دانشگاه علم و صنعت - دانشکده ریاضی و علوم کامپیوتر

سیستم‌های عامل

فصل اول: مقدمه‌ای بر سیستم‌های عامل

استاد: حمید حاج سیدجوادی

ایمیل: h.s.javadi.hamid@gmail.com

یکشنبه و سه‌شنبه ساعت ۱۰:۳۰ تا ۱۲:۰۰

۱۴۰۲ - ۱۴۰۳

بخش اول
ساختار سیستم‌های کامپیوتری

بخش دوم
سیستم عامل

بخش سوم
تاریخچه سیستم عامل

بخش چهارم
عملیات سیستم عامل

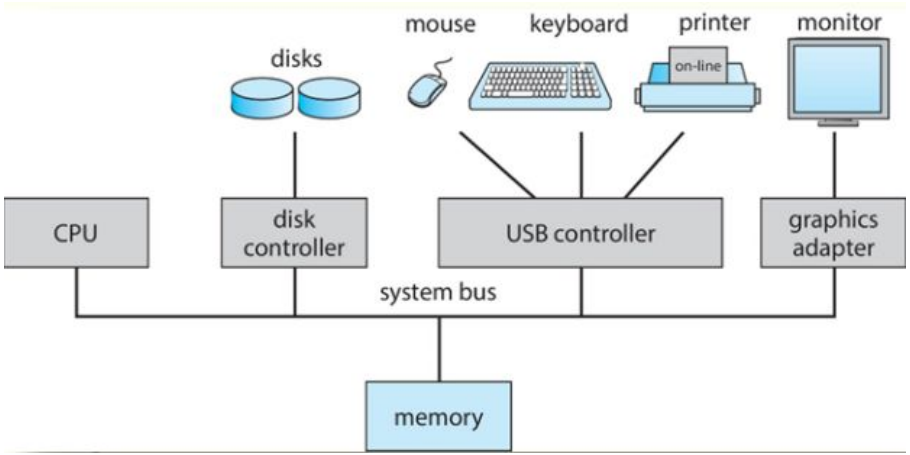
بخش پنجم
ساختار سیستم عامل

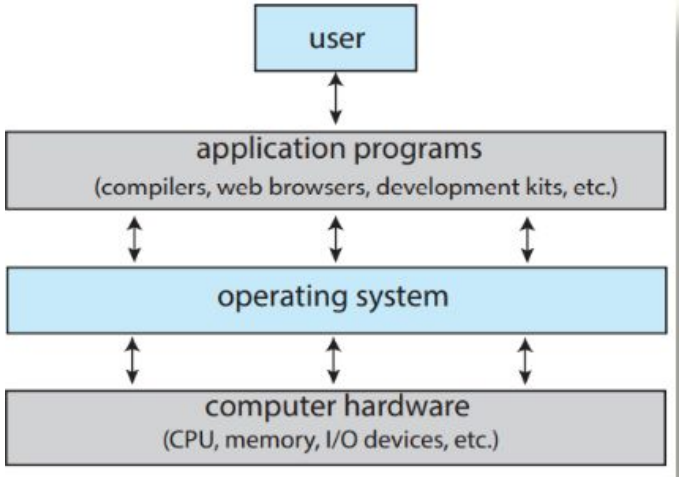
بخش ششم
مفاهیم اولیه

بخش هفتم
جمع‌بندی و پایان

ساختار سیستم‌های کامپیوتری

- یک یا چند CPU، کنترل کننده‌های دستگاه از طریق گذرگاه مشترک به هم متصل می‌شوند و به حافظه مشترک دسترسی دارند.





سیستم عامل

تعریف سیستم عامل :

یک سیستم عامل به عنوان یک واسطه بین کاربر کامپیوتر و سخت افزار کامپیوتر عمل می‌کند. هدف یک سیستم عامل فراهم کردن محیطی است که در آن کاربر بتواند برنامه‌ها را به شیوه ای راحت و کارآمد اجرا کند.

هسته (Kernel) : تنها برنامه‌ای که همیشه بر روی کامپیوتر اجرا می‌شود، هسته، بخشی از سیستم عامل است
برنامه‌های سیستمی : عملیات سیستم را کنترل می‌کنند و مهمترین آنها سیستم عامل است.
برنامه‌های کاربردی : این برنامه‌ها ، کارها و عملیاتی را که کاربران بخواهند انجام می‌دهند.

سیستم‌های عامل امروزی برای مقاصد عمومی و محاسبات سیار نیز شامل میان‌افزار هستند.
میان افزار (Middleware) : مجموعه‌ای از چارچوب‌های نرم‌افزاری که خدمات اضافی را به توسعه‌دهندگان برنامه‌ها مانند پایگاه‌های داده، چند رسانه‌ای، گرافیک و غیره ارائه می‌کنند.

اهداف سیستم عامل :

- اجرای برنامه های کاربر و رفع مشکلات کاربران
- استفاده راحت از سیستم کامپیوتری
- استفاده کارآمد از سخت افزار کامپیوتر

تاریخچه سیستم عامل

سیستم‌های عامل مجموعه ای از توابع و پیوندهای مورد نیاز برای کنترل و همگام سازی سخت افزار کامپیوتر را فراهم می کنند. این توابع توسط اکثر برنامه های کاربردیاستفاده می شود. می‌خواهیم با تاریخچه سیستمهای عامل از ابتدا تا به امروز آشنا شویم و مراحل پیشرفت و تکامل زمانی آن هارا بدانیم.

- نسل اول (دهه ۱۹۴۰ تا اوایل ۱۹۵۰)
- نسل دوم (۱۹۵۵ تا ۱۹۶۵)
- نسل سوم (۱۹۶۵ تا ۱۹۸۰)
- نسل چهارم (۱۹۸۰ تا کنون)

نسل اول (دهه ۱۹۴۰ تا اوایل ۱۹۵۰)

- اولین کامپیوترهای دیجیتال الکترونیکی Z1 که در سال های 1936-1938 ساخته شده بودند؛ هیچ سیستم عاملی نداشتند.
- هر برنامه ای که روی این کامپیوترها اجرا می شد باید ضمن برقراری ارتباط با سخت افزار، تمام کدهای مورد نیاز را برای اجرا روی کامپیوتر شامل می شد، همچنین کامپیوترها به طور کلی برای حل محاسبات ساده ریاضی استفاده می شدند و تمام برنامه نویسی ها به زبان ماشین مطلق، اغلب با سیم کشی پلاگین ها برای کنترل عملکردهای اساسی دستگاه انجام می شد. این وضعیت باعث شد حتی ساده ترین برنامه ها نیز بسیار پیچیده شوند.
- در پاسخ به این مشکل با توجه به تکامل و پیچیده تر شدن سخت افزارها، صاحبان کامپیوترهای مرکزی شروع به توسعه نرم افزارهای سیستمی کردند که نوشتن و اجرای برنامه های موجود در کامپیوتر را تسهیل می کرد و بدین ترتیب اولین سیستم های عامل متولد شدند.

نسل دوم (۱۹۵۵ تا ۱۹۶۵)

- اولین سیستم عامل که در اوایل دهه 1950(در سال 1956)معرفی شد، GMOS نام داشت و توسط رابرت ال. پاتریک از جنرال موتورز برای ماشین IBM 701 ایجاد شد. در دهه 1960 ،IBM اولین تولید کننده کامپیوتر شد و وظیفه توسعه سیستم های عامل را بر عهده گرفت و شروع به توزیع سیستم های عامل موجود در کامپیوترهای خود کرد.
- سیستم های عامل در دهه 1950 سیستم های پردازش دسته ای تک جریانی نامیده می شدند زیرا داده ها به صورت گروهی ارسال می شدند. این ماشین های جدید frame main نامیده می شدند و توسط اپراتورهای حرفه ای در اتاق های بزرگ کامپیوتر استفاده می شدند. از آنجایی که قیمت این ماشین ها زیاد بود، تنها سازمان های دولتی یا شرکت های بزرگ قادر به خرید آن ها بودند.
- پیدایش این سیستم عامل زمانی اتفاق افتاد که کامپیوترها می توانستند تنها یک برنامه را در یک زمان اجرا کنند. در دهه های بعدی، کامپیوترها شروع به گنجاندن برنامه های نرم افزاری بیشتر و بیشتری کردند که برای ایجاد سیستم های عامل امروزی گرد هم آمدند.
- کامپیوترهای این نسل اکثراً به زبان فورترن و اسمبلی برنامه نویسی شده بودند.
- نمونه هایی از سیستم های عامل بکار رفته در این کامپیوترها عبارتند از :

FMS (Fortran Monitor System)

IBSYS(IBM’s Operating System for 7094)

نسل سوم (۱۹۶۵ تا ۱۹۸۰)

- سیستم های دهه 1960 نیز سیستم های پردازش دسته ای بودند، اما توانستند با اجرای چندین کار به طور همزمان از منابع کامپیوتر بهره ببرند. بنابراین طراحان سیستم عامل مفهوم چند برنامه نویسی (Multi Programming) را توسعه دادند که در آن چندین کار به طور همزمان در حافظه اصلی صورت می گرفت. معرفی چند برنامه نویسی بخش عمده ای در توسعه سیستم های عامل بود، زیرا اجازه می داد تا CPU تقریباً 100 درصد مواقعی که کار می کرد، مشغول باشد
- در اواخر دهه 1960، آزمایشگاه های بل شروع به کاربر روی منشأ یونیکس کرد و اولین نسخه از سیستم عامل چند وظیفه ای و چند کاربره یونیکس توسط برنامه نویسان Mcllroy، T&T Bell Labs Ken Thompson، Dennis Ritchie، Douglas و Joe Ossanna توسعه یافت و در دهه ۷۰ ابتدا در شرکت بزرگ T&AT و سپس توسط کالج ها و دانشگاه ها مقبولیت گسترده ای به دست آورد و در دسترس قرار گرفت. این سیستم عامل به زبان C نوشته شده است و به صورت آزاد در دسترس عموم می‌باشد (اشاره به نرم‌افزار آزاد).
- بسیاری از سیستم عامل های مدرن، از جمله لینوکس، OS MAC ، اندروید، iOS ، سیستم عامل کروم و تمامی نسخه های مختلف لینوکس به سیستم عامل یونیکس متکی هستند (اشاره به قضیه Unix Like Operating Systems)
- به عنوان نمونه ای از کامپیوترهای نسل سوم می توان به PDP-11 اشاره کرد.

نسل چهارم (۱۹۸۰ تا کنون)

- شرکت مایکروسافت اولین سیستم عامل خود را در سال 1981 تحت عنوان DOS-MS ساخت.
- نسل چهارم سیستم های عامل شاهد ایجاد محاسبات شخصی بودند و بر روی کامپیوترهای شخصی مورد استفاده قرار می گرفتند. یک کامپیوتر شخصی به قدری مقرون به صرفه بود که این امکان را برای یک فرد فراهم می کرد تا بتواند برای استفاده شخصی یک کامپیوتر داشته باشد، در حالی که مینی کامپیوترها هنوز به اندازه ای قیمت دارند که فقط شرکت ها می توانند آن ها را داشته باشند.
- یکی از عوامل مهم در ایجاد محاسبات شخصی، تولد مایکروسافت و سیستم عامل ویندوز بود. نام ویندوز برای اولین بار در سال 1985 استفاده شد، زمانی که یک رابط کاربری گرافیکی ایجاد شد و به DOS-MS پیوست. ویندوز با انتشار نسخه های 95 ،98 ،10 ، 8 ، 7 ، XP و 11 امروزه تبدیل به بزرگ ترین سیستم عامل مورد استفاده در فناوری شد.
- اپل دیگر سیستم عامل اصلی است که در دهه 1980 ساخته شد. استیو جابز، یکی از بنیانگذاران اپل، اپل مکینتاش را ایجاد کرد که به دلیل این واقعیت که کاربر پسند بود، موفقیت بزرگی داشت.

عملیات سیستم عامل

سیستم های عامل به طور کلی وظایفی دارند و عملیاتی را انجام می دهند:

Process management	مدیریت پردازه
Memory Management	مدیریت حافظه
File System Management	مدیریت سیستم فایل
IO Management	مدیریت ورودی / خروجی
Mass-Storage Management	مدیریت ذخیره سازی انبوه
Protection and Security	حفاظت و امنیت
Virtualization	مجازی سازی

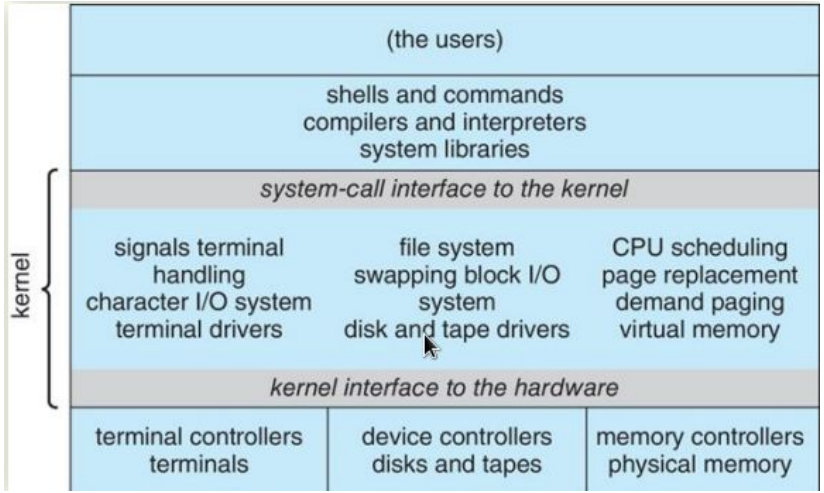
ساختار سیستم عامل

سیستم های عامل دارای ساختارهای متفاوتی هستند که به بررسی آنها می پردازیم:

- سیستم‌های یکپارچه Monolithic System
- سیستم‌های لایه‌ای Layered Systems
- سیستم‌های خادم و مخدوم Client/Server
- سیستم‌های جفت جفت Peer to Peer Kernel
- سیستم‌های محاسبات ابری Cloud Computing
- سیستم‌های ریز هسته Micro Kernel

سیستم‌های یکپارچه Monolithic System

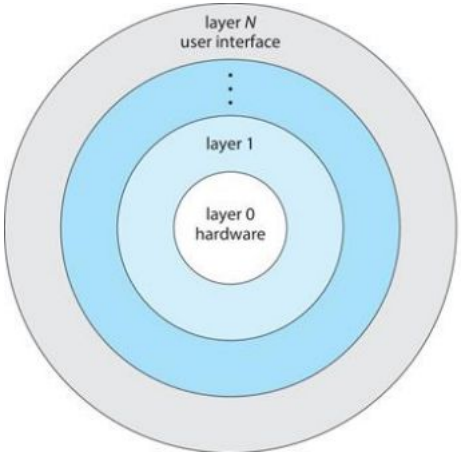
- برخی سیستم ها فاقد ساختار شناخته شده ای هستند. سیستم عامل DOS-MS یک نمونه از این سیستم ها می باشد.
- به دلیل محدود بودن سخت افزار، هدف از نوشتن این سیستم عامل عبارت بود از ایجاد بیشترین عملکرد و کارایی در کمترین فضا و با کمترین منابع
- یونیکس سیستم عامل دیگری است که با محدودیت سخت افزار مواجه می باشد. این سیستم شامل دو بخش است: برنامه های سیستم و هسته (Kernel) هسته مواردی نظیر سیستم فایل، زمانبندی پردازه ها، مدیریت حافظه و غیره را فراهم می کند.
- سیستم های یکپارچه را سیستم های ساده (Simple) نیز می نامند.
- ساختار یونیکس در شکل زیر نشان داده شده است:



سیستم‌های لایه‌ای Layered Systems

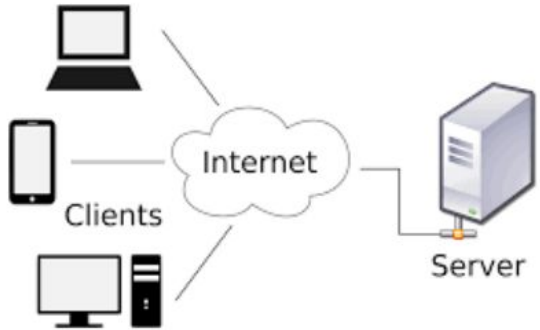
یک روش سازمان‌دهی کردن سیستم.

- ماژولار کردن یک سیستم می تواند به روش های مختلفی انجام شود. یکی از شیوه هایی که برای این منظور ارائه شده است، روش لایه بندی است.
- در روش لایه‌بندی، سیستم عامل به تعدادی لایه (سطح) شکسته می‌شود
- هر یک از این لایه‌ها در بالای لایه پایینی ساخته می‌شود
- پایین ترین لایه (لایه صفر)، سخت افزار و بالاترین لایه (لایه N)، واسط کاربر (User interface) می‌باشد
- مزیت اصلی سیستم‌های لایه‌ای قابلیت ماژولاریتی است



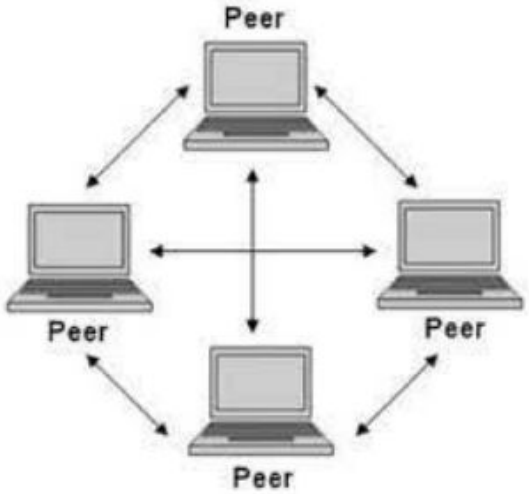
سیستم‌های خادم و مخدوم Client/Server

- در این نوع ساختار، سیستم ها به دو دسته تقسیم می شوند:
  - کلاینت
  - سرور
- Client ها درخواست خود را به server ها ارسال می کنند و server ها نیز به درخواست client ها پاسخ می دهند.
- سرورها دو دسته‌اند:
  - دسته اول: server-Compute: واسطی برای client ها فراهم می کنند که بتوانند سرویس ها را درخواست کنند مانند database
  - دسته دوم: File-server: واسطی برای Clientها فراهم می‌کنند که بتوانند ذخیره و بازیابی انجام دهند.



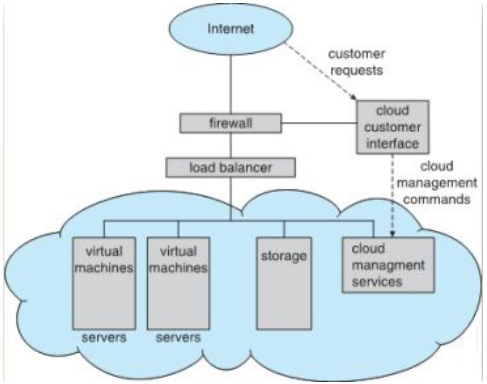
سیستم‌های جفت Peer to Peer Kernel

- مدل دیگری برای سیستم های توزیع شده مدل peer-to-peer می باشد.
- در این مدل، client ها و sever ها قابل تشخیص نیستند و تمام سیستم ها (گره ها) همتا در نظر گرفته می شوند.
- هر گره ممکن است به عنوان server، client و یا هر دو عمل کند.
- گره ها باید به شبکه P2P متصل باشند.
- Skype نمونه ای از این مدل می باشد.



سیستم‌های محاسبات ابری Cloud Computing

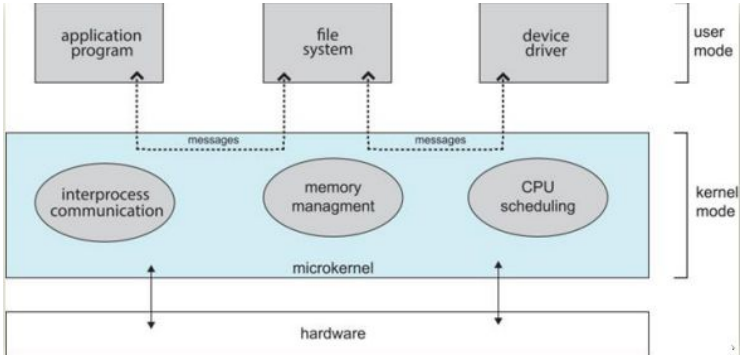
- این ساختار، محاسبات، ذخیره سازی، حتی برنامه ها را به عنوان یک سرویس در سراسر شبکه ارائه می دهد.
- این ساختار به طور گسترده ای از مجازی سازی استفاده می کند.
- برای مثال، آمازون EC2 دارای هزاران سرور، میلیون ها ماشین مجازی و چندین پتابایت فضای ذخیره سازی در سراسر اینترنت می باشد.
- این ساختار انواع مختلفی دارد:
  - Public Cloud: از طریق اینترنت برای هر کسی که مایل به پرداخت هزینه است، در دسترس می باشد.
  - Private Cloud: توسط یک شرکت و فقط برای استفاده خود شرکت اداره می شود.
  - نرم افزار به عنوان سرویس(SaaS): یک یا چند برنامه کاربردی از طریق اینترنت در دسترس هستند.
  - پلتفرم به عنوان سرویس (PaaS): پشته نرم افزار برای استفاده برنامه کاربری از طریق اینترنت آماده است. برای مثال، یک سرور پایگاه داده
  - زیرساخت به عنوان سرویس (IaaS): سرورها یا فضای ذخیره سازی از طریق اینترنت در دسترس هستند.



- محیط های محاسبات ابری متشکل از سیستم های عامل سنتی، به عالوه ماشین های مجازی، به عالوه ابزارهای مدیریت ابری می باشند.
- اتصال به اینترنت نیاز به امنیت مانند فایروال دارد.
- balancer Load ترافیک را در چندین برنامه پخش می کند.

سیستم‌های ریز هسته Micro Kernel

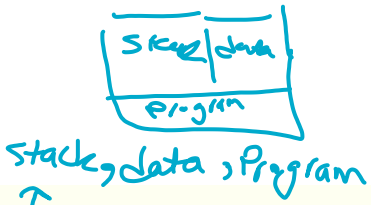
- این روش، با حذف اجزای غیرضروری از هسته و پیاده سازی آن ها به عنوان برنامه های کاربر و برنامه های سیستم، سیستم عامل را ساختاردهی می کند.
- ریزهسته عالوه بر شفافیت ارتباطات، حداقل مدیریت پردازش و حافظه را موجب می شود.
- ارتباطات توسط Passing Message انجام می شود.
- مزیت ریزهسته عبارتند از:
  - گسترش سیستم عامل به سادگی امکان پذیر است.
  - همه سرویس های جدید به فضای کاربر اضافه می شود و نیازی به اصالح هسته نیست.
  - سیستم عامل حاصل به راحتی از یک سخت افزار به سخت افزار دیگر منتقل می شود.
  - امنیت بیشتر است.
  - قابلیت اطمینان بیشتر است (کد کمتری در هسته اجرا می شود)



مفاهیم اولیه

- پردازه : برنامه در حال اجرا
- نخ
- فایل
- فراخوان سیستم
- پوسته
- بافر
- سیستم توزیع شده
- سیستم چند پردازنده
- سیستم بلادرنگ
- چند برنامگی
- اشتراک زمانی
- ثبات
- اجرای دستورالعمل
- وقفه
- حافظه نهان
- سلسله مراتب حافظه

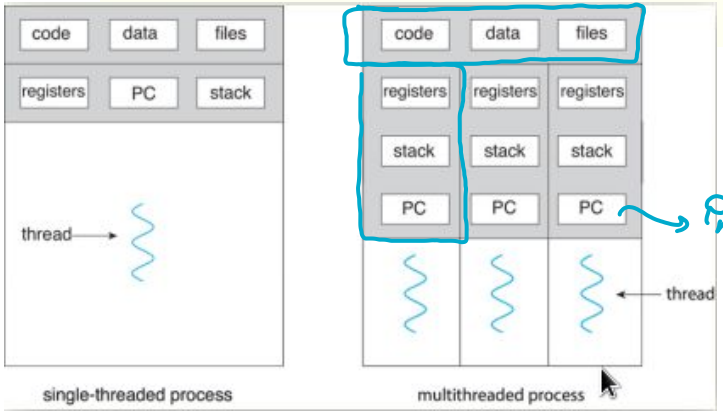
پردازه



- یکی از مفاهیم کلیدی در تمام سیستم‌های عامل، مفهوم پردازه می‌باشد.
- پردازه، یک برنامه در حال اجرا می‌باشد. به عبارت دیگر، پردازه موجودیتی است که می‌توان پردازنده را به آن اختصاص داد و آن را اجرا کرد.
- هر پردازه فضای آدرس خاص خود را دارد. فضای آدرس پردازه شامل برنامه قابل اجرا، داده برنامه و پشته می‌باشد.
- هر پردازه مجموعه ای از ثبات‌ها (شامل شمارنده برنامه، اشاره‌گر پشته و سایر ثبات‌های سخت افزاری) و دیگر اطلاعات مورد نیاز برای اجرای برنامه را در اختیار دارد.
- هر پردازه خود می‌تواند پردازه دیگری را ایجاد کند. پردازه تولید کننده را والد (parent) و پردازه تولید شده را فرزند (child) می‌نامیم.
- پردازه‌های وابسته به هم برای انجام برخی کارها نیاز دارند که با هم در ارتباط باشند. این ارتباط را Inter-process Communication می‌نامند.
- ارتباط بین پردازه‌ها را می‌توان به‌وسیله یک ساختار درختی نشان داد.
- پردازه در حال اجرا عناصر زیر را به همراه دارد:
  - شناسه (Identifier) : یک شناسه منحصر به فرد است که به هر پردازه داده می‌شود و آن را از بقیه متمایز می‌کند.
  - حالت یا وضعیت (State) : اگر پردازه در حال اجرا باشد، در وضعیت Running قرار دارد.
  - اولویت _ (Priority) : سطح و میزان اولویت پردازه را نسبت به سایر پردازه‌ها مشخص می‌کند.
  - شمارنده برنامه (Program Counter) : آدرس دستورالعمل بعدی را در برنامه در حال اجرا مشخص می‌کند.
  - اشاره‌گر حافظه (Memory Pointer) : شامل اشاره‌گرهایی به کد برنامه و داده‌ها می‌باشد.
  - اطلاعات وضعیت ورودی/خروجی (IO Status Information) : شامل درخواست‌های ورودی/خروجی خاص و دستگاه‌های ورودی/خروجی مورد نیاز پردازه می‌باشد. همچنین شامل لیست فایل‌های به‌کار رفته توسط پردازه نیز می‌باشد.
  - اطلاعات حسابداری (Accounting Information) : ممکن است شامل میزان زمان پردازنده، زمان ساعت سیستم و .. باشد.
- این اطلاعات در ساختار داده‌ای به نام بلوک کنترل پردازه (PCB – Process Control Block) ذخیره می‌شود که توسط سیستم عامل ایجاد و مدیریت می‌گردد.
- در سیستم‌های تک پردازنده‌ای در هر لحظه فقط یک پردازه می‌تواند در حال اجرا باشد.

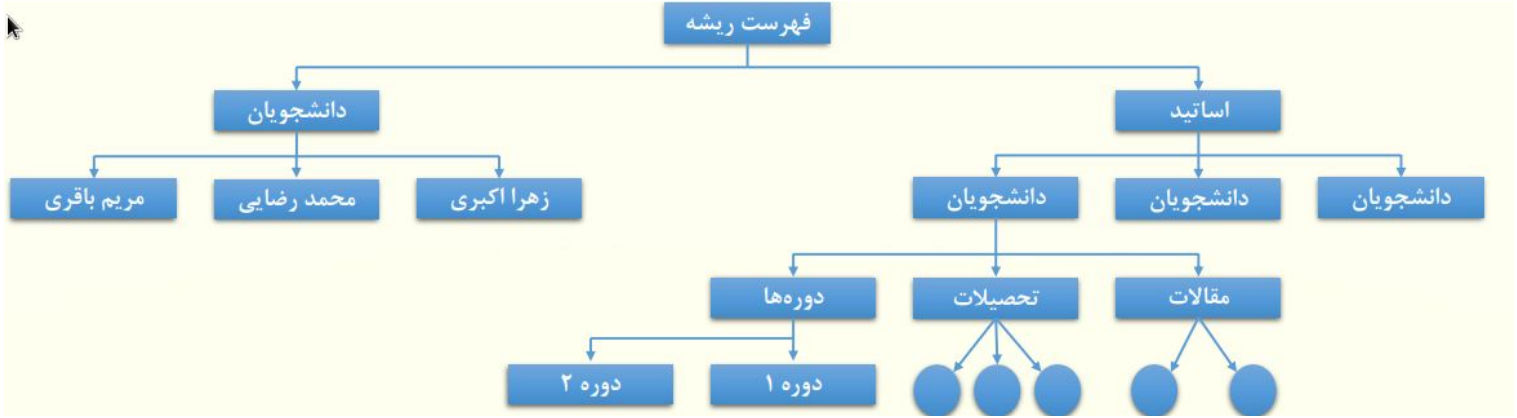
نخ یا ریسه (Thread)

- در سیستم‌های عامل سنتی، هر پردازه یک فضای آدرس و یک رشته کنترلی دارد که این رشته کنترلی را نخ یا ریسه می‌نامیم. البته برخی به آن پردازه سبک وزن (Light weight Process) نیز می‌گویند.
- هر پردازه می‌تواند فقط شامل یک نخ باشد یا اینکه چندین نخ داشته باشد.
- **ویژگی‌ها و خصوصیات نخ :**
  - نخ‌های مرتبط با یک پردازه از فضای آدرس، کد برنامه و منابع سیستم عامل یکسان و مشترک برخوردار هستند.
  - اجرای یک نخ در داخل یک وظیفه (Task) انجام می‌شود.
  - هر نخ ثبات‌ها، شمارنده برنامه و پشته خاص خود را دارد.
  - جابجایی بین نخ‌ها در پردازه بسیار راحت‌تر و سریع‌تر از جابجایی بین پردازه‌ها می‌باشد.
  - ایجاد نخ جدید در یک پردازه عملی آسان و کم هزینه است.
  - نخ‌ها می‌توانند به صورت همگان و یا ناهمگام با یکدیگر اجرا شوند.



فایل (File)

- یکی از وظایف سیستم عامل پنهان کردن جزئیات کار دیسک و سایر دستگاه‌های ورودی/خروجی است. اطلاعات و داده‌های کاربران در قالب ساختاری خاص بر روی دیسک ذخیره می‌شود. این ساختار را فایل می‌نامند.
- سیستم عامل برای ایجاد، باز کردن، خواندن، نوشتن، بستن و یا حذف فایل‌ها یکسری فراخوان‌های سیستمی (System Call) در اختیار دارد.
- برای دسته بندی فایل‌ها می‌توان از ساختار فهرست (Directory) استفاده کرد. هر فهرست می‌تواند شامل چندین فایل و یا فهرست دیگر باشد.
- برای ایجاد و یا حذف فهرست‌ها و همچنین قراردادن فایل‌ها در فهرست‌ها و یا حذف فایل‌ها از فهرست‌ها فراخوان‌های سیستمی خاصی وجود دارد.
- در یک سیستم می‌توان نحوه قرار گرفتن فایل‌ها و فهرست‌ها و ارتباط آن‌ها با یکدیگر را توسط یک ساختار درختی نمایش داد.

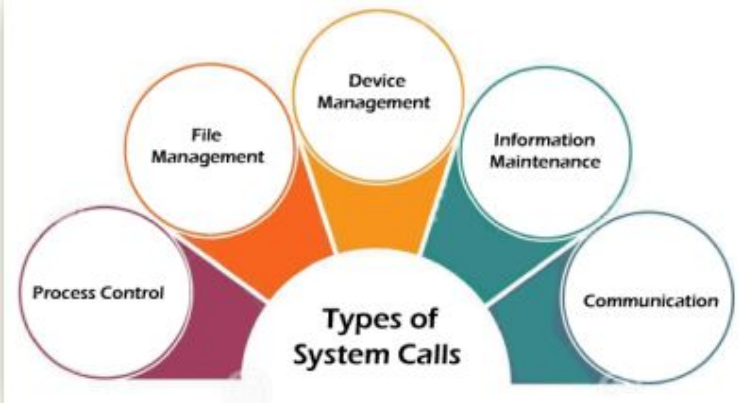


فراخوان سیستم (System Call)

فراخوان سیستم، تابعی است که با اجرای آن، کار خاصی انجام می شود. سیستم عامل برای انجام هر یک از امور خود، فراخوان های سیستمی خاص خود را دارد.

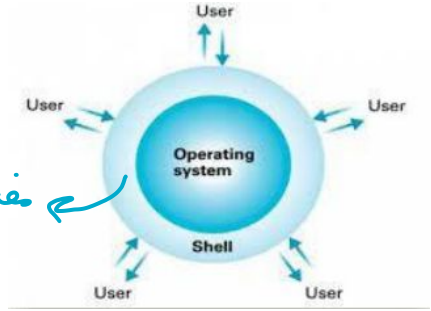
فراخوان های سیستم را می توان به صورت زیر دسته بندی کرد:

- فراخوان های سیستمی برای مدیریت پردازه
- فراخوان های سیستمی برای فایل ها و فهرست ها
- فراخوان های سیستمی برای مدیریت دستگاه ها
- فراخوان های سیستمی برای نگهداری اطلاعات
- فراخوان های سیستمی برای ارتباطات



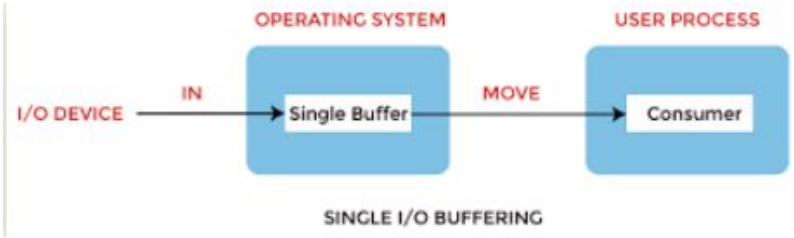
پوسته (Shell)

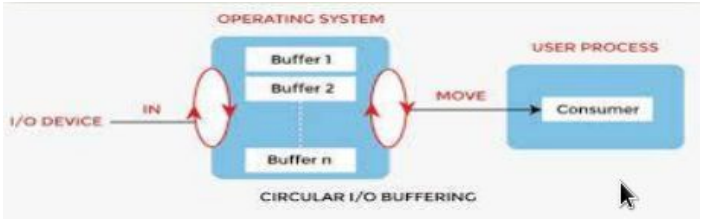
- سیستم عامل کدی است که فراخوان های سیستم را انجام می دهد. ویراستارها، کامپایلرها، اسمبلرها و مفسرهای فرمان قطعاً جزئی از سیستم عامل نیستند
- ولی بسیار مهم و سودمند هستند.
- پوسته در واقع یک مفسر فرمان است که دستورات صادره را تفسیر می کند. با وجود این که جزئی از سیستم عامل نیست، در اکثر سیستم های عامل به کار
- می رود. چگونگی اجرای فراخوان های سیستم به پوسته وابسته است.
- همچنین پوسته واسط بین کاربر و سیستم عامل است، مگر این که کاربر از یک واسط گرافیکی استفاده کند.
- برخی از پوسته های موجود عبارتند از : zsh ، ksh ، csh و bash.
- وقتی یک کاربر وارد سیستم می شود، یک پوسته راه اندازی می شود. پوسته ترمینالی به عنوان ورودی و خروجی استاندارد دارد.
- برای مثال، اگر کاربر دستور date را تایپ کند، پوسته یک پردازه فرزند ایجاد می کند و برنامه date را به عنوان فرزند اجرا می کند. پوسته منتظر می ماند
- تا پردازه فرزند که در حال اجرا می‌باشد به پایان برسد و terminate شود. با پایان کار پردازه، پوسته سعی می کند ورودی بعدی را بخواند.



بافر (Buffer)

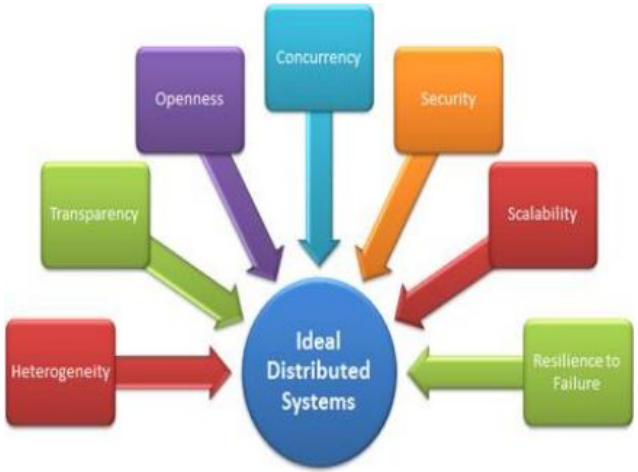
- عملیات ورودی/خروجی کند بوده و کارایی سیستم را کاهش می دهند. با استفاده از حافظه های میانی (بافرها)، عملیات ورودی/خروجی یک زیر برنامه با اجرای آن همزمان می شود. این عمل برای کارهایی که عملیات ورودی/خروجی مناسبی دارند سودمند است.
- عموماً کارها به سه دسته تقسیم می شوند:
  - کارهای Bound O/I: کارهایی هستند که بخش زیادی از آن ها در ارتباط با دستگاه های ورودی/خروجی می باشد و محاسبات و پردازش کمتری دارند.
  - کارهای Bound CPU: کارهایی هستند که محاسبات و پردازش زیادی دارند و عملیات ورودی/خروجی آن‌ها کم است.
  - کارهایی که عملیات ورودی/خروجی و حجم محاسبات و پردازش آن ها متعادل و متناسب است.
- در کارهای Bound O/I و Bound CPU استفاده از بافر چندان تاثیری در بهبود کارایی سیستم ندارد.

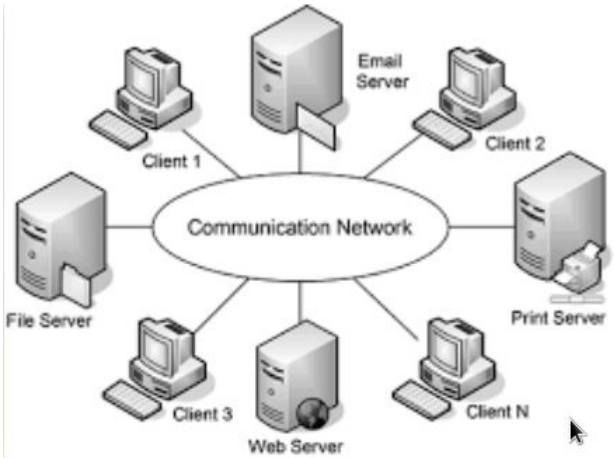




سیستم‌های توزیع شده (Distributed Systems)

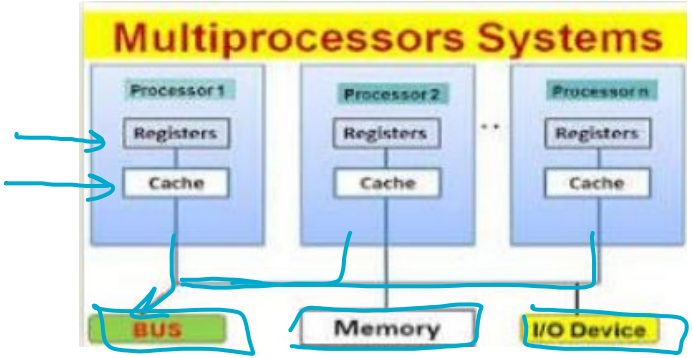
- سیستم های توزیع شده از دید کاربر همانند یک سیستم تک پردازنده ای هستند در صورتی که چنین نیست.
- در این نوع سیستم های چندین کامپیوتر در محل های مختلف قرار دارند و از طریق شبکه با هم در ارتباط هستند.





سیستم‌های چند پردازنده (Multiprocessor Systems)

- سیستم های تک پردازنده فقط یک CPU دارند ولی سیستم های چندپردازنده دارای چندین CPU هستند.
- در سیستم چندپردازنده، هر پردازنده ثبات ها و حافظه نهان خاص خود را دارد ولی از خطوط Bus ، حافظه و دستگاه های ورودی/خروجی به طور مشترک استفاده می کنند.



- مقایسه سیستم های چند پردازنده و سیستم های توزیع شده:

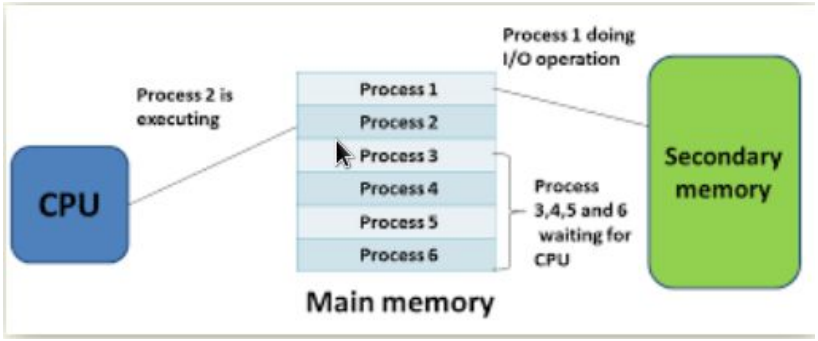
	Multiprocessor Parallel System	Distributed Parallel System
Scalability	Harder	Easier
Programmability	Easier	Harder
Degree of transparency	High	Very high
Architecture	Homogeneous	Heterogeneous
Basic of communication	Shared memory	Message passing

سیستم‌های بلادرنگ (Real-Time System)

- این سیستم‌ها باید در بازه زمانی معین، سرویس دهی لازم را داشته باشند و اگر عملیات در زمان معین انجام نشود، سودی ندارد و ممکن است خسارت‌های غیرقابل جبرانی به همراه داشته باشد.
- **انواع سیستم‌های بلادرنگ :**
  - ❖ **نوع اول : Hard Realtime :**
    - در این نوع سیستم‌ها، ضرب الاجل بسیار دقیق انجام می‌شود.
    - در واقع باید وظیفه داده شده در زمان برنامه ریزی شده مشخص اجرا شود و در مدت زمان تعیین شده تکمیل گردد.
    - مثال : سیستم مراقبت‌های ویژه پزشکی، سیستم‌های هواپیما و غیره.
  - ❖ **نوع دوم : Firm Realtime :**
    - این نوع سیستم‌ها نیز باید از مهلت‌های زمانی پیروی کنند.
    - با این حال، از دست دادن ضرب الاجل ممکن است تاثیر زیادی نداشته باشد، اما می‌تواند اثرات نامطلوب، مانند کاهش شدید کیفیت یک محصول، ایجاد کند.
    - مثال : انواع مختلف برنامه‌های چندرسانه‌ای
  - ❖ **نوع سوم : Soft Realtime :**
    - این نوع سیستم‌ها برخی تاخیرها را توسط سیستم می‌پذیرند.
    - در اینجا نیز برای هر کار خاص مهلت تعیین شده است اما تاخیر برای مدت کمی قابل قبول است.
    - بنابراین ضرب الاجل توسط این نوع سیستم‌ها به آرامی انجام می‌شود.
    - مثال : سیستم معاملات آنلاین

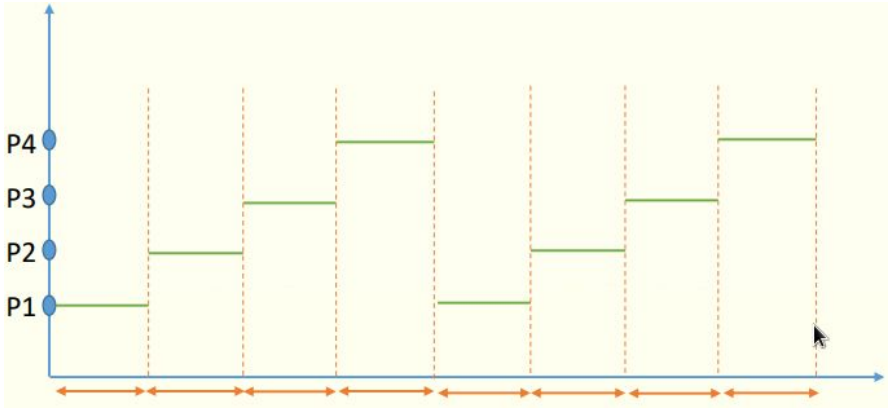
چند برنامگی (Multi Programming)

- در این سیستم ها، چندین برنامه می توانند در آن واحد در سیستم وجود داشته باشند و به طور همزمان اجرا شوند.
- البته منظور از همزمان، انجام پردازش و استفاده از CPU نیست، زیرا در سیستم های تک پردازنده ای در هر لحظه فقط یک برنامه می تواند CPU را در اختیار داشته باشد.
- منظور از اجرای همزمان این است که وقتی پردازه در حال اجرا به عملیات ورودی/خروجی نیاز داشته باشد CPU از آن گرفته شده، در اختیار پردازه دیگر قرار می گیر. در این لحظه هر دو پردازه در حال اجرا هستند ولی یکی مشغول عملیات ورودی/خروجی و دیگری مشغول پردازش است.



اشتراک زمانی (Time Sharing)

- اشتراک زمانی حالت خاصی از چند برنامگی است که در آن تعویض پردازه ها بر اساس یک معیار زمانی معین انجام می شود، نه نیاز به عمل ورودی/خروجی.
- به هر پردازه به اندازه یک بازه زمانی معین پردازنده اختصاص داده می شود.
- اگر کار پردازه در این بازه به پایان نرسد باید پردازنده از آن گرفته شده، در اختیار پردازه دیگر قرار گیرد.
- در این حالت پردازه قبلی باید منتظر بماند تا دوباره نوبت به او برسد و بتواند کارش را ادامه دهد.

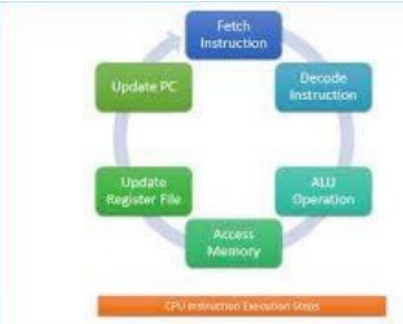


ثبات (Register)

- هر پردازنده شامل مجموعه ای از ثبات ها می باشد.
- ثبات، حافظه ای سریع تر و کوچک تر از حافظه اصلی است.
- ثبات های پردازنده از نظر وظایفی که بر عهده دارند به دو دسته کلی تقسیم می شوند:
- دسته‌اول:ثبات‌هایقابل‌رویتبرای‌کاربر:
  - برنامه نویسان زبان اسمبلی می توانند با استفاده مناسب از این ثبات ها، مراجعه به حافظه اصلی را به حداقل برسانند.
    - ثبات های داده: برنامه نویس می تواند این ثبات ها را به برخی توابع نسبت دهد.
    - ثبات های آدرس: این ثبات ها حاوی آدرس دستورالعمل ها و داده ها در حافظه هستند.
- دسته‌دوم:ثبات‌هایوضعیت‌وکنترل:
  - پردازنده از این ثبات ها برای کنترل عملیات خود استفاده می کنند.
  - برخی از این ثبات ها عبارتند از:
    - ثبات آدرس حافظه (MAR - Memory Address Register)
    - ثبات بافر حافظه (MBR - Memory Buffer Register)
    - ثبات آدرس ورودی/خروجی (I/OAR - Input/Output Address Register)
    - ثبات بافر ورودی/خروجی (I/OBAR - Input/Output Buffer Register)
    - ثبات شمارنده برنامه (PC - Program Counter): آدرس دستورالعمل بعدی که باید واکشی شود
    - ثبات دستورالعمل (IR - Instruction Register): آخرین دستورالعملی که باید واکشی شود
    - ثبات کلمه وضعیت برنامه (PSW - Program Status Word): اطلاعات وضعیت برنامه

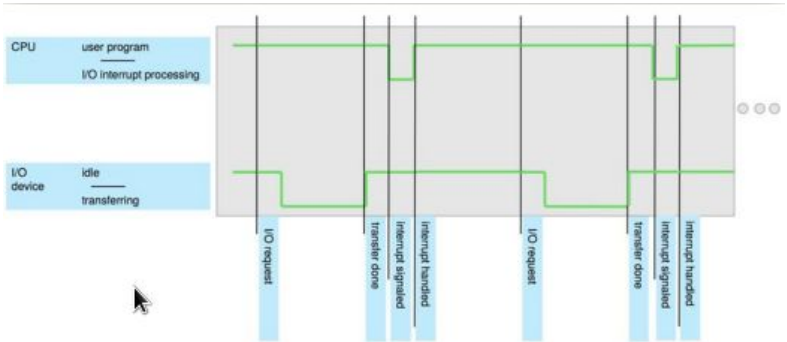
اجرای دستورالعمل (Instruction Execution)

- هر برنامه ای که توسط پردازنده اجرا می شود، شامل مجموعه ای از دستورالعمل هایی است که در حافظه ذخیره شده اند.
- پردازش دستورالعمل طی مراحل زیر انجام می شود :
  - مرحله اول: پردازنده یک دستورالعمل را از حافظه واکشی می کند (Fetch)
  - مرحله دوم: دستورالعمل رمزگذاری شده در IR توسط رمزگشا تفسیر می شود (Decode)
  - مرحله سوم: دستورالعمل واکشی شده را اجرا می کند (Execute)
  - مرحله چهارم: ذخیره نتایج در حافظه
  - مرحله پنجم: به روز رسانی ثبات ها
  - مرحله ششم: به روز رسانی شمارنده برنامه



وقفه (Interrupt)

- هدف از وقفه، استفاده بهینه از وقت پردازنده است. وقتی در حین کار عادی سیستم دچار وضعیت غیرعادی می شود، وقفه ای رخ می دهد و عملیات عادی سیستم مختل می شود.
- همیشه بعد از وقوع وقفه تغییر حالت پردازنده و احتمالا در برخی مواقع تعویض پردازه انجام می شود.
- نحوه کار وقفه در عملیات ورودی/خروجی به صورت زیر است:
  - پردازنده با راه‌اندازی دستگاه جانبی، اطلاعاتی در مورد آنچه باید نوشته یا خوانده شود، در ثبات های دستگاه ورودی/خروجی قرار می دهد و تقاضای خود را به دستگاه جانبی اطلاع می دهد.
  - بدون بروز وقفه، پردازنده کار خود را ادامه می دهد و دستگاه جانبی عملیات ورودی/خروجی را آغاز می کند.
  - دستگاه جانبی پس از انجام عملیات خواندن یا نوشتن، پایان کار خود را با ارسال سیگنالی به پردازنده اطلاع می دهد.
  - پردازنده در اولین فرصت، کار انتقال اطلاعات دستگاه جانبی را جهت پردازش انجام می دهد.
  - عملکرد وقفه در محیط چند برنامگی بسیار موثر است، زیرا در هنگام نیاز یک پردازه به عملیات ورودی/خروجی، پردازنده بیکار نبوده و پردازه دیگری را اجرا می کند

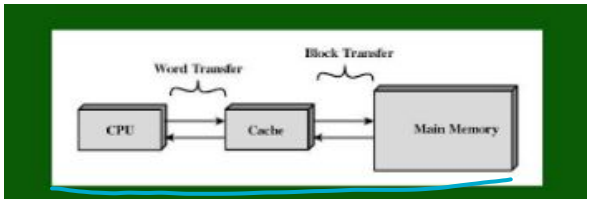


حافظه نهان (Cache)

اگر چه حافظه نهان قابل رویت نیست ولی با دیگر سخت افزارهای مدیریت حافظه در تعامل می باشد.

هدف از ایجاد حافظه نهان، ارائه حافظه ای به مراتب سریع تر از حافظه اصلی است.

در هنگام پردازش بخشی از حافظه اصلی به حافظه نهان منتقل می شود و دسترسی به آن بخش سریع تر انجام می گردد.



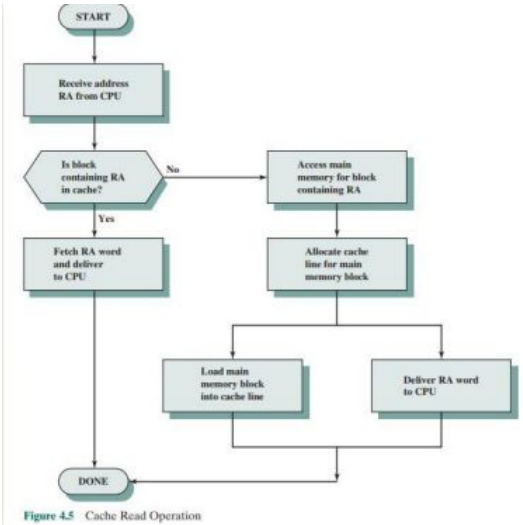
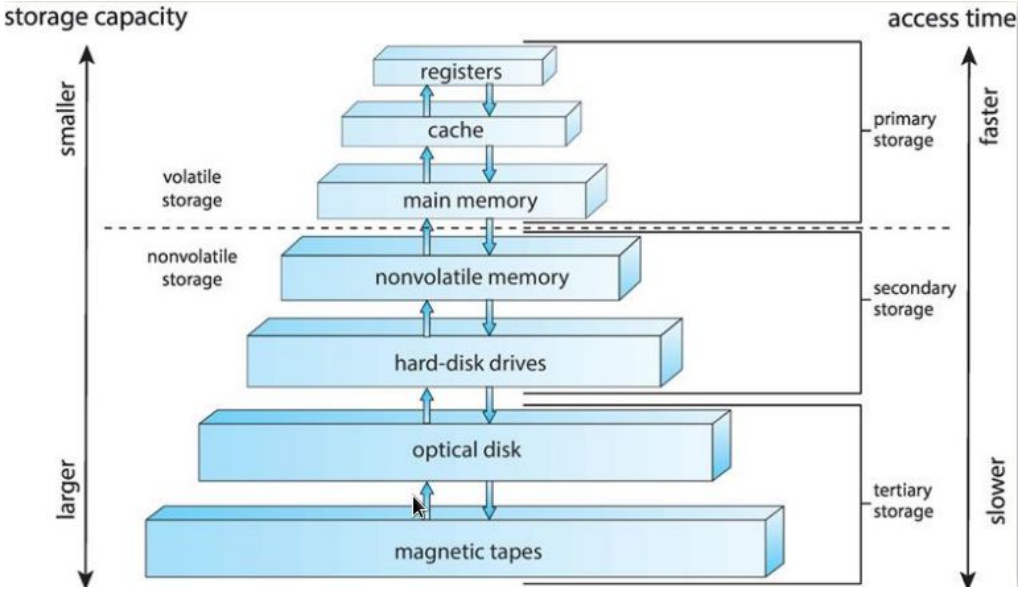


Figure 4.5 Cache Read Operation

سلسله مراتب حافظه

شکل زیر سلسله مراتب انواع حافظه را نشان می دهد:



پایان درس اول