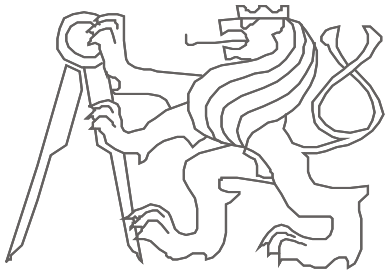
Architektury počítačů

Sběrnice PCI a PCIe

+ HD a DMA

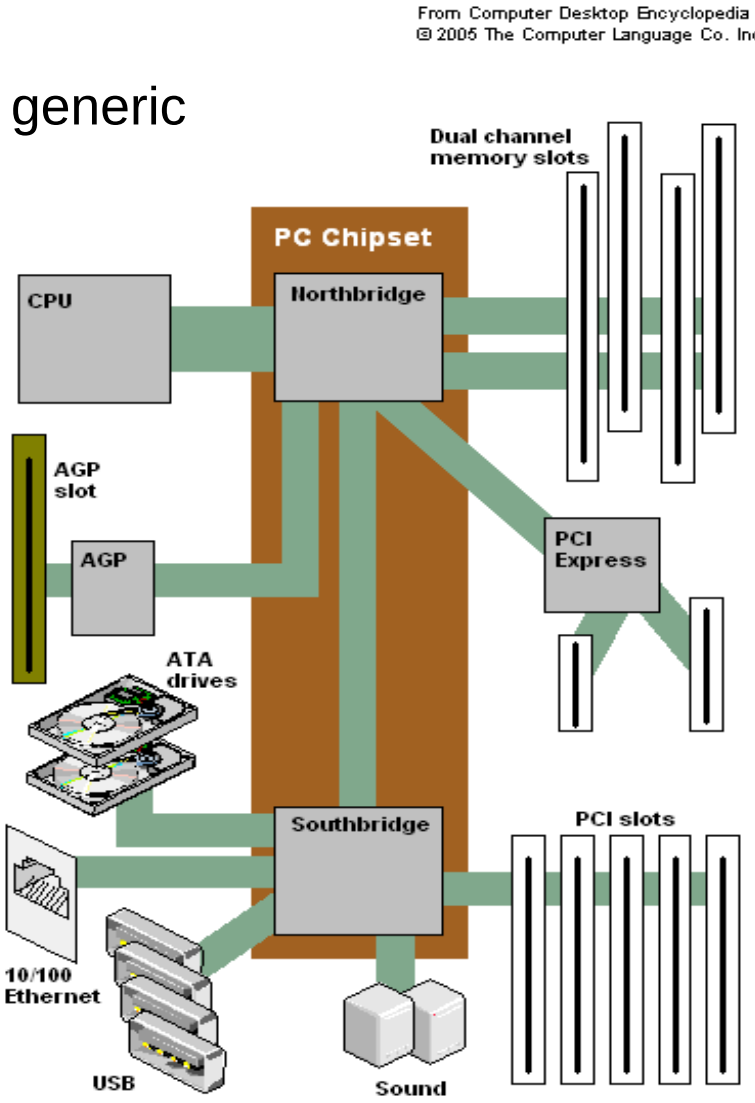
Richard Šusta, Pavel Píša



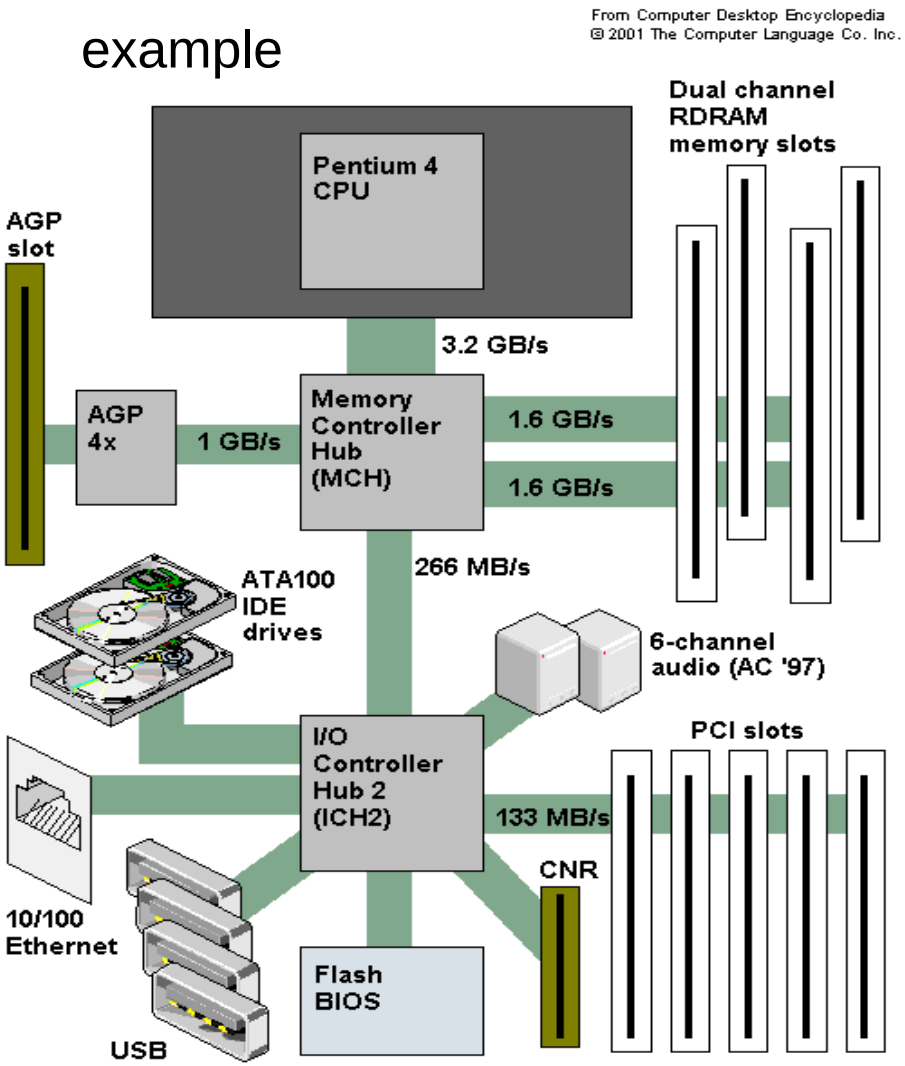
České vysoké učení technické, Fakulta elektrotechnická

4th lecture: Computer architecture (desktop x86 PC)

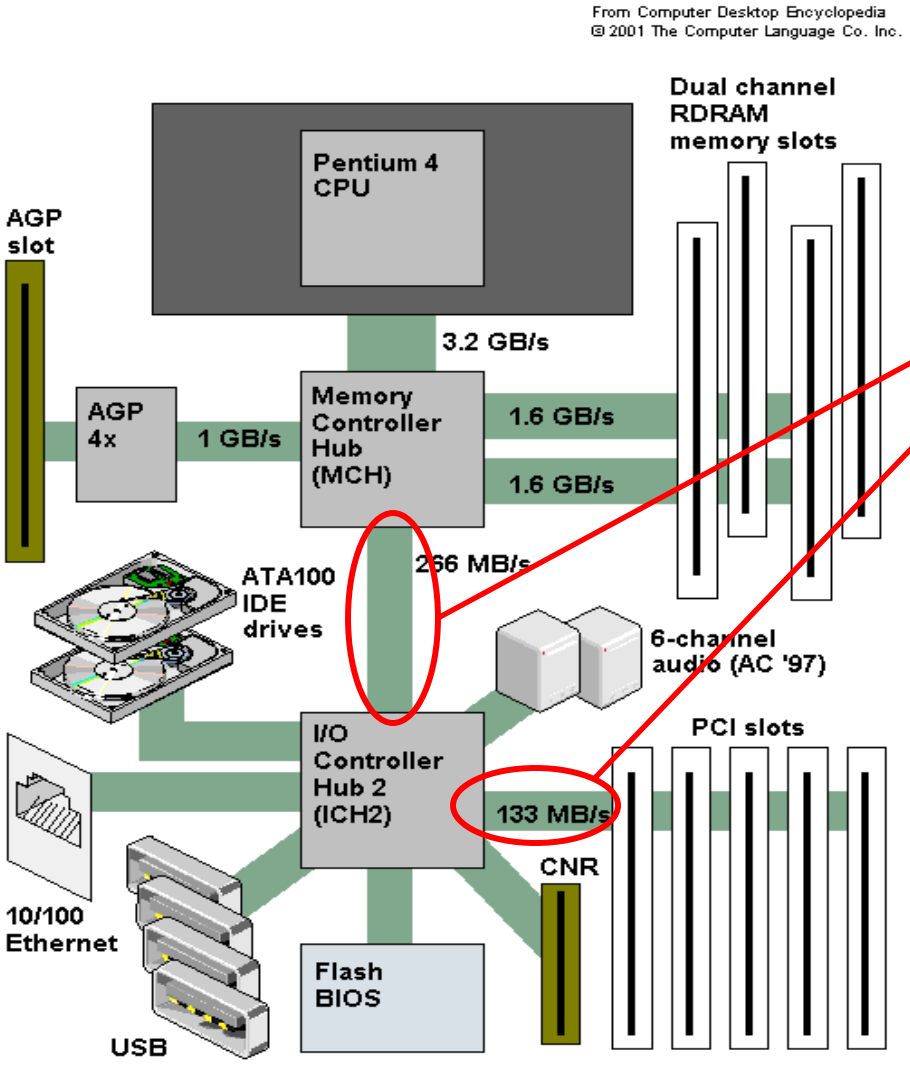
generic



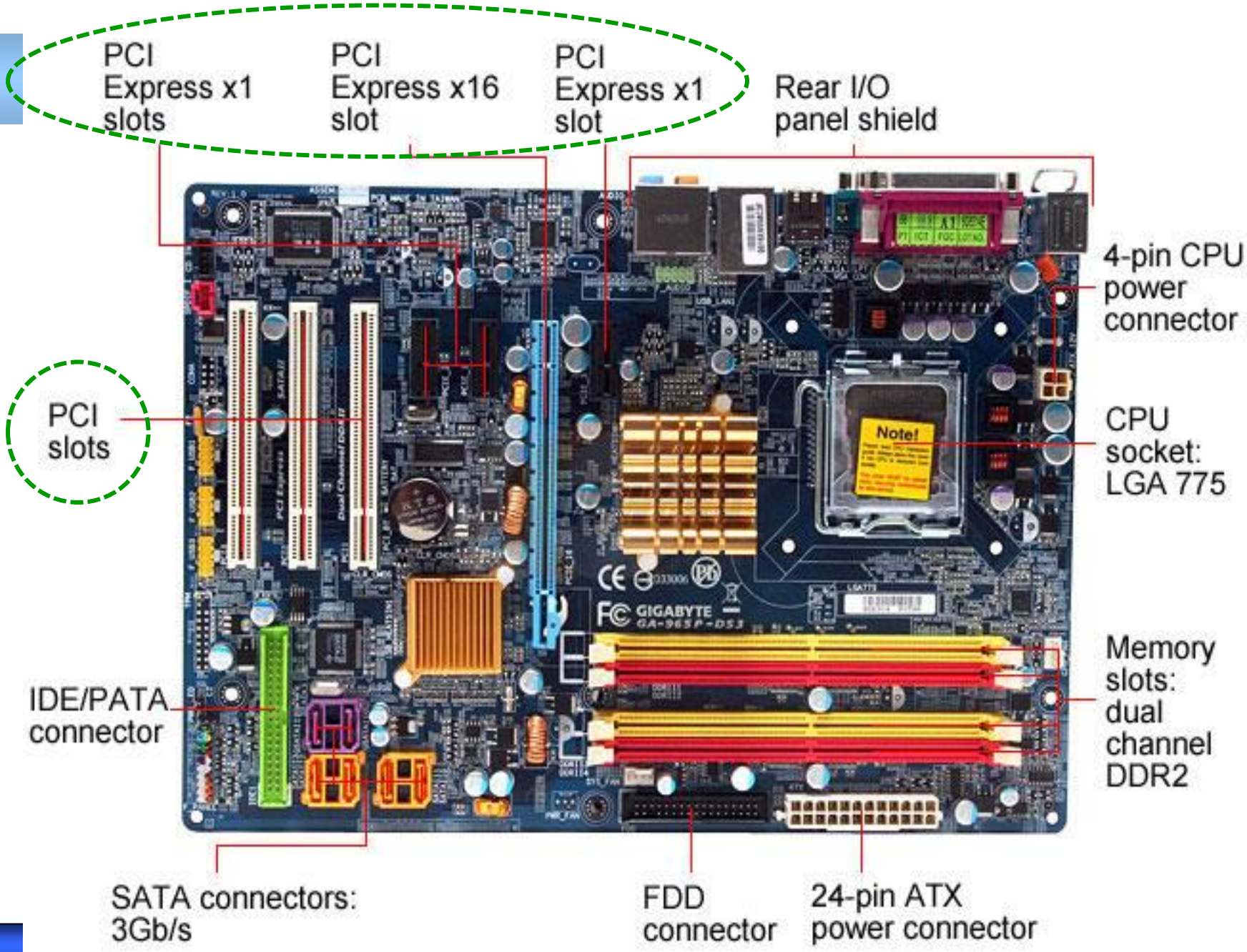
example



Motivation



Goal of today lecture
understand
bus interconnection
technologies

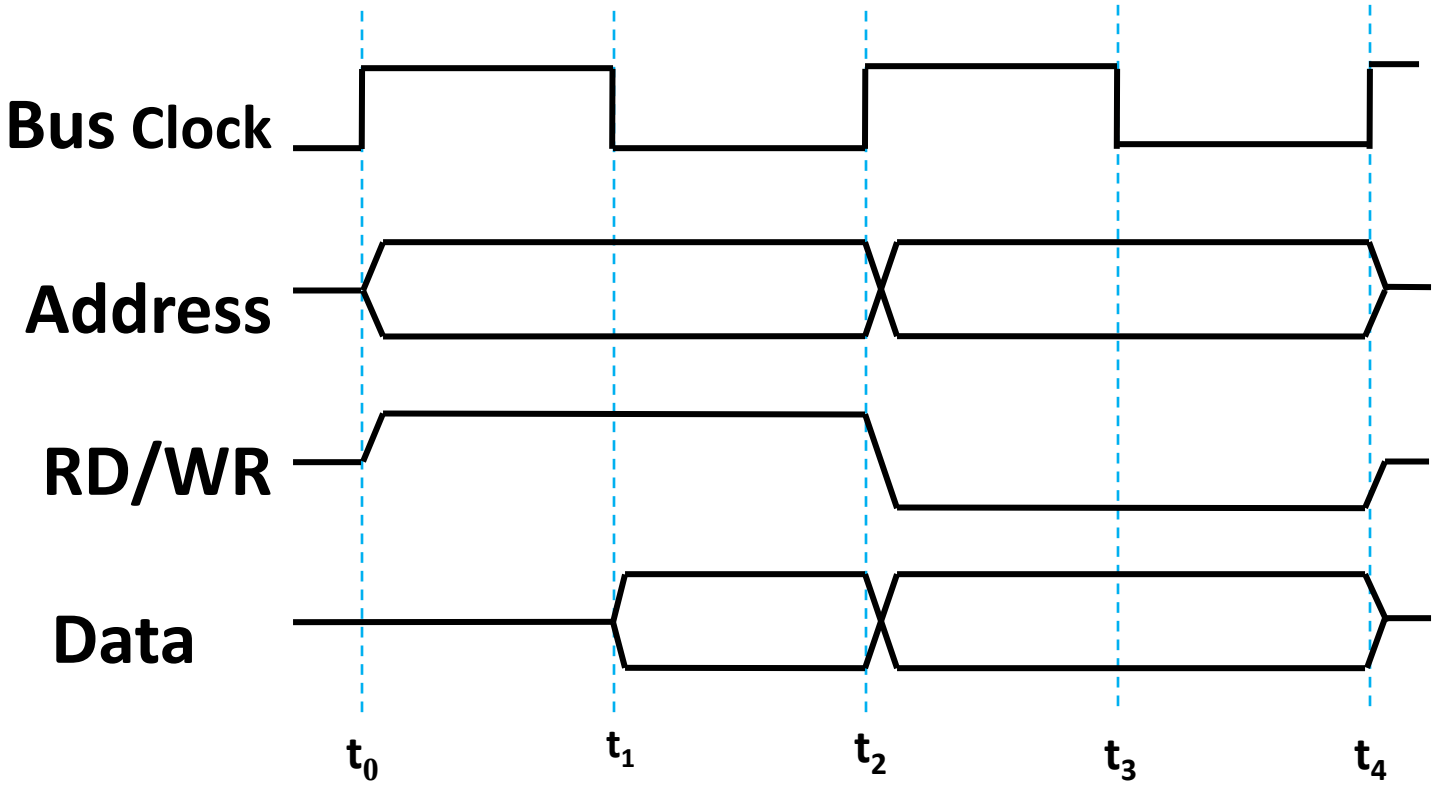

PCI Express x1 slots
PCI Express x16 slot
PCI Express x1 slot
Rear I/O panel shield
4-pin CPU power connector
PCI slots
CPU socket: LGA 775
Memory slots: dual channel DDR2
IDE/PATA connector
SATA connectors: 3Gb/s
FDD connector
24-pin ATX power connector
Note!
GIGABYTE
GA-965P-DS3

Source:

Synchronous Parallel Data Transfer

The events are determined by a clock!

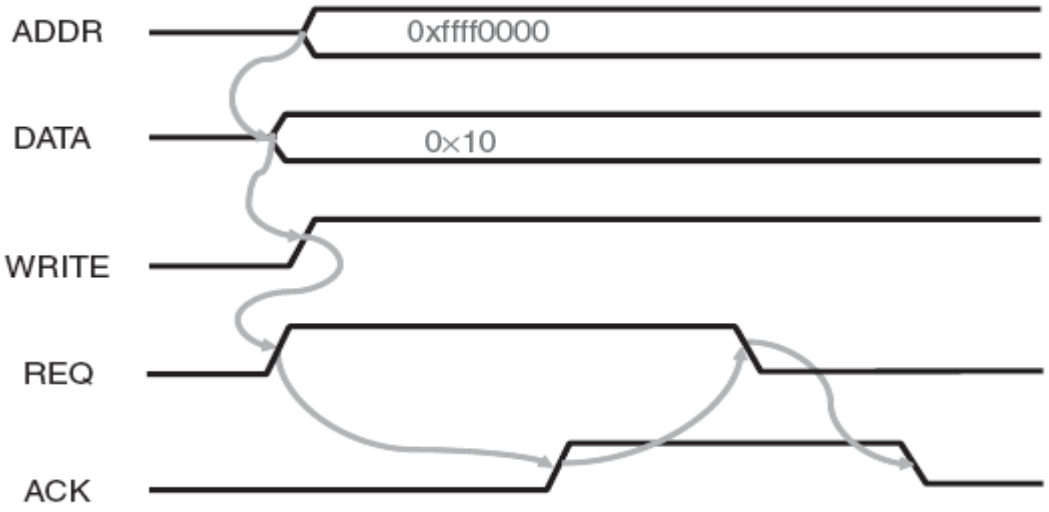
Good synchronization of all signals is absolutely essential!



Source: Sudeep Pasricha,
On Chip Communication, Colorado 2011

Asynchronous Bus

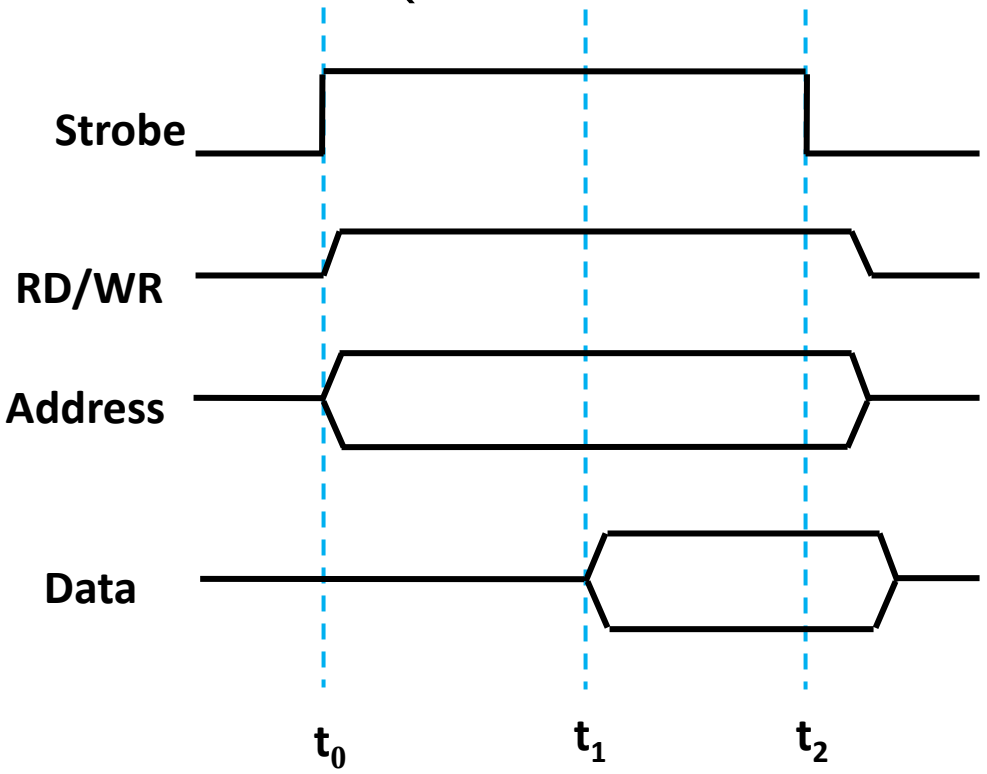
- Not clocked
- Requires a handshaking protocol (*It behaves as TCP internet protocol*)
  - performance not as good as that of synchronous bus
  - No need for frequency converters, but does need extra lines
- Does not suffer from clock skew like the synchronous bus



Source: Sudeep Pasricha,
On Chip Communication, Colorado 2011

Asynchronous Parallel Data Transfer by Strobing

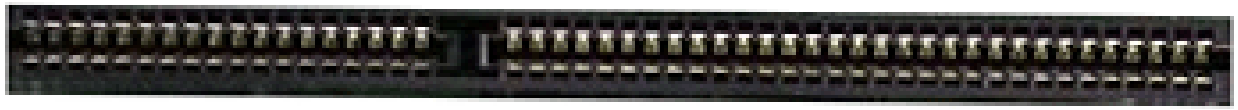
Strobing accelerates asynchronous bus operations,
but it is less reliable. (*It behaves as UDP internet protocol*)



Source: Sudeep Pasricha,
On Chip Communication, Colorado 2011

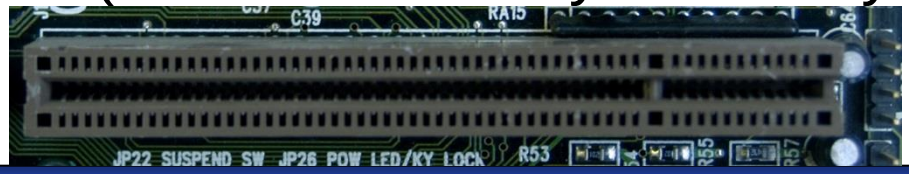
Older Computer Busses

* **ISA bus**
  16-bit wide
  7 MHz speed



* **EISA Extended ISA (EISA)**
  32-bit wide
  8,3 MHz speed

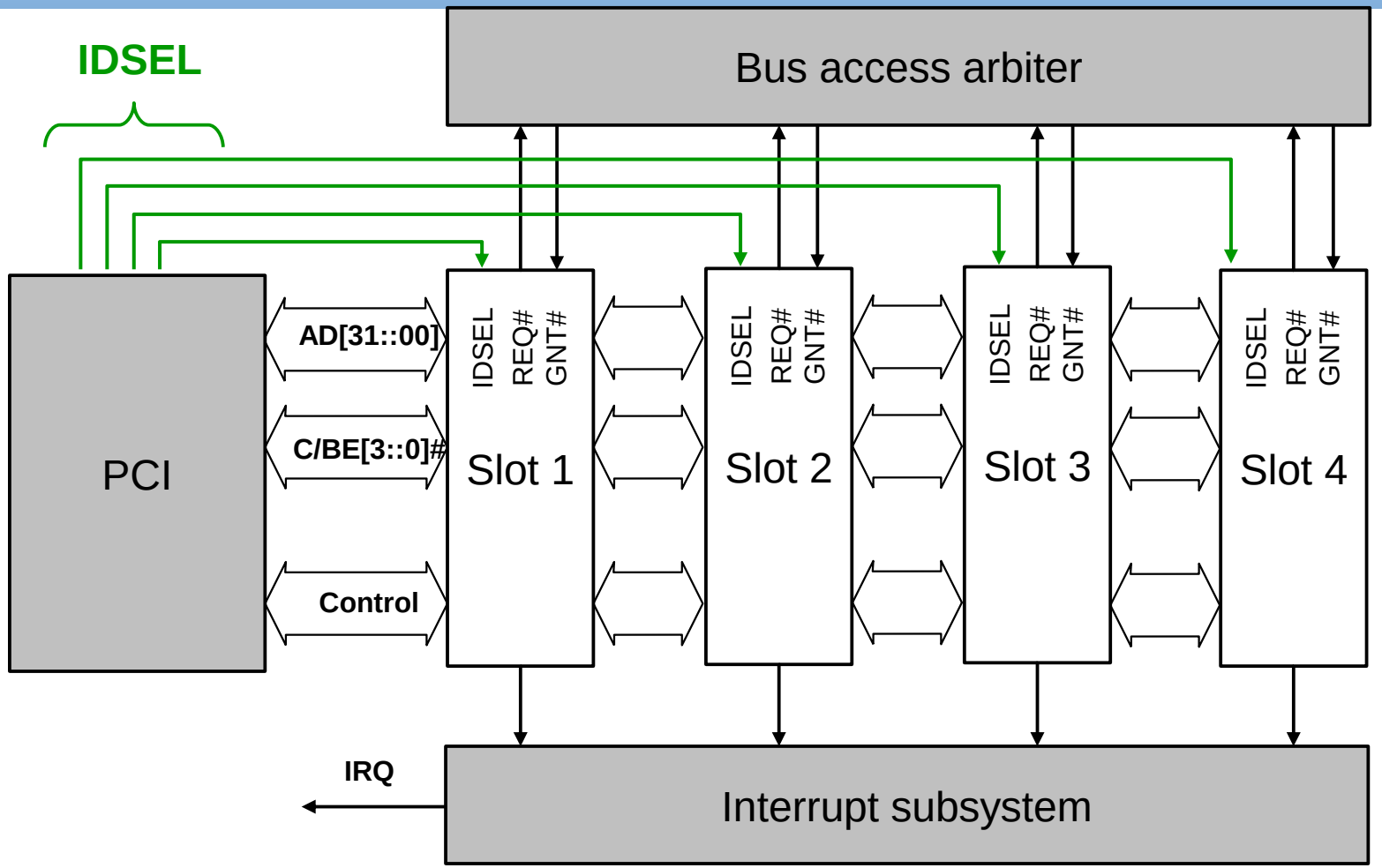
* **VESA** - Video Electronics Standards Association's VESA
  32-bit wide
  66 MHz speed (but it was usually used only at 33 MHz)



* PCI Bus

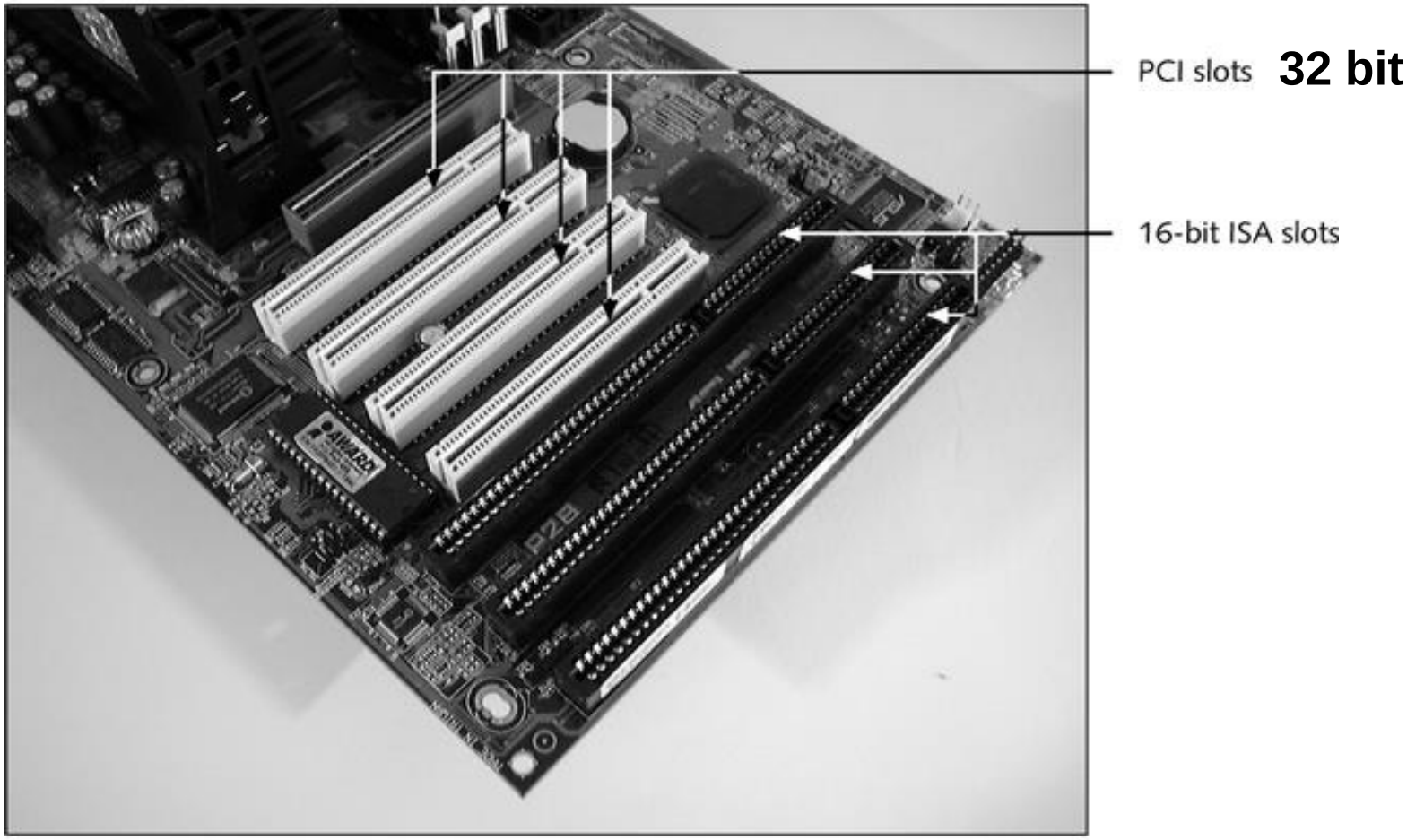
Peripheral Component Interconnect

PCI - architecture



Note: *During initial configuration transactions only, the IDSEL is used to indicate to a PCI endpoint (or bridge) that it is currently selected.*

ISA and PCI on Board



PCI terms and definitions I.

- Two devices participate in each bus transaction:
  - **Initiator** (starts the transaction) × **Target** (obeys request)
  - **Initiator** = Bus Master,
  - **Target** = Slave for current transaction.
- Initiator and target role does not directly impose data source and receiver role!

PCI terms and definitions II.

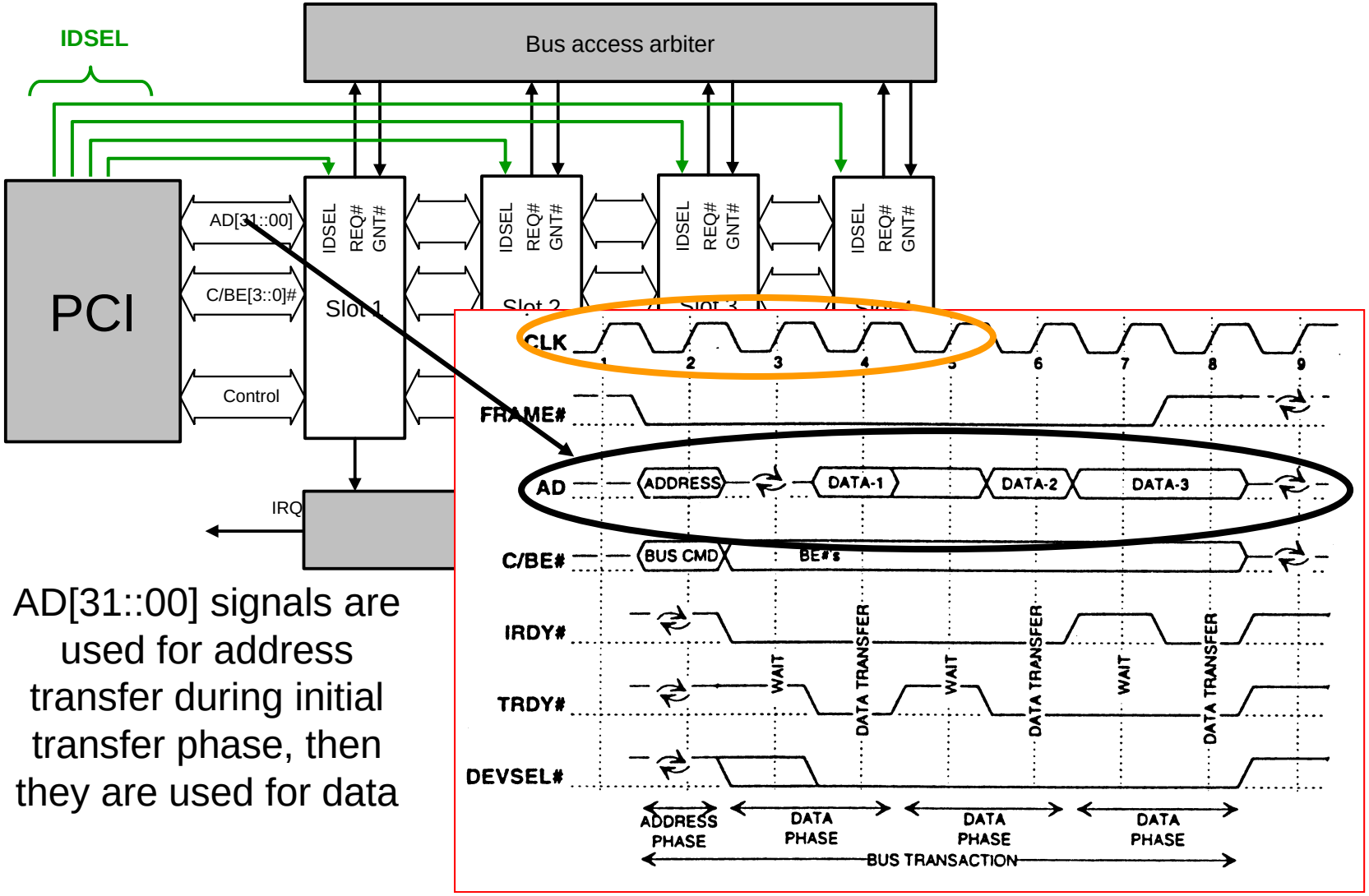
- What does it mean that bus is multimaster?
  - More participants can act as Bus Master – initiate transfers!
- When bus signals are shared, then transactions need to be serialized and only one device can act as master at any given time instant
  - Bus master is responsible to grant bus to requesting participant
- Who takes care of bus arbitration?
  - One dedicated device or bus backplane

Note: A decentralized (distributed) bus arbitration also exists, i.e., all devices participate in the selection of the next bus master, but not for PCI.

PCI terms and definitions III.

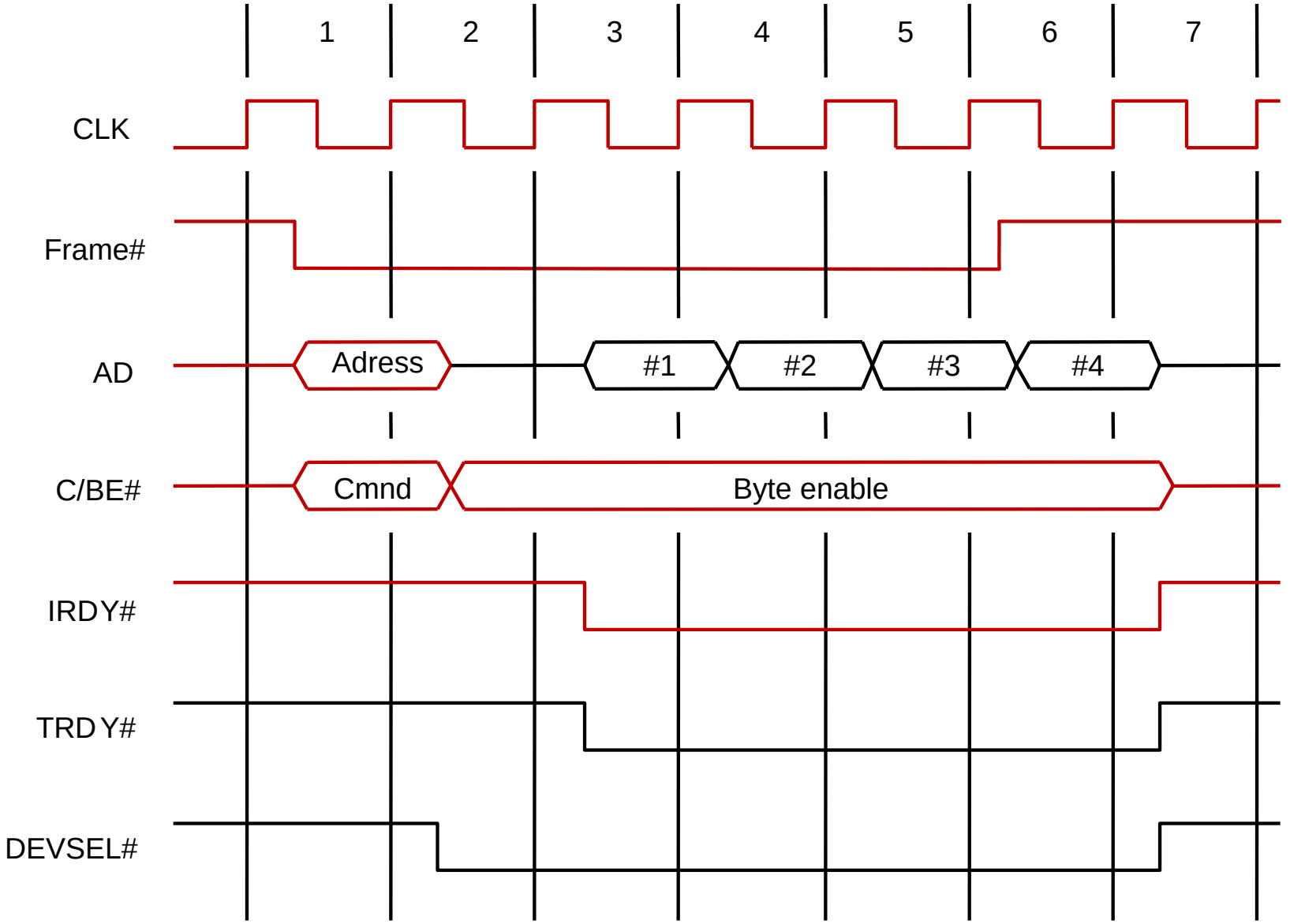
- The rising clock edge is reference/trigger point for all phase of the bus cycle/transaction timing
- Bus cycle is formed in most cases by
  - **address phase**
  - **data phase**
- Premature termination of data transfer is possible during bus cycle
- Transfer synchronization itself is pseudo-synchronous

Bus/signal lines reuse/multiplexing

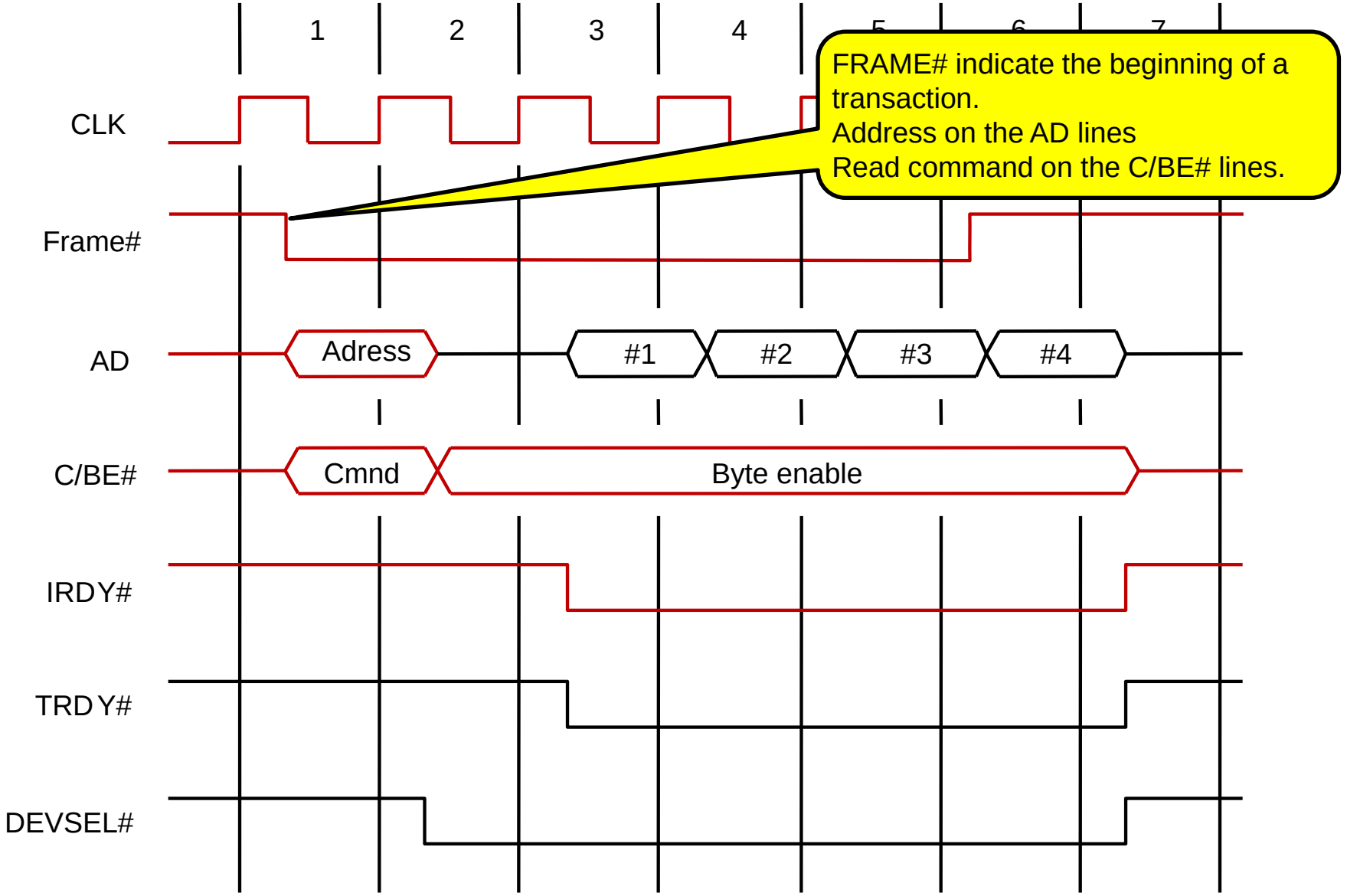


AD[31::00] signals are used for address transfer during initial transfer phase, then they are used for data

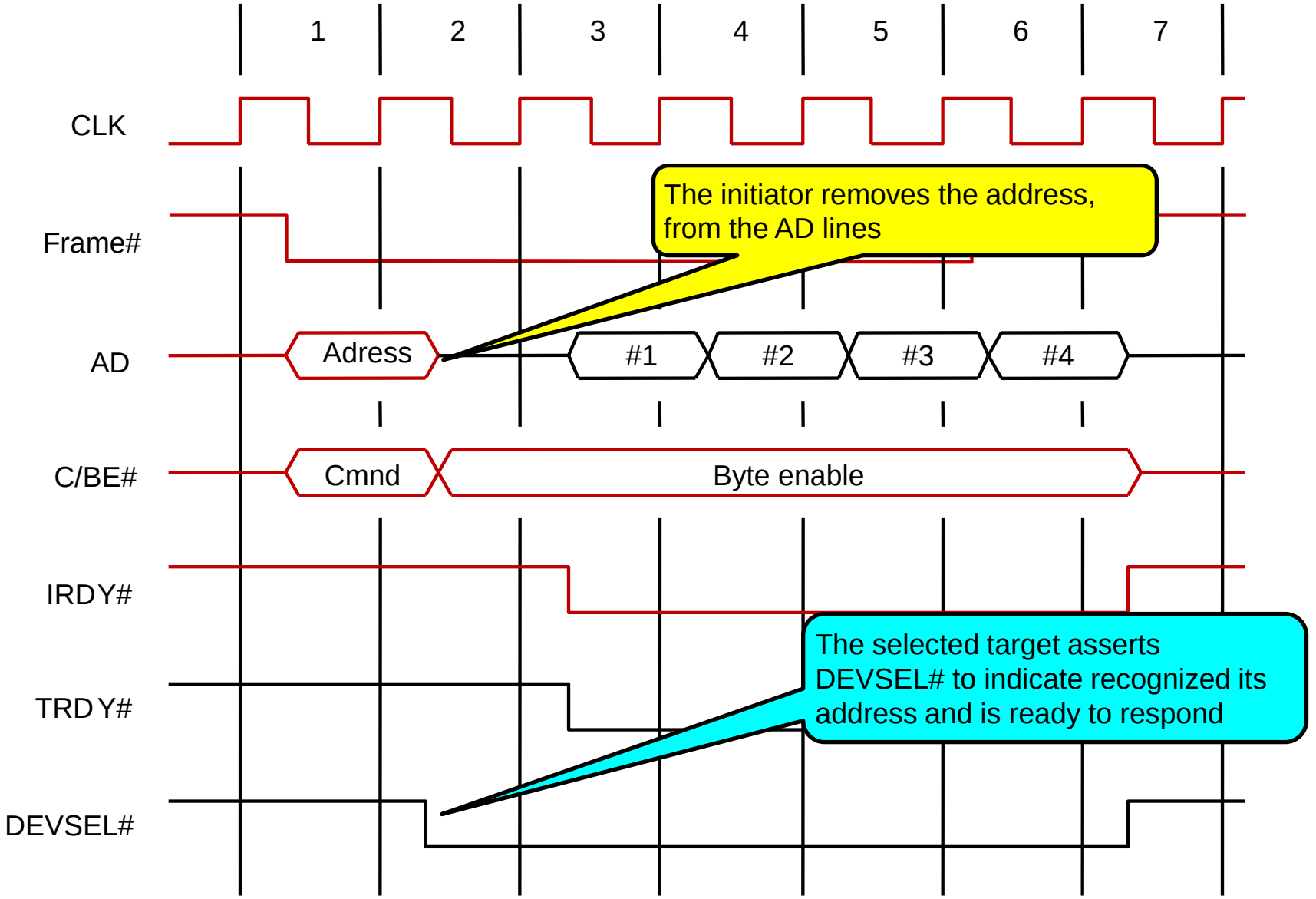
A read operation on the PCI bus



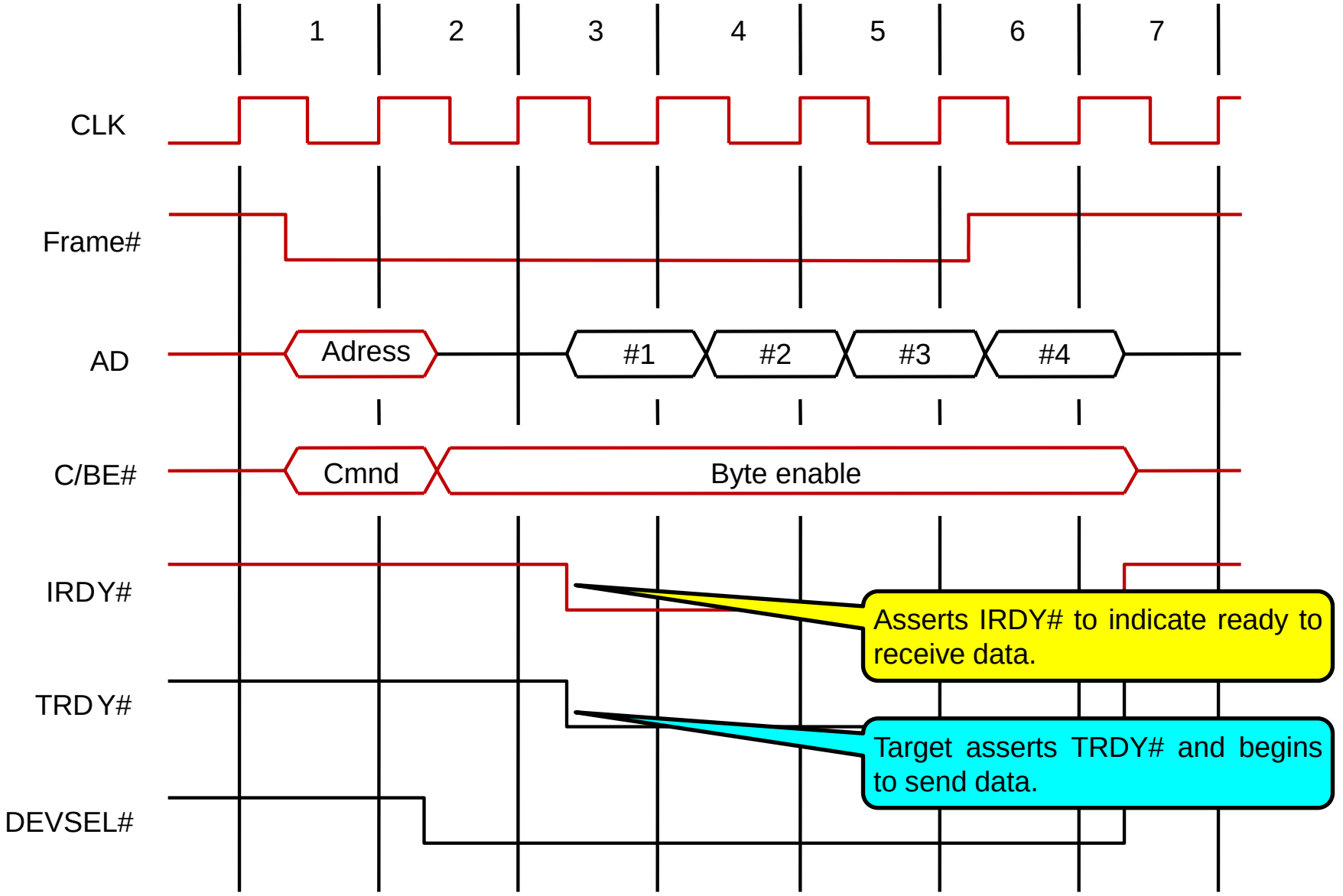
A read operation on the PCI bus



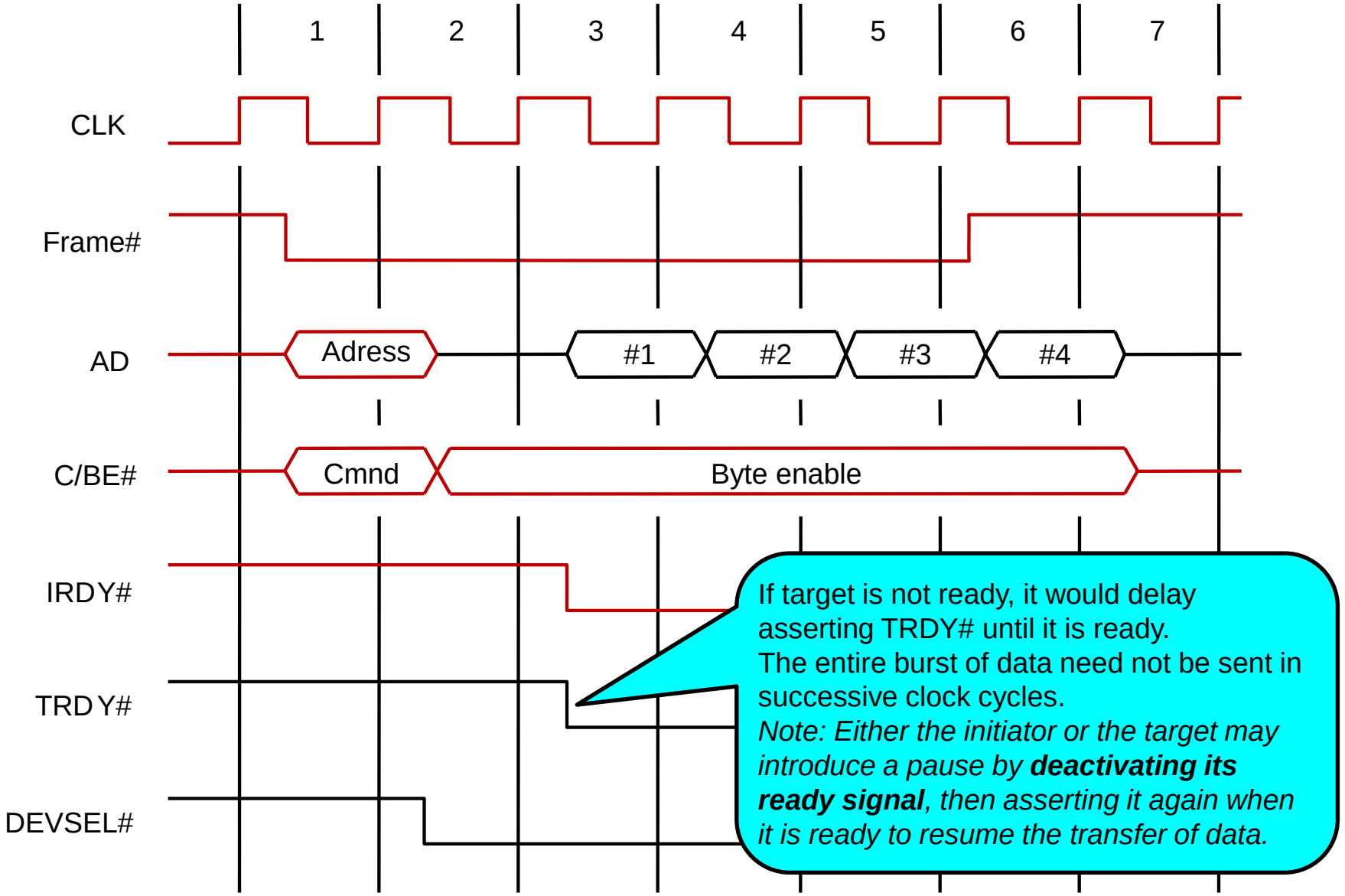
A read operation on the PCI bus



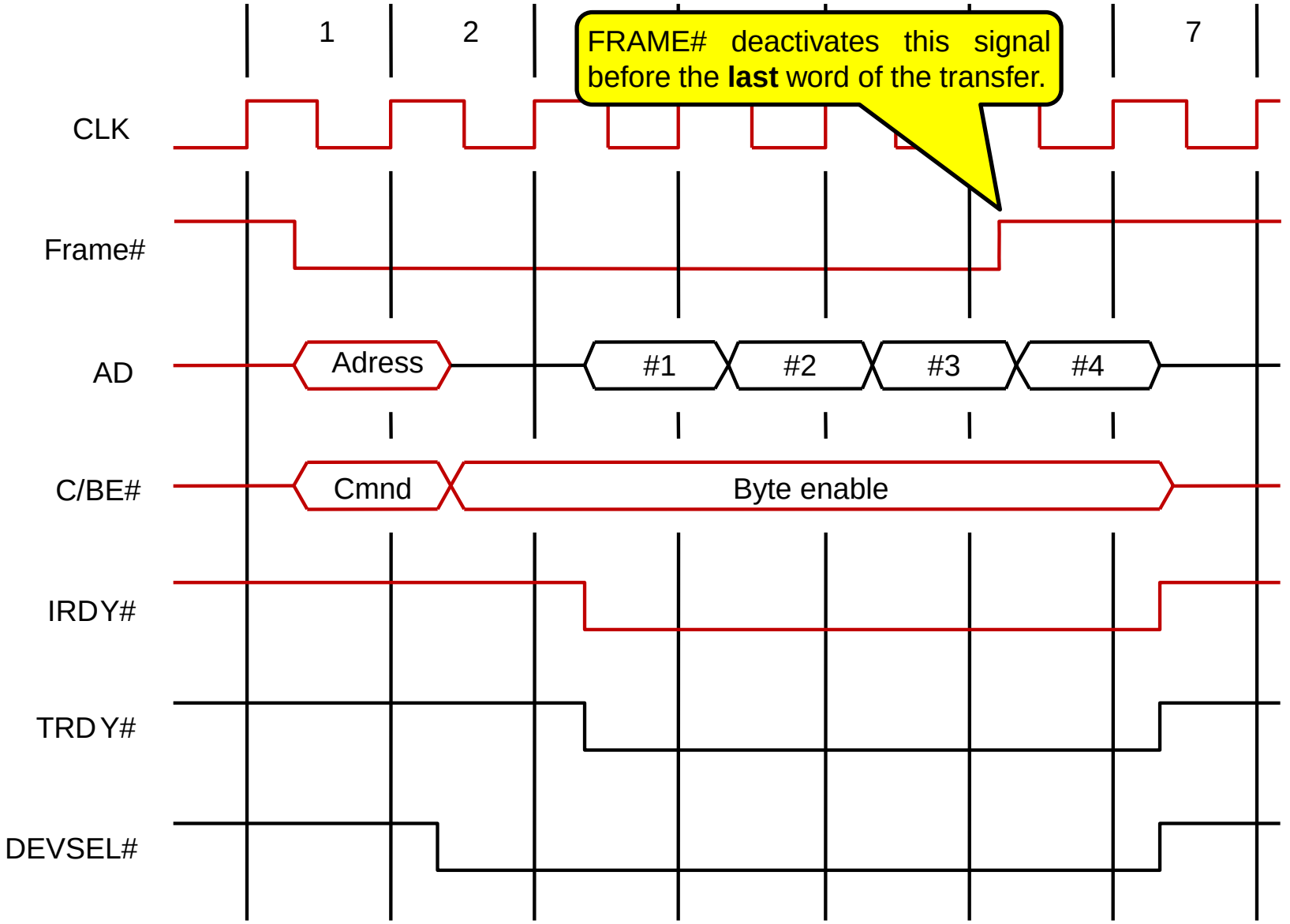
A read operation on the PCI bus



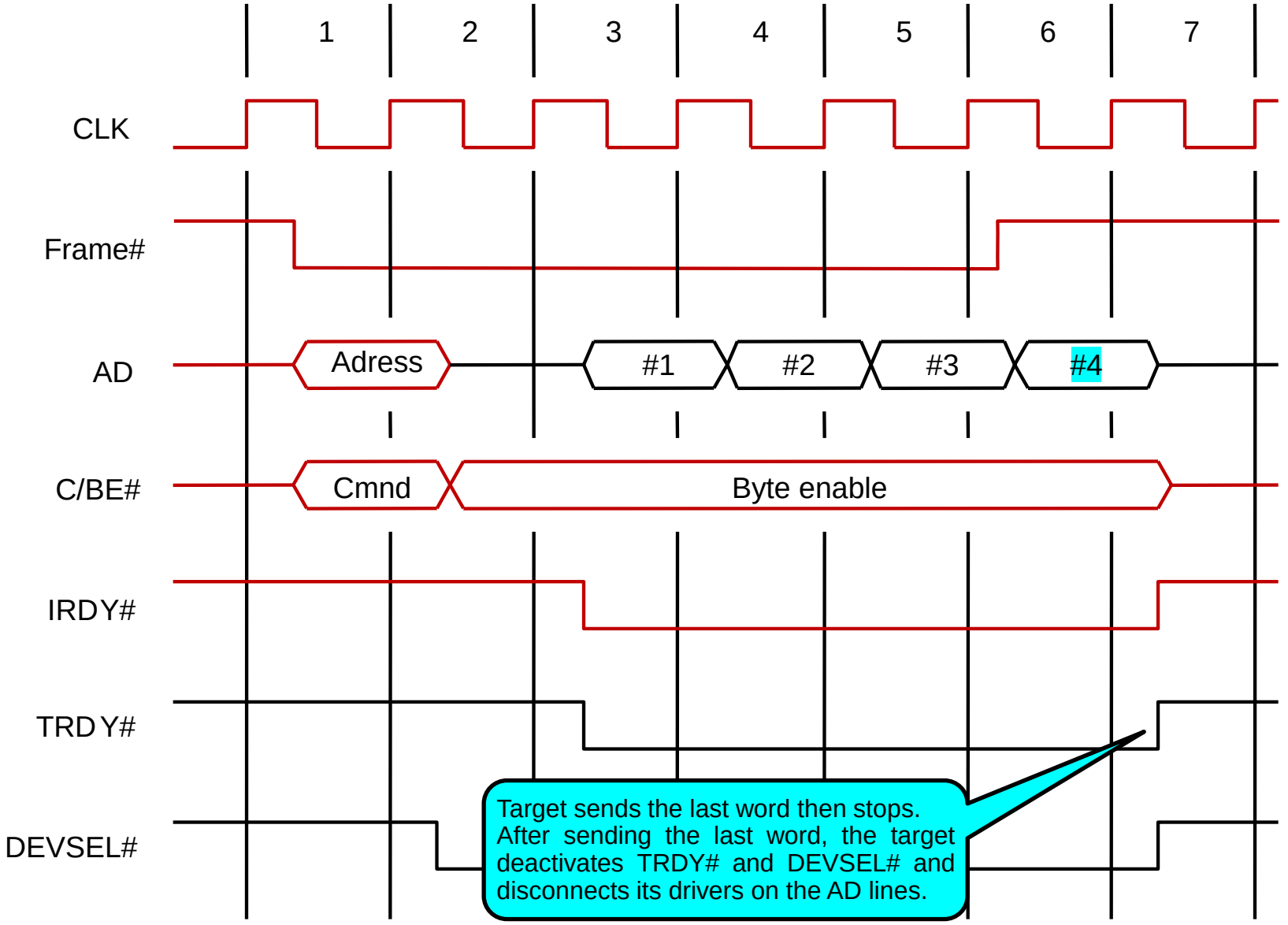
A read operation on the PCI bus



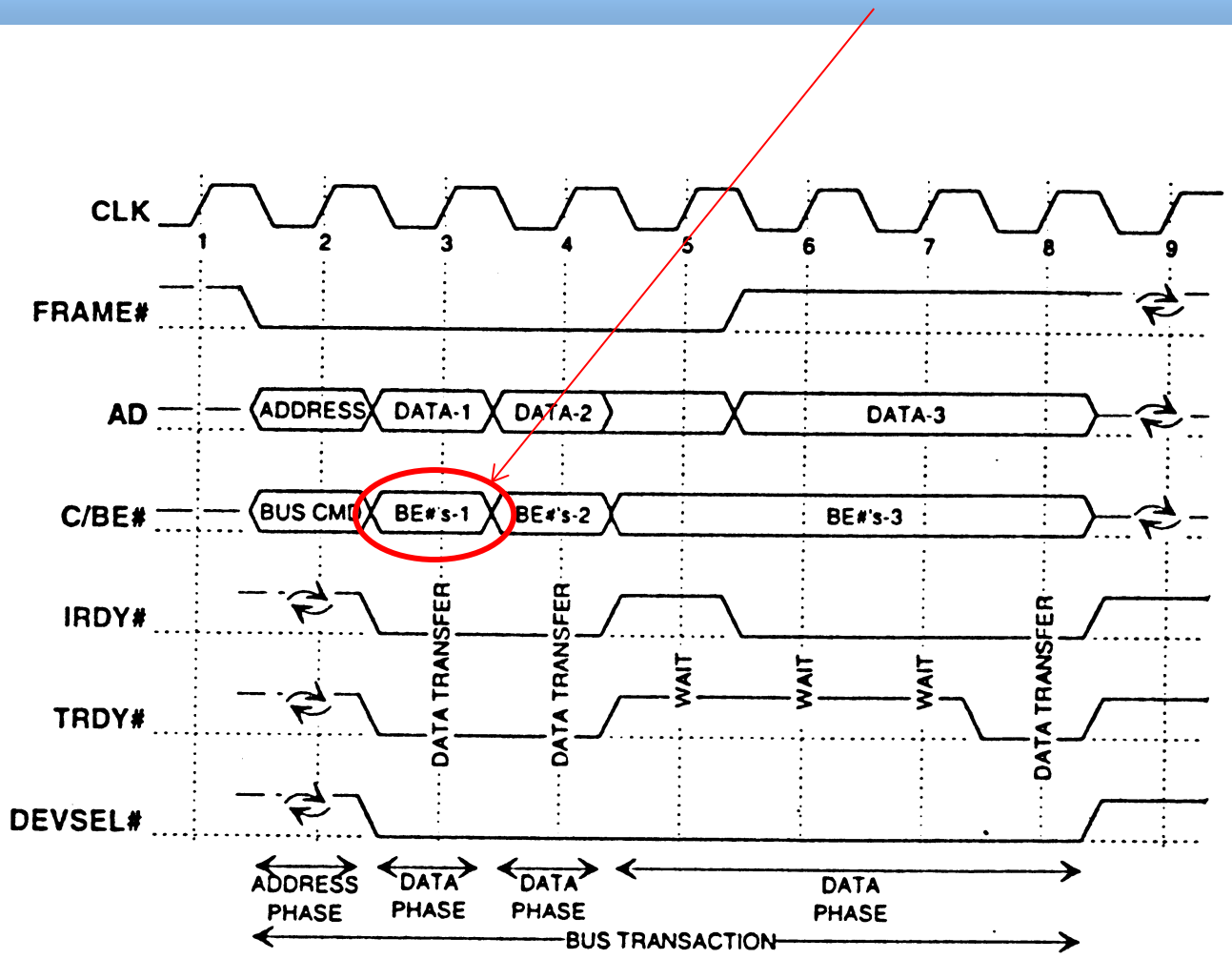
A read operation on the PCI bus



A read operation on the PCI bus



PCI bus memory **write** timing



It begins immediately after address

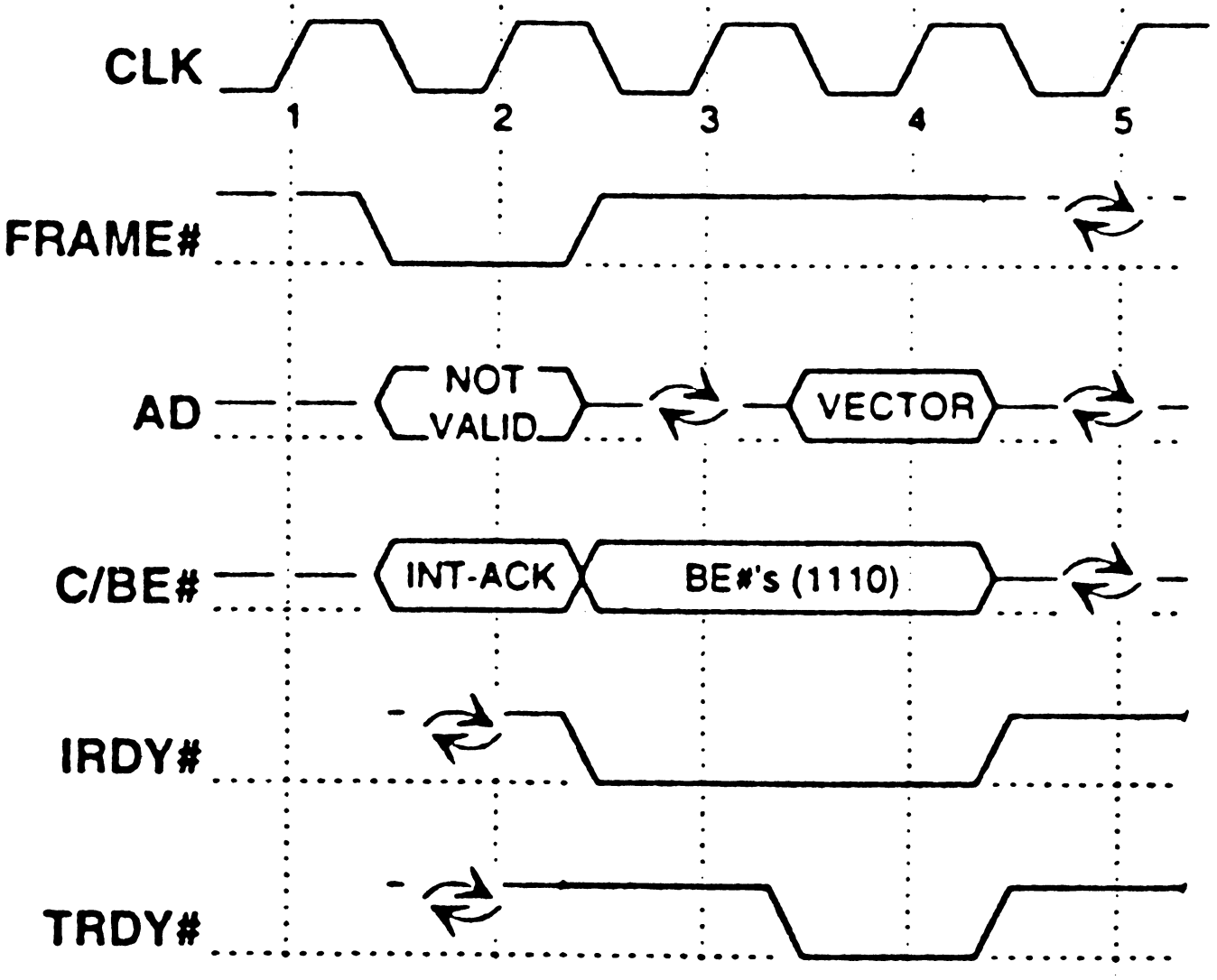
Bus cycle kind/direction – command – specified by C/BE

C/BE[0::3]#	Bus command (BUS CMD)
0000	Interrupt Acknowledge
0001	Special Cycle
0010	I/O Read
0011	I/O Write
0100	Reserved
0101	Reserved
0110	Memory Read
0111	Memory Write
1000	Reserved
1001	Reserved
1010	Configuration Read (only 11 low addr bits for fnc and reg + IDSEL)
1011	Configuration Write (only 11 low addr bits for fnc and reg + IDSEL)
1100	Memory Read Multiple
1101	Dual Address Cycle (more than 32 bits for address – i.e. 64-bit)
1110	Memory Read Line
1111	Memory Write and Invalidate

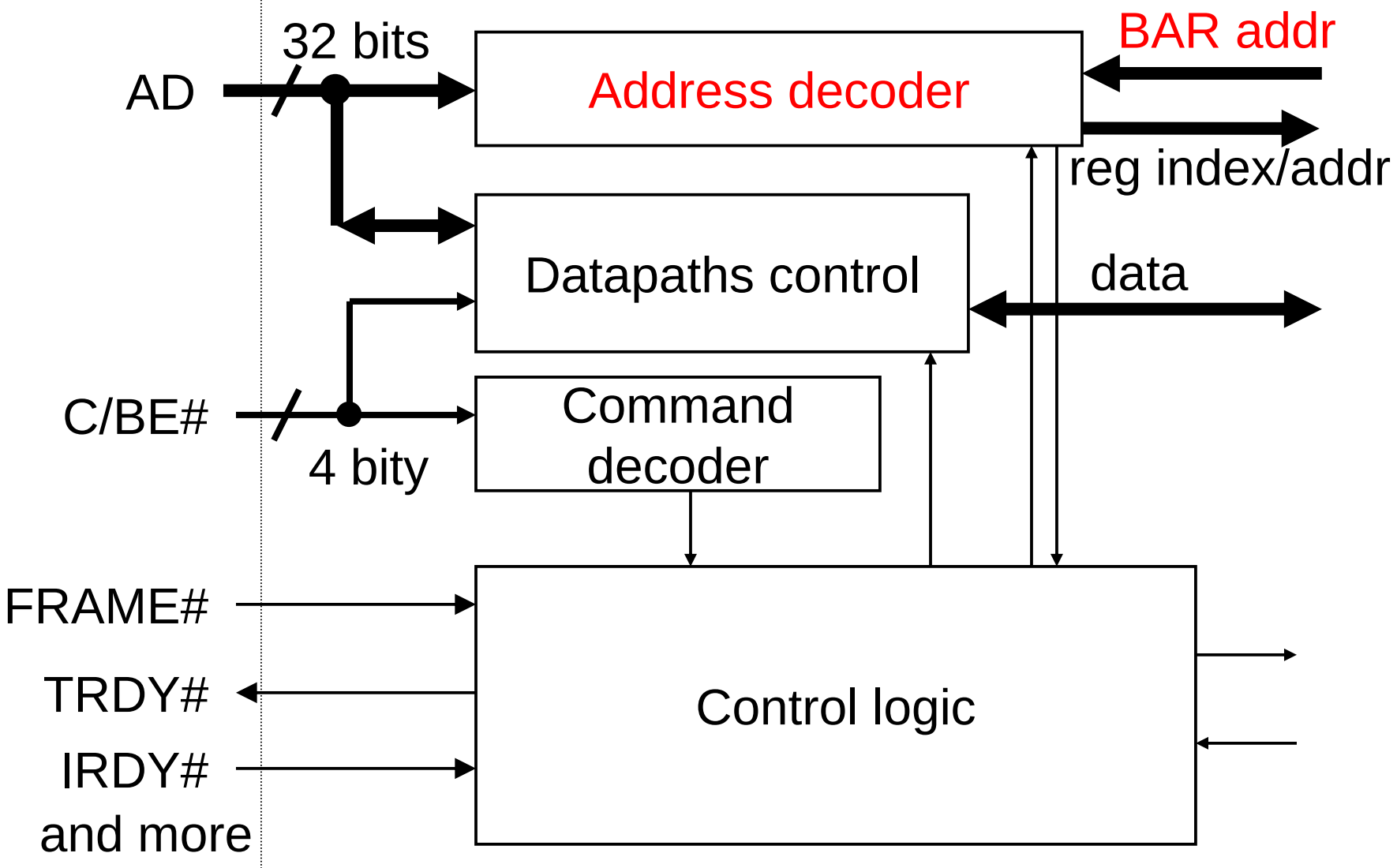
Interrupt acknowledge cycle

- Processor needs time to save interrupted execution state to allow state restoration after return from service routine
- Interrupt controller provides vector number (assigned to the asynchronous event) and CPU is required to translate it to the interrupt service routine start address
- All these activities require some time

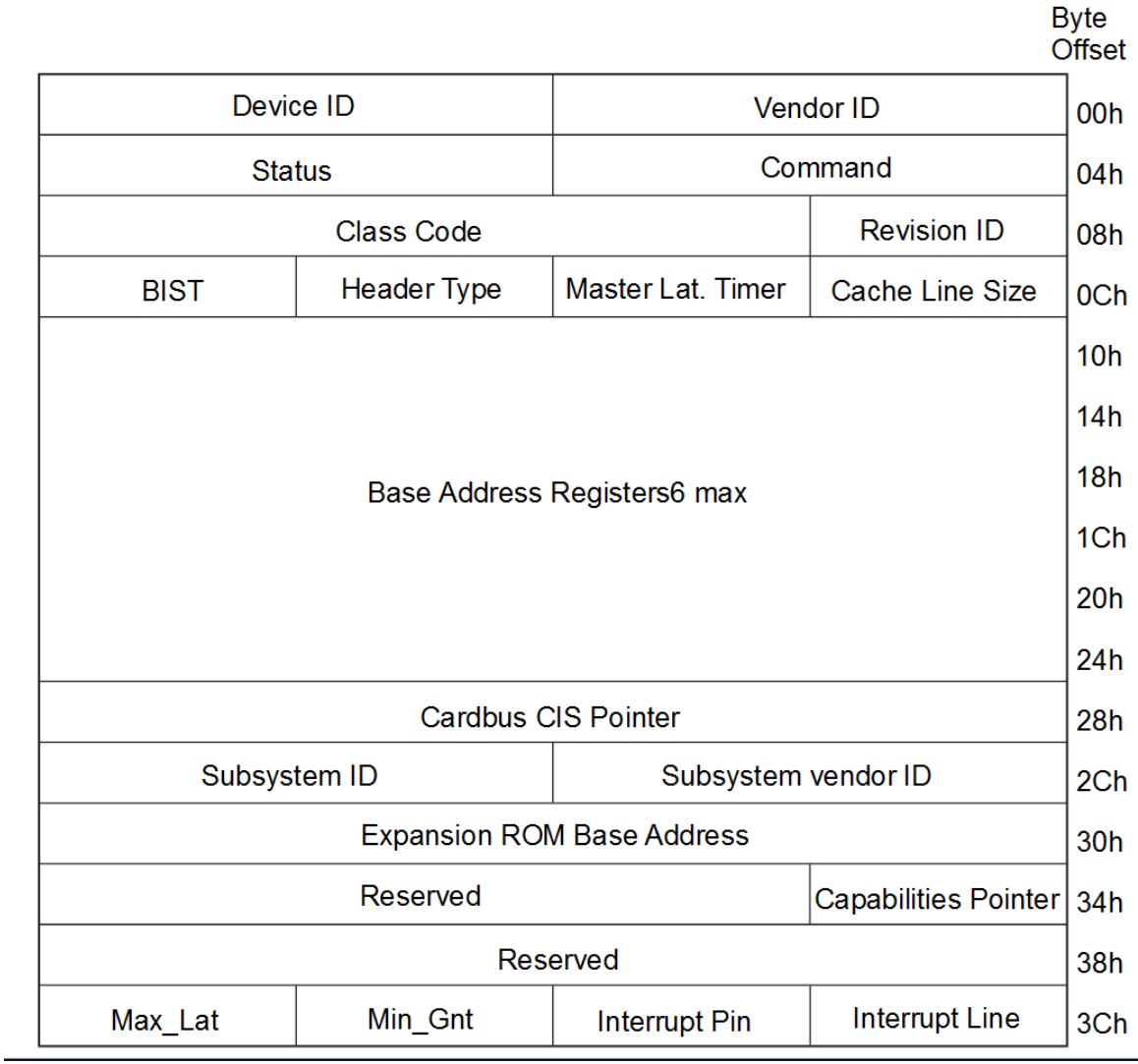
Interrupt acknowledge cycle timing



PCI Device/Card Interface

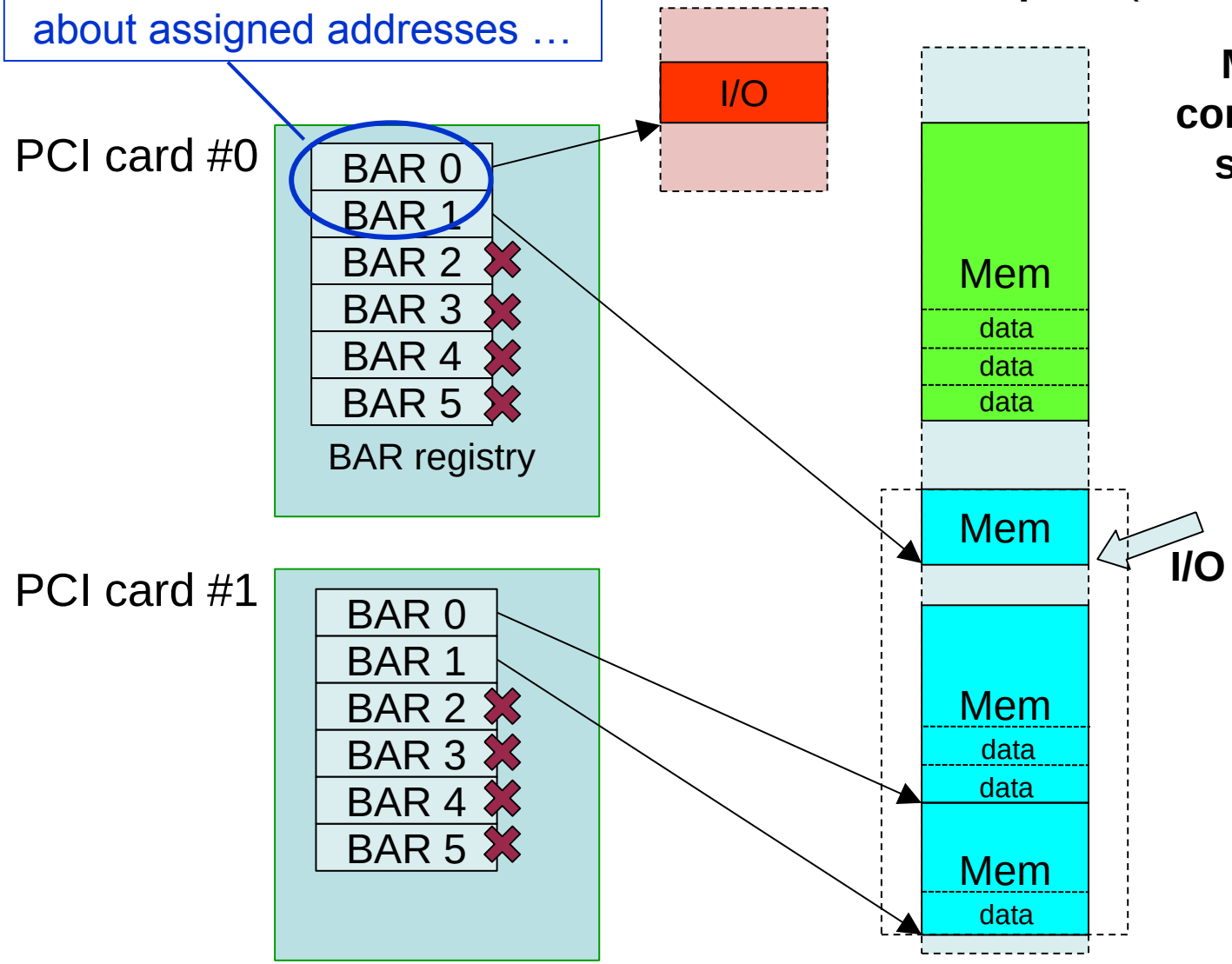


PCI Header


Byte Offset
Device ID
Vendor ID
00h
Status
Command
04h
Class Code
Revision ID
08h
BIST
Header Type
Master Lat. Timer
Cache Line Size
0Ch
Base Address Registers6 max
10h
14h
18h
1Ch
20h
24h
Cardbus CIS Pointer
28h
Subsystem ID
Subsystem vendor ID
2Ch
Expansion ROM Base Address
30h
Reserved
Capabilities Pointer
34h
Reserved
38h
Max_Lat
Min_Gnt
Interrupt Pin
Interrupt Line
3Ch

PCI Configuration Space - BAR = Base Address Registers

PCI device/card is informed about assigned addresses …



Memory space: common for I/O and system memory

Memory

If CPU writes to this location, write is recognized by PCI device/card #0. Its effect depends on card logic, i.e., for a graphic card frame-buffer, it behaves as regular memory, but data are seen on the screen.

PCI Configuration Space - BAR = Base Address Registers

PCI device/card is informed about assigned addresses …

PCI card #0

BAR 0
BAR 1
BAR 2
BAR 3
BAR 4
BAR 5

BAR registry

This is physical /bus address

PCI card #1

BAR 0
BAR 1
BAR 2
BAR 3
BAR 4
BAR 5

Study mmap() function manual.
Do not forget to munmap()…

I/O address space (x86 in, out instructions)

I/O

Memory space: common for I/O and system memory

Mem
data
data
data

Mem

mmap(BAR1)

I/O

Mem
data
data

Mem
data

base addr. +4
mmap(BAR0)

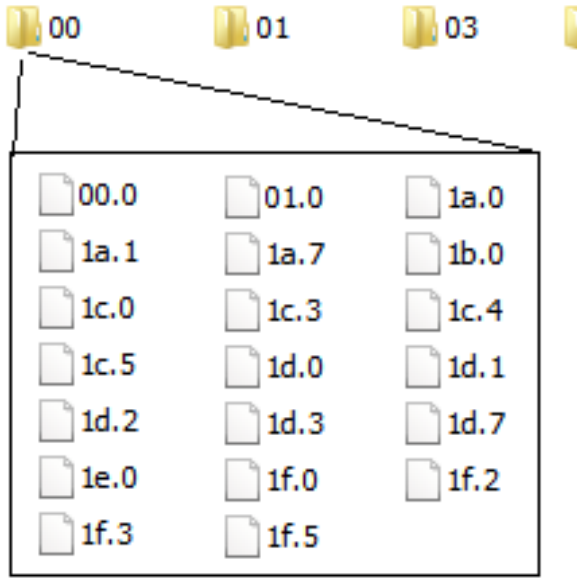
mmap(BAR1)

Memory

If CPU writes to this location, write is recognized by PCI device/card #0. Its effect depends on card logic…

CPU/code use virtual addresses and are translated by MMU !!!

Linux **/proc/bus/pci** directory




```
          00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  72 11 32 1f  07 00 10 00  01 00 00 ff  08 00 00 00  r.2.....'....
00000010  00 00 8f fe  00 00 00 00  00 00 00 00  00 00 00 00  ..Ź.....
00000020  00 00 00 00  00 00 00 00  00 00 00 00  72 11 32 1f  ....r.2.
00000030  00 00 00 00  50 00 00 00  00 00 00 00  0b 01 00 00  ....P.....
00000040  .. .. .. ..  .. .. .. ..  .. .. .. ..  .. .. .. ..  ....
00000050  .. .. .. ..  .. .. .. ..  .. .. .. ..  .. .. .. ..  ....
```

- Each directory represents one PCI bus (with its number assigned) and each file mirrors one PCI device function PCI header (the first 64 bytes)

Linux **/proc/bus/pci** directory – command **lspci -vb**

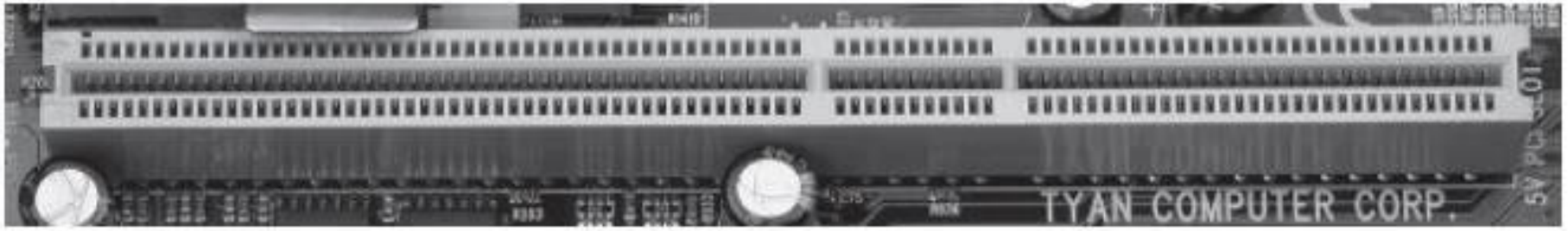
```
00:00.0 Host bridge: Intel Corporation 82X38/X48 Express DRAM Controller
        Subsystem: Hewlett-Packard Company Device 1308
        Flags: bus master, fast devsel, latency 0
        Capabilities: [e0] Vendor Specific Information <?>
        Kernel driver in use: x38_edac
        Kernel modules: x38_edac

00:01.0 PCI bridge: Intel Corporation 82X38/X48 Express Host-Primary PCI
Express Bridge
        Flags: bus master, fast devsel, latency 0
        Bus: primary=00, secondary=01, subordinate=01, sec-latency=0
        I/O behind bridge: 00001000-00001fff
        Memory behind bridge: f0000000-f2ffffff
        Kernel driver in use: pcieport
        Kernel modules: shpchp

00:1a.0 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI
Controller #4 (rev 02)
        Subsystem: Hewlett-Packard Company Device 1308
        Flags: bus master, medium devsel, latency 0, IRQ 5
        I/O ports at 2100
        Capabilities: [50] PCI Advanced Features
        Kernel driver in use: uhci_hcd
```

PCI-X

PCI-X 64 bit wide

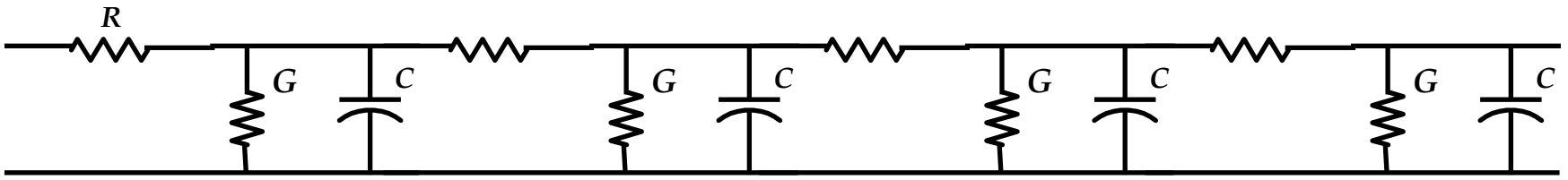


PCI 32 bit wide

* Serial Bus PCI-e

Peripheral Component Interconnect Express

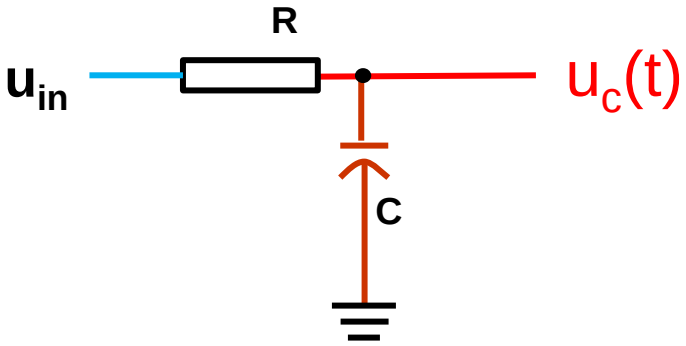
RC Delay in Wire

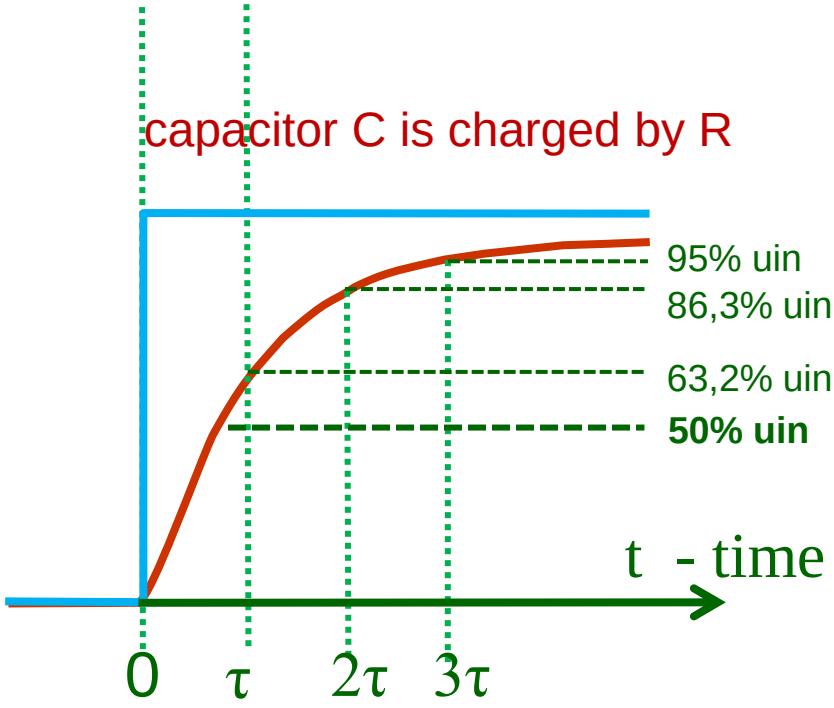


Wire: The length of one wire element → 0 and the number of elements → ∞

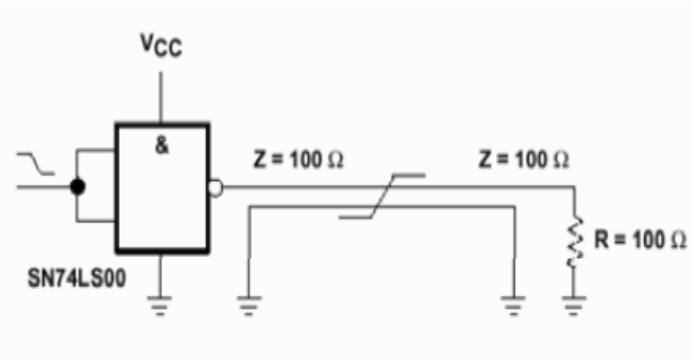
$u_c(t) = u_{in}(1 - e^{-t/\tau}),\ \tau = R.C$

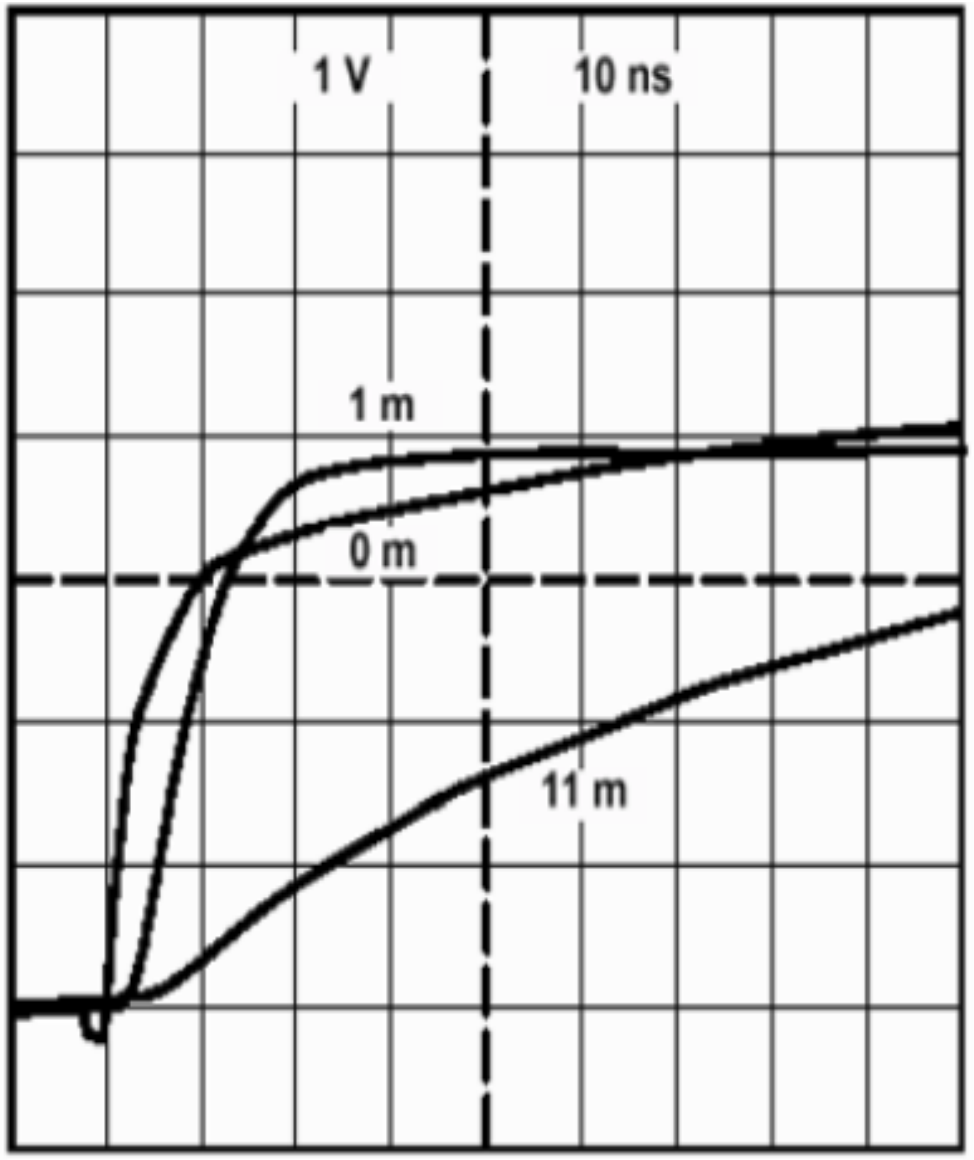
50 % = 0.69 RC





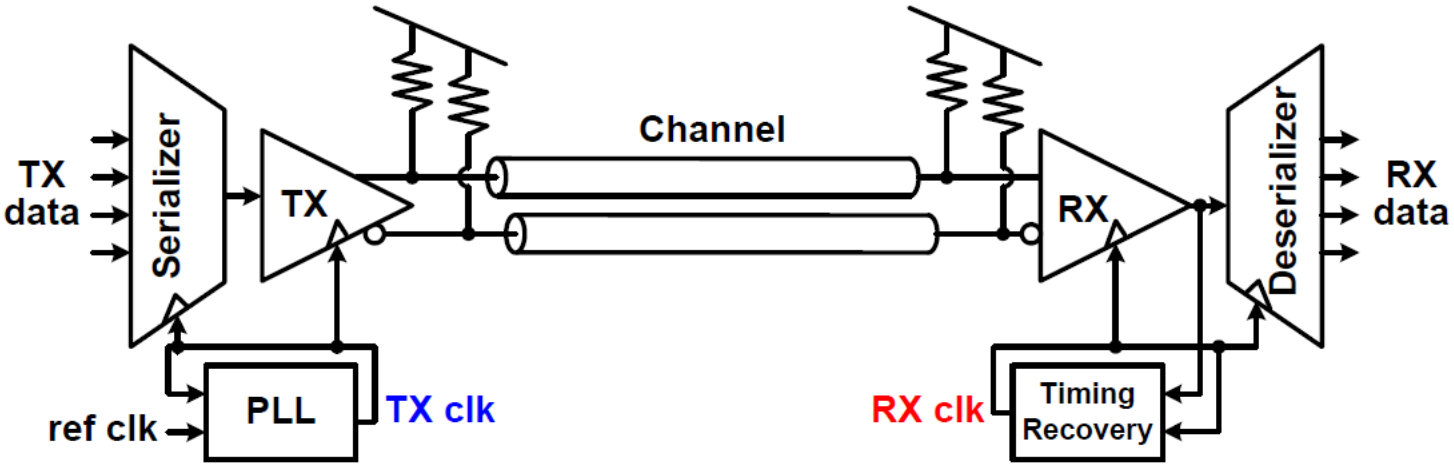
Deformation of Signals



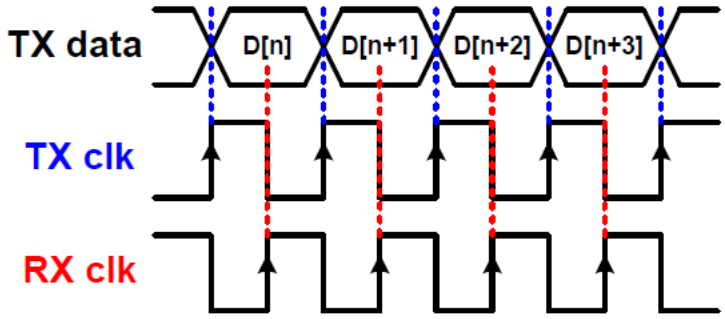


Source:

High Speed Serial Link



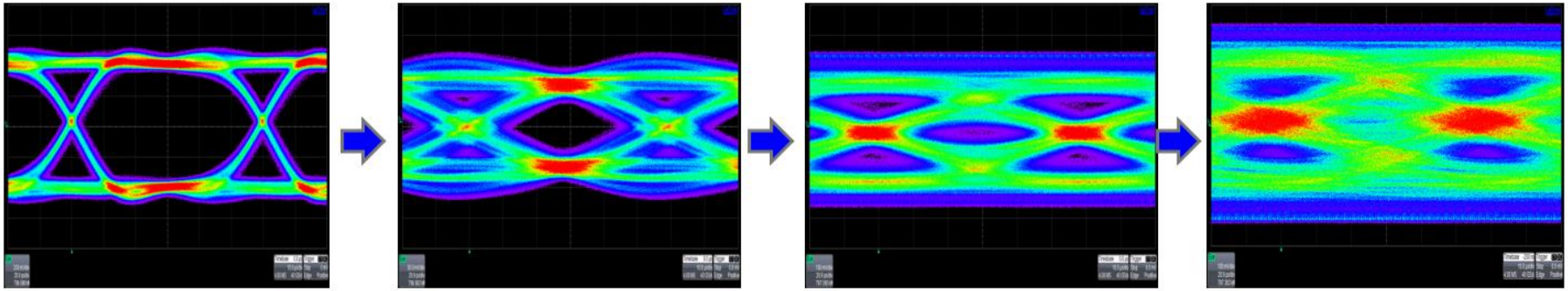
PLL
= Phase-locked Loop
(cz: fázový závěs)
generates frequency with phase related to input reference signal ref clk.



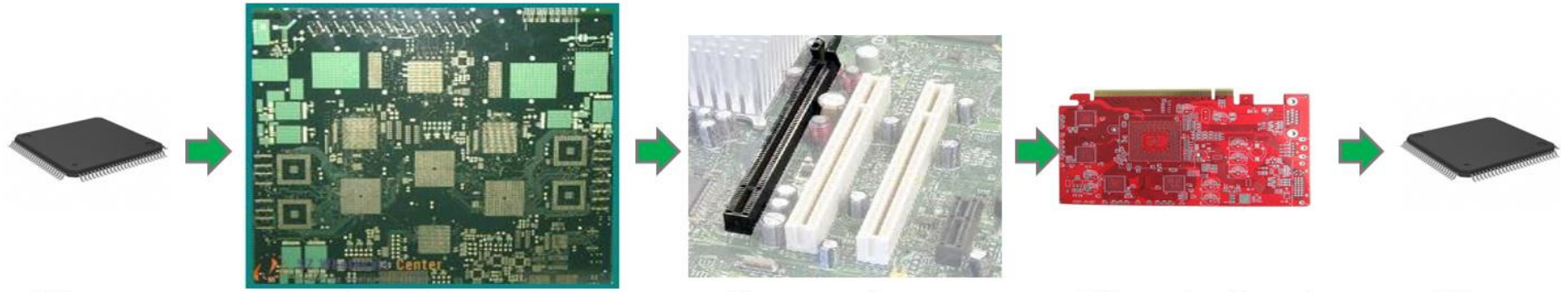
Only ideal theoretical signals:-)

Source: S. Palermo: **High-Speed Serial I/O Design for Channel-Limited and Power-Constrained Systems ,** Texas A&M University 2010

High Speed Serial Link Reality



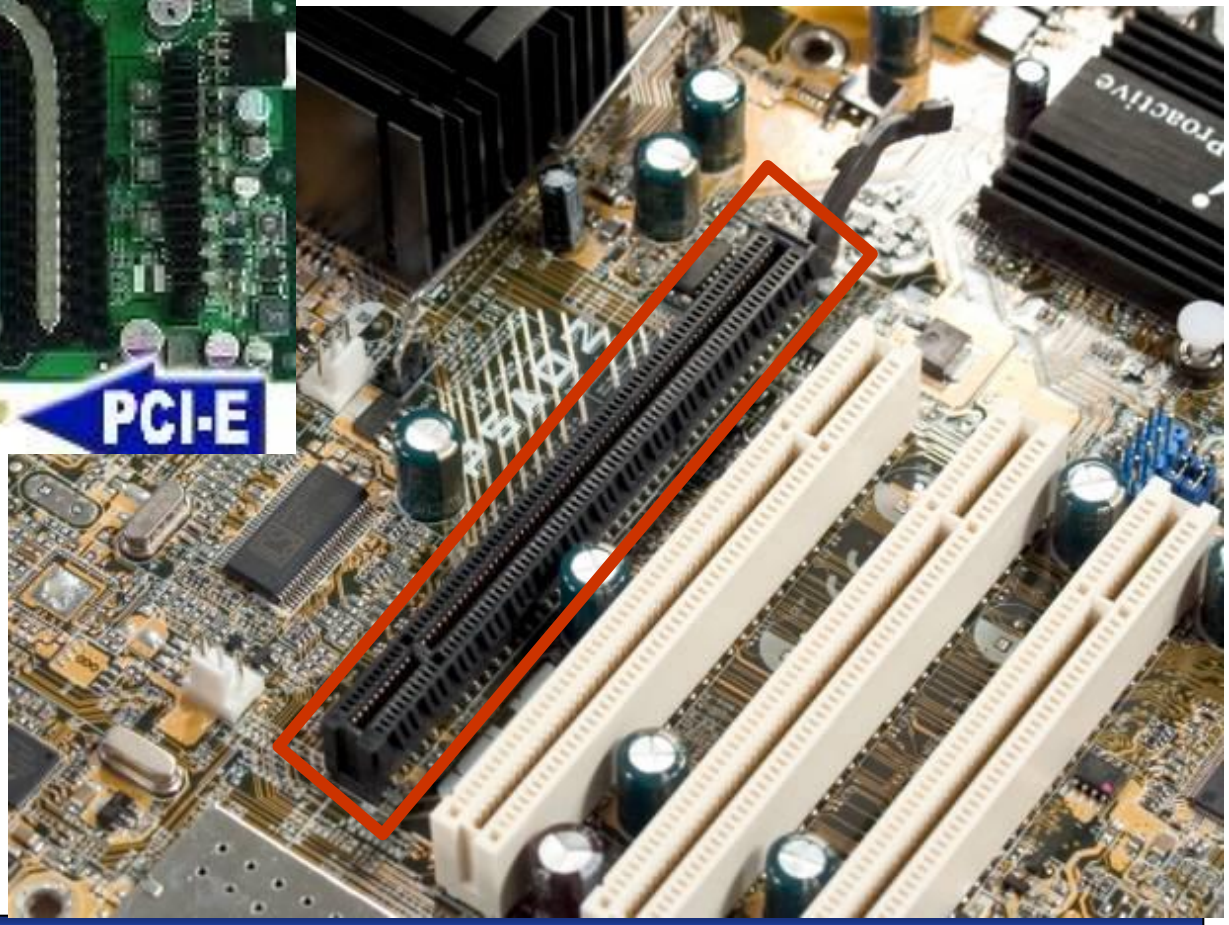
Signal degrades over long transmission path and connectors



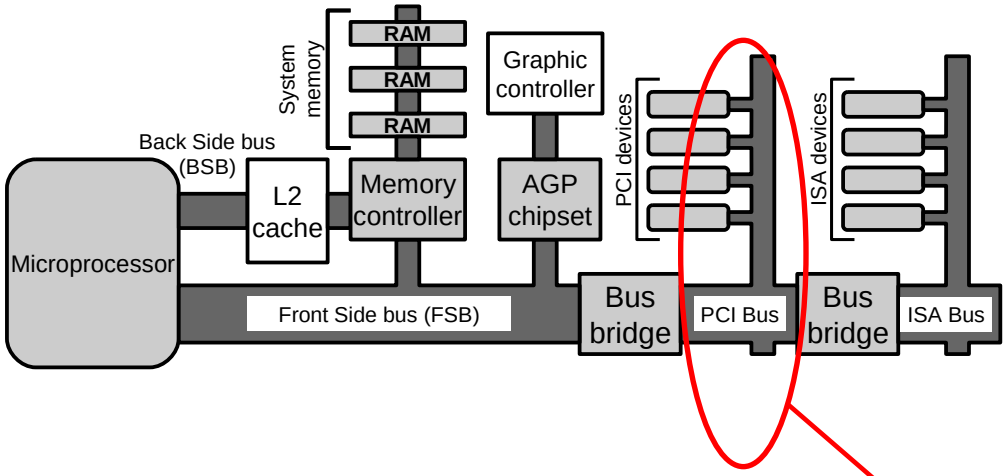
IC System Board Connector Plug-In Card IC

Source: TELEDYNE LECROY, 2018

PCI Express PCI-E

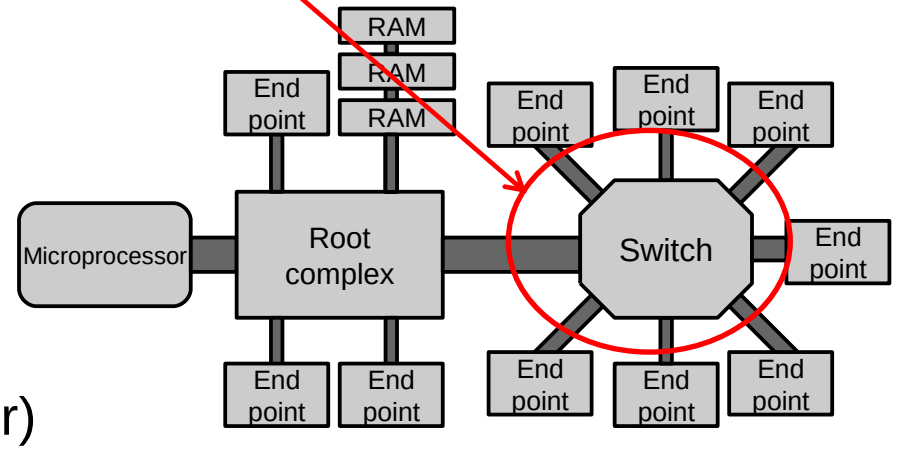


PCIe architecture - bus is replaced by shared switch



Control space access is routed by IDSEL

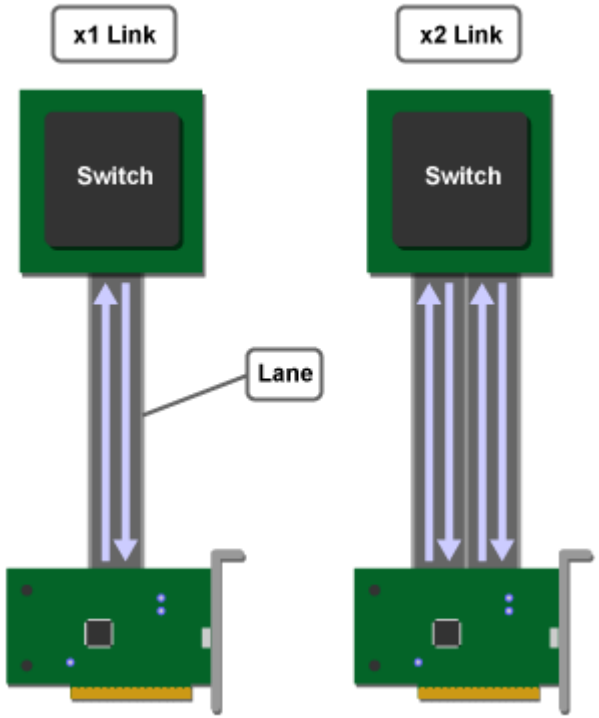
PCIe = PCI Express



Control space access is routed by switch (port number)

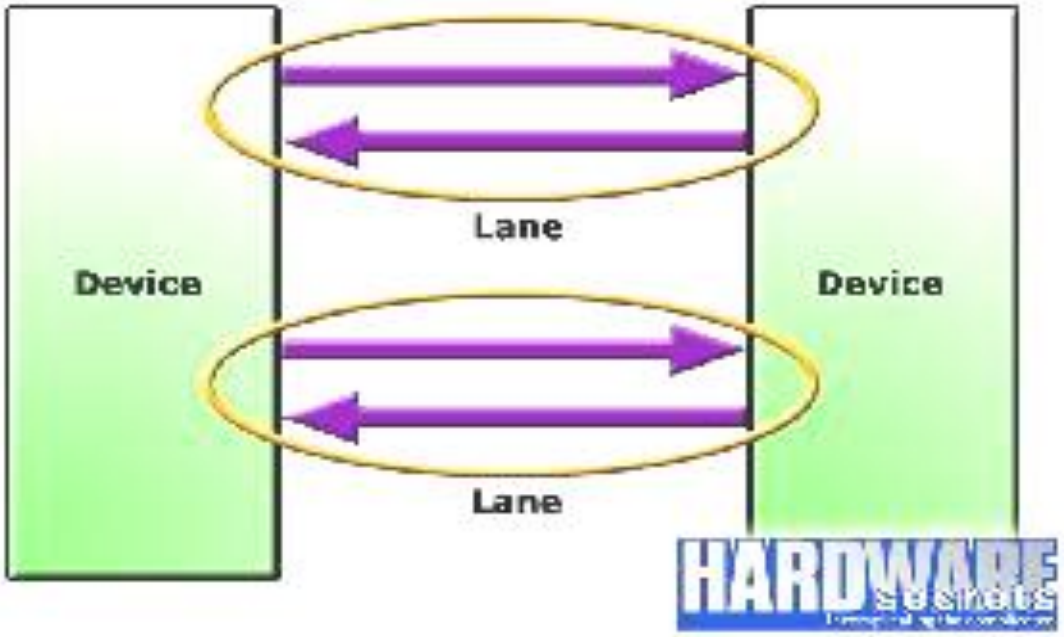
PCIe transfers signaling, PCIe lanes

- Link interconnect switch with exactly one device
- Differential AC signaling is used – two wires for single direction
- Each link consists of one or more lanes
- **The lane consists of two pairs of wires** - one pair for Tx and other one for Rx
- Data are serialized by 8/10 code
- The separate pairs allow **full-duplex** operation/transfers



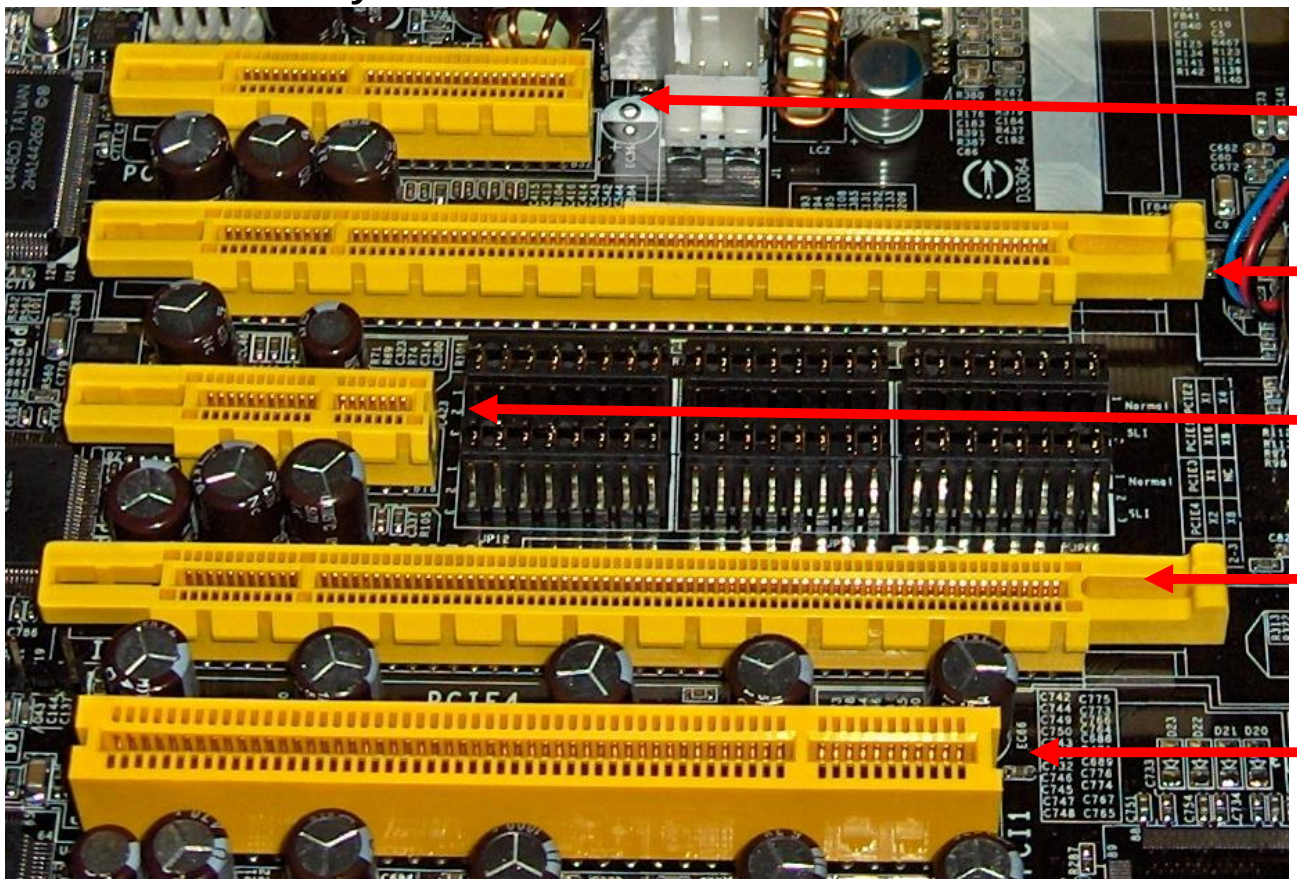
PCI Express x2 bus

- PCI-E with two lanes.
- Highly scalable because bandwidth can be increased by increasing the number of lanes.



PCI-E Slots

The PCI Express bus defines a different types of slot based on the number of lanes in the system.



- PCIe x4
- PCIe x16
- PCIe x1
- PCIe x16
- 32-bit classic PCI slot

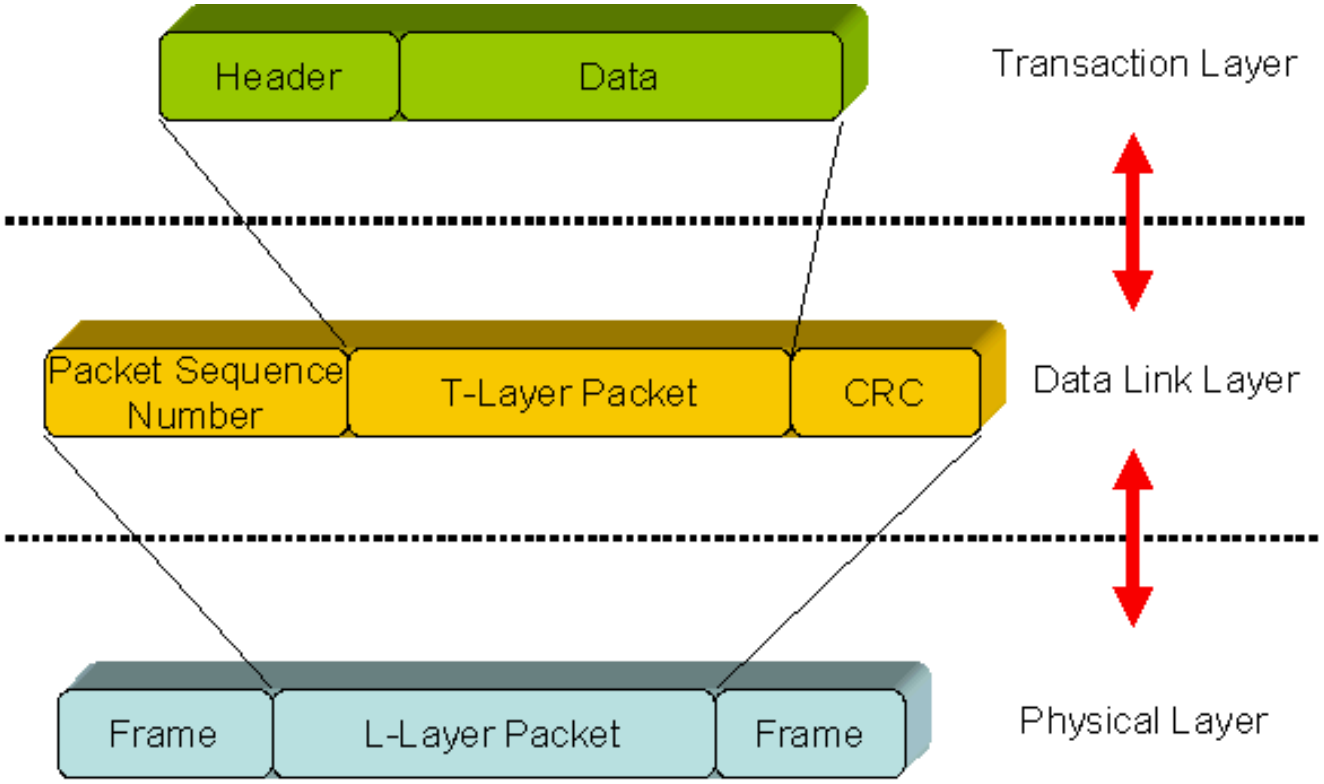
- Picture source: Wikipedia

LanParty nF4 Ultra-D mainboard from DFI

Full and Half Duplex

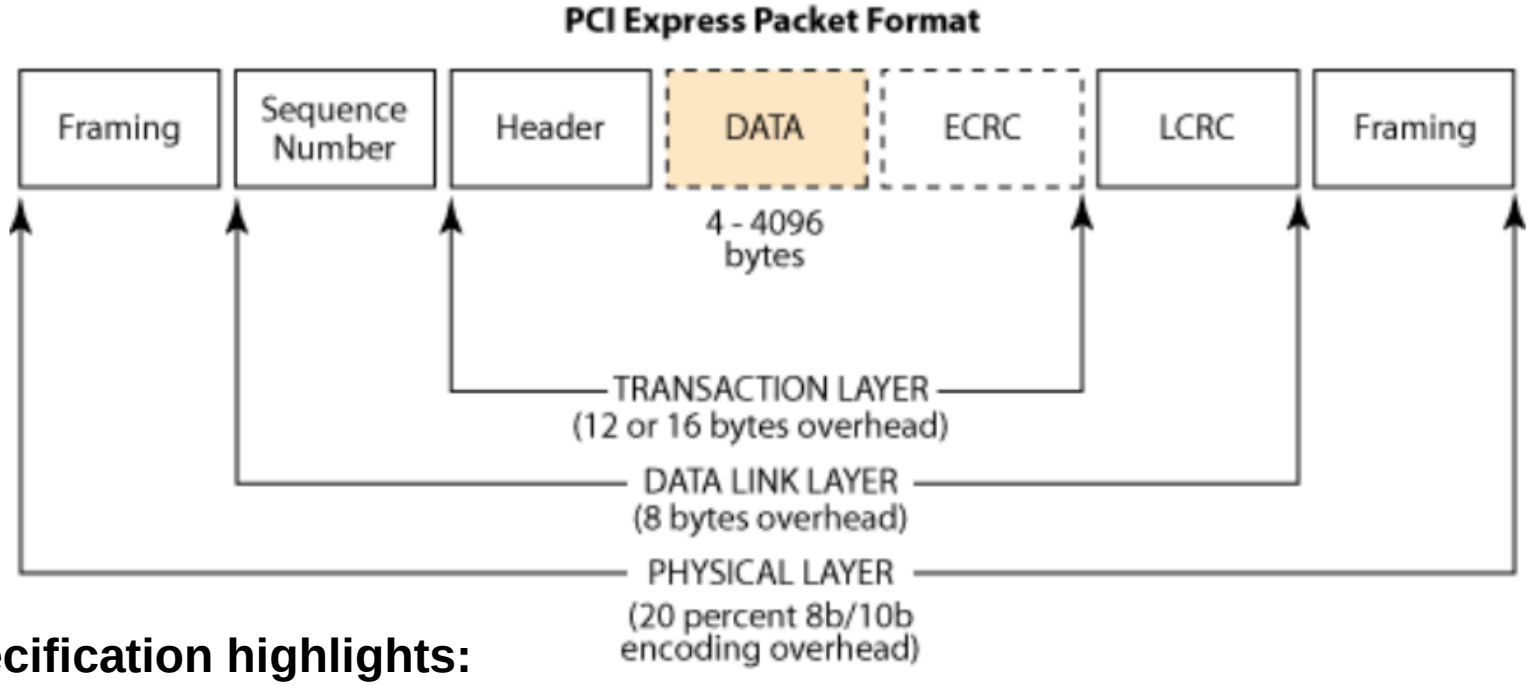
- A half-duplex system allows communicating in both directions, but only one direction at a time (not simultaneously).
- Full duplex means communicating in two directions simultaneously at once.
- Example:
  - PCI Express can use 1,2,4,8,12,16 and 32 lanes to achieve max. bandwidth of 160 Gbps
  - Max bandwidth
    = (32 lanes X 2.5GB) X 2 (*Full duplex*)
    = 160 GBPS.

Data packet structure and transport layers



- Source: http://zone.ni.com/devzone/cda/tut/p/id/3767#toc0

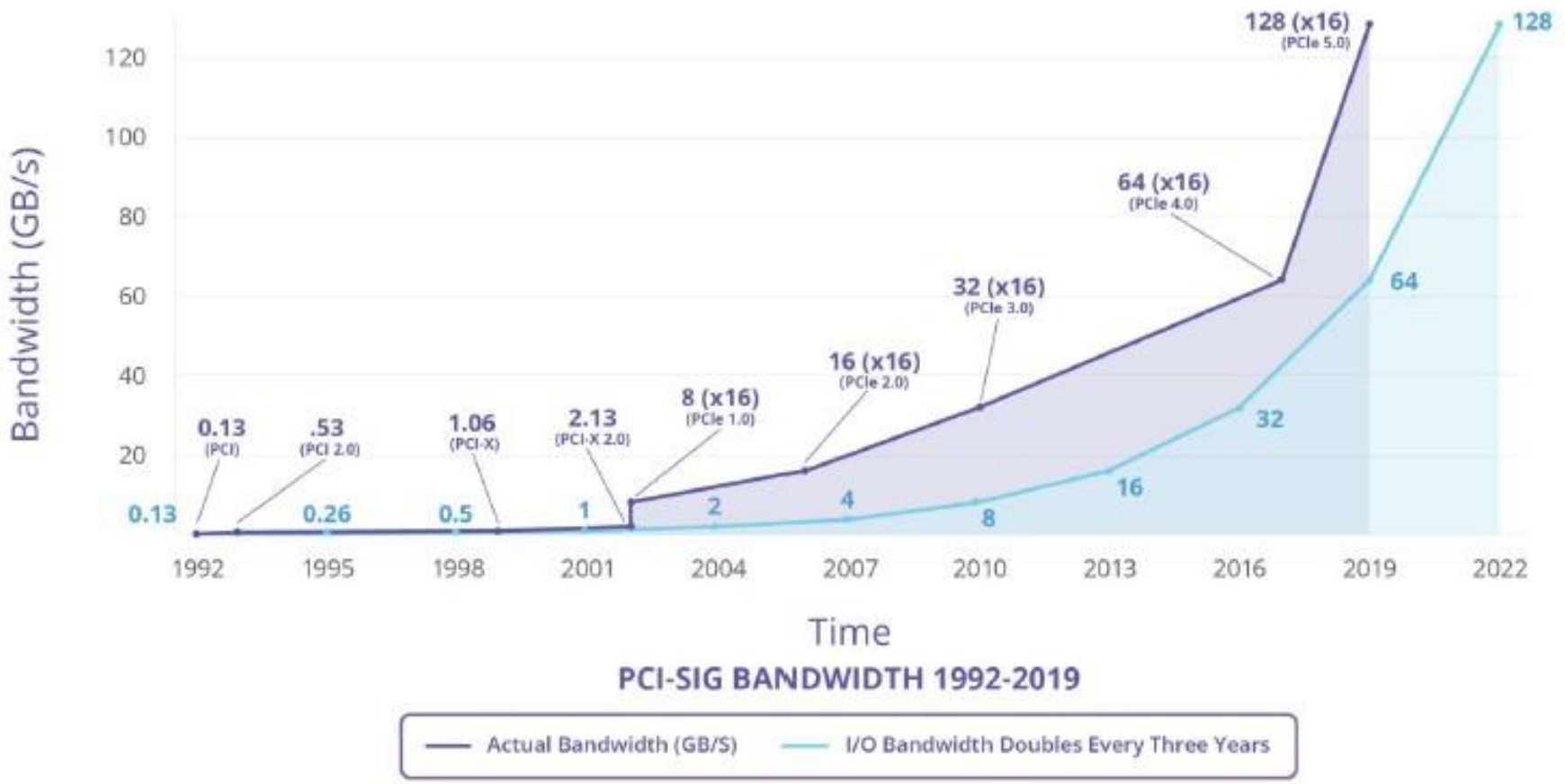
PCIe packet format



Specification highlights:

- Packet length from 4B to 4096B
- PCIe packed has to be exchanged as the whole (no option to preempt when higher priority transfer is requested)
- Long packed can cause increase of latencies in the system
- On the other hand, short packets overhead (frame/data) is considerable

PCI(e) I/O bandwidth expectations and reality



PCIe buses overcome theoretical expectations

Source: Nextpatform

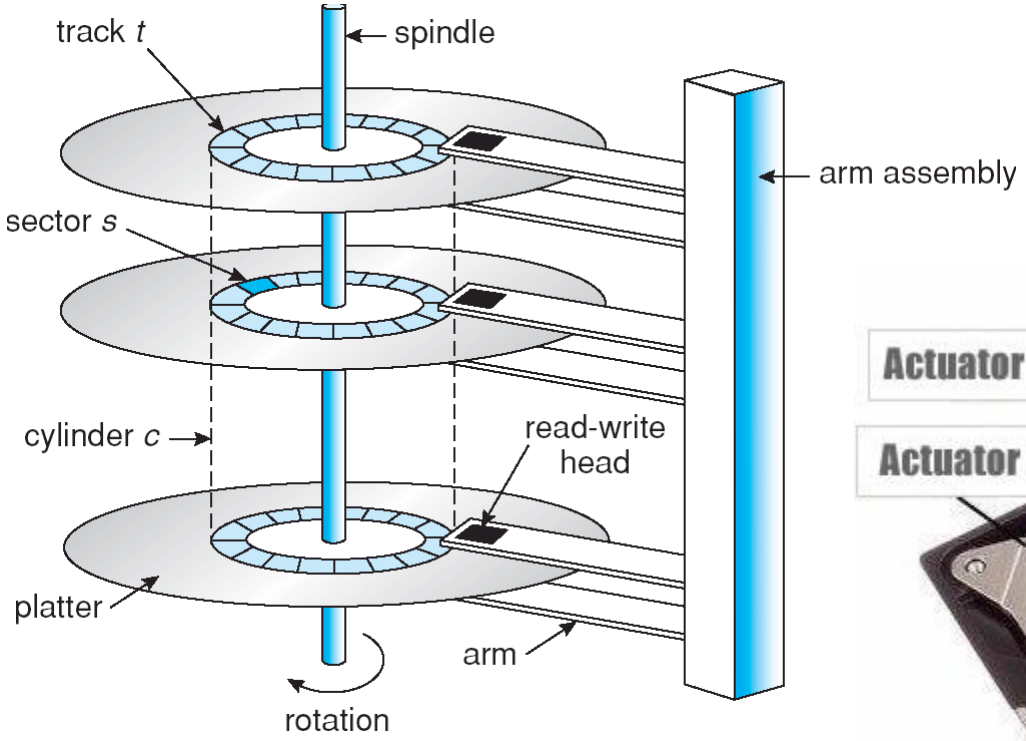
PCI-signal Bandwidth in Numbers

Year	Bandwidth	Frequency/Speed
1992	133MB/s (32 bit simplex)	33 Mhz (PCI)
1993	533MB/s (64 bit simplex)	66 Mhz (PCI 2.0)
1999	1.06GB/s (64 bit simplex)	133 Mhz (PCI-X)
2002	2.13GB/s (64 bit simplex)	266 Mhz (PCI-X 2.0)
2002	8GB/s (x16 duplex)	2.5 GHz (PCIe 1.x)
2006	16GB/s (x16 duplex)	5.0 GHz (PCIe 2.x)
2010	32GB/s (x16 duplex)	8.0 GHz (PCIe 3.x)
2017	64GB/s (x16 duplex)	16.0 GHz (PCIe 4.0)
2019	128GB/s (x16 duplex)	32.0 GHz (PCIe 5.0)

Source: Nextpatform

* Důležitou periferií je datové úložiště, a tak ho musíme zrychlit…

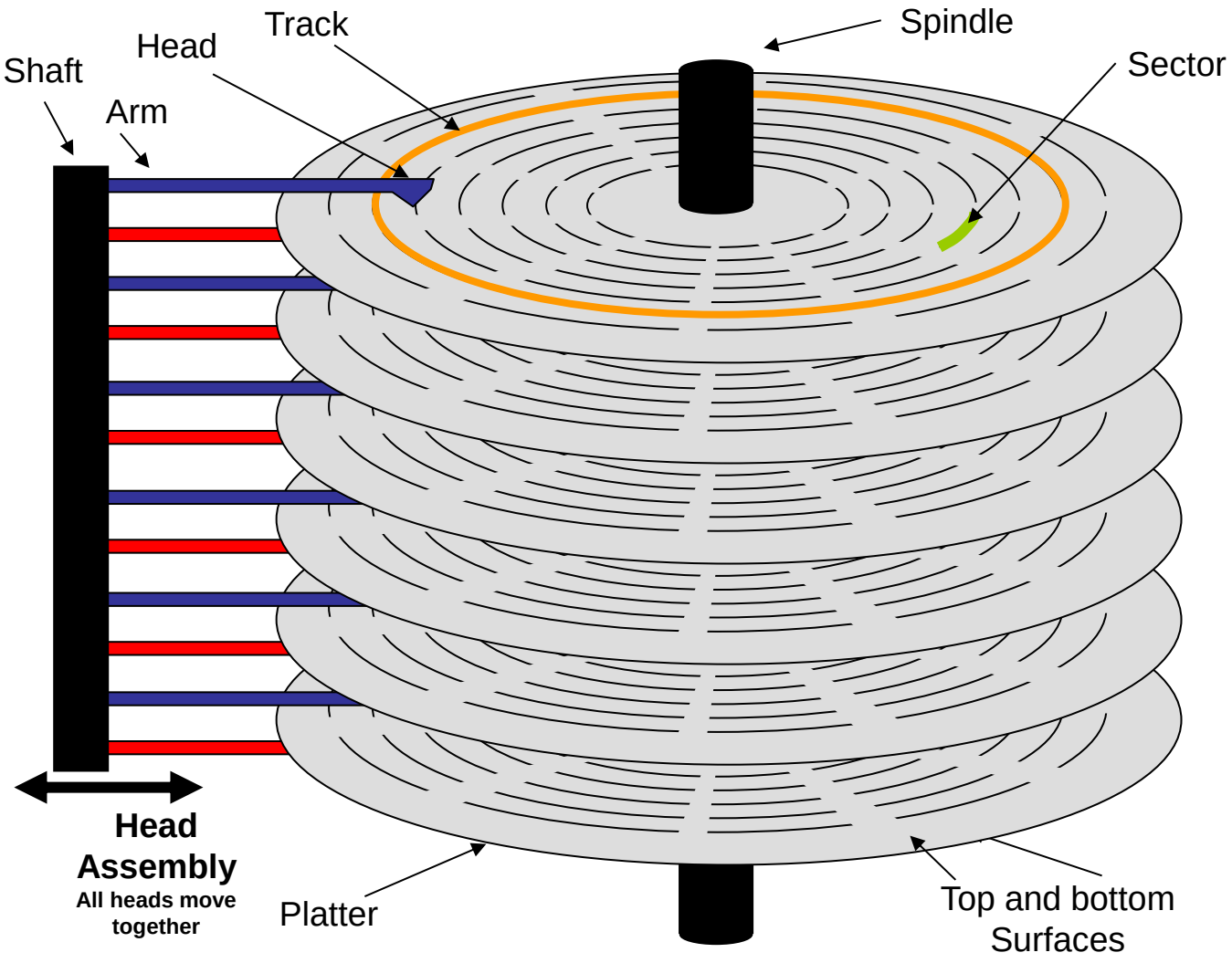
Hard Drive



Drawing source: Junfeng Yang

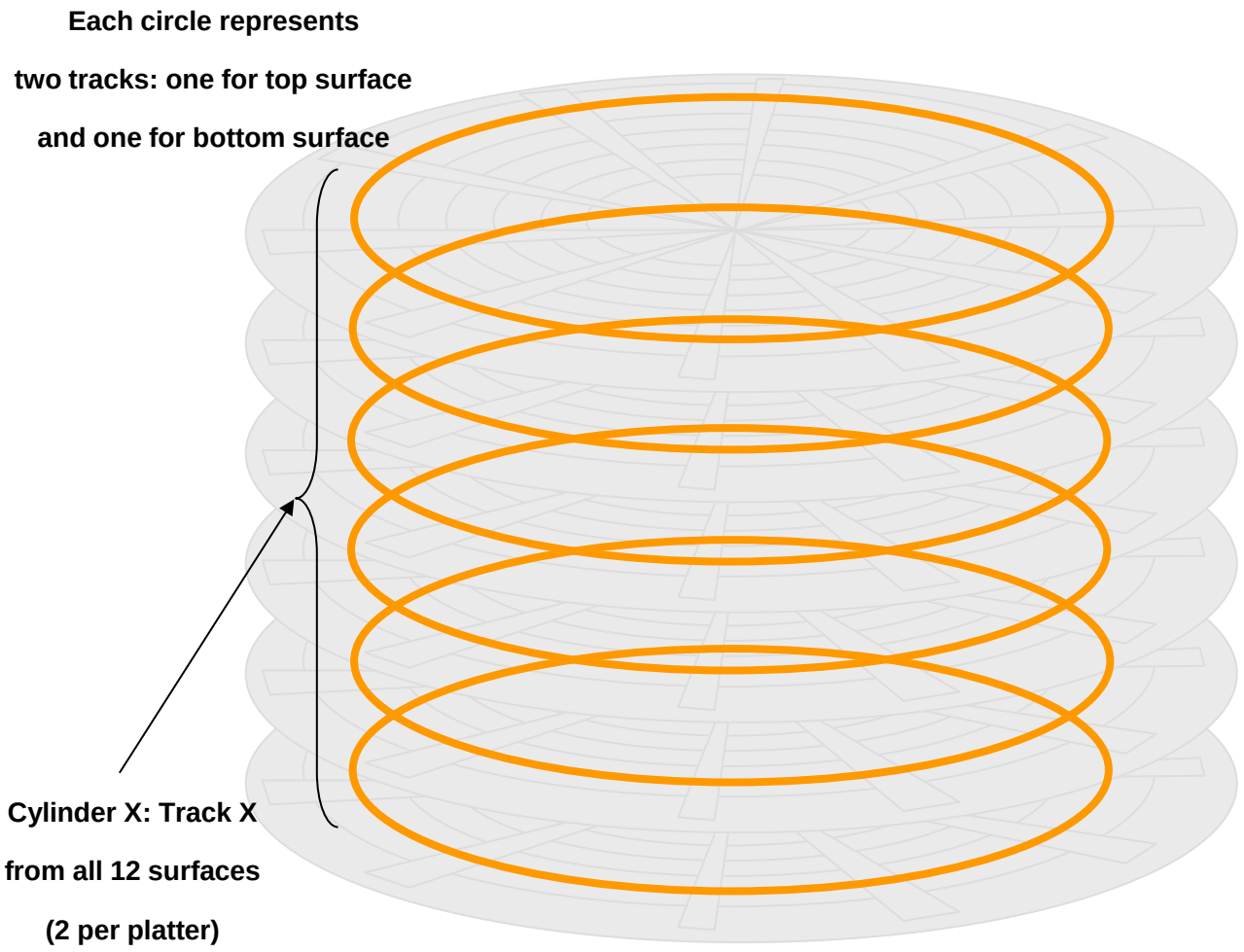
Picture source: https://www.hddzone.com/hard_disk_drive_compontents.html

Disk Storage



Source: U. Ramachandran and W. D. Leahy Jr.: An Integrated Approach to Architecture and Operating Systems , 2008

Disk Storage



Source: U. Ramachandran and W. D. Leahy Jr.: An Integrated Approach to Architecture and Operating Systems , 2008

Sequential v.s. Random

A sequential access

- Seek to the right track
- Rotate to the right sector
- Transfer

A random access of the same amount of data

- Seek to the right track
- Rotate to the right sector
- Transfer
- Repeat

Since seek and rotate are slow, sequential access is much faster than random access

Examples of HD Parameters

	Barracuda 180	**Cheetah X15 36LP**
Capacity	181GB	36.7GB
Disk/Heads	12/24	4/8
Cylinders	24247	18479
Sectors/track	~609	~485
Speed	**7200 RPM**	**15000 RPM**
Rotational latency (ms)	4.17	2.0
Avg seek (ms)	7.4	3.6
Track-2-track(ms)	0.8	0.3

Table source: Junfeng Yang

Maximum RPM

Sonar boom - source: U.S. Navy/Travis K. Mendoza

Rychlosti zvuku se na obvodu HD dosáhne při RPM

Průměr	RPM
3.5 palce	73105
2.5 palce	102347

Ale RPM zvyšuje i energii nutnou k roztočení a odstředivou sílu

IOPS

IOPS stands for **input/output operations per second**

IOPS=1 / (Average Seek Time + Average Latency)

Pameter	HD	SSD
IOPS	55-180	3000-40000
IOPS/Watt	10-30	2000-60000
$/GB	$0.02 - $0.04	$0.25-$0.50
Data retention if unplugged	10 - 60 years	1 - ?100? years / 25 °C *SSD are used for short time, reliable data are unknown yet*

IOPS

Google test of SSD:

In summary, we find that the flash drives in our study experience significantly lower replacement rates (within their rated lifetime) than hard disk drives.

On the downside, they experience significantly higher rates of noncorrectable errors than hard disk drives.

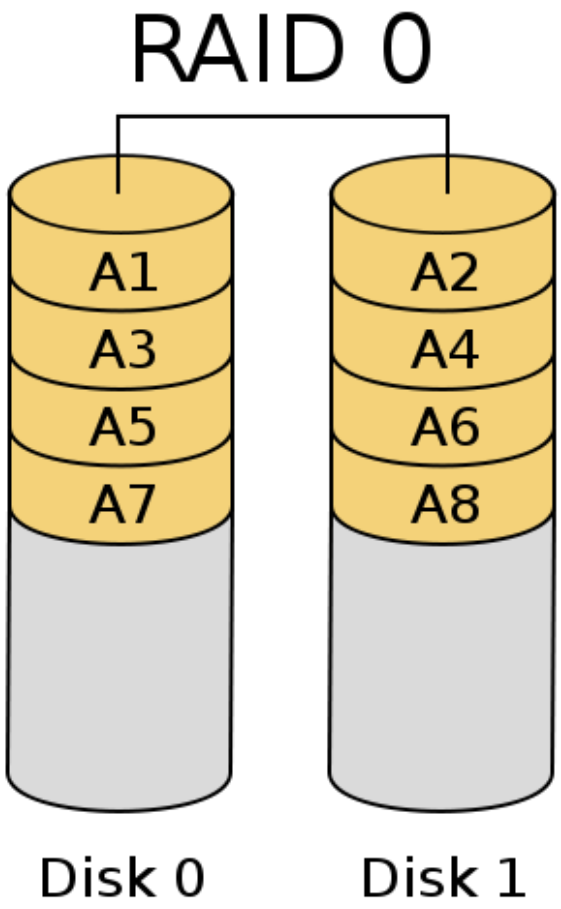
[Result of 4-year test in Toronto data center]

Flash Reliability in Production: The Expected and the Unexpected
Bianca Schroeder, *University of Toronto;* Raghav Lagisetty and Arif Merchant, *Google Inc.*

An analysis of latent sector errors in disk drives. BAIRAVASUNDARAM, L. N., GOODSON, G. R., PASUPATHY, S., AND SCHINDLER, JIn *Proceedings of the 2007 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems* (New York, NY, USA, 2007), SIGMETRICS '07, ACM, pp. 289–300.

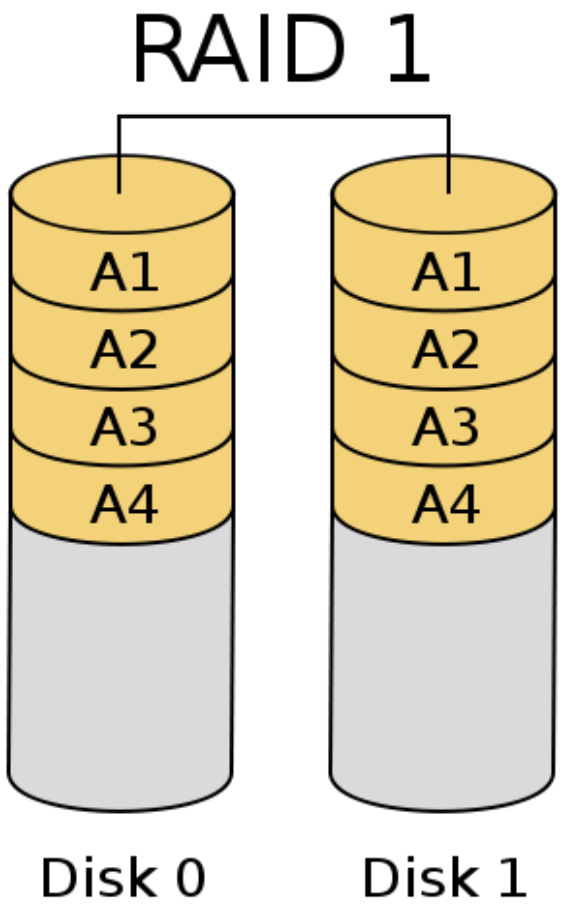
RAID 0

- Pro zvýšení výkonu systému pevných disků.
- tzv. “stripping” (proužkování)
- A1,A2,..., A7, A8 označuje bloky soubory
- Zdvojnásobí rychlost
- Havárie 1 disku -> ztráta všech dat



RAID 1

- Pro zvýšení spolehlivosti uložených dat.
- Označuje se jako “Mirroring”.
- Nezrychluje, ale zvyšuje spolehlivost.

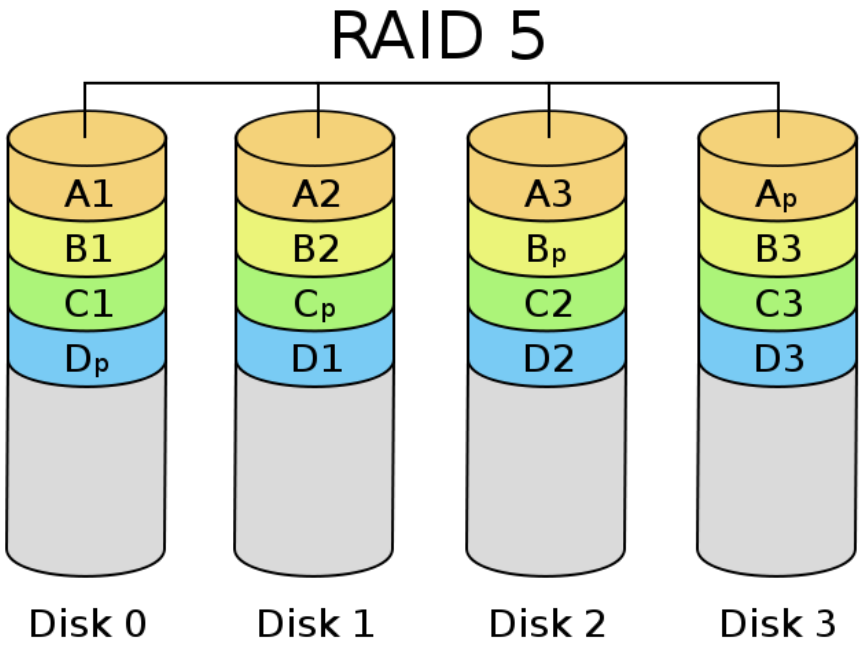


RAID 10

- Kombinace obou výše popsaných.
- Vytvoří se RAID 0 a ten se pak zrcadlí na RAID 1. Výsledkem jsou vlastně dva RAID 0 obsahující identická data.
- RAID 10 zvyšuje jak výkon, tak spolehlivost, musíte ale použít nejméně čtyři disky, a nejlépe se stejnými parametry.

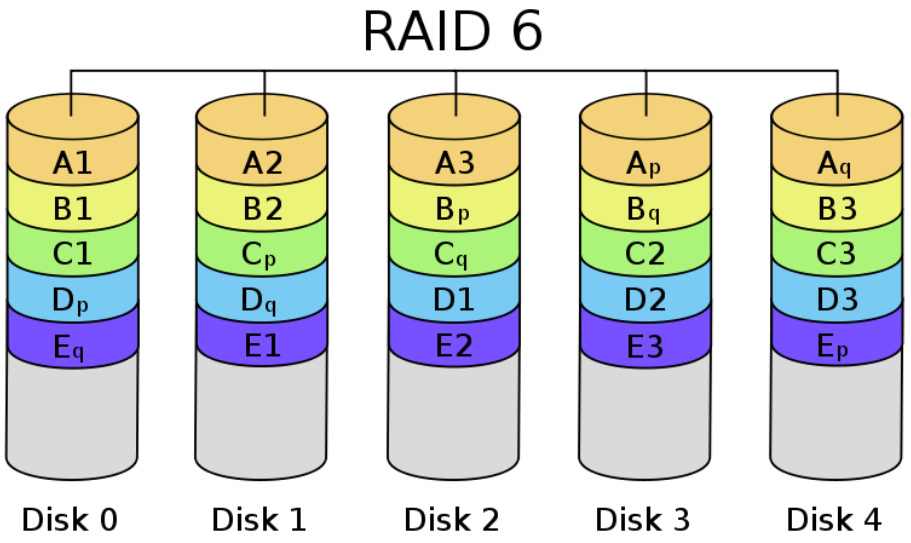
RAID 5

- Ukládá paritní informace, nikoli však na jeden vyhrazený disk.
- V degradovaném režimu se musí data uložená na vadném disku odvodit z dat zbývajících disků a parity.
- Zrychluje čtení, ale zpomaluje zápis.



RAID 6

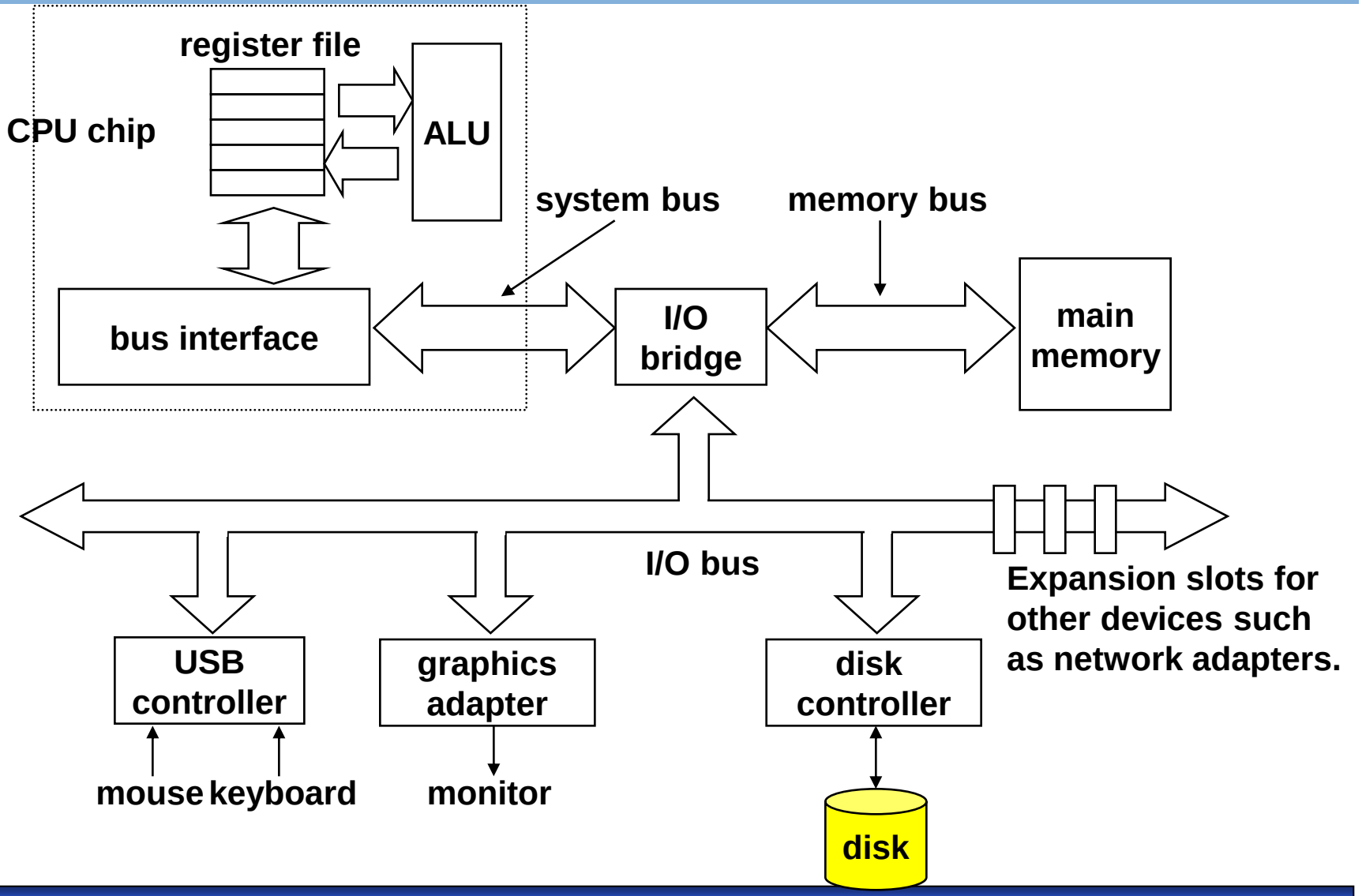
- Obdoba RAID 5, používá dva paritní disky s různě vypočtenou paritou.
- Odolný proti výpadkům 2 disků.
- Zrychlí čtení jako RAID 5, ale zápis je ještě pomalejší.



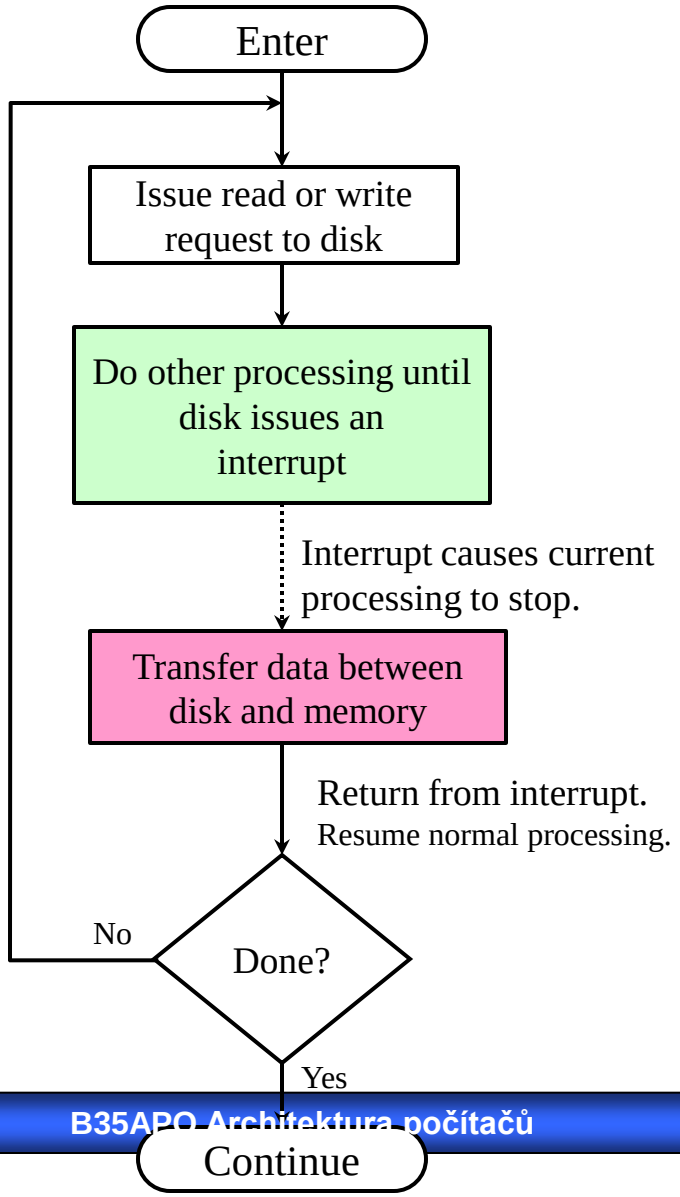
* DMA

Direct Memory Access

A Typical Hardware System

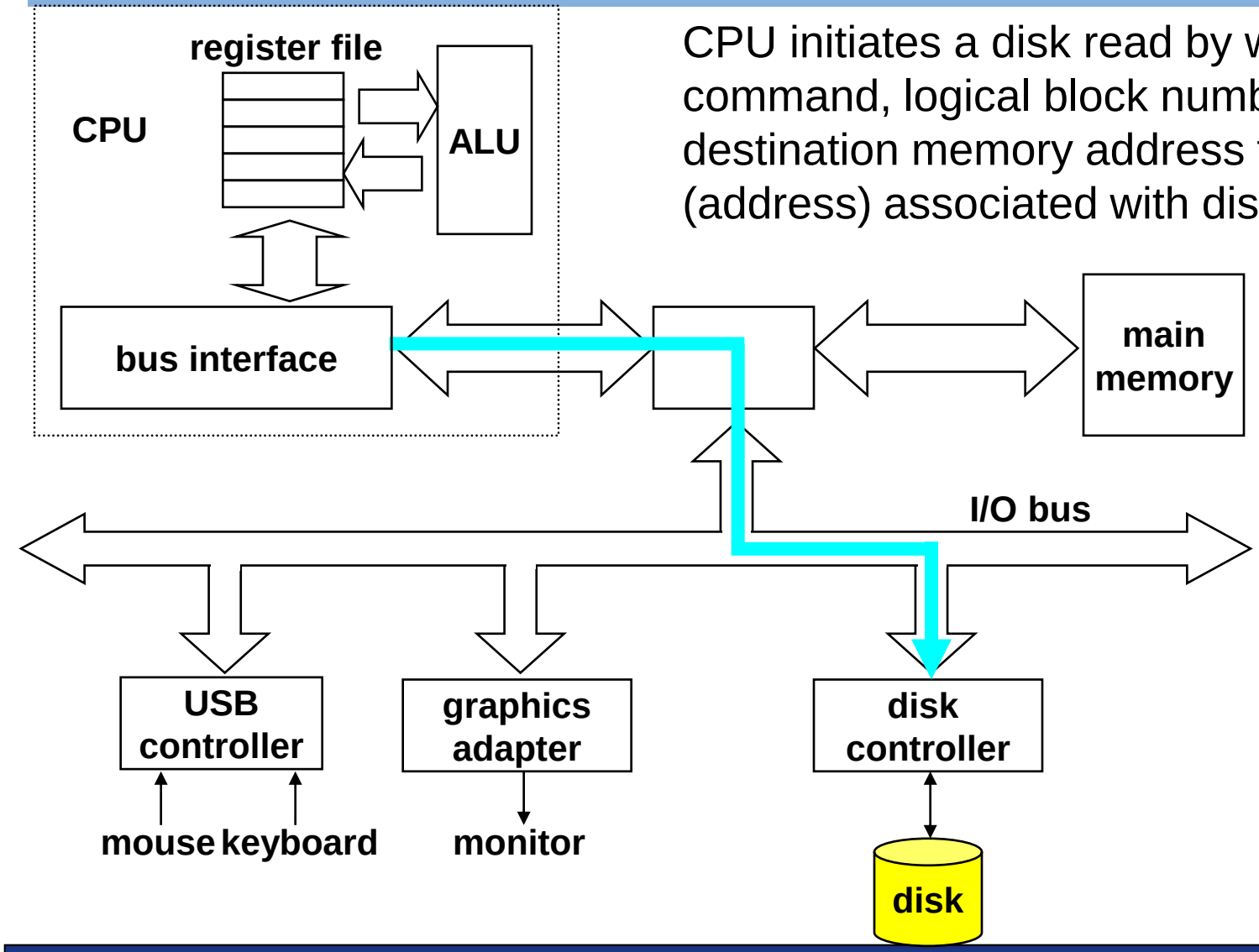


Slow Interrupt Driven Disk Transfer



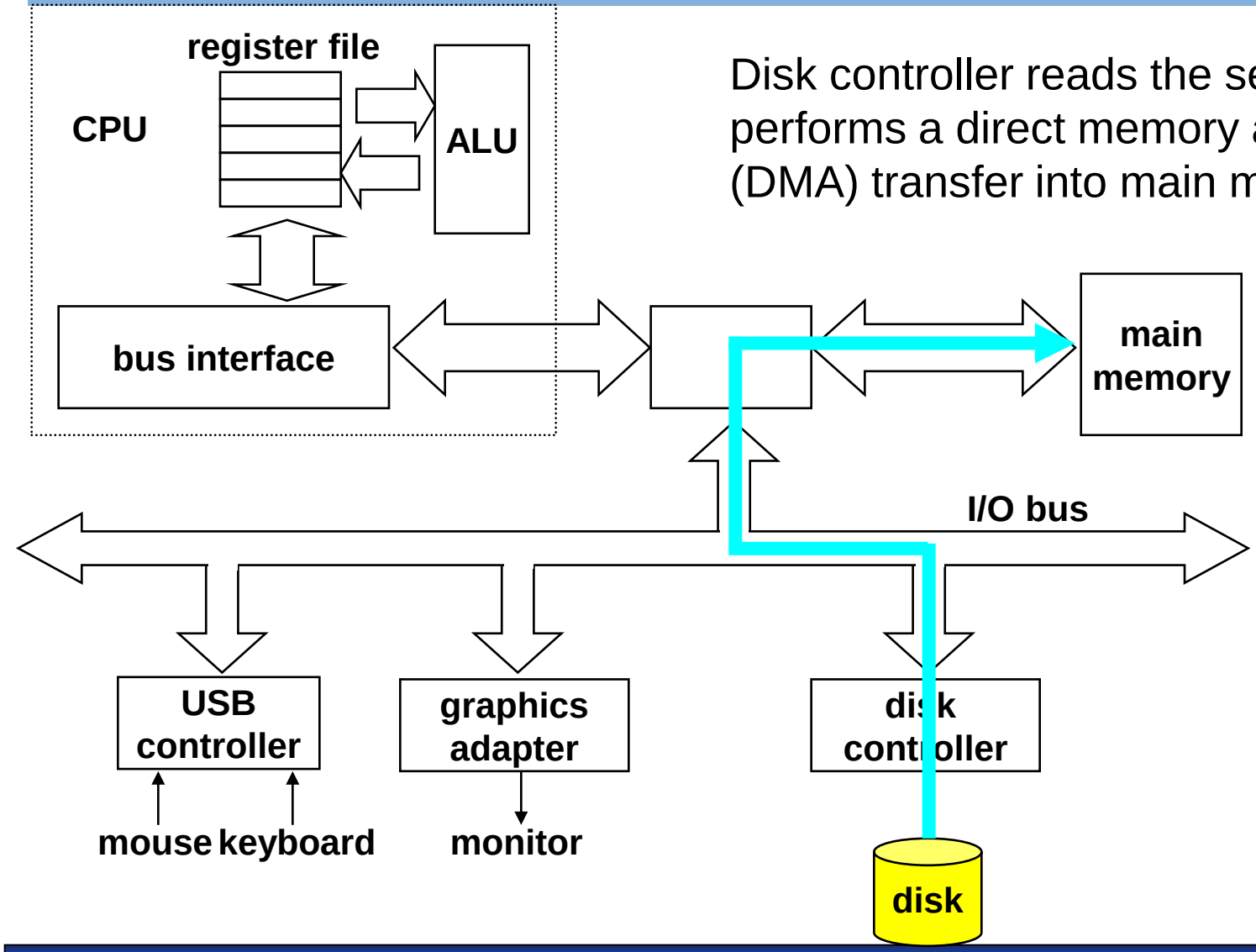
1. CPU issues read command
2. I/O module gets data from peripheral while CPU performs other work
3. I/O module interrupts CPU
4. CPU requests data
5. I/O module transfers data

DMA Reading a Disk Sector: Step 1



CPU initiates a disk read by writing a command, logical block number, and destination memory address to a port (address) associated with disk controller.

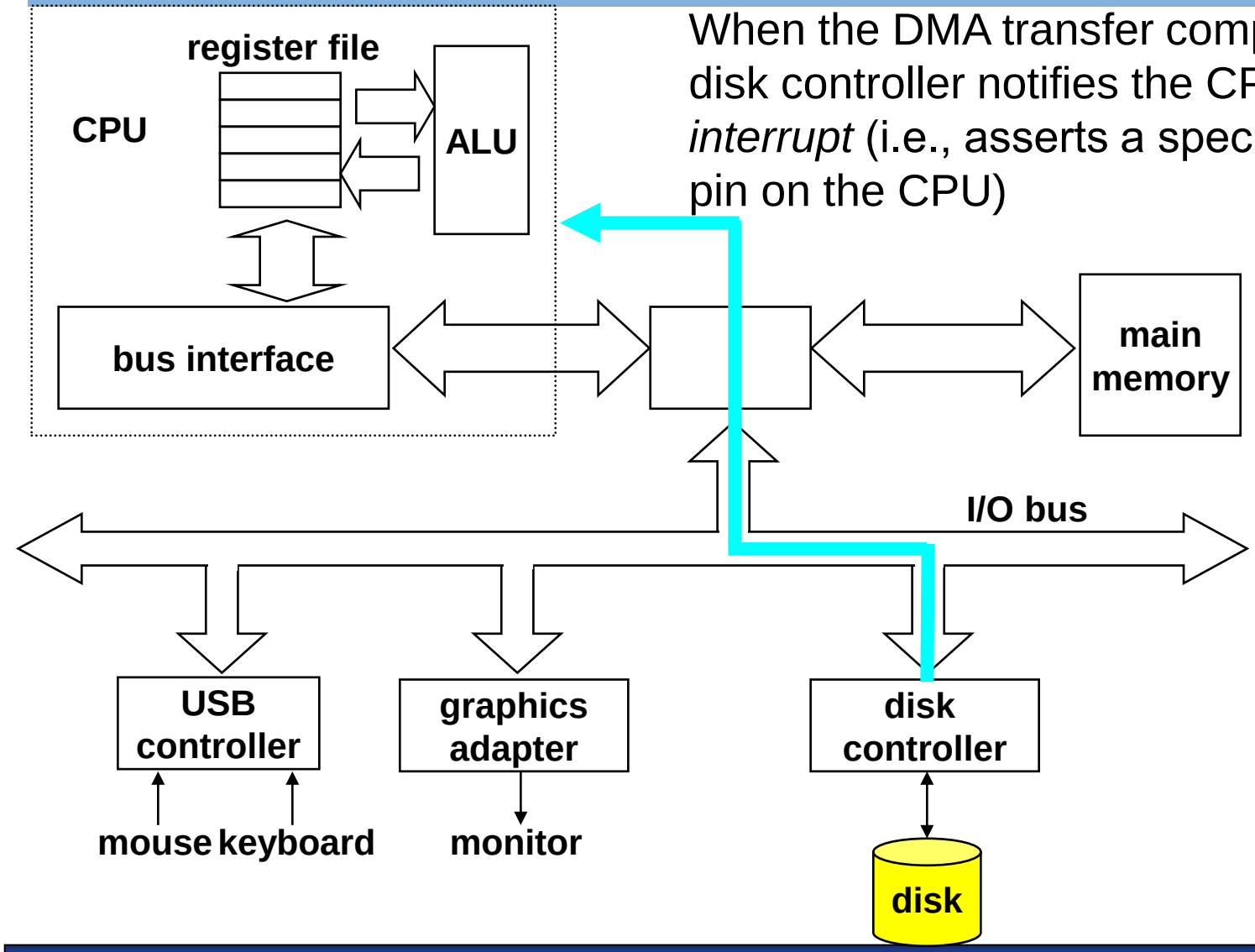
DMA Reading a Disk Sector: Step 2



Disk controller reads the sector and performs a direct memory access (DMA) transfer into main memory.

Reading a Disk Sector: Step 3

When the DMA transfer completes, the disk controller notifies the CPU with an *interrupt* (i.e., asserts a special “interrupt” pin on the CPU)



Direct Memory Access (DMA)

Direct Memory Access (DMA)

1. Device wishing to perform DMA asserts the processors bus request signal.
2. Processor completes the current bus cycle and then asserts the bus grant signal to the device.
3. The device then asserts the bus grant ack signal.
4. The processor senses in the change in the state of bus grant ack signal and starts listening to the data and address bus for DMA activity.
5. The DMA device performs the transfer from the source to destination address.
6. During these transfers, the processor monitors the addresses on the bus and checks if any location modified during DMA operations is cached in the processor. If the processor detects a cached address on the bus, it can take one of the two actions:
   - Processor invalidates the internal cache entry for the address involved in DMA write operation
   - Processor updates the internal cache when a DMA write is detected
7. Once the DMA operations have been completed, the device releases the bus by asserting the bus release signal.
8. Processor acknowledges the bus release and resumes its bus cycles from the point it left off.

Source: http://www.eventhelix.com/