

به نام ایزد دانا

**WWW.SOFTECH.IR**

## **جزوه درسی سیستم عامل**

( ویژه داوطلبان آزمون کادانی به کارشناسی - نرم افزار )

**منبع:**

کلیه حقوق این جزوه متعلق به وب سایت کارشناسی دات کام میباشد

هرگونه کپی برداری بدون ذکر سایت فوق ممنوع است.

**www.karshenasi.Com**

## 1- نگاه کلی به سخت افزار

عناصر اصلی

1-1- پردازنده

2-1- حافظه اصلی

3-1- دستگاه های ورودی / خروجی

4-1- اتصالات داخلی سیستم شامل ساختارها و راهکارهایی که ارتباط بین اجزاء ( پردازنده - حافظه - دستگاه های ورودی / خروجی )

1-1-1- ثبات پردازنده

الف- ثباتهای قابل دسترس توسط کاربر : این ثباتها در دسترس تمام برنامه های کاربردی و برنامه های سیستم  
ب- ثبات های کنترلی وضعیت: این ثباتها برای کنترل عملیات پردازنده بکار گرفته می شود و قابل دسترسی بوسیله کاربران نیستند..مانند ثبات دستوالعمل ، شمارنده برنامه ( PC ) ، ثبات کلمه وضعیت ( PSW )

## وقفه (interrupt)

نکته 1: وقفه توسط سخت افزار کامپیوتر تولید می شود هر چند علت آن می تواند نرم افزار باشد.

نکته 2: یک هدف عمده ، از راهکار وقفه ، افزایش کارائی پردازنده می باشد.

نکته 3: در هنگام اجرای خود هسته نیز وقفه اتفاق می افتد. CPU آنها را آشکار کرده و به جای پشته درون جدول ، فرآیند از پشته هسته استفاده می شود.

نکته 4: امکان وقوع وقفه های تو در تو نیز وجود دارد و هرگاه روال سرویس دهی به وقفه بعدی خاتمه می یابد ، می توان وقفه قبل از آن را تکمیل نمود.  
{تنباومر 169}

در حالت کلی سه نوع وقفه در کامپیوتر وجود دارد:

1- وقفه خارجی

2- وقفه داخلی

3- وقفه نرم افزاری

## 1- وقفه خارجی

الف- زمان سنج (Timer): وقفه ای که توسط تایمر داخلی پردازنده تولید می شود. این وقفه به سیستم عامل اجازه می دهد ، بعضی اعمال را به طور مرتب انجام دهد. در واقع به منظور تعیین زمان اجرای پردازنده در هر وهله کاری  
ب- وقفه تکمیل ورودی - خروجی: هنگامی تولید می شود که ورودی - خروجی کامل شده باشد یا خطائی در آنها رخ داده باشد.

ج- وقفه سخت افزار : وقفه ای است که بر اثر بروز خطا در سخت افزار ماشین رخ می دهد. مثل نقص برق

د- وقفه Restart: وقفه ای است که بر اثر فشار دادن دکمه Reset بر روی کنسول ایجاد می گردد.

## 2- وقفه داخلی (تله های حفاظتی و استثنا، Trap):

بر اثر خطا های مختلف در برنامه کار بران رخ می دهد . این وقفه ها تله (Trap) نیز نامیده می شود مانند تقسیم بر صفر ، سرریز شدن محاسباتی ، تلاش برای اجرای یک دستوالعمل ماشین غیر مجاز و مراجعه به آدرسی خارج از فضای مجاز کاربر

مثال : الف- فرض کنید برنامه ای می خواهد خلاف کند به حافظه دیگران دسترسی داشته باشد.

مثال : ب- می خواهیم وقفه ها را از کار بندازیم .

مثل : ج- هارد دیسک را فرمت کنیم

Trap چیست؟ هر گونه رفتن از مد کاربر به مد هسته را تله گویند.

چه کسی Trap را کشف میکند؟

الف- هسته      ب- پوسته      ج- CPU      د- Compiler

### 3- وقفه نرم افزاری :

این وقفه ها همان فراخوانی سیستمی (System Call) نامیده می شوند دستوراتی از برنامه کاربر هستند که با فراخوانی آنها نیاز به استفاده از حالت ناظر( ) و امکانات آن فراهم می گردد مثل در خواست اجرا ی یک ورودی - خروجی که با یک وقفه به ناظر (سیستم) واگذار می شود.

جدول بردار وقفه ( Vector Table):

بخشی از مکانیسم وقفه شامل یک جدول بردار می باشد که آدرس حافظه سرویس دهنده های وقفه را نگه می دارد . جدول آدرس به عنوان مکانی مطمئن برای نگهداری آدرس ها ی مهمی که کاری با وقفه ندارند ، استفاده می شود. برای هر کدام از این آدرس ها یک شماره وقفه متناظر وجود دارد ، اما تا وقتی که روتین سرویس دهنده وقفه موجود نباشد ، هرگز نمی تواند استفاده شود.

وقفه به عنوان مکانیسمی استفاده می شوند که کامپیوتر را به دنیای اطرافش متصل می کند. وقفه ها کامپیوتر را به گردش در می آورند، زیرا تمام کارها به شکل وقفه ها به کامپیوتر وارد می شوند. مهمتر اینکه تمام سازمان دداخلی کامپیوتر بر اساس وقفه طراحی می شود. عامل کنترل لی که تعیین می کند توجه پردازنده در کجا متمرکز خواهد شد.

### مکانیسم وقفه :

هر وقفه یک شماره دارد که نوع وقفه را مشخص میکند . به عنوان مثال یک شماره وقفه برای دیسک گردانها استفاده می شود ( تمام گرداننده ها دارای یک وقفه می باشند ) ساعت، صفحه کلید ، چاپگر ها نیز هر کدام یک شماره مخصوص دارند، برای هر شماره وقفه یک برنامه مخصوص به نام سرویس دهنده وقفه ( Interrupt handler ) جدول مخصوص که در اوایل حافظه کامپیوتر قرار دارد آدرس کامل سرویس دهنده های وقفه را ضبط می کند . هنگامیکه وقفه ای رخ می دهد ، از شماره وقفه برای پیدا کردن برنامه سرویس دهنده وقفه استفاده می شود . اما پیش از اینکه سرویس دهنده وقفه شروع به کار کند ، که مکانیسم پردازش وقفه ، رکوردی از وضعیت کاری که در جریان بود را بر روی پشته ذخیره می کند . بعد از اجرای وقفه ، کنترل پردازنده از روتین سرویس دهنده وقفه انتقال می یابد . سرویس دهنده وقفه اغلب در برنامه تعبیه شده ROM\_BIOS یا به عنوان بخش از سیستم عامل ظاهر میشوند.

## تفاوت سیستم عامل با ROM\_BIOS چیست؟

سیستم عامل بطور مستقیم کاربر را سرویس دهی میکند در صورتیکه ROM\_BIOS چنین کاری را انجام نمی دهد. تفاوت دیگر در این است که ROM\_BIOS سرویس های خود را در سطحی پایین تر برای برنامه ها عرضه می نماید. در صورتیکه بیشتر سرویس های سیستم عامل سرویس های پیچیده ایی است و در سطحی بالا عرضه می گردد. کنترل کننده دیسک دستوالعمل را از BIOS یا درایور ها به سیگنال الکتریکی ترجمه می کند که باعث حرکت هد خواندن / نوشتن روی مکان دیسک می شود. این عمل باعث ایجاد سیگنالهای مغناطیسی شده و بدین ترتیب داده ها ی یک سند روی سطح دیسک ثبت و ضبط می شود.

BIOS را میتوان یک نرم افزار خاص برای برقراری ارتباط میان سیستم عامل و اجزای سخت افزاری نامید که معمولا بر روی یک حافظه از نوع Flash بر روی مادر برد ذخیره میشود ولی گاهی اوقات از نوع ROM نیز ساخته میشود که تفاوت این دو با هم در این است که BIOS از نوع ROM قابلیت تغییر یافتن را ندارد ولی نوع Flash را میتوان به روز رسانی کرد .

به طور کلی برای BIOS می توان چهار وظیفه مهم را بر شمرد :

1. می توان گفت مهمترین وظیفه BIOS بارگذاری سیستم عامل است . همانطور که می دانید سیستم عامل بر روی هارد دیسک ذخیره شده است و هنگامی که کامپیوتر را روشن میکنیم و ریز پردازنده تلاش میکند تا اولین دستور را اجرا کند باید این دستورات را از جایی بخواند و اجرا نماید و مسلما از سیستم عامل نمی تواند این کار را انجام دهد، BIOS این دستورات را فراهم میکند .
2. تست خودکار تمام اجزای سخت افزاری سیستم جهت اطمینان از صحت کارکرد آنها .
3. فعال کردن سایر BIOS های موجود در کارت های مختلفی که بر روی سیستم نصب است.
4. BIOS علاوه بر کارهای فوق یکسری زیر برنامه ها ی سطح پایینی را هم در اختیار قرار می دهد که سیستم عامل برای برقراری ارتباط با برخی سخت افزار ها نیاز دارد مثلا صفحه کلید ، صفحه نمایش ، پورت های سریال و موزی را کنترل میکند،مخصوصا هنگام بوت شدن سیستم !!  
(همین زیر برنامه ها نام BIOS – Basic Input/Output System- را به آن داده اند )

هنگامی که **سیستم روشن می شود BIOS** اعمال مختلفی را انجام می دهد که ما در اینجا یک روال معمول را به شما گوشزد میکنیم :

1. بررسی تنظیمات دستی CMOS (Complementary Metal-Oxide Semiconductor): اولین کاری که BIOS انجام می دهد بررسی اطلاعات ذخیره شده بر روی یک RAM 64 بایتی به نام CMOS می باشد که جزئیات سیستم را تامین میکند و میتواند با توجه به مشخصات سیستم تغییر یابد ، BIOS به این اطلاعات برای اجرای صحیح دستورات خود نیاز دارد .
2. بار گذاری کنترل کننده های وقفه و راه اندازی سخت افزاری: کنترل کننده وقفه قطعه نرم افزاری کوچکی است که به عنوان مترجم میان سخت افزار و سیستم عامل عمل میکند به عنوان مثال هنگامی که یک کلید بر روی صفحه کلید فشرده میشود یک سیگنال به کنترل کننده وقفه فرستاده شده که به **CPU** می گوید که چه کلیده فشرده شده و آنرا به سیستم عامل ارسال میکند .  
راه انداز های سخت افزار ، قطعات نرم افزاری هستند که اجزای سخت افزاری پایه از جمله صفحه کلید و ماوس و هارد و فلاپی و را تعریف می کند که برای افزایش سرعت بر روی **RAM** بارگذاری می شوند . بد از بار گذاری کنترل کننده وقفه نوبت به کارت گرافیک می رسد، در این هنگام بررسی می شود که آیا کارت گرافیک فعال است یا خیر ؛ در صورت فعال بودن **BIOS** کارت گرافیک ، مادربرد این BIOS را بارگذاری میکند .
3. مقدار دهی اولیه ثبات ها و مدیریت جریانهای الکتریکی داخل سیستم
4. انجام تست خود کار هنگام روشن شدن (POST)  
سیس BIOS چک میکند که آیا بوت شدن ، یک بوت شدن سرد است (Cold Boot) یا بوت مجدد (ReBoot) . اگر از نوع دوم باشد بایوس اعمال مربوط به POST را دیگر انجام نمی دهد زیرا قبلا یک بار انجام گرفته اند . اما اگر بوت سرد باشد BIOS حافظه RAM را با انجام تست خواندن /نوشتن بر روی هر آدرس حافظه بررسی میکند . سپس BIOS پورت های PS/2 و USB را برای صفحه کلید و ماوس چک میکند و در صورت وجود گذرگاه PCI ( Peripheral Component Interconnect ) همه کارت های PCI را بررسی میکند اگر بایوس هنگام **POST** به مشکل بر بخورد این خطا با بوق یا متن به کاربر اطلاع داده میشود .
5. نمایش تنظیمات دستی : بعد از مراحل فوق BIOS برخی جزئیات سیستم را نمایش میدهد از جمله :
  - پردازنده
  - فلاپی درایو و هارد درایو
  - حافظه
  - نسخه و تاریخ ساخت BIOS
  - نوع سخت افزار نمایشی
6. تعیین اینکه کدام دستگاهها قابل بوت شدن هستند
7. شروع کردن روال نوار بوت : بعد از نمایش جزئیات ، BIOS راه انداز قطعاتی مانند SCLS (**Small Computer System interface** ) را که همراه خود قطعه است را بارگذاری میکند . پس از طی این مراحل ، برای اجرای سیستم عامل BIOS ترتیب قطعاتی را که به عنوان قطعات بوت کننده در CMOS انتخاب شده اند را بررسی میکند و سعی میکند تا روال بوت شدن را از اولین قطعات پیدا کند و اگر هیچ قطعه ای را پیدا نکند روند بوت قفل میشود .  
این آخرین فعالیت BIOS در هنگام بوت شدن می باشد پس از آن در صورت یافتن سیستم عامل ، مدیریت سیستم به دست سیستم عامل سپرده میشود .

#### :ROM\_BIOS

مجموعه ای از برنامه های تعبیه شده در کامپیوتر می باشد که اصلی ترین و سطح پایین ترین عملیات کنترل و نظارت را برای کامپیوتر انجام می دهند. ROM\_BIOS در پایین ترین لایه قرار دارد لایه ای که در زیر تمام نرم افزار ها و عملیات در جدار کردن سایر برنامه ها از جزئیات نحوه کار سخت افزار می باشد. ROM\_BIOS بطور اساسی یک رابط ، متصل کننده ، و مترجم بین سخت افزار کامپیوتر و برنامه های نرم افزاری می باشد. نرم افزار هایی اینچنین چون بطور ثابت در تراشه های سخت افزاری ROM ذخیره می شوند **FIREWARE** گفته می شوند آنچه را که ROM\_BIOS باید انجام دهد کنترل مستقیم سخت افزار و پاسخ به هر تقاضایی از سوی سخت افزار است . نحوه انجام اینکار به مقدار زیاد ی توسط در گاهها انجام می گیرد.

مجموعه از برنامه های مهم و اصلی پشتیبان کلی عملیات کامپیوتر در ROM\_BIOS نگهداری می گردند. این برنامه ها از دویخیش اصلی تقسیم می شوند

1- بخش اول فقط در هنگام روشن شدن کامپیوتر استفاده می شوند : این گروه برنامه های تست و مقدار دهی اولیه هستند که مطمئن می شوند کامپیوتر در وضعیت کاری خوبی قرار دارد. تاخیری که بین زمان روشن کردن کامپیوتر تا شروع کار آن وجود دارد توسط عملیات این برنامه های تست و مقدار دهی یا **POST ( Power On Self Test )** بوجود می آید.

2- بخش دوم گروهی از روتینها به نام سرویس های اصلی ورودی / خروجی یا **( Basic Input /Output Service )** BIOS می باشد. این برنامه ها کنترل درونی و مفصلی را بر روی بخشهای مختلف کامپیوتر ، مخصوصاً وسایل ورودی / خروجی ، مانند دیسک گردانها فراهم می کنند.

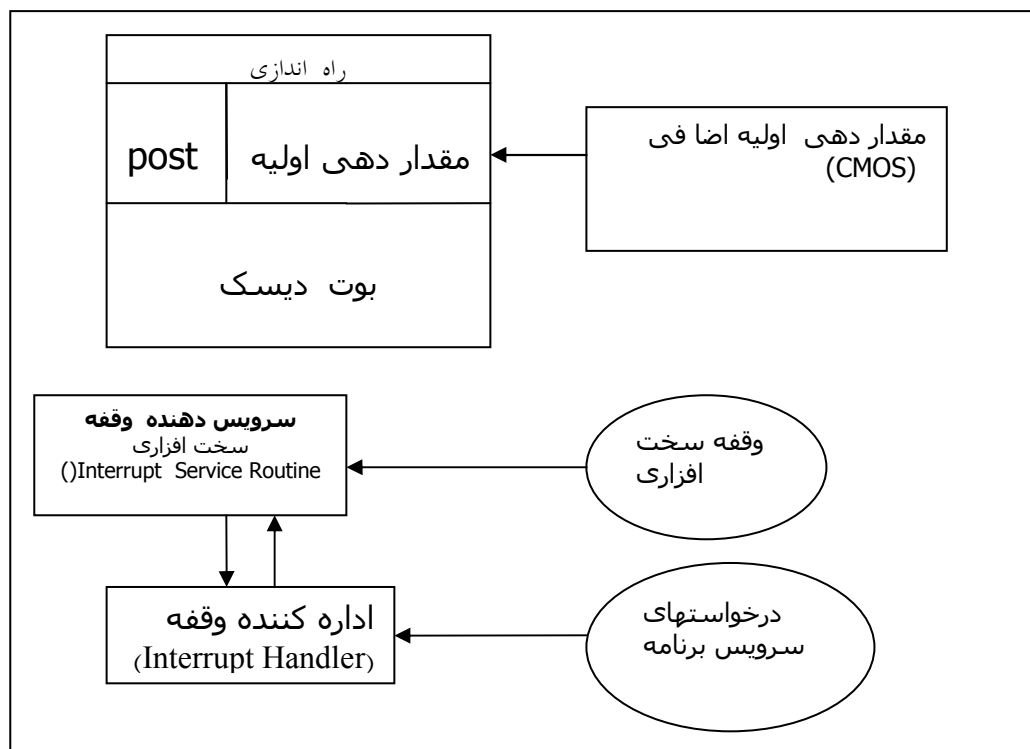
گرداننده ها( ) احتیاج به نظارت دقیقی (شامل نظارت جامع برای خطا ها ) دارند.

ROM-BIOS برای کمک به کل عملیات کامپیوتر سرویس های متعدد و مفیدی را ارائه می کند که قابل دسترس برای استفاده توسط سیستم عامل (Dos,windows)

و برنامه های کاربردی می باشند.آخرین بخش روتین های راه اندازی در ROM\_BIOS روتین بوت می باشد که سعی میکند سیستم عامل را در کامپیوتر بوت کند. عمل بوت تلاش ROM\_BIOS برای خواندن رکورد بوت کننده از آغاز دیسک را در بر دارد.

دو بخش دیگر ROM\_BIOS بخش اصلی عملیات اجرایی کامپیوتر را شکل می دهد. این دو بخش سرویس دهنده وقفه سخت افزار (– hardware interrupt handler) و دستگذاری سرویس ( service – handling ) می باشد. آنها به عنوان دو نوع روتین مجزا اما همکار عمل می کنند.

3-بخش سوم ROM-BIOS که فقط برای اعضا خانواده PC ساخت IBM به کار می رود.

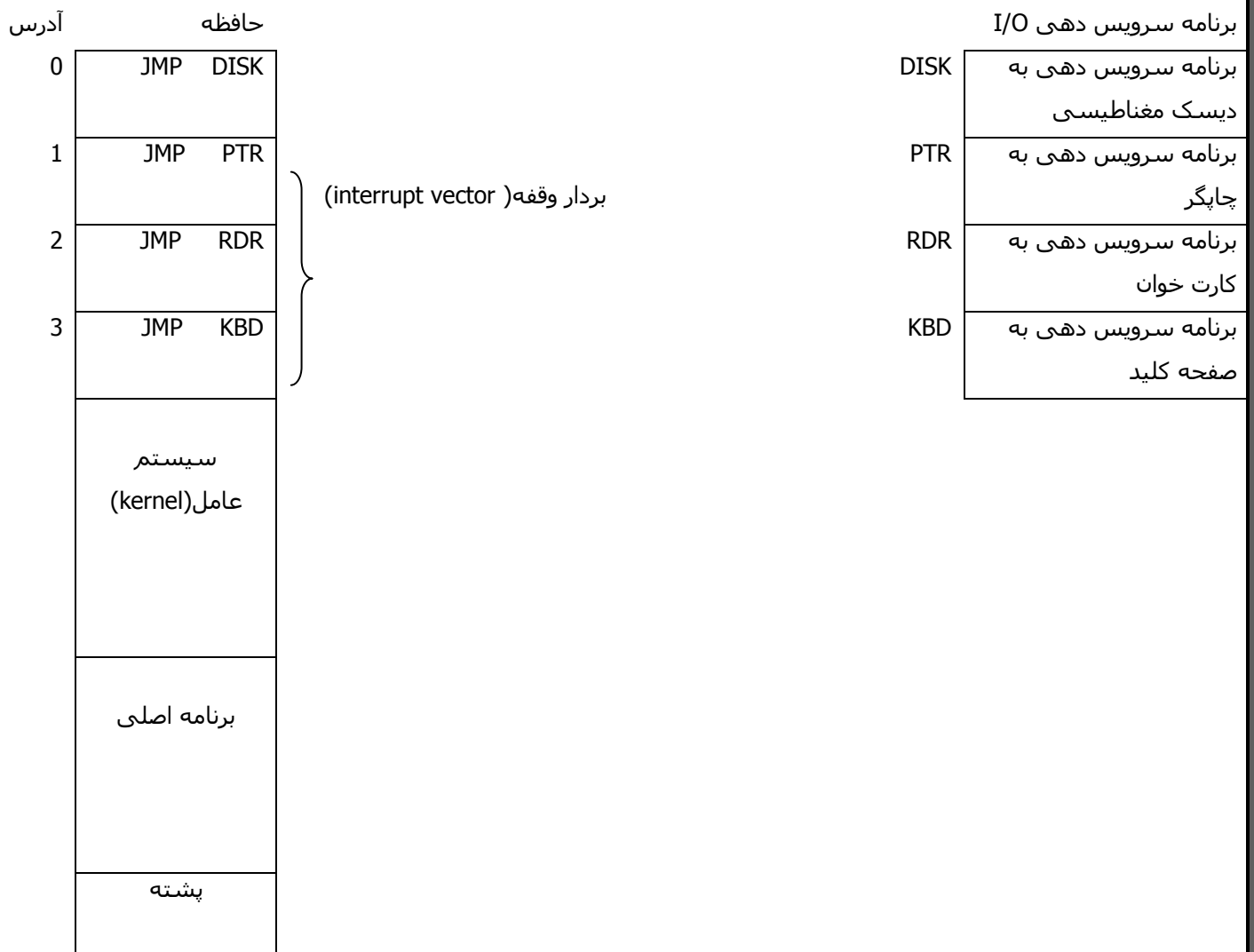


## حافظه اصلی کامپیوتر

- 1- اولین بخش جدول بردار (Interrupt Vector) می باشد که محل روتین های سرویس دهنده وقفه را تعریف می کند. این جدول آدرس 0 تا 400 هگزا را اشغال می کند.
- 2- دومین فضا به عنوان محل کار روتین های ROM\_BIOS استفاده می شود. از بین چیز های زیادی که در فضای داده ROM\_BIOS ذخیره می شود. به یک BUFFER که ضربه های کلید را پیش از اینکه برنامه برای دریافت آنها حاضر باشد در خود نگه می دارد .
- 3- سومین بخش فضای پایین حافظه فضای کار کامپایلرها و سیستم عامل می باشد.
- 4- بخش اصلی فضای کاری حافظه قسمتی است که برنامه ها و داده های آنها استفاده می شود.

هر وسیله برنامه سرویس دهی خاص خود را دارد و از طریق یک دستورالعمل (JMP) ، آدرس بردار قرار دارد قابل دسترسی است. مثال:

فرض کنید که صفحه کلید بیت وقفه را 1 کند. در پایان سیکل دستورالعمل ، کامپیوتر به سیکل وقفه میرود. ابتدا آدرس بازگشت برنامه در حال اجرا را در پشته ذخیره می کند و سپس CPU آدرس بردار 00000011 (3) را از گذرگاه پذیرفته و آنرا به ثبات PC (ثبات شمارنده) انتقال میدهد. آنگاه دستورالعمل واقع در مکان 3 اجرا شده ، و در نتیجه کنترل به روال KBD (برنامه سرویس دهی به صفحه کلید) منتقل می شود. آخرین دستور در هر برنامه ، دستورالعمل بازگشت از وقفه است. { معماری کامپیوتر - مورس مانو }



در رابطه با هر کلاس از دستگاه های I/O در قسمت پایینی حافظه ناحیه ای به نام بردار وقفه ( Interrupt ) وجود دارد که شامل آدرس روتین سرویس وقفه ( Interrupt Service Procedure ) می باشد. فرض کنید فرآیند 3 در حال اجرا است که ناگهان یک وقفه در رابطه با دیسک اتفاق می افتد . این وقفه باعث می شود که PC (شمارنده برنامه )، PSW(کلمه وضعیت برنامه ) و چند رجیستر دیگر توسط وقفه سخت افزاری در پشته (جاری) قفزار گیرند. سپس کامپیوتر به آدرس مشخص شده در بردار وقفه ( Interrupt Verctor ) پرش میکند. تا این مرحله کاملاً توسط سخت افزار انجام می شود ولی از این مرحله به بعد به عهده نرم افزار است. روتین سرویس وقفه با ذخیره سازی کلیه رجیستر ها در درابه ای از جدول فرآیند که مربوط به فرآیند جاری است آغاز می شود. اعمالی نظیر ذخیره سازی رجیستر ها و تغییر اشاره گر پشته (به راحتی) در زبان C انجام نمی شود. بنابراین این موارد توسط رویه کوچکی که به زبان اسمبلی نوشته شده است انجام می شود. { تنبناوم 65 }

## روشهای انتقال ورودی / خروجی {معماری کامپیوتر - موریس مانو}

تبادل داده با وسایل جانبی به سه طریق زیر امکان پذیر است.

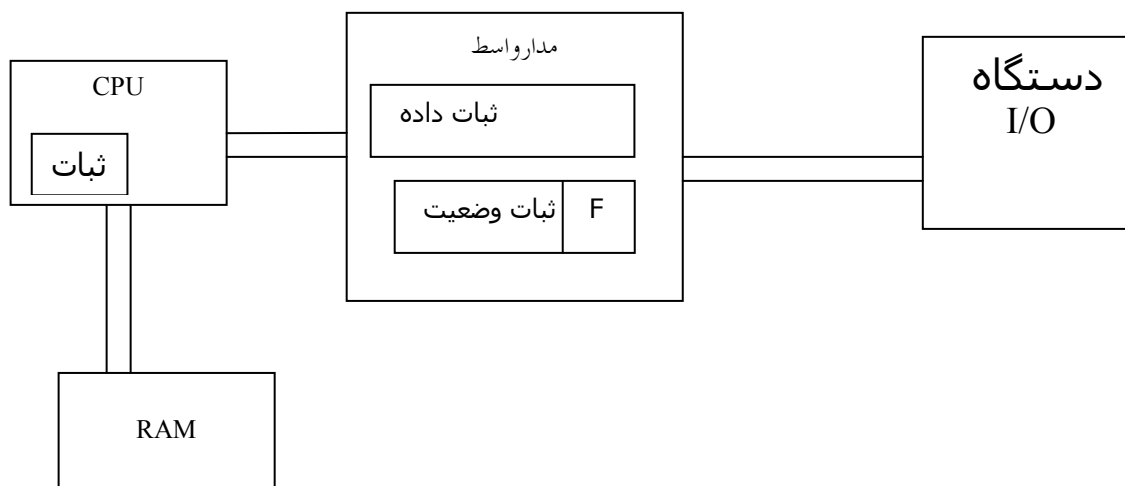
- 1-ورودی خروجی برنامه نویسی شده
- 2-ورودی /خروجی بطریقه وقفه
- 3-دستیابی مستقیم به حافظه (DMA (Direct Memory Access

### 1-ورودی /خروجی برنامه سازی شده

در این روش ، CPU در یک حلقه از برنامه باقی می ماند تا واحد I/O مشخص نماید که برای انتقال داده آماده است . این فرآیند موجب اتلاف قابل توجه وقت می شود. زیرا پردازنده را بی آنکه لازم باشد مشغول نگه می دارد . برای جلوگیری از بهدر رفتن وقت پردازنده از امکانات وقفه ( Interrupt ) استفاده کرد. در این روش انتقال داده بین CPU و دستگاه جانبی انجام می شود. در این روش ، برنامه ای برای کامپیوتر نوشته شده است تا پرچم را در ثبات وضعیت واریسی نموده و تعیین کند که آیا وسیله I/O بایستی در ثبات قرارداده است یا خیر . این عمل با خواندن ثبات وضعیت و قراردادن آن در یک ثبات CPU و واریسی بیت پرچم صورت می گیرد. اگر پرچم برابر 1 باشد ، CPU داده را از ثبات داده می خواند . سپس بیت پرچم با 0 شدن به وسیله ، CPU ، پاک می شود. .

بعد از صدور فرمان لازم برای اجرای ورودی / خروجی به مولفه مربوطه ، عمل درخواست شده توسط مولفه ورودی / خروجی انجام میشود و بیتهای مناسب از ثبات وضعیت ورودی / خروجی مقدار گذاری میشود و هیچ اطلاعی به پردازنده داده نمی شود. این مسئولیت پردازنده است که وضعیت مولفه ورودی / خروجی را متناوباً بررسی نماید تا اتمام آن عمل را دریابد .

F : بیت پرچم



## 2- ورودی / خروجی بطریقه وقفه

بعد از صدور فرمان به مولفه ورودی / خروجی ، پردازنده به کار مفید دیگری میپردازد . بعد از تکمیل عمل ورودی / خروجی هنگامی که داده ها در بافر داده های آن مولفه قرار گرفت مولفه ورودی / خروجی از طریق یک خط کنترلی به پردازنده وقفه میدهد، سپس داده هایش توسط پردازنده ، درخواست شده و به حافظه منتقل می گردد. در این روش CPU مادامی که مشغول اجرای یک برنامه است پرچم را واریسی نمی کند. با این وجود ، هنگامیکه پرچم 1 شده باشد، یک وقفه CPU را لحظه ای از اجرای برنامه جاری متوقف ساخته و 1 شدن پرچم را به آن اطلاع می دهد. CPU از آنچه در حال انجام آن است منصرف شده و به انتقال ورودی یا خروجی می پردازد. پس از تکمیل شدن انتقال ، کامپیوتر به برنامه اصلی بر می گردد تا آنچه را که قبل از وقفه انجام می داد ادامه دهد.

## 3- ورودی / خروجی مبتنی بر وقفه همراه با مولفه DMA

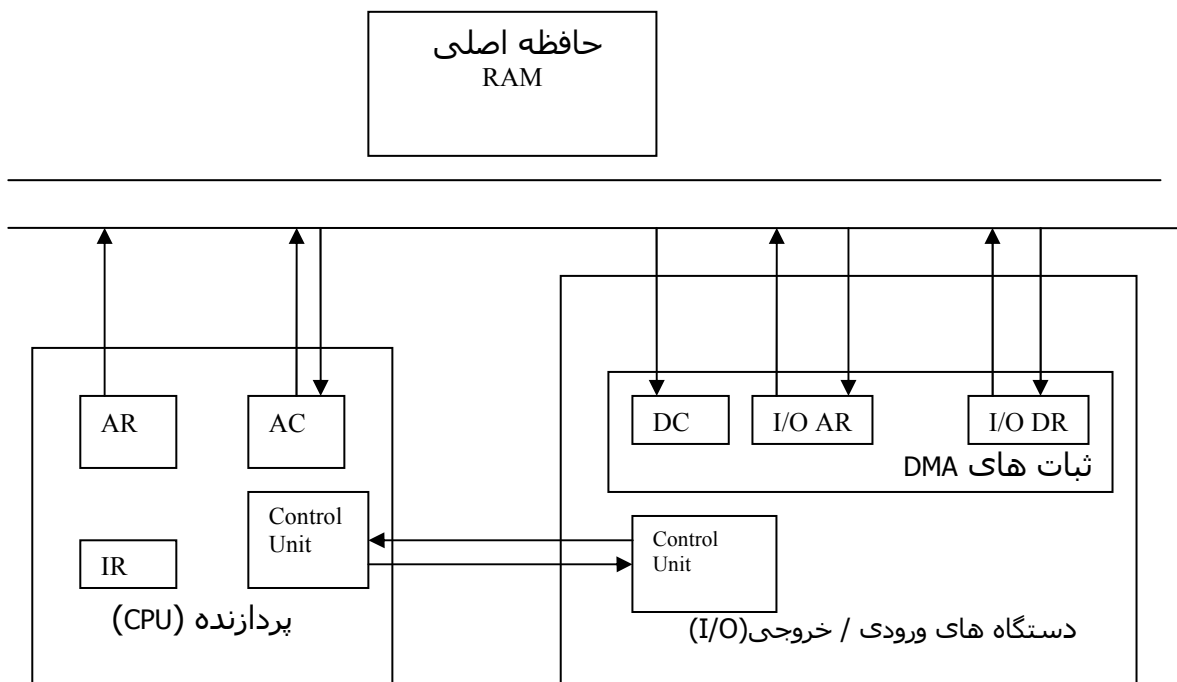
انتقال داده بین یک وسیله ذخیره سازی سریع مانند دیسک مغناطیسی و حافظه اغلب بوسیله سرعت CPU محدود می شود. حذف CPU از مسیر و ایجاد امکان کنترل مستقیم گذرگاه حافظه توسط وسیله جانبی سرعت انتقال را بهبود می بخشد. این تکنیک انتقال دستیابی مستقیم به حافظه (DMA) نام دارد در حین انتقال بروش DMA ، CPU بیکار است و کنترلی بر گذرگاه های حافظه ندارد . یک کنترل کننده به منظور اداره مستقیم امر انتقال بین حافظه و وسیله I/O ، کنترل گذرگاه را بدست می گیرد.

در این روش که برای انتقال داده های حجیم کارآمد تر است ، پردازنده با ارسال اطلاعات شامل ، عمل ورودی / خروجی ، آدرس دستگاه ورودی / خروجی ، محل حافظه برای خواندن یا نوشتن و تعداد کلماتی که باید خوانده ویا نوشته شود ، مولفه DMA را فعال میسازد . سپس پردازنده به کار دیگری می پردازد ، زمانی که انتقال ورودی / خروجی به اتمام رسید مولفه DMA علامت وقفه را صادر می نماید . بنابراین بدون دخالت پردازنده داده ها مستقیماً به حافظه منتقل می شود. پس نتیجه می گیریم که برای انتقال ورودی / خروجی های چند کلمه ای DMA بسیار کارآمد تر از روش های ورودی / خروجی برنامه سازی شده یا مبتنی بر وقفه است.

توجه نمایند که مولفه DMA به منظور انتقال اطلاعات به حافظه یا بالعکس می بایست از گذرگاه استفاده نماید.

بنابراین مولفه DMA باید زمانی از گذرگاه استفاده نماید که پردازنده به آن نیازی ندارد و یا باید پردازنده را وادار کند که موقتاً عملیاتش را با تاخیر اندازد . روش اخیر را سرفقت سیکل ( Cycle Stealing ) می گویند زیرا مولفه DMA در اصل یک چرخه گذرگاه را می رباید . مولفه DMA هر بار که گذرگاه را در اختیار می گیرد (زمان واکنشی دستور و یا واکنشی عملوند ) یک کلمه را منتقل می کند و کنترل را به پردازنده باز می گرداند توجه نماید که این وقفه نیست ، بلکه به اندازه یک چرخه گذرگاه منتظره می ماند. CPU صرفاً عملکرد خود را به اندازه یک سیکل حافظه به تاخیر می اندازد تا عمل انتقال حافظه I/O به طور مستقیم صورت گرفته و یک سیکل حافظه را بریابد.





## مفاهیم سیستم عامل

- به طور کلی نرم افزارهای کامپیوتر به دو گروه تقسیم می‌شوند:
- 1- یکی برنامه‌های سیستمی که عملیات کامپیوتر را مدیریت می‌کنند
  - 2- دیگری برنامه‌های کاربردی .

سیستم عامل ( **operating system = os** ) اصلی ترین برنامه سیستمی است که به عنوان رابط بین کاربر و سخت افزار کامپیوتر عمل می‌کند .

سیستم عامل دو وظیفه (یا هدف) اصلی دارد:

1- سیستم عامل ستفاد هاز 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 حذف کند . این کار در واقع با به کاربردن دستورات ساده‌ای که فراخوان های سیستمی ( **System Calls** ) را صدا می‌زنند انجام پذیرد.

در صورت عدم وجود سیستم عامل کاربر یا برنامه نویس می‌بایست آشنایی کاملی با سخت افزارهای مختلف کامپیوتر (مثل مونیتر، فلاپی ، کی بورد و غیره) داشته باشد و روتین‌هایی برای خواندن و یا نوشتن آنها به زبانهای سطح پائین بنویسد. از این جنبه به سیستم عامل با عنوان ماشین توسعه یافته ( **Extended machine** ) یا

نکته :ماشین مجازی ( **Virtual machine** ) یاد می‌شود که واقعیت سخت افزار را از دید برنامه نویسان مخفی می‌سازد.

2- وظیفه دوم سیستم عامل مدیریت منابع ( **Resource Management** ) می‌باشد ،یعنی سیستم عامل باعث استفاده بهینه و سودمند ( اقتصادی ) از منابع سیستم می‌گردد .منظور از منابع پردازنده‌ها ،حافظه‌ها ،دیسکها،ماوس ها ، چاپگرها ، فایلها ، پورتها و غیره هستند. یک سیستم کامپیوتری منابع نرم افزاری و سخت افزاری بسیار دارد که ممکن است در حین اجراء برنامه لازم باشند ، سیستم عامل همانند مدیر منابع عمل کرده و آنها را بر حسب نیاز به برنامه‌های مشخصی تخصیص می‌دهد .

سیستم عامل معمولاً اولین برنامه‌های است که پس از بوت شدن در حافظه بار می‌شود. پس از بار شدن قسمتی از سیستم عامل بطور دائم در حافظه باقی(Resident) می‌ماند. قسمتهای دیگر با توجه به کاربرد کامپیوتر توسط کاربر از دیسک به حافظه آورده می‌شود .

به قسمت اصلی سیستم عامل که وظایف مهم آن را انجام می‌دهد هسته یا Kernel گفته می‌شود.

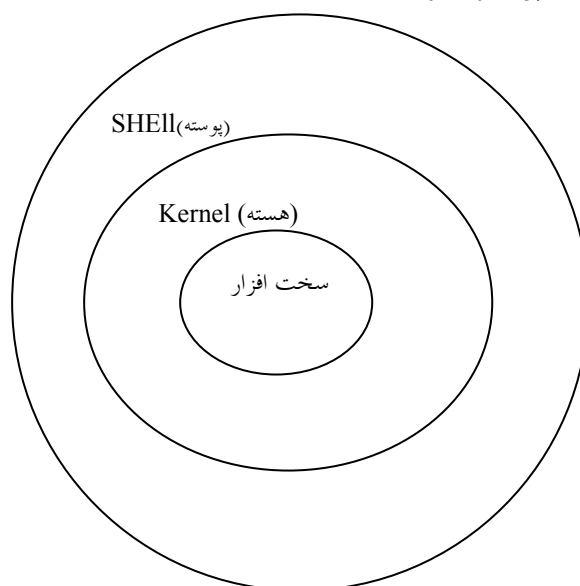
هسته سیستم عامل برنامه‌ای است که در تمامی اوقات بر روی کامپیوتر در حال اجراست . سیستم عامل و معماری کامپیوتر اثر زیادی بر روی یکدیگر داشته‌اند. یعنی جهت سهولت کار با سخت افزارهای جدید، سیستم عامل‌ها توسعه یافتند و همچنین در اثبات طراحی سیستم عامل‌ها ، مشخص شد که تغییراتی در طراحی سخت افزار می‌تواند سیستم عاملها را ساده تر و کارآمدتر سازد .

هر چند که تطبیق نسلهای کامپیوتر با نسلهای سیستم عامل کار درستی نیست ولی این تطبیق که در ادامه انجام می‌دهیم علت ایجاد سیستم عاملهای جدید را مشخص می‌سازد.

سیستم بانکی	رزرواسیون هوایی	مرورگر وب	نرم افزار های کاربردی
مترجم ها	ویرایشگر ها	مفسر های فرمان	نرم افزار های سیستمی
سیستم عامل			
ریز برنامه نویسی (BIOS) در ROM			سخت افزار
دستگاه های فیزیکی			

سیستم عامل از دو بخش مهم تشکیل شده است

- 1- هسته (Kernel)
- 2- پوسته (Shell)

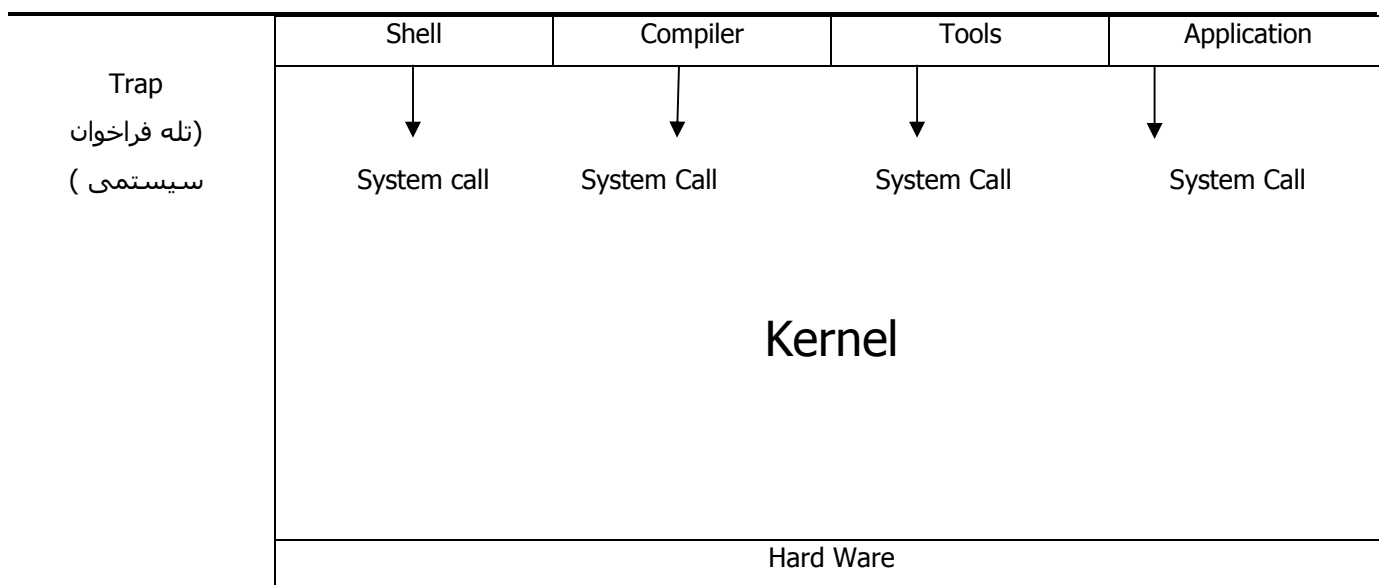


**هسته سیستم عامل** : قلب سیستم عامل که کلیه وظایف مدیریتی سیستم عامل را بر عهده دارد و واسط بین پوسته (و سایر نرم افزار های سطح کاربر )، سخت افزار است.

**پوسته سیستم عامل** : فقط واسط بین کاربر ( نه برنامه های کاربردی ) با هسته را دارد و فرامین کاربر را گرفته در صورت صحیح بودن آنها یکی از توابع درون هسته را برای انجام آن درخواست فراخوانی می کند.

پوسته های سیستم عامل

- 1 Command Driver مثل DOS
- 2 Menu driver مثل DOS SHELL
- 3 Icon Driver مثل Windows



نکته : در واقع سیستم عامل خود هسته است. پوسته تقریباً ارزش علمی ندارد. **Interrupt Handler** ها و **Device Driver** ها همه در Kernel هستند.

## فراخوان سیستمی ( System Call ) چیست؟

هرگاه یک نرم افزار سطح کاربر نیاز به دسترسی به منابع سیستم داشته باشد و سخت افزار یکی از **Function** های درون سیستم را فراخوانی می کند. که به این عمل فراخوان سیستمی ( **System Call** ) می گویند.

## فرق System call با یک فراخوان Function عادی چیست؟

- همه در سطح کاربرند ولی در حالت فراخوان سیستم باید مد آن تغییر کند از مد کاربر به مد هسته برود . چرا مد آن عوض می شود ؟ برای امنیت ، حدود اختیارات برنامه های سطح کاربر خیلی کمتر از سطح هسته است .
- نکته : وقتی برنامه در سطح کاربر انجام میشود و بخواهد خلاف کند سیستم عامل وجود ندارد تا جلوییش را بگیرد .

به سطح هسته ، سطح راهبری ( **Supervisor Mode** ) هم گفته می شود..  
در **Unix** فراخوان سیستمی ( ) وجود دارد به نام: Fork()  
Fork() که یک فرآیند فرزند می سازد و عین پدرش به شماره فرزند بر می گردد و به فرزند شماره صفر .  
و صفر شماره هیچ فرآیندی نیست یعنی فرزند است.

If fork() !=0

پدر

Else

فرزند

نکته : Source همه Function ها در هسته است.

#### نکته مهم :

سیستم عامل، نرم افزاری است که در مود هسته ( **Kernel mode** ) یا راهبری (Supervisor mode) کار می کند. که این بخش توسط سخت افزار (CPU) در مقابل مداخله کاربران محافظت می شود.

مترجم ها و ویرایشگر ها در مود کاربر (User Mode) اجرا می شوند.

کاربر حق ندارد که اداره کننده وقفه ( Interrupt handler ) دیسک را که بخشی از سیستم عامل است تغییر دهد. ولی کاربر آزاد است که برای خودش یک مترجم بنویسد.

## اهداف سیستم عامل به عنوان مدیر منابع چیست؟

- 1- استفاده بهینه از منابع
- 2- تخصیص و آزاد سازی منابع
- 3- زمانبندی فرآیند ها برای استفاده از منابع
- 4- به اشتراک گذاری منابع
- 5- حفاظت و ایجاد امنیت: جلوگیری از دسترس فرآیند ها به یکدیگر . ایجاد لیست ها و فهرست و طبقه بندی منابع و نام گذاری آنها مثل ایجاد فایل ها ، درخت و لیست های مختلف
- 6- طبقه بندی منابع و نام گذاری آنها در درخت ها
- 7- جلوگیری از بن بست ، قحطی ، تداخل

## زبان های پیاده سازی سیستم عامل

سیستم عاملهای اولیه به زبان اسمبلی نوشته می شدند ولی امروز، اکثر سیستم عاملها به زبان **C** یا **C++** نوشته می شوند . سیستم عامل UNIX OS/2، و ویندوز بیشتر به زبان C نوشته شده اند و قسمت اندکی از آنها به زبان اسمبلی است.  
مهمترین مزیت استفاده از زبان سطح بالا برای پیاده سازی سیستم عامل قابلیت حمل آن بر روی انواع کامپیوترها و سادگی پیاده سازی ، تغییر و بسط دادن سیستم عامل می باشد.

ممکن است ادعا شود پیاده سازی سیستم عامل به زبان C باعث کاهش سرعت و افزایش مصرف حافظه می گردد . اگر چه یک برنامه نویس ماهر زبان اسمبلی ، می تواند برنامه های کوچک و بسیار بهینه بنویسد ولی برای برنامه های بزرگ یک **کامپایلر** خوب، می تواند تحلیل پیچیده تری نسبت به مغز انسان ماهر انجام داده و بهینه سازی های کاملی را انجام دهد .

لذا در عمل برنامه های بزرگ C کد اسمبلی بهینه تر و کمتری را تولید می کنند، نسبت به حالتی که برنامه نویس بخواهد همان کاری به زبان اسمبلی انجام دهد . از طرف دیگر در عمل کارایی اصلی نتیجه ساختمان داده و **الگوریتم** های بهتر است نه نتیجه نوشتن برنامه به زبان اسمبلی .

همچنین اگر چه سیستم عاملها برنامه های بزرگی هستند ولی تنها بخش کوچکی از کد آنها، نسبت به کارایی ، بحرانی ( **Critical** ) می باشد مثل **مدیریت حافظه** و زمانبندی . CPU

لذا پس از آنکه سیستم عامل به زبان سطح بالا نوشته شد و به درستی عمل کرد می توان روتین های گلوگاه ( **bottleneck** ) و مهم را شناسایی کرد و سپس آنها را با روتین های معادل زبان اسمبلی جایگزین نمود.

## تاریخچه سیستم عامل

اولین کامپیوتر رقمی واقعی توسط یک ریاضیدان انگلیسی به نام چارلز بابیج ( **charles babbage** ) 1792-1871 طراحی شده است با بیج بیشتر عمر و ثروت خود را برای ساختن موتور تحلیلی صرف کرد ولی به علت اینکه فقط از مکانیک محض استفاده کرد به نتیجه دلخواه خود نرسید. موتور تحلیلی وی سیستم عامل نداشت بابیج فهمیده بود که موتور تحلیلی اش به نرم افزار نیاز دارد بنابراین این ار خانم جوانی را به نام ( ) استخدام نمود . او دختر شاعر مشهور انگلیسی لرد بایرون بود که به عنوان اولین برنامه نویس جهان شناخته شد. {تنباوم}

### 1- نسل اول (1945-1955): لامپ خلا و تخته های مدار های سوراخ دار

اندازه ماشین ها بسیار بزرگ بود و به همراه دهها هزار لامپ خلا داخل اتاق ها را پر می کرد کلیه برنامه به طور مطلق به زبان ماشین نوشته می شد. و اغلب به وسیله سیم بندی تخته مدار های سوراخ دار و به منظور کنترل عملیات باید ماشین پایه ای ماشین انجام می شد. زبانهای برنامه نویسی شناخته نشده بود ( حتی زبان اسمبلی ) و هیچ کس نامی از سیستم عامل نشنیده بود . اکثر برنامه ها مثل تهیه جدول سینوس و کسینوس بود. تنباوم} برنامه ساز مستقیما با سخت افزار در تراکنش بود ، سیستم عاملی در کار نبود.

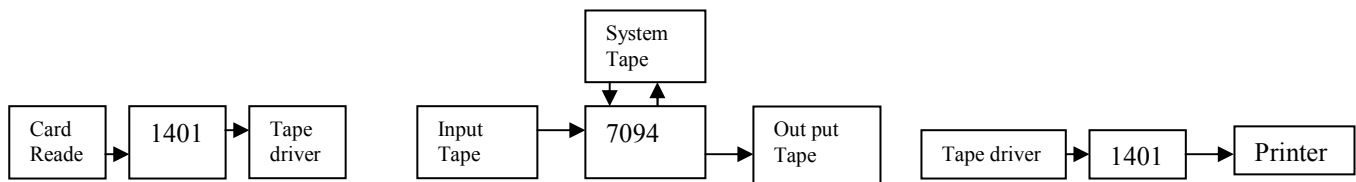
### 2- نسل دوم (1955-1965): ترانزیستور و سیستم های دسته ای

ظهور ترانزیستور در اواسط دهه 1950 یک انقلاب بنیادی بود . این ماشین ها در اتاق های مخصوص کامپیوتر با نهویه مطبوع مستقر می شود و گروهی از اپراتورها ی حرفهای راهبری آن را بر عهده می گرفتند. برای اجرای یک کار (job) ، یک برنامه نویسی برنامه (به زبان فرترن یا اسمبلی) خود را بر روی کاغذ می نوشت سپس آنر بر روی کارت ها منگنه می کرد و یک دسته کارت را به اتاق کامپیوتر می آورد و به یکی از اپراتورها تحویل میداد. در صورتیکه برنامه به مترجم فرترن نیاز داشت وی دسته کارت مترجم فرترن را نیز از داخل کمد فایل ها برداشته و در دستگاه کارتخوان قرار می داد . هنگامیکه اپراتورها برای انجام این کار ها در اتاق ماشین راه می رفتند ، مدت زمان زیادی از وقت کامپیوتر تلف می شد. راه حلی که به طور کلی پذیرفته شد ، سیستم های دسته ای ( **Batch system** ) بود.

ایده این روش این بود که یک سید پر از دسته کارتها در اتاق ورودی جمع آوری شود سپس کلیه آنها به وسیله دستگاه کارتخوان یک کامپیوتر کوچک و ارزان قیمت ها IBM 1401 خوانده شده و از طریق یک نوار گردان بر روی نوار مغناطیس ذخیره گردد . این IBM 1401 برای محاسبات مناسب نبودند و از یک ماشین گران قیمت IBM 7094 کامپیوتر ها برای پردازش و محاسبات واقعی استفاده می شد.

پس از حدود یک ساعت که برای جمع آوری یک دسته از کارها برای نوار صرف می شد ، یک اپراتور نوار را از اتاق کامپیوتر می آورد و یکی از اپراتور ها تحویل می داد.

در صورتیکه برنامه به مترجم فرترن نیاز داشت . وی دسته کارت مترجم فرترن را نیز از داخل کمد فایل ها برداشته و در دستگاه کارتخوان قرار می داد . هنگامیکه اپراتور ها برای انجام این کارها در اتاق ماشین راه می رفتند ، مدت زمان زیادی از وقت کامپیوتر تلف می شد. راه حلی که به طور کلی پذیرفته شد ، سیستم های دسته ای ( ) بود. ایده این روش این بود که یک سید پر از دسته کارتها در اتاق ورودی جمع آوری شود ، سپس کلیه آنها به وسیله دستگاه کارتخوان یک کامپیوتر کوچک و ارزان قیمت IBM 1401 خوانده شده و از طریق یک نوار گردان بر روی نوار مغناطیسی ذخیره گردد. این کامپیوتر ها ( ) رای محاسبات مناسب نبودند و از یک ماشین گران قیمت (7094) برای پردازش و محاسبات واقعی استفاده می شود.



حدود یک ساعت طول میکشد که یک دسته از کارها بر روی نوار ضبط شود پس از یک ساعت یک اپراتور نوار را از اتاق برداشته و به اتاق ماشین منتقل می کرد و در آنجا در یک نوار گردان قرار می داد. سپس یک برنامه مخصوص (سیستم عامل) را بار می کرد تا اولین کار از روی نوار بخواند و اجرا نماید. خروجی به جای چاپگر بر روی یک نوار دیگر نوشته می شد. پس از اتمام هر کار، سیستم عامل به صورت خودکار، کار بعدی را از نوار می خواند و شروع به اجرای آن می کرد. وقتی که تمامی دسته کارها اجرا می شد، اپراتور نوارهای ورودی و خروجی را بر می داشت، نوار ورودی را با دسته بعدی جایگزین کرده، نوار خروجی را به ماشین 1401 منتقل میکرد تا عملیات چاپ به صورت OFF Line (به کامپیوتر اصلی متصل نیست) انجام شود. {تنباوم} سیستم عامل در این گونه کامپیوترها بسیار ساده بود. کار اصلی آن، انتقال از یک کار به کار دیگر بود. سیستم عامل همیشه در حافظه قرار داشت. {سیلبر شاتس}

**ویژگی های سیستم عامل دسته ای: { سیلبر شاتس }**

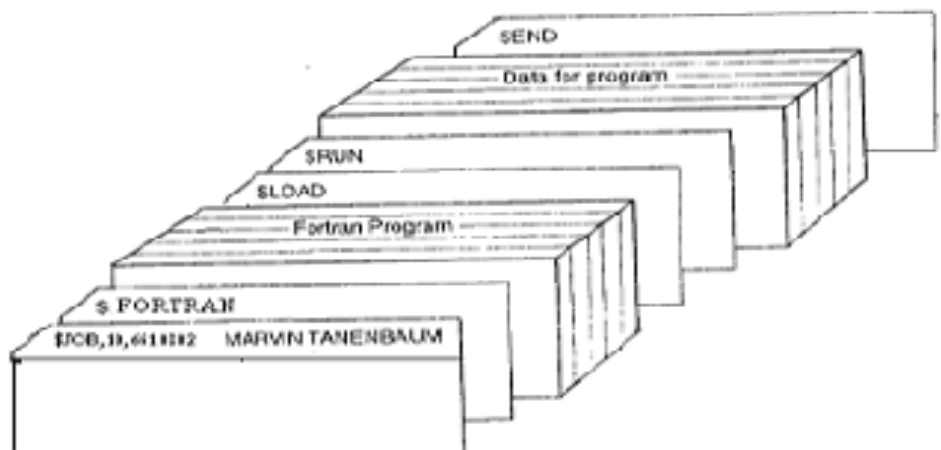
- 1- کاربر با کاری که در حال اجرا بود تعاملی نداشت.
- 2- درچنین محیطی CPU اغلب بیکار است چون سرعت دستگاه های I/O مکانیکی کمتر از سرعت دستگاه های الکترونیکی مثل CPU است. مثلا CPU در یک ثانیه هزاران دستورالعمل انجام می دهد. در حالکه دستگاه کارت خوان سریع در یک دقیقه 1200 کارت را می خواند. در نتیجه تفاوت سرعت بین CPU و دستگاه های I/O بسیار زیاد است.
- 3 - تخصیص CPU در چند برنامه ای انحصاری (Non Preemptive) است .

استفاده از فناوری دیسک، موجب افزایش سرعت دستگاه های I/O شده است. به جای اینکه کارت ها از طریق کارت خوان به حافظه خوانده شوند. سپس کاربر پردازش شود کارت ها مستقیماً به دیسک خوانده میشوند. {سیلبر شاتس}

**Job چیست؟**

مجموعه ای از یک 1- برنامه (program) 2- داده ورودی (input data) 3- دستورات سیستم عامل (O.S Command) (Job Control Language) یک زبان کنترل است.

مثال: ساختار یک کار ورودی در شکل 3 نشان داده شده است.



ساختار یک کار نوعی در FSM

1- این ساختار بایک کارت \$JOB شروع می شود. که مشخص کننده حداکثر زمان اجرا به دقیقه ، شماره حساب برای حسابداری هزینه ها و نام برنامه نویس می باشد.

\$JOB , 10, 6610802, MARVIN TANENBAUM

2- سپس یک کارت \$FORTRAN قرار دارد که به سیستم عامل اعلام می کند که کترجم فرترن را از روی نوار سیستم بار نماید.

3- سپس کارتهای برنامه ای که باید ترجمه شود قرار دارد و کارت \$LOAD

به دنبال آن قرار گرفته است. این کارت سیستم عامل را ودار میکند که برنامه OBJECT (مستقیماً ترجمه شده ) را بار نماید

4- سپس نوبت به کارت \$RUN می رسد که به سیستم عامل می گوید اجرای برنامه ها را به همراه داده هایش که در کارتهای بعدی قرار دارد، آغاز نماید.

5- در انتها کارت \$End قرار گرفته است که نشان دهنده پایان کار است.

کامپیوتر های بزرگ نسل دوم بیشتر برای محاسبت مهندسی و علمی بکار می رفت . برنامه آنها بیشتر به زبان فرترن و اسمبلی نوشته می شد. و معمولاً از سیستم عامل ( **Fortran Monitor System** ) FSM و IBSYS (سیستم عامل IBM برای 7094) استفاده می شد.

### 3-نسل سوم (1965-1980):مدارات مجتمع و چند برنامهگی

IBM سری 360 اولین خط تولید کامپیوتر های بزرگ بود که از مدارات مجتمع با مقیاس کوچک استفاده کرد بنابراین نسبت به کامپیوتر های نسل دوم که از ترانزیستور های جداگانه ساخته می شود نسبت به کارایی از هزینه بالاتری برخوردار بود.

IBM یک سیستم عامل عظیم الجثه و بسیار پیچیده بود که این سیستم عامل از میلیون ها خط اسمبلی که توسط هزاران برنامه نویس نوشته شده است . سیستم عامل OS/360 چندین تکنیک کلیدی جدید که در سیستم عامل های نسل دوم وجود نداشت

متداول نمود. شاید مهمترین این تکنیک ها ، چند برنامهگی ( Multi Programming ) بود.

در محاسبات علمی سنگین تنگنای محاسباتی CPU\_BOUND (Multi Limited) ،

I/O به ندرت به کار می رود.

اما در پردازش داده های تجارتي ، زمان انتظار I/O اغلب 80 تا 90 درصد کل زمان را به خود اختصاص می دهد. و CPU بیشتر بیکار است. برای حل مشکل ، حافظه را به چند تکه تقسیم بندی نماییم.

سیستم عامل
Job 1
Job 2
Job3

اگر چه سیستم های نسل سوم برای محاسبات علمی بزرگ و پردازش داده های تجاری مناسب بود ، ولی هنوز سیستم های دسته ای بودند.

### ویژگی سیستم عامل نسل سوم :

- 1- بلا فاصله پس از ورود کارها به اتاق کامپیوتر ، می توانند کارتها را خوانده و به دیسک منتقل نمایند.
- 2- بنابراین هرگاه یک کار ، در حال اجرا به پایان می رسد ، سیستم عامل می تواند یک کار جدید را از روی دیسک برداشته و در یک بخش خالی شده از حافظه بار نماید و سپس آنرا ، به اجرا در آورد . این تکنیک که Spooling نامیده می شود.
- 3- برای خروجی نیز ، به کار گرفته شده با SPOOLING دیگر نیازی به 1401 و نوار گردان های اضافی و حمل نوارها نبود. {تنباوم }

### سیستم های چند برنامه ای Multi programming

در نسل سوم کامپیوترها (80-1965) از **مدارات مجتمع (Integrated Circuit=IC)** برای ساخت کامپیوترها استفاده شد. به طور کلی برنامه ها را می توان به دو دسته تقسیم کرد : یکی برنامه ها با تنگنای محاسباتی

(CPU bound یا CPU Limiter) مانند محاسبات علمی سنگین که بیشتر زمان کامپیوتر صرف محاسبات CPU می‌شود و دیگری برنامه های تنگنای I/O (I/O Limited) مانند برنامه های تجاری که بیشتر زمان کامپیوتر صرف ورود داده ها و خروج اطلاعات می‌شود.

یک اشکال مهم سیستم های دسته ای این است که وقتی کار جاری برای تکمیل یک عملیات I/O مثلاً بر روی نوار گردان منتظر می‌شود. در این حال CPU بیکار می‌ماند و مجبور است صبر کند تا عملیات I/O به اتمام برسد. در برنامه های CPU Limited این اتلاف وقت اندک است ولی در برنامه های I/O Limited ممکن است حدود 80 تا 90 درصد وقت CPU به هدر برود.

برای رفع این مشکل از تکنیک **multiprogramming** استفاده می‌شود. بدنی ترتیب که حافظه به چند قسمت تقسیم شده و در هر قسمت یک برنامه مجزا قرار داده می‌شود. وقتی که یک کار برای تکمیل عملیات I/O منتظر می‌ماند، پردازنده به کار دیگری داده می‌شود. اگر تعداد کارهای موجود در حافظه کافی باشد می‌توان CPU را تقریباً صد در صد مشغول نگه داشت. البته نگهداری همزمان چند برنامه در حافظه نیاز به مدیریت خاص حافظه دارد تا برنامه ها بر همدیگر اثر سوء نداشته باشند. لذا مدیریت حافظه بحث مهمی در سیستم عامل می‌باشد.

کدامیک از موارد زیر برای ایجاد یک سیستم چند برنامه‌گی (Multi Programming) کاملاً ضروری است؟

- الف – وقفه
- ب- بافر
- ج- دیسک

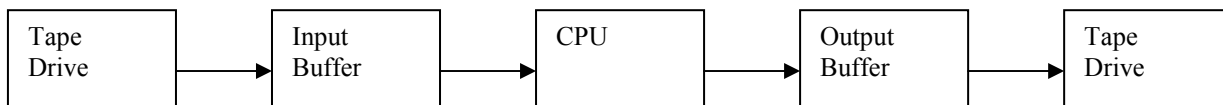
چرا حتماً باید دیسک باشد؟

چون باید Random Access باشد که Tape این توانایی را ندارد چون مراجعات مکرر و عقب و جلو کردن Tape کار بیهوده است.

### با فر کردن (Buffering)

علیرغم استفاده از نوار های مغناطیسی بازهم عملیات ورودی و خروجی کند بود ه، بهروری سیستم را کاهش می دهد. با استفاده از حافظه ای بافر ، عملیات ورودی و خروجی یک برنامه با اجرای آن همزمان می شود..

الف- کارهای I/O bound (I/O Limited): کارهایی که بخش زیادی از اجرای آنها در ارتباط با دستگاهای ورودی / خروجی بوده و محاسبات زیادی ندارند.  
ب- کارهای CPU bound (I/O Limited): کارهایی که حجم زیادی محاسبات داشته و بخش عمده نیاز آنها برای اجرا ، وقت پردازنده است.



### تکنیک SPOOLING

یکی دیگر از ویژگیهای سیستم عامل نسل سوم Spooling یا (On Line Spooling) است که معمولاً همراه چند برنامه گی استفاده می‌شود. این کلمه مخفف عبارت ( **Simultaneous Peripheral Operation onLine** ) می‌باشد. در این سیستم به جای آنکه کارتها از دستگاه کارت خوان مستقیماً وارد حافظه گردند و توسط CPU پردازش شوند ابتدا کاراکتر به کاراکتر در بافری در حافظه قرار گرفته و سپس به صورت بلوکی بر روی دیسک نوشته می‌شود. وقتیکه برنامه کاربر اجرا می‌شود و از سیستم عامل تقاضای ورودی می‌کند، اطلاعات ورودی به صورت بلوکی و با سرعت زیاد از دیسک خوانده می‌شوند. به طور مشابه هنگامی که برنامه برای خروجی چاپگر را احضار می‌کند، خط خروجی در یک بافر کپی شده و سپس در دیسک نوشته می‌شود. پس اطلاعات خروجی از دیسک بر اساس ترتیب و اولویت در چاپگر چاپ می‌شوند. در واقع اسپولینگ عمل I\O یک کار را با عمل محاسباتی کار دیگر روی هم می‌اندازد . (overlap) در سیستم اسپولینگ د رحالیکه ورودی یک کار از دستگاه ورودی خوانده می‌شود، کار دیگری در حال چاپ شدن است ، در همین بین حتی کار دیگری می‌تواند در حال پردازش و اجرا باشد . در اسپولینگ برنامه



عملیات ورودی و خروجی اش را متناسب با سرعت دیسک (که سریع است) انجام می‌دهد و نه متناسب با سرعت کارتخوان یا چاپگر (که خیلی کند هستند).

بنابراین سیستم مذکور باعث استفاده بهینه از CPU و سایل I/O می‌شود و سرعت عمل را بالا می‌برد. در این سیستم دیگر نیازی به کامپیوترهای 1401، نوار گردانهای اضافی و حمل نوارها (مانند سیستم‌های دسته‌ای) نداریم.

### سه مکانیزم برای پیاده سازی Spooling:

1- بافر

2- دیسک

3- وقفه (Interrupt)

بلوک دباگرام یک تکنیک اسپولینگ می‌تواند به صورت زیر باشد:

1- سیستم ورودی کاراکترهایی که توسط کارتخوان وارد می‌شود را در بلوکهایی جمع آوری کرده و به کمک مدیر دیسک این بلوکها را بر روی دیسک می‌نویسد . درانتهای هر مدرک ورودی اطلاعاتی راجع به آن فرآیند (مانند محل آن بر روی دیسک , اولویت ,اسم استفاده کننده ) به قسمت زمانبند کار فرستاده می‌شود.

2- زمانبند کار (Job scheduler) این زمانبند یک لیست از کارهای موجود در ماشین و اطلاعات لازم در مورد مدرک ورودی مورد نیاز هر یک را نگه می‌دارد. به این لیست انبار کار یا Jobpool یا Joblist نیز گفته می‌شود. زمانبند کار به پردازنده کار می‌گوید که کدام کار بعدی را اجرا کند. برای این منظور اطلاعاتی در مورد محل کار و مدرک ورودی آن بر روی دیسک را به پردازنده کار می‌دهد . همچنین اگر کارهای متعددی منتظر ورود به حافظه باشند و فضای کافی برای همگی در دسترس نباشد , زمانبند کار تعدادی از آنها را انتخاب کرده و به حافظه می‌آورد.

3- پردازنده کار (Job processor) کار داده شده را اجراء می‌کند. این پردازنده محل کامپایلرها و سایر نرم افزارهای سیستم را بر روی دیسک می‌داند. هنگام اجراء پردازنده کار خروجی های خود را به صورت بلوکی بر روی دیسک می‌نویسد و مدارک خروجی را تشکیل می‌دهد . پردازنده کار اطلاعاتی راجع به محل و اولویت مدارک خروجی به زمانبند خروجی می‌دهد .

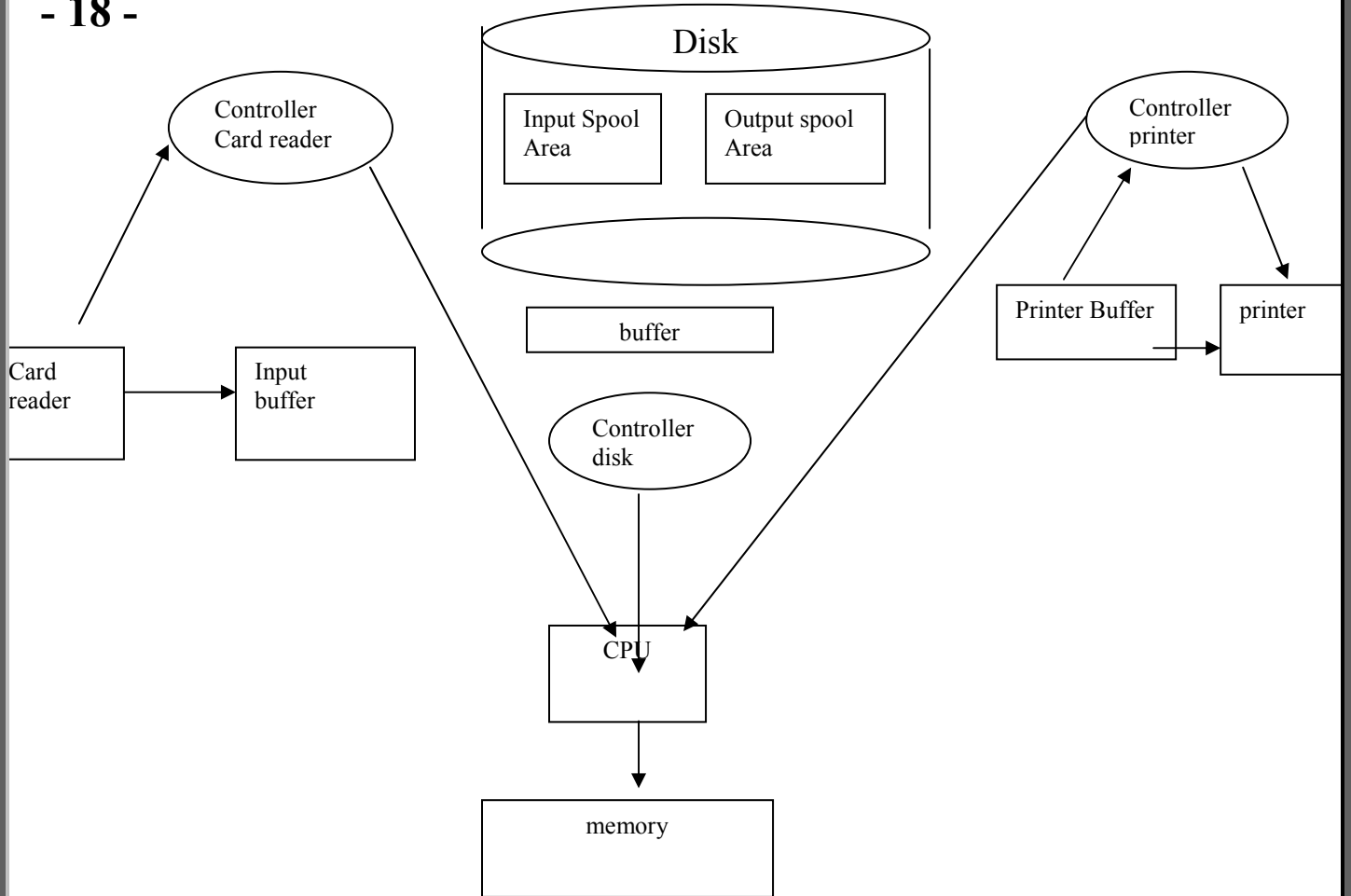
4- زمانبند خروجی (output scheduler) لیستی از مدارکی که باید چاپ شود را نگه می‌دارد. وقتی که چاپگر آزاد شد , این زمانبند مدرک بعدی را برای چاپ انتخاب کرده و محل مدارک بر روی دیسک را به سیستم خروجی می‌گوید.

5- سیستم خروجی بلاکهای خروجی را از روی دیسک خوانده و کاراکتر به کاراکتر (یا خط به خط) آنها را به چاپگر می‌فرستد.

6- مدیر دیسک (Disk Manager) که وظایف خواندن و نوشتن یک بلاک بر دیسک , تخصیص یک بلاک خالی روی دیسک و برگرداندن یک بلاک به مجموعه فضای آزاد دیسک را بر عهده دارد. در خواستههای مربوطه به دیسک در یک صف به نام

DTQ (Disk Transfer Queue) ذخیره می‌گردد.

البته هر سیستم اسپولینگ یک هماهنگ کننده (Coordinator) دارد که مسئول زمانبندی پردازش های سیستم و فراهم کردن عملیاتی که جهت همگام کردن بکار می‌آیند می‌باشد. این عملیات توسط دو روال انجام می‌پذیرد



## تفاوت Buffering و Spooling چیست؟

1- بافر کردن ( Buffering )، امکان همپوشانی ( همزمانی ) عمل I/O یک کار را با عملیات پردازش همان کار بوجود می آورد. در حالیکه spooling امکان همزمانی پردازش ورودی - خروجی چندین کار را با یکدیگر فراهم می کند.

2- هر Spooling الزاما بافر دارد ولی بافر Spooling ندارد .

سیستم Spooling راه حلی برای کار کردن با **دستگاه های I/O انحصاری** ، در یک سیستم چند برنامه ای است. یک دستگاه Spool شده نوعی را در نظر بگیرید: یک چاپگر هر چند از نظر تکنیکی آسان خواهد بود که به هر فرآیند کاربر اجازه دهیم که فایل مخصوص کاراکتری را برای چاپگر باز کند، ولی فرض کنید که فرآیند آن را باز کند و برای ساعت ها کار نکند، در این حالت هیچ یک از دیگر فرآیندها نیز قادر به چاپ کردن نخواهند بود. به جای این روش کاری که انجام گرفته است ، ایجاد یک فرآیند ویژه می باشد که شیخ (Deamon) نامیده می شود و یک فهرست مخصوص که فهرست Spooling نامیده می شود. برای چاپ کردن یک فایل ، ابتدا یک فرآیند تمامی فایل را برای چاپ شدن ایجاد می نماید و آن را در فهرست Spooling قرار می دهد. چاپ فایل های درون این فهرست بر عهده شیخ است که تنها فرآیندی است که اجازه استفاده از فایل مخصوص چاپگر را دارد . با حفاظت فایل مخصوص از دسترسی مستقیم کاربر، مسئله باز نگهداشتن طولانی و بیهوده آن نیز حل می شود.

Spooling: تنها در چاپگر به کار نمی رود ، بلکه در وضعیت های دیگری نیز استفاده می شود . برای مثال ، انتقال فایل از طریق یک شبکه معمولاً توسط یک شیخ شبکه ای انجام می شود. برای ارسال فایل به مقصد مشخص ، کاربر آن را در فهرست Spooling شبکه قرار می دهد. سپس شیخ شبکه آن را خارج ساخته و منتقل می کند.

### نکته:

یک کاربرد و استفاده ویژه از انتقال فایل Spool شده ، سیستم پست الکترونیکی (Email) اینترنت می باشد. این شبکه شامل میلیونها ماشین در سراسر جهان است که با استفاده از شبکه های کامیوتری با یکدیگر ارتباط برقرار می کنند. برای ارسال Mail به شخصی ، باید برنامه ای مانند را صدا بزیند که نامه را

جهت ارسال شدن می پذیرد و سپس آن را به صورت امانت در فهرست Spooling قرار می دهد تا بعداً از سال شود. تمام سیستم Mail در خارج از سیستم عامل اجرا می شود. {تنباوم 187}

Spooling در تمام سیستم عامل ها قابل استفاده است.

## سیستم های اشتراک زمانی (time sharing)

این سیستمها از اوایل سالهای 1970 در نسل سوم کامپیوترها معمول شدند. سیستم اشتراک زمانی در واقع تعمیم سیستم چند برنامه‌نگی است .

در سیستمهای چند برنامه‌نگی کاربر ارتباطی با کامپیوتر نداشت و خطایابی برنامه‌ها مشکل بود چرا که زمان برگشت نسبتاً طولانی اجازه آزمایش کردنهای متعدد را نمی‌داد. در سیستم اشتراک زمانی کاربر به کمک دو ترمینال (Terminal) که شامل کی بورد برای ورودی و مونیتر برای خروجی است با کامپیوتر به صورت محاوره‌ای (interactive) رابطه برقرار می‌سازد.

کاربر مستقیماً دستوراتی را وارد کرده و پاسخ سریع آن را روی مونیتر دریافت می‌کند. در این سیستمها چندین کاربر به کمک ترمینالهایی که به کامپیوتر وصل است همزمان می‌توانند از آن استفاده کنند .

در سیستم اشتراک زمانی فقط یک پردازنده وجود دارد که توسط مکانیزمهای زمانبندی بین برنامه‌های مختلف کاربرها با سرعت زیاد (مثلاً در حد میلی ثانیه) سوئیچ می‌شود و بنابراین هر کاربر تصور می‌کند کل کامپیوتر در اختیار اوست . در اینجا تأکید بر روی میزان عملکرد کاربر است یعنی هدف فراهم کردن وسایل مناسب برای تولید ساده نرم افزار و راحتی کاربرد می‌باشد و نه بالا بردن میزان کاربرد ماشین.

کاربر می‌تواند در هر زمان دلخواه برنامه خود را آغاز یا متوقف سازد و یا برنامه را به صورت قدم به قدم اجرا و اشکال زدایی (debug) کند . سیستمهای دسته‌ای برای اجرای برنامه‌های بزرگ که نیاز محاوره‌ای کمی دارند مناسب است ولی سیستمهای اشتراک زمانی برای مواردی که زمان پاسخ کوتاه لازم است ، استفاده می‌شوند .

در زمانی که کاربری در حال تایپ برنامه‌اش یا فکر کردن روی خطاهای برنامه اش می‌باشد CPU به برنامه کاربر دیگری اختصاص یافته تا آن را اجرا کند

**نکته 1:** سیستم های اشتراک زمانی عملاً فراگیر نشد تا اینکه استفاده از سخت افزار لازم برای حفاظت در کامپیوتر های نسل سوم رایج شد. {تنباوم}

**نکته 2:** در سیستم های اشتراک زمانی اندازه برهه زمانی ( Quantum ) را سیستم عامل معین می کند. همه جا مدیریت با سیستم عامل است . هیچ اتفاقی را سیستم عامل نمی فهمد بلکه فقط از طریق وقفه

(interrupt) می فهمد.

**نکته 3:** CPU، Timer ندارد بلکه Timer در روی مادر برد است یعنی Inrupt Timer به CPU می دهد و بعد برای اجرا سیستم عامل وارد عمل می شود.

نکته 4: در سیستم های محاوره‌ای (interactive) اولین کاری که انجام می دهیم عمل Login است و گرنه اجازه ورود به سیستم را نمی دهد. ابتدا UserID و Password را چک میکند. بعد به کاربر یک Token می دهد این Token

می گوید از چه ثبات هایی می توانی استفاده کنی و به کجا ها دسترسی داشته باشی مجوز صادر می کند.

زمانبند کار دائماً کنترل می کند که تعداد کاربران محاوره‌ای چند تاست ؟ اگر از تعداد مجاز بیشتر باشد جلوی ورود کاربران دیگر را می گیرد.

### نسل چهارم (1980 تا کنون) : کامپیوتر های شخصی

سال 1980 تاکنون که مدارات مجتمع با مقیاس بزرگ (Large Scale Integrated Circuit) ابداع شدند، به عنوان نسل چهارم کامپیوترها شناخته می‌شود. در این سالها کامپیوترهای شخصی با قیمتی ارزان و کارایی بالا و محیط گرافیکی و محاوره‌ای بسیار خوب به سرعت گسترش یافتند. سیستم عاملهای اولیه بر روی PCها مانند DOS فقط تک کاربره و تک برنامه‌ای بودند.

ولی سیستم عاملهای امروزی آن مانند Windows NT خاصیتهای چند برنامه‌نگی، چند کاربرته (multiuser) و شبکه‌ای را دارا هستند. با توجه به هزینه اندک سخت افزار اهداف سیستم عامل در طول زمان تغییر کرده است و برای PC ها به جای ماکزیمم کردن درصد استفاده CPU و وسایل جانبی ، سیستم به سمت راحتی کاربر پیش می‌رود.

به تدریج ویژگی‌های مهم سیستم عاملهای قدیمی در کامپیوترهای بزرگ (مانند محافظت حافظه ، حافظه مجازی، محافظت فایلها ، همزمانی پردازشها...) بر روی سیستم های PC نیز پیاده سازی شده است .

هنگامی که کامپیوترها از طریق شبکه به هم وصل شوند. به آنها ایستگاههای کاری (Work stations) می‌گویند . در یک سیستم عامل شبکه ، کاربران از

وجود ماشین های مختلف در شبکه با خیرند. آنها می توانند از دور وارد یک ماشین شوند و همچنین فایل های یک ماشین را روی ماشین دیگر کپی کنند. هر کامپیوتر سیستم عامل محلی خودش را اجراء می کند و کاربر یا کاربران محلی مخصوص به خود را دارد .

### سیستم های بی درنگ Real Time

یک سیستم بلا درنگ سیستمی است که در آن زمان نقش مهمی را ایفا می کند . نوعاً اگر یک یا چند دستگاه فیزیکی را از خارج کامپیوتر تحرکاتی ایجاد کنند کامپیوتر باید طی مدت زمانی معین در برابر آنها واکنش مناسب نشان دهد. مثل سیستم مانیتورینگ بخش آی سی یو بیمارستان ، سیستم خلبان اتوماتیک هواپیما و سیستم کنترل امن و مطمئن یک نیروگاه اتمی می باشد. {تنبیاهوم 106}

سیستم های بی درنگ معمولاً به عنوان یک کنترل کننده در یک کاربرد خاص استفاده می شوند. سیستم در این حالت می بایست در زمانی مشخص و معین حتماً جواب مورد نظر را بدهد. سیستم های کنترل صنعتی، پزشکی، کنترل موشک و غیره از این دسته اند .

نکته: در سیستم های بی درنگ زمان پاسخ باید سریع و تضمین شده باشد ولی در سیستم اشتراک زمانی مطلوبست که زمان پاسخ سریع باشند (ولی اجباری نیست). در سیستم دسته ای هیچ محدودیت زمانی در نظر گرفته نمی شود.

در سیستم های بی درنگ معمولاً وسایل ذخیره سازی ثانویه وجود ندارد و به جای آن از حافظه های ROM استفاده می شود. سیستم عامل های پیشرفته نیز در این سیستمها وجود ندارند چرا که سیستم عامل کاربر را از سخت افزار جدا می کند و این جدا سازی باعث عدم قطعیت در زمان پاسخگویی می شود. سیستم های بلا درنگ معمولاً به دو گروه تقسیم میشوند.

1- بلا درنگ سخت ( Hard Real Time )

2- بلا درنگ نرم ( Soft Real Time )

سیستم های بی درنگ با سیستم های اشتراک زمانی تناقض دارند لذا نمی توانند هر دو توأمأ وجود داشته باشند . به دلیل نیاز به پاسخ دهی سریع و تضمین شده سیستم های بلا درنگ از حافظه مجازی و اشتراک زمانی استفاده نمی کنند به این سیستمها «بی درنگ سخت» (Hard Real Time) نیز گفته می شود.

در سیستم بلا درنگ نرم بعضی مواقع آماده نشدن پاسخ در مهلت زمانی تعیین شده قابل تحمل است.

در سیستم های «بی درنگ نرم» یک وظیفه بی درنگ بحرانی، نسبت به سایر وظایف الویت دارد و تا پایان تکمیل شدنش این ارجحیت را دارا خواهد بود . از آنجا که این سیستمها مهلت زمانی (deadline) را پشتیبانی نمی کنند استفاده آنها در کنترل صنعتی ریسک آور است . هر چند که این سیستم های بی درنگ نرم می بایست پاسخی سریع داشته باشند ولی مساله پاسخ دهی به حادی سیستم های بی درنگ سخت نمی باشد.

از کاربردهای سیستم بی درنگ نرم می توان رزرواسیون شرکتهای هواپیمایی، چند رسانه ای (multimedia) واقعیت مجازی (Virtual reality) را نام برد . این سیستمها به ویژگی های سیستم عامل های پیشرفته (که توسط بیدرنگ سخت حمایت نمی شوند) نیازمندند . بعضی از نسخه های UNIX مانند solaris 2 خاصیت بیدرنگ نرم را دارا می باشند .

در برخی کاربردها (مثل کنترل صنعتی) در کامپیوترها از سیستم عامل استفاده نمی شود. از آنجا که در سیستم های کنترل صنعتی برنامه می بایست در اسرع وقت در مقابل یک اتفاق ، از خود عکس العمل نشان دهد ، وجود واسطه سیستم عامل باعث کند شدن مراحل می گردد .

### سیستم های چند پردازنده های Multi processing

کامپیوترها می توانند به جای یک CPU چندین CPU داشته باشند که در اینصورت به آنها سیستم multiprocessing میگویند. جهت استفاده از این سیستم های نیاز به یک سیستم عامل خاص می باشد که بتواند چندین برنامه یا بخشهای یک فرآیند ( را به صورت موازی واقعی روی آنها اجراء کند . )

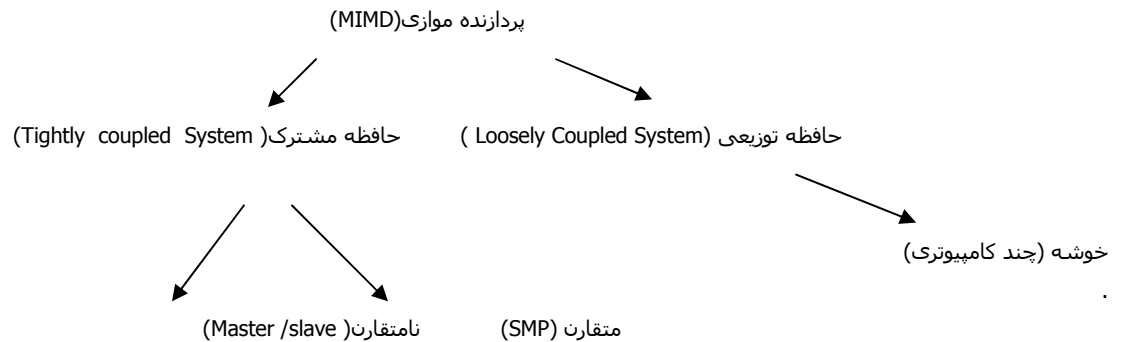
نکته : سیستم عامل multitasking برای اجراء چند نخ بر روی یک CPU و سیستم عامل multiprocessing برای اجرای چند نخ بر روی چند CPU به کار می‌روند .

برای جلوگیری از ایجاد مزاحمت ناشی از استفاده منبع مشترک به وسیله چندین پردازنده ، باید اولویتی در تخصیص منابع به پردازنده وجود داشته باشد . این وظیفه به سیستم عامل محول شده است. در طراحی سیستم عامل برای چندین پردازنده ها ، از سه سازمان استفاده شده است . {معمار-مویس مانو }

1- سیستم حاکم تابع (Master Slave Mode)

2-سیستم عامل توزیع شده با ارتباط Loosely Coupled System

3- سیستم عامل توزیع شده با ارتباط Tightly coupled System



1-در روش حاکم و تابع (Master Slave Mode) ، همیشه یک پردازنده ، که حاکم (Master) نامیده می شود، عملیات سیستم عامل را اجرا می کند . پردازنده های دیگر بمانند یک تابع (Slave) عمل کرده و عملیات سیستم عامل را اجرا نمی کنند. اگر پردازنده تابع (Slave) به سرویس سیستم عامل نیاز داشته باشد ، باید آن را با وقفه دادن به حاکم (Master) تقاضا کند و تا وقفه برنامه جاری منتظر بماند.

سیستم های توزیع شده در دو دسته قرار می گیرند: {کتاب مهندسی کامپیوتر- نگارش سمیع - کاجی}

الف- سیستم های با ارتباط محکم (Tightly coupled System): در این سیستم ها پردازنده ها دارای پالس ساعت یکسان و حافظه مشترک هستند . اجرای کار در آنها مشکل بوده ولی سرعت اجرا بالاست.

گاهی اوقات به سیستم های چند پردازنده ای ، سیستم های Tightly Coupled یا ارتباط محکم نیز گفته می شود. چرا که پردازنده ها کلاک (Clock) گذرگاه و همچنین حافظه مشترکی دارند روال های سیستم عامل بین پردازنده های موجود توزیع شده اند. با این وجود ، هر تابع خاص از سیستم عامل در هر زمان فقط به یک پردازنده اختصاص داده می شود . این نوع سازمان دهی ، سیستم عامل شناور ( Floating Operating System ) خوانده می شود زیرا روال ها از یک پردازنده به پردازنده دیگر شناورند و ممکن است اجرای روال ها در زمان های مختلف به عهده پردازنده های مختلف باشد. {معماری کامپیوتر مویس مانو }

سیستم عامل توزیع شده در یک محیط شبکه ای اجراء می شود. در این سیستم قسمتهای مختلف برنامه کاربر بدون آنکه خود او متوجه شود می توانند همزمان در چند کامپیوتر مجزا اجراء شده و سپس نتایج نهایی به کامپیوتر اصلی کاربر برگردند. کاربران نباید از این موضوع باخبر شوند که برنامه آنها در کجا به اجراء در می آید و یا فایل های آنها در کجای شبکه قرار دارد و همه این کارها باید توسط سیستم عامل به صورت خودکار انجام گیرد. به عبارتی دیگر سیستم باید از دید کاربر شفاف باشد و هرچیز را با نام آن فراخوانی کند و کاری به آدرس آن نداشته باشد.

ب-سیستم هایی با ارتباط سست (Loosely Coupled System): در این سیستم ها تعدادی پردازنده با خطوط ارتباطی مناسب وجود دارند و هر پردازنده دارای پالس ساعت و حافظه مستقل است و سرعت اجرا پائین می باشد . به سیستم های توزیع شده گاهی اوقات سیستم های Loosely

Coupled یا ارتباط ضعیف نیز می‌گویند، چرا که هر پردازنده کلاک و حافظه مستقلی دارد. پردازنده‌ها از طریق خطوط مخابراتی مختلفی مثل گذرگاه‌های سریع یا خطوط تلفن ارتباط دارند.

هر پردازنده می‌تواند روالهایی از سیستم عامل را که به آنها نیاز دارد اجرا کند. این سازمان برای سیستم‌های با کوپل سست (Lossely Coupled) که هر پردازنده ممکن است نسخه کامل از سیستم عامل خود را داشته باشد بیشتر مناسب است.

یکی از مزایای مهم سیستم‌های توزیع شده سرعت بالای اجرای برنامه‌هاست چرا که یک برنامه همزمان می‌تواند از چندین کامپیوتر برای اجراء شدنش استفاده کند.

همچنین به علت توزیع شدن اطلاعات، بانکهای اطلاعاتی حجیم می‌توانند روی یکسری کامپیوترهای شبکه شده قرار بگیرند. و لازم نیست که همه اطلاعات به یک کامپیوتر مرکزی فرستاده شود (که در نتیجه این نقل و انتقالات حجیم زمان زیادی به هدر می‌رود).

به علت تأخیرهای انتقال در شبکه و نوزهای احتمالی در خطوط انتقالی قابلیت اعتماد اجرای یک برنامه در یک سیستم تنها، بیشتر از قابلیت اجرای آن در یک سیستم توزیع شده است.

همچنین در سیستم توزیع شده اگر یکی از کامپیوترهایی که وظیفه اصلی برنامه جاری را برعهده دارد خراب شود کل عمل سیستم مختل خواهد شد. از طرف دیگر اگر اطلاعاتی همزمان در چند کامپیوتر به صورت یکسان ذخیره گردد و یکی از کامپیوترها خراب شود، داده‌ها را می‌توان از کامپیوترهای دیگر بازیابی کرد از این نظر امنیت افزایش می‌یابد.

سیستم‌های توزیع شده به دلیل افزایش سرعت اجرا، اشتراک منابع، افزایش قابلیت اطمینان و ایجاد ارتباط بین سیستم‌های مختلف، مورد استفاده قرار می‌گیرند.

در سیستم چند پردازنده‌ای، CPUها باید بتوانند از حافظه، امکانات ورودی و خروجی و گذرگاه Bus سیستم به صورت اشتراکی استفاده کنند. مزایای این سیستم‌های عبارتند از:

- زیاد شدن توان عملیاتی. (throughput) منظور از throughput تعداد کارهایی است که در یک واحد زمانی تمام می‌شوند. بدیهی است هر چقدر تعداد پردازنده‌ها بیشتر باشد تعداد کارهای تمام شده در یک پرود زمانی نیز بیشتر خواهد بود. البته این نسبت خطی نیست، مثلاً اگر تعداد پردازنده‌ها n باشد سرعت اجراء برنامه‌ها n برابر نمی‌شود چرا که بخشی از وقت پردازنده‌ها جهت مسائل کنترلی و امنیتی و سوییچ کردنها به هدر می‌رود.
- صرفه جویی در هزینه‌ها، از آنجا که پردازنده‌ها منابع تغذیه، دیسکها، حافظه‌ها و ادوات جانبی را به صورت مشترک استفاده می‌کنند در هزینه‌های سخت افزاری صرفه‌جویی می‌شود.
- تحمل پذیری در برابر خطا (fault-tolerant) سیستم‌های مالتی پروسسور قابلیت اعتماد را افزایش می‌دهند چرا که خرابی یک CPU سبب توقف سیستم نمی‌شود بلکه تنها سبب کند شدن آن خواهد شد. استمرار عمل با وجود خرابی نیازمند مکانیزمی است که اجازه دهد خرابی جستجو شده، تشخیص داده شده و در صورت امکان اصلاح شود (با کنار گذاشته شود). این توانایی به ادامه سرویس، متناسب با سطح بقای سخت افزار، تنزل مطبوع یا graceful degradation نامیده می‌شود.

سیستم‌های عاملهای چند پردازنده‌ای به دو دسته کلی متقارن و نامتقارن تقسیم می‌شوند.:

در سیستم چند پردازنده‌ای نامتقارن (Asymmetric Multi Processing = ASMP) یا Master/ Slave یک پردازنده جهت اجراء سیستم عامل و پردازنده‌های دیگر جهت اجراء برنامه‌های کاربران استفاده می‌شود. از آنجا که کد سیستم عامل تنها روی یک پروسسور اجراء می‌شود، ساخت این نوع سیستم عامل نسبتاً ساده است و از تعمیم سیستم عامل تک پردازنده‌ای به دست می‌آید.

این نوع سیستم عامل‌ها برای اجراء روی سخت افزارهای نامتقارن مناسب هستند، مانند کمک پردازنده و پردازنده‌ای که به هم متصل هستند یا دو پردازنده‌ای که از تمام حافظه موجود مشترکاً استفاده نمی‌کنند. یکی از معایب سیستم عامل نامتقارن غیر قابل حمل بودن (non-portable) آن است. یعنی برای سخت افزارهای مختلف باید سیستم عاملهای مختلفی نوشته شود چرا که نامتقارنی می‌تواند حالات مختلف داشته باشد.

در سیستم چند پردازنده‌ای متقارن (symmetric Multi Processing = ASMP) سیستم عامل می‌تواند روی هر یک از پروسسورهای آزاد یا روی تمام پردازنده‌ها همزمان اجراء شود. در این حالت حافظه بین تمام آنها مشترک می‌باشد. تمام پردازنده‌ها اعمال یکسانی را می‌توانند انجام دهند. سیستم متقارن از چند جنبه نسبت به نوع نامتقارن برتری دارد:

- از آنجا که سیستم عامل خود یک پردازش سنگین است اگر فقط روی یک CPU اجراء شود باعث می‌گردد که آن پردازنده همواره بار سنگینی داشته باشد، در حالیکه احتمالاً پردازنده‌های دیگر بی کار هستند لذا اجراء سیستم عامل روی چند پردازنده باعث متعادل شدن (balancing) بار سیستم می‌شود.

- در سیستم نامتقارن اگر پردازنده اجراء کننده سیستم عامل خراب شود کل سیستم خراب می‌شود ولی در سیستم متقارن از این نظر امنیت بیشتر است چرا که اگر یک پردازنده از کار بیفتد سیستم عامل می‌تواند روی پردازنده‌های دیگر اجراء شود .
- بر عکس سیستم عامل نامتقارن ، سیستم عامل قابل حمل (portable) بر روی سیستم های سخت افزاری مختلف است .

سیستم عامل SUNOS ورژن 4 از نوع نامتقارن و سیستم عامل Solaris2 ورژن و همچنین windows NT از نوع متقارن می‌باشند.

وجود پردازنده‌های متعدد از دید کاربر مخفی است و زمانبندی نخه ( Thread ) یا فرآیندها (process) روی هر یک از پردازنده‌ها به عهده سیستم عامل است .

گرچه multiprocessing و multithreading امکانات مستقلی هستند ولی معمولاً با هم پیاده سازی می‌شوند. حتی در یک ماشین تک پردازنده‌ای ، چند نخه کارایی را افزایش می‌دهد. همچنین ماشین چند پردازنده‌ای حتی برای فرآیندهای غیر نخه هم کارآمد است .

( /Slave )

### سئوال: {معماری کامپیوتر موريس مانو}

شباهت سیستم های چند پردازنده ای و چند کامپیوتری چیست؟

هر دو آنها عملیات همزمان را پشتیبانی می کنند.

تفاوت سیستم های چند پردازنده ای و چند کامپیوتری چیست؟

شبکه از چندین کامپیوتر مستقل تشکیل می شود که می توانند با هم ارتباط داشته یا نداشته باشند .

یک سیستم چند پردازنده توسط یک سیستم عامل کنترل می شود تا اتصالات بین پردازشگرها برقرار شده و در نتیجه تمام اجزاء سیستم در حل یک مسئله همکاری نمایند.

### فرق بین فرآیند و برنامه چیست؟ {60 تنبؤم }

مثال:

دانشمند کامپیوتری را مجسم کنید که می خواهد برای دخترش یک کیک تولد درست کند. او دستورالعمل پخت کیک و مواد لازم از قبیل آرد ، تخم مرغ ، شکر ، کمی وانیل ،... را در اختیار دارد. در اینجا دستورالعمل پخت کیک یک "برنامه "

(الگوریتمی که به زبان مناسب بیان شده است ) است. دانشمند هم در حکم CPU است و مواد لازم داده های ورودی هستند .

فرآیند فعالیتی است که شامل خواندن از روی دستورالعمل ، آوردن مواد لازم و پختن کیک می باشد.

حال تصور کنید که پسر این دانشمند باشتاب وارد می شود و می گوید که یک زنبور او گزیده است . دانشمند به خاطر می سپارد که تا کجای کار پیش رفته است . و از روی دستورالعمل چه کارهایی را انجام داده و الان بر سر کدام دستورالعمل (وضعیت فرآیند جاری ذخیره می شود) . سپس کتاب کمکهای اولیه را بر می دارد و بر اساس آن کارهای لازم را برای پسرش انجام می دهد. در اینجا مشاهده می شود CPU (دانشمند) از یک فرآیند (پختن کیک ) به فرآیند دیگر با اولویت بالاتر (رسیدگی به پسرش) سوئیچ میکند که البته هر کدام برنامه خاص خود را دارد (دستورالعمل پخت کیک و کتاب کمک های اولیه ). دانشمند وقتی که مداوای پسرش را به اتمام رساند، دوباره به سر کار قبلی خود بر می گردد. و از همان وضعیتی که کارها کرده بود ، دوباره شروع به کار می کند. در اینجا یکنکته کلیدی وجود دارد و آن اینست که یک فرآیند ، به نوعی فعالیت است . این فرآیند شامل یک برنامه ، ورودی ، خروجی ، و یک حالت (وضعیت) می باشد.

### فرق بین Multi Program و Multi Task چیست؟

سئوال 1: در multi program چه موقع عمل switching انجام می گیرد؟ وقت I/O

سئوال 2: در multi task چه موقع عمل switching انجام می گیرد؟ در یک برهه زمانی یعنی فقط منتظر عمل I/O

نمی ماند اگر به I/O بر خورد کند switch می کند. ولی اگر به I/O بر خورد نکند و آن برهه زمانی بگذرد بازهم switch می کند. یعنی پردازنده از آن گرفته می شود.

نکته 1: یک سیستم عامل میتواند چند برنامه (Multi program) باشد ولی چند وظیفه ای (Multi Task) نباشد. مثل ویندوز 1 و 2 و 3

چون فقط به I/O برخورد می کردند پردازنده Switch می کرد.

نکته 2: یک سیستم عامل که هم چند برنامه (Multi program) و هم چند وظیفه ای (Multi Task) باشد. مثل ویندوز 95 و 98

چون اگر چند پنجره را باز کنید می بینید همه پنجره ها در حال فعلیت هستند. یعنی پردازنده ما بین آنها Switch می کند .

نکته 3: اشتراک زمانی (Time Sharing) وقتی است که سیستم چند کاربره (Multi User) باشد.

مثال : وینوز 2000 و time sharing و multi task است. N تا کاربر بتوانند از طریق شبکه به یک سیستم وصل شوند و کاملاً منابع اش را به اشتراک بگذارند. در ویندوز NT - n تا کاربر به آن سیستم می یابند و از منابع سیستم استفاده می کنند.

نکته 4: در ویندوز 95 و 98 همه منابع غیر از حافظه و پردازنده را می توانیم به اشتراک بگذاریم. ولی در ویندوز NT و ویندوز 2000 و یونیکس همه منابع را می توانیم به اشتراک بگذاریم.

### دو نکته ضرورت چند وظیفه ای بودن را روی کامپیوترهای شخصی موجب شده اند. {استالینگ 87}

1- با افزایش سرعت و ظرفیت ریز پردازنده ها و با حمایت از حافظه مجازی ، کاربر ده پیچیده تر و مرتبتر شده اند .

برای مثال ممکن است کاربران بخواهند یک برنامه ، word ، یک برنامه Excel ، یک برنامه Autocad

را به طور همزمان برای ایجاد یک سند به کار برند.

1- باز کردن برنامه Autocad

2- رسم شکل و ذخیره آن

3- بستن برنامه Autocad

4- بازکردن Word

5- درج شکل در محل مناسب

در محیط چند وظیفه ای کاربر هر یک از کاربردهای مورد نیاز خود را باز کرده و به صورت بازها می کند . و

اطلاعات به سادگی می تواند بین چند کاربرد حرکت کند.

ولی در تک وظیفه ای اگر تغییری در شکل لازم باشد کاربر باید برنامه Word را ببندد، برنامه Autocad را باز کند ، شکل را ویرایش و ذخیره نماید ، برنامه Autocad را ببندد، برنامه Word را باز کند و در نهایت شکل ویرایش شده را درج کند . که خیلی ملال آور است.

2- برای عملکرد چند وظیفه ای ، رشد به کارگیری کامپیوتر به صورت مشتری / خدمتگزار (Client/Server) می باشد. در این دیدگاه یک کامپیوتر شخصی ( Client ) و یک سیستم میزبان (Server) مشتری کا، برای انجام یک کاربرد، خاص، بک کار گرفته شده اند. این ترتیب در یک کاربرد ممکن است یک یا چند کامپیوتر شخصی و یک یا چند دستگاه خدمتگزار ( Client ) درگیر باشند.

برای هر کدام از وقفه ها ما سرویس روتین مربوط به خودش را داریم .

**تعویض فرآیند:** زمانیکه فرآیند در حال اجرا با وقفه مواجه می شود و سیستم عامل فرآیند دیگری را برای اجرا در نظر می گیرد و کنترل را به آن منتقل می کند. ولی چند نکته طراحی مطرح می شود . اولاً ، چه حادثه های موجب شروع یک تعویض می گردد؟ دوماً تفاوت بین تعویض متن (Context Switching) و تعویض فرآیند (Context Process چیست؟ سوماً برای تعویض فرآیند سیستم عامل چه اعمالی را روی ساختمان داده ای که تحت کنترل دارد، باید انجام دهد؟

تفاوت بین تعویض متن (Context Switching) و تعویض فرآیند (Context Process چیست؟ {استالینگ 143}

فرآیندی که در حال اجراست ، به وسیله یک وقفه (حادثه ای که خارج از فرآیند بروز کرده و پردازنده آن را تشخیص می دهد) یا با اجرای یک System Call از سیستم عامل ، دچار وقفه می شود. در هر صورت ، پردازنده یک عمل تعویض حالت ( ) انجام می داده و کنترل را به



یک روال سیستم عامل منتقل می کند. بعد از انجام عمل لازم ، ممکن است سیستم عامل فرآیند ی که دچار وقفه شده بود را از سر گیرد و یا فرآیند دیگری را فعال نماید. {استالینگ 156}

چه چیزی متن (وضعیت) ذخیره شده را تشکیل می دهد؟  
هر اطلاعی که بتواند با اجرای سرویس روتین وقفه تغییر کند و برای از سرگیری برنامه وقفه داده شده لازم باشد باید ذخیره گردد. بنابراین شامل بلوک کنترل فرآیند (PCB) باید ذخیره گردد. {استالینگ 143}

### مفاهیم فرآیند (Process-فرآروند- پردازش- پروسه -پردازش)

فرآیند چیست؟ برنامه در حال اجرا را Process گویند.  
تعریف مفیدتر فرآیند: هر وقت یک Job را برای اجرا انتخاب می کند ، گوئیم فرآیند تا زمانی که Job خاتمه پیدا کند.  
فرآیند (process) مفهومی دینامیکی است منظور اینست Running عوض می شود یعنی دائماً در CPU ، Flagها و Register ها ، Stack ها و Data ها عوض می شوند.  
هر فرآیند دارای فضای آدرس (Address Space) مخصوص به خود است . فضای آدرس شامل برنامه اجرایی ، داده ای برنامه و پشته آن است. همچنین هر فرآیند برای خودش یک مجموعه از رجیسترها دارد، شامل شمارنده برنامه ، اشاره گر پشته (Stack Pointer) و دیگر رجیسترها ی سخت افزاری ، همینطور همه اطلاعات دیگری که برای اجرای برنامه مورد نیاز است.

عیناً یک کپی از فضای آدرس فرآیند بر روی فضای Swap دیسک وجود دارد که Process Core Image گویند .  
در واقع تصویر حافظه اصلی فرآیند بر روی حافظه جانبی است.

Core: حافظه اصلی (چمره ای) که در حال حاضر استفاده نمی شود..

1- هر وقت فرآیند بخواهد اجرا شود یک کپی را از دیسک می آورد در حافظه می گذارد.

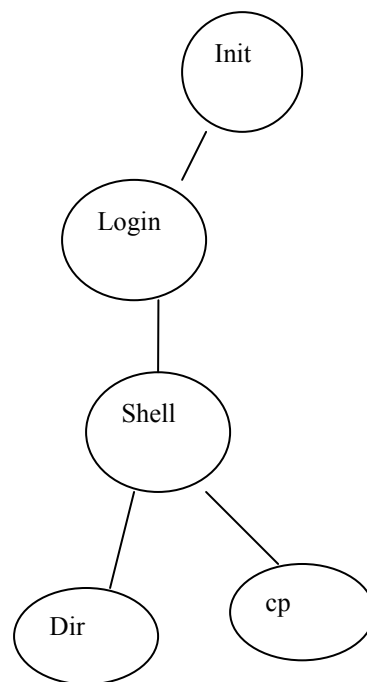
2- Core Image

بازگشت به دیسک Suspond خواهد بود . در سیستم های قبلی که سیستم Swap باشند.

نکته : اگر فرآیند ها نیاز به اجرای برنامه خارجی داشته باشند (جزو خودشان نیست) باید یک فرزند ایجاد کنند، فرزند یک فرآیند، خودش فرآیند است . و فرزندان ، فرزندان هم می توانند فرزند ایجاد کند یعنی ساختار درختی داریم . در Unix

این ساختار واحد است و حتی درخت واحد فایلها هم دارد.

مثال: در بعضی unix ها وقتی Start up می کند ، فرزند ی می سازد به نام Init بعد سیستم عامل می بیند کدام ترمینالها روشن هستند به تعداد آنها فرزند به نام Login می سازد.



نکته : در سیستم Unix : اگر فرآیند پدر از بین برود . فرآیند فرزند اطلاعات را به فرآیند Init داده می شود . هنگامیکه سیستم بوت می شود فرآیند Init که جد تمام فرآیند ها است ، به وجود می آید. فرآیند پدر همیشه منتظر فرآیند فرزند است . هر فرآیندی یک پدر دارد که اطلاعات به آن بر می گردد.

} بلوک کنترل فرآیند	شناسایی فرآیند
	اطلاعات وضعیت پردازنده
	اطلاعات کنترل فرآیند
	پشته کاربر
	فضای آدرس خصوص کاربر
	(برنامه ها، داده ها)
	فضای آدرس مشترک

**بلوک کنترلی فرآیند ( Process Control Block )**

هر فرآیند در سیستم عامل با یک بلوک کنترلی فرآیند خاص خود نشان داده می شود . یک PCB یک بلوک داده ای است که اطلاعاتی متناظر با یک فرآیند معین در بر می گیرد.  
از جمله این اطلاعات عبارتند از:

- 6- اطلاعات هویتی فرآیند: شامل نام ، شماره شناسایی فرآیند و فرآیند پدر
- 7- وضعیت فرآیند : شامل آماده ، اجرا ، منتظر
- 8- شماره برنامه : آدرس دستوالعمل بعدی
- 9- ثبات های پردازنده : شامل تمام اطلاعات وضعیتی برنامه که در هنگام رخ دادن یک وقفه حفظ میشود.
- 10- اطلاعات زمانبند پردازنده ها : شامل پارامتر های زمانبندی و الویت فرآیند
- 11- اطلاعات درباره مدیریت حافظه : در بر گیرنده تمام اطلاعاتی در رابطه با حافظه و فرآیند مربوطه

- 12- اطلاعات وضعیت ورودی - خروجی : شامل لیستی از فایلها ی باز ، دستگا های ورودی - خروجی اختصاص یافته به این فرآیند  
13- بطور کلی PCB مرکز اطلاعاتی است که سیستم عامل تمام اطلاعات کلیدی در باره فرآیند را در آن قرار میدهد.

## فرآیند ها سه وضعیت دارند؟

- 1- Ready : آماده اجرا
- 2- Running: در حال اجرا
- 3- Block : بلوکه شده (مسدود) یا Wait

Ready : فرآیندی هایی که منتظر هیچ منبعی بجز CPU نیستند ، CPU به آن بدهید اجرا می شود. تمام منابع را دارد (فعالاً) ممکن است CPU داده شود ولی بعد درخواست I/O کند و متوقف شود.

Running: فرآیندی که در حال حاضر CPU و سایر منابع مورد نیازش را در اختیار دارد و در حال اجرا است.  
Blocked: فرآیند هایی هستند که منتظر منبعی بجز CPU (مثلاً I/O) و یا یک وقفه خارجی هستند.

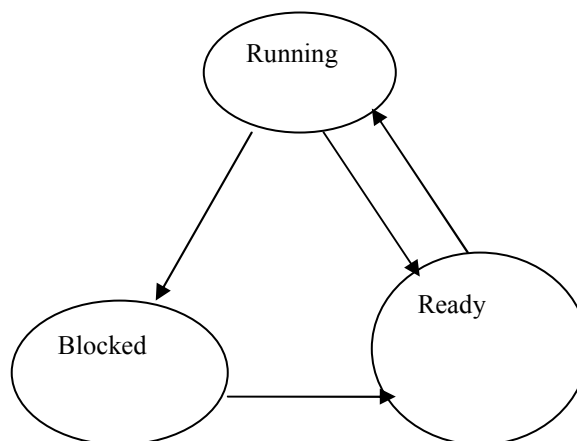
فرآیند منتظر دو راه دارد:

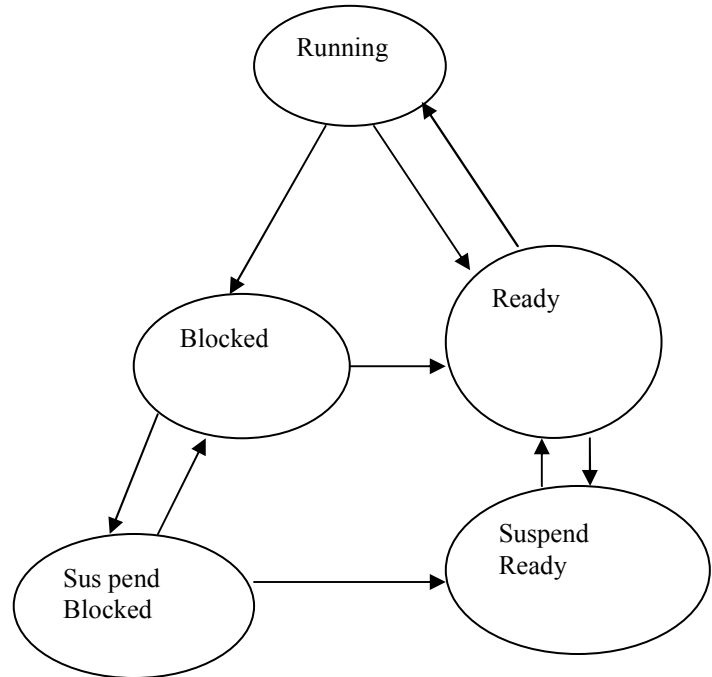
- 1- Busy Waiting : Cpu را Loop می زند. وقت Cpu را تلف می کند.
- 2- Blocked : به فرآیند هایی که منتظرند CPU را داوطلبانه پس می دهد.

نکته مهم 1: رفتن فرآیند از Blocked به Running غلط است چون اگر منابع را دریافت نمود Ready می شود.  
نکته مهم 2: رفتن از حالت Ready به حالت Block غلط است . فرآیند Ready در صورتی Block می شود که اول اجرا شود (Running) ، تا برسد به جایی که نیاز به منبع دارد.

نکته 3: از حالت Ready به حالت Running امکان پذیر است یعنی زمانبند ( Scheduler ) تصمیم می گیرد که Ready به Running وارد شود یعنی CPU می دهد که اصطلاحاً Dispatch می گویند.

از نظر تنبایوم





نظر ویلیام استالینگ

3- فرآیند در وضعیت آماده و معلق ( **Suspend Ready** ) فقرار می گیرد اگر حافظه اصلی از آن گرفته شود و بر روی حافظه ثانوی قرار گیرد و بهمحض با ر شدن در حافظه اصلی در وضعیت آماده قرار گیرد.

4- فرآیند در وضعیت مسدود و معلق (Suspend Wait): قرار می گیرد اگر در حافظه ثانوی قرار گیرد و منتظر حادثه ایی باشد. نحوه تبدیل وضعیت فرآیند

اختصاص یافتن پردازنده به اولین فرآیند در لیست فرآیند های آماده را توزیع ( **Dispatch** ) می نامند. این عمل توسط بخشی از سیستم به نام توزیع کننده ( **Dispatcher** ) انجام میگردد.

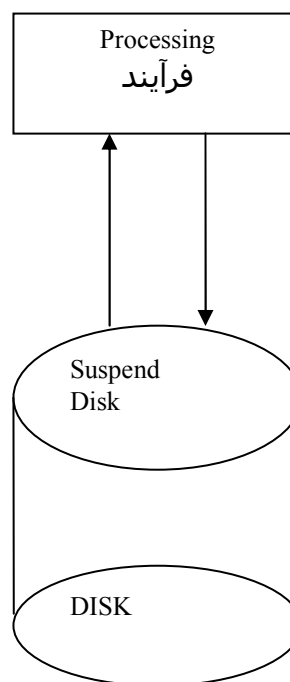
**Time Run Out**: برای جلوگیری از اینکه فرآیندی سیستم را منحصراً به خود اختصاص می دهد، سیستم عامل از یک تایمر استفاده می نماید. این تایمر به فاصله زمانی معین (برش زمانی ) تنظیم می گردد. چنانچه فرآیند به طور داو طلبانه قبل از سپری شدن فاصله زمانی ، پردازش را آزاد نکند، این تایمر ، وقفه خارجی را تولید می کند که سبب می شود تا سیستم عامل کنترل را بدست گیرد.سیستم در اینصورت فرآیند در حال اجرا را به حالت آماده تبدیل می نماید.

**DISPATCH :** Ready--> Running  
**Time Run Out :** Running --> Ready  
**Wait :** Running ---> Block  
**Wake up :** Block --->Ready  
**Suspend :** Block ---->Suspend Wait  
**Suspend :** Ready---->suspend Ready

سئوال : اولویت با Ready یا Blocked است؟

به دلیل کمبود حافظه فرآیندی که Blocked است به دیسک منتقل می کنیم و اولویت به برنامه Ready داده می شود. نکته: در وضعیت Ready و Blocked صف وجود دارد.. Suspend

: وقتی فضای حافظه اصلی کم می شود. تمام تصویر فرآیند روی دیسک منتقل می شود یعنی عمل را Swap out انجام می شود. چون حافظه اصلی محدود است از Suspend استفاده می شود. که زمانبند میان مدت این کار را انجام می دهد.



نکته 1: فرآیندی که Block است اگر Swap out کنیم Suspend می شود.. (CPU که ندارد روی دیسک هم منتقل می شود)  
نکته 2: اگر در حافظه اصلی جا کم داشته باشیم و همه فرآیند ها هم Ready هستند باید بعضی از Ready ها را روی دیسک ببریم. که Suspend Ready گفته می شود.

نکته 3: فرآیندی که Suspend Ready است هرگز Ready نمی شود. بلکه باید RUN شود. بعد به حالت Ready برود.

نکته 4: از حالت Suspend Blocked به حالت blocked بسیار نادر است اگر فرآیندی آماده اجرا نیست و در حافظه اصلی هم قرار ندارد، آوردن آن به حافظه چه امتیازی دارد؟ ممکن است فرآیندی پایان یابد و مقداری از حافظه اصلی آزاد شود. فرآیندی در صف Blocked Wait وجود دارد که اولویتش از تمام فرآیند های صف Suspend Ready بیشتر است. آوردن یک فرآیند مسدود به حافظه نسبت به فرآیند آماده، معقول به نظر می رسد.  
نکته 5: از حالت blocked wait به حالت Suspend Ready: موقعی یک فرآیند مسدود و معلق به حالت آماده و معلق تغییر حالت می دهد که حادثه ای که منتظرش بوده است، اتفاق افتاده است.

## مسائل

مسائل {تنتیابوم}

- 1- دو وظیفه اصلی یک سیستم عامل چیست؟
- 2- چند برنامه‌گی (Multi Program) چیست؟

3- **SPOOLING** چیست؟

4- چرا اشتراک زمانی (Time Sharing) در کامپیوتر های نسل دوم خیلی رایج نشد؟

الف- غیر فعال کردن کلیه وقفه ها

ب- خواندن ساعت (زمان -از- روز)

ج- تنظیم ساعت (زمان -از- روز)

د- تغییر نگاشت حافظه

5- مدل مشتری -خدمت گذار در سیستم های توزیع شده رایج است. آیا می توان از آن در سیستم های تک کامپیوتری هم استفاده کرد؟

6- چرا در سیستم اشتراک زمانی به جدول فرآیند نیاز داریم؟ آیا در سیستم های کامپیوتر شخصی که در آن فقط یک فرآیند وجود دارد ، و آن فرآیند کل ماشین را در اختیار می گیرد تا به اتمام برسد، نیز به آن جدول نیاز داریم ؟

\*\*\*\*\* فصل دوم \*\*\*\*\*

**سیستم های چند نخی multithreading**

در تکنیک چندنخی (multithreading) یک فرآیند (process) که برنامه ای در حال اجراست ، می تواند به بخشها یا نخهایی (بندهایی) تقسیم شود که می توانند به صورت همزمان اجراء شوند .

برنامه هایی که چند وظیفه مستقل از هم را انجام می دهند می توانند به صورت چند نخی نوشته شوند.  
گ

فرآیند (process) یا پردازش اساس یک برنامه در حال اجراست که منابعی از سیستم به آن تخصیص داده شده است (شامل رجیسترها، حافظه، فایلها و دستگاهها). فرآیند می تواند مجموعه ای از یک یا چند نخ باشد.

به نخ، رشته یا بند هم گفته می شود . کلیه اطلاعات مربوط به هر پروسس ، در یکی از جداول سیستم عامل به نام **process Control Block=PCB** ذخیره می شود. این جدول یک آرایه یا لیست پیوندی از ساختارهاست که هر عضو آن مربوط به یکی از پروسس هاست که در حال حاضر موجودیت دارد.

اطلاعات موجود در PCB عبارتند از :

- حالت جاری پردازش
- شماره شناسایی پردازش
- اولیت پردازش
- نشانی حافظه پردازش
- نشانی محل برنامه پردازش بر روی دیسک
- نشانی سایر منابع پردازش
- محلی برای حفظ ثباتها .

\*\*\*\*\* **کنکور سال 74** \*\*\*\*\*

**سال 74**

1- در یک سیستم مبادله ساده که برش زمانی فقط یک پروسس کاربر به همراه سیستم عامل در حافظه اصلی قرار دارد از یک دیسک با مشخصات زیر استفاده شده است متوسط زمان حرکت هد 30 میلی ثانیه ، مینیم زمان حرکت 10 میلی ثانیه ، سرعت چرخش دیسک 15 میلی ثانیه در هر دور و گنجایش هر شیار 10 کیلو بایت . در صورتی که اندازه همه پروسس ها کاربران ثابت و طول هر برش زمانی یک دهم ثانیه باشد . میزان به کارگیری مفید CPU چقدر است؟

(ا) بیش از 30 درصد و کمتر از 40 درصد

2- آیا کد صوری زیر برای مسئله Critical Section بین دو فرآیند همروند قابل قبول است؟ چرا؟

```
{
Int turn;
Boolean flag [2];
Proc(i);
Int (i);
{
While (TRUE)
{
Compute;
Flag[i]=TRUE;
Turn:=(i+1) mod 2;
While (flag [(i+1) mod 2] & turn =i);
Critical -section;
Flag [i]=FALSE;
}
}
Turn:=0;
Flag[0]=FALSE;
Flag[1]=FALSE;
Proc(0) AND Proc (1);
}
```

(توضیح اینکه اپراتور AND به معنای اجرای همروند می باشد).

1)خیر- زیرا Dead Lock وجود دارد.

2)بلی- زیرا شرط Mutual Exclusion برقرار است.

3)خیر زیرا شرط Mutual Exclusion برقرار است.

4) بلی زیرا Deadlock وجود ندارد.

3- پنج کار در وضعیت آماده ، در انتظار اجرا شدن روی یک کامپیوتر هستند. زمان تخمین زده شده برای اجرای این کارها برابر 10 ، 6، 5، 8، X میکرو ثانیه ( X مجهول است) بنظر شما استفاده از کدام روش زمان بندی ، متوسط زمان پاسخگویی ( Response\_Time ) این کارها را حداقل می کند؟

1-First Come First Served(FCFS)

2-Shortest Job First(SJF)

3-Shotest Remaining time (SRT)

4-Round Robin (RR)

4- به دو برنامه زیر دقت کنید:

یک پراسس

IF jobcount = 1 then

روتین اینتراپت

Jobcount:=1;

WWW.SOFTECH.IR  
Return from interrupt

wake up (jobscheduler);  
Jobcount :=0

ELSE

Wait for interrupt;

Jobscheduler (1) همیشه بیدار است

Jobscheduler (2) هیچ وقت بیدار نمی شود.

Jobscheduler(3) گاهی اوقات اشتباهاً بیدار می شود.

Jobscheduler (4) گاهی اوقات اشتباهاً بیدار نمی شود.

5- به فرض آن که اندازه متوسط هر فرآروند برابر  $p$  بایت ، اندازه هر صفحه برابر  $q$  بایت و اندازه هر مدخل از جدول صفحه برابر  $e$  بایت باشد ، اندازه بهینه صفحه ( ) چند بایت خواهد بود ؟ (متوسط فضای هدر رفته در آخرین صفحه ، فرآروند را برابر نصف اندازه صفحه در نظر بگیرید).

$P^2/2e$  (4)

$2pe$  (3)

$2ep/q$  (2)

$p/2e$  (1)