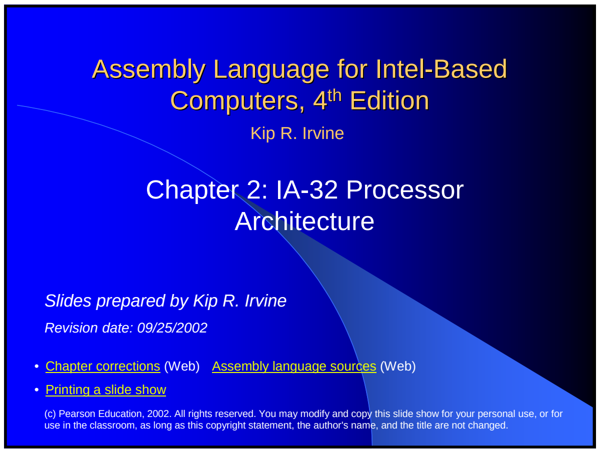

Assembly Language for Intel-Based Computers, 4th Edition

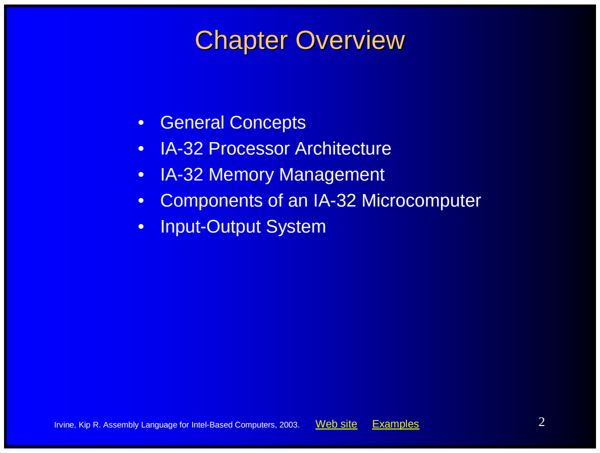
Kip R. Irvine

Chapter 2: IA-32 Processor Architecture

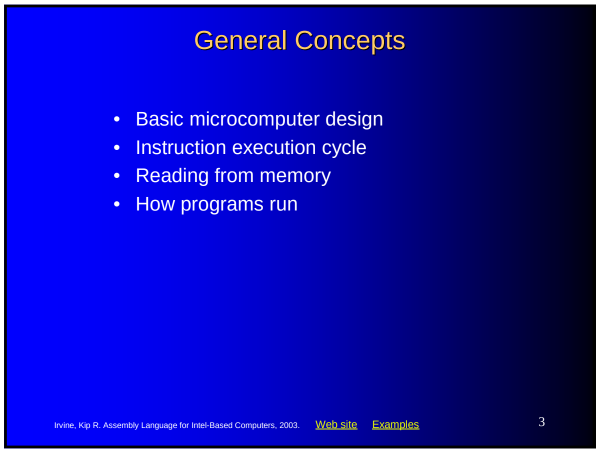
Slides prepared by Kip R. Irvine

Revision date: 09/25/2002

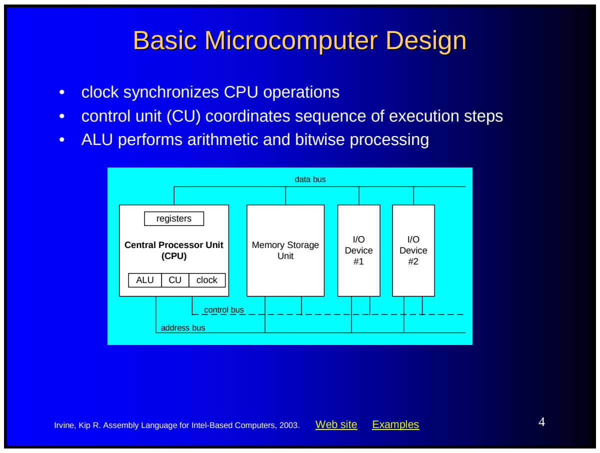
- Chapter corrections (Web)   Assembly language sources (Web)
- Printing a slide show


Chapter Overview

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

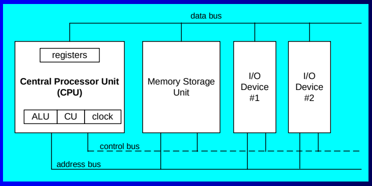

General Concepts

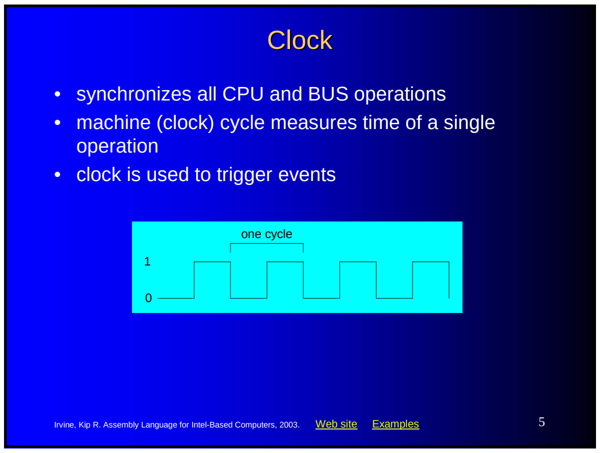
- Basic microcomputer design
- Instruction execution cycle
- Reading from memory
- How programs run



Basic Microcomputer Design

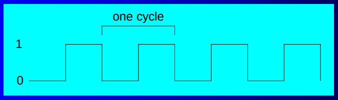
- clock synchronizes CPU operations
- control unit (CU) coordinates sequence of execution steps
- ALU performs arithmetic and bitwise processing

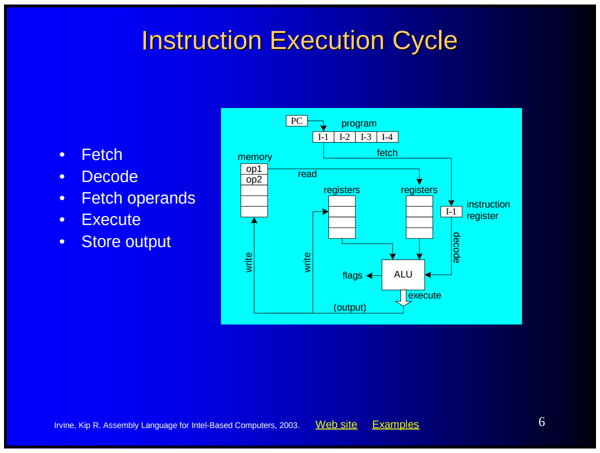




Clock

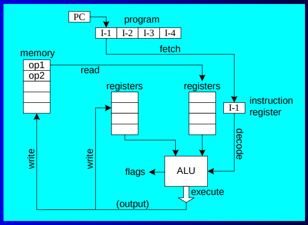
- synchronizes all CPU and BUS operations
- machine (clock) cycle measures time of a single operation
- clock is used to trigger events

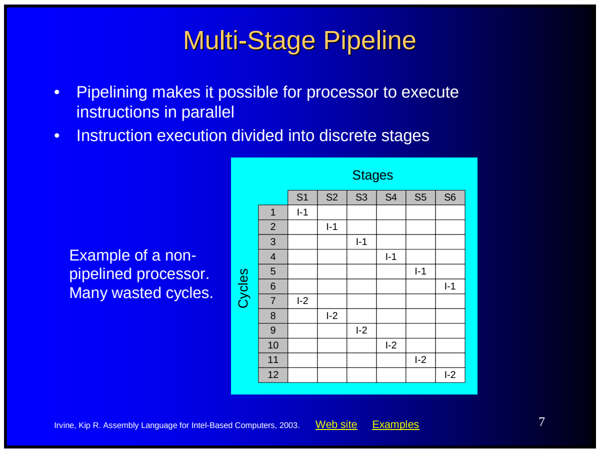




Instruction Execution Cycle

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





Multi-Stage Pipeline

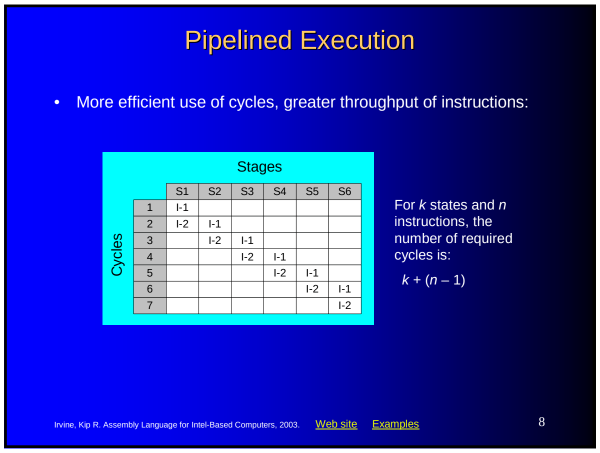
- Pipelining makes it possible for processor to execute instructions in parallel
- Instruction execution divided into discrete stages

Example of a non-pipelined processor. Many wasted cycles.

Stages

Cycles	S1	S2	S3	S4	S5	S6
1	I-1					
2		I-1				
3			I-1			
4				I-1		
5					I-1	
6						I-1
7	I-2					
8		I-2				
9			I-2			
10				I-2		
11					I-2	
12						I-2

Irvine, Kip R. Assembly Language for Intel-Based Computers, 2003. Web site Examples 7



Pipelined Execution

- More efficient use of cycles, greater throughput of instructions:

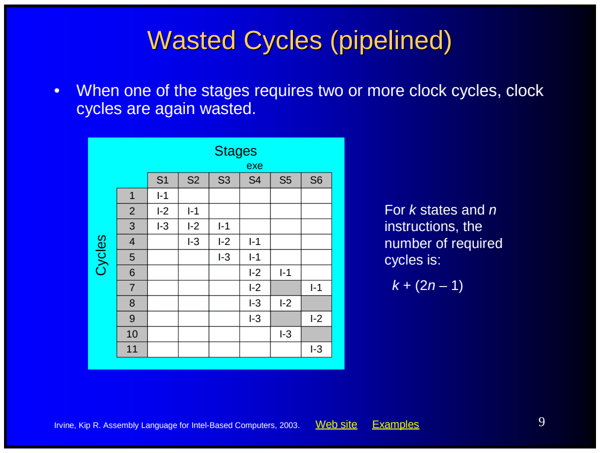
Stages

Cycles	S1	S2	S3	S4	S5	S6
1	I-1					
2	I-2	I-1				
3		I-2	I-1			
4			I-2	I-1		
5				I-2	I-1	
6					I-2	I-1
7						I-2

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

$k + (n - 1)$

Irvine, Kip R. Assembly Language for Intel-Based Computers, 2003. Web site Examples 8



Wasted Cycles (pipelined)

- When one of the stages requires two or more clock cycles, clock cycles are again wasted.

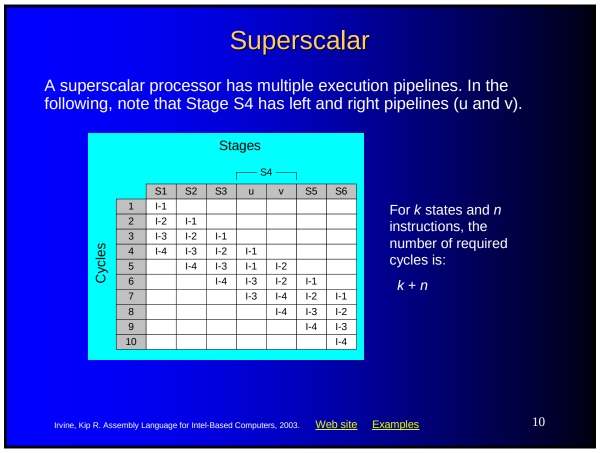
Stages

				exe		
Cycles	S1	S2	S3	S4	S5	S6
1	I-1					
2	I-2	I-1				
3	I-3	I-2	I-1			
4		I-3	I-2	I-1		
5			I-3	I-1		
6				I-2	I-1	
7				I-2	▨	I-1
8				I-3	I-2	▨
9				I-3	▨	I-2
10					I-3	▨
11						I-3

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

$k + (2n - 1)$

Irvine, Kip R. Assembly Language for Intel-Based Computers, 2003. Web site Examples 9



Superscalar

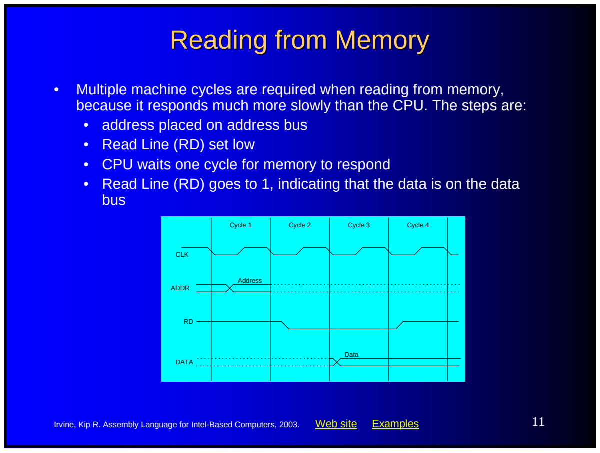
A superscalar processor has multiple execution pipelines. In the following, note that Stage S4 has left and right pipelines (u and v).

Stages

Cycles	S1	S2	S3	S4 u	S4 v	S5	S6
1	I-1						
2	I-2	I-1					
3	I-3	I-2	I-1				
4	I-4	I-3	I-2	I-1			
5		I-4	I-3	I-1	I-2		
6			I-4	I-3	I-2	I-1	
7				I-3	I-4	I-2	I-1
8					I-4	I-3	I-2
9						I-4	I-3
10							I-4

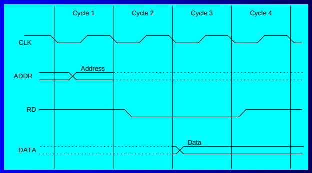
For *k* states and *n* instructions, the number of required cycles is:

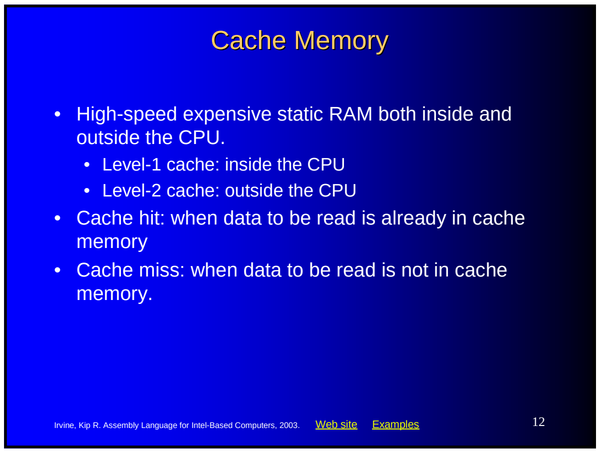
k + *n*



Reading from Memory

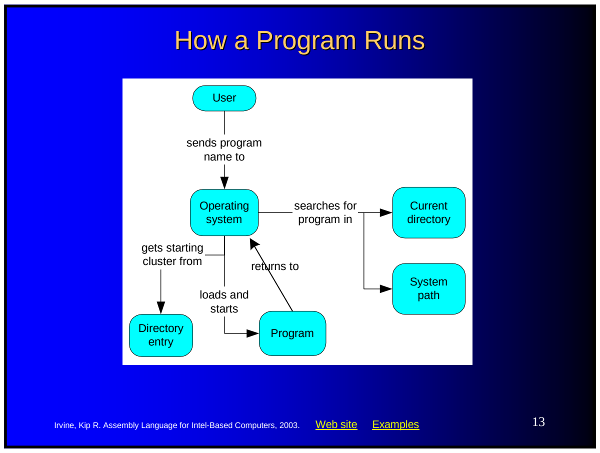
- Multiple machine cycles are required when reading from memory, because it responds much more slowly than the CPU. The steps are:
  - address placed on address bus
  - Read Line (RD) set low
  - CPU waits one cycle for memory to respond
  - Read Line (RD) goes to 1, indicating that the data is on the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



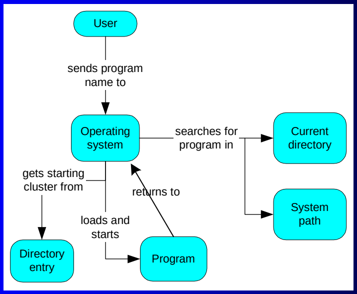


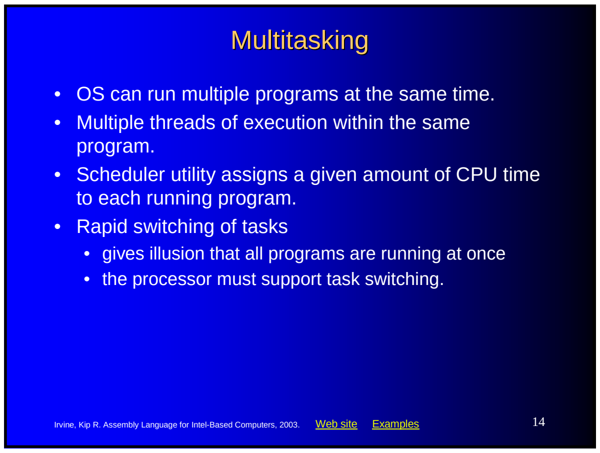
Cache Memory

- 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.



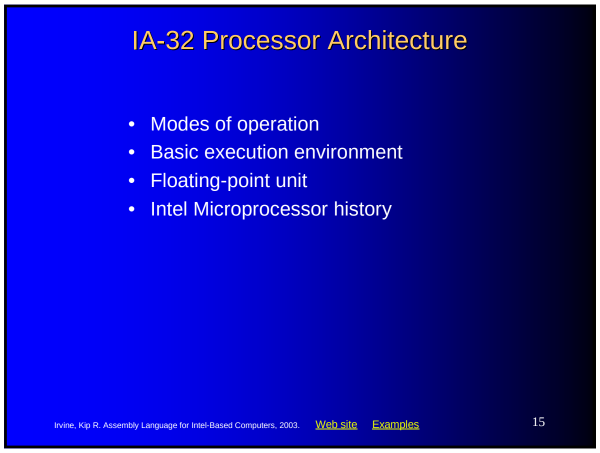
How a Program Runs





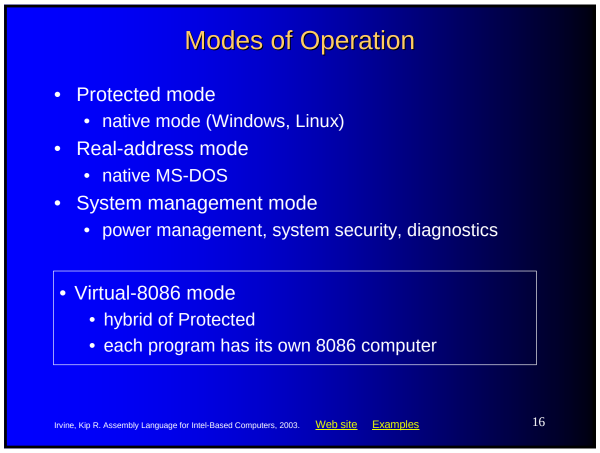
Multitasking

- 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

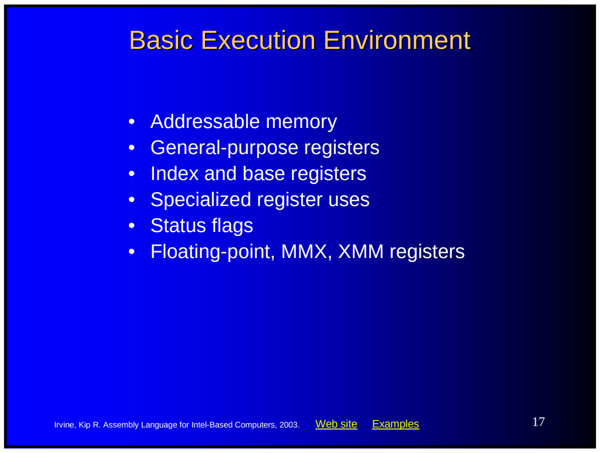
- Modes of operation
- Basic execution environment
- Floating-point unit
- Intel Microprocessor history



Modes of Operation

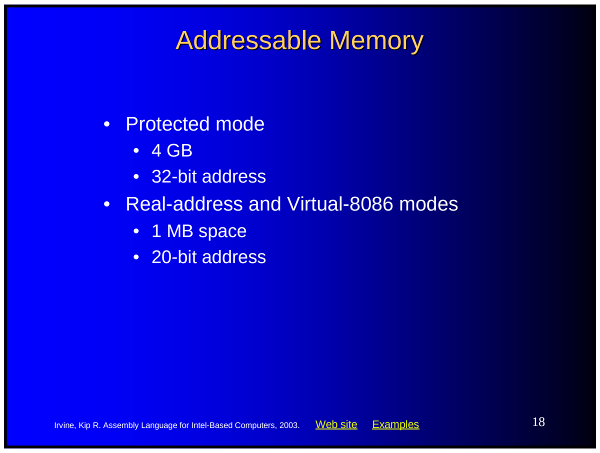
- Protected mode
  - native mode (Windows, Linux)
- Real-address mode
  - native MS-DOS
- System management mode
  - power management, system security, diagnostics

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



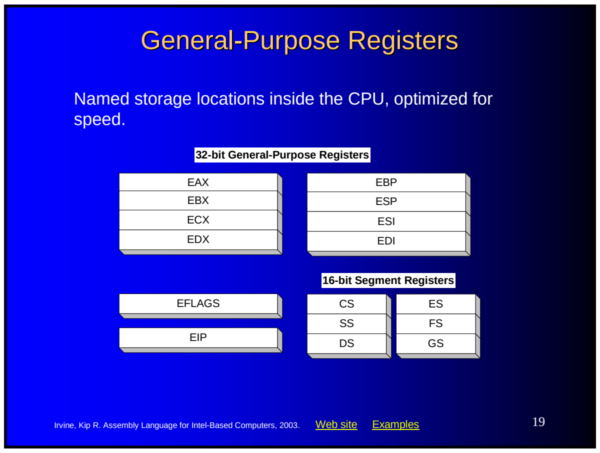
Basic Execution Environment

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



Addressable Memory

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



General-Purpose Registers

Named storage locations inside the CPU, optimized for speed.

32-bit General-Purpose Registers

EAX	EBP
EBX	ESP
ECX	ESI
EDX	EDI

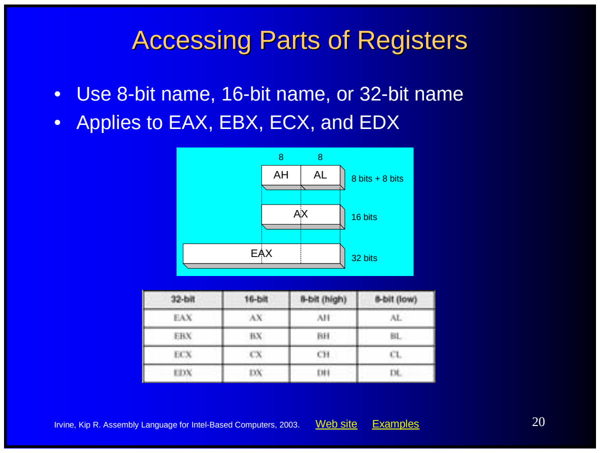
EFLAGS

EIP

16-bit Segment Registers

CS	ES
SS	FS
DS	GS

Irvine, Kip R. Assembly Language for Intel-Based Computers, 2003. Web site Examples 19



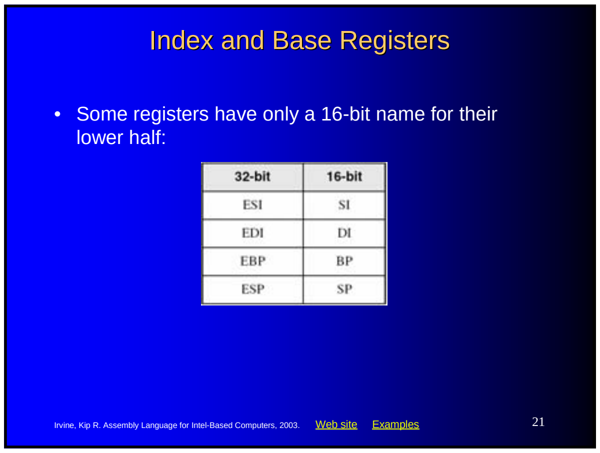
Accessing Parts of Registers

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

8
8
AH
AL
8 bits + 8 bits
AX
16 bits
EAX
32 bits

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

Irvine, Kip R. Assembly Language for Intel-Based Computers, 2003. Web site Examples 20

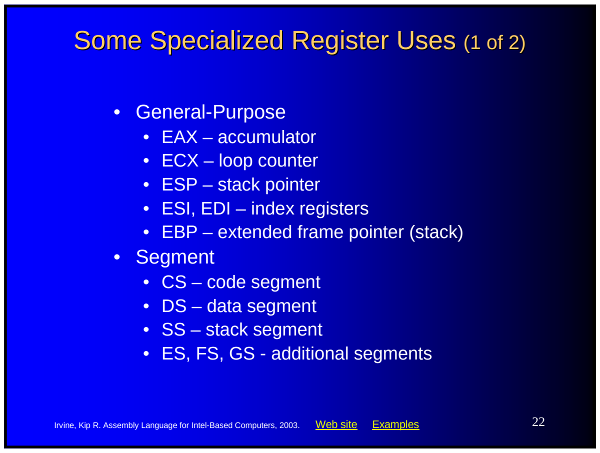


Index and Base Registers

- Some registers have only a 16-bit name for their lower half:

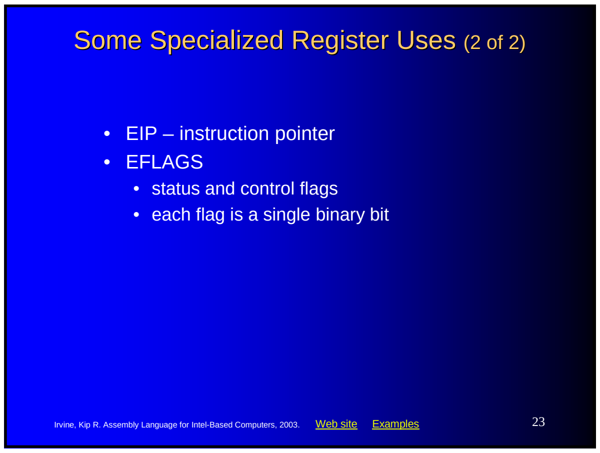
32-bit	16-bit
ESI	SI
EDI	DI
EBP	BP
ESP	SP

Irvine, Kip R. Assembly Language for Intel-Based Computers, 2003. Web site Examples 21



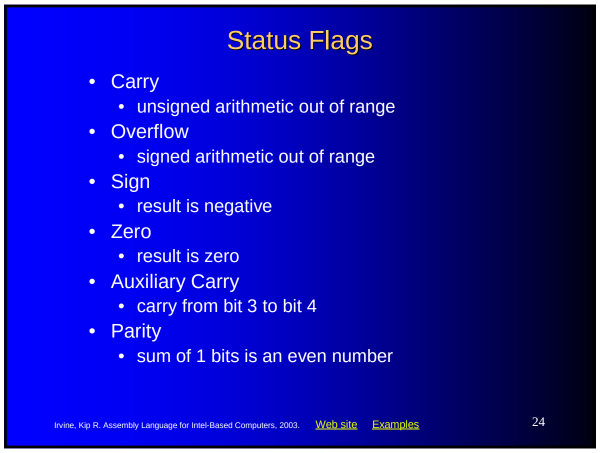
Some Specialized Register Uses (1 of 2)

- General-Purpose
  - EAX – accumulator
  - ECX – loop counter
  - ESP – stack pointer
  - ESI, EDI – index registers
  - EBP – extended frame pointer (stack)
- Segment
  - CS – code segment
  - DS – data segment
  - SS – stack segment
  - ES, FS, GS - additional segments



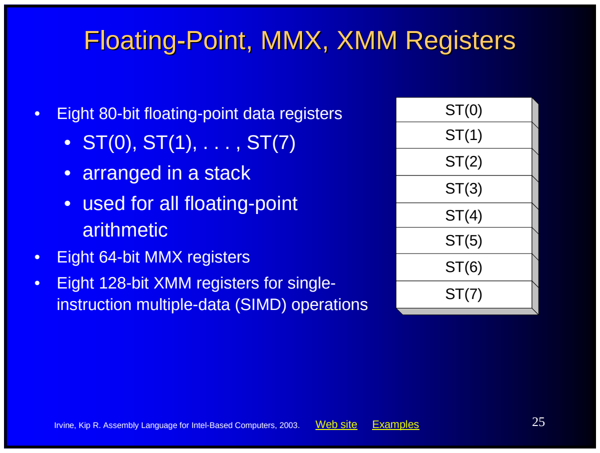
Some Specialized Register Uses (2 of 2)

- EIP – instruction pointer
- EFLAGS
  - status and control flags
  - each flag is a single binary bit



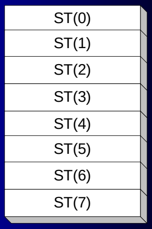
Status Flags

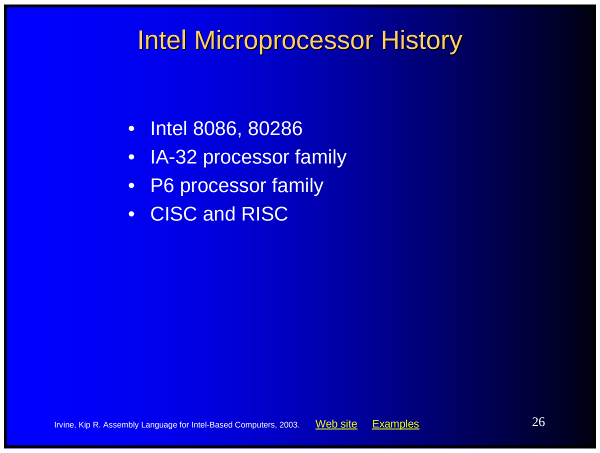
- Carry
  - unsigned arithmetic out of range
- Overflow
  - signed arithmetic out of range
- Sign
  - result is negative
- Zero
  - result is zero
- Auxiliary Carry
  - carry from bit 3 to bit 4
- Parity
  - sum of 1 bits is an even number



Floating-Point, MMX, XMM Registers

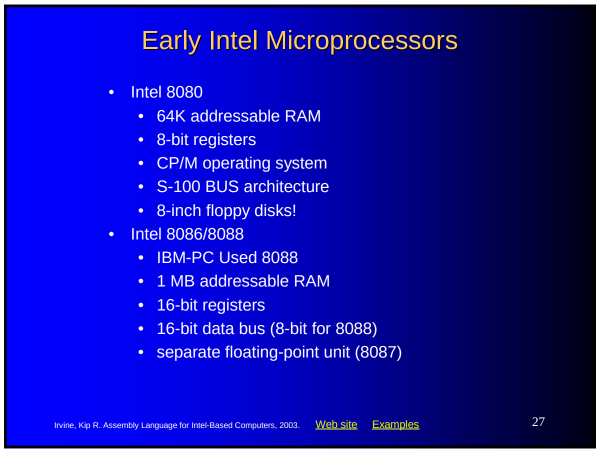
- Eight 80-bit floating-point data registers
  - ST(0), ST(1), . . . , ST(7)
  - arranged in a stack
  - used for all floating-point arithmetic
- Eight 64-bit MMX registers
- Eight 128-bit XMM registers for single-instruction multiple-data (SIMD) operations





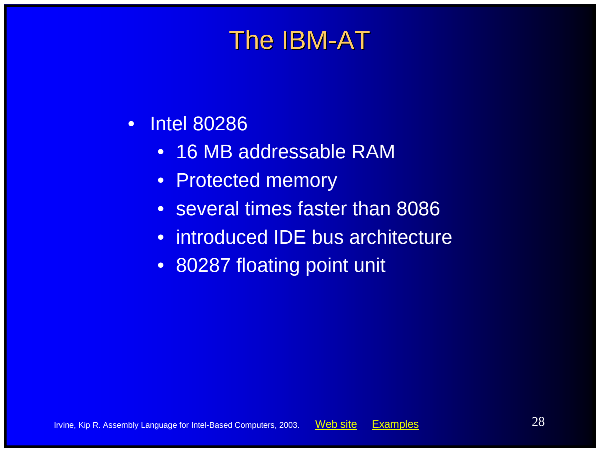
Intel Microprocessor History

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



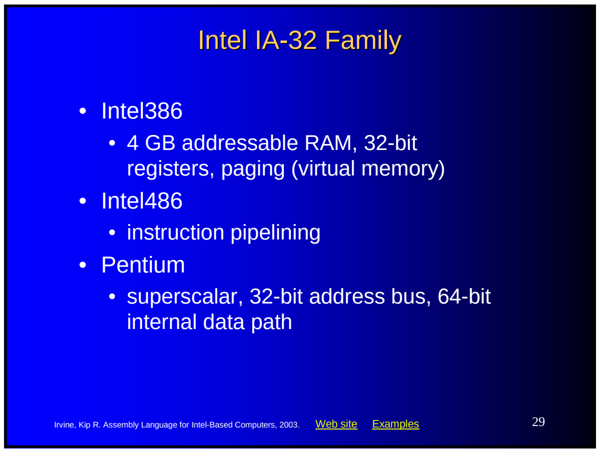
Early Intel Microprocessors

- 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)



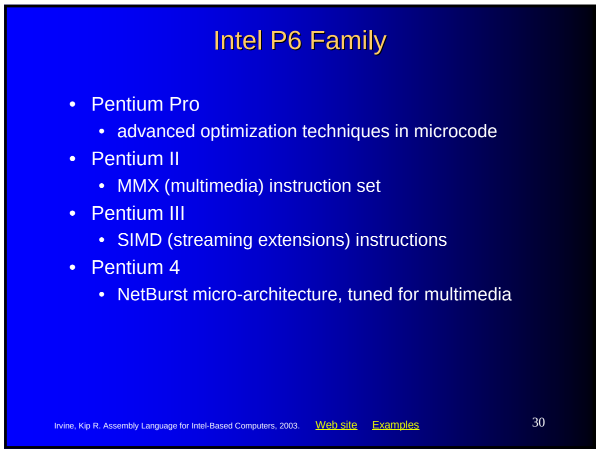
The IBM-AT

- Intel 80286
  - 16 MB addressable RAM
  - Protected memory
  - several times faster than 8086
  - introduced IDE bus architecture
  - 80287 floating point unit



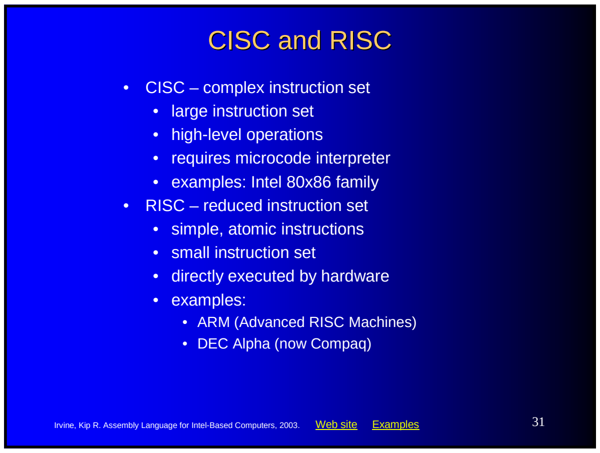
Intel IA-32 Family

- 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



Intel P6 Family

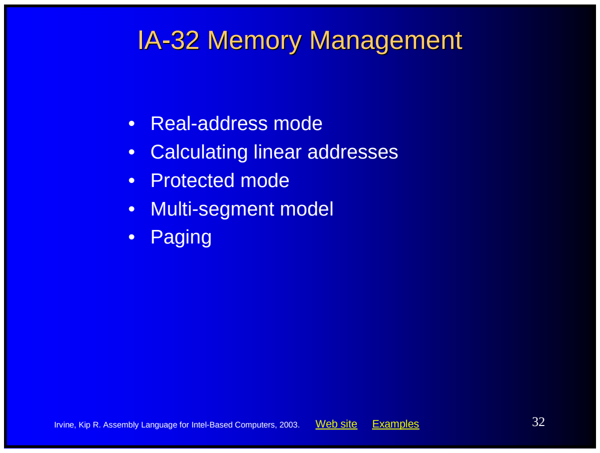
- Pentium Pro
  - advanced optimization techniques in microcode
- Pentium II
  - MMX (multimedia) instruction set
- Pentium III
  - SIMD (streaming extensions) instructions
- Pentium 4
  - NetBurst micro-architecture, tuned for multimedia



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)

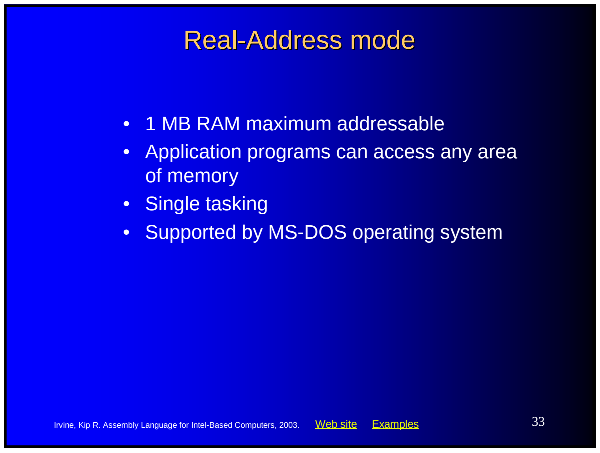
Irvine, Kip R. Assembly Language for Intel-Based Computers, 2003. Web site Examples 31



IA-32 Memory Management

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

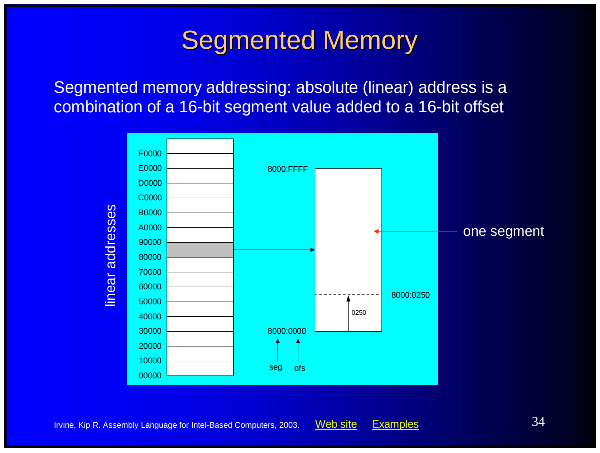
Irvine, Kip R. Assembly Language for Intel-Based Computers, 2003. Web site Examples 32



Real-Address mode

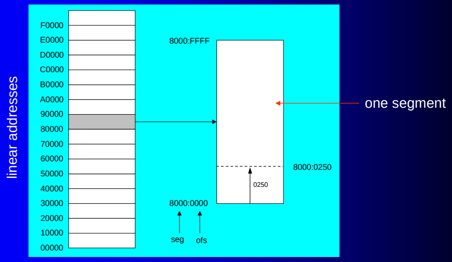
- 1 MB RAM maximum addressable
- Application programs can access any area of memory
- Single tasking
- Supported by MS-DOS operating system

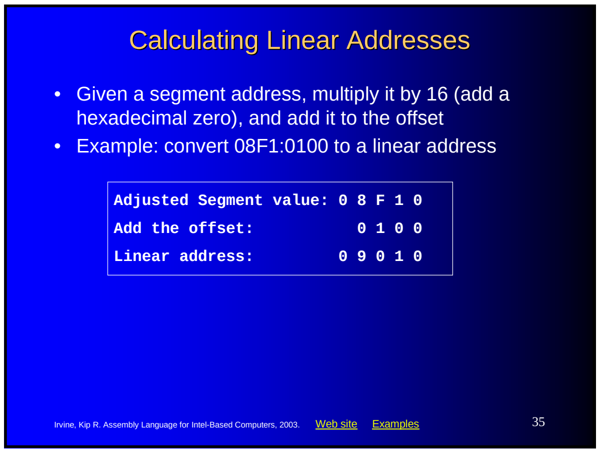
Irvine, Kip R. Assembly Language for Intel-Based Computers, 2003. Web site Examples 33



Segmented Memory

Segmented memory addressing: absolute (linear) address is a combination of a 16-bit segment value added to a 16-bit offset

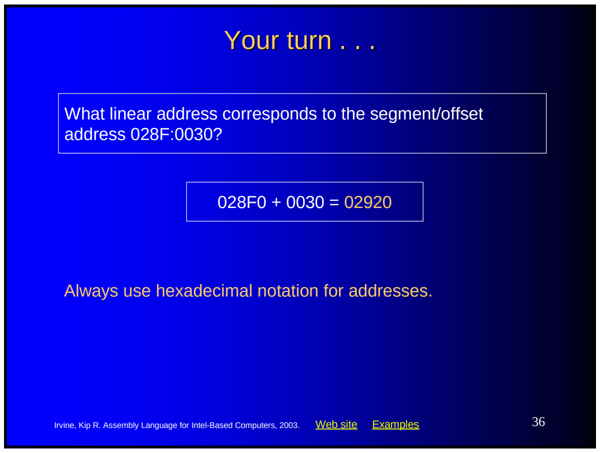




Calculating Linear Addresses

- 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
```

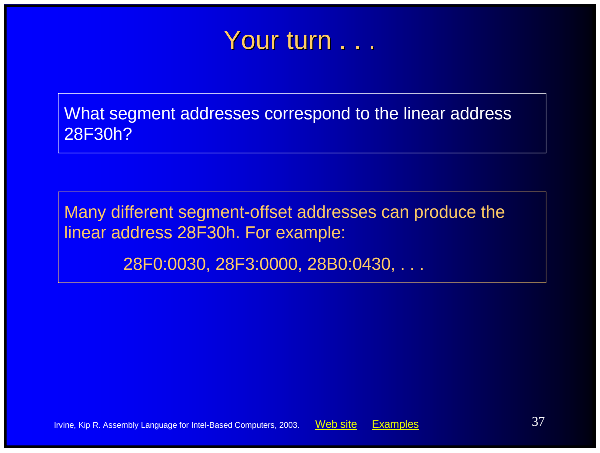


Your turn . . .

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

028F0 + 0030 = 02920

Always use hexadecimal notation for addresses.

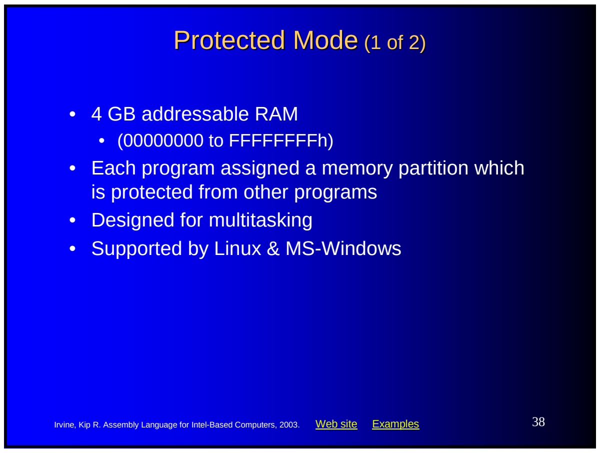


Your turn . . .

What segment addresses correspond to the linear address 28F30h?

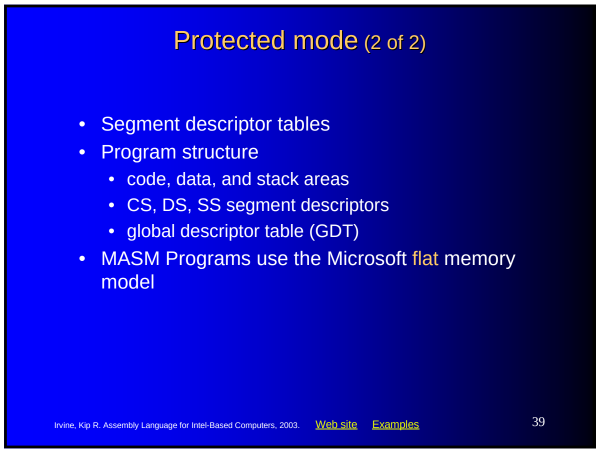
Many different segment-offset addresses can produce the linear address 28F30h. For example:

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



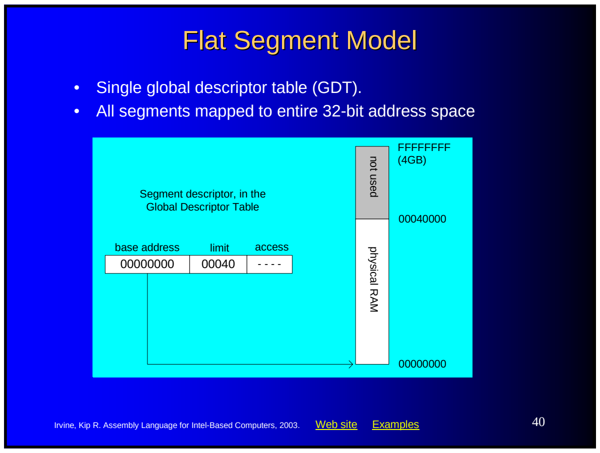
Protected Mode (1 of 2)

- 4 GB addressable RAM
  - (00000000 to FFFFFFFFh)
- Each program assigned a memory partition which is protected from other programs
- Designed for multitasking
- Supported by Linux & MS-Windows



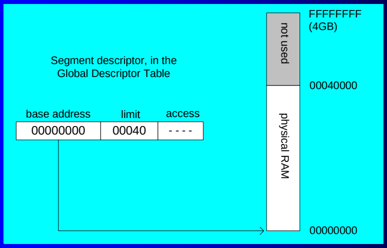
Protected mode (2 of 2)

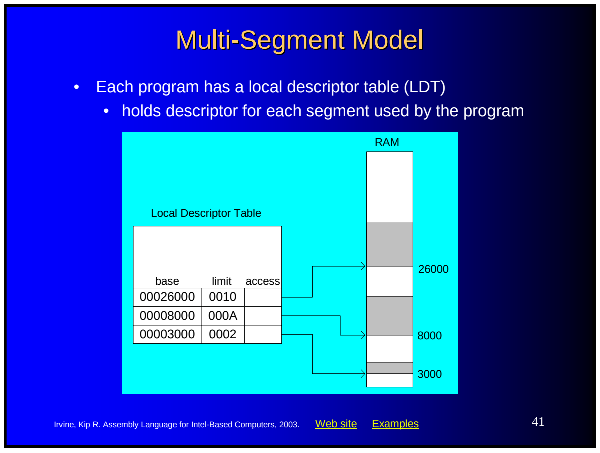
- Segment descriptor tables
- Program structure
  - code, data, and stack areas
  - CS, DS, SS segment descriptors
  - global descriptor table (GDT)
- MASM Programs use the Microsoft flat memory model



Flat Segment Model

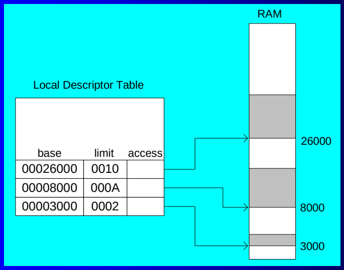
- Single global descriptor table (GDT).
- All segments mapped to entire 32-bit address space



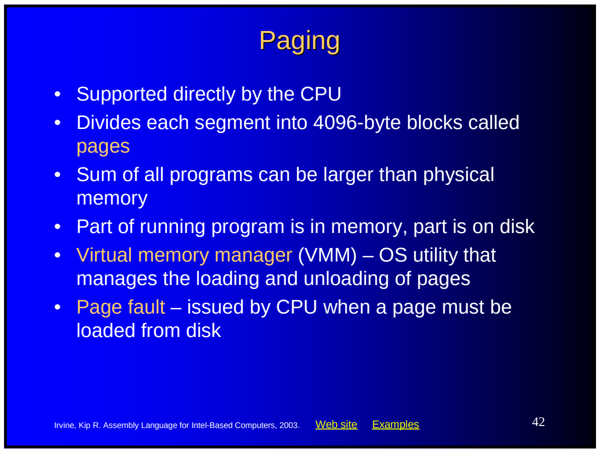


Multi-Segment Model

- Each program has a local descriptor table (LDT)
  - holds descriptor for each segment used by the program



base	limit	access
00026000	0010	
00008000	000A	
00003000	0002	



Paging

- Supported directly by the CPU
- Divides each segment into 4096-byte blocks called pages
- Sum of all programs can be larger than physical memory
- Part of running program is in memory, part is on disk
- Virtual memory manager (VMM) – OS utility that manages the loading and unloading of pages
- Page fault – issued by CPU when a page must be loaded from disk

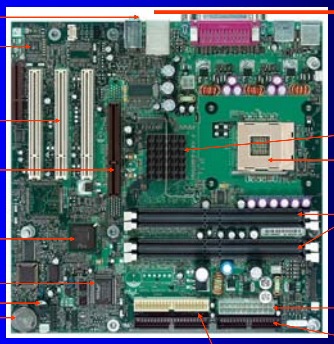
Components of an IA-32 Microcomputer

- 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

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

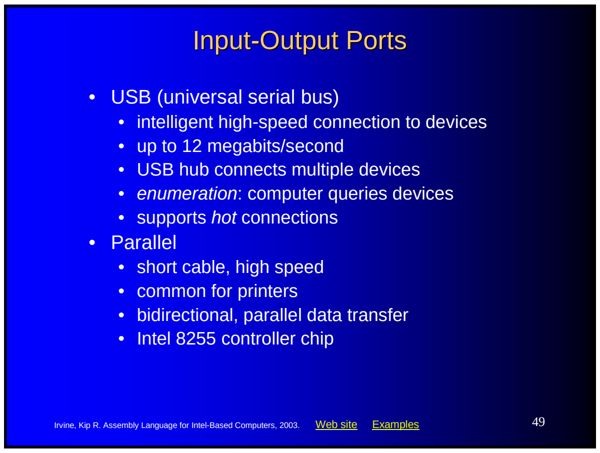
* This link may change over time.

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

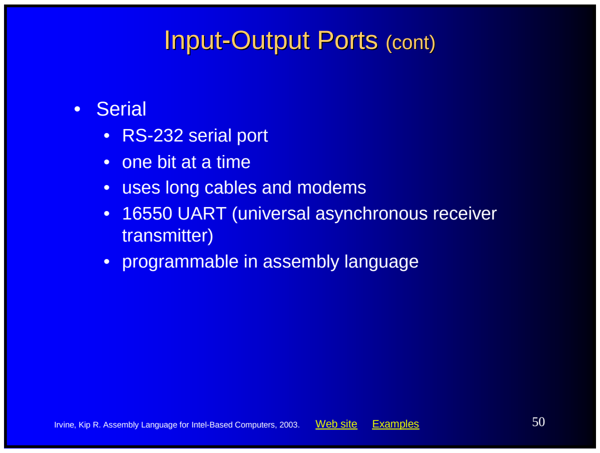
Memory

- 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)



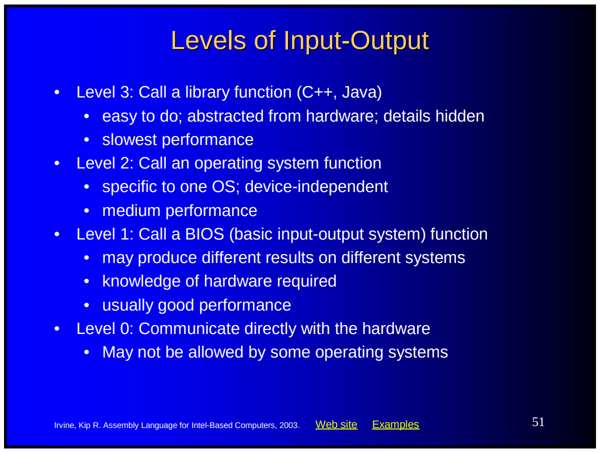
Input-Output Ports

- 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



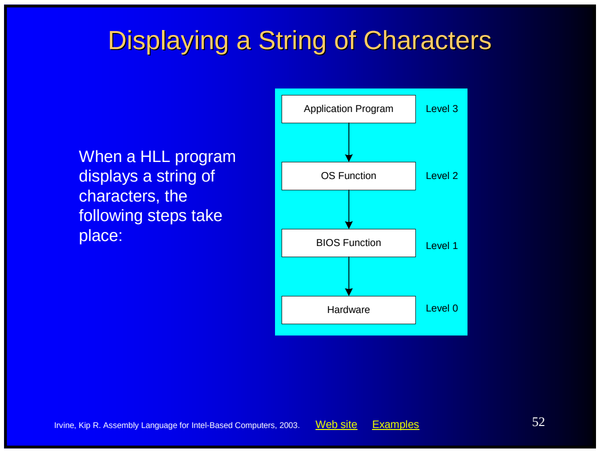
Input-Output Ports (cont)

- 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



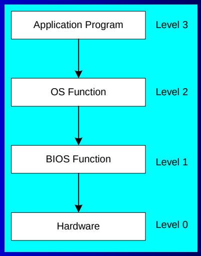
Levels of Input-Output

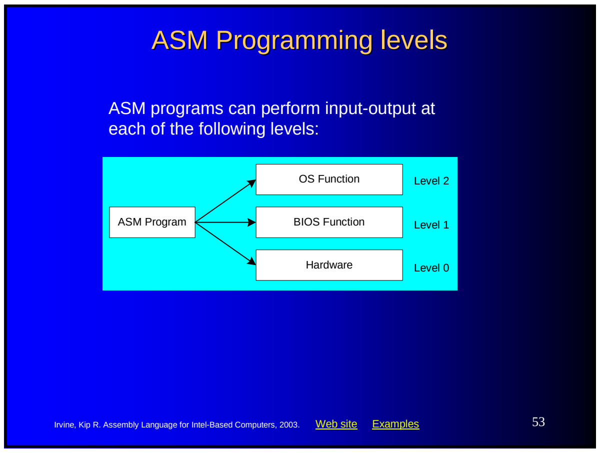
- Level 3: Call a library function (C++, Java)
  - easy to do; abstracted from hardware; details hidden
  - slowest performance
- Level 2: Call an operating system function
  - specific to one OS; device-independent
  - medium performance
- Level 1: Call a BIOS (basic input-output system) function
  - may produce different results on different systems
  - knowledge of hardware required
  - usually good performance
- Level 0: Communicate directly with the hardware
  - May not be allowed by some operating systems



Displaying a String of Characters

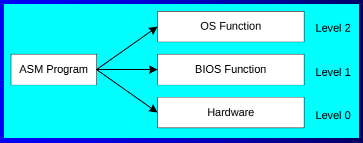
When a HLL program displays a string of characters, the following steps take place:

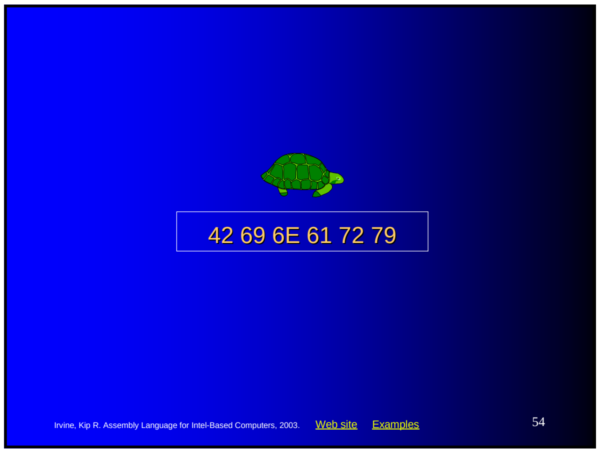


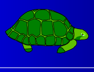


ASM Programming levels

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







42 69 6E 61 72 79