Subject:
Year: Month: Date: ۱

طراحی مدارهای واسط ۲

* ارزیابی

- فعالیت کلاسی و تمارین ٪ 20 - 25
- امتحان میان ترم ٪ 30
- امتحان پایان ترم ٪ نمره

Take Home Design ←

Subject:
Year. Month. Date. ( )

* تعریف مدار واسط (I/F ≡ Interface)

A H.W and/or S.W which allows a microcomputer system to communicate with outside world.

- سخت افزار TTL Gate Interface ☑
- نرم افزار GUI ☑
- سخت افزار + نرم افزار

* انواع دسته بندی ها : سخت افزار یا سخت افزار

I.F
- S.W
- H.W
  - Analog → Sensors, Sound Card.
  - Digital

Analog I.F : به سیستم کامپیوتر اجازه می دهد سیگنال آنالوگ ـ مثل داده یا از آن خارج شود. Microphone ، Headphone ☑

* انواع دسته بندی : جهت ارتباط

I.F
- Unidirectional → Mouse, Keyboard
- Bidirectional → Serial port, USB
- Multidirectional → Bus,

* سطح اینترفیس (Level)

I. Application

نرم افزارهایی که یک برنامه کاربری ارتباط دارد مانند GUI

II. Software

مانند API و Device Driver و (برنامه‌هایی در داخل OS که حلقه ارتباطی بین سخت افزار و کیس OS می باشد)

III. logical

IV. Physical (Board / Chip)

ارتباط کارت با کارت (Board) یا چیپ با چیپ (Chip)

V. Electrical

• تعداد I,F های فیزیکی محدود است هنگامی که به کاربر تعداد بیشتر I,F نشان داده شود آن I,F ها منطقی خواهند بود.

☑ به عنوان مثال با استفاده از Modulation

* مبدل Interface

I. Ports

☑ پورت‌های سریال ، موازی ، General Purpose I/O ، ...

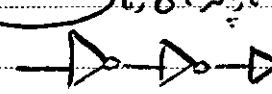
II. Drivers (hardware drivers)

☑ Line Driver ← LS 244

Motor Drivers ← فرمان‌های دیجیتال را به فرمان‌های آنالوگ (مثلاً on/off) تبدیل می‌کند.

III. Buffer

☑ برای اصلاح پارامترهای زمانی



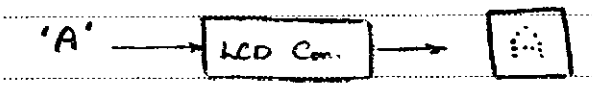
Subject:
Year. Month. Date. ( )

IV. Bridges

PCI Bridges ☑

V. Controllers

☑ LCD یک دستگاه آنالوگ است که صفحه‌ای از کریستال مایع است که پوشانده و ترسط اطلاعات آنالوگ پیکسل‌های آن روشن شده دستگاهی که فرمان دیجیتال را به آنالوگ تبدیل می‌کند LCD Controller نام دارد.



(MAC) Ethernet Controller ، DMA Controller ، CRT Controller

VI. Adapters

☑ عمدتاً برای تطبیق سطح ولتاژ

VII. Communication I/F

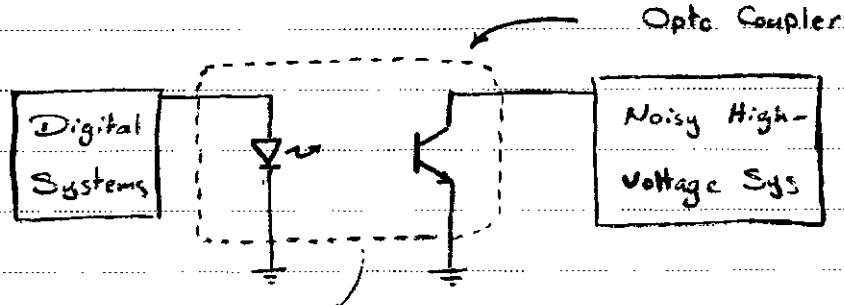
☑ سیستم‌های آنالوگ ← مکانیزم handshaking را پیاده می‌کنند.

VIII. Buses

AMBA AHB ☑

IX. Couplers

Opto Couplers ☑



Subject:
Year Month Date 3

X > Sensors

☑ واسط دنیای خارج با سیستم مورد نظر مثلاً سنسور حرارت و فشار ...

XI > Timer / Counter / Interrupt

☑ اساس برخواری ارتباط زمانی را برقرار می‌کند ← اندازه‌گیری کلاک CPU
وسایل کارت خوان

XII > Firmware

☑ حلقه ارتباطی بین سخت افزار و نرم افزار مثلاً Device Driver

* Interface Standard
جنبه‌های فیزیکی، سخت افزاری و نرم افزاری یک I.F را به دقت تعریف می‌کند
جامع و کامل و شامل نکات ریز پیاده سازی

* Interface Protocol
نحوه ارتباطات بین اجزاء در سطحی بالاتر از استاندارد می‌پردازد.

✎ 10 اینترفیس معمول را نام برده، مشخص کنید در کدام دسته قرار می‌گیرد (در صورت نیاز دسته اضافه کنید.)

Subject:
Year. Month. Date. ( )

BUS

* تعریف Bus

Serves as a shared communication link between the subsystems.

- comm. link ← دوطرفه
- Shared ←
- sub sys. ← subsystem یا ماژول مثل سیستم اصلی کامپیوتر ولی component نیست

* مزایای Bus

I> Low Cost

کم هزینه تر ← USB ، میزهای متصل به Board مادربرد می باشد

II> Versatile

دستگاههای جدیدی می توان به آن اضافه یا از آن حذف نمود.

* معایب Bus

I> Common Bandwidth

چون Bus اختصاص به یک device ندارد ← همه ی دستگاهها باید منتظر بمانند

Bus ← مرور زمانی برای چند برابر شدن

II> Security

همه ی دستگاهها به Bus می بینند همه اطلاعاتی رد و بدل می شود.

* پارامترهای مؤثر در طراحی Bus

I> Length

- Chip ← μm
- Board ← cm
- PC ← m
- طول ↑ ⋈ نرخ ارسال اطلاعات ↓
- هزینه Bus در این عامل اصلی است

Subject:

Year. Month. Date. (4)

II > Width

✓ w : 32 bit ← ℓ : 5 cm ← 100 Mega Byte/sec ☑

✗ w : 32 bit ← ℓ : 50 cm ← 100 Mega Byte/sec

بیشتر نویز cross talk

✓ w : 8 bit ← ℓ : 50 cm ← 100 Mega Byte/sec

با بالا فرکانس کلاک می‌توان آنرا جبران کرد.

8 bit × 100 MHz = 100 MBps

32 bit × 25 MHz = 100 MBps

1 bit × 800 MHz = 100 MBps

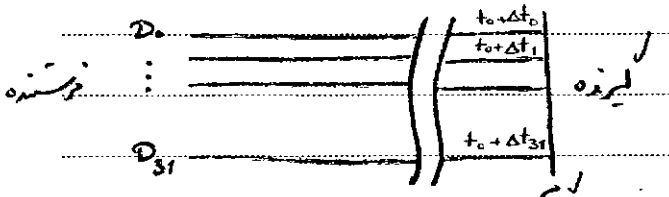
Length ↑ ⋈ Width ↓

می‌توان با بالا بردن کلاک تا حدی آنرا جبران کرد ← پیشرفت تکنولوژی

- علل این trade-off

Cross talk Noise *

Skew *



$Skew = \Delta t_{latest} - \Delta t_{earliest}$

۱- مسئله data validity برای مدتی بیشتر

۲- باندویدث طول اختلاف بین Δt با بالا می‌رود.

III > Number of Devices

- تعداد دستگاه‌ها روی طراحی مؤثر است.

IV > Arbitration

(نوبت دهی)

- مکانیسم جلوگیری یا برخورد با Collision

Subject:
Year. Month. Date. ( )

- پیکاربندی مختلف

device ← بدون درنظرگرفتن وضعیت ارسالی بودن آن ارسال کند ← برخورد
device ← ابتدا وضعیت را چک کند ← عدم ارسال همزمان
device ← یک arbiter درخواست گرفتن Bus بدهد ← حجم محاسباتی بالا
device ← همه device ها درخواست بدهند ← هزینه باید هم متقل باشد

centralized arbiter ← AMBA ☑
arbiter نداشتن ← CSMAC

V> Speed (Frequency)

- هر Bus یک فیلتر مابین گذر را شبیه‌سازی می‌کند
- کابل مابین رسانه فیزیکی است ← Twisted Pair ، USB ، Coax ...
- هر کابل یک محدودیت فرکانسی خاص خود دارد

VI> Type of Devices

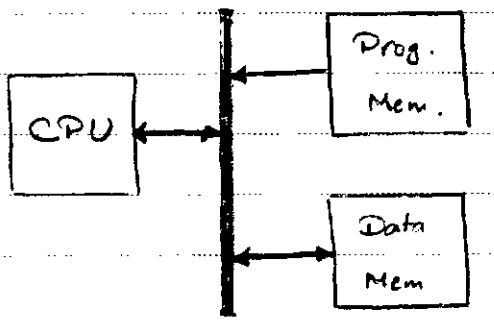
- پوشندگی

☑ دو کارت گرافیک و یک نمایشگر / دو نمایشگر و یک کارت گرافیک

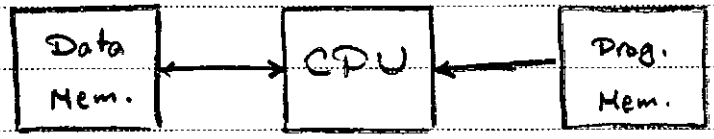
* دسته‌بندی Bus ها

I> CPU - Memory Bus
II> I/O Bus

طول : کوتاه
عرض : بالا
تعداد : کم
arbitration : غیر
سرعت : (فیلر) GB صد



طراحی بالا یک CPU-Mem Bus است که ابتدا به Prog Mem دسترسی دارد و سپس به Data Mem دسترسی دارد.



طراحی بالا یک CPU-Mem Bus است که CPU بدون pipeline برای مرحله fetch بدون دارا بودن data بخواند. در اینجا سربار بیشتری برای Bus و pin ها (دو برابر) برای استفاده از این امکان pipeline تحمیل می شود. پس باید ارزش اضافیش کاربرد را داشته باشد.

- پارامترهای CPU-Mem Bus
  - تعداد محدود
  - طول کوتاه (روی یک Motherboard)
  - پهنای باند زیاد
    - سرعت بالا
  - عرض زیاد
  - توپولوژی ساده

- خصوصیت I/O Bus
  - تعداد دستگاه‌های بیشتر
  - طول معمولاً بلندتر (10 cm ~ 10 m)
  - معمولاً عرض کمتر (≤ 16 bit)
  - توپولوژی وابسته به تعداد دستگاه‌ها
  - ☑ IDE ، USB

Subject :

Year . Month . Date . ( )

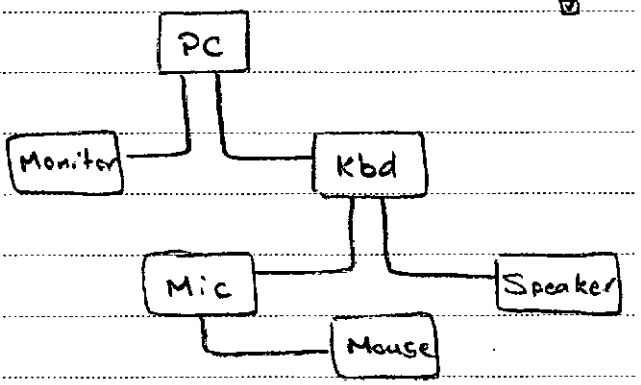
* برای اتصال دو host باید یک bus داشته باشیم.
  - یک bus کوتاه باید و نقش device را بازی کند.
  - برای اتصال بیش از دو device استفاده می‌شود.

* Topology
توپولوژی این استاندارد یک Tiered Star است یعنی نقاط اتصال پخش شده (Distributes Connectivity Points) دارد.

* 127 Logical Connection
شاید تعداد پورت‌های یک یا چند باشد و بقیه device‌ها بین این نقاط اتصال متصل باشند.

* Configuration
  - Dynamic Insertion / Removal
  - Automatic Configuration On Change

با جدا شدن یا اتصال هر دستگاه طرف دستگاه‌های متصل شده شبکه خود را دوباره پیکربندی (re-configure) می‌کند.



* Physical layer
  - 2-wire Differential Signalling for Data → مناسب برای خط زیری
  - NRZI Coded with Bit-Stuffing
  - 4-pin Connector, 4-wire Cable

Subject:
Year. Month. Date. ( )

- Shielded Connector & Cable, Shield has to be grounded

در استاندارد گفته شده است که این حفاظ دیتا فلزی باید GND باشد، علاوه بر آن روی Mother board، 5 سیم وجود دارد که یکی از آنها باید GND باشد برای حفاظ تأمین کند.

- Supply Voltage of +5V

☑ ضمانت تأمین 5 ولت برعهده دستگاهی است که از پشت آن کرنته می‌شود مثلاً اگر در دوربین دیجیتالی در پرت وجود داشته باشد یکی خروجی کامپیوتر است ولی برای ذخیره، دوربین باید 5 ولت را تأمین کنیم.

- Up to 5 meter Physical Connection

با افزایش طول و یا بازکردن حفاظ سرعت دستگاه پایین می‌آید.

↳ اتا خارجی استفاده نمی‌شود ← تا حل مشکل در الکترومغناطیس

↳ تعداد خرابی ها بالا می‌رود و مجبور به ارسال مجدد می‌شود.

حفاظ، نویز نویز را به خط تبدیل می‌کند.

محدودیت طول از دو عامل ناشی می‌شود: I) نویز II) تضعیف (افت ولتاژ)

Speed / Bandwidth *

- Early Versions : 1.5 Mbps ⟹ 12 Mbps (USB 1)
- Later Versions : 480 Mbps (USB 2)

Transaction Protocal *

- Host based

فقط host می‌تواند ترانزکشن را آغاز کند.

- Token Polling

host به ازای هر token یک دستگاه می‌فرستد و از آن می‌پرسد که آیا چیزی برای گفتن دارد یا خیر؟ اگر داشت می‌گوید وگرنه token به دیگری انتقال می‌یابد. چون مدل پ

Subject:
Year. Month. Date. (4

بس است ، token به این معنی است که در هر لحظه تنها می‌تواند یکی از دیود تنها
« host » می‌تواند به دیگران token بفرستد. هر device دو پورت دارد
و آدرس token مربوط به آن بندر آنها روی پورت دیگر خود کپی می‌کند.

- Host handles most of the protocol complexity.
- Device design is simple & low cost.

Robustness *

- Differential Signaling
- Shielded
- Handshake to Acknowledge Data Transfer
- Very Low Raw Physical Bit Error

این میزان دوچکتر از 10^{-12} است به شرطی که لایه فیزیکی به خوبی پیاده سازی
شود (طول 5 متر ، NRZI خوب ، Bit Stuffing ، کابل ، ...)
raw : تعداد بیتهای ارسالی (شامل هر نوع بیت داده و کنترلی)

- CRC Protection for Error Detection

با وجود این نرخ خطای کم ، تنها احتمال است که خطا را پیدا کرد و صرف نماید
که آنها با تحمل هزینه بیتهای بیشتر (کریتیک و ...) اصلاح کرد بلکه از راه پایین
استفاده می‌شود. در پروتکل این CRC 16 بیتی در نظر گرفته می‌شود.

- Retry Option

Bounded Transfer Charasteristics *

- Data Transfer Bandwidth & Latency Pre-negotiated

با توجه به ماهیت dynamic ارتباط ، با هر تغییر در topology دوباره با هم
negotiate می‌کنند.

Subject:

Year. Month. Date. ( )

Transaction *

در پروتکل USB هر Transaction (انتقال اطلاعات) تشکیل می‌شود از سه بسته‌ی زیر که عبارتند از:

I. Token

II. Data

III. Handshake

- بسته Token نوع داده، آدرس و مشخصات انتقال و ... را مشخص می‌کند

وقتی host در ابتدای هر Transaction است بسته Token را می‌فرستد، انواع توکن را در نظر دارید:

1. IN ← بسته‌ای را می‌خواند
2. OUT ← بسته‌ای را می‌نویسد
3. SETUP ← برای configure کردن یک device
4. SOF (Start of Frame) ← برای سنکرون کردن (مربوط به Bit Stuff) broadcast

هر Token Packet که می‌خواهد فرستاده شود:

1. احتیاج به فیلدی به نام Sync. دارد ← یک الگوی ثابت است که در ماشین حالات

2. PID ← Packet ID
   - نوع بسته را مشخص می‌کند
   - 8 بیت ← $2^8 = 256$ حالت ← انتقال بسته‌ی دیتا پروتکل
3. Address ← آدرس ← چون هر host می‌داند 127 آدرس را پشتیبانی می‌کند
   - 7 bit
4. End Point ← یک دستگاه ممکن است چندین استفاده کننده باشد که هر یک دریت متفاوت می‌شود این استفاده کننده‌ها End Point مشخص می‌شوند
   - آدرس: آدرس بسته / End P: دریافت کننده
   - 4 bit ← پرداخته حداکثر 15 استفاده کننده
   - دستگاه: Printer / Scanner ☑ Fax / Tel

P4PCO

Subject:

Year. Month. Date. (8)

5. CRC ← 5 bit ← تبلا گفته شد

6. EOP ← End of Packet

IN, OUT, SETUP Frame:

Sync. Field	PID	Address	End Point	CRC	EOP
	8 bit	7 bit	4 bit	5 bit	

SOF Frame:

Sync. Field	PID	Frame NO	CRC	EOP
		11 bit		

Frame No. ← شخص می کند که کدام Frame قرار است تبادل شود.

• لیست Data:

1. داده‌ای فرستاده می‌شود می‌تواند بین 0 تا 1023 بیت طول داشته باشد.

2. PID ← Data 0 ← شماره packet زوج است.

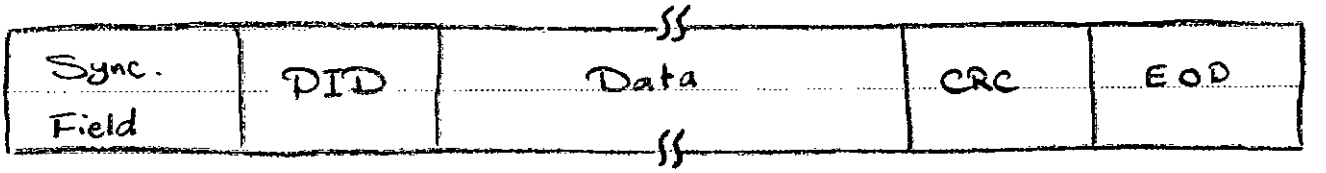
Data 1 ← شماره packet فرد است.

اگر بسته‌ای از دست برود می‌توان با CRC آنرا تشخیص داد. اگر بسته‌ای گم شود، گیرنده متوجه نمی‌شود. تک این نوع PID حالتهای 00 و 11 نشانه از دست رفتن یک بسته است.

اگر بسته‌ای با احتمال P از دست برود، احتمال از دست رفتن دو بسته P^2 است. بفرض $P = 10^{-3}$ آنگاه احتمال از دست رفتن دو بسته متوالی 10^{-6} است که برای جبران آن سربار کمی تحمل می‌کنند.

Subject :

Year . Month . Date .



← 0 ~ 1023 Byte →

• بسته Hand Shake

PID .1 ← ACK ← دریافت شد

NAK ← دریافت نشد

STALL ← فرستاده وقتی که بسته را فرستاده منتظر برگشتن ACK یا NAK است. این را با فرستادن STALL مشخص می‌کند. STALL اجازه آغاز Transaction را به device می‌دهد. گیرنده بدون این بسته Transaction را آغاز می‌کند و ACK یا NAK را می‌فرستد

اگر در یک بازه زمانی وقتی host منتظر چیزی است دریافت نمی‌کند time-out می‌شود (به علت گم شدن ACK یا STALL یا NAK حالت بن بست رخ می‌دهد) و اطلاعات از آخرین ACK یا NAK دریافت شده ارسال می‌شود.

CRC ندارد.

• بسته PRE ← Enables hub for low speed communication

Subject:

Year Month Date 9

USB Transaction ★

- USB نیمه سریع است.
- شامل بسته‌های کوچک و سریع و یا بزرگ و کند.
- نوع آن در config مشخص می‌شود.

1. Isochronous → Real Time Applications
   - ☑ Audio, Telephony
   - Periodic, Bounded Latency, Guaranteed BW
   - Tokens
   - Time Out

   • سریعترین نوع و با اطلاعاتی که سرعت در آنها حیاتی است.

2. Interrupt → Non-Real Time Applications & Slow
   - ☑ Joystick, Mouse, keyboard
   - Non-periodic (Burst)
   - Asynchronous
   - Low Bandwidth

   • دستگاه‌هایی که بر از حجم کوچکی از اطلاعات را می‌فرستند و نوعاً کند است.

3. Bulk → Large Amount of Data, Usually Storage
   - ☑ Cool Disk, Printer, Scanner
   - Non-periodic
   - Asynchronous
   - High Bandwidth
   - No Guranteed Access IF Bus Busy → Starved

   • برای مواردی که حجم زیادی از اطلاعات جابجا می‌شود و سرعت بالایی هم طلب می‌کند.

Subject:
Year. Month. Date. ( )

4. Control → Configuration Message
- → Bursty
- → Low Bandwidth
- → High Pericrity , Host Initiated
- → Retries Supported

* در پروتکل USB ما باس داریم که بین همه دستگاه‌ها مشترک است. پس نحوه قرارداد اطلاعات روی BUS راه‌حل‌های گوناگونی را می‌طلبد. برای همین ۱ هیتی به نام USB Frame تعریف می‌شود که مسائل فوق را تا حدی حل می‌کند.

ساختار این فریم به صورت زیر است:

1 ms

SOF	Isochronous	Interrupt	Control	Bulk
Fix	variable	variable	variable > 10%	تعریف !!!

مثال می‌خواهیم داده‌ها را در این فریم قرار بچینیم. این قرارگیری باید به نحوی پیش برود که بهترین جواب را بدهد. پس باید به نحوی اولویت بندی کرد. سطوح اولویت، مثال انواع transaction ای موجود است.

- اولویت اول به Isochronus تخصیص دارد. تمام data ای این نوع در فریم چیده می‌شود و در فریم تجاوز نمی‌کند. چون در این صورت، دستگاه مربوطه اجازه نمی‌گیرد.

حداکثر زمان مورد نیاز برای ۱ داده صوتی با نرخ 64 bps، برابر 83 ms خواهد شد (با توجه به پهنای باند بیشینه برای USB 1.0)

- اولویت دوم به Interrupt تخصیص دارد.
- اولویت سوم مربوط به Control می‌باشد. 10 % پهنای باند برای این نوع رزرو شده است. در شرایطی بسیار نادر که داده‌ی Control بیش از 10 % باشد، امکان تخصیص پهنای باند بیشتر باس به آن امکان‌پذیر است.
- اولویت آخر مربوط به Bulk است. اینکه است در سرعت ارسال داده‌ی storage انجام زمان اینترایپتی کم زیاد می‌کند. برای همین پهنای باند نوع Bulk تضمین نشده است.

Subject:

Year Month Date 10

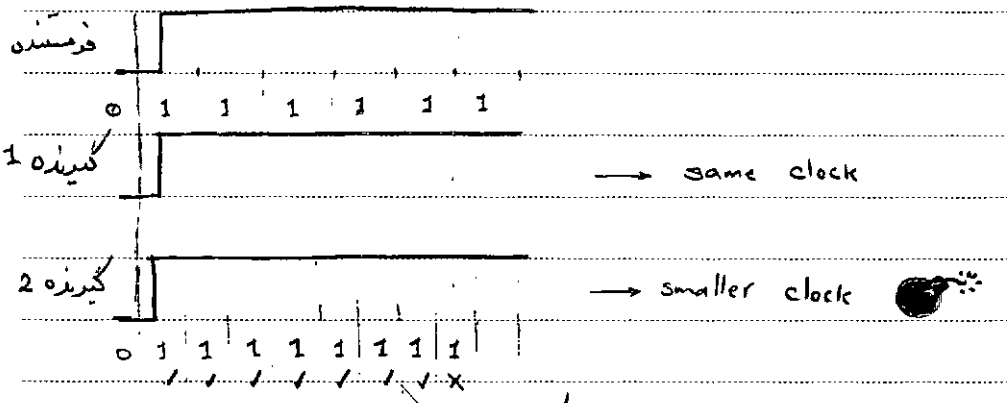
در این USB از Multiplexing نوع TDM استفاده می‌شود.

* جهت ارسال 0 و 1

از استاندارد NRZI استفاده می‌شود.

Toggle : 0

Don't Change: 1



برای حل این مشکل از Bit Staffing استفاده می‌کنیم.

- تا 6 تا "1" پشت سر هم مانعی ندارد.
- هفتمین "1" متوالی اجبار force می‌شود به جای آن "0" می‌فرستد.
- گیرنده نیز از این موضوع خبر دارد و با دیدن ششمین "1" بعدی داده را نمی‌خواند و منتظر toggle می‌شود. بدینگونه گیرنده و فرستنده سنکرون می‌شوند.
- پس همواره بجای 6 تا "1" بیت هفتم اضافه است آن را ignore می‌کنیم. چون در غیر این صورت تمایز بین bit stuffing و حالتهای input pattern مشکل می‌شود.

* USB Power Requirement

با دستگاههای خودتان کار داریم.

I. Low Power ← average over 1 sec intervals, peak = 100 mA, 0.5 mA

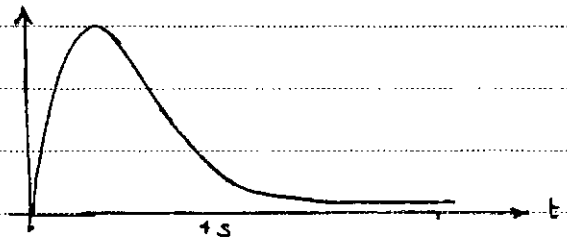
II. High Power ← average over 1 sec intervals, peak = 500 mA, 2.5 mA

خود device به میزانی خود می‌داند توان بین چه بازه‌ای است.

Subject:

Year. Month. Date. ( )

☒ دستگاهی در حالت اول 100mA را می‌کشد، پس از 999ms بعدی می‌تواند 400mA را بکشد.



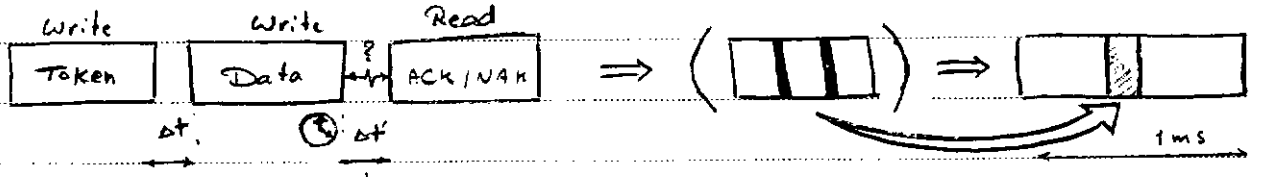
اگر این شرایط را داشت low power محسوب می‌شود.
حتی اگر جریان Peak آن از 100mA هم شد، low power است.

قبل از زمان configuration، حتی جریان پیک باید کمتر از حالت configured باشد، چون host نمی‌داند چقدر باید به آن تخصیص دهد. حتی ممکن است host نتواند آن را پیدا کند.
اگر توان بیش از حد High Power مورد نیاز باشد، باید دستگاه خودتان تغذیه مستقل داشته باشد.

Suspended State.

after 3ms of no bus activity
جریان در این حالت باید در حد device کم توان باشد.

* هر transaction یک نسبت ارسال یک قسمت دریافت است ☒ host می‌خواهد چیزی را بنویسد.
پس اول token می‌فرستد، بعد داده را می‌فرستد و بعد ack دریافت می‌کند. host در حال خریم باید ack را دریافت کند تا بفهمد که ack مربوط به کدام خریم است. برای همین پیام‌ها ترانزیشن را صورت می‌دهد (آسان کننده پروتکل) و برای پیاده‌سازی (time out ، state machine ، ...) استفاده می‌شود.



host ← ناصله زمانی مربوط به (max ای) device → ناصله زمانی مربوط به
Transaction
Frame

Subject:
Year Month Date

Device State Machine *

I. Attached ← دارد حسی جریان ورودی USB است به آشکار کشیدن جریان trigger می‌شود و باعث
- انتقال اطلاعات هدر device به host
- Hub تغییرات خود را به host اطلاع می‌دهد.
- وضعیت Unconfigured است و حداکثر جریان 100mA است

II. Powered ← Port Enabled
- host تغذیه مناسب را به دستگاه می‌دهد.

III. Default ← Reset می‌شود و سپس بسته‌ها را می‌پذیرد.

IV. Address Assigned ← آدرس اختصاصی به دستگاه داده می‌شود.

V. Configured ← آدرس به آن تخصیص یافته است

VI. Suspend ← اگر بیش از 3 ms غیرفعال باشد به طور خودکار وارد می‌شود.

Device Class *

برای اینکه device بداند درایور از کدام دسته‌ها باید استفاده کند category

I. Human Interface Class (HID)
- ☑ Keyboard, Mouse, Joystick
- @ Interrupt

II. Communication Device Class
- ☑ Audio Devices, Analog Modems, ...
- @ Isochronus

Subject:

Year. Month. Date. ( )

III. Printer Device Class

☑ Printer

© Bulk

IV. Mass Storage Device Class

☑ HDD, CD-ROM, Flash Disks

© Bulk

* اطلاعات ردو بدل شده Configuration سیستم

• Device Class

• Vender ID ← کارخانه سازنده + نوع محصول

Subject :
Year . Month . Date . ۱۲

◎ استاندارد : IEEE 1394

* کاربردی USB :

۱- Mouse / key board

۲- Audio

۳- Still Image

۴- ...

* برای برخی کاربردها باید پهنای باند بیشتری در نظر گرفت مثلاً Video . USB (حداقل نسخه 1 آن) جوابگوی آن نیست.

پهنای باند برحسب این استاندارد از حدود 50 Mbps تا 4 Gbps خواهد بود.

← نوع سیم انتقالش می‌باید.

← قابل استفاده برای LAN , دستگاه‌های بیرح که مثلاً خیلی کاربردی نیست

* Characteristics :

- Topology : Tiered Star
- پهنای USB
- Peer to Peer
- بدون نیاز به host
- Guaranteed Bandwidth for Real Time Data
- Up to 63 Devices per Bus

Subject:

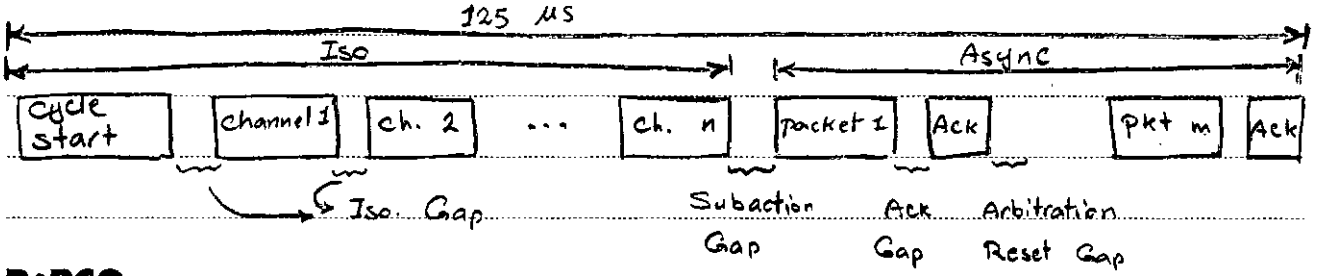
Year. Month. Date. ( )

Subject:

Year Month Date 13

Cycles in 1394

- معادل فریم در USB
- بازه زمانی که بتوان اطلاعات گنجانده شده است: حدود 125 μs
- در آغاز یک cycle ← root یک cycle start می‌فرستد
- مدتی سیستم صبر می‌کند ← Isochronous Gap
- بعد از آن اطلاعات Isochronous فرستاده می‌شود:
  - channel ≠ n ← اطلاعات خود را می‌فرستد
    - ← قبلاً درخواست کرده است
    - ← چون نوبت شده است پس ← یا کسی درخواست نداده بود
    - ← یا به root نزدیکتر بود
- سایر نودها به channel ≠ n گوش می‌دهند
  - ← n ≠ n' به غیر از یک استثناء
- بین هر دو channel فاصله‌ای وجود دارد.
- طول زمان مختص به هر channel برابر است.
- پس از پایان Isochronous، زمان را منتظر می‌ماند ← Subaction Gap
- حال نوبت به بسته‌های Asynchronous می‌رسد.
- پس از ارسال هر packet مدتی را صبر می‌کند ← Ack Gap
- بسته Acknowledge را دریافت می‌کند.
- مدت خاصی را برای ارسال بسته بعد منتظر می‌ماند ← Arbitration Reset Gap
- پس از پایان 125 μs اطلاعاتی دیگر فرستاده نمی‌شود و چرخه بعد آغاز می‌شود
  - ← ممکن است تمام مدت صرف Isochronous شود.
  - ← به دلیل ثابت نبودن زمان دریافت Ack زمان چرخه ممکن از 125 μs بیشتر شود.



Subject:

Year. Month. Date. ( )

* انواع Gap :

I> Iso :

- Idle Time
- Before the start of isochronous arbitration
- 0.04 ~ 0.05 μs ⟶ $\frac{1}{2500}$ interval

II> Subaction :

- Idle Time
- Between the isochronous & asynchronous arbitration
- max 10 μs

III> Ack :

- Idle Time
- Between end of packet and start of Ack.
- 0.04 ~ 0.05 μs

IV. Arbitration Reset :

- Idle Time
- Between fairness intervals
- max 20 μs

* یک بیت اطلاعاتی است که device ها وجود دارد ⟵ Arbitration Enable
• در آغاز دوره تنظیم این بیت 1 است ⟵ در هنگام هر بیت 1 می‌شود cycle
• هر گره تغییر می‌کند پس در این device در هر cycle ، این بیت را صفر می‌کند
• اگر هیچ device ای نشان داده نشد و همه‌ی گره‌ها ساکت بودند ، زمان را تا fairness interval
• صبر می‌کند تا root به آنها فرمان می‌دهد بیت خود را 1 کنند

Subject:
Year Month Date 14

• از برادام از این ببهایی که root وجود دارد، والد دستش دیجارجزایی شود، time out زمان به آن تخصیص داده می‌شود و بعد آن root صدا می‌شود و نوبت دی می‌شود.

• این نکته وجود داشته که اصلاً درخواست بعد نداده باشد و بازهم بخواهد، آنگاه دوره fairness interval تمام شده و اصلاً در نظر گرفته نمی‌شود.

1394.a Arbitration Enhancement ✱

1. Ack Acceleration Arbitration

A device acknowledge packet may append another outgoing packet.

2. Fly-by Arbitration

Nodes can append their own packet to packets recieved from children.

البته در صورتی که Ack وجود نداشته باشد می‌توان Request را در بازه Gap فرستاد که ممکن است بین آنها Collision پیش بیاید و باید سخت افزار Detect شود.

3. Token Arbitration

Groups of nodes "conspire" to use fly-by arbitration.

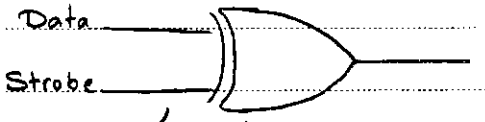
✱ در بازه Arbitration Enable Bit چون هر device می‌تواند چند channel داشته باشد و با هر Request این بیت را ۰ می‌کند و ممکن است چند درخواست Isochronous داشته باشد پس داریم:

bit { Per channel ← Isochronous .
Per device ← Asynchronous .

Subject :

Year . Month . Date . ( )

* با استفاده از ترکیب سیگنال‌های Data و Strobe می‌توان به سیگنال کلاک دست یافت.
برای همین استاندارد IEEE 1394 برای انتقال سریال اطلاعات است.

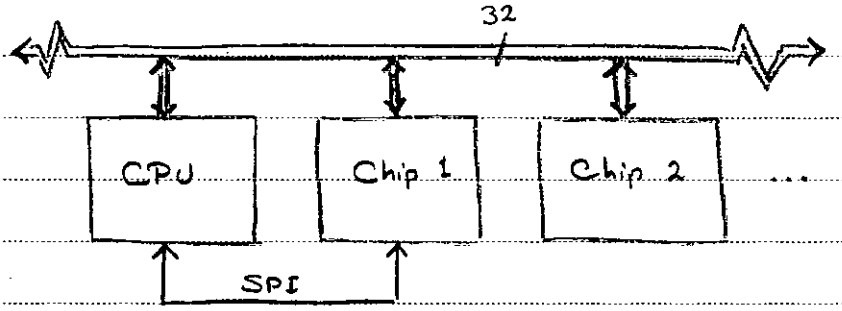


Strobe به تنهایی کافی نیست چون آغاز داده‌ها را مشخص می‌کند، ولی بیت به بیت را مشخص می‌کند.

Subject:
Year Month Date 15

② Serial Peripheral Interface (SPI)

• ایجاد ارتباط بین CPU و peripheral های درون دی به اجباراً برای استال داده



• نیاز به Debug برون سخت افزار منجر به ایجاد این باس می‌شود:
- نوشتن داده در حافظه
- device برون Configure

• این باس استاندارد نیست.
• اولین بار توسط شرکت Motorolla به وجود آمد
• بعدی توسط شرکت National Semiconductor با نام MicroWire عرضه شد

* Features

- Serial
- Synchronous
- low Cost → ساده پیاده سازی می‌شود Shift Register با یک
- ~1 Mbps
- For Debug & Configuration

* Connectivity

• CPU ← Master
• Peripheral ← Slave

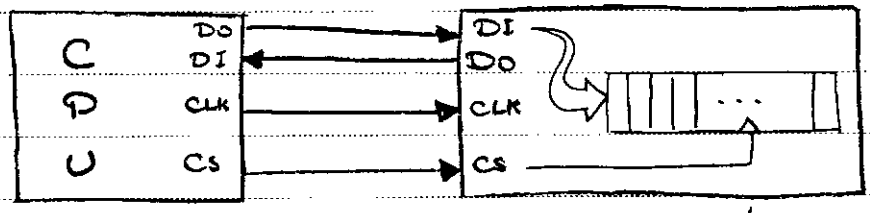
: استاندارد سیم ارتباطات

Subject :

Year . Month . Date . ( )

Clock : Master ⇒ Slave Synchronous

Chip Select : Master ⇒ Slave Enable



- داخل دستگاه جانبی یک Shift Reg وجود دارد.
- اطلاعات ورودی به دو دسته : Command و Data
- برای خواندن Status ، از بس SPI یک low level است آنها می خوانیم تا دستگاه را امتحان کنیم. با دستورات داخلی به peripheral (که CPU به آن می دهد) آنها Status Reg داخل Shift Reg می رود و به خارج منتقل می شود.
- شما می توانید خود این اصلی مثلاً ۲ خطه ، می توان از SPI استفاده کرد.

* این باس از نوع on-chip هستند.

* نسخه‌ای از AHB نام AHB Light ، خاصیت Multi Master حذف می‌شود و تنها یک Master حضور دارد. سیم بندی و طراحی Arbiter بسیار ساده می‌شود.

* سیگنال‌های AHB

• HCLK → کلاک سیستم که Master آن را تولید می‌کند.
در صورت حضور چند Master تنها یکی از آن‌ها به HCLK متصل است.
سنکرون شدن باس با کلاک است.

• HReset n → active low : n
منبع آن Reset Controller است که ماژولی مجزا است.

• HADDR → بایی 32 bit
MSB = 0 , LSB = 31 ⇐ استاندارد Intel
منبع آن Master است.

• HTRANS → نوع ترانزکشن
non sequential ① sequential ② idle ③ busy ④
دو بیت برای نشان دادن ۴ حالت

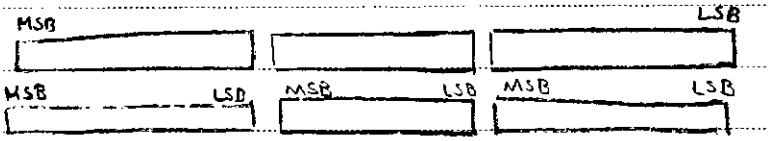
• HWRITE → جهت transfer
Master write : high Master Read : low

• HSIZE → (8, 16, 32 bit) عرض مجازی باس : 3bit
برای استفاده بهینه از زمان توسط device و arbiter داده است
و نشان over head بیشتر از آن استفاده می‌شود (گسترش عرض)

Subject:
Year. Month. Date. ( )

- HBURST → 3bit
  burst : تعدادی از اطلاعات که اطلاعات برای جلوگیری از سربار arbitration ایجاد می کند.

HSIZE : اندازه‌ی یک داده
HBURST : داده‌های ذخیره شده

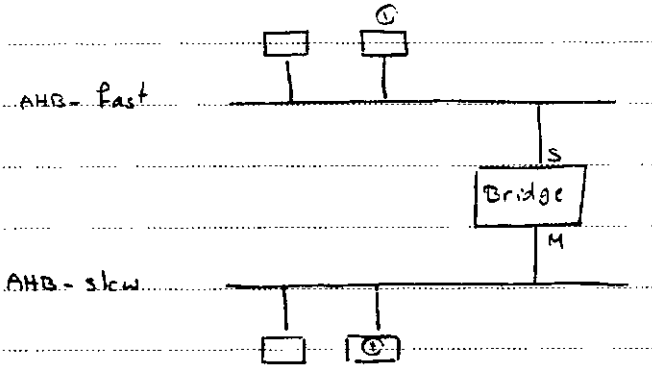


- HWDATA → Master خروجی 32bit

- HSELECT x → x : شماره‌ی اسلاو
  سیگنال انتخاب یک slave که توسط decoder تولید می شود.

- HRDATA → Slave خروجی 3bit

- HREADY → Slave اعلام آمادگی
  active high ☑



اگر ① Master و ② Slave باشد پیداست Bridge خواهد بود.
چون تمام پین پین روی آن نوشته شده است. پس Bridge Slave می تواند با این سیگنال پین بالا یا پایین داده. ضمن این سیگنال پین را idle می گذارد.

Subject:

Year. Month. Date. 17

• HRESP → Slave بیخ

SPLIT ⑤ RETRY ④ ERROR ① OKAY ⓪

در پاسخ Master یک HRESP + HREADY می‌تواند بفهمد که چرا slave کار را انجام نمیدهد.

• HBUSREQx → x : تعداد Master ها

درخواست یک Master درخواست می‌کند

• HLOCK x → x : به تعداد Master ها

این Master درخواست می‌کند که بدون تخصیص باید برای بدی از کنترل آن خارج نشود.

LOCK : high

• HGRANT x → x : به تعداد Master ها

به Master x می‌گوید که باس در اختیار است

• HMASTER → 4 bit : در Master های debug 16 تا

وقتی سیگنال اعطای باس به Master اشتباهی می‌رود (خطای سیستمی)

این ۴ بیت نشان می‌دهد که توسط Arbiter اعطای باس به کدام بود است.

• HMAST LOCK → بیخ Arbiter به درخواست LOCK (برای debug)

• HSPLIT x → بستگی به Slave ها و قابلیت Split Enable

به Arbiter می‌گوید که در وسط تراکنش است و می‌تواند Resume کند.

Master باید بداند که slave آن state را ذخیره کند تا بتواند عملیات را ادامه دهد.

Subject:
Year. Month. Date. ( )

AHB Transfer ✱

1. Master ⟵ request ⟶ Arbiter
2. Arbiter ⟵ bus grant ⟶ Master :
3. Master ⟵ Address & Control ⟶ : در لبه بالا رونده باید valid باشند
4. Slave ⟵ Status Data ⟶
5. Master ⟵ Data ⟶ Slave : write
6. Slave ⟵ Data ⟶ Master : Read → next address prepaired too.

Wait Cycles ✱

- ممکن است device ی وجود باشد که نتواند به سرعت مورد نظر پاسخ دهد.
- حتماً باید قبل از لبه بالارونده عمل خواندن، دیتا زمانی داده‌ی ورودی پایدار شده تا ثبت شوند، یا برای عمل خواندن، خروجی حتماً ی Master باید زمانی به پایدار تبدیل شود.

Read : Setup Time Master = Hold Time Slave

Setup Time → زمانی است که قبل از trigger داده باید ورودی stable شده باشد

Hold Time → زمانی که پس از trigger شدن، داده باید ورودی ثبت باقی بماند تا حالت داخلی FF پایدار شود

علت آن propagation delay است

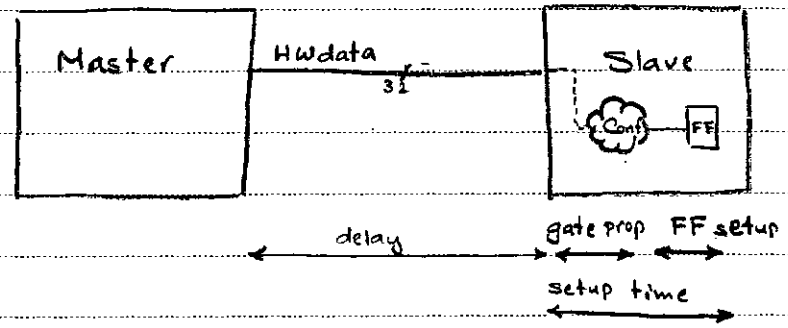
اگر قبل از زمان Hold داده تغییر کند، به حالت Meta Stable رفته و نمی‌شود تشخیص داد

- Master می‌خواهد بنویسد ← data A روی خط
- Slave حاضر شنید ← HReady را می‌اندازد
- Master تا لبه بالارونده بعدی نمی‌تواند data B روی خط ← که آن‌ها عوض می‌کند
  control C روی خط
- هیچ چیز متوقف می‌شود ← تا لبه‌ای که HReady را بالا ببیند
- Slave آماده می‌شود و HReady را بالا می‌برد ← هیچ چیز را می‌خواند.

- Master می‌خواهد بخواند ← Control را می‌فرستد.
  Slave حاضر نیست ← HReady را می‌اندازد.
  Master توقع دارد در این لبه بعد، داده را بخواند ← HReady برابر Low است ← نمی‌خواند!
  همه چیز متوقف می‌شود ← تا لبه‌ای که HReady را بالا ببیند.
  Slave حاضر می‌شود ← داده را می‌خواند
  Slave به اندازهٔ Setup Time صبر می‌کند ← HReady را بالا می‌برد.

چند wait cycle برای حاضر شدن داده Slave نیاز است؟

Wait_Time = n * Clock - Setup Time - Hold Time ⟹ n=?

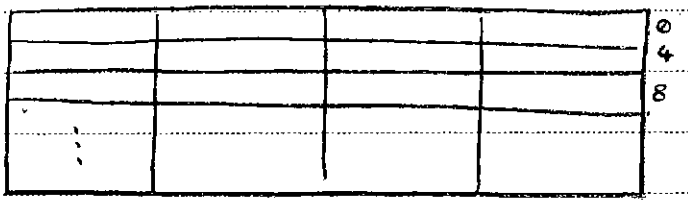


* Burst

- الگوریتم تراکنش آغاز شد، تبادل بدون کلاک زدن و برداشتن، تعدادی از اطلاعات را منتقل کند.
- کاربرد آن، در یک فضای نگاشت‌یافته است، در حالت‌های آن

Single, Inc, Wrap4, Inc4, ..., Wrap16, Inc16

INCR4, SIZE = Word



وابسته به سایز حافظه است

Subject:
Year: Month: Date: ( )

- اول فرآیند Burst آغاز می‌شود ← چون خودمان آدرس را می‌گذرد.
  در cycle بعد SEQUENTIAL می‌شود ← آدرس بعدی را با توجه به Increment می‌سازد.

Responses *

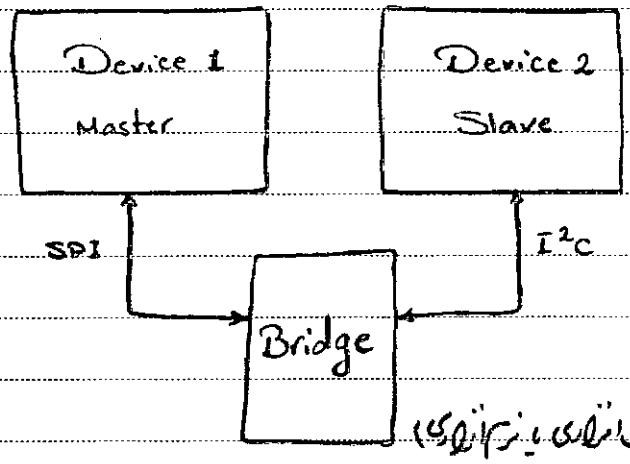
- Okay ← کامل
- Error ← خطا (همزمان با HReady)
- Retry ← مشکلی (glitch ، ...) پیش آمده و نیاز به ارسال مجدد داشت.
- Split ← ترانزکشن طولانی ← به Arbiter می‌گوید که من را آزاد کند، بعداً مراجعه کند.
- علت اینکه HReady در سطح Low است را توضیح می‌دهد.
- در مورد Spl ، Ret ، Err
  - حداقل دو سیکل
  - سیکل اول ← Bus : idle Hready : 0 Hresp : E/R/S
  - سیکل دوم ← Bus : non seq Hready : 1 Hresp : E/R/S
- هر Retry ، ۲ سیکل نیاز دارد.

Split Transaction *

- برای زمان‌بر یا دستگاه‌های کند، از مکانیزم wait استفاده می‌کند ← تا ۵ سیکل
  - این را تحمل می‌کند.
  - کارایی افت می‌کند.
- اگر Slave جواب Split داد:
  - جواب را بعداً خواهد داد.
  - آنگاه Arbiter ، من را آزادی کند ← HGRANT بلافاصله آزاد می‌شود.
  - وقتی آماده شد به Arbiter خبر می‌دهد ... ← آن هم نوبتی به من Split می‌دهد.
  - Arbiter انتقال Master درخواست دهنده را برمی‌گرداند.
  - در پایان هر ترانزکشن Arbiter بررسی می‌کند که آیا Slave آماده شده یا خیر.

Subject:

Year. Month. Date. 19



هدف از طراحی Bridge طراحی سخت افزاری یا نرم افزاری است که بتواند به طور همزمان بین دو دستگاه ① و ② تبدیل پروتکل را انجام دهد.

۱- تعریف کارکرد دستگاه

۲- تعریف یک ماژول SPI (سخت افزاری یا نرم افزاری)

۳- تعریف یک ماژول I2C (")

۴- برداشت درخواست‌های Master به Slave در بین پروتکل‌ها

می‌توان از پورت پارالل PC به عنوان یک پروتکل ساده استفاده کرد. چند بیت آن مربوط به SPI و چند بیت آن به I2C تخصیص می‌یابد. سپس با روش‌های نرم افزاری اعمال لازم را انجام داده، FSM نرم افزاری تولید می‌شود.

راه حل مورد نظر، طراحی سخت افزار مورد نظر است.

Subject:

Year. Month. Date. ( )

MII

* در بررسی سخت لایه OSI به دو لایه آخری پرداریم.

Layer n → Software

Data Link — Layer 2 →

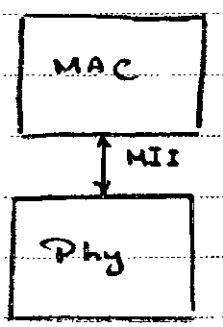
Phy — Layer 1 → Hardware

Ethernet ⟹ twisted pair — Phy

Wireless ⟹ antenna

* لایه 2 غالبا سخت افزاری پیاده می شود.

IEEE 802.3 (Ethernet)



Media Access Controller , Digital

Analog

برای اتصال این دو لایه نیاز به Interface است که بتواند آنها را پشتیبانی کند.

Media Independant Interface : MII *

Board در سطح .

Subject:

Year. Month. Date. 20

* اولین اصول طراحی MII

- تعداد پین ← در پروتکل Ethernet تعداد زیادی پین استفاده می‌شود.
- پهنای باند ← خودکار Battleneck نشود.

لـ حداکثر سرعت پروتکل Fast Eth : 100 Mbps

- انعطاف پذیری ← کاربرد در جاهای دیگر

Subject :

Year . Month . Date . ( )

Subject:

Year. Month. Date. 21

* در درون چیپ یک منطق دیجیتال می‌شود که کارهای مدار را دنبال می‌کند به آن Internal Logic می‌گویند. برای تست آن، ورودی و خروجی آنها باید دسترسی داخلی قرار دارد.
  - دسترسی موازی ← محدودیت پین
  - دسترسی سری ← در واحدهای دسترسی محلی به ترتیب داده‌ها چیده می‌شود و پس به منطق داخلی اعمال می‌شود. مثلاً اگر 1011 باشد به طور سریال به این واحد وارد می‌شود و خروجی نیز به طور سریال خارج می‌شود. هر این واحد boundry cell می‌باشد

http://www.gaepel.com/content/html_en/popup_en/bs_tut_#_.htm

1 ~ 9

* یکی از مزایای JTAG نسبت به ISC این است که نقاط داخلی منطق داخلی نیز می‌توان به آنها خروجی اضافی کرد و در آنها خروجی آنها بررسی کرد ولی در ISC امکان‌پذیر نیست.

* این چرخه تست چیزی حدود ۵ تا ۱۰ درصد است ولی در مقابل با Primary I/O که مرتبط با Pin Driver است، این رقم قابل تحمل است.

* TCK: (input) Test Clock
  TMS: (input) Test Mode Select
  TDI: (input) Test Drive In
  TDO: (output) Test Drive Out
  TRST: (optional) Test Restet

  TAP: Test Access Port

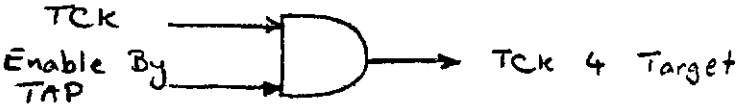
* برای اوقات به‌غیر از تست حالت درونی چیپ نخواهیم به آن دسترسی کنیم که به حالت خاصی برود آن را تست کنیم، Instruction Reg به همین منظور تعبیه شده است. (IR)

Subject :

Year . Month . Date . ( )

* مجموعه Test Data Reg از boundry cell تشکیل شده که طول آن برابر مجموع ورودی‌ها و خروجی‌هاست.

* TAP Controller به سیگنال‌های ورودی می‌گوید که به کدام قسمت دارد شود.
  - انبکار را با FSM انجام می‌دهد.
  - مسیر ورودی به این جاها می‌رود:
    - IR
    - TDR
    - Internal Logic
    - Bypass Reg
  - انبکار را اینجوری انجام می‌دهد:



Subject:

Year. Month. Date. 22

Device Drivers :

* Kernel
- هسته داخلی سیستم عامل
- شامل زیرروالهایی که کنترل سخت افزار را به عهده دارد.
- برای برنامه نویسی سخت افزار دو حالت داریم:
  * مود هسته ← بدون محافظ، آزادی عمل کامل، نیاز به تسلط زیاد
  * مود کاربر ← محافظت شده، برخی از دستورالعمل ها در دسترس نیست
- نوشتن DD در مود هسته است و احتمال خرابی سخت افزار در اثر آن است.
- وظایف هسته
  - دسترسی نرم افزار به سخت افزار ← System Call
  - I/O Management, Virtual Memory, Scheduling ☑
  - کنترل منابع تخصیص داده شده ← Resource Management
  - مدیریت تخت، صفحات حافظه، حافظه Swap ← Mem Manage
  - اولویت بندی و خدمت دهی منابع و ... ← Periority
  - کنترل منابع سخت افزاری ← Physical Devices

* تعریف DD

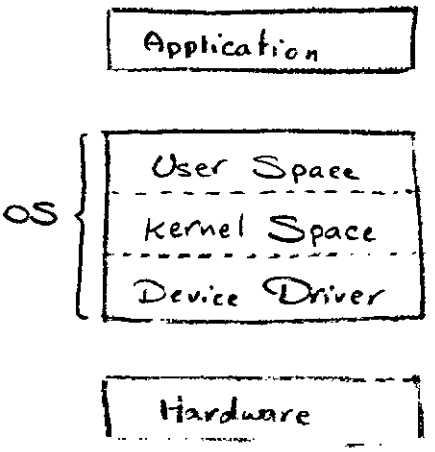
- Loadable Kernel Module that...

نیازی ندارد از اول در سیستم باشد و می تواند بعداً به آن اضافه شود.

... Manage data transfer between device and OS.

Subject :

Year Month Date ( )



* Device Driver ≡ Firmware

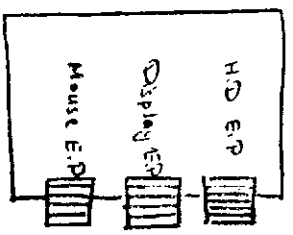
- دارای انعطاف کمتر از نرم‌افزار به دلیل وابستگی به سخت‌افزار
- تحویل‌های سخت‌افزار را تحمل می‌کند
- ولی به نوعی نرم‌افزار است.

* Plug & Play

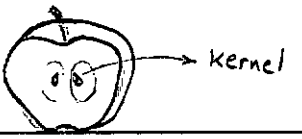
- تبادل سخت‌افزاری اطلاعات
- دستگاه خود را معرفی می‌کند
- OS و تبادل DD مربوطه برقرار می‌گردد.
  - ★ اگر پیدا کرد نصب می‌کند و تمام.
  - ★ اگر پیدا نکرد به کاربر اطلاع می‌دهد.

* Mail Box

- ارتباط Kernel و DD
- دستورات High level به تعدادی از دستورات تبدیل می‌شوند که در این جدول می‌آید.
- یک Data Structure است.
- به هر سطر جدول این DC یک Entry Point (E.P.) می‌گویند.
- کلاً وظیفه‌ی آن Abstraction است.



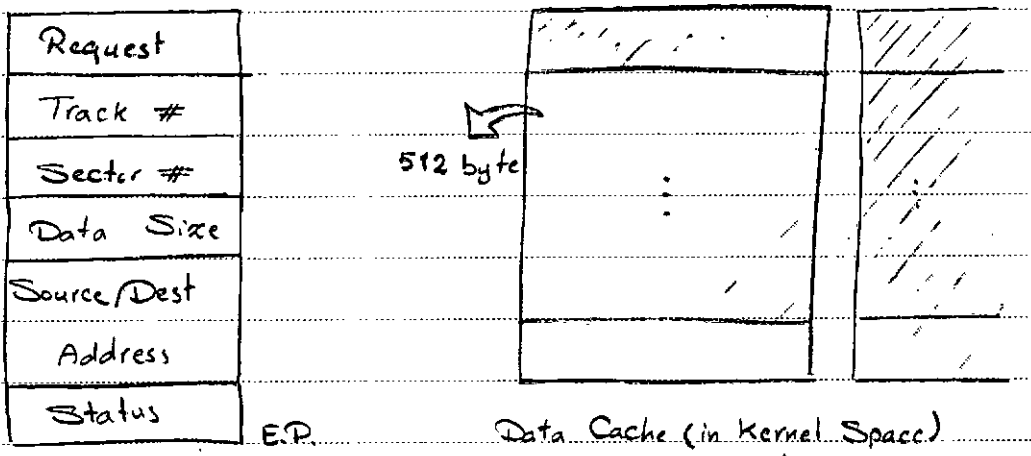
- کلیک دوبل روی پوشه
- نشان Mouse را بخوان
- محتویات پوشه را بخوان
- روی Display نشان بده
- برنامه‌ای از این جدول باید دارای E.P باشند.



* انواع Device Drive

I. Block

- درایو I/O را با استفاده از بافرهای بزرگ انجام می دهد.
- بافر از فضای داخل kernel space می گیرد.
- برای device ها که اطلاعات زیادی را جابجا می کنند، OS باید cache بسیار بزرگی داشته باشد.
- kernel موظف است علاوه بر اختصاص فضای حافظه، نقل و انتقال اطلاعات را هم به عهده بگیرد.
- مثال: Flash Driver و Disk Driver



- تخصیص حافظه و پس گرفتن آن در DD به عهده OS است.
- DD از دید کاربر مخفی است و OS واسط میان آنهاست.
- داده ها در DD ها از نوع positional است ← آدرس شروع + جابجایی

II. Character

- بافر به اندازه یک بایت است.
- مثال: Interactive Terminal, Line Printer
- برنامه اطلاعات را می تواند جابجا کند، ولی معمولاً درخواست کاری را دریافت کرده و کار را کنترل و مدیریت می شود.

III. Network

- کپسوله سیستم شبکه را به یک NI منتقل می‌کند.
- NI را برای عمل آماده می‌کند.
- دریافت اطلاعات را کنترل می‌کند.
- مثال: LAN

IV> Pseudodevice

- مثال: PDF Distiller
- عملاً سخت افزاری را مدل می‌کند درحالی که سخت افزاری وجود ندارد.
- معمولاً روی باس کار نمی‌کند ←

* مدارد فراخدای DD

I> Auto configuration

invoke شامل روال‌ها برای بارشدن در حافظه

II> I/O Operation

فرمان I,O به OS توسط کاربر، برنامه یا خود دستگاه

III> Interrupt Handling

مواجهه با رویداد خارجی (مثل قطع برق)

IV> Special Request

دستورات درخواست شده توسط OS به فرمان کاربر توسط IOCH

V> Reinitialization

نصب دستگاه جدید

VI> User Level Request ...

الکتا این مدارد به طور غیرمستقیم از OS می‌آید.

Subject:

Year. Month. Date. 24

* تنظیم DD
  - برای هر دستگاه تنظیمات خاص نیاز است.
  - پس از تنظیمات باید compile شود.
  - سپس روی kernel پیاده می‌شود.
  - روشهای تنظیم
    * static ← تنظیم ثابت ← وارد kernel می‌شود ← بدون تغییر
    * dynamic ← داخل کد DD بسیاری تعاملات با kernel پیش‌بینی می‌شود و از طریق kernel می‌توان روی کار دستگاه تأثیر می‌گذارد.
  - در Auto Config، تنظیمات روی kernel روی DD تغییری داده نمی‌شود، بلکه تنها این تغییرات را در دست کاربر قرار می‌دهد.

Subject :

Year . Month . Date . ( )