

# BİLGİSAYAR ORGANİZASYONU ve TASARIMI

YRD. DOÇ. DR. FATİH KELEŞ

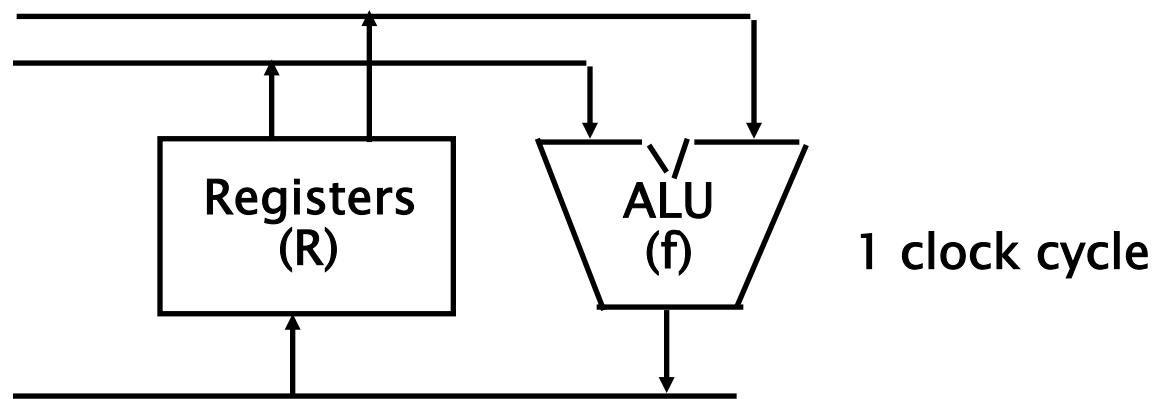
- ▶ Register (Saklayıcı)
- ▶ Register Transfer Dili (RTL)
- ▶ Register Transferi
- ▶ Ortak Yol (Bus) ve Bellek Transferi
- ▶ Aritmetik Mikroişlemleri
- ▶ Lojik Mikroişlemleri
- ▶ Öteleme Mikroişlemleri
- ▶ Aritmetik Lojik Öteleme Birimi (ALU)

- ▶ Saklayıcı Gösterimi
- ▶ Saklayıcı Aktarım Dili (RTL: Register Transfer Lang.)
- ▶ Saklayıcılar Arası Aktarım
- ▶ Ortak Yol (Bus) ve Bellek Aktarımı

# Mikroişlemler

- ▶ Saklayıcılarda tutulan verilerle yapılan en temel işlemlerdir.
- ▶ Örnek Mikroişlemler:
  - Shift
  - Load
  - Clear
  - Increment
  - ...

# Mikroişlemeler



# Register Transfer Dili - RTL

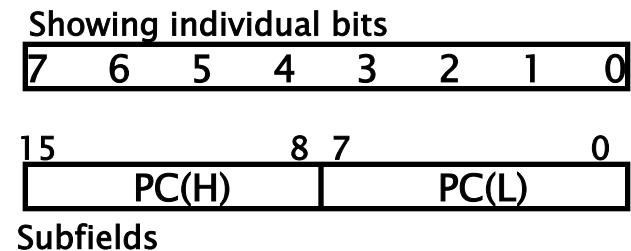
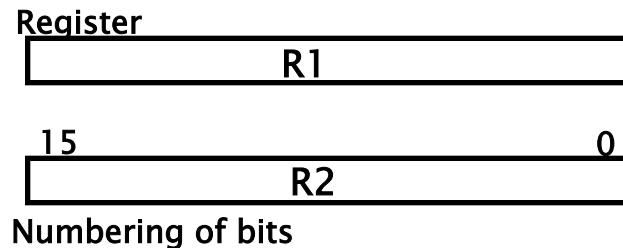
- ▶ Bir dijital sistemi kelimelerle tanımlamak yerine özel bir gösterim kullanılır:  
***saklayıcı aktarım dili - register transfer dili (RTL)***
- ▶ Register Transfer Dili, bilgisayarın herhangi bir fonksiyonu için gereken mikroişlemleri tanımlamada kullanılır.
- ▶ Register transfer dili, sembolik bir dil olarak
  - Dijital bilgisayar iç organizasyonunu tanımlamak için en uygun yoldur.
  - Dijital sistemin tasarım prosedürünü basitleştirmek için de kullanılır.

# Register Gösterimi

- ▶ Saklayıcılar büyük harflerle gösterilir. Bazen büyük harfleri rakamlar takip eder (örneğin A, R1, IR)
- ▶ Çoğunlukla saklayıcı isimleri fonksiyonunu gösterir:
  - MAR = memory address register
  - PC = program counter
  - IR = instruction register
- ▶ R1, R2, ... Buradaki R, Register' dan gelmektedir.
- ▶ Bir saklayıcı, genellikle dikdörtgen ve ismi içinde olarak gösterilir. Her biti ayrıca gösterilebilir.
  - Saklayıcı ve içerikleri çeşitli şekillerde gösterilebilir ve simgelenebilir:

# Register Gösterimi

- Bir register gösterimi
  - register
  - register bölümü
  - register bitleri
- Bir register blok diyagramını çizmenin değişik yolları vardır.



# Register Gösterimi

- ▶ Saklayıcı



- ▶ 16-bit Saklayıcı



- ▶ 8-bit bir Saklayıcının bitleri



- ▶ 16-bit Saklayıcının ikiye bölünmesi:

- 0-7nci bitler **Düşük anlamlı**,
- 8-15nci bitler **Yüksek anlamlı**.



# Register Transferi

- ▶ Register transferi bir saklayıcıdan diğerine içerik kopyalaması şeklinde olabilir:
- ▶ Bu durumda register transfer aşağıdaki gibi gösterilir:

$R2 \leftarrow R1$

- register R1 (kaynak) içeriği register R2 ye (hedef) kopyalanır (load)
- kaynaktan (R1) hedefe (R2) tüm bitlerin aynı anda transferi bir saat çevriminde gerçekleşir.
- Bu durumda R1 içeriği **değişmez** yani işlem *non-destructive* bir işlemidir.

# Register Transferi

- register transferi aşağıdaki gibi ise;

$$R1 \leftarrow R2$$

## Dijital sistem

- Kaynaktan (R2) hedefe (R1) tüm data hatlarına (data lines)
- Hedef register (R1) için paralel yükleme özelliğine
- İşlemi sağlayacak kontrol hatlarına (Control lines) sahip olmalıdır.

# Kontrol Fonksiyonları

- ▶ Pek çok işlem belirli bir şartın doğrulanmasına bağlı olarak icra edilir.
- ▶ Yüksek seviyeli programlama dilinde bu durum “if” ifadelerine (statement) karşılık gelir.
- ▶ Dijital Sistemlerde bu ifade, kontrol fonksiyonu adı verilen kontrol işaretleri yardımıyla sağlanır.
  - Kontrol işaretti 1 ise, işlem gerçekleşir.  
RTL kullanılarak; aşağıdaki gibi yazılabilir.

if ( $P=1$ ) then  $R2 \leftarrow R1$

P:  $R2 \leftarrow R1$

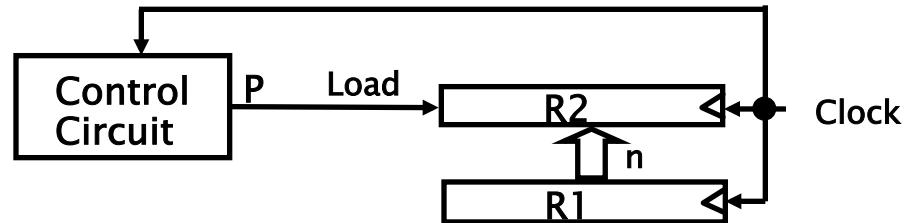
$P = 1$  ise, register  $R1$  içeriğinin register  $R2$  ye transfer işlemi gerçekleşir.

# Transferin Donanımsal Gerçeklemesi

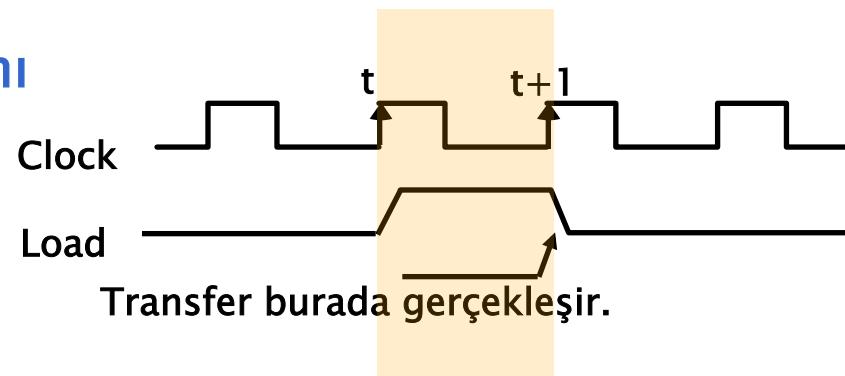
Kontrol İşareti P

Blok diyagram

P:  $R2 \leftarrow R1$



Zamanlama diyagramı



- Saat işaretini, hem hedef registeri hem de kontrol işaretini üreten devreyi kontrol eder.
- Registerler pozitif kenar tetiklemeli flip floplardan oluşmaktadır.



# Aynı Anda Gerçeklenen İşlemler

- ▶ İki yada daha fazla işlem aynı anda gerçekleşirse, (,) işaretini kullanılarak işlemler birbirinden ayrılır.

P: R1  $\leftarrow$  R2 , MAR  $\leftarrow$  IR

- ▶ Burada, kontrol işaretti P = 1 ise, load R2 içeriği R1 e transfer edilirken, aynı saat çevriminde, IR içeriği de register MAR ye transfer edilir.

# Register Transfer İçin Temel Semboller

Semboller	Tanım	Örnekler
Büyük Harfler & rakamlar	Bir register simgeler	MAR, R2
Parentez ()	Registerin bir kısmını simgeler	R2(7-0), R2(L)
Ok ←	Transfer bildirir (kaynaktan-hedefe )	R2 ← R1
Kolon :	Kontrol fonksiyonunun sonlandırımı	P:
Virgül ,	İki mikroişlemi ayırır	A ← B, B ← A

# Sistem Modüllerini Bağlama

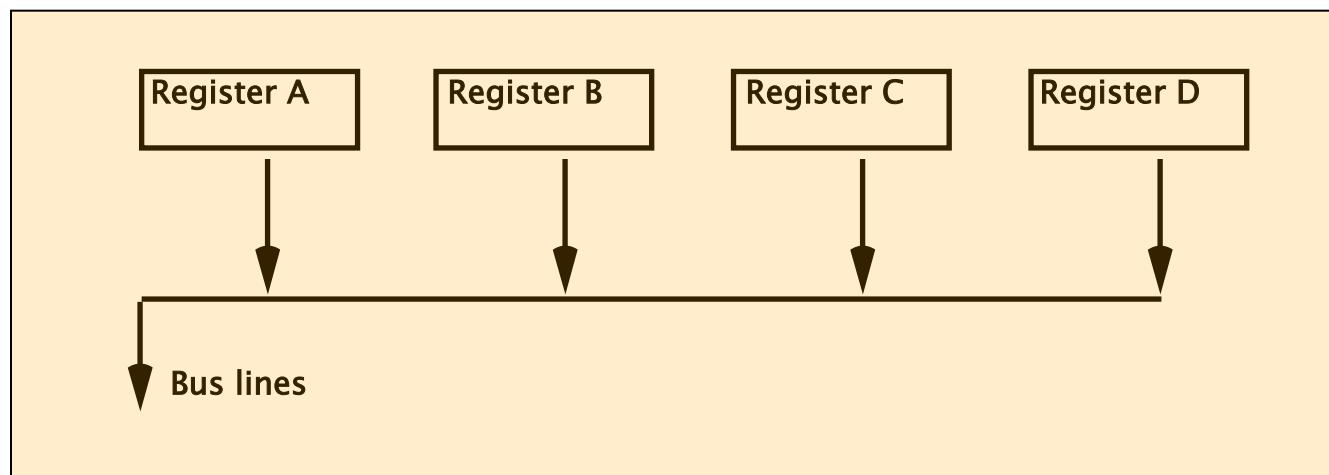
- ▶ Çok saklayıcılı bir dijital sistemde, tüm saklayıcıların birbirlerine içeriklerini transfer etmeyi sağlayacak data ve kontrol hatlarını doğrudan oluşturmak olanaksızdır.
  - ▶ Böyle bir sistem için,  
 **$n$  saklayıcı  $\rightarrow n(n-1)$  hat**  
 **$O(n^2)$**  ile orantılı bir maliyet gereklidir.
    - Büyük dijital sistemler için gerçekleştirilebilir bir yaklaşım değildir.
    - Bunun yerine, data transferi için bir ortak yol (bus) kullanımı esas alınır.
- Kaynak ve hedef saklayıcıları seçmek için ise, kontrol devreleri kullanılır.

# Ortak Yol (Bus) ve Transferi

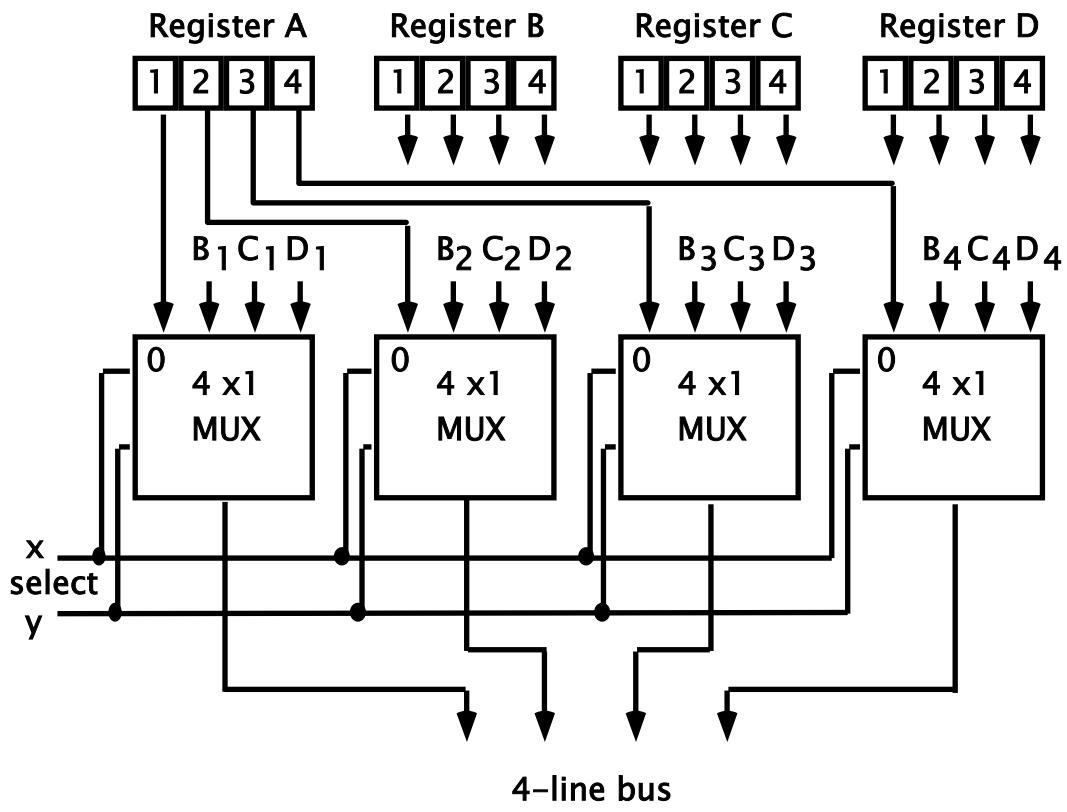
Ortak Yol bir grup telden oluşan bir hattır. Ortak yol üzerinden gönderilen veriler, pek çok kaynak saklayıcısından pek çok hedef saklayıcılarına transfer edilir.

Saklayıcıdan ortak yola data transferi

BUS ← R

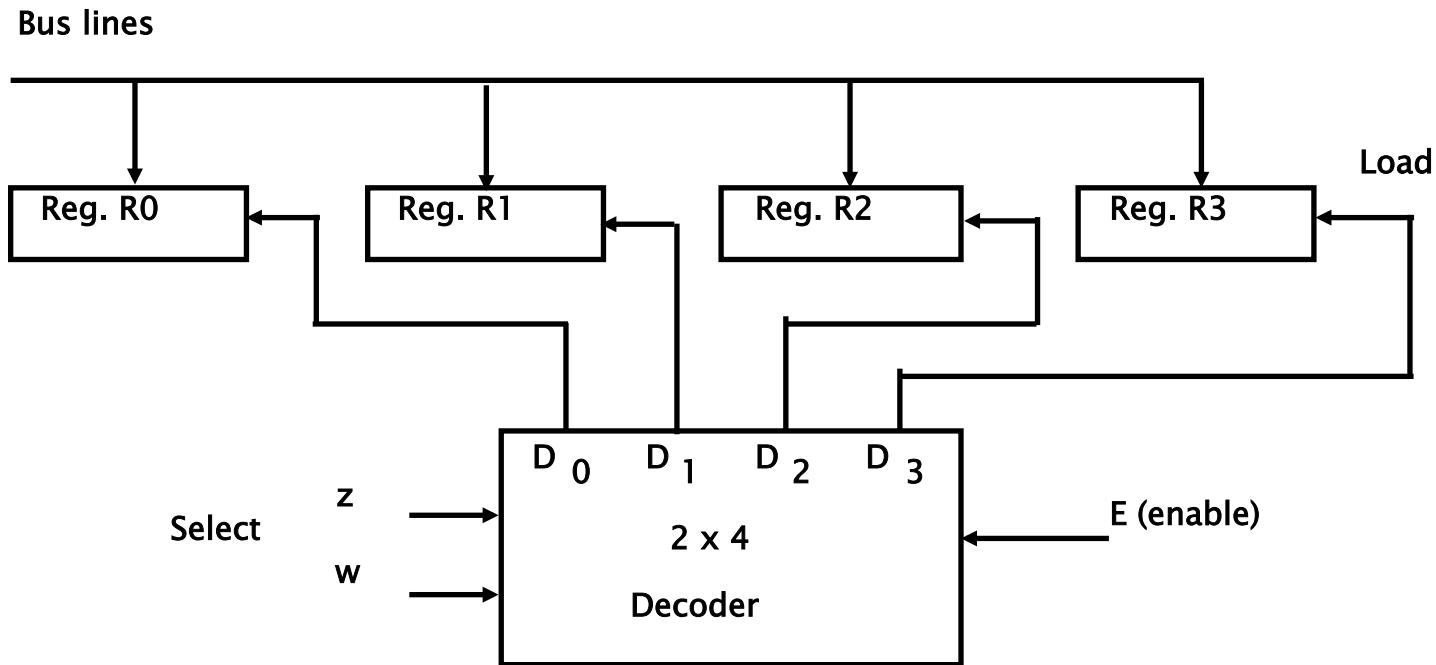


# Ortak Yol ve Transferi



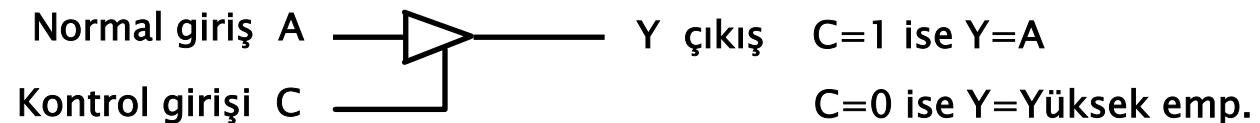
- ▶ 4-bit veriyolu yapmak için;
  - 4 tane 4-bit saklayıcı
  - 4 tane 4x1 MUX
  - Veriyolu seçimi S1 ve S0
  
- ▶ 16-bit 8 saklayıcı ile
  - 16 tane 8x1 MUX
  - Bit sayısı=MUX sayısı
  - Saklayıcı sayısı=MUX büyütüğü

# Ortak Yoldan Bir Hedef Saklayıcıya Transfer

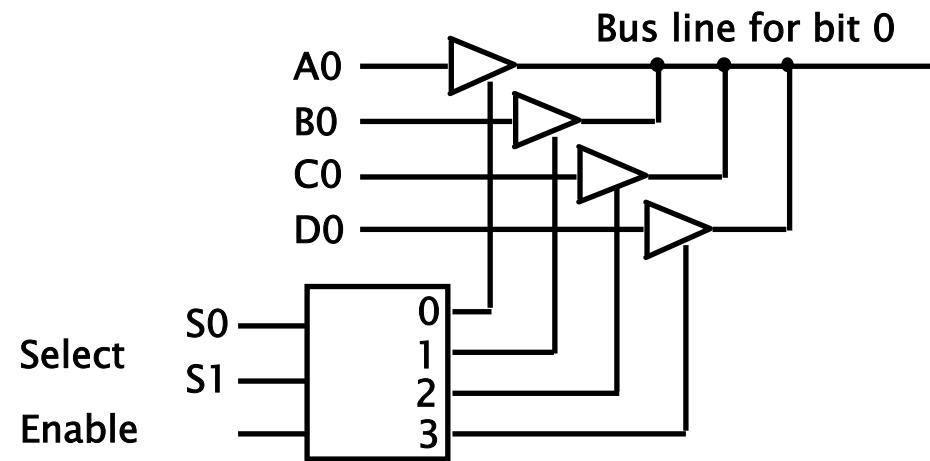


# Üç Durumlu Yol Ayırıcıları ile Ortak Yol Sistemi

## Üç durumlu Yol Ayırıcıları (Three-State Bus Buffers)



## Üç durumlu Yol Ayırıcıları ile oluşturulan Ortak Yol Sistemi



# Ortak Yol Transferi (RTL Gösterimi)

- ▶ Ortak Yol üzerinden register transferi aşağıdaki gibi gösterilebilir:

$R2 \leftarrow R1$

Implicit gösterim

veya

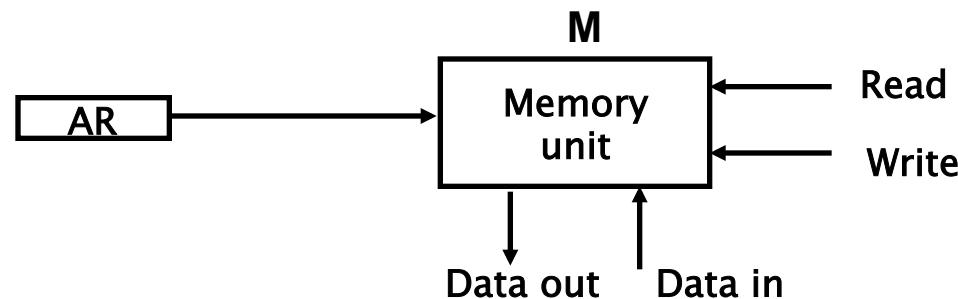
$BUS \leftarrow R1, R2 \leftarrow BUS$

Explicit gösterim

- ▶ Implicit gösterimde, ortak yol görünmez. (kapalı gösterim)
- ▶ Explicit gösterimde, dolaylı olarak ortak yol görünür. (açık gösterim)

# Bellek Transferi

- ▶ Bellek (M) register düzeyinde bir eleman olarak görülebilir.
- ▶ Bellek pekçok gözden oluştuğu için, bellek adresinin belirlenmesi gereklidir.
- ▶ Bu da bellek referansı kullanılarak gerçekleştirilir.
- ▶ *Memory Address Register (MAR, or AR)* gibi özel bir saklayıcıda hedef adresi tutarak bir bilgisayar sisteminde belleğe erişim sağlanır.
- ▶ Belleğe erişildiğinde, MAR içeriği bellek ünitesinin adres hattına gönderilir.



# Bellek Okuma

- ▶ Bellekte bir gözden bir değer okuma ve onu bir saklayıcıya yükleme işlemi RTL olarak gösterimi;

Read: DR  $\leftarrow$  M[AR]

- ▶ Bu işlem aşağıdaki alt işlemlere neden olur:
  - AR içeriği bellek adres hattına gönderilir.
  - Read (= 1) işaretini bellek ünitesine gönderilir.
  - Belirlenen adressteki data bellek çıkış hattına gönderilir.
  - Bu data DR saklayıcısına yüklenmek üzere ortak yol üzerinden gönderilir.

# Bellek Yazma

- ▶ Bellekte bir alana değer yazma RTL dilinde

Write:  $M[AR] \leftarrow DR$

Bu işlem aşağıdaki alt işlemlere neden olur:

- AR içeriği bellek adres hattına gönderilir.
- Write (= 1) işaretini bellek ünitesine gönderilir.
- DR saklayıcısındaki data ortak yol üzerinden bellek data giriş hattına gönderilir.
- Bu data bellekte belirlenen adrese yüklenir.

# Mikroişlemeler (RTL Gösterimi)

$A \leftarrow B$

$AR \leftarrow DR(AD)$

$A \leftarrow \text{constant}$

$ABUS \leftarrow R1$

$R2 \leftarrow ABUS$

AR

DR

$M[R]$

M

$DR \leftarrow M$

$M \leftarrow DR$

Transfer content of reg. B into reg. A

Transfer content of AD portion of reg. DR into reg. AR

Transfer a binary constant into reg. A

Transfer content of R1 into bus A

Transfer content of bus A into R2

Address register

Data register

Memory word specified by reg. R  
Equivalent to  $M[AR]$

Memory *read* operation: transfers content of  
memory word specified by AR into DR

Memory *write* operation: transfers content of  
DR into memory word specified by AR