Estructura de Computadores (ETC)

# Unit 10
# Interconnection busses

Course 2018/2019

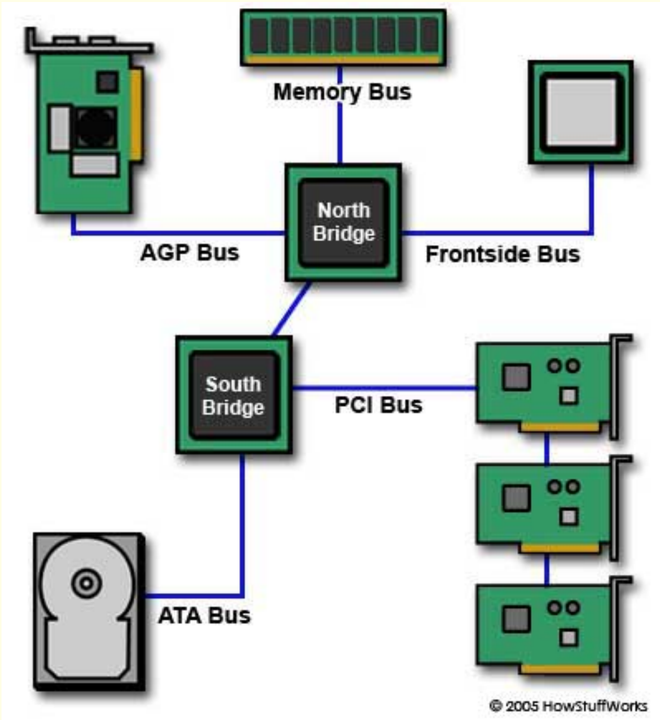
# Contents

- Busses
  - The bus concept
  - Technological aspects
  - Topologies
  - Interconnecting busses
  - Bus hierarchy
- Current busses
  - Trends
  - PCI and PCIe
  - SATA
  - USB and Firewire
  - Today's bus hierarchy
- Transfers within the computer
  - The roles of the bus controller and OS
  - Examples of timing

# Busses

# The need for interconnection

- CPU, memories and I/O devices have varying bandwidth requirements
- They are interconnected with different types of busses




Source: http://computer.howstuffworks.com/

# The bus concept

- A bus is a means for communicating two or more devices, enabling:
  - Addressing: selecting one of the connected devices and the addressable elements within them
  - Synchronization: a means of signalling that a device is ready to operate
  - Transfer: the effective transmission of data among devices
- Other, optional functions are:
  - Power supply
  - Hot plug capability …
- **Bus cycle:** time interval taken to complete an elementary data transfer between interconnected devices

# Technological aspects

- **Bus requirements and specifications**
  - **Bandwidth**: bus speed – must be enough for supporting the working speed of the connected devices
  - **Length**
    - Some devices are close to each other, within few cm. (CPU, memory controller, graphics adapter, etc.)…
    - …whereas others may be farther apart (printer, scanner, network devices, etc.) A flexible cable is needed to connect them to CPU and memory
  - **Standardization**
    - Busses interconnecting *motherboard* devices (CPU, system clock, memory controller, etc.) need not be subject to a standard specification
    - Interchangeable devices (disks, graphics adapter, keyboard, etc.) need to accommodate the specification of standard busses

# Technological aspects

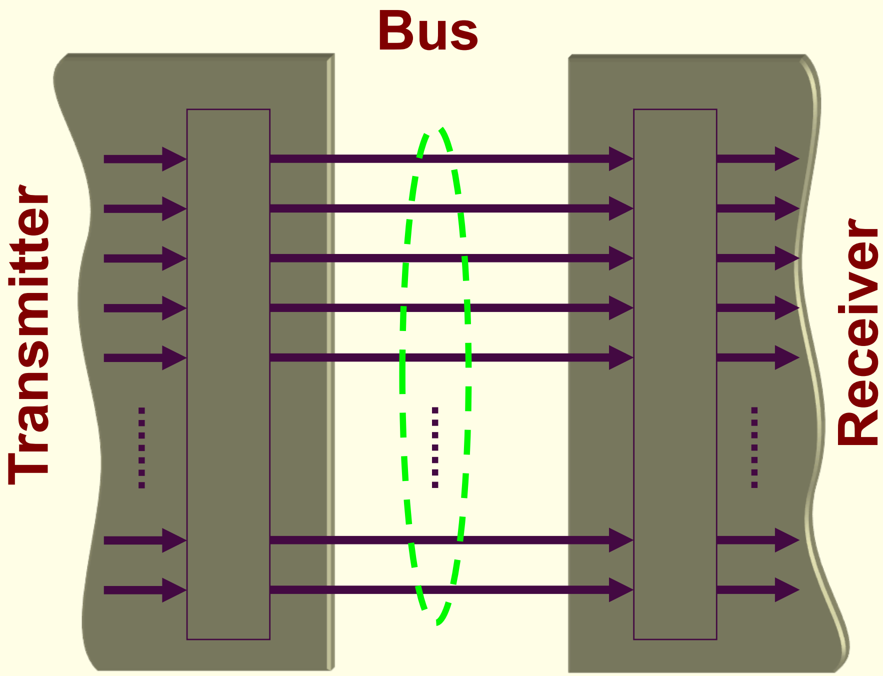
- **Electrical issues**
  - **Electrical noise**: Computer components and neighbouring equipment produce electromagnetic interference
    - The problem grows with cable length, and is reduced with shielding
  - **Degradation** and **clock skew**: signals lose shape and synchronicity when they travel the bus cables. The problem grows with:
    - Faster bus transfers (shorter bus cycles)
    - Cable bending, that alters the geometry and electrical characteristics of cables in the bus
  - **Crosstalk**: lines in a bus interfere each other. The problem grows:
    - With the number of wires in the bus
    - With insufficient shielding (e.g. when it is sacrificed for cable flexibility)

# Technological aspects

- **Physical aspects**
  - A bus is composed by a set of electric wires
    - Often times, a *shielding* conductor wraps the others to reduce electromagnetic interference
    - Wires are spaced from each other to reduce crosstalk
    - Cable length is limited
  - A bus specification includes detailed description of the allowed connectors
  - Each bus uses a **protocol**: a particular way of commanding signals in the bus, including also error detection
  - A bus may be **serial** or **parallel**, according to the number of data bits transferred in a singe *bus clock* cycle

# Technological aspects

- **Parallel busses**
  - All word bits are transferred simultaneously in a single bus clock cycle
  - Parallel bus example (unidirectional)

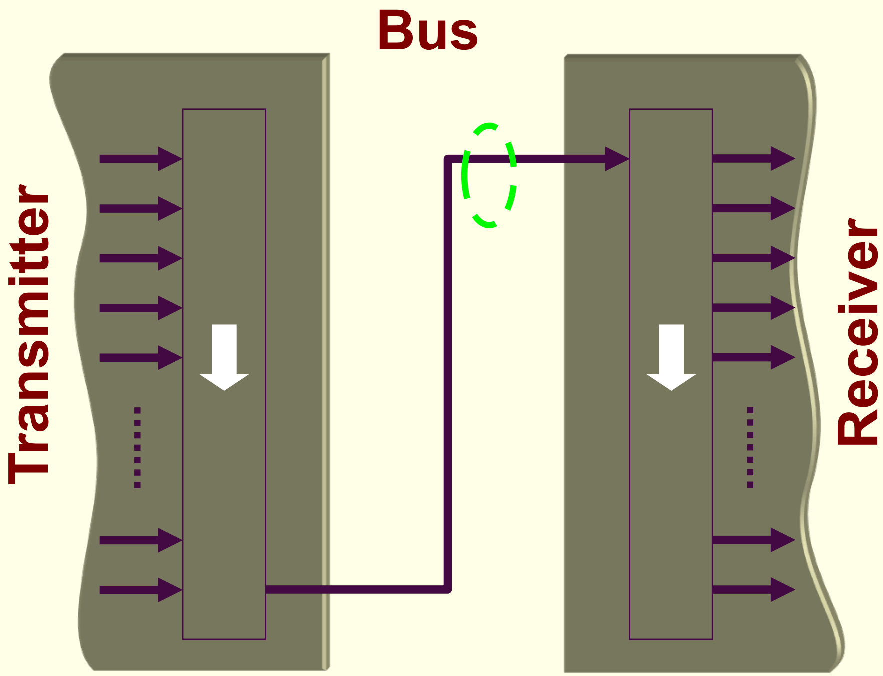



Implementation using two registers with parallel load

1. Transmitter writes its register
2. Data arrives in the receiver register and is written there
3. Receiver can then read the transmitted data from its own register

# Technological aspects

- **Serial bus**
  - Bits are transmitted serially. For *n*-bit words, *n* elementary transmissions (bus clock cycles) are needed to transfer a full word
  - A serial bus requires only one wire for data
    - Few more lines for signalling, power, etc.




Implementation with two shift registers

1. Transmitter writes the output register
2. Word is transferred bit by bit from the transmitter's serial output to the receiver's serial input
3. Receiver can read the whole word only at the end of transfer

# Technological aspects

- **Serial and parallel busses: quick comparative**
  - The control unit for the serial connection is more complex
  - Parallel cables are heavier and more rigid. Connectors are more fragile and less convenient
  - Under ideal conditions, parallel is faster, but…
    - …"ideal" means no noise and perfect wires (no capacitance, no inductance)
  - For high frequencies (~ GHz), parallel busses can only be very short (~ cm) due to clock skew and cross-talk

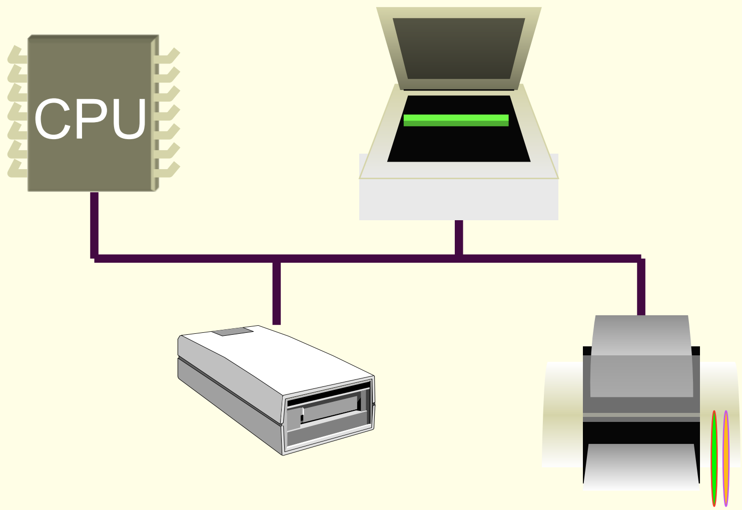
| Type | Control complexity | Electrical issues | No. of wires and pins |
|---|---|---|---|
| Parallel | low | serious | many |
| Serial | higher | simple | few |

# Technological aspects

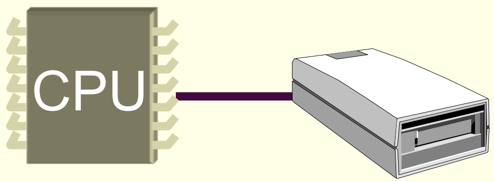
- **Bus bandwidth**
  - A bus has an associated clock frequency, $f$
  - For a parallel bus of width $w$ data bits, the resulting bandwidth is $B = f \times w/8$ Bps
    - Examples:
      - PCI-X. $f$ = 133.3 MHz, $w$ = 64 bits, $B$ = 1066.6 MBps
      - Parallel ATA-133. $f$ = 66 MHz, $w$ = 16 bits, $B$ = 133 MBps
  - For a serial bus, $B = f$ bps, BUT redundant bits for synchronization and error control need be subtracted to obtain the *effective* bandwidth
    - Examples
      - PCIe-1x (version 2). $f$ = 5 GHz, using 10 bits/byte → B = 500 MBps
      - SATA-3 Gbps. $f$ = 3 GHz, using 10 bits/byte → B = 300 MBps

# Bus topologies

- **Multipoint** (or *multidrop*)
  - Limited number of addressable devices
  - Example: ATA (limit = 2), PCI (limit = $2^{32}$ or $2^{64}$)

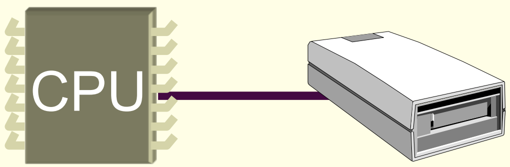



- **Point to point**
  - Direct connection between two devices
    - No need for device selection
  - Example: RS-232, AGP

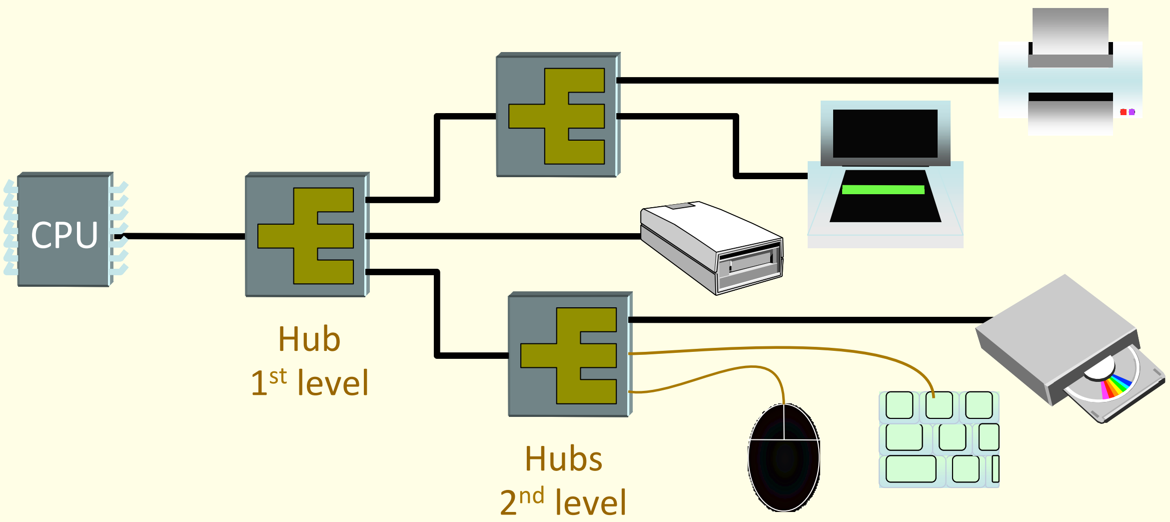


# Bus topologies

- **Star busses**
  - In their simplest form, point to point

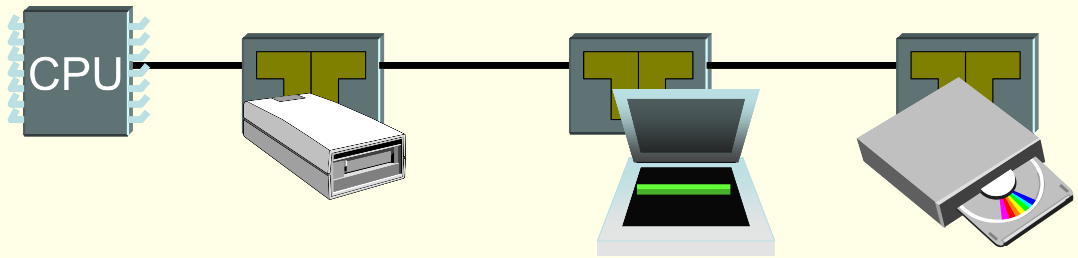



  - Extended with additional switching elements (*hubs*)
  - Example: USB, SATA

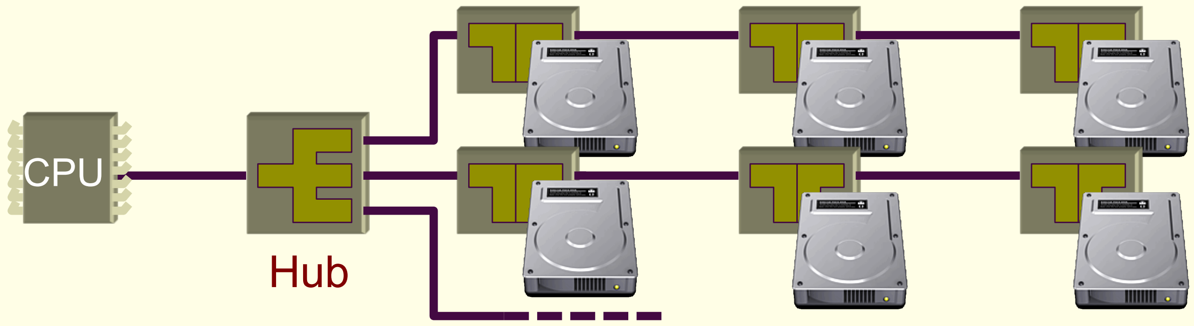


# Bus topologies

- **Daisy chain**
  - Point-to-point connection, with transducers (two connections)
    - If address on the bus is not *mine*, pass it on to next in the chain
  - Examples: SCSI, Firewire

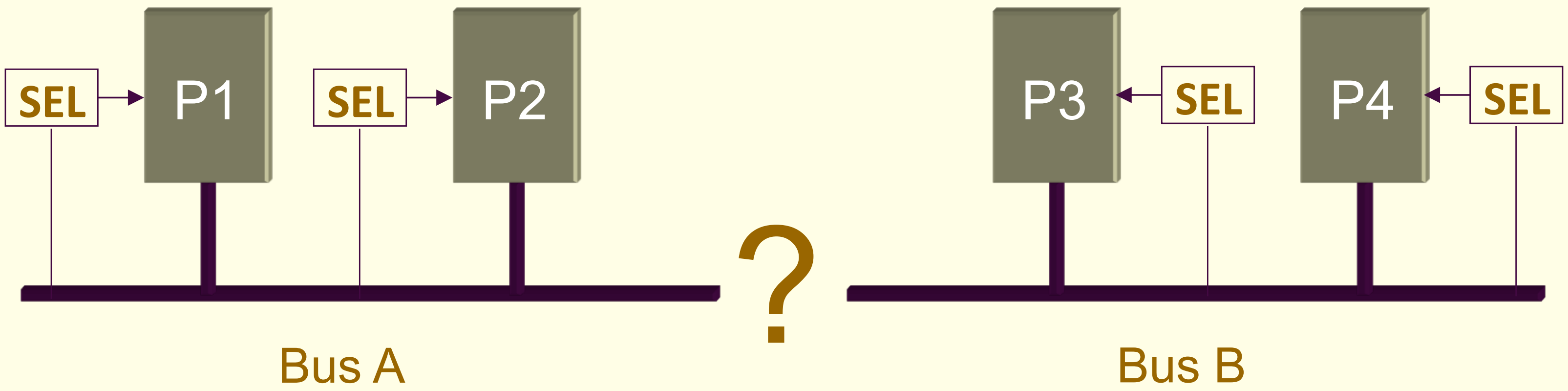



  - With hubs: tree topology (Firewire)



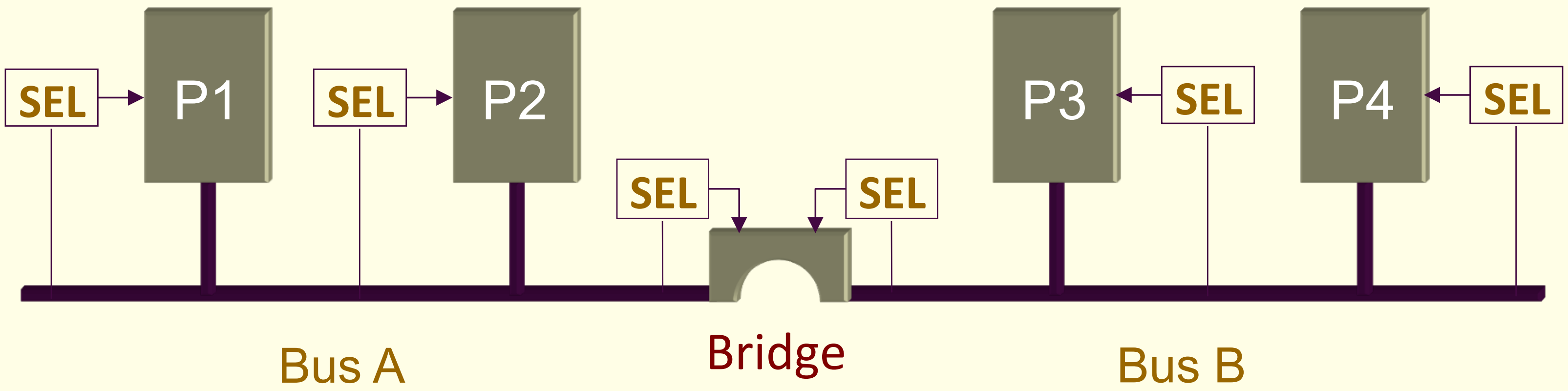
# Interconnecting busses

- Interconnecting two busses requires solving two issues:
  - **Physical matching**: each bus has its own specs for signals (frequencies, voltages…) Signals must be adapted from one bus to the other
  - **Logical matching**: Devices in one bus must be addressable from devices in the other



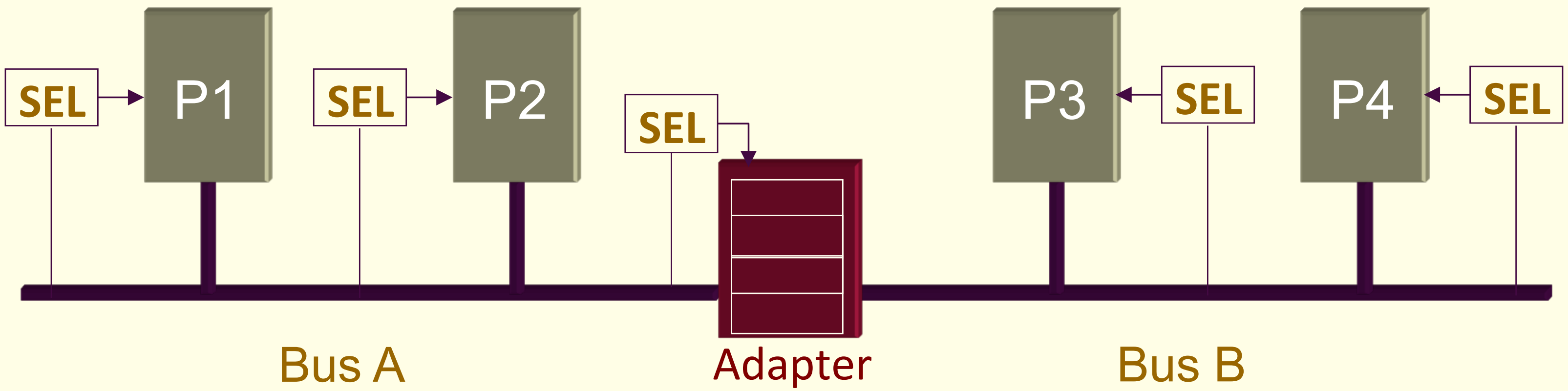
# Interconnecting busses

- **Bridges**
  - Bridges help preserving logical unity
    - They give the view of a single addressing space. Programs do not care about which bus a peripheral is physically connected to
  - A bridge is seen as another device on each bus. It provides access to devices on the other bus
    - On bus A, the addresses for P3 and P4 select the bridge
    - On bus B, the addresses for P1 and P2 select the bridge
  - When selected, the bridge translates the signals to the other bus



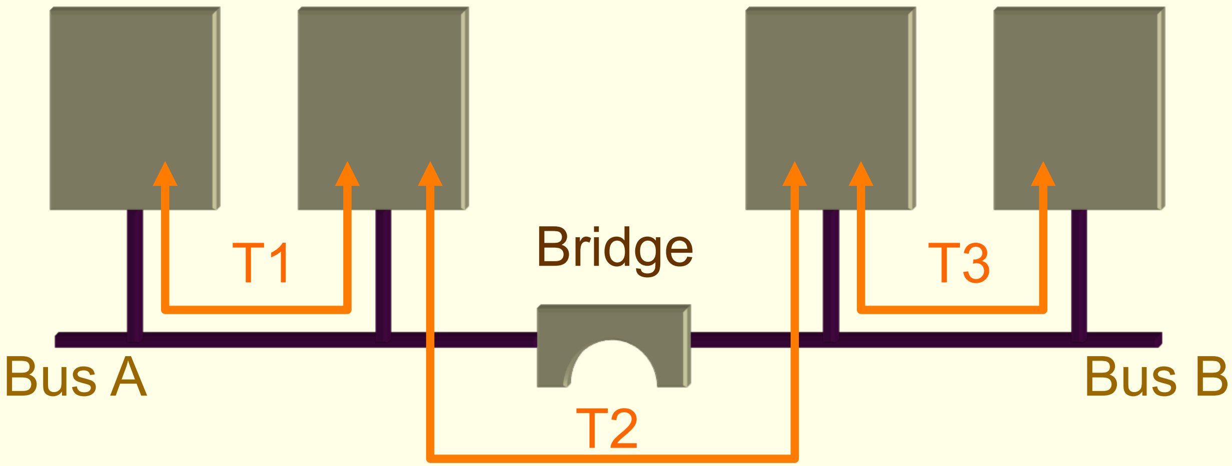
# Interconnecting busses

- **Bus adapters**
  - Adapters offer an interface to access devices on a different bus
  - The interconnected busses have separate addressing spaces
  - Programs must select the adapter and program its registers in order to access devices in the other bus through the adapter
  - Example: P1, P2 and adapter share the same addressing space on bus A. P3 and P4 can be selected from bus A by writing their addresses in the adapter



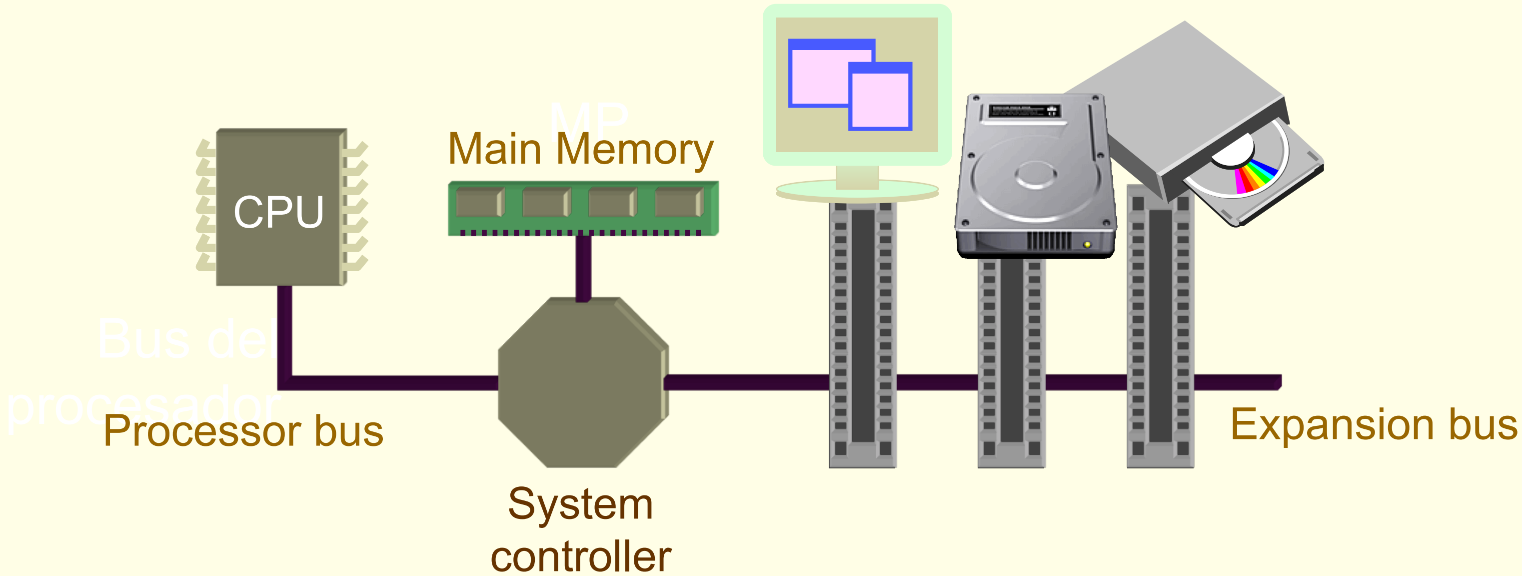
# Interconnecting busses

- **Bandwidth**
  - When two busses are interconnected by a bridge, the maximum bandwidth is:
    - The bandwidth of the used bus, when the transfer does not cross the bridge
    - The minimum bandwidth in the path, when the transfer needs to cross the bridge
  - Examples: T1 → $B_A$ T2 → $\min(B_A, B_B)$ T3 → $B_B$



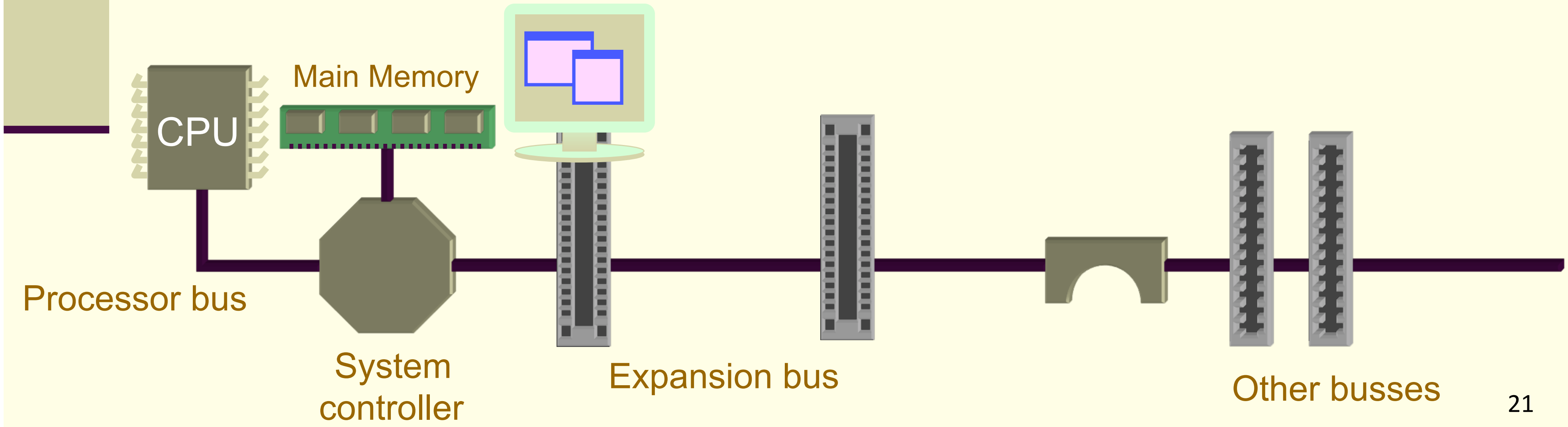
# Bus hierarchy

- **The *system controller***
  - Combination of DRAM controller plus a set of bridges connecting:
    - **The processor bus**. Proprietary design, depends on the particular CPU. Targets maximum bandwidth
    - **The main memory bus**. Complies with a given standard (e.g., DDR3)
    - **The standard expansion bus**. Targets maximum compatibility. Provides standard connectors for compatible peripheral adapters



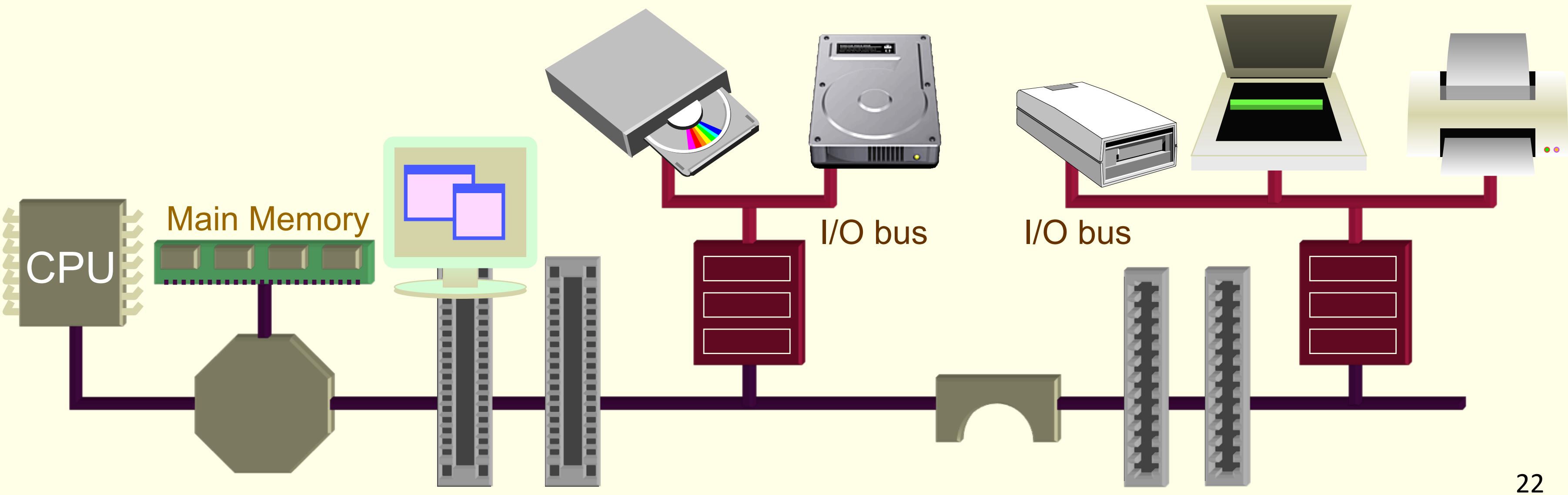
# Bus hierarchy

- **The system bus**
  - A system bus connects the devices that are mapped in the addressing space of the processor
  - It is typically formed by the processor bus, the system controller and the expansion bus
    - The expansion bus may connect to other busses via bridges



# Bus hierarchy

- **I/O busses (or peripheral busses)**
  - A set of standard I/O busses connected to the system bus via bus adapters
  - Each I/O bus has its own addressing space. Programs must use the bus adapter interface in order to access the peripherals



# Bus hierarchy

- **Summary**
  - Typical computers include a set of busses
    - Busses are interconnected by bridges and adapters
    - Each bus is chosen to satisfy certain criteria: compatibility, bandwidth, etc.
  - The closer to CPU-memory, the faster the bus

# Current busses

# Trends

- **Years 1980…2000 (approx.):**
  - Parallel, multipoint expansion busses (ISA - PCI, NuBus…)
  - Parallel peripheral busses (Centronics, SCSI, ATA) for higher bandwidth demands (scanner, hard disk, etc.)
  - Serial peripheral busses (RS-232) for slower devices (mouse, keyboard) or remote devices (modems, printers)
- **Years 2000…2005**
  - Parallel expansion busses (PCI and AGP)
  - Parallel peripheral busses only for internal discs (ATA)
  - Serial peripheral busses (USB and Firewire)
- **Currently (2005…)**
  - Serial expansion busses (PCI express)
  - Serial peripheral busses (SATA, USB and Firewire)
  - Only the processor bus is still mostly parallel (not clear what the future will be in this regard…)

# Trends

## Devices’ bandwidth

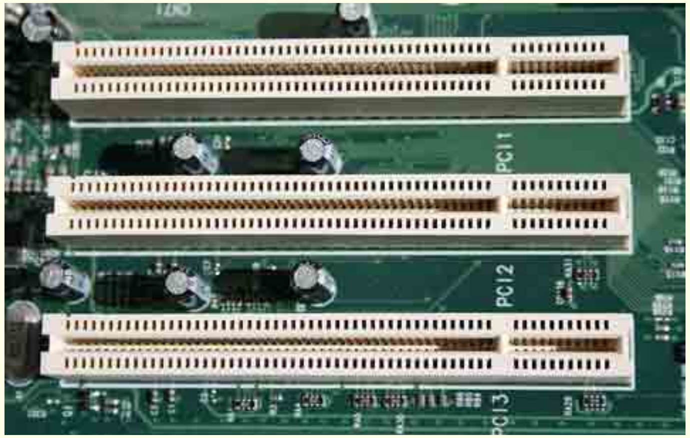
| Device | MBps (approx.) |
|---|---|
| processor (Core Duo 2GHz) | 10000 |
| SDRAM channel DDR3 400 MHz | 6400 |
| Graphics display (1600×1200, 50 fps) | 300 |
| Hard disk (7200 rpm, 1000 sectors/track) | 100 |
| DVD (20x) | 27 |
| CD-ROM (52x) | 7,8 |

## Peripheral busses

| Peripheral | old | current |
|---|---|---|
| mouse, keyboard | RS232 | USB |
| Graphics display | PCI, AGP | PCIe |
| Internal hard disk | ATA | SATA |
| External hard disk | SCSI | USB, Firewire, eSATA |
| Optical drive | ATA | SATA |

# PCI

- **Features**
  - Multipoint parallel bus designed to operate as system bus
  - Introduced in 1993, still in use for compatibility
- **Bandwidth**
  - In its many versions, bandwidth has evolved from the initial 133 MBps to 4GBps
    - PCI 2.3 (conventional): 533 MBps (64bits/66MHz)
    - PCI-X 1.0: 1GBps (64bits/133MHz)
    - PCI-X 2.0: 2GBps (64bits/266MHz)
      4GBps (64bits/533MHz)

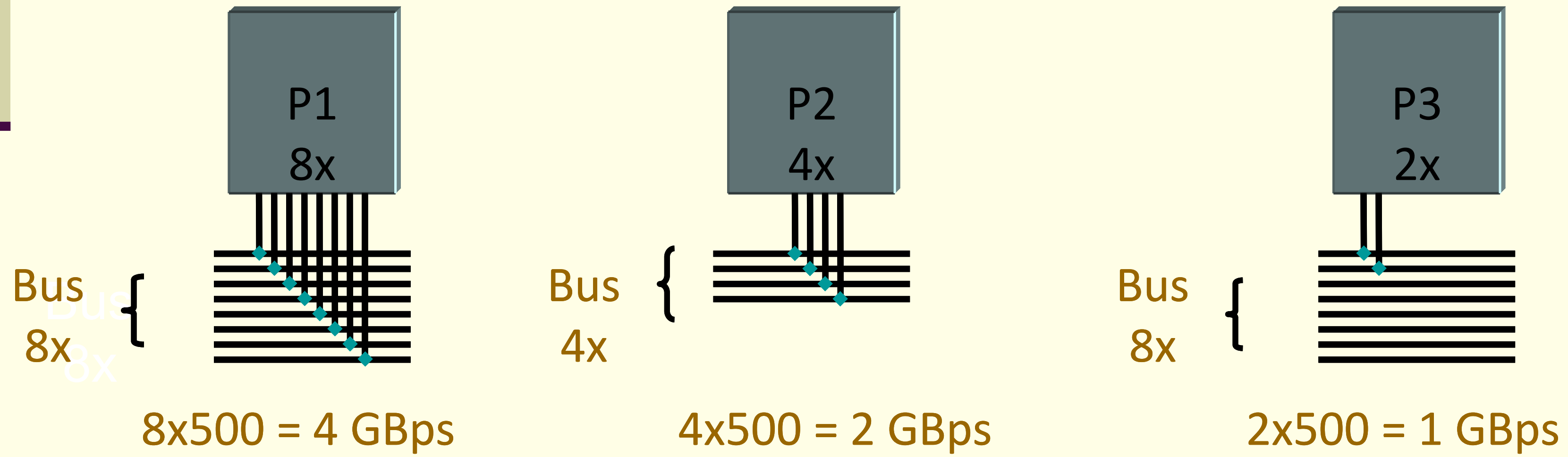


# PCI-express (PCIe)

- Evolved version of the classical PCI as system bus
- Serial bus, using 10 bits per byte (128b/130b in PCIe 3.0, 4.0)
- Point-to-point bus, characterized by the number of *lanes* $L_B$ (1x, 2x, 4x, 8x, 12x, 16x or 32x)
  - Each lane enables serial transfers at
    - 250 MB/s – 2.5 GT/s (PCIe versions 1.0 and 1.1),
    - 500 MB/s – 5 GT/s (version 2.0),
    - 984.6 MB/s – 8 GT/s (v. 3.0) [128b/130b]

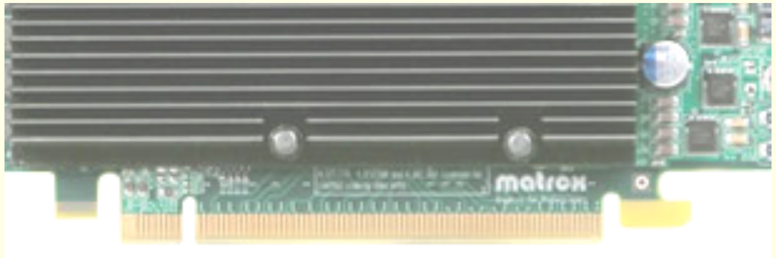
# PCI-express (PCIe)

- **PCIe peripheral adapters**
  - Each adapter has a given nr. $L_P$ of connections (1x, 2x, etc…) to bus lanes
  - At start-up, the system checks the number of available lanes to talk with the peripheral through the bus: min{$L_B$,$L_P$}
  - Common values for $L_P$: 8x for the graphics adapter, 1x for audio
  - Examples with PCIe 2.0:



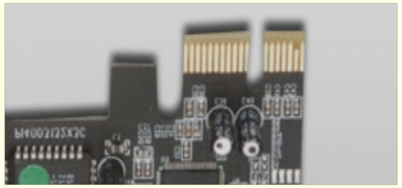
# PCI-express

- **PCI-express adapters and connectors**




16x adapter


1x adapter

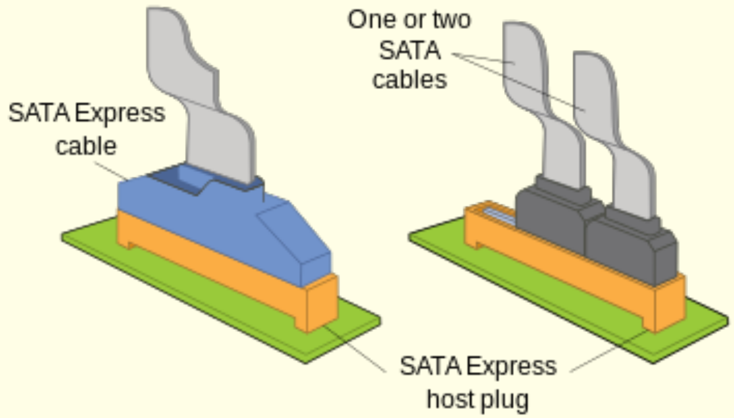


# Peripheral busses

- **SATA bus**
  - Most frequently used with storage units: hard disks and optical drives
  - Serial connection (maximum length of 1 m), 8b/10b encoding
  - Two topologies:
    - Point-to-point: one peripheral per bus
    - Star, with one level of *multipliers* (~hubs). Up to 15 peripherals
  - Versions and bandwidth:
    - SATA 1.5 Gb/s: 150 MBps (effective)
    - SATA 3 Gb/s “SATA II”: 300 MBps
    - SATA 6 Gb/s: 600 MBps
  - A version (*eSATA*) exists for external peripherals, for cables up to 2 m and bandwidth 3 Gbps (300 MBps)

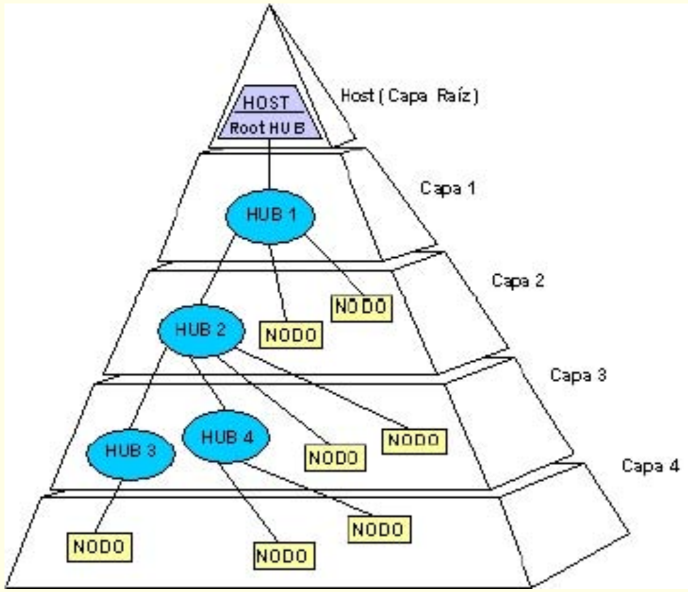
# Peripheral busses

- SATA Express bus
  - Supports SATA and PCIe 3.0
  - Up to 1969 MBps (2 PCIe lanes)
  - SATA-compatible connector
  - Conceived for solid state disks (SSDs)



# Peripheral busses

- **USB bus**
  - General purpose peripheral bus with serial transfer, star topology
  - One bus controller, several hubs where peripherals are connected
  - Cables:
    - Maximum length = 5 m (w/o power)
    - Asymmetrical connectors
    - Power lines included (5 V, 0.5 A typ.)
  - Up to 6 levels of hubs
  - Up to 127 devices per bus
  - Versions and bandwidth:
    - USB 1.0 and 1.1: 12 Mbps
    - USB 2.0: 480 Mbps
    - USB 3.0: 4.8 Gbps (8b/10b)
    - USB 3.1: 10 Gbps (128b/132b)



# Peripheral busses

- M.2 connectors
  - ✓ Enable multiple interfaces via a 75-pin connector

| KEY | INTERFACES | COMMON USES |
|---|---|---|
| A | PCIe x2, USB 2.0, I2C, DisplayPort x4 | Wi-Fi/Bluetooth, cellular cards |
| B | PCIe x2, SATA, USB 2.0, USB 3.0, audio, PCM, IUM, SSIC, I2C | SATA and PCIe x2 SSDs |
| E | PCIe x2, USB 2.0, I2C, SDIO, UART, PCM | Wi-Fi/Bluetooth, cellular cards |
| M | PCIe x4, SATA | PCIe x4 SSDs |

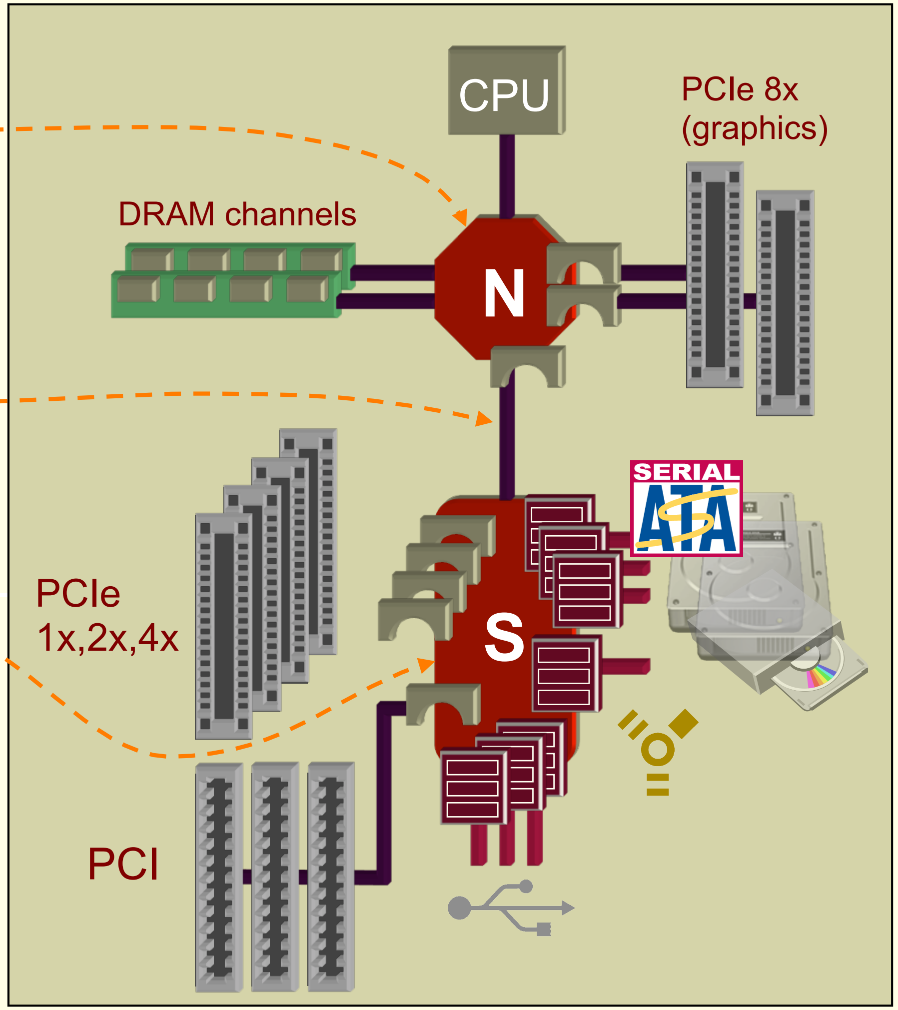

Two B- and M-keyed SSDs (left)
M-keyed SSD (right)

# Peripheral busses

- **Firewire (IEEE 1394, i.Link)**
  - General purpose peripheral bus
    - Up to 63 peripherals
  - Serial transfer, daisy chain topology
    - Maximum length: 4,5 m for one cable, 72 m the whole bus
  - Very versatile
    - Allows connection between computers
    - Allows direct communication between devices connected to the bus
    - Widely used for professional video
  - Versions and bandwidth:
    - Firewire 400 Mbps
    - Firewire 800 Mbps
    - Firewire 1600 and 3200 Mbps

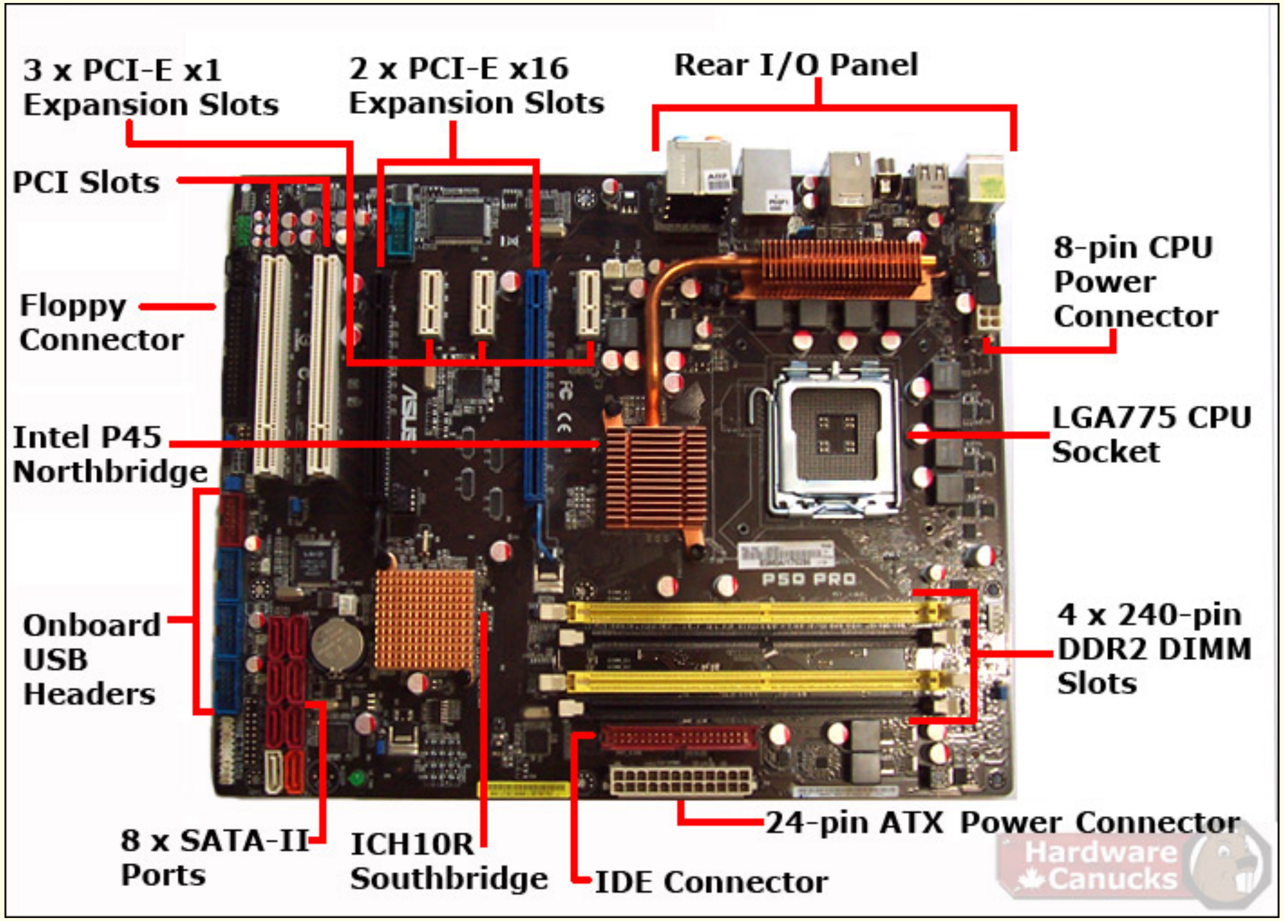
# Trends

- **Current bus hierarchy**
  - The system controller (*northbridge*) controls access to the fastest busses: 2 DRAM channels and video adapter
  - A (usually proprietary) bus connects the *northbridge* with the *southbridge*
  - The *southbridge, system hub* or I/O controller is a collection of bridges and I/O bus adapters

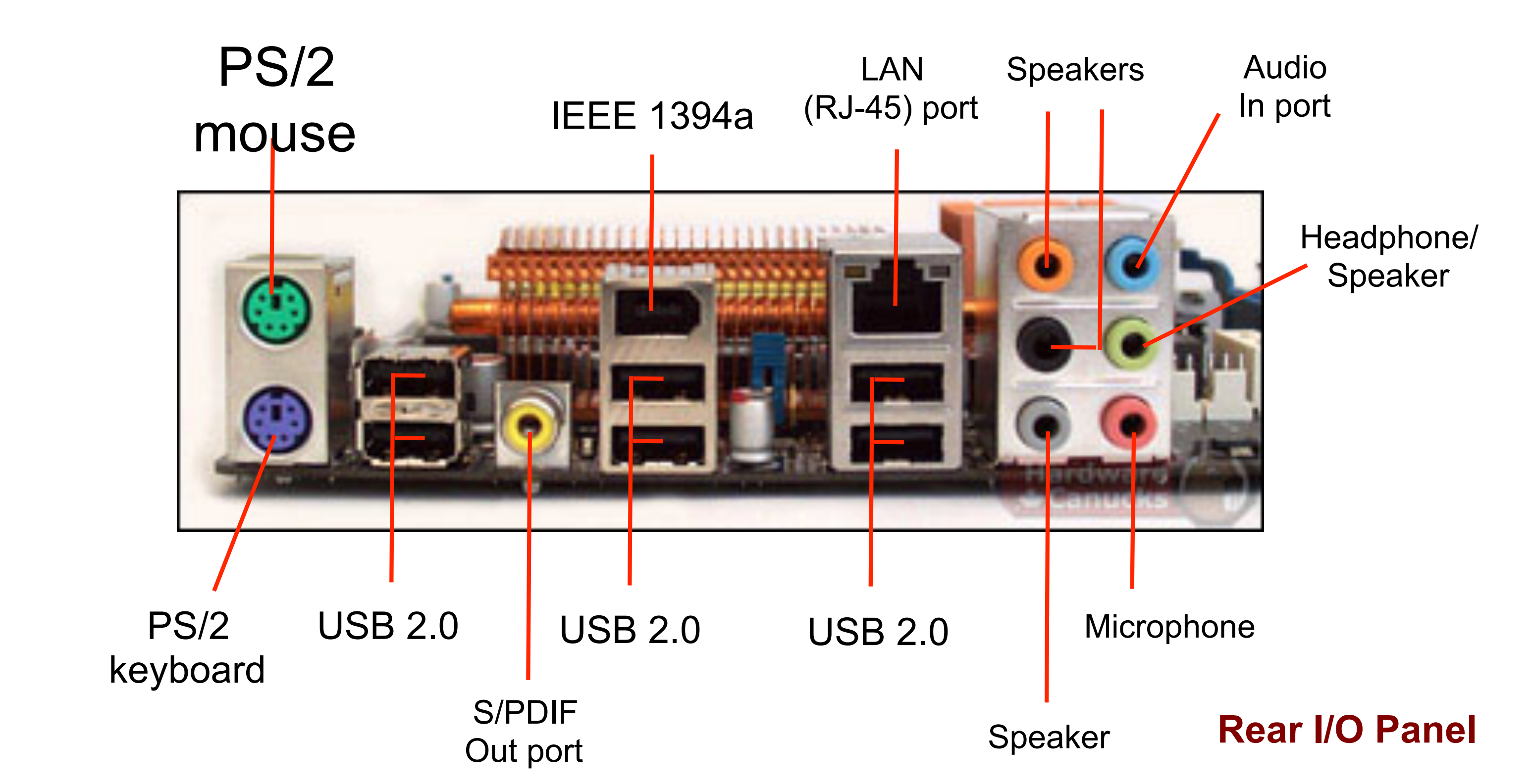


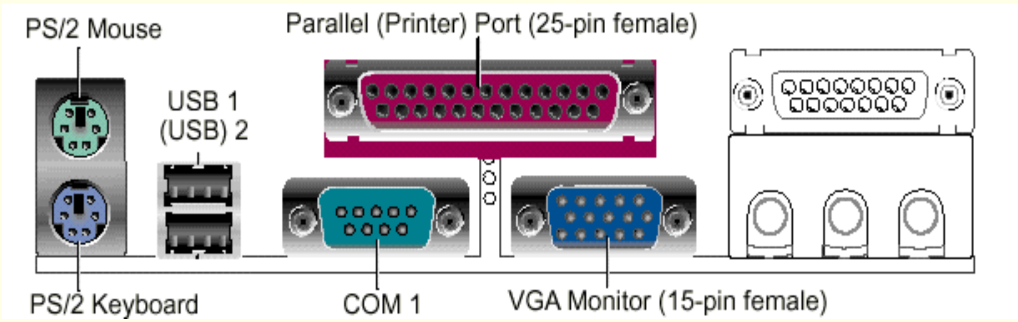
# Motherboard connections

ASUS P5Q PRO



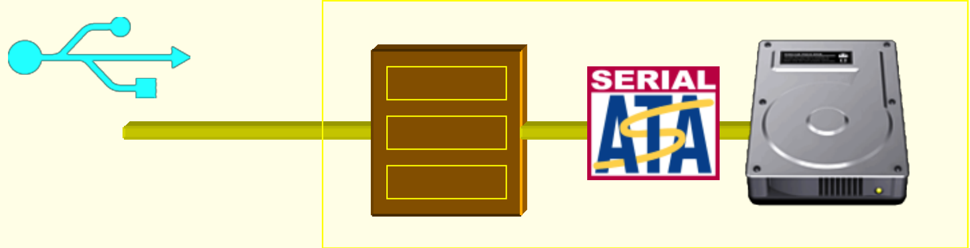
# Motherboard connections



# Trends

- **External units**
  - **Via bus adapter:**
    - Combine a disk unit and a bus adapter (e.g. *external USB disk drive*)
    - The included bus adapter makes the translation between the general-purpose I/O bus (USB, Firewire) and a specific bus (e.g. SATA)
    - The applicable IODTR* is limited by the slowest bus (the I/O bus; e.g. USB is slower than SATA)




  - **Without bus adapter**
    - Sometimes a faster connection is directly available for external units (e.g. eSATA)
    - The applicable IODTR is high, greater than the SDTR **

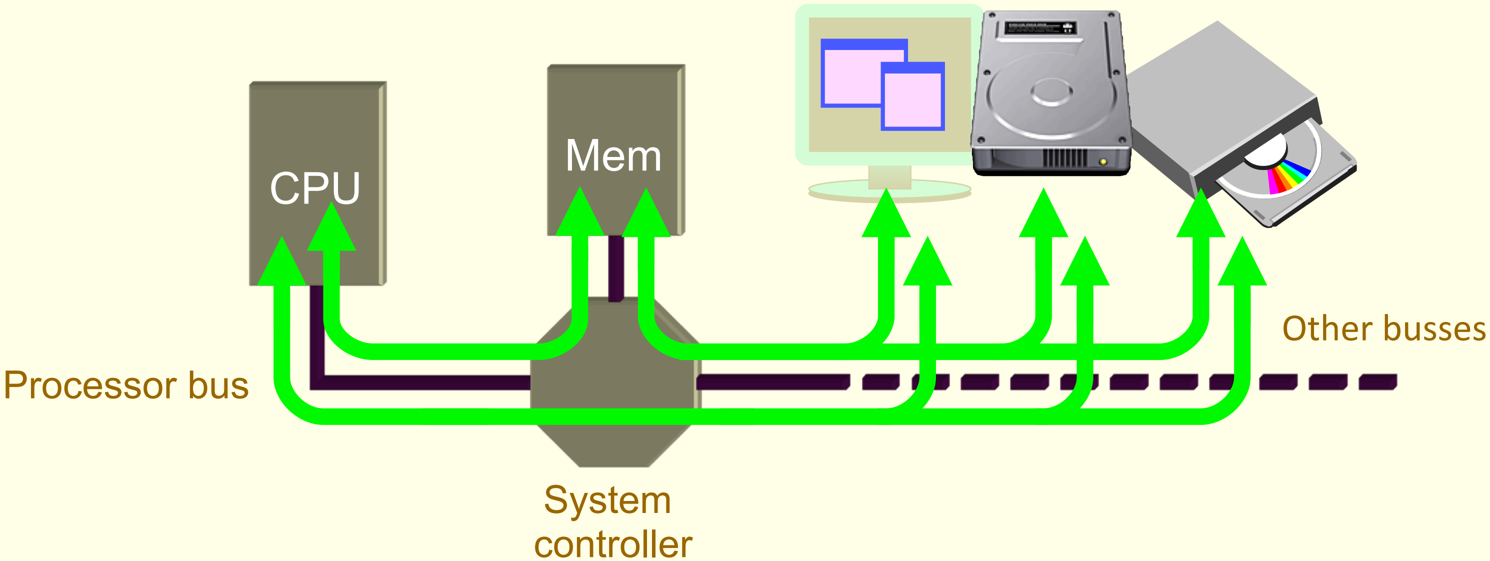
**IODTR = Input-Output Data Transfer Rate,* a parameter of the bus

***SDTR = Sustained Data Transfer Rate,* a parameter of the device

# Transfers within the computer

# Transfers within the computer

- **Data flow within the computer**
  - Main memory is the *central* resource
  - All traffic crosses the system controller
    - The CPU reads/writes data and fetches instructions from there (when cache misses)
    - DMA block devices make transfers Main Memory ↔ Peripheral
    - PIO devices make transfers Main Memory ↔ CPU ↔Peripheral



# Transfers within the computer

- **Data flow control**
  - The system bus and many I/O busses support concurrent transfers
    - Elements connected to the same bus compete for it
    - Controllers and bus arbiters handle the multiplexing of concurrent transfers
    - On each bus, the bandwidth consumption is the sum of bandwidths consumed by all transfers taking place
    - Utilization of the bus cannot surpass the maximum bus bandwidth. The bus controller/arbiter can limit the speed of individual transfers to keep the transit within the bus limits
  - When a transfer traverses several busses, the effective bandwidth is limited by the slowest bus involved

# Transfers within the computer

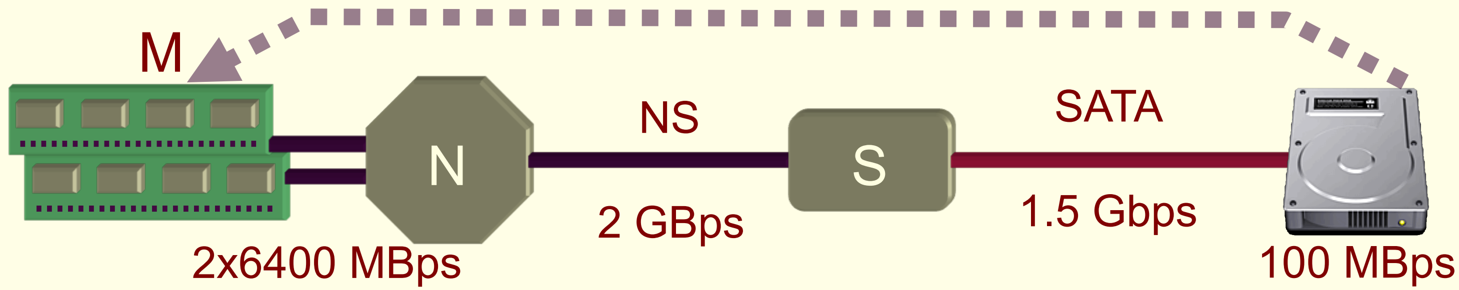
- **The role of the OS**
  - Regular programs use OS services to use the peripherals
    - Especially the file-system-related functions
  - The OS services program the peripheral adapters and carry the corresponding PIO or DMA transfers
    - With file-system services, the OS performs additional housekeeping functions (directory modifications, file table updating, etc.)
  - Under ideal circumstances, the **theoretical bandwidth** depends only on the speeds of the busses (IODTR) and peripherals (SDTR) involved
  - In actual fact, the **effective bandwidth** is reduced by other factors (time taken to program the devices, arbitration conflicts, etc.)

# Transfers within the computer

- **Real-time aspects**
  - Transfers with no real-time restrictions (*as fast as possible*)
    - File transfers, file read/write, internet browsing...
    - Transfers occur at the maximum available bandwidth, taking a time T = (amount of data)/B
  - Transfers with real-time restrictions (*within given deadlines*)
    - Typically, multimedia: audio/video playing/recording, streaming…
      - They must comply with some given real-time restriction (frames per second, audio samples per second)
    - If the available bandwidth is sufficient, transfers occur at the appropriate speed. Otherwise, the results will be defective
    - A special case are critical real-time applications, where failing to meet the timing requirements can lead to catastrophic results
      - e.g. control systems in aviation, air traffic control, medical equipment, satellites, power stations, etc.

# Examples

- **Example 1**
  - A program reads a full 1 GB file from a hard disk connected to a SATA bus (1.5 Gbps)
    - Disk: SDTR = 100 MBps; DMA transfer
    - Context:

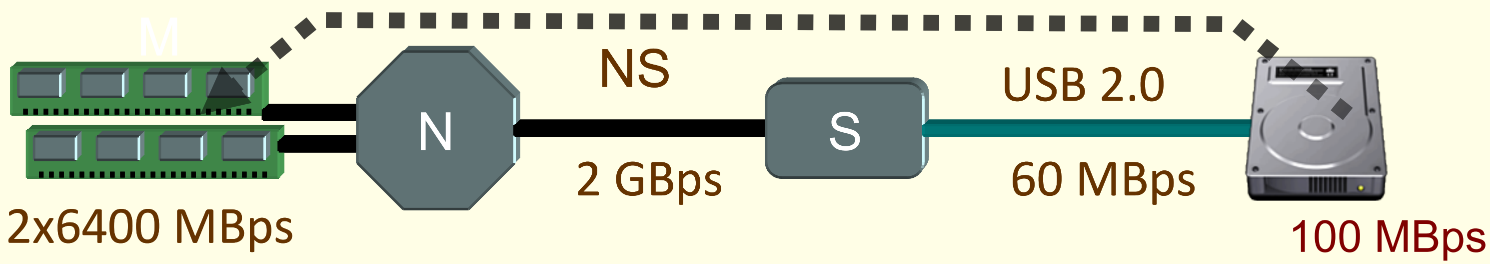



**What we can conclude**

- The maximum bandwidth is limited by the hard disk
- Minimum time for transfer is 10 s (1 GB / 100 MBps)
- Bandwidth consumption in the busses:
  100/12800 = 0.78% (M);   100/2000 = 5% (NS);   100/150 = 67% (SATA)

# Examples

- **Example 2**
  - Full read of a 1 GB file in a external disk, connected via USB 2.0
    - Disk: SDTR = 100 MBps; DMA transfer
    - Context:

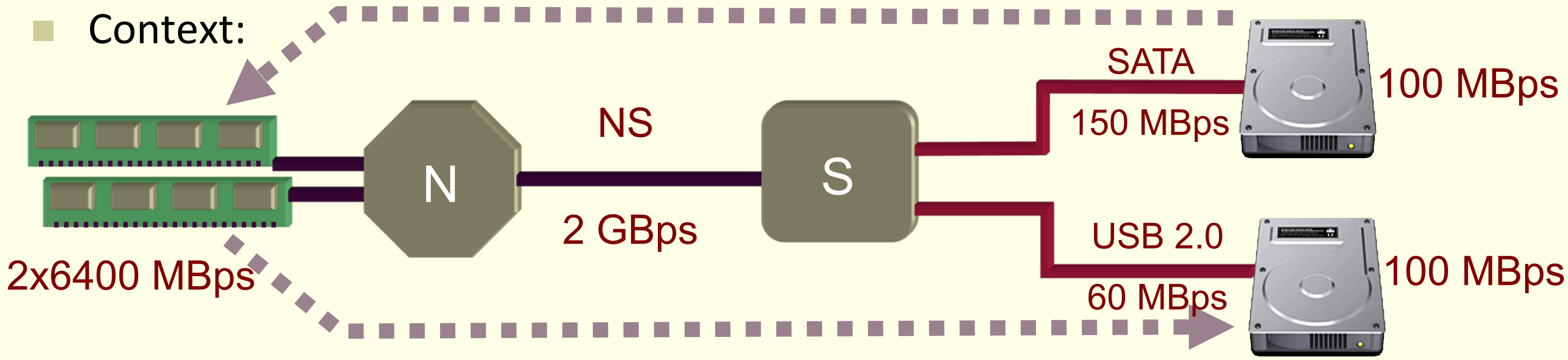



**What we can conclude:**

- Bandwidth limited by the USB bus
- Minimum time for transfer is 16.7 s (1 GB / 60 MBps)
- Bandwidth consumption in the busses:
  60/12800 = 0.47% (M); 60/2000 = 3% (NS); 100% (USB)

# Examples

- **Example 3**
  - Copy a 1 GB file from disk (connected to SATA 1.5 Gbps) to another disk (connected to bus USB 2.0)
  - Disks: 100 MBps (SDTR); DMA transfer
  - Context:

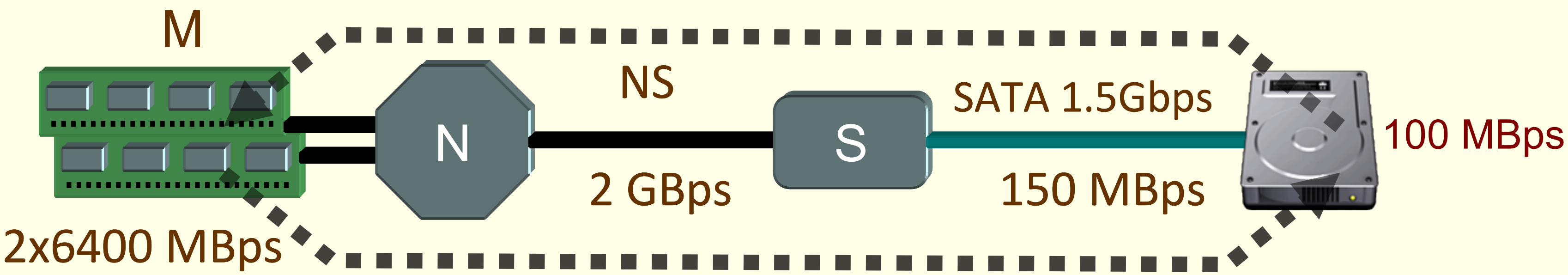



**What we can conclude:**

- Bandwidth limited by USB bus
- Minimum transfer time is 16.7 s (1 GB / 60 MBps)
- Bandwidth consumption in the busses:
  2*60/12800 = 0.94% (M); 2*60/2000 = 6% (NS); 60/150 = 40% (SATA); 100% (USB)

# Examples

- **Example 4**
  - Copy a 1 GB file to the same disk (connected to SATA 1.5 Gbps)
  - Disk: 100 MBps (SDTR); DMA transfer
  - Context:




**What we can conclude:**

- Bandwidth limited by the disk
- Minimum transfer time is 20 s (2*1GB / 100 MBps)
- Bandwidth consumption in the busses:
  100/12800 = 0.78% (M); 100/2000 = 5% (NS); 67% (SATA)

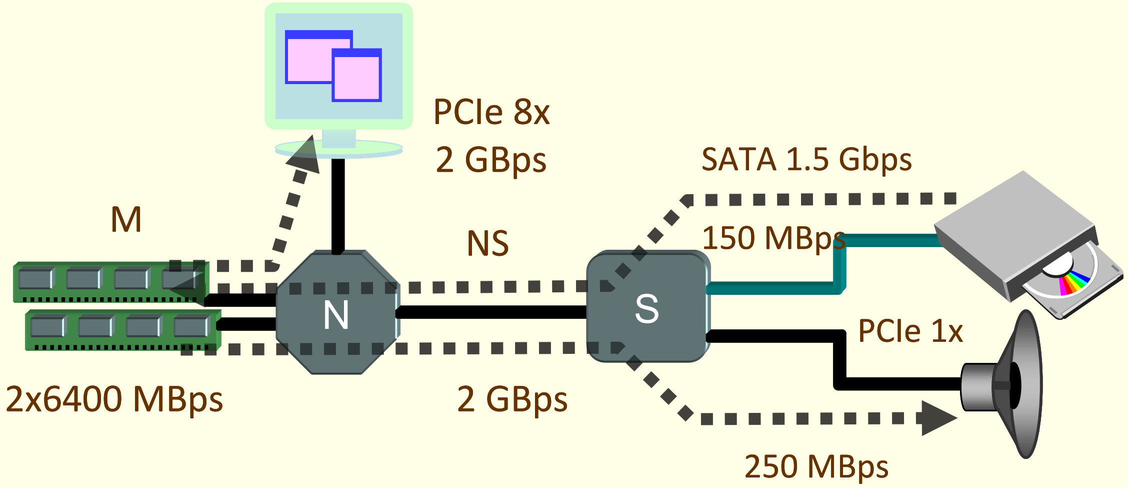
# Examples

## Example 5 (with real-time restrictions)

- Playing a DVD movie (video + audio)
  - Transfer contents (mpeg-2) from DVD to Main Memory
  - Meanwhile, the CPU decodes the mpeg-2 to obtain the video frames and PCM audio
  - Video frames need be transferred to the display at the proper frequency
  - Audio needs also be transferred to the sound card
- Context:
  - Movie at 30 fps (frames per second)
  - DVD: mpeg-2 encoding at 10 Mbps
  - Assume the CPU is fast enough to decode mpeg-2 at proper speed
  - Audio 5.1, 16 bits at 48 KHz
  - Frame size is 1600 x 1200 pixels, 24-bit colour

# Examples

- **Example 5**
  - Context:




**The transfer is NOT limited by the busses:**

- DVD read: 10 Mbps = 1.25 MBps
- Graphics adapter: 1600 x 1200 x 3 x 30 = 172.8 MBps
- Audio: 6 x 48000 x 2 = 576 KBps = 0.576 MBps

**Busses utilization:**

- 1.25/150 = 0.83% (SATA); 172.8/2000=8.6% (PCIe-graphics); 0.576/250 = 0.23% (PCIe-audio); (1.25 + 0.576)/2000 = 0,091% (NS); (1.25 + 0.576 + 172.8)/12800 = 0.14% (Memory)

# Examples

- **Example 6 (with real-time restrictions)**
  - Play an uncompressed, high definition silent movie from hard disk (30 fps, 1920 x 1080 pixels, 24-bit colour depth)
    - Each second, read 30 frames of 1920x1080x3 bytes from disk
    - No need to decode frames
    - Each second, write 30 frames in graphics memory
    - The bandwidth required by each transfer is ≈ 187 MBps
  - Context:
    - Display: 1920 x 1080 pixels, 24-bit colour
    - Disk: 100 MBps connected to SATA 3Gbps

**The disk is not fast enough**
**Playback will be defective**