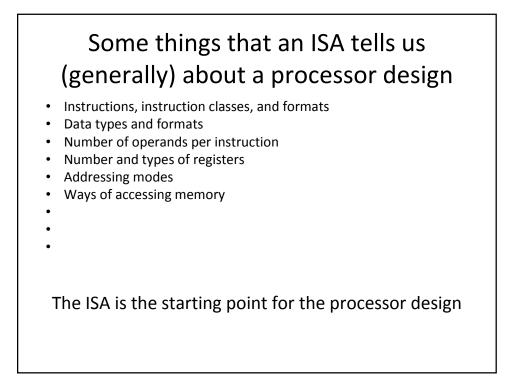
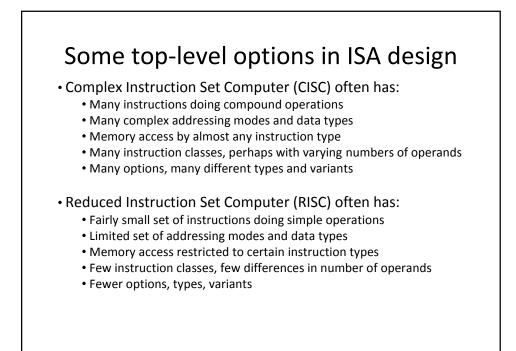


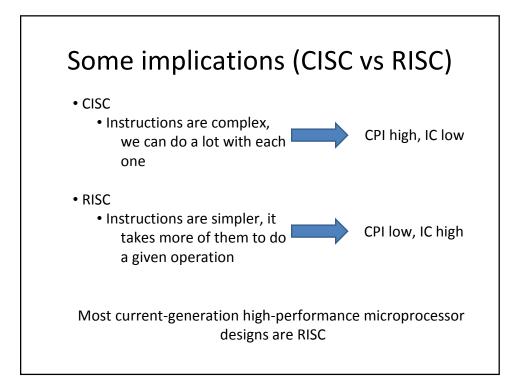
ISA Defined

- Patterson and Hennessy:
 - "interface between the hardware and the lowest level software"
 - "includes anything programmers need to know to make a binary machine language program work correctly, including instructions, I/O devices, and son on."
 - "enables many implementations of varying costs and performances to run identical software"



"x-Address" Machines		
x	to do "a=b+c"	
3	ADD a,b,c	"Natural" for most arithmetic ops
2	COPY a, c ADD a,b	Smaller instructions, but we need more
1	COPY1 c ADD b COPY2 a	Need still more instructions Pretty much un-natural
0	PUSH b PUSH c ADD POP a	Still more instructions Very un-natural
		1





Some things to think about when we're designing (1)

"Simplicity favors regularity" (P & H Principle 1)

- The more variations in instructions (formats, ...), the more logic it takes to identify which variation we have in a particular instruction.
- The more regular the instruction set, the less time we have to spend decoding the instruction type

Some things to think about when we're designing (2)

"Smaller is faster" (P&H Principle 2)

- As we add more and more logic to a design, max speed tends to drop due to:
 - More "things" to select from (e.g, more registers)
 - More logic levels needed to decode identifiers
 - Wider instructions to specify more units
 - Longer path lengths needed
 - Signal propagation increases

Some things to think about when we're designing (3)

"Make the common case fast" (P&H Principle 3)

- Based on Amdahl's Law (Chapter 1)
- Speeding up things you do often gives greater payoff than speeding up things you do do infrequently.

Some things to think about when we're designing (4)

- "Good design demands good compromises" (P&H Principle 4)
 - Much of the time, an improvement in one area compromises another (ex: Adding registers makes programmers happy, but may slow the processor down)
 - The best designs exhibit a balance of features

Some things to think about when we're designing (5)

- Memory access is much slower than register access
 - Register access typically 1 clock cycle
 - RAM access may be dozens or hundreds of clock cycles