Input/output

DSC 315: Computer Organization & Operating Systems

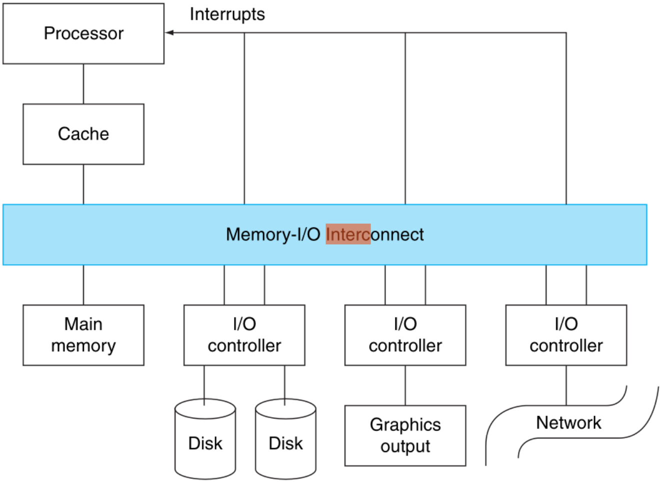
Dr. Laltu Sardar

School of Data Science,
Indian Institute of Science Education and Research Thiruvananthapuram (IISER TVM)

February 11, 2026

Input-output and I/O System Design
Section 6.1

I/O Connections



- processor, memory, I/O devices
- **Bus**: shared communication link, single set of wires
- **Bus types**:
  - Processor–memory bus: short, high speed, bandwidth optimized
  - I/O bus: long, many devices, varied bandwidth and latency
  - Special-purpose buses: e.g. graphics
- **Advantages**: versatility, low cost, easy device addition
- **Disadvantage**: communication bottleneck, limited I/O throughput

Diversity of I/O devices.

Device	Behavior	Partner	Data rate (Mbit/sec)
Keyboard	Input	Human	0.0001
Mouse	Input	Human	0.0038
Voice input	Input	Human	0.2640
Sound input	Input	Machine	3.0000
Scanner	Input	Human	3.2000
Voice output	Output	Human	0.2640
Sound output	Output	Human	8.0000
Laser printer	Output	Human	3.2000
Graphics display	Output	Human	800.0000–8000.0000
Cable modem	Input or output	Machine	0.1280–6.0000
Network/LAN	Input or output	Machine	100.0000–10000.0000
Network/wireless LAN	Input or output	Machine	11.0000–54.0000
Optical disk	Storage	Machine	80.0000–220.0000
Magnetic tape	Storage	Machine	5.0000–120.0000
Flash memory	Storage	Machine	32.0000–200.0000
Magnetic disk	Storage	Machine	800.0000–3000.0000

- I/O devices are incredibly **diverse**
- Can be organized with Three **characteristics**
  1. **Behavior**: Input or output or storage
  2. **Partner**: human or a machine
  3. **Data rate**: The peak rate at which data can be transferred

I/O Performance Measurements

- **Throughput (Bandwidth)**
  - Total Data transferred per unit time
  - Units: MB/s, GB/s
- **Response Time (Latency)**
  - Time to complete a single I/O request
  - Includes: Queueing + Service time
- **IOPS**
  - I/O Operations Per Second
  - Important for SSD and databases
- **Utilization**

$$\text{Utilization} = \frac{\text{Busy Time}}{\text{Total Time}}$$

Connecting Processors, Memory, and I/O Devices
Section 6.5

Bus-Based Organization

Bus

- Shared communication link
- Connects CPU, Memory, I/O

Advantages

- Versatility
- Standard interface
- Low cost (shared wiring)

Limitation

- Communication bottleneck
- Limited bandwidth
- Device contention

Types of Buses

- Processor–Memory Bus
- I/O Bus

Challenges and Modern Interconnects

Design Challenges

- Bus length limits speed
- Number of devices
- Clock skew
- Signal reflection

Heterogeneity

- Different latencies
- Different bandwidth needs

Industry Shift

- From parallel shared buses
- To serial point-to-point links
- Switched interconnects

Current Focus

- Processor interconnects
- Memory connections
- I/O networking

Characteristics of I/O devices

Synchronous Bus vs Asynchronous Interconnect

Synchronous Bus

- Shared global clock
- Fixed, clocked protocol
- Read / Write transactions
- Simple finite state control
- Fast over short distances

Limitations

- All devices same clock rate
- Clock skew limits length

Asynchronous Interconnect

- No global clock
- Handshaking protocol
- Variable latency support
- Scales to longer distances

Characteristics

- More complex control
- Flexible timing
- Suitable for modern serial links

I/O Transactions

I/O Transactions

- Address phase + Data phase
- Input: Device $\rightarrow$ Memory
- Output: Memory $\rightarrow$ Device

Examples (Standards)

- USB, PCIe, SATA, SAS, FireWire

Key characteristics of five dominant I/O standards.

Characteristic	Firewire (1394)	USB 2.0	PCI Express	Serial ATA	Serial Attached SCSI
Intended use	External	External	Internal	Internal	External
Devices per channel	63	127	1	1	4
Basic data width (signals)	4	2	2 per lane	4	4
Theoretical peak bandwidth	50 MB/sec (Firewire 400) or 100 MB/sec (Firewire 800)	0.2 MB/sec (low speed), 1.5 MB/sec (full speed), or 60 MB/sec (high speed)	250 MB/sec per lane (1x); PCIe cards come as 1x, 2x, 4x, 8x, 16x, or 32x	300 MB/ sec	300 MB/sec
Hot pluggable	Yes	Yes	Depends on form factor	Yes	Yes
Maximum bus length (copper wire)	4.5 meters	5 meters	0.5 meters	1 meter	8 meters
Standard name	IEEE 1394, 1394b	USB Implementors Forum	PCI-SIG	SATA-IO	T10 committee

Interfacing I/O Devices to the Processor, Memory, and Operating System

Section 6.6

Key Questions in I/O System Design

- How is a user I/O request transformed into a device command and communicated to the device?
- How is data actually transferred to or from a memory location?
- What is the role of the operating system?

Interfacing I/O Devices

Processor, Memory, and Operating System

- I/O communication uses a bus or network protocol
- Protocol defines data format and signal timing
- Extra mechanisms are needed for actual data transfer

Key Characteristics of I/O Systems

- **Shared Resource**
  - Multiple programs share the I/O system
  - Requires coordination and protection
- **Interrupt-Driven**
  - Devices generate interrupts
  - Causes switch to kernel mode
  - Managed by the Operating System
- **Complex Device Control**
  - Concurrent events
  - Strict timing and protocol requirements
  - Detailed low-level management

Defines **Responsibilities** of the OS

Operating System Functions in I/O Systems

OS acts as an interface between hardware and programs

- **Protection and Access Control**
  - Enforces user permissions
  - Prevents unauthorized device or file access
  - Disallows direct user-level I/O
- **Device Abstraction**
  - Provides high-level interfaces
  - Hides low-level device complexity
- **Interrupt Handling**
  - Handles device-generated interrupts
  - Executes in kernel mode
- **Resource Scheduling**
  - Fair sharing of I/O devices
  - Improves overall system throughput

OS and I/O Device Communication

To manage I/O securely and efficiently, the OS must control all communication between user programs and devices.

- **Command Issuing**
  - OS sends device commands
  - Examples: read, write, disk seek
- **Device Notification**
  - Device signals completion or error
  - Typically via interrupts
- **Data Transfer**
  - Data moved between memory and device
  - Example: disk block to main memory

Command Issuing

Purpose: OS initiates and controls device operations

- OS writes to **device control registers**
  - Command register
  - Status register
  - Data register
- Commands include:
  - Read / Write
  - Seek (disk positioning)
  - Reset / Configure device
- Access methods:
  - Memory-mapped I/O
  - Isolated I/O instructions
- Executed in **kernel mode** to enforce protection

Device Notification

Purpose: Device reports completion, status, or errors

- Implemented via **hardware interrupts**
- Typical flow:
  1. Device asserts interrupt line
  2. CPU saves context
  3. Control transfers to interrupt handler
  4. OS services device
- Advantages:
  - Avoids busy waiting
  - Improves CPU utilization
- May include:
  - Priority levels
  - Interrupt vector table

Data Transfer

Purpose: Move data between device and main memory

- Data path:
  - Device buffer $\leftrightarrow$ Memory buffer
- Mechanisms:
  - **Programmed I/O**
    - CPU moves each word
    - Simple but CPU intensive
  - **Interrupt-Driven I/O**
    - CPU notified per block or word
  - **DMA (Direct Memory Access)**
    - DMA controller transfers block
    - CPU interrupted only on completion
- Tradeoff: CPU overhead vs hardware complexity

Design of an I/O system

Section 6.8

Designing an I/O System: Specifications

Two Key Performance Constraints

- **Latency Constraints**
  - Bound the time to complete an I/O operation
  - Critical for real-time or device safety
  - Unloaded system: sum component delays
  - Loaded system: queueing delay becomes dominant
- **Bandwidth Constraints**
  - Sustain required data throughput
  - Depends on workload characteristics
  - May require balancing an existing configuration

Performance Evaluation Tools

- Queueing theory for analytical estimation
- Simulation for complex traffic patterns

Designing an I/O System: Design Strategy

Workload Awareness

- Read/write ratio
- Request size distribution
- Burstiness and concurrency level

General Design Method

1. Identify the **weakest link**
   - CPU, memory, controller, interconnect, or device
2. Configure that component to sustain required bandwidth
3. Scale remaining components to avoid new bottlenecks

Goal

- Meet latency bounds
- Maximize sustainable throughput
- Maintain balanced system design

Parallelism and I/O

Parallelism and I/O: Motivation

Why Parallelism in I/O?

- I/O devices are much slower than CPUs
- Single I/O path becomes performance bottleneck
- Modern workloads demand high throughput

Goals of I/O Parallelism

- Increase aggregate bandwidth
- Reduce effective latency
- Improve system utilization
- Avoid single weakest-link bottleneck

Sources of Parallelism

- Multiple devices
- Multiple I/O channels or controllers

Forms of I/O Parallelism

Device-Level Parallelism

- Multiple disks in RAID
- Multiple network interfaces
- Independent device queues

Controller-Level Parallelism

- Multiple DMA engines
- Independent I/O controllers
- Separate I/O buses or lanes

Request-Level Parallelism

- Multiple outstanding I/O requests
- Queue depth increases throughput
- Out-of-order completion

Overlapping and System-Level Parallelism

Computation and I/O Overlap

- CPU executes while DMA transfers data
- Interrupt signals completion
- Improves processor utilization

Pipeline Parallelism

- Stage 1: Issue command
- Stage 2: Transfer data
- Stage 3: Process result
- Multiple operations in flight

Challenges

- Contention on interconnect
- Synchronization overhead

Introduction to multicores and multiprocessors

Multicores and Multiprocessors: Motivation

Why Multiple Processors?

- Frequency scaling hit power and thermal limits
- Instruction-level parallelism is limited
- Workloads increasingly parallel

Goal

- Increase performance via parallel execution
- Improve throughput rather than single-thread speed

Key Distinction

- **Multicore:** Multiple cores on a single chip
- **Multiprocessor:** Multiple physical CPUs in one system

Shared Memory Multiprocessors

Basic Organization

- Multiple processors
- Shared main memory
- Interconnected via bus or network

Types

- **SMP (Uniform Memory Access)**
  - Equal memory access latency
- **NUMA (Non-Uniform Memory Access)**
  - Local vs remote memory latency differs

Key Issues

- Cache coherence
- Memory consistency model

Cache Coherence and Consistency

Cache Coherence Problem

- Multiple caches may hold same memory block
- Writes must be visible to all processors

Coherence Mechanisms

- Snooping protocols
- Directory-based protocols

Memory Consistency

- Defines order of memory operations
- Sequential consistency vs relaxed models

Critical for correctness in parallel programs.

Interconnect and Scalability

Interconnect Options

- Shared bus (limited scalability)
- Crossbar switch
- Ring, mesh, torus networks

Scalability Challenges

- Bandwidth contention
- Latency growth
- Power consumption

Design Tradeoff

- Core count vs memory bandwidth
- Coherence traffic vs performance gain

Dr. Laltu Sardar, Assistant Professor, IISER Thiruvananthapuram
`laltu.sardar@iisertvm.ac.in`, `laltu.sardar.crypto@gmail.com`

Course webpage: https://laltu-sardar.github.io/courses/corgos_2026.html.

Carl Hamacher, Zvonko Vranesic, and Safwat Zaky.
Computer Organization.
McGraw-Hill, 6th edition, 2012.

John L. Hennessy and David A. Patterson.
Computer Architecture: A Quantitative Approach.
Morgan Kaufmann, 6th edition, 2017.

Vincent P. Heuring and Harry F. Jordan.
Computer System Design and Architecture.
Pearson, 2nd edition, 2007.

David A. Patterson and John L. Hennessy.
Computer Organization and Design: The Hardware/Software Interface.
Morgan Kaufmann, 4th edition, 2014.

William Stallings.
Computer Organization and Architecture: Designing for Performance.
Pearson, 10th edition, 2016.