بسم اللّه الرّحمن الرّحیم

USB در چند کلمه

ویرایش اوّل

برای اینکه چگونگی ارتباط USB را بهتر درک کنید

فرشید سفیدگران

کارشناسی کامپیوتر سخت افزار

مقدمه

به نام او که همه چیز از آن اوست.پروردگار بزرگ را شاکرم که دوباره به من این فرصت و انرژی را داد تا قدمی دیگر در جهت آموزش علوم رایانه ای بردارم.امیدوارم که شما خواننده عزیز نهایت استفاده را از این قسمت ببرید و راضی باشید.

واسط USB به خاطر ساده بودن برای کاربران معمولی و اینکه می توان بدون راه اندازی دوباره[1] سیستم عامل ، دستگاه را به آن متصل کرد و شروع به کار کرد بسیار مورد علاقه مردم قرار گرفته و روز به روز به کاربران آن افزوده می شوند.واسط USB در مقایسه با واسط RS-232[2] برای کسانی که می خواهند روی دستگاههای خود ازتباط با USB را داشته باشند بسیار پیچیده ودشوار است به علاوه برای این منظور به یک نرم افزار اداره کننده برای USB که همان درایور[3] است احتیاج دارند که در اختیار نداشتن این درایور نرم افزاری برای RS-232 باعث گسترش محبوبیت این استاندار انتقال داده شده است.ولی امروزه در سیستمهای کامپیوتری جدید دیگر اثری از خروجی های RS-232 نیست تقریبا می توان گفت که استاندارد RS-232 با وجود محبوبیت گسترده اش باز در مقابل استاندارد USB کبودهای زیادی دارد و کم کم میدان را به استاندارد پر سرعت USB تحویل داده است.

[1] Restart
[2] A popular communications protocol for connecting computers to modems and varied peripherals
[3] Device Driver

Interface	Format	Number of Devices (maximum)	Length (maximum, feet)	Speed (maximum, bits/sec.)	Typical Use
USB	asynchronous serial	127	16 (or up to 96 ft. with 5 hubs)	1.5M, 12M, 480M	Mouse, keyboard, disk drive, modem, audio
RS-232 (EIA/TIA-232)	asynchronous serial	2	50-100	20k (115k with some hardware)	Modem, mouse, instrumentation
RS-485 (TIA/EIA-485)	asynchronous serial	32 unit loads (up to 256 devices with some hardware)	4000	10M	Data acquisition and control systems
IrDA	asynchronous serial infrared	2	6	115k	Printers, hand-held computers
Microwire	synchronous serial	8	10	2M	Microcontroller communications
SPI	synchronous serial	8	10	2.1M	Microcontroller communications
I2C	synchronous serial	40	18	3.4M	Microcontroller communications
IEEE-1394 (Fire Wire)	serial	64	15	400M (increasing to 3.2G with IEEE-1394b	Video, mass storage
IEEE-488 (GPIB)	parallel	15	60	8M	Instrumentation
Ethernet	serial	1024	1600	10M/100M/ 1G	Networked PC
MIDI	serial current loop	2 (more with flow-through mode)	50	31.5k	Music, show control
Parallel Printer Port	parallel	2 (8 with daisy-chain support)	10-30	8M	Printers, scanners, disk drives

توصیف سخت افزاری USB

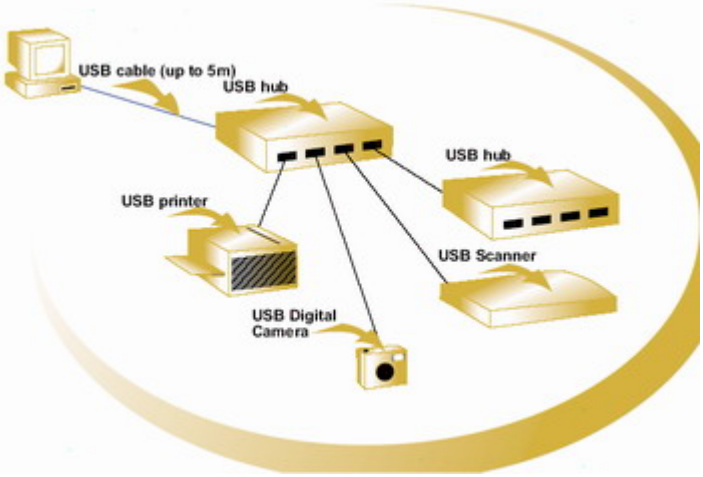
استاندارد USB یک واسط Plug-and-Play[4] بین یک کامپیوتر و یک دستگاه جانبی مانند:
audio players, joysticks, keyboards, mousses, telephones, scanners, printers, cool disks, cameras,...

به منظور برقراری یک ارتباط پر سرعت قرار می گیرد.

[4] یک استاندارد سخت افزاری برای اتصال آسان دستگاه جانبی به رایانه و شناختن خودکار نوع دستگاه توسط رایانه ، PNP

بوسیله ی پایانه ی USB می توان یک دستگاه جانبی جدید را بدون نیاز به خاموش کردن رایانه یا استفاده از یک کارت سخت افزاری به رایانه متصل کرد.استاندارد گذرگاه جانبی USB توسط شرکتهای Compaq, IBM, DEC, Intel, Microsoft, NEC, Northern Telecom ارائه شد و فناوری آن بدون اضافه کردن وسیله ی خاصی برای تمام رایانه ها و فروشنده ها فراهم شد.

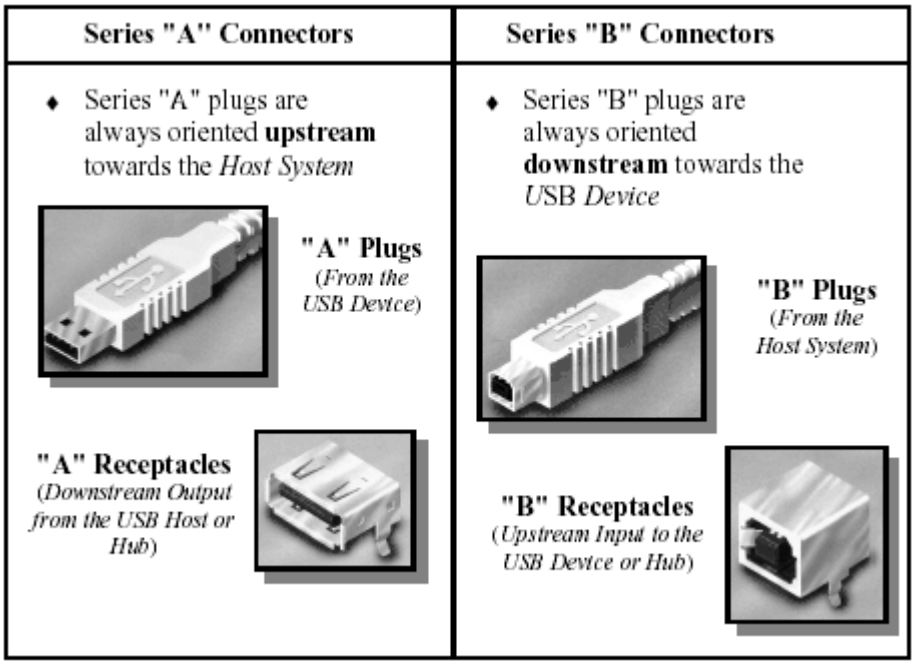
گذرگاه USB براي اتصال ۱۲۷ دستگاه مختلف طرح ريزي شده كه اين تعداد دستگاه به دو روش زنجيره اي يا استفاده از HUB به پورت متصل مي شوند البته خود USB در سوكتهاي مختلفي موجود است كه اين خود بر تعداد دستگاههاى اتصالي مي افزايد.ما مي توانيم به هر دستگاه HUB هفت ابزار جانبي وصل كنيم يا يك يا چند تا از اين هفت تا را به HUB ثانوي به منظور داشتن هفت مورد اتصال ديگر وصل مي كنيم.مي توان از طريق پورت USB يك منبع تغذيه ۵ ولتي را منتقل كرد البته براي دستگاههاي كوچك و كم مصرف استفاده مي شود زيرا دستگاههاي بزرگتر مانند اسكنر خود منبع تغذيه داخلي دارند.همچنين قادر به جرياندهي به اندازه ۵۰۰ ميلي آمپر است.



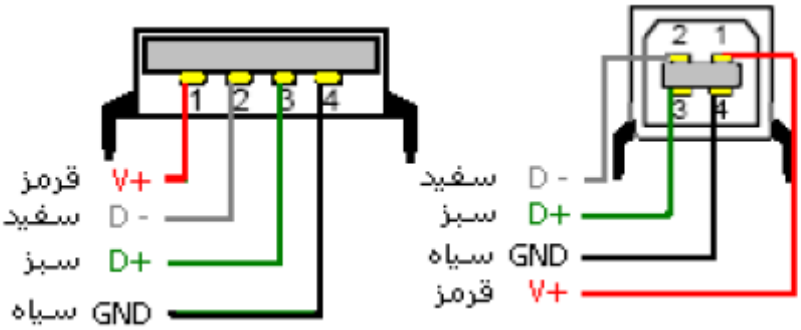
تصویر 1 : نحوه اتصال چندین دستگاه جانبی به رایانه از طریق HUB تا سقف ۱۲۷ دستگاه

تصویر 2 : انواع و اقسام فیشهای رایج USB .

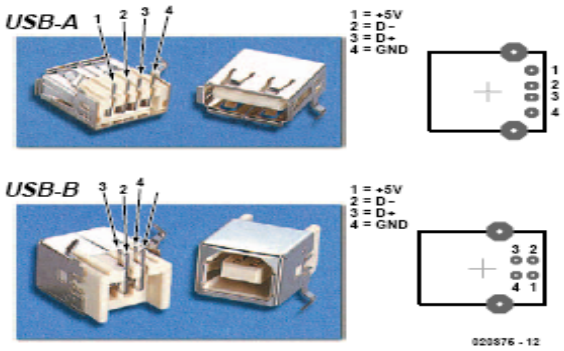
ابزارهاي جانبي مي توانند به طور مستقيم توسط سوكت A كه چهار پينی است و بعضاً سوكت B به كامپيوتر وصل شوند.نقش سوكت A و سوكت B در تصوير زير توضيح داده شده است:



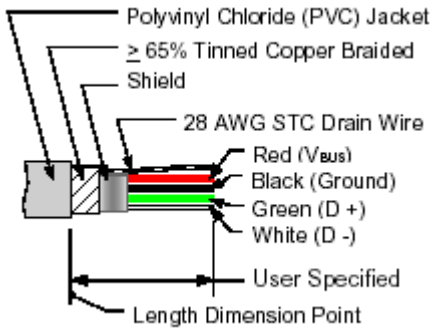
تصویر 3 : دو نوع سوکت رایج برای USB سمت راست نوع B برای دستگاههای کوچک که فضای کمی دارند و سمت چپ نوع A برای سایر دستگاههای بزرگتر لازم به ذکر است که این دو نوع از نظر اتصالات داخلی کاملا شبیه هستند و هیچ تفاوتی با هم ندارند فقط شکل فیزیکی آنها متفاوت است.



تصویر 4 : سمت راست سوکت روبردی مدل B و سمت چپ سوکت روبردی مدل A .



تصویر 5 : نمای واقعی از دو نوع فیش روبردی رایج USB



تصویر 6 : نمایش برشی طولی از یک کابل USB

طول کابل USB نباید بیشتر از ۵ متر باشد در غیر اینصورت به خاطر فرکانس بالای انتقال در USB ، داده ها همراه با خطا منتقل می شوند که این ناشی از ضعیف شدن توان سیگنال حاوی اطلاعات در طول کابل است که در نهایت باعث اثر گذاری انواع نویز ها روی سیگنال می شود.در ادامه توضیحات تخصصی تری در مورد نحوه انتقال داده ها و پروتکلها خواهد آمد.

پروتکلهای USB

بر خلاف استاندارد RS-232 یا سایر استانداردهای سریال که در آنها قالب داده های فرستاده شده تعریف نشده است ولی USB از چندین لایه ی پروتکل ساخته شده است.
هر انتقال USB شامل شده است از:

- Token Packet (سرباری برای تعریف بسته ی بعدی)
- Optional Data Packet (شامل بار مفید)
- Status Packet (به منظور انتقال ACK و فراهم کردن قابلیت تصحیح خطا)

همینطور که قبلاً هم گفتیم USB یک گذرگاه تک مدیر است یعنی فقط یک Host می تواند به آن متصل شود.تمام انتقالات را Host آماده سازی می کند.اولین بسته همان بسته Token است که توسط Host ایجاد می شود تا یک سری اطلاعات کنترلی مانند خواندنی یا نوشتنی بودن عملیات، آدرس دستگاه مقصد و ... رد و بدل شود.بسته ی بعدی معمولاً یک بسته داده به منظور حمل بار مفید[5] است که بلافاصله پس از این بسته ها یک بسته Handshake فرستاده می شود تا وضعیت دستگاه مقابل را در قبال سوال دیگری به او برساند تا وی بتواند تصمیم درستی را به اجرا درآورد.

فیلدهای مشترک در بسته های USB

- **Sync** [6]

تمام بسته ها باید با یک فیلد Sync شروع شوند.طول این فیلد 8 بیت است که به منظور همزمان سازی Clock بین فرستنده و گیرنده استفاده می شود.دو بیت آخر شروع فیلدهای PID را نشان می دهند.

- **PID** [7]

فیلد PID برای تعیین هویت بسته هاست.این فیلد به گیرنده نشان می دهد که نام و هویت بسته ای که هم اکنون دریافت کرده چیست.مقادیر ممکن در این فیلد به صورت زیر است:

Group	PID Value	Packet Identifier
Token	0001	OUT Token
	1001	IN Token
	0101	SOF Token
	1101	SETUP Token
Data	0011	DATA0
	1011	DATA1
	0111	DATA2
	1111	MDATA
Handshake	0010	ACK Handshake
	1010	NAK Handshake
	1110	STALL Handshake
	0110	NYET (No Response Yet)
Special	1100	Preamble
	1100	ERR
	1000	Split
	0100	Ping

[5] Payload
[6] Synchro field
[7] Packet ID field

طول فیلد PID معمولاً 4 بیت است ولی به منظور اطمینان بیشتر در یک فیلد 8 بیتی به صورت دو 4 بیت پشت سر هم تکرار می شوند.به این صورت:

PID0	PID1	PID2	PID3	nPID0	nPID1	nPID2	nPID3

- **ADDR** [8]

فیلد آدرس به منظور تعیین مقصد بسته ایجاد می شود.طول این فیلد 7 بیت است و مقدار آدرس صفر به هیچ دستگاهی نسبت داده نمی شود بنابراین می توان حداکثر 127 دستگاه را آدرس دهی کرد البته دستگاههایی که به گذرگاه متصل هستند ولی هیچ آدرسی به آنها تعلق نگرفته است باید به بسته هایی که با آدرس صفر می رسند پاسخ دهند.

- **ENDP** [9]

طول این فیلد 4 بیت است بنابراین می توان 16 نقطه ی پایانی ایجاد کرد.دستگاههایی با سرعت Low می توانند فقط 2 تا حداکثر 4 نقطه پایانی داشته باشند.

- **CRC** [10]

عملیات CRC روی محتوای داده در بسته ی بار مفید انجام می شود.تمام بسته های Token زمانی که بسته داده آنها 16 بیتی است شامل 5 بیت CRC هستند.

- **EOP** [11]

این فیلد پایان بسته را تعیین می کند.در حقیقت یک SE0 [12] است یعنی تقریباً به مدت زمان دو بیت سیگنال صفر می شود.

انواع بسته های USB

پروتکل USB دارای چهار نوع بسته مختلف است.بسته Token به منظور تعیین نوع انتقال و بسته داده شامل بار مفید و بسته Handshake شامل اطلاعات مربوط به پاسخ و ACK همچنین گزارش خطاها و آخرین بسته یعنی بسته شروع فریم به منظور تعیین نقطه شروع فریم جدید است.

- **Token Packets**

سه نوع مختلف از بسته Token وجود دارد:
In: درخواست Host را مبنی بر خواندن اطلاعات از دستگاه، را به دستگاه می رساند.
Out: در خواست Host را مبنی بر فرستادن اطلاعات به دستگاه، را به دستگاه می رساند.
Setup: به منظور شروع نوع انتقال کنترل[13] استفاده می شود.

بسته های از نوع Token باید از قالب زیر تبعیت کنند:

Sync	PID	ADDR	ENDP	CRC5	EOP

- **Data Packets**

دو نوع از بسته داده وجود دارد که هر کدام قادر به انتقال 0 تا 1023 بایت داده هستند.
DATA0
DATA1
این بسته های داده بر طبق قالب زیر هستند:

Sync	PID	DATA	CRC16	EOP

[8] Address field
[9] Endpoint field
[10] Cyclic Redundancy Check
[11] End of Packet
[12] Single Ended Zero
[13] Control Transfer

- **Handshake Packets**

سه نوع از بسته های Handshake که با PID بسادگی معرفی می شوند وجود دارد:
ACK: به منظور گزارش و تصدیق دریافت صحیح بسته بکار می رود.
NAK: برای گزارش اینکه فعلاً درخواست شما قابل اجرا نیست یا گزارش ناتوانی در پاسخ بکار می رود.همچنین در نوع انتقال وقفه[14] به منظور فهماندن به Host که در حال حاضر هیچ داده ای برای فرستادن وجود ندارد.
STALL: زمانی که دستگاه در وضعیتی قرار دارد که نمی تواند برای خود تصمیم بگیرد با این فیلد به Host می گوید که به مداخله وی احتیاج دارد.
بسته های Handshake به قالب زیر هستند:

Sync	PID	EOP

- **Start of Frame Packets[15]**

بسته ی SOF دارای 11 بیت به منظور شماره فریم است که در هر 1mS ± 500nS توسط Host فرستاده می شود.

Sync	PID	Frame Number	CRC5	EOP

توابع USB[16]

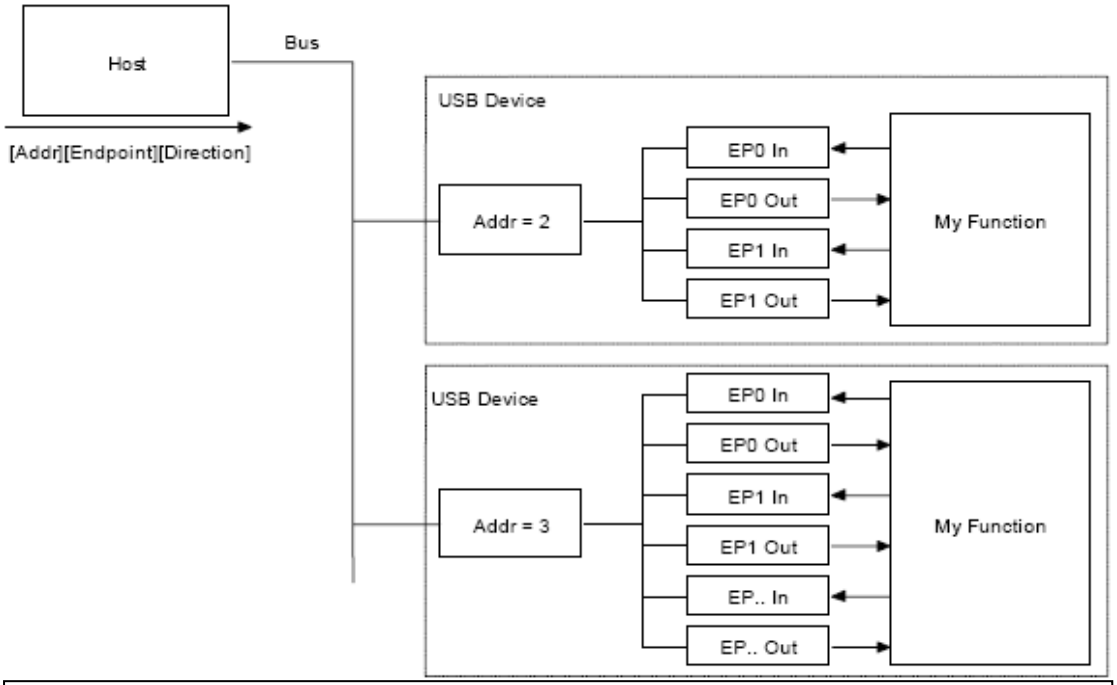
وقتی ما راجع به یک دستگاهی که با USB کار می کند فکر می کنیم برایمان فرقی نمی کند که آن دستگاه چه باشد فقط کافیست دارای یک ارتباط دو طرفه با پروتکل USB باشد ممکن است این دستگاه یک اسکنر، پرینتر و یا هر چیز دیگری باشد مهم توابع USB است که مطابق با نیازهای ارتباطی دستگاه اجرا می شوند.
بیشتر توابع یک سری بافر دارند که معمولاً طول آنها 8 بایت است که هر بافر به یک نقطه پایانی یا endpoint یعنی EP0 IN و EP0 OUT و ... وابسته خواهد بود. به عنوان مثال Host یک تقاضای واصف دستگاه[17] می فرستد.ابتدا دستگاه توسط سخت افزار تابع یک بسته Setup را می خواند و در آن فیلد ADDR را چک می کند اگر آدرس متعلق به او بود سپس بسته های بعدی را هم که رسید می خواند و بارمفید درون بسته داده را در یک بافر نقطه پایانی ذخیره می کند مشخصات این نقطه پایانی در فیلد نقطه پایانی از بسته Token قرار گرفته است.سپس با فرستادن یک بسته ACK دیافت آن را به فرستنده اعلام می کند پس از آن با ایجاد یک وقفه داخلی به روش خودش مثلاً در یک چیپ یا میکروکنترلر روتین های بعدی را از آماده بودن اطلاعات دریافت شده برای پردازش مطلع می کند.این کارها همه معمولاً در سخت افزار انجام می شود.حالا نرم افزار باید محتوای بافر نقطه پایانی را پارس کند تا تقاضای واصف آن را بدست آورده و عملکرد لازم را داشته باشد.

نقاط پایانی یا endpoints

نقاط پایانی منابع داده ها هستند. نقاط پایانی می توانند در یک تابع USB البته در انتهای کانال ارتباطی قرار گیرند.در لایه نرم افزاری مثل درایور می توان یک بسته مثل EP1 به دستگاه فرستاد.بنابراین داده ای در خروجی Host جریان می یابد در بافر EP1 OUT پایان می یابد.سپس firmware قادر است سر فرصت این داده را از بافر بخواند و به دقت پردازش کند.اگر او بخواهد داده ای را باز گرداند روند کار مانند خروج داده از Host ساده نیست.تا زمانی که Host تقاضای ارسال داده از سوی firmware یا دستگاه را نکرده است firmware داده مورد نظر را در بافر EP1 IN قرار می دهد به محض آمدن تقاضا و آماده بودن firmware داده درون بافر EP1 IN ارسال خواهد شد.

---

[14] Interrupt transaction
[15] SOF Packet
[16] USB Functions
[17] Device Descriptor



تصویر 7 : نحوه ارتباط توابع USB با نقاط پایانی

لوله ها[18]

زمانی که دستگاه از طریق نقاط پایانی داده ها را می فرستد و دریافت می کند، نرم افزار در کلاینت همین داده ها را از طریق لوله ها می فرستد و دریافت می کند.لوله ها ارتباط منطقی بین Host و نقاط پایانی هستند.لوله ها دارای مجموعه ای پارامتر هستند مانند: میزان پهنای باند اختصاص داده شده به آن، نوع انتقال USB (Control, Isochronous, Bulk or Interrupt) ، جهت انتقال داده ها و حداکثر سایز بسته ها و بافرها. به عنوان مثال لوله ی پیش فرض یک لوله ی دوطرفه[19] با یک نقطه پایانی صفر ورودی[20] یا EP0 IN و نقطه پایانی صفر خروجی [21] یا EP0 OUT به همراه یک انتقال USB از نوع کنترل ساخته شده است.

واسط USB دو نوع لوله دارد:

- **Stream Pipes**

قالب از پیش تعیین شده ای ندارد.این نوع لوله ها فقط در انواع انتقال Bulk, Isochronous, Interrupt تعریف شده اند و می توانند از طریق Host یا دستگاه کنترل شوند.

- **Message Pipes**

این نوع لوله ها قالب تعریف شده ای دارند.آنها از سمت Host کنترل می شوند و با یک تقاضای فرستاده شده از host آماده سازی می شوند.سپس داده در جهت خواسته شده در این تقاضا منتقل می شود.این لوله ها به داده ها اجازه انتقال در هر دو جهت را می دهند ولی فقط نوع انتقال کنترل را پشتیبانی می کنند.

[18] Pipes
[19] Bi - Directional
[20] endpoint zero in
[21] endpoint zero out

انواع نقاط پایانی یا انواع انتقال در USB

گذرگاه USB دارای 4 نوع انتقال یا نقاط پایانی است:

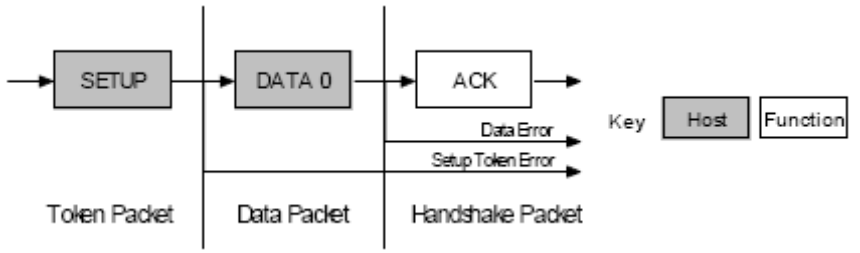
- Control Transfer
- Interrupt Transfer
- Isochronous Transfer
- Bulk Transfer

نوع انتقال کنترل یا Control Transfer

نوع انتقال کنترل معمولاً برای فرستادن فرامین تغییر وضعیت استفاده می شود.طول بسته ها در نوع سرعت Low باید 8 بایت باشد و سرعت High سایز های 8 ، 16 ، 32 و 64 بایت را هم پشتیبانی می کند همچنین در دستگاههایی با سرعت Full طول بسته ها فقط 64 بایت است.نوع انتقال کنترل دارای سه طبقه است.

- **Setup Stage**

در این طبقه تقاضا فرستاده می شود که شامل سه بسته است.ابتدا بسته ی Token که حاوی فیلدهای آدرس و شماره ی نقطه پایانی است فرستاده می شود.پس از آن بسته داده که همیشه دارای PID از نوع DATA0 است و همچنین شامل یک بسته Setup از جزئیات مربوط به نوع تقاضا است فرستاده می شود.در ادامه بسته ی Setup را توضیح خواهیم داد.بسته ی آخر یا سوم بسته ی Handshake به منظور تصدیق صحت دریافت یا گزارش یک خطا فرستاده می شود.



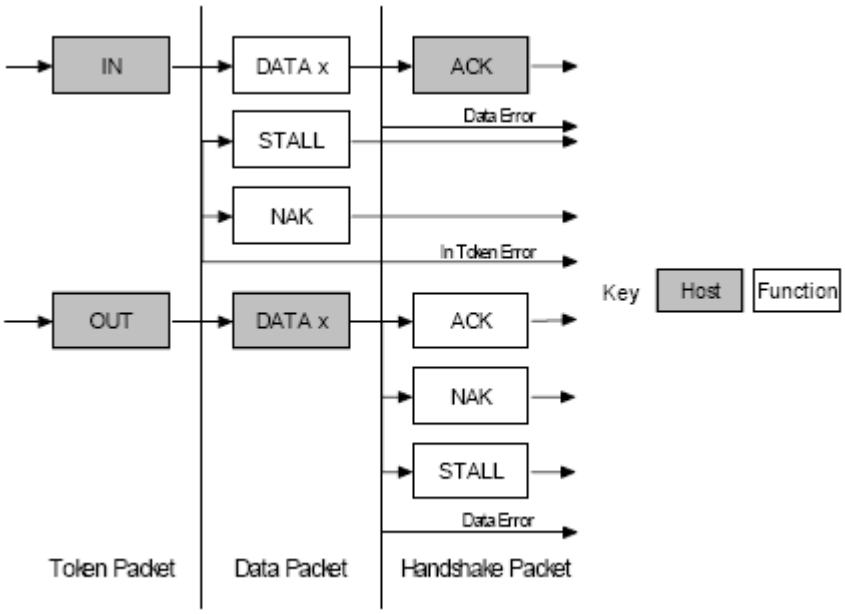
تصویر 8 : روند فرستادن سه بسته ی مربوط به طبقه ی Setup .

- **Data Stage (Optional)**

این طبقه شامل یک یا چند انتقال IN یا OUT است.تقاضای Setup میزان داده ای که در این طبقه باید منتقل شود مشخص می کند.اگر این میزان از مقدار ماکسیمم مجاز تجاوز کرد داده ها باید طی چند مرحله منتقل شوند و هرکدام از میزان داده در آخرین بسته تبعیت می کنند.طبقه ی داده بر طبق جهت انتقال داده ها دارای دو سناریوی مختلف است.

IN: زمانی که Host برای دریافت داده کنترلی آماده است یک IN Token منتشر می کند.اگر تابع این IN Token را با خطا دریافت کرد معلوم می شود که PID با بیتهای معکوس PID برابر نیستند سپس این بسته Ignore می شود.اگر Token صحیح دریافت شود دستگاه می تواند با فرستادن یک بسته داده شامل داده کنترلی یا بسته stall شامل نقطه پایانی که در آن نوع خطا اعلام شده است یا یک بسته NAK به منظور مطلع کردن Host از اینکه داده ای برای فرستادن موجود نیست پاسخ دهد.

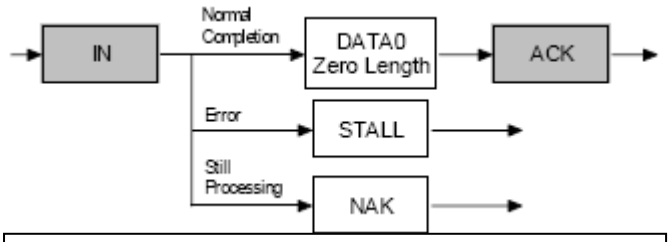
OUT: زمانی که Host می خواهد به سمت دستگاه یک بسته داده کنترلی بفرستد باید ابتدا یک بسته OUT Token و پشت سر آن یک بسته داده که بار مفید آن همان داده کنترلی است بفرستد.اگر هر کدام از محتویات این دو بسته در گیرنده خطا داشت دستگاه آنها را Ignore می کند و نادیده می گیرد اگر بافر نقطه پایانی خالی بود داده دریافت شده در آن قرار می کیرد سپس دستگاه با ارسال بسته ACK به سمت Host صحت عملیات را اعلام می کند.اگر بافر نقطه پایانی خالی نبود و دستگاه مشغول پردازش بسته های قبلی بود آماده نبودن خود را با فرستادن یک بسته NAK به Host اعلام می کند.اگر نقطه پایانی شامل خطا بود تابع باید این خطا را با فرستادن بسته STALL به Host اعلام کند.



تصویر 9 : نحوه فرستادن بسته ها در طبقه ی داده

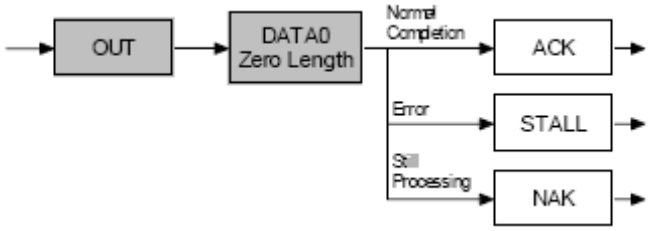
- **Status Stage**

در این طبقه وضعیت تمام تقاضاهای موجود تعیین می شود که البته بر مبنای جهت انتقال متفاوت است.گزارش وضعیت همیشه توسط تابع اعلام می شود.



تصویر 10 : روند فرستادن بسته ها در طبقه وضعیت نوع IN .

IN: اگر Host درحین دریافت داده در طبقه داده یک یا چند IN Token فرستاد باید صحت دریافت آن داده ها را با ACK به دستگاه اعلام کند این کار با فرستادن یک OUT Token به همراه یک بسته داده به طول صفر انجام می شود.اکنون تابع می تواند وضعیت خود را توسط طبقه Handshake گزارش دهد.یک ACK نشان می دهد که حالا تابع اجرای فرمان کنونی را تمام کرده و آماده پذیرفتن فرمان بعدی است.اگر در طی اجرای این فرمان خطایی رخ دهد تابع باید یک STALL منتشر کند.اگر چه تابع هنوز در حال اجرا است ولی با فرستادن NAK به Host می گوید که طبقه وضعیت را در یک وقت دیگر تکرار کند.



تصویر 11 : روند فرستادن بسته ها در طبقه وضعیت نوع OUT .

OUT: اگر Host یک یا چند OUT Token در حین انتقال داده در طبقه داده فرستاد تابع صحت دریافت داده را در پاسخ به یک IN Token با یک بسته با طول صفر اعلام می کند.اگر خطایی رخ داد باید یک STALL بفرستد و اگر همچنان در حال پردازش داده بود با یک NAK به Host می گوید که فاز وضعیت را وقت دیگری تکرار کند.

ذهنیتی بزرگتر از نوع انتقال کنترل

در اینجا برای روشن شدن مطلب مثالی در مورد درخواست یک واصف از دستگاه توسط Host در حین کار، وجود دارد که توجهتان را به آن جلب می کنیم.
Host یک Setup Token می فرستد تا به تابع بگوید که بسته ای که در ادامه می فرستد یک بسته Setup است.فیلد آدرس شامل آدرس دستگاهی است که Host از آن تقاضای واصف کرده است.شماره ی نقطه پایانی صفر است که استفاده از لوله ی پیش فرض را نشان می دهد.سپس Host بسته DATA0 را می فرستد.این بسته شامل بار مفید 8 بایتی که نوع واصف درخواستی است می باشد.پس از آن تابع USB صحت دریافت بسته Setup را اعلام می کند.اگر این بسته با خطا دریافت شد دستگاه آن را نادیده می گیرد سپس Host پس از تاخیری کوچک دوباره آن را می فرستد.

1. Setup Token

Address & Endpoint Number

Sync	PID	ADDR	ENDP	CRC5	EOP

2. DATA0 Packet

Device Descriptor Request

Sync	PID	DATA0	CRC16	EOP

3. ACK Handshake

Device Ack. Setup Packet

Sync	PID	EOP

سه بسته فوق اولین انتقال USB را تشکیل می دهند.دستگاه پس از دریافت آنها باید 8 بایت داده را کدگشایی کند و تعیین کند که آن تقاضای چه نوع واصفی است.پس از تشخیص نوع واصف تقاضا شده باید آن را به سمت Host بفرستد که این مرحله بعدی از انتقال USB است.

1. IN Token

Address & Endpoint Number

Sync	PID	ADDR	ENDP	CRC5	EOP

2. DATA1 Packet

First 8 Bytes of Device Descriptor

Sync	PID	DATA0	CRC16	EOP

3. ACK Handshake

Host Acknowledges Packet

Sync	PID	EOP

1. IN Token

Address & Endpoint Number

Sync	PID	ADDR	ENDP	CRC5	EOP

2. DATA0 Packet

Second 8 Bytes of Device Descriptor

Sync	PID	DATA1	CRC16	EOP

3. ACK Handshake

Host Acknowledges Packet

Sync	PID	EOP

1. **IN Token**

Address & Endpoint Number

Sync	PID	ADDR	ENDP	CRC5	EOP

2. **DATA1/0 Packet**

Last 8 Bytes of Device Descriptor

Sync	PID	DATA0/1	CRC16	EOP

3. **ACK Handshake**

Host Acknowledges Packet

Sync	PID	EOP

در اینجا لازم به ذکر است که حداکثر سایز بار مفید 8 بایت است.Host با فرستادن یک IN Token به دستگاه می گوید که حالا می تواند داده را برای این نقطه پایانی بفرستد.همینطور که ماکسیمم سایز بسته 8 بایت است پس ما باید یک واصف 12 بایتی را به قطعاتی تقسیم کنیم و آن قطعات را پشت سر هم بفرستیم.هر قطعه به جز در آخرین انتقال باید 8 بایت باشد.Host برای هر بسته داده ACK می فرستد.
زمانی که واصف دستگاه فرستاده شد یک انتقال وضعیت در پی خواهد داشت.اگر انتقال موفقیت آمیز باشد Host یک بسته به طول صفر می فرستد تا بگوید تمام انتقالات موفقیت آمیز بوده است.سپس تابع به این بسته با فرستادن وضعیتش پاسخ می دهد.

1. **OUT Token**

Address & Endpoint Number

Sync	PID	ADDR	ENDP	CRC5	EOP

2. **DATA0 Packet**

Zero Length Packet

Sync	PID	DATA0	CRC16	EOP

3. **ACK Handshake**

Function Ack. Entire Transactions

Sync	PID	EOP

بسته Setup

از این بسته استاندارد به منظور شناسایی و تنظیم دستگاه پس از روشن شدن استفاده می شود.این بسته از نوع تقاضای استاندارد استفاده می کند به همین خاطر در فیلد bmRequestType بیتهای D6 و D5 برابر با صفر است.تمام فیلدهای زیر مانند bRequest و wValue و wIndex و wLength در خصوصیات پروتکل USB بررسی می شوند.هر بسته Setup هشت بایت دارد که مشخصات آن را در جدول زیر مشاهده می کنید.

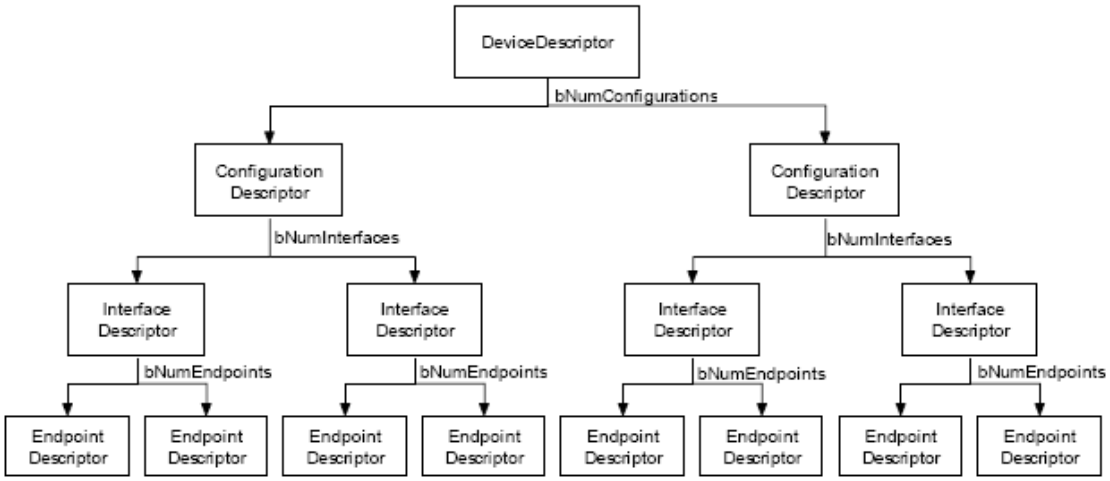
Offset	Field	Size	Value	Description
0	bmRequestType	1	Bit-Map	**D7 Data Phase Transfer Direction** 0 = Host to Device 1 = Device to Host **D6..5 Type** 0 = Standard 1 = Class 2 = Vendor 3 = Reserved **D4..0 Recipient** 0 = Device 1 = Interface 2 = Endpoint 3 = Other 4..31 = Reserved
1	bRequest	1	Value	Request
2	wValue	2	Value	Value
4	wIndex	2	Index or Offset	Index
6	Wlength	2	Count	Number of bytes to transfer if there is a data phase

انواع واصف ها در USB

تمام دستگاههای USB دارای یک مجموعه واصفهایی هستند که از طریق آنها اطلاعات درونی خود را به Host اعلام می کنند تا Host بتواند خود را با آنها هماهنگ کند.این اطلاعات از این قبیل هستند:نام دستگاه، نام سازنده دستگاه، نوع انتقال USB که دستگاه پشتیبانی می کند، با چه راههایی می تواند تنظیم شود، تعداد نقاط پایانی آن چقدر است و چه انواعی دارد.
انواع واصفهایی که بیشتر استفاده می شوند به صورت زیر است:

- واصفهای دستگاه یا Device Descriptor
- واصفهای تنظیمات یا Configuration Descriptor
- واصفهای واسطه یا Interface Descriptor
- واصفهای نقطه پایانی یا Endpoint Descriptor
- واصفهای رشته یا String Descriptor

تصویر 12 : سلسله مراتب مربوط به انواع واصفها و چگونگی ارتباط آنها



قابل توجه صنعتگران، دانشجویان و گروه های رباتیک، مدیران پروژه های مکاترونیک و ...

اگر متقاضی دریافت مشاوره و هر گونه پشتیبانی علمی در رابطه با چگونگی یک ارتباط راحت با رایانه از طریق پایانه USB هستید و قصد دارید دستگاه خود را از طریق پایانه پر سرعت USB، توسط رایانه کنترل کنید می توانید با پست الکترونیک Sefidgaran@Gmail.com مکاتبه فرمایید تا پس از بررسی سریعاً پاسخ خود را دریافت کنید.

منابع

USB Complete: Everything You Need to Develop USB Peripherals, Third Edition
by Jan Axelson

گردآورنده

فرشید سفیدگران
کارشناسی کامپیوتر سخت افزار
مرداد ۱۳۸۵

Copyright © 2006, Farshid Sefidgaran (Sefidgaran@Gmail.com)
First Release 20 August 2006