Computer Organization & Assembly Language - EE2003

Lecture 03

Week 02

Chapter Overview

- **General Concepts**
- IA-32 Processor Architecture
- IA-32 Memory Management
- Components of an IA-32 Microcomputer
- Input-Output System

General Concepts

- Basic microcomputer design
- Instruction execution cycle
- Reading from memory
- How programs run

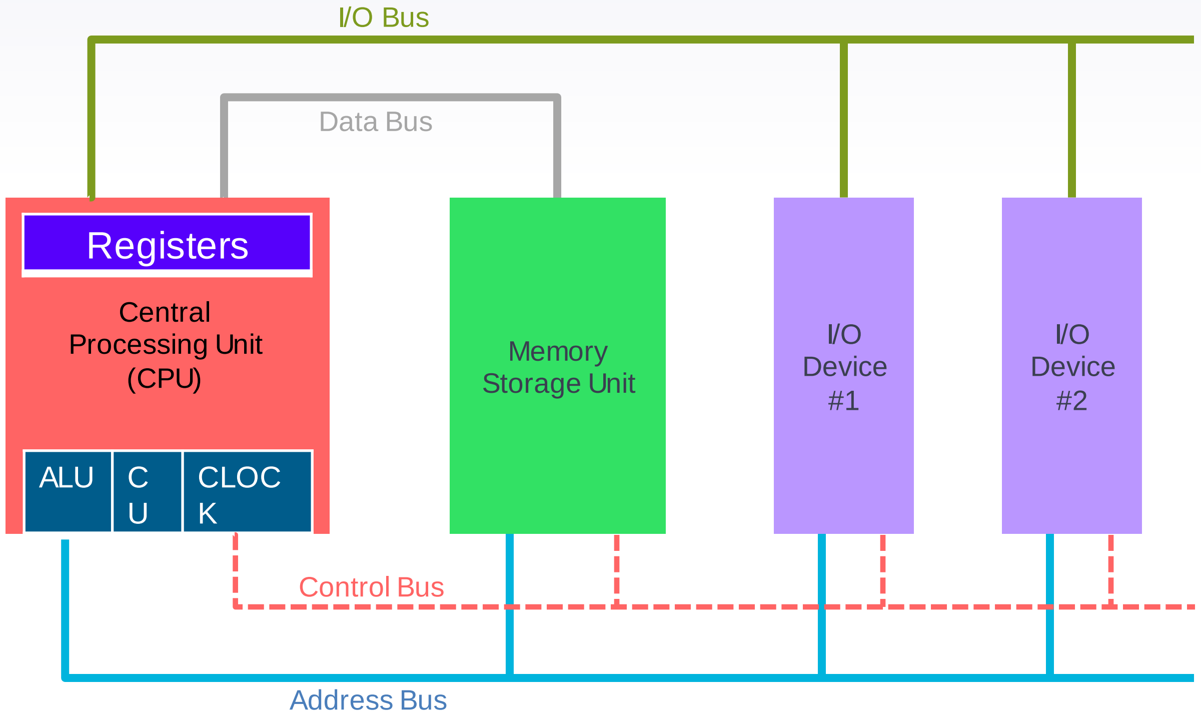
Organization and Architecture

- Computer Architecture
  - Attributes of a computer that have a direct impact on the logical program execution
  - e.g. I/O mechanisms, memory addressing techniques
- Computer Organization
  - Operational units and their interconnection that realizes the architectural specifications
  - e.g. control signals, interconnection between computer and its peripherals

Basic Computer Organization (1/2)

- Computer has 3 main components
  - Processor, also called Central Processing Unit (CPU)
  - Memory and Storage Devices
  - I/O Devices
- These components communicate with each other through
  - Data Bus
  - I/O Bus
  - Address Bus
  - Control Bus

Basic Computer Organization (2/2)



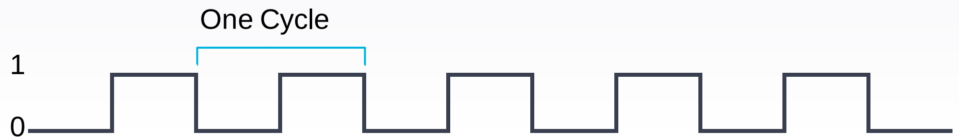
Central Processing Unit (CPU)

- Generally called Processor
- Contains
  - Clock
  - Registers
  - Arithmetic Logic Unit (ALU)
    - Performs arithmetic and logic instructions
  - Control Unit (CU)
    - Generates the control signals required to execute the instructions

Clock (1/3)

- Clock cycle is basic unit of time for machine instructions
- Synchronizes Processor and Bus operations
- Clock cycle = Clock period = 1/(Clock Rate)
- Clock Rate = Clock Frequency = Cycles per sec
  - 1 Hz clock produces 1 Clock Cycle in 1 second
  - 1 KHz clock produces 1000 Clock Cycles in 1 second
  - 1 MHz clock produces 1,000,000 cycles in 1 second
  - 1 GHz clock produces 1,000,000,000 cycles in 1 second

Clock (2/3)



Clock Frequency	Clock Period (sec)	Time to execute one instruction
1 Hz	1	1 sec
1 KHz	0.001	1 ms (millisecond)
1 MHz	0.000,001	1 µs (microsecond)
1 GHz	0.000,000,001	1 ns (nanosecond)

What's Next

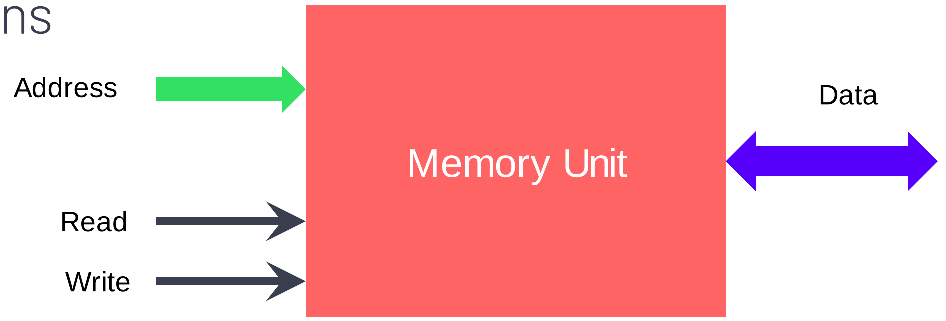
- General Concepts
- **IA-32 Processor Architecture**
- IA-32 Memory Management
- Components of an IA-32 Microcomputer
- Input-Output System

Memory (1/2)

- Ordered sequence of bytes
  - Sequence number is called memory address
- Byte addressable memory
  - Each byte has a unique address
- Physical address space
  - Determined by the address bus width
  - Pentium has 32-bit address bus
    - Physical address space of Pentium = 2^{32} bytes = 4 GB
  - Itanium with 64-bit address bus can support up to 2^{64} bytes of physical address space

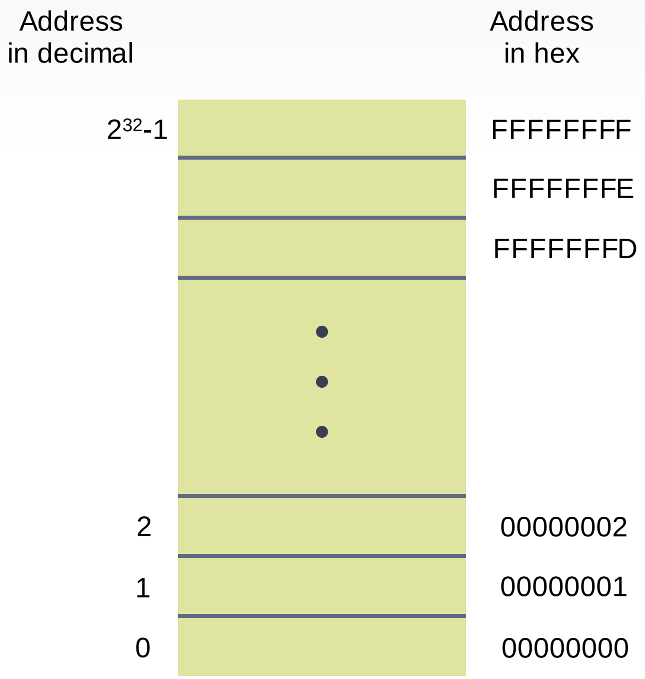
Memory (2/2)

- Address of location to be read/written is placed on the address bus
- Data to be read/written is placed on the data bus by memory/processor
- Two control signals read and write decide the reading or writing operations



Physical Address Space

- Address space is the set of addressable memory locations (bytes)



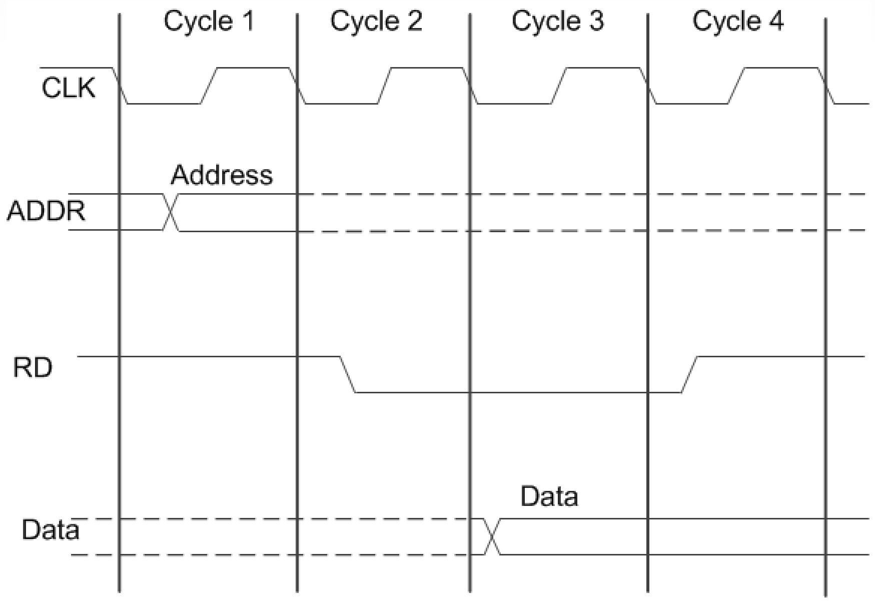
Reading from Memory

- Multiple clock cycles are required
- Memory responds much more slowly than CPU
- Reading process is carried out in this way
  - Address is placed on the address bus
  - Read Line (RD) goes low indicating that processor wants to read
  - CPU waits for memory to respond
  - Read Line (RD) goes high indicating that data has been placed on the data bus

Memory Read and Write Cycles

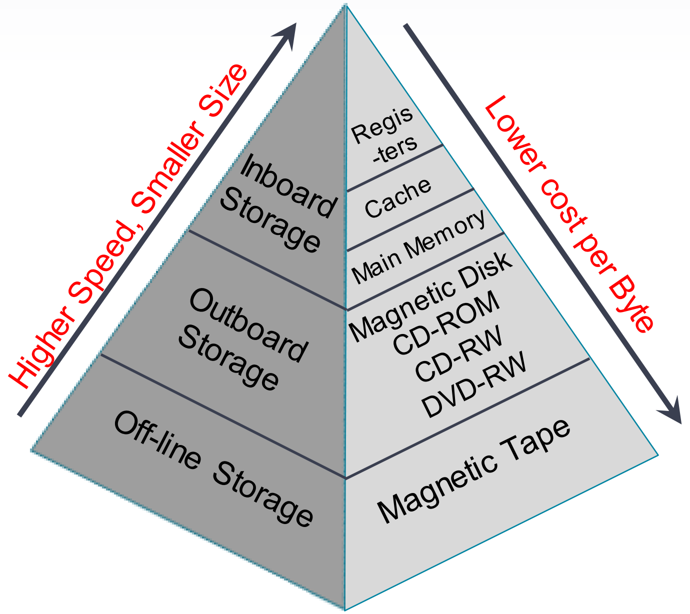
- In Read Cycle, the Processor
  - places address on address bus
  - asserts the memory read control signal
  - waits for memory to place data on the data bus
  - reads the data from data bus
  - drops the memory read signal
- In Write Cycle, the Processor
  - places address on the address bus
  - asserts the memory write control signal
  - places the data on the data bus
  - waits for memory to store the data
  - drops the memory write signal

Reading from Memory



Memory Hierarchy

- Registers
  - Fastest storage elements
- Cache Memory
  - Stores recently used instructions
- Main Memory
- Magnetic Disk
  - Stores data permanently
- Magnetic Tape



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
- Cache hit: when data to be read is already in cache memory
- Cache miss: when data to be read is not in cache memory.

Multitasking

- OS can run multiple programs at the same time.
- Multiple threads of execution within the same program.
- Scheduler utility assigns a given amount of CPU time to each running program.
- Rapid switching of tasks
  - gives illusion that all programs are running at once
  - the processor must support task switching.

IA-32 Processor Architecture

- Modes of operation
- Basic execution environment
- Floating-point unit
- Intel Microprocessor history

Basic Execution Environment

- Addressable memory
- General-purpose registers
- Index and base registers
- Specialized register uses
- Status flags
- Floating-point, MMX, XMM registers

Registers

- Very high speed memory units inside CPU
- Can be categorized into
  - General Purpose Registers
  - Segment Registers
  - Instruction Pointer
  - EFLAGS Register
    - Control Flags
    - Status Flags

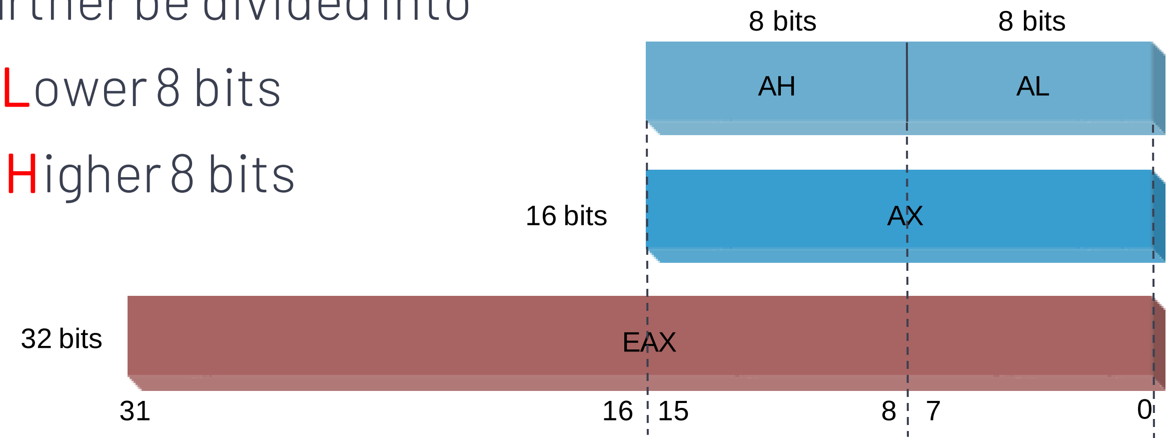
General Purpose Registers

- Used primarily for arithmetic and data movement
- Eight 32-bit ***Extended*** general purpose registers

Register	Description	Specialized Usage
E**A**X	**A**ccumulator	Used in *mul* and *div* instruction
E**B**X	**B**ase	
E**C**X	**C**ounter	Used in *LOOP* instruction
E**D**X	**D**ata	
E**BP**	**B**ase **P**ointer	Used to reference local variables on stack
E**SP**	**S**tack **P**ointer	Points at top of the stack, used by *PUSH* and *POP*
E**SI**	**S**ource **I**ndex	Used by high speed memory transfer instruction
E**DI**	**D**estination **I**ndex	Used by high speed memory transfer instruction

Accessing Parts of Registers (1/2)

- Portions of General Purpose Registers can also be accessed
  - Lower 16 bits of EAX are named AX
  - AX can further be divided into
    - AL = Lower 8 bits
    - AH = Higher 8 bits



Accessing Parts of Registers (2/2)

- EAX, EBX, ECX, EDX can be accessed at word and byte level
- EBP, ESP, ESI, EDI can be accessed at word level

32-bit	16-bit	8-bit **H**igher	8-bit **L**ower
EAX	AX	A**H**	A**L**
EBX	BX	B**H**	B**L**
ECX	CX	C**H**	C**L**
EDX	DX	D**H**	D**L**

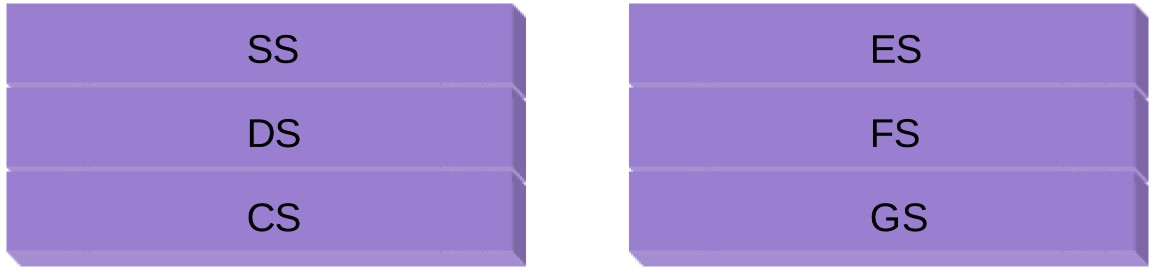
32-bit	16-bit Lower
ESP	SP
EBP	BP
ESI	SI
EDI	DI

Segment Registers (1/2)

- Six 16-bit segment registers
- Indicate base addresses of pre-assigned memory areas
- An assembly program usually contains 3 segments
  - Stack Segment: holds local function variables and function parameters
  - Data Segment: holds variables
  - Code Segment: holds program instructions

Segment Registers (2/2)

- SS = Stack Segment
- DS = Data Segment
- CS = Code Segment
- 3 extra segment registers
  - ES, FS, GS (Extra Segment)



Instruction Pointer

- Represented by EIP
- Also called Program Counter (PC)
- Contains the address of next instruction to be executed
- Its value changes with the execution of program
- Certain machine instructions influence the value of EIP

Status Flags (1/2)

- EFLAGS register consists of individual binary bits
- Can be categorized into
  - Control Flags: control the operation of CPU
  - Status Flags: reflect the outcome of arithmetic and logical operations

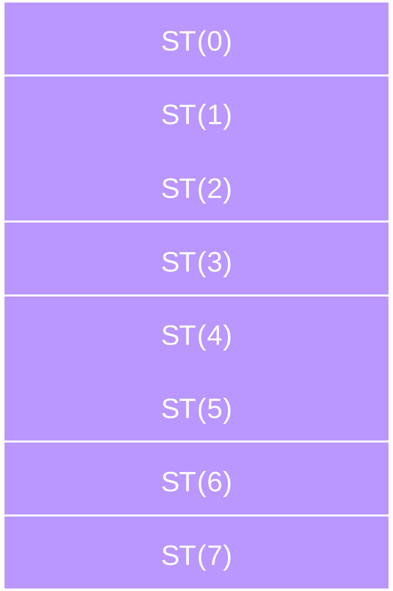
0	0	0	0	0	0	0	0	0	0	ID	VIP	VIF	AC	VM	RF	0	NT	IOPL	OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

Status Flags (2/2)

- Arithmetic and Logical operations set/reset them
- Status Flags are
  - Carry Flag (CF): set when unsigned arithmetic operation produces a carry
  - Overflow Flag (OF): sets when signed arithmetic result is out of range
  - Sign Flag (SF): sets when result of operation is –ve
  - Zero Flag (ZF): sets when result of operation is zero
  - Auxiliary Flag (AF): sets when there is a carry from bit 3 to bit 4
  - Parity Flag (PF): sets when parity is even

Floating Point Unit (FPU)

- FPU performs high speed floating point operations
- Integrated into main processor chip from Intel486 onward
- Eight 80-bit floating point data registers
- All arranged as a stack



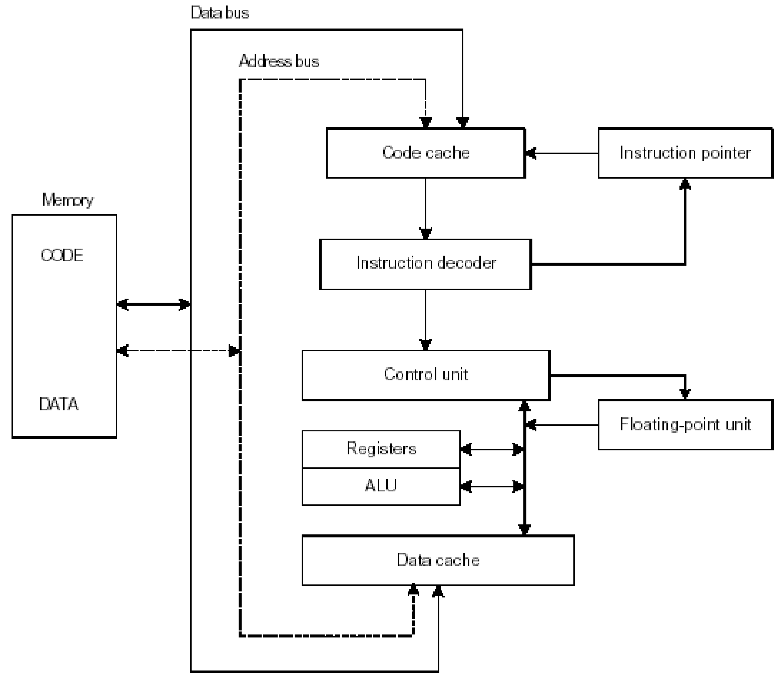
Lecture 04

Week 02

Instruction Execution Cycle

- Fetch
- Decode
- Fetch operands
- Execute
- Store output

Figure 2–2 Simplified Pentium CPU Block Diagram.



Instruction Execution Cycle (1/2)

- A program is loaded into memory before execution
- Instruction Pointer (IP) contains the address of next instruction to be executed
- Instruction Queue contains the group of instructions about to be executed
- Three basic steps in execution of a machine instruction
  - Fetch
  - Decode
  - Execute
- Two more steps if the instruction uses an operand located in memory
  - Fetch Operand
  - Store Output

Instruction Execution Cycle (2/2)

- Order of steps when instruction uses an operand located in memory
  - Instruction Fetch (IF)
  - Instruction Decode (ID)
  - Operand Fetch from Memory (OF)
  - Instruction Execute (IE)
  - Write Back in Memory (WB)
- Instruction Execution Cycle (IEC) is the combination of all above operations

Instruction Fetch

- Program Counter (PC) or IP holds address of next instruction to be executed
- Processor fetches instruction from memory pointed to be PC
- Increment PC to the next instruction
- Fetching operation concludes here

Instruction Decode

- Control Unit (CU) determines the function of the instruction
- Instruction's input operands are passed to ALU
- Signals sent to ALU to perform the desired operation

Operand Fetch from Memory

- Sometimes instructions may involve some operands located in memory
- In this case, data should be retrieved from memory
- The CU uses a memory *read* operation
- CU copies these operands into Instruction Registers which are invisible to user programs

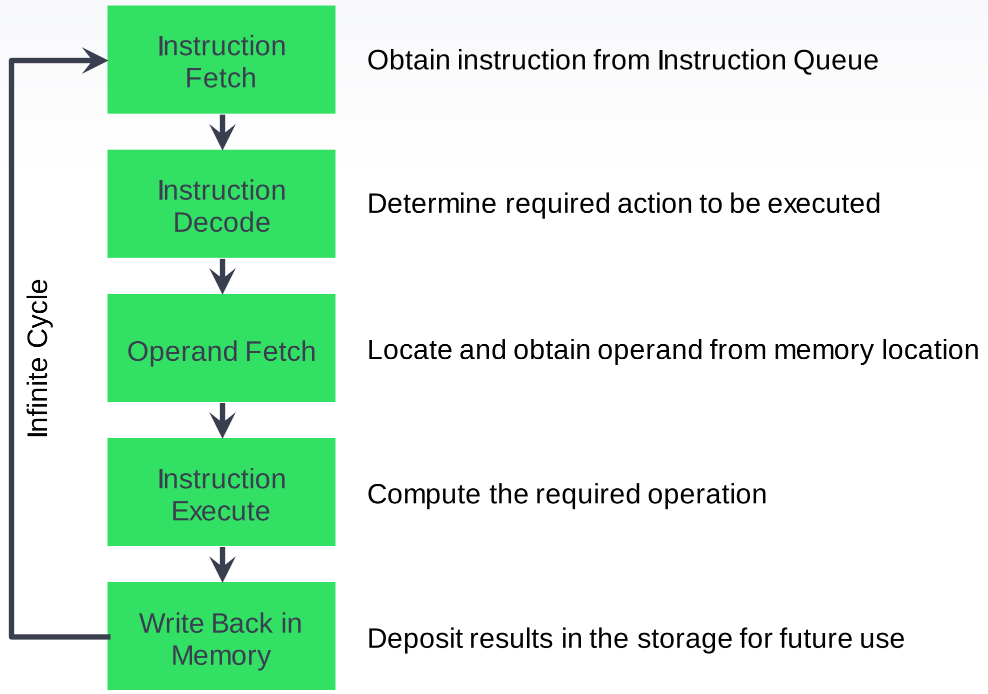
Instruction Execute

- The instruction is ready for execution at this stage
- ALU executes the instruction
- ALU updates the status flags providing information about the result

Write Back in Memory

- The output may be required to save in memory for future uses
- CU performs a *write* operation to save this output into memory

IEC Graphical View


Instruction Fetch
Instruction Decode
Operand Fetch
Instruction Execute
Write Back in Memory
Infinite Cycle
Obtain instruction from Instruction Queue
Determine required action to be executed
Locate and obtain operand from memory location
Compute the required operation
Deposit results in the storage for future use

Intel Microprocessor History

- Intel 8086, 80286
- IA-32 processor family
- P6 processor family
- CISC and RISC

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, paging (virtual memory)
- Intel486
  - instruction pipelining
- Pentium
  - superscalar, 32-bit address bus, 64-bit internal data path

64-bit Processors

- Intel64
  - 64-bit linear address space
  - Intel: Pentium Extreme, Xeon, Celeron D, Pendium D, Core 2, and Core i7
- IA-32e Mode
  - Compatibility mode for legacy 16- and 32-bit applications
  - 64-bit Mode uses 64-bit addresses and operands

Intel Technologies

- HyperThreading technology
  - two tasks execute on a single processor at the same time
- Dual Core processing
  - multiple processor cores in the same IC package
  - each processor has its own resources and communication path with the bus

Intel Processor Families

Currently Used:

- Pentium & Celeron – dual core
- Core 2 Duo - 2 processor cores
- Core 2 Quad - 4 processor cores
- Core i7 – 4 processor cores

CISC and RISC

- CISC – complex instruction set
  - large instruction set
  - high-level operations
  - requires microcode interpreter
  - examples: Intel 80x86 family
- RISC – reduced instruction set
  - simple, atomic instructions
  - small instruction set
  - directly executed by hardware
  - examples:
    - ARM (Advanced RISC Machines)
    - DEC Alpha (now Compaq)

What's Next

- General Concepts
- IA-32 Processor Architecture
- **IA-32 Memory Management**
- Components of an IA-32 Microcomputer
- Input-Output System

Modes of Operation

- Protected mode
  - native mode (Windows, Linux)
- Real-address mode
  - native MS-DOS
- System management mode
  - power management, system security, diagnostics

- Virtual-8086 mode
  - hybrid of Protected
  - each program has its own 8086 computer

Addressable Memory

- Protected mode
  - 4 GB
  - 32-bit address
- Real-address and Virtual-8086 modes
  - 1 MB space
  - 20-bit address

IA-32 Memory Management

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

x86 Operation Modes

- x86 architecture operates in 3 basic different modes
  - Protected Mode
    - All instructions and features of processor are available
    - Programs cannot reference memory outside their segments
    - Virtual-8086 is special mode of Protected Mode
  - Real-Address Mode
    - Programs can have direct access to memory and hardware devices
    - ... so operating system can crash in this mode
  - System Management Mode
    - Implemented in processors customized for a particular system setup
    - Supports power management and system security

x86 Memory Management

- Memory is managed according to the operation mode
  - Real-Address Mode
  - Protected Mode

Real-Address Mode Memory Management

- 16-bit registers
- 20-bit address bus
  - Physical Address Space = 1MByte
  - Linear or Absolute Address range is 00000h to FFFFFh
- How 20-bit address can be accommodated in 16-bit Registers?
  - Solution lies in Segmented Memory

Real-Address Mode Memory Management

- Programs cannot use linear address directly because of 16-bit registers
- Addresses are represented using 2 16-bit registers
  - One 16-bit value is called Segment Value which is placed in segment register
  - Other 16-bit value is called Offset value

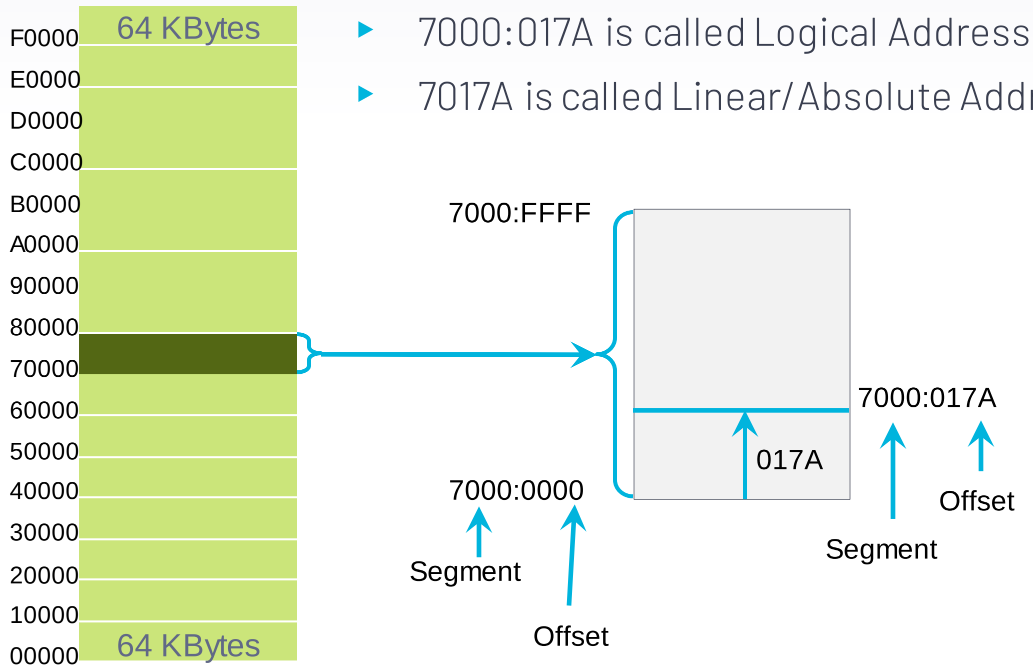
Segmented Memory Scheme

- All memory is divided in 64KBytes segments
- Each segment begins at an address having a zero in its last hex digit
  - … this zero is omitted when representing segment value in 16-bit registers
    - A segment value of $E000_{16}$ refers to the segment address of $E0000_{16}$

- If we have Physical Address Space of 2^{20} bytes, and size of each segment is 64KBytes (2^{16}bytes), how many segments do we have?

Segmented Memory Map

- 7000:017A is called Logical Address
- 7017A is called Linear/Absolute Address



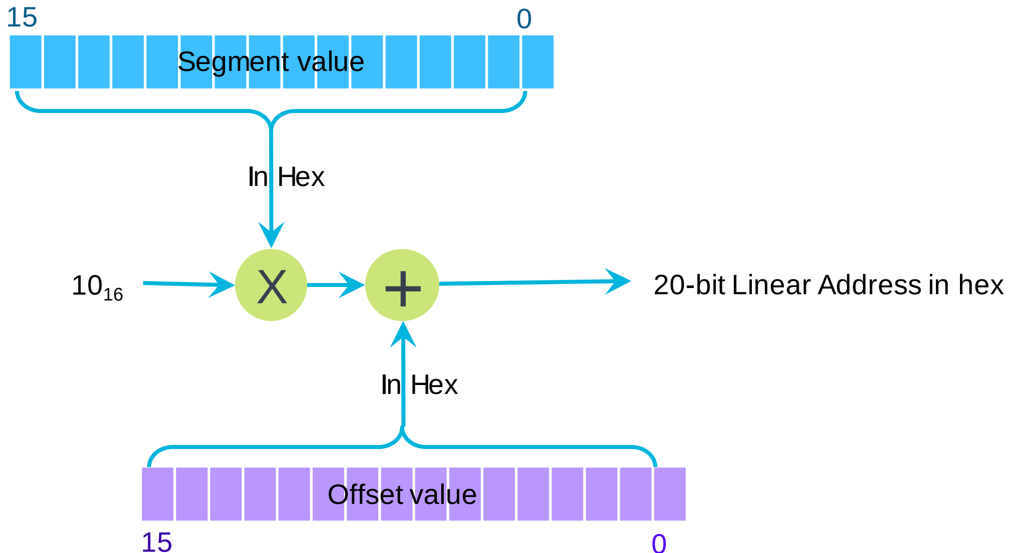
Calculating Linear Addresses

- 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
```

20-bit Linear Address Calculation

- How to convert a 16-bitSeg:16-bitOffset logical address into 20-bit linear/absolute address



Protected Mode Memory Management

- Restricts application programs from accessing system hardware
- Logical Address consists of
  - 16-bit segment selector (Segment Registers)
  - 32-bit offset value
- Program's linear address space is 4GBytes
  - What is address bus width?
- A typical protected-mode program has three segments
  - Code Segment
  - Data Segment
  - Stack Segment
- Segment Registers point to the Segment Descriptor Table
  - … Segment Descriptor Tables are created by OS to keep track of individual program segments

Your turn . . .

What linear address corresponds to the segment/offset address 028F:0030?

028F0 + 0030 = 02920

Always use hexadecimal notation for addresses.

Your turn . . .

What segment addresses correspond to the linear address 28F30h?

Many different segment-offset addresses can produce the linear address 28F30h. For example:

28F0:0030, 28F3:0000, 28B0:0430, . . .

Paging

- Paging divides the linear address space into fixed-sized block called pages
- IA-32 uses pages of size 4 KB
- Pages are allocated space on main memory
  - ... if main memory is full, then on hard disk
- Complete set of pages mapped by OS is called Virtual Memory

Paging

- We want to run many programs in parallel
- Main memory may not be enough to load all of them at the same time
- Disk storage is cheaper and we have plenty of it
- Paging gives an illusion that we have unlimited main memory
- The more a program depends on paging, the slower it runs
- Higher amount of main memory means less usage of paging

What's Next

- General Concepts
- IA-32 Processor Architecture
- IA-32 Memory Management
- **Components of an IA-32 Microcomputer**
- Input-Output System

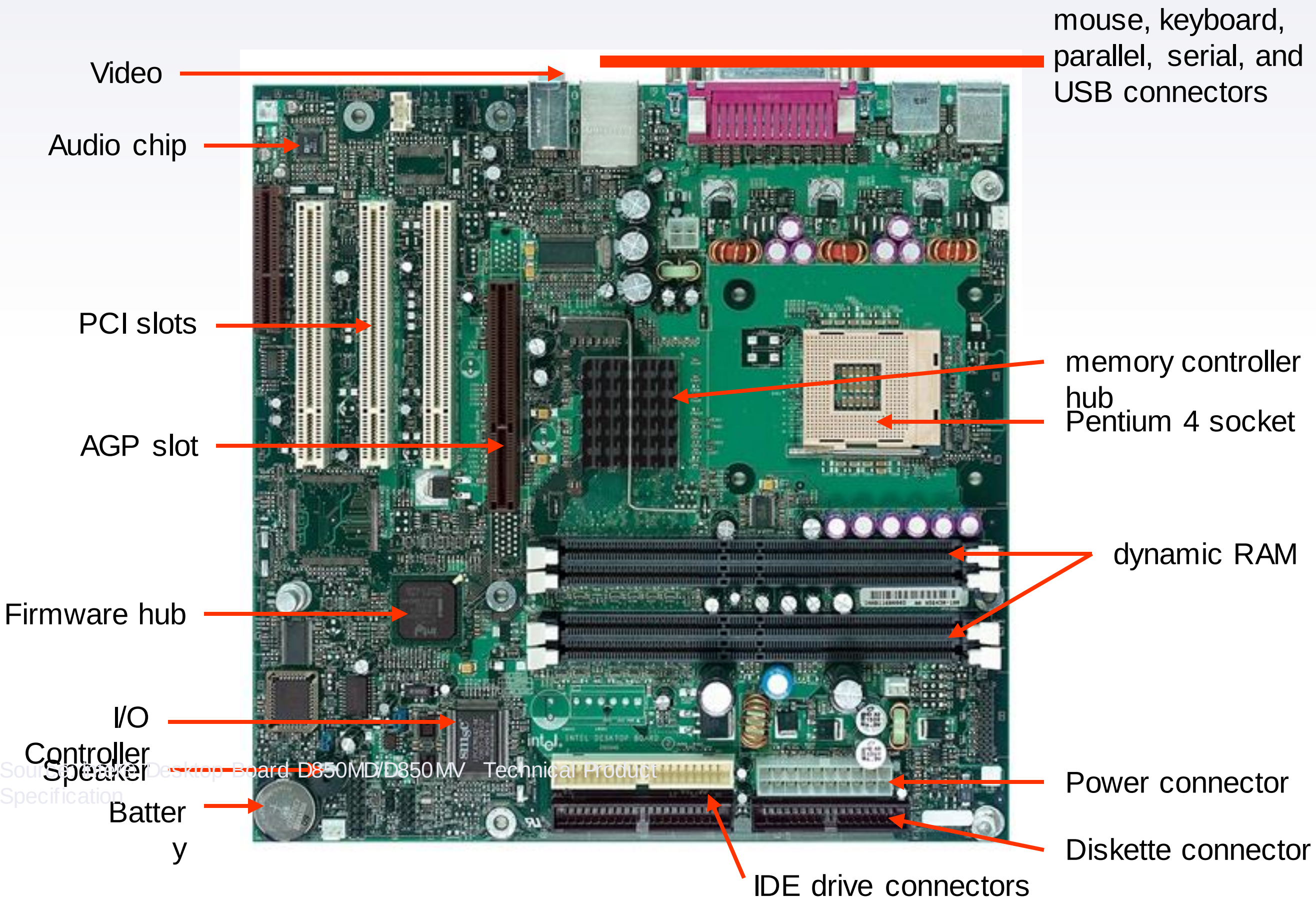
Components of an IA-32 Microcomputer

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

Motherboard

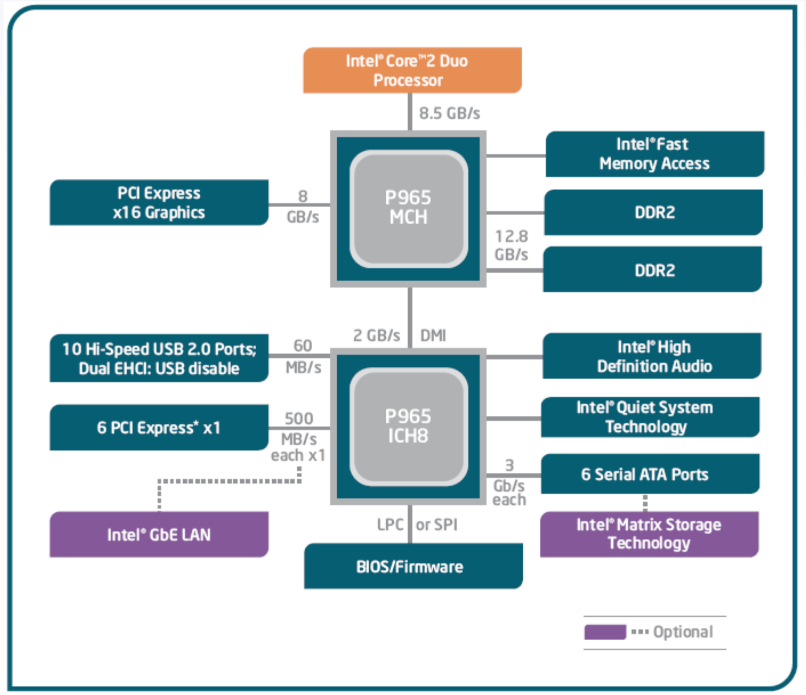
- CPU socket
- External cache memory slots
- Main memory slots
- BIOS chips
- Sound synthesizer chip (optional)
- Video controller chip (optional)
- IDE, parallel, serial, USB, video, keyboard, joystick, network, and mouse connectors
- PCI bus connectors (expansion cards)

Intel D850MD Motherboard



Source: Intel Desktop Board D850MD/D850MV Technical Product Specification

Intel 965 Express Chipset

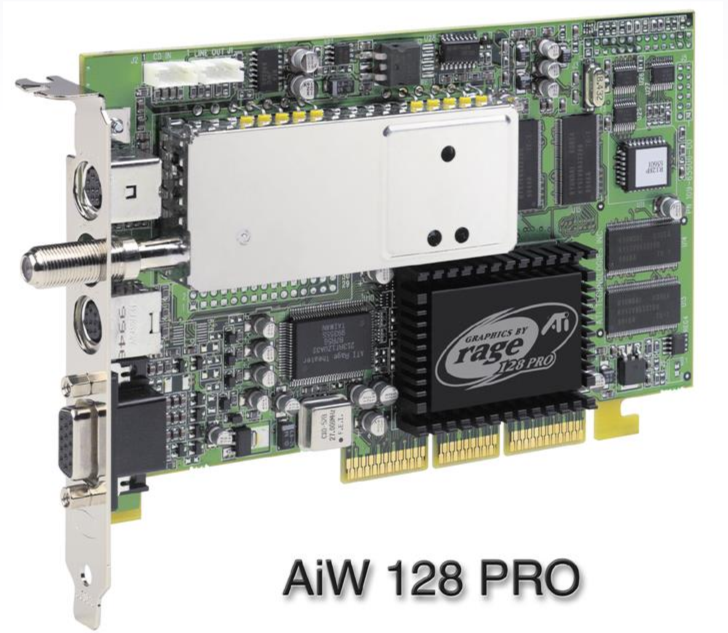


Video Output

- 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

Sample Video Controller (ATI Corp.)

- 128-bit 3D graphics performance powered by RAGE™ 128 PRO
- 3D graphics performance
- Intelligent TV-Tuner with Digital VCR
- TV-ON-DEMAND™
- Interactive Program Guide
- Still image and MPEG-2 motion video capture
- Video editing
- Hardware DVD video playback
- Video output to TV or VCR



AiW 128 PRO

Memory

- ROM
  - read-only memory
- EPROM
  - erasable programmable read-only memory
- Dynamic RAM (DRAM)
  - inexpensive; must be refreshed constantly
- Static RAM (SRAM)
  - expensive; used for cache memory; no refresh required
- Video RAM (VRAM)
  - dual ported; optimized for constant video refresh
- CMOS RAM
  - complimentary metal-oxide semiconductor
  - system setup information
- See: Intel platform memory (Intel technology brief: link address may change)

Input-Output Ports

- USB (universal serial bus)
  - intelligent high-speed connection to devices
  - up to 12 megabits/second
  - USB hub connects multiple devices
  - *enumeration*: computer queries devices
  - supports *hot* connections
- Parallel
  - short cable, high speed
  - common for printers
  - bidirectional, parallel data transfer
  - Intel 8255 controller chip

Input-Output Ports (cont)

- 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

Device Interfaces

- ATA host adapters
  - intelligent drive electronics (hard drive, CDROM)
- SATA (Serial ATA)
  - inexpensive, fast, bidirectional
- FireWire
  - high speed (800 MB/sec), many devices at once
- Bluetooth
  - small amounts of data, short distances, low power usage
- Wi-Fi (wireless Ethernet)
  - IEEE 802.11 standard, faster than Bluetooth

What's Next

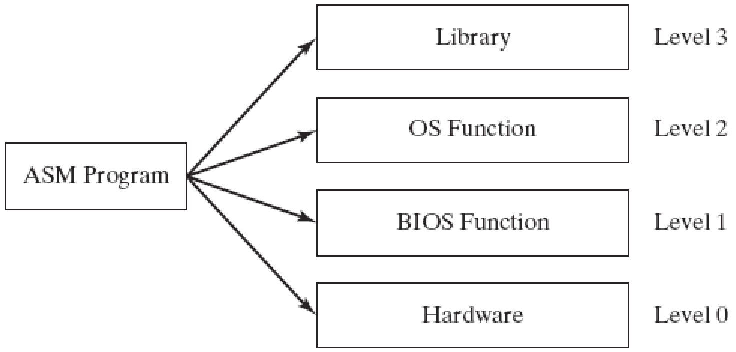
- General Concepts
- IA-32 Processor Architecture
- IA-32 Memory Management
- Components of an IA-32 Microcomputer
- **Input-Output System**

Levels of Input-Output

- Level 3: High-level language function
  - examples: C++, Java
  - portable, convenient, not always the fastest
- Level 2: Operating system
  - Application Programming Interface (API)
  - extended capabilities, lots of details to master
- Level 1: BIOS
  - drivers that communicate directly with devices
  - OS security may prevent application-level code from working at this level

Programming levels

Assembly language programs can perform input-output at each of the following levels:



Summary

- Central Processing Unit (CPU)
- Arithmetic Logic Unit (ALU)
- Instruction execution cycle
- Multitasking
- Floating Point Unit (FPU)
- Complex Instruction Set
- Real mode and Protected mode
- Motherboard components
- Memory types
- Input/Output and access levels