IA-32 Architecture

CSC321

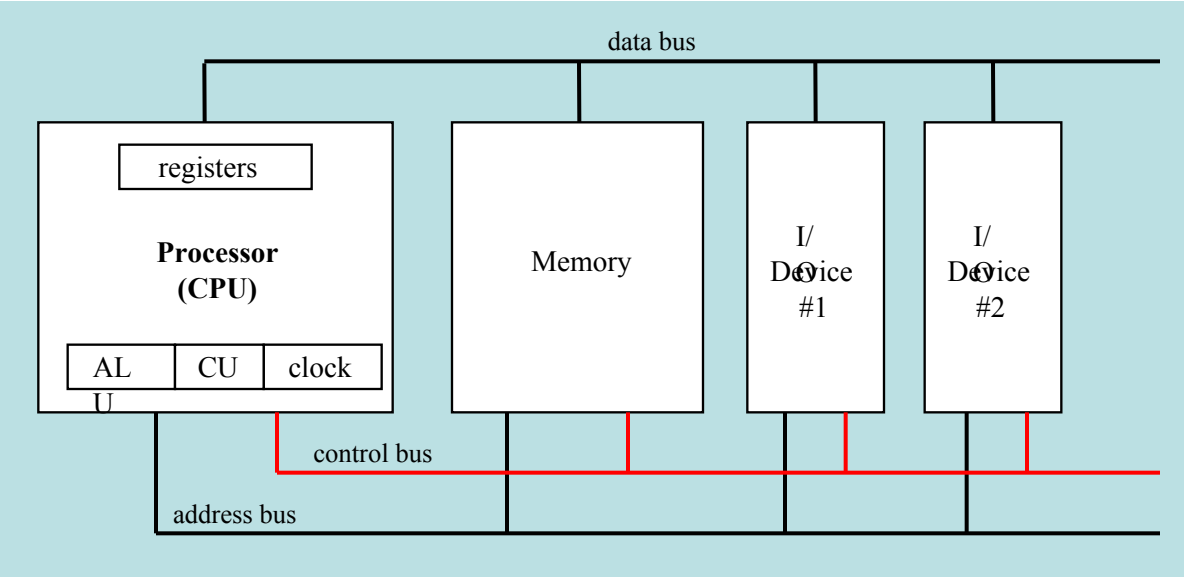
Microprocessor and Assembly Language

Presentation Outline

- Basic Computer Organization
- Intel Microprocessors
- IA-32 Registers
- Instruction Execution Cycle
- IA-32 Memory Management

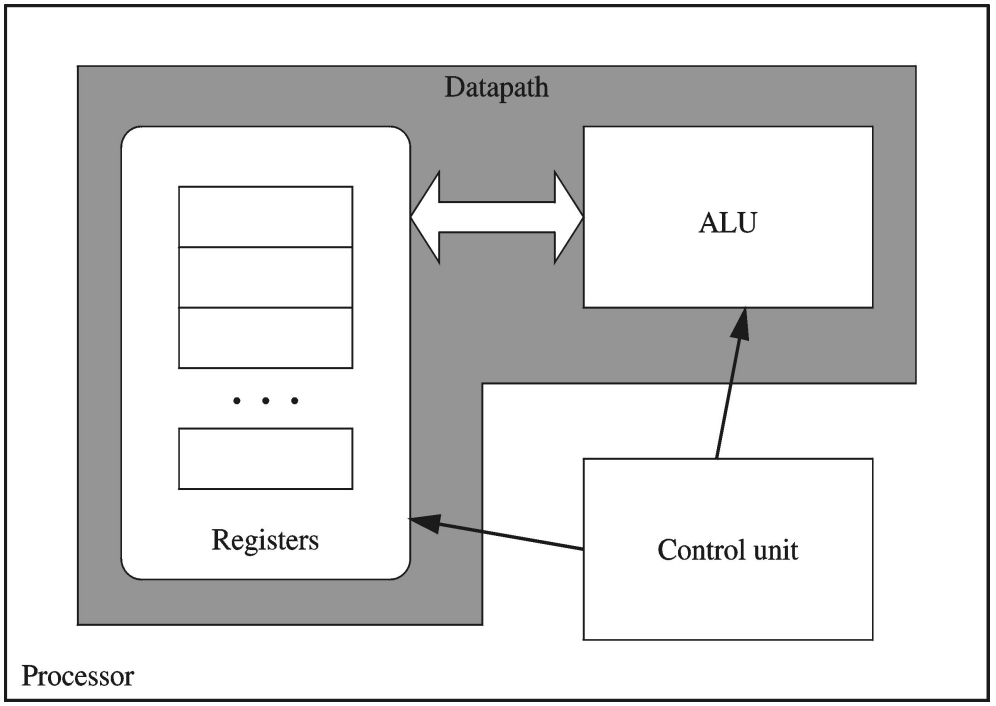
Basic Computer Organization

- Since the 1940's, computers have 3 classic components:
  - Processor, called also the CPU (Central Processing Unit)
  - Memory and Storage Devices
  - I/O Devices
- Interconnected with one or more buses
- Bus consists of
  - Data Bus
  - Address Bus
  - Control Bus



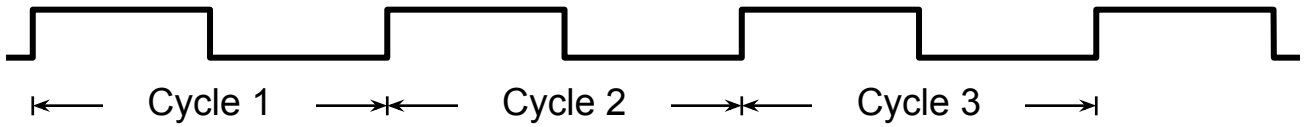
Processor

- Processor consists of
  - Datapath
    - ALU
    - Registers
  - Control unit
- ALU
  - Performs arithmetic and logic instructions
- Control unit (CU)
  - Generates the control signals required to execute instructions
- Implementation varies from one processor to another



Clock

- Synchronizes Processor and Bus operations

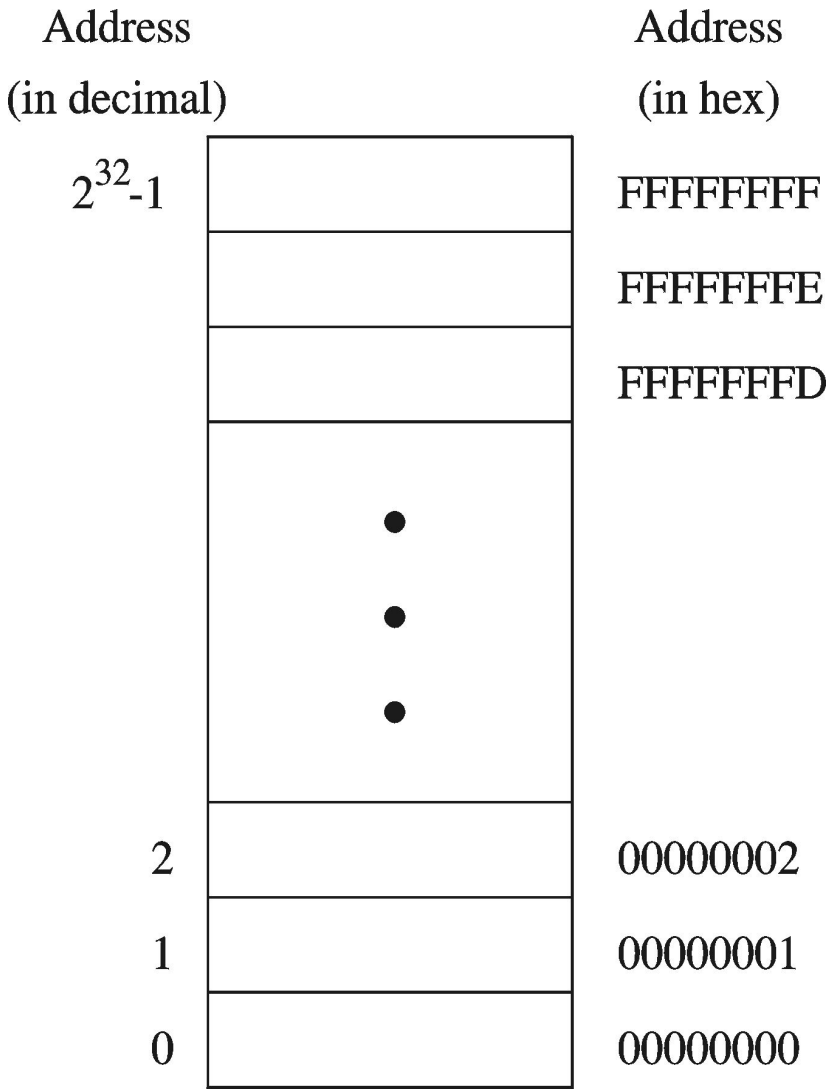


- Clock rate = Clock frequency = Cycles per second
  - 1 Hz = 1 cycle/sec 1 KHz = 10^3 cycles/sec
  - 1 MHz = 10^6 cycles/sec 1 GHz = 10^9 cycles/sec
  - 2 GHz clock has a cycle time = $1/(2\times10^9)$ = 0.5 nanosecond (ns)
- Clock cycles measure the execution of instructions

Memory

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

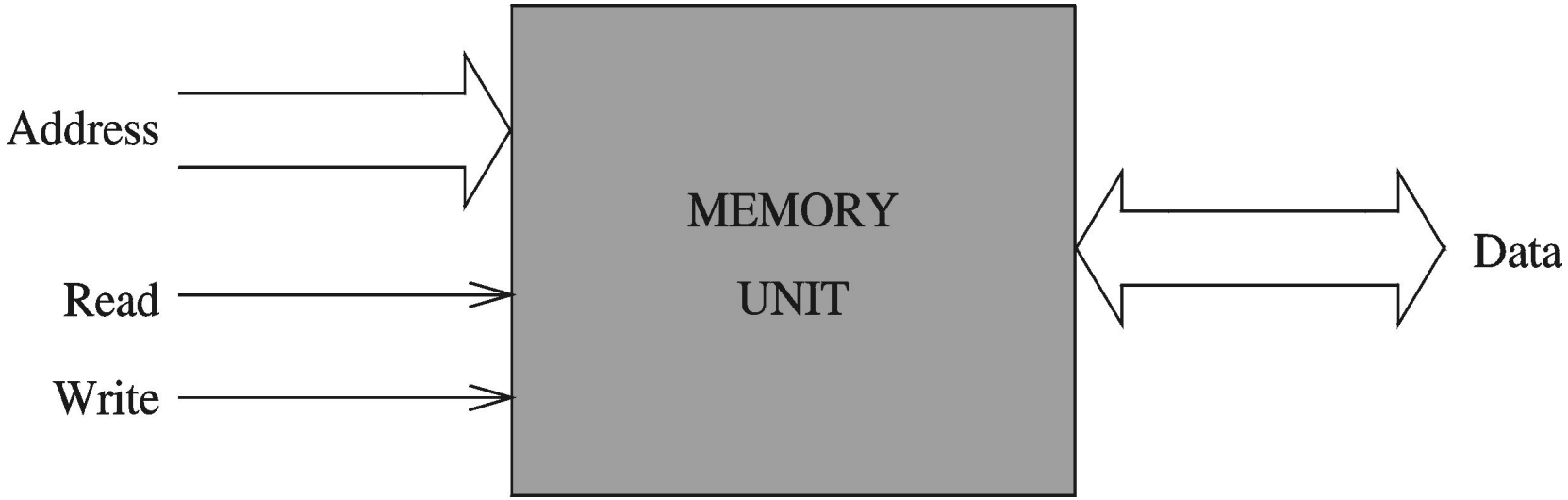
Address Space



Address Space is the set of memory locations (bytes) that can be addressed

Memory Unit

- Address Bus
  - Address is placed on the address bus
  - Address of location to be read/written
- Data Bus
  - Data is placed on the data bus
- Two Control Signals
  - Read
  - Write
  - Control whether memory should be read or written



Memory Read and Write Cycles

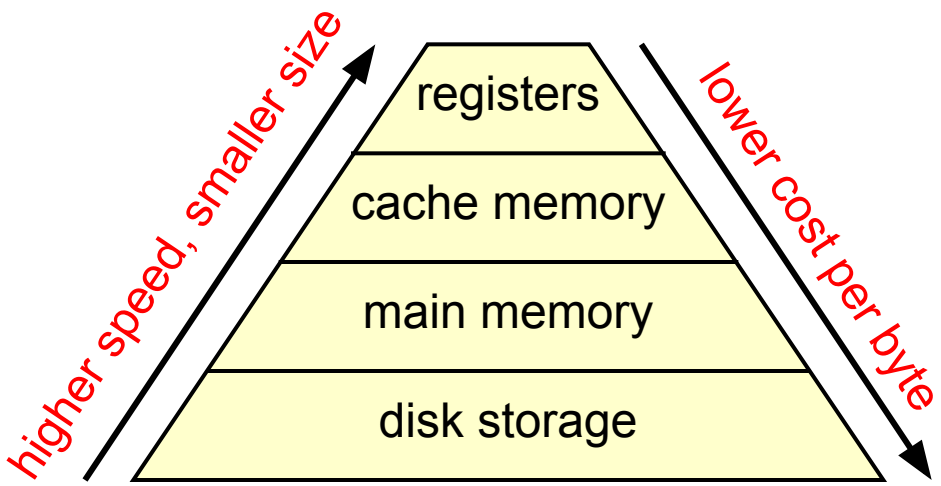
- Read cycle
  1. Processor places address on the address bus
  2. Processor asserts the memory read control signal
  3. Processor waits for memory to place the data on the data bus
  4. Processor reads the data from the data bus
  5. Processor drops the memory read signal
- Write cycle
  1. Processor places address on the address bus
  2. Processor asserts the memory write control signal
  3. Processor places the data on the data bus
  4. Wait for memory to store the data (wait states for slow memory)
  5. Processor drops the memory write signal

Memory Devices

- ROM = Read-Only Memory
  - Stores information permanently (non-volatile)
  - Used to store the information required to startup the computer
  - Many types: ROM, EPROM, EEPROM, and FLASH
- RAM = Random Access Memory
  - Volatile memory: data is lost when device is powered off
  - Dynamic RAM (DRAM)
    - Inexpensive, used for main memory, must be refreshed constantly
  - Static RAM (SRAM)
    - Expensive, used for cache memory, faster access, no refresh
  - Video RAM (VRAM)
    - Responsible for graphical related tasks

Memory Hierarchy

- Registers
  - Fastest storage elements, stores most frequently used data
  - General-purpose registers: accessible to the programmer
  - Special-purpose registers: used internally by the microprocessor
- Cache Memory
  - Fast SRAM that stores recently used instructions and data
- Main Memory (DRAM)
- Disk Storage
  - Permanent magnetic storage for files



Next ...

- Basic Computer Organization
- Intel Microprocessors
- IA-32 Registers
- Instruction Execution Cycle
- IA-32 Memory Management

Intel Microprocessors

- Intel introduced the 8086 microprocessor in 1979
- 8086, 8087, 8088, and 80186 processors
  - 16-bit processors with 16-bit registers
  - 16-bit data bus and 20-bit address bus
    - Physical address space = 2^{20} bytes = 1 MB
  - 8087 Floating-Point co-processor
  - Uses segmentation and real-address mode to address memory
    - Each segment can address 2^{16} bytes = 64 KB
  - 8088 is a less expensive version of 8086
    - Uses an 8-bit data bus
  - 80186 is a faster version of 8086

Intel 80286 and 80386 Processors

- 80286 was introduced in 1982
  - 24-bit address bus ⇒ 2^{24} bytes = 16 MB address space
  - Introduced protected mode
    - Segmentation in protected mode is different from the real mode
- 80386 was introduced in 1985
  - First 32-bit processor with 32-bit general-purpose registers
  - First processor to define the IA-32 architecture
  - 32-bit data bus and 32-bit address bus
  - 2^{32} bytes ⇒ 4 GB address space
  - Introduced paging, virtual memory
    - Segmentation can be turned off

Intel 80486 and Pentium Processors

- 80486 was introduced 1989
  - Improved version of Intel 80386
  - On-chip Floating-Point unit (DX versions)
  - Uses Pipelining: can execute up to 1 instruction per clock cycle
- Pentium (80586) was introduced in 1993
  - Wider 64-bit data bus, but address bus is still 32 bits
  - Two execution pipelines: U-pipe and V-pipe
    - Superscalar performance: can execute 2 instructions per clock cycle
  - MMX instructions (later models) for multimedia applications

Intel P6 Processor Family

- P6 Processor Family: Pentium Pro, Pentium II and III
- Pentium Pro was introduced in 1995
  - Three-way superscalar: can execute 3 instructions per clock cycle
  - 36-bit address bus ⇒ up to 64 GB of physical address space
- Pentium II was introduced in 1997
  - Added MMX instructions (already introduced on Pentium MMX)
- Pentium III was introduced in 1999
  - Added SSE instructions and eight new 128-bit XMM registers

CISC and RISC

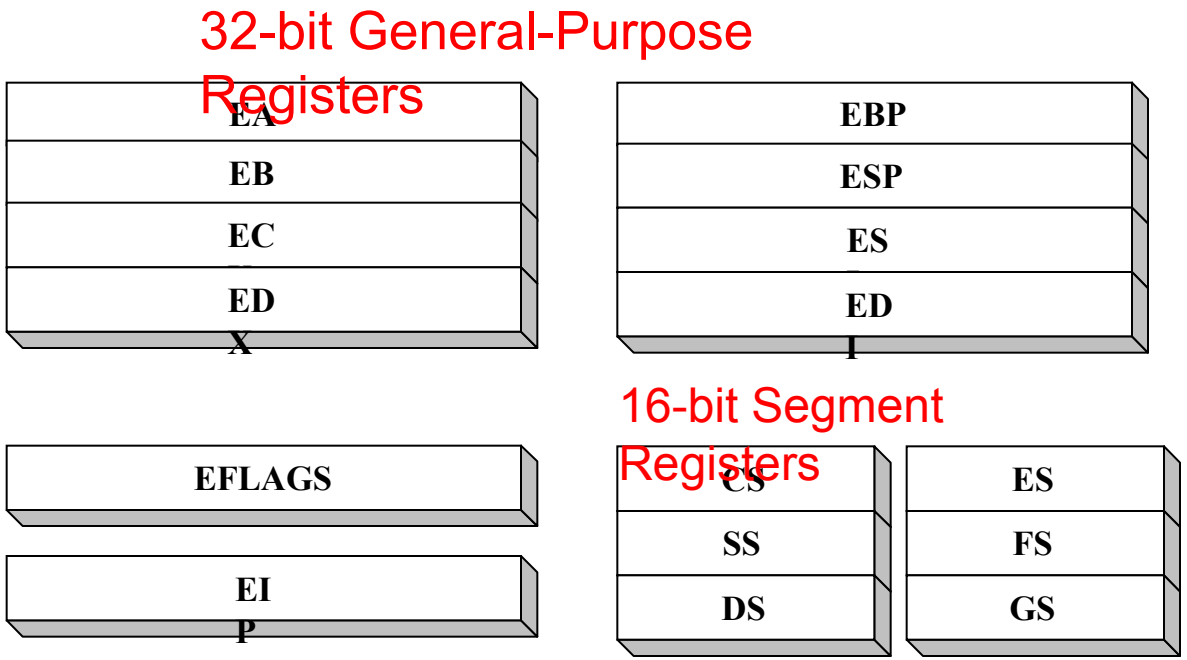
- CISC – Complex Instruction Set Computer
  - Large and complex instruction set
  - Variable width instructions
  - Example: Intel x86 family
- RISC – Reduced Instruction Set Computer
  - Small and simple instruction set
  - All instructions have the same width
  - Simpler instruction formats and addressing modes
  - Examples: ARM, MIPS, PowerPC, SPARC, etc.

Next ...

- Basic Computer Organization
- Intel Microprocessors
- IA-32 Registers
- Instruction Execution Cycle
- IA-32 Memory Management

Basic Program Execution Registers

- Registers are high speed memory inside the CPU
  - Eight 32-bit general-purpose registers
  - Six 16-bit segment registers
  - Processor Status Flags (EFLAGS) and Instruction Pointer (EIP)

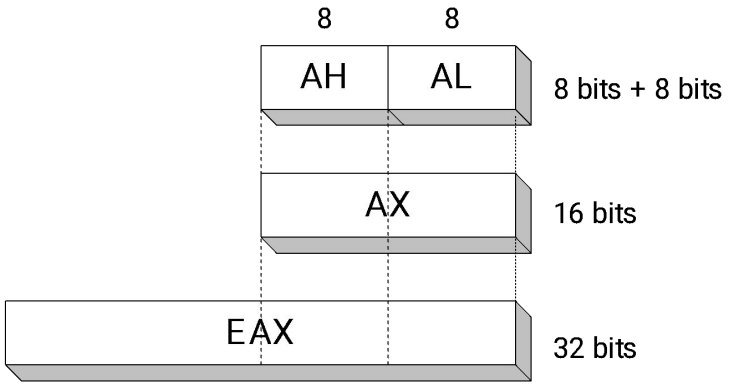


General-Purpose Registers

- Used primarily for arithmetic and data movement
  - `mov eax, 10` move constant 10 into register eax
- Specialized uses of Registers
  - EAX – Accumulator register
    - Automatically used by multiplication and division instructions
  - ECX – Counter register
    - Automatically used by LOOP instructions
  - ESP – Stack Pointer register
    - Used by PUSH and POP instructions, points to top of stack
  - ESI and EDI – Source Index and Destination Index register
    - Used by string instructions
  - EBP – Base Pointer register
    - Used to reference parameters and local variables on the stack

Accessing Parts of Registers

- EAX, EBX, ECX, and EDX are 32-bit Extended registers
  - Programmers can access their 16-bit and 8-bit parts
  - Lower 16-bit of EAX is named AX
  - AX is further divided into
    - AL = lower 8 bits
    - AH = upper 8 bits
- ESI, EDI, EBP, ESP have only 16-bit names for lower half

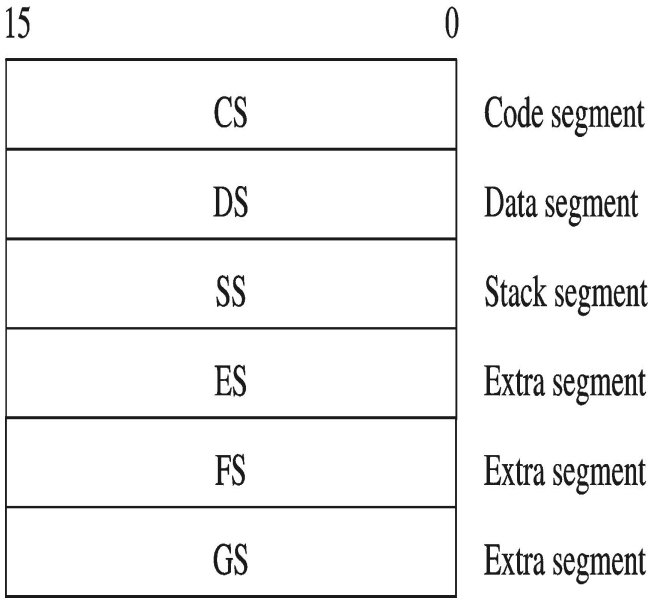


32-bit	16-bit	8-bit (high)	8-bit (low)
EAX	AX	AH	AL
EBX	BX	BH	BL
ECX	CX	CH	CL
EDX	DX	DH	DL

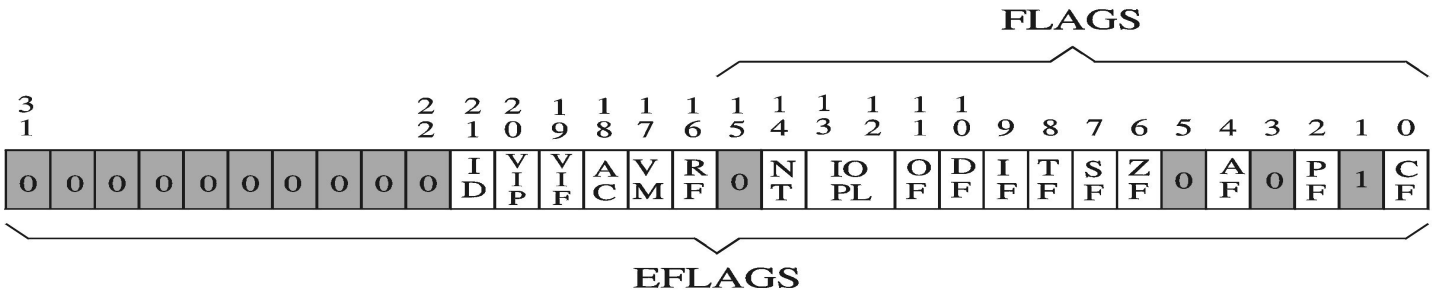
32-bit	16-bit
ESI	SI
EDI	DI
EBP	BP
ESP	SP

Special-Purpose & Segment Registers

- EIP = Extended Instruction Pointer
  - Contains address of next instruction to be executed
- EFLAGS = Extended Flags Register
  - Contains status and control flags
  - Each flag is a single binary bit
- Six 16-bit Segment Registers
  - Support segmented memory
  - Six segments accessible at a time
  - Segments contain distinct contents
    - Code
    - Data
    - Stack



EFLAGS Register



Status flags

CF = Carry flag
PF = Parity flag
AF = Auxiliary carry flag
ZF = Zero flag
SF = Sign flag
OF = Overflow flag

Control flags

DF = Direction flag

System flags

TF = Trap flag
IF = Interrupt flag
IOPL = I/O privilege level
NT = Nested task
RF = Resume flag
VM = Virtual 8086 mode
AC = Alignment check
VIF = Virtual interrupt flag
VIP = Virtual interrupt pending
ID = ID flag

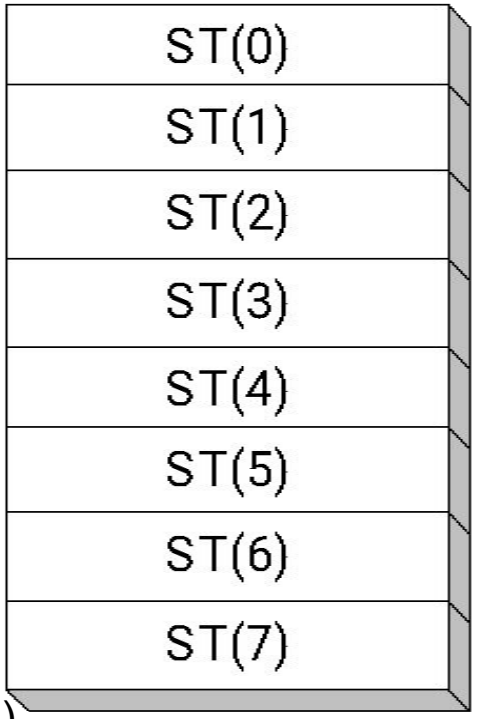
- Status Flags
  - Status of arithmetic and logical operations
- Control and System flags
  - Control the CPU operation
- Programs can set and clear individual bits in the EFLAGS register

Status Flags

- Carry Flag
  - Set when unsigned arithmetic result is out of range
- Overflow Flag
  - Set when signed arithmetic result is out of range
- Sign Flag
  - Copy of sign bit, set when result is negative
- Zero Flag
  - Set when result is zero
- Auxiliary Carry Flag
  - Set when there is a carry from bit 3 to bit 4
- Parity Flag
  - Set when parity is even
  - Least-significant byte in result contains even number of 1s

Floating-Point, MMX, XMM Registers

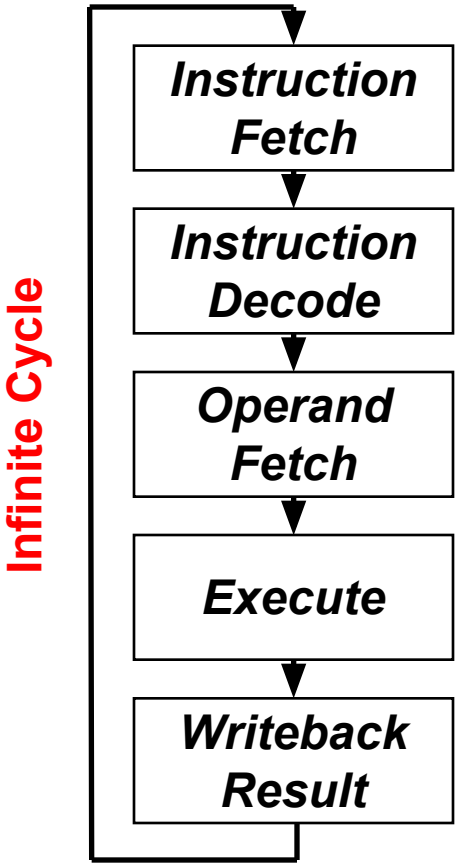
- Floating-point unit performs high speed FP operations
- Eight 80-bit floating-point data registers
  - ST(0), ST(1), . . . , ST(7)
  - Arranged as a stack
  - Used for floating-point arithmetic
- Eight 64-bit MMX registers
  - Used with MMX instructions
  - For advanced multimedia and communications applications
- Eight 128-bit XMM registers
  - Used with SSE instructions (Streaming SIMD Extension)
  - Used for calculations on data.



Next ...

- Basic Computer Organization
- Intel Microprocessors
- IA-32 Registers
- Instruction Execution Cycle
- IA-32 Memory Management

Instruction Execute Cycle



Obtain instruction from program storage

Determine required actions and instruction size

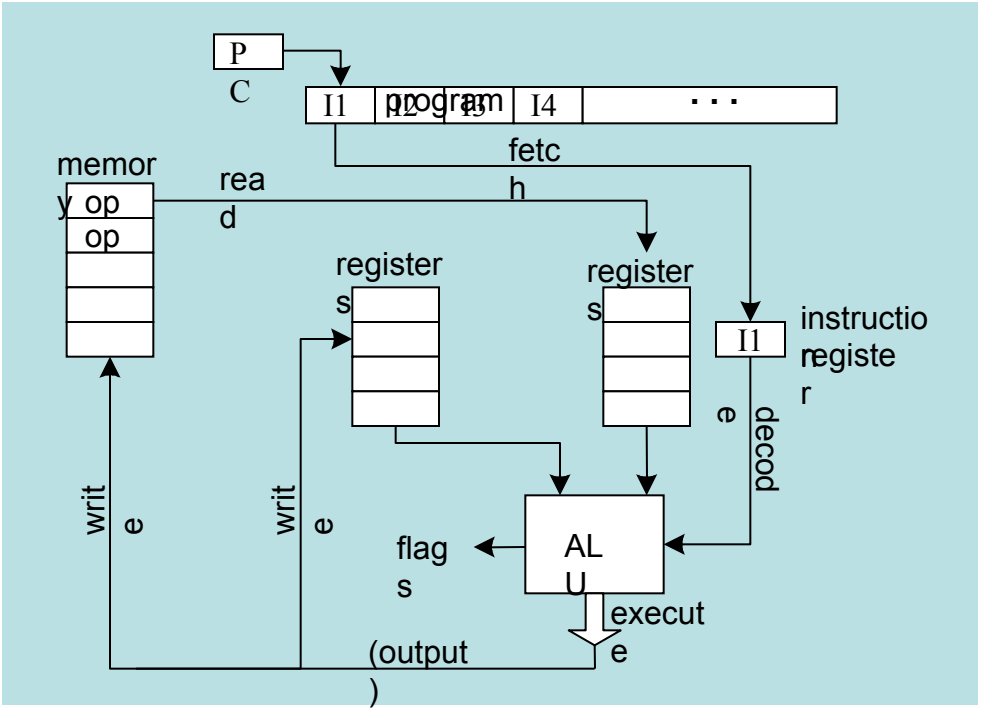
Locate and obtain operand data

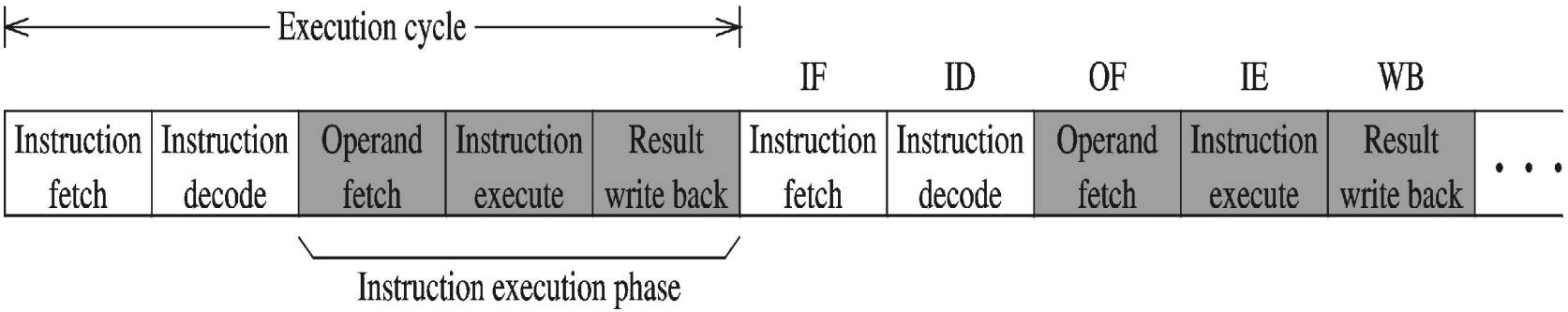
Compute result value and status

Deposit results in storage for later use

Instruction Execution Cycle – cont'd

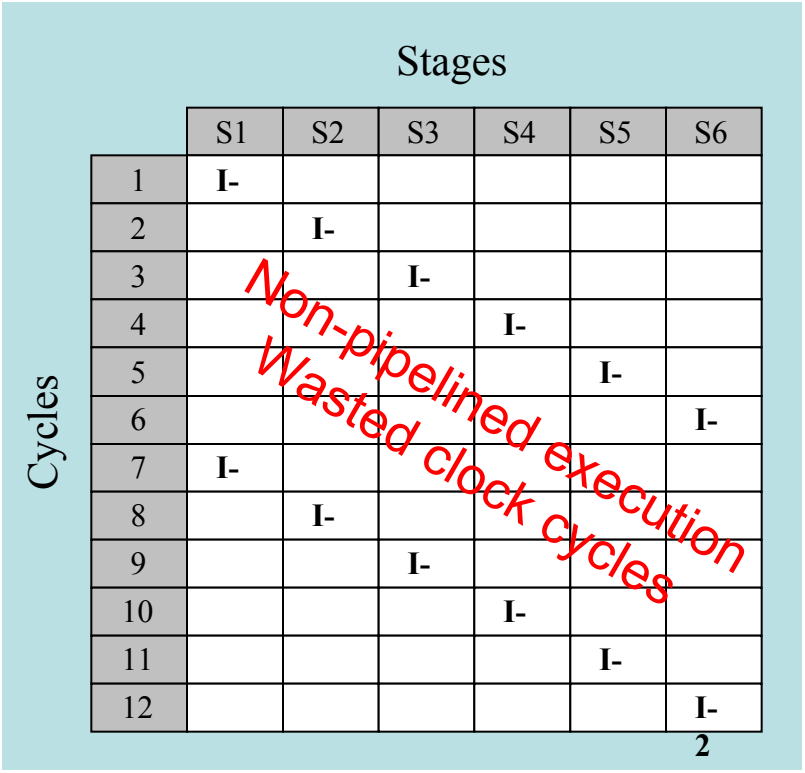
- Instruction Fetch
- Instruction Decode
- Operand Fetch
- Execute
- Result Writeback



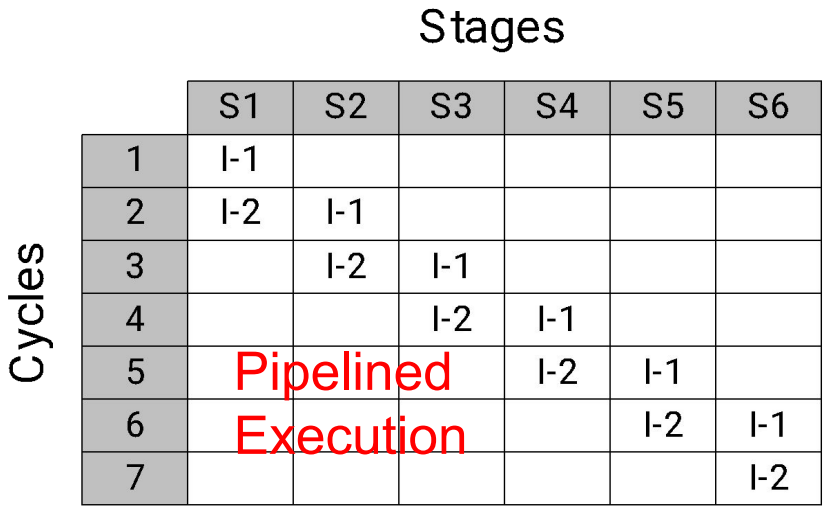


Pipelined Execution

- Instruction execution can be divided into stages
- Pipelining makes it possible to start an instruction before completing the execution of previous one



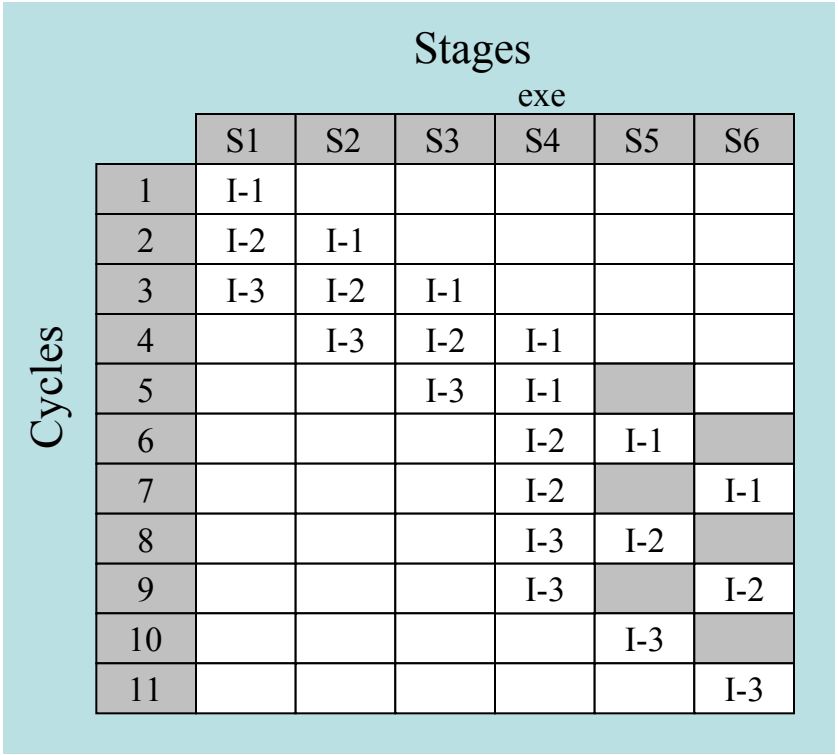
For *k* stages and *n* instructions, the number of required cycles is: $k + n - 1$



Wasted Cycles (pipelined)

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

  - Assume that stage S4 is the execute stage
  - Assume also that S4 requires 2 clock cycles to complete
  - As more instructions enter the pipeline, wasted cycles occur
  - For *k* stages, where one stage requires 2 cycles, *n* instructions require $k + 2n - 1$ cycles



Superscalar Architecture

- A superscalar processor has multiple execution pipelines
- The Pentium processor has two execution pipelines
  - Called U and V pipes
- In the following, stage S4 has 2 pipelines
  - Each pipeline still requires 2 cycles
  - Second pipeline eliminates wasted cycles
  - For *k* stages and *n* instructions, number of cycles = *k* + *n*

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

Next ...

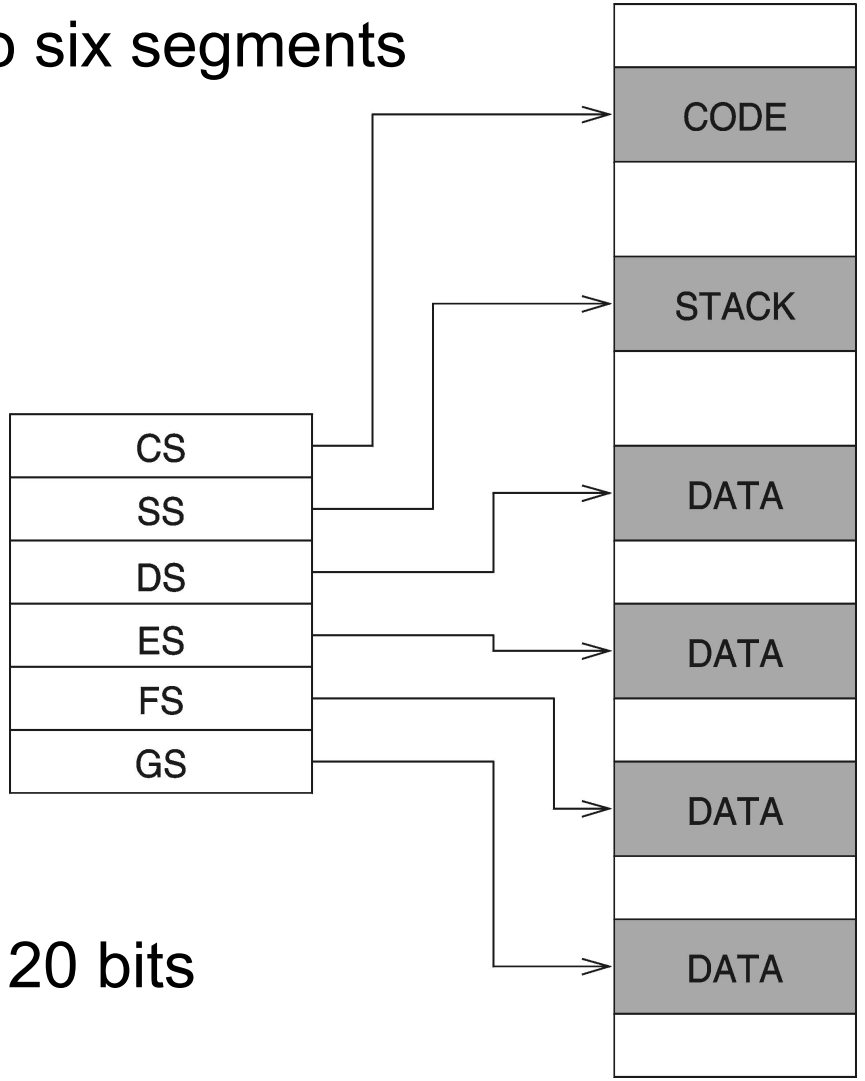
- Basic Computer Organization
- Intel Microprocessors
- IA-32 Registers
- Instruction Execution Cycle
- IA-32 Memory Management

Modes of Operation

- Real-Address mode (original mode provided by 8086)
  - Only 1 MB of memory can be addressed, from 0 to FFFFF (hex)
  - Programs can access any part of main memory
  - MS-DOS runs in real-address mode
- Protected mode (introduced with the 80386 processor)
  - Each program can address a maximum of 4 GB of memory
  - The operating system assigns memory to each running program
  - Programs are prevented from accessing each other's memory
  - Native mode used by Windows NT, 2000, XP, and Linux

Real Address Mode

- A program can access up to six segments at any time
  - Code segment
  - Stack segment
  - Data segment
  - Extra segments (up to 3)
- Each segment is 64 KB
- Logical address
  - Segment = 16 bits
  - Offset = 16 bits
- Linear (physical) address = 20 bits



Logical to Linear Address Translation

Linear address = Segment × 10 (hex) + Offset

Example:

segment = A1F0 (hex)

offset = 04C0 (hex)

logical address = A1F0:04C0 (hex)

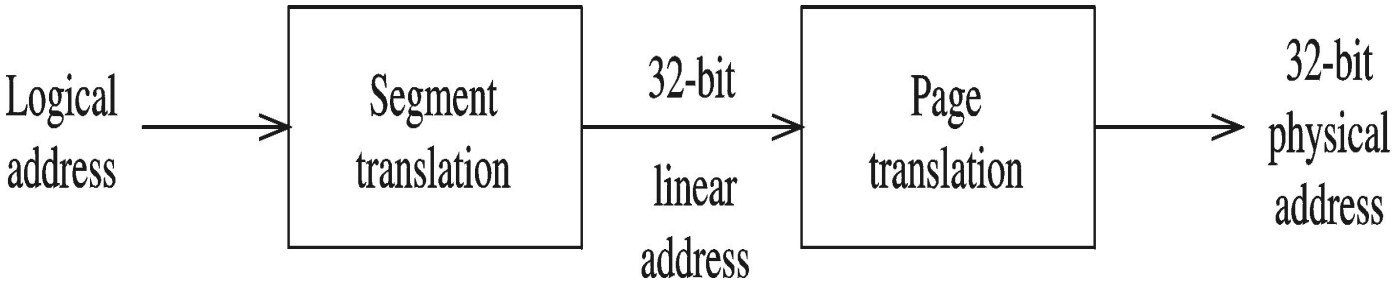
what is the linear address?

Solution:

```
  A1F00  (add 0 to segment in hex)
+ 04C0   (offset in hex)
  A23C0  (20-bit linear address in hex)
```

Protected Mode Architecture

- Logical address consists of
  - 16-bit segment selector (CS, SS, DS, ES, FS, GS)
  - 32-bit offset (EIP, ESP, EBP, ESI ,EDI, EAX, EBX, ECX, EDX)
- Segment unit translates logical address to linear address
  - Linear address is 32 bits (called also a virtual address)
- Paging unit translates linear address to physical address
  - Using a page directory and a page table

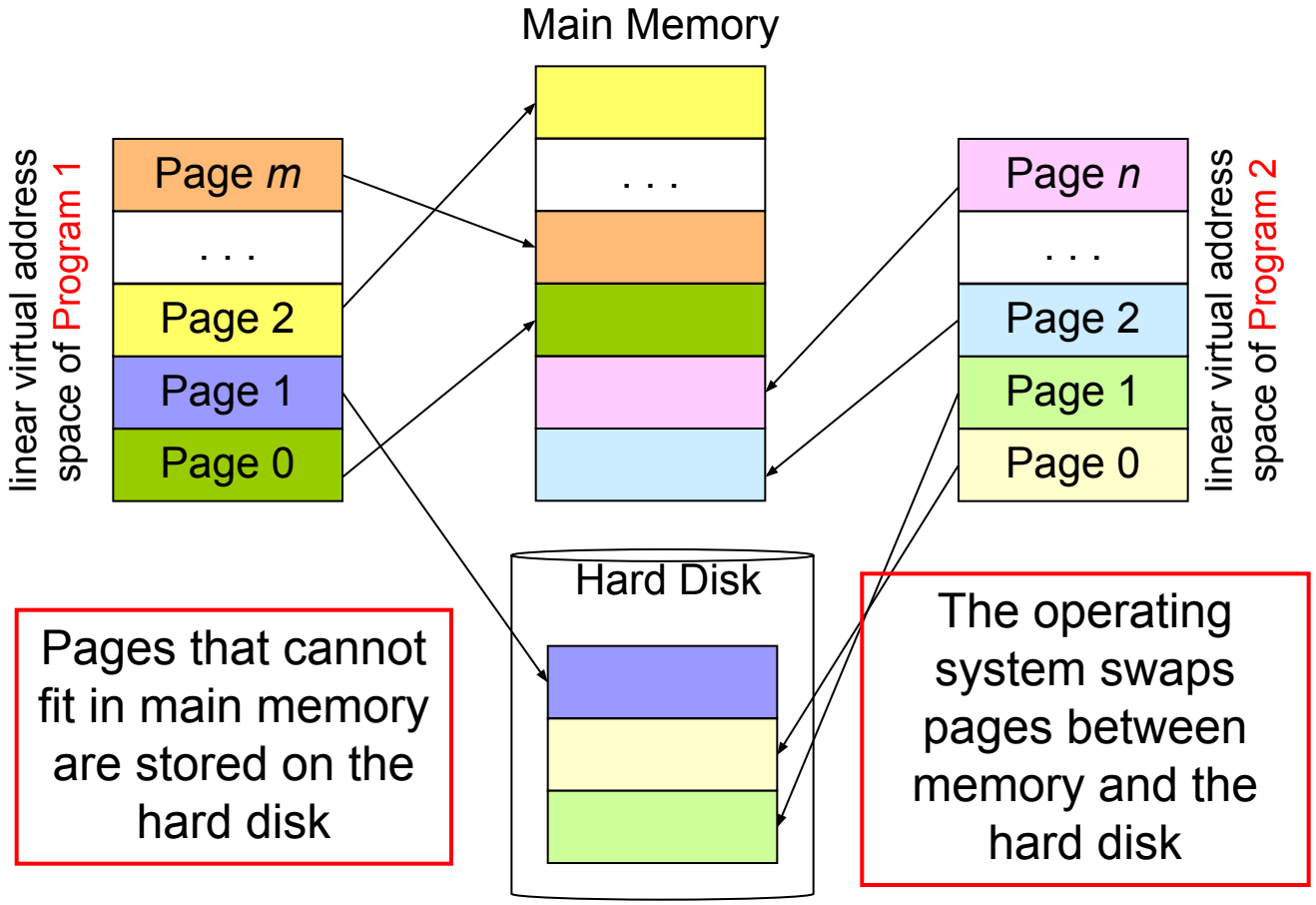


Paging

- Paging divides the linear address space into …
  - Fixed-sized blocks called pages, Intel IA-32 uses 4 KB pages
- Operating system allocates main memory for pages
  - Pages can be spread all over main memory
  - Pages in main memory can belong to different programs
  - If main memory is full then pages are stored on the hard disk
- OS has a Virtual Memory Manager (VMM)
  - Uses page tables to map the pages of each running program
  - Manages the loading and unloading of pages
- As a program is running, CPU does address translation
- Page fault: issued by CPU when page is not in memory

Paging – cont'd

The operating system uses page tables to map the pages in the linear virtual address space onto main memory



Each running program has its own page table

Pages that cannot fit in main memory are stored on the hard disk

The operating system swaps pages between memory and the hard disk

As a program is running, the processor translates the linear virtual addresses onto real memory (called also physical) addresses