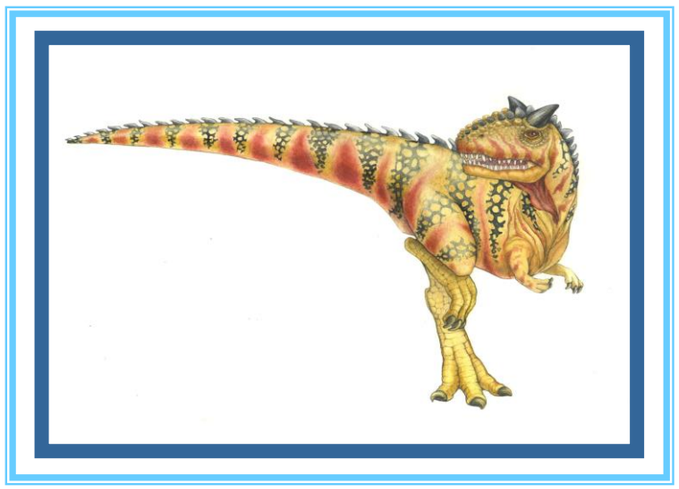
Bölüm 8: Ana Bellek (Main Memory)



Bölüm 8: Bellek Yönetimi

- Arka plan
- Takas (Swapping)
- Ardışık Bellek Tahsisi (Contiguous Memory Allocation)
- Sayfalama (Paging)
- Sayfa Tablosunun Yapısı
- Segmentasyon
- Örnek: Intel Pentium

Hedefler

- Bellek donanımını organize etme yollarını detaylı bir şekilde açıklamak
- Sayfalama ve segmentasyon da dahil olmak üzere çeşitli bellek yönetim teknolojilerini tartışmak
- Sadece segmentasyon ve sayfalamalı segmentasyon tekniklerinden her ikisini de destekleyen Intel Pentium'u detaylı bir şekilde tanımlamak

Arkaplan

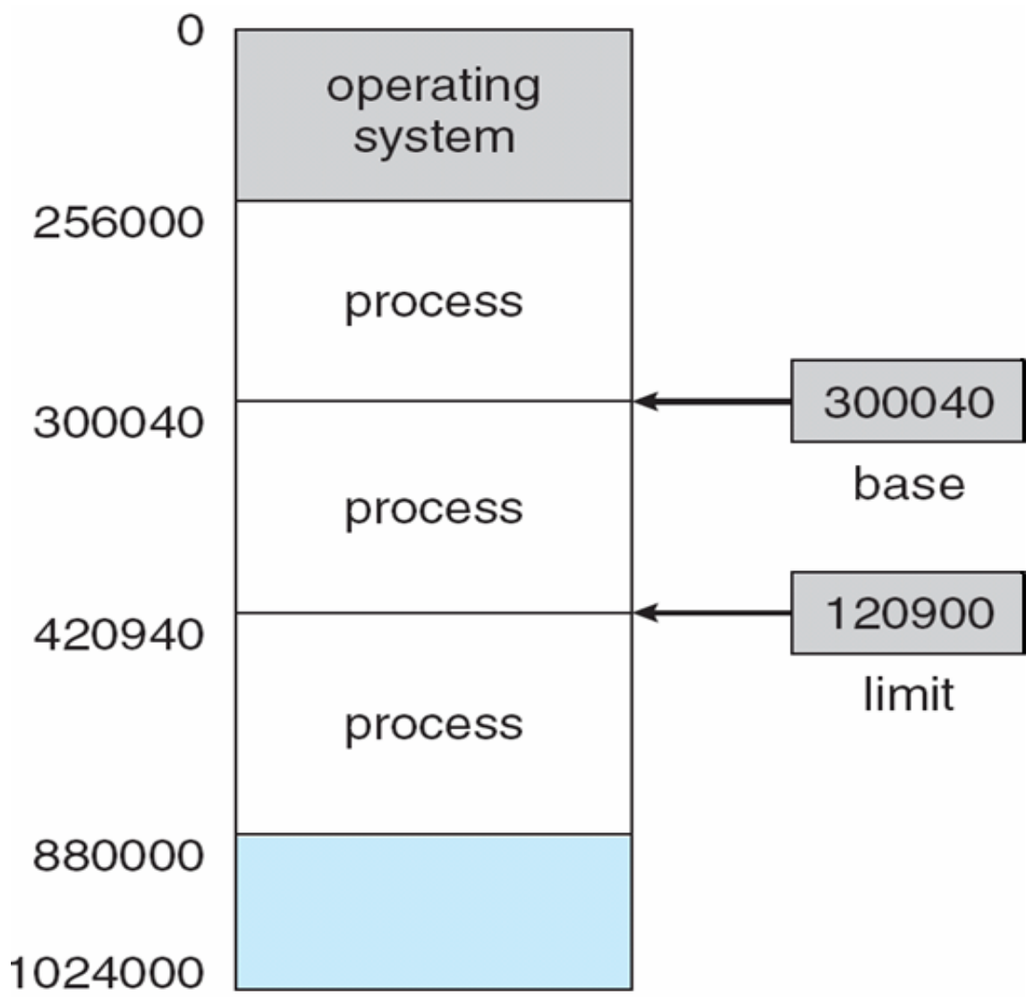
- Bir bilgisayar sisteminin ana amacı, programları çalıştırmaktır.
- Bu programlar, eriştikleri verilerle birlikte, yürütme sırasında en azından kısmen ana bellekte olmalıdırlar.
- Hem CPU'nun kullanımını hem de kullanıcılara verdiği yanıtın hızını iyileştirmek için, genel amaçlı bir bilgisayar, birkaç işlemin bellekte tutulması gerekir.
- Çeşitli yaklaşımları yansıtan bir çok bellek yönetim şeması mevcuttur ve her bir algoritmanın etkinliği duruma bağlıdır.
- Bir sistem için bir bellek yönetimi şemasının seçilmesi birçok faktöre bağlıdır, özellikle sistemin donanım tasarımı üzerinde.
- Çoğu algoritma, donanım desteği gerektirir.

Arkaplan

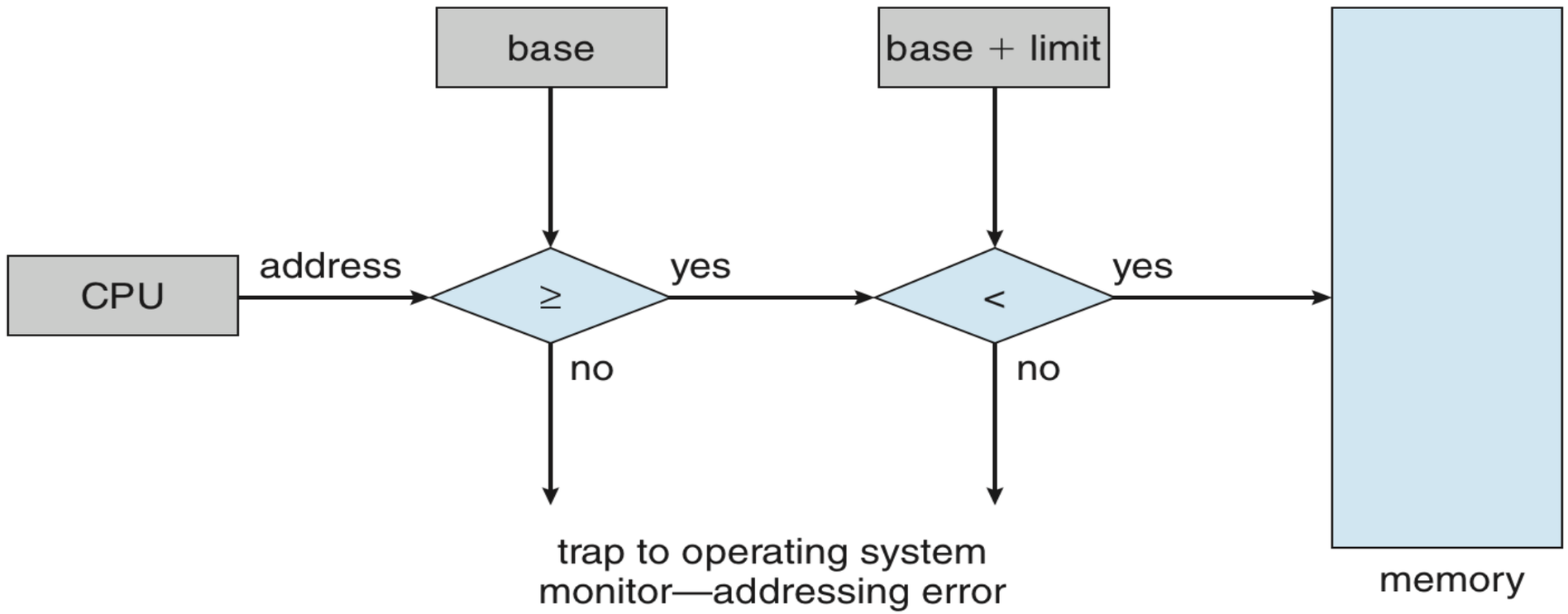
- Program, diskten belleğe getirilip çalışması için bir process'e yerleştirilmelidir.
- Ana bellek ve kaydediciler CPU'nun doğrudan erişebildiği kayıt ortamlarıdır
- Bellek ünitesi yalnızca adresler + okuma istekleri veya adres + veri ve yazma istekleri ile ilgilenir
- Kaydedici erişimi bir CPU çevriminde (veya daha az) yerine getirilir
- Ana bellek bir den fazla çevrimde erişilebilir
- **Ön bellek (Cache),** ana bellek ve CPU kaydedicileri arasında yer alır
- Belleğin korunması belleğin doğru çalışmasını sağlamak için gereklidir.

Taban ve Limit Kaydedici

- **Base** ve **limit** kaydedici çifti mantıksal adres alanı olarak tanımlanırlar.
- CPU'nun, kullanıcı modunda üretilen her bir bellek erişimini denetlemesi gerekir (bu kullanıcı için tanımlı olan base ve limit sınırları içinde olduğunu garantilemek için)



Base ve Limit Kaydedicileri ile Donanım Adresi Koruma



Adres Bağlama

- Genellikle, bir program, bir disk üzerinde ikili (binary) bir yürütülebilir dosya olarak bulunur.
- Uygulanacak program, belleğe alınmalı ve bir prosese yerleştirilmelidir.
- Kullanılan bellek yönetimine bağlı olarak, işlem yürütülürken disk ve bellek arasında taşınabilir.
- Diskte duran ve yürütülmek için belleğe alınmayı bekleyen prosesler, girdi kuyruğunu (input queue) oluşturur.
- Normal prosedüre göre, girdi kuyruğundaki işlemlerden birini seçilir ve bu işlem belleğe yüklenir.
- İşlem yürütülürken, bellekteki komutlara ve verilere erişir.
- Sonunda proses sona erer ve bellek alanı kullanılabilir olarak bildirilir. Yani adres alanı geri verilir.

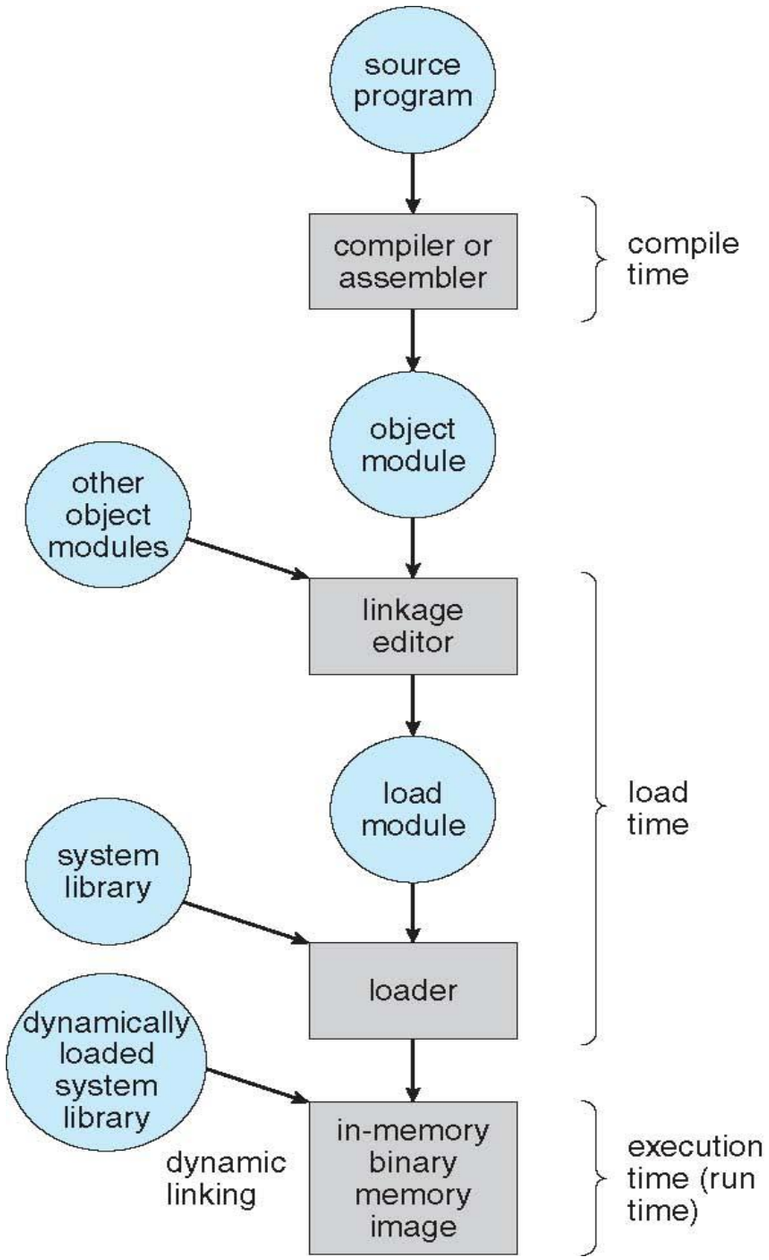
Adres Bağlama

- Çoğu sistem bir kullanıcı prosesinin fiziksel belleğin herhangi bir bölümünde bulunmasına izin verir.
- Böylece, bilgisayarın adres alanı 00000'de başlayabilmesine rağmen, kullanıcı işleminin ilk adresi 00000 olmasına gerek yoktur. Daha sonra, bir kullanıcı programının bir işlemi gerçekte fiziksel bellekte nasıl bulunduğunu göreceksiniz.
- Ayrıca, adresler bir programın yaşamının farklı aşamalarında farklı yollarla gösterilir.
  - Kaynak kod adresleri genellikle semboliktir. («Sayaç» değişkeni gibi)
  - Bir derleyici, genellikle bu sembolik adresleri yeniden taşınabilir adreslere **bağlar**.
    - örneğin. “bu modülün başından itibaren 14 bytes”
  - Bağlayıcı (linkage) veya Yükleyici (loader) yeniden konumlandırılabilir bu adresleri değişmez adreslere bağlar
    - örneğin. 74014
  - Her bir bağlama bir adres uzayını diğerine dönüştürür

Komutları ve Veriyi Belleğe Bağlama

- Komutların ve verinin bellek adreslerine bağlanması üç durumda olabilir:
  - **Derleme zamanı**: Eğer bellek konumu önceden bilinirse, **mutlak kod** üretilebilir. Eğer başlangıç konumu değişirse yeniden derlenmelidir.
  - **Yükleme zamanı:** Bellek konumu derleme zamanında bilinmiyorsa **yeniden konumlandırılabilir kod** üretilmelidir.
  - **Çalışma zamanı**: Eğer proses çalışma esnasında bir bellek kesiminden diğerine hareket ederse, bağlama çalışma anına kadar geciktirilir.
    - Adres haritalama için donanım desteği gerekir (örneğin, taban (base) ve tavan (limit) kaydedicileri)

Bir Kullanıcı Programının Çok Adımlı Çalışması



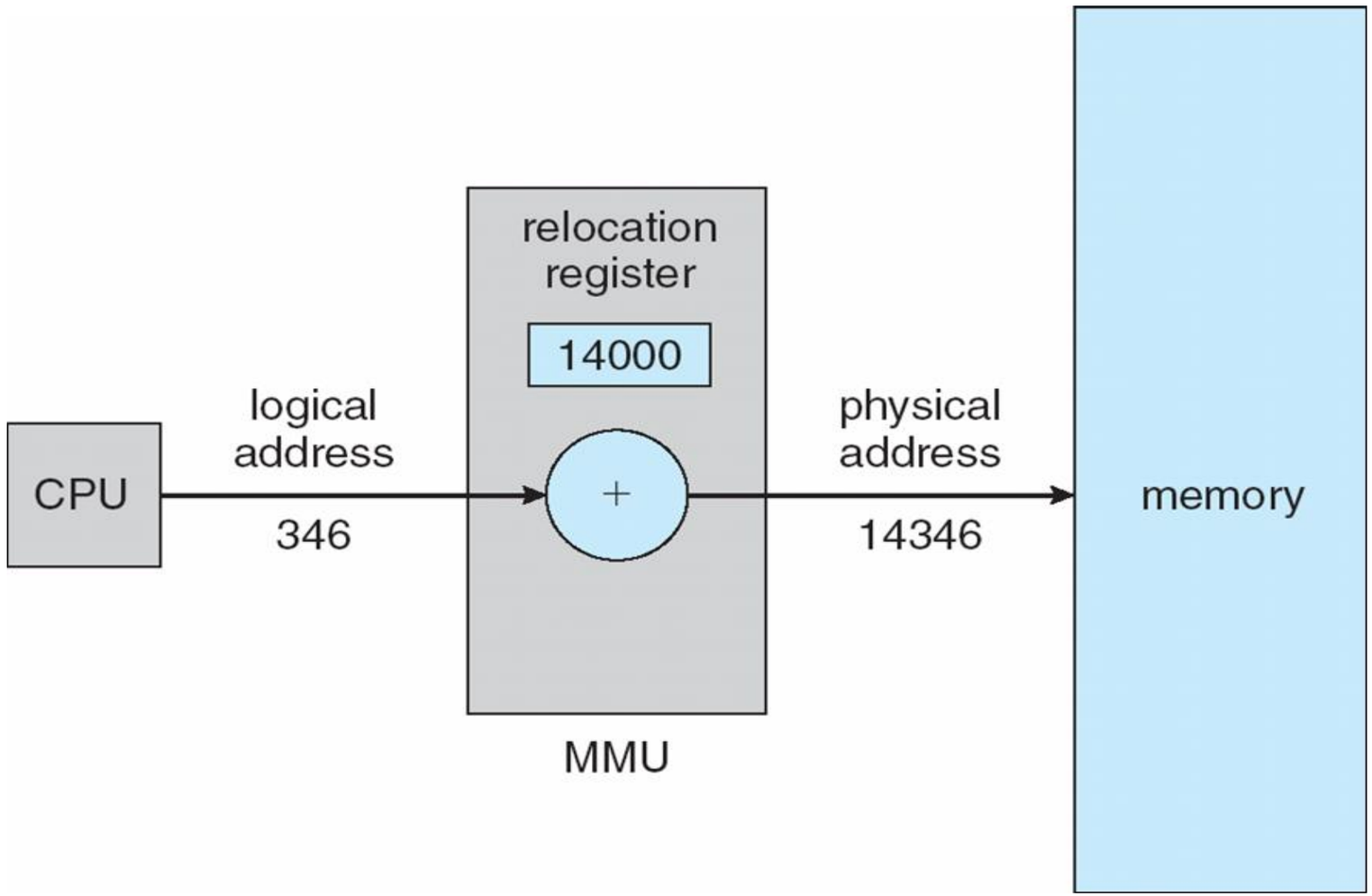
Mantıksal vs. Fiziksel Adres Uzayı

- Fiziksel adres uzayının mantıksal adres uzayı kavramından ayrılması, sağlam bir bellek yönetiminin merkezinde yer alır.
  - **Mantıksal adres (Logical address)** – CPU tarafından oluşturulur; ayrıca sanal adres (**virtual address**) olarak ta adlandırılır.
  - **Fiziksel adres (Physical address)** – adres, bellek birimi tarafından görülür.
- Derleme zamanı ve yükleme zamanı adresiyle bağlama yöntemleri aynı mantıksal ve fiziksel adresleri üretir. Bununla birlikte, yürütme zamanı adres-bağlama şeması farklı mantıksal ve fiziksel adreslere neden olur.
- **Mantıksal adres uzayı** bir program tarafından oluşturulan tüm mantıksal adreslerin kümesidir.
- **Fiziksel adres uzayı** bir program tarafından oluşturulan bu mantıksal adreslere karşılık gelen tüm fiziksel adres kümesidir.

Bellek Yönetim Birimi
(Memory-Management Unit - MMU)

- Çalışma anında sanal adresi fiziksel adrese dönüştüren donanım
- Bu bölümün ilerleyen kısımlarında göreceğiniz üzere pek çok yöntem mevcuttur.
- Başlangıç için, yeniden konumlandırma kaydedicisindeki bir değerin bir kullanıcı prosesi tarafından belleğe gönderildiği anda üretilen her bir adrese eklendiği basit bir düzeni ele alalım.
  - Taban(base) kaydedicisi **yeniden konumlandırma kaydedicisi** olarak adlandırılır
  - Intel 80x86 üzerinde MS-DOS, 4 adet yeniden konumlandırma kaydedicisi kullanmıştır.
- Kullanıcı programları *mantıksal* adreslerle çalışırlar; asla *gerçek* fiziksel adresleri göremezler.
  - Çalışma anında bağlama bellekteki bir konuma referans yapıldığında meydana gelir
  - Mantıksal adres, fiziksel adrese bağlıdır.

Konumlandırma Kaydedicisi Kullanılarak Dinamik Konumlandırma



Dinamik Yükleme

- Rutin çağrılana kadar yüklenmez.
- Bellek alanının daha iyi kullanımını sağlar. Kullanılmamış rutin asla yüklenmez.
- Tüm rutinler yeniden konumlandırılabilir yük biçiminde hazır durumda diskte tutulur.
- Nadir meydana gelen olayları yönetmek için büyük miktarda koda ihtiyaç duyulduğunda kullanışlıdır.

Dinamik Bağlama

- **Statik bağlama**– sistem kütüphaneleri ve program kodunun yükleyici (loader) tarafından ikilik (binary) program görüntüsü altında birleştirilmesidir.
- **Dinamik bağlama**– bağlama işleminin çalışma zamanına kadar ertelenmesidir.
- Küçük bir kod parçası olan *stub (kalıntı)*, uygun olan hafıza-yerleşim yerini tespit etmek için kullanılır.
- Stub rutinin adresi ile kendisinin yerini değiştirerek rutini yürütür.
- İşletim sistemi rutinin bellek adresinde bulunup bulunmadığını kontrol eder.
  - Adres alanında değilse, adres alanına ekler.
- Dinamik linking özellikle kütüphaneler için kullanışlıdır.
- Sistem aynı zamanda **shared libraries (paylaşımlı kütüphaneler)** olarak da bilir.
- Sistem kütüphanelerini güncellemeleri için uygulanabilirliğini düşünün.
  - Sürümleme (versiyonlar oluşturma) gerekebilir.

Swapping (Takas)

- Bir process geçici olarak bellekten bir yedekleme deposuna alınabilir ve sonra yürütmeye devam etmek için belleğe geri gönderilebilir
  - Process'lerin toplam fiziksel bellek alanı fiziksel bellek miktarını aşabilir.

- **Backing store (yedekleme deposu)** – Tüm kullanıcılar için tüm bellek görüntülerini (image) kopyalarını barındıracak kadar büyük ve hızlı disk ; bu hafıza görüntülerini (image) doğrudan erişim sağlanmalıdır.

- **Roll out, roll in (Dışa taşıma, içe taşıma)** – Öncelik tabanlı planlama algoritmaları için kullanılır. Düşük öncelikli bir process dışa taşınır (swap out) çok daha yüksek öncelikli işlem takas edilip yürütülür.

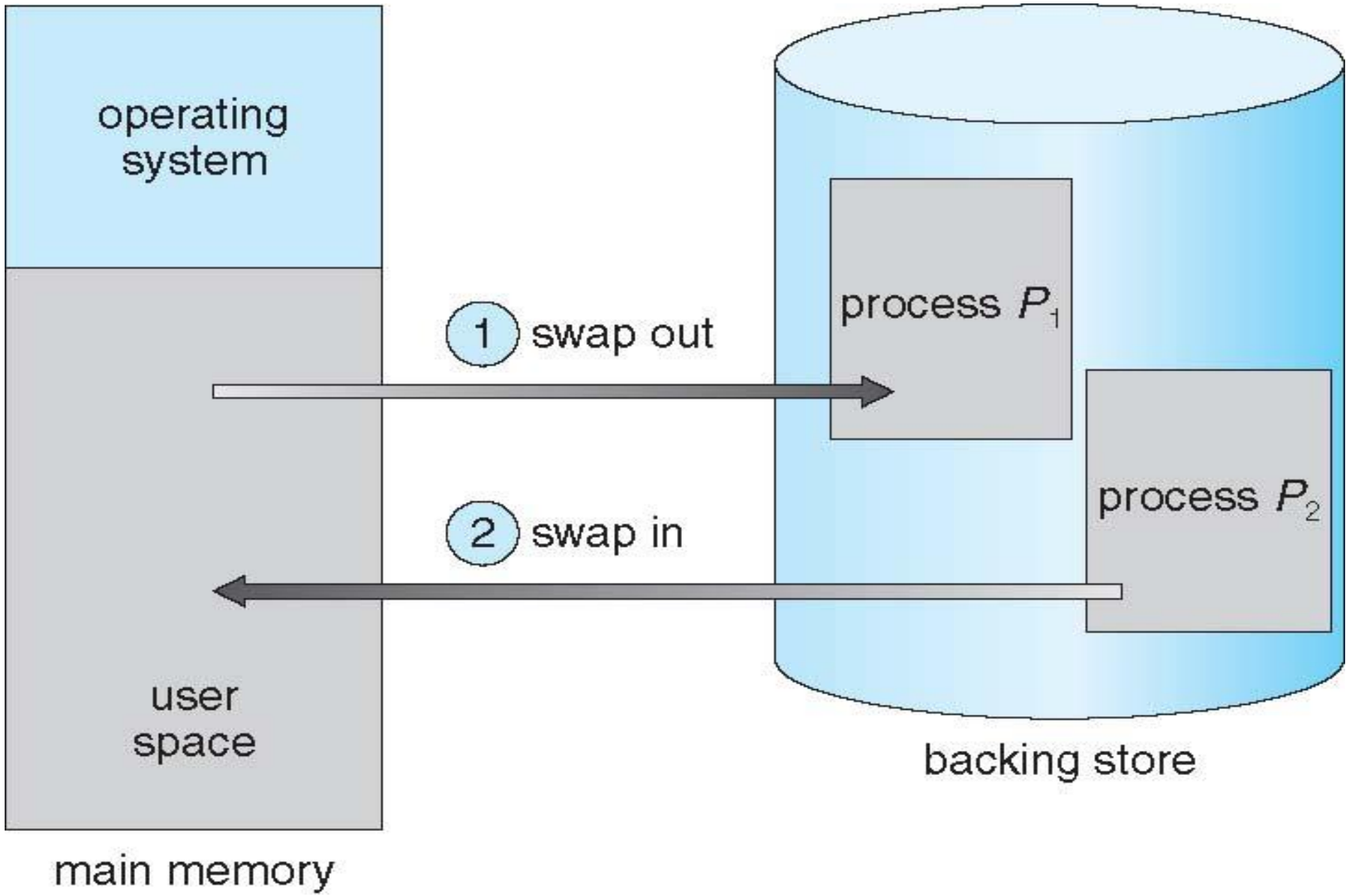
- Takas süresinin büyük bir kısmı transfer süresidir; toplam transfer süresi takas edilen bellek miktarı ile doğru orantılıdır.

- Sistem, disk üzerinde bellek görüntüleri olan çalışmaya hazır (ready-to-run) prosesleri hazır kuyruğunda tutar.

Swapping (Takas)

- Değiştirilen sürecin aynı fiziksel adreslere geri dönmesi gerekiyor mu?
- Adres bağlama yöntemine bağlı
- Swapping işleminin değiştirilmiş versiyonları pek çok sistemde bulunur.(ör., UNIX, Linux, ve Windows)
  - Takas işlemi normalde devre dışıdır.
  - Ayrılmış bellek alanı eşik değerinden fazla ile başlatılır.
  - Talep edilen bellek miktarı eşik değerinin altına inerse tekrar devre dışı bırakılır.

Swapping Şematik Görünümü



Context Switch Time including Swapping

- CPU'ya konacak sonraki işlemler bellekte değilse, bir prosesi dışa takas etmeye (dışa taşıma) ve hedef prosesinde içe takas etmeye ihtiyaç duyarız
- İçerik geçiş zamanı daha sonra çok yüksek olabilir
- 50MB / sn aktarım hızıyla sabit diske 100MB prosesin takası;
  - Artı 8 ms'lik disk gecikmesi ile
  - 2008 ms dışa takas (swap out) zamanı
  - + Aynı boyutlu prosesi de içe takas yapın
  - Toplam bağlam değiştirme bileşeni süresi 4016ms (> 4 saniye)

- Gerçekten ne kadar bellek kullanıldığını bilerek, bellek takas boyutu ve tabiki zamanı azaltabilir.
- Sistem çağrıları, (`request memory` ve `release memory`) OS'yi bellek kullanımı hakkında bilgilendirilmesi için kullanır

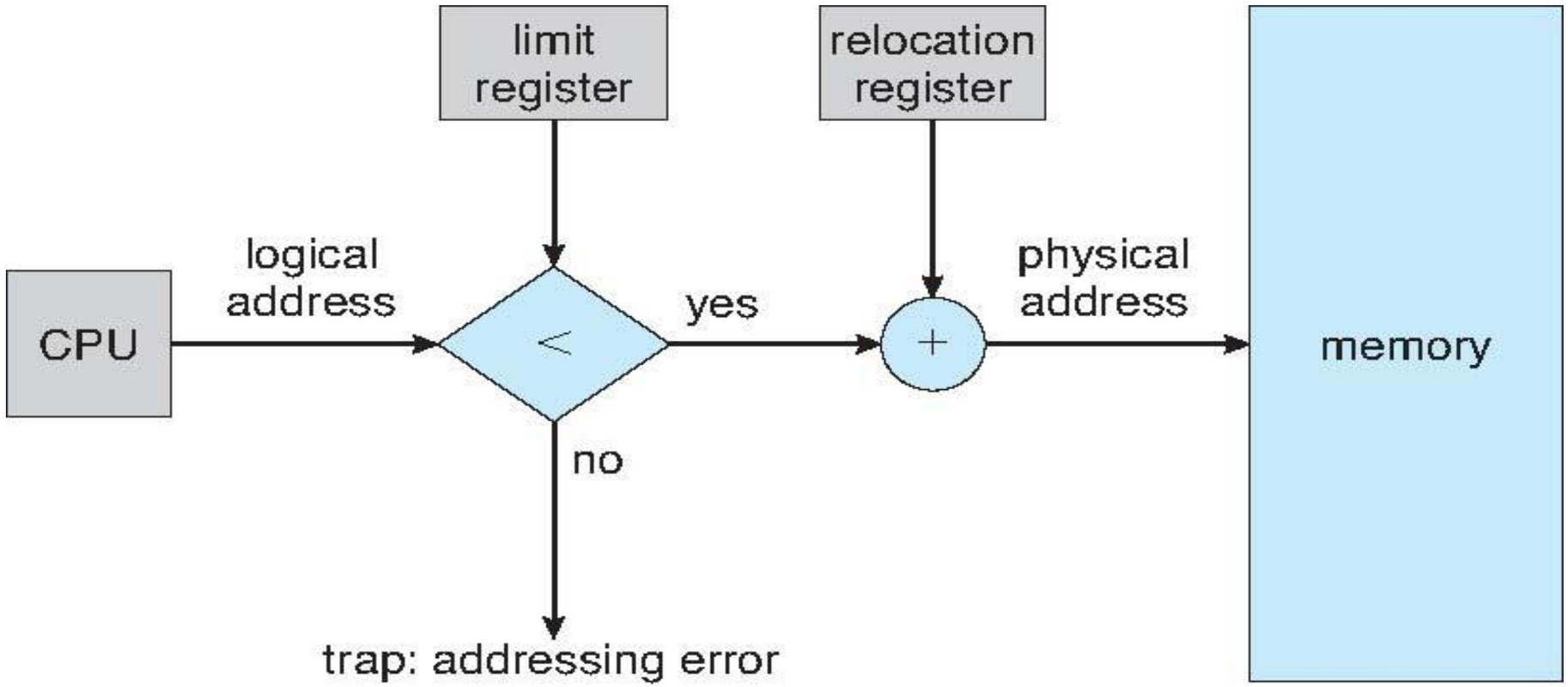
Bitişik Tahsis (Contiguous Allocation)

- Ana bellek, hem OS hem de kullanıcı süreçlerini desteklemelidir
- Sınırlı kaynak verimli bir şekilde tahsis edilmelidir
- Bitişik tahsis, eski bir yöntemdir
- Ana bellek genellikle iki bölümden oluşur:
  - Yerleşik işletim sistemi genellikle kesme vektörü ile düşük bellekte (low memory) tutulur.
  - Kullanıcı işlemleri ise yüksek hafızada (high memory) tutulur
  - Her process belleğin bitişik tek bir bölümünde yer alır.

Bitişik Tahsis (Contiguous Allocation)

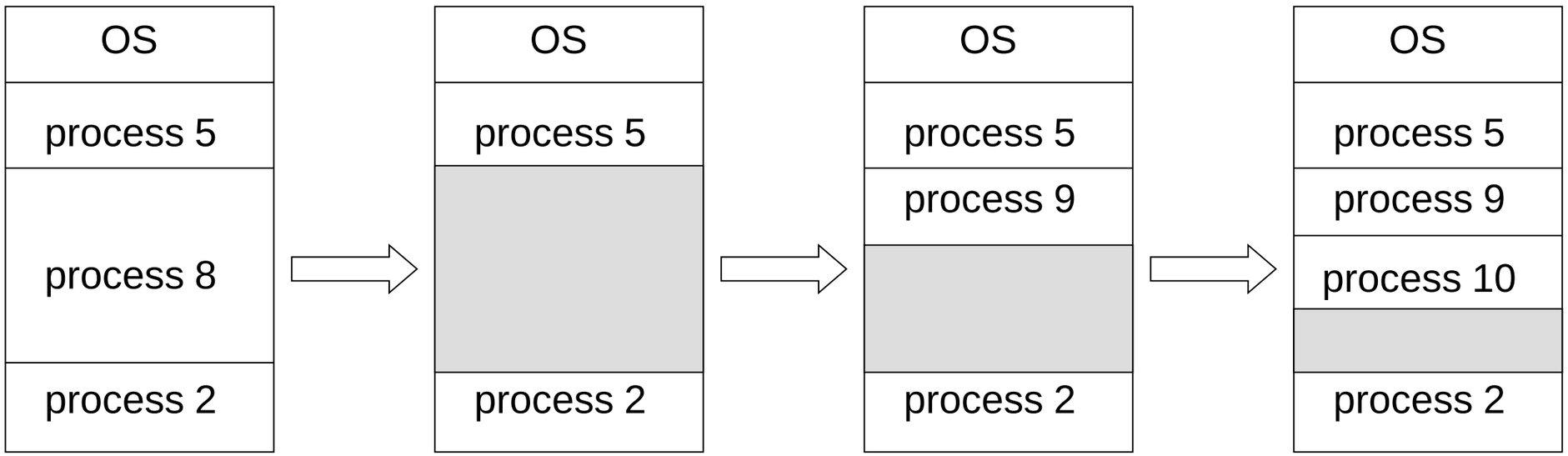
- Konumlandırma register'ı (relocation register) kullanıcı process'lerini bir diğerinden korumak için kullanılır.
  - Base register en küçük fiziksel adres değerini içerir.
  - Limit register mantıksal adres aralığını içerir - her mantıksal adres limit register'dan daha kısa olmalıdır.
  - MMU mantıksal adresi *dinamik olarak* haritalar.

Konumlandırma Register'ları ve Limit Register'lar İçin Donanım Desteği



Bitişik Tahsis (Devam)

- Çoklu-bölüm tahsisi
  - Verimlilik için değişken bölüm boyutları (belirli bir prosesin ihtiyaçlarına göre boyutlandırılır)
  - Hole (Delik) – kullanılabilir bellek bloğu;çeşitli büyüklükteki holes(boşluklar) bellek boyunca dağılmıştır.
  - Bir process geldiğinde, onun sığabileceği büyüklükte bir hole(delik) bellek tahsis edilir.
  - Process serbest bölüme geçerken komşu serbest bölümle birleştirilir.
  - İşletim sistemi şu bilgileri tutar:
    a) ayrılan bölümleri b) serbest bölümleri (hole)



Dinamik Depolama-Tahsis Problemi

Serbest alanlara gelen N boyutlu bir istek nasıl yerine getirilir ?

- **First-fit (İlk durum)**: İlk bulduğu yeterli alana yerleştirir.

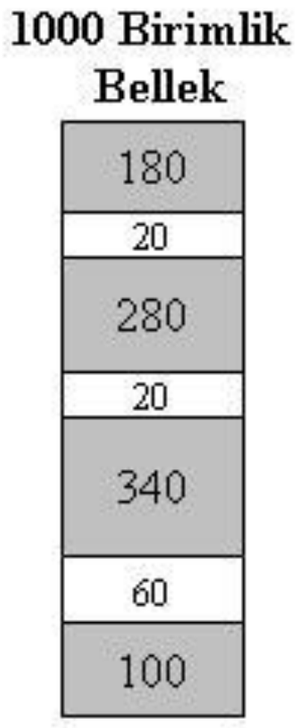
- **Best-fit (En uygun durum)**: Tüm liste aranır boyutlarına bakarak en az boşluk bırakacak şekilde yerleştirilir.
  - En az artık alan üretir.

- **Worst-fit (En kötü durum)**: En büyük alana yerleştirir. Aynı zamanda tüm liste aranır.
  - En fazla artık alan üretir.

Hız ve depolama alanı açısından first-fit ve best-fit, worst-fit'ten daha iyidir.

Fragmentation (Parçalanma)

- Veriler pek çok sistemde belleğe ardışıl bir biçimde yerleştirilmez. Çünkü ardışıl yerleştirme işlemi bir süre sonra parçalanma (*fragmentation*) denilen soruna yol açmaktadır.
- **External Fragmentation (Dış Parçalanma)**– Toplam bellek alanı çalıştırılacak programa yettiği halde boşluklar farklı bölgelerde olduğundan yerleştirilemez.



- **Internal Fragmentation (İç Parçalanma)**– Ayrılan bellek alanı istenen bellek alanından biraz daha büyük olabilir; bu boyut farkı bellekte bir bölüm olarak mevcuttur ancak kullanılmamıştır.

- «İlk uyan» incelendiğinde *N* blokluk alan tahsis edilmiş, 0.5 *N* blokluk alan fragmentation nedeniyle kaybedilmiştir.
  - 1/3'ü kullanılamaz olabilir -> **yüzde 50 kuralı**

Fragmentation (Devam)

- **Compaction (sıkıştırma)** ile dış parçalanmayı azaltın.
  - Boş belleği tek bir büyük blokta bir araya getirmek için bellek içeriğini karıştır
  - Sıkıştırma işlemi mümkündür ancak, sadece takas (relocation) işlemi dinamik ise yapılır ve yürütme zamanında gerçekleştirilir.
  - I/O (Giriş / Çıkış Sorunu)
    - I/O işlem isteği olduğunda görevi belleğe sabitle

- Şimdi yedekleme deposunun aynı parçalanma sorunlarına sahip olduğunu düşünün

- Sıkıştırma işlemi yükü fazla, bütün proseslerin yeri değişiyor ve zaman gerektirir. Bu yüzden çözüm için; sayfalama, segmentasyon

Sayfalama

- Bir process'in fiziksel bellek alanı bitişik olmayabilir, sonraki bellek alanı kullanılabilir olduğu sürece fiziksel bellek alanı tahsis edilir.

- Fiziksel belleğin sabit boyutlu bloklar halinde bölünmüş haline **frames (çerçeveler)** denir.
  - Boyut 2'nin kuvvetleri şeklinde, 512 bytes ve 16 Mbytes arasında

- Mantıksal adres eş boyutlara bölünür ve bu bölümlere **pages (sayfalar)** deriz.

Sayfalama

- *N* sayfa boyutundaki bir programı çalıştırmak için, *N* tane serbest frame'e ve programın yüklenmesine ihtiyaç vardır
- Mantıksal adresi fiziksel adrese çevirmek için **page table** (sayfa tablosu) gerekli
- Yedekleme deposu aynı şekilde sayfalara bölünür.
- Hala iç parçalanma (Internal fragmentation) mevcuttur.

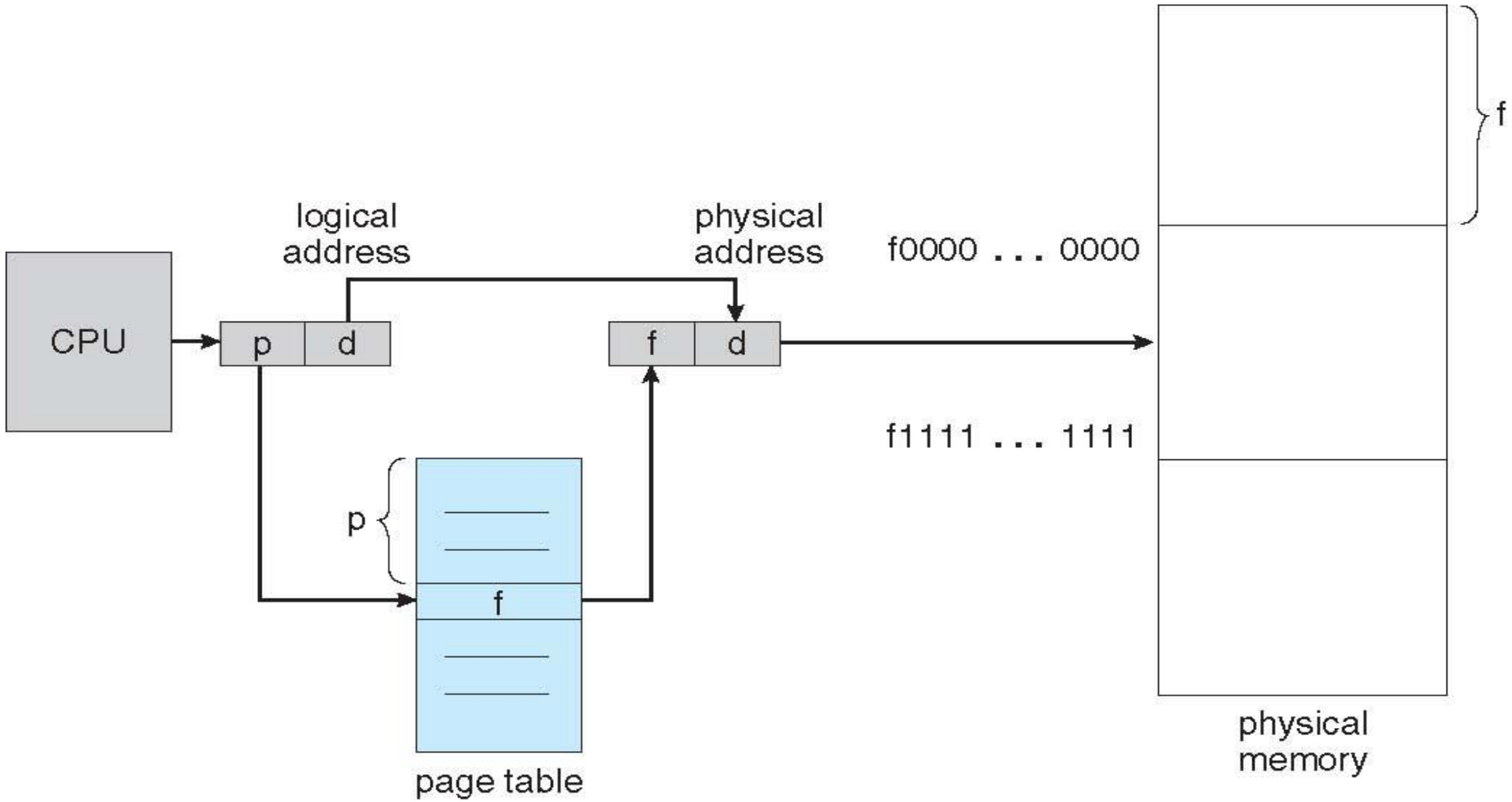
Adres Çeviri Şeması

- İşlemci tarafından üretilen adres aşağıdaki gibi bölünmüştür:
  - **Page number (Sayfa numarası -*p*)**– Fiziksel bellekteki her sayfanın base addressini içeren bir sayfa tablosunda index olarak kullanılır.
  - **Page offset (Sayfa ofset -*d*)** – Bellek birimine gönderilen fiziksel bellek adresini tanımlamak için taban adresi ile birleştirilir.

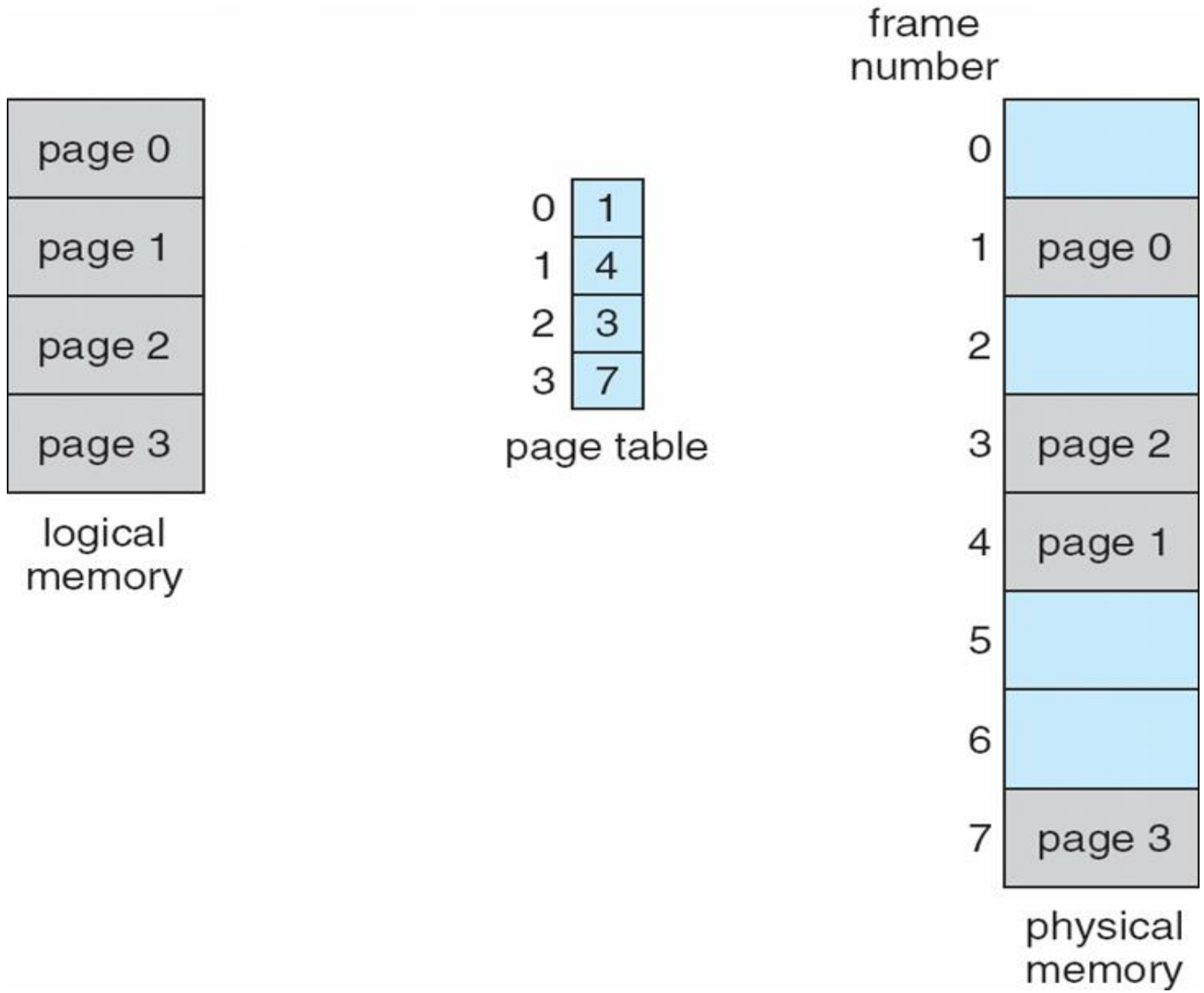
page number	page offset
p	*d*
m - n	*n*

- Verilen mantıksal adres alanı 2^m ve sayfa boyutu 2^n için

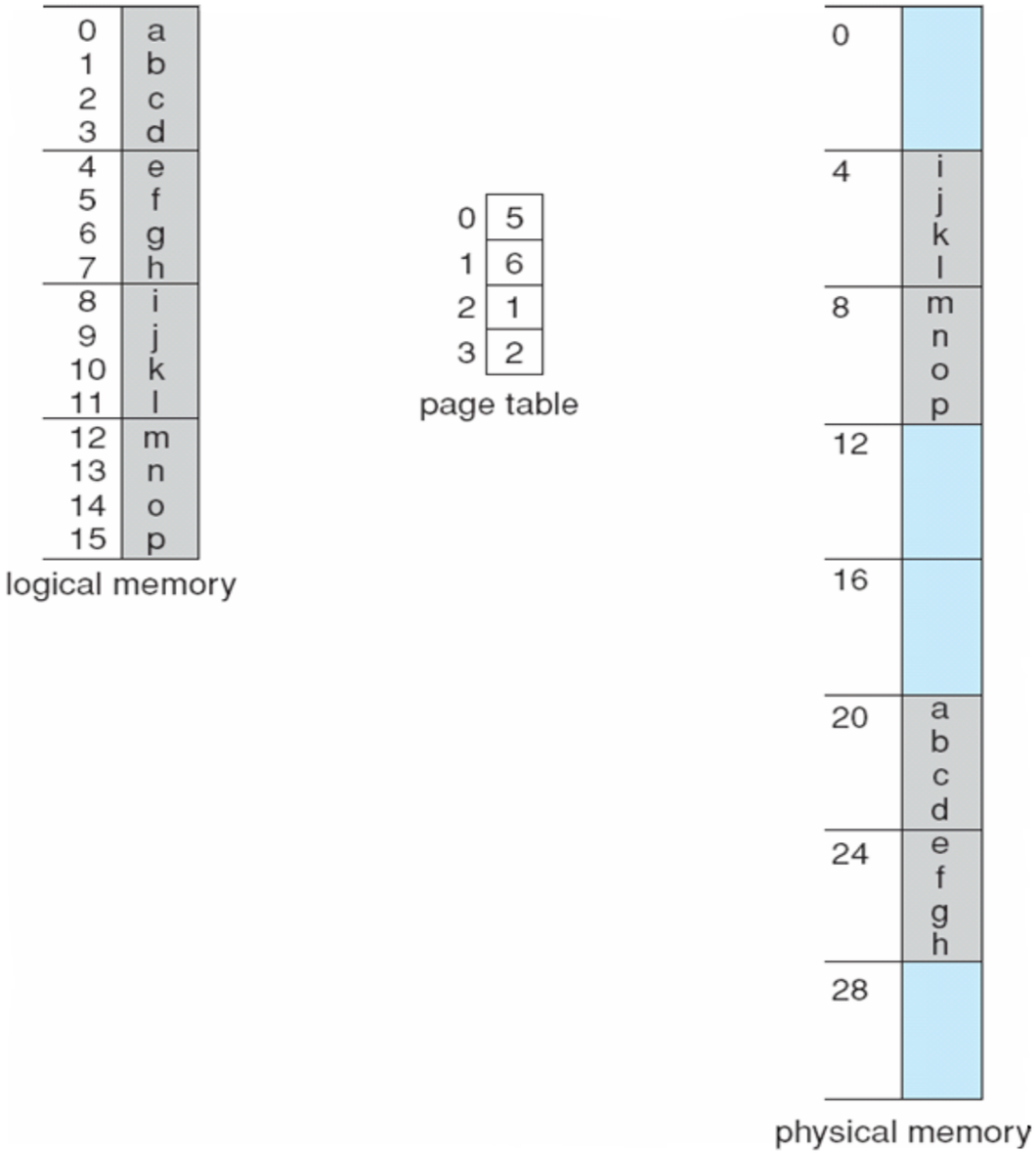
Donanım Sayfalama



Mantıksal ve Fiziksel Bellek Sayfalama Modeli



Sayfalama Örneği

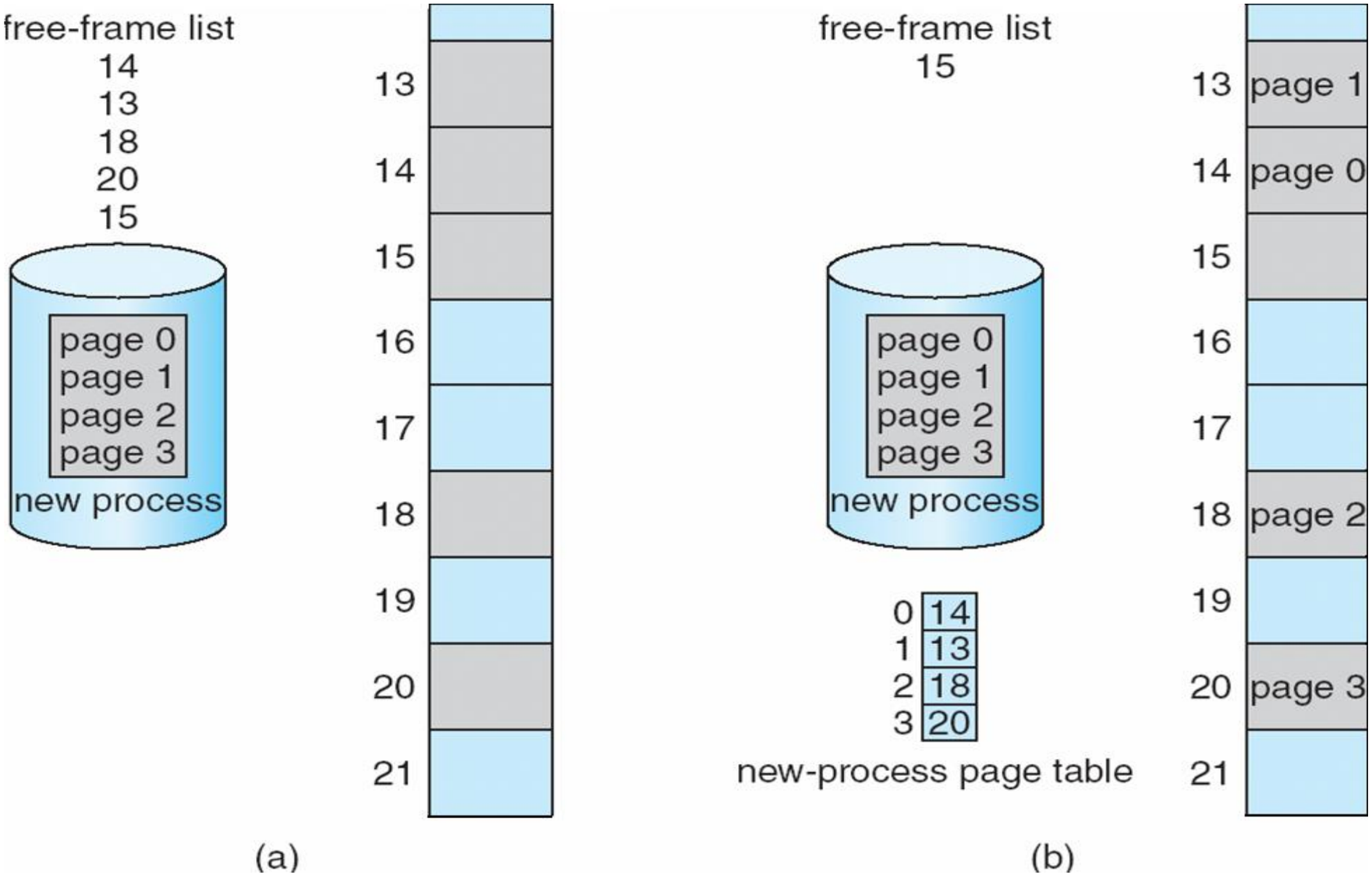


n=2 ve *m*=4 32-byte bellek ve 4-byte'lık sayfalar

Sayfalama (Devam)

- İç parçalanma hesaplanıyor
  - Sayfa boyutu = 2,048 bytes
  - Process boyutu = 72,766 bytes
  - 35 sayfa + 1,086 byte
  - İç parçalanma 2,048 - 1,086 = 962 bytes
  - Çok küçük çerçeve boyutları arzulanır mı?
  - Ancak her sayfa tablosu girişi, izlemek için bellekte yer tutar
  - Solaris iki sayfa boyutu destekler. – 8 KB ve 4 MB
- İşlem görünümü ve fiziksel hafıza artık çok farklı
- Uygulama process'i sadece kendi belleğine erişebilir.

Serbest Frame'ler



Before allocation

After allocation