Akdeniz University
Computer Engineering Department

CSE206 Computer Organization
Week03: Computer Systems

Assoc.Prof.Dr. Taner Danışman
tdanisman@akdeniz.edu.tr

Course program (Textbook: Stalling 10th Edt.)

Week 1	19-Feb-24	Introduction	Ch1
Week 2	26-Feb-24	Computer Evolution	Ch2
Week 3	4-Mar-24	Computer Systems	Ch3
Week 4	11-Mar-24	Cache Memory, Direct Cache Mapping	Ch4
Week 5	18-Mar-24	Associative and Set Associative Mapping	Ch4
Week 6	25-Mar-24	Internal Memory, External Memory, I/O	Ch5-Ch6-Ch7
Week 7	1-Apr-24	Number Systems, Computer Arithmetic	Ch9-Ch10
Week 8	8-Apr-24	Midterm (Expected date, may change)	Ch1-…-Ch10
Week 9	15-Apr-24	Digital Logic	Ch11
Week 10	22-Apr-24	Instruction Sets	Ch12
Week 11	29-Apr-24	Addressing Modes	Ch13
Week 12	6-May-24	Processor Structure and Function	Ch14
Week 13	13-May-24	RISC, Inctruction Level Parallelism	Ch15-Ch16
Week 14	20-May-24	Assembly Language (TextBook : Assembly Language for x86 Processors)	Kip Irvine
Week 15	27-May-24	Assembly Language (TextBook : Assembly Language for x86 Processors)	Kip Irvine

Computer Components

- Contemporary computer designs are based on concepts developed by John von Neumann at the Institute for Advanced Studies, Princeton
- **Referred to as the *von Neumann architecture* and is based on three key concepts:**
  - Data and instructions are stored in a single read-write memory
  - The contents of this memory are addressable by location, without regard to the type of data contained there
  - Execution occurs in a sequential fashion (unless explicitly modified) from one instruction to the next
- ***Hardwired program***
  - The result of the process of connecting the various components in the desired configuration

Hardware and Software Approaches

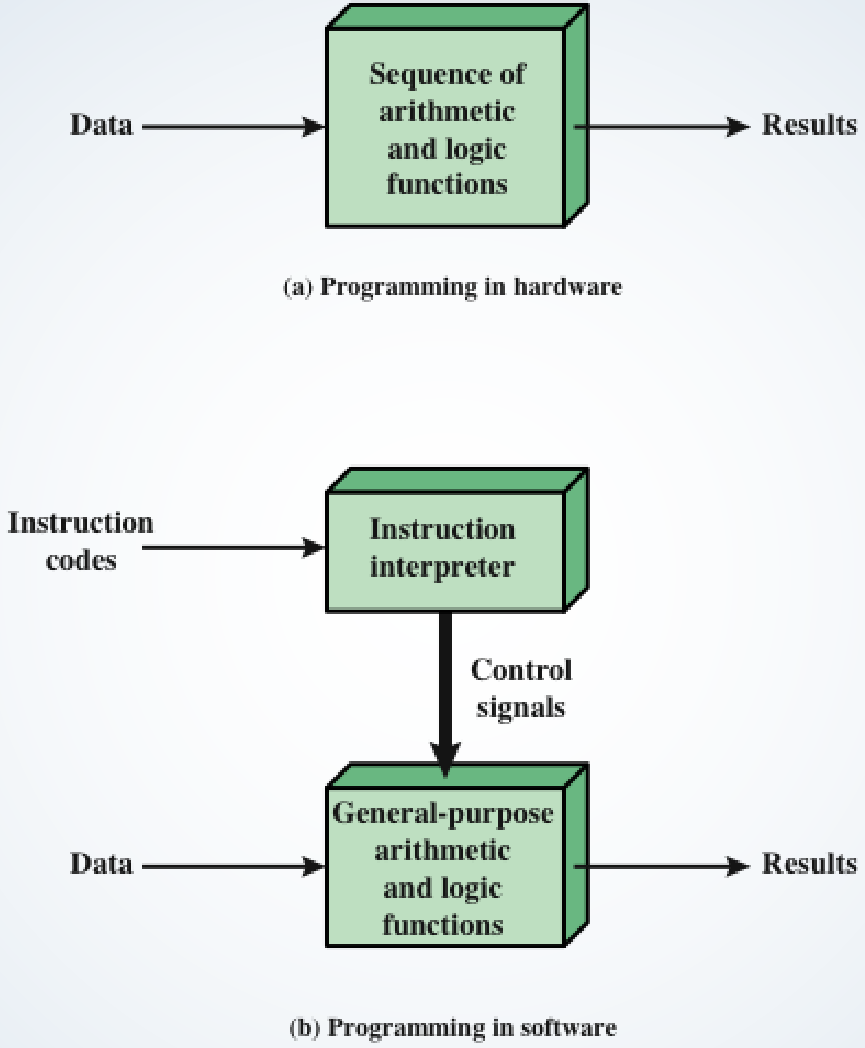


Figure 3.1 Hardware and Software Approaches

I/O Components

Software

- A sequence of codes or instructions
- Part of the hardware interprets each instruction and generates control signals
- Provide a new sequence of codes for each new program instead of rewiring the hardware

Major components:

- CPU
  - Instruction interpreter
  - Module of general-purpose arithmetic and logic functions
- I/O Components
  - Input module
    - Contains basic components for accepting data and instructions and converting them into an internal form of signals usable by the system
  - Output module
    - Means of reporting results

MEMORY

Memory address register (MAR)

- Specifies the address in memory for the next read or write

Memory buffer register (MBR)

- Contains the data to be written into memory or receives the data read from memory

I/O address register (I/OAR)

- Specifies a particular I/O device

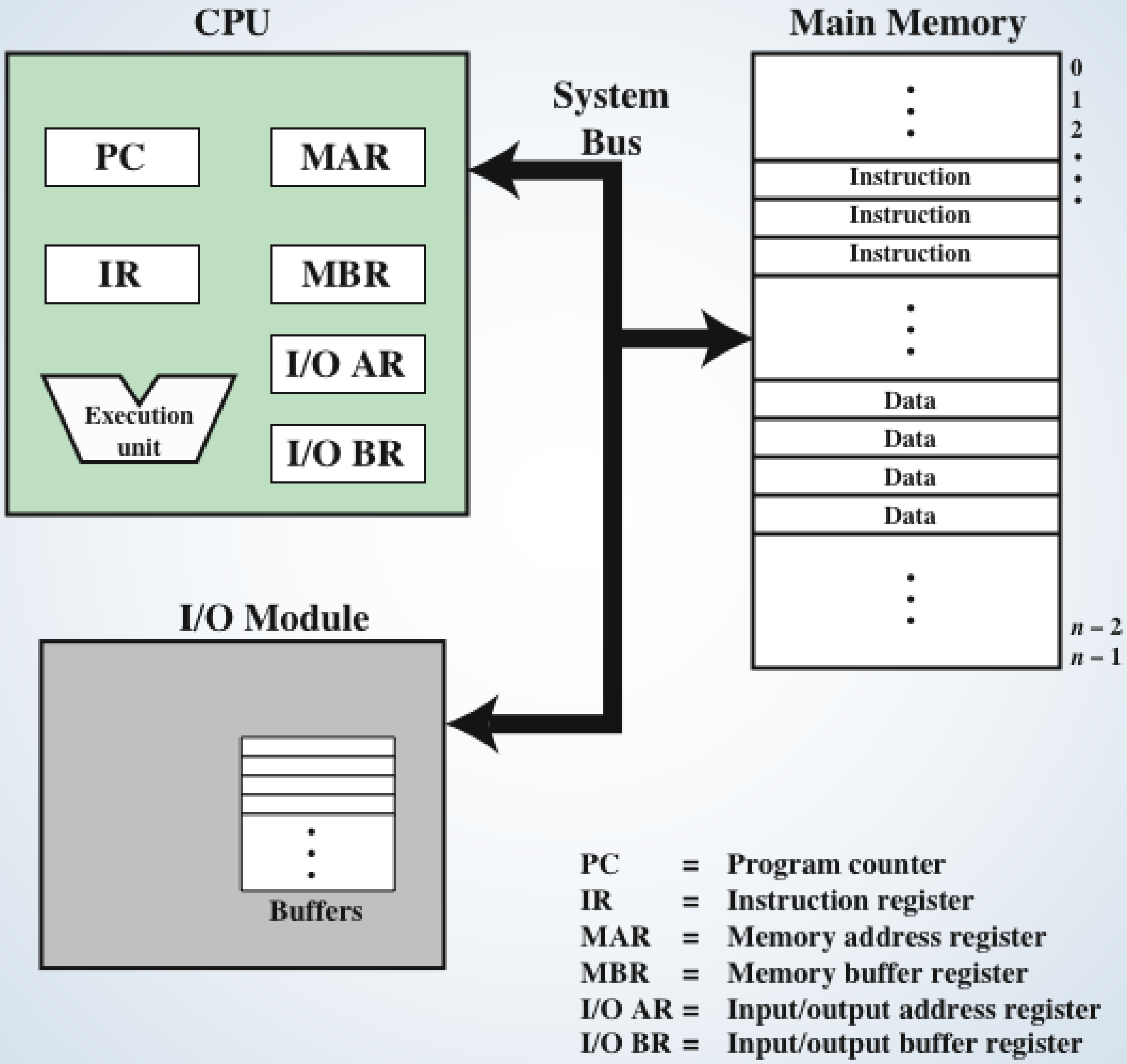
I/O buffer register (I/OBR)

- Used for the exchange of data between an I/O module and the CPU

MAR

MBR

Computer Components: Top Level View



Basic Instruction Cycle

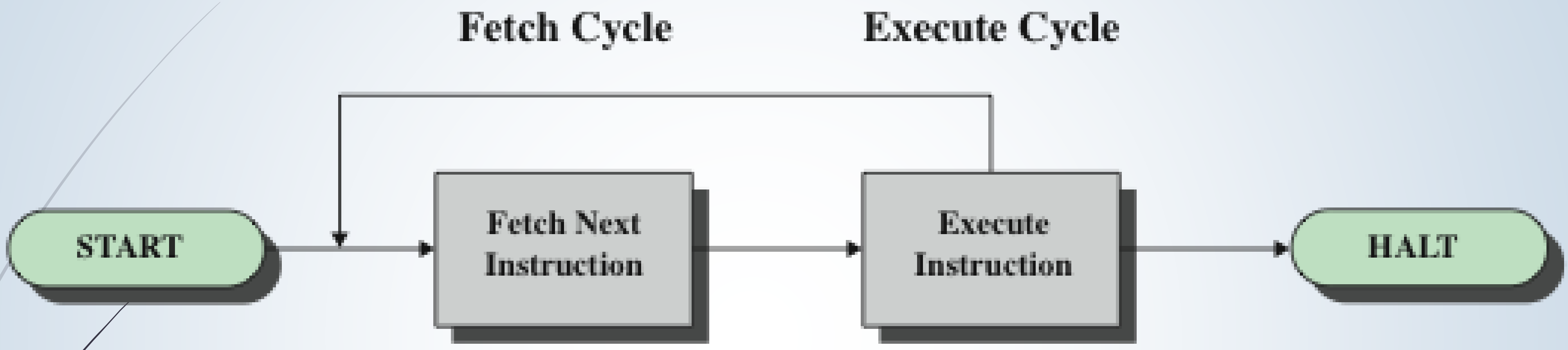


Figure 3.3 Basic Instruction Cycle

Fetch Cycle

- At the beginning of each instruction cycle the processor fetches an instruction from memory
- The **program counter** (PC) holds the address of the instruction to be fetched next
- The processor increments the PC after each instruction fetch so that it will fetch the next instruction in sequence
- The fetched instruction is loaded into the **instruction register** (IR)
- The processor interprets the instruction and performs the required action

Action Categories

Processor-memory

Data transferred from processor to memory or from memory to processor

Processor-I/O

Data transferred to or from a peripheral device by transferring between the processor and an I/O module

Data processing

The processor may perform some arithmetic or logic operation on data

Control

An instruction may specify that the sequence of execution be altered

Characteristics of a Hypothetical Machine

(a) Instruction format

(b) Integer format

Program Counter (PC) = Address of instruction
Instruction Register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

(c) Internal CPU registers

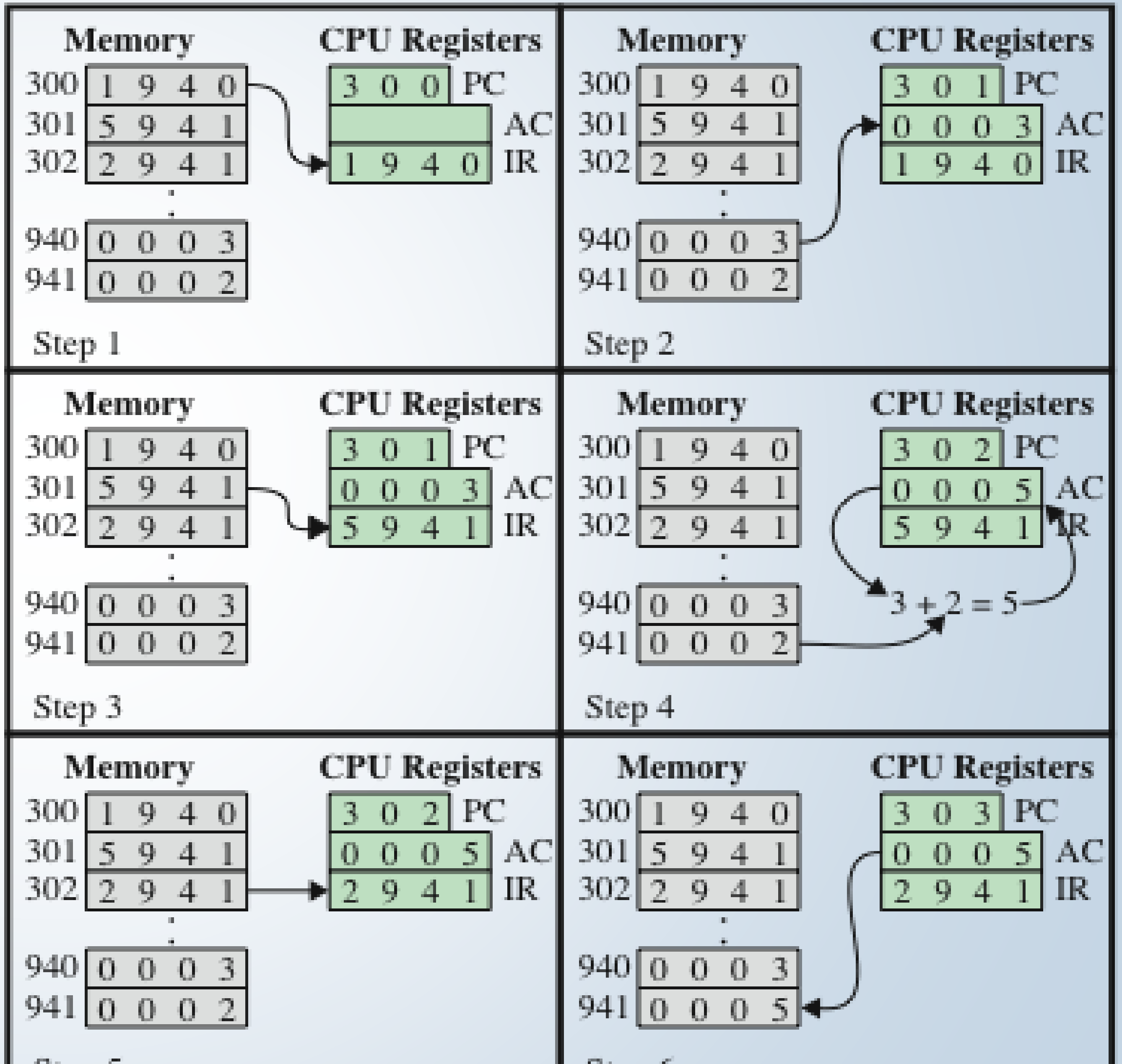
0001 = Load AC from Memory
0010 = Store AC to Memory
0101 = Add to AC from Memory

(d) Partial list of opcodes

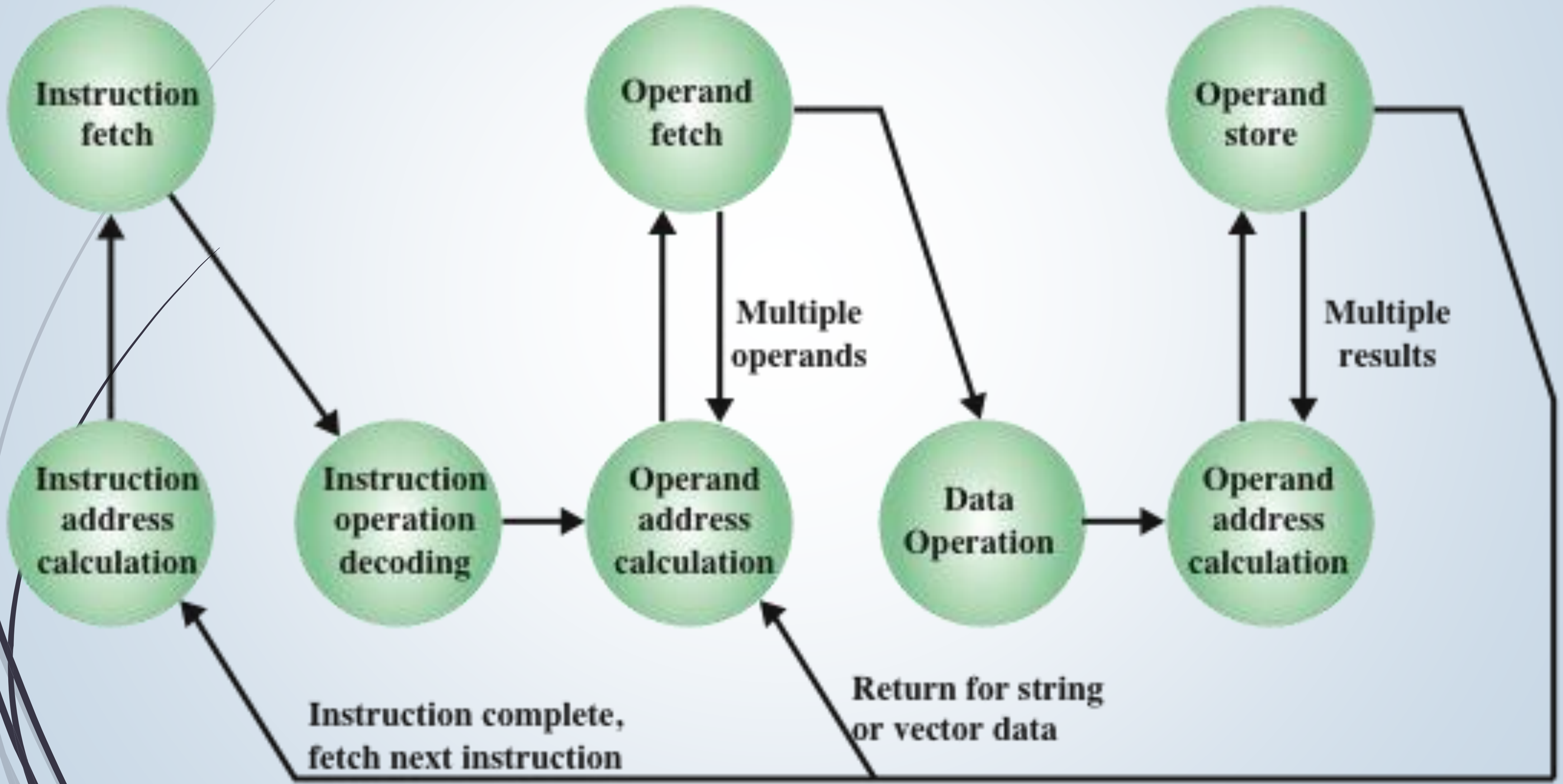
Example of Program Execution

Figure 3.5 Example of Program Execution (contents of memory and registers in hexadecimal)

0001 = Load AC from Memory
0010 = Store AC to Memory
0101 = Add to AC from Memory



Instruction Cycle State Diagram



Classes of Interrupts

Program	Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, or reference outside a user's allowed memory space.
Timer	Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis.
I/O	Generated by an I/O controller, to signal normal completion of an operation, request service from the processor, or to signal a variety of error conditions.
Hardware failure	Generated by a failure such as power failure or memory parity error.

Program Flow Control

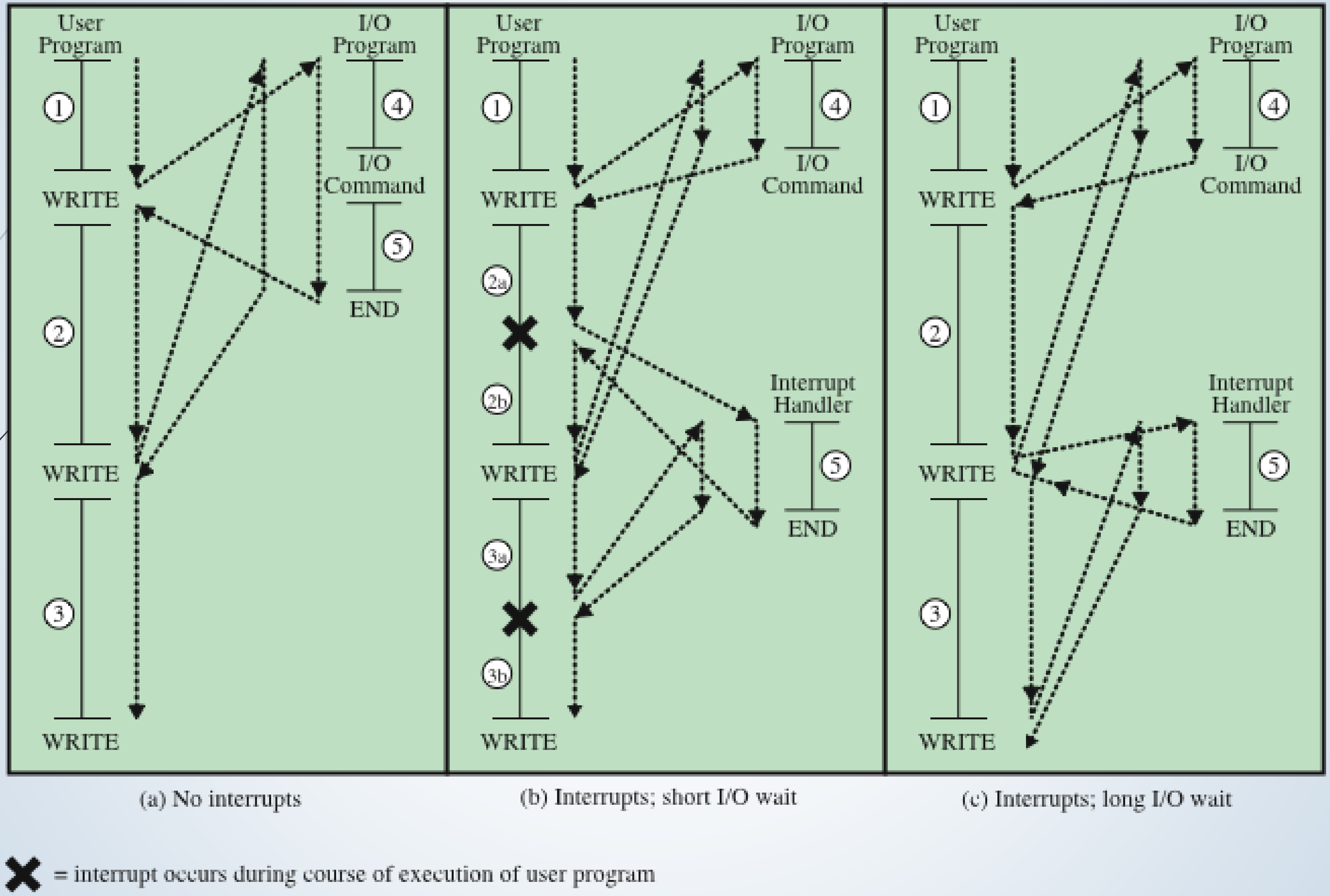


Figure 3.7 Program Flow of Control Without and With Interrupts

Transfer of Control via Interrupts

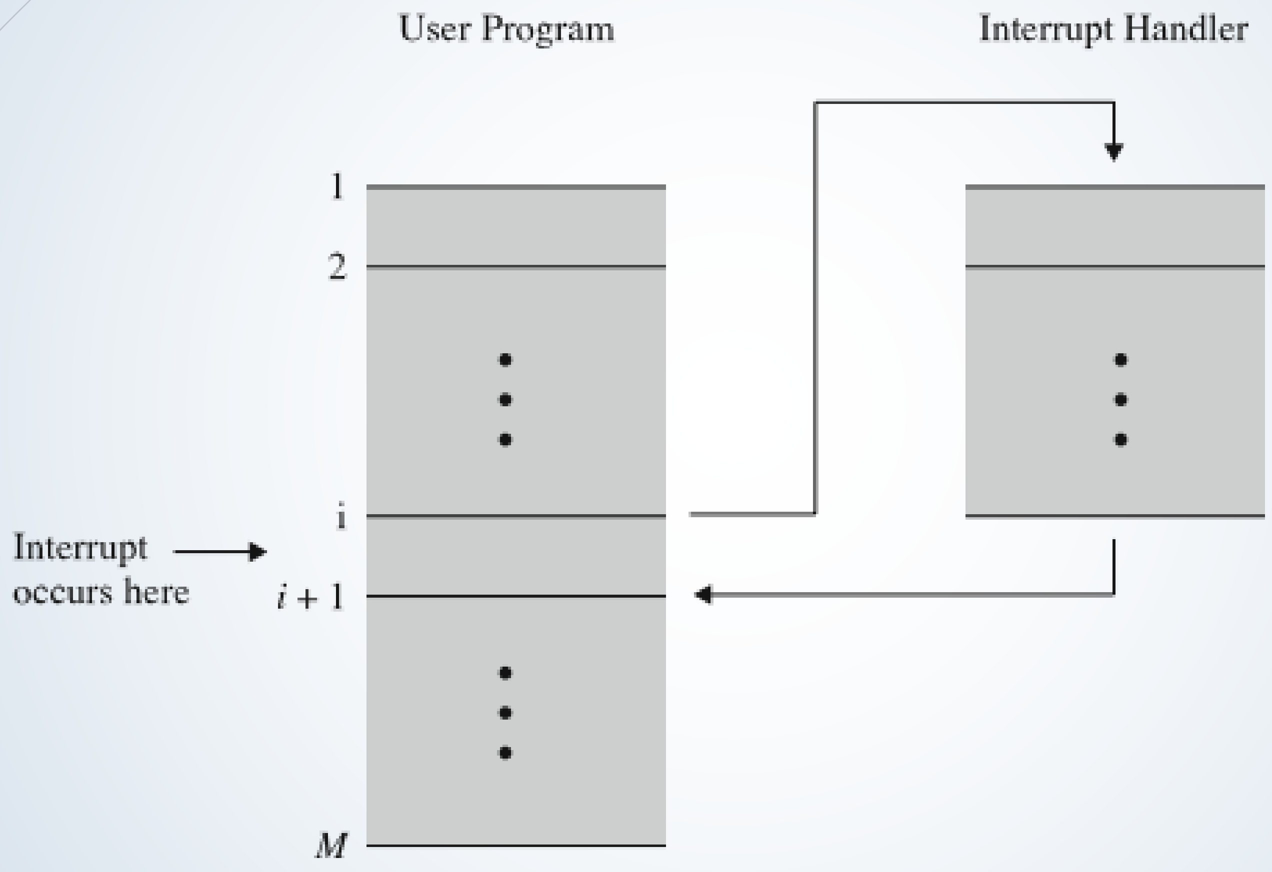


Figure 3.8 Transfer of Control via Interrupts

Instruction Cycle With Interrupts

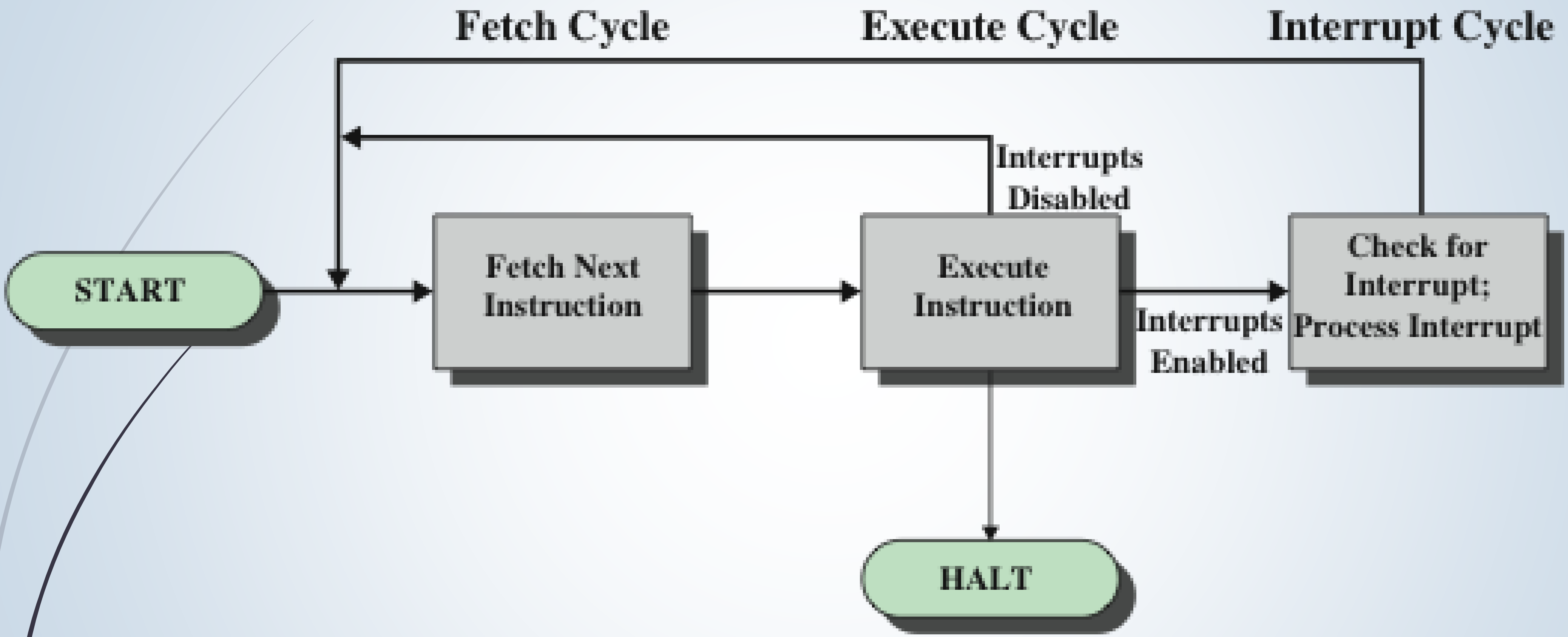
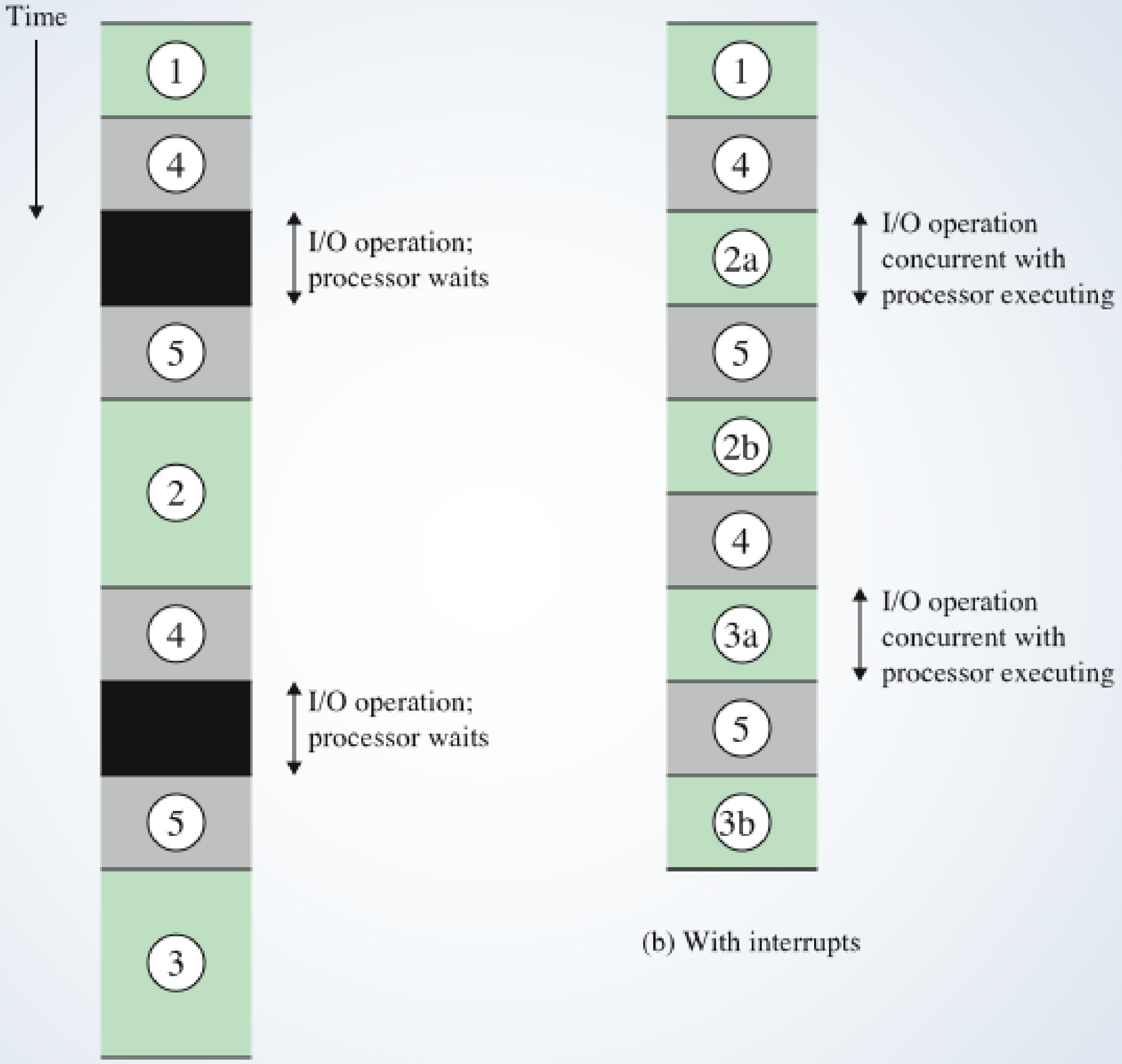


Figure 3.9 Instruction Cycle with Interrupts

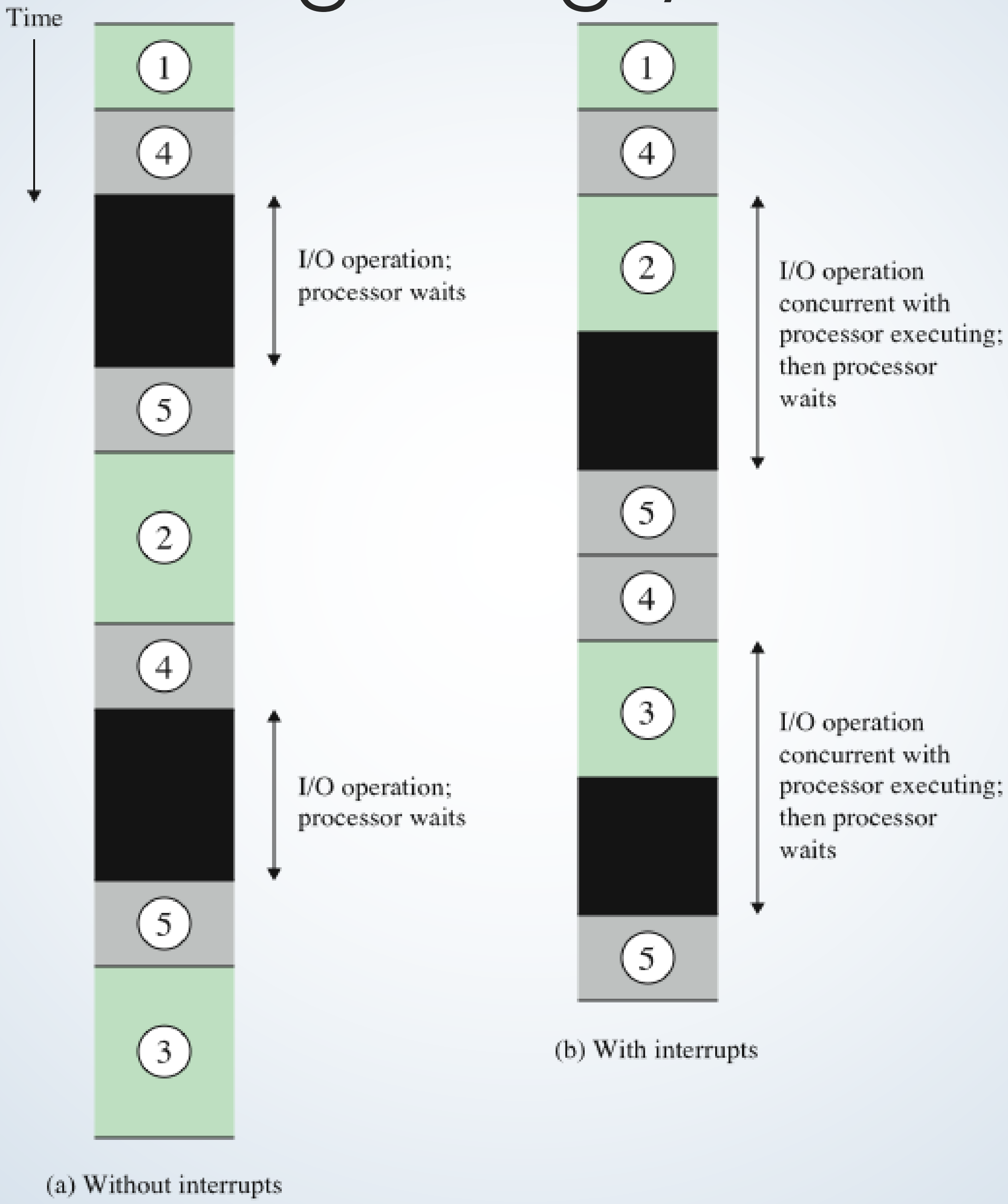
Program Timing: Short I/O Wait



(a) Without interrupts

(b) With interrupts

Program Timing: Long I/O Wait



Instruction Cycle State Diagram With Interrupts

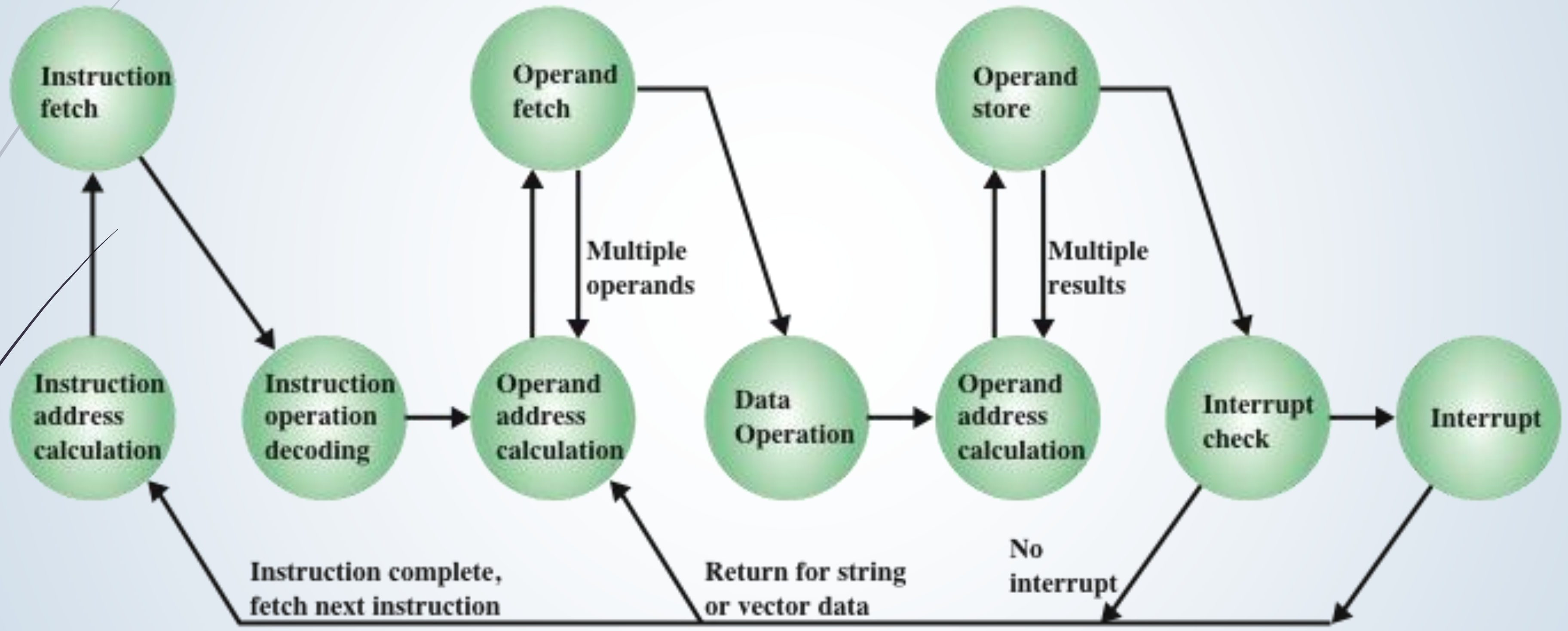


Figure 3.12 Instruction Cycle State Diagram, With Interrupts

Transfer of Control with Multiple Interrupts

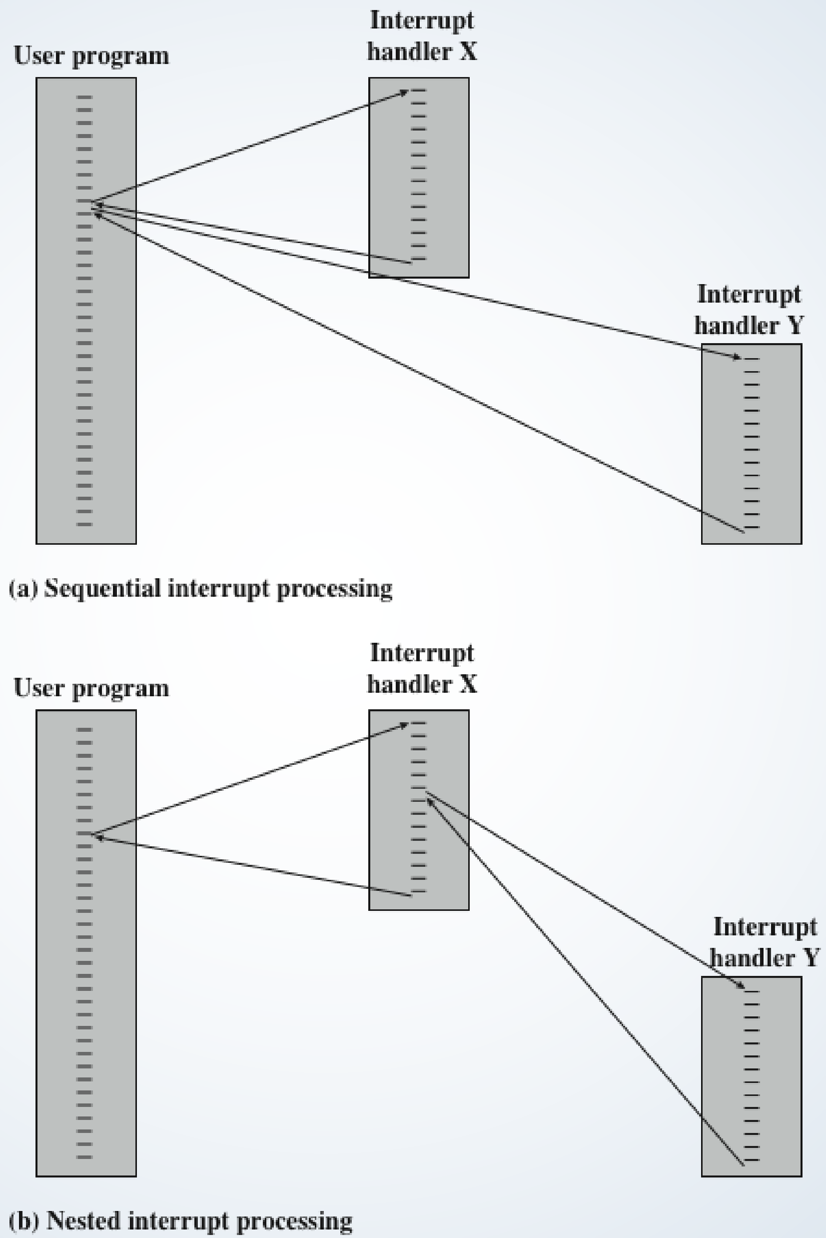
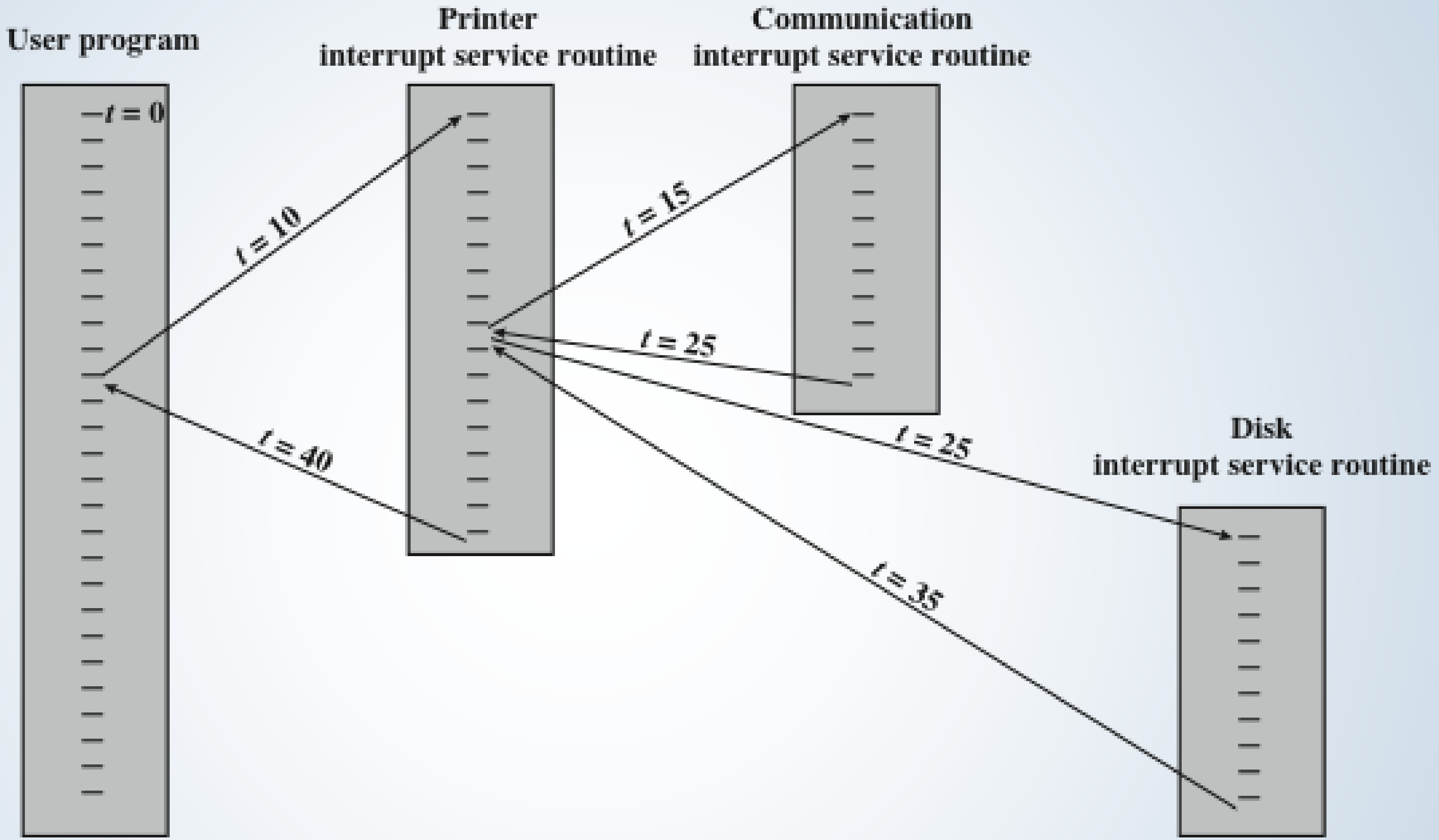


Figure 3.13 Transfer of Control with Multiple Interrupts

Time Sequence of Multiple Interrupts



Priorities: Communication > Disk > Printer
Interrupts: t=10 Printer, t=15 Communication, t=20 Disk

Figure 3.14 Example Time Sequence of Multiple Interrupts

I/O Function

- I/O module can exchange data directly with the processor
- Processor can read data from or write data to an I/O module
  - Processor identifies a specific device that is controlled by a particular I/O module
  - I/O instructions rather than memory referencing instructions
- In some cases it is desirable to allow I/O exchanges to occur directly with memory
  - The processor grants to an I/O module the authority to read from or write to memory so that the I/O memory transfer can occur without tying up the processor
  - The I/O module issues read or write commands to memory relieving the processor of responsibility for the exchange
  - This operation is known as direct memory access (DMA)

Computer Modules

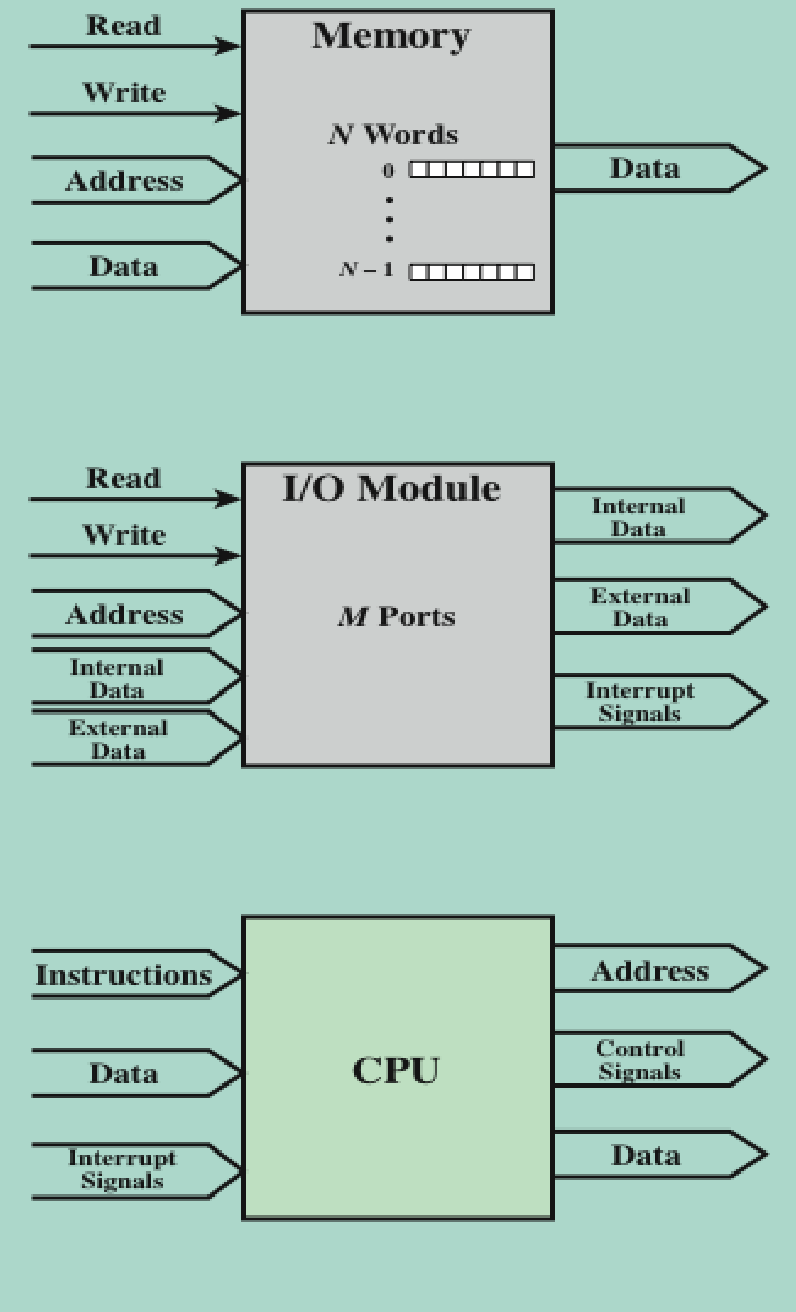


Figure 3.15 Computer Modules

The interconnection structure must support the following types of transfers:

Memory to processor

Processor reads an instruction or a unit of data from memory

Processor to memory

Processor writes a unit of data to memory

I/O to processor

Processor reads data from an I/O device via an I/O module

Processor to I/O

Processor sends data to the I/O device

I/O to or from memory

An I/O module is allowed to exchange data directly with memory without going through the processor using direct memory access

Bus Interconnection

A communication pathway connecting two or more devices

- Key characteristic is that it is a shared transmission medium

Signals transmitted by any one device are available for reception by all other devices attached to the bus

- If two devices transmit during the same time period their signals will overlap and become garbled

Typically consists of multiple communication lines

- Each line is capable of transmitting signals representing binary 1 and binary 0

Computer systems contain a number of different buses that provide pathways between components at various levels of the computer system hierarchy

System bus

- A bus that connects major computer components (processor, memory, I/O)

The most common computer interconnection structures are based on the use of one or more system buses

Data Bus

- Data lines that provide a path for moving data among system modules
- May consist of 32, 64, 128, or more separate lines
- The number of lines is referred to as the *width* of the data bus
- The number of lines determines how many bits can be transferred at a time
- The width of the data bus is a key factor in determining overall system performance

Address Bus

- Used to designate the source or destination of the data on the data bus
  - If the processor wishes to read a word of data from memory it puts the address of the desired word on the address lines
- Width determines the maximum possible memory capacity of the system
- Also used to address I/O ports
  - The higher order bits are used to select a particular module on the bus and the lower order bits select a memory location or I/O port within the module

Control Bus

- Used to control the access and the use of the data and address lines
- Because the data and address lines are shared by all components there must be a means of controlling their use
- Control signals transmit both command and timing information among system modules
- Timing signals indicate the validity of data and address information
- Command signals specify operations to be performed

Bus Interconnection Scheme

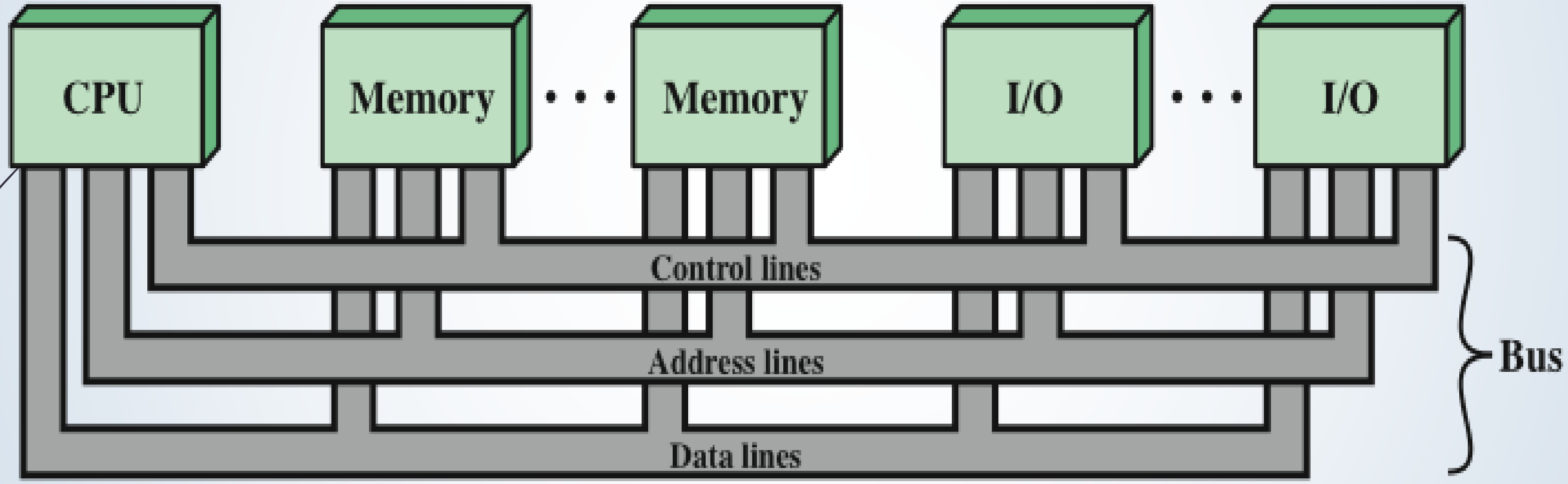


Figure 3.16 Bus Interconnection Scheme

Bus Configurations

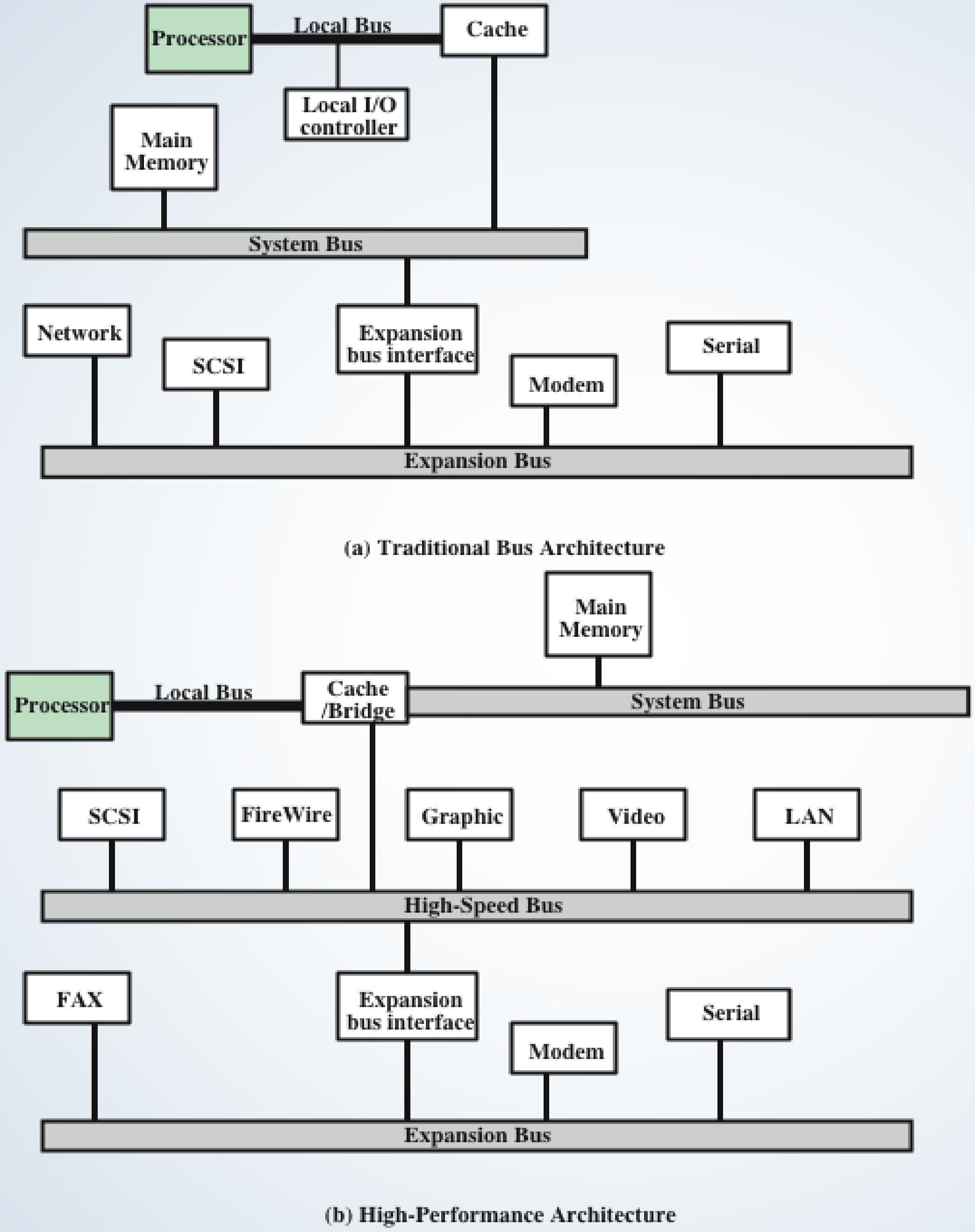


Figure 3.17 Example Bus Configurations

Elements of Bus Design

Type	**Bus Width**
Dedicated	Address
Multiplexed	Data
Method of Arbitration	**Data Transfer Type**
Centralized	Read
Distributed	Write
Timing	Read-modify-write
Synchronous	Read-after-write
Asynchronous	Block

Timing of Synchronous Bus Operations

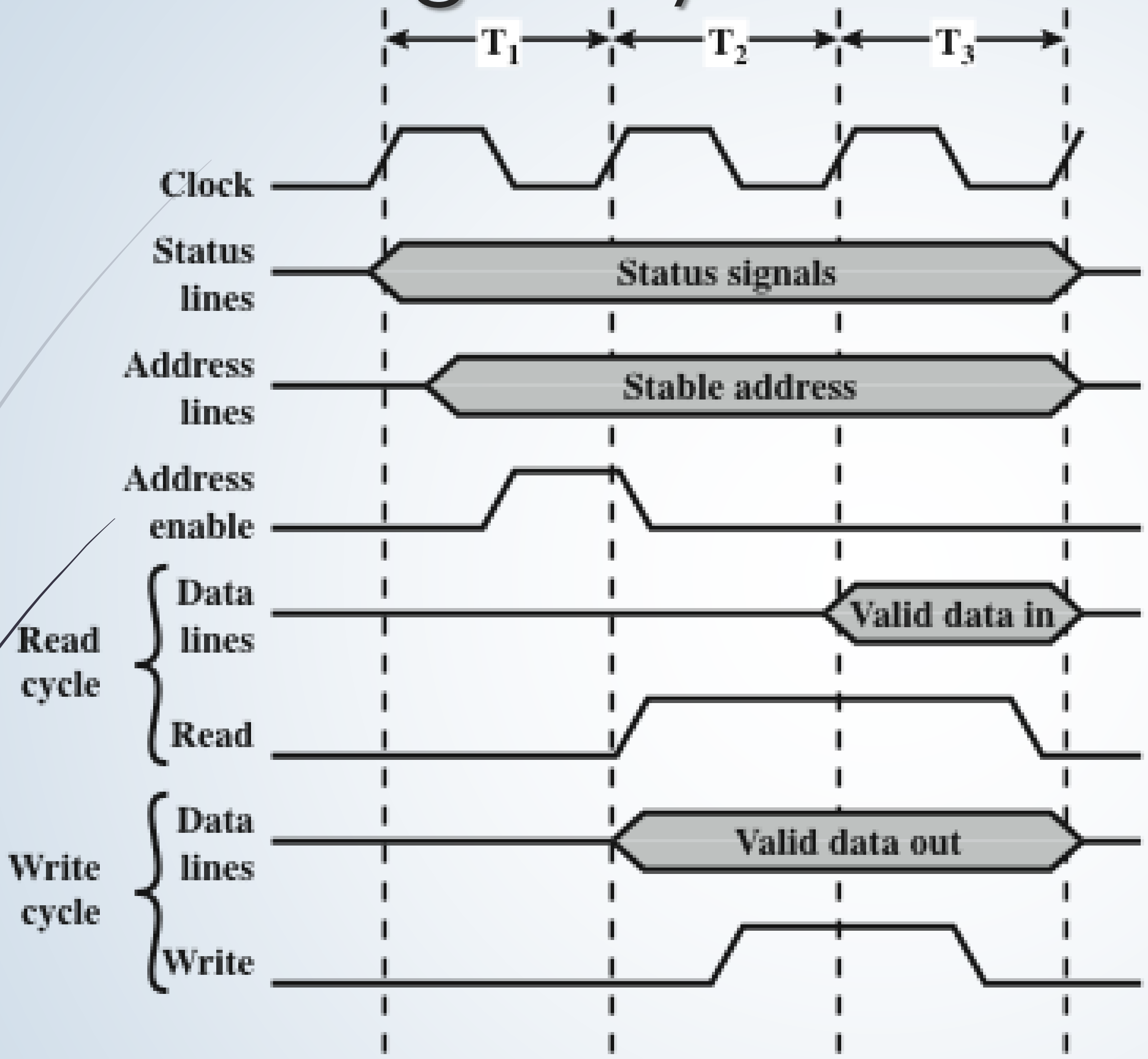


Figure 3.18 Timing of Synchronous Bus Operations

Timing of Asynchronous Bus Operations

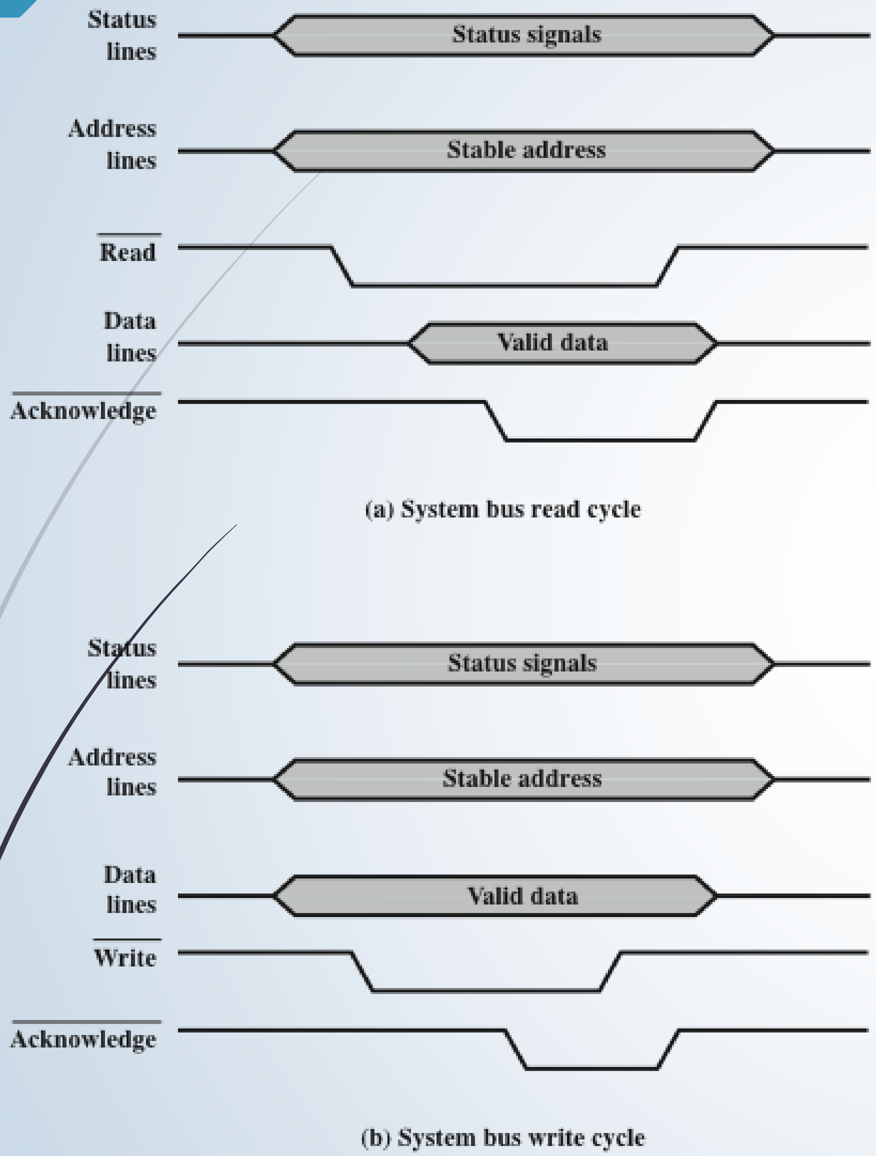


Figure 3.19 Timing of Asynchronous Bus Operations

Point-to-Point Interconnect

Principal reason for change was the electrical constraints encountered with increasing the frequency of wide synchronous buses

At higher and higher data rates it becomes increasingly difficult to perform the synchronization and arbitration functions in a timely fashion

A conventional shared bus on the same chip magnified the difficulties of increasing bus data rate and reducing bus latency to keep up with the processors

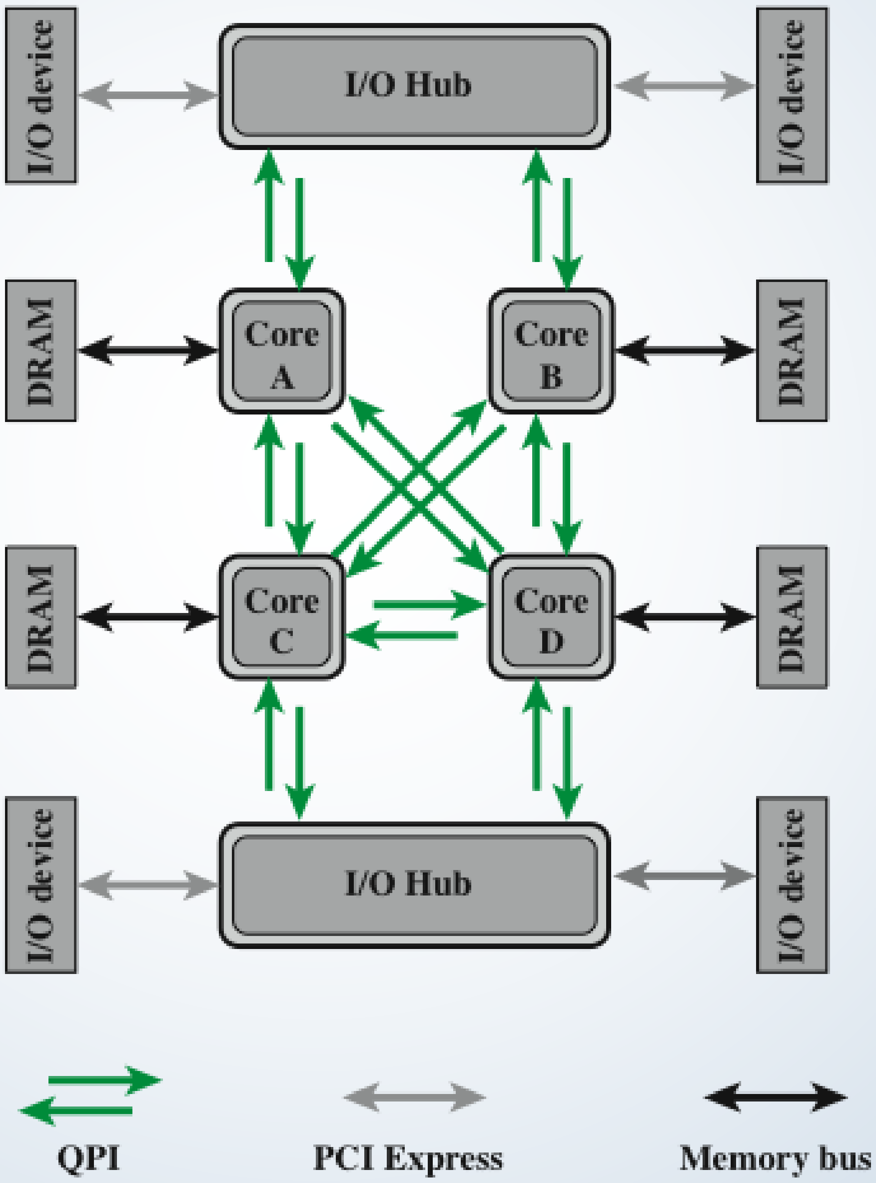
Has lower latency, higher data rate, and better scalability

Quick Path Interconnect

QPI

- Introduced in 2008
- Multiple direct connections
  - Direct pairwise connections to other components eliminating the need for arbitration found in shared transmission systems
- Layered protocol architecture
  - These processor level interconnects use a layered protocol architecture rather than the simple use of control signals found in shared bus arrangements
- Packetized data transfer
  - Data are sent as a sequence of packets each of which includes control headers and error control codes

Multicore Configuration Using QPI



QPI Layers

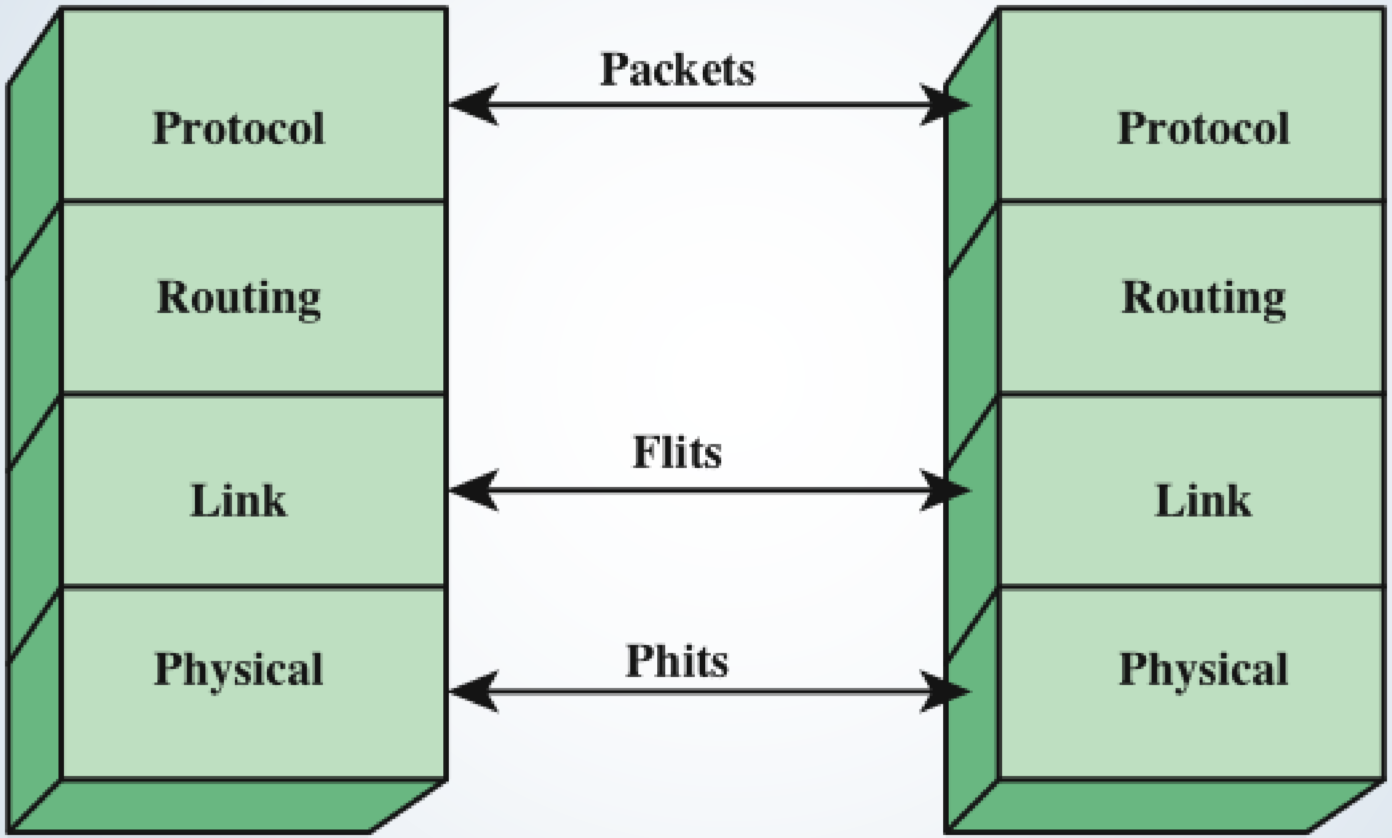


Figure 3.21 QPI Layers

Physical Interface of the Intel QPI Interconnect

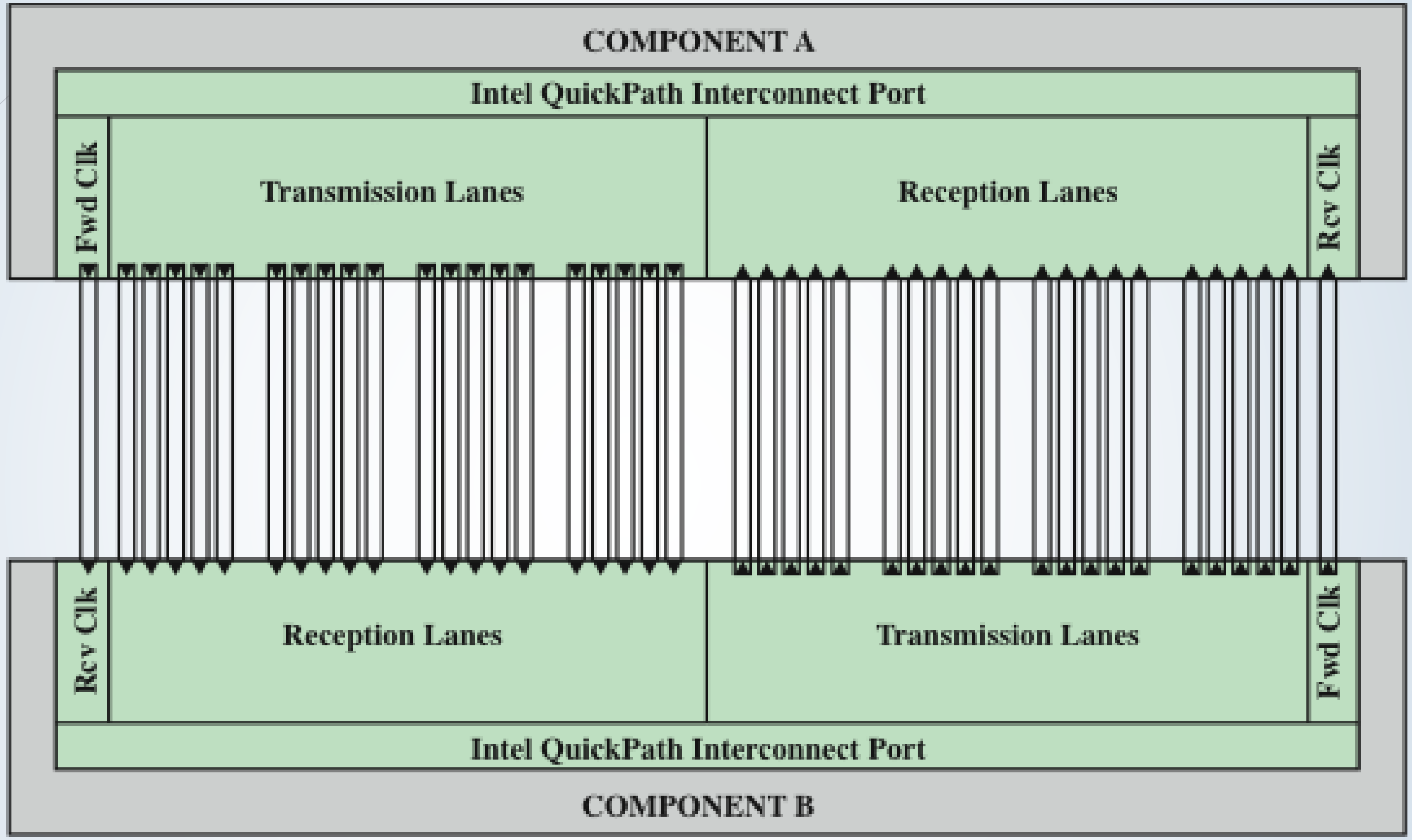


Figure 3.22 Physical Interface of the Intel QPI Interconnect

QPI Link Layer

- Performs two key functions: *flow control* and *error control*
  - Operate on the level of the flit (flow control unit)
  - Each flit consists of a 72-bit message payload and an 8-bit error control code called a *cyclic redundancy check* (CRC)

- Flow control function
  - Needed to ensure that a sending QPI entity does not overwhelm a receiving QPI entity by sending data faster than the receiver can process the data and clear buffers for more incoming data

- Error control function
  - Detects and recovers from bit errors, and so isolates higher layers from experiencing bit errors

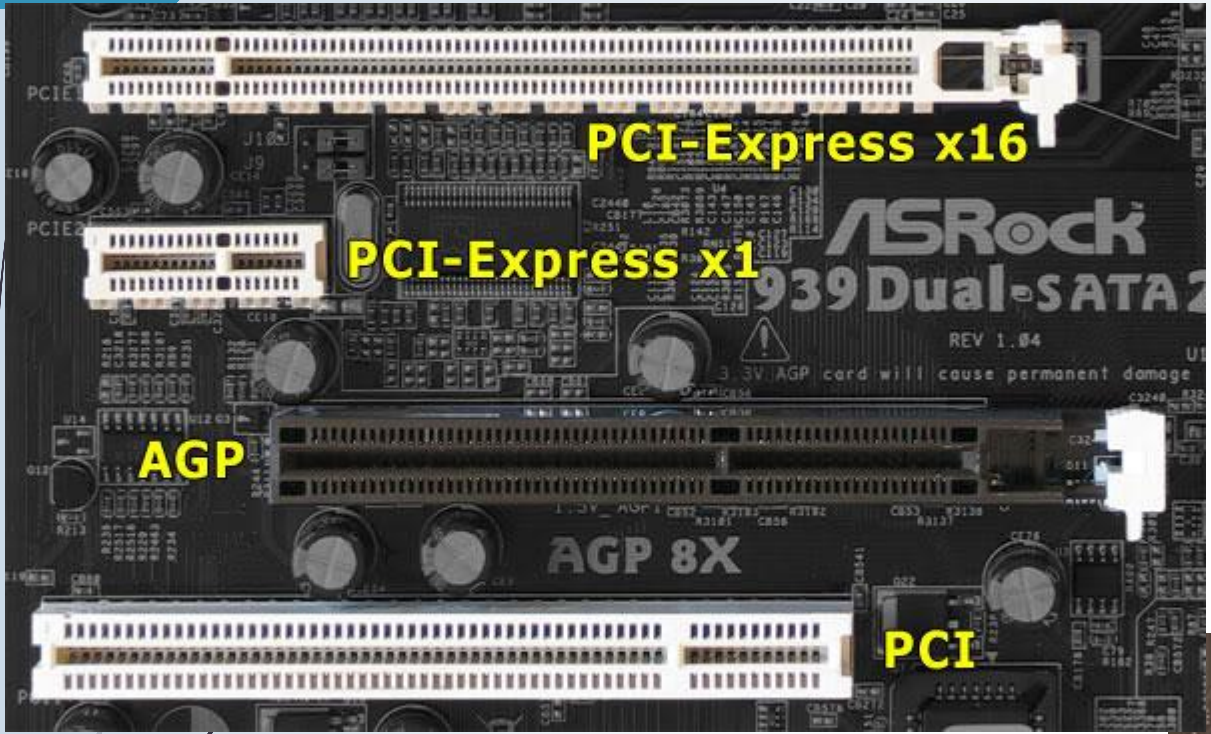
QPI Routing and Protocol Layers

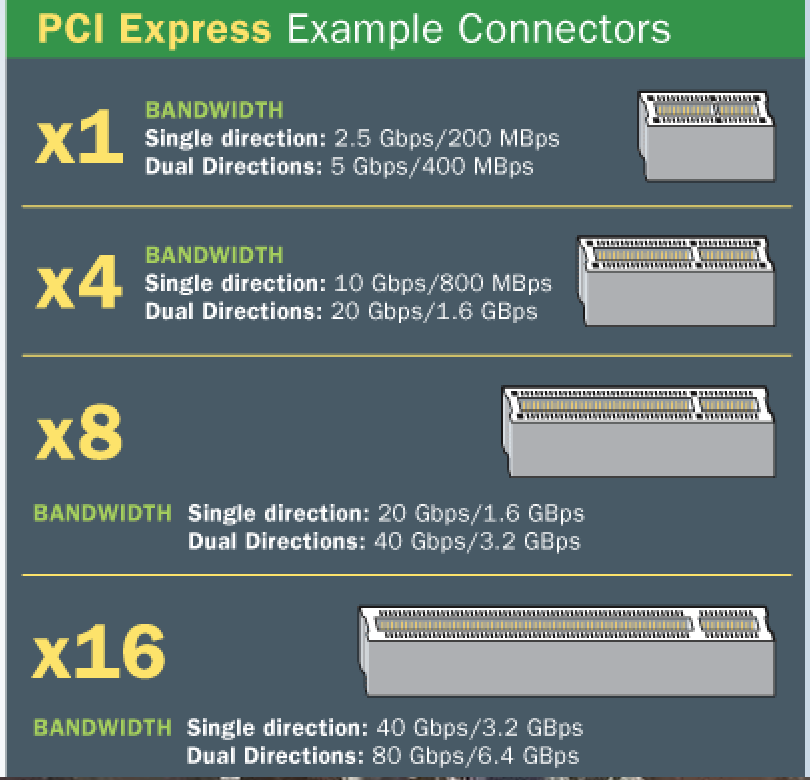
- Routing Layer
- Used to determine the course that a packet will traverse across the available system interconnects
- Defined by firmware and describe the possible paths that a packet can follow

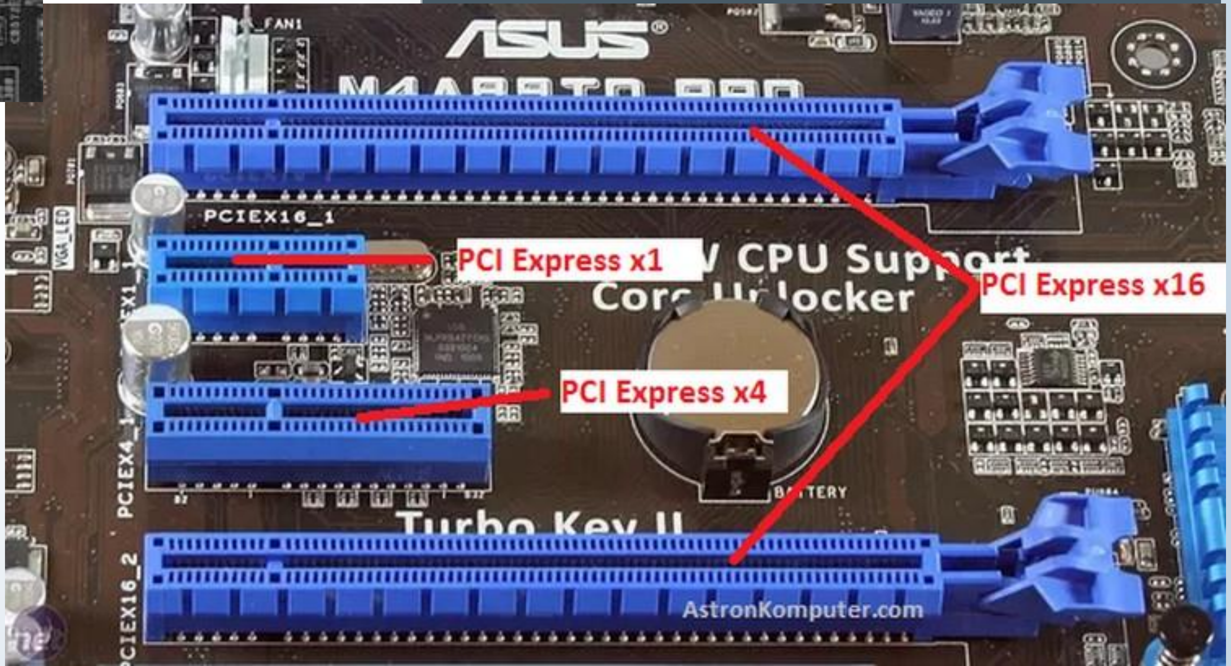
- Protocol Layer
- Packet is defined as the unit of transfer
- One key function performed at this level is a cache coherency protocol which deals with making sure that main memory values held in multiple caches are consistent
- A typical data packet payload is a block of data being sent to or from a cache

Peripheral Component Interconnect (PCI)

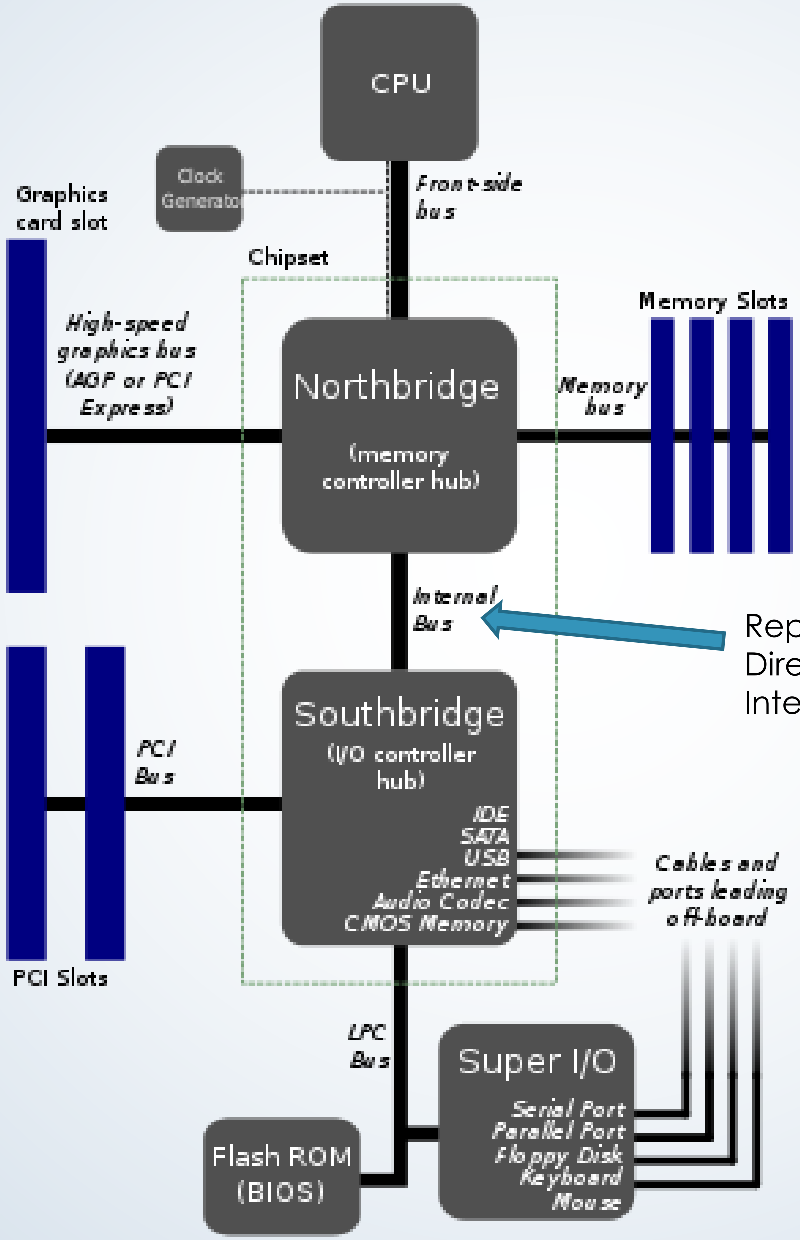
- A popular high bandwidth, processor independent bus that can function as a mezzanine or peripheral bus
- Delivers better system performance for high speed I/O subsystems
- PCI Special Interest Group (SIG)
  - Created to develop further and maintain the compatibility of the PCI specifications
- PCI Express (PCIe)
  - Point-to-point interconnect scheme intended to replace bus-based schemes such as PCI
  - Key requirement is high capacity to support the needs of higher data rate I/O devices, such as Gigabit Ethernet
  - Another requirement deals with the need to support time dependent data streams


PCI-Express x16
PCI-Express x1
ASRock
939Dual-SATA2
REV 1.04
3.3V AGP card will cause permanent damage
AGP
AGP 8X
PCI

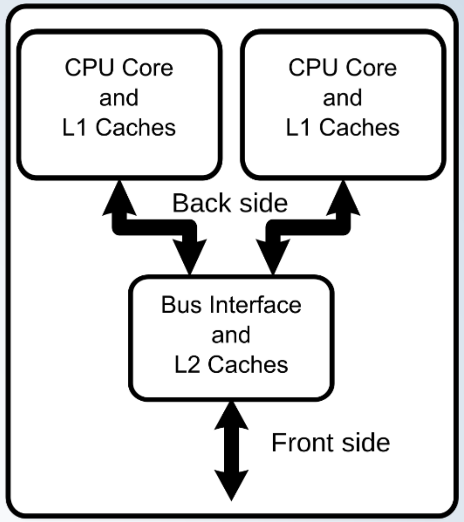

PCI Express Example Connectors
x1
BANDWIDTH
Single direction: 2.5 Gbps/200 MBps
Dual Directions: 5 Gbps/400 MBps
x4
BANDWIDTH
Single direction: 10 Gbps/800 MBps
Dual Directions: 20 Gbps/1.6 GBps
x8
BANDWIDTH
Single direction: 20 Gbps/1.6 GBps
Dual Directions: 40 Gbps/3.2 GBps
x16
BANDWIDTH
Single direction: 40 Gbps/3.2 GBps
Dual Directions: 80 Gbps/6.4 GBps

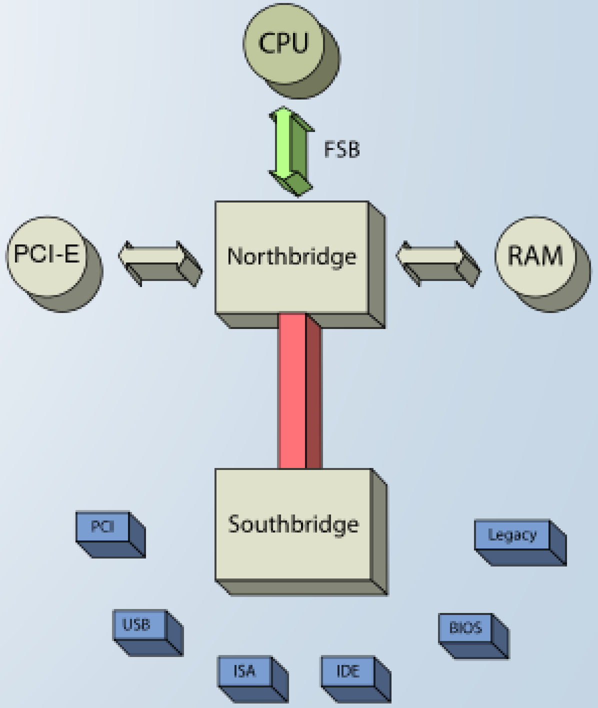

ASUS
PCIEX16_1
PCI Express x1
PCI Express x16
PCI Express x4
PCIEX4
PCIEX16_2
AstronKomputer.com

Computer Organization



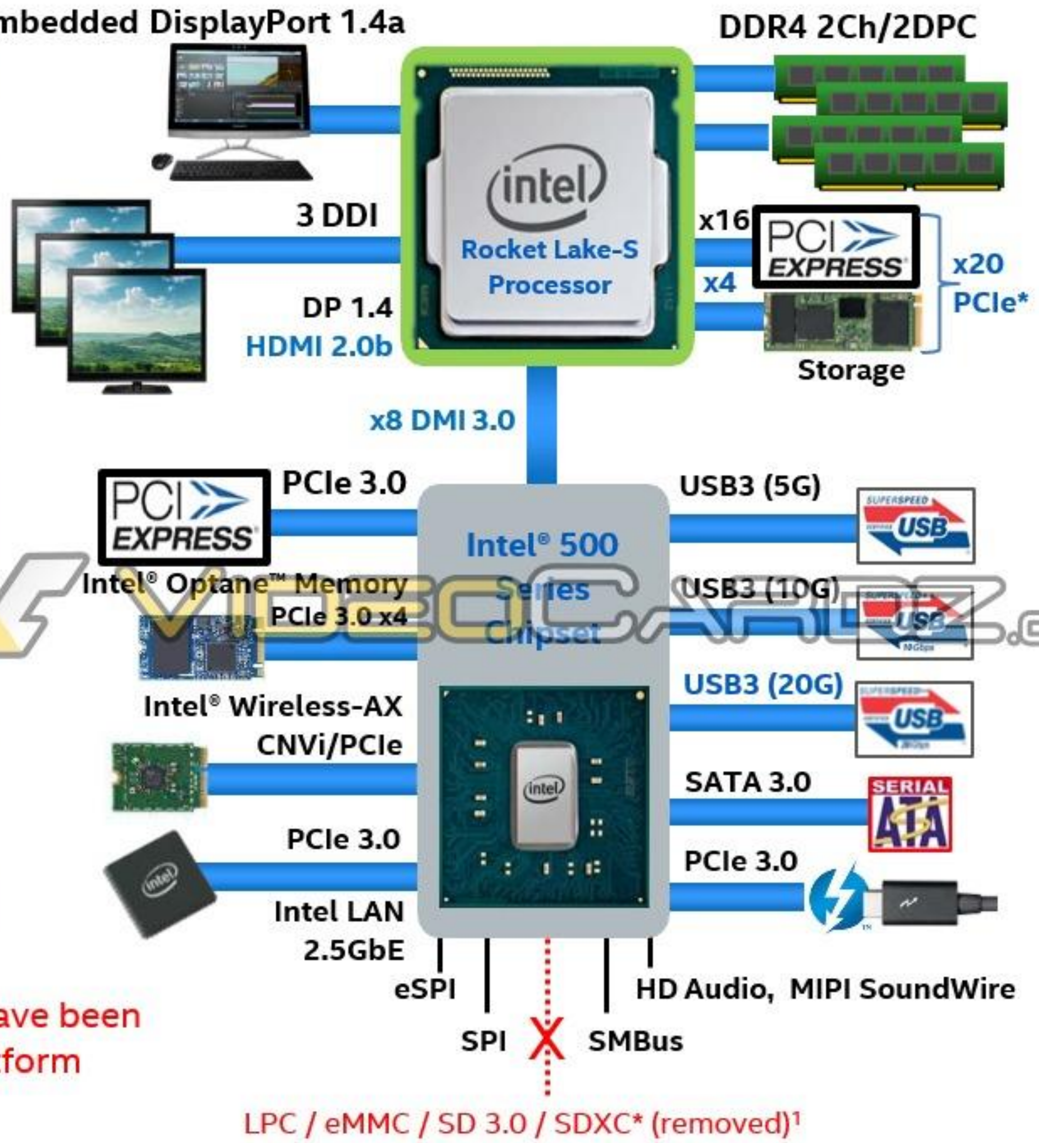
Replaced by Direct Media Interface





Up to date Microarchitecture examples

- Increased performance with new processor core architecture
- New X^e graphics architecture
- Enhanced Display (Integrated HDMI 2.0, HBR3)
- Enhanced Media (12bit AV1/HEVC, E2E compression)
- New Overclocking Features and Capabilities
- Integrated CNVi & Wireless-AX
- 2.5Gb Ethernet Discrete LAN



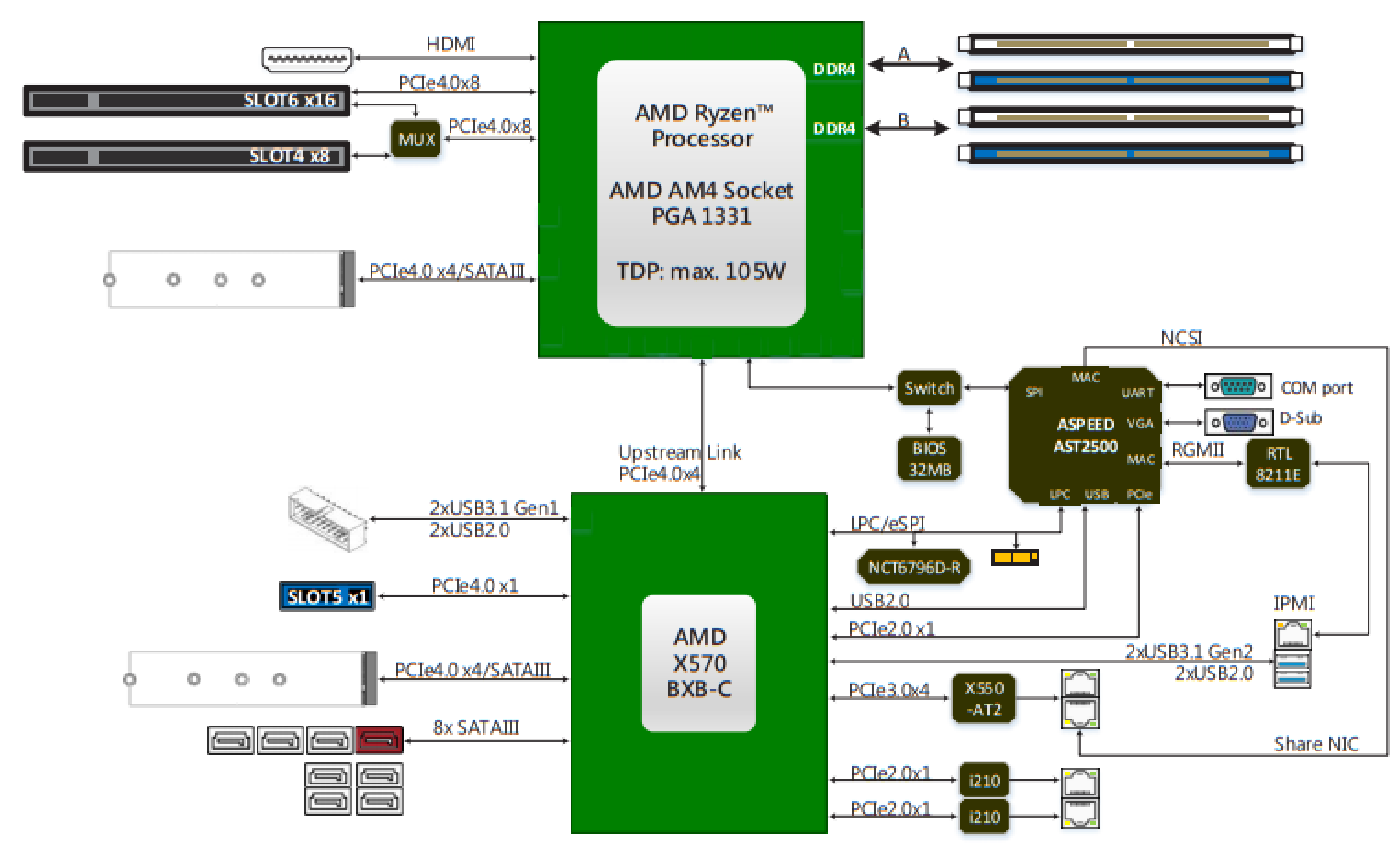
Intel® Software Guard Extensions (SGX) have been removed from Rocket Lake-S CPUs & platform

- Increased DDR4 speed
- CPU PCIe* 4.0 lanes
- Added x4 CPU PCIe* lanes = 20 total CPU PCIe 4.0 lanes
- CPU Attached Storage or Intel® Optane™ Memory
- USB Audio Offload
- Integrated USB 3.2 Gen 2x2 (20G)
- Discrete Intel® Thunderbolt 4 (USB4 compliant)

New with Rocket Lake-S Platform

1) Intel® Rocket Lake-S Chipset does not support the LPC, eMMC, SD3.0 or SDXC interfaces.

Up to date Microarchitecture examples



PCIe Configuration

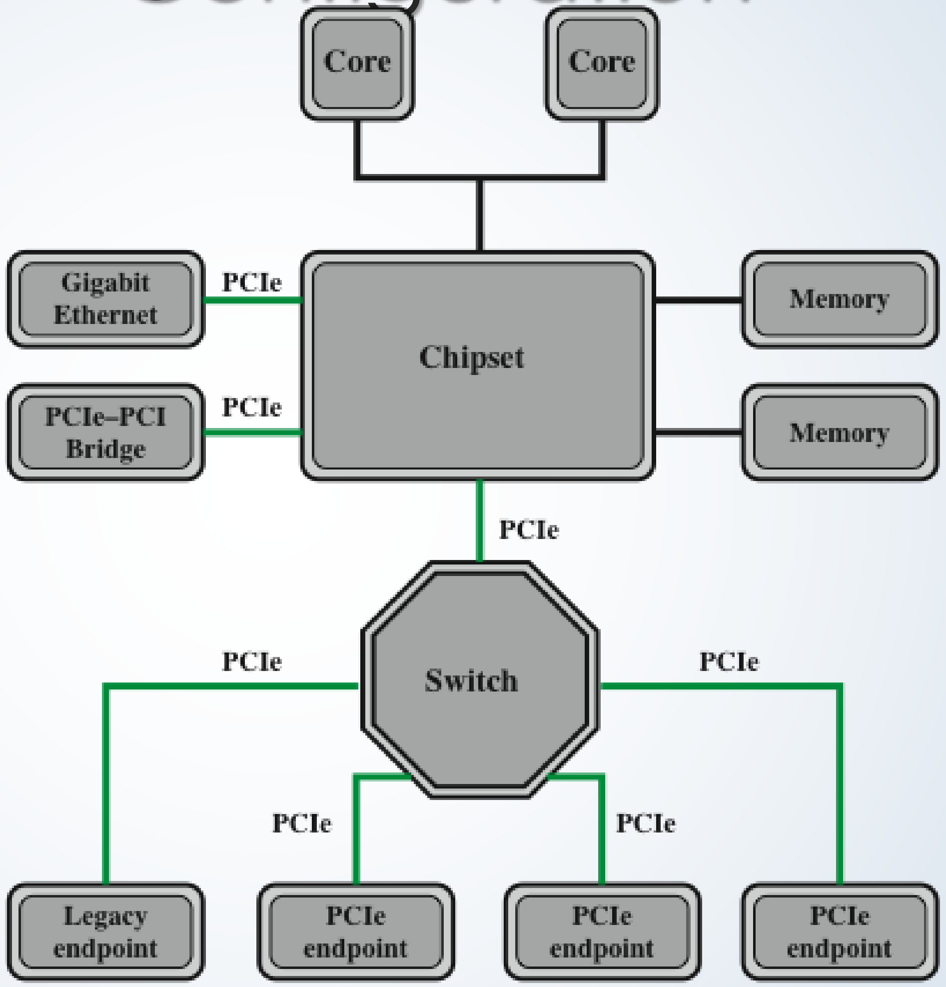


Figure 3.24 Typical Configuration Using PCIe

PCIe Protocol Layers

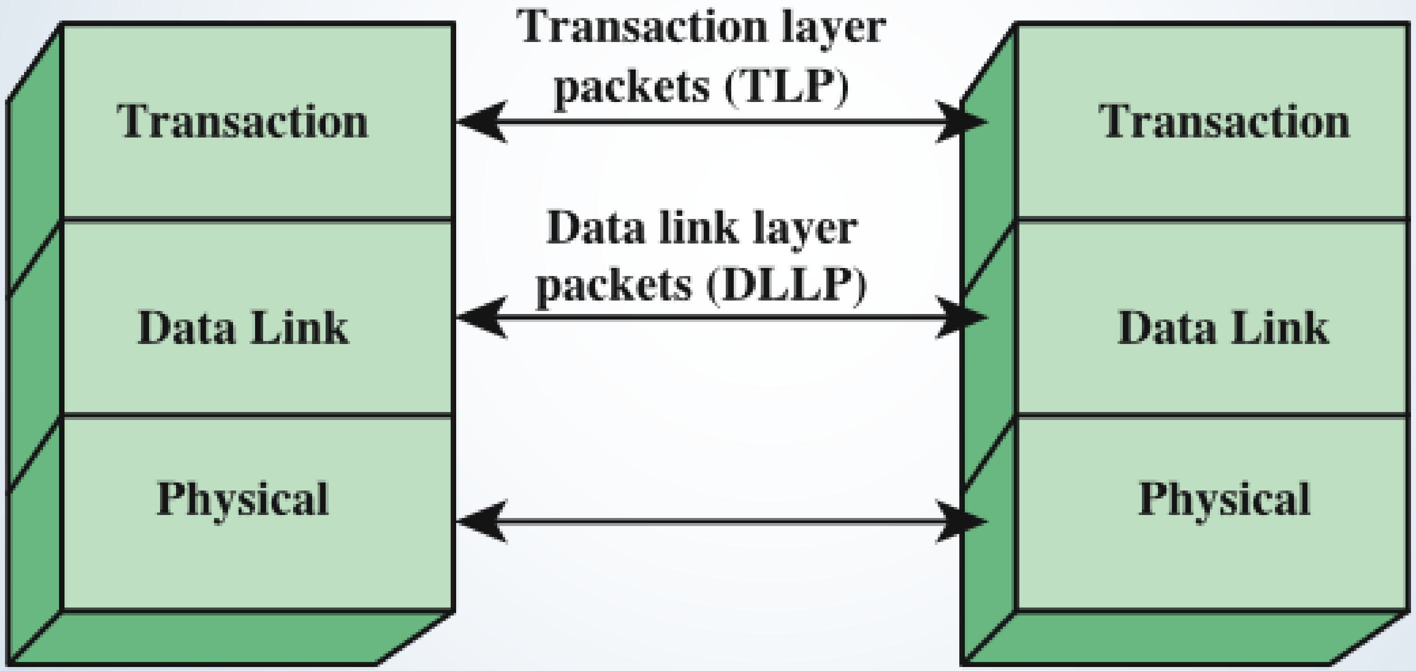


Figure 3.25 PCIe Protocol Layers

PCIe Protocol Data Unit Format

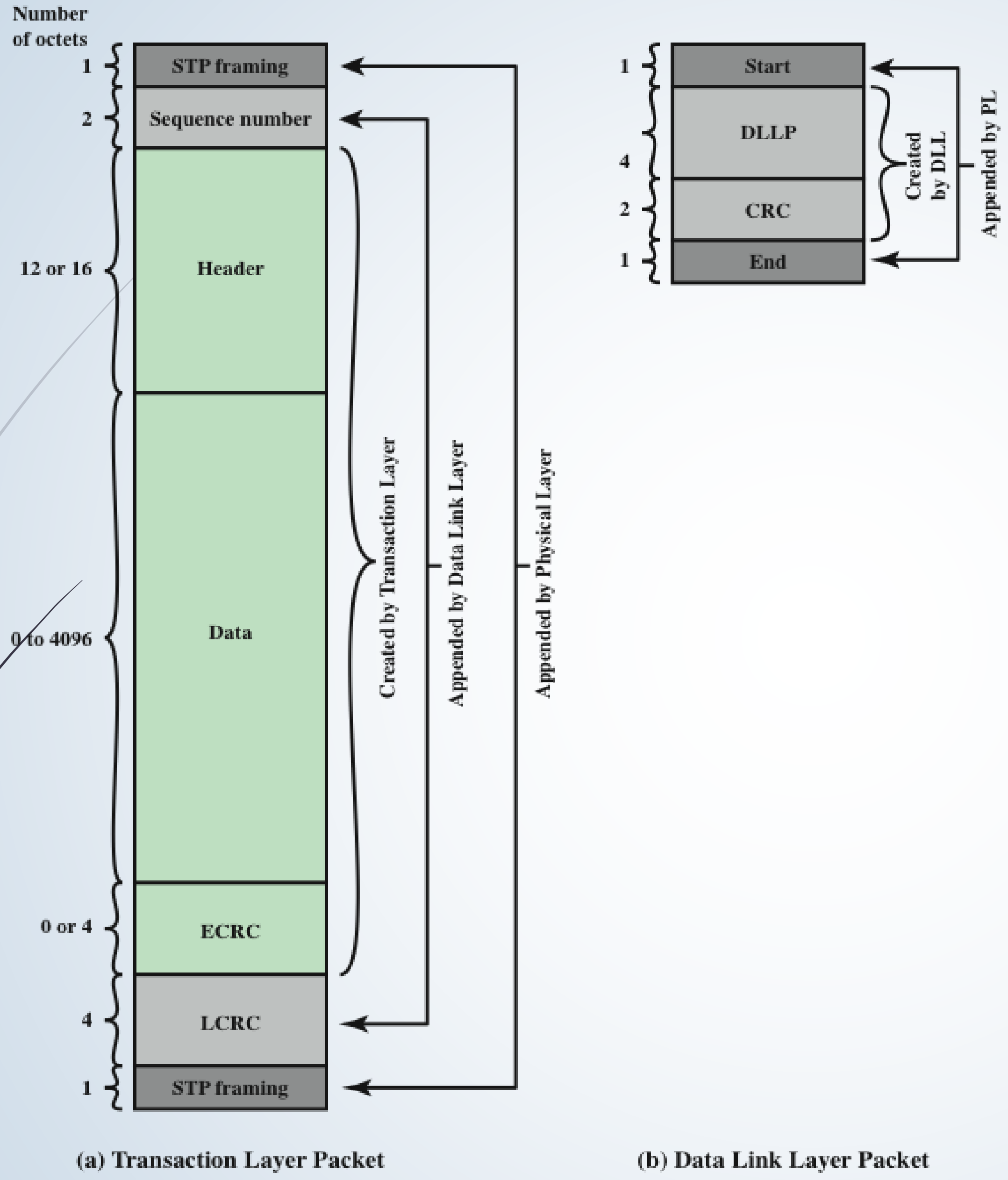


Figure 3.28 PCIe Protocol Data Unit Format

Summary

- Chapter 3

- Computer components
- Computer function
  - Instruction fetch and execute
  - Interrupts
  - I/O function
- Interconnection structures
- Bus interconnection
  - Bus structure
  - Multiple bus hierarchies
  - Elements of bus design

- A Top-Level View of Computer Function and Interconnection

- Point-to-point interconnect
  - QPI physical layer
  - QPI link layer
  - QPI routing layer
  - QPI protocol layer
- PCI express
  - PCI physical and logical architecture
  - PCIe physical layer
  - PCIe transaction layer
  - PCIe data link layer