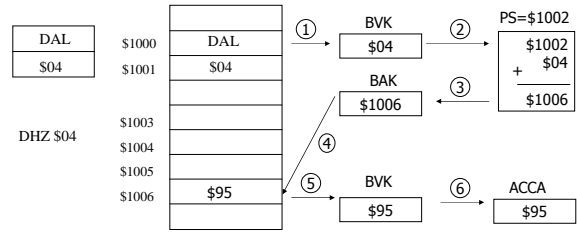


www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

9

Bağlı Adresleme

- Bu adresleme yöntemi dallanma komutları için kullanılır. İşlenen yerinde görülen sayı bir sonraki komutun program sayacındaki adresten ne kadar uzakta olduğunu gösterir. Dallanma komutları durum kütüğündeki bazı bitlere göre koşullu veya koşulsuz (DHZ) olabilir.



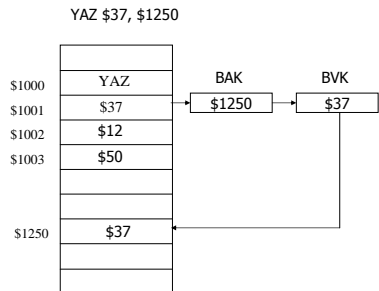
www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

10

Gelişmiş Adresleme Yöntemleri

- Belleğe İvedi Adresleme:

- Veri doğrudan bellekte bir adrese yazılır.
- Akümülatör ya da yardımcı kütüklerin içeriği bozulmaz.



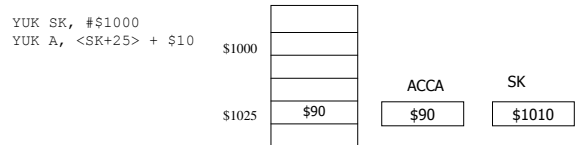
www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

11

Gelişmiş Adresleme Yöntemleri

- Artırmalı Sıralı Adresleme

- Sıralı adreslemeye olduğu gibi etkin adres hesaplanır, ardından sıralama kütüğü içeriği buyrukta belirtilen değer kadar artırılır.
- Artırma değeri \$00-\$FF arasında olabilir.



www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

12

Veri Aktarma Buyrukları

- Yazma:** MİB içindeki kütüklerin içerikleri bellek gözlerine aktarılır.

YAZ K_i , <BELLEK> <BELLEK> $\leftarrow K_i$

YAZ A, \$1000 \$1000 \leftarrow ACC A

YAZ C, \$E000 \$E000 \leftarrow C

YAZ SK, \$C000 \$C000 + \$C001 \leftarrow SK

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

19

Veri Aktarma Buyrukları

- Takas:** MİB içindeki bazı kütüklerin içeriklerini birbirleri ile takas etmelerini sağlar.

TKS K_i , K_j $K_i \leftrightarrow K_j$

TKS A, B ACC A \leftrightarrow ACC B

TKS C, D C \leftrightarrow D

TKS SK, YG SK \leftrightarrow YG

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

20

Aritmetik İşlem Buyrukları

- Toplama:** İki akümülatörün içeriği, bir akümülatör ile bir yardımcı kütüğün içeriği, bir akümülatörün içeriği ile bir veri veya bir akümülatör ile bir bellek gözünün içeriği eldeli veya eldesiz olarak toplanabilir. Sonuç buyruktaki birinci işlenen durumunda olan akümülatöre yazılır.

TOP A, B ACCA \leftarrow ACCA+ACCB

TOP A, K_i ACCA \leftarrow ACCA+ K_i

TOP A, VERİ ACCA \leftarrow ACCA+VERİ

TOPA, <BELLEK> ACCA \leftarrow ACCA+ <BELLEK>

Eldeli Toplama

TOPE A, B ACCA \leftarrow ACCA+ACCB+E

TOPE A, K_i ACCA \leftarrow ACCA+ K_i +E

TOPE A, VERİ ACCA \leftarrow ACCA+VERİ+E

TOPE A, <BELLEK> ACCA \leftarrow ACCA+ <BELLEK>+E

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

21

Toplama İşlemi

TOP A, B ACC A \leftarrow ACC A + ACC B

TOPE A, B ACC A \leftarrow ACC A + ACC B + E

TOP A, # \$25 ACC A \leftarrow ACC A + \$25

TOP A, <\$1000> ACC A \leftarrow ACC A + <\$1000>

TOPE A, <SK+10> ACC A \leftarrow ACC A + <SK+10> + E

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

22

Toplama İşlemi

- Örnek:** X=\$40A0 ve Y=\$1BC1 sayılarının toplanması

<10>	<11>	X	<10>:0100 0000
<12>	<13>	Y	<11>:1010 0000
+			<12>:0001 1011
<14>	<15>	SONUÇ	<13>:1100 0001

YUK A, <\$0010> ACCA \leftarrow 0100 0000

YUK B, <\$0011> ACCB \leftarrow 1010 0000

TOP B, <\$0013> ACCB \leftarrow 0110 0001 **E=1**

TOPE A, <\$0012> ACCA \leftarrow 0101 1100

YAZ A, <\$0014> \$0014 \leftarrow 0101 1100

YAZ B, <\$0015> \$0015 \leftarrow 0110 0001

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

23

Aritmetik İşlem Buyrukları

- Çıkarma:** İki akümülatörün içeriği, bir akümülatör ile bir yardımcı kütüğün içeriği veya bir akümülatör ile bir bellek gözünün içeriği eldeli veya eldesiz olarak birbirinden çıkarılabilir. Sonuç buyruktaki birinci işlenen durumunda olan akümülatöre yazılır.

ÇIK A, B ACCA \leftarrow ACCA-ACCB

ÇIK A, K_i ACCA \leftarrow ACCA- K_i

ÇIK A, VERİ ACCA \leftarrow ACCA-VERİ

ÇIK A, <BELLEK> ACCA \leftarrow ACCA- <BELLEK>

Borçlu Çıkarma

ÇIK E A, B ACCA \leftarrow ACCA-ACCB-E

ÇIK E A, K_i ACCA \leftarrow ACCA- K_i -E

ÇIK E A, VERİ ACCA \leftarrow ACCA-VERİ-E

ÇIK E A, <BELLEK> ACCA \leftarrow ACCA- <BELLEK>-E

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

24

Aritmetik İşlem Buyrukları

Çarpma:

- Birinci işlenen A akümülatörü olmalıdır.
- İkinci işlenen, B akümülatöründe, yardımcı kütüklerden birinde, bir bellek gözünde bulunabilir ya da bir veri olabilir.
- İşlem sonucu A ve B akümülatör çiftinde yer alır.
- İşlenenlerin işaretli olması gerekir.

ÇAR A, B ACCA + ACCB ← ACCA * ACCB
 ÇAR A, Ki ACCA + ACCB ← ACCA * Ki
 ÇAR A, VERİ ACCA + ACCB ← ACCA * VERİ
 ÇAR A, <BELLEK> ACCA + ACCB ← ACCA * <BELLEK>

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

25

Aritmetik İşlem Buyrukları

Bölme:

- Birinci işlenen A ve B akümülatör çiftinde bulunur.
- İkinci işlenen yardımcı kütüklerden birinde, bir bellek gözünde bulunabilir ya da bir veri olabilir.
- Sonuç A ve B akümülatörlerinde, kalan C yardımcı kütüğünde oluşur.
- İşlenenlerin işaretli olması gerekir.
- 0'a bölme durumunda taşma bayrağı 1 olur.

BÖL AB, Ki ACCA + ACCB ← <ACCA + ACCB> / Ki
 BÖL AB, VERİ ACCA + ACCB ← <ACCA + ACCB> / VERİ
 BÖL AB, <BELLEK> ACCA + ACCB ← <ACCA + ACCB> / <BELLEK>

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

26

Örnek

- Bellekte \$1000 ve \$1001 bellek gözlerinde bulunan sayı \$1002 bellek gözünde bulunan sayıya bölünecek, sonuç \$1005-\$1006 bellek gözlerine, kalan ise \$1007 bellek gözüne yazılacaktır.

başla yük sk, \$1000
 yük a, <sk+0> + \$01
 yük b, <sk+0> + \$01
 böl ab, <sk+0>
 yaz ab, <\$1005>
 yaz c, <\$1007>

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

27

Örnek

- Bellekte \$1000 ve \$1001 bellek gözlerinde bulunan sayılar çarpılarak \$1002 bellek gözünde bulunan sayıya bölünecek, sonuç \$1005-\$1006 bellek gözlerine, kalan ise \$1007 bellek gözüne yazılacaktır.

başla yük sk, \$1000
 yük a, <sk+0> + \$01
 çar a, <sk+0> + \$01
 böl ab, <sk+0>
 yük sk, \$1005
 yaz ab, <sk+0> + \$02
 yaz c, <sk+0>

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

28

Lojik İşlem Buyrukları

- VE, VEYA, YADA:** Lojik buyruklarda birinci işlenen akümülatör olmak zorundadır. Bir akümülatörün içeriği bir veri, diğer bir akümülatör, yardımcı kütük veya bir bellek gözünün içeriği ile lojik işleme sokulabilir.

İŞLEM A, VERİ ACCA ← ACCA.İŞLEM.VERİ
 İŞLEM A, B ACCA ← ACCA.İŞLEM.ACCB
 İŞLEM A, Ki ACCA ← ACCA.İŞLEM.Ki
 İŞLEM A, <BELLEK> ACCA ← ACCA.İŞLEM. <BELLEK>

VE A, \$25 ACC A ← ACC A .VE. \$25
 VEYA A, B ACC A ← ACC A .VEYA. B
 YADA A, <\$1000> ACC A ← ACC A .YADA. <\$1000\$

www.cs.itu.edu.tr/~gunduz/courses/mikroisl/

29