فصل ۱۱
تشکیلات ورودی خروجی

وهاب صمدی
دانشگاه صنعتی قم
پاییز ۹۱

- دستگاه های ورودی/خروجی مکانیزمی برای ارتباط کامپیوتر با دنیای بیرون هستند.
- اطلاعات از طریق این ادوات به کامپیوتر وارد و پس از پردازش به کاربر بازگردانده می شوند.
- صفحه کلید یک نمونه از مهمترین دستگاه های ورودی و صفحه نمایش و چاپگر نمونه های از دستگاه های خروجی هستند.

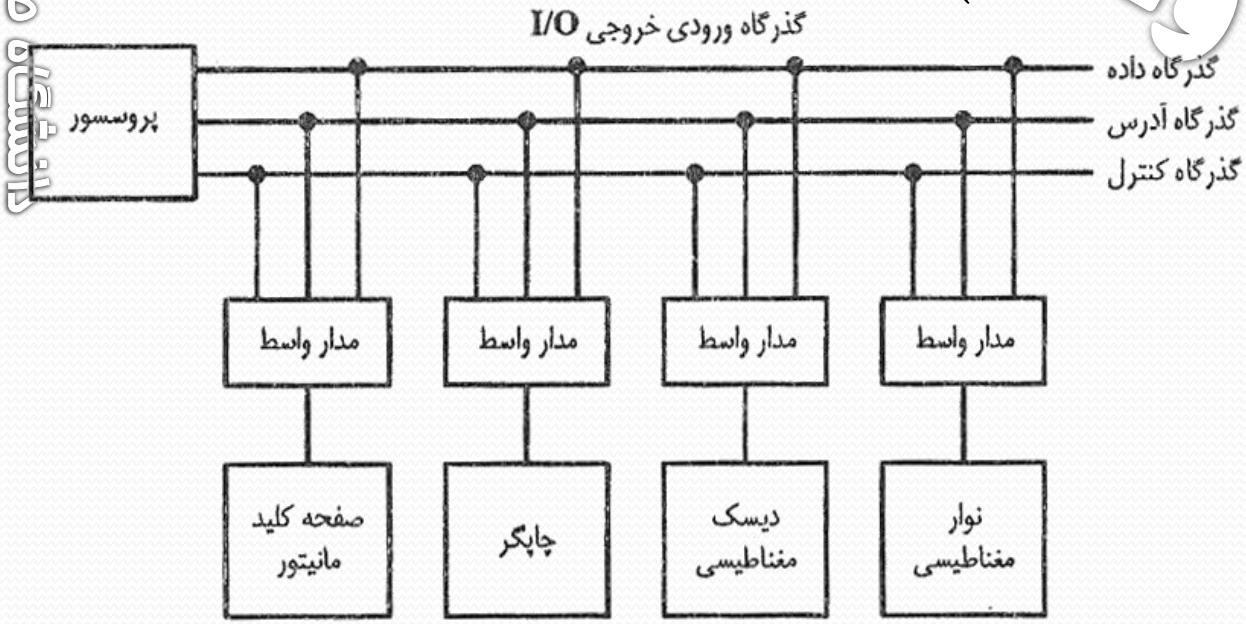
مدارهای واسط

- **دستگاه های جانبی نمی توانند به صورت مستقیم به CPU متصل شود چون:**

1. مکانیزمهای دستگاه های ورودی/خروجی (مکانیکی، الکترومکانیکی)، ولتاژ کاری و مشخصات دیگر آنها با پردازنده و مدار داخلی کامپیوتر متفاوت است.
2. سرعت انتقال اطلاعات متفاوت است (دستگاههای I/O کندتر هستند و نیاز به سنکرون کردن CPU و ادوات I/O وجود دارد).
3. فرمت نمایش اطلاعات در دستگاه های ورودی و خروجی با فرمت نمایش اطلاعات در حافظه/پردازنده متفاوت است.
4. هر دستگاه ورودی/خروجی سیستم عملکرد مخصوص به خود را داراست و احتیاج به واسطی هست که بدون اختلال در عملکرد سایر دستگاه های جانبی بتوانند به کار خود ادامه دهند (به عنوان مثال حتی دو پرینتر با این که هر دو وظیفه چاپ را بر عهده دارند، اما سیستم عملکرد و دستورات خاص خود را دارند)

مدارهای واسط (Interfaces)

- ارتباط بین پردازنده و دستگاههای ورودی/خروجی به مدارهای واسط یا Interfaceها نیازمند است.
- BUS ورودی خروجی متشکل است از:
  1. گذرگاه داده (Data Bus)
  2. گذرگاه آدرس (Address Bus)
  3. گذرگاه کنترل (Control Bus)



- **به عنوان مثال وقتی قرار است اطلاعاتی در پرینتر چاپ شود:**
  - ابتدا آدرس پرینتر بر روی گذرگاه آدرس قرار می گیرد و پرینتر تشخیص می دهد که باید نسبت به اطلاعات درگاه داده و کنترل حساس باشد.
  - سایر دستگاه ها تشخیص می دهند آدرس متعلق به آنها نیست و بنابراین تبعیت از دستورات و داده هایی که روی گذرگاه داده/کنترل قرار می گیرند استفاده نمی کنند.
  - دستور مرتبط به پرینتر ارسال می شود.

انواع دستورات ورودی/خروجی (I/O Commands)

- **دستورات ورودی/خروجی بر چهار نوع هستند:**

1. دستورات کنترل (Control Command)
   - دستوری که به دستگاه جانبی اطلاع میدهد باید کار خاصی را انجام دهد. مثلاً به هارددیسک می گوید که Head خود را به مکانی خاص منتقل کند تا آماده خواندن/نوشتن شود
2. استعلام وضعیت (Status)
   - دستوری که وضعیت دستگاه جانبی را تشخیص می دهد. به عنوان مثال کنترل می کند آیا پرینتر کاغذ دارد تا بتواند پرینت کند یا نه؟
3. خواندن داده (Input Data)
   - با ارسال این دستورات از سمت CPU، اطلاعات از دستگاه جانبی خوانده می شود و به مدار واسط انتقال می یابد. CPU با کنترل کردن بیت های پرچم از رسیدن اطلاعات آگاه می شود و آنها را می خواند. مثلاً دستور خواند اطلاعات از هارددیسک
4. خارج کردن داده (Output Data)
   - اطلاعات بر روی ثباتهای مدار واسط قرار می گیرند و سپس به دستگاه خروجی منتقل می شوند. به عنوان مثال، زمانی که باید کاراکترهایی بر روی پرینتر باید چاپ شوند، دستور خروج داده ها باعث انتقال اطلاعات بر روی بافر پرینتر می شود که نهایتا منجر به چاپ کاراکترها می گردند.

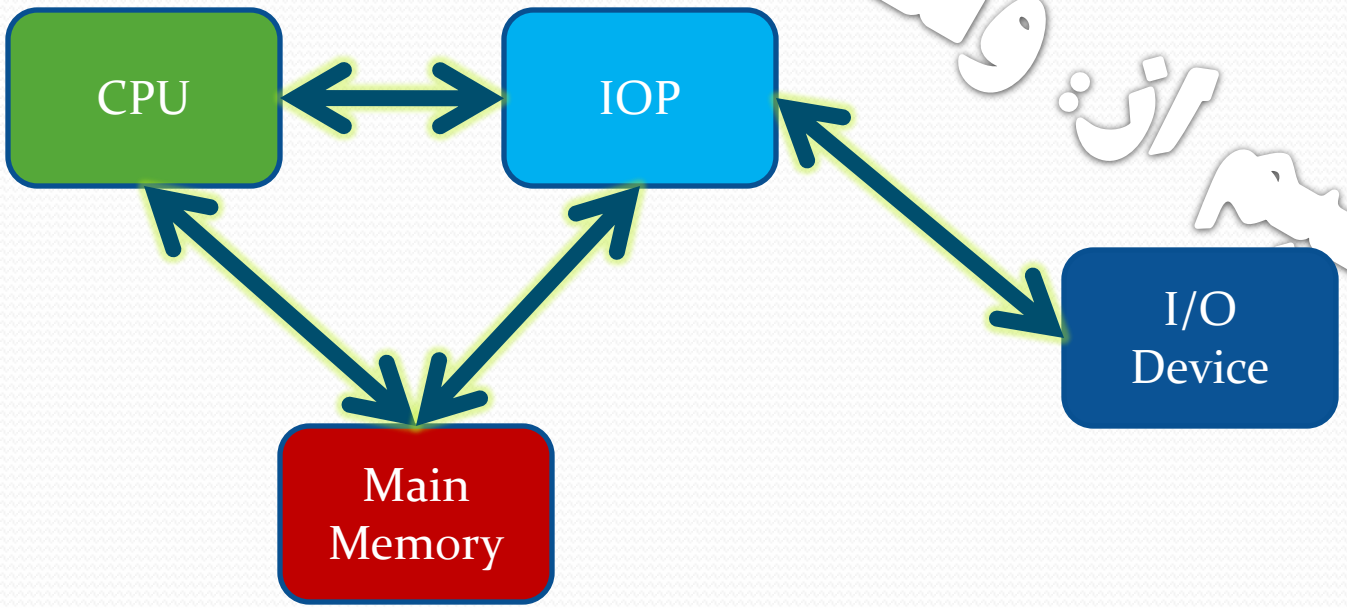
گذرگاه حافظه و گذرگاه ورودی/خروجی

- پردازنده علاوه بر ارتباط با دستگاههای جانبی باید با حافظه اصلی نیز در ارتباط باشد.
- گذرگاه حافظه نیز (مشابه گذرگاه خارجی/ورودی) دارای گذرگاههای داده و آدرس و همچنین خطوط خواندن/نوشتن است.
- سه رویکرد در مورد نحوه پیاده سازی این دو نوع گذرگاه وجود دارد:

1. دو گذرگاه مستقل داشته باشیم، یکی برای ادوات جانبی یکی برای حافظه
2. یک گذرگاه مشترک برای هر دو داشته باشیم اما خطوط کنترل هر کدام مستقل باشند
3. یک گذرگاه مشترک برای هر دو داشته باشیم و خطوط کنترل هم مشترک باشند

I/O Processor

- روش اول (گذرگاه های مستقل) برای سیستمهایی استفاده می شود که پردازنده ورودی خروجی (IOP) داشته باشند.
- دلیل وجود این پردازنده، ایجاد امکان ارتباط مستقیم بین دستگاههای جانبی و حافظه است.



ورودی/خروجی: نگاشت حافظه یا ایزوله

الف: ورودی/خروجی ایزوله

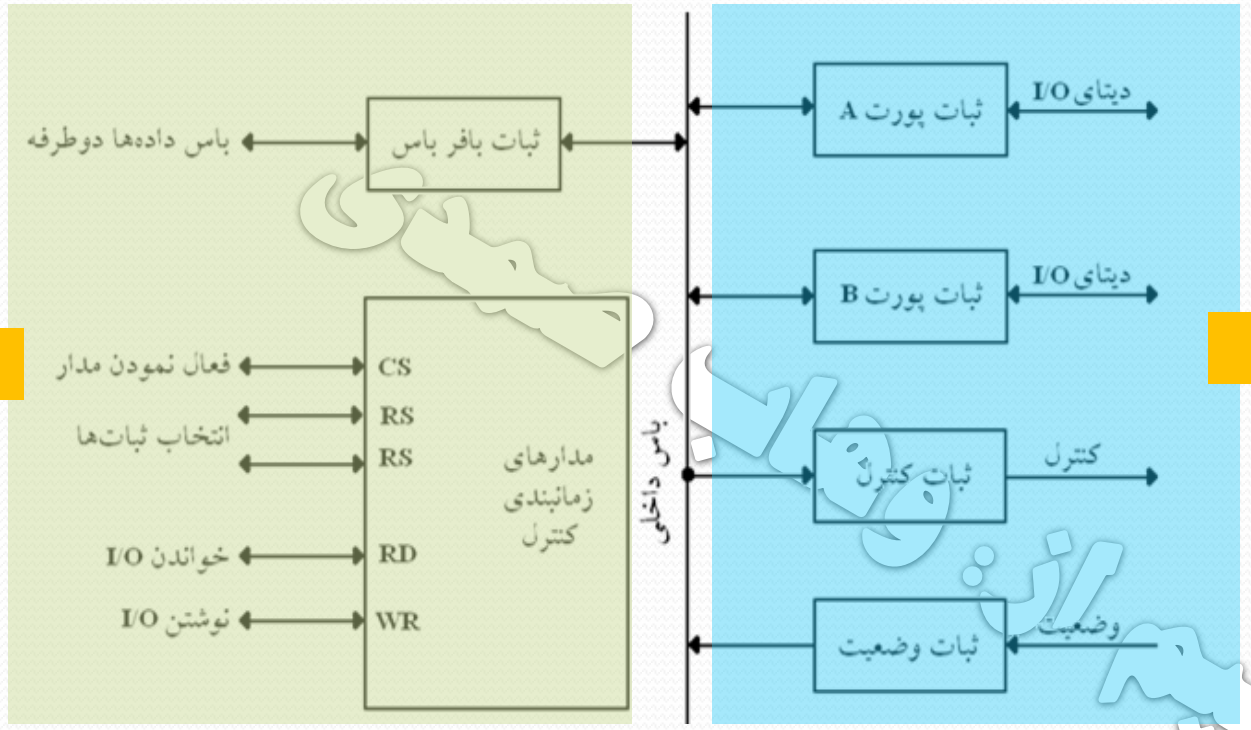
- در این روش علیرغم مشترک بودن گذرگاه آدرس، فضای آدرسهای حافظه و I/O از هم جدا هستند و خطوط نوشتن و خواندن حافظه و I/O جداست (یعنی آدرس ۱۰۰ حافظه با آدرس ۱۰۰ در دستگاه I/O متفاوت است).
- گذرگاه آدرس بین دستگاه های I/O و حافظه یکسان است اما چیزی که مشخص می کند، آدرس مربوط به کدام دسته است، فعال بودن خطوط خواندن/نوشتن آنها است.
- اگر آدرس ۱۰۰ بر روی گذرگاه قرار بگیرد و خط نوشتن حافظه فعال شود، یعنی اطلاعاتی باید بر روی آدرس ۱۰۰ حافظه نوشته شود
- اگر آدرس ۱۰۰ بر روی گذرگاه قرار بگیرد و خط نوشتن I/O فعال شود، یعنی باید بر روی بافر ورودی دستگاهی که آدرس آن ۱۰۰ است، اطلاعات نوشته شود.

ورودی/خروجی: نگاشت حافظه یا ایزوله

ب: نگاشت حافظه (Memory Mapped)

- در این روش فضای آدرس دستگاه های I/O و حافظه با هم آمیخته است و خطوط خواندن و نوشتن حافظه و I/O مشترک است (یعنی آدرس ۱۰۰ تنها می تواند برای حافظه و یا دستگاه I/O باشد).
- در این حالت CPU در هنگام خواندن و نوشتن تمایزی بین حافظه و I/O قائل نمی شود و وجه تمایز آنها صرفا آدرسها متفاوت آنهاست.
- معمولاً بخشی از آدرس حافظه برای ثباتهای واسطها کنار گذاشته و رزرو می شود ولی در عمل الزامی به این قضیه نیست (مثلا آدرسهای صفر تا FFFF برای حافظه و بعد از آن برای ثباتهای واسط)
- در کامپیوترهای که ساختار memory-mapped دارند، دستورات ارجاع به حافظه می توانند برای دسترسی به I/O استفاده شوند. این یک مزیت حساب می شود چون معمولا در کامپیوترها معمولی، تعداد دستورات ارجاع به حافظه و تنوع آنها نسبتا به دستورات I/O بالاتر است.

مثالی از یک مدار واسط (interface)ورودی/خروجی



به سمت CPU

به سمت دستگاه I/O

ثباتی که انتخاب می‌شود	RS۰	RS۱	CS
هیچ ثباتی انتخاب نمی‌شود	x	x	۰
ثبات پورت A	۰	۰	۱
ثبات پورت B	۱	۰	۱
ثبات کنترل	۰	۱	۱
ثبات وضعیت	۱	۱	۱

تبادل اطلاعات آسنکرون

- اگر دستگاه های که به هم متصل می گردند، از کلاک یکسانی استفاده نکنند، گفته می شود که آنها آسنکرون هستند.
- غالباً دستگاه هایی که به CPU متصل می شوند، کلاک داخلی خود را دارند و سرعت عملکرد آنها با CPU متفاوت است، لذا به صورت آسنکرون با CPU متصل هستند.
- برای تبادل اطلاعات بین ادوات آسنکرون، احتیاج به سیگنالهایی کمکی می باشد که آماده بودن دستگاه ها برای ارسال و تبادل اطلاعات را مشخص کند.

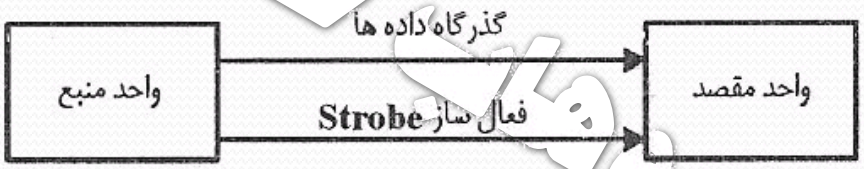
روشهای تبادل اطلاعات آسنکرون

1. استفاده از سیگنال Strobe
   - اعلام آغاز ارسال اطلاعات توسط سیگنال Strobe
2. استفاده از سیگنالهای به همراه داده جهت اعلام ارسال اطلاعات و سیگنالی از سمت گیرنده به مفهوم تأیید دریافت (HandShaking)

- این دو روش تنها بین CPU و I/O استفاده نمی شود بلکه بین هر دو واحدی که به صورت آسنکرون با هم در تبادل داده باشند، استفاده می شود.
- واحد فرستنده اطلاعات را منبع و واحد گیرنده اطلاعات را مقصد نامگذاری می کنیم.
- ترتیب تبادل اطلاعات و سیگنالهای بین منبع و مقصد را با دیاگرامهای زمانی (Timing Diagram) نمایش می دهند.

کنترل Strobe – آغازگر منبع

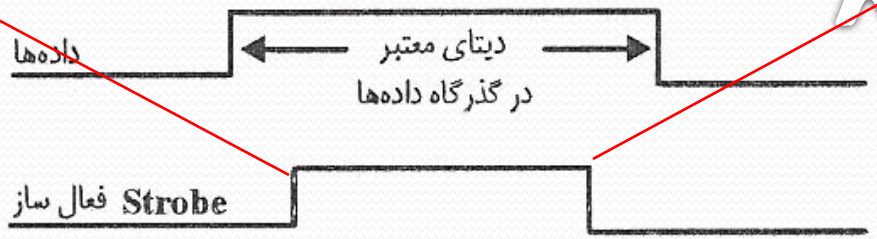
- **آغاز گر ارسال اطلاعات می تواند منبع یا مقصد باشد:**
  - اگر آغازگر منبع باشد Strobe به معنی آغاز ارسال اطلاعات است
  - اگر آغازگر مقصد باشد Strobe به معنی آغاز دریافت اطلاعات است



الف : ساختار تشکیلاتی

آغاز ارسال اطلاعات معتبر

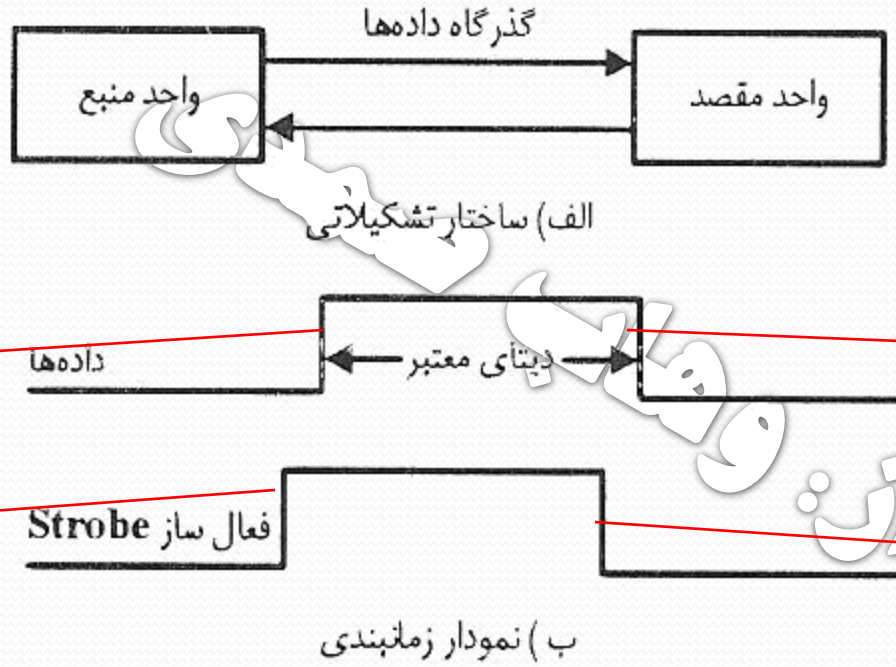
پایان مخابره اطلاعات معتبر



ب : نمودار زمانبندی

انتقال اطلاعات با فعال کردن سیگنال Strobe توسط واحد منبع

کنترل Strobe – آغازگر مقصد



شکل (۱۱-۴) انتقال اطلاعات با فعال کردن سیگنال Strobe توسط واحد مقصد

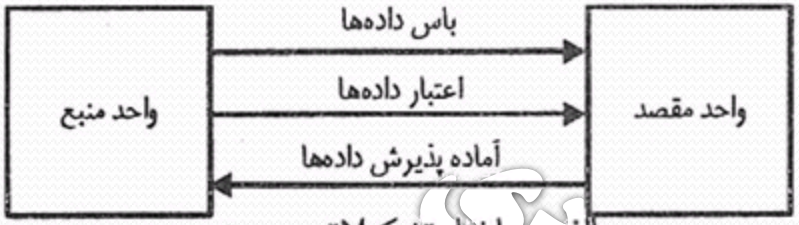
ارتباط طرفین از طریق سیگنال Strobe معمولاً برای تبادل اطلاعات بین CPU و Interface استفاده می شود.

HandShaking

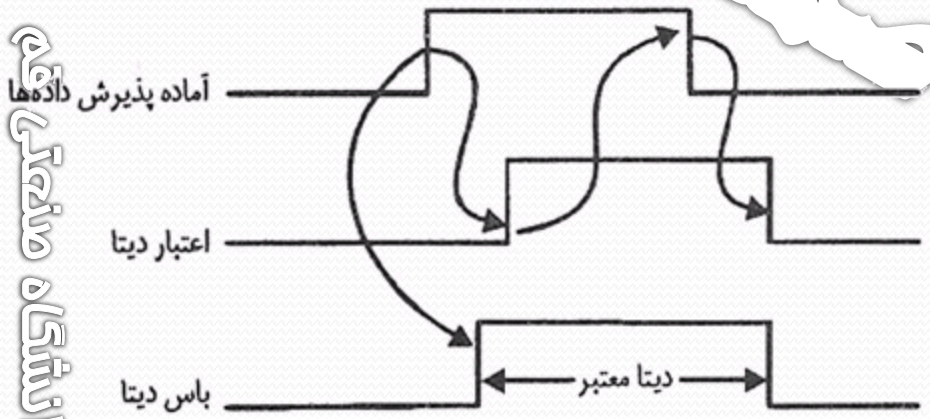
- از اشکالات روش کنترل Strobe این است که واحد فرستنده مکانیزمی برای کنترل صحت دریافت اطلاعات در گیرنده ندارد
  - اگر گیرنده آماده دریافت نباشد به دلایلی نظیر پربودن بافر دریافت یا خاموش بودن
  - نویز در جریان اطلاعات
- در روش Handshaking به جای استفاده از یک سیگنال، از دو سیگنال کنترل (یا دو سیم) استفاده می شود. در این حالت پس از اعلان آمادگی یکی از طرفین توسط سیگنال اول، طرف دیگر با استفاده از سیگنال دوم آمادگی خود را اعلام می کند.

این روش برای ارتباط بین Interface و دستگاه I/O استفاده می شود

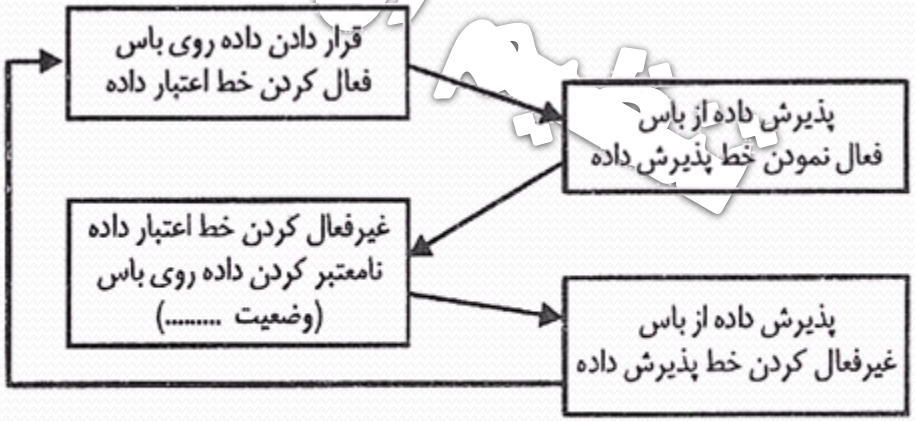
Handshaking – آغازگر مقصد



الف : ساختار تشکیلاتی سیستم

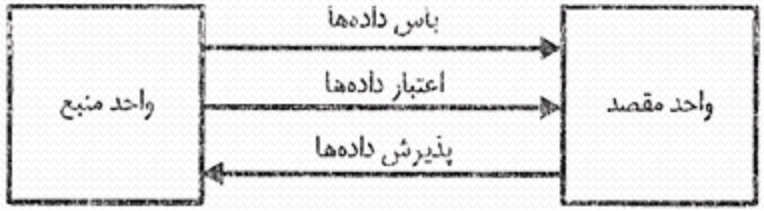


ب : نمودار زمانبندی

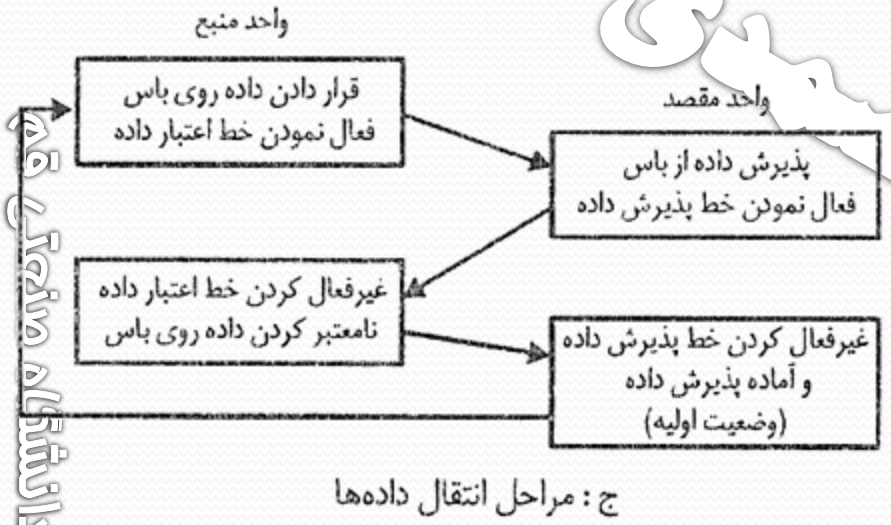


ج : مراحل انتقال داده‌ها

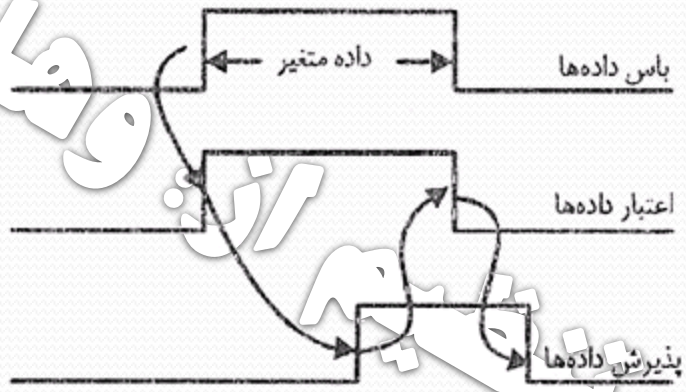
Handshaking – آغازگر منبع



الف) ساختار تشکیلاتی



ج : مراحل انتقال داده‌ها



ب) نمودار زمانی

در روش HandShaking اگر در زمان مشخصی پاسخ از طرف مقابل دریافت نشود، پیغام TimeOut به CPU ارسال می شود که در قبال آن این سیگنال می تواند آغازگر وقفه ای جهت انجام اقدام مناسب در CPU باشد

انتقال اطلاعات آسنکرون سری (Serial)

- در انتقال اطلاعات موازی برای ارسال n بیت، صرفاً به n سیم برای ارسال داده در آن واحد نیاز داریم.
- انتقال اطلاعات سریال ارزانتر است چون علاوه بر سیم زمین از یک سیم دیگر جهت تبادل داده استفاده می کند.
- در روش انتقال اطلاعات سری در فواصل دور، سیستمهای پالس ساعت متفاوتی دارند که در لحظات خاص با یکدیگر هم زمان می شوند.
- در ارسال آسنکرون اطلاعات تنها زمانی که انتقال اطلاعات در جریان است خط انتقال اشغال است و در بقیه حالات خط در وضعیت بیکار است (idle).

انتقال اطلاعات آسنکرون سری (Serial)

- سرعت مخابره اطلاعات بین دو طرف مشخص است.
- آغاز ارسال اطلاعات با بیتهای شروع نمایش داده می شود
- پس از آن داده ارسال می شود و اگر لازم باشد بیتهای توازن پس از آن
- در انتها تعدادی بیت به معنی پایان ارسال اطلاعات ارسال می شود

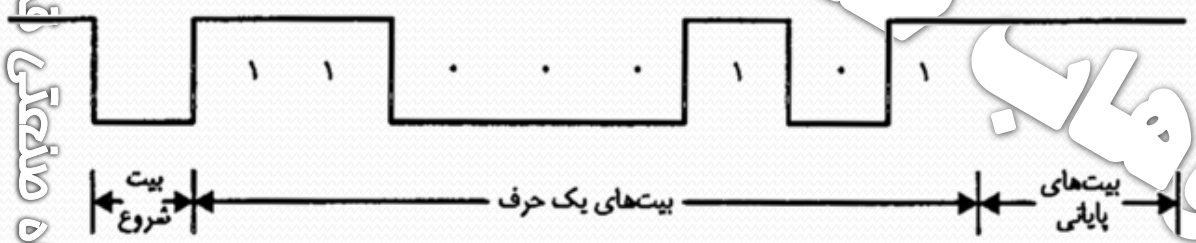
- تعداد بیتهای شروع، پایان، داده و توازن مشخص است.
  - بیتهای شروع: ۱ بیت
  - بیتهای پایان: ۱ بیت یا ۱/۵ بیت – در سیتمهای قدیمی ۲ بیت

انتقال اطلاعات آسنکرون سری (Serial)

- **در هنگام ارسال اطلاعات:**
  - بیت آغازین صفر است
  - سپس بیتهای داده مخابره می شوند
  - در انتها بیتهای پایان که مقدار آن یک است ارسال می شود.
  - یک بودن خط به معنی idle بودن است.
- **قوانین سمت گیرنده:**
  - موقعی که داده ای ارسال نمی شود، خط در حالت یک قرار دارد
  - شروع ارسال بوسیله بیت شروع صفر مشخص می شود
  - بیتهای حرف همیشه به دنبال بیت شروع مخابره می شوند
  - بعد از ارسال آخرین بیت، بیت پایان که حداقل به اندازه یک بیت زمان می برد و مقدار آن یک است مخابره می گردد.

سرعت مخابره اطلاعات

- فرض کنید سیستمی ۱۰ حرف در ثانیه ارسال می کند
- هر حرف متشکل است از:
  - یک بیت شروع
  - هفت بیت داده
  - یک بیت توازن
  - دو بیت پایان



- تعداد بیت انتقالی در ثانیه را Baud Rate یا bps می گویند.

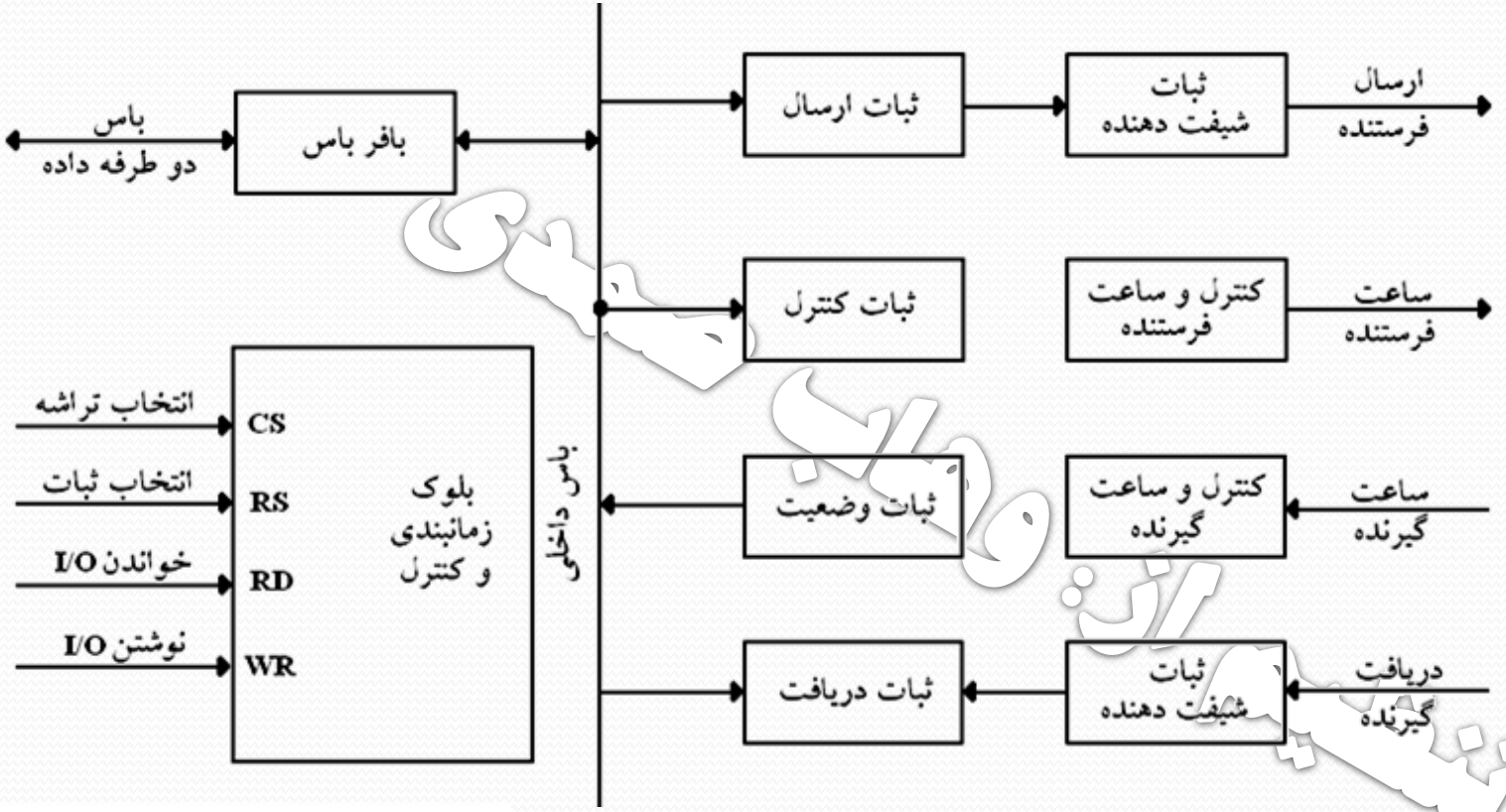
Baud Rate = Characters per second × bits in each character

Baud Rate = 10 × 11 = 110

نرخ انتقال اطلاعات معادل ۱۱۰ باد (Baud)

- مدار واسط مخابره کننده، ۱۱ بیت را دریافت می کند و ۸ بیت داده آن را برای پردازشگر می فرستد.
- در هنگام دریافت اطلاعات از CPU، ۸ بیت داده را دریافت و آن را در قالب ۱۱ بیتی ارسال می کند.
- این نوع مدار واسط در قالب IC ساخته شده است و نام آن UART است (Universal Asynchronous Receiver Transmitter)

مدار واسط آسنکرون (مثال)



CS	RS	Operation	Register selected
0	×	×	None: data bus in high-impedance
1	0	WR	Transmitter register
1	1	WR	Control register
1	0	RD	Receiver register
1	1	RD	Status register

ثبات کنترل:

کنترل مکانیزم عملکرد مدار واسط (سرعت، تعداد بیتهای شروع و پایان)

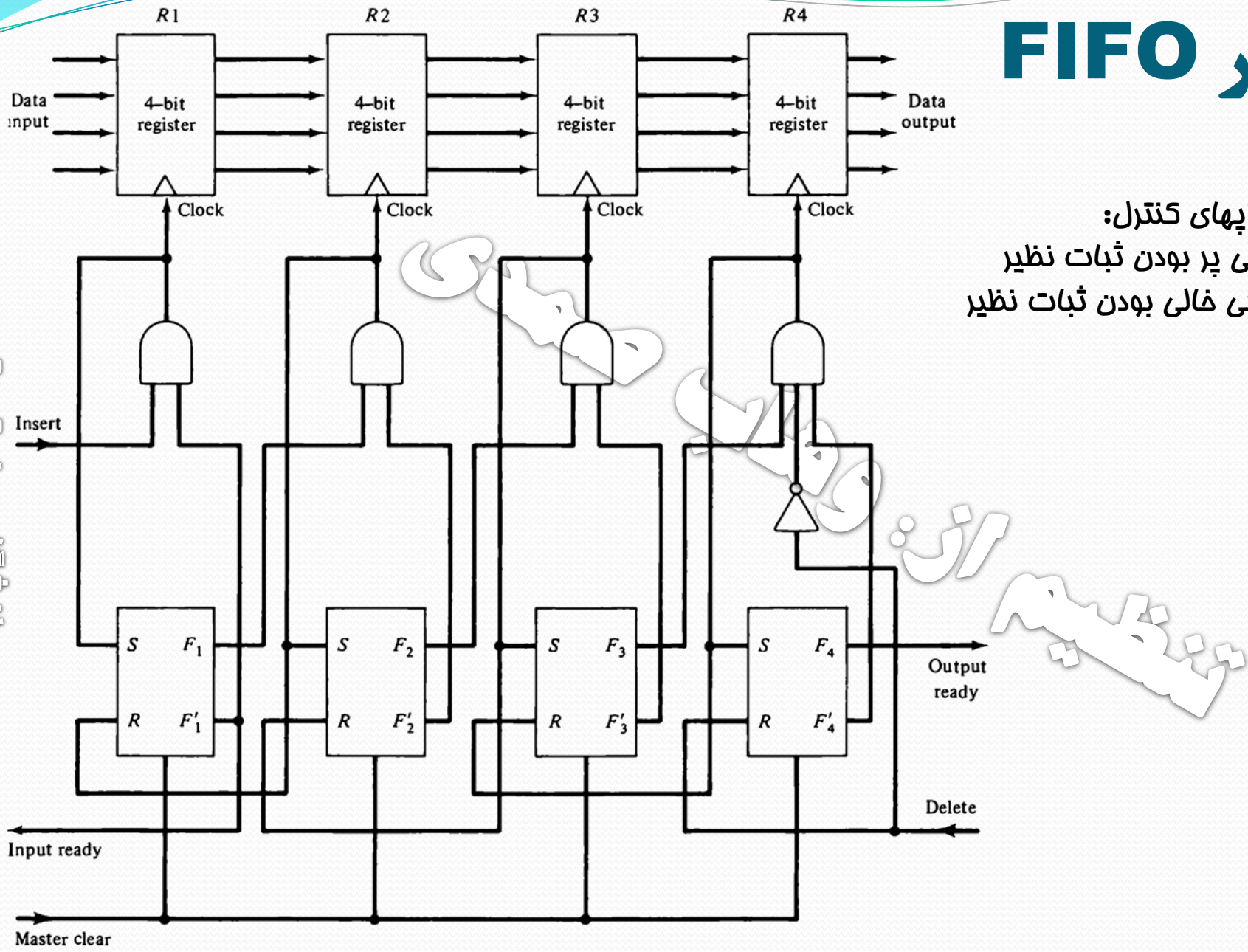
ثبات وضعیت:

نمایانگر وضعیت ارسال /دریافت

خطاهای متداول:

Overwrite Error ،Parity Error ،Faming Error

مدار FIFO



فلیپ فلاپهای کنترل:

۱ به معنی پر بودن ثبات نظیر

۰ به معنی خالی بودن ثبات نظیر

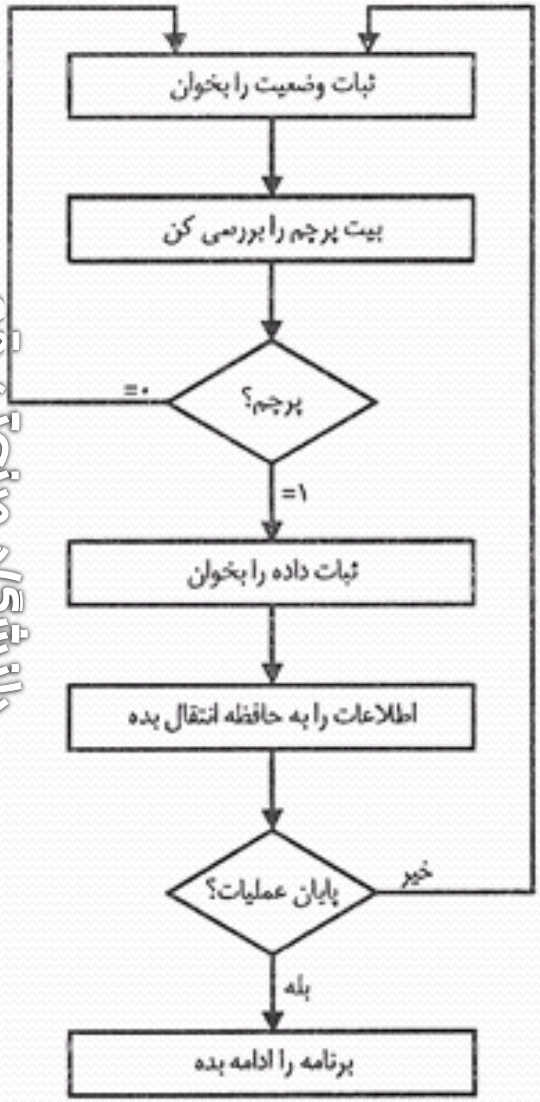
روشهای تبادل داده I/O

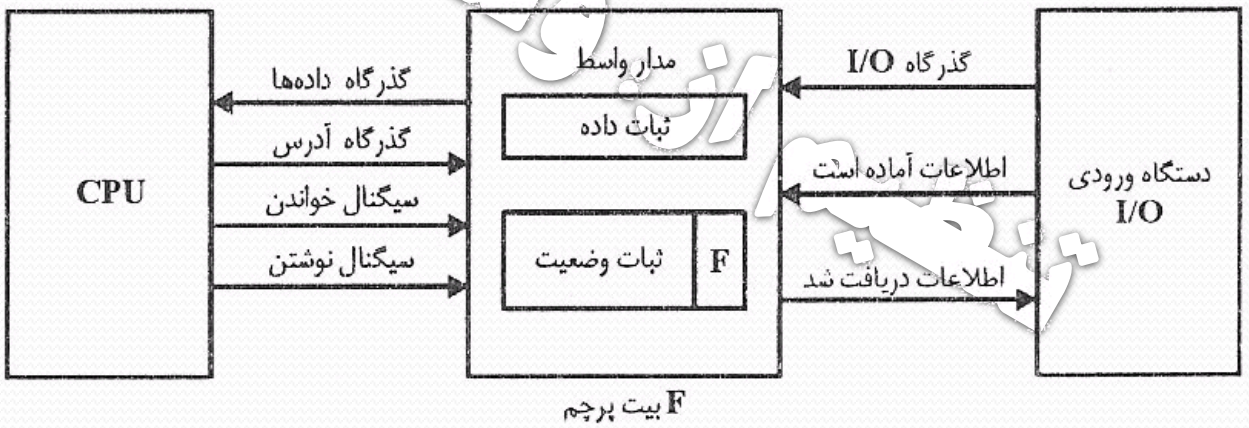
- تبادل داده با I/O از طریق برنامه (Programmed I/O)
  - زمانبر
- تبادل داده با I/O از طریق وقفه (Interrupted Initiated I/O)
  - پاسخ به درخواستهای I/O از طریق روتینهای وقفه
- دسترسی مستقیم به حافظه (DMA)
  - دسترسی دستگاه های I/O به اطلاعات به صورت مستقیم بدون دخالت پردازنده

- برخی سیستمهای مدارهای منطقی لازم برای پیاده سازی Interface و DMA را با هم در یک مدار پیاده سازی می کنند که به آن پردازنده ورودی/خروجی (IOP) می گویند.

مثال تبادل داده با I/O از طریق برنامه

- برنامه ای جهت ورود اطلاعات به CPU
  - اشکال این روش در انتظار فعال (Busy Waiting) پردازنده است





Interrupted Initiated I/O

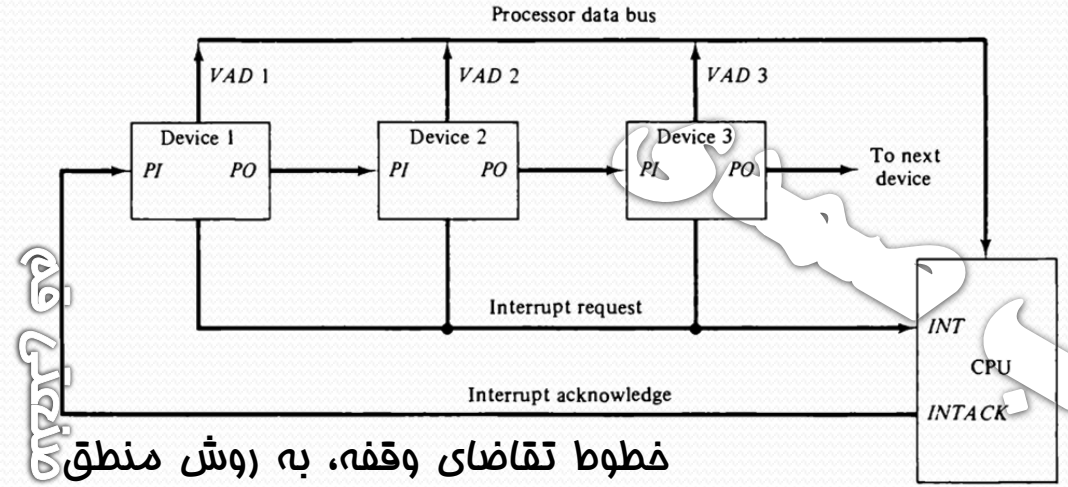
- **هنگامی که وقفه ای ایجاد می شود، پردازنده روند کاری خود را متوقف می کند و باید در PC آدرس روتین وقفه را ذخیره کند تا آدرس روتین وقفه شروع به اجرا کند.**
- **حال نوبت به آن می رسد که روتین وقفه فراخوانی شود. سیستمها به دو نوع آدرس ابتدای روتین وقفه را پیدا می کنند:**
  - الف- روش غیربرداری: بدون توجه به این که منبع وقفه کجاست، آدرس روتین وقفه ثابت است.
  - ب- روش برداری: با توجه به منبع وقفه آدرس شروع روتین وقفه از بردار وقفه استخراج می شود.
    - بردار وقفه یا آدرس ابتدای روتین وقفه را دارد یا به صورت غیر مستقیم آدرس آن را ذخیره کرده (یعنی آدرس خانه ای را دارد که آدرس روتین وقفه در آن است)

اولویت وقفه

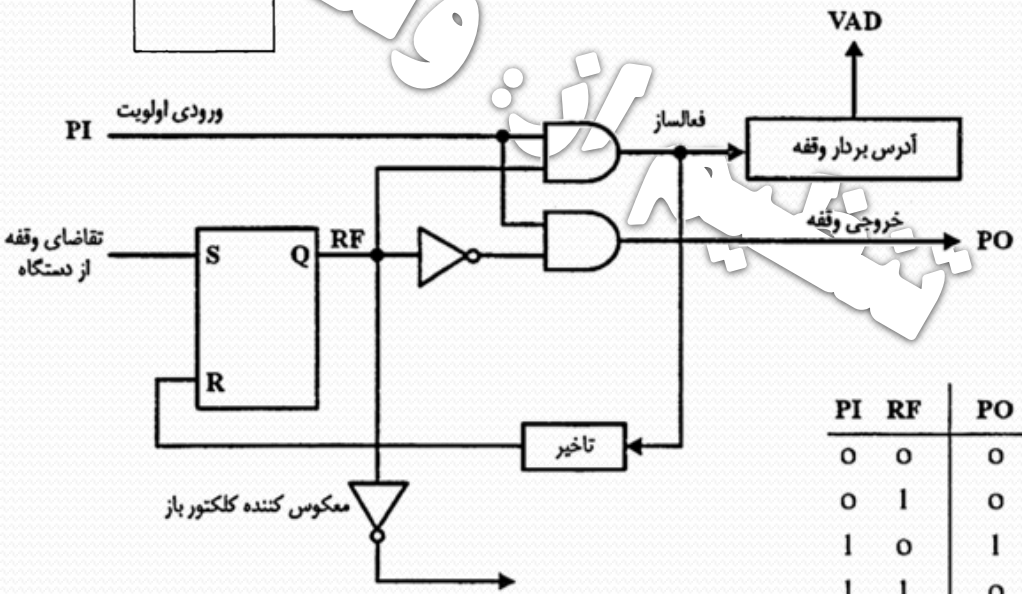
- بعد از دریافت وقفه در پردازنده و ذخیره اطلاعات بازگشت، نوبت به سرویس دهی به منبع وقفه می رسد.
- اگر تعداد منابع تولیدکننده وقفه در آن واحد بیش از یکی باشد، باید انتخاب کنیم به کدامیک سرویس بدهیم.
- روش تعیین اولویت می تواند نرم افزاری باشد، یعنی پردازنده جستجو کند که منبع وقفه کدام است و به اولویت مشخص شده خود به آنها سرویس بدهد (به این روش Poolingمی گویند)
- روش دیگر استفاده از مدارات سخت افزاری است که پردازنده درگیر تعیین اولویت نشود.

مدارهای تعیین اولویت

- تعیین اولویت زنجیره ای (Daisy Chain)



خطوط تقاضای وقفه، به روش منطق سیم بندی به هم متصل هستند.
اگر یکی از سیمهای خروجی صفر شود، مقدار روی سیم صفر می شود و تقاضای وقفه فعال در نظر گرفته می‌شود.



PI	RF	PO	فعالساز
0	0	0	0
0	1	0	0
1	0	1	0
1	1	0	1

اولویت وقفه موازی

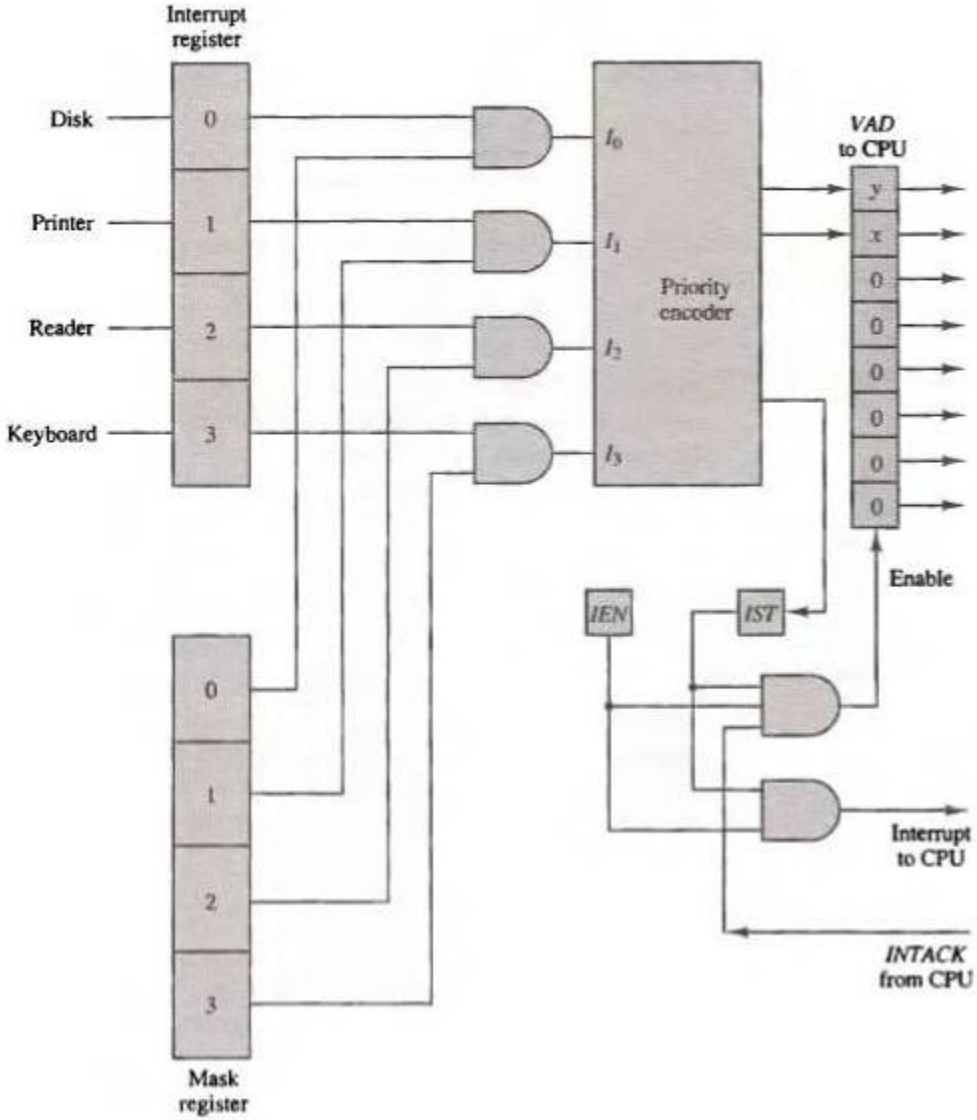


TABLE 11-2 Priority Encoder Truth Table

Inputs				Outputs			Boolean functions
I_0	I_1	I_2	I_3	x	y	IST	
1	×	×	×	0	0	1	
0	1	×	×	0	1	1	$x = I_0' I_1'$
0	0	1	×	1	0	1	$y = I_0' I_1 + I_0' I_2'$
0	0	0	1	1	1	1	$(IST) = I_0 + I_1 + I_2 + I_3$
0	0	0	0	×	×	0	

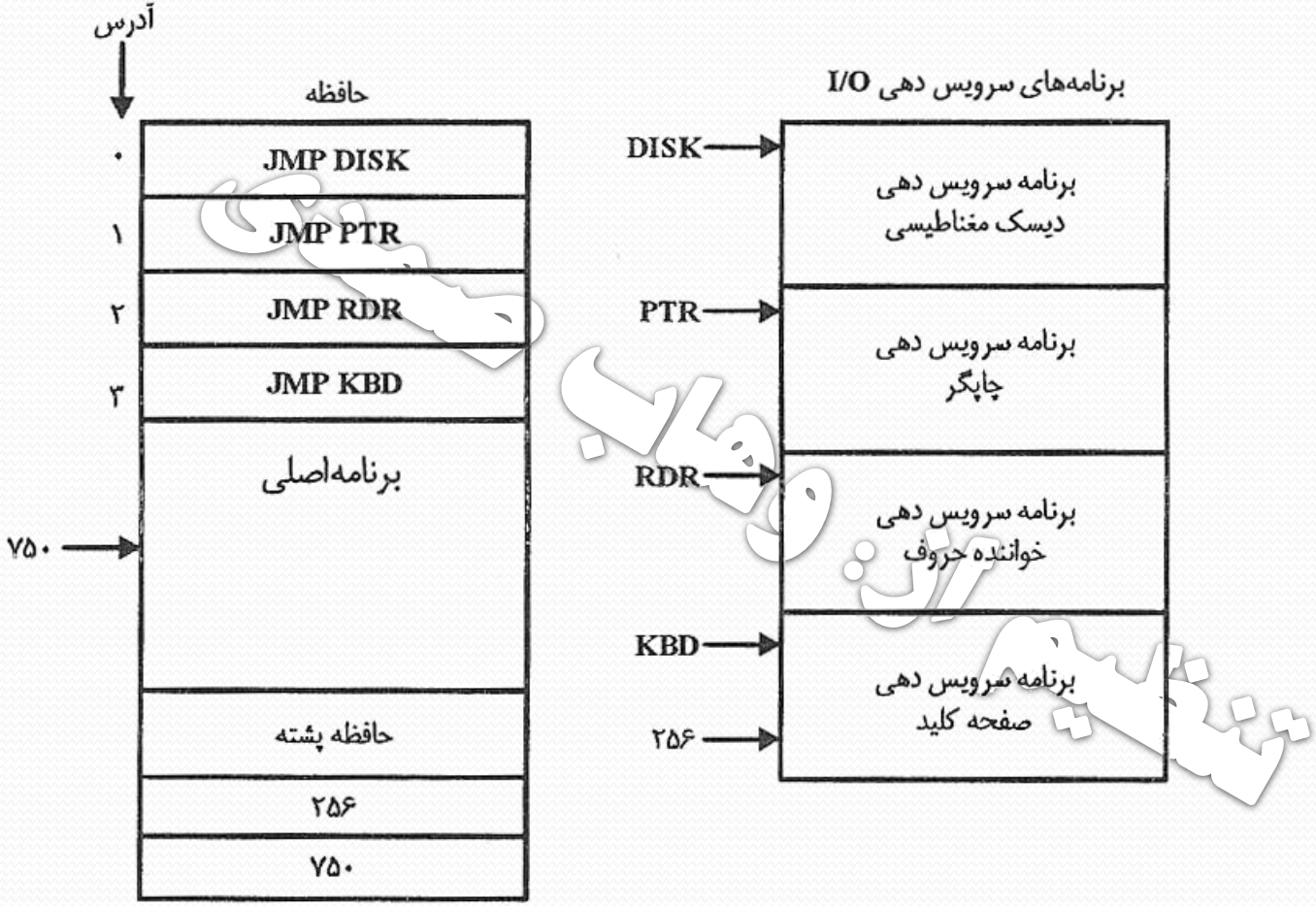
سیکل وقفه (Interrupt Cycle)

$SP \leftarrow SP - 1$	Decrement stack pointer
$M[SP] \leftarrow PC$	Push PC into stack
$INTACK \leftarrow 1$	Enable interrupt acknowledge
$PC \leftarrow VAD$	Transfer vector address to PC
$IEN \leftarrow 0$	Disable further interrupts

Go to fetch next instruction

ملاحظات نرم افزاری

- سیستم می تواند به گونه ای طراحی شود که وقفه های با اولویت بالاتر روتین وقفه های با اولویت پایین تر را دچار وقفه کند.
- چون وقفه های با اولویت بالاتر به ادوات با سرعت بالاتر سرویس می دهند، ایجاد وقفه در روتین یک وقفه اولویت پایین که به دستگاهی با سرعت پایین متصل است، خللی در کار دستگاه سرعت پایین تر (با اولویت پایین تر ایجاد نخواهد کرد)
- در هر روتین وقفه در انتهای کار پس از پایان روتین، یک دستور Return قرار داده می شود تا برنامه روند کاری عادی خود را از سر گیرد



شکل (۱۱-۱۵) برنامه‌هایی که برای سرویس وقفه، در حافظه قرار گرفته‌اند

آغاز وقفه/پایان وقفه

- **آغاز وقفه**
- بیتهای ماسک وقفه های اولویت پایین تر صفر می شوند (تا درخواست وقفه اولویت پایین به پردازنده نرسد)
- فلیپ فلاپ IST صفر می شود (یعنی وقفه دریافت شد و آماده دریافت وقفه های بعد هستیم)
- ثباتهای پردازنده ذخیره می شوند (معمولاً در پشته نگهداری می شوند)
- IEN یک می شود
- روتین سرویس وقفه اجرا می شود

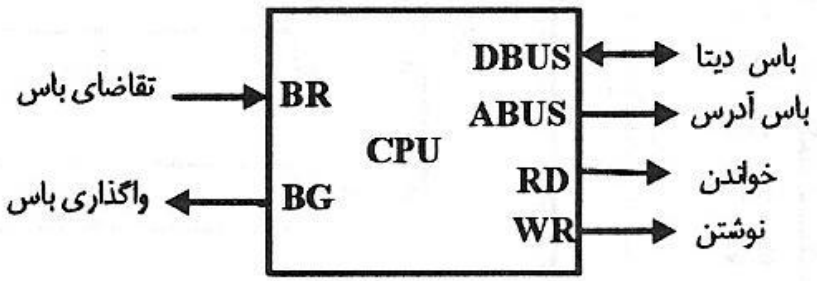
- **پایان وقفه**
- IEN صفر می شود
- ثباتهای پردازنده بازیابی می شوند
- بیت وقفه مرتبط با وقفه فعلی صفر می شود (یعنی سرویس این وقفه اجرا شد و منبع وقفه فعلی می تواند دوباره وقفه تولید کند)
- بیتهای ماسک وقفه های با اولویت پایینتر یک می شوند
- IEN یک می شود

دسترسی مستقیم به حافظه (DMA)

- ورود پردازنده به چرخه انتقال اطلاعات بین حافظه و ادوات I/O به دلیل ماهیت نرم افزاری فرآیند، موجب کند شدن انتقال اطلاعات می شود.
- در موارد زیر DMA کاربرد دارد:
  - سرعت انتقال وتولید اطلاعات آنقدر زیاد که باعث می شود CPU از پاسخگویی به این جریان داده جا بماند
  - سرعت دستگاه I/O آنقدر پایین است که زمان زیادی از CPU هدر می شود
- علاوه بر تبادل اطلاعات با I/O، DMA جهت تبادل اطلاعات بین چیپها در کامپیوتر، تبادل اطلاعات حافظه-به-حافظه و تبادل اطلاعات درون چیپ در سیستمهای چند هسته ای استفاده می شود.

دسترسی به گذرگاه در روش DMA

- هنگامی که دستگاه جانبی می خواهد به حافظه دسترسی پیدا کند، از طریق کنترل کننده DMA به حافظه اصلی دسترسی می یابد.
- این کنترلر نیز توانایی نوشتن روی گذرگاه حافظه را دارد و جهت جلوگیری از اختلال در عملکرد CPU باید با CPU هماهنگ باشد.
- این هماهنگی توسط سیگنالهای Bus Request و Bus Granted انجام می شود.



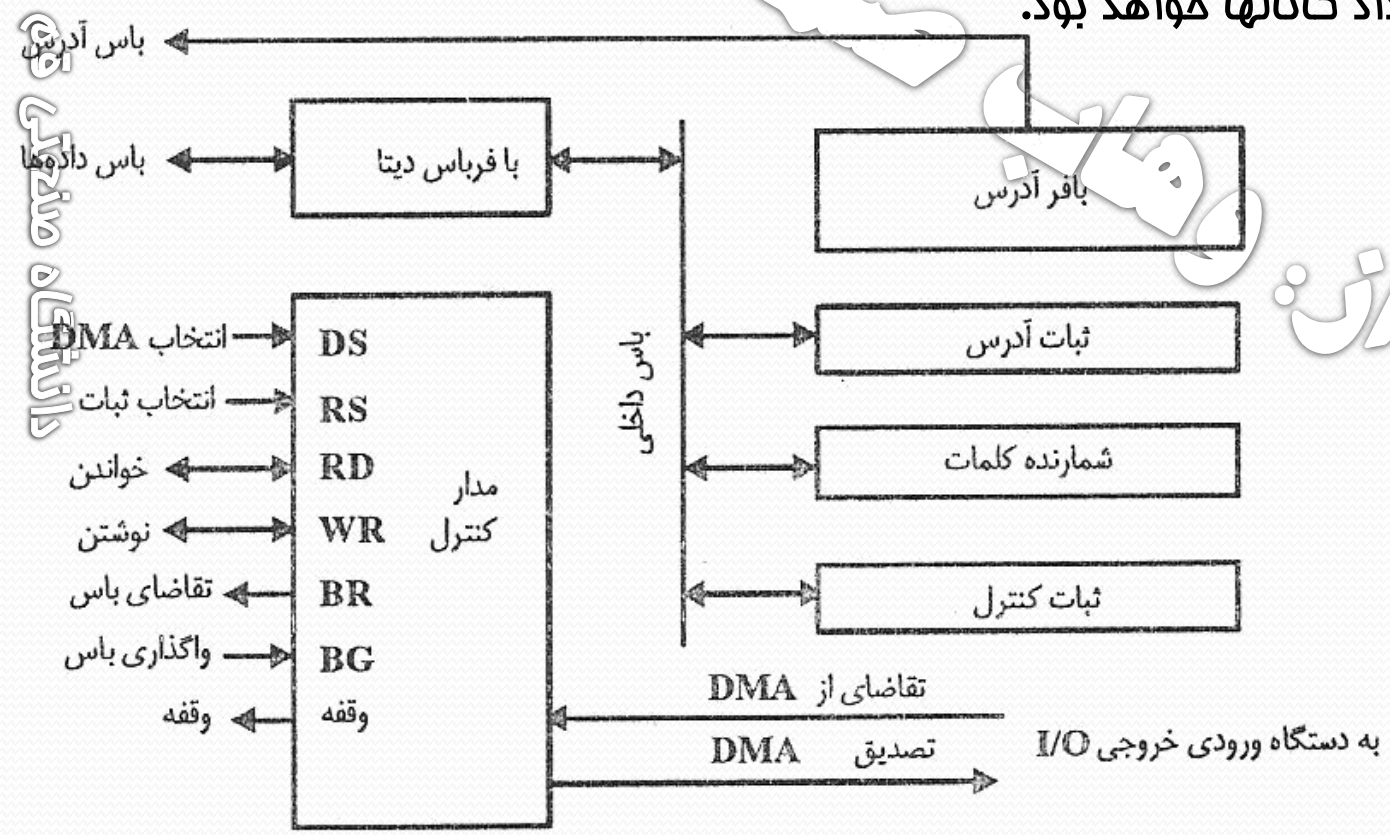
شکل (۱۱-۱۶) سیگنالهای باس CPU برای انتقال با DMA

دسترسی به Bus در DMA

- **کنترل کننده DMA به سه روش به حافظه دسترسی پیدا می کند:**
- Burst: بعد از گرفتن گذرگاه حجم بزرگی از اطلاعات به صورت بلوک به بلوک به حافظه منتقل می شود. این روش برای ادواتی که اطلاعات را عمده می خوانند مانند هارددیسک مناسب است اما اشکال این روش در این است که پردازنده در خلال این مدت باید منتظر برای امکان دسترسی دوباره به حافظه بماند.
- Cycle Stealing: در این روش DMA Controller بعد از هر بار بدست آوردن گذرگاه یک بایت می فرستد و دوباره گذرگاه را آزاد می کند و مجدداً در فرصتی دیگر دوباره گذرگاه را درخواست می کند. در این روش پردازنده می تواند به کارهای خود بپردازد اما روند انتقال اطلاعات در سمت I/O نسبت به حالت Burst کندرت است. این روش برای ادواتی مناسب است که وضعیت آنها باید مداوماً کنترل و Monitor شود.
- Transparent: در این روش DMA Controller با استفاده از یک سرسی مدارات جانبی توانایی تشخیص این را به دست می آورد که چه زمانی پردازنده نیازی به استفاده از گذرگاه حافظه ندارد و در این مقطع ارسال اطلاعات را انجام می دهد. این روش به دلیل نیاز به مدارات اضافه تر، هزینه بیشتری برای پیاده سازی نیاز دارد

کنترلر DMA

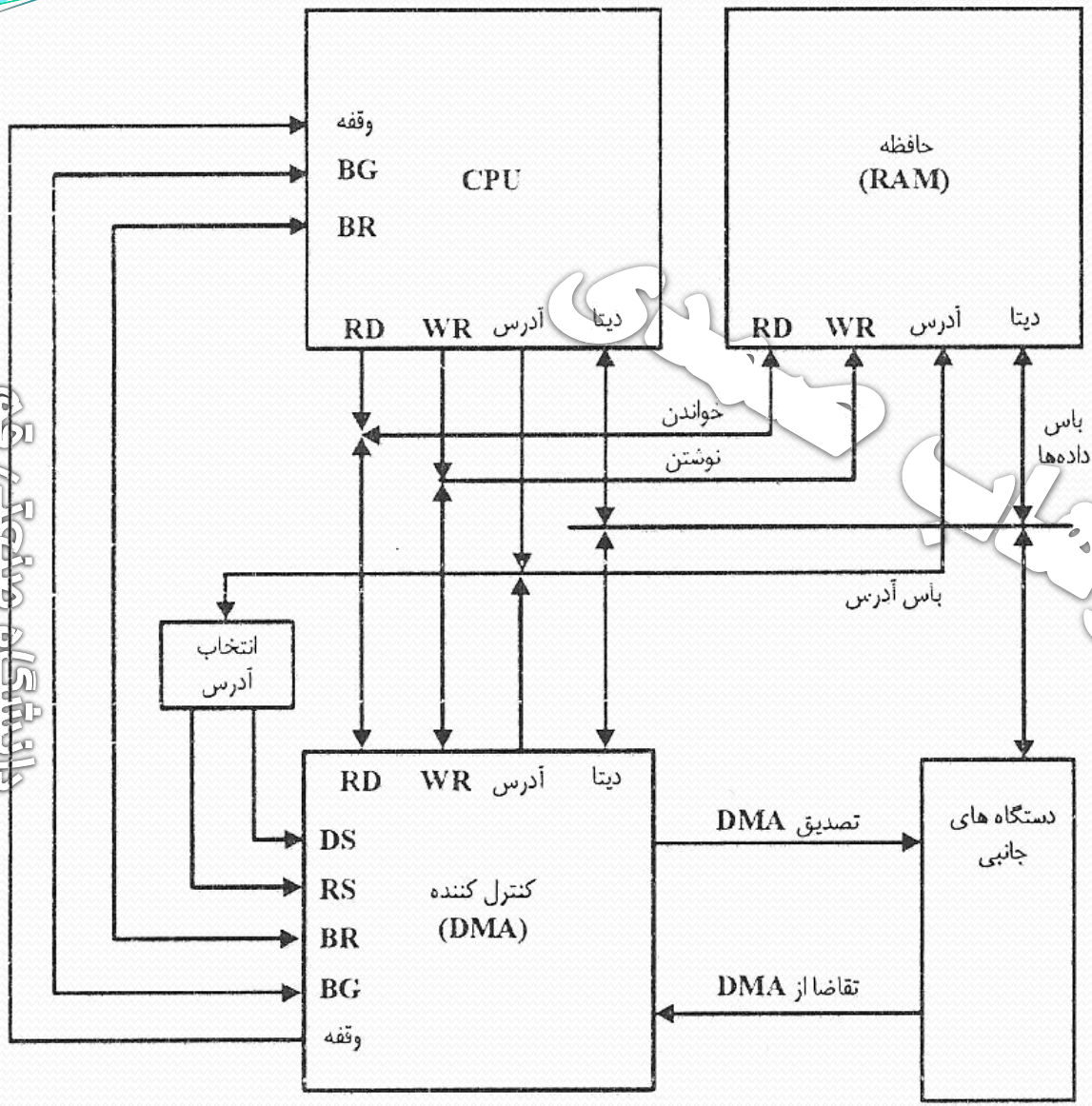
- کنترلر DMA خود به عنوان یک دستگاه جانبی به پردازنده متصل است و پردازنده در آغاز عملیات انتقال اطلاعات آن را مقداردهی می کند و بقیه اعمال توسط کنترلر DMA مدیریت می شود.
- خطوط RD و WR دوطرفه هستند.
- کنترلر ممکن است دارای چند کانال باشد که به دستگاه های مختلف متصل باشد در این صورت ثبات آدرس و شمارنده به تعداد کانالها خواهد بود.



چرخه مقداردهی کنترلر

- آدرس شروع بلوک داده در حافظه مشخص می شود
- تعداد کلمات در شمارنده کلمات ذخیره می شود
- اطلاعات کنترلی که نوع نوشتن/خواندن را مشخص می کنند در ثبات کنترل ثبت می شوند
- اطلاعات کنترلی که بیانگر آغاز عملیات هستند، نوشته می شوند

انتقال اطلاعات DMA

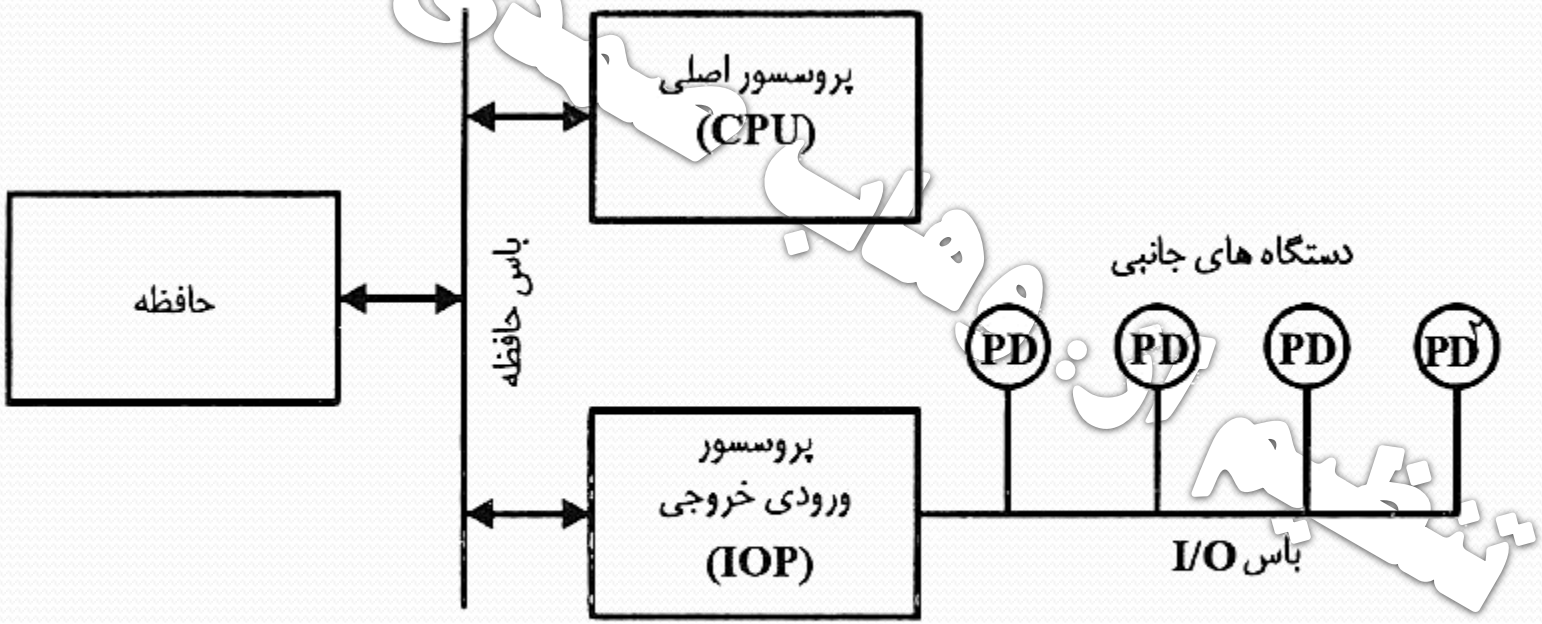


شکل (۱۱-۱۸) انتقال اطلاعات با DMA، در یک سیستم کامپیوتری

۱- مقدار دهی اولیه کنترلر DMA توسط پردازنده
۲- فعال شدن خط تقاضا از DMA
۳- فعال شدن BR
۴- آزادسازی گذرگاه توسط پردازنده و فعال کردن BG
۵- گذاشتن آدرس روی گذرگاه و در اختیار گرفتن خطوط RD و WR توسط کنترلر DMA
۶- خواندن/نوشتن اطلاعات از روی گذرگاه توسط وسیله جانبی
۷- به روز رسانی آدرس در کنترلر
۸- کنترل خط تقاضا از DMA
۹- در صورت ۰ بودن، BR صفر می شود وگرنه به گام ۵ می رویم
۱۰- پردازنده باس را در اختیار می گیرد و کنترل می کند آیا ارسال تمام شده یا خیر

پردازنده ورودی/خروجی (IOP)

- پردازنده می تواند عملیات ورودی/خروجی را به پردازنده ورودی/خروجی واگذار کند. این پردازنده ها خود تواناییهایی نظیر اجرای دستورات I/O، واکشی دستورات از حافظه و اجرای دستورات ساده منطقی و ریاضی را دارند.
- این پردازنده ها برای انجام عملیات I/O بهینه سازی شده اند.
- به دلیل امکانات واکشی و اجرای دستور، دیگر مانند DMA احتیاج به مداخله پردازنده مرکزی (CPU) در انجام عملیات I/O نیست.
- معمولاً CPU وظیفه مقداردهی اولیه برنامه های I/O را بر عهده دارد و بعد از آن کارها به پردازنده I/O واگذار می شود. پردازنده I/O در صورت لزوم وقفه های لازم جهت اطلاع دادن به CPU را صادر می کند.

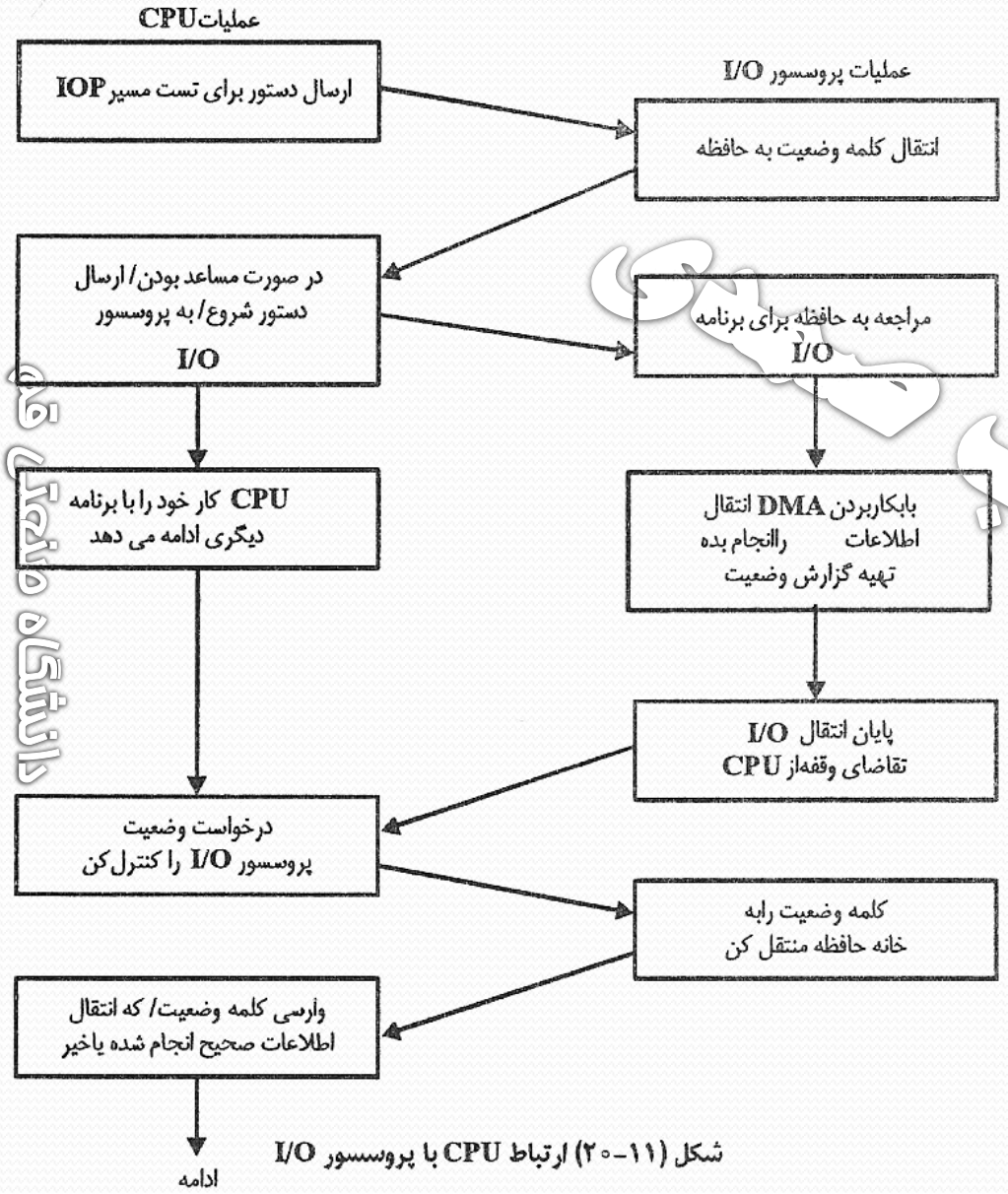


شکل (۱۱-۱۹) ساختار تشکیلاتی یک کامپیوتر با پروسسورهای I/O

- **مدل ارتباطی IOP:**
- **IOP با تجهیزات I/O به روش Program Controlled عمل می کند.**
- **ارتباط IOP با حافظه مانند DMA است.**
- **ارتباط IOP با پردازنده به مدلهای مختلفی است:**
  - در سیستمهای خیلی پیچیده هر کدام از آنها مستقل عمل می کنند
  - در غالب سیستم ها CPU حالت Master و IOP حالت Slave را دارد.
  - پردازنده وظیفه آغاز تمامی عملیات را دارد و IOP وظیفه اجرای عملیات I/O را.
  - دستورات پردازنده منجر به عملیات I/O می شوند و پردازنده وضعیت IOP را بررسی می کند تا نسبت به آن، تصمیم مناسب را بگیرد.
  - در سوی دیگر IOP برای اعلان به CPU از وقفه استفاده می کند

- برای تفکیک عناوین کلمات، دستوراتی که را IOP از حافظه می خواند command(فرمان) می نامیم.
- فرمانها توسط برنامه نویسهای خبره تهیه می شوند و در حافظه قرار می گیرند. کلمات فرمان جنبه برنامه اجرایی IOP را دارند.
- پردازنده هنگامی که احتیاج به اجرای عملیات I/O پیش می آید، آدرس ذخیره فرمانها در حافظه را به IOP اعلام می کند.

ارتباط CPU و IOP



شکل (۱۱-۲۰) ارتباط CPU با پروسسور I/O

- ارتباط بین CPU و IOP معمولاً با استفاده از حافظه مشترک صورت می گیرد. در این حالت حافظه نقش یک مرکز تبادل پیام بین دو سمت را بازی می کند.
- IOP مسؤول کنترل تبادل داده بین I/O وحافظه است. به همین خاطر در زمانهایی متفاوت با CPU در رقابت جهت دسترسی به حافظه است.
- به همین دلیل تعداد ادواتی که در آن واحد می توانند با CPU کار کنند با زمان دسترسی حافظه در ارتباط است و نمی توان آن را خیلی افزایش داد.
- بعضی ادوات جانبی مانند هارددیسک به سیکلهای زیادی برای تبادل داده با حافظه نیاز دارند، به همین خاطر در مدت ارتباط آنها با حافظه، CPU به ناچار مجبور به کاهشِ عملکرد خورد است تا زمانی که مجدداً بتواند به حافظه دسترسی پیدا کند.