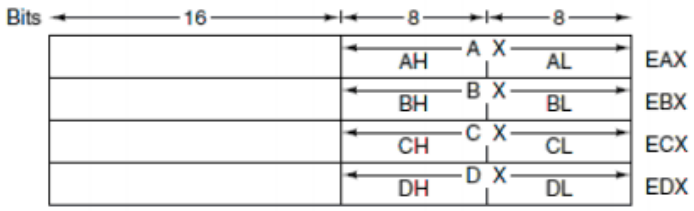
Ch 5 - Instruction set architecture

5.1 Overview

Core i7 - CISC machine

From p 347 - The reference document for Intel's Core i7 architecture "weighs in at 4,161 pages, reminding us once again that the Core i7 is a *complex* instruction set computer."

Core i7 registers...`eax, ebx, ecx, edx, ebp, esp, esi, edi`



OMAP - RISC machine

32 bit machine, ARM architecture, load/store architecture
16 general-purpose registers, R0-R15:

Register	Alt. name	Function
R0–R3	A1–A4	Holds parameters to the procedure being called
R4–R11	V1–V8	Holds local variables for the current procedure
R12	IP	Intraprocedure call register (for 32-bit calls)
R13	SP	Stack pointer
R14	LR	Link register (return address for current function)
R15	PC	Program counter

Separate bank of floating point registers: 32 32-bit single precision or 16 64-bit double precision
Quiz - How are params and local vars handled differently than the Intel arch? Related… why is this called a load/store architecture?

ATmega168 - microcontroller

Tiny - 8-bit machine, 16KB of program memory, 1KB data memory,
32 general-purpose registers: R0-R31, values mirrored in memory (weird)

5.2 Data types

Why does Intel Core i7 have instructions for 8 bit, 16 bit, 32 bit, and 64 bit integers?

5.3 Instruction formats

Definitions: expanding opcode

5.4 Addressing

Modes: immediate, direct, register, register indirect, indexed, based-indexed, stack

5.5 Instruction types

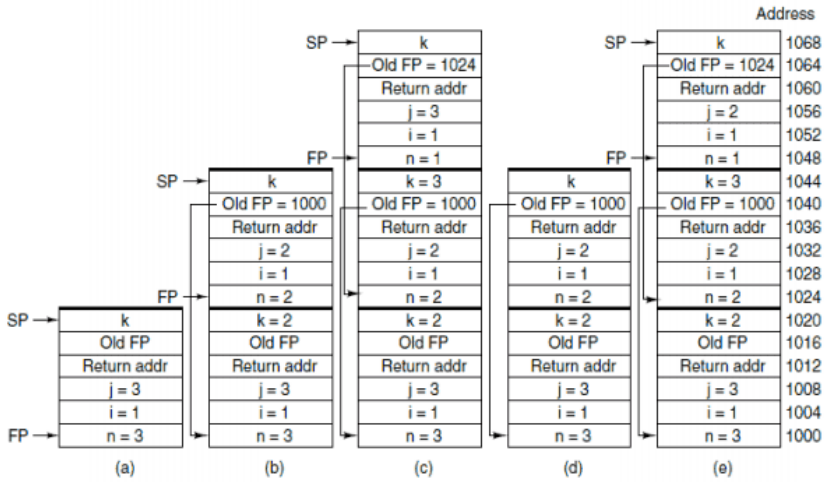
Types: Data movement, dyadic, monadic, comparison, conditional branch, call, loop, input/output

Page 404 - “The Core i7 design was determined by three major factors: 1. Backward compatibility, 2. Backward compatibility, 3. Backward compatibility.”

5.6 Flow of control

Definitions: coroutine, trap, trap handler, interrupt, interrupt vector

Follow the bouncing frame pointer!



5.7 Towers of Hanoi example

Page 419, Figure 5-45, Towers of Hanoi for Core i7 - some note:

- Follow along with the Java for this code: page 409, Figure 5-39
- Good - comments at side, uses local var and param names in comments
- Bad - need space between related chunks of code, like each function call
- See function guards at beginning (push ebp, mov ebp, esp) and end (leave, ret)
- See `eax` is used as a tmp register
- See printf call. See its format string at the bottom.
- See cleanup after each function call, for example after printf: `ADD ESP,12`

Author’s 3 strikes against Intel architecture:

1. Intel breaks its complex instructions down into RISC-size chunks, but that still takes time
2. Memory-oriented ISA slows down the clock
3. Small and irregular register set

And more railing… “A huge fraction of all the transistors on the Core i7 are devoted to decomposing CISC instructions...”

Bonus quiz - What software company has this backward compatibility problem?