CMSC 313
COMPUTER ORGANIZATION
&
ASSEMBLY LANGUAGE
PROGRAMMING

LECTURE 03, FALL 2012

TOPICS TODAY

- Moore's Law
- Evolution of Intel CPUs
- IA-32 Basic Execution Environment
- IA-32 General Purpose Registers
- "Hello World" in Linux Assembly Language
- Addressing modes
- gdb debugger demo

INTEL CPUS

1.5 Historical Development

- Moore's Law (1965)
  - Gordon Moore, Intel founder
  - "The density of transistors in an integrated circuit will double every year."
- Contemporary version:
  - "The density of silicon chips doubles every 18 months."

But this "law" cannot hold forever ...

1.5 Historical Development

- Rock's Law
  - Arthur Rock, Intel financier
  - "The cost of capital equipment to build semiconductors will double every four years."
  - In 1968, a new chip plant cost about $12,000.

At the time, $12,000 would buy a nice home in the suburbs.

An executive earning $12,000 per year was "making a very comfortable living."

1.5 Historical Development

- Rock's Law
  - In 2010, a chip plants under construction cost well over $4 billion.

    $4 billion is more than the gross domestic product of some small countries, including Barbados, Mauritania, and Rwanda.

  - For Moore's Law to hold, Rock's Law must fall, or vice versa. But no one can say which will give out first.

Table 2-3. Key Features of Previous Generations of IA-32 Processors

Intel Processor	Date Intro-duced	Max. Clock Frequency/ Technology at Introduction	Tran-sistors	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: 32 KB L2: 256 KB
Pentium 4 Processor	2000	1.50 GHz, Intel NetBurst Microarchitecture	42 M	32 GP 80 FPU 64 MMX 128 XMM	64	64 GB	12K µop Execution Trace Cache; L1: 8KB L2: 256 KB
Intel Xeon Processor	2001	1.70 GHz, Intel NetBurst Microarchitecture	42 M	32 GP 80 FPU 64 MMX 128 XMM	64	64 GB	12K µop Execution Trace Cache; L1: 8KB L2: 512KB
Intel Xeon Processor	2002	2.20 GHz, Intel NetBurst Microarchitecture, HyperThreading Technology	55 M	32 GP 80 FPU 64 MMX 128 XMM	64	64 GB	12K µop Execution Trace Cache; L1: 8KB L2: 512KB
Pentium M Processor	2003	1.60 GHz, Intel NetBurst Microarchitecture	77 M	32 GP 80 FPU 64 MMX 128 XMM	64	4 GB	L1: 64KB L2: 1 MB
Intel Pentium 4 Processor Supporting Hyper-Threading Technology at 90 nm process	2004	3.40 GHz, Intel NetBurst Microarchitecture, HyperThreading Technology	125 M	32 GP 80 FPU 64 MMX 128 XMM	64	64 GB	12K µop Execution Trace Cache; L1: 16KB L2: 1 MB

NOTE:

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

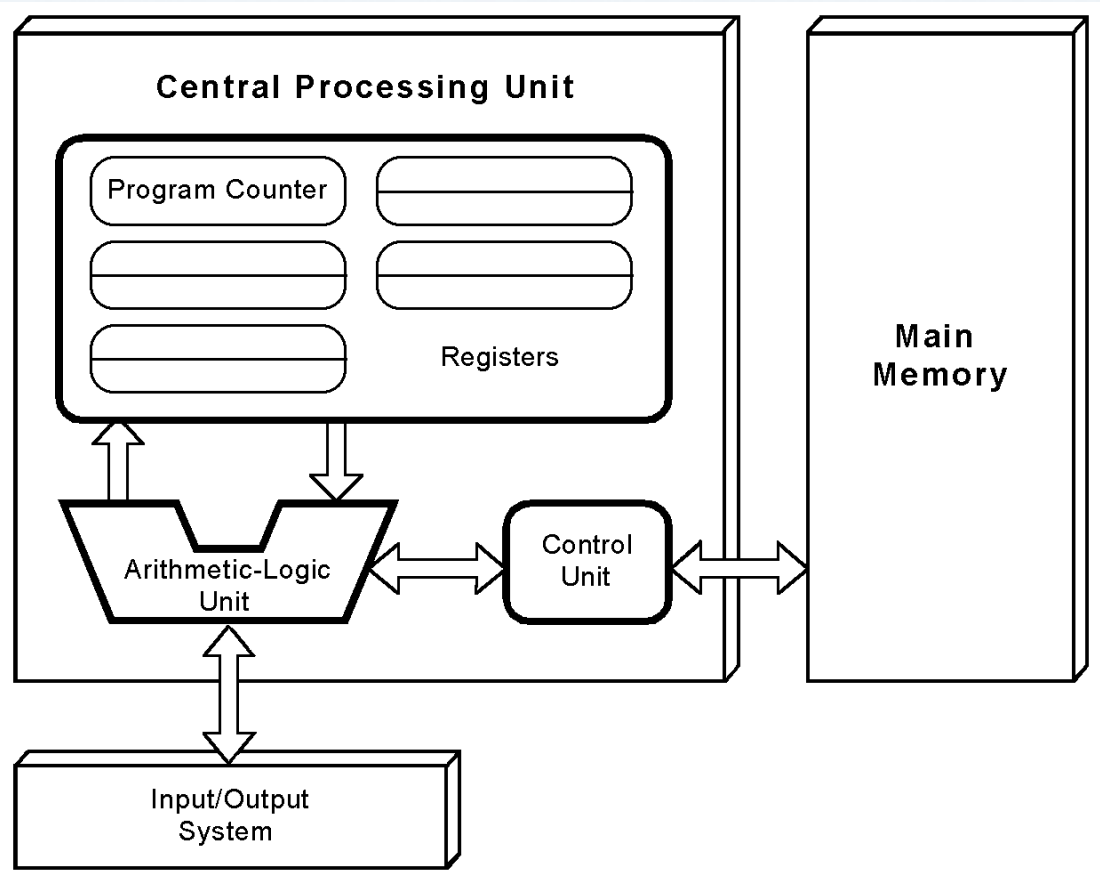
Intel Processor	Date Intro-duced	Micro-architecture	Top-Bin Clock Fre-quency at Intro-duction	Tran-sistors	Register Sizes[1]	Syste m Bus Band-width	Max. Extern. Addr. Space	On-Die Caches[2]
Intel Pentium M Processor 755[3]	2004	Intel Pentium M Processor	2.00 GHz	140 M	GP: 32 FPU: 80 MMX: 64 XMM: 128	3.2 GB/s	4 GB	L1: 64 KB L2: 2 MB
Intel Core Duo Processor T2600[3]	2006	Improved Intel Pentium M Processor Microarchitecture; Dual Core; Intel Smart Cache, Advanced Thermal Manager	2.16 GHz	152M	GP: 32 FPU: 80 MMX: 64 XMM: 128	5.3 GB/s	4 GB	L1: 64 KB L2: 2 MB (2MB Total)
Intel Atom Processor Z5xx series	2008	Intel Atom Microarchitecture; Intel Virtualization Technology.	1.86 GHz - 800 MHz	47M	GP: 32 FPU: 80 MMX: 64 XMM: 128	Up to 4.2 GB/s	4 GB	L1: 56 KB[4] L2: 512KB

Table 2-2. Key Features of Most Recent Intel 64 Processors (Contd.)

Intel Processor	Date Intro-duced	Micro-architec-ture	Top-Bin Fre-quency at Intro-duction	Tran-sistors	Register Sizes	System Bus/QPI Link Speed	Max. Extern. Addr. Space	On-Die Caches
Intel Core i7-620M Processor	2010	Intel Turbo Boost Technology, Intel microarchitecture code name Westmere; Dualcore; HyperThreading Technology; Intel 64 Architecture; Intel Virtualization Technology., Integrated graphics	2.66 GHz	383 M	GP: 32, 64 FPU: 80 MMX: 64 XMM: 128		64 GB	L1: 64 KB L2: 256KB L3: 4MB
Intel Xeon-Processor 5680	2010	Intel Turbo Boost Technology, Intel microarchitecture code name Westmere; Six core; HyperThreading Technology; Intel 64 Architecture; Intel Virtualization Technology.	3.33 GHz	1.1B	GP: 32, 64 FPU: 80 MMX: 64 XMM: 128	QPI: 6.4 GT/s; 32 GB/s	1 TB	L1: 64 KB L2: 256KB L3: 12MB
Intel Xeon-Processor 7560	2010	Intel Turbo Boost Technology, Intel microarchitecture code name Nehalem; Eight core; HyperThreading Technology; Intel 64 Architecture; Intel Virtualization Technology.	2.26 GHz	2.3B	GP: 32, 64 FPU: 80 MMX: 64 XMM: 128	QPI: 6.4 GT/s; Memory: 76 GB/s	16 TB	L1: 64 KB L2: 256KB L3: 24MB
Intel Core i7-2600K Processor	2011	Intel Turbo Boost Technology, Intel microarchitecture code name Sandy Bridge; Four core; HyperThreading Technology; Intel 64 Architecture; Intel Virtualization Technology., Processor graphics, Quicksync Video	3.40 GHz	995M	GP: 32, 64 FPU: 80 MMX: 64 XMM: 128 YMM: 256	DMI: 5 GT/s; Memory: 21 GB/s	64 GB	L1: 64 KB L2: 256KB L3: 8MB
Intel Xeon-Processor E3-1280	2011	Intel Turbo Boost Technology, Intel microarchitecture code name Sandy Bridge; Four core; HyperThreading Technology; Intel 64 Architecture; Intel Virtualization Technology.	3.50 GHz		GP: 32, 64 FPU: 80 MMX: 64 XMM: 128 YMM: 256	DMI: 5 GT/s; Memory: 21 GB/s	1 TB	L1: 64 KB L2: 256KB L3: 8MB
Intel Xeon-Processor E7-8870	2011	Intel Turbo Boost Technology, Intel microarchitecture code name Westmere; Ten core; HyperThreading Technology; Intel 64 Architecture; Intel Virtualization Technology.	2.40 GHz	2.2B	GP: 32, 64 FPU: 80 MMX: 64 XMM: 128	QPI: 6.4 GT/s; Memory: 102 GB/s	16 TB	L1: 64 KB L2: 256KB L3: 30MB

1.7 The von Neumann Model

- **This is a general depiction of a von Neumann system:**
- **These computers employ a fetch-decode-execute cycle to run programs as follows . . .**



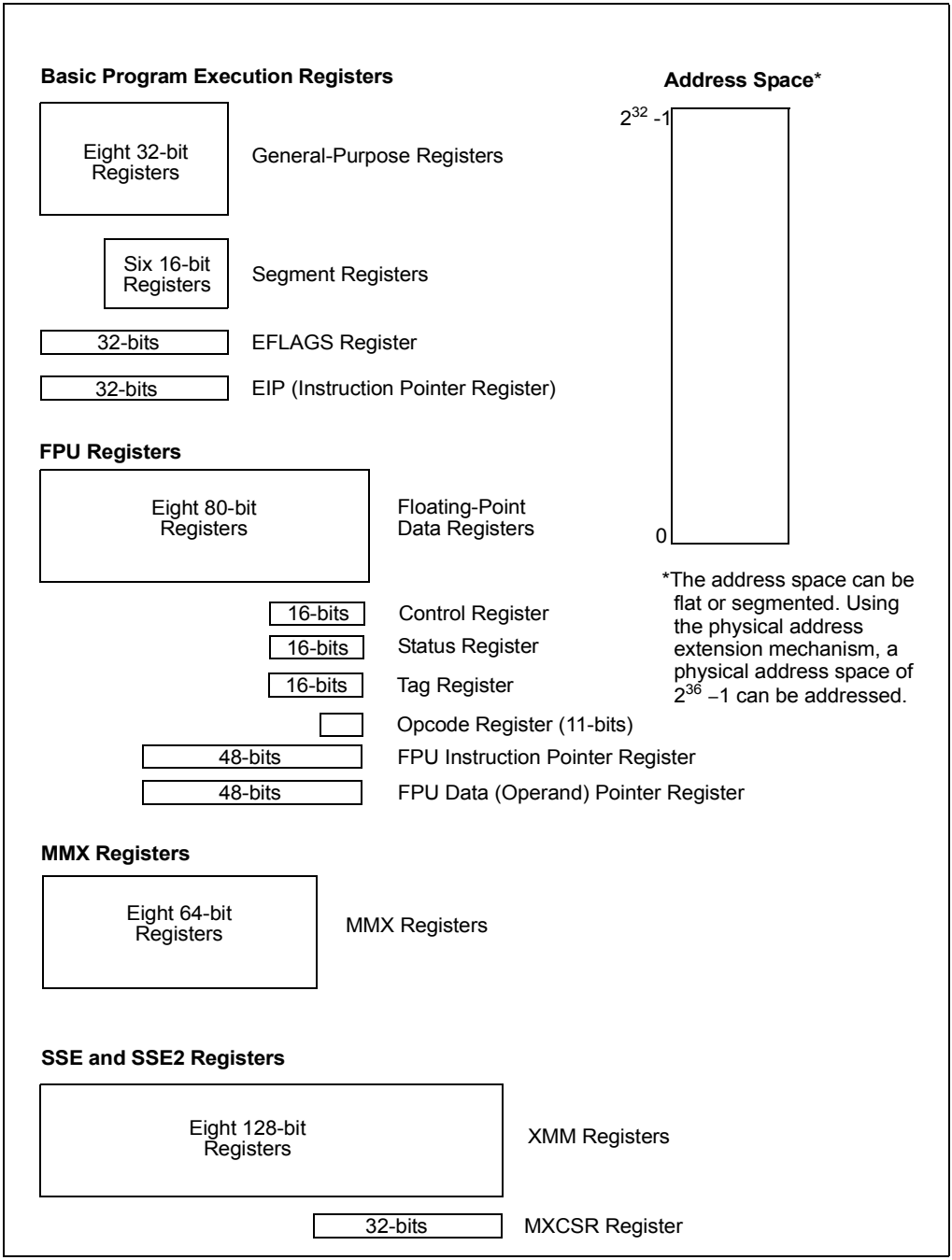


Figure 3-1. IA-32 Basic Execution Environment

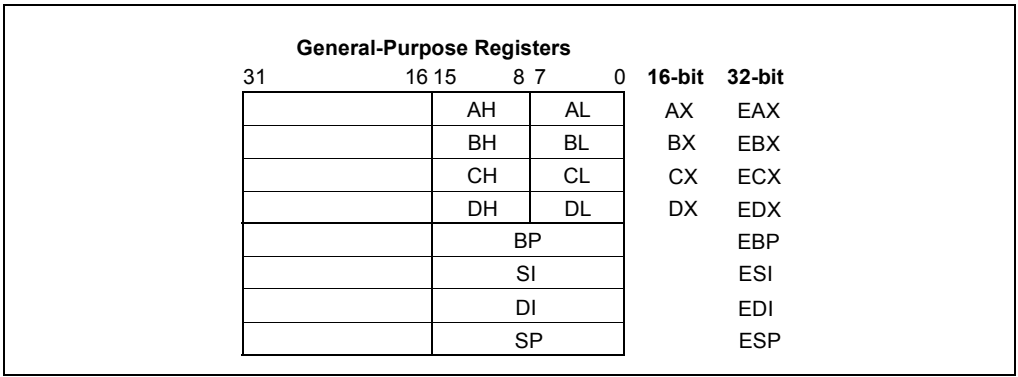


Figure 3-4. Alternate General-Purpose Register Names

- 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).

"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

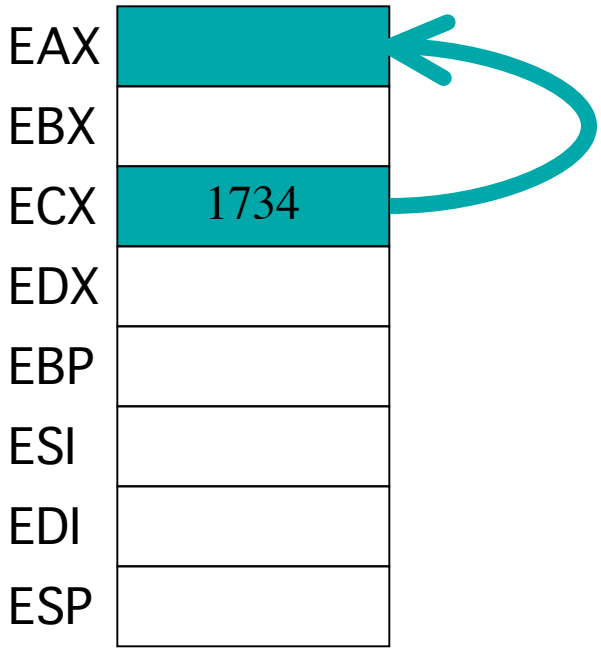
ADDRESSING MODES

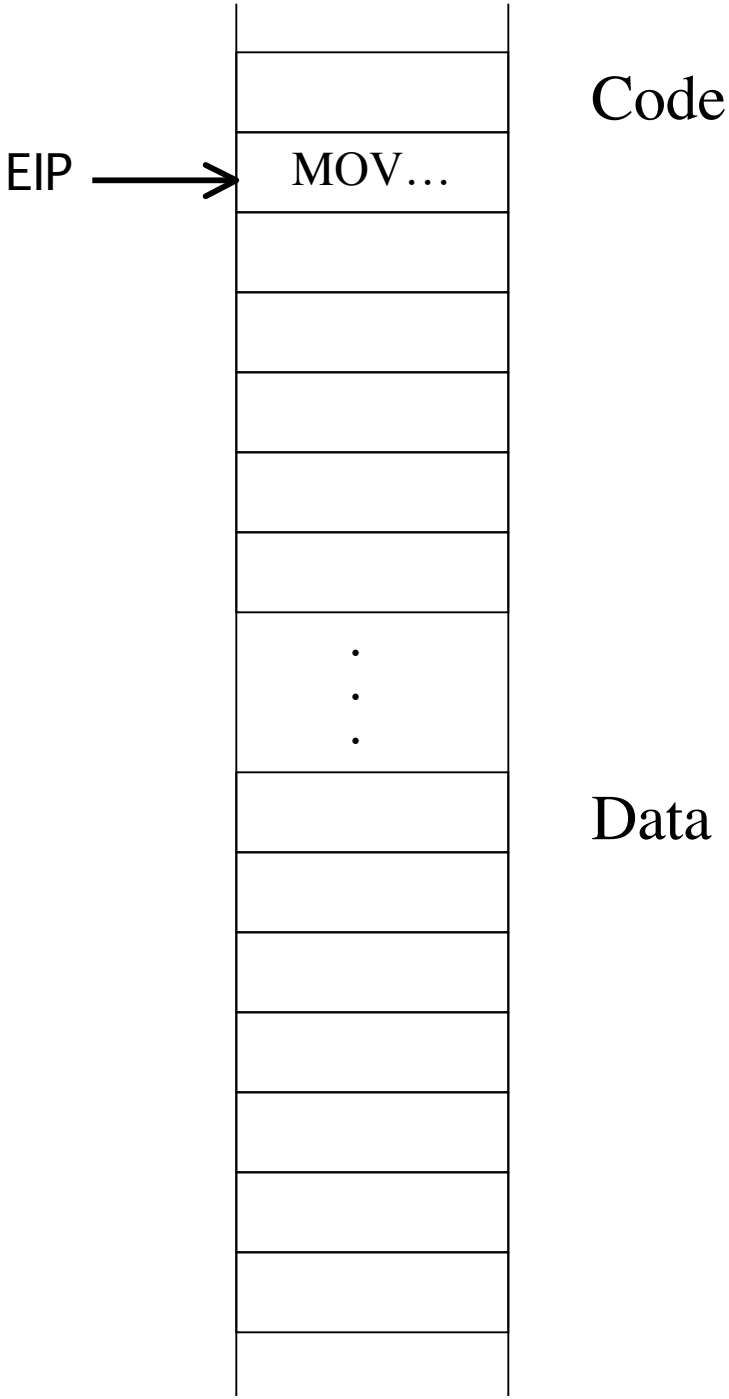
80x86 Addressing Modes

- 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

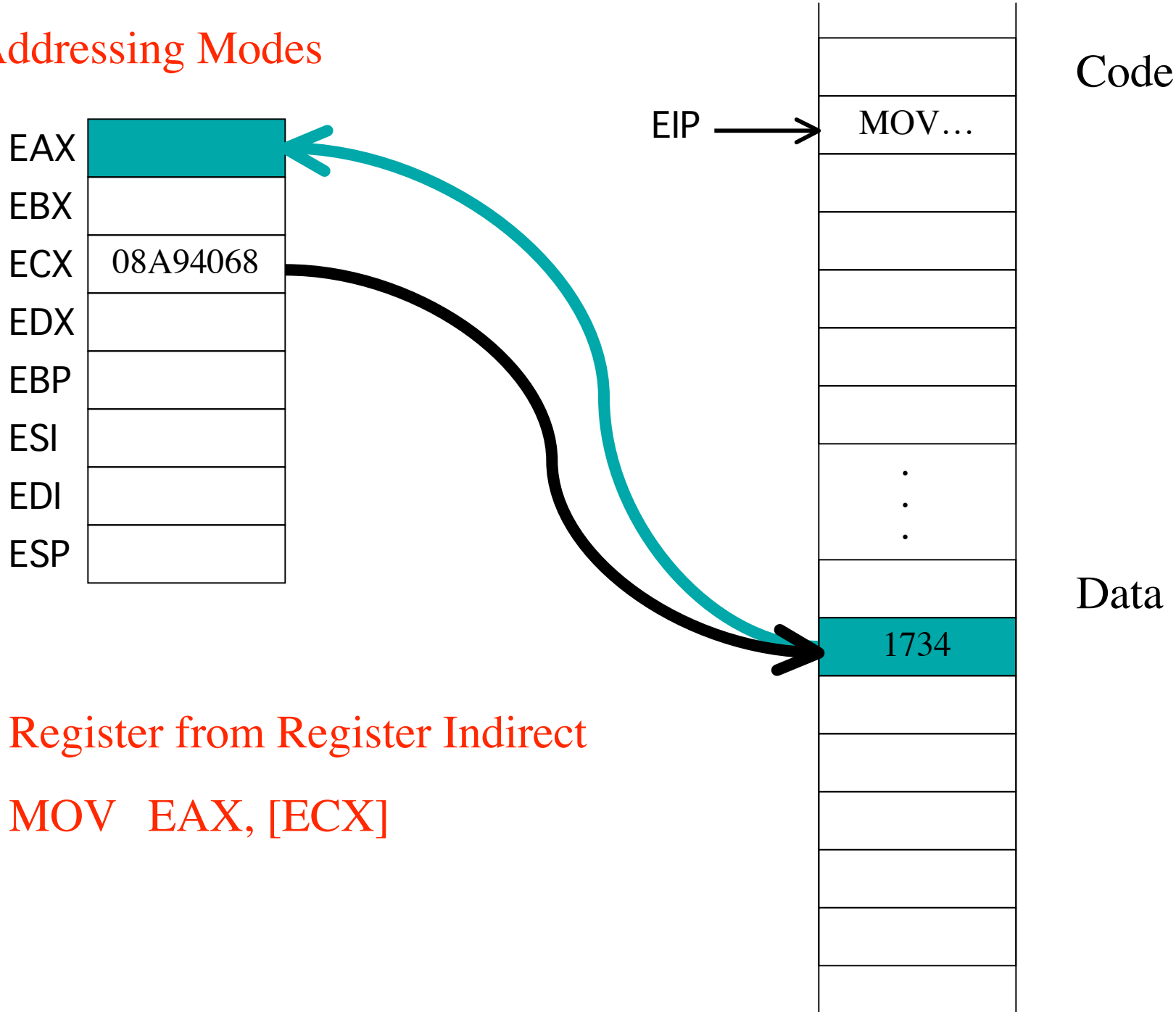




Register from Register

MOV EAX, ECX

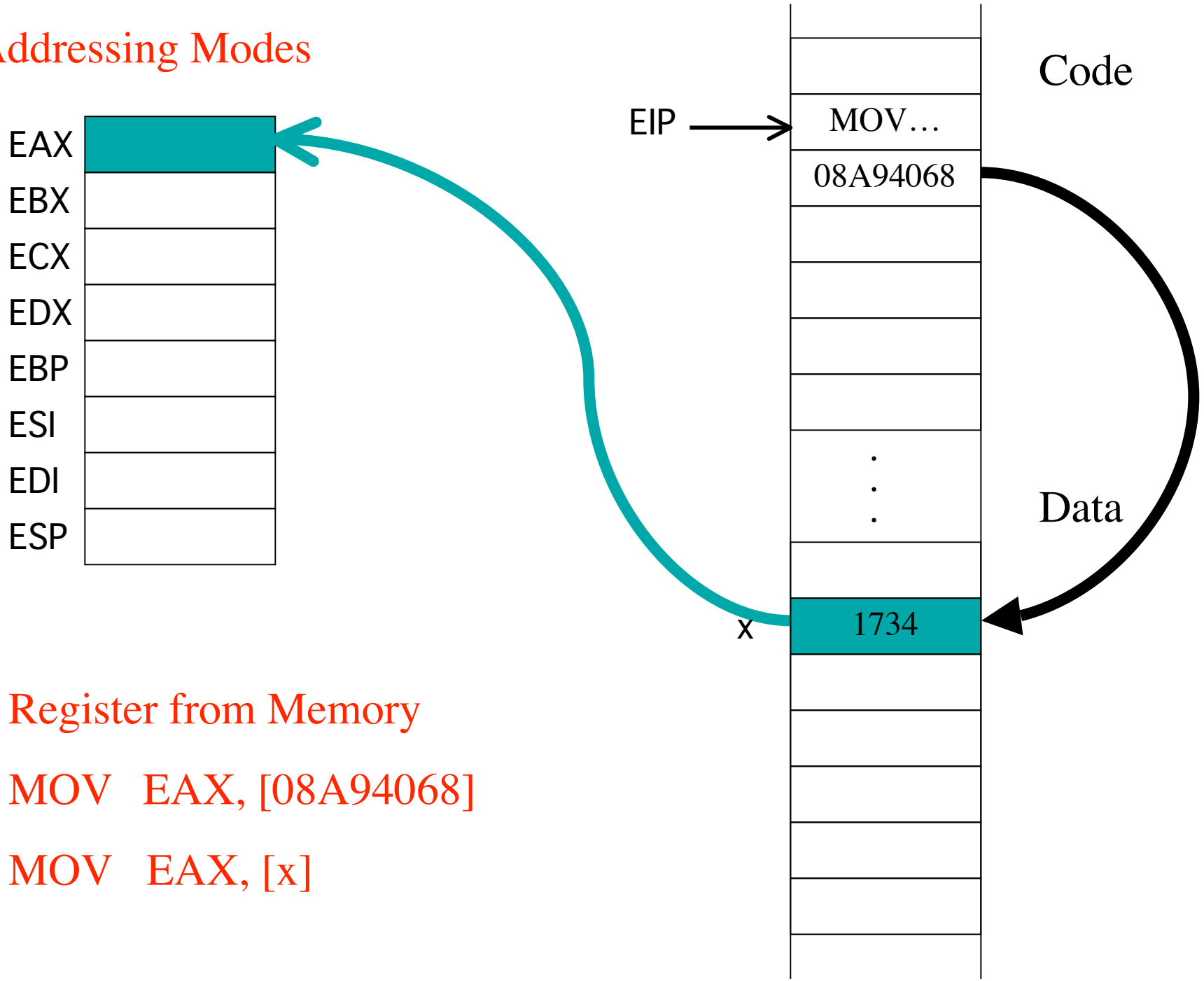
Addressing Modes



Register from Register Indirect

MOV EAX, [ECX]

Addressing Modes

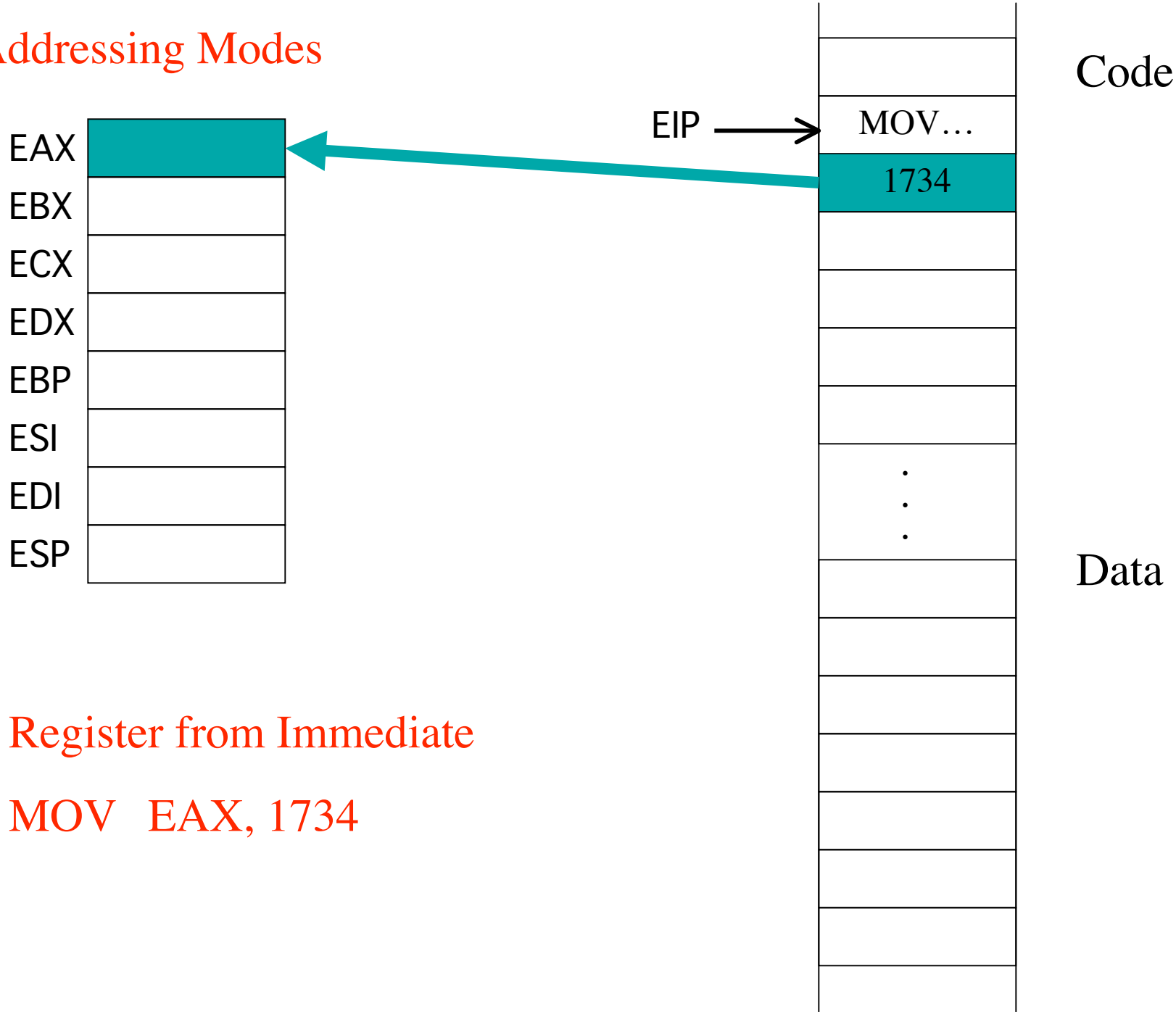


Register from Memory

MOV EAX, [08A94068]

MOV EAX, [x]

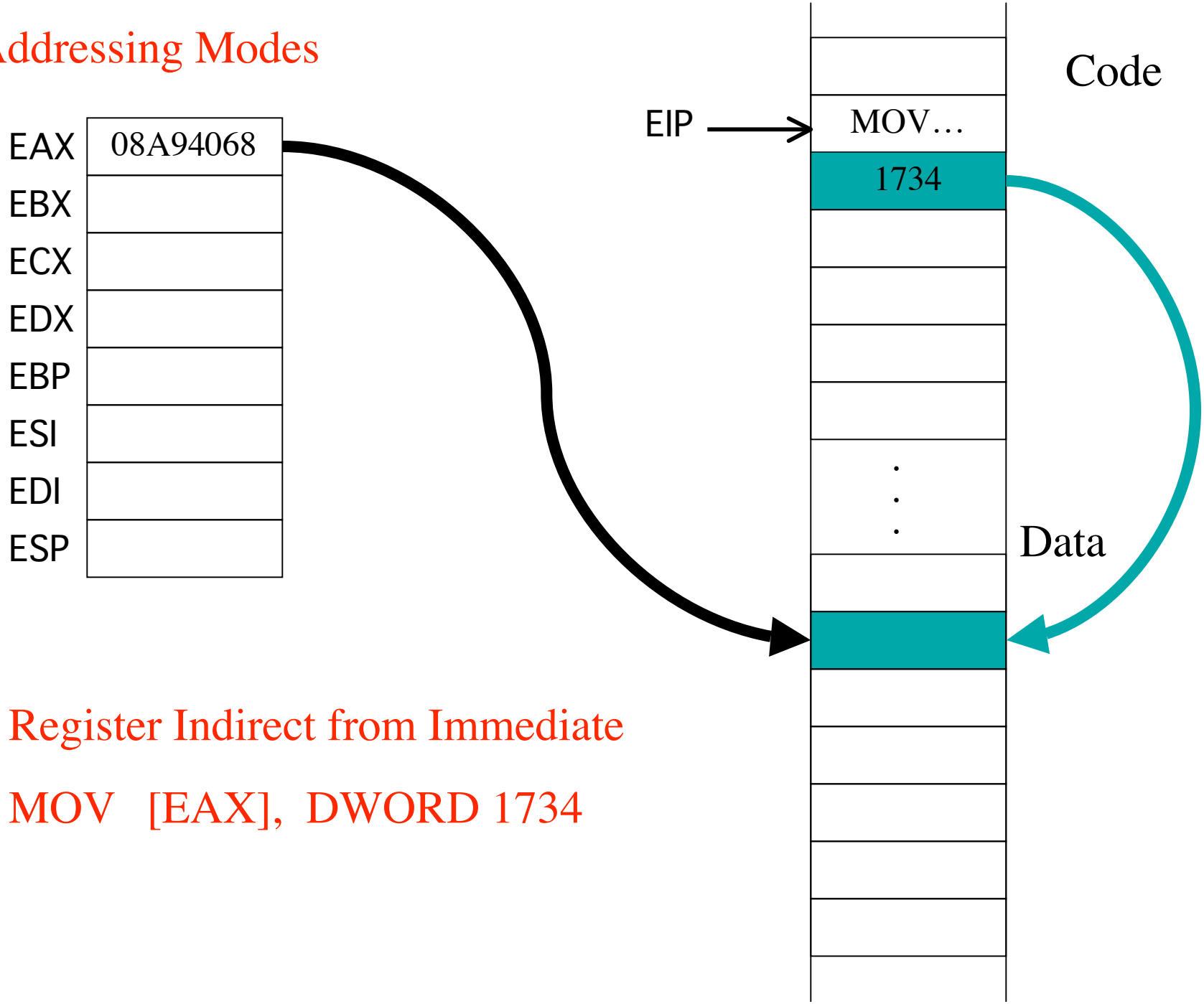
Addressing Modes



Register from Immediate

MOV EAX, 1734

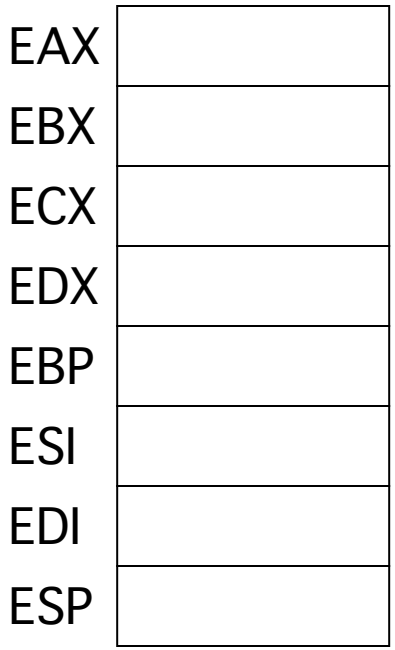
Addressing Modes

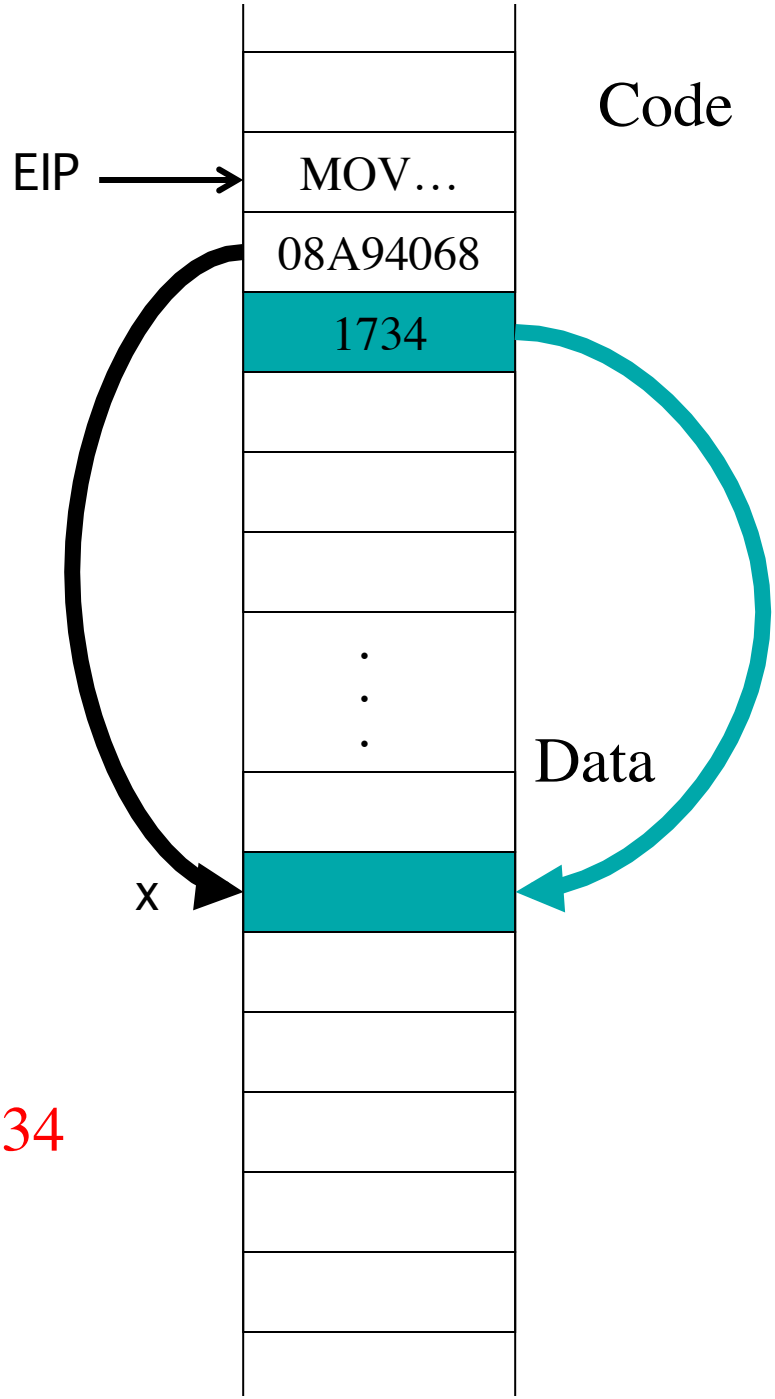


Register Indirect from Immediate

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.

toupper.asm

- Prompt for user input.
- Use Linux system call to get user input.
- Scan each character of user input and convert all lower case characters to upper case.
- How to:
  - work with 8-bit data
  - specify ASCII constant
  - compare values
  - loop control

THE GDB DEBUGGER

Debugging Assembly Language Programs

- Cannot just put print statements everywhere.
- Use gdb to:
  - examine contents of registers
  - exmaine contents of memory
  - set breakpoints
  - single-step through program
- READ THE GDB SUMMARY ONLINE!

Summary of `gdb` commands

Command	Example	Description
run		start program
quit		quit out of gdb
cont		continue execution after a break
break [addr]	break *_start+5	sets a breakpoint
delete [n]	delete 4	removes nth breakpoint
delete		removes all breakpoints
info break		lists all breakpoints
list _start		list a few lines of the source code around _start
list 7		list 10 lines of the source code starting on line 7
list 7, 20		list lines 7 thru 20 of the source code
stepi		execute next instruction
stepi [n]	stepi 4	execute next n instructions
nexti		execute next instruction, stepping over function calls
nexti [n]	nexti 4	execute next n instructions, stepping over function calls
where		show where execution halted
disas [addr]	disas _start	disassemble instructions at given address
info registers		dump contents of all registers
print/d [expr]	print/d $ecx	print expression in decimal
print/x [expr]	print/x $ecx	print expression in hex
print/t [expr]	print/t $ecx	print expression in binary
x/NFU [addr]	x/12xw &msg	Examine contents of memory in given format
display [expr]	display $eax	automatically print the expression each time the program is halted
info display		show list of automatically displays
undisplay [n]	undisplay 1	remove an automatic display

Next Time

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