

Bilgisayar Mimarileri

İşlemci Tasarımına Giriş

İşlemci Tasarımına Giriş

Okuma Listesi

Gerekli

- Computer Organization and Design: The Hardware Software Interface [RISC-V Edition] David A. Patterson, John L. Hennessy
 - 4. Bölüm (4.1, 4.3)

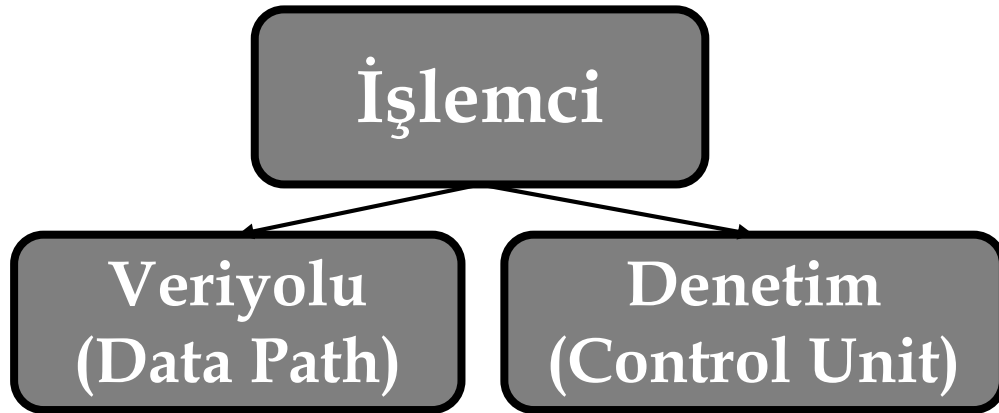
Önerilen

- Computer Organization and Design: The Hardware Software Interface [RISC-V Edition] David A. Patterson, John L. Hennessy
 - 4. Bölüm (4.2)

Giriş

İşlemci bir bilgisayarın **en temel** parçasıdır.

- Aritmetik işlemler,
- Bellekteki veri aktarımları ve
- Giriş çıkış verilerinin kaydedilmesi işlemci tarafından gerçekleştirilir.



Giriş

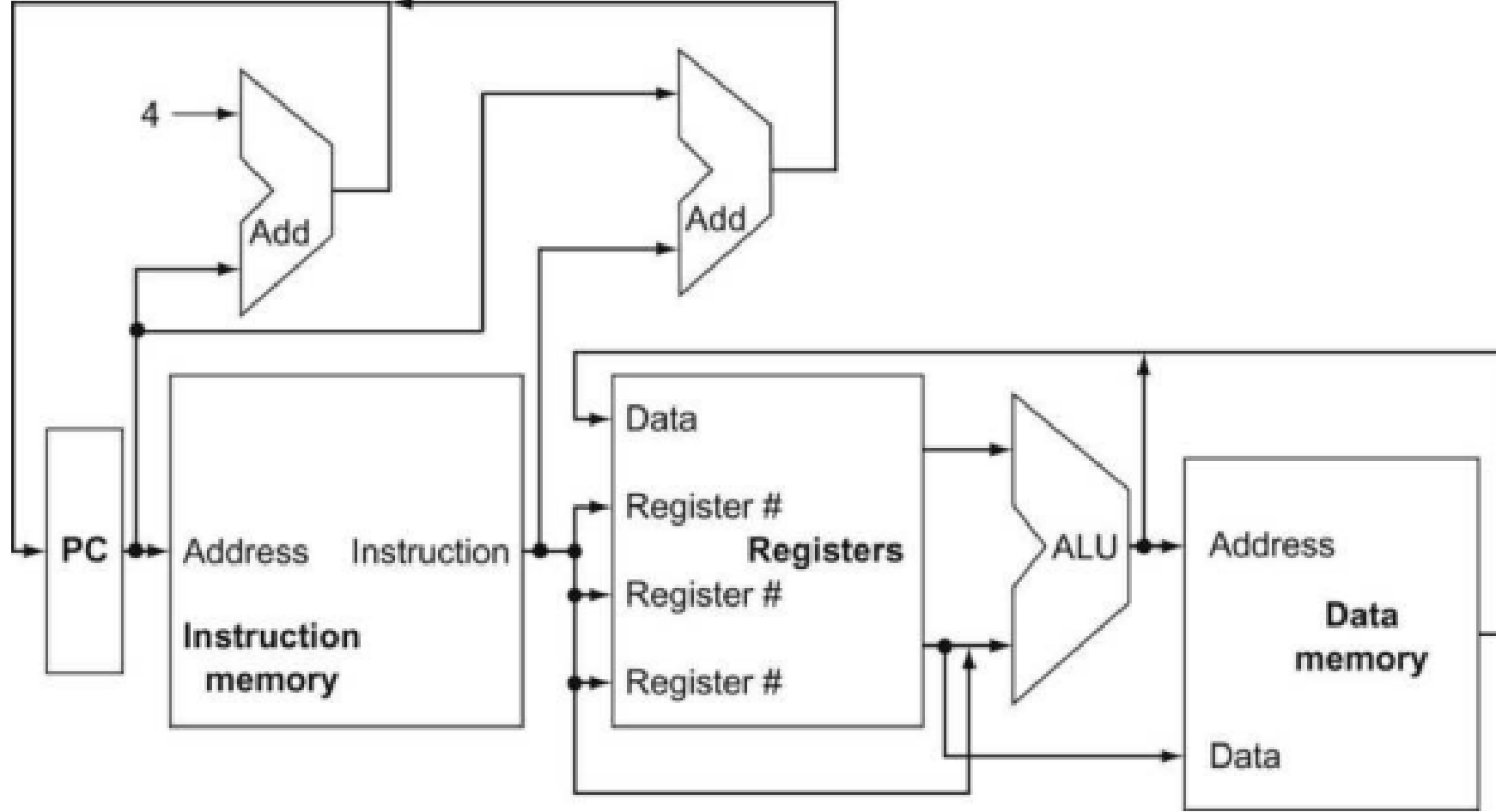
- Bu bölümde bir işlemcinin tasarımında kullanılan ilke ve teknikleri inceleyecek ve açıklamaya çalışacağız.
- Bunun sonunda, bir veri yolu oluşturan ve RISC-V gibi bir komut setini uygulamaya yetecek kadar basit bir işlemci versiyonunu oluşturacağız.

Uygulamaya Genel Bakış

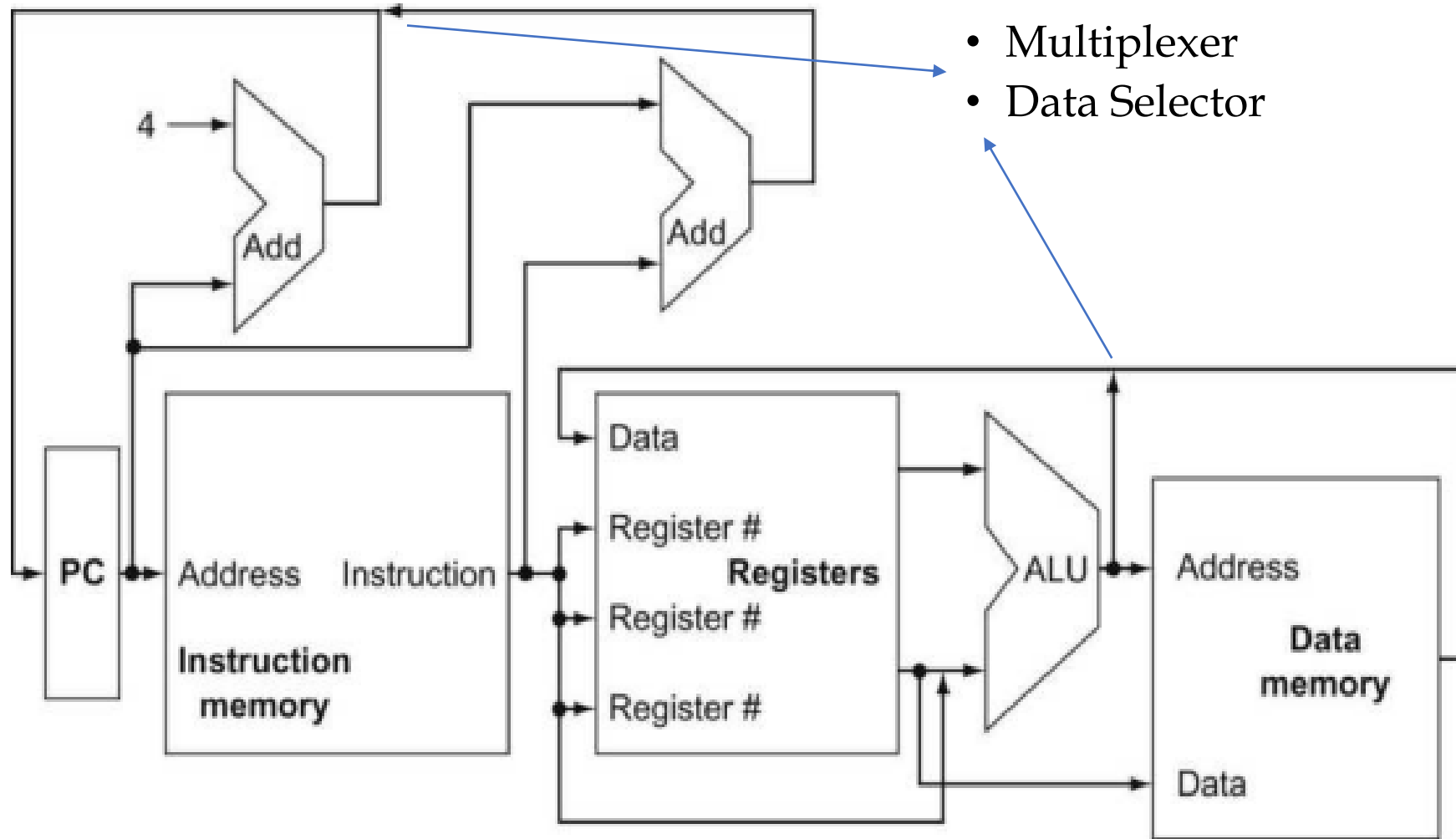
- Her komutun işlenmesinde ilk iki adım aynıdır:
- 1. Program sayacını (PC) kodu içeren hafızaya gider ve komutu bu hafızadan alır (Fetch).
- 2. Okunacak registerları seçmek için komut alanlarını kullanarak bir veya iki registerı okur.
- **ld** komutu için yalnızca bir Registerı okumamız gerekir, ancak diğer komutların çoğu iki Registerın okunmasını gerektirir.

Uygulamaya Genel Bakış

- Ana fonksiyonel birimleri ve bunlar arasındaki ana bağlantıları gösteren RISC-V alt kümesinin uygulanmasına ilişkin soyut bir görünüm.



Uygulamaya Genel Bakış



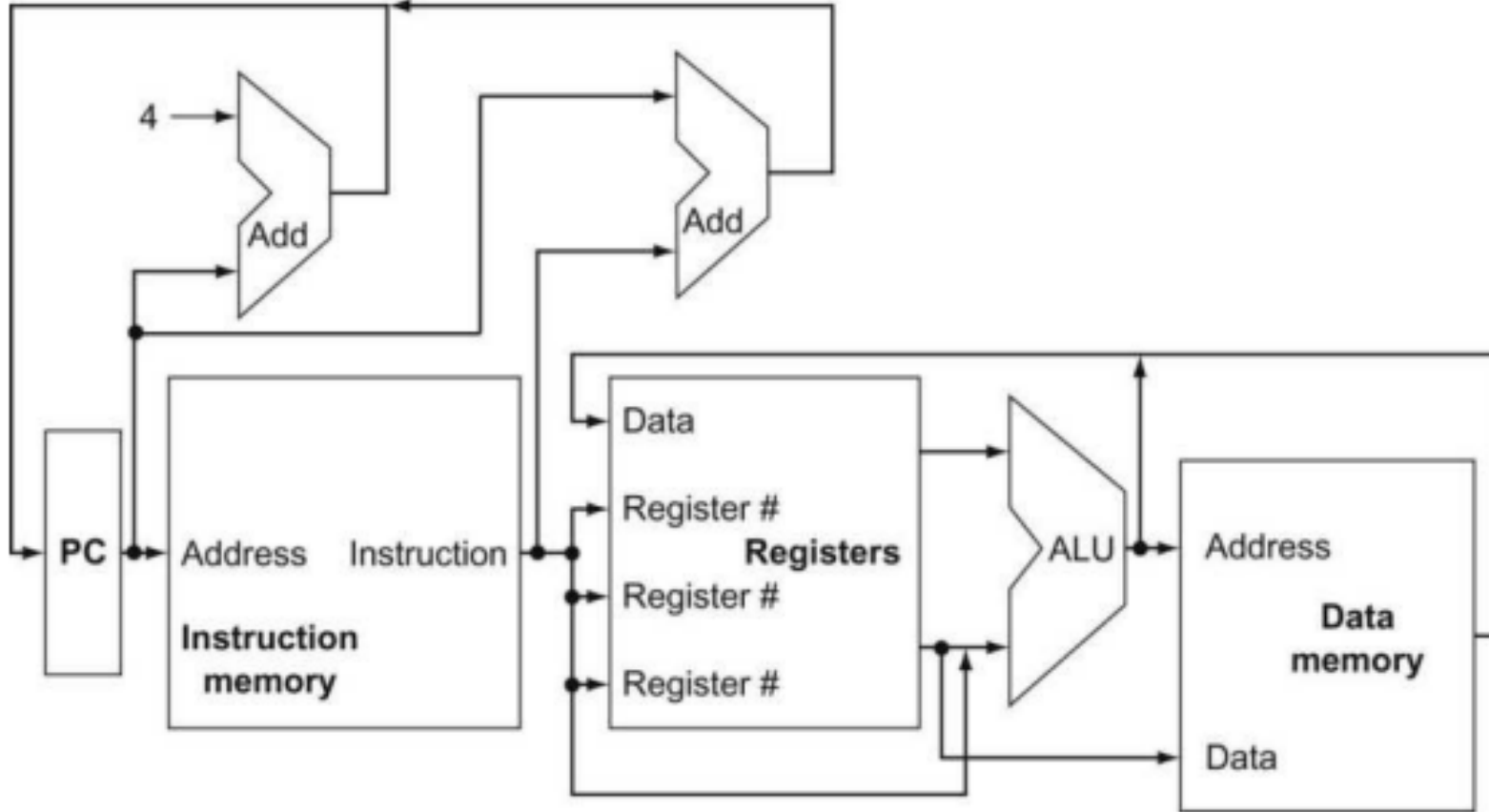
- Multiplexer
- Data Selector

- Belirli bir birime giden verilerin iki farklı kaynaktan geldiğini görüyoruz.

Pratikte bu veri hatları basitçe birbirine bağlanamaz.

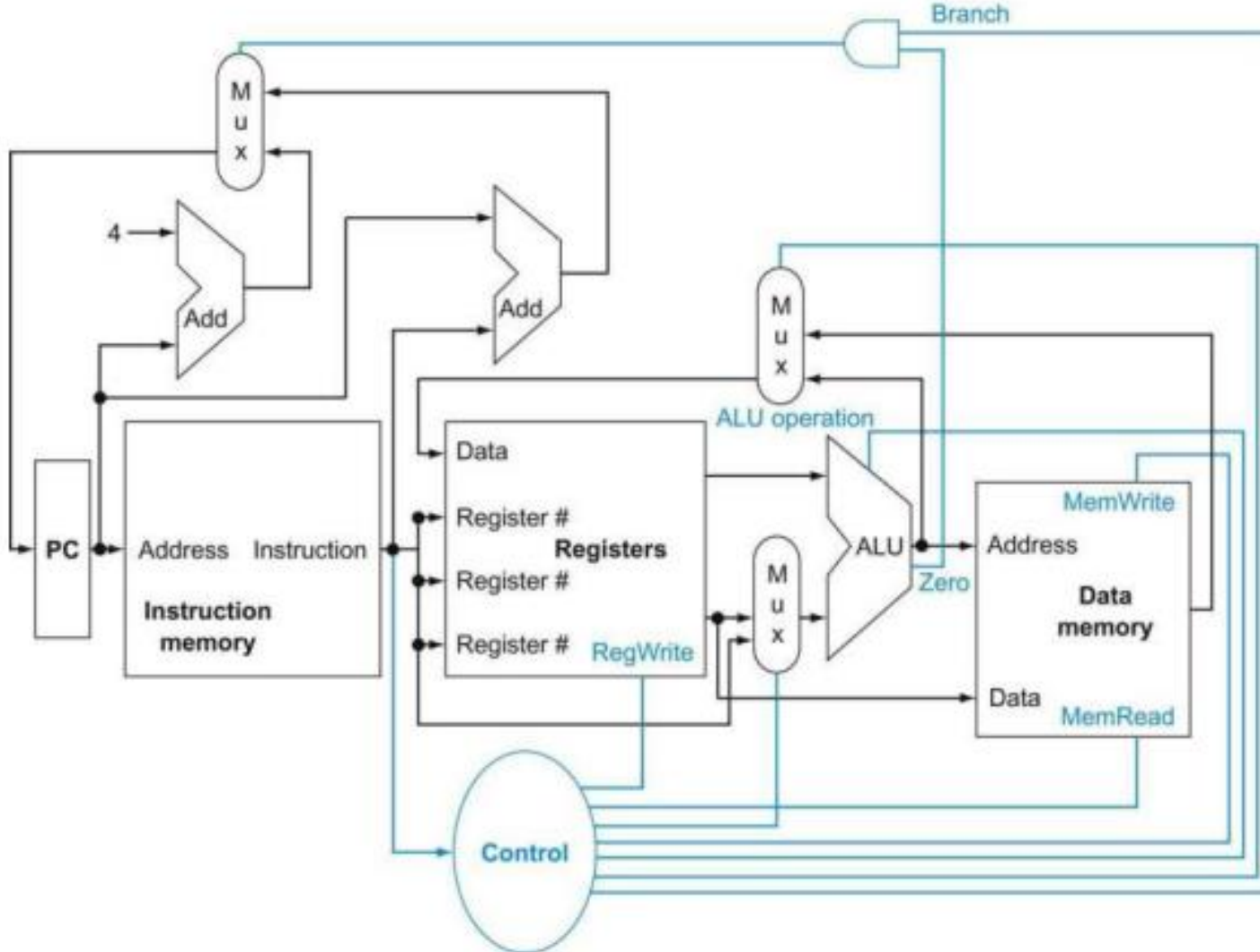
Birden fazla kaynak arasından seçim yapan ve bu kaynaklardan birini hedefine yönlendiren bir mantık unsuru eklemeliyiz.

Uygulamaya Genel Bakış



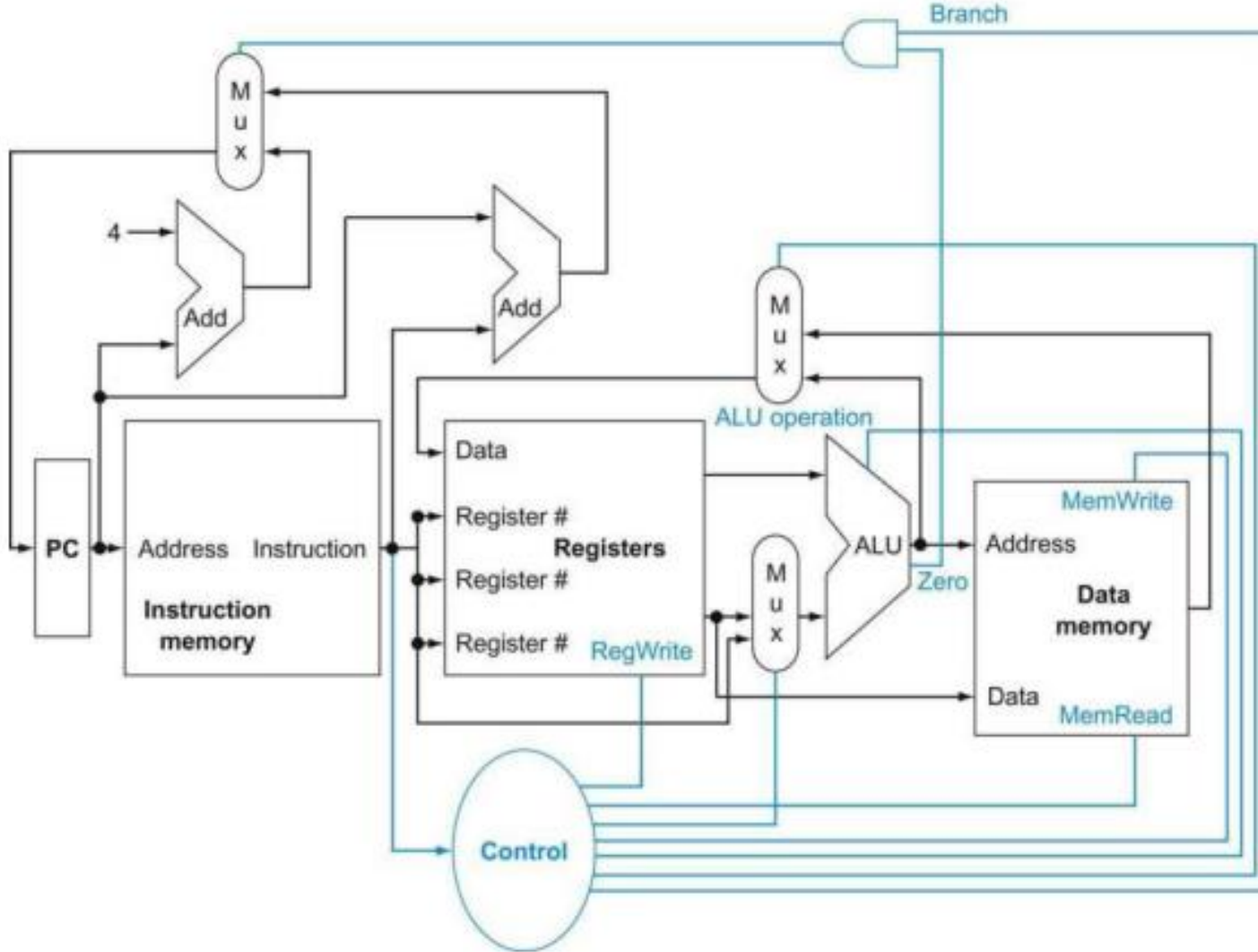
- Komut türüne bağlı olarak birkaç birimin **kontrol edilmesi** gerektirir.
- Örneğin veri belleğinin bir load komutunda okuması ve bir store komutunda yazması gerekir.
- Register dosyası yalnızca bir load komutu veya aritmetik-mantıksal komuta yazılmalıdır ve ALU'nun çeşitli işlemlerden birini gerçekleştirmesi gerekir.

Uygulamaya Genel Bakış



- Yandaki şekilde, bir önceki şekildeki veri yolunu gerekli üç multiplexerın yanı sıra ana işlevsel birimler için kontrol hatları eklenmiş olarak göstermektedir.
- Giriş olarak komut içeren bir **kontrol ünitesi(Control Unit)**, fonksiyonel üniteler ve iki multiplexer için kontrol hatlarının nasıl ayarlanacağını belirlemek için kullanılır.

Uygulamaya Genel Bakış

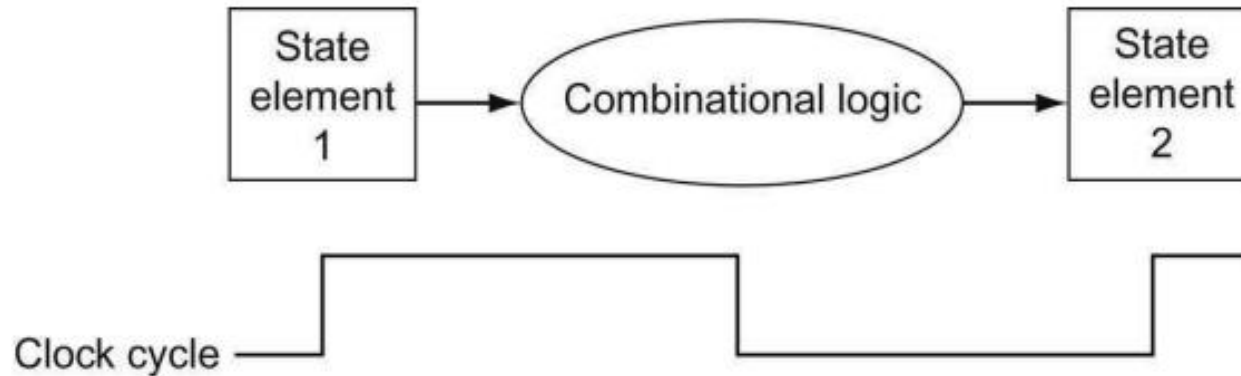


- PC 4'ün veya Branch (dallanma) hedef adresinin PC'ye yazılıp yazılmayacağını belirleyen üst multiplexer, beq komutunun karşılaştırmasını gerçekleştirmek için kullanılan ALU'nun (Zero) **Sıfır çıkışına** göre ayarlanır.

- RISC-V komut setinin düzenliliği ve basitliği, kontrol hatlarının nasıl ayarlanacağını belirlemek için basit bir kod çözme işleminin kullanılabileceği anlamına gelir.

Mantıksal Tasarım Kuralları

- Bir bilgisayarın tasarımına başlamadan önce;
- bilgisayarı uygulayan **donanım mantığının** nasıl çalışacağına ve
- bilgisayarın nasıl bir **clock sistemine** sahip olması gerektiğine karar vermeliyiz.



- kenar tetiklemeli saat palsi

Mantıksal Tasarım Kuralları

- RISC-V uygulamasındaki **veri yolu** bileşenleri iki farklı türde mantık bileşeninden oluşur:
- 1- **veri değerleri** üzerinde çalışan ögeler ve
- 2- **durumları** içeren ögeler.
- Veri değerleri üzerinde çalışan bileşenlerin tümü kombinasyonel devre bileşenidir. Bu durum, bu devrelerin çıktılarının yalnızca **mevcut girdilere bağlı** olduğu anlamına gelir.
- Örnek: VE Kapısı veya ALU gibi

Mantıksal Tasarım Kuralları

- Bir durum (State) elemanının en az iki girişi ve bir çıkışı vardır.
- Gerekli girişler, **elemana yazılacak veri değeri** ve veri değerinin ne zaman yazılacağını belirleyen **saat palsi girişi**dir.
- Örneğin mantıksal olarak en basit durum elemanlarından biri **D tipi bir flip-flop**tur.
- RISC-V uygulamamızda, flip-flop'lara ek olarak iki tür durum ögesi daha kullanır:
- Bellekler ve Registerlar

Tek Vuruşluk İşlemci (Single Cycle Processor)

İşlemci içindeki her komut tek bir peryotta (çevrim) tamamlanan işlemciye denir.

Komutların çalıştırılma döngüsü:

Topla Komutu: Getir → Çöz → Yürüt → Yaz

Yükle Komutu: Getir → Çöz → Yürüt → **Bellek** → Yaz

Topla

Yükle



Topla ve yükle komutlarının gecikmeleri farklıdır.

$$Yürütme\ Zamanı = Komut\ Sayısı \times KBÇ \times Çevrim\ Zamanı(Peryot)$$

↓
1

Saat Vuruş Sıklığı Nasıl Belirlenir?

- İşlemci ısındıkça silikon malzeme üzerinden akan elektron akımı da yavaşlayacaktır.
- Transistörlerin üretim aşamasında üretim kalınlıkları transistörün anahtarlama süresini etkiler. 22nm teknolojisi ile üretilen bir transistör ortalama 100 atom genişliğindedir. Fakat malzemenin kesin aşamasında bu değerler değişkenlik gösterebilir.
- Her komutun yürütme zamanı değişkenlik gösterebilir. Bu nedenle de en yavaş komuta göre süre belirlenmelidir.
- Bu gibi nedenlere bağlı olarak saat vuruşu ve saat vuruşları sonrası ek süreler belirlenir.

İşlemci Tasarımı Aşamaları

1. *Komut Kümesi Mimarisi (ISA) Belirlenmesi*

1. ARM
2. RISC-V
3. MIPS
4. INTEL
5. Transmeta Crusoe (2000 yılları)
https://en.wikipedia.org/wiki/Transmeta_Crusoe
6. CYRIX (1988'de kuruldu)

2. *KKM Gereksinimleri Belirlenmesi*

1. Toplama/Çıkarma
2. Load/Store için Bellek
3. Registerları koymak.. Intel için 8 adet, RISC-V için 32 adet

İşlemci Tasarımı Aşamaları

3. *Veriyolu Oluşturulması*

1. Gereksinimleri karşılayan veri yolu bileşenlerinin belirlenmesi
2. Veri yolu bileşenlerinin birleştirilmesi

4. *Denetim Birimlerinin Oluşturulması*

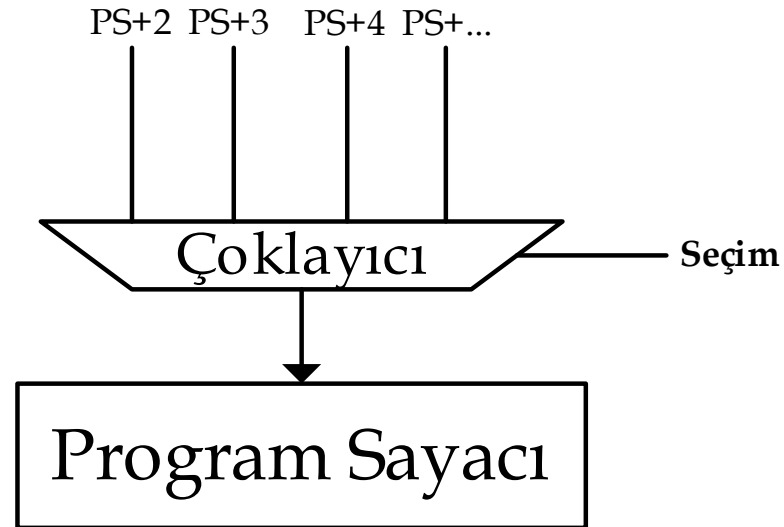
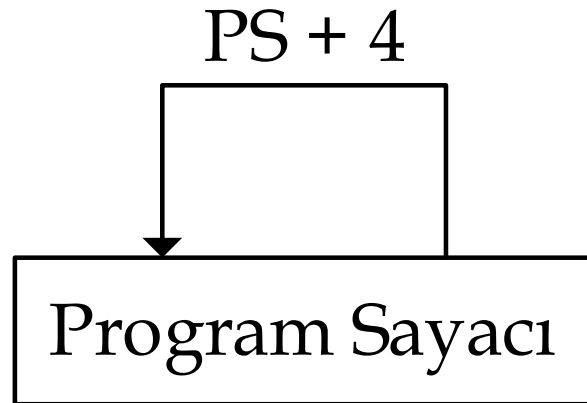
3. Denetim noktalarının belirlenmesi
4. Denetim işaretlerini sağlayacak birimin tasarlanması

Komut Kümesi Mimarisi Belirlenmesi

Komut kümesi mimarisi işlemci tasarım kararlarını etkiler.

Örneğin:

- Değişken boyutta komutlar **getir (-ing. fetch)** ve **çöz (-ing. decode)** aşamalarının tasarımını etkiler.



- Mimarideki register sayısı işlemcideki **register öbeğinin (-ing. register file)** boyutunu belirler.

Komut Kümesi Mimarisi Belirlenmesi

İşlemciler seçilen komut kümesindeki tüm komutları yürütebilmelidir.

Tasarlayacağımız işlemcinin komut kümesi (RV32-I'nın alt kümesi):


Tipi	Komut	İşlenenler	Açıklama
Aritmetik	add	hr, kr1, kr2	kr1 ve kr2'deki değerleri toplayıp hr yazmacına yazar.
	sub	hr, kr1, kr2	kr1 yazmacındaki değerden kr2 yazmacındaki değeri çıkarır ve hr yazmacına yazar.
	addi	hr, kr1, anlık ₁₂	kr1 ve anlık değeri toplayıp hr yazmacına yazar.
	subi	hr, kr1, anlık ₁₂	kr1 yazmacındaki değerden anlık değeri çıkarıp hr yazmacına yazar.
Mantık	and	hr, kr1, kr2	kr1 ve kr2'deki değerlerin mantıksal ve sonucunu hr yazmacına yazar.
	xor	hr, kr1, kr2	kr1 ve kr2'deki değerlerin mantıksal dışlayan veya sonucunu hr yazmacına yazar.
Bellek	lw	hr, anlık ₁₂ (kr1)	Bellek'te (kr1 + anlık) ile işaretlenen sözcüğü hr yazmacına yazar.
	sw	kr2, anlık ₁₂ (kr1)	Bellek'te (kr1 + anlık) ile işaretlenen sözcüğe kr2 yazmacının değerini yazar.
Dallanma	beq	kr1, kr2, anlık ₁₂	kr1 ve kr2 eşit ise (PS + anlık) adresine zıplar, değilse program sayacını dört artırır.
	jal	hr, anlık ₂₀	(PS + 4) değerini hr yazmacına yazar sonra (PS + anlık) adresine zıplar.

İşlemci Tasarımı Aşamaları

1. Komut Kümesi Mimarisi Belirlenmesi → RISC-V
- 2. KKM Gereksinimleri Belirlenmesi**
3. Veriyolu Oluşturulması
4. Denetim Birimlerinin Oluşturulması

KKM Gereksinimlerinin Belirlenmesi

Tasarıma başlanmadan önce cevaplanması gereken sorular:

- Register öbeğinde (File) kaç register var?
 - RV32-I: 32 register.
- Registerlar kaç bit genişliğinde?
 - RV32-I: 32-bit.
- Aynı anda kaç Register okunabiliyor?
 - RV32-I: En fazla **iki Register** işleneni var (Örn. add rt, kr1, kr2)
- Hangi işlemler desteklenmeli?  **Adres hesaplaması**
 - Aritmetik: Topla (add, addi, **ld**, **st**) , çıkar (sub, subi).
 - Mantık: VE (and), DIŞLAYAN VEYA (xor).
 - Karşılaştırma: Eşittir (beq).

İşlemci Tasarımı Aşamaları

1. Komut Kümesi Mimarisi Belirlenmesi → RISC-V
2. KKM Gereksinimleri Belirlenmesi
- 3. Veriyolu Oluşturulması**
4. Denetim Birimlerinin Oluşturulması

Veriyolu Oluşturulması

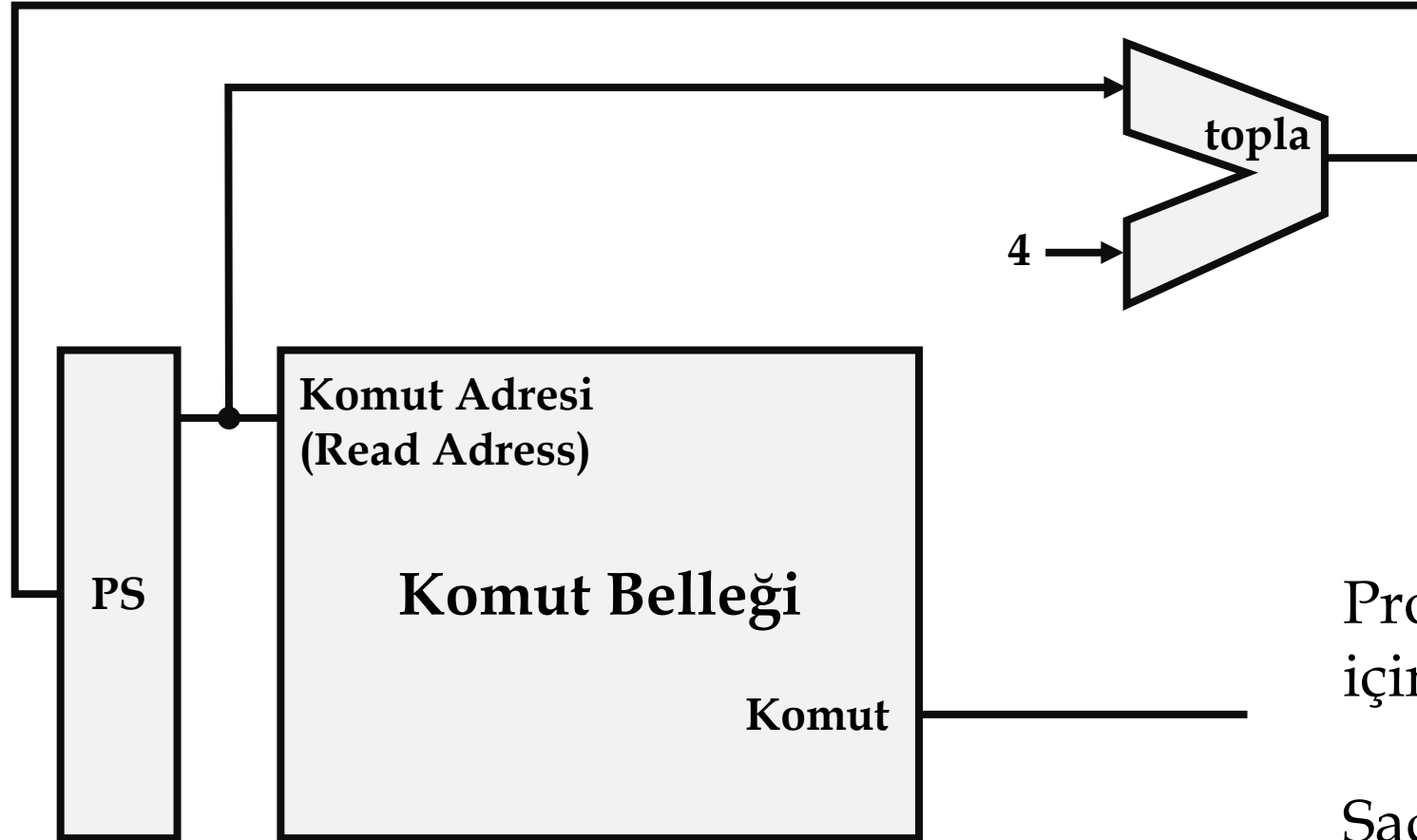
İşlemcide veri saklamak veya veri üzerinde işlem yapmak için kullanılan birimlere **veriyolu birimleri** adı verilir. İşlemci **veriyolunu** bu birimler oluşturur.

RV32-I işlemcisinin veriyolu birimleri:

- Program Sayacı ve Komut Belleği
- Register Öbeği
- Aritmetik mantık birimi
- Anlık genişletme birimi
- Veri Belleği

Program Sayacı ve Komut Belleği

Komut belleği program komutlarını tutar ve verilen komut adresine karşılık gelen komutu çıktı olarak verir.



fetching instructions

Program sayacı (PS) ise o anki komut adresini tutar. Komutlar **32-bit sabit genişlikte** olduğundan ve **bayt adresleme** kullanıldığı için **program sayacı** her saat vuruşunda 4 artar.

Program hafızası sadece **okuma** içindir.

Sadece program yüklenirken komut belleğine **yazma** gerçekleşir.

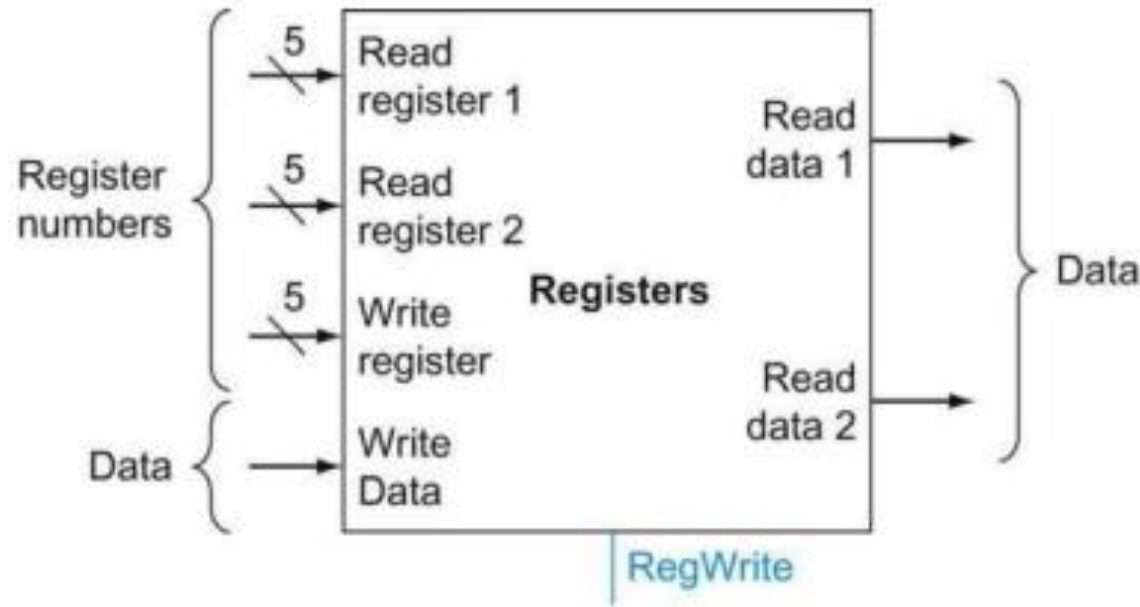
Register Öbeği (Register File)

Register Öbeği, register numaraları verilen kaynak Registerlarındaki verileri çıktı olarak verir veya Register numarası verilmiş hedef Registera verilen veriyi yazar.



Register Öbeği (Register File)

Ancak yazma işlemleri, **yazma kontrol sinyali** tarafından kontrol edilir.

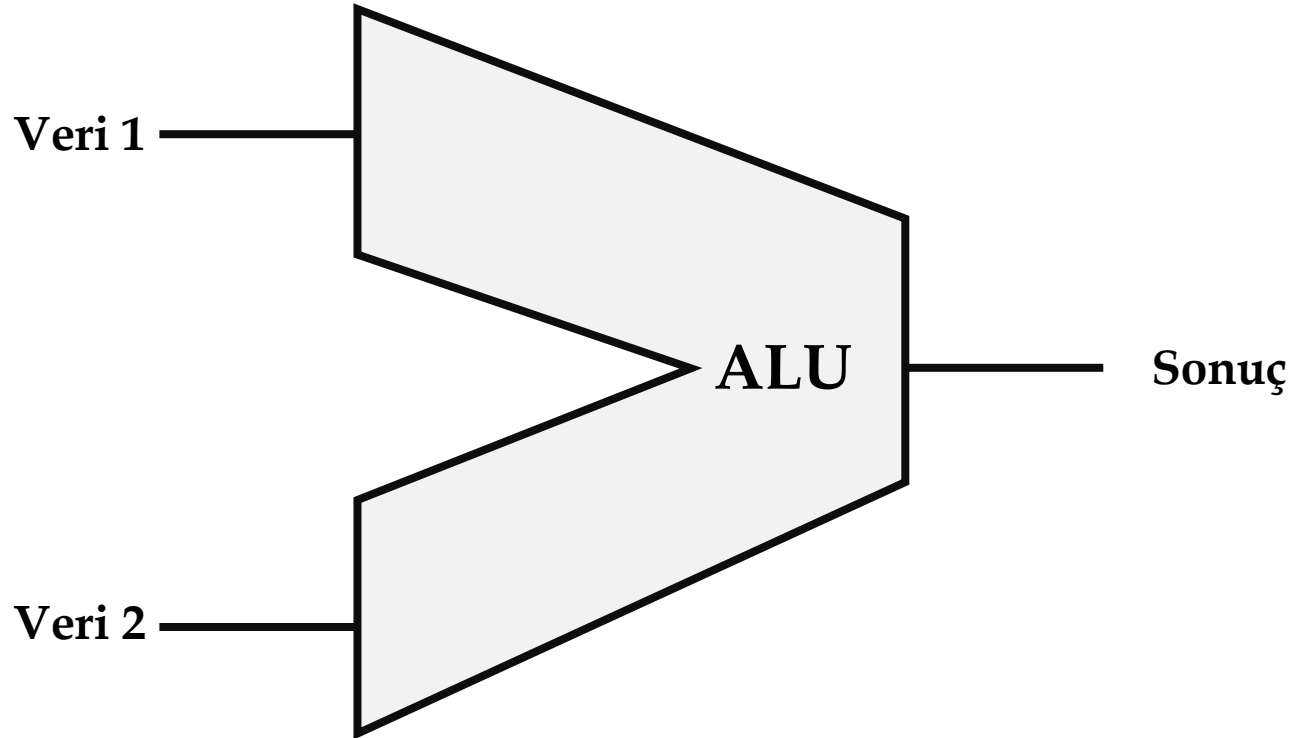


R-formatlı komutların **üç Register** alanı vardır, dolayısıyla her komut için Register öbeğinden iki word boyutunda veri okumamız ve Register öbeğine bir word'lük veri yazmamız gerekecektir.



Aritmetik Mantık Birimi (ALU)

Aritmetik – mantık işlemlerinin yapıldığı birimdir.



Bu birimin yapması gereken işlemler:

1. Toplama
2. Çıkarma
3. VE
4. DIŞLAYAN VEYA

Anlık Oluşturma Birimi

Komut kümesi mimarisine göre farklı komutların anlık değerleri farklı bit uzunluklarında olabilir ve Aritmetik Mantık Birimi 32 bitlik verilerle işlem yapmaktadır. Bu anlık değerlerin işleme girebilmesi için genişletilmesi gerekmektedir.

Anlık Oluşturma Birimi, işleme girecek olan bu anlık değerleri komuttan seçer işaretle genişletir.



Veri Belleği

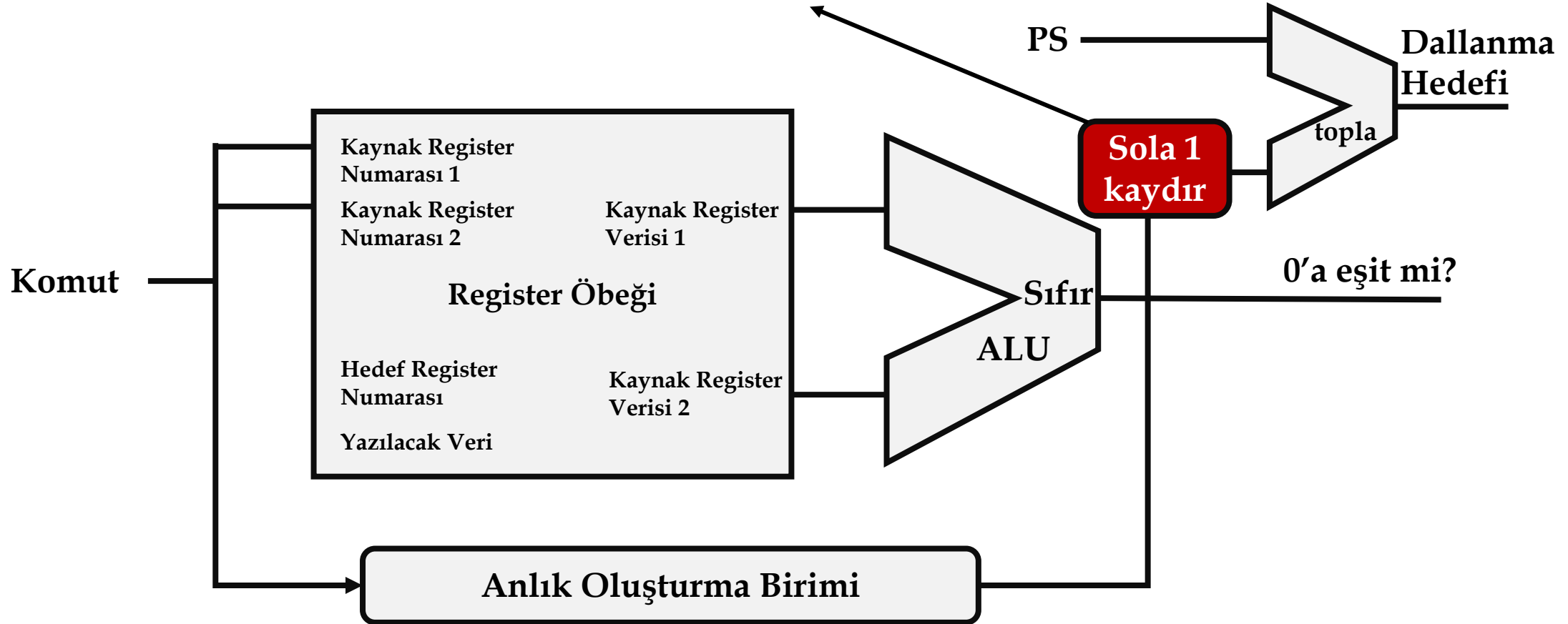
Veri Belleği, çalışan programların bilgilerini tutar ve bu adreslere erişimi sağlayan birimdir. Verilen adrese okuma ya da yazma (R/W) yapabilir.



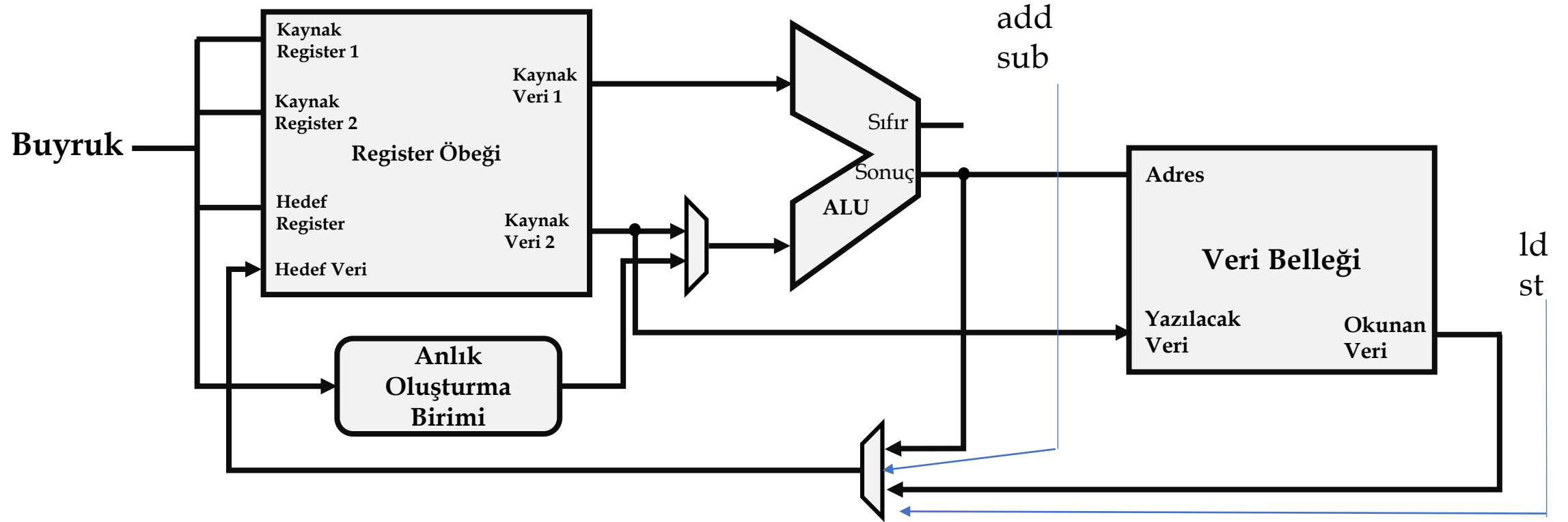
Dallanmaların Veriyolu

RISC-V Tanımlaması: Dallanma komutlarındaki anlık değerlerin kodlaması 2-bayt katlarında göreceli konum belirtir.
“The 12-bit B-immediate encodes signed offsets in multiples of 2 bytes.”

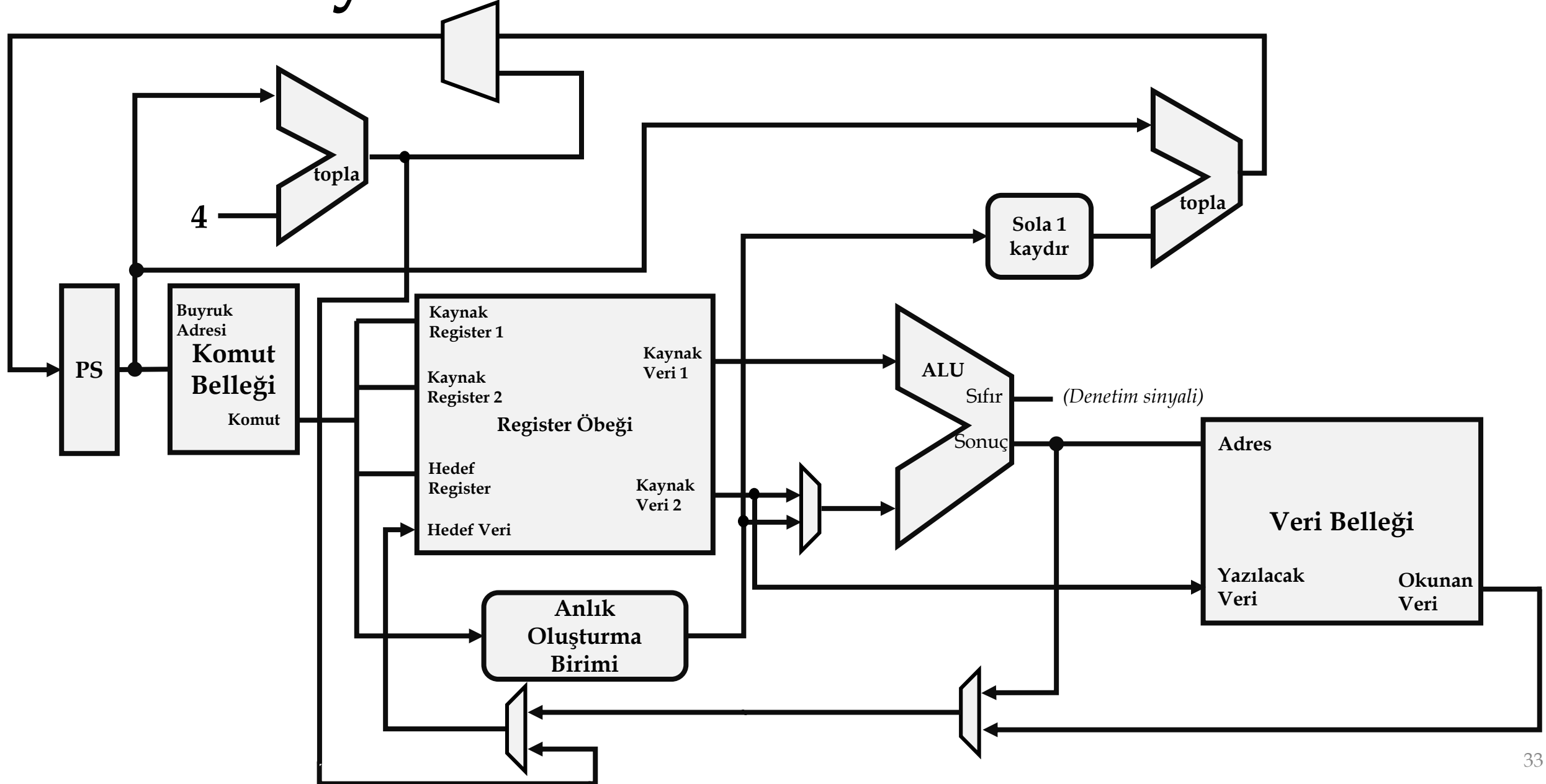
Örn: İki buyruk öteye zıplamak için **anlık değer** = 4 olmalıdır. $4 \times 2 = 8$ bayt, her buyruk 4 bayt.



Bellek, Aritmetik ve Mantık İşlemlerinin Veriyolu



Tüm Veriyolu



Özet

- İşlemci Tasarımına Giriş
- İşlemci Tasarımı Aşamaları
 - Buyruk Kümesi Mimarisi Belirlenmesi
 - BKM Gereksinimleri Belirlenmesi
 - Veriyolu Oluşturulması