Chapter 2

IA-32 Processor Architecture

Chapter 2

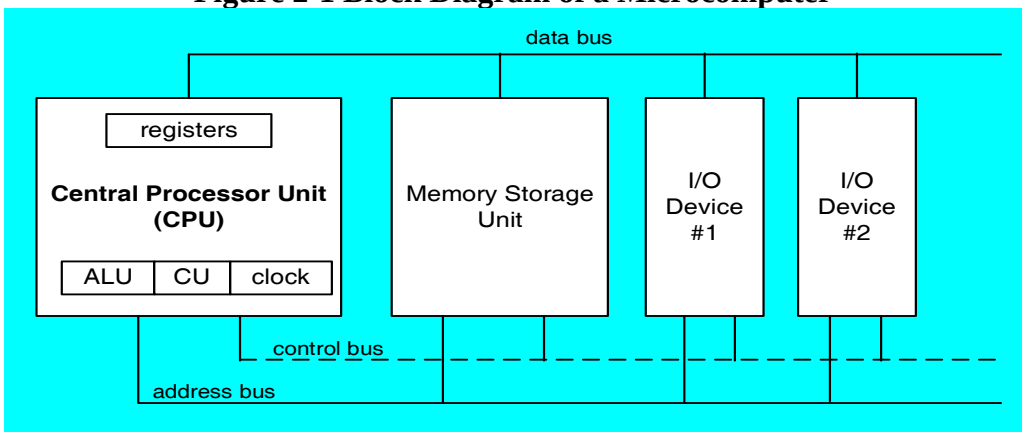
IA-32 Processor Architecture

Objectives

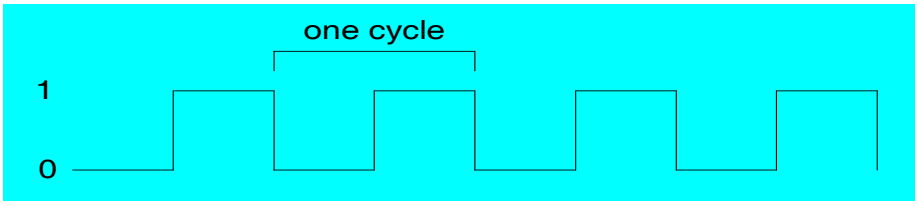
- Understand the basic structure of a microcomputer
- Be familiar with the instruction execution cycle
- Understand how computers read from memory
- Understand how the operating system loads and executes programs
- Know the modes of operand and basic execution environment of the IA-32 processors
- Be familiar with the floating-point unit and the history of Intel Processors
- Understand how memory is addressed in protected mode and real-address mode
- Know the basic components of a microcomputer
- Understand the different levels of input-output

- The Central Processor Unit (CPU), where calculations and logic operations take place, contains a limited number of storage locations name registers, a high-frequency clock, a control unit, and an arithmetic logic unit.
  - **Registers**: Storage locations
  - **Clock**: The clock **synchronizes** the internal operations of the CPU with other system components.
  - **CU**: The Control Unit (CU) **coordinates** the sequencing of steps involved in executing machine instructions.
  - **ALU**: The Arithmetic Logic Unit (ALU) performs **arithmetic operations** such as addition and subtraction and **logical operations** such as AND, OR, NOT.

Figure 2-1 Block Diagram of a Microcomputer



- **Memory Storage Unit**: The Memory Storage Unit is where instructions and data are held while a computer program is running.
- **Bus**: A bus is group of parallel wires that transfer data from one part of the computer to another. A computer's system bus usually consists of there separate buses:
  - **Data bus**: The data bus transfers **instructions and data** between the CPU and memory.
  - **Control bus**: The control bus uses binary signals **synchronize** actions of all device attached to the system bus.
  - **Address bus**: The address bus holds the **addresses** of instructions and data when the currently executing instruction transfers data between the CPU and memory.
- **Clock**: Each operation involving the CPU and the system bus is synchronized by an internal clock pulsing at a constant rate.
  - The basic unit of time for machine instruction is a machine cycle (or clock cycle).

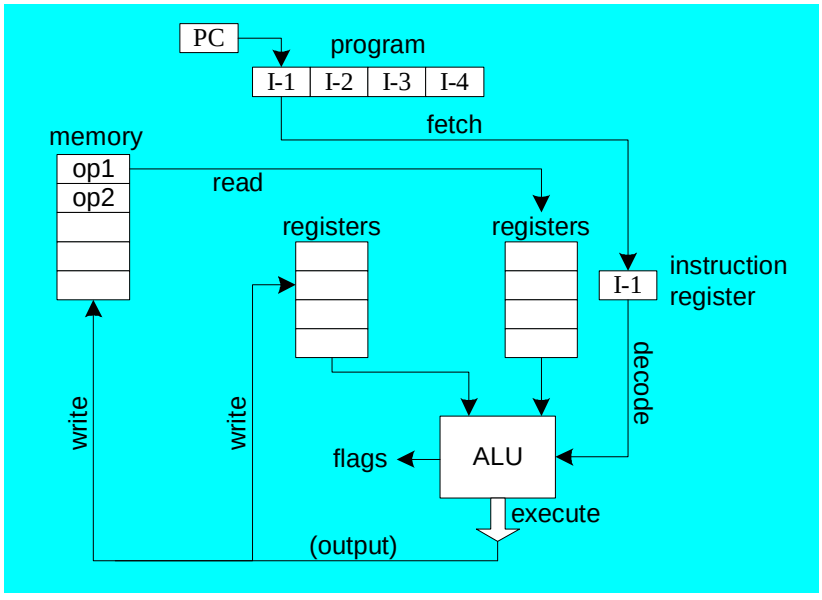


  - A clock that oscillates 1 billion times per second (1 GHz), for example, produces a clock cycle with a duration of one billionth of a second (1 nanosecond).
  - A machine instruction required **at least** one clock cycle to execute, and a few require in excess of 50 clocks (the multiply instruction on the 8088 processor, for example).
  - Instructions requiring memory access often have empty clock cycles called **wait states** because of the differences in the speeds of the CPU, the system bus, and memory circuits.

2.1.2 Instruction Execution Cycle 27

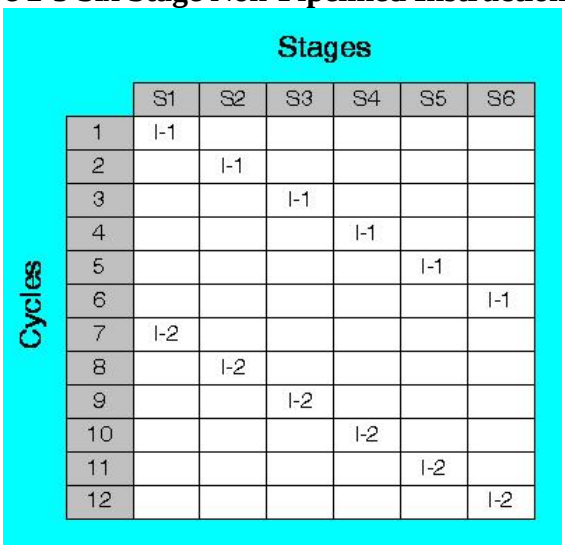
- The execution of a single machine instruction can be divided into a sequence of individual operations called the **instruction execution cycle**.
- Each of the steps is described as follows:
  - **Fetch**: The control unit fetches the instruction from the **instruction queue** and increments the **instruction pointer** (IP).
    - The instruction queue holds a **group** of instructions about to be executed.
    - The instruction is also known as the **program counter**.
    - The instruction pointer contains the address of the **next** instruction.
  - **Decode**: The control unit decodes the instruction's function to determine **what** the instruction will do.
  - **Fetch operands**: It the instruction uses an input operand located in **memory**, the control unit uses a **read** operation to retrieve the operand and copy it into **internal registers**.
  - **Execute**: The ALU executes the instruction using the named **registers** and **internal registers** as operands and sends the output to named **registers and/or memory**. The ALU updates **status flags** providing information about the processor state.
  - **Store output operand**: If the output operand is in **memory**, the control unit uses a **write** operation to store the data.
- The sequence of steps can be expressed neatly in pseudocode:

```
loop
        fetch next instruction
        advance the instruction pointer (IP)
        decode the instruction
        if memory operand needed, read value from memory
        exectue the instruction
        if result is memory operand, write result to memory
continue loop
```



- Non-Pipelined
  - Suppose each execution state in the processor requires a single clock cycle.

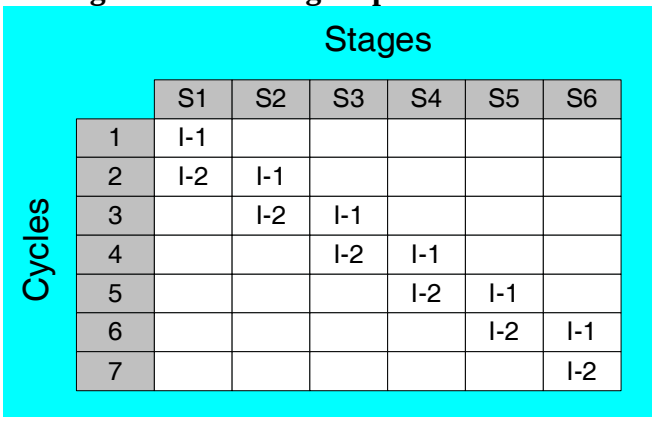
Figure 2-3 Six Stage Non-Pipelined Instruction Execution



For *k* states and *n* instructions, the number of required cycles is: **n * k**

- Multi-Stage Pipeline
  - Pipelining makes it possible for processor to execute instructions in parallel
  - Instruction execution divided into discrete stages
  - More efficient use of cycles, greater throughput of instructions:

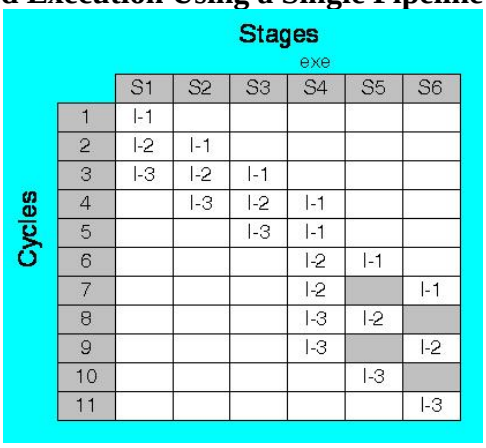
Figure 2-4 Six Stage Pipelined Execution



For *k* states and *n* instructions, the number of required cycles is: **k + (n – 1)**

- Superscalar
  - A superscalar or multi-core processor has **two or more** execution pipelines, making it possible for two instructions to be in the execution state at **the same time**.
  - Wasted Cycles (pipelined):
    - When one of the stages requires two or more clock cycles, clock cycles are again wasted.

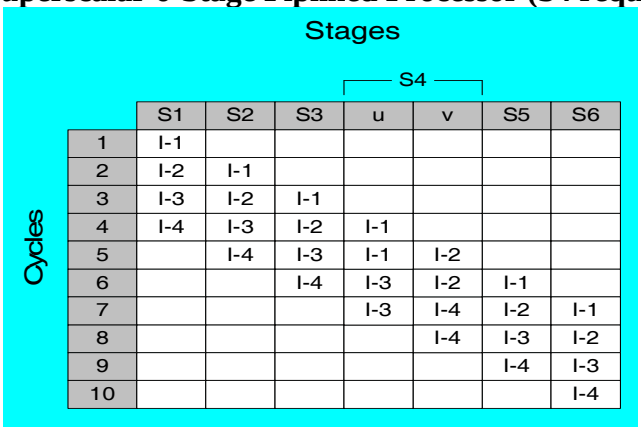
Figure 2-5 Pipelined Execution Using a Single Pipeline (S4 required two clock cycles)



For *k* states (where one stage requires two cycles) and *n* instructions, the number of required cycles is: **k + (2n – 1)**

  - A superscalar processor:
    - For n pipelines, n instructions can execute during the same clock cycle.
    - The Intel Pentium, with two pipelines, was the first superscalar processor in the IA-32 family. The Pentium Pro processor was the first to use three pipelines.
    - Figure 2-6 shows an execution scheme with **two pipelines** in a six-stage pipeline.
      - S4 requires two cycles
      - Odd-numbered instructions enter the **u-pipeline** and even-numbered instructions enter the **v-pipeline**.

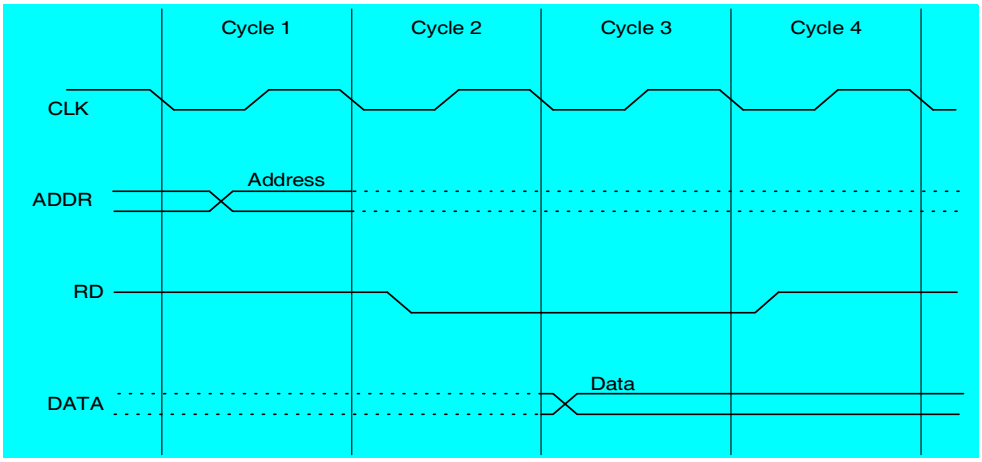
Figure 2-6 Superscalar 6-Stage Piplined Processor (S4 required two clock cycles)



For *k* states (where one stage requires two cycles) and *n* instructions, the number of required cycles is: **k + n**

2.1.3 Reading from Memory 30

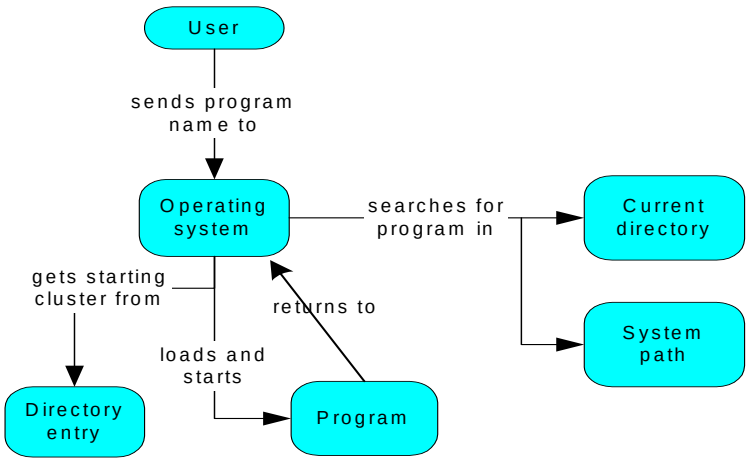
- Multiple machine cycles are required when reading from memory, because it responds much more slowly than the CPU
- The steps are:
  - Cycle1: address placed on address bus
  - Cycle2: Read Line (RD) set low
  - Cycle3: CPU waits one cycle for memory to respond
  - Cycle4: Read Line (RD) goes to 1, indicating that the data is on the data bus



- Cache Memory
  - Because conventional memory is so much slower than the CPU, computers use high-speed cache memory to hold **the most recently** used instruction and data.
  - The first time a program reads **a block** of data, it leaves a copy in the cache.
  - If the program needs to read the same data a second time, it looks for the data in cache.
    - **Cache hit***:* when data to be read is already in cache memory
    - **Cache miss***:* when data to be read is **not** in cache memory.
  - IA-32 processors have two type of cache memory (high-speed expensive **static** RAM both inside and outside the CPU).
    - Level-1 cache: inside the CPU
    - Level-2 cache: outside the CPU

2.1.4 How Programs Run 31

- Load and execute process
  - The program begins running, it is called a **process**.
  - The OS assigns the process an identification number (process ID), which is used to keep track of it while running.



- Multitasking
  - OS can run multiple tasks at the same time.
  - Task:
    - A task is defined as **either** a program (a process) **or** a thread of execution.
    - A process has it own memory area and many contain **multiple threads**. Multiple threads of execution within the same program.
    - A thread **shares** its memory with other threads belonging to the same process.
  - Scheduler:
    - A CPU can really execute **only one** instruction at a time, so a component of the OS named the **schedule**r allocation a slice of CPU time (called a **time slice**) to each task.
    - Scheduler utility assigns a given amount of CPU time to each running program.
    - One type of scheduling used by the OS is called **round-robin** scheduling.
  - Task switching:
    - A multitasking OS runs on a processor that supports task switching.
    - The processor saves the **state** of each task before switching to a new one.
    - A task's state consists of the contents of the processor registers, program counter, and status flags, along with references to the task's memory segments.
  - Priority:
    - A multitasking OS will usually assign varying priorities to tasks, giving them relatively larger or smaller **time slices**.
    - A **preemptive** multitasking OS (such as Windows XP or Linux) permits a higher priority task to interrupt a lower-priority one, leading to better system stability.
      - Example: Suppose an application in locked in loop and has stopped to responding to input.

2.2 IA-32 Processor Architecture 33

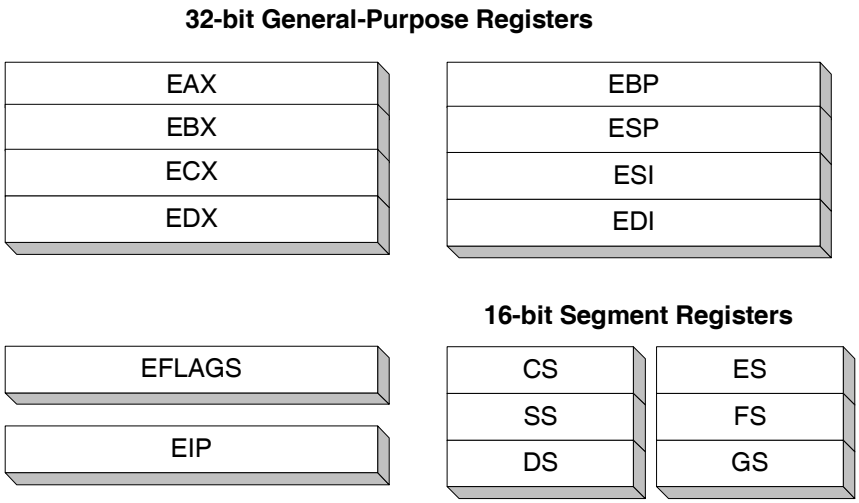
- IA-32 refers to a family of processors beginning with the Intel386 and continuing up to the latest 32-bit processor, the Pentium 4.

2.2.1 Modes of Operation 33

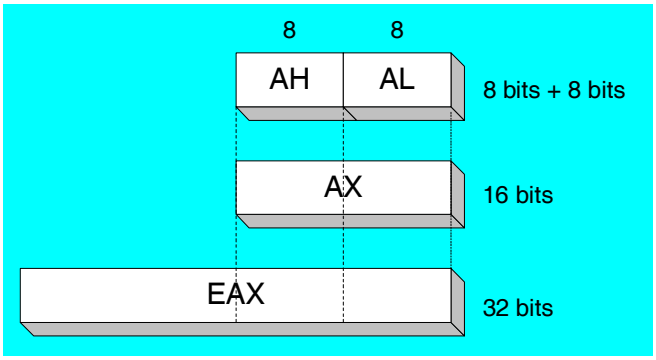
- IA-32 processors have three primary modes of operation: protected mode, real-address mode, and system management mode. Another mode, named virtual-8086, is a special case of protected mode.
- Protected mode
  - Native mode (Windows, Linux) of the processor, in which **all** instructions and features are available.
  - Programs are given separate memory areas named **segments**, and the processor **prevents** programs from referencing memory outside their assigned segments.
- Virtual-8086 mode
  - Hybrid of Protected: While in protected mode, the processor can directly execute real-address mode software such as **MS-DOS program** in a safe multitasking environment.
  - If an MS-DOS program crashes or attempts to write data into the system memory are, it will **not** affect other programming running at the same time.
  - Each program has its own 8086 computer: **Windows XP** can execute **multiple** separate virtual-8086 sessions at the same time.
- Real-address mode
  - Real-address mode implements the programming environment of the Intel **8086** processor.
  - **Native MS-DOS**
  - All Intel processors **boot** in Real-address mode
  - This mode is available in **Windows 98**, and can be used to run MS-DOS program that requires direct access to system memory and hardware devices.
  - Programs running in real-address mode can cause the OS to **crash**.
- System management mode
  - System Management mode (SMM) provides an OS with a mechanism for implementing functions such as **power management**, **system security**, and **diagnostics**.

2.2.2 Basic Execution Environment 34

- Addressable memory
  - Protected mode
    - 4 GB space
    - 32-bit address: 0 to 2^{32} - 1
  - Real-address and Virtual-8086 modes
    - 1 MB space
    - 20-bit address: 0 to 2^{20} - 1
- General-Purpose Registers: Used for arithmetic and data movement



- Accessing Parts of Registers
  - Use 8-bit name, 16-bit name, or 32-bit name
  - Applies to EAX, EBX, ECX, and EDX



32-bit	16-bit	8-bit (high)	8-bit (low)
EAX	AX	AH	AL
EBX	BX	BH	BL
ECX	CX	CH	CL
EDX	DX	DH	DL

  - Index and Base Registers: some registers have only a 16-bit name for their lower half and cannot be divided further

32-bit	16-bit
ESI	SI
EDI	DI
EBP	BP
ESP	SP

- Some Specialized Register Uses
  - General-Purpose Registers
    - EAX accumulator
    - ECX loop counter
    - ESP stack pointer
    - ESI, EDI source and destination index registers
    - EBP extended frame pointer (stack)
  - Segment Registers
    - CS code segment
    - DS data segment
    - SS stack segment
    - ES, FS, GS additional segments
  - EIP Register instruction pointer
  - EFLAGS Register
    - Status and control flags (control the CPU operations)
    - Each flag is a single binary bit
    - Status Flags: reflect the outcomes of arithmetic and logical operations
      - Carry flag (CF): **Unsigned** arithmetic out of range
      - Overflow flag (OF): **Signed** arithmetic out of range
      - Sign flag (SF): Result is negative
      - Zero flag (ZF): Result is zero
      - Auxiliary Carry flag (AC): Carry from bit 3 to bit 4
      - Parity flag (PF): Sum of 1 bits is an even number

2.2.3 Floating-Point Unit 36

- Floating-Point, MMX, XMM Registers
  - Eight 80-bit floating-point data registers
    - ST(0), ST(1), . . . , ST(7)
    - Arranged in a stack
  - Used for all floating-point arithmetic
    - Eight 64-bit MMX registers
    - Eight 128-bit XMM registers for single-instruction multiple-data (SIMD) operations

2.2.4 Intel Microprocessor History 37

- Early Intel Microprocessors
  - Intel 8080
    - 64K addressable RAM
    - 8-bit registers
    - CP/M operating system
    - S-100 BUS architecture
    - 8-inch floppy disks!
  - Intel 8086/8088
    - IBM-PC Used 8088
    - 1 MB addressable RAM
    - **16-bit** registers
    - 16-bit data bus (8-bit for 8088)
    - Separate floating-point unit (8087)
  - The IBM-AT
    - Intel 80286
    - 16 MB addressable RAM
    - Protected memory
    - several times faster than 8086
    - introduced IDE bus architecture
    - 80287 floating point unit
- **Intel IA-32 Family**
  - Intel386
    - 4 GB addressable RAM, **32-bit** registers, 32-bit address but and external data path, paging (virtual memory)
  - Intel486
    - Instruction **pipelining**
  - Pentium
    - Superscalar design with two parallel execution pipelines, 32-bit address bus, 64-bit internal data path
- Intel P6 Family
  - Pentium Pro
    - Advanced optimization techniques in microcode
  - Pentium II
    - MMX (multimedia) instruction set
  - Pentium III
    - SIMD (streaming extensions) instructions
  - Pentium 4 and Xeon
    - Intel NetBurst micro-architecture, tuned for multimedia

- **CISC and RISC**
  - **CISC** (Complex Instruction Set)
    - The Intel 8086 processor was the first in a line of processors using a CISC design
    - The instruction set is **large.**
    - The **major disadvantage** to CISC design is that complex instructions require a relatively **long time** to decode and execute.
    - An interpreter inside the CPU written in a language called **microcode** decodes and executes each machine instruction.
    - Once Intel released the 8086, it became necessary for all subsequent Intel processors to **compatible** with the first one. **Customers did not** want to throw away their existing software every time a new processor was released.
  - **RISC** (Reduced Instruction Set)
    - A RISC consists of a relatively small number of **short**, **simple** instructions that execute relatively **quickly**.
    - The instruction set is **small.**
    - Rather that using a microcode interpreter to decode and execute machine instructions, a RISC processor **directly** decodes and executes instructions using hardware.
    - **High-speed** engineering and graphics workstations have been built using RISC processors for many years.
    - The systems have been **expensive** because the processors were produced in small quantities.
- **Why Intel Microprocessor?**
    - Because of the **huge popularity** of IBM-PC-compatible computers, Intel was able to **lower the price** of its processors and dominate the microprocessor market.
    - The IA-32 instruction set continues to be **complex** and constantly expanding.

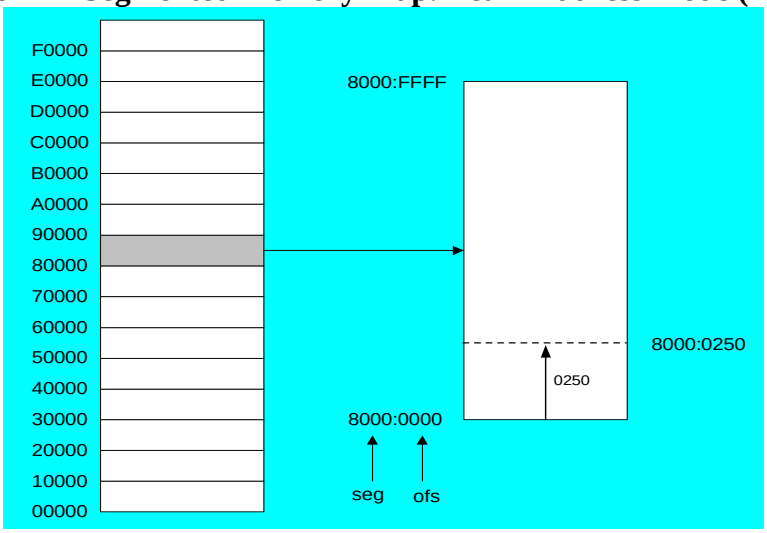
2.3 IA-32 Memory Management 39

- Real-address mode
- Calculating linear addresses
- Protected mode
- Multi-segment model
- Paging

2.3.1 Real-Address Mode 39

- Real-address mode
  - **1 MB** RAM maximum addressable, from hexadecimal 00000 to FFFFF.
  - Application programs can access **any** area of memory
  - **Single tasking**
  - Supported by **MS-DOS** operating system: **Windows 95 and 98** can be booted into this mode.
  - Segmented Memory
    - All of memory is divided into **64-kilobyte** (2^{16} bytes) units called segments.
    - Segmented memory addressing: **absolute (linear)** address is a combination of a **16-bit** segment value added to a **16-bit** offset

Figure 2-11 Segmented Memory Map. Real –Address Mode (2^{20} Bytes)



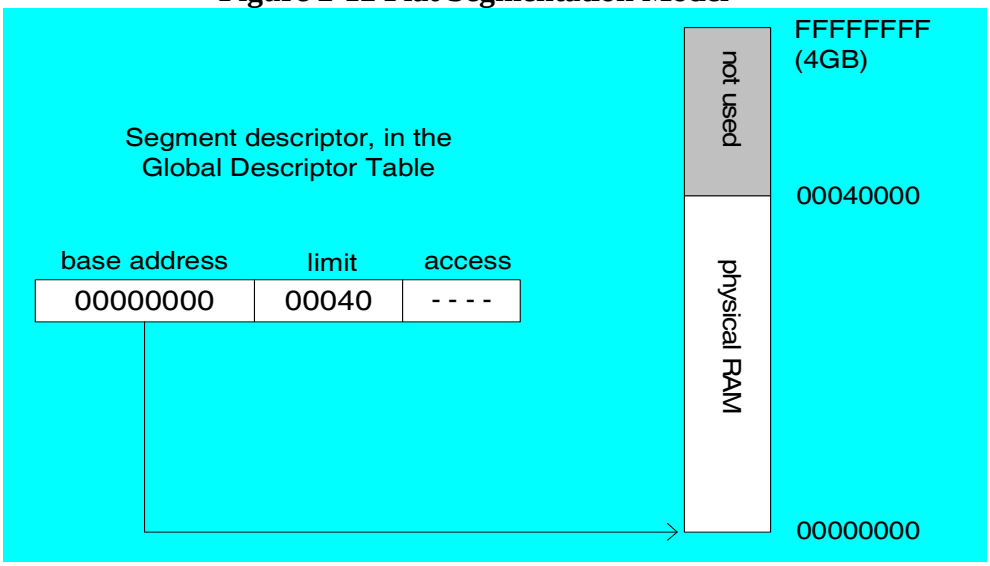
- Calculating Linear Addresses
  - In real-address mode, the linear (or absolute) address is **20** bits.
  - Given a segment address, **multiply it by 16** (add a hexadecimal zero), and add it to the offset
  - Example: convert **08F1:0100** to a linear address

Adjusted Segment value:	0	8	F	1	**0**
Add the offset:		0	1	0	0
Linear address:	**0**	**9**	**0**	**1**	**0**

2.3.2 Protected Mode 41

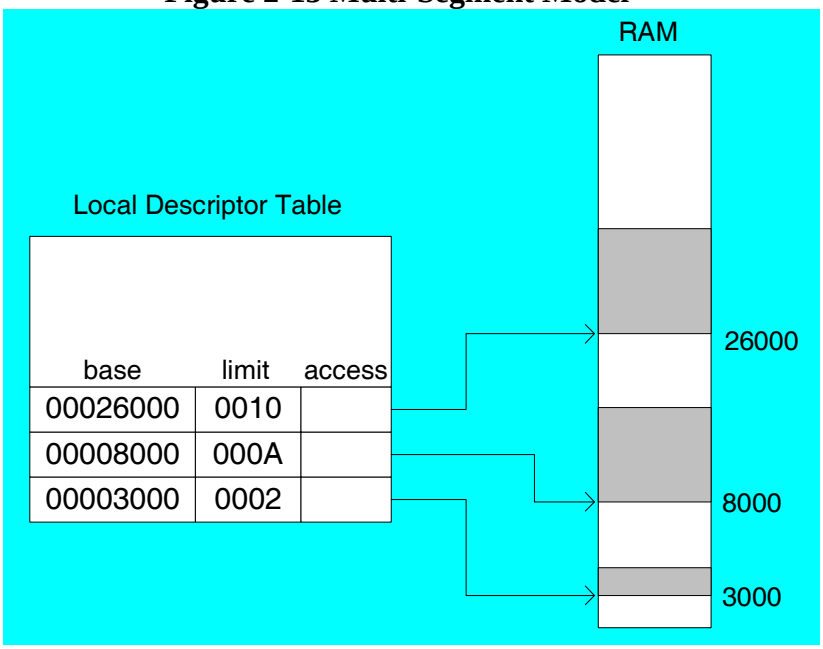
- Protected mode
  - 4 GB addressable RAM (00000000 to FFFFFFFF)
  - Each program assigned a memory partition which is protected from other programs
  - Designed for multitasking
  - Supported by Linux & MS-Windows
  - Segment descriptor tables
  - Program structure
    - code, data, and stack areas
    - CS, DS, SS segment descriptors
    - global descriptor table (GDT)
  - **MASM Programs** use the Microsoft **flat** memory model
  - Flat Segment Model
    - Single Global Descriptor Table (GDT)
    - **All** segments mapped to entire 32-bit address space
    - Suppose a computer had 256MB of RAM. The segment limit field would contain 10000 hex because its value is **implicitly multiplied by 1000 hex**, producing 10000000 hex (256MB)

Figure 2-12 Flat Segmentation Model



- Multi-Segment Model
  - **Each** program has a Local Descriptor Table (LDT)
  - holds descriptor for each segment used by the program

Figure 2-13 Multi-Segment Model



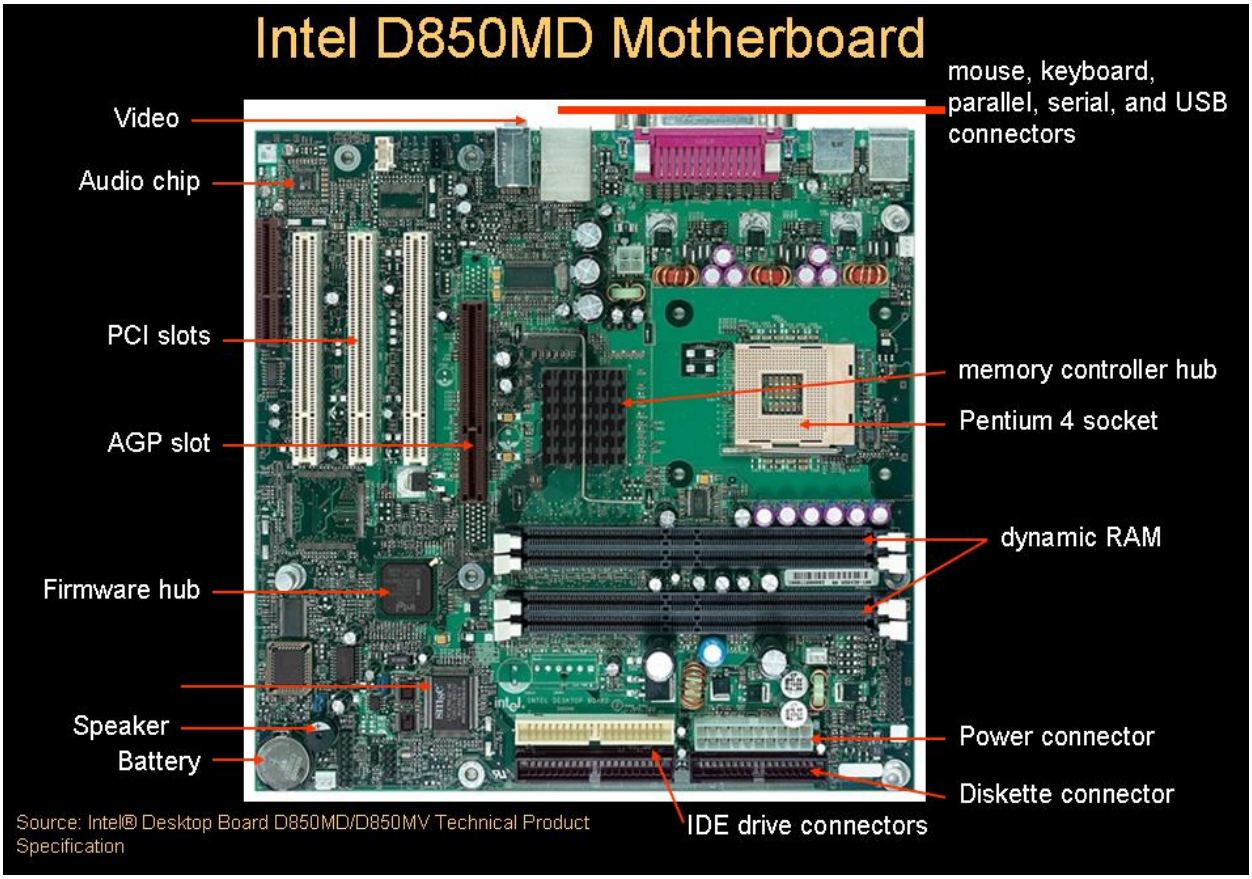
- Paging
  - IA-32 processors support paging, a feature that permits segment to be divided into **4,096-byte** blocks of memory called **pages**.
  - Paging permits the total memory used by all programs running at the time to be much **larger** than the computer's **physical memory**.
  - Virtual Memory: The complete collection of pages mapped by the OS is called virtual memory.
  - Virtual memory manager (VMM): OS utility that manages the loading and unloading of pages
  - When a task is running, part of running program is in **memory**, part is on **disk**
  - **Page fault:** Issued by CPU when a page must be loaded from disk

2.4 Components of an IA-32 Microcomputer 43

- Motherboard
- Video output
- Memory
- Input-output ports

2.4.1 Motherboard 43

- Motherboard
  - CPU socket
  - External cache memory slots
  - Main memory slots
  - BIOS (Basic Input-Output System) chips
  - Sound synthesizer chip (optional)
  - Video controller chip (optional)
  - IDE, parallel, serial, USB, video, keyboard, joystick, network, and mouse connectors
  - PCI bus connectors (expansion cards)
  - Intel 8042 keyboard and mouse microcontroller



2.4.2 Video Output 44

- Video Output
  - Video controller
    - On motherboard, or on expansion card
    - AGP (accelerated graphics port technology)
  - Video memory (VRAM)
  - Video CRT Display
    - Uses raster scanning
    - Horizontal retrace
    - Vertical retrace
  - Direct digital LCD monitors
    - No raster scanning required

2.4.3 Memory 45

- Memory
  - ROM (Read-Only Memory)
    - permanent, cannot be erased
  - EPROM (Erasable Programmable Read-Only Memory)
    - erased by ultraviolet light and reprogrammed
  - Dynamic RAM (Ramdom Access Memory) (DRAM)
    - commonly known as **main memory**, Inexpensive; must be refreshed constantly with 1 millisecond or less
  - Static RAM (SRAM)
    - Expensive; used for **cache memory**; no refresh required
  - Video RAM (VRAM)
    - Dual ported; optimized for constant video refresh
  - CMOS (Complimentary Metal-Oxide Semiconductor) RAM
    - System setup information; refreshed by a battery, so its contents are retained when the computer's power is off.

2.4.4 Input-Output Ports and Device Interfaces 45

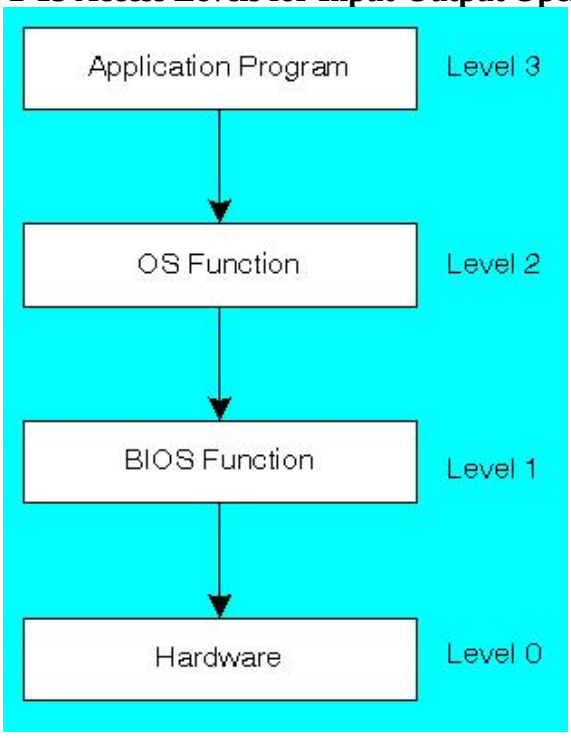
- Input-Output Ports and Device Interfaces
  - USB (universal serial bus)
    - intelligent high-speed connection to devices
    - USB Version 2.0 supports data transfer speeds of 480 megabits/second
  - Parallel
    - Short cable, high speed (1MBbyte per second) over short distances, usually no more than 10 feet.
    - Common for **printers**
    - Bidirectional, parallel data transfer
    - Intel 8255 controller chip
  - Serial
    - RS-232 serial port
    - **One bit at a time**
    - Uses long cables and modems
    - 16550 UART (universal asynchronous receiver transmitter)
    - Programmable in assembly language
  - IDE (Intelligent Drive Electronics or Integrated Drive Electronics)
    - parallel ATA (advanced technology attachment) devices, in which the drive controller is located on the drive itself.
    - SATA (serial ATA), which provides higher data transfer rates than parallel ATA.
  - FireWire
    - FireWire is a high-speed external bus standard supporting data transfer speeds up to **800MB** per second.

2.5 Input-Output System 46

2.5.1 How It All Works 46

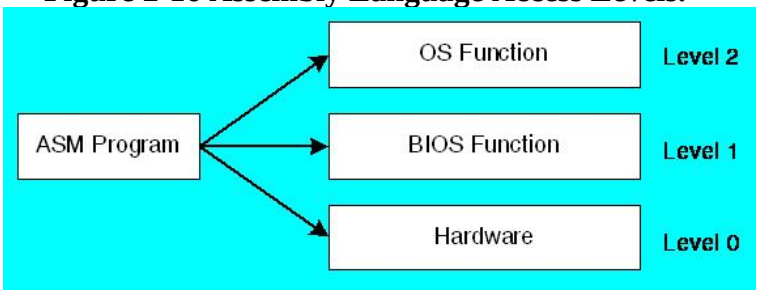
- Levels of Input-Output
  - *Level 3*: Call a library function (C++, Java)
    - Easy to do; abstracted from hardware; details hidden
    - Slowest performance
  - *Level 2:* Call an operating system function (API: Application Programming Interface)
    - Specific to one OS; device-independent
    - Medium performance
  - *Level 1:* Call a *BIOS (Basic Input-Output System)* function
    - May produce different results on different systems
    - Knowledge of hardware required
    - Usually good performance
  - *Level 0:* Communicate directly with the hardware
    - A device driver works much like the BIOS. An example is CDROM.SYS, which enables MS-DOS to read CD-ROM drives.

Figure 2-15 Access Levels for Input-Output Operations



- ASM Programming levels
  - Level 2: Call OS functions to perform generic text I/O and file-based I/O.
  - Level1: Call BIOS function to control device-specific features such as color, graphics, sound, keyboard input, and low-level disk I/O
  - Lovel 0: Send and receive data from hardware points, having absolute control over specific devices.
    - Programs using this level must **extend their coding** logic to handle variations in I/O devices.
    - **Real-mode game programs** are prime examples because they usually take control of the computer.
  - ASM programs can perform input-output at each of the following levels:

Figure 2-16 Assembly Language Access Levels.



  - Example: You wanted to play a WAV file suing an audio controller device:
    - At the OS level, you would **not** have to know what type of device was installed and you would not be concerned with nonstandard features.
    - At the BIOS level, you would query the sound card (**using its installed device driver software**) and find out where it belonged to a certain class of sound having known features.
    - At the hardware level, you would **fine tune** the program for certain brands of audio cards, taking advantage of each card's special features.

2.6 Chapter Summary 49

- CPU contains a limited number of storage locations called **registers**, a high-frequency **clock** to synchronize its operations, a **control unit**, and the **arithmetic logic unit**.
- The execution of a single machine instruction can be divided into a sequence of individual operations called the **instruction execution cycle**.
- Three primary operations are **fetch**, **decode**, and **execute**.
- Each step in the instruction cycle takes at least one tick of the system clock, called a **clock cycle**.
- **Pipelined** execution greatly improves the throughput of multiple instructions in a CPU by permitting the overlapped execution of multi-stage instructions.
- A **superscalar** processor is a pipelined processor with multiple execution pipelines.
- A multitasking OS can run multiple tasks at the same time.
- IA-32 processors have three basic modes of operation: **protected mode**, **real-address mode**, and **system management mode**. In addition, **virtual-8086** mode is a special case of protected mode.
- The earliest Intel processors for the IBM PC were based on the complex instruction set (**CISC**) approach.
- A reduced instruction set (**RISC**) machine language consists of a relatively small number of short, simple instructions that can be executed **quickly** by the processor.
- In real-address mode, only **1MB** of memory can be addressed, using hexadecimal addresses 00000 to FFFFF.
- In **protected mode**, the processor can run multiple programs at the same time. It assigns each processor (running program) a total **4GB** of **virtual memory**.
- In virtual-8086 mode, the computer runs in **protected mode** and creates a virtual 8086 machine with its own **1MB** address space that simulates an 80 X 86 computer running in real-address mode.
- In the **flat** segmentation model, **all** segments are mapped to the entire physical address space of the computer. **MASM Programs** use the Microsoft **flat** memory model
- In the **multi-segment** mode, each task is given its own table of segment descriptions, called a local descriptor table (LDT).
- The IA-32 supports a feature called paging, which permits a segment to be divided into **4096-byte** blocks of memory called **pages**.
- Paging the total memory used by all programs running at the same time to be much **larger** than the computer's actual (physical) memory).
- A parallel port transmits **8 or 16 data bits** simultaneously from one device to another.
- An RS-232 serial port sends binary bits **one at a time**.
- The BIOS (Basic Input-Output System) is a collection of functions that communicate **directly** with hardware devices.
- Assembly programs can also **directly** access input-output devices.