EECE416 :Microcomputer Fundamentals and Design ("Microcomputer & Microprocessor")

IA 32

Basic Architecture

Dr. Charles Kim

Department of Electrical and Computer Engineering

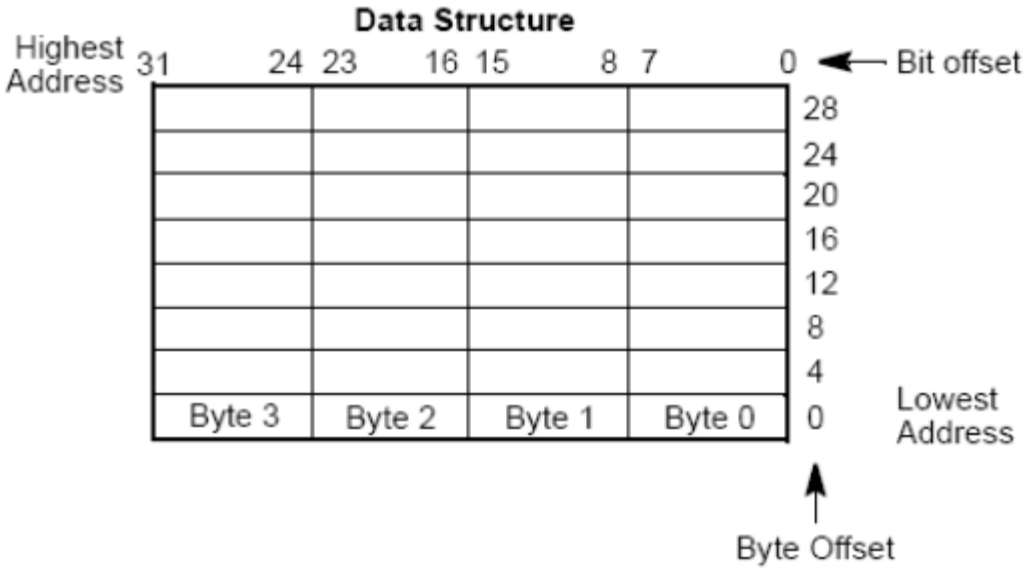
Howard University

Contents to be covered

- Overview and Notational Convention
- Introduction to the Intel Architecture
- Basic Execution Environment
- Procedure Calls, Interrupts, and Exceptions
- Data Types and Addressing Modes
- Instruction Set Summary
- Floating-Point Unit
- Programming with Intel MMX Technology
- Programming with the Streaming SIMD Extensions

Notational Conventions

- Bit and Byte Oder
  - Smaller address at the bottom of figure
  - Address increases toward top
  - Bit positions numbered from right to left
- Little-Endian Machine
  - the bytes of a word are numbered starting from the least significant byte



Conventions

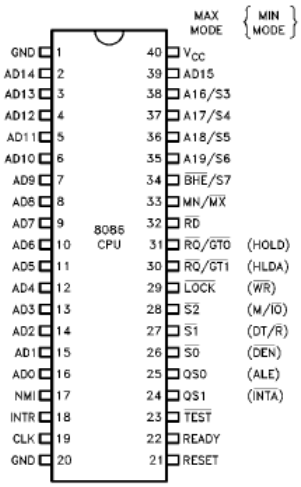
- IA Assembly Language
- Instruction Format
  - Label: mnemonic argument1, argument2, argument3
  - Label: Identifier (followed by a colon)
  - Mnemonic: a reserved name for a class of instruction opcodes which have the same function
  - Operands (arguments): The operands argument1, argument2, and argument3 are optional. There may be from zero to three operands, depending on the opcode. When present, they take the form of either literals or identifiers for data items. Operand identifiers are either reserved names of registers or are assumed to be assigned to data items declared in another part of the program.
  - When two operands are present in an arithmetic or logical instruction, the right operand is the source and the left operand is the destination.
  - Example: LOADREG: MOV EAX, SUBTOTAL
  - label mnemonic dst src
- Binary and Hexadecimal Numbers
  - Base 2 (binary) numbers are represented by a string of 1s and 0s, sometimes followed by the character B (for example, 1010B). The “B” designation is only used in situations where confusion as to the type of number might arise.
  - Base 16 (hexadecimal) numbers are represented by a string of hexadecimal digits followed by the character H (for example, F82EH). A hexadecimal digit is a character from the following set: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F.
- Segmented Addressing
  - Byte addressing
  - Address space: Range of memory that can be addressed
  - Segmented Addressing: where a program may have many independent address spaces.
  - Example: Byte address within a segment
    - Segment-register:Byte-address
    - DS:FF79H (byte at address FF79H in the segment pointed by the DS register)

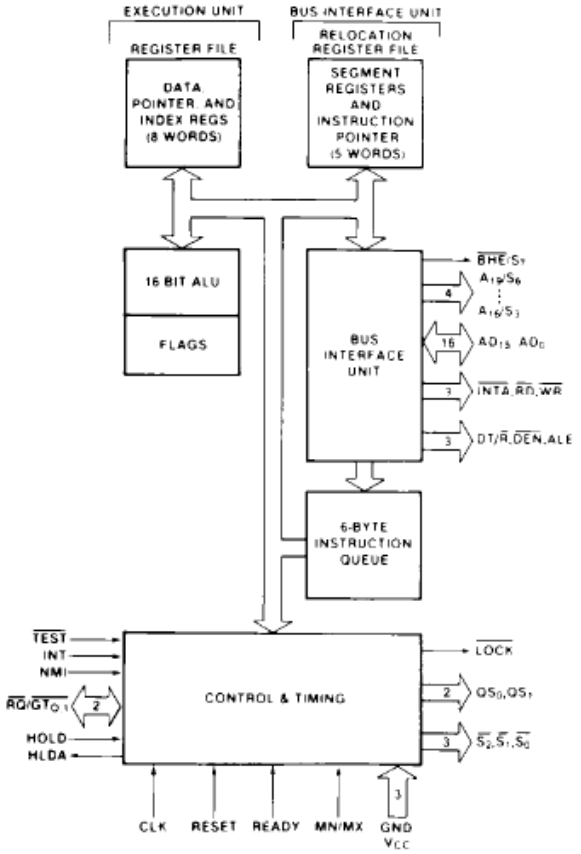
Conventions

- Exceptions
  - an event that typically occurs when an instruction causes an error.
  - Example: an attempt to divide by zero generates an exception.
  - Some exceptions, such as breakpoints, occur under other conditions.
  - Some types of exceptions may provide error codes. An error code reports additional information about the error.
  - #PF(fault code)
    - This example refers to a page-fault exception under conditions where an error code naming a type of fault is reported.
  - Under some conditions, exceptions which produce error codes may not be able to report an accurate code. In this case, the error code is zero, as shown below for a general-protection exception.
    - #GP(0)

Intel Architecture - History

- First IA Family member: 8086 (→ 8088). 1978
  - Cf. 4004 → 8080 → 8085
- 8086
  - 16-bit registers, external data bus
  - 20-bit addressing (→ 1MB address space)
  - Segmentation (by 16-bit) : 64KB
    - 4 Segmentation registers hold 4*64KB =256KB
    - Upto 256KB can be addressed without switching between Segments





IA History -continued

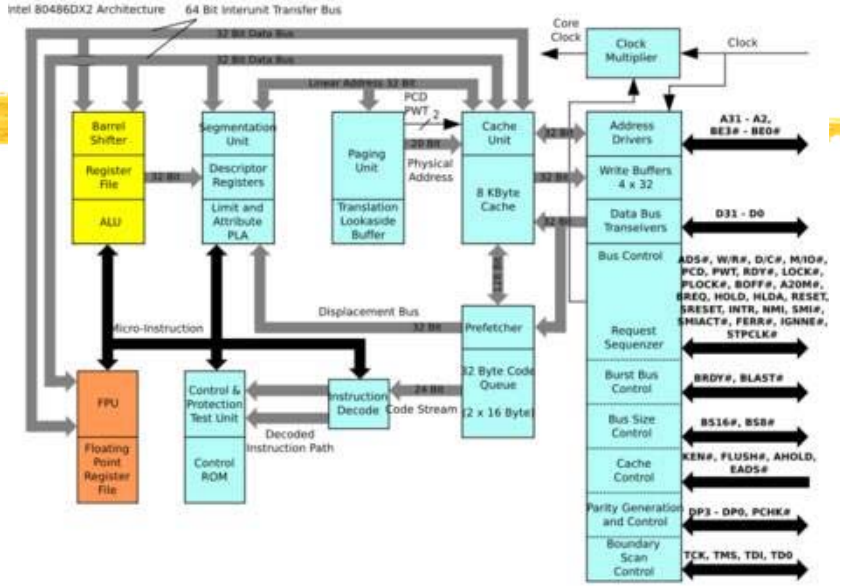
- 80286
  - Protected Mode
    - Segment register contents as selector or pointer → discriptor table
  - 24-bit base address → 16MB memory size
  - Application protection
- 386
  - 32-bit registers for operands and addressing(→4GB space)
  - Lower half of 32 bits is equivalent to 16 bits of earlier generations [Backward (upward) compatibility with 16-bit registers]
  - Some new instructions was added (like bit manipulation)
  - Max 4GB segmentation of physical space
  - New Parallel Processing Stages introduced: Bus Interface Unit, Code Prefetch Unit, Instruction Decode Unit, Execution Unit, Segment Unit (logical address → Linear address), Paging Unit (Linear address → physical address)

IA History - Continued

- 486
  - More parallel execution capability
    - Expansion of 386's Instruction Decode and Execution Units into 5 pipeline stages – 1 instruction per 1 CPU clock
  - L1 cache added
    - 8 KB
  - Integration of floating-point math unit on to the same chip
  - More pins added to support multiprocessors
  - Energy saving and management capability added to 486 SL Enhanced processors --- for battery operated notebook PC market
    - Stop clock and auto halt powerdown features – reduced clock rate to save power

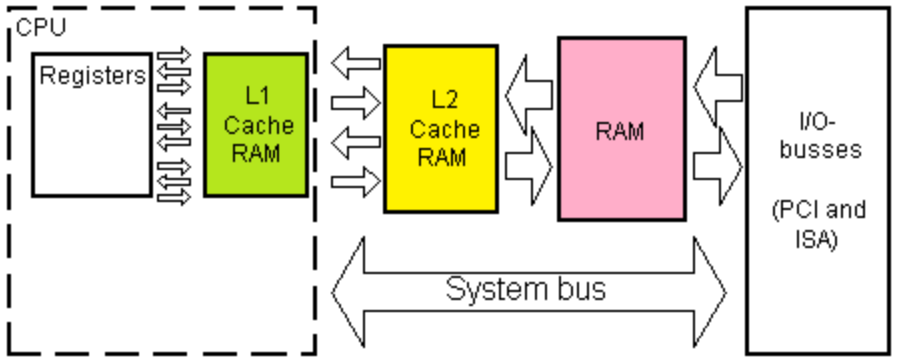
IA History - Continued

- 486
  - More parallel execution capability
    - Expansion of 386's Instruction Decode and Execution Units into 5 pipeline stages – 1 instruction per 1 CPU clock
  - L1 cache added
    - 8 KB
  - Integration of floating-point math unit on to the same chip
  - More pins added to support multiprocessors
  - Energy saving and management capability added to 486 SL Enhanced processors --- for battery operated notebook PC market
    - Stop clock and auto halt powerdown features – reduced clock rate to save power



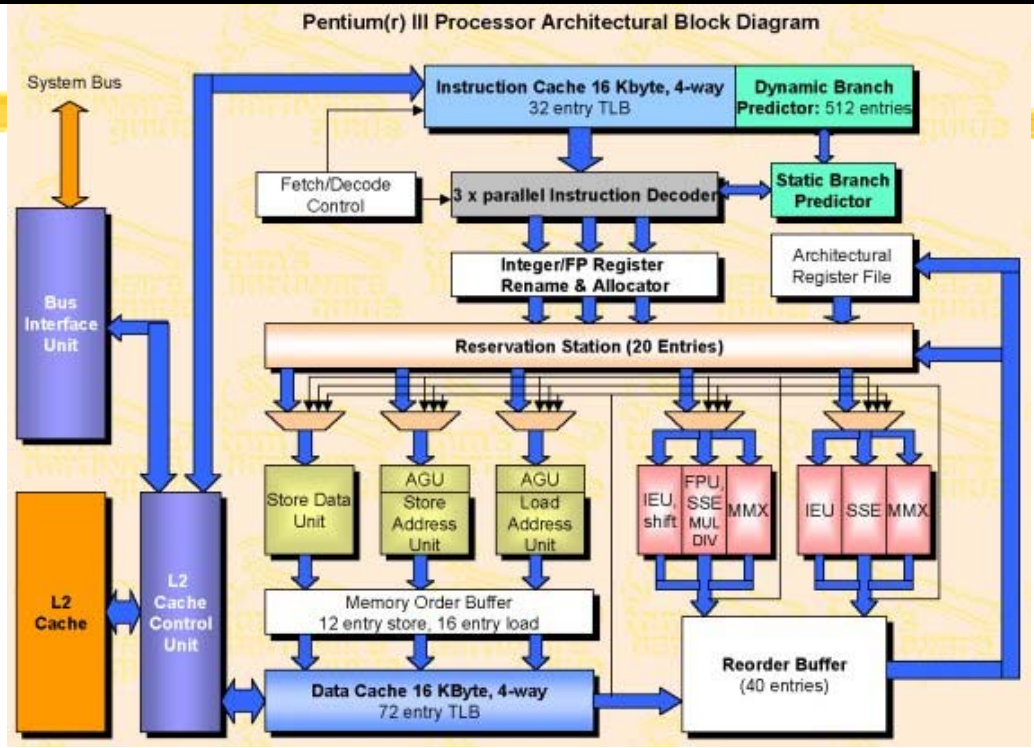
Insert – Cache Memory

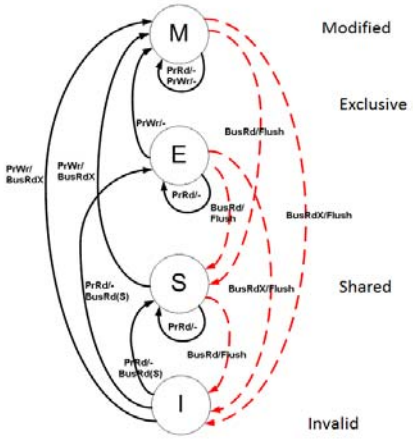
- CPU Cache Memory
  - a cache used by CPU to reduce the average time to access memory.
  - a smaller, faster memory which stores copies of the data from the most frequently used main memory locations. As long as most memory accesses are to cached memory locations, the average latency of memory accesses will be closer to the cache latency than to the latency of main memory.
  - Multi-level caches generally operate by checking the smallest Level 1 (L1) cache first; if it hits, the processor proceeds at high speed. If the smaller cache misses, the next larger cache (L2) is checked, and so on, before external memory is checked.
  - As the latency difference between main memory and the fastest cache has become larger, some processors have begun to utilize as many as three levels of on-chip cache. For example, in 2003, Itanium 2 began shipping with a 6 MB unified level 3 (L3) cache on-chip. The IBM Power 4 series has a 256 MB L3 cache off chip, shared among several processors.



IA History - continued

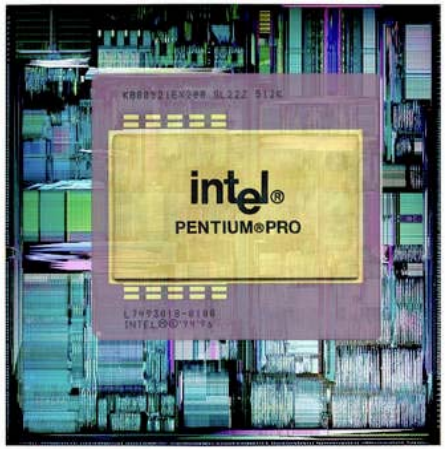
- Pentium
  - Second Execution Pipeline –2 pipelines (u and v) → 2 instructions per clock
- L1 Cache
  - 8KB for code
  - 8KB for data –
    MESI protocol for more efficient write-back mode
- Registers: 32 bits
- Internal Data paths: 128 and 256 bits

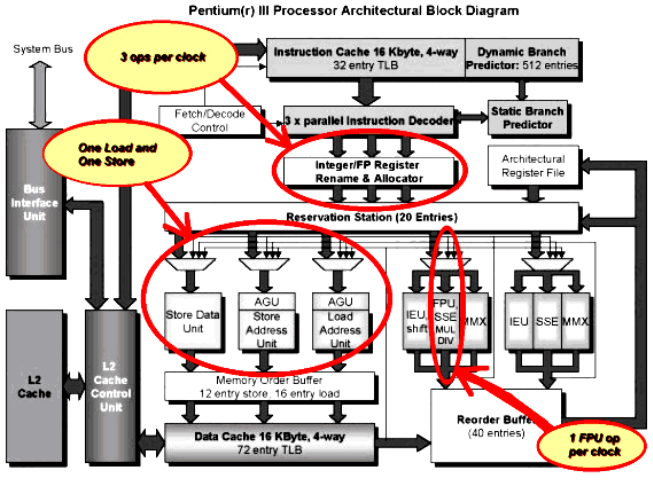




IA History - continued

- Pentium Pro
  - "Dynamic Execution" – 3 instructions per CPU clock
  - 8-KB L1 cache
  - 256 KB L2 cache
  - 36-bit address bus → 64 GB Physical Address





The Intel Pentium !!!

IA History - continued

- Pentium II
  - MMX instructions added
    - MMX is a single instruction, multiple data (SIMD) instruction set designed by Intel, introduced in 1996
    - MMX is officially a meaningless initialism trademarked by Intel.
    - MMX defined eight registers, known as MM0 through MM7 (henceforth referred to as MMn)
    - Each of the MMn registers holds 64 bits (the mantissa-part of a full 80-bit FPU register). The main usage of the MMX instruction set is based on the concept of packed data types, which means that instead of using the whole register for a single 64-bit integer, two 32-bit integers, four 16-bit integers, or eight 8-bit integers may be processed concurrently.
  - 16 KB L1 Instruction, 16 KB L1 Data
  - 256 (512, 1000) KB L2
  - Power Management: AutoHALT, Stop-Grant, Sleep, Deep Sleep

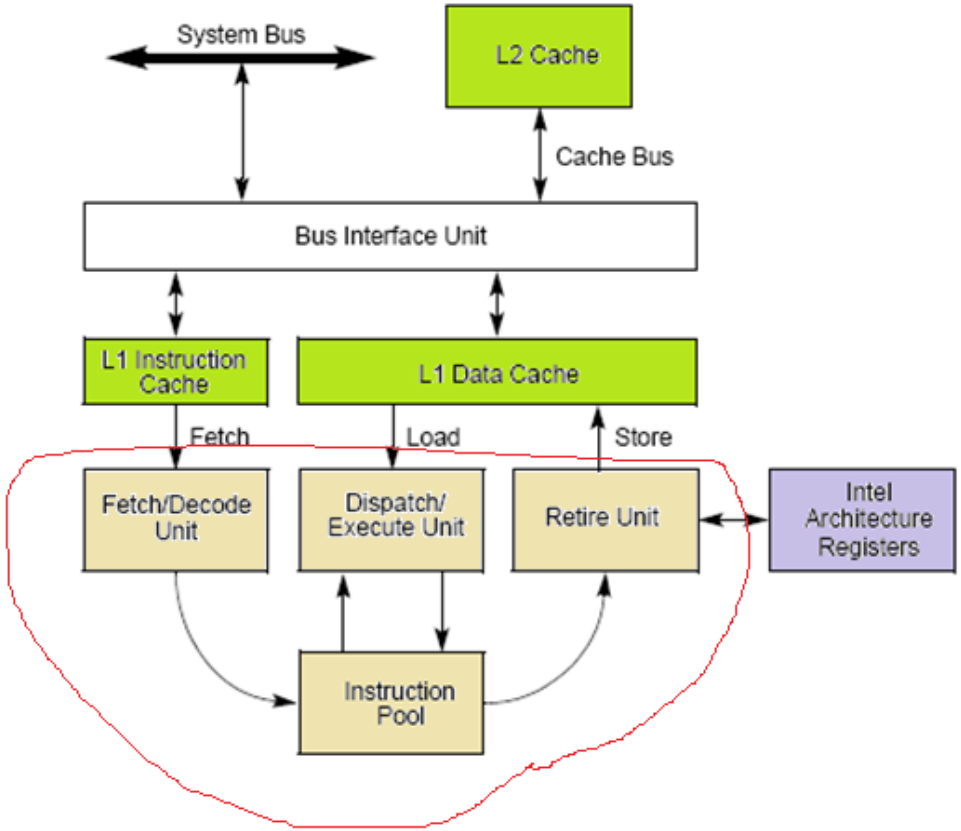
IA History - continued

- Pentium III
  - Pentium Pro + Pentium II
  - 70 new instructions
    - For New SIMD-floating-point unit
- Summary

Intel Processor	Date of Product Intro-duction	Perform -ance in MIPs[1]	Max. CPU Frequency at Intro-duction	No. of Transis -tors on the Die	Main CPU Register Size[2]	Extern. Data Bus Size[2]	Max. Extern. Addr. Space	Caches in CPU Pack-age[3]
8086	1978	0.8	8 MHz	29 K	16	16	1 MB	None
Intel 286	1982	2.7	12.5 MHz	134 K	16	16	16 MB	Note 3
Intel386™ DX	1985	6.0	20 MHz	275 K	32	32	4 GB	Note 3
Intel486™ DX	1989	20	25 MHz	1.2 M	32	32	4 GB	8KB L1
Pentium®	1993	100	60 MHz	3.1 M	32	64	4 GB	16KB L1
Pentium® Pro	1995	440	200 MHz	5.5 M	32	64	64 GB	16KB L1; 256KB or 512KB L2
Pentium II®	1997	466	266	7 M	32	64	64 GB	32KB L1; 256KB or 512KB L2
Pentium® III	1999	1000	500	8.2 M	32 GP 128 SIMD-FP	64	64 GB	32KB L1; 512KB L2

IA History - continued

- P6 Family Processor
  - 1995
  - Most recent processor in IA family
  - 3-way superscalar, pipelined architecture: 4 units



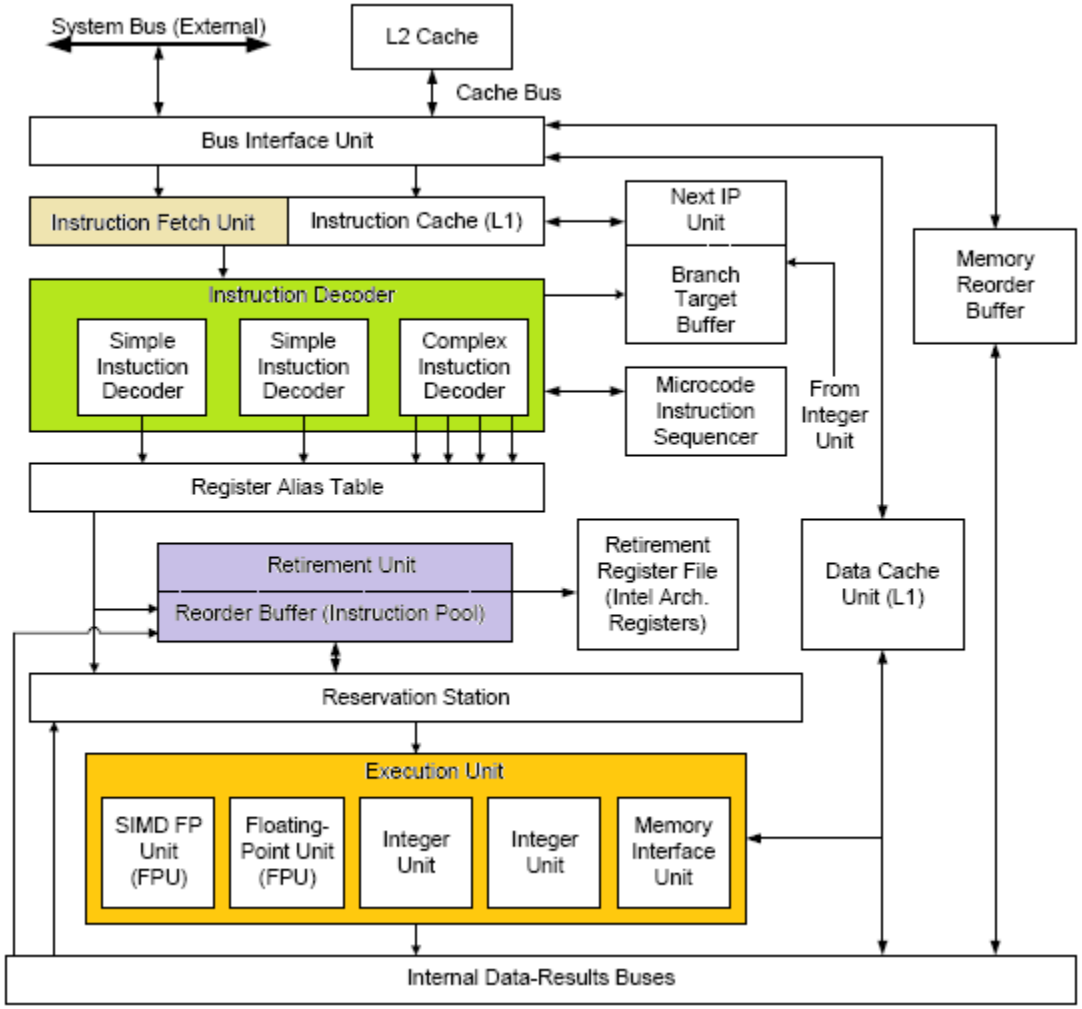
- 8 KB L1 Instruction cache
- 8 KB L1 Data cache
- 256 (512, 1000) KB SRAM L2

→ 64 bit cache bus

- Dynamic Execution (out-of-order execution mechanism)
  - Deep branch prediction
  - Dynamic data flow analysis
  - Speculative execution-ahead of program counter

IA History - continued

- P6 Micro-architecture
  - Instruction fetch/decoder unit
  - Instruction Pool (reorder buffer)
  - Dispatch/Execute Unit
  - Retirement Unit



EECE416 :Microcomputer Fundamentals and Design ("Microcomputer & Microprocessor")

IA 32

Basic Execution Environment

As seen by assembly-language programmers

Dr. Charles Kim

Department of Electrical and Computer Engineering

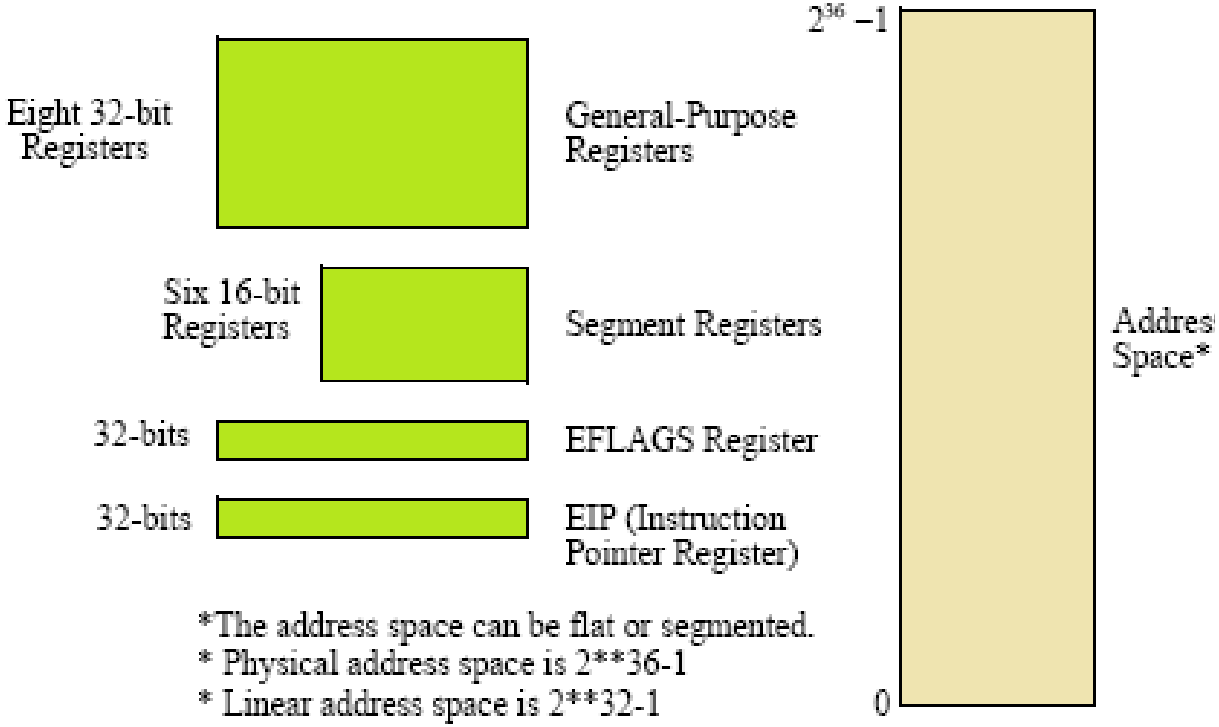
Howard University

Modes of Operation

- Operating mode determines which instructions and architectural features are accessible - 3 Operating modes
- Protected mode
  - Native State of Processor
  - All instructions and architectural features are available – highest performance and capability
  - Recommended mode
- Real-address mode
  - Programming environment of Intel 8086
  - Processor is in this mode following power-up or reset
- System management mode (SMM)
  - Power management and system security
  - Enters SMM by SMM interrupt (SMI) or APIC (Advanced Programmable Interrupt Controller)

Overview of Basic Execution

- Set of resources for Executing instructions and for Storing code, data, and state information
- Resources:
  - Address space: 36 address lines
  - 8 General data registers
  - 6 Segment registers
  - Status and control registers



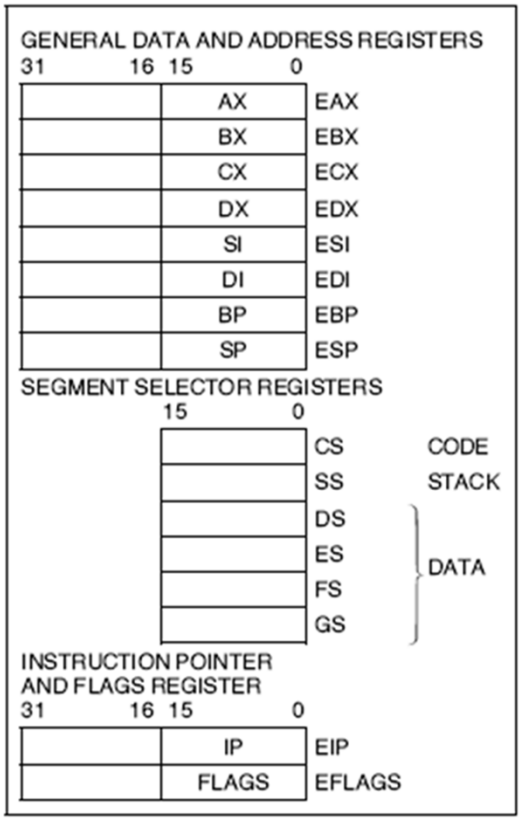


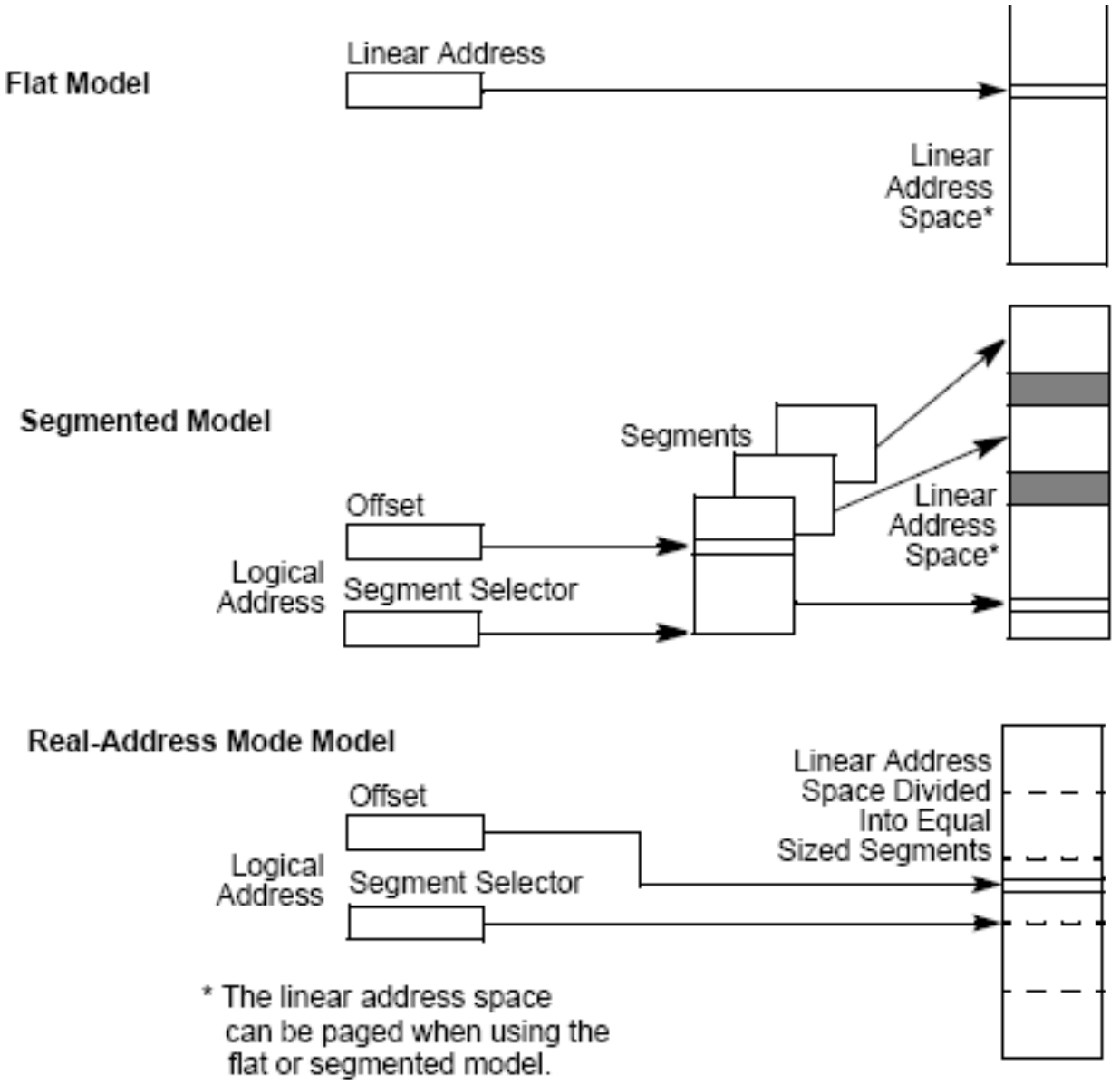
Figure 2-1. Intel386™ DX Base Architecture Registers

Memory Organization

- Physical Memory
  - The memory -- the processor addresses on its bus
  - Organized as a sequence of 8-bit bytes
  - Each byte is assigned a unique address, a physical address
  - Range: 36 address lines → 64 GB
- Flat memory model (a single continuous address space) → linear address space
  - Code, data, stack are all contained in this address space
  - Byte accessible
- Segmented memory model (memory grouped into independent address spaces, segments)
  - Code, data, stacks are contained in separate segments
  - Logical address (segment selector and an offset) to address
  - Up to 16K segments of different sizes (max 64 GB)
  - Why segmentation:
    - Increase reliability of programs and systems – avoid overwriting
- Real-Address Mode (Intel 8086 model)

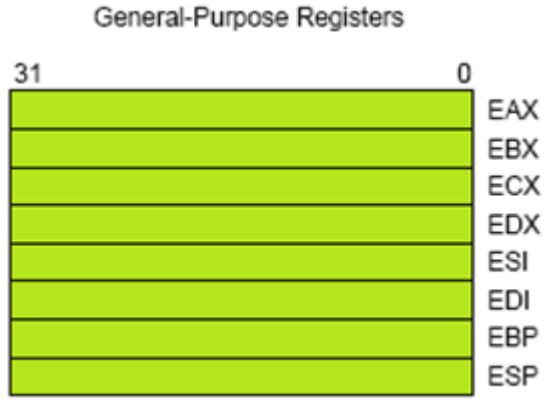
```
.386
.MODEL   FLAT

.STACK   4096

.DATA
```

Memory Management Model



General Purpose Data Registers

- Holding the following items (for all):
  - Operands for logical and arithmetic operations
  - Operands for address calculations
  - Memory pointers
- ESP (Stack pointer)holds the stack pointer (restricted use)
- ECX (Counter), ESI (Source pointer) , EDI (data pointer) for string instructions
- EBP (base pointer to data on the stack in DS segment)
- EDX (for I/O pointer)
- EAX (accumulator for operands and results data)
- EBX (Pointer to data in Segment)
- ESP points to the top item on the stack and the EBP points to the "previous" top of the stack before the function was called.

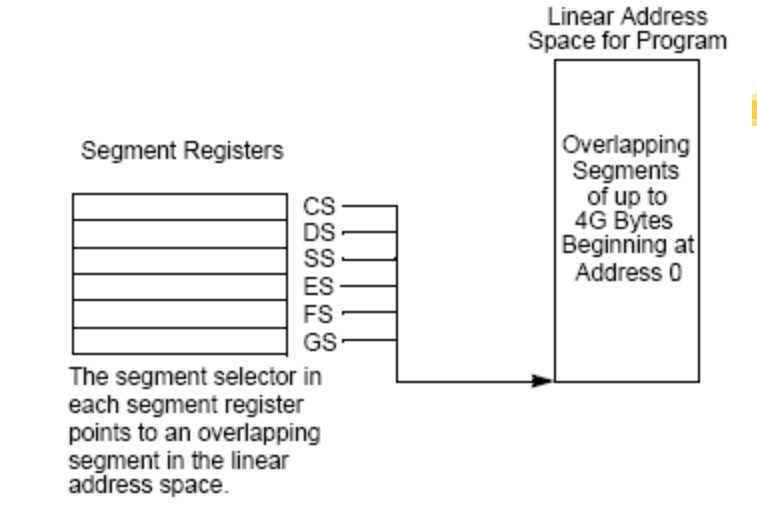


General-Purpose Registers

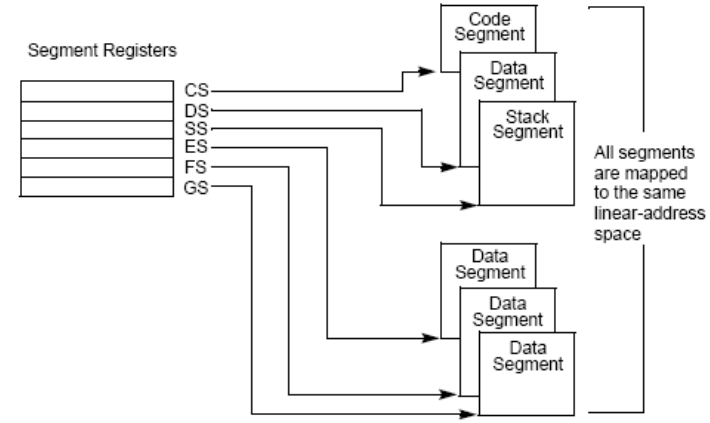
31 16	15 8	7 0	16-bit	32-bit
	AH	AL	AX	EAX
	BH	BL	BX	EBX
	CH	CL	CX	ECX
	DH	DL	DX	EDX
	BP			EBP
	SI			ESI
	DI			EDI
	SP			ESP

Segment registers

- Hold 16-bit segment selectors
- Segment selector: a special pointer that identifies a segment in memory
- Associated with 3 types of storage:
  - Code (instructions are stored): CS + EIP (offset)
  - Data : DS, ES, FS, and GS
  - Stack (Procedure Stack is stored): SS
- Segment selector ← by Assembler directive
- Flat (un-segmented) Memory Model Case:
  - Overlapped and starts at 0: Code Seg and Data Seg and Stack Seg
- Segmented Memory Model Case:
  - Loaded with different segments, pointing different segments
  - Program can access 6 different segments
  - To access a segment not pointed by the Segment registers? Load a segment selector to a segment register first.



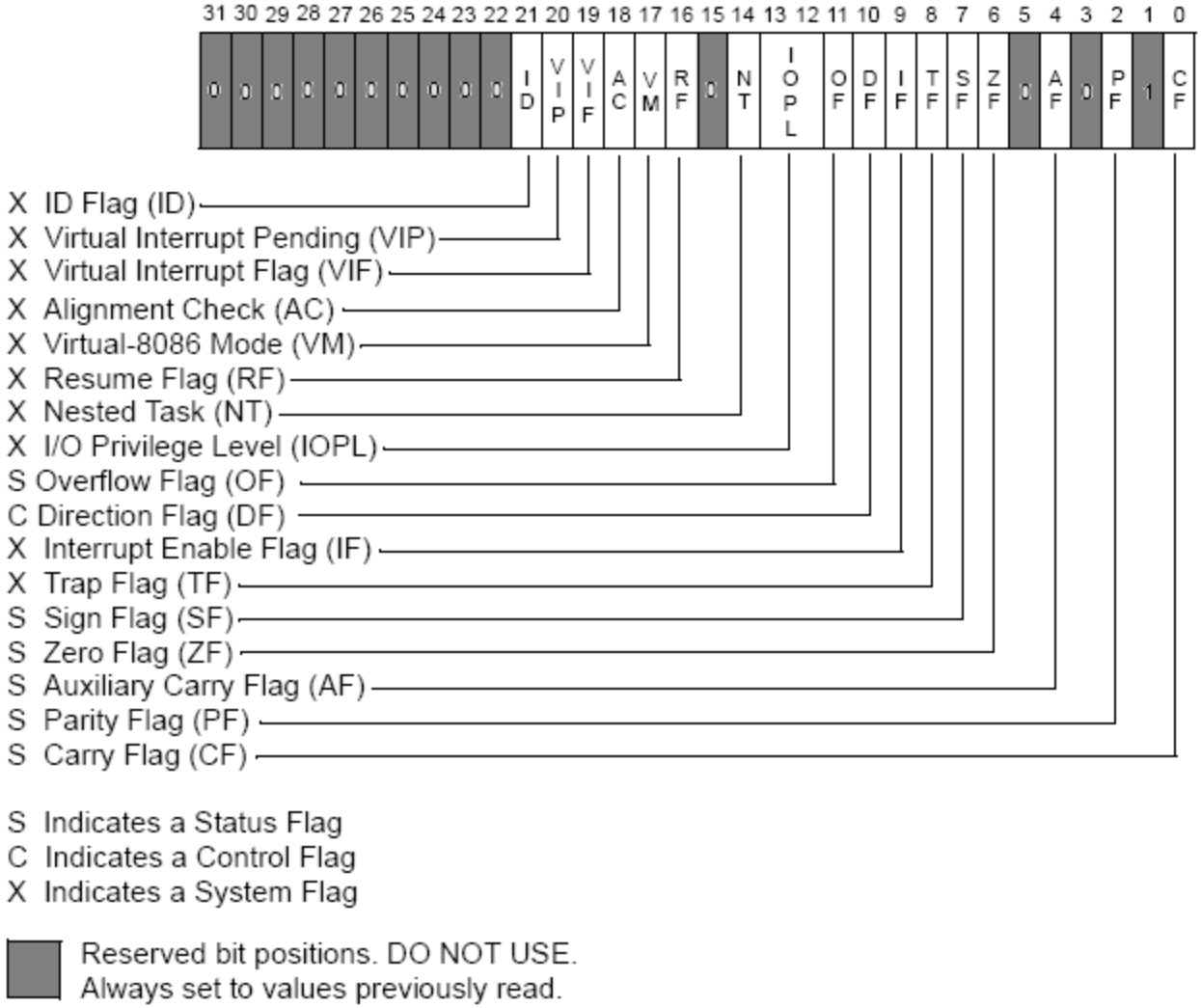
Use of Segment Registers for Flat Memory Model



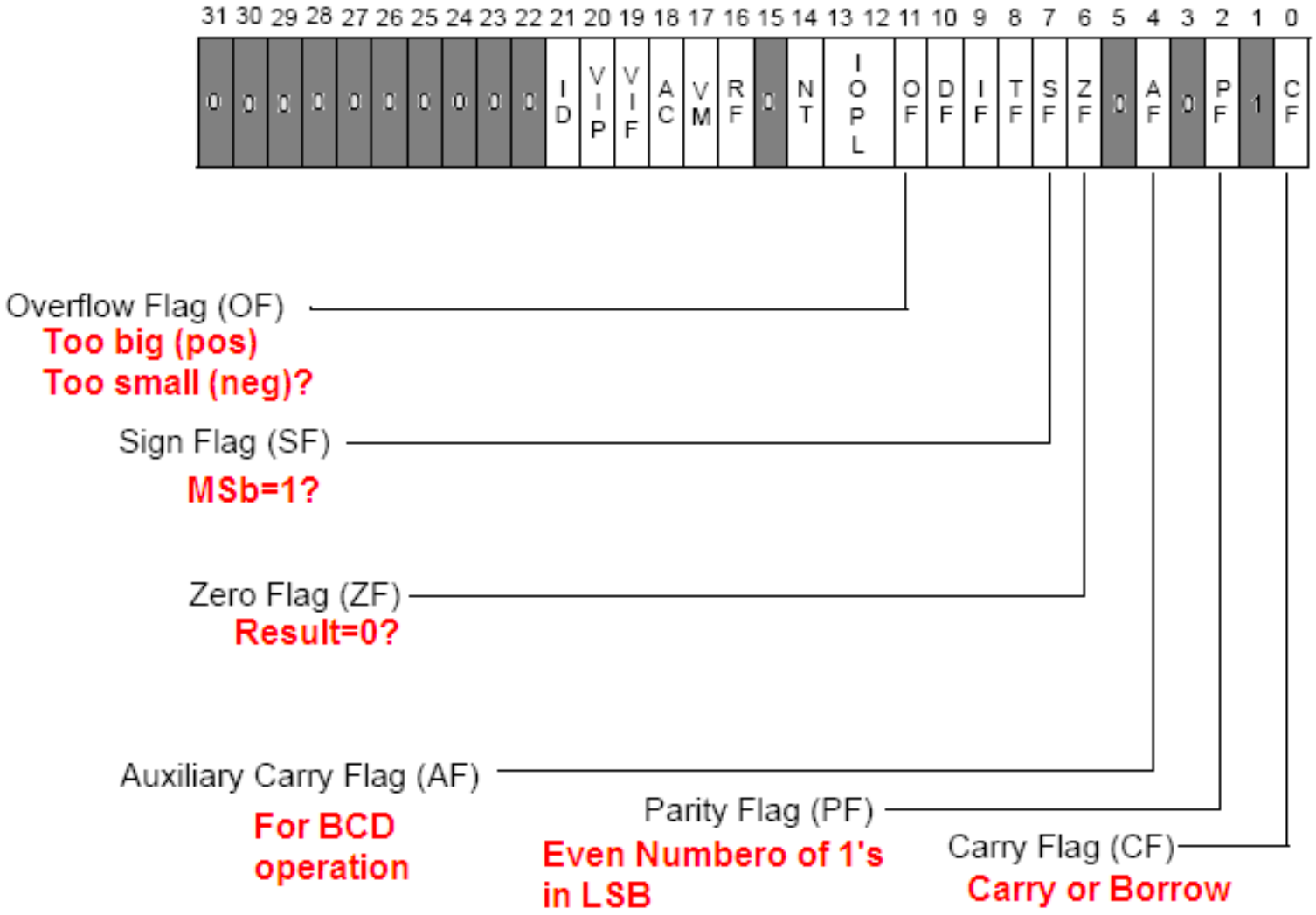
Use of Segment Registers in Segmented Memory Model

EFLAG Register

- 32-bit register
  - Initial state: 00000002H
  - Contains a group of **status flags**, a **control flag**, and a group of **system flags**



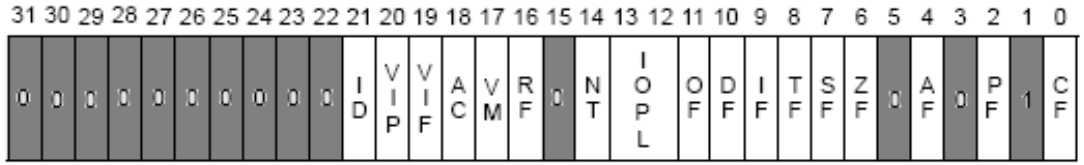
Status Flags



Control Flag (DF)

- DF (Direction Flag)
  - The direction flag controls the string instructions (MOVS, CMPS, SCAS, LODS, and STOS).
  - DF=1 → string instructions to auto-decrement (that is, to process strings from high addresses to low addresses).
  - DF=0 → string instructions to auto-increment (process strings from low addresses to high addresses).
  - STD → Set DF flag
  - CLD → Clear DF flag

System Flags



ID Flag (ID)
CPUID

Virtual Interrupt Pending (VIP)
Int. Pending?

Virtual Interrupt Flag (VIF)

Alignment Check (AC)
Memory reference aligned?

Virtual-8086 Mode (VM)
1: 8086 mode
0: Protected mode

Resume Flag (RF)
Response to debug exception

Nested Task (NT)
Task linked to another task?

I/O Privilege Level (IOPL)
Current Proviledge Level <= I/O Privilige level

Interrupt Enable Flag (IF)
Int. Respond/Inhibit

Trap Flag (TF)
Enable/Disable Single-Step Debug Mode