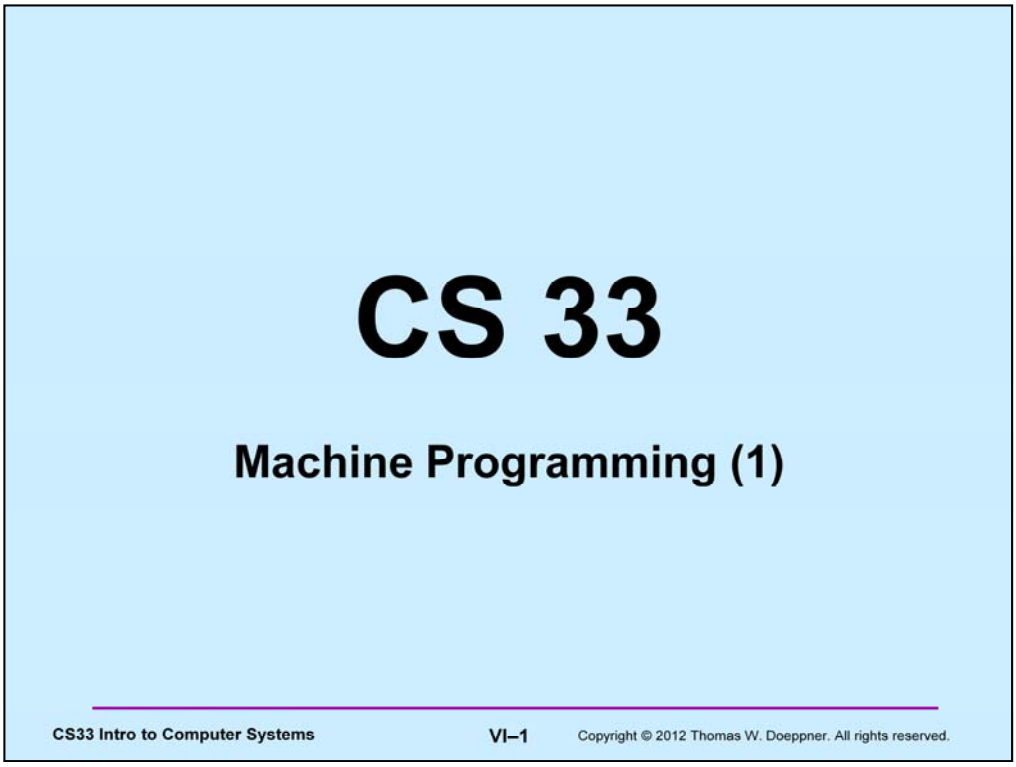
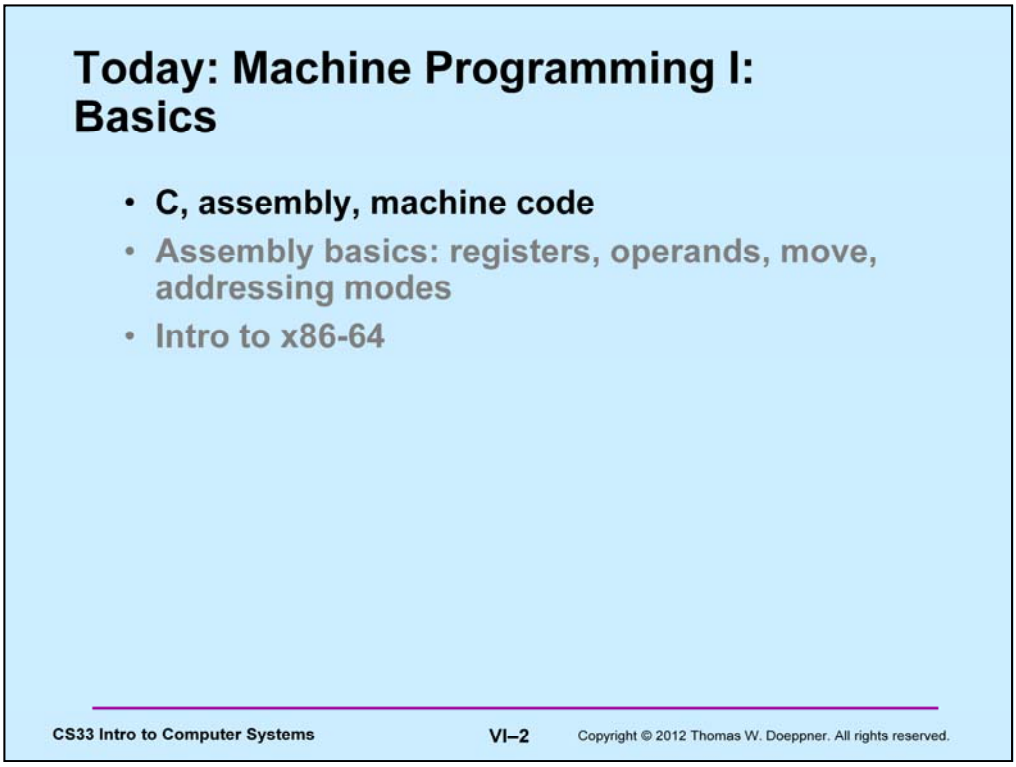
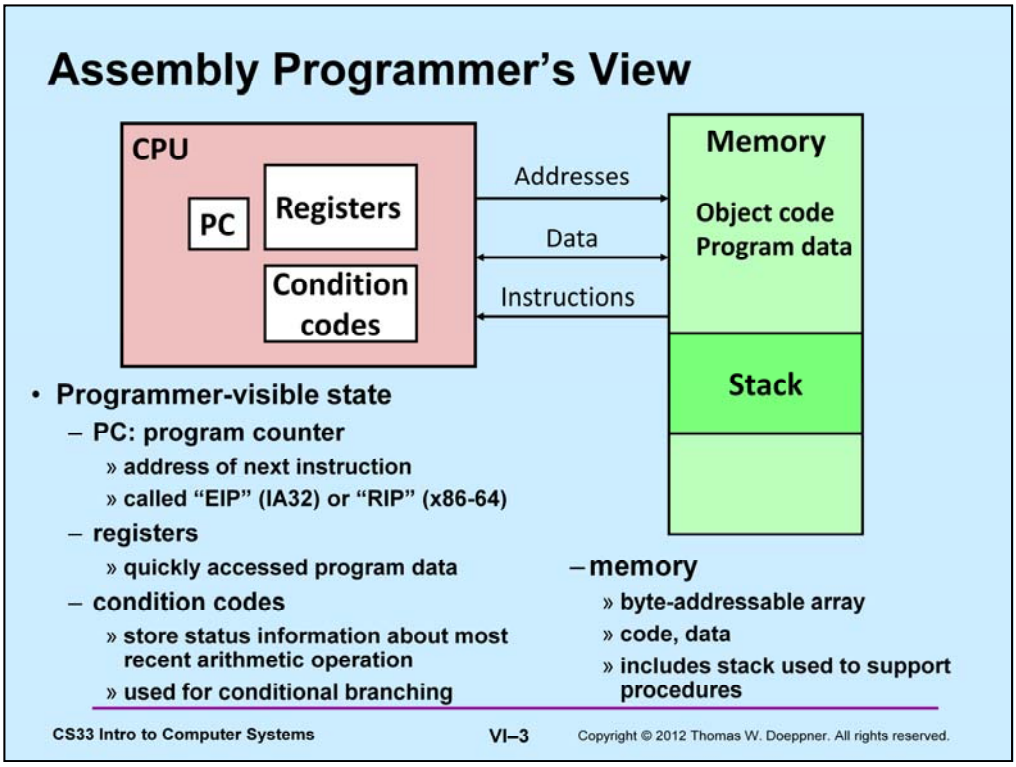


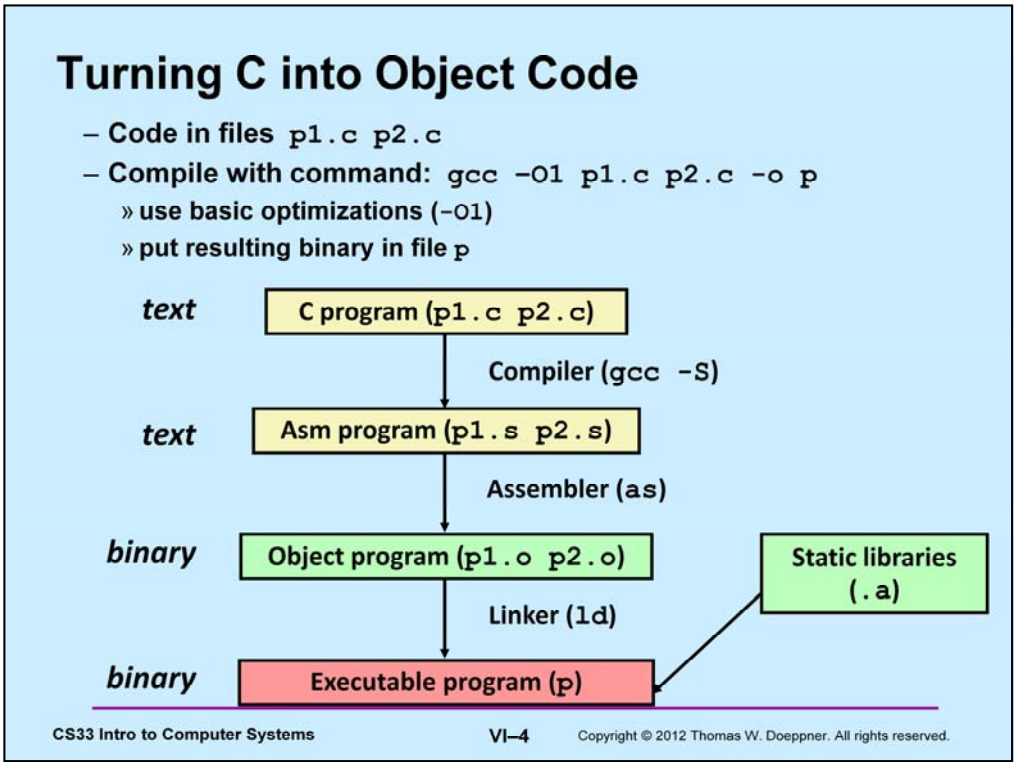
Many of the slides in this lecture are either from or adapted from slides provided by the authors of the textbook "Computer Systems: A Programmer's Perspective," 2nd Edition and are provided from the website of Carnegie-Mellon University, course 15-213, taught by Randy Bryant and David O'Hallaron in Fall 2010. These slides are indicated "Supplied by CMU" in the notes section of the slides.



Supplied by CMU.

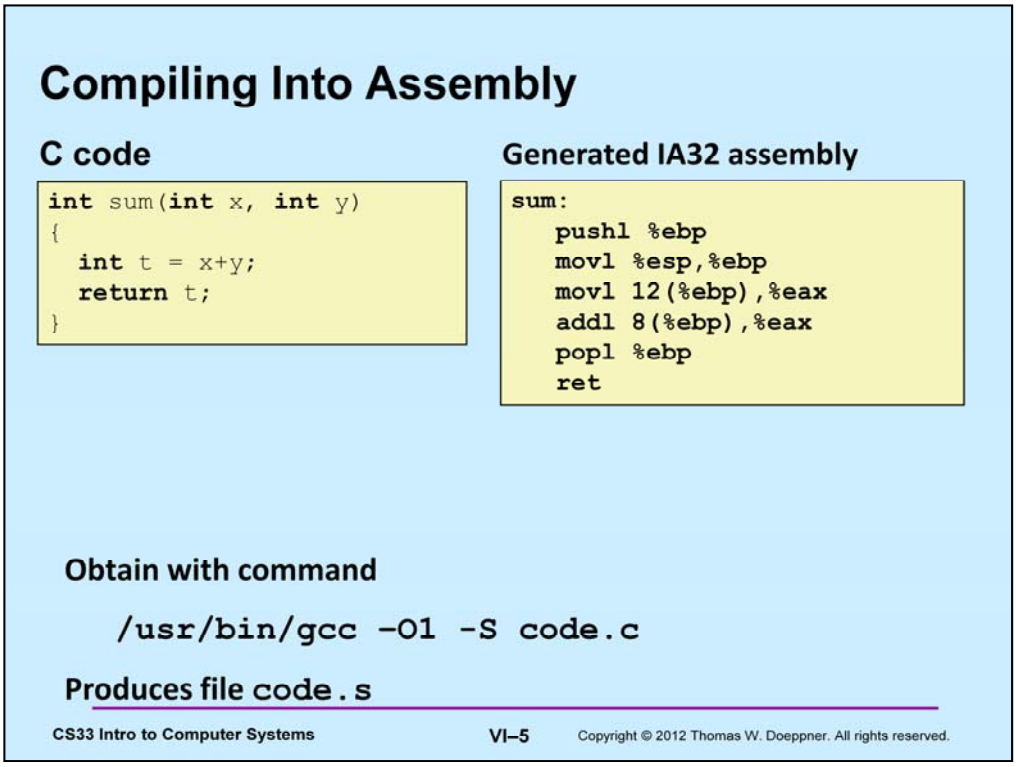


Supplied by CMU.



Supplied by CMU.

Note that normally one does not ask gcc to produce assembler code, but instead it compiles C code directly into machine code (producing an object file). Note that the gcc command actually invokes a script; the compiler (also known as gcc) compiles code into either assembler code or machine code; if necessary, the assembler (as) assembles assembler code into a object code. The linker (ld) links together multiple object files (containing object code) into an executable program.



Supplied by CMU.

Assembly Characteristics: Data Types

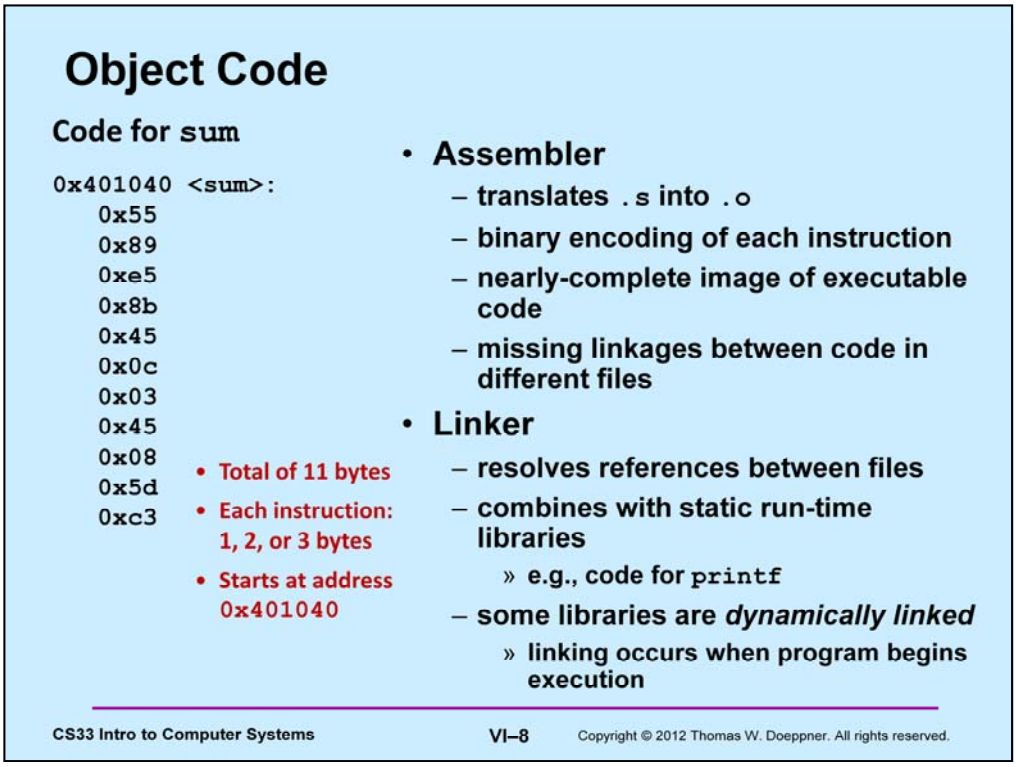
- "Integer" data of 1, 2, or 4 bytes
  - data values
  - addresses (untyped pointers)
- Floating-point data of 4, 8, or 10 bytes
- No aggregate types such as arrays or structures
  - just contiguously allocated bytes in memory

Supplied by CMU.

Assembly Characteristics: Operations

- Perform arithmetic function on register or memory data
- Transfer data between memory and register
  - load data from memory into register
  - store register data into memory
- Transfer control
  - unconditional jumps to/from procedures
  - conditional branches

Supplied by CMU.



Supplied by CMU.

Disassembling Object Code

Disassembled

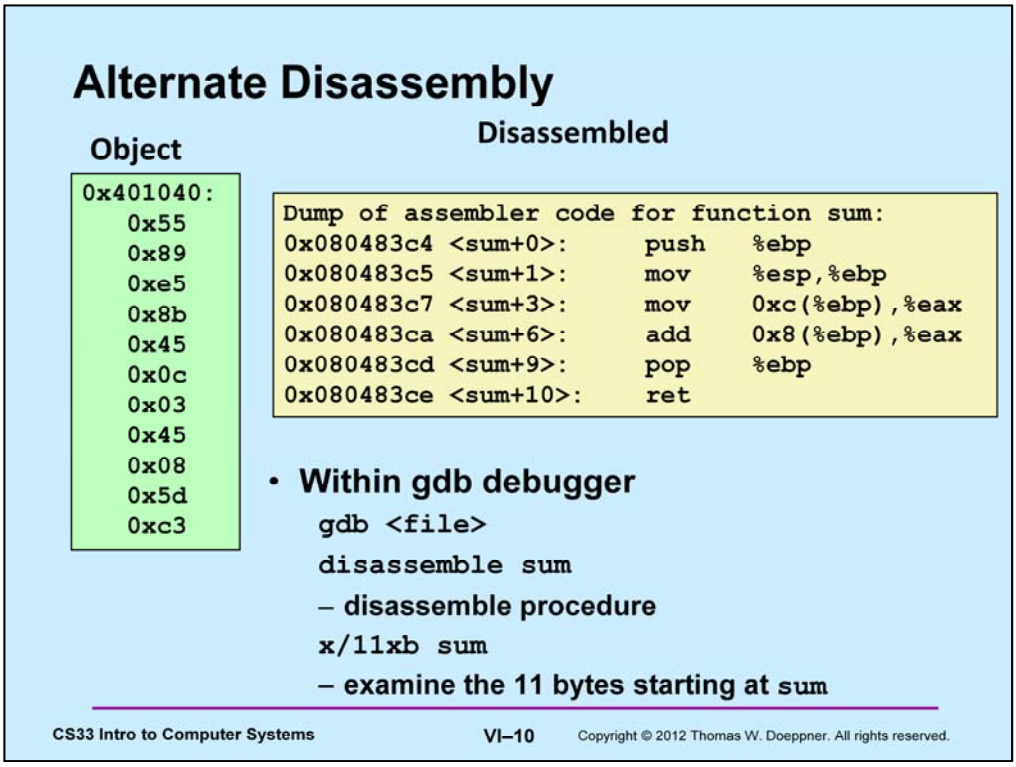
```
080483c4 <sum>:
 80483c4:   55          push   %ebp
 80483c5:   89 e5       mov    %esp,%ebp
 80483c7:   8b 45 0c    mov    0xc(%ebp),%eax
 80483ca:   03 45 08    add    0x8(%ebp),%eax
 80483cd:   5d          pop    %ebp
 80483ce:   c3          ret
```

- **Disassembler**

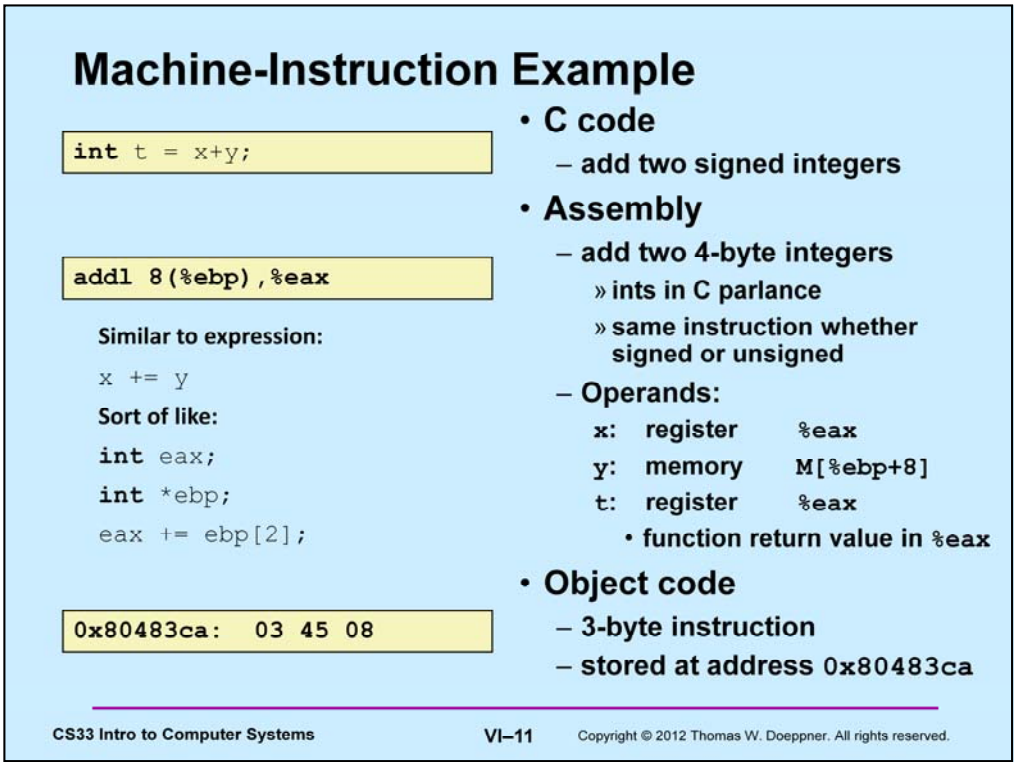
  `objdump -d <file>`

  - useful tool for examining object code
  - analyzes bit pattern of series of instructions
  - produces approximate rendition of assembly code
  - can be run on either executable or object (`.o`) file

Supplied by CMU.



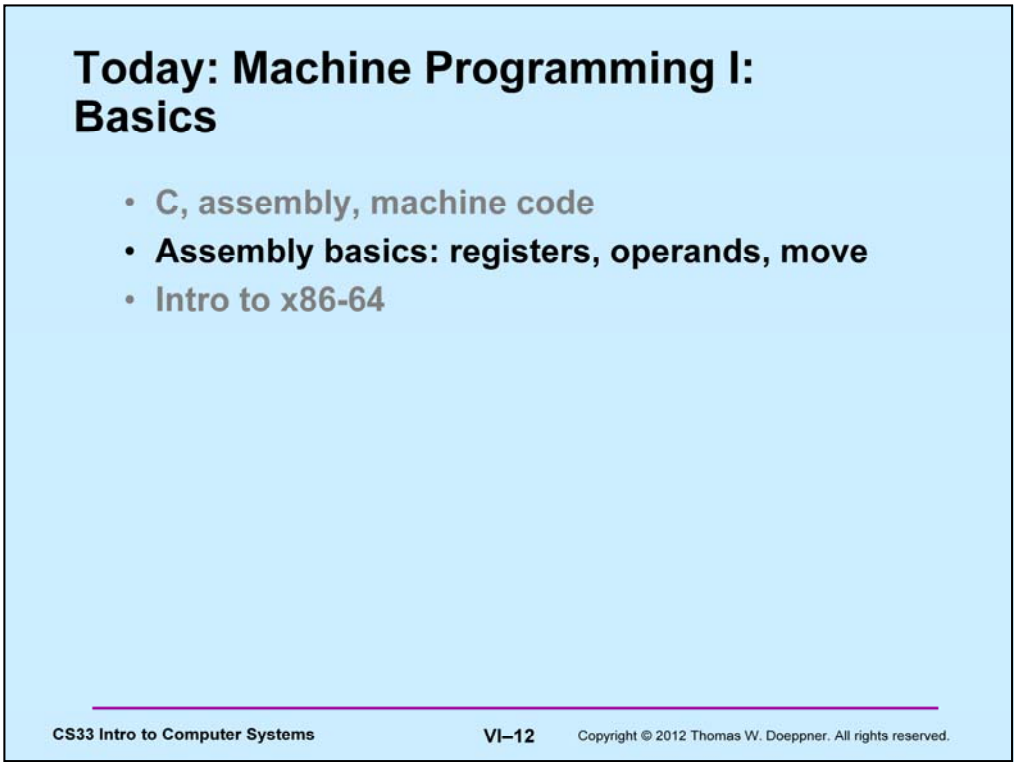
Supplied by CMU.



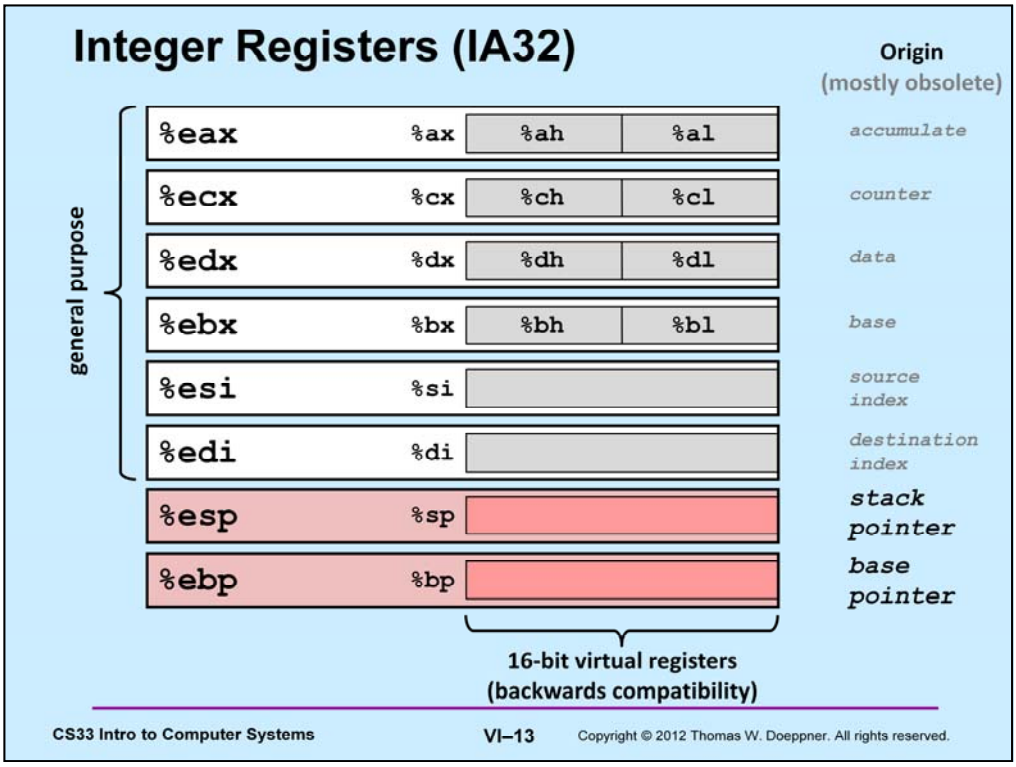
Supplied by CMU.

The "M" in "M[%ebp+8]" is the conceptual memory array. In the case of this instruction, we are adding 8 to the contents of register ebp to produce an address, which is treated as an index into the "memory array."

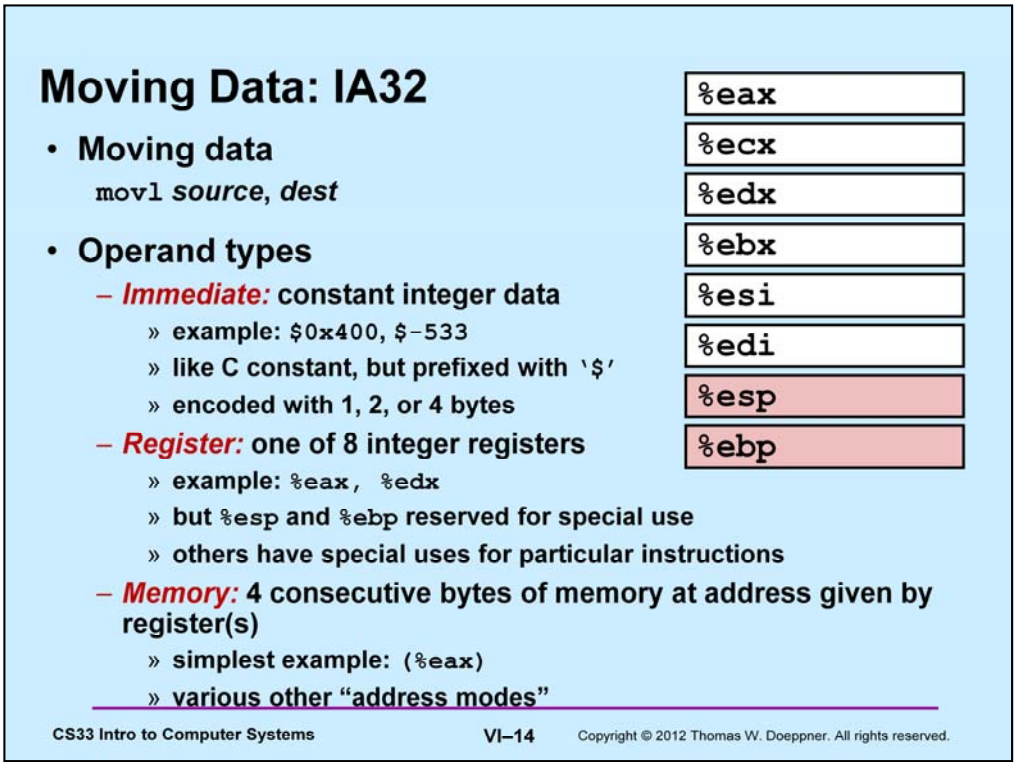
Note that some assemblers (in particular, those of Intel and Microsoft) place the operands in the opposite order. Thus the example of the slide would be "addl %eax,8(%ebp)". The order we use is that used by gcc.



Supplied by CMU.



Supplied by CMU.



Supplied by CMU.

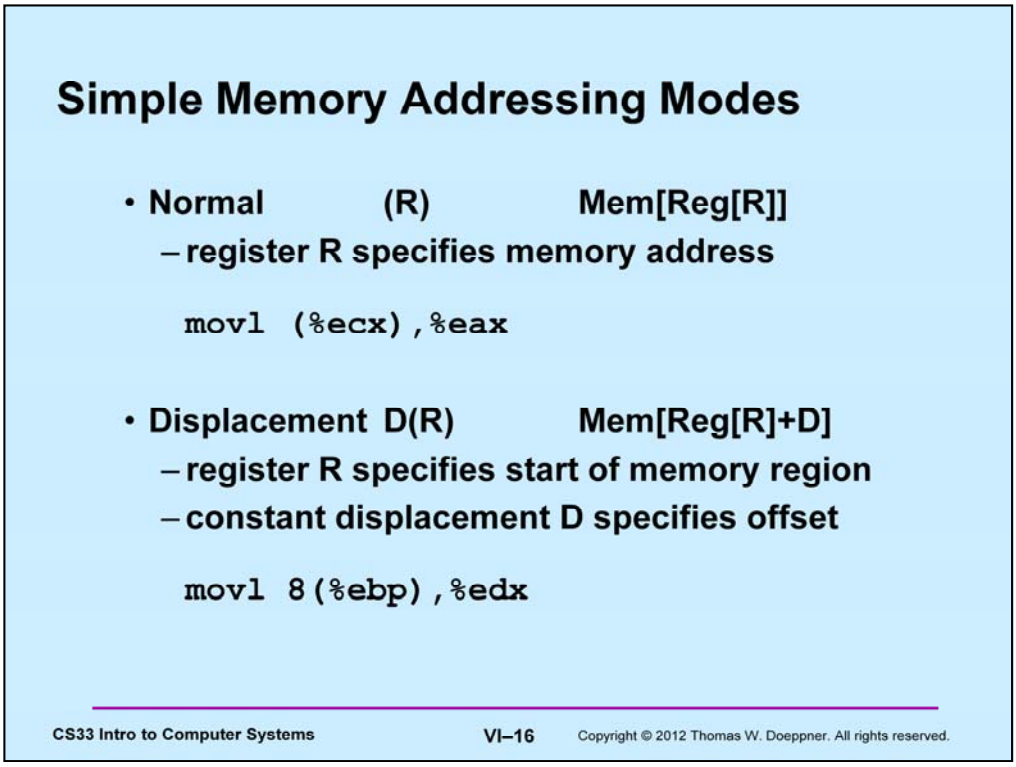
Note that though esp and ebp have special uses, they may also be used in both source and destination operands.

`movl` Operand Combinations

	Source	Dest	Src, Dest	C Analog
movl	Imm	Reg	`movl $0x4,%eax`	`temp = 0x4;`
		Mem	`movl $-147,(%eax)`	`*p = -147;`
	Reg	Reg	`movl %eax,%edx`	`temp2 = temp1;`
		Mem	`movl %eax,(%edx)`	`*p = temp;`
	Mem	Reg	`movl (%eax),%edx`	`temp = *p;`

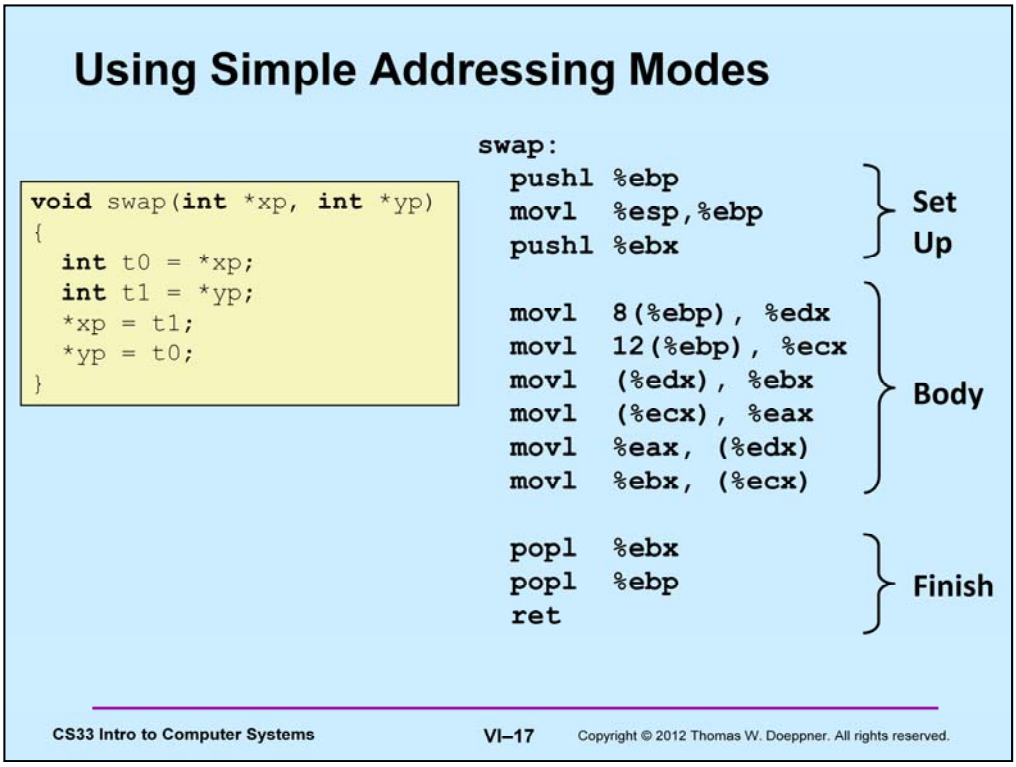
Cannot do memory-memory transfer with a single instruction

Supplied by CMU.

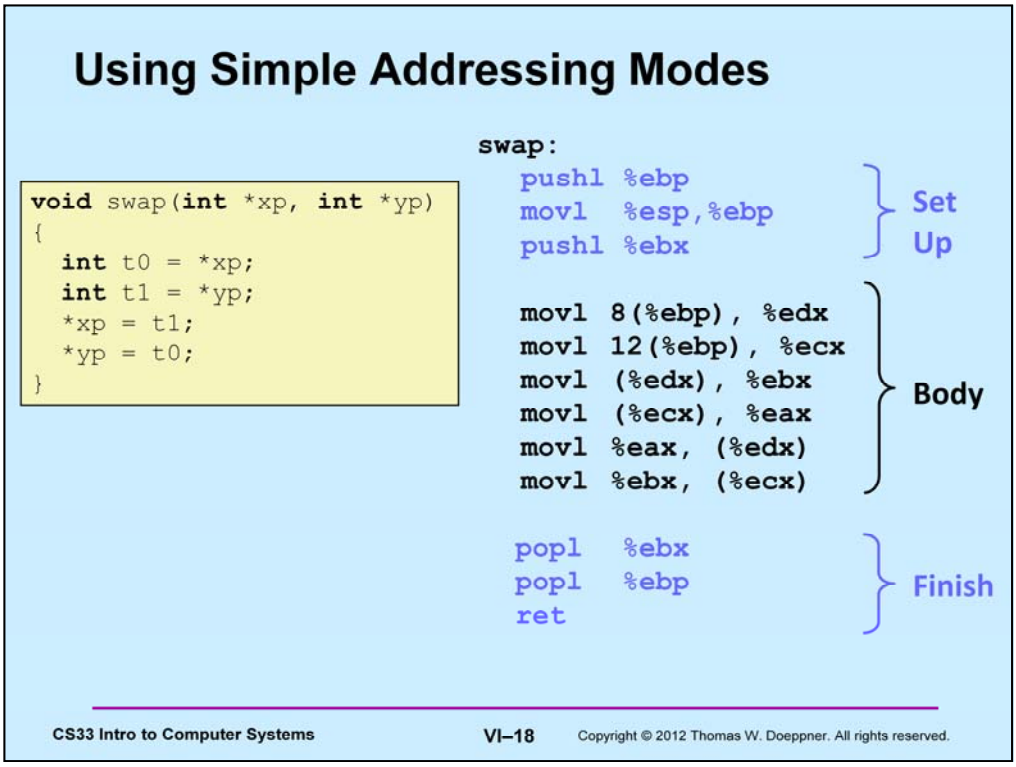


Supplied by CMU.

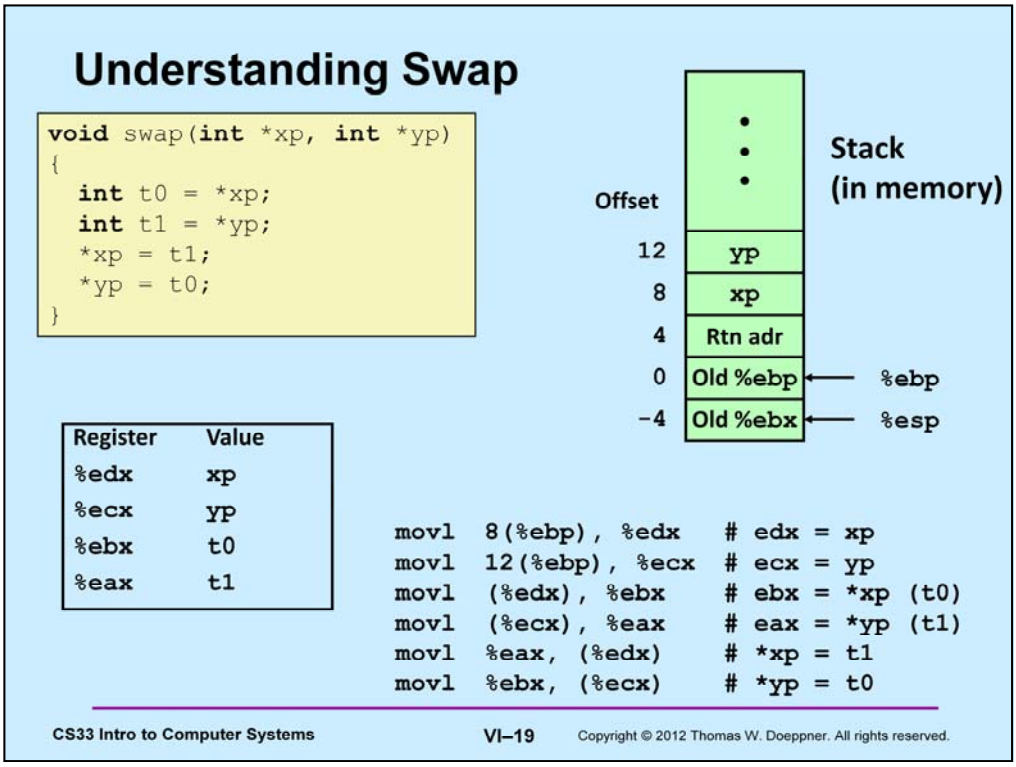
If one thinks of there being an array of registers, then “Reg[R]” selects register “R” from this array.



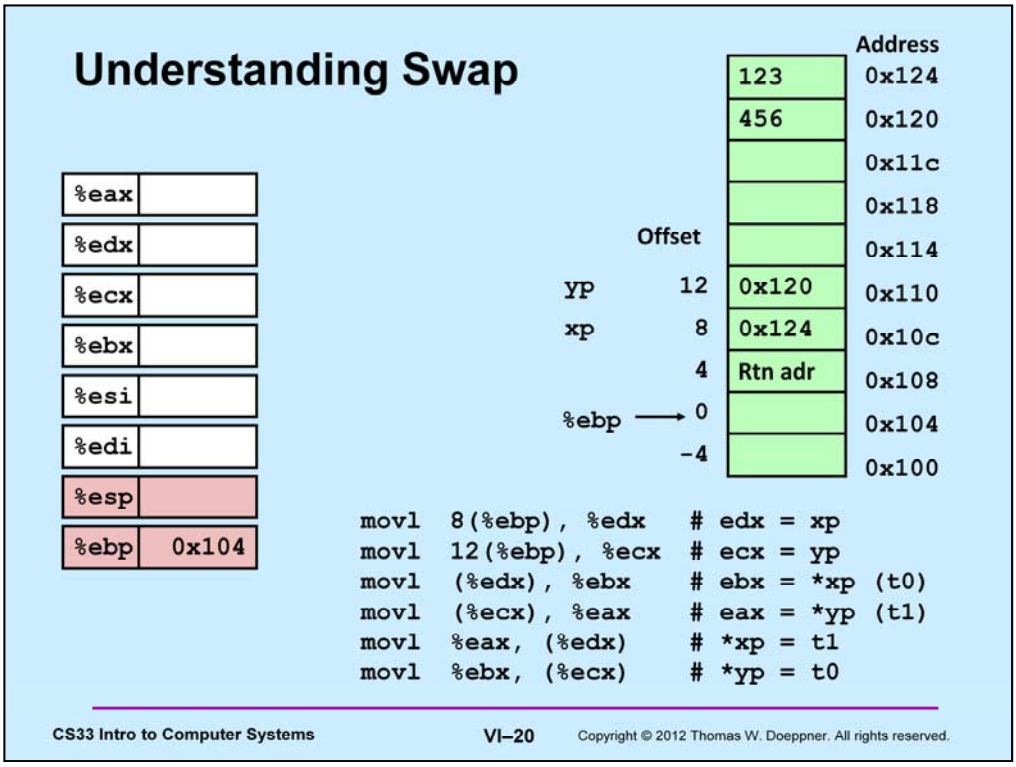
Supplied by CMU.



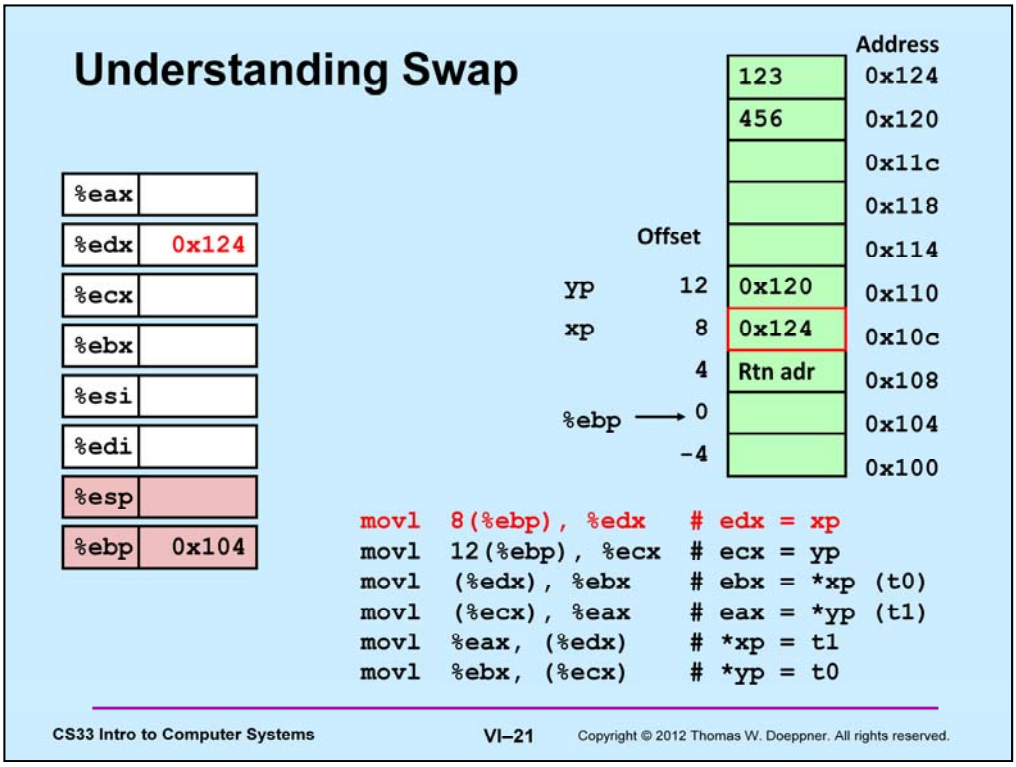
Supplied by CMU.



Supplied by CMU.

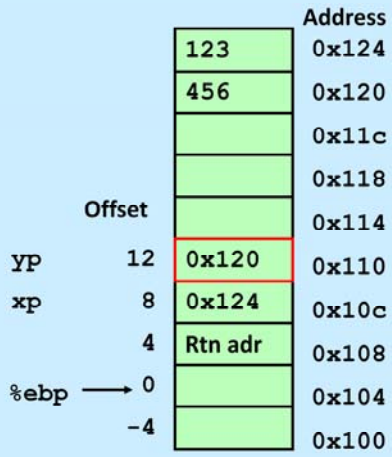


Supplied by CMU.

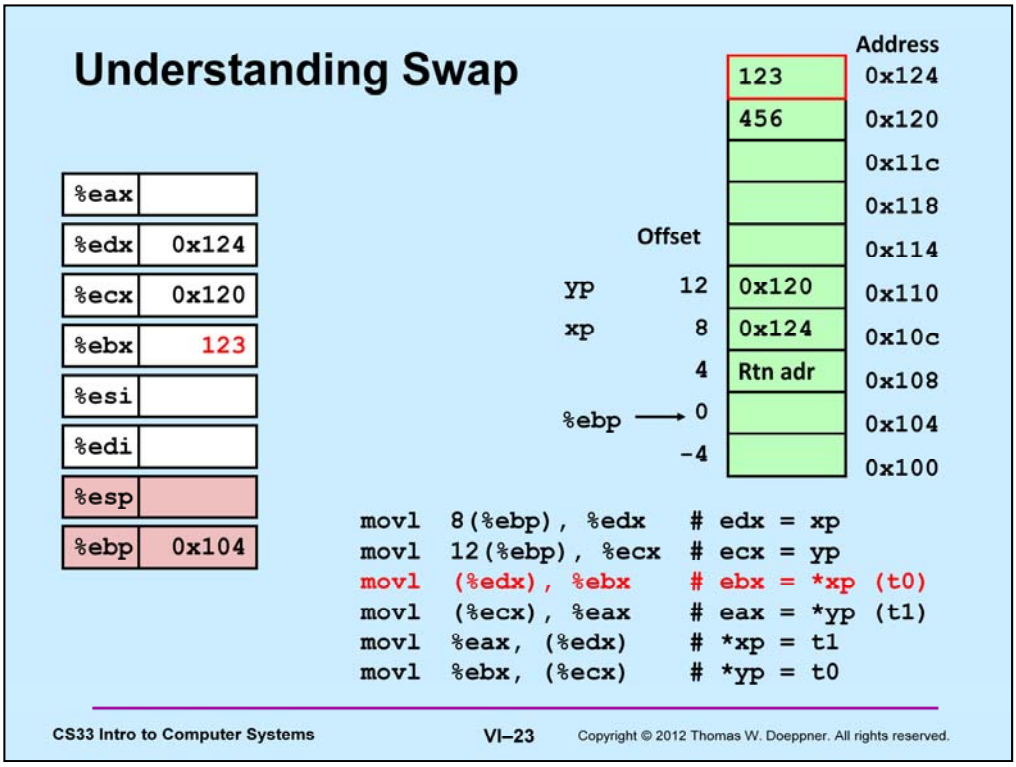


Supplied by CMU.

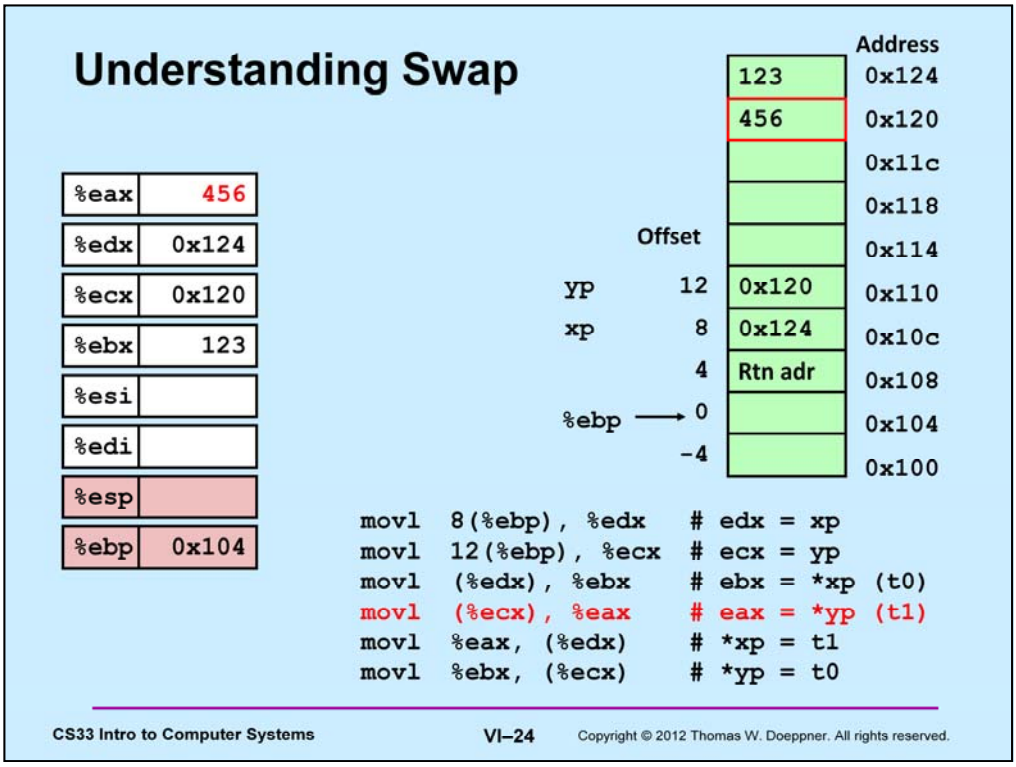
Understanding Swap



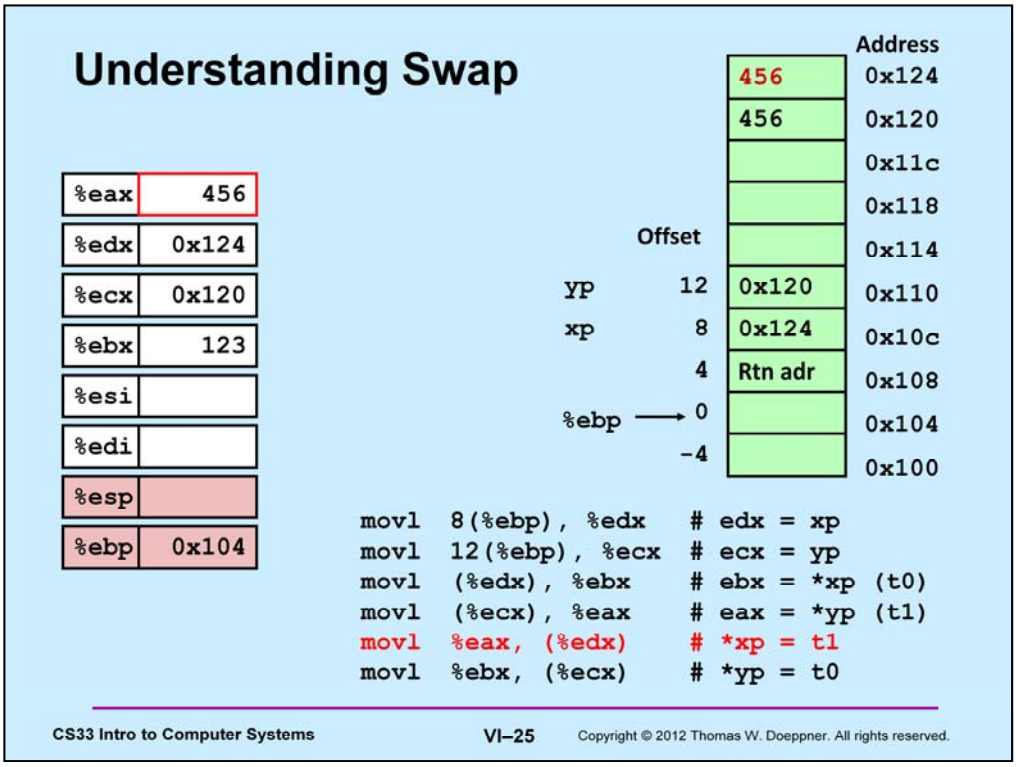

```
movl  8(%ebp), %edx    # edx = xp
movl  12(%ebp), %ecx   # ecx = yp
movl  (%edx), %ebx     # ebx = *xp (t0)
movl  (%ecx), %eax     # eax = *yp (t1)
movl  %eax, (%edx)     # *xp = t1
movl  %ebx, (%ecx)     # *yp = t0
```



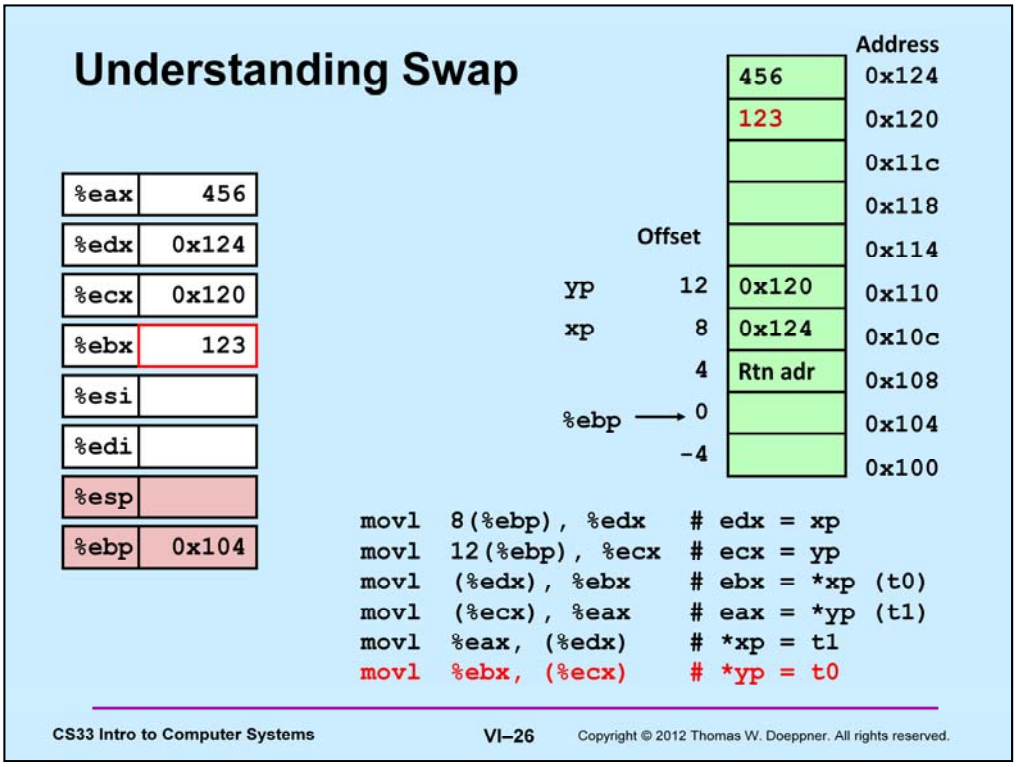
Supplied by CMU.



Supplied by CMU.



Supplied by CMU.



Supplied by CMU.

Complete Memory-Addressing Modes

- Most general form

  D(Rb,Ri,S) Mem[Reg[Rb]+S*Reg[Ri]+D]

  - D: constant "displacement"
  - Rb: base register: any of 8 integer registers
  - Ri: index register: any, except for `%esp`
    - » unlikely you'd use `%ebp` either
  - S: scale: 1, 2, 4, or 8

- Special cases

(Rb,Ri)	Mem[Reg[Rb]+Reg[Ri]]
D(Rb,Ri)	Mem[Reg[Rb]+Reg[Ri]+D]
(Rb,Ri,S)	Mem[Reg[Rb]+S*Reg[Ri]]

Supplied by CMU.

Address-Computation Examples

%edx	0xf000
%ecx	0x0100

Expression	Address Computation	Address
0x8(%edx)	0xf000 + 0x8	0xf008
(%edx,%ecx)	0xf000 + 0x0100	0xf100
(%edx,%ecx,4)	0xf000 + 4*0x0100	0xf400
0x80(,%edx,2)	2*0xf000 + 0x80	0x1e080

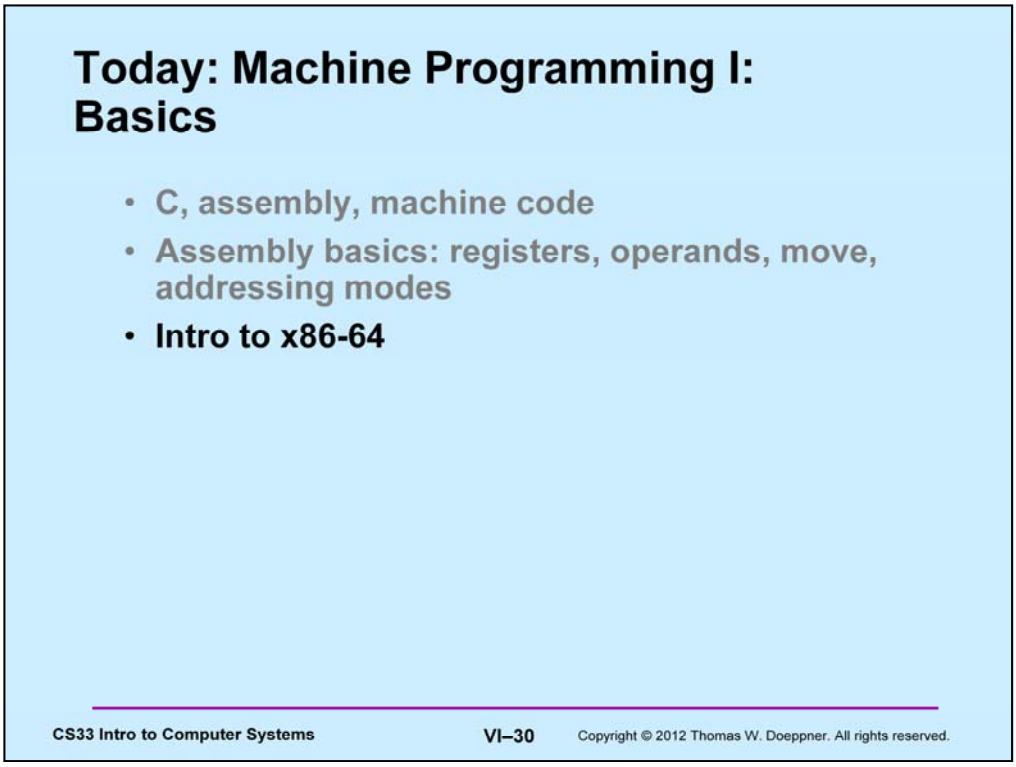
Address-Computation Instruction

- `leal` *src*, *dest*
  - *src* is address mode expression
  - set *dest* to address denoted by expression
- Uses
  - computing addresses without a memory reference
    - » e.g., translation of `p = &x[i];`
  - computing arithmetic expressions of the form x + k*y
    - » k = 1, 2, 4, or 8
- Example

```
int mul12(int x)
{
  return x*12;
}
```

Converted to ASM by compiler:

```
movl 8(%ebp), %eax          # get arg
leal (%eax,%eax,2), %eax    # t <- x+x*2
sall $2, %eax               # return t<<2
```



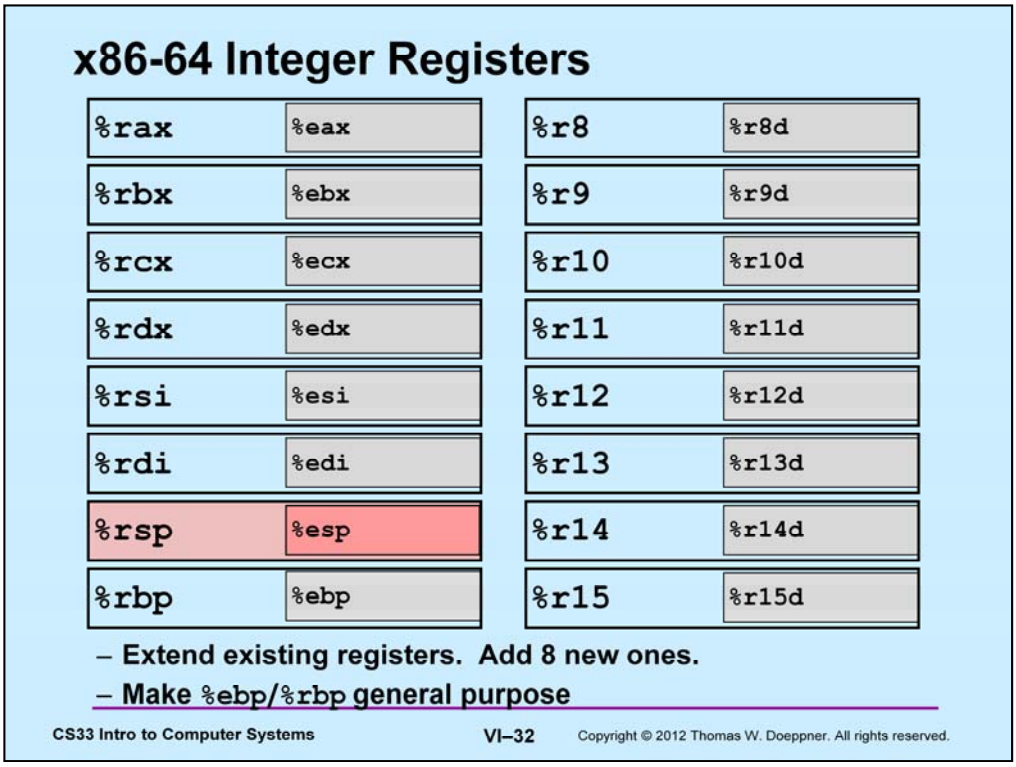
Supplied by CMU.

Data Representations: IA32 + x86-64

- Sizes of C objects (in bytes)

C Data Type	Generic 32-bit	Intel IA32	x86-64
unsigned	4	4	4
int	4	4	4
long int	4	4	8
char	1	1	1
short	2	2	2
float	4	4	4
double	8	8	8
long double	8	10/12	16
char *	4	4	8
or any other pointer			

Supplied by CMU.

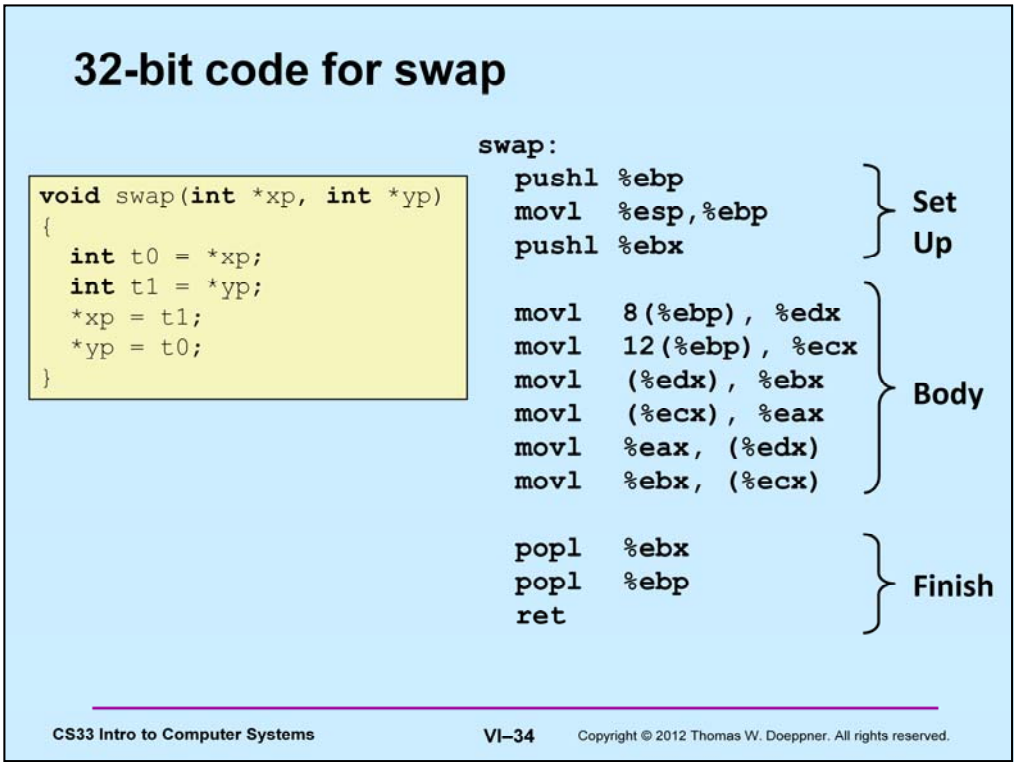


Supplied by CMU.

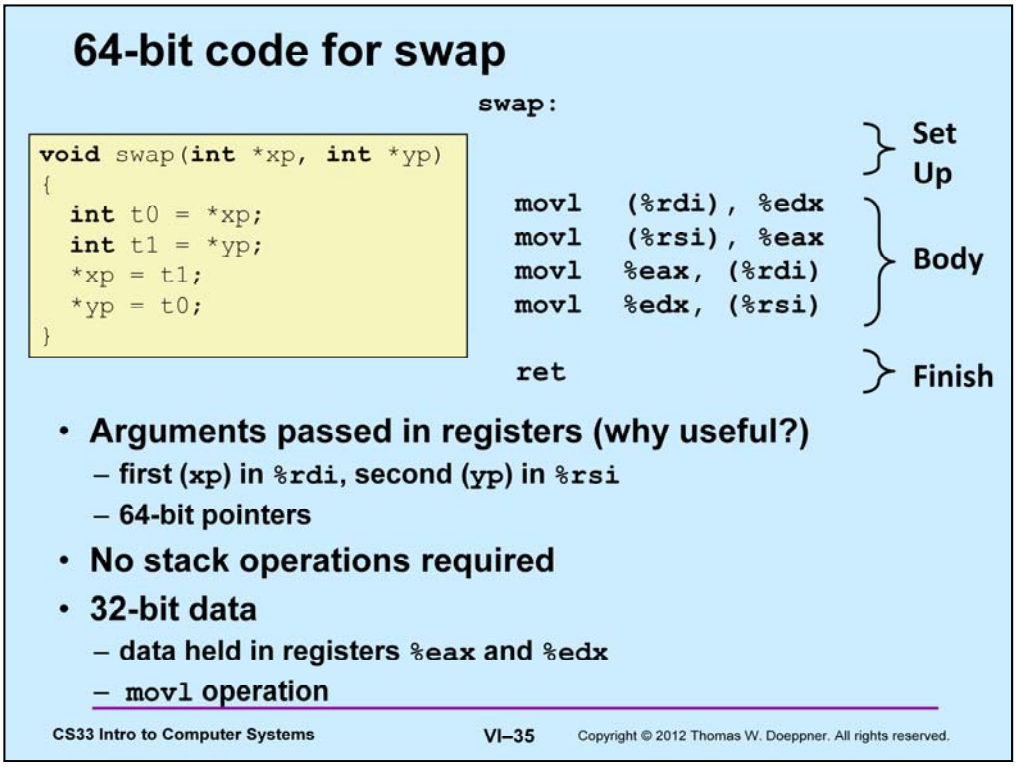
Instructions

- Long word `l` (4 Bytes) ↔ Quad word `q` (8 Bytes)
- New instructions:
  - `movl` → `movq`
  - `addl` → `addq`
  - `sall` → `salq`
  - etc.
- 32-bit instructions that generate 32-bit results
  - set higher order bits of destination register to `0`
  - example: `addl`

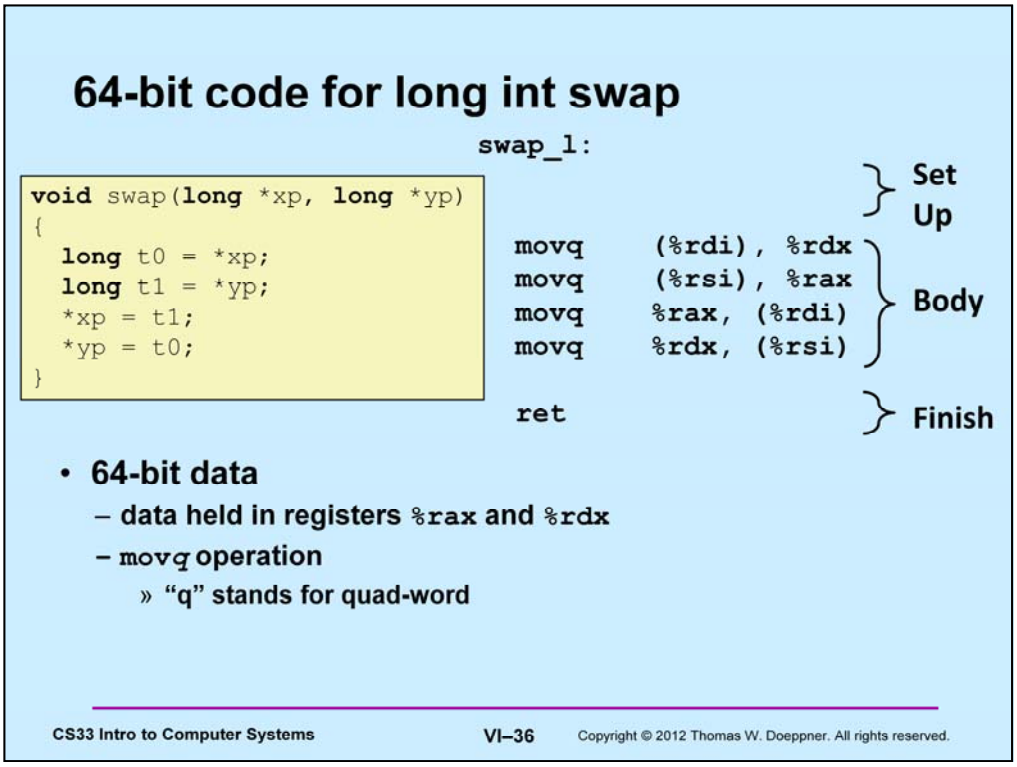
Supplied by CMU.



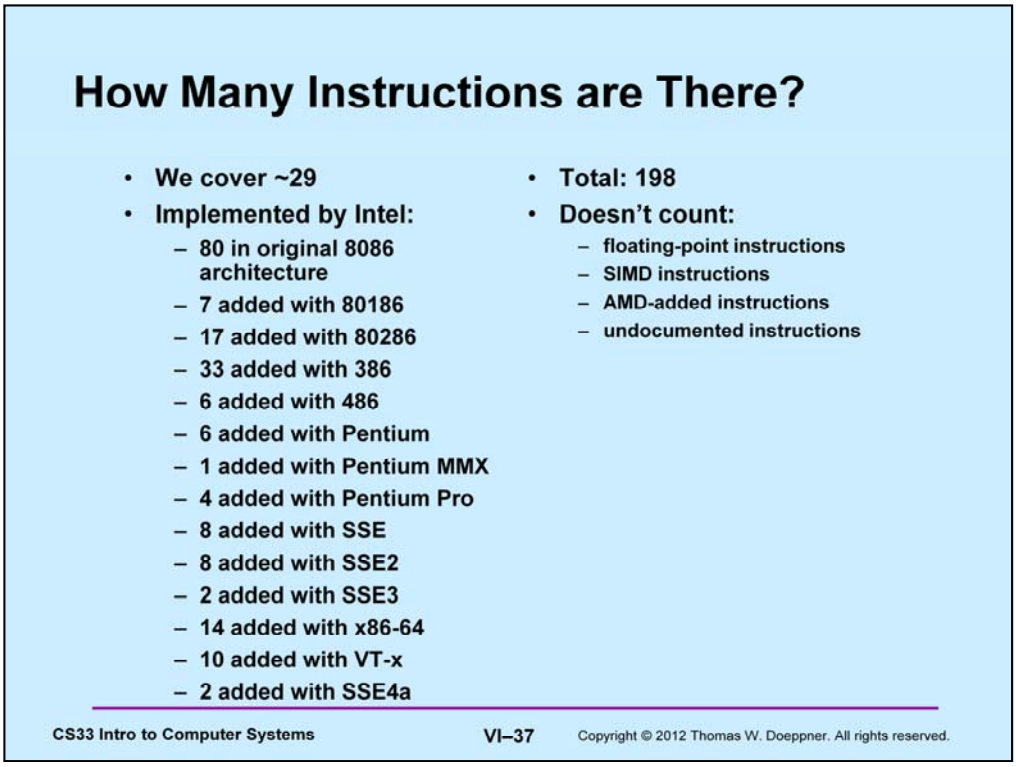
Supplied by CMU.



Supplied by CMU.



Supplied by CMU.



The source for this is http://en.wikipedia.org/wiki/X86_instruction_listings, viewed on 9/18/2012, and also depends upon my ability to count.