دانشگاه کردستان

دانشکده مهندسی

گروه مهندسی کامپیوتر

جزوه درس:

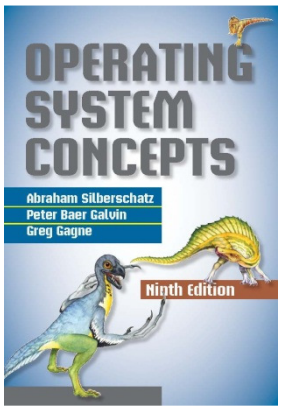
سیستم‌های عامل

تهیه کننده و مدرس:

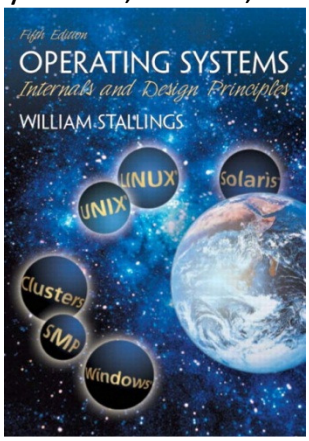
دکتر علیرضا عبداله‌پوری

References:

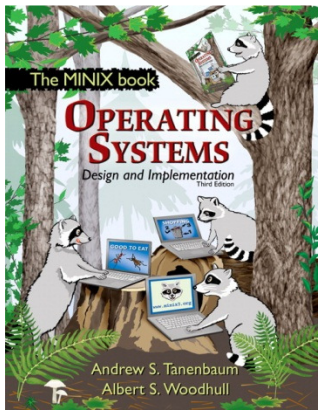
- Abraham Silberschatz, Galvin, and Gagne. “Operating System Concepts”, 9th ed., John Wiley and Sons.



- William Stallings. “Operating Systems”, 4th ed., Prentice Hall, 2001.



- Andrew S. Tanenbaum. “Operating Systems design and implementation”, 3rd ed., Prentice Hall, 2006.



Course Homepage: **http://prof.uok.ac.ir/abdollahpouri/os.html**

۱
۲
۳
۴
۵
۶
۷
۸

فصل اول

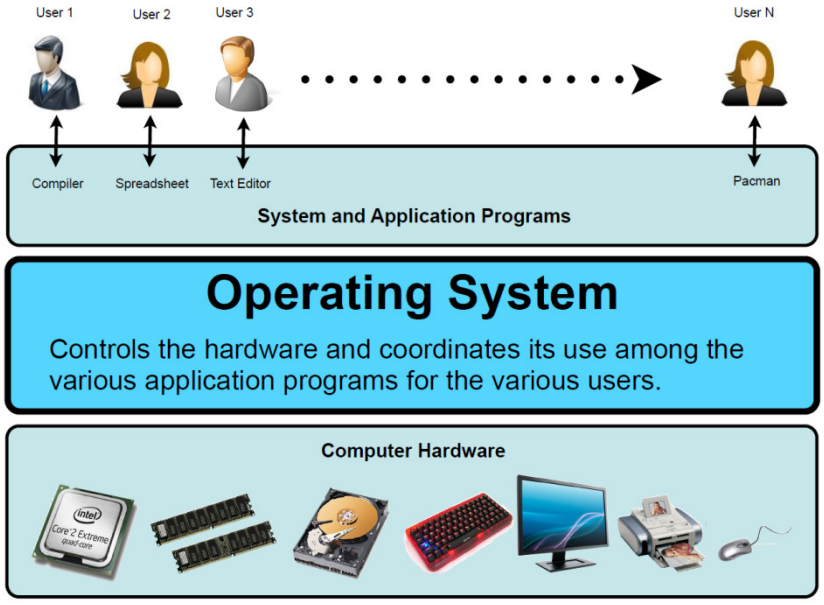
مقدمه و تاریخچه

تعریف سیستم عامل در کتاب سیلبرشاتس (Silberschatz)

سیستم عامل مهم‌ترین برنامه‌ای است که در یک سیستم کامپیوتری مدام در حال اجرا است و یک واسط بین کاربر و سخت افزار فراهم می‌کند؛ همچنین، مدیریت تمامی منابع سیستمی سخت‌افزارها و سایر برنامه‌ها را انجام می‌دهد.

ایراد این تعریف این است که چون بحث ما بر روی سیستم تک پردازنده‌ای است، اجرای واقعی همزمان سیستم عامل در کنار برنامه‌های دیگر ممکن نیست. بلکه سیستم عامل در بازه‌های زمانی مختلف می‌آید و نقش کنترلی خود را انجام می‌دهد و در زمانی که CPU در اختیار برنامه دیگر است، سیستم عامل یا قبلاً تأثیر خود را گذاشته یا بعداً می‌گذارد. بنابراین اگر می‌گوییم سیستم عامل برنامه‌ای است که همیشه در حال اجراست از دید سطح بالا و انسان است و نه از دید CPU.

سیستم عامل استفاده از کامپیوتر را ساده می‌سازد. این بدان معناست که مثلاً کاربر یا برنامه نویس بدون درگیر شدن با مسائل سخت افزاری دیسکها به راحتی فایلی را بر روی دیسک ذخیره و یا حذف کند. این کار در واقع با به کاربردن دستورات ساده‌ای که فراخوان‌های سیستمی (System Calls) را صدا می‌زنند، انجام می‌پذیرد. در صورت عدم وجود سیستم عامل، کاربر و یا برنامه نویس می‌بایست آشنایی کاملی با سخت افزارهای مختلف کامپیوتر (مثل مونیتور، فلاپی، صفحه کلید و غیره) داشته باشد و روتین‌هایی برای خواندن و یا نوشتن آنها به زبانهای سطح پائین بنویسد. بنابراین، سیستم عامل باعث می‌شود کسی که از بالا نگاه می‌کند یک دید انتزاعی، سطح بالا و کاربرپسند نسبت به سیستم پیدا کند.



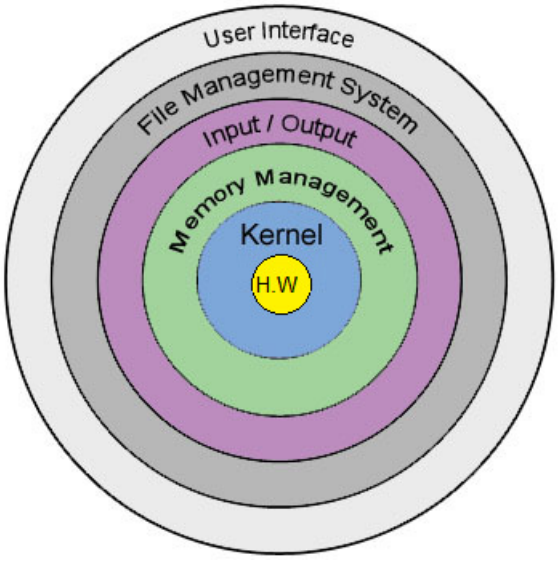
منابع
- **فیزیکی:** CPU و دستگاههای ورودی/خروجی و حافظه
- **منطقی:** فایلها و دادهها، سوکتها، سمافورها و ...

مزایای سیستم عامل

- ✓ جزئیات سخت افزار را از کاربر پنهان می‌کند (برنامه نویسی سخت افزار بسیار مشکل است)
- ✓ سهولت استفاده از کامپیوتر
- ✓ استفاده بهینه وعادلانه از منابع
- ✓ سهولت ارتقاء و توسعه پذیری
- ✓ جلوگیری از خرابکاری کاربران/برنامه‌ها روی همدیگر

ساختار و جایگاه سیستم عامل

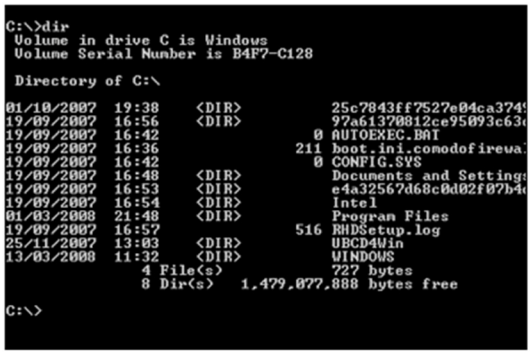
مدلهای متفاوتی در کتابهای مختلف برای ساختار سیستم عامل و جایگاه آن در سیستمهای کامپیوتری پیشنهاد شده است مانند: مدل هرمی، مدل لایه‌ای و مدل پله‌ای.



مدل لایه‌ای در سیستم عامل

در مدل لایه‌ای، بیرونی‌ترین لایه واسط کاربر و درونی‌ترین لایه سخت افزار می‌باشد. واسط کاربر می‌تواند بـه دو صورت "خط فرمان" و "گرافیکی" باشد.

User interface
- Command Line (DOS)
- Graphical (Windows)



هسته سیستم عامل که به آن Kernel می‌گویند، در حافظه اصلی مقیم است و شامل توابعی است که مکرراً استفاده می‌شود. هسته به سخت افزار دسترسی مستقیم دارد. سیستم حداقل از دو حالت اجرا حمایت می‌کند.

۱- حالت ممتاز (حق بالا):

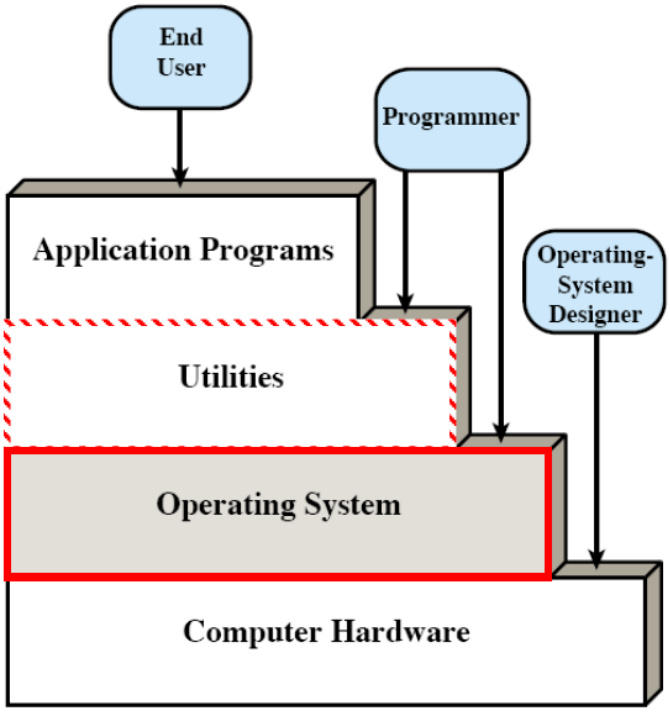
یک وضعیت اجرایی است که به تمام دستورالعمل‌های سخت افزار اجازه اجرا می‌دهد، که به این حالت سیستم، حالت ناظر یا حالت هسته (Supervisor/kernel mode) می‌گویند.

۲- حالت عادی (حق پایین):

حالتی است که اجازه اجرای دستورالعمل‌های حساس سخت افزاری مثل دستورالعمل توقف و دستورالعمل‌های ورودی/خروجی را نمی‌دهد. این حالت را حالت کاربر (user mode) نیز گویند چرا که برنامه‌های کاربران معمولاً در این حالت اجرا می‌شوند.

ویژگی	حالت عادی (User Mode)	حالت ممتاز (Kernel Mode)
اجراکننده	برنامه‌های کاربردی	هسته سیستم‌عامل
سطح دسترسی	محدود و کنترل‌شده	کامل و بدون محدودیت
دسترسی به سخت‌افزار	غیرمستقیم (از طریق سیستم‌عامل)	مستقیم
دسترسی به حافظه	فقط فضای آدرس خود برنامه	تمام حافظه فیزیکی

مدل پله‌ای stallings



تقسیم بندی سیستم عامل از جهت ارتباط با کاربر

- ✓ **محاوره‌ای** (Interactive): به سیستم‌هایی گویند که در آن کاربر به صورت مستقیم با کامپیوتر در ارتباط است، دستوراتی وارد می‌کند و منتظر پاسخ می‌ماند. پس از دریافت پاسخ، مجدداً دستوراتی را وارد می‌نماید.
- ✓ **دسته‌ای** (Batch) سیستم‌های بَچ از اولین نمونه‌های سیستم‌عامل بودند که با هدف بهینه‌سازی استفاده از منابع کامپیوترهای پرهزینه اولیه ایجاد شدند. در این پارادایم، برنامه‌های مختلفی که دارای نیازهای مشابه هستند در یک دسته قرار گرفته و به صورت یک‌جا توسط کامپیوتر پس از بار شدن کامپایلر مورد نیازشان اجرا می‌شوند. مکانیزم اصلی کار بر اساس اجرای متوالی وظایف توسط یک مانیتور دائمی در حافظه بود که پس از اتمام هر کار، کار بعدی را به طور خودکار بارگذاری می‌کرد. مهم‌ترین مزیت این سیستم‌ها افزایش چشمگیر کارایی از طریق حذف زمان‌های مرده بین اجرای برنامه‌های مختلف بود، اما مهم‌ترین محدودیت آن زمان پاسخگویی بسیار طولانی و عدم امکان تعامل بلادرنگ با کاربران محسوب می‌شد. این معماری پایه‌ای برای توسعه سیستم‌عامل‌های پیشرفته‌تر مانند سیستم‌های چندبرنامه‌ای شد که با بهره‌گیری از زمان‌بندی هوشمند، امکان اجرای همزمان چندین برنامه را فراهم می‌کنند.

نمونه‌ای ازیک سیستم دسته‌ای ساده

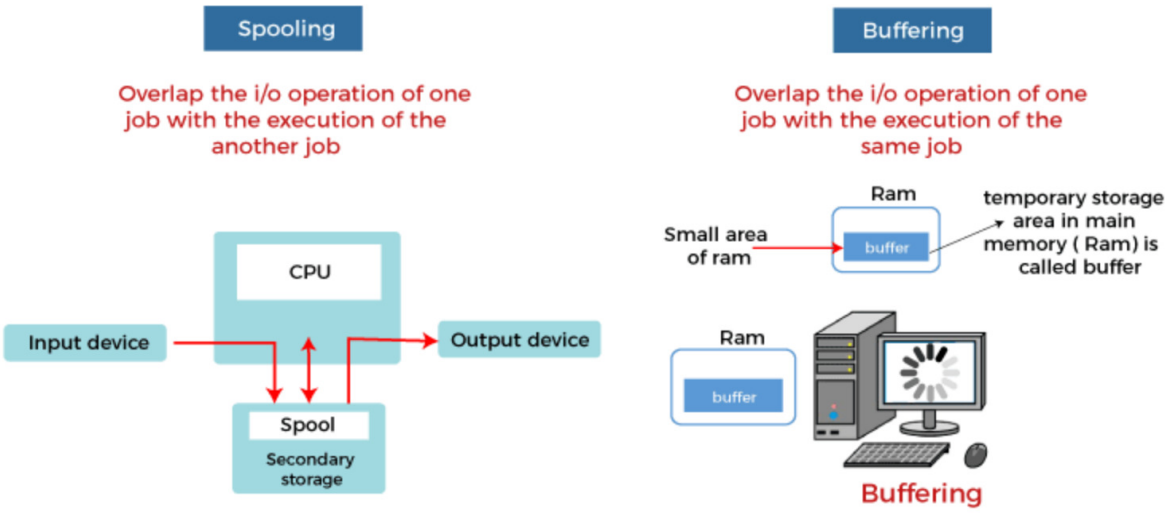
تقسیم بندی سیستم عامل از جهت ارتباط با دستگاههای ورودی/خروجی

- ✓ **مستقیم** (online): پردازنده به صورت مستقیم با دستگاه ورودی/خروجی در ارتباط است. بـه دلیـل کند بودن دستگاههای I/O کارایی پردازنده به صورت قابل توجهی کاهش می‌یابد.
- ✓ **غیر مستقیم** (offline): در این دسته پردازنده به صورت مسـتقیم بـا دسـتگاه ورودی و خروجـی در ارتباط نیست. بلکه ابتدا عمل خواندن ورودی توسط یک سری کارت خوان انجام می‌گیرد. داده ها به حافظه جانبی یا نوار مغناطیسی انتقال می‌یابد. سپس پردازنده داده‌ها را از این حافظه جـانبی یـا نـوار مغناطیسی می‌خواند. در این روش امکان همپوشانی عملیات ورودی/خروجی و کار پردازنـده وجـود دارد.

بافر کردن (Buffering)

علی‌رغم استفاده از نوارهای مغناطیسی باز هم بهره‌وری سیستم به دلیل کند بـودن عملیـات ورودی و خروجـی کاهش می‌یابد. به این دلیل می‌توان از حافظه‌های میانی (بافرها) عملیات ورودی و خروجی یک برنامه را با اجرای آن به صورت همزمان انجام داد. این عمل در مورد کارهایی که دارای حجم عملیات محاسباتی و ورودی/خروجی متناسب می‌باشند، باعث افزایش کارایی می‌شود.

Spooling: در سیستم‌هایی که دارای دیسک‌های سریع و با حجم زیاد باشند می‌توان بجای انتقال داده به نوار مغناطیسی و سپس استفاده از آن، داده را روی دیسک ذخیره نمود و به این ترتیب در حین پردازش یک کار عمل انتقال داده‌ها و برنامه‌های دیگر به روی دیسک به وجود آمده و در نتیجه همزمانی اجرایی چندین برنامه امکان پذیر می‌شوند.



مزایای Spooling

۱) افزایش بهره‌وری چرا که همه دستگاه ها با هم کار می‌کنند.

۲) استفاده از راه دور ساده شد یعنی شرکت‌های کوچک می‌توانند یک کامپیوتر ارزان بخرند برنامه را بوسیله کارت‌خوان خوانده و برنامه را به صورت نوار به سرور اصلی انتقال دهند.

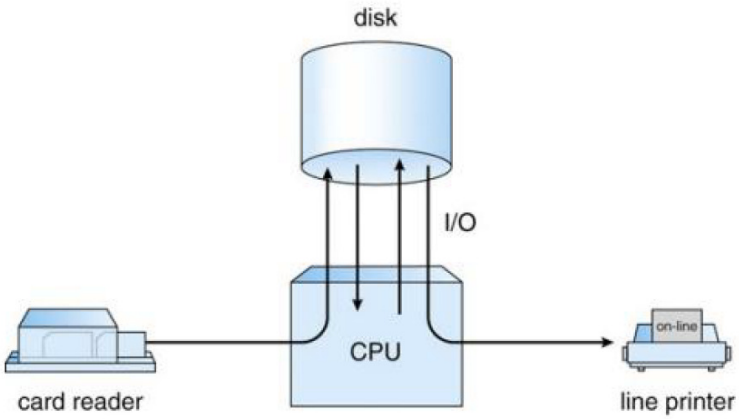
معایب Spooling

۱- ارتباط با کاربر همچنان offline است.

۲- زمان برگشت کار طولانی است.

۳- چک کردن و برطرف کردن ایرادات وقت‌گیر است.

ویژگی	Buffering	Spooling
هدف اصلی	همگام‌سازی سرعت	ایجاد صف برای دستگاه‌های غیرقابل اشتراک
منبع ذخیره‌سازی	**حافظه اصلی(RAM)**	**حافظه ثانویه (دیسک)**
مقیاس کاربرد	برای **یک فرآیند** خاص	برای **چندین فرآیند/کاربر** در کل سیستم
ظرفیت	محدود (به اندازه حافظه اختصاص یافته)	بسیار بزرگ (به اندازه فضای دیسک)
مدت زمان نگهداری	**کوتاه‌مدت** - موقتی	**بلندمدت**- تا زمان سرویس‌دهی
اصلی‌ترین فایده	جلوگیری از بیکاری CPU	امکان چندبرنامگی و اشتراک‌گذاری دستگاه‌ها
مثال رایج	پخش ویدیو، خواندن/نوشتن داده از/روی دیسک	سیستم صف چاپ(Print Spooling)

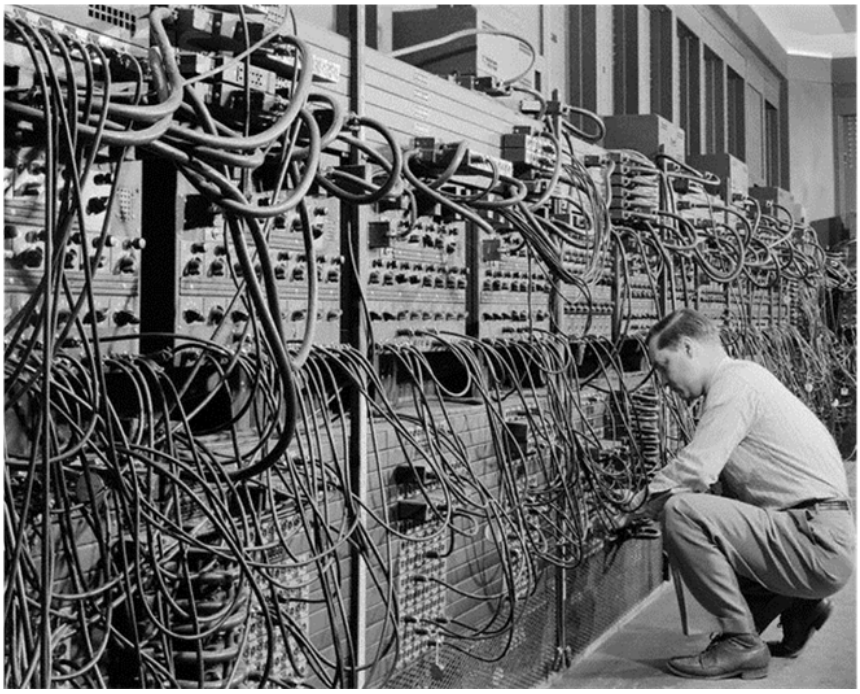


بررسی تاریخی سیر تکاملی سیستم عامل

بررسی تاریخچه سیستم‌های کامپیوتری، مشکلات پیش روی طراحان و اپراتورهای اولیه را نشان می‌دهد و ایده ها و راه‌حلهای کلیدی را بیان می‌کند و نشان می‌دهد که علم کامپیوتر اساساً یک علم تجربی است و ویژگی‌ها عمدتا از نیازها تکامل یافته‌اند. سیستم های عامل به طور کامل وابسته به ساختار معماری کامپیوترهایی هستند که روی آنها اجرا می‌شوند.

نسل اول کامپیوترها (۱۹۵۵-۱۹٤٥)

در نسل اول کامپیوترها که از لامپ خلأ برای ساخت آنها استفاده می‌شد(مانند کامپیوتر ENIAC). زبانهای برنامه نویسی (حتی اسمبلی) ابداع نشده بودند و سیستم عامل نیز اصلاً وجود نداشت. روند کار به این صورت بود که برنامه نویسان تنها در یک فاصله زمانی مشخص حق استفاده از کامپیوتر بزرگ و گران قیمت را داشتند. آنها برنامه‌های خود را توسط تخته مدار سوراخدار و به زبان ماشین به کامپیوتر می‌دادند. برنامه نویس در واقع خودش یک اسمبلر بود یعنی برنامه را با زبان ماشین و با صفر و یک‌ها باید می-نوشت. اطلاعات به صورت دستی و مستقیم با استفاده ازسوییچ وارد می‌شوند.



ایراد سیستم عامل های نسل اول

۱) منابع را هدر می‌دادند چون وقتی ورودی کار می‌کند خروجی بیکار است و ...

۲) Turnaround time آن‌ها بالا بود یعنی زمان پاسخ خیلی طولانی بود.

۳) ارتباط با کاربر offline بود.

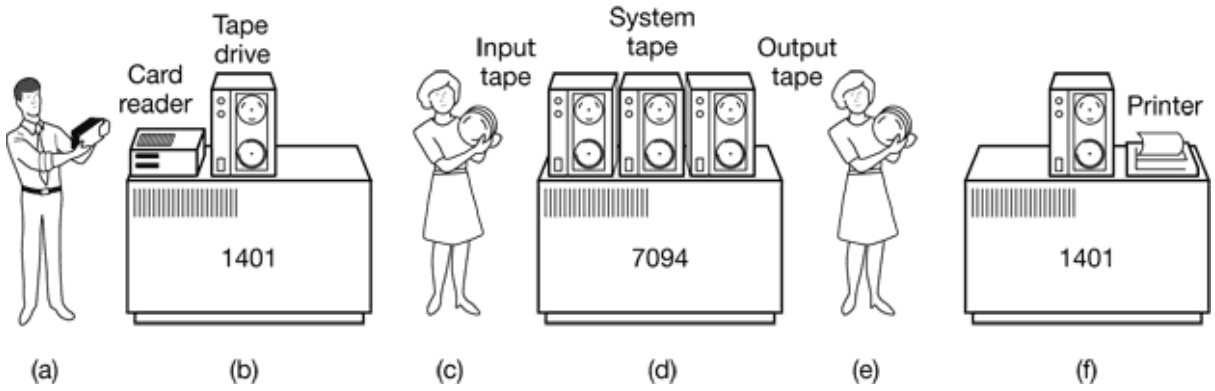
٤) Debug کردن برنامه‌ها سخت بود.

نسل دوم کامپیوترها (۱۹٦٥-۱۹٥٥)

در این نسل سخت افزارهای جدیدی پدید آمد مثلاً ترانزیستور به جای لامپ خلاء، card reader به جای تخته مدار سوراخ‌دار، پرینتر به جای لامپ‌هایی که خاموش و روشن می شدند، tape drive به عنوان ورودی که می توانیم OS آن را لود کنیم. همچنین، کامپایلر به وجود آمد و زبان‌هایی مثل PL1 ،Fortran، COBOL.

OS و کامپایلر بر روی tape هایی ذخیره می‌شدند که به آن System tape می‌گفتند. برنامه کاربر بر روی کارت پانچ‌ها و به وسیله Card reader خوانده و در حافظه Load می‌شد. بعد از پایان هر کار، CPU به سراغ کار بعدی می‌رفت.

در این نسل، تمایز بین برنامه نویس و اپراتور مشخص گردید.

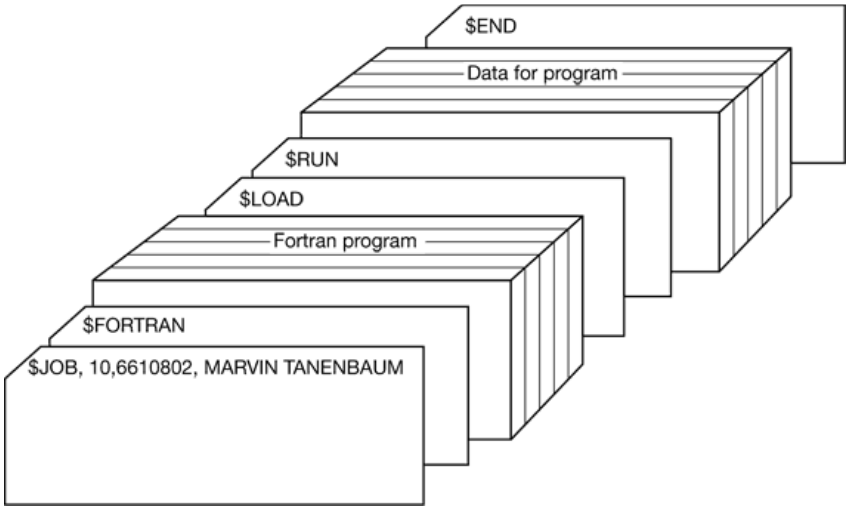


برنامه نویس، برنامه خود را به صورت دسته‌ای از کارتهای سوراخ‌دار (job) تحویل کارت خوان می‌دهد. اپراتور دسته‌ای از jobها را به نوار مغناطیسی منتقل می‌کند (مفهوم Batch). سپس برنامه‌ها از نوار خوانده شده و پس از اجرا، خروجی به نوار دیگری منتقل می‌گردد. در نهایت، اپراتور نوار خروجی را برای چاپ بوسیله پرینتر به IBM 1401 تحویل می‌دهد.

IBM 7094: برای پردازش (گرانتر و سریعتر) IBM 1401: برای I/O (ارزانتر و کندتر)

IBM 7094

- ✓ نسل دوم اولین نسلی است که در آن سیستم عامل به وجود آمد. سیستم‌های عامل این نسل را Batch می‌گویند.
- ✓ دستگاه‌های جانبی مستقیماً به پردازنده وصل نیست و اپراتور این کار را انجام می‌دهد. برنامه‌نویس‌ها، برنامه‌های خود را در اختیار اپراتور قرار می‌دهند و سپس اپراتور برنامه‌هایی که الزامات یکسان دارند در دسته‌هایی گروه‌بندی می‌کند.

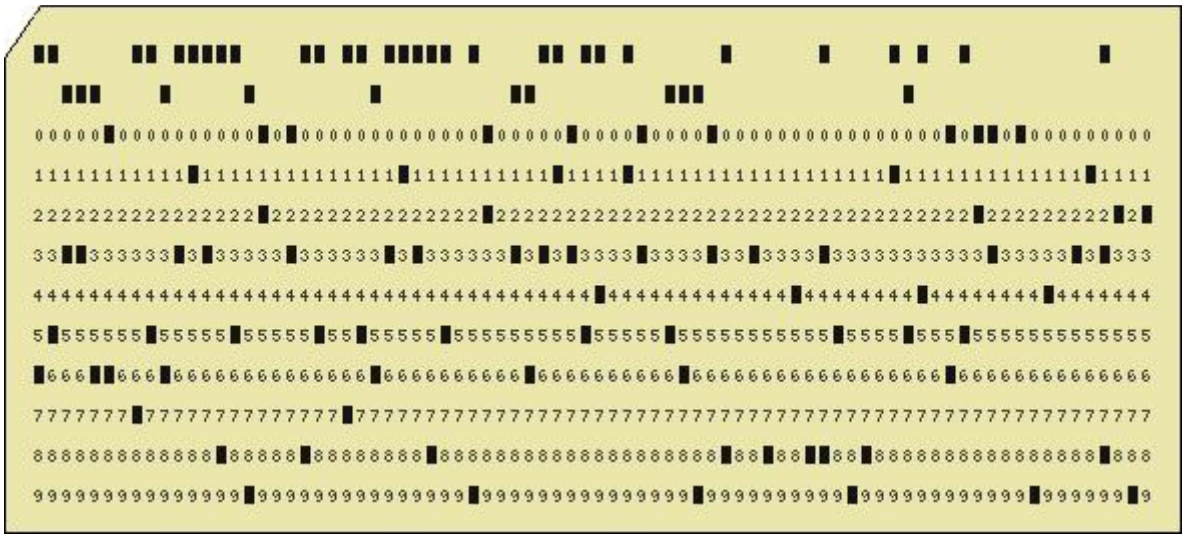


ساختار یک job به صورت دسته‌ای از کارت‌ها (job)

(کارت‌های کنترلی باید به زبان JCL باشند)

Job control language (JCL)

در سیستم‌های دسته‌ای قدیمی، کاربر نیازهای خود را از طریق یک سری دستور به نام JCL به سیستم عامل می داد. سیستم عامل نیز بر مبنای دستورات JCL منابع مورد نیاز را در اختیار آن برنامه می‌گذاشت. به عبارتی دیگر اطلاعات یک Job به صورت یک بسته متشکل از JCL ، برنامه‌ها و داده‌ها (Data + JCL + Program) به سیستم عامل داده می‌شود. بنابراین JCL حاوی اطلاعاتی است که کاربر به سیستم عامل می‌دهد و به آن می‌گوید چه کار کرده (از چه کامپایلری استفاده کند، روی چه داده‌ای کار کند و ...) و از چه منابعی استفاده کند.



شمای یک کارت پانچ

نسل سوم کامپیوترها (۱۹۸۰-۱۹٦۵)

در این نسل در سخت افزار اتفاقات جالبی افتاده بود:

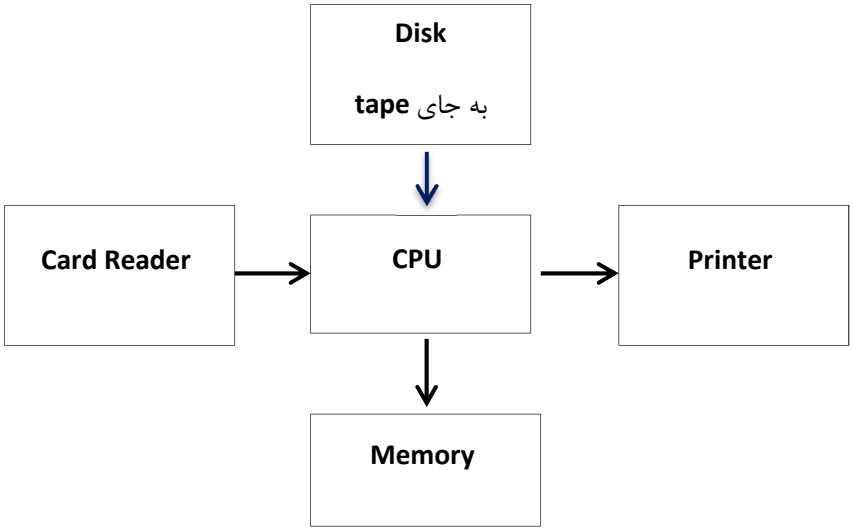
۱- ظهور ICها

۲- ظهور هارددیسک که یکی از تحولات این نسل است. آنقدر مهم که مایکروسافت نام اولین سیستم عامل خود را DOS به معنای Disk operating System گذاشت.

۳- مکانیزم‌هایی مانند buffering، interrupt، online spooling در این نسل آمده است.

٤- در اواخر این نسل ظهور ترمینال‌ها باعث شد ارتباط با کاربر online و interactive شود.

سازمان کامپیوترهای این نسل:



چند برنامگی (Multi Programming)

اصولا کارها در کامپیوتر به دو دسته تقسیم می‌شود:

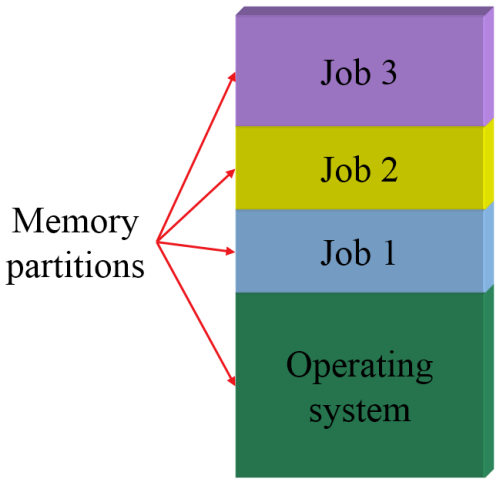
۱- کارهای محدود به محاسبه یا **محدود به CPU** (CPU-bound) که به محاسبات علمی سنگین می پردازند و I/O به ندرت در آن‌ها به کار می‌رود. زمان تلف شده در این نوع کارها اهمیت چندانی ندارد.

۲- برنامه‌های **محدود به I/O** (I/O-bound) مانند پردازش داده‌های تجاری که زمان انتظار I/O اغلب ۸۰ تا ۹۰ درصد کل زمان را به خود اختصاص می‌دهد.

در سیستم‌های تک برنامگی، درصد بالایی از وقت CPU به هدر می‌رود؛ بنابراین باید برای پرهیز از این همه بیکاری پردازنده چاره‌ای بیابیم. راه حل ارائه شده این بود که حافظه را به چند تکه تقسیم کنیم و همان طور که در شکل دیده میشود در هر پارتیشن یک کار مجزا قرار دهیم. البته باید ترکیب مناسبی از کارهای I/O bound و CPU bound برای بارگذاری در حافظه انتخاب شود. وقتی که یک کار برای تکمیل عملیات I/O منتظر می-ماند بلافاصله یکی دیگر از کارهای درون حافظه، پردازنده را در اختیار می‌گیرد. اگر تعداد کارهای موجود در حافظه کافی باشد می‌توان پردازنده را صد در صد مشغول نگه داشت. البته نگهداری همزمان چند برنامه در حافظه نیاز به مدیریت خاص حافظه دارد تا برنامه‌ها بر همدیگر اثر سوء نداشته باشند.

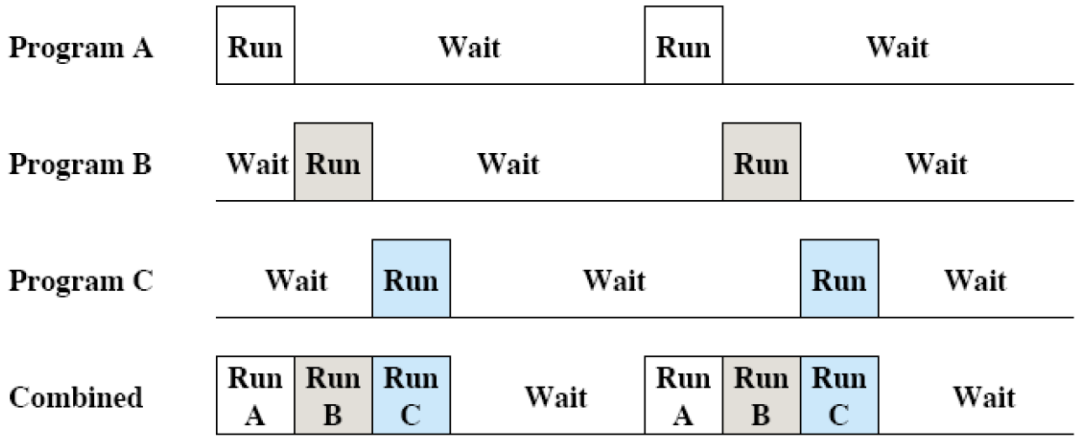
کاربر درخواست خود را وارد می‌کند. بعد کامپیوتر آن را اجرا می‌کند و کاربر نتایج خود را بعداً می‌گیرد (چند دقیقه یا چند ساعت بعد). در چند برنامگی اجرای یک کار تا زمان نیاز به ورودی یا خروجی انجام می‌گیرد. سپس پردازنده اجرای کار دیگری را شروع کرده یا ادامه می‌دهد.

- Multiple jobs in memory
- Memory partitioning
- Protected from one another
- Resources (time /hardware) split between jobs
- Still **not** interactive



مزیت Multi Programming

- ✓ زمان کمتری هدر می‌رود.
- ✓ استفاده بهینه از منابع (حافظه، CPU ، I/o)



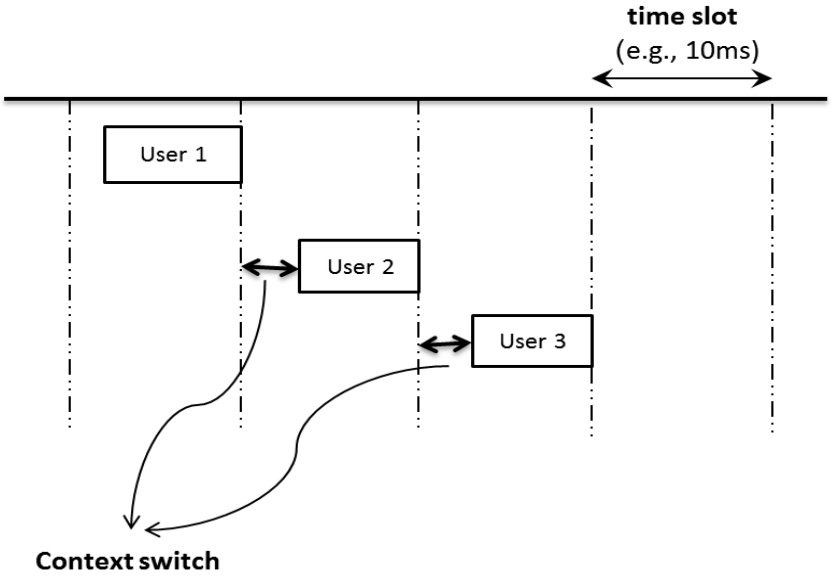
سیستم عامل دارای قابلیت چند برنامگی به مراتب پیچیده‌تر از حالت تک برنامگی است:

- نیاز به مدیریت حافظه داد تا حافظه را به برنامه های متعددی اختصاص دهد.
- نیاز به زمانبندی CPU دارد تا از بین job های آماده، یکی را برای اجرا انتخاب نماد.
- متکی به تواناییهای سخت افزاری ویژه‌ای برای ورودی/خروجی می‌باشد. (مانند وقفه و DMA)

مفهوم Time_ sharing (اشتراک زمانی)

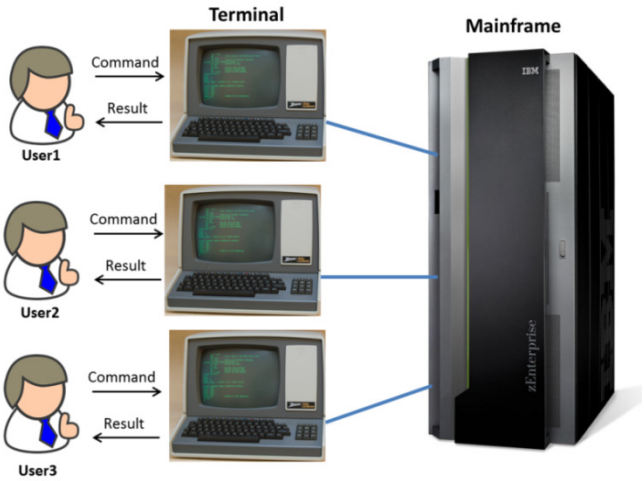
برای کاربردهای محاوره‌ای (Interactive) لازم است که میزان معطلی اجرای برنامه ها کم شود. پس زمان اجرا بین آنها پخش می‌شود. در این روش زمان به بازه‌های کوچکی به نام کوانتوم یا time slice تقسیم می‌شود. در پایان کوانتوم، زمان‌سنج، یک Interrupt سخت افزاری می‌فرستد. وقتی برنامه‌ای وقتش به پایان رسید، کنترل به زمانبند سپرده می شود و برنامه بعدی را مشخص می‌کند.

در این حالت هر کاربر فکر می‌کند فقط خودش با سیستم کار می‌کند. چون معمولا پردازنده بسیار سریع است.



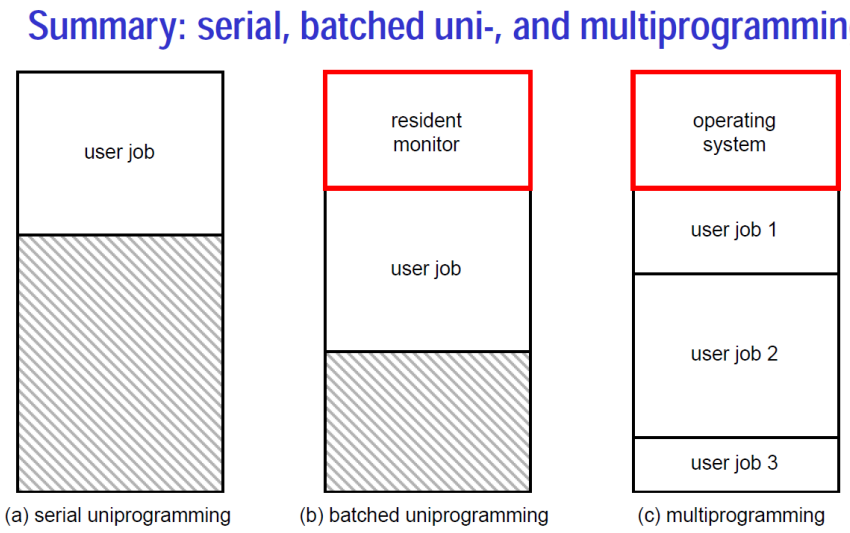
مزایای Time sharing

- ✓ ارزان بودن ترمینال‌ها (Cheap terminals)
- ✓ Interactive(تعاملی) بودن: یعنی اینکه جواب درخواستی که داده‌ایم زود بـه مـا داده مـی‌شـود. مثـل کامپیوترهای امروزی
- ✓ زمان پاسخ (response time) بهینه‌تر می‌گردد.



توجه: Terminal یک کامپیوتر نیست بلکه یک دستگاه ورودی/خروجی است.

Context switch time: زمان تعویض متن. هنگامی که وقفه در سیستم عامل رخ می‌دهد ابتدا سیستم عامل وضعیت کامل برنامه در حال اجرا را حفظ می‌کند. سپس وقفه را بررسی می‌کند و کنترل را به یک روال وقفه مناسب تحویل می‌دهد. پس از آن سپس به برنامه جدید مراجعه می‌شود. به این جریان کاری **"تعویض متن"** گفته می‌شود.



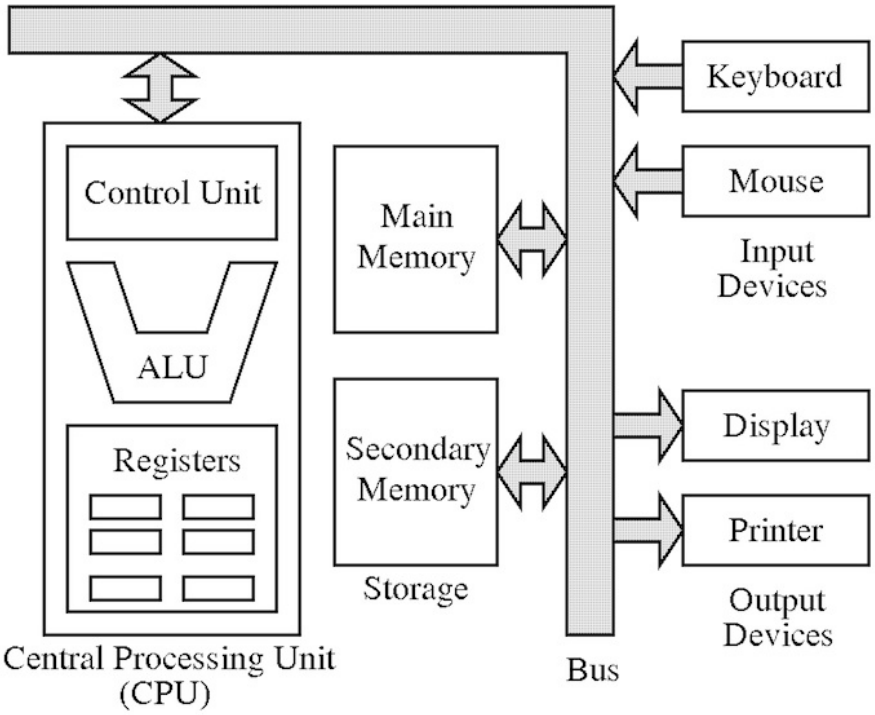
نسل چهارم (ظهور تکنولوژی VLSI)

در این نسل مدارهای مجتمع بسیار فشرده VLSI ظهور کردند که به آنها میکرو پروسسور یا ریزپردازنده می‌گویند. تراکم و اندازه IC ها کوچک‌تر می شود که خود موجب کاهش حجم و کاهش قیمت کامپیوترها می شود و درنتیجه باعث **ظهور کامپیوترهای شخصی** و **حرکت از چندکاربره به تک‌کاربره** گردید. این امر خود باعث سهولت استفاده از کامپیوترهای شخصی می‌شود.

سیستم عامل و معماری کامپیوتر اثر زیادی بر روی یکدیگر داشته‌اند. یعنی جهت سهولت کار با سخت افزارهای جدید، سیستم عامل‌ها توسعه یافتند و همچنین در اثنای طراحی سیستم عامل‌ها، مشخص شد که تغییراتی در طراحی سخت افزار می‌تواند سیستم عاملها را ساده تر و کارآمدتر سازد. به همین دلیل، یادآوری مختصری در مورد ساختار و معماری سیستمهای کامپیوتری در ادامه آورده شده است.

اجزای یک سیستم کامپیوتری (مروری بر برخی مفاهیم درس معماری کامپیوتر)

یک سیستم کامپیوتری دارای واحدهای ورودی و خروجی، پردازنده و حافظه است.



(معماری Von-neumann)

واحد پردازش مرکزی خود به دو بخش واحد کنترل و مسیر داده(datapath) تقسیم می‌گردد. مسیرداده شامل واحدهای اجرایی (مانند ALU)، بخشهایی برای نگهداری اطلاعات (cache ها و رجیسرها) و مدارتی برای انتقال اطلاعات(باسها و MUX ها) می‌باشد. واحد کنترل وظیفه تولید سیگنالهای کنترلی برای مسیرداده را برعهده دارد و می‌تواند به دو صورت پیاده سازی شود:

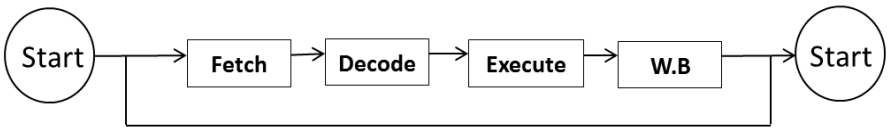
Control Unit
- Hardwired
- Micro-programming

رجیسترها را می‌توان به دو دسته تقسیم نمود:

- ✓ رجیسترهایی که توسط برنامه نویس قابل مشاهده هستند مثل DR و یا رجیسترهای R0-R31 در پردازنده MIPS
- ✓ رجیسترهایی که برنامه نویس نمی‌تواند از آن‌ها استفاده کند و توسط واحد کنترل استفاده می‌شوند.

مثل PC , IR

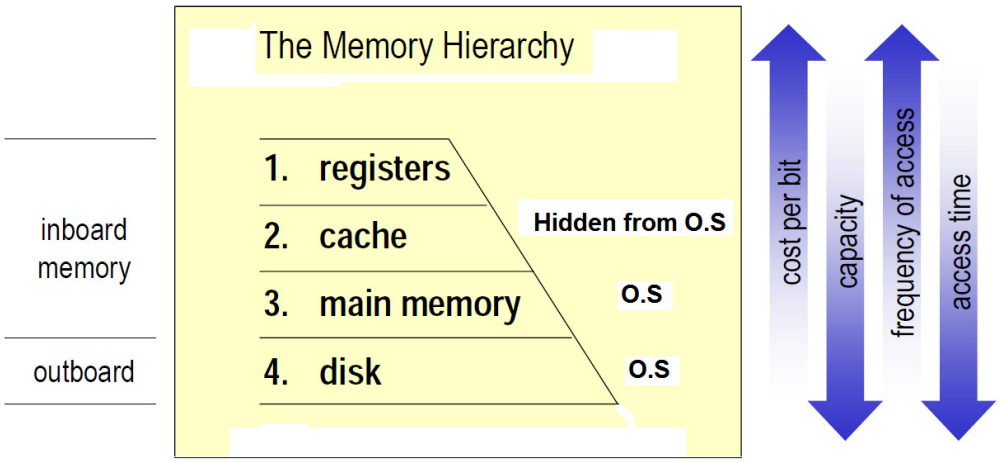
Instruction Cycle



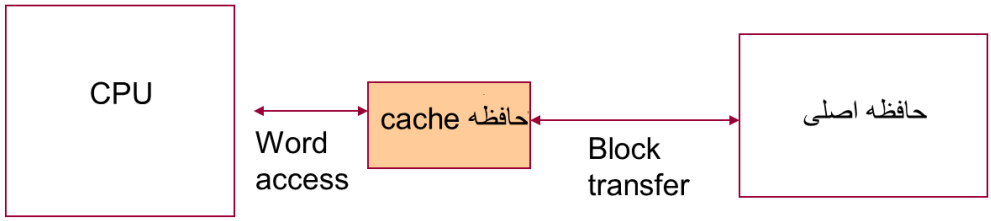
Fetch: واکشی دستورالعمل از حافظه (یا cache) به یک رجیستر خاص در داخل پردازنده

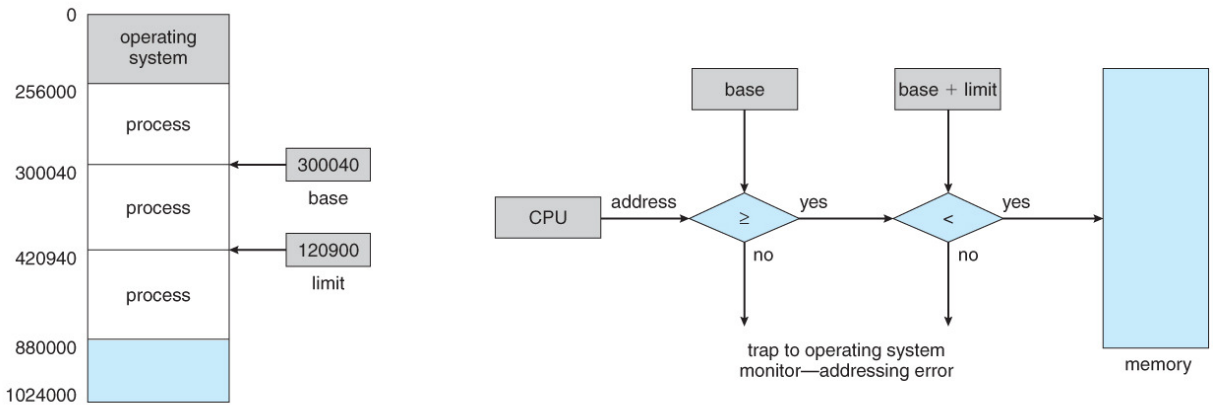
W.B: نوشتن نتیجه در حافظه یا رجیستر

سلسله مراتب حافظه:



سیستم عامل وظیفه مدیریت حافظه اصلی و ثانویه (دیسک) را بر عهده دارد. حافظه cache کاملا از دید سیستم عامل پنهان است.

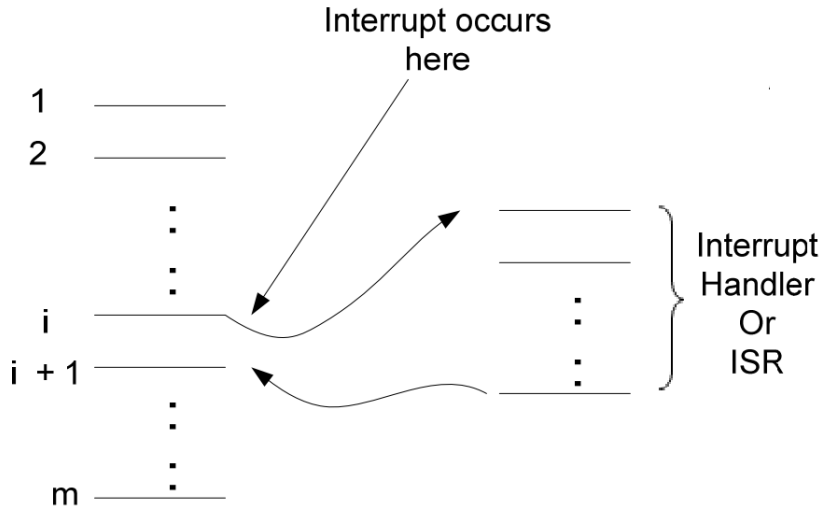




استفاده از رجیسترهای base و limit برای محافظت برنامه‌ها از دسترسی غیرمجاز همدیگر

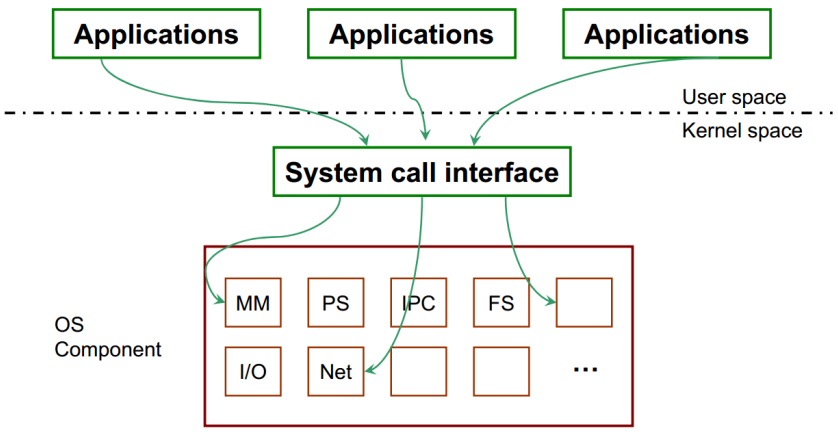
مفهوم وقفه:

سیگنال یا دستوری است که باعث توقف قرایند جاری و رفتن به روالی موسوم به ISR می‌گردد. وقفه یک مکانیسم حیاتی برای سیستم عامل است.

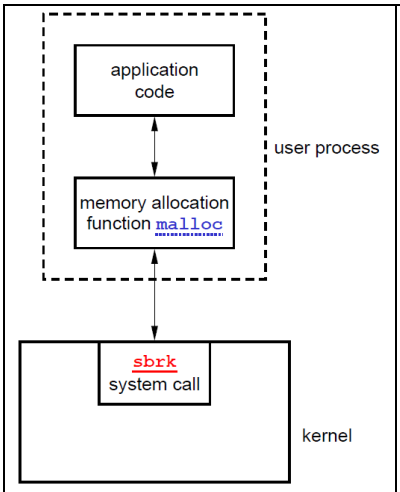


انواع وقفه:
- سخت افزاری (وقفه های I/O – تایمر): *ناهمگام*
- نرم افزاری (تقسیم بر صفر– system calls – overflow): *همگام*

System call یا فراخوانی سیستمی نوعی وقفه نرم افزاری بوده و درخواستی است که یک فرایند سطح کاربر از هسته سیستم عامل (برای هدف خاصی) می‌نماید. در حقیقت فراخوانی سیستمی نقاط ورود به هسته سیستم هستند.



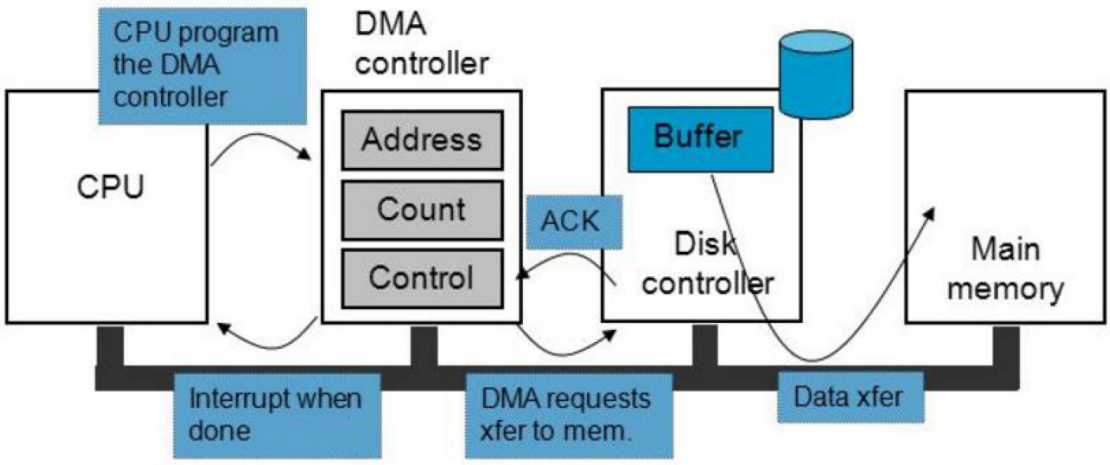
در سیستم عامل Unix بیش از ۳۰۰ فراخوانی سیستمی و Windows نزدیک به ۲۰۰۰ فراخوانی سیستمی دارد. در جدول زیر تعدادی از فراخوانیهای سیستمی در سیستم عاملهای Unix و Windows و همچنین ارتباط یک تابع کتابخانه‌ای در زبان C با یک فراخوانی سیستمی در هسته سیستم عامل نشان داده شده است.



UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

مفهوم DMA (دسترسی مستقیم به حافظه)

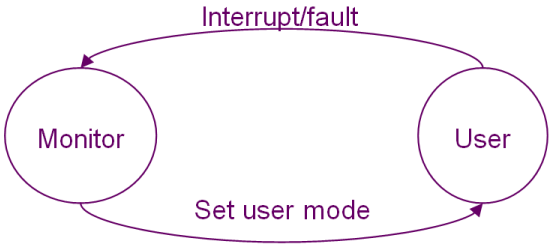
یک ویژگی در سیستمهای کامپیوتری است که به بعضی از دستگاههای I/O اجازه دسترسی به حافظه سیستم را بصورت مستقل از واحد پردازش‌مرکزی می‌دهد (تحت نظارت DMA controller). با استفاده از DMA، پردازنده تنها فرایند انتقال را آغاز کرده، در هنگامی که انتقال در جریان است مشغول کارهای دیگر می‌شود.



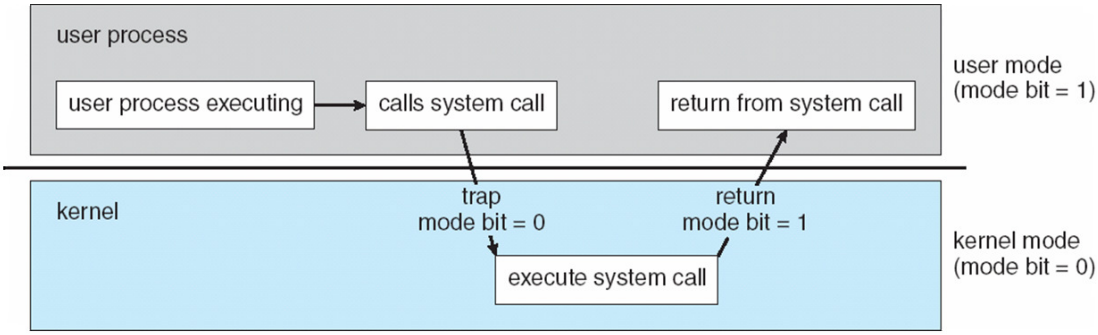
پشتیبانی سخت افزاری از سیستم عامل

سیستم عامل برای عملکرد مؤثر به چندین ویژگی کلیدی از سخت افزار زیرین متکی است. در اینجا برخی از پشتیبانی‌های سخت افزاری ضروری آورده شده است:

حالت‌های پردازش چندگانه: CPU باید حداقل از دو سطح امتیاز پشتیبانی کند که معمولاً به عنوان حالت هسته و حالت کاربر شناخته می شود. حالت هسته به سیستم عامل اجازه می‌دهد تا دستورالعمل‌های ممتاز را با دسترسی کامل به منابع سیستم اجرا کند. حالت کاربر کاری را که برنامه‌ها می‌توانند انجام دهند محدود می‌کند و از تداخل آنها با سیستم عامل یا سایر برنامه‌ها جلوگیری می‌کند. در سیستمهای کامپیوتری بین کد مربوط به هسته سیستم عامل و کد مربوط به برنامه کاربر تمایز وجود دارد. برای این کار در سخت افزار یک بیت به نام بیت حالت (mode bit) وجود دارد. با این کار دوحالت ممتاز و عادی خواهیم داشت. به حالت ممتاز monitor mode یا kernel mode یا system mode گفته می‌شود.



برخی دستورات به طور خاص مجاز (designated as privileged) به حساب می آیند که تنها در مد هسته اجرا می شوند. فراخوانی سیستمی (System Call) مد را به هسته تغییر می دهد و پس از خروج از تابع مربوطه مد به حالت کاربری تغییر می کند.



مدیریت حافظه: سیستم عامل برای مدیریت کارآمد حافظه به پشتیبانی سخت افزاری نیاز دارد. این شامل ویژگی‌هایی مانند واحدهای مدیریت حافظه (MMU) برای ترجمه آدرس‌های حافظه مجازی مورد استفاده برنامه‌ها به آدرس‌های حافظه فیزیکی است. علاوه بر این، پشتیبانی سخت افزاری از حافظه مجازی به سیستم عامل اجازه می‌دهد تا برنامه‌های بزرگتر از حافظه فیزیکی موجود را با مبادله داده ها بین حافظه رم و دستگاه های ذخیره سازی اجرا کند. همچنین، پشتیبانی سخت افزاری برای مکانیسم های حفاظت از حافظه بسیار مهم است. این کار تضمین می‌کند که برنامه‌هایی که در حالت کاربر اجرا می‌شوند نمی‌توانند به حافظه مورد استفاده هسته یا سایر برنامه‌ها دسترسی پیدا کنند و از خرابی‌ها و آسیب‌پذیری‌های امنیتی جلوگیری می‌کند (با استفاده از ازجیسترهای مخصوص). تکنیک هایی مانند بخش بندی حافظه(segmentaion) و صفحه‌بندی (paging) با کمک و حمایت سخت افزار اجرا می شوند.

مدیریت وقفه: وقفه به دستگاه‌ها اجازه می‌دهد زمانی که نیاز به توجه دارند، مانند زمانی که انتقال داده‌ها کامل شده است، CPU را مطلع کنند. سیستم عامل برای مدیریت کارآمد دستگاههای ورودی/خروجی به وقفه ها متکی است. برای مدیریت وقفه نیاز به یک جدول سخت افزاری است (جدول بردار وقفه) که آدرس‌روتین‌های سرویس وقفه خاص (ISR) را ذخیره کند. هنگامی که یک وقفه رخ می‌دهد، CPU برای مکان‌یابی و اجرای ISR مناسب برای مدیریت رویداد به این جدول مراجعه می‌کند.

تایمر: تایمر یک دستگاه سخت افزاری است که وقفه‌های دوره‌ای را در فواصل زمانی مشخص ایجاد می‌کند. سیستم‌عامل می‌تواند از تایمر برای کارهای مختلف از جمله زمان‌بندی فرآیندها و جابجایی متن بین برنامه های در حال اجرا استفاده کند.

دسترسی مستقیم به حافظه (DMA): همانطور که قبلا گفته شد، این امکان به دستگاه‌ها اجازه می‌دهد تا داده‌ها را مستقیماً و بدون دخالت CPU به حافظه منتقل کنند و عملکرد کلی سیستم را بهبود بخشد.

فراهم کردن دستورالعمل‌های اتمی (Atomic Instructions): دستورالعمل‌های اتمی یک قطعه حیاتی از پشتیبانی سخت افزاری برای سیستم عامل‌ها، به ویژه در محیط‌های چند هسته‌ای هستند. دستورالعمل‌های اتمی برای اطمینان از اینکه عملیات خاصی روی داده‌های مشترک به عنوان واحدهای تقسیم‌ناپذیر اجرا می‌شوند، وارد عمل می‌شوند. به عبارت ساده‌تر، یا کل عملیات با موفقیت کامل می‌شود یا اصلاً اتفاق نمی‌افتد. با استفاده از دستورالعمل‌های اتمی، سیستم عامل‌ها می‌توانند عملکردهای حیاتی مانند همگام‌سازی و قفل کردن را انجام دهند. همگام سازی به معنای اطمینان از دسترسی رشته‌ها یا فرآیندها به منابع مشترک به شیوه ای کنترل شده، می‌باشد.

الزامات سخت افزاری خاص برای یک سیستم عامل بسته به پیچیدگی و پلتفرم هدف آن می تواند متفاوت باشد. سیستم‌های تعبیه‌شده ساده ممکن است به حداقل قابلیت‌ها متکی باشند، در حالی که سیستم‌عامل‌های سرور از طیف وسیع‌تری از ویژگی‌های سخت‌افزاری برای بهبود عملکرد، امنیت و قابلیت‌های مجازی‌سازی استفاده می‌کنند. با این حال، عملکردهای اصلی ذکر شده در بالا برای هر سیستم عاملی برای مدیریت موثر منابع و فراهم کردن بستری برای اجرای برنامه ها ضروری است.

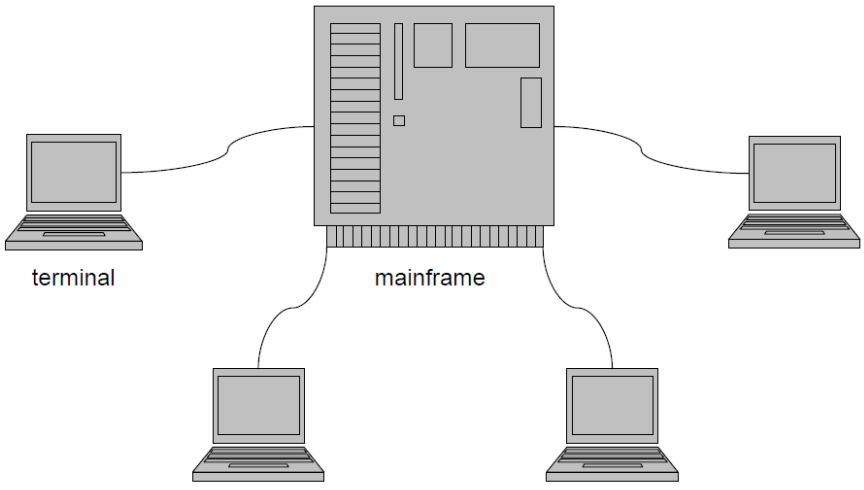
انواع سیستمهای عامل:

- **سیستم عاملهای Mainframe**

سیستم‌های عامل mainframe بیشتر برای برنامه‌هایی که شامل بارهای کاری سنگین ورودی/خروجی هستند، مانند مدیریت پایگاه داده، پردازش تراکنش و انبارداری داده، مناسب هستند. مین فریم‌ها کانال‌های اختصاصی برای مدیریت عملیات ورودی/خروجی دارند که امکان انتقال داده‌های موازی را فراهم می‌کنند و دخالت پردازنده را کاهش می‌دهند. آن‌ها می‌توانند حجم عظیمی از داده‌ها را ذخیره کنند و نیاز به عملیات ورودی/خروجی مکرر را کاهش دهند.

- ظرفیت I/O : ۱۰۰۰ دیسک و هزاران Gigabyte داده
- به عنوان Server های دارای کاربران همزمان خیلی زیاد
- به صورت اشتراک زمانی (Time sharing)

مثالهایی از Mainframe operating systems : MVS , OS/360 , OS/390



- **سیستم عاملهای Desktop و laptop**

– تک کاربره
– هدف اصلی راحتی کاربر است

مثال: Windows , Linux, Mac OS , Free BSD

این نوع سیستم‌های عامل، به کاربر امکان می‌دهند تا به راحتی با کامپیوتر و نرم‌افزارها تعامل کند. برای این منظور، رابط‌های کاربری گرافیکی (GUI) و متنوع که از ماوس، منوها، و صفحات لمسی و حتی ورودی صوتی استفاده می‌کنند، برای تعامل با کامپیوترها ارائه می‌دهند. همچنین، امکاناتی برای مدیریت فایل‌ها، شبکه‌سازی و امنیت را فراهم می‌کنند.

- **سیستم‌های چند پردازشی، شبکه‌ای و توزیع شده**

در سازمان‌هایی که نیازمند انجام محاسبات سنگین یا سرویس دهنده‌هایی با حجم وسیع درخواست‌ها هستند، تمایلاتی به سمت سیستم‌های چند پردازشی یا موازی وجود دارد. در این سیستم‌ها چندین پردازنده به طور موازی به اجرای کارها می‌پردازند.

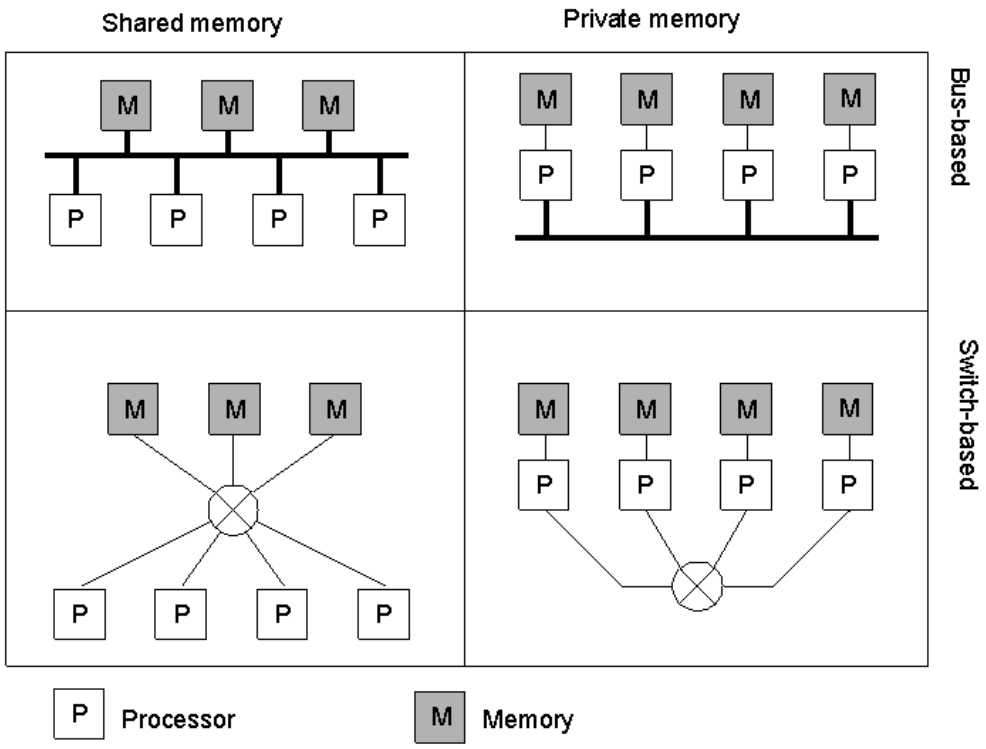
سیستم‌های موازی به دو دسته تقسیم می‌شوند :

۱- **چند پردازنده (Multi-processors)**: این نوع سیستم‌ها خود به دو دسته تقسیم می‌شوند.

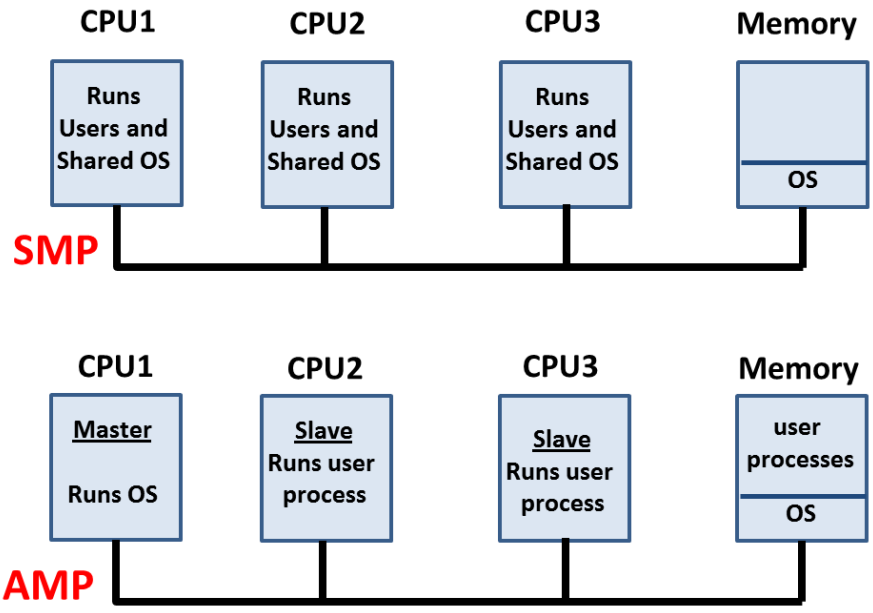
✓ چند پردازنده متقارن (Symmetric): هر پردازنده به طور مجزا می‌تواند هسته سیستم عامل واحد روی حافظه مشترک را اجرا کند و مجموعه فرایندها و نخ‌های خود را زمانبندی کند.

✓ چند پردازنده‌ای نامتقارن(Asymmetric) : مانند پردازنده‌های (MASTER/SLAVE) که معمولا سیستم عامل روی یک پردازنده MASTER اجرا می‌شود و زمان بند واحد آن، برنامه های سطح کاربر را بر روی سایر پردازنده‌ها توزیع می‌کند.

۲- **چند کامپیوتری (Multi-computers)**: چندین پردازنده که هر کدام دارای حافظه اختصاصی خود هستند.



- ***Symmetric multiprocessing (SMP)***
  - Each processor runs an identical copy of the operating system.
  - Many processes can run at once without performance deterioration.
  - Most modern operating systems support SMP
- ***Asymmetric multiprocessing***
  - Each processor is assigned a specific task; master processor schedules and allocates work to slave processors.
  - More common in extremely large systems



نقاط قوت سیستم‌های موازی عبارتند از :

۱- توان عملیاتی بالا
۲- صرفه جویی اقتصادی به علت امکان به اشتراک گذاشتن منابع
۳- افزایش قابلیت اطمینان و تحمل پذیری خطا

معایب اساسی سیستم‌های موازی عبارتند از:

۱- پیچیدگی در ایجاد توازی و همگام سازی فرآیندها و نخ‌ها
۲- مشکلات در راه ایجاد توازی و همگام سازی در مدیریت حافظه

- **سیستم‌های عامل بلادرنگ**

در سیستم‌های بلادرنگ (Real-Time)، هر فرایند مهلت زمانی خاصی برای انجام دارد. زمان پاسخ در این سیستم‌ها الزاماً باید به موقع و تضمین شده باشد. در مقابل، در سیستم‌های اشتراک زمانی، داشتن زمان پاسخ کوتاه مطلوب است ولی الزامی نیست. در سیستم‌های دسته‌ای نیز اصلا محدودیت زمانی وجود ندارد. این سیستم‌ها به دو دسته تقسیم می‌شوند:

۱- **بلادرنگ سخت:** در این سیستم‌ها در تمام مواردی که رویدادی به وقوع می‌پیوندد که نیازی به اجرای یک وظیفه بلادرنگ(Real-Time Task) دارد، اجرای آن وظیفه باید حتما قبل از پایان مهلت تعیین شده تمام شود؛ در غیر این صورت، ممکن است فاجعه‌ای به بار آید و با مشکل یا خرابی غیر قابل جبرانی مواجه شویم. یعنی پس از پایان مهلت زمانی، اجرای وظیفه بی‌معنی است.

مانند بعضی از رویدادها در یک نیروگاه هسته‌ای، هواپیماهای نظامی، روبات‌های صنعتی، سیستم کنترل پرواز، پرتونگاری پزشکی و ... .

این سیستم‌ها تضمین می‌کند که کارهای بحرانی را به موقع انجام دهد. این سیستم‌ها به دلیل محدودیت زمانی بسیاری از خصوصیات سیستم‌های عامل مدرن را استفاده نمی‌کنند.

۲- **بلادرنگ نرم:** در این سیستم‌ها، رعایت مهلت زمانی مطلوب است ولی اجباری نیست و تضمین نمی شود. به عبارت دیگر، سیستم سعی می‌کند کار خود را در مهلت زمانی خاص انجام دهد، ولی اجباری در آن وجود ندارد. حتی پس از پایان مهلت زمانی، اجرای وظیفه معنا دارد. مانند: سیستم‌های چند

رسانه‌ای (صوتی و تصویری)، واقعیت مجازی و غیره ... . از ویژگی‌های این روش آن است که برخی کارها اولویت بالایی دارند. اصولا این سیستم‌ها قابل ترکیب با سیستم‌های دیگر نیز هستند. در هر حال، در اینجا احتمال خطا یا تأخیر وجود دارد.

- **سیستم‌های تعبیه شده (توکار)**

معمولاً سیستم‌های تعبیه شده (Embedded System) به این منظور ایجاد می‌شوند که اجرای وظایف معینی را در داخل یک دستگاه خاص، مدیریت کنند. درواقع این سیستم‌ها، خاص منظوره و معمولا کوچک هستند. برای مثال برای کنترل وسایلی مانند وسایل خانگی و اتومبیل‌ها استفاده می‌شوند. سخت افزار این سیستم ها، اغلب به صورت یک تراشه یا یک برد کوچک است. این سیستم‌ها معمولاً دو مد کاربر و هسته ندارند و اجرای وظایف در بعضی از آن‌ها به صورت بلادرنگ است.

- **سیستم عامل های شبکه‌ای**

پیشرفت‌های جالبی که در اواسط دهه‌ی 1980 انجام شد، گسترش شبکه‌های کامپیوترهای شخصی بود که سیستم عامل‌های شبکه‌ای را به اجرا در می‌آورد. بر خلاف سیستم عامل تک کاربره که امور اساسی مربوط به عملکرد یک کامپیوتر را بر عهده می گیرد، سیستم عامل شبکه باید به درخواستهای ایستگاههای کاری متعدد پاسخ داده و اموری چون ایجاد و اداره حساب کاربران، دستیابی به شبکه، استفاده و به اشتراک گذاری منابع، حفاظت از داده ها و کنترل خطاها را نیز انجام دهد. از سیستمهای عامل شبکه به عنوان مثال می‌توان به Novel netware و Windows 2000 server اشاره کرد.

- **سیستم های عامل توزیع شده**

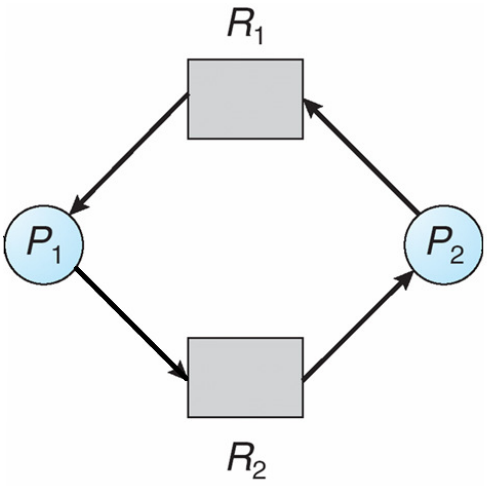
این سیستم‌های عامل در محیط‌های چند پردازنده و چند کامپیوتری همگن به وجود آمده‌اند. اگرچه در این محیط‌ها چندین پردازنده مستقل وجود دارد اما بر خلاف سیستم‌های عامل شبکه‌ای (که در آن کاربران از وجود چندین کامپیوتر باخبرند)، یک **سیستم عامل توزیع شده (Distributed Operating System)** ، خود را مانند یک سیستم تک پردازنده‌ای قدیمی به کاربرانش نشان می‌دهد. در واقع این سیستم‌ها برای کاربران مانند یک سیستم یکتا، متمرکز و منسجم به نظر می‌آیند. کاربران نباید از این امر آگاه شوند که برنامه آن‌ها در کجا به اجرا

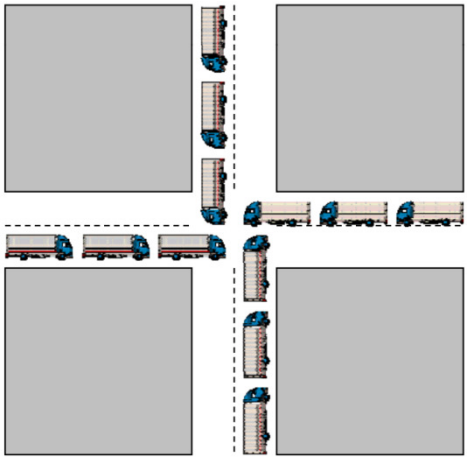
در می‌آید و یا فایل‌های آن‌ها در کجا قرار دارد به عبارت دیگر سیستم باید از دیدگاه کاربر، شفاف یا نامرئی باشد. سیستم‌های متمرکز و توزیع شده، از اساس با یکدیگر متفاوت‌اند و سیستم‌های توزیع شده بسیار پیچیده‌تر هستند.

سیستمهای دیگری نیز وجود دارند مانند: peer-to-peer , Clustered , Handheld , ...

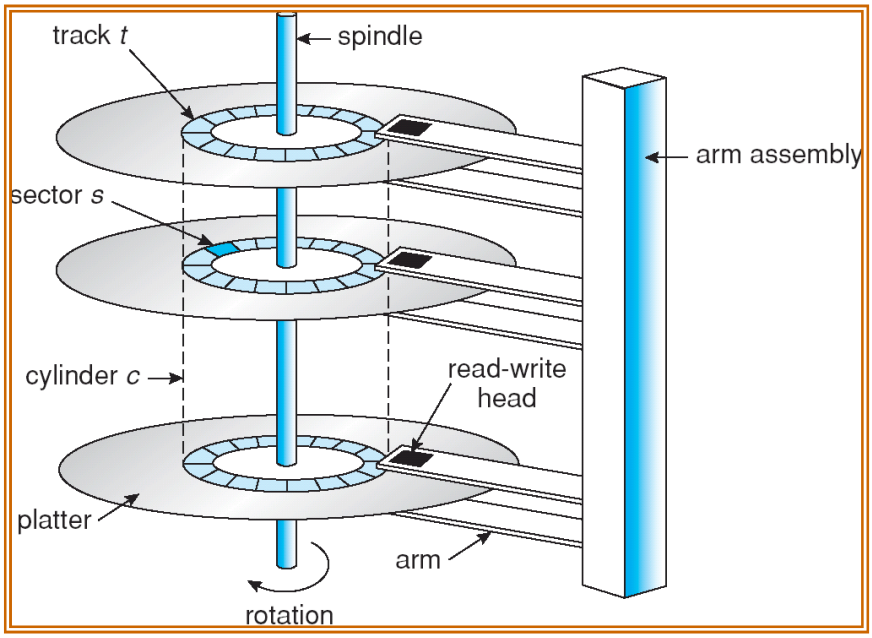
اجزای یک سیستم عامل:

- مدیریت فرایند (Process management): شامل ایجاد و حذف فرایندها – همگام سازی فرآیندها – معلق کردن و شروع مجدد یک فرآیند – ارتباط فرآیندها
- بن بست: شامل روشهایی برای کشف بن بست – جلوگیری و یا اجتناب از بن بست

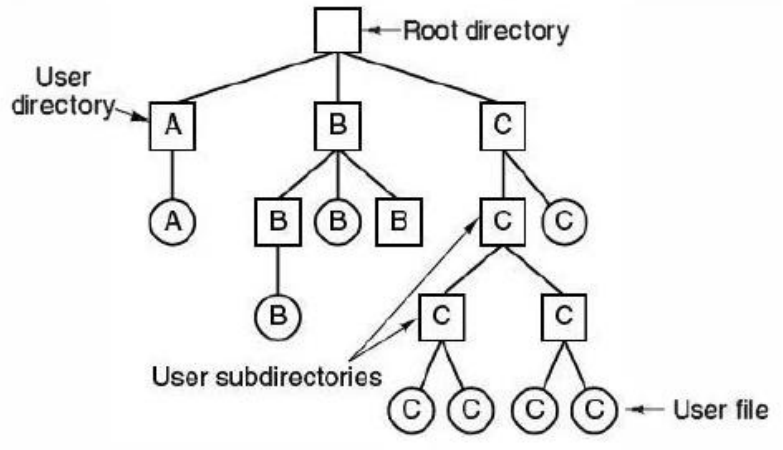




- مدیریت حافظه: شامل تخصیص حافظه و پس گیری آن- پیگیری قطعات استفاده شده و قطعات آزاد حافظه - مکانیسمهای صفحه بندی و قطعه بندی
- مدیریت حافظه ثانویه (دیسک): شامل الگوریتمهایی برای پاسخ به درخواستهای استفاده از دیسک با توجه به موقعیت فعلی head و لیست سکتورهای درخواست شده



- مدیریت ورودی/خروجی: نحوه ارتباط با پورتهای I/O – مدیریت صفهای I/O
- مدیریت فایل: ارائه یک دید منطقی و یکسان از دیتا به نام "فایل" به کاربر، مستقل از رسانه ای که روی آن ذخیره شده است. شامل ایجاد و حذف فایلها – ایجاد و حذف directoryها – مکانیسمهایی برای کار کردن با فایل



- شبکه‌سازی: شامل مکانیسمهایی برای تعریف حساب کاربران و اشتراک منابع و ...
- زمانبند فرآیند: شامل الگوریتمهایی برای اولویت بندی برای اجرای فرآینهای آماده
- مفسر فرمان: واسطه ای برای کاربر که به وسیله آن می‌تواند به سیستم عامل فرمان دهد. در حقیقت فرمانهای کاربر به مجموعه‌ای از system call ها تبدیل می‌شود.

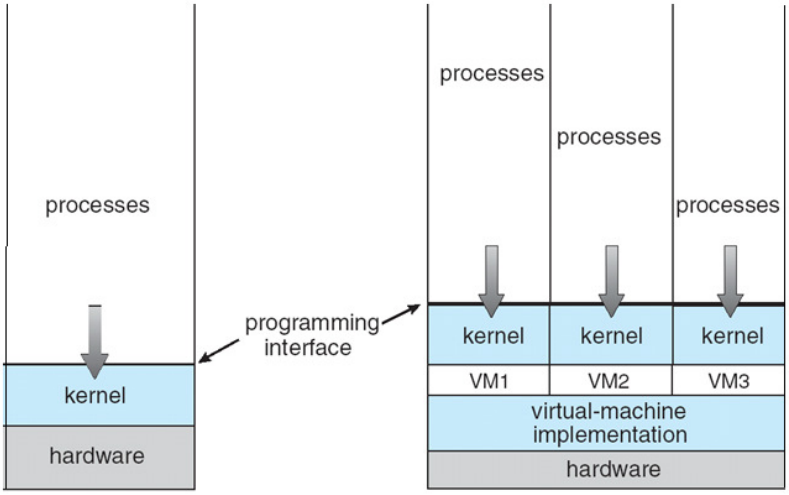
(Copy F1 F2 → open- read- write-creat-close تبدیل میشود به مجموعه از فراخوانیها مانند)

- مدیریت امنیت: شامل برای تامین امنیت کاربران از دسترسیهای غیرمجاز

مفهوم ماشین مجازی (Virtual machine)

یک ماشین مجازی در حکم واسطی است که بر روی سخت افزار قرار می گیرد و آن را بین کاربران مختلف به گونه ای تقسیم می کند که هریک از کاربران فکر می کنند به شکل انحصاری بر روی سخت افزار کار می کنند.

- برنامه هایی از یک سیستم عامل و سخت افزار خاص را می‌توان روی دیگری اجرا نمود.
- هر پروسه فکر می‌کند که تمام منابع سیستم را در اختیار دارد.
- دستگاههای متفاوت به نظر می‌رسد که یک interface واحد دارند.
- هر وقفه‌ای در اجرای برنامه‌ای ایجاد میشود، برنامه نمی‌داند که به دلیل اشتراک منابع است و آن را به صورت تاخیر احساس می‌کند.



(a) Nonvirtual machine (b) virtual machine

مثال: ماشین مجازی VMware