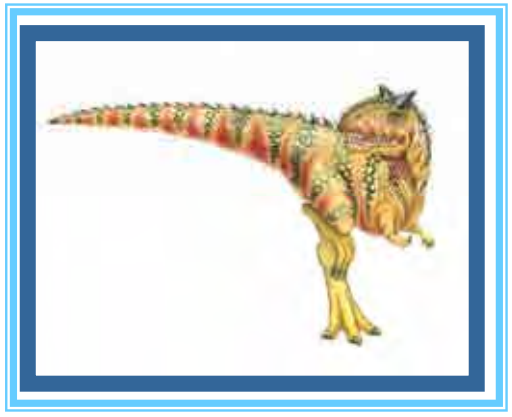
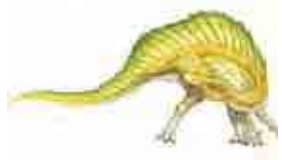
فصل اول: مقدمه



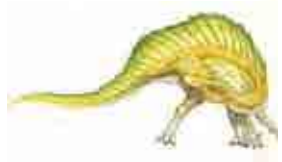
یک سیتم عامل چیست؟

- سیستم عامل به عنوان واسطی بین کاربر کامپیوتر و سخت افزار کامپیوتر عمل می کند.
- اهداف سیستم عامل:
  - فراهم آوردن محیطی که کاربر بتواند در آن به آسانی برنامه های خود را اجرا کند و مشکلات خود را بر طرف سازد.
  - فراهم کردن محیط های کامپیوتری برای آسایش کاربران
  - افزایش کارآیی سخت افزار کامپیوتر

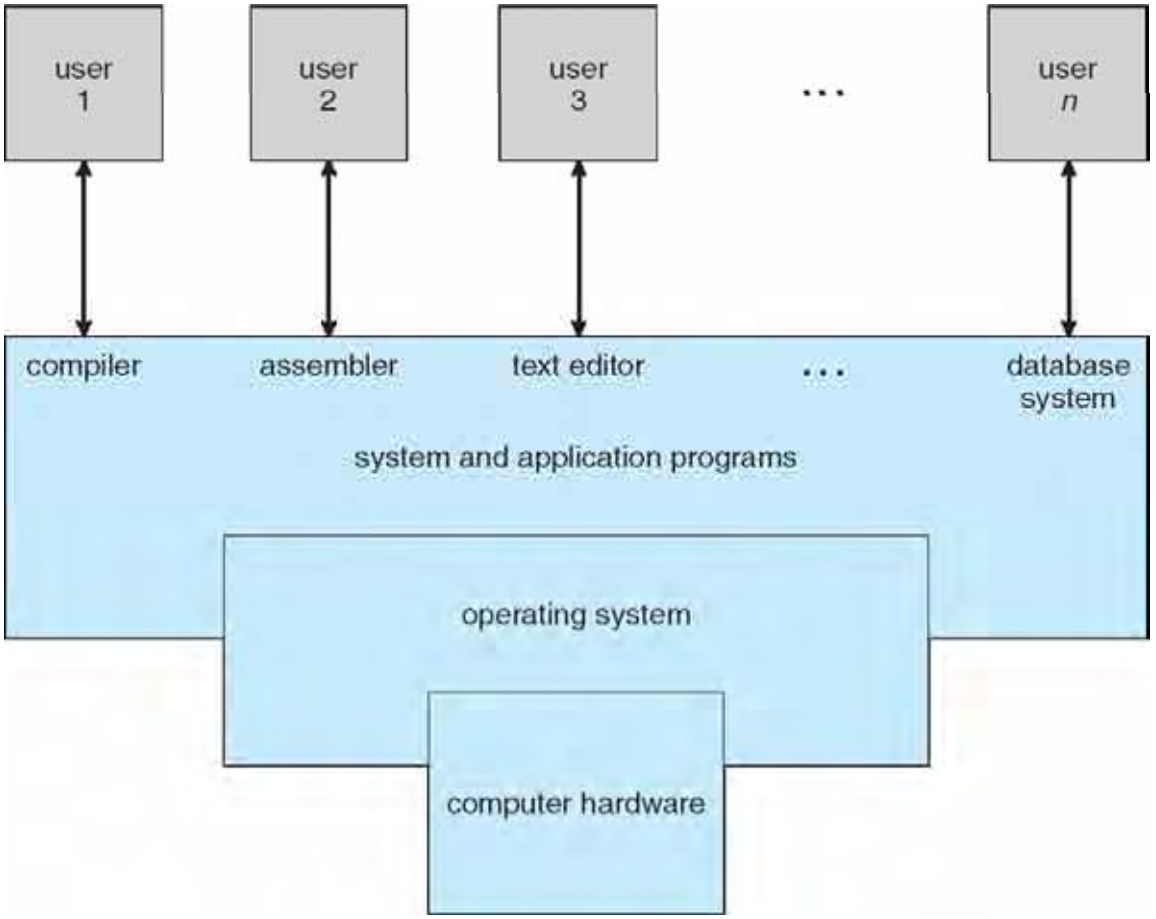


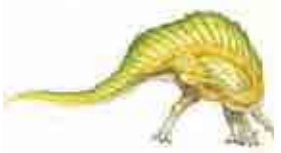
ساختار سیستم کامپیوتری

- سیستم کامپیوتری می تواند به چهار قسمت تقسیم کرد
  - سخت افزار – فراهم آوردن منابع محاسباتی اساسی
    - CPU, memory, I/O devices
  - سیستم عامل
    - کنترل هماهنگی و به کارگیری سخت افزار در میان کاربران و برنامه های کاربردی
  - برنامه های کاربردی – تعیین روش به کارگیری منابع سیستم برای برطرف کردن مشکلات کاربران
    - Word processors, compilers, web browsers, database systems, video games
  - کاربران
    - مردم ، ماشین ها، دیگر کامپیوترها



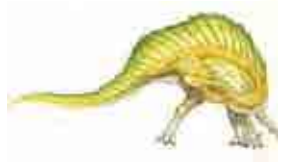
چهار قسمت یک سیستم کامپیوتری





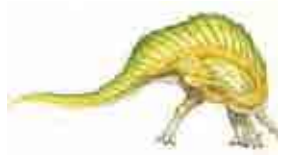
سیستم عامل از دیدگاه های مختلف

- **سیستم عامل ازدیدگاه کاربر – دیدگاه کاربر نسبت به کامپیوتر، بر حسب واسطی که مورد استفاده قرار می گیرد، تفاوت دارد. اما به طورکلی می توان موارد زیر را برشمرد:**
  - مدیریت تمام منابع
  - تصمیم گیری برای درخواست های مختلف و حتی متضاد به منظور استفاده بهینه و عادلانه منابع در بین کاربران
- **سیستم عامل از دیدگاه سیستم – سیستم عامل برنامه ای است که به شدت با سخت افزار عجین است. سیستم عامل را می توان تخصیص دهنده منابع در نظر گرفت. سیستم عامل یک برنامه کنترلی است.**
  - برنامه کنترلی، برنامه های کاربران را کنترل می کند تا از کامپیوتر به درستی استفاده شود و استفاده از آن بهبود داده شود.



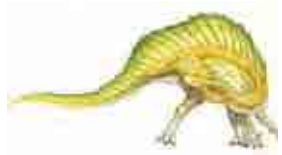
تعریف سیستم عامل

- **تعریف قابل قبول جهانی برای سیستم عامل وجود ندارد. اما می توان تعریف زیر را برای آن در نظر گرفت.**
  - هر چیزی که یک فروشنده در قبال درخواست سیستم عامل به شما تحویل میدهد می تواند به عنوان سیستم عامل محسوب شود.
  - ویژگی هایی که ارائه می شوند در سیستم عاملهای مختلف متفاوت هستند
- **یک تعریف متداولتر که معمولا از آن پیروی می کنیم، این است که، سیستم عامل برنامه ای است که همیشه در کامپیوتر در حال اجرا است و معمولا هسته (کرنل) نامیده می شود.**
  - همراه با هسته دو نوع برنامه وجود دارد: برنامه های سیستمی و برنامه های کاربردی



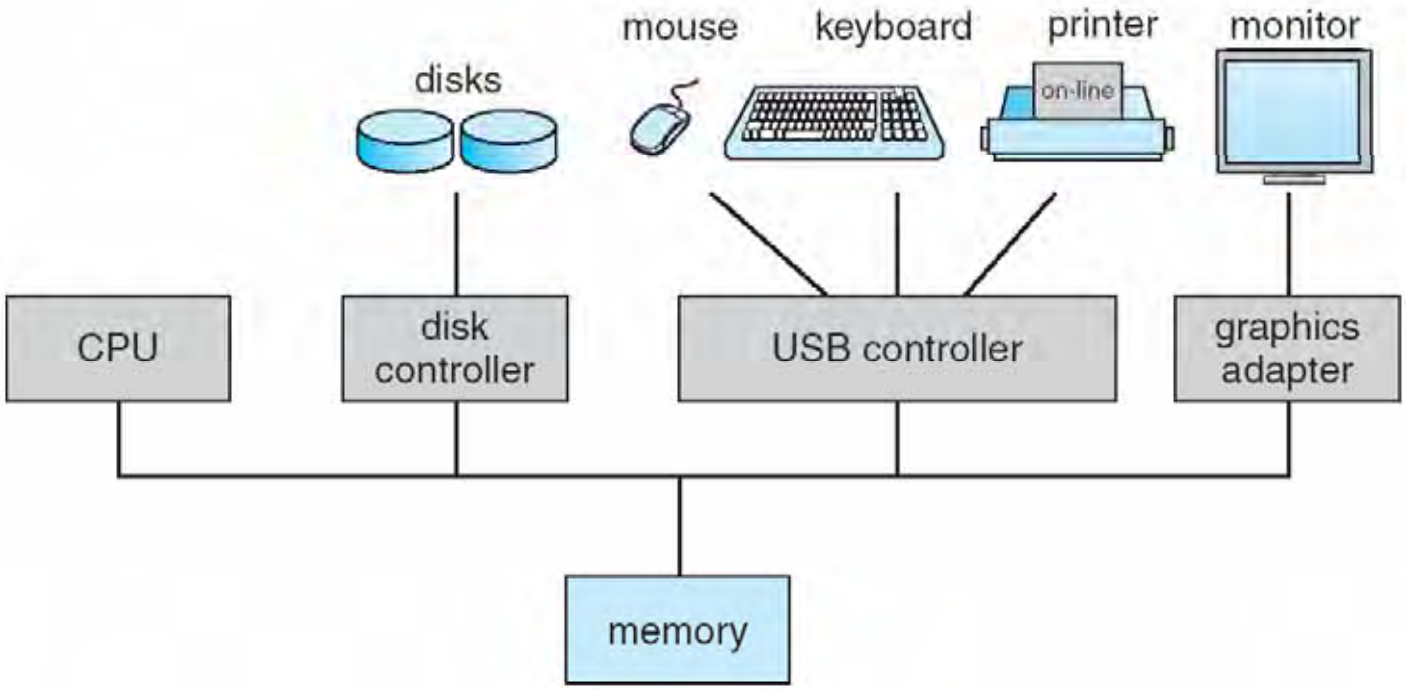
شروع به کار کامپیوتر

- **برنامه راه انداز** – در زمان روشن کردن کامپیوتر و یا ری استارت شدن آن کامپیوتر را راه اندازی می کند.
  - به طور معمول در ROM و یا EEPROM که به میان افزار نیز آن را می شناسند ذخیره می شود.
  - تمام جنبه ها و قسمت های سیستم توسط آن مقدار دهی اولیه می شود.
  - باید بداند که سیستم عامل را چگونه بارکند و اجرای آن سیستم را آغاز نماید.



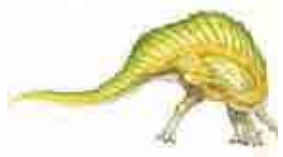
سازمان سیستم کامپیوتری

- عملیات سیستم کامپیوتری
  - پردازنده و کنترلگر دستگاه ها از طریق گذرگاه خاصی(باس) به هم متصل شده که دستیابی به حافظه مشترک را امکان پذیر می سازد
  - پردازنده (CPU) و کنترلگرهای دستگاه می توانند به طور همزمان اجرا شوند و برای بدست آوردن حافظه با هم رقابت کنند



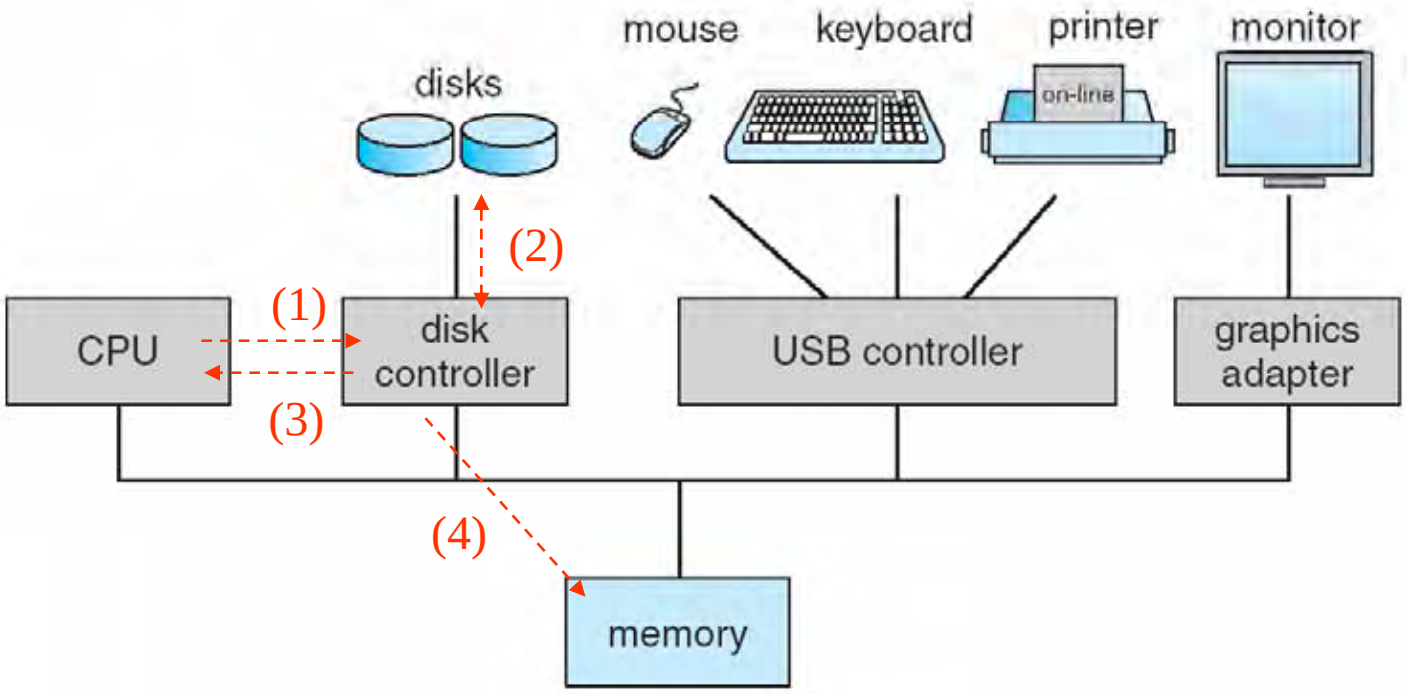
عملیات سیستم کامپیوتری

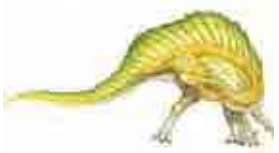
- وسایل I/O و CPU می توانند به طور همروند اجرا شوند
  - هر کنترلگر دستگاه به دستگاه خاصی تعلق دارد. ممکن است چند دستگاه به یک کنترلگر متصل باشند.
  - کنترلگر هر وسیله یک بافر محلی و مجموعه ای از ثبات ها را برای اهداف خاص دارد
  - معمولا سیستم های عامل برای هر کنترلگر دستگاه، دارای یک گرداننده ی دستگاه (device driver) است.
- عملیات I/O
  - برای شروع یک عملیات I/O ،گرداننده دستگاه، ثبات های مناسبی از کنترلگر دستگاه را بار می کند.
  - کنترلگر دستگاه به نوبه خود، محتویات این ثباتها را بررسی می کند تا مشخص کند چه کاری باید انجام دهد (مثل خواندن یک کارکتر از صفحه کلید)
  - داده ها را از وسیله به بافر محلی کنترلگر منتقل می شود
  - پس از انتقال داده ها، کنترلگر دستگاه از طریق یک وقفه به CPU خبر می دهد که کارش تمام شده است.
  - پردازنده داده ها را از بافر محلی به حافظه اصلی انتقال می دهد و یا برعکس



Interrupt-Driven I/O Operation

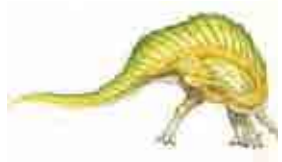
- Ex: Disk I/O operation





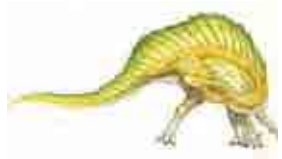
Common Functions of Interrupts

- وقفه باعث انتقال کنترل به روال سرویس دهنده وقفه می شود.
  - بردار وقفه شامل آدرس شروع روتین های سرویس دهنده وقفه می باشد.
  - معماری وقفه باید آدرس دستوری را ذخیره کند که هنگام اجرای آن، وقفه صادر شده است.
  - وقتی یک وقفه در حال پردازش است ورود وقفه های دیگر غیر فعال می شود و این کار برای جلوگیری کردن از دست رفتن وقفه است.
- یک تله یک وقفه نرم افزاری است که به دلیل یک خطا و یا درخواست کاربر ایجاد شده است.
- به دلیل استفاده از وقفه، سیستم عامل های جدید را هدایت شده به وسیله وقفه گویند.

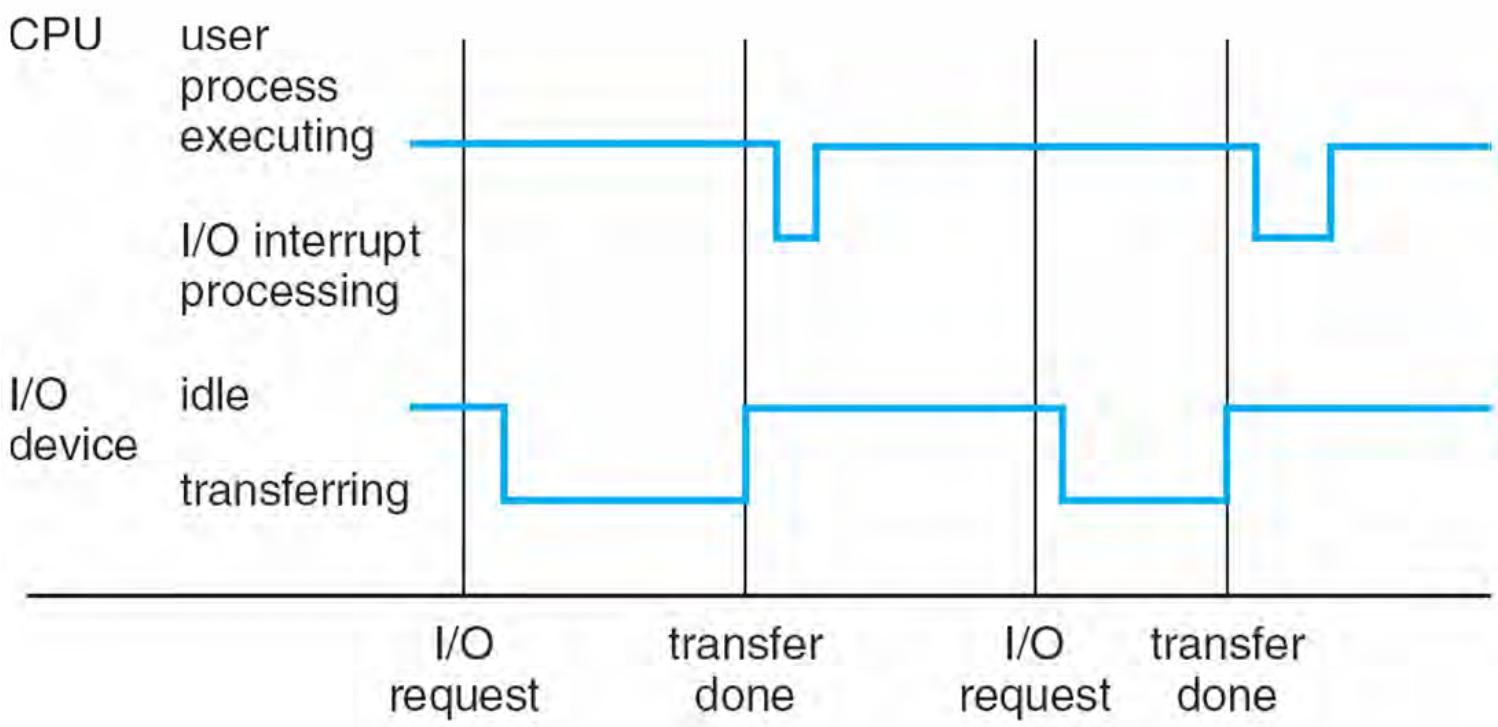


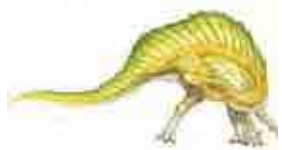
رسیدگی به وقفه

- سیستم عامل به وسیله ذخیره مقدار ثبات ها و شمارنده برنامه وضعیت پردازنده را برای اجراهای بعدی حفظ می کند.
- Determines which type of interrupt has occurred:
  - **polling**
  - **vectored** interrupt system
- Separate segments of code determine what action should be taken for each type of interrupt



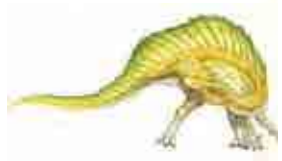
Interrupt Timeline





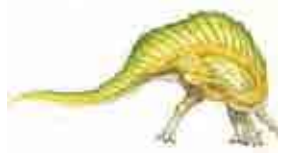
I/O Structure

- *Synchronous*: After I/O starts, control returns to user program only upon I/O completion
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access)
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing
- *Asynchronous*: After I/O starts, control returns to user program without waiting for I/O completion
  - **System call** – request to the OS to allow user to wait for I/O completion
  - **Device-status table** contains entry for each I/O device indicating its type, address, and state
  - OS indexes into I/O device table to determine device status and to modify table entry to include interrupt

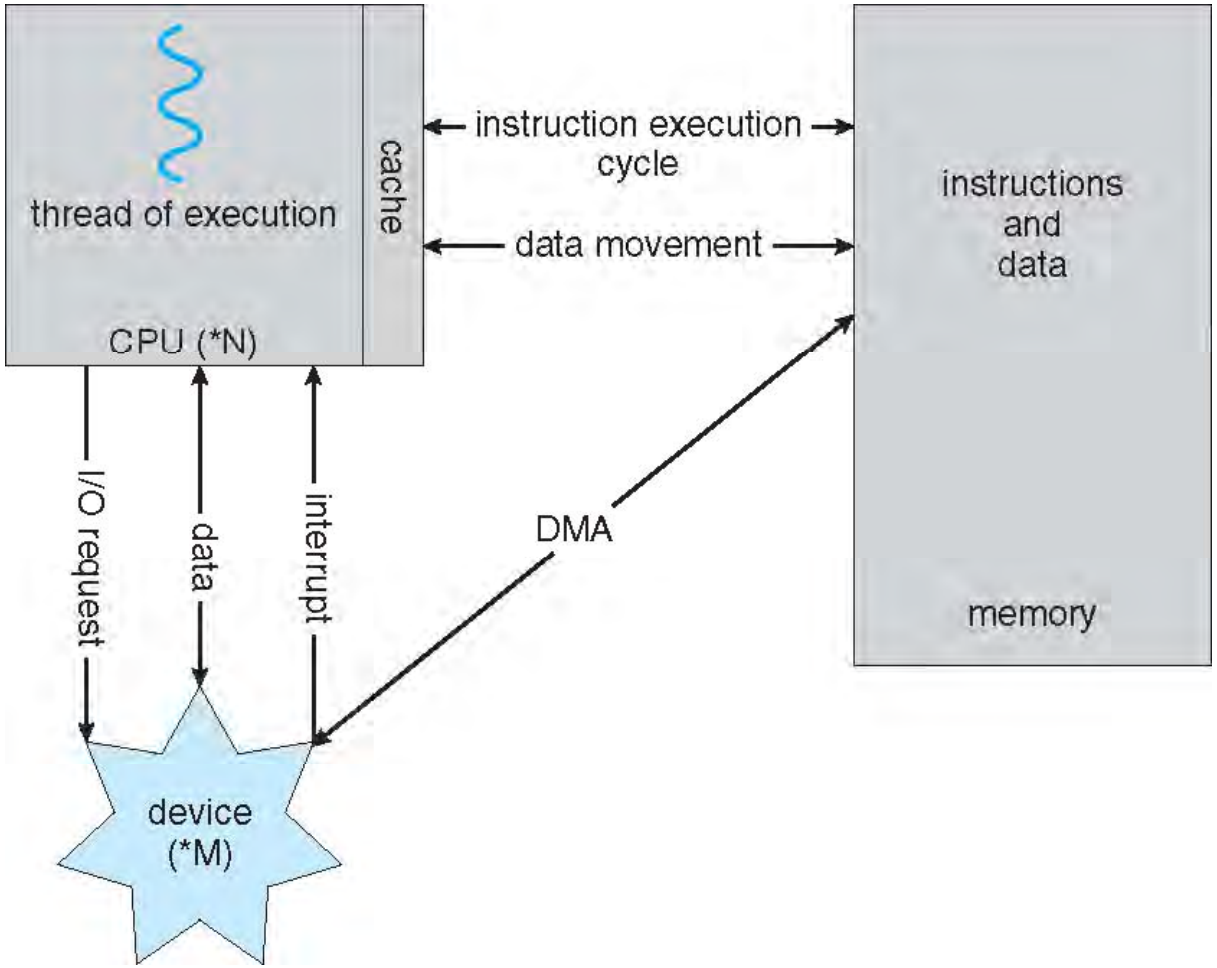


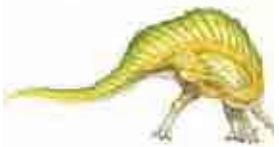
دسترسی مستقیم به حافظه

- I / O مبتنی بر وقفه برای انتقال حجم کمی از داده ها مناسب است. در I / O با حجم زیاد از دسترسی مستقیم به حافظه استفاده می شود.
- کنترلگر دستگاه یک بلاک از داده ها به طور مستقیم و بدون دخالت CPU از بافر خود به حافظه منتقل می کند.
- در این حالت به جای تولید یک وقفه برای هر کارکتر یک وقفه برای هر بلاک داده تولید می شود.(هر بلاک داده از چندین کارکتر تشکیل شده است)



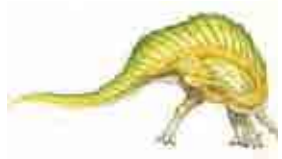
How a Modern Computer Works





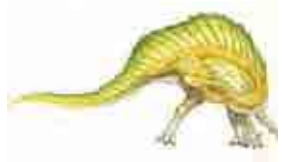
ساختار دخیره سازی

- حافظه اصلی تنها رسانه ذخیره سازی بزرگ است که پردازنده به طور مستقیم به آن دسترسی دارد.
- رسانه ذخیره سازی ثانویه- حافظه اصلی را گسترش می دهد و امکان ذخیره کردن غیر میرا را فراهم می آورد.
  - دیسک های مغناطیسی- دیسک فلزی سخت و یا پلاستیکی که روی آن با مواد مغناطیسی پوشیده شده است
    - سطح دیسک به طور منطقی به قسمت هایی به نام تراک (track) تقسیم می شوند که آن نیز به زیر قسمت هایی به نام سکتور تقسیم می شود.
    - کنترلگر دیسک تعیین کننده تعامل منطقی میان وسیله و کامپیوتر است.

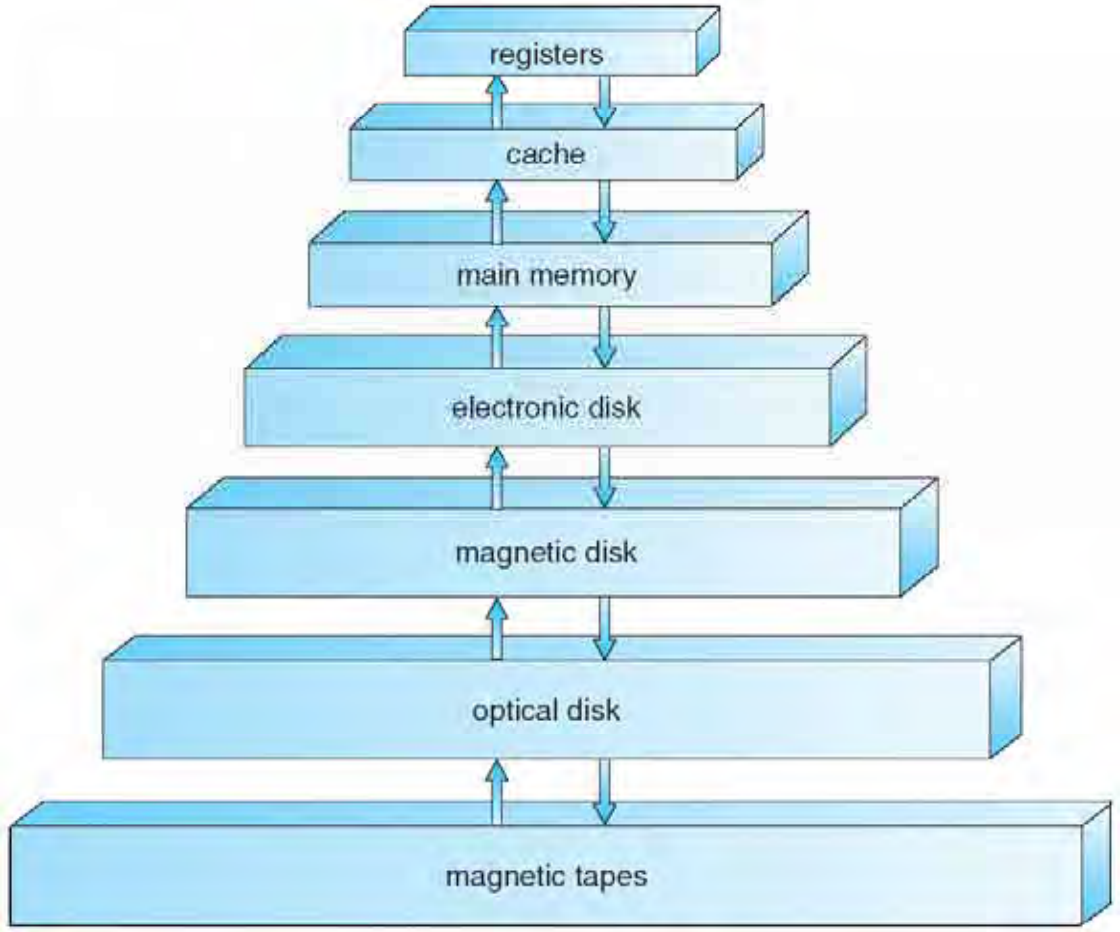


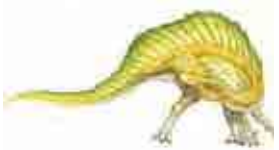
سلسله مراتب ذخیره سازی

- سازماندهی سلسله مراتب حافظه بر اساس عوامل زیر است
  - سرعت
  - هزینه
  - ظرفیت



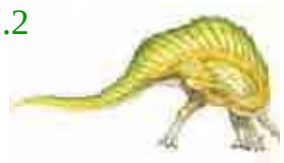
سلسله مراتب دستگاه های حافظه



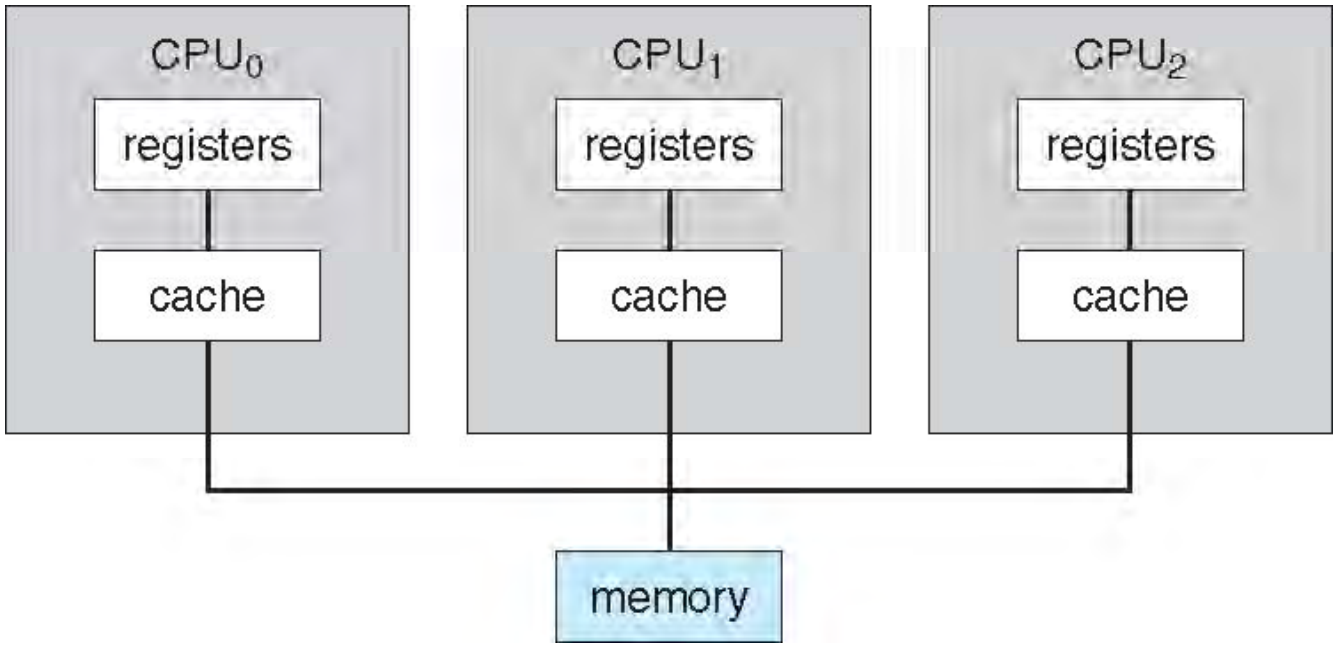


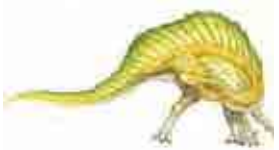
معماری سیستم کامپیوتری

- اغلب سیستم ها تنها از یک پردازنده عمومی و یا همنظوره استفاده می کنند. از کامپیوترهای دستی گرفته تا mainframeها
  - اغلب سیستم ها دارای پردازنده های خاص نیز هستند که کارهای خاصی را انجام می دهند و نمی توانند یک فرایند را به تنهایی انجام دهند.
- سیستم های چند پردازنده- از نظر کاربرد و بهبود عملکرد در حال رشد هستند
  - سیستم های موازی و یا سیستم های به هم وابسته نیز خوانده می شوند
  - سه مزیت عمده این سیستم ها
    1. افزایش توان عملیاتی
    2. صرفه جویی اقتصادی
    3. افزایش قابلیت اطمینان- تنزل مطبوع ، تحمل کننده خطا
  - سیستم های چند پردازنده بر دو نوع هستند
    1. چند پردازنده ای نامتقارن: ارباب برده ای
    2. چند پردازنده ای متقارن (SMP)

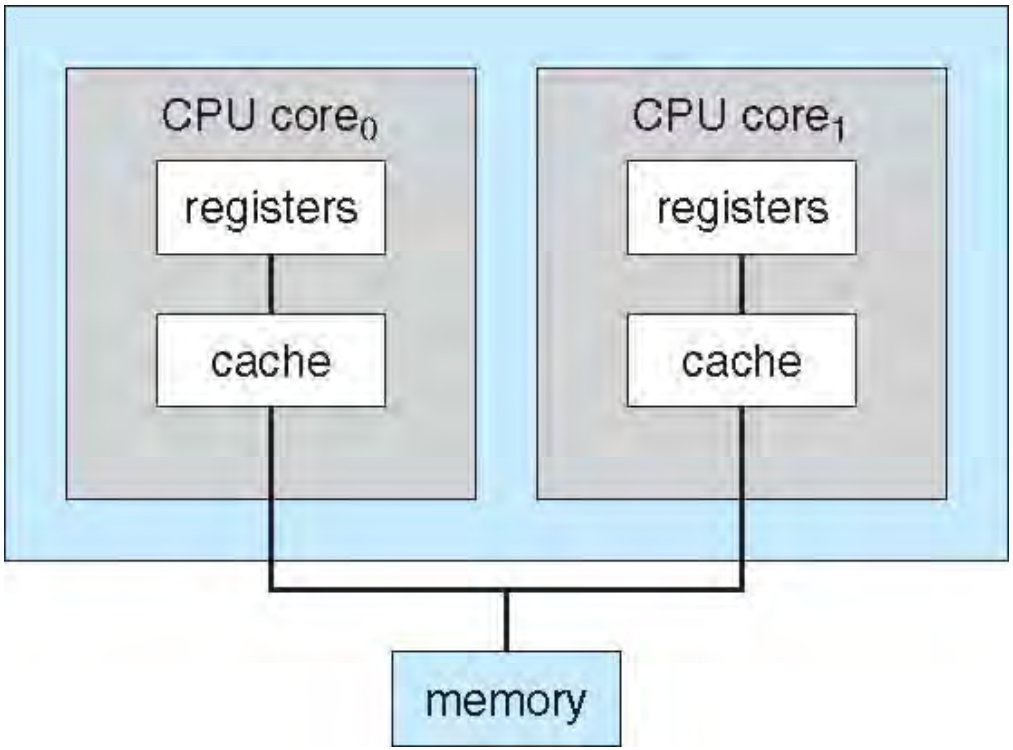


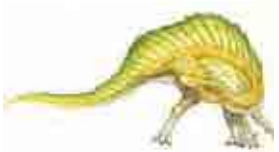
معماری چند پردازنده ای متقارن





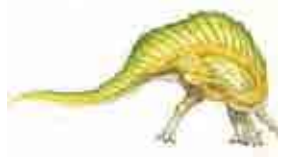
طراحی دو هسته ای به طوری که دو هسته در یک تراشه قرار دارند





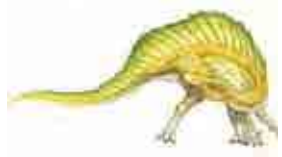
سیستم های خوشه ای

- شبیه سیستم های چند پردازنده ای هستند اما در حقیقت چند سیستم هستند که با یکدیگر کار می کنند
  - حافظه مشترکی دارند و از طریق شبکه های محلی (LAN) یا ارتباط داخلی سریع فراهم می شود .
  - یک سرویس همیشگی و ادامه دار فراهم می کند
    - در خوشه بندی نامتقارن، یک ماشین در حالت آماده قرار دارد، در حالی که ماشین دیگر در حال اجرای برنامه کاربردی است
    - در خوشه بندی متقارن- دو یا چند میزبان در حال اجرای برنامه کاربردی هستند و بر یکدیگر نظارت می کنند.
  - خوشه ها ممکن است برای فراهم کردن محیط محاسباتی با کارایی بالا (HPC)مورد استفاده قرار گیرند.
    - برنامه ها باید با استفاده از تکنیک موازی ساز نوشته شوند.



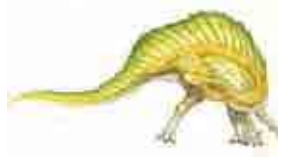
ساختار سیستم عامل

- **چند برنامگی برای افزایش کارآیی لازم است.**
  - یک کاربر آنقدر کار برای انجام ندارد که هم CPU و هم وسایل I/O را برای همیشه مشغول نگهدارد.
  - چند برنامگی بهروری CPU را افزایش می دهد زیر کارها را طوری سازماندهی می کند که در هر زمان کاری برای انجام باشد.
  - یک زیر مجموعه از کارها در حافظه نگه داشته می شود
  - زمانبند کار یک کار را انتخاب می کند و آن را اجرا می کند.
  - اگر کار در حال اجرا به حالت انتظار برود به عنوان مثال برای تکمیل شدن یک درخواست I/O سیستم عامل به کار دیگر سوئیچ می کند

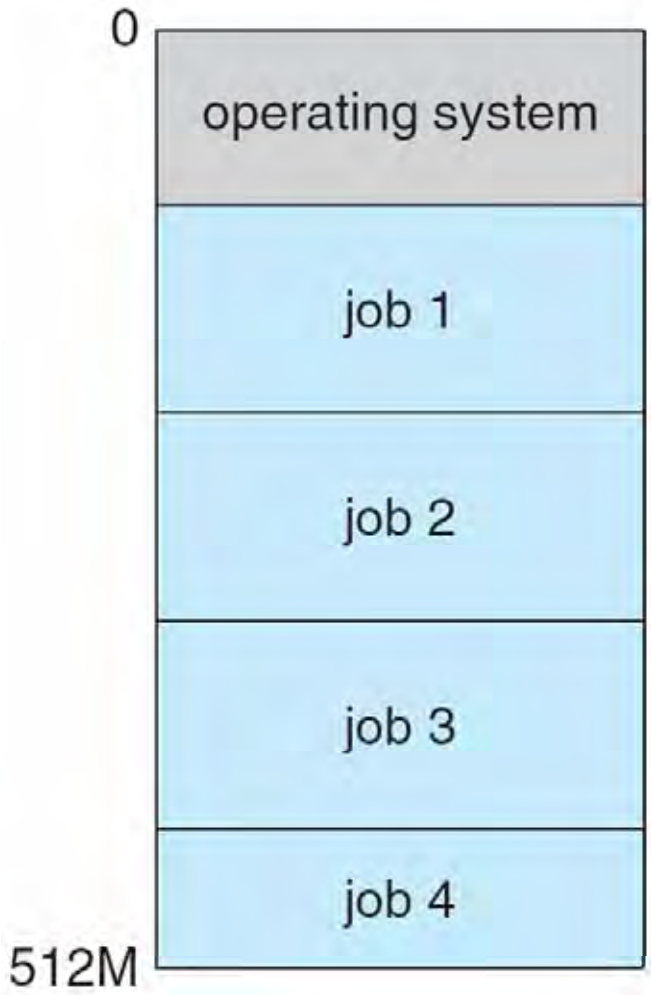


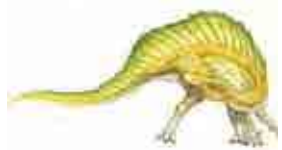
اشتراک زمانی(چند وظیفه ای): بسط منطقی چند برنامگی است

- پردازنده به طور مرتب در بین کارها سوئیچ می کند و این باعث میشود کاربر بتواند با برنامه در حال اجرا ارتباط داشته باشد.
- **زمان پاسخ باید کمتر از ۱ ثانیه باشد.**
- هر کاربر حداقل یک برنامه در حال اجرا در حافظه دارد که به آن پردازش میگویند.
- اگر چندین برنامه آماده برای اجرا در یک زمان داشته باشیم از زمانبند CPU استفاده می شود.
- اگر پردازش به طور کامل در حافظه قرار نگیرد از روش تعویض Swapping برای رفع مشکل استفاده میشود.
- **حافظه مجازی** امکان اجرای پردازش هایی را که بطور کامل در حافظه نیستن را می دهد.



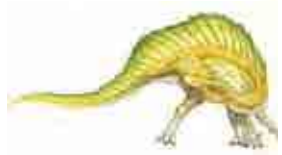
Memory Layout for Multiprogrammed System





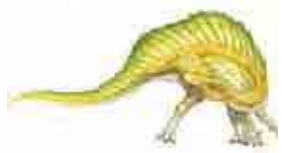
کارهای سیستم عامل

- وقفه به وسیله سخت افزار هدایت میشود.
- تله یا استثناء یک وقفه نرم افزاری است که ناشی از یک خطا مثل تقسیم بر صفر و یا ناشی از درخواست خاصی از برنامه کاربر است.



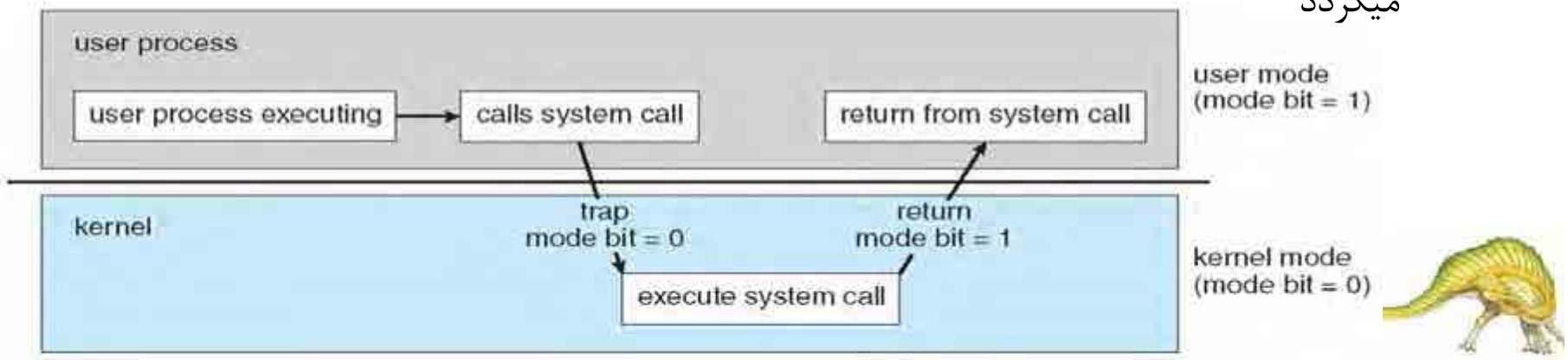
عملیات دو حالته

- **عملیات دو حالته** برای سیستم عامل امکان حفاظت خود و سایر قسمت های سیستم را فراهم می آورد.
  - **حداقل دو حالته جداگانه وجود دارد: حالت کاربر و حالت هسته**
  - **بیت حالت** به وسیله سخت افزار کامپیوتر فراهم میشود تا حالت فعلی را نشان دهد.
    - با توجه به بیت حالت قادر هستیم بین کاری که از طرف سیستم عامل اجرا میشود و بین کاری که از طرف کاربر انجام میگیرد تمایز قائل شویم.
    - بعضی از دستورات به عنوان دستورات ممتاز طراحی شده اند که تنها در حالت هسته اجرا میشوند.
    - وقتی برنامه کاربردی کاربر، سرویسی را از سیستم عامل درخواست می کند باید از حالت کاربر به حالت سیستم برویم تا درخواست انجام شود و سپس به حالت کاربر بر میگردیم.



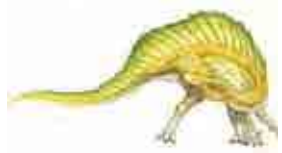
انتقال از حالت کاربر به حالت هسته

- زمان سنج اجازه نمی دهد برنامه کاربر در حلقه ی نامتناهی باقی بماند یا در فراخوانی سرویس های سیستم با شکست مواجه شود و هرگز کنترل را به سیستم عامل برنگرداند
  - تایمر پس از دوره ی معین کامپیوتر را دچار وقفه می نماید
  - سیستم عامل، شمارنده را مقدار دهی می کند.
  - هر وقت ساعت تیک می کند، از شمارنده یک واحد کم میشود
  - زمانی که شمارنده صفر میشود یک وقفه رخ می دهد
  - قبل از انتقال کنترل به کاربر، سیستم عامل تضمین می کند که تایمر برای صدور وقفه تنظیم شده است. اگر تایمر وقفه ای را تولید نماید، کنترل به طور خودکار به سیستم عامل منتقل میگردد

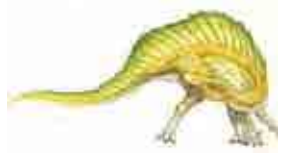


مدیریت فرآیند

- یک پردازش (موجودیت فعال) در حقیقت یک برنامه (موجودیت غیرفعال) در حال اجرا است.
  - پردازش واحد کار در سیستم است.
  - هر فرآیند به منابعی نیاز دارد تا وظیفه اش را انجام دهد.
    - CPU, حافظه ، I/O, فایل,
    - مقداردهی اولیه داده ها
  - وقتی فرآیند خاتمه می یابد، سیستم عامل تمام منابع قابل استفاده مجدد را بازیابی می کند



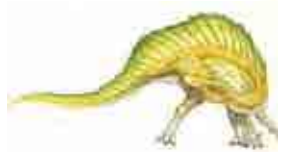
- بطور معمول سیستم دارای چندین پردازش است که بعضی از آنها متعلق به کاربر و بعضی دیگر متعلق به سیستم عامل می باشد که به صورت همروند بر روی یک و یا چند پردازنده اجرا میشوند
  - همروندی به وسیله تقسیم CPU در میان پردازش ها و بندها انجام میشود
- فرآینده های تک نخی دارای یک شمارنده برنامه هستند که محل دستور العمل بعدی را برای اجرا مشخص می کند.
  - دستورات فرآیند یکی پس از دیگری اجرا میشود تا فرآیند به طور کامل اجرا شود، علاوه بر این، در هر زمان، حداکثر یک دستور از فرآیند اجرا میشود.
- فرآیند چندنخی، چندین شمارنده ی برنامه دارد، که هر کدام به دستورالعمل بعدی که باید برای نخی اجرا شود، اشاره میکند.



مدیریت فرآیند توسط سیستم عامل

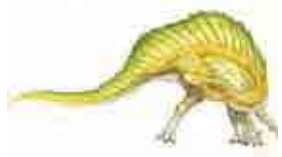
سیستم عامل در رابطه با مدیریت فرآیند، مسئول کارهای زیر است:

- زمان بندی فرآیندها و نخ ها در پردازنده
- ایجاد و حذف فرآیندهای کاربر و سیستم
- به تعویق انداختن و از سرگیری فرآیندها
- فراهم کردن راهکاری برای همگام سازی فرآیند
- فراهم کردن راهکاری برای برقراری ارتباطات



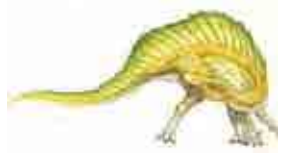
مدیریت حافظه

- تمام داده ها قبل از اجرا باید در حافظه قرار گیرند.
- دستورالعمل ها برای اجرا شدن باید در حافظه قرار گیرند
  - برای بهبود بهره وری CPU و سرعت پاسخ کامپیوتر برای کاربران، کامپیوترهای همه منظوره باید چندین برنامه را در حافظه نگه دارند، و در نتیجه نیاز به مدیریت حافظه است
- سیستم عامل در رابطه با مدیریت حافظه، مسئول فعالیت های زیر است:
  - چه بخش هایی از حافظه و توسط چه کسانی در حال استفاده است.
  - تصمیم گیری راجع به این که کدام فرآیندها و داده ها باید به حافظه منتقل و یا از آن خارج شوند.
  - تخصیص و آزادسازی فضای حافظه در صورت نیاز



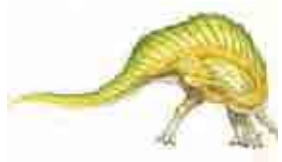
مدیریت ذخیره سازی

- سیستم عامل یک دید منطقی و یکنواخت از ذخیره سازی اطلاعات را فراهم می سازد.
  - فایل- سیستم عامل از خواص فیزیکی دستگاه های ذخیره سازی، یک واحد ذخیره سازی اطلاعات را فراهم می سازد.
  - هر رسانه مانند دیسک مغناطیسی، دیسک نوری و نوار مغناطیسی توسط دستگاهی کنترل میشود
    - هر یک از رسانه ها ویژگی و سازمان فیزیکی خاص خودشان را دارند.



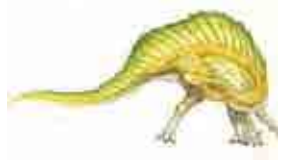
مدیریت سیستم فایل

- فایل ها به طور معمول در دایرکتوری ها سازماندهی می شوند
- کنترل کننده دسترسی در بیشتر سیستم ها تعیین کننده این است که چه کسی/کسانی می توانند به چه فایل یا فایل هایی دسترسی داشته باشند
- وظایف مدیریت سیستم فایل
  - ایجاد و حذف فایل ها و دایرکتوری ها
  - اجازه دستکاری فایل ها و داده ها
  - نگاشت فایل ها بر روی حافظه ثانویه
  - پشتیبان گیری از فایل ها بر روی رسانه های ذخیره سازی پایدار



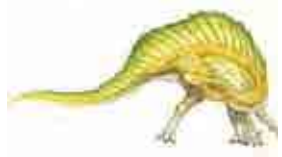
مدیریت حافظه انبوه

- بطور معمول دیسک برای ذخیره اطلاعاتی که در حافظه اصلی جا نمی شود و یا باید برای مدت طولانی نگهداری شوند به کار می رود
  - مدیریت درست دیسک از اهمیت ویژه ای برخوردار است
- سیستم عامل در رابطه با مدیریت دیسک مسئول کارهای زیر است:
  - مدیریت فضای آزاد Free-space management
  - تخصیص حافظه Storage allocation
  - زمانبندی دیسک Disk scheduling
- حافظه های جانبی دیگری که از دیسک ها کندتر و ارزانتر هستند را حافظه های ثالث گویند
  - حافظه های ثالث مانند: وسایل ذخیره سازی نوری، نوارهای مغناطیسی
  - این رسانه ها دارای فرمت های مختلفی مانند WORM یک بار نوشتن چند بار خواندن ، RW خواندن – نوشتن



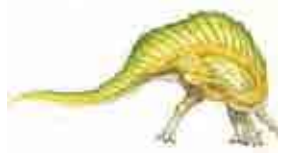
کارآیی سطوح مختلف حافظه

Level	1	2	3	4
Name	registers	cache	main memory	disk storage
Typical size	< 1 KB	> 16 MB	> 16 GB	> 100 GB
Implementation technology	custom memory with multiple ports, CMOS	on-chip or off-chip CMOS SRAM	CMOS DRAM	magnetic disk
Access time (ns)	0.25 – 0.5	0.5 – 25	80 – 250	5,000.000
Bandwidth (MB/sec)	20,000 – 100,000	5000 – 10,000	1000 – 5000	20 – 150
Managed by	compiler	hardware	operating system	operating system
Backed by	cache	main memory	disk	CD or tape



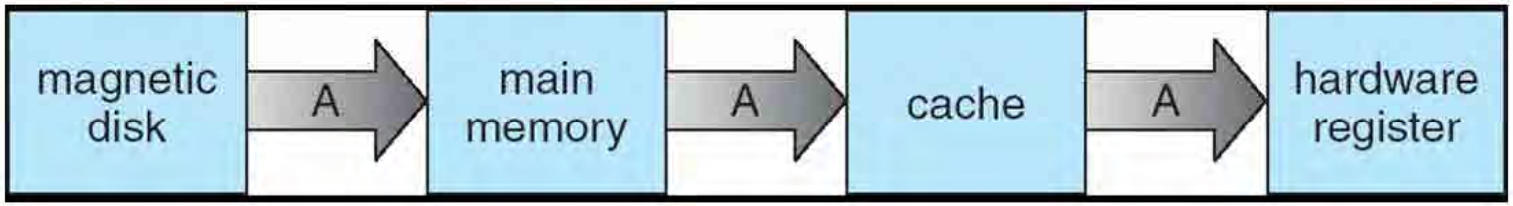
حافظه نهان

- انتقال اطلاعات بین سطوح سلسله مراتب حافظه، می تواند به صورت صریح و یا ضمنی باشد که به طراحی سخت افزار، سیستم عامل و نرم افزار بستگی دارد مانند انتقال داده از حافظه نهان به CPU که سیستم عامل دخالت ندارد یا از دیسک به حافظه اصلی
- اطلاعات در حال استفاده به طور موقت از حافظه های کند به حافظه های سریع منتقل می شوند
- در هنگام جستجوی اطلاعات ابتدا حافظه های سریعتر برای تعیین بودن و یا نبودن اطلاعات بررسی می شوند
  - اگر داده های مورد نیاز در حافظه سریع Cacheباشد از آن ها استفاده میشود در غیر اینصورت داده ها به داخل حافظه سریع کپی شده و سپس استفاده می شود

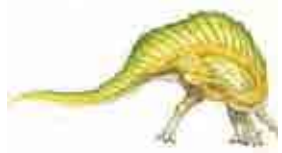


انتقال مقدار صحیح A از دیسک به ثبات سخت افزاری

- در محیط های چند پردازنده ای باید بسیار مراقب بود چون ممکن است کپی های مختلفی از یک داده در حافظه های نهان مختلف ذخیره شده باشد و بر روی آن ها عملیات مختلف انجام شود

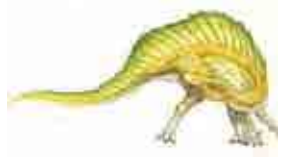


- در محیط های چند پردازنده ای باید اطمینان حاصل کنیم که به هنگام سازی یک مقدار در یک حافظه نهان فورا در تمام حافظه های نهان دیگر منعکس می شود این وضعیت **انسجام حافظه نهان** نام دارد و معمولا یک مسئله سخت افزاری است.
- در محیط های توزیع شده مانند شبکه های کامپیوتری وضعیت پیچیده تر است چون ممکن است یک کپی از یک فایل در چند جا موجود باشد



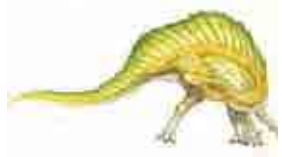
زیر سیستم های ورودی- خروجی I/O

- یکی از اهداف سیستم عامل مخفی کردن ویژگی های خاص دستگاه های سخت افزاری از کاربر است (همانطور که در یک شبکه فایل ها را جا به جا می کنید بر سیستم خود نیز این کار را انجام می دهید)
- زیر سیستم I/O شامل چندین قطعه است
  - مدیریت حافظه I/O
    - بافر کردن (ذخیره موقتی داده ها در هنگام نقل و انتقال آن ها)
    - حافظه نهان (ذخیره بخشی از داده ها در حافظه های سریعتر برای افزایش کارآیی)
    - اسپولینگ (روی هم انداختن (همزمان کردن) ورودی یک کار با خروجی کار دیگر)
  - واسط عمومی گرداننده دیسک
  - کنترل کننده مربوط به دستگاه های سخت افزاری



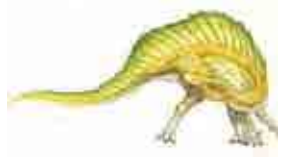
حفاظت و امنیت

- **حفاظت – قواعدی که برای کنترل دسترسی کاربر و یا پردازش ها به منابع سیستم به وسیله سیستم عامل تعریف می شود**
- **امنیت – دفاع از سیستم در برابر حمله های داخلی و خارجی**
  - مانند حملات ویروس ها و کرم ها ، هویت سرقت شده، استفاده از تمام منابع سیستم و عدم امکان استفاده ی کاربران مجاز از سیستم
- سیستم ها به طور معمول ابتدا کاربر را از بین کاربرهای خود تمیز می دهد تا بتواند تشخیص دهند چه کاربری چه کاری را انجام می دهد
  - هویت کاربر (**user IDs**, security IDs) : نام و شماره ای که به کاربر داده میشود
  - شناسه کاربر به تمام فایل ها و پردازش های او انتساب داده میشود تا با آن بتوان سطح دسترسی را تعیین کرد
  - شناسه گروه اجازه می دهد تا مجموعه ای از کاربران را تعریف کنیم و آن ها را مدیریت کرده و سطح دسترسی آن ها را تعیین کنیم



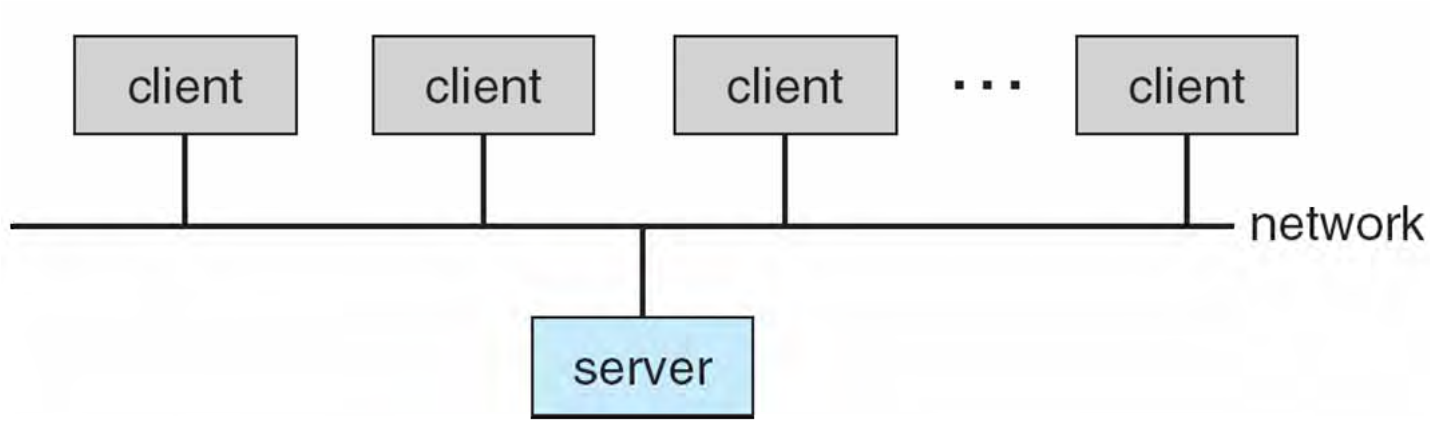
محیط های محاسباتی

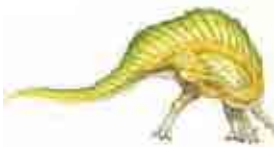
- سیستم های سنتی
  - سیستم های سنتی در حال رنگ باختن هستند
    - در گذشته در یک اداره PC ها از طریق شبکه به کامپیوتر سرور وصل می شدند و کاربر می توانست کار خود را انجام دهد مانند انتخاب واحد سنتی و یا سیستم بانک ها PCs
    - امروز شرکت ها و موسسات پورتال هایی را فراهم آورده اند که از طریق اینترنت از هر کجایی می توان به آن وصل شد و کار خود را انجام داد
    - امروز یک کاربر با استفاده از یک مودم میتواند از منزل خود به شبکه های مختلف وصل شود و فایروال هایی را برای امنیت استفاده که در گذشته بسیار گران بودن و قابل استفاده برای هر کس نبود



محیط های محاسباتی

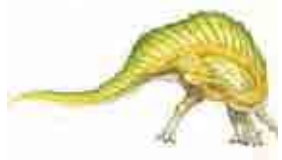
- محاسبات کلاینت – سرور
  - با پیشرفت PC ها به تدریج آن ها جایگزین ترمینال ها در شبکه ها شدن
  - بسیاری از سیستم ها امروزی، به عنوان سیستم های سرور (سرویس دهنده)عمل می کنند تا به درخواست های تولیده شده توسط سیستم های کلاینت (مشتری) را برآورده نمایند
    - **سیستم های سرور** واسطی را فراهم می کنند تا کاربر از طریق آن بتواند درخواست های خود را مثلا به پایگاه داده بفرستد
    - **سرور فایل File-server: واسطی برای دخیره و بازیابی اطلاعات کاربر فراهم می کند.**





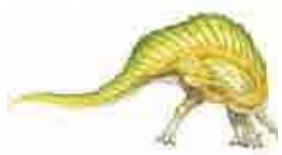
Peer-to-Peer محاسبات همتا به همتا (نظیر به نظیر)

- یک مدل از سیستم های توزیع شده می باشد
- در این سیستم ها کلاینت و سرور مشخصی وجود ندارد هر سیستم هم کلاینت است و هم سرور و همه همتا و هم مرتبه هستند

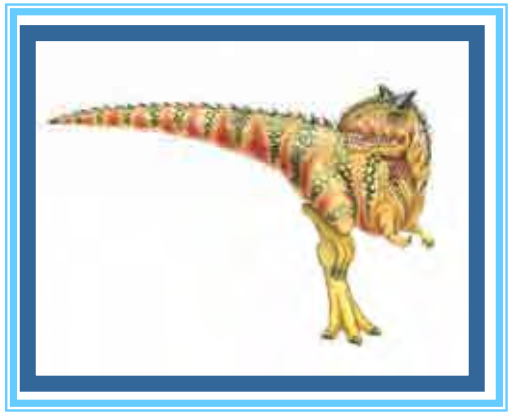


محاسبات مبتنی بر وب Web-based

- وب در حال همه گیر شدن است و در بسیاری از دستگاه ها قابل دستیابی است.
  PC ها هنوز متداول ترین دستگاه های دستیابی وب هستند
  - More devices becoming networked to allow web access
- پیاده سازی محاسبات مبتنی بر وب، دسته های جدیدی از دستگاه ها ،مثل متوازن کننده بار **load balancers** را پدید آورده است ، که اتصالات شبکه را بین مخزنی از سرورها مشابه توزیع می کند .



End of Chapter 1



فصل دوم : ساختار سیستم عامل

Outline

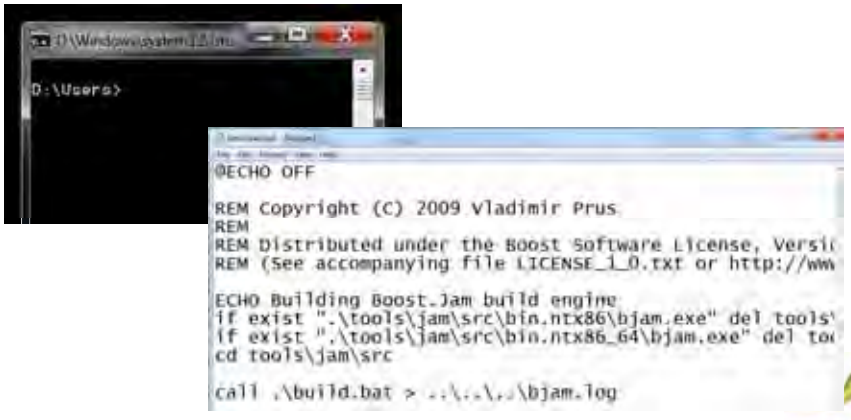
- Operating-System Services
- User Operating-System Interface
- System Calls
- Types of System Calls
- System Programs
- Operating-System Design and Implementation
- Operating-System Structure
- Virtual Machines
- Operating-System Debugging
- Operating-System Generation
- System Boot

اهداف این فصل

- توصیف سرویس هایی که سیستم برای کاربران،فرآیندها و سایر سیستم ها فراهم می آورد.
- بحث درباره روش های مختلف سازماندهی سیستم عامل
- چگونگی نصب سیستم عامل و راه اندازی آن

Operating System Services

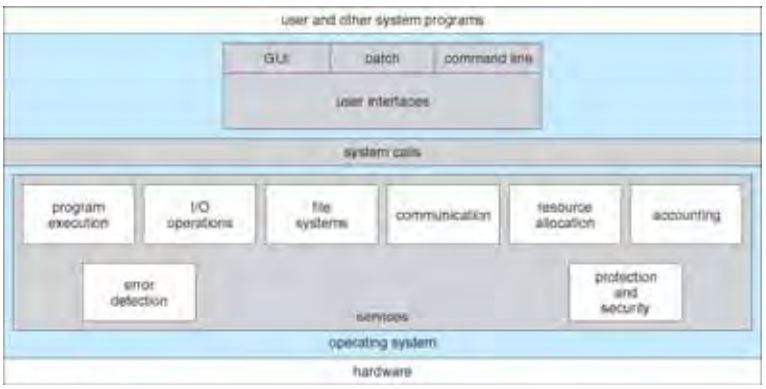
- سرویس های سیستم عامل برای کمک به کاربران
  - واسط کاربر (User interface (UI
    - واسط خط فرمان (CLI) ، واسط گرافیکی کاربر(GUI)، واسط دسته ای مانند bat فایل ها



Operating System Services

- سرویس های سیستم عامل برای کمک به کاربران
  - اجرای برنامه - سیستم باید قادر باشد برنامه ای را وارد حافظه کند و آن را اجرا نماید. خاتمه دادن به برنامه چه بصورت نرمال و یا غیر نرمال
  - عملیات های I/O - برنامه در حال اجرا ممکن است نیاز به I/O داشته باشد، که شامل فایل ها یا دستگاه I/O می باشد
  - پردازش سیستم فایل- خواندن و نوشتن فایل ها و دایرکتوری ها، جستجوی آن ها ، لیست اطلاعات فایل ها و اجازه دسترسی به آن ها

A View of Operating System Services



Operating System Services (Cont)

- سرویس های سیستم عامل برای کمک به کاربران
  - ارتباطات- مبادله اطلاعات بین فرآیندهای یک کامپیوتر و یا کامپیوترهای مختلف روی یک شبکه
    - ارتباط ممکن است از طریق حافظه مشترک و یا از طریق ارسال پیام باشد
  - تشخیص خطا- سیستم عامل همیشه باید خطا های ممکن را تشخیص دهد
    - خطا ممکن است در پردازنده، حافظه، سخت افزار، وسایل I / O و برنامه های کاربر رخ دهد
    - برای هر نوع خطا سیستم عامل باید عمل مناسبی را انجام دهد تا محاسبات درست و سازگار باشند
    - امکانات خطا یابی میتواند توازن کاربر و برنامه نویس را در استفاده از سیستم افزایش دهد.

Operating System Services (Cont)

- بعضی از سرویس ها در OS برای تضمین عملیات کارآمد خود سیستم عامل فراهم شده مانند اشتراک منابع
  - **تخصیص منابع**-تخصیص منابع به چند ین کاربر و یا کار به طور همزمان
    - انواع مختلف منابع
      - پردازنده، حافظه اصلی، فایل دخیره شده، وسایل I / Oو ...
  - **حسابرسی- چه کاربرانی از چه منابعی از کامپیوتر به چه میزان استفاده کرده اند**

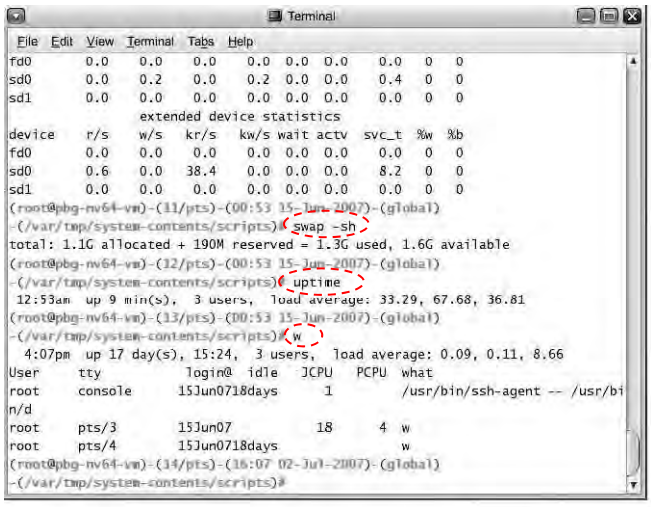
- **حفاظت و امنیت – کنترل کاربرد اطلاعات ، پردازش های همزمان نباید با هم تداخل داشته باشند**
  - **حفاظت** : اطمینان از این که دسترسی به تمام منابع کنترل شده است
  - **امنیت: به وسیله شناسایی کاربران به وسیله کلمه عبور و رمز آن ها علاوه بر این دستگاه های I/O خارجی مانند مودم ها و کارت شبکه را از دستیابی های غیر مجاز مصون نگهداریم**
  - پیش گیری باید در سراسر سیستم انجام گیر قدرت زنجیر به ضعیف ترین پیوند آن بستگی دارد

User Operating System Interface - CLI

- واسط خط فرمان یا مفسر فرمان اجازه می دهد تا کاربران به طور مستقیم دستورات خود را وارد کنند و سیستم عامل آن ها را اجرا نماید
  - در بعضی از سیستم عامل ها به عنوان هسته در نظر گرفته می شود و در بعضی دیگر به عنوان یک برنامه
  - پوسته- در بعضی از موراد چندین مفسر فرمان وجود دارد که میتوان بین آن ها انتخاب کرد که به آن پوسته گویند.
  - کار اصلی مفسر فرمان گرفتن و اجرای دستورات کاربر است
    - یا خود مفسر فرمان شامل کدی است برای اجرای فرمان یا به وسیله برنامه های سیستم انجام می شود
      - در این حالت مفسر فرمان به هیچ وجه فرمان را نمیشناسد و به راحتی میتوان دستورات را اضافه کرد

Bourne Shell Command Interpreter

- An example in Solaris 10



User Operating System Interface - GUI

- User-friendly desktop metaphor interface
  - به طور معمول از موس، صفحه کلید و مانیتور استفاده میشود
  - آیکن ها – نشان دهنده ی فایل ها، برنامه ها، دایرکتوری ها و اعمال سیستم است
  - بسته به محلی که اشاره گر موس قرار دارد فشار دادن دکمه موس باعث انجام اعمال مختلفی میشود مانند نمایش اطلاعات، تنظیمات،اجرای توابع، باز شدن پوشه ها
  - واسط های کاربر گرافیکی، نتیجه تحقیق Xerox PARCدر(in 1970s)
- بیشتر سیستم ها دارای هر دو واسط GUI و CLI است مانند Microsoft Windows، Apple Mac OS X و Solaris

The Mac OS X GUI

System Calls

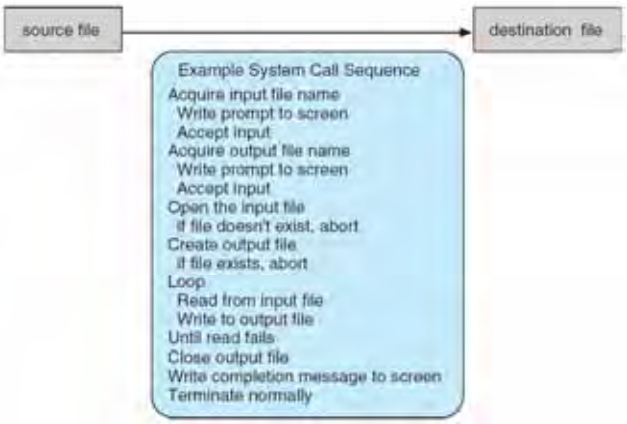
- فراخوانی های سیستمی، واسطی برای استفاده برنامه نویسان از سرویس ها سیستم است
  - این فراخوانی ها معمولا به صورت روال هایی نوشته شده در C و C++ وجود دارند البته کارهای سطح پایین با دستورات زبان اسمبلی نوشته میشود
- البته بیشتر برنامه نویسان از این جزئیات خبر ندارند و آن ها از واسط برنامه نویسی کاربردی Application Program Interface (API) به جای فراخوانی مستقیم استفاده می کنند
- API، مجموعه ای از توابع است تا برنامه نویس کاربردی کار خود را با آن انجام دهد

System Calls

- سه API مهم برای برنامه نویسان کاربردی عبارت است از:
  - Win32 API for Windows
  - POSIX API for POSIX-based systems(شامل یونیکس، لینوکس و Mac OS X versions of UNIX, Linux, and Mac OS X
  - Java API <= برای طراحی برنامه هایی که در ماشین مجازی جاوا اجرا میشود
- برای چه از APIها به جای فراخوانی های سیستمی استفاده میشود
  - قابلیت حمل برنامه Portability
  - فراخوانی های سیستمی دارای جزئیات زیادی هستند و کار کردن با آن ها مشکل است

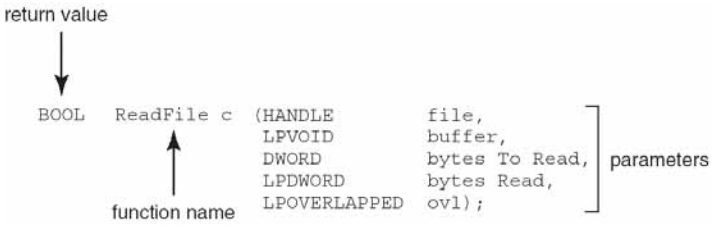
Example of System Calls

- مثالی از چگونگی استفاده از فراخوانی ها سیستمی برای کپی یک فایل



Example of Standard API

- یک تابع API برای خواندن فایل در سیستم عامل ویندوز ماکروسافت
-

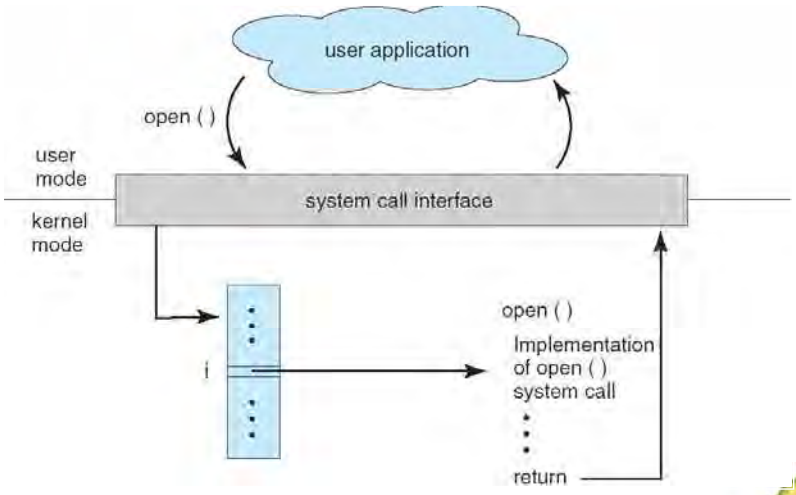


- توصیف پارامترهای تابع
  - HANDLE file—the file to be read
  - LPVOID buffer—a buffer where the data will be read into and written from
  - DWORD bytesToRead—the number of bytes to be read into the buffer
  - LPDWORD bytesRead—the number of bytes read during the last read
  - LPOVERLAPPED ovl—indicates if overlapped I/O is being used

System Call Implementation

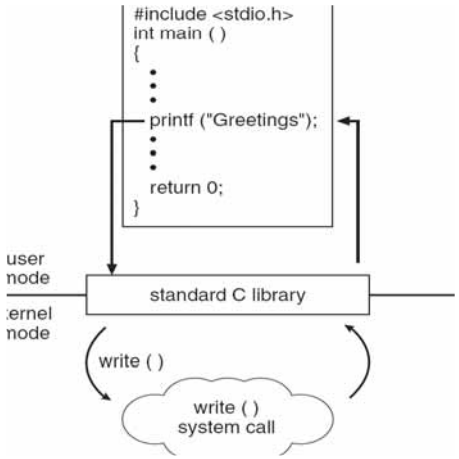
- معمولا به هر فراخوانی سیستمی یک عدد نسبت داده میشود
  - واسط فراخوان سیستم جدولی را نگهداری میکند که این عدد به عنوان شاخص در آن عمل می کند
- واسط فراخوان سیستم، فراخوانی سیستمی مورد نظر را در هسته ی سیستم عامل احضار می کند و وضعیت فراخوان سیستمی و هر مقدار برگشتی را بر می گرداند
- فراخوانی کننده، به هیچ اطلاعاتی راجع به چگونگی پیاده سازی فراخوان سیستمی نیاز ندارد
  - فقط لازم است از API تبعیت کند و پی ببرد که سیستم عامل در اثر اجرای آن فراخوان سیستم چه کاری را انجام می دهد
  - اغلب جزئیات واسط سیستم عامل، توسط API از برنامه نویس مخفی می شود و توسط کتابخانه های زمان اجرا مدیریت می گردد

API – System Call – OS Relationship



Standard C Library Example

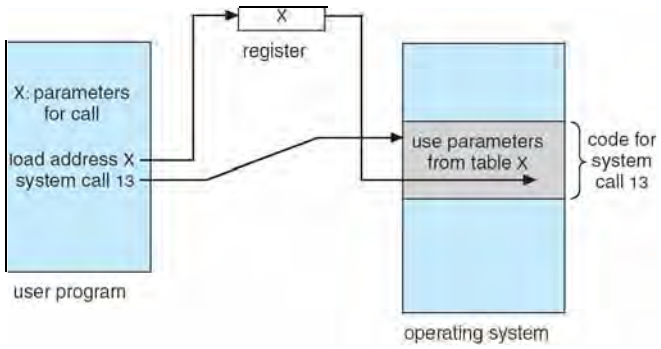
- شکل زیر نشان می دهد که دستور printf() در زبان C چگونه به یک فراخوانی سیستمی تبدیل می شود



System Call Parameter Passing

- فراخوان های سیستم، به روش های مختلفی انجام میگردد، که غالبا، به اطلاعاتی بیش از هویت فراخوان سیستمی نیاز است
  - نوع و میزان دقیق اطلاعات، بسته به نوع سیستم عامل و فراخوانی، فرق می کند
- از سه روش کلی برای ارسال پارامترها به سیستم عامل استفاده می شود:
  - ساده ترین روش ارسال پارامترها از طریق ثبات ها است
    - اما ممکن است تعداد پارامترها از ثبات ها بیشتر باشد
  - پارامترها معمولا در بلوک و یا جدولی از حافظه ذخیره می شوند و آدرس بلوک به عنوان پارامتر در ثبات قرار مگیرد
    - مثال Ex: Linux and Solaris
  - پارامترها می توانند توسط برنامه در پشته قرار گیرند و توسط سیستم عامل از پشته برداشته شودند
    - سیستم های عامل روش بلوکی یا پشته را ترجیح می دهند، زیرا این روش ها تعداد و طول پارامترهای ارسالی را محدود نمی کند

Parameter Passing via Table



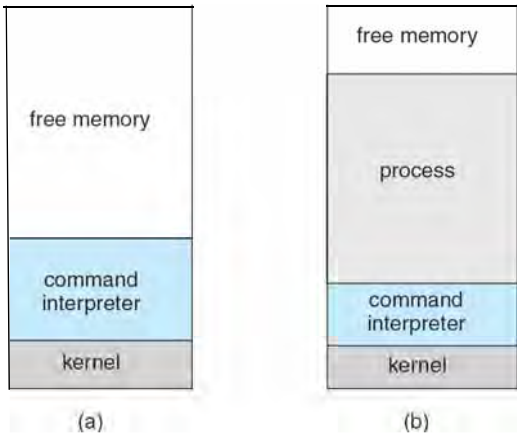
انواع فراخوانی های سیستمی

- کنترل فرایند Process control
- مدیریت فایلFile management
- مدیریت دستگاه Device management
- نگهداری اطلاعات Information maintenance
- ارتباطات Communications
- امنیت Protection

مثالی از فراخوانی های سیستمی که در ویندوز و لینوکس فراهم شده

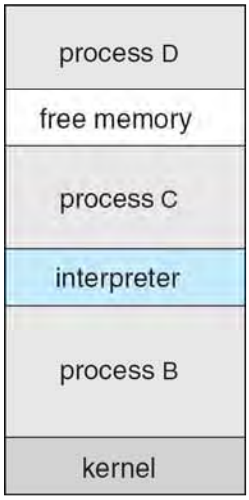
	Windows	Unix
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Manipulation	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shmget() mmap()
Protection	SetFileSecurity() InitializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

MS-DOS execution



(a) At system startup (b) running a program

FreeBSD Running Multiple Programs



برنامه های سیستمی

- برنامه های سیستمی، که امکانات سیستمی نیز خوانده می شوند، محیط آسانی را برای تولید و اجرای برنامه فراهم می سازند
  - کار کردن با فایل ها
  - اطلاعات وضعیتی
  - تغییر فایل
  - پشتیبانی از زبان های برنامه سازی
  - بارکردن و اجرای برنامه ها
  - ارتباطات
- نمایی از سیستم عامل که توسط اغلب کاربران دیده میشود، توسط برنامه های سیستمی و کاربردی تعریف می گردد

System Programs

- ارائه یک محیط مناسب برای توسعه ، نوشتن و اجرای برنامه ها
  - برخی از آن ها ساده بوده به سادگی یک فراخوانی سیستمی و بعضی دیگر پیچیده هستند
- مدیریت فایل- ایجاد، حذف، کپی، تغییر نام، چاپ و دستکاری فایل ها و دایرکتوری ها
- اطلاعات وضعیت
  - اطلاعات سیستم-زمان، تاریخ، مقدار حافظه در دسترس، فضای دیسک و تعداد کاربران
  - جزئیات کارکرد، اشکالزدایی اطلاعات
  - بعضی از سیستم ها از ریجستری استفاده می کنند و اطلاعات پیکربندی را در آن ذخیره و از آن بازیابی می کنند.

System Programs (cont'd)

- اصلاح فایل ها
  - ویرایشگرهای متن برای ایجاد و اصلاح فایل ها
  - دستورات خاص برای جستجوی محتویات فایل ها یا انجام تغییرات در متن ها
- پشتیبانی از زبان های برنامه نویسی- کامپایلرها، اسمبلرها، برنامه های اشکالزدایی و مفسرها
- بارگزاری و اجرای برنامه ها – لود شدن در حالت مطلق، لود شدن در حالت قابل جابه جایی اشکالزدایی سیستم برای زبان های ماشین سطح بالا و پایین
- ارتباطات- مکانیزمی برای ایجاد ارتباط مجازی در بین پردازش ها، کاربران، و سیستم های کامپیوتری
  - این قسمت به کاربران اجازه می دهد تا پیامی را به صفحه نمایش دیگر کاربران، مرورگرهای وب بفرسند حتی امکان ارسال پیام های و نامه های الکترونیکی را فراهم می آورد همچنین انتقال فایل از یک سیستم به سیستم دیگر

طراحی و پیاده سازی سیستم عامل

- جواب های کاملی برای مسئله طراحی و پیاده سازی سیستم عامل ها وجود ندارد اما روش هایی وجود دارند که اثبات شده اند و موفق هستند.
  - ساختار درونی سیستم عامل های مختلف به طور گسترده ای می تواند متفاوت باشد
  - شروع کار طراحی و پیاده سازی به وسیله تعریف اهداف و مشخصات است
  - اما تحت تاثیر انتخاب سخت افزار و نوع سیستم است.
- اهداف طرح
  - اهداف کاربر - سیستم عامل باید برای استفاده راحت، آسان برای یادگیری، قابل اعتماد، امن و سریع
  - اهداف سیستم – سیستم عامل باید آسان برای طراحی، پیاده سازی ، و نگهداری، و همچنین به عنوان قابل انعطاف، قابل اعتماد و عاری از خطا، و کارآمد

طراحی و پیاده سازی سیستم عامل

- سیاست باید از مکانیزم جدا باشد
  سیاست ها : چه باید کرد؟ (هر برنامه باید مدت زمان مشخص و محدودی اجرا شود)
  مکانیسم : چگونه می توان آن را انجام دهد؟ (این کار را با زمان سنج انجام دهیم)
- این یک اصل بسیار مهم است و باعث حداکثر شدن انعطاف میشود و می توانیم اگر خواستیم در آینده سیاست را به آسانی تغییر دهیم .

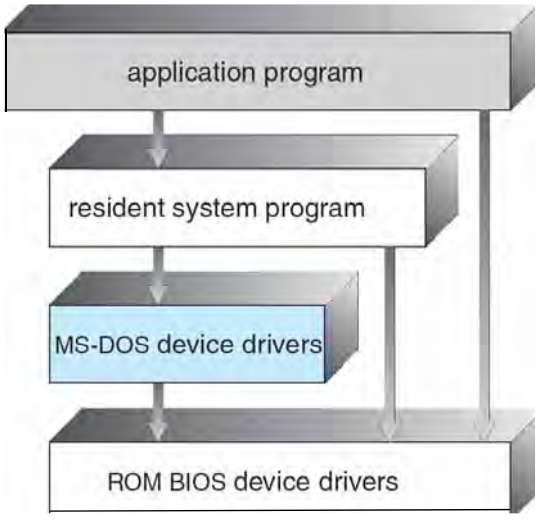
ساختار سیستم عامل OS Structure

- ساختار ساده
- لایه ای
- ریز هسته یا ریز کرنل
- پیمانه ها (ماژول ها)
- ترکیبی

ساختار ساده

- MS-DOS: به منظوره ارائه بیشترین کارآیی در حداقل فضا نوشته شده است
  - به ماژول های مختلف تقسیم نشده است
  - در این سیستم عامل واسط ها و سطوح عملکرد به خوبی تفکیک نشده اند برای مثال، برنامه های کاربردی قادرند به روال های I/O پایه نیز دستیابی داشته باشند یا مستقیما در نمایشگر و گرداننده های دیسک بنویسند

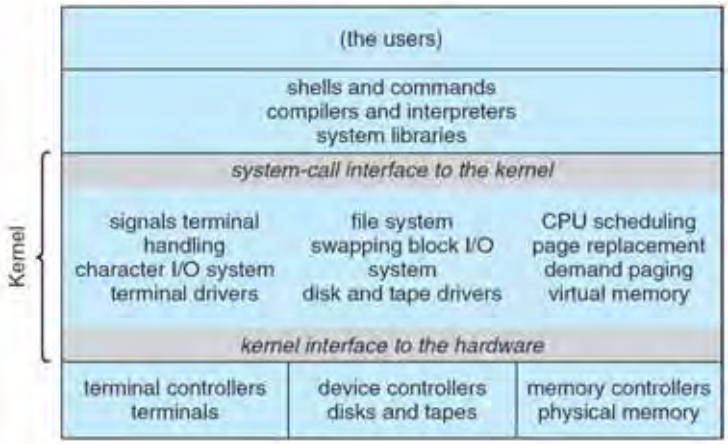
MS-DOS Layer Structure



UNIX

- برای سخت افزارها با قابلیت محدود نوشته شده است. سیستم عامل یونیکس اصلی دارای ساختار محدودی بوده است.
- سیستم عامل یونیکس دارای دو بخش مجزا است:
  - برنامه های سیستم ها
  - هسته
    - به هر چیز موجود در زیر، واسط فراخوانی سیستمی و بالای سخت افزار فیزیکی، هسته گفته می شود
    - هسته، سیستم فایل زمانبندی CPUو حافظه را فراهم می آورد و همچنین تعداد زیادی تابع برای یک لایه

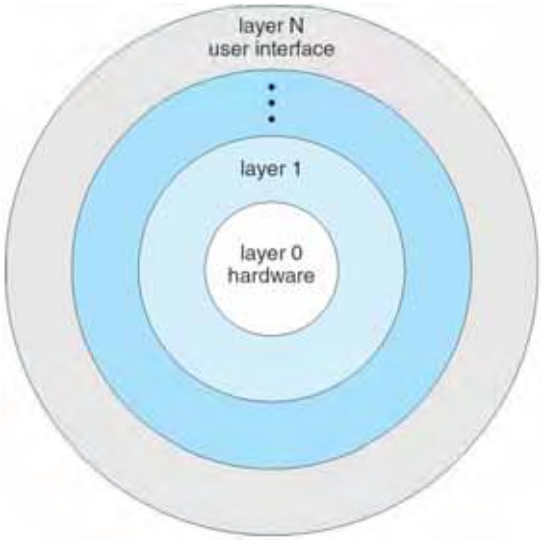
Traditional UNIX System Structure



راهبرد لایه ای

- در این راهبرد سیستم عامل به چند لایه تقسیم میشود و هر لایه بر روی لایه های پایینی قرار میگیرد.
  - پایین ترین لایه یا لایه صفر سخت افزار است
  - بالاترین لایه یا لایه N واسط کاربر است
- مزیت ها: سادگی ساخت و اشکالزدایی
  - با ماژول سازی، هرلایه فقط از توابع و سرویس های فراهم شده به وسیله لایه های سطح پایین تر استفاده می کنند
- مشکل عمده :تعریف لایه ها به طور مناسب
  - تعریف نامناسب موجب کاهش شدید کارآیی می شود.

Layered Operating System



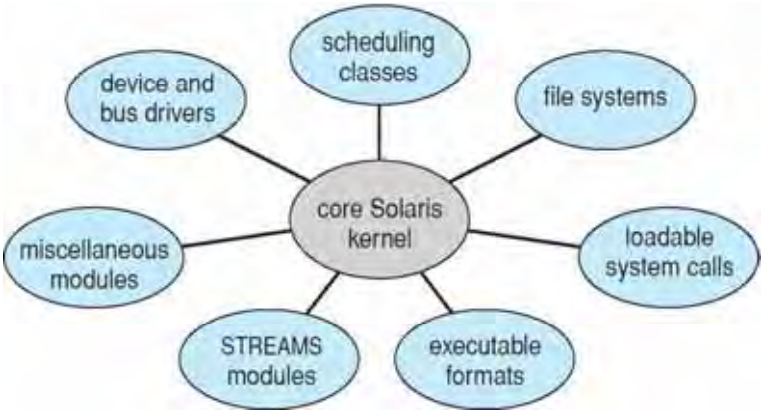
ساختار ریز هسته ها یا ریز کرنل ها

- قطعات اضافی و غیر اساسی از هسته حذف شده و پیاده سازی آن به صورت برنامه های سطح کاربر و سیستمی، سازمان دهی می شود.
- برای ارتباط میان ماژول های کاربر ارسال پیام *message passing* است.
- مزیت ها
  - آسانتر شدن گسترش ریز هسته
  - راحت شدن انتقال سیستم عامل به معماری های جدید
  - افزایش اطمینان(کاهش کد در حال اجرا در مد هسته)
  - امنیت بیشتر
- ضعف ها
  - سربار ارسال پیام از فضای کاربر به فضای هسته به طور مرتب

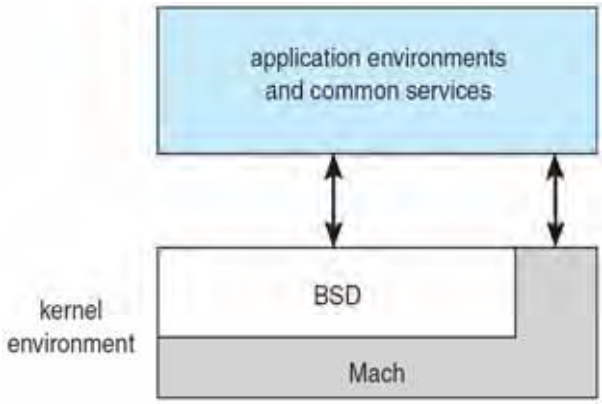
Modules

- بیشتر سیستم عاملهای مدرن دارای هسته ماژولار هستند
  - در آن ها از رهیافت برنامه نویسی شی گرا استفاده میشود
  - هسته از قطعه های جداگانه ساخته شده است
  - ماژولها به طور پویا قابل بارکردن هستند
- شبیه سیستم لایه ای است اما انعطاف پذیری در آن بیشتر است
  - هر ماژول می تواند ماژول های دیگر را فراخوانی کند
- شبیه ریز هسته ها کار می کنند اما کارآمدیی بیشتری دارد
  - هیچ پیامی در بین ماژولها ارسال نمیشود

Solaris Modular Approach



Mac OS X Structure



- ساختار ترکیبی- ترکیب ساختار لایه ای با ریز هسته

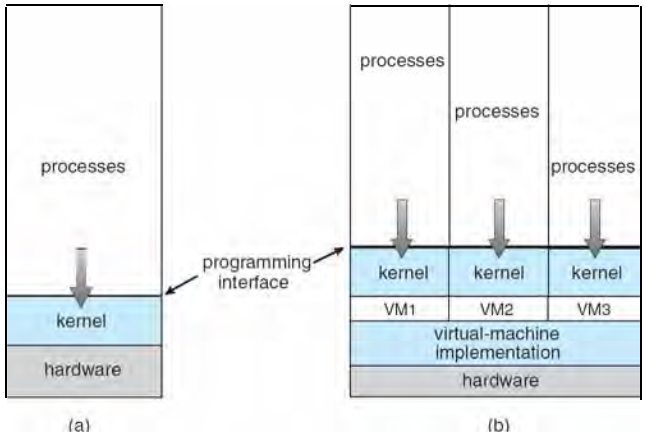
ماشین مجازی

- روش لایه از نتیجه منطقی در مفهوم ماشین مجازی به دست آمده
  - ماشین مجازی سخت افزار را به چند قسمت تقسیم می کند به گونه ای که هسته سیستم عامل تصور می کند تمام سخت افزار را در اختیار دارد.
  - ماشین مجازی واسطی را ایجاد می کند که هم ارز سخت افزار عریان است.
- بخشی از ماشین مجازی که میزبان سیستم عامل است این تصور را ایجاد میکند که هر پردازش پردازنده و حافظه خود را دارد و صاحب آن ها می باشد.
- در ماشین مجازی برای هر فرآیند میهمان یک کپی از فرآیند مورد نظر فراهم میشود

تاریخچه و مزیت های ماشین مجازی

- ماشین مجازی، ابتدا از طریق سیستم عامل VM در سال ۱۹۷۲ ، در کامپیوترهای بزرگ IBM مطرح شد.
- مزیت ها :
  - با یک سخت افزار میتوانیم چندین محیط (سیستم عامل های مختلف) را با هم اجرا کنیم.
  - حفاظت سیستم های میزبان از یکدیگر
  - امکان به اشتراک گذاشتن منابعی مانند فایل ها که به دو روش می توان این کار را انجام داد
    - از طریق به اشتراک گذاشتن ولوم یا مکان ذخیره سازی
    - از طریق ایجاد ارتباط مجازی (شبکه مجازی)
  - برقراری ارتباط از طریق شبکه ای از کامپیوترهای مجازی
  - مناسب بودن برای تست و توسعه سیستم عامل ها
  - بسیاری از سیستم ها با کاربرد کمتر می توانند با هم ادغام شوند تا یک سیستم پر کاربرد را ایجاد کنند
- "Open Virtual Machine Format": شکل استانداردی از ماشین مجازی است
  - استاندارد سازی باعث می شود بتوان بر روی قسمت های مختلف ماشین مجازی سیستم های مختلف را اجرا کرد

Virtual Machines (Cont)

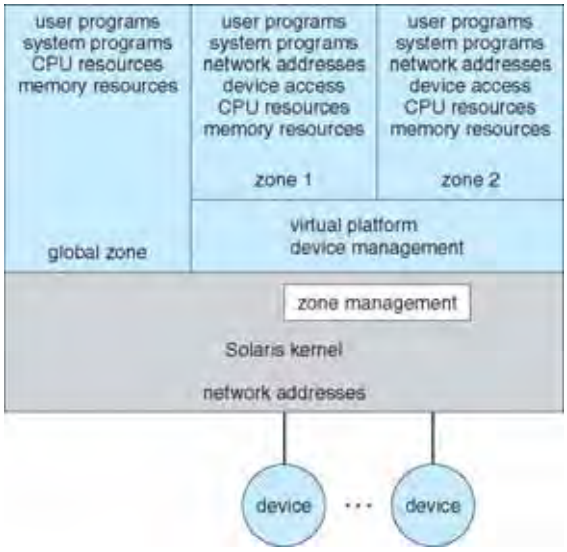


(a) Nonvirtual machine (b) virtual machine

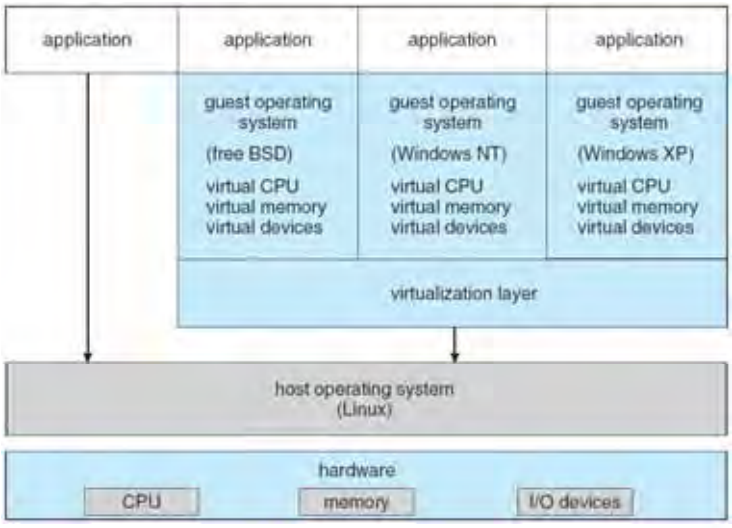
پارا مجازی سازی (مجازی سازی جزئی)

- در این روش جای این که کاری کنیم که سیستم عامل(مهمان) فکر کند که سیستمی در اختیارش قرار دارد، پارامجازی ساز، سیستمی را فراهم می کند که شبیه سیستم مورد انتظار است
- مهمان باید مطابق سخت افزاری که پارامجازی ساز ارائه می دهد تغییر کند
- در سولاریس ۱۰ یک لایه مجازی بین سیستم عامل و برنامه های کاربردی قرار میگیر در این سیستم ها فقط یک هسته نصب می شود و سخت افزار مجازی نمیشود و در عمل این هسته است که به صورت مجازی تبدیل می شود.

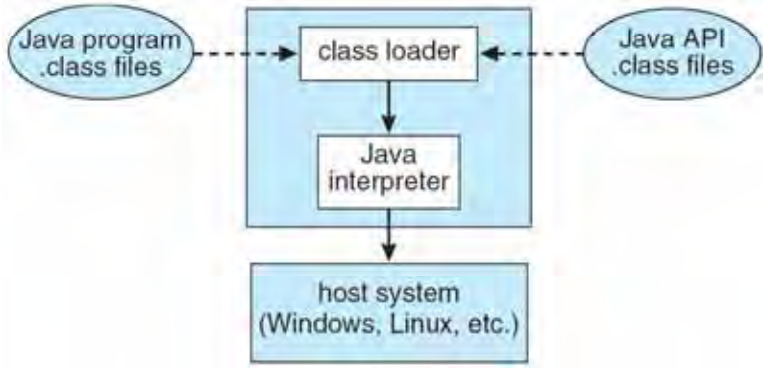
Solaris 10 with Two Containers



VMware Architecture



The Java Virtual Machine



اشکالزدایی سیستم عامل

- اشکالزدایی فعالیت یافتن و تصحیح خطا یا عیب ها است
  - سیستم عامل ها یک لاگ فایل(فایل کارنامه)log files تهیه می کنند که در آن اطلاعاتی درباره خطاها رخ داده وجود دارد
  - هنگامی که یک فرایند با مشکل مواجعه شود می توان تصویری از حافظه متعلق به پردازش ذخیره کرد (core dump)
  - خرابی هسته، فروپاشیcrash نام دارد. همانند شکست فرایند، در این جا نیز اطلاعات خطا در فایل کارنامه دخیره میشود و حالت حافظه در روبرداری فروپاشیcrash dump ذخیره می گردد
- تنظیم کارآیی می تواند عملکرد سیستم را بهبود ببخشد.

قانون کرنیگان: اشکالزدایی دو برابر سخت از نوشتن کد است. بنابراین، اگر کد را با هوشمندی هر چه بیشتر بنویسید، بنا به تعریف، به اندازه کافی باهوش نیستید که ئن را اشکال زدایی کنید.

- DTrace ابزاری است که در سیستم عامل هایی مانند Solaris, FreeBSD, Mac OS X وجود دارد و اجازه سنجش زنده سیستم های تولیدی را می دهد
  - کاوشگر زمانی که کدی اجرا می شود تصویری از وضعیت داده های آن تهیه می کند و در اختیار ما قرار می دهد.

Solaris 10 dtrace Following System Call

```
# ./all.d 'pgrep xclock' XEventsQueued
dtrace: script './all.d' matched 52377 probes
CPU FUNCTION
  0 -> XEventsQueued                      U
  0   -> _XEventsQueued                   U
  0     -> _X11TransBytesReadable         U
  0     <- _X11TransBytesReadable         U
  0     -> _X11TransSocketBytesReadable   U
  0     <- _X11TransSocketBytesreadable   U
  0     -> ioctl                          U
  0       -> ioctl                        K
  0         -> getf                       K
  0           -> set_active_fd            K
  0           <- set_active_fd            K
  0         <- getf                       K
  0         -> get_udatamodel             K
  0         <- get_udatamodel             K
...
  0         -> releasef                   K
  0           -> clear_active_fd          K
  0           <- clear_active_fd          K
  0           -> cv_broadcast             K
  0           <- cv_broadcast             K
  0         <- releasef                   K
  0       <- ioctl                        K
  0     <- ioctl                          U
  0   <- _XEventsQueued                   U
  0 <- XEventsQueued                      U
```

خروجی Dtrace در سولاریس ۱۰ که فراخوانی های سیستمی را در هسته ردیابی می کند

تولید سیستم عامل

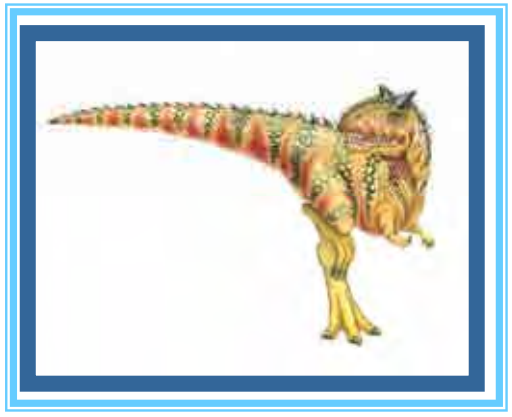
- به طور معمول سیستم عامل ها به گونه ای طراحی میشوند که بر روی سیستم ها گوناگون با پیکربندی های مختلف اجرا شوند
  - سیستم باید بتواند بر روی هر کامپیوتر با هر پیکربندی اجرا شود. سپس سیستم باید برای کامپیوترهایی با پیکربندی خاص تولید شود. این فرایند را تولید سیستم (SYSGEN) گویند
- راه اندازی سیستم (*Booting*)- شروع به کار کامپیوتر به وسیله بار کردن هسته
- برنامه راه انداز (*Bootstrap program*)- کدی است که در ROM ذخیره میشود و توانایی پیدا کردن و در حافظه قرار داده هسته را دارد آن همچنین هسته را نیز اجرا می کند

راه انداز سیستم System Boot

- سیستم عامل باید در دسترس سخت افزار ساخته شود تا سخت افزار بتواند آن را راه انداز کنند
  - بخش کوچکی از کد به نام بارکننده راه انداز در هسته قرار دارد، هسته را در حافظه اصلی بار می کند و اجرای آن را آغاز می نماید.
  - بعضی از سیستم های کامپیوتری، مثل PC ها، از فرآیند دو مرحله ای استفاده می کنند که در آن، یک بار کننده ی راه انداز، برنامه ی راه انداز پیچیده تری را از دیسک بار می کند، که آن نیز به نوبه ی خود، هسته را بار می کند.
    - میان افزار ها برای نگهداری کد راه انداز اولیه استفاده میشوند boot code

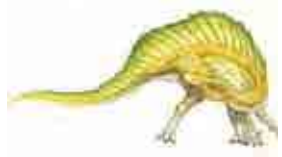
End of Chapter 2

Chapter 3: Processes



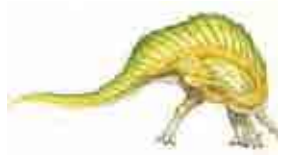
Outline

- مفهوم فرآیند
- زمان بندی فرآیندها
- عملیات بر روی فرآیند ها
- ارتباط بین فرآیندها
- مثال هایی از سیستم های IPC
- ارتباط در سیستم های کلاینت –سرور



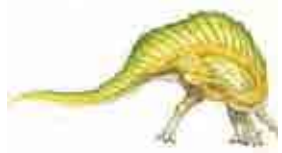
اهداف فصل

- معرفی مفهوم فرایند- یک برنامه در حال اجرا که مبنای تمام محاسبات است.
- توصیف ویژگی های مختلف فرایندها، از جمله زمانبندی، ایجاد و خاتمه و ارتباطات
- توصیف ارتباطات در سیستم های کلاینت - سرور

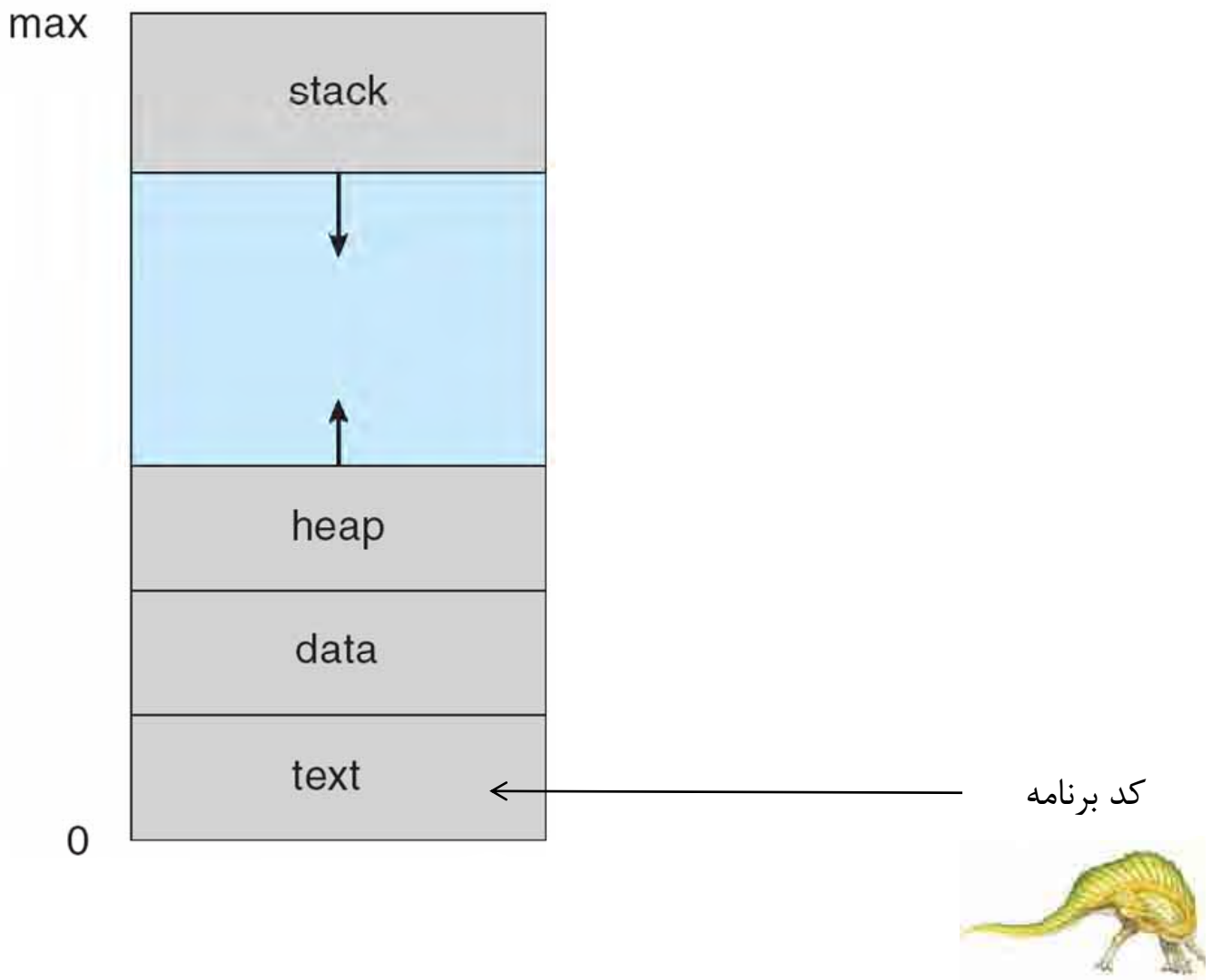


مفهوم فرایند

- انواع سیستم عامل ها، برنامه های مختلفی را اجرا می کند :
  - سیستم های دسته ای کارها را اجرا می کنند
  - سیستم های اشتراک زمانی برنامه های کاربر یا وظیفه ها را اجرا می کنند
  - در این کتاب، غالبا کار و فرایند به جای یکدیگر به کار گرفته می شوند.
- فرایند – یک برنامه در حال اجرا؛ فرایند با یک ترتیب کارهای خود را انجام می دهد تا پایان بپذیرد
  - یک فرایند شامل قسمت های زیر است:
    - شمارنده برنامه: نشان دهنده فعالیت فعلی است
    - پشته:بخش داده های موقت است
    - بخش داده ها: حاوی متغیرهاسراسری می باشد



Process in Memory



حالت فرایند

- هر فرایند ممکن است در یکی از حالت های زیر باشد:
  - جدید (**new**) : فرایند ایجاد می شود
  - **اجرا running** : دستورات در حال اجرا هستند
  - **انتظار waiting**: فرایند منتظر وقوع رویدادی است (مثل کامل شدن I / O)
  - **آماده ready**: فرایند منتظر است تا به یک پردازنده نسبت داده شود
  - **پایان terminated**: اجرای فرایند خاتمه یافته است.

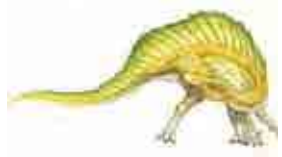
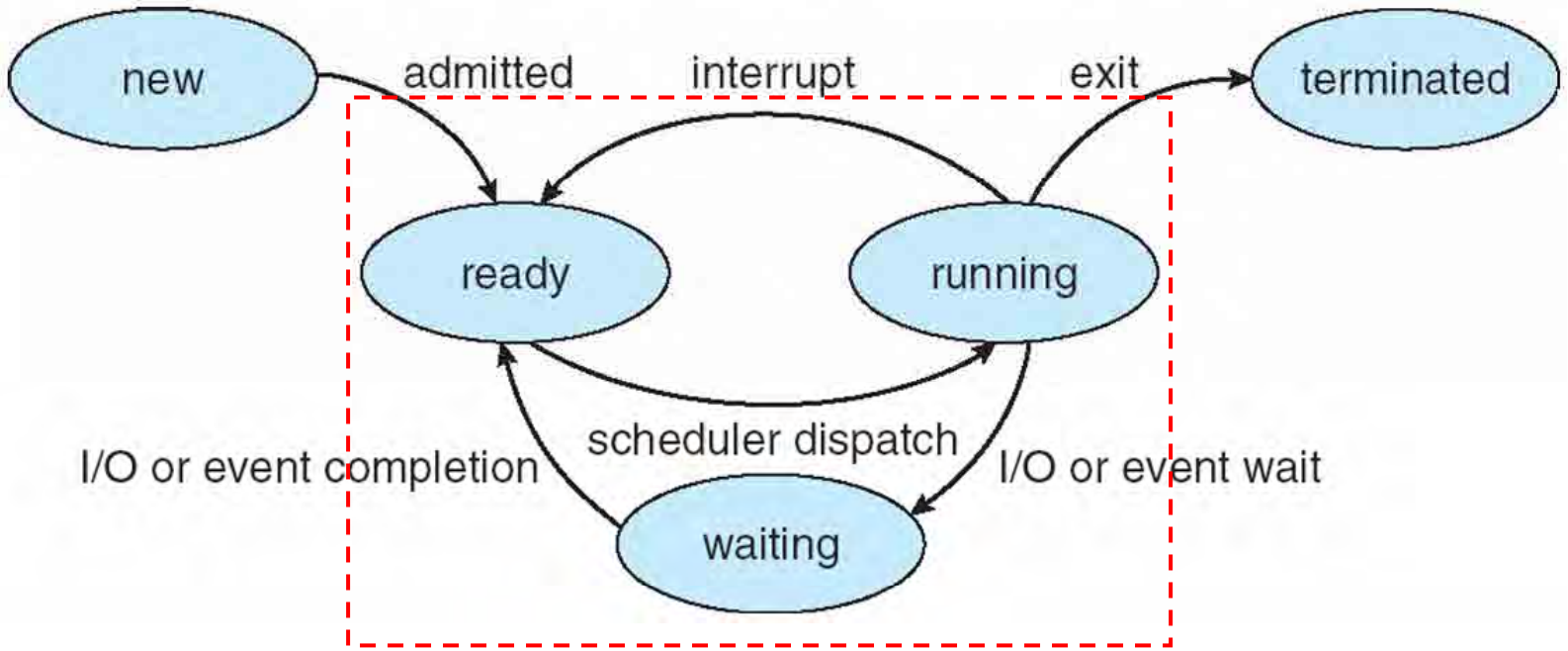
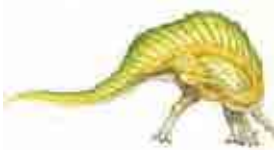


Diagram of Process State

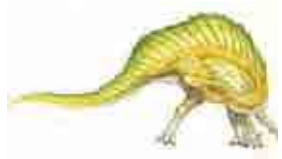




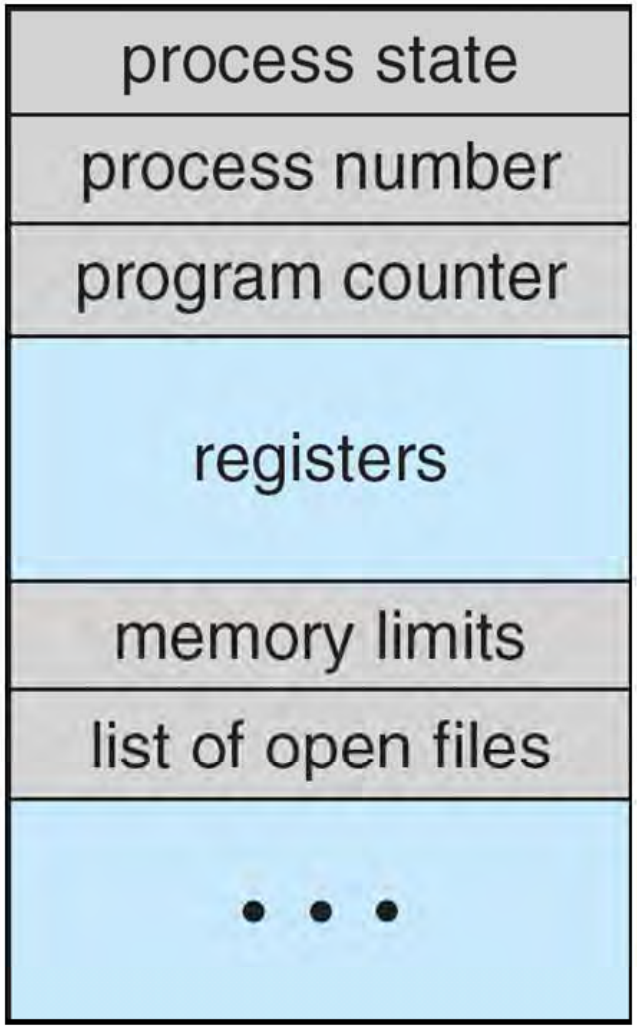
بلوک کنترل فرایند (PCB)

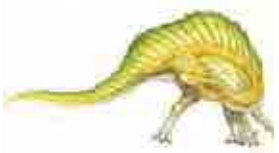
حاوی اطلاعاتی درباره فرایند است

- حالت فرایند
- شمارنده برنامه
- ثبات های CPU
- اطلاعات زمانبندی CPU
- اطلاعات مدیریت حافظه
- اطلاعات حسابرسی
- اطلاعات وضعیت I/O

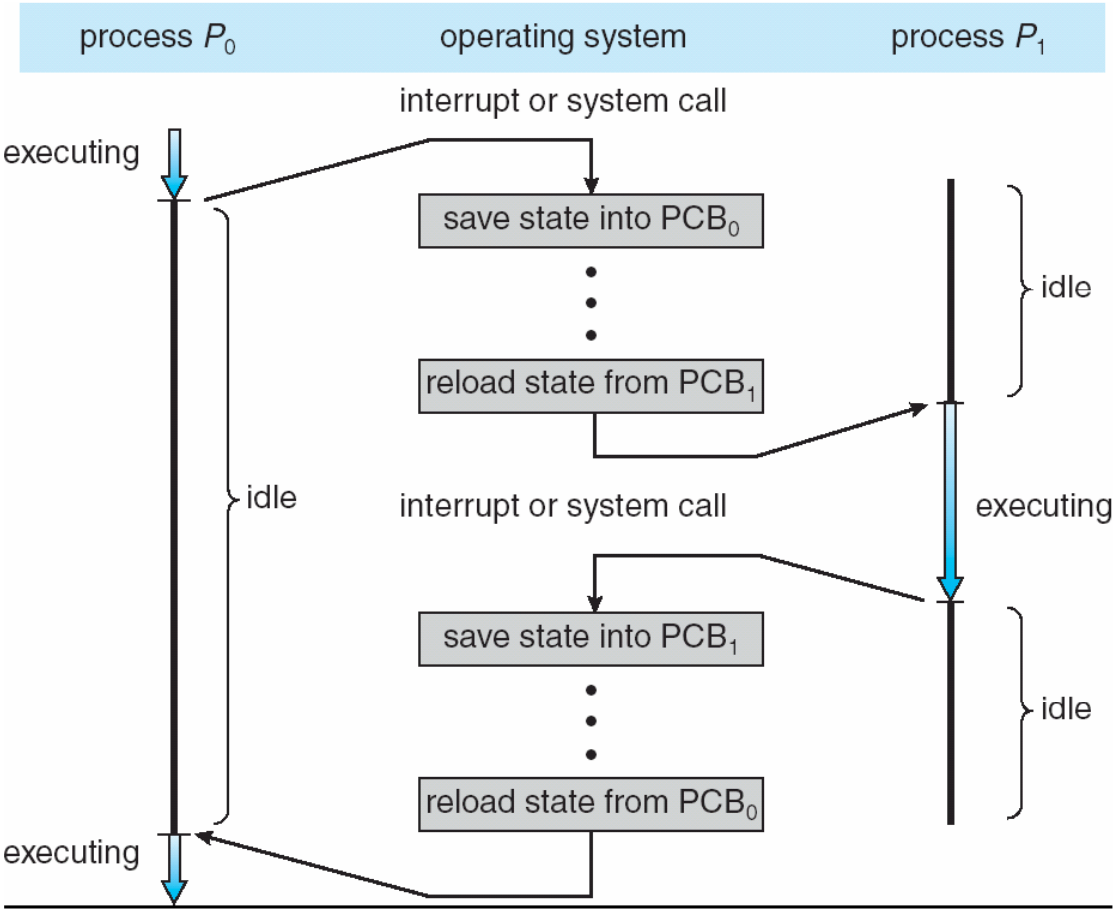


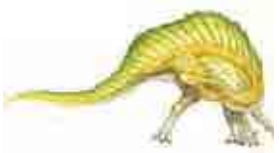
Process Control Block (PCB)





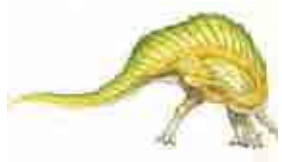
CPU Switch from Process to Process



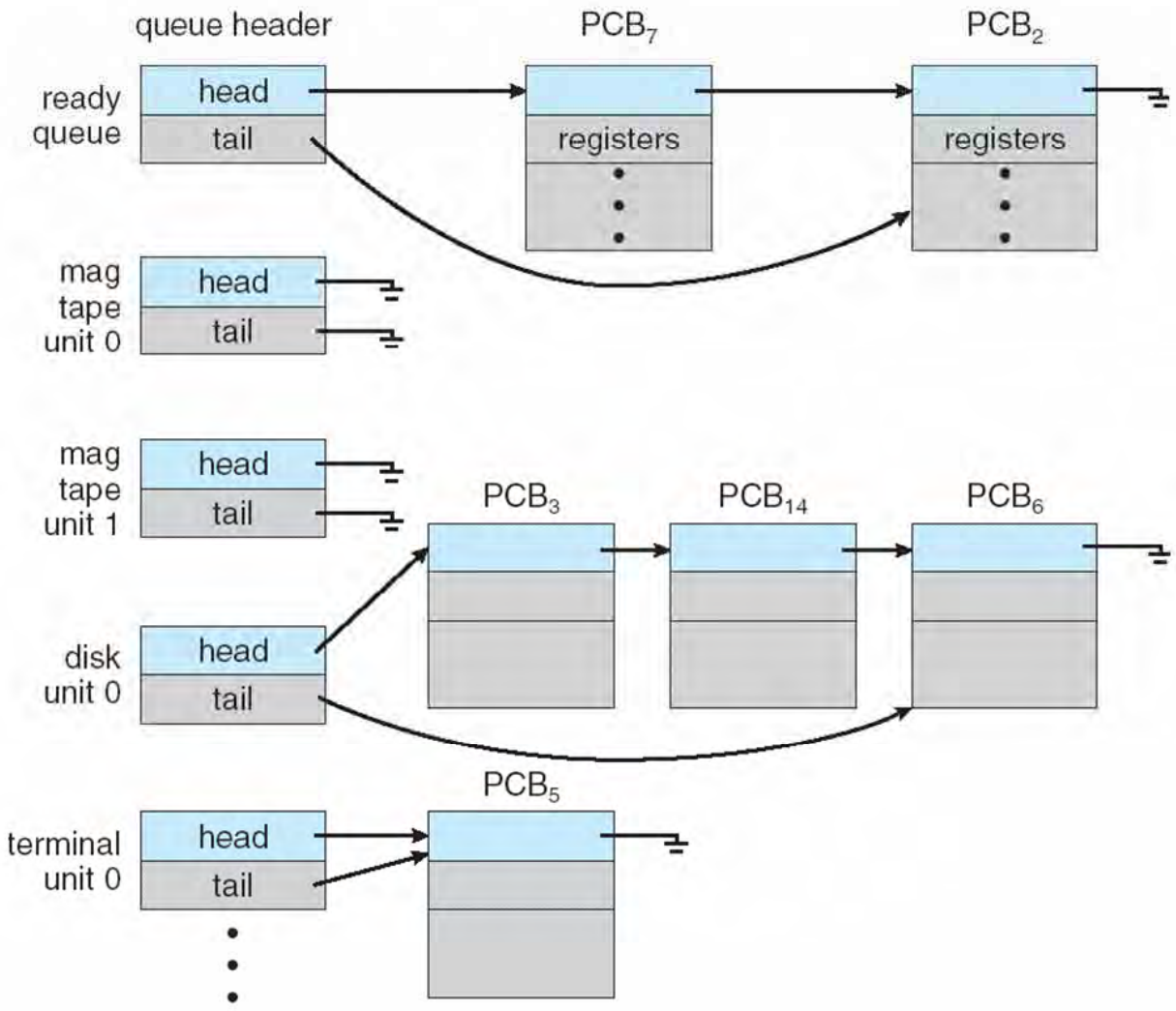


صف های زمانبندی فرآیند

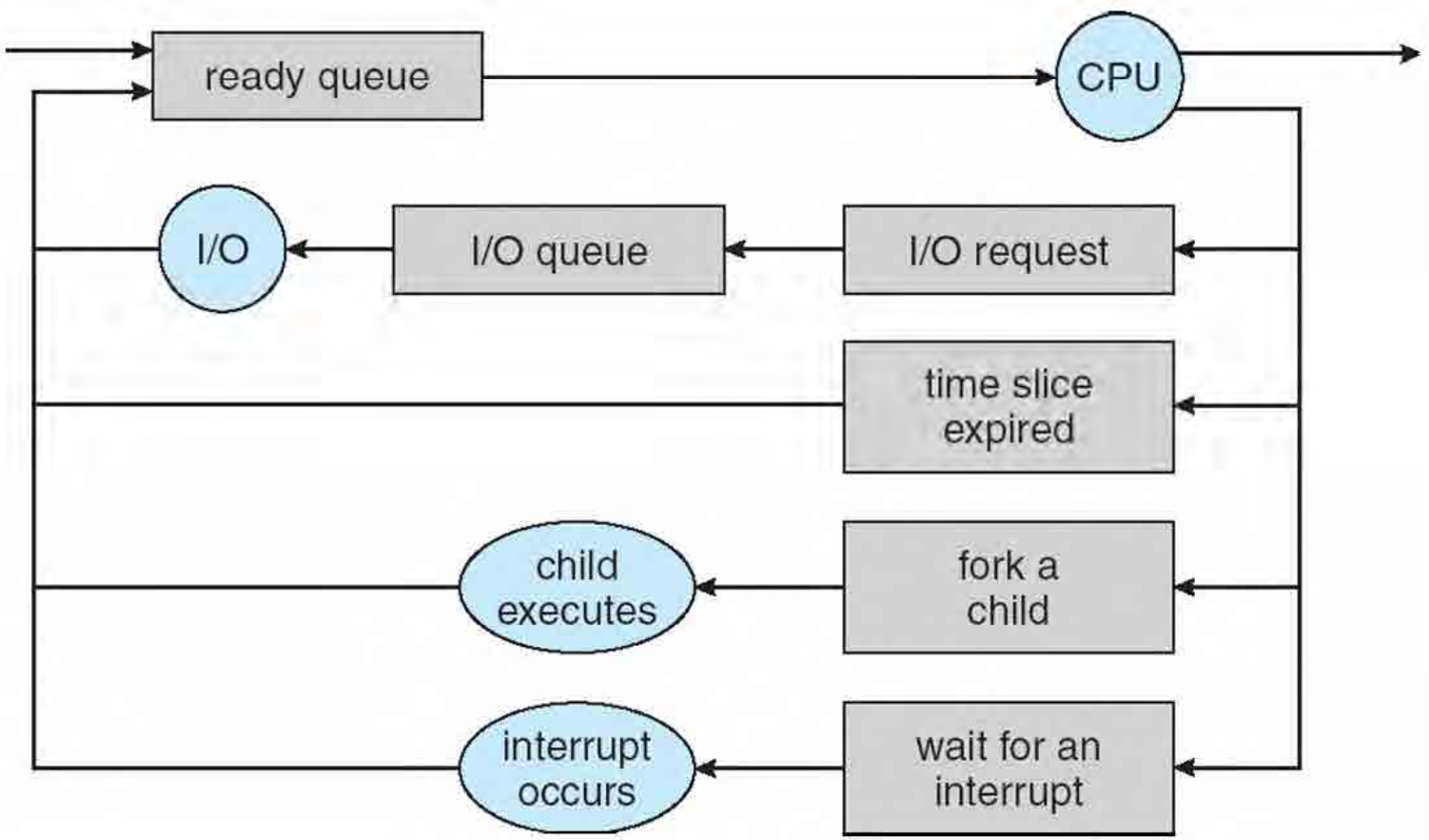
- **صف کار** – مجموعه همه پردازش های سیستم
- **صف آماده** – فرایندهایی که در حافظه قرار می گیرند و آماده و منتظر اجرا هستند
- **صف دستگاه**– لیستی از فرایندهایی که منتظر یک دستگاه I/O خاص هستند
- فرایندها در بین صف های مختلف جابه جا می شوند

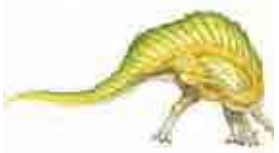


Ready Queue and Various I/O Device Queues



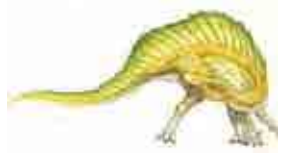
Representation of Process Scheduling



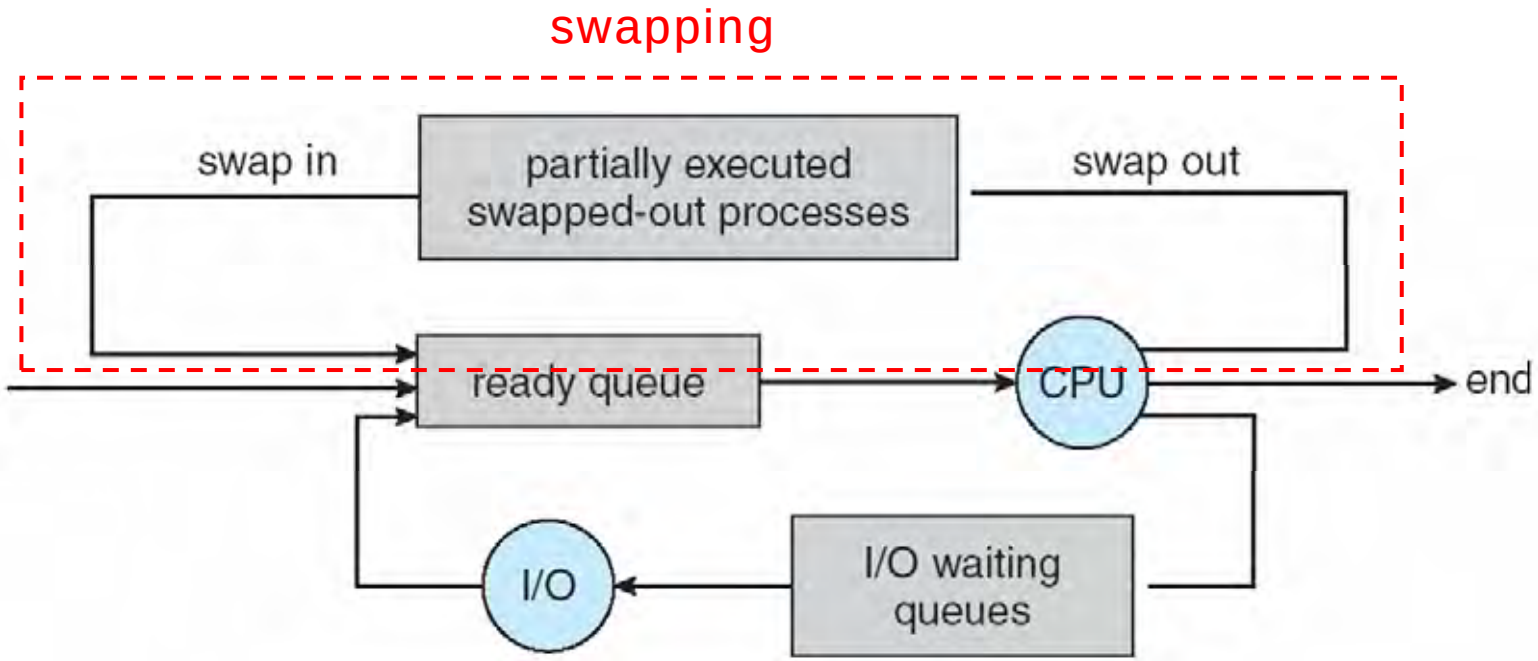


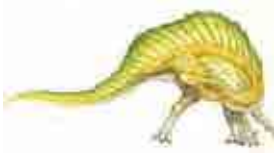
زمان بندها

- **زمانبند بلند مدت(زمانبند کار)**– تعیین می کند کدام پردازش باید در صفحه آماده قرار بگیرد .
- **زمانبند کوتاه مدت (زمانبند CPU)**– تعیین می کند کدام پردازش می تواند پردازنده را در اختیار بگیرد و اجرا شود.



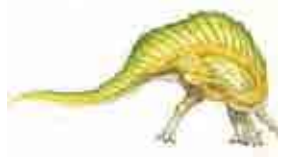
Addition of Medium Term Scheduling





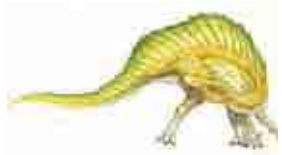
Schedulers (Cont)

- زمانبند کوتاه مدت: باید خیلی سریع باشد و فرکانس آن چند میلی ثانیه است یعنی هر چند میلی ثانیه یک بار اجرا می شود (must be fast) $\Rightarrow$
- زمانبند بلند مدت: در هر دقیقه و یا ثانیه چند بار اجرا میشود البته ممکن است آهسته تر نیز باشد.
  - زمانبند بلند مدت درجه چند برنامه گی را کنترل می کند.
- پردازش ها را می توان با هر یک از دو قسمت زیر توصیف کرد:
  - **مقید به I/O (تنگنای I/O)**– زمان بیشتری صرف I / O می شود تا محاسبات
  - **مقید به CPU (تنگنای CPU)**– فرایند بیشتر وقت خود را صرف انجام محاسبات می کند .



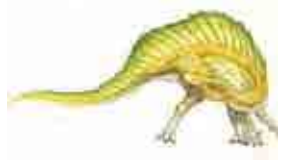
تعویض متن

- در زمان تعویض CPU به فرایند دیگر ، باید وضعیت فرایند فعلی دخیره شده و وضعیت فرایند جدید بارگزاری شود که به این کار تعویض بستر یا تعویض متن گویند.
- وقتی تعویض متن صورت میگیرد حالت پردازش در PCB بلاک کنترل پردازش ذخیره میشود
- زمان تعویض متن زمان سربار است و سیستم نمی تواند در این زمان کار مفیدی انجام دهد
- مدت زمان لازم برای تعویض متن به سخت افزار بستگی دارد.



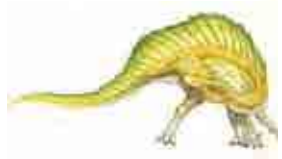
ایجاد فرایند

- **فرآیندی که فرایند های جدید را ایجاد می کند، فرایند والد، و فرایندهای جدید، فرزندان آن فرایند نامیده می شوند. هر یک از این فرایندهای جدید نیز ممکن است فرایندهایی را به وجود آورند و درختی از فرایندها را تشکیل دهند.**
  - به طور معمول فرایند ها با شناسه فرایند(pid)، شناسایی و مدیریت میشوند
- اشتراک منابع
  - والد و فرزندان همه منابع را به اشترک میگذارند
  - فرزند از زیر مجموعه ای از منابع اشتراکی والد استفاده می کند
  - والد و فرزندان هیچ منبع اشتراکی ندارند.
- روند اجرا
  - والد و فرزندان به صورت همروند اجرا میشوند
  - والد منتظر می ماند تا فرزند به کارش پایان دهد

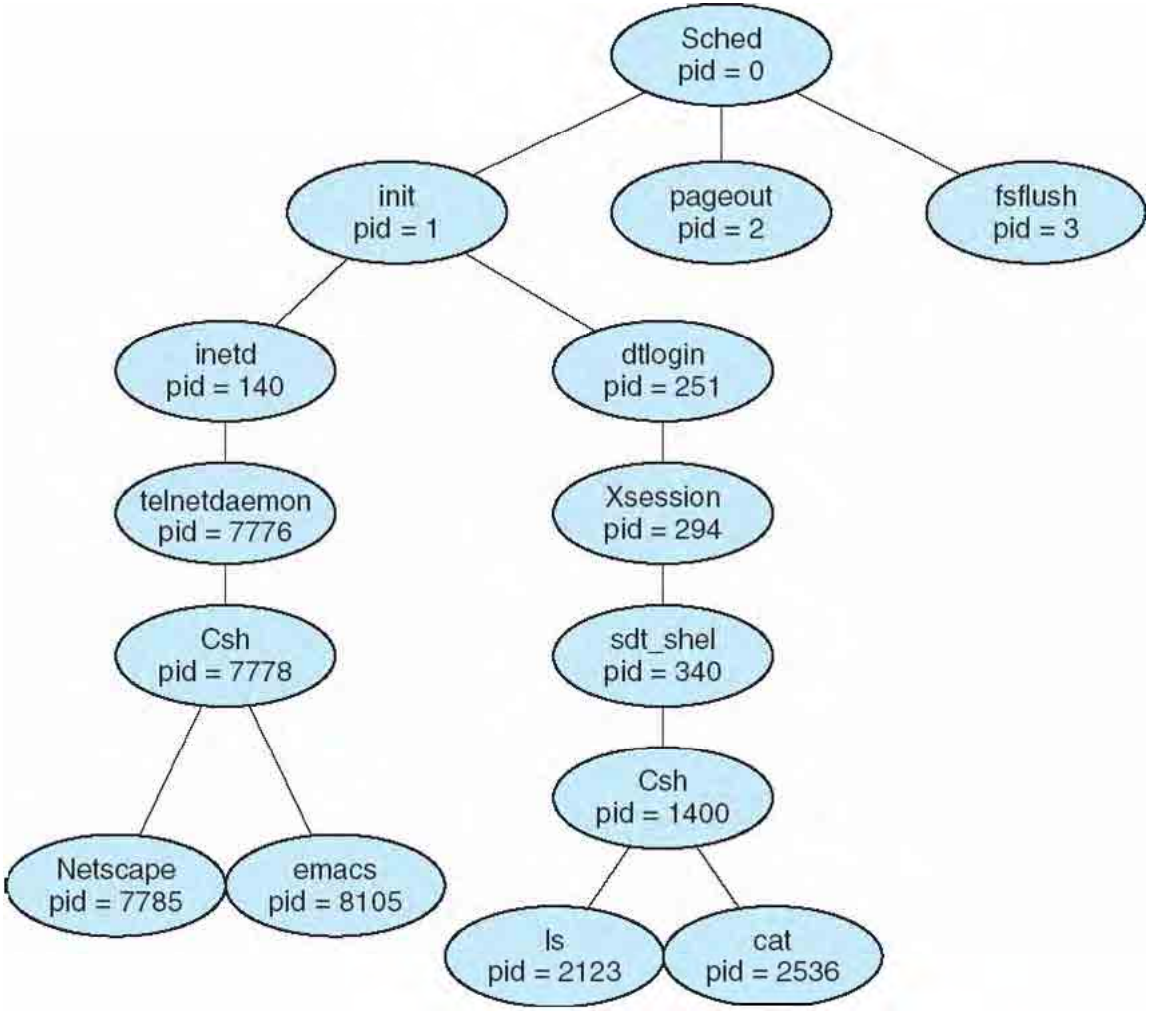


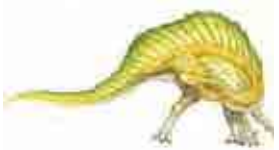
Process Creation (Cont)

- فضای آدرس
  - فرایند فرزند، یک کپی از فرایند والد است (داده ها و برنامه های یکسان با والد دارد)
  - فرایند فرزند، برنامه را در فضای آدرس خود بار کرده است.
- مثال هایی از دستورات UNIX
  - **fork()** فراخوانی سیستمی برای ایجاد پردازش **fork()** system call creates new process
  - **معمولا پس از فراخوانی fork() ، فراخوانی سیستمی exec()** توسط یکی از دو فرایند اجرا می شود تا فضای حافظه فرایند را با برنامهی جدید جایگزین کند.



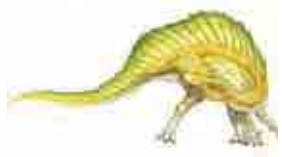
A Tree of Processes on a Typical Solaris





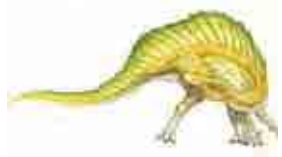
پایان فرایند

- فرایند وقتی پایان می یابد که آخرین دستورش اجرا شده باشد و با فراخوان سیستم exit() از سیستم عامل می خواهد که آن را حذف کند.
  - در آن نقطه داده از فرایند فرزند به والد انتقال داده میشود (با استفاده از wait)
  - تمام منابع فرایند توسط سیستم عامل پس گرفته میشوند.
- فرایند ممکن است اجرای فرایندهای فرزند را پایان دهد را به دلایل زیر پایان دهد:
  - فرزند، از منابعی بیش از آنچه که در اختیارش قرار گرفته است استفاده می کند
  - وظیفه ای که بر عهده ی فرزند گذاشته شد، دیگر ضروری نیست
  - اگر والد پایان یابد بعضی از سیستم عامل ها اجازه نمی دهند فرایند فرزند ادامه یابد If parent is exiting
    - در این سیستم ها اگر والد خاتمه یابد آن گاه تمام فرزندان آن نیز باید خاتمه یابد. این پدیده را پایان آبشاری گویند

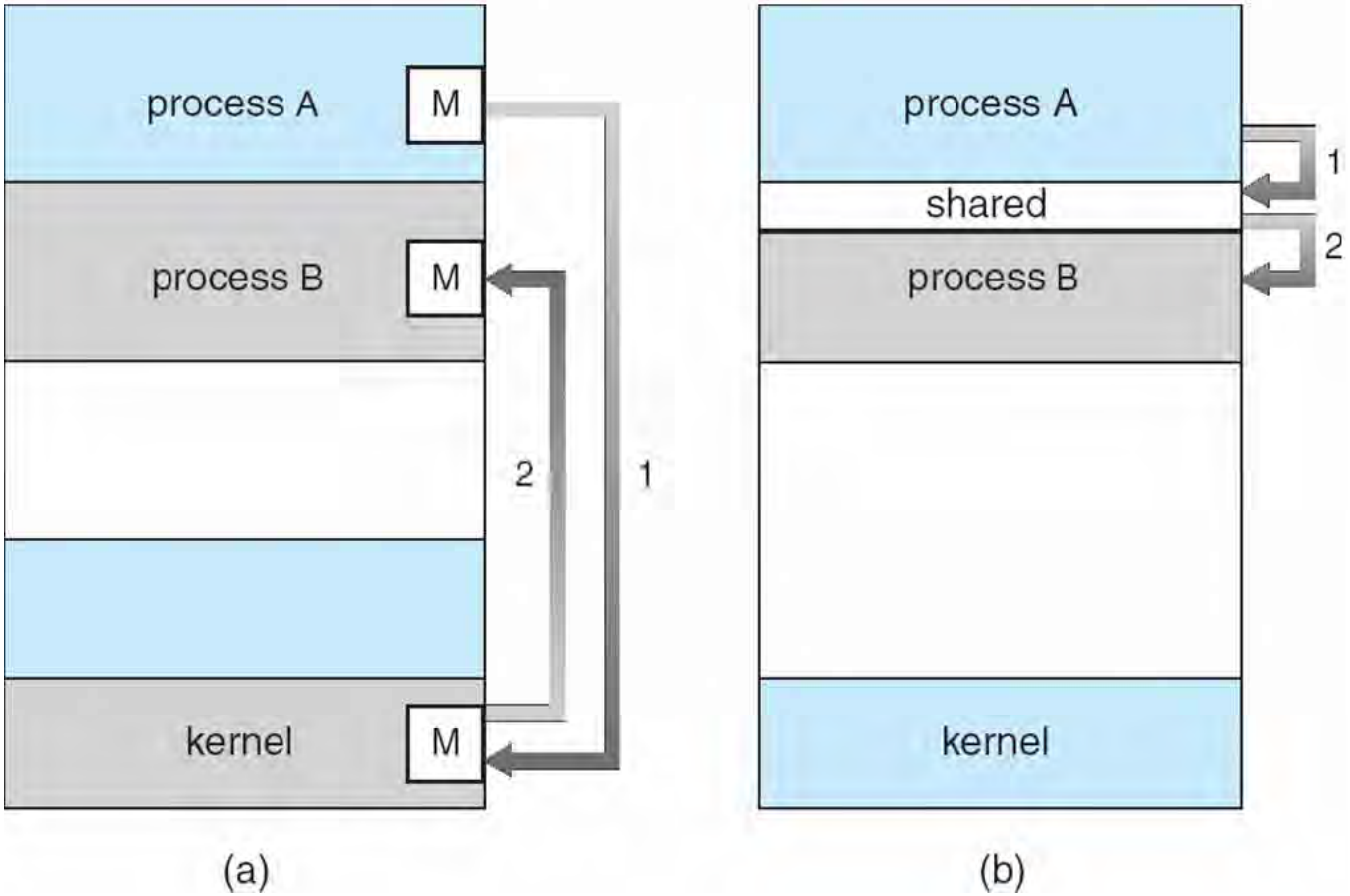


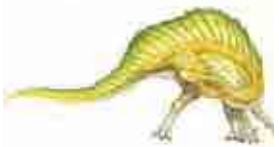
ارتباط بین فرایندها

- فرایندهایی که به طور همزمان در سیستم عامل اجرا می شوند، ممکن است فرایندهای مستقل یا فرایندهای همکار باشند.
  - فرایند در صورتی همکار است که در فرایندهای دیگر موجود در سیستم تأثیر بگذارد یا از آن ها تاثیر پذیرد. مانند فرایندهایی که داده مشترک دارند
  - فرایندهای همکار ،نیازمند یک راهکار **ارتباط بین فرایندها** (IPC) هستند که به آن ها اجازه می دهد داده ها و اطلاعات را مبادله کنند.
- دو مدل از IPC عبارتند از:
  - حافظه ی مشترک
  - مبادله پیام



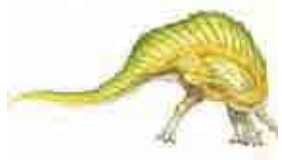
Communications Models





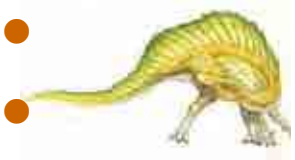
پردازش های همکار

- **پردازش ها مستقل** نمی توانند بر دیگر پردازش ها اثر بگذارند و یا از آن ها تأثیر بپذیرند.
- **پردازش های همکار** می توانند بر اجرای دیگر پردازش ها اثر بگذارند و یا از آن ها تأثیر بپذیرند.
- مزیت پردازش های همکار
  - اشتراک اطلاعات
  - تسریع محاسبات
  - پیمانه ای (ماژولار)
  - سهولت



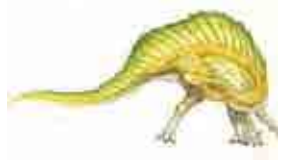
سیستم های مبادله پیام

- مکانیزمی برای فرایندها فراهم می آورد تا با یکدیگر در ارتباط باشند و کارهای خود را با هم هماهنگ کنند.
- مبادله پیام، راهکاری را فراهم می کند که اجازه می دهد فرایندها بدون به اشتراک گذاشتن فضای آدرس با هم در ارتباط باشند
- روش مبادله پیام، حداقل دو عملیات را فراهم می سازد
  - **send**(*message*)- اندازه پیام های ازسال شده توسط فراین میتواند ثابت یا متغیر باشد
  - **receive**(*message*)
- اگر فرآیندهای P وQ بخواهند با یکدیگر ارتباط بر قرار کنند، If *P* and *Q* wish to communicate, they need to:
  - باید یک پیوند ارتباطی بین آن ها برقرار باشد. establish a *communication link* between them
  - پیام ها را به وسیله send و receive مبادله می کنند.
- پیاده سازی پیوند ارتباطی
  - پیاده سازی فیزیکی (مثل حافظه ی مشترک، گذرگاه سخت افزاری یا شبکه)
  - پیاده سازی منطقی از طریق ویژگی های منطقی



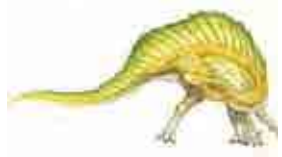
ارتباط مستقیم

- در ارتباط مستفیم، هر فرایندی که می خواهد ارتباط برقرار کند باید صریحا نام گیرنده یا فرستنده ی ارتباط را ذکر نماید:
  - **send**(*P*, *message*)- ارسال پیام به پردازش P
  - **receive**(*Q*, *message*)- دریافت پیام از پردازش Q
- خواص پیوند ارتباطی
  - پیوند به صورت خودکار برقرار میشود این دو فرایند باید هویت یکدیگر را بدانند.
  - هر پیوند دقیقا به دو فرایند وابسته است.
  - بین هر جفت از فرایندها فقط یک پیوند وجود دارد



ارتباط غیر مستقیم

- پیام ها از طریق صندوق های پستی یا پورت ها ارسال و دریافت میشوند
  - هر صندوق پستی یک id منحصر به فرد دارد
  - دو فرایند تنها در صورتی می توانند ارتباط برقرار کنند که صندوق پستی مشترکی داشته باشند.
- این پیوند ارتباطی دارای خواص زیر است
  - وقتی پیوند بین دو فرایند بر قرار می شود که صندوق پستی مشترکی داشته باشند
  - یک پیوند ممکن است به بیش از دو فرایند مربوط شود
  - بین هر جفت از فرایندهایی که با هم ارتباط بر قرار میکنند ممکن است چندین پیوند وجود داشته باشدبه طوری که هر پیوند متناظر با یک صندوق پستی است

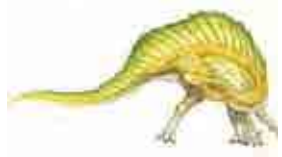


ارتباط غیر مستقیم

- در این ارتباط فرایندها باید بتوانند کارهای زیر را انجام دهند:
  - صندوق پستی جدید ایجاد کنند.
  - پیام ها را از طریق صندوق پستی ارسال و دریافت کنند.
  - صندوق پستی را حذف کند.
- تعریف های اصلی عبارتند از :

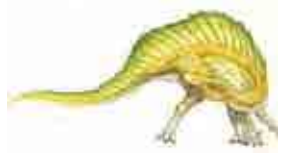
send(*A*, *message*)–ارسال پیام به صندوق پستی A

receive(*A*, *message*)- دریافت پیام از صندوق پستی A



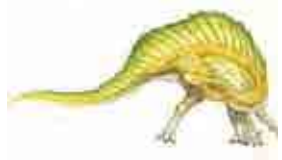
Indirect Communication

- فرض کنید پردازش های P1,P2 و P3 در صندوق پستی A مشترک هستند. فرایند P1 پیامی را به A می فرستد، در حالی که P1 وP2 عملیات receive() را از A اجرا می کنند. کدام فرایند ، پیام ارسال شده توسط P1 را دریافت می کند؟
- راه حل
  - اجازه داده شود که هر پیوند حداکثر به دو فرایند وابسته باشد.
  - اجازه داده شود که در هر زمان حداکثر یک فرایند عملیات receive() را اجرا کند
  - اجازه داده شود که سیستم به دلخواه انتخاب کند کدام فرایند پیام را دریافت کند
    - فرستنده تعیین کند چه کسی پیام را دریافت کند



همگام سازی

- مبادله پیام ممکن است انسدادی و یا غیرانسدادی باشد
- **انسدادی ، همگام** نیز خوانده می شود
  - **در ارسال انسدادی (مسدود کننده):** فرایند فرستنده مسدود میشود تا پیام توسط فرایند گیرنده یا توسط صندوق پستی دریافت شود.
  - **دریافت انسدادی (مسدود کننده): گیرنده مسدود می شود تا زمانی که پیامی در دسترس است.**
- **غیرانسدادی ، ناهمگام** نیز خوانده می شود
  - **ارسال غیر انسدادی:** فرایند فرستند پیام را می فرستد و عملیات را از سر می گیرد .
  - **دریافت غیرانسدادی:** گیرنده یک پیام معتبر و یا تهی را دریافت می کند



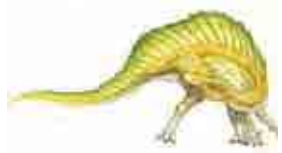
Buffering میانگیر

- ارتباط چه مستقیم باشد و چه غیر مستقیم باشد، پیام های مبادله شده توسط فرایند ها در یک صف موقت قرار می گیرند
- چنین صف هایی به سه روش پیاده میشوند

1. ظرفیت صفر- طول صف صفر است و هیچ پیامی نمی تواند در آن منتظر بماند. فرستنده باید منتظر بماند تا گیرنده پیام را دریافت کند.

2. با ظرفیت محدود- طول صف محدود است. اگر پیوند پر باشد فرستنده باید منتظر بماند تا محل خالی در صف پیدا شود.

3. با ظرفیت نامحدود-طول صف نامحدود است و فرستنده هیچ گاه منتظر نمی ماند



Examples of IPC Systems - Mach

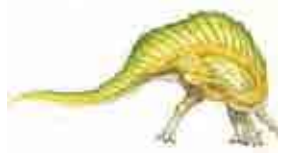
- Mach: سیستم عامل مبتنی بر پیام است
  - حتی فراخوانی های سیستمی با پیام انجام می شود
  - وقتی کاری ایجاد می شود دو صندوق پستی خاص-صندوق پستی Kernelو صندوق پستی Notify نیز ایجاد می شوند.
  - تنها سه فراخوانی سیستمی برای انتقال اطلاعات لازم است

`msg_send()` (پیامی را می فرستد و منتظر می ماند تا دقیقا یک پیام از فرستنده برگردد)
`msg_receive()`,

`msg_rpc()`

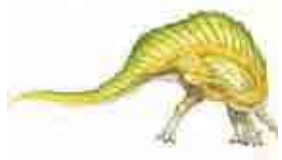
  - صندوق پستی مورد نیاز برای ارتباط با فراخوانی زیر ایجاد می شود

`port_allocate()`

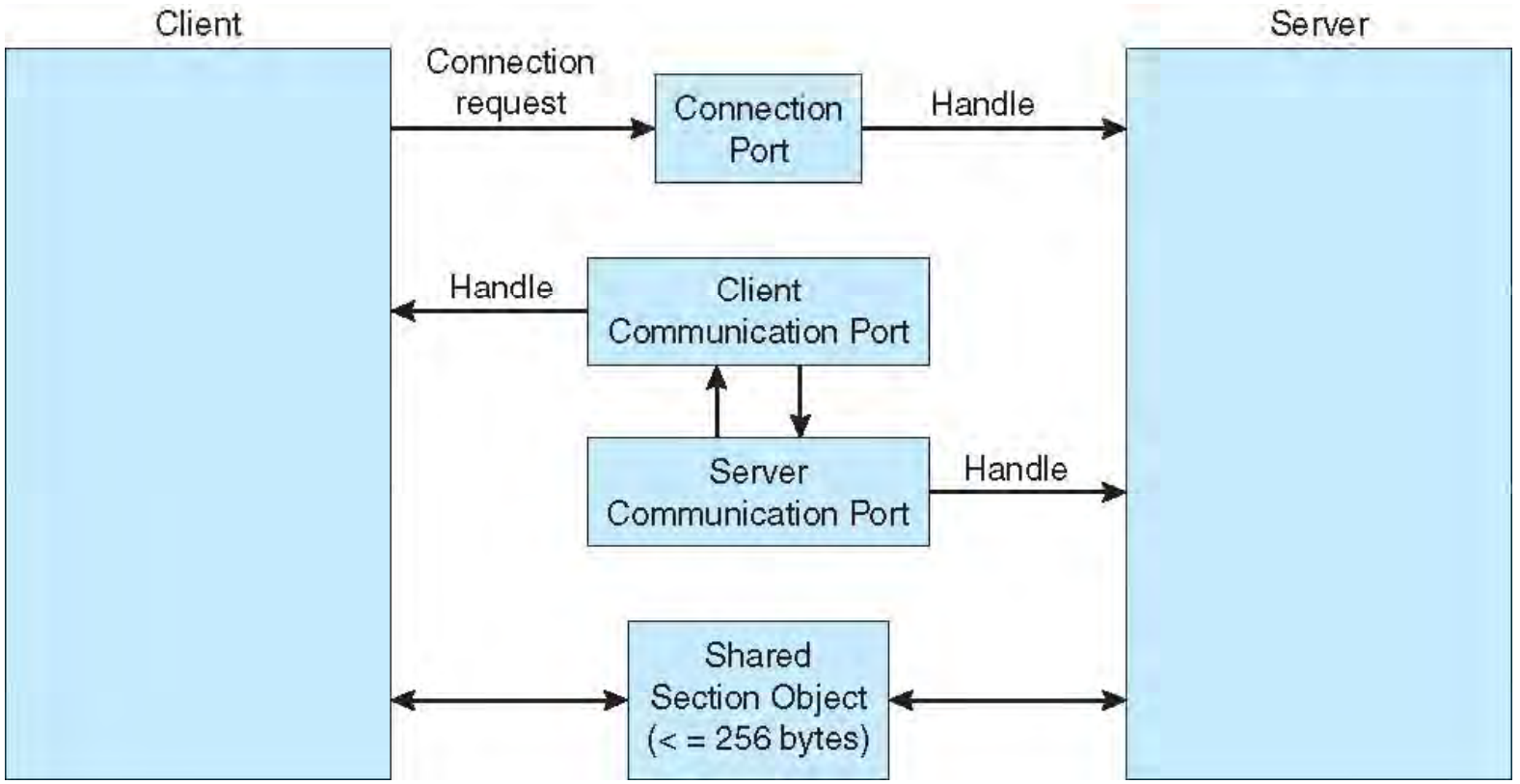


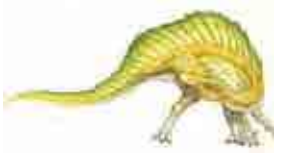
Examples of IPC Systems – Windows XP

- مبادله پیام در ویندوز XP، امکان فراخوانی رویه ی محلی(LPC) نامیده میشود
  - LPC بین دو فرایندها در سیستم های مشابه ارتباط بر قرار می کند.
  - برای برقراری و نگهداری اطلاعات از شئ پرت (مشابه صندوق پستی) استفاده می کند.
  - کارهایی که در برقراری ارتباط انجام میشود عبارت است از:
    - کلاینت، دستگیره ای به شئ پورت اتصالی زیر سیستم را باز می کند.
    - کلاینت، درخواست اتصال را می فرستد.
    - سرور، دو پورت ارتباطی اختصاصی را ایجان می کند و دستگیره ی یکی از آن ها را به کلاینت می فرستد.
    - کلاینت و سرور برای ارسال پیام ها یا callbacks و گوش دادن به پاسخ ها، از پورت متناظر استفاده می کنند.



Local Procedure Calls in Windows XP





End of Chapter 3