CMSC 313 Lecture 04

- Homework 2
- IA-32 Basic Execution Environment
- IA-32 General Purpose Registers
- Moore's "Law"
- Evolution of the Pentium Chip
- "Hello World" in Linux Assembly Language
- Addressing Modes

Due: Tuesday, September 21, 2004

Instructions: For the following questions, *show all of your work*. It is not sufficient to provide the answers.

Exercise 1. Convert the following decimal numbers to hexadecimal representations of 16-bit two's complement numbers.

a. 1293

b. 31249

c. -24752

d. -4096

Exercise 2. Convert the following 16-bit two's complement numbers in hexadecimal representation to decimal.

a. $FFF5_{16}$

b. $7CD9_{16}$

c. $00BB_{16}$

d. 8000_{16}

Exercise 3. Write the following decimal numbers in IEEE-754 single precision format. Give your answers in binary.

a. 14.125

b. 3.14159

c. -58.375

d. -4096

Exercise 4. Write the decimal equivalents for these IEEE-754 single precision floating point numbers given in binary.

a. `0 10000001 01100000000000000000000`

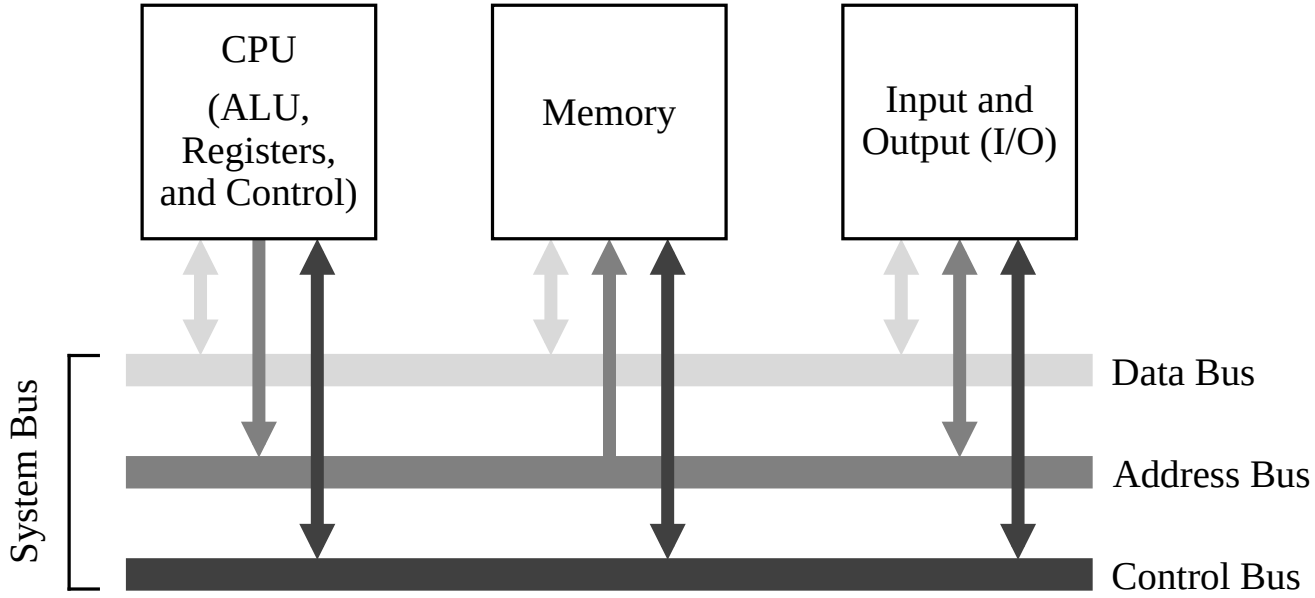
b. `1 10000001 00010000000000000000000`

c. `1 10000000 00000000000000000000000`

d. `0 00000001 01011000000000000000000`

The System Bus Model

- **A refinement of the von Neumann model, the system bus model has a CPU (ALU and control), memory, and an input/output unit.**
- **Communication among components is handled by a shared pathway called the *system bus*, which is made up of the data bus, the address bus, and the control bus. There is also a power bus, and some architectures may also have a separate I/O bus.**



The Fetch-Execute Cycle

- **The steps that the control unit carries out in executing a program are:**

  (1) Fetch the next instruction to be executed from memory.

  (2) Decode the opcode.

  (3) Read operand(s) from main memory, if any.

  (4) Execute the instruction and store results.

  (5) Go to step 1.

This is known as the *fetch-execute cycle*.

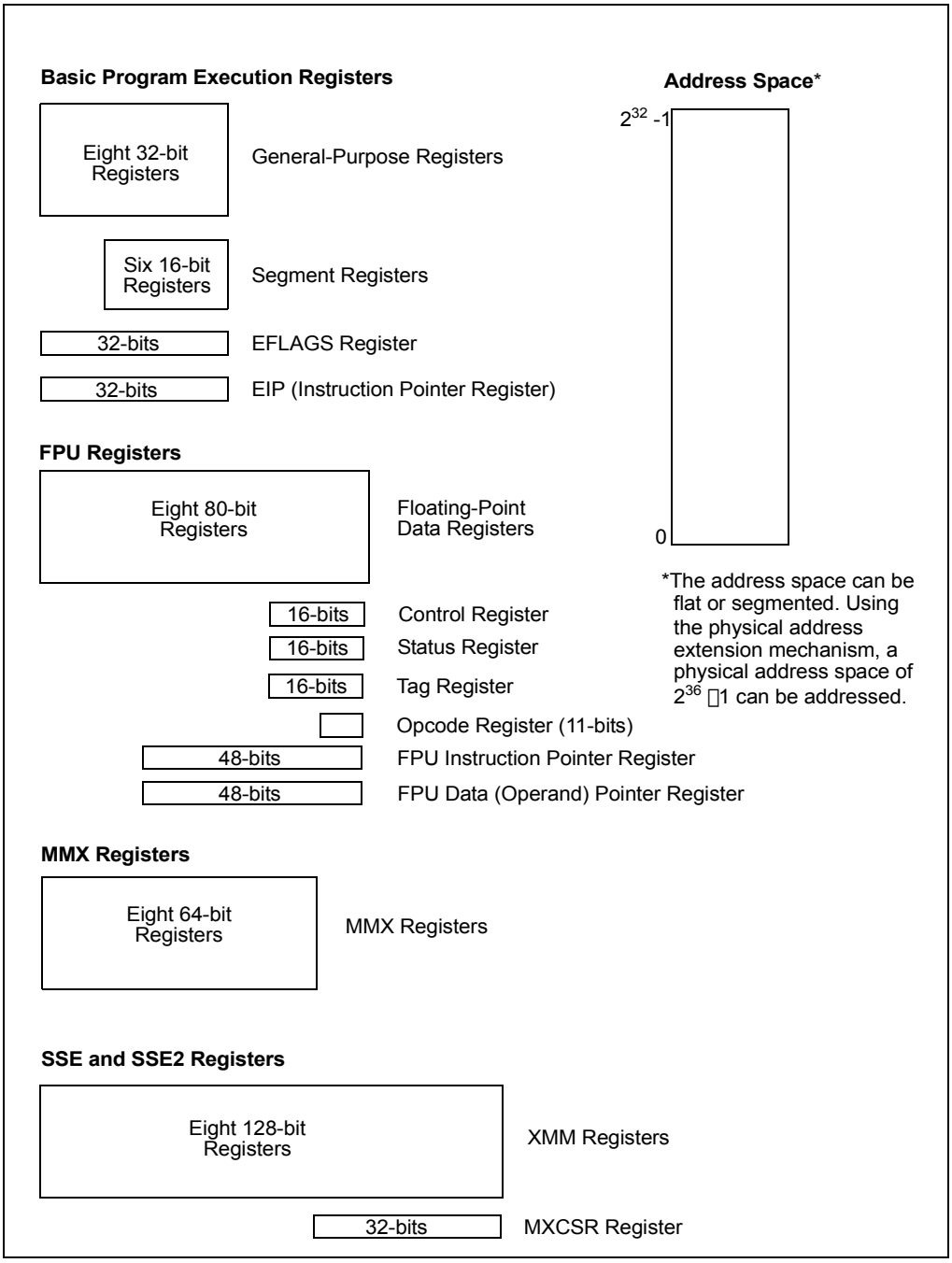


Figure 3-1. IA-32 Basic Execution Environment

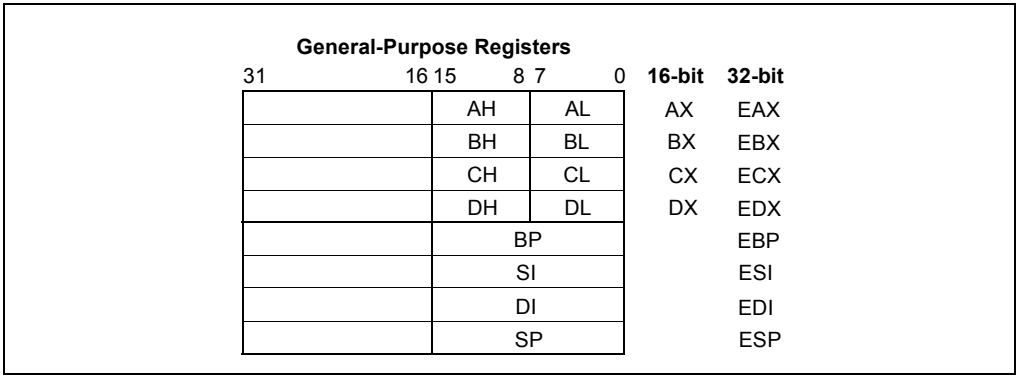


Figure 3-4. Alternate General-Purpose Register Names

- EAX—Accumulator for operands and results data.
- EBX—Pointer to data in the DS segment.
- ECX—Counter for string and loop operations.
- EDX—I/O pointer.
- ESI—Pointer to data in the segment pointed to by the DS register; source pointer for string operations.9
- EDI—Pointer to data (or destination) in the segment pointed to by the ES register; destination pointer for string operations.
- ESP—Stack pointer (in the SS segment).
- EBP—Pointer to data on the stack (in the SS segment).

Table 2-2. Key Features of Previous Generations of IA-32 Processors

Intel Processor	Date Intro-duced	Max. Clock Frequency at Intro-duction	Transis-tors per Die	Register Sizes[1]	Ext. Data Bus Size[2]	Max. Extern. Addr. Space	Caches
8086	1978	8 MHz	29 K	16 GP	16	1 MB	None
Intel 286	1982	12.5 MHz	134 K	16 GP	16	16 MB	Note 3
Intel386 DX Processor	1985	20 MHz	275 K	32 GP	32	4 GB	Note 3
Intel486 DX Processor	1989	25 MHz	1.2 M	32 GP 80 FPU	32	4 GB	L1: 8 KB
Pentium Processor	1993	60 MHz	3.1 M	32 GP 80 FPU	64	4 GB	L1:16 KB
Pentium Pro Processor	1995	200 MHz	5.5 M	32 GP 80 FPU	64	64 GB	L1: 16 KB L2: 256 KB or 512 KB
Pentium II Processor	1997	266 MHz	7 M	32 GP 80 FPU 64 MMX	64	64 GB	L1: 32 KB L2: 256 KB or 512 KB
Pentium III Processor	1999	500 MHz	8.2 M	32 GP 80 FPU 64 MMX 128 XMM	64	64 GB	L1: 32 KB L2: 512 KB
Pentium III and Pentium III Xeon Processors	1999	700 MHz	28 M	32 GP 80 FPU 64 MMX 128 XMM	64	64 GB	L1: 32KB L2: 256KB

NOTES:

1. The register size and external data bus size are given in bits. Note also that each 32-bit general-purpose (GP) registers can be addressed as an 8- or a 16-bit data registers in all of the processors.
2. Internal data paths are 2 to 4 times wider than the external data bus for each processor.

Table 2-1. Key Features of Most Recent IA-32 Processors

Intel Processor	Date Intro-duced	Microarhi-tecture	Clock Frequency at Intro-duction	Tran-sistors Per Die	Register Sizes[1]	System Bus Band-width	Max. Extern. Addr. Space	On-Die Caches[2]
Pentium 4 Processor	2000	Intel NetBurst Microarchi-tecture	1.50 GHz	42 M	GP: 32 FPU: 80 MMX: 64 XMM: 128	3.2 GB/s	64 GB	12K µop Execution Trace Cache; 8KB L1; 256-KB L2
Intel Xeon Processor	2001	Intel NetBurst Microarchi-tecture	1.70 GHz	42 M	GP: 32 FPU: 80 MMX: 64 XMM: 128	3.2 GB/s	64 GB	12K µop Trace Cache; 8-KB L1; 256-KB L2
Intel Xeon Processor	2002	Intel NetBurst Microarchi-tecture; Hyper-Threading Technology	2.20 GHz	55 M	GP: 32 FPU: 80 MMX: 64 XMM: 128	3.2 GB/s	64 GB	12K µop Trace Cache; 8-KB L1; 512-KB L2
Intel Xeon Processor MP	2002	Intel NetBurst Microarchi-tecture; Hyper-Threading Technology	1.60 GHz	108 M	GP: 32 FPU: 80 MMX: 64 XMM: 128	3.2 GB/s	64 GB	12K µop Trace Cache; 8-KB L1; 256-KB L2; 1-MB L3
Intel Pentium 4 Processor with Hyper-Threading Technology	2002	Intel NetBurst Microarchi-tecture; Hyper-Threading Technology	3.06 GHz	55 M	GP: 32 FPU: 80 MMX: 64 XMM: 128	4.2 GB/s	64 GB	12K µop Execution Trace Cache; 8-KB L1; 512-KB L2
Intel Pentium M Processor	2003	Intel Pentium M Processor	1.60 GHz	77 M	GP: 32 FPU: 80 MMX: 64 XMM: 128	3.2 GB/s	4 GB	L1: 64 KB L2: 1 MB

Table 2-1. Key Features of Most Recent IA-32 Processors (Contd.)

Intel Processor	Date Intro-duced	Microarhi-tecture	Clock Frequency at Intro-duction	Tran-sistors Per Die	Register Sizes[1]	System Bus Band-width	Max. Extern. Addr. Space	On-Die Caches[2]
Intel Pentium 4 Processor Supporting Hyper-Threading Technology at 90 nm process	2004	Intel NetBurst Microarchi-tecture; Hyper-Threading Technology	3.40 GHz	125 M	GP: 32 FPU: 80 MMX: 64 XMM: 128	6.4 GB/s	64 GB	12K µop Execution Trace Cache; 16 KB L1; 1 MB L2
Intel Pentium M Processor 755[3]	2004	Intel Pentium M Processor	2.00 GHz	140 M	GP: 32 FPU: 80 MMX: 64 XMM: 128	3.2 GB/s	4 GB	L1: 64 KB L2: 2MB

NOTES

1. The register size and external data bus size are given in bits.
2. First level cache is denoted using the abbreviation L1, 2nd level cache is denoted as L2. The size of L1 includes the first-level data cache and the instruction cache where applicable, but does not include the trace cache.
3. Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See http://www.intel.com/products/processor_number for details.

Moore's "Law"

- In the mid-1960's, Intel Chairman of the Board Gordon Moore observed that "the number of transistors that would be incorporated on a silicon die would double every 18 months for the next several years."
- His prediction has continued to hold true.
- Perhaps a self-fulfilling prophecy?

"Hello World" in Linux Assembly

- Use your favorite UNIX editor (vi, emacs, pico, ...)
- Assemble using NASM on gl.umbc.edu

  ```
  nasm -f elf hello.asm
  ```

- NASM documentation is on-line.
- Need to "load" the object file

  ```
  ld hello.o
  ```

- Execute

  ```
  a.out
  ```

- CMSC 121 Introduction to UNIX

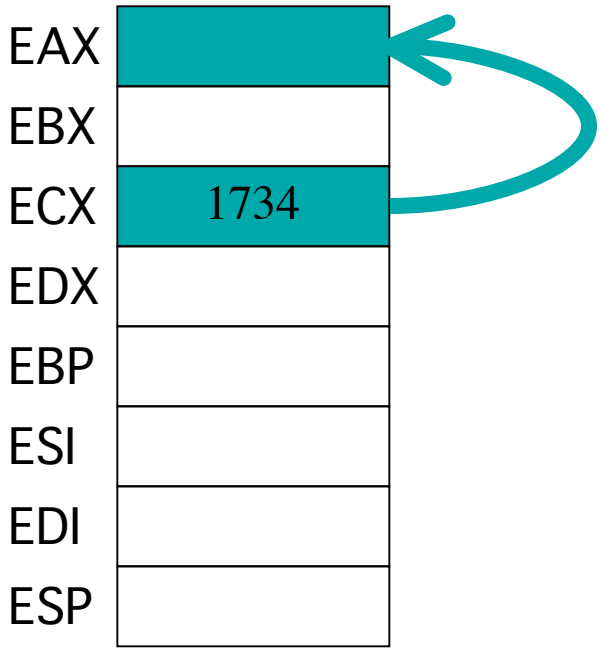
```
 1   ;
 2   ; Linux style "fast call". Assemble using NASM
 3   ;
 4           SECTION .data                   ; Data section
 5
 6   msg:    db "Hello, world", 10           ; The string to print.
 7   len:    equ $-msg
 8
 9           SECTION .text                   ; Code section.
10           global _start
11   _start: nop                             ; Entry point.
12           mov     edx, len                ; Arg 3: length of string.
13           mov     ecx, msg                ; Arg 2: pointer to string.
14           mov     ebx, 1                  ; Arg 1: file descriptor.
15           mov     eax, 4                  ; Write.
16           int     080H
17
18           mov     ebx, 0                  ; exit code, 0=normal
19           mov     eax, 1                  ; Exit.
20           int     080H                    ; Call kernel.
```

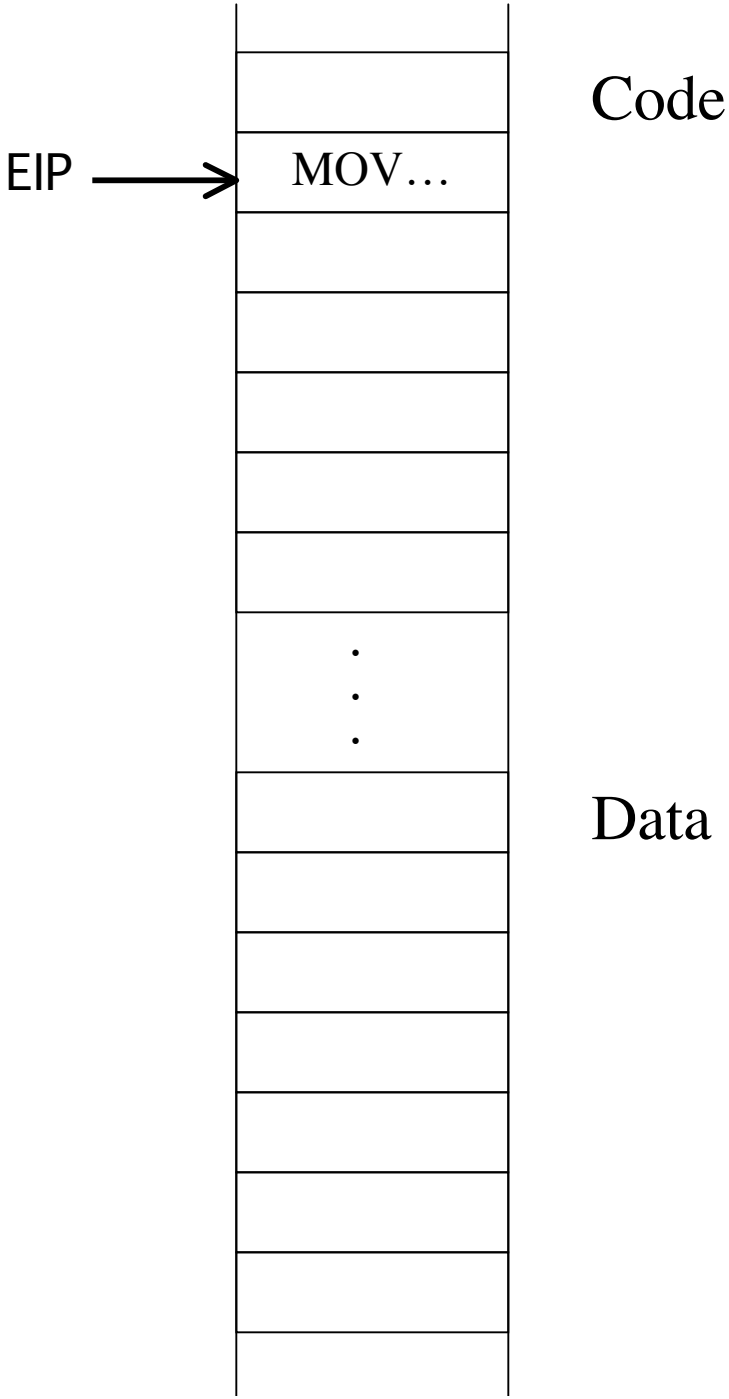
80x86 Addressing Modes

- We want to store the value 1734h.
- The value 1734h may be located in a register or in memory.
- The location in memory might be specified by the code, by a register, …
- Assembly language syntax for MOV

```
MOV       DEST,  SOURCE
```

Addressing Modes

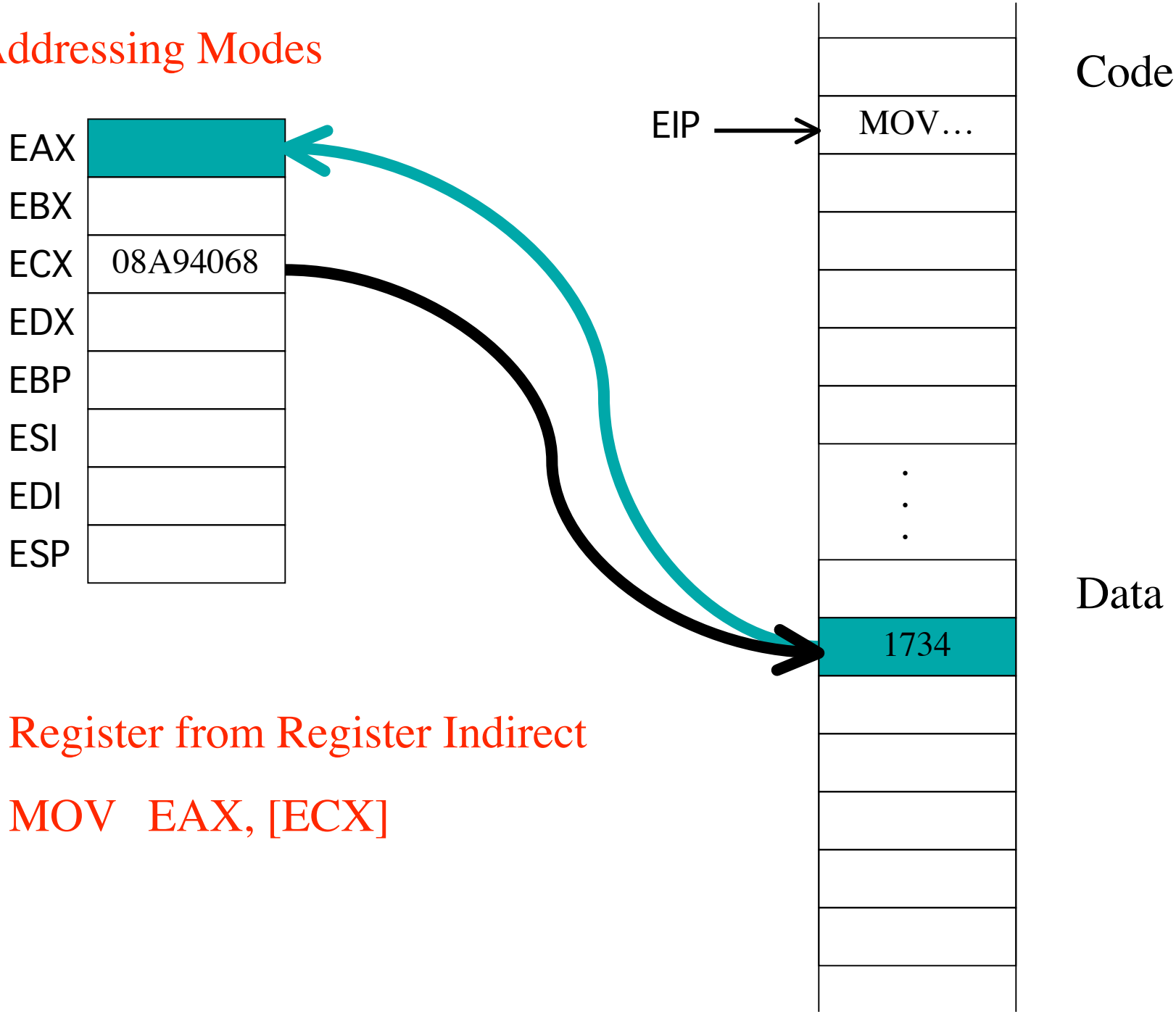




Register from Register

MOV EAX, ECX

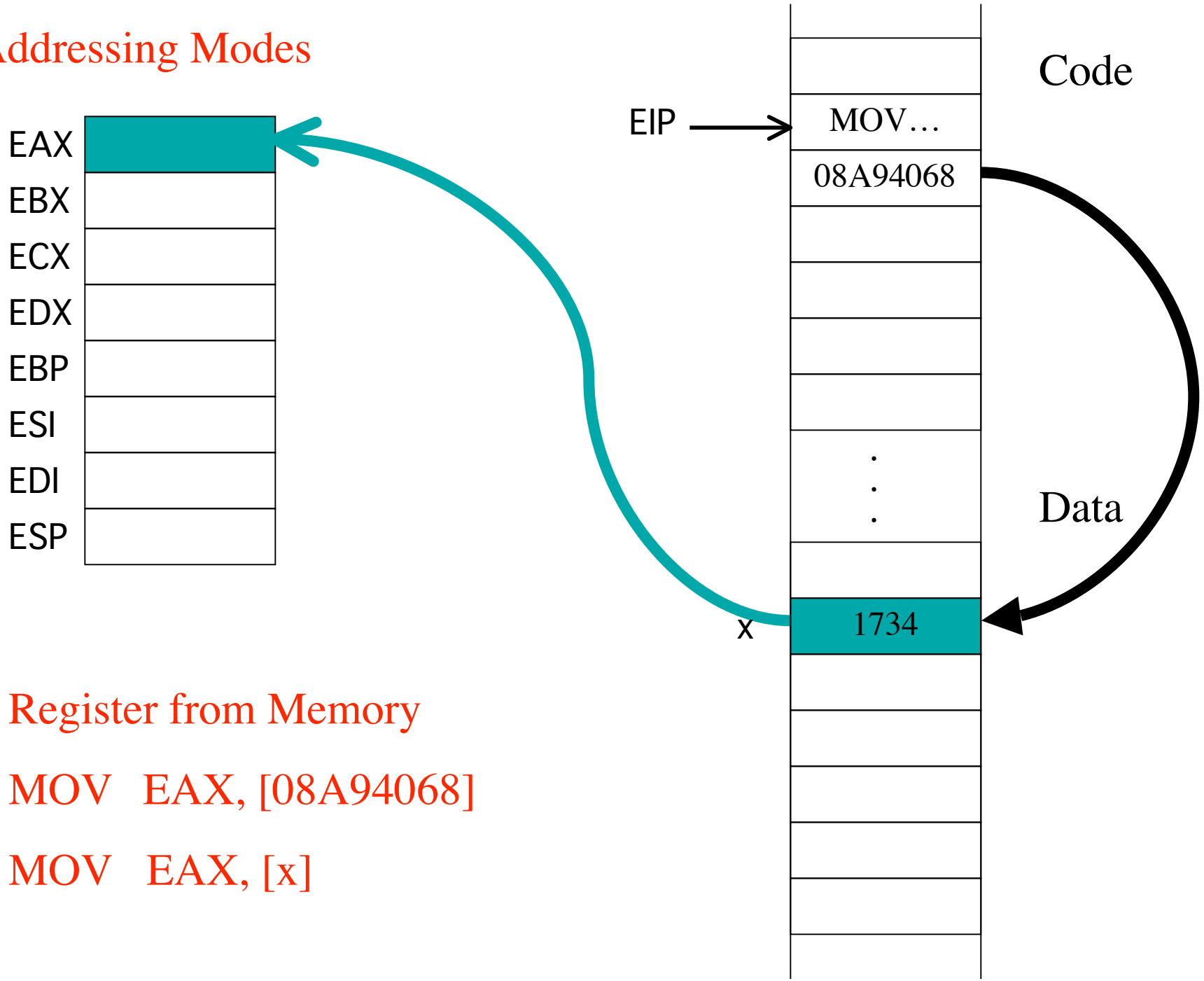
Addressing Modes



Register from Register Indirect

MOV EAX, [ECX]

Addressing Modes

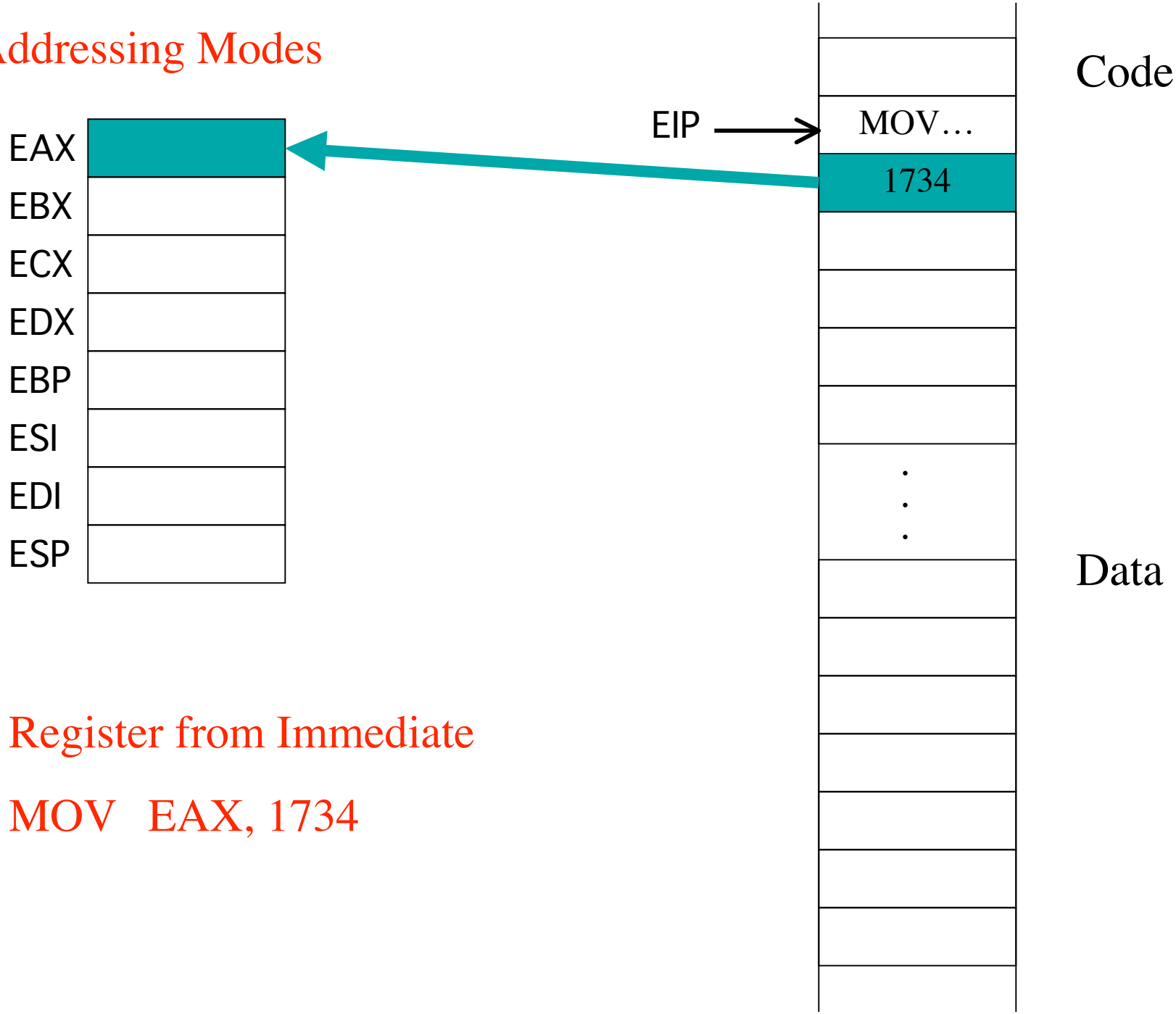


Register from Memory

MOV EAX, [08A94068]

MOV EAX, [x]

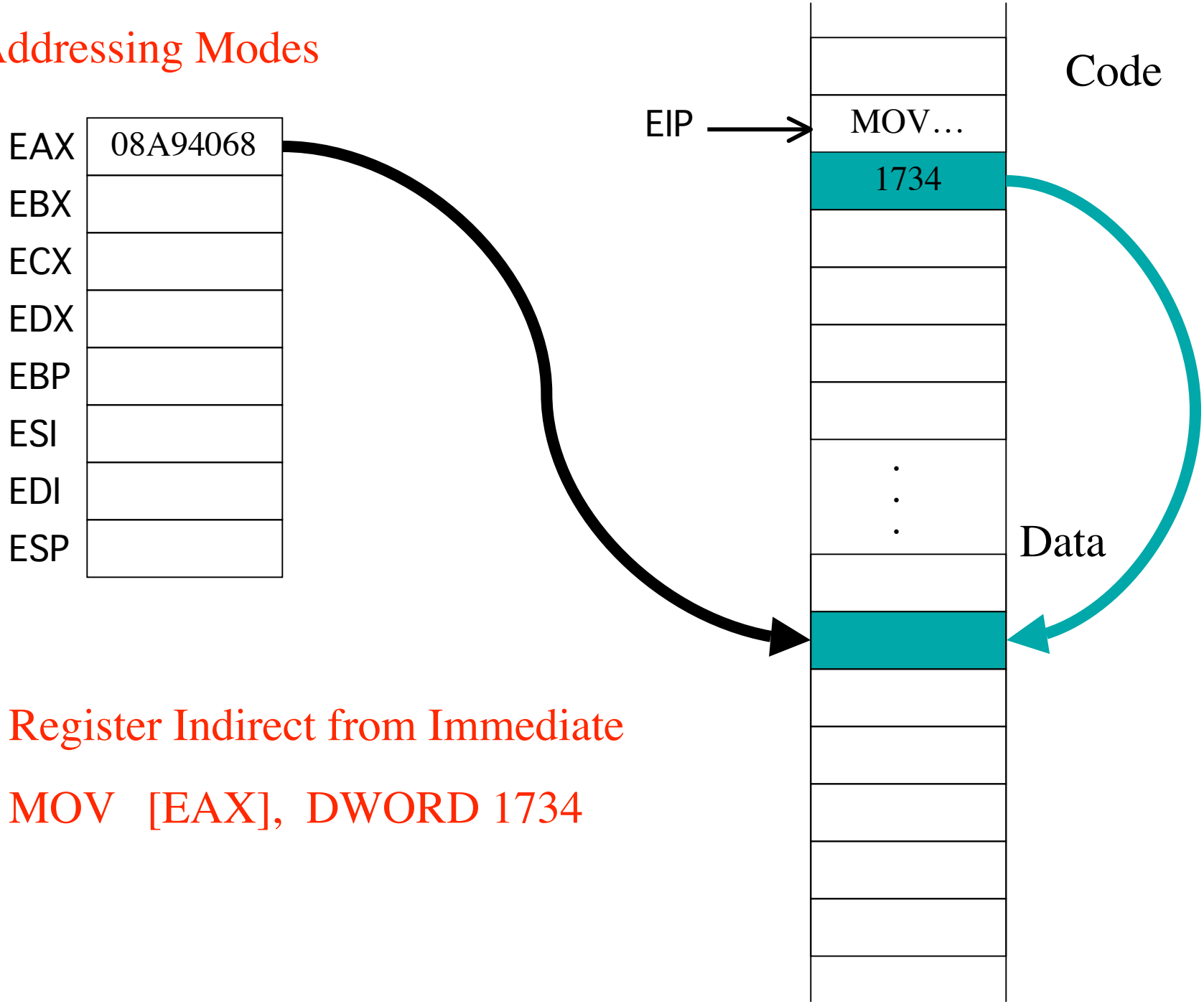
Addressing Modes



Register from Immediate

MOV EAX, 1734

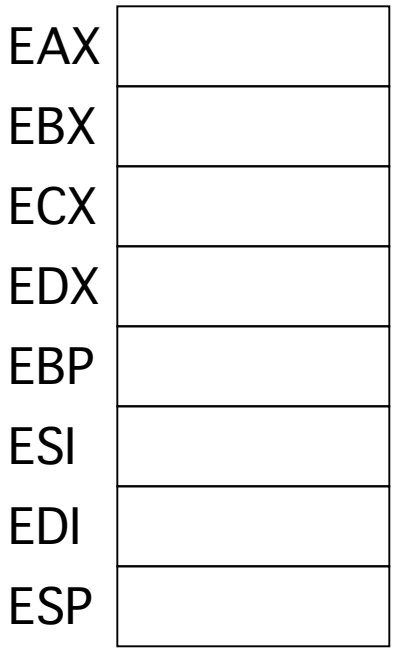
Addressing Modes

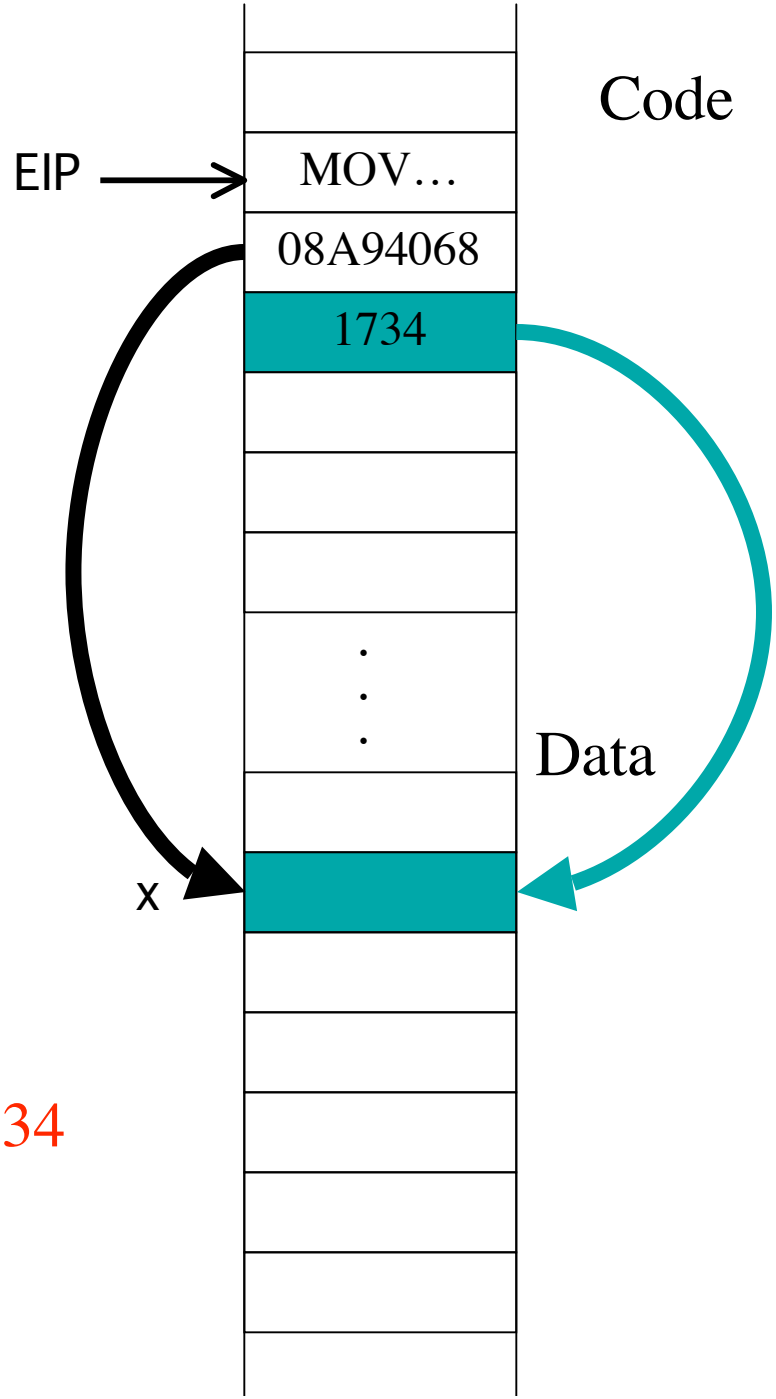


Register Indirect from Immediate

MOV [EAX], DWORD 1734

Addressing Modes





Memory from Immediate

MOV [08A94068], DWORD 1734

MOV [x], DWORD 1734

Notes on Addressing Modes

- More complicated addressing modes later:

```
MOV        EAX, [ESI+4*ECX+12]
```

- Figures not drawn to scale. Constants 1734h and 08A94068h take 4 bytes (little endian).
- Some addressing modes are not supported by some operations.
- Labels represent addresses not contents of memory.

Next Time

- Overview of i386 instruction set.
- Arithmetic instructions, logical instructions.
- EFLAGS register