IA-32 Registers

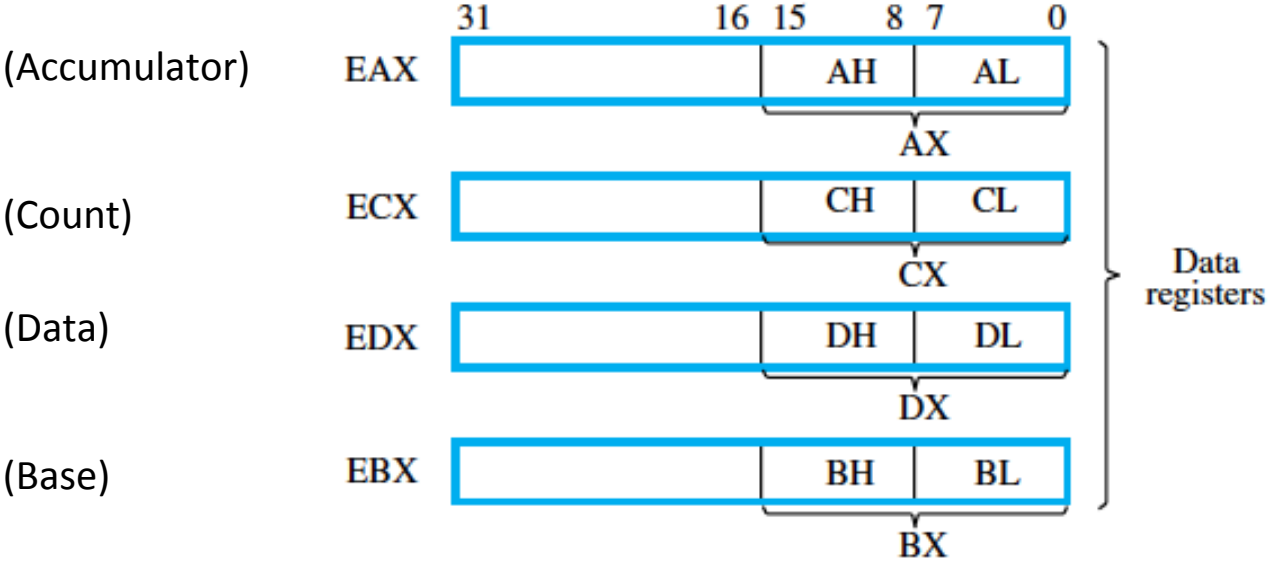
- Eight 32-bit General Purpose Registers
  - 4 Data Registers
  - 2 Pointer Registers
  - 2 Index Registers
- 32-bit Program counter
  - called Instruction pointer (EIP)
- 32-bit Status Register
    - Contains condition flags
- Register names, not numbers, are used

IA-32 Registers

- 8-bit Intel processors used 8-bit registers
  - Data registers were called A, B, C, and D.
- Later 16-bit processors used 16-bit registers
  - Data registers were called AX, BX, CX and DX.
- IA-32 Architecture uses 32-bit registers
  - Data registers are called EAX, EBX, ECX and EDX
    - E-prefix indicates 32-bit “extended” versions

IA-32 Registers

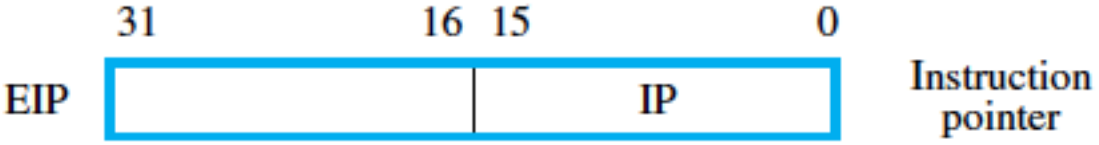
Data Registers



- EAX, ECX, EDX & EBX are 32-bit (extended)
- AX, CX, DX, and BX used for 16-bit items
- AH,AL, CH, CL, DH, DL, BH, BL are for 8-bit items
  - Maintains compatibility with earlier x86
  - Correctly runs 16-bit code on IA-32 processors

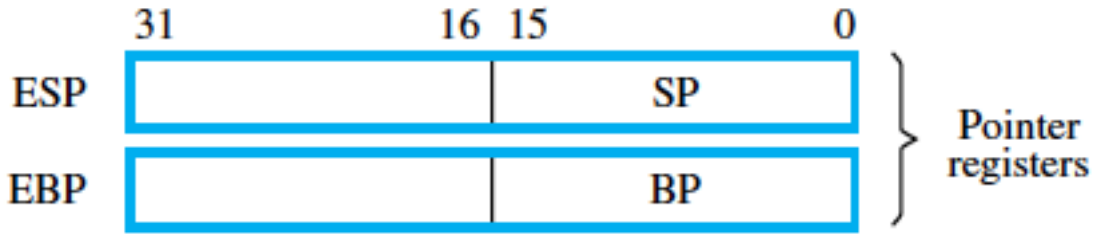
IA-32 Instruction Pointer

- Points to next instruction to execute
  - IP refers to the 16-bit instruction pointer
    - Limits code segment to 64 KB
  - EIP is the 32-bit extended instruction pointer
    - Code segment as large as 4 GB



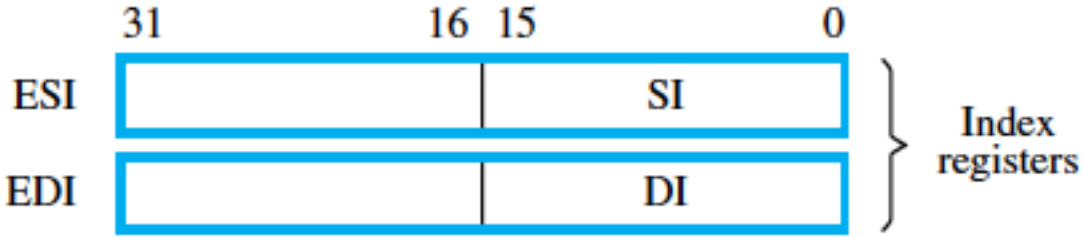
IA-32 Pointer Registers

- ESP & SP are 32-bit & 16-bit stack pointers
  - Points to stack in memory
- EBP & BP are 32-bit & 16-bit base registers
  - Used for call frames and data

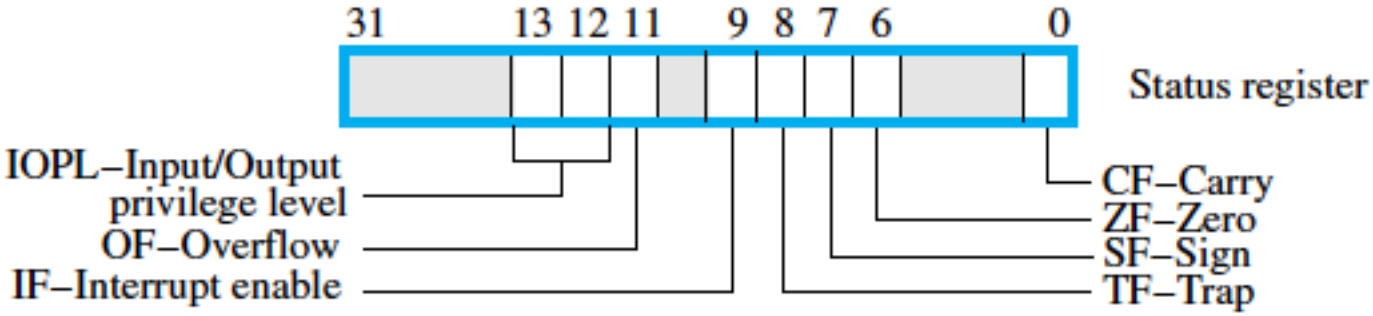


IA-32 Index Registers

- ESI & SI are 32-bit & 16-bit source index
- EDI & DI are 32-bit & 16-bit destination index
  - Used to reference memory operands
    - Array and string indices



IA-32 Status Register

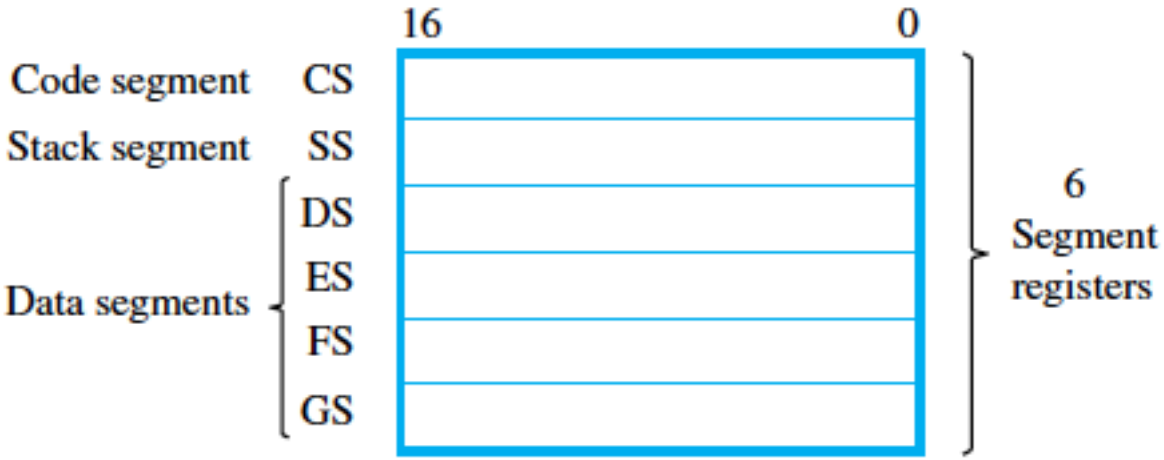


Also known as Flags register

- CF, ZF, SF, & OF are condition code flags
  - Reflect the result of arithmetic operations
- IOPL, IF, TF for I/O operations & interrupts
  - IF is interrupt enable/disable
  - TF is for single stepping through code
  - IOPL is set by the OS to control which operations are allowed

IA-32 Segment Registers

- Earlier x86 processors used memory segment
  - Segment starting address was held in a register
    - Segment registers are 16-bit
  - Segments were limited to 64 KB size

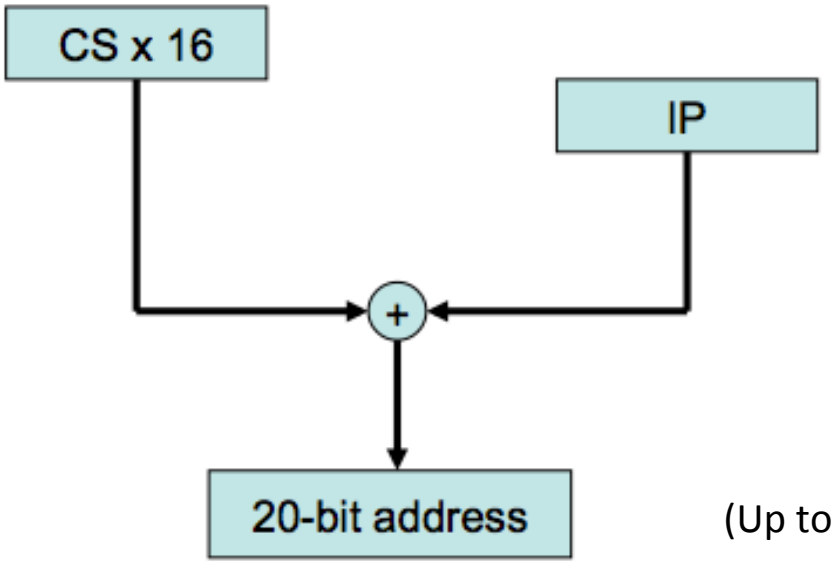


  - Memory address were specified as segment/offset
    - Segment_reg_contents:offset_value (xxxx:yyyy)

IA-32 Segment Registers

Address Generation Example:

The instruction Pointer register (IP) contains the offset within the current code segment from which the next instruction is fetched. On the 8086 this was a 16-bit register which is combined with the code segment register as shown below to produce the 20-bit memory address:



(Up to 1 MB address space)

Memory operand addresses are generated in a similar way, but one of the other segment registers is combined with a 16-bit offset.

IA-32 Segment Registers

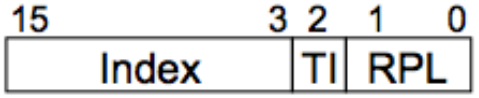
- Use of registers is based on Processor Mode
  - Real Mode
    - Segment registers are used the same as on x86 processors
      - Addresses are 20 bits
  - Protected Mode uses more complex segmentation
    - Segment registers contain index into descriptor table
      - 1 Local Descriptor table (LDT) for each task
      - Global Descriptor table (GDT) shared by all tasks
      - Interrupt Descriptor table (IDT) used by O.S.
      - Table entries contain 32-bit base address
      - Start address + 32-bit offset = 32-bit linear address
    - Overlapping segments provides a flat address space
    - Segments can be up to 4 GB in size

IA-32 Segment Registers

Protected Mode Segment Registers

Segment selector	Segment base address, size, access rights, etc.
Segment selector	Segment base address, size, access rights, etc.
Segment selector	Segment base address, size, access rights, etc.
Segment selector	Segment base address, size, access rights, etc.
Segment selector	Segment base address, size, access rights, etc.
Segment selector	Segment base address, size, access rights, etc.

In protected mode, every segment register has a "visible" part and an "invisible" The visible part is referred to as the segment selector. The invisible part is automatically loaded by the processor from a descriptor table.



Segment register Contents (protected mode)

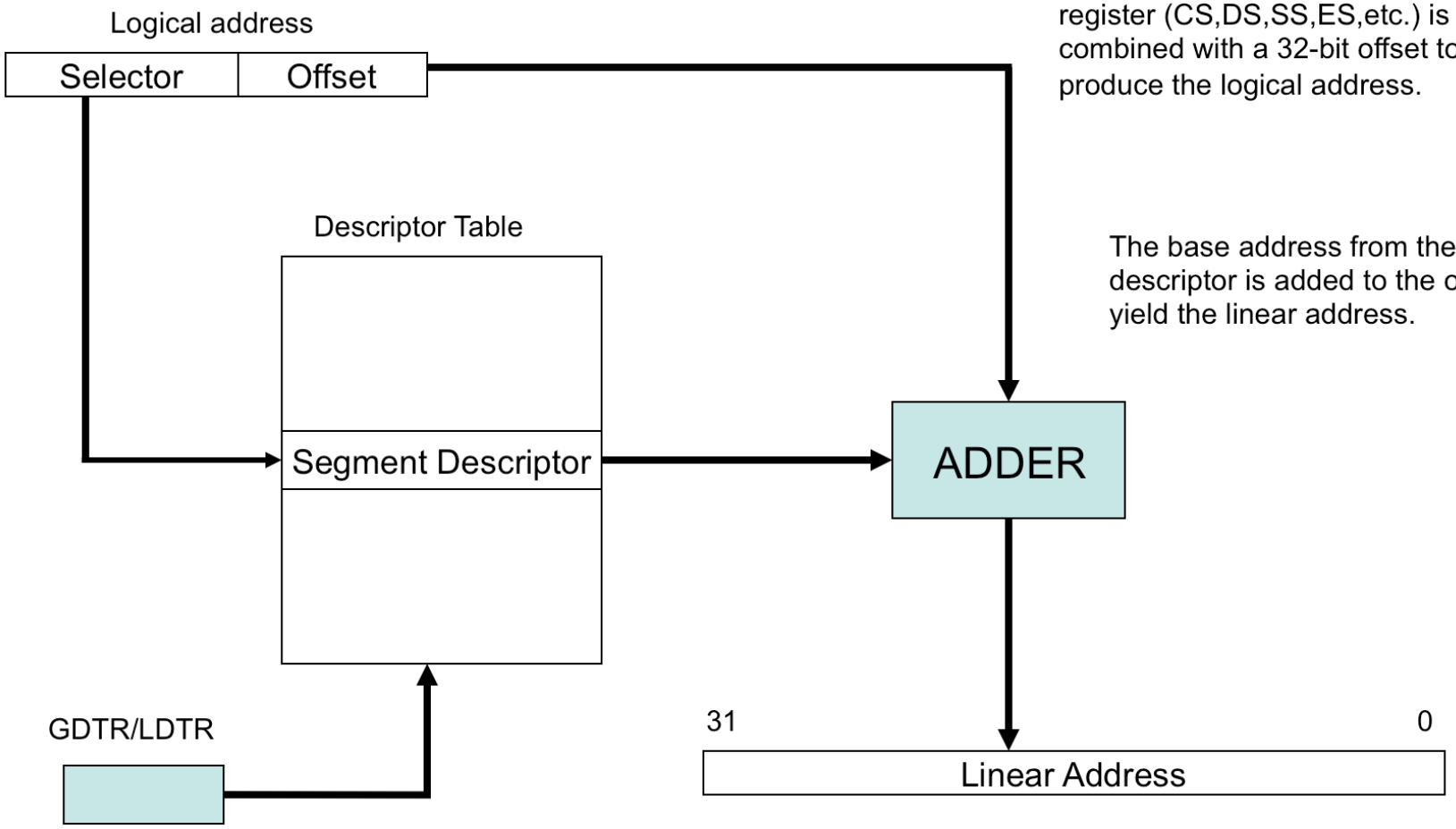
RPL – request privilege level

TI - table indicator 0=GDT, 1=LDT

Index – points to descriptor in table

IA-32 Segment Registers

Protected Mode Address Translation



A segment selector in the segment register (CS,DS,SS,ES,etc.) is combined with a 32-bit offset to produce the logical address.

The base address from the segment descriptor is added to the offset to yield the linear address.

Contains address of descriptor table

If paging is not used, then the linear address is the physical address.