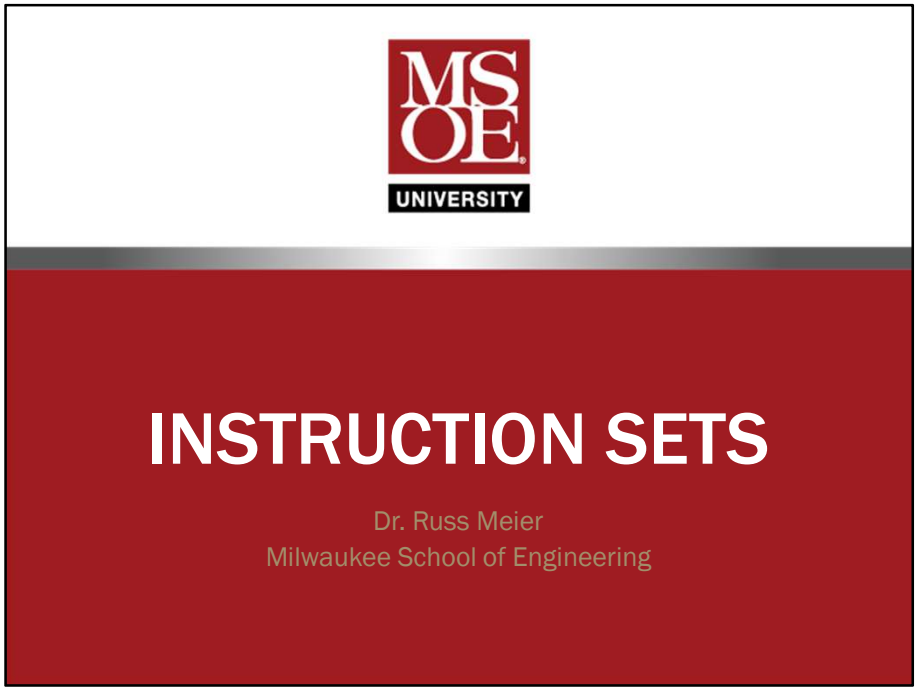
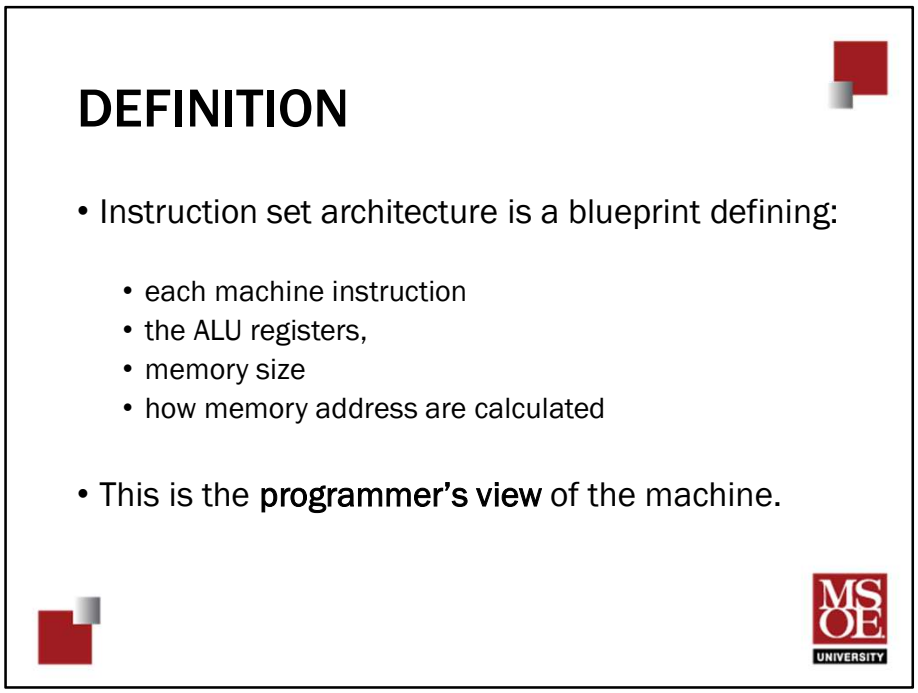
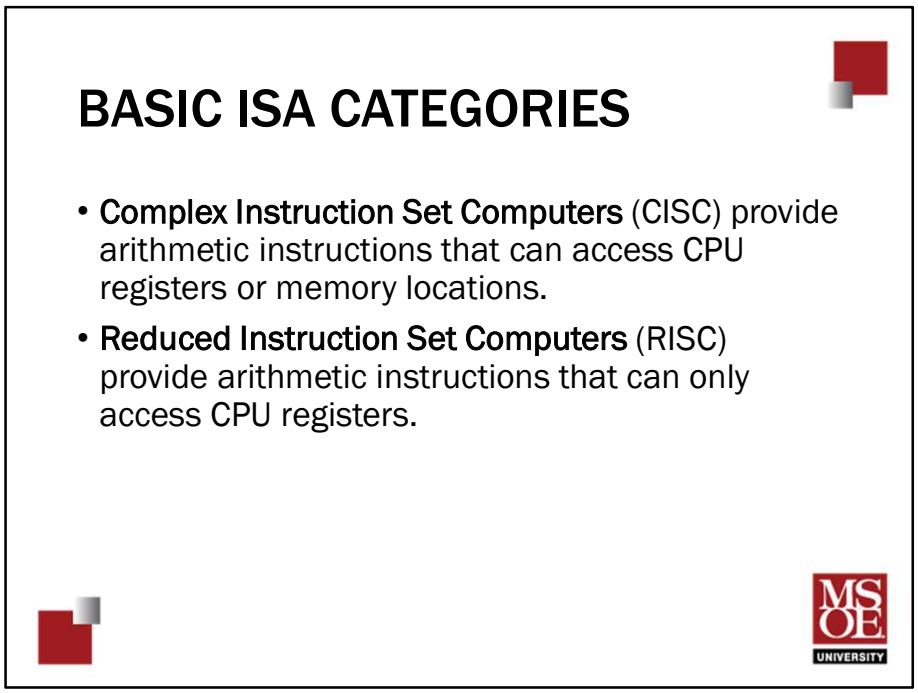

MSOE
UNIVERSITY
INSTRUCTION SETS
Dr. Russ Meier
Milwaukee School of Engineering



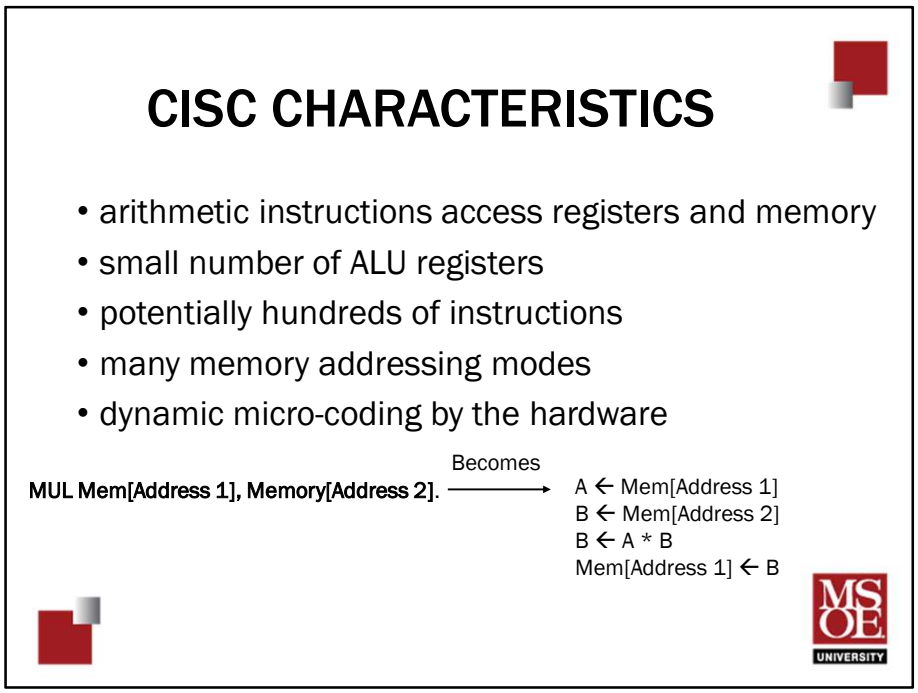
Instruction set architecture is a blueprint that begins the design of any computer. The instruction set architecture defines the machine the way the programmer would see it and not the way a circuit designer would see it.

- The instruction set architecture **uses** tables, lists, words, and binary numbers to document what every instruction looks like.
- The instruction set architecture **defines** what CPU registers are available to store data before it is moved to larger memories.
- The instruction set architecture **may suggest** how programmers group registers into sets and use them when programming the machine.
- The instruction set architecture describes abstractly how big memory is and how memory is accessed by instructions.

This programmer's view of the machine is often called the **programmer's model**. It is this programmer's model that must then be implemented by circuit designers.



Instruction set architectures are placed into two categories called CISC and RISC. The categorization is done based on the arithmetic instructions. The instruction set is called complex if arithmetic instructions can access both CPU registers and main memory. It is called **reduced** if arithmetic instructions can access only CPU registers.

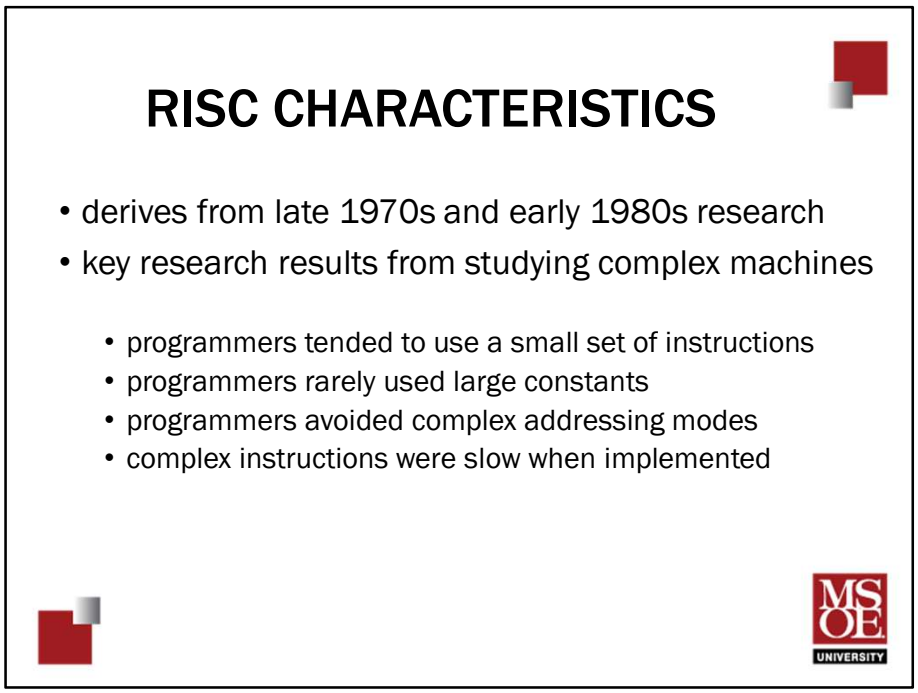


Historically, memory was expensive and thus more limited than what we have today. The total set of binary numbers that form a program stored in instruction memory is called the **instruction memory image**. Complex instructions kept the instruction memory image small at a time when memory was limited. By the late 1970s, most microprocessors implemented complex instruction sets. **CISC machines have these general characteristics**:

- The most important identifying characteristic is arithmetic instructions that can access both ALU registers and memory.
- Another characteristic is a small number of ALU registers – generally less than 20 and sometime far fewer.
- CISC machines usually provided hundreds of instructions. The original x86 ISA had just over 80 instructions and current versions of the x86-64 ISA have over 1500 instructions. The Motorola 68000 ISA had just over 100 instructions.
- Another characteristic is how programmers specify memory locations as addresses. CISC machines provide many **addressing modes**. Lots of addressing modes adds complexity to address calculation which leads to delay.
- A final characteristic is dynamic code generation by the hardware. Because the complex instruction specified complex behavior, the circuitry had to automatically rewrite the instruction into a set of simpler steps. This is called **micro-coding**. Micro-coding adds complexity to the circuit design. An **example** of a micro-coding is shown as the **MUL**

Mem[Address 1], Memory[Address 2] is converted automatically by the circuit into four simpler instructions using ALU registers A and B and requiring at least four clock periods.

Remember the architectural rule of thumb: **simple → fast**. Because CISC machines accessed slower main memory, had to identify and microcode hundreds of instructions, and calculate memory addresses in multiple ways, circuit delay was significant.



By the late 1970s, memory densities were beginning to rise as the industry kept pace with Moore's Law. Several key researchers in computer architecture began to ask if it made sense to provide so many instructions and addressing modes. Working independently, these researchers discovered the key points noted in this slide.

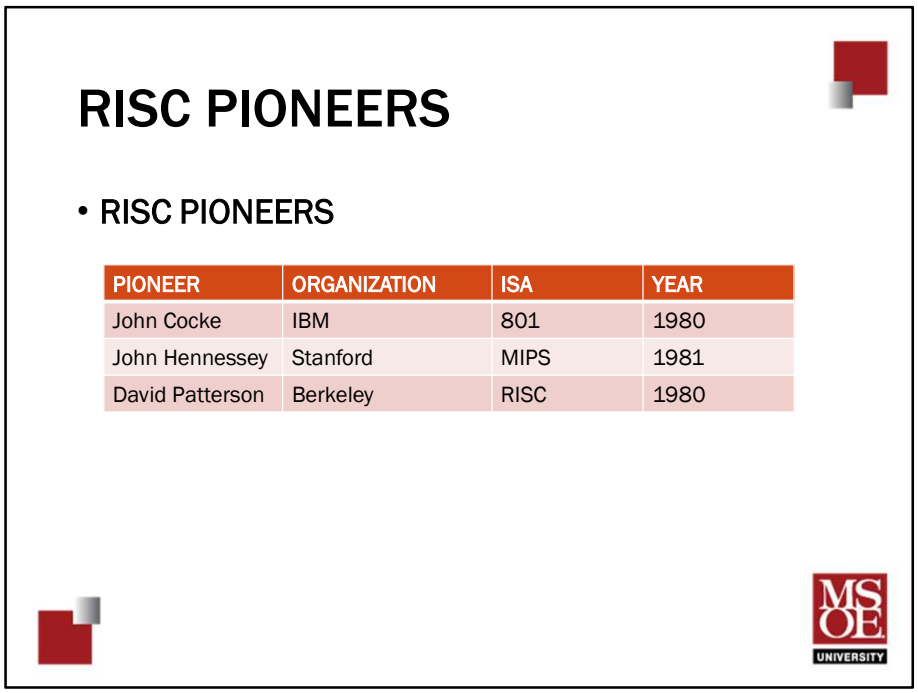
- Assembly language programmers tended to use a smaller set of faster basic instructions.
- Assembly programmers rarely used large constants in for-loops or comparisons. Sometimes, yes, but on the average constant numbers tended to be smaller.
- Many of the complex addressing modes confused assembly language programmers and they didn't use them.
- Complex circuitry had significant delay and didn't scale well to increased clock speeds.

These researchers proposed a change in the way machine design should progress as companies moved forward. This is known as the RISC movement.

- A team of people at IBM (led by John Cocke) investigated reducing the complexity of the instruction set. This led to the IBM 801 in 1980. This was not well publicized as it was an internal company project.
- An academic team at Stanford (led by John Hennessey) began a project in 1981 that led to the founding in 1984 of MIPS Computer Systems – a fabless company that designed

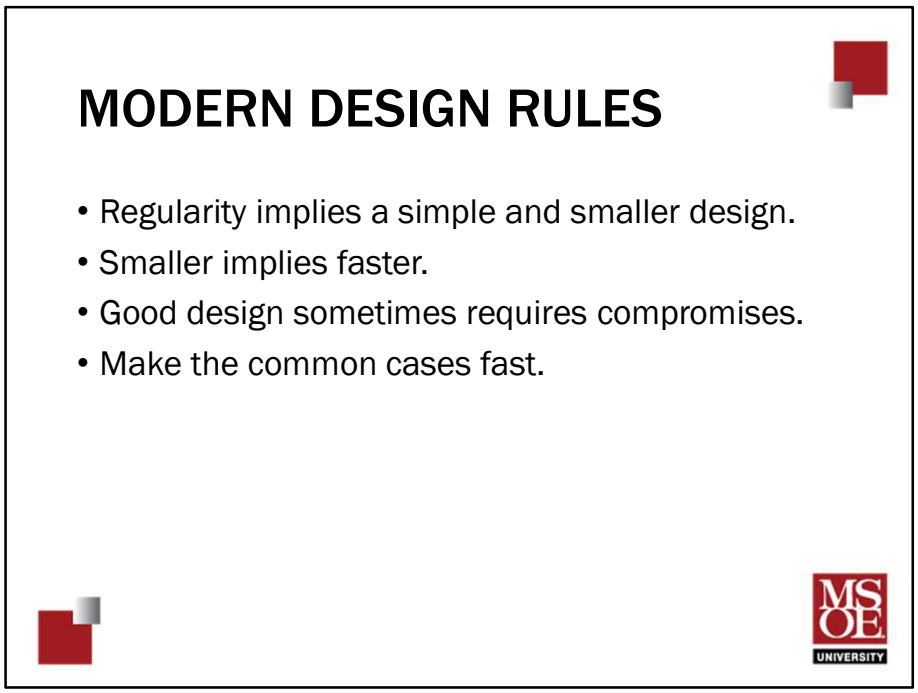
the MIPS ISA and microarchitectures it licensed to other manufacturers.

- An academic team at the University of California Berkeley (led by David Patterson) began a project in 1980. David Patterson defined the acronym RISC. His project results in the the RISC 1, which is later commercialized by Sun Microsystems as the SPARC ISA and microarchitectures.
- Cocke, Hennessey and Patterson are now considered the pioneers that encouraged the industry to rethink design.

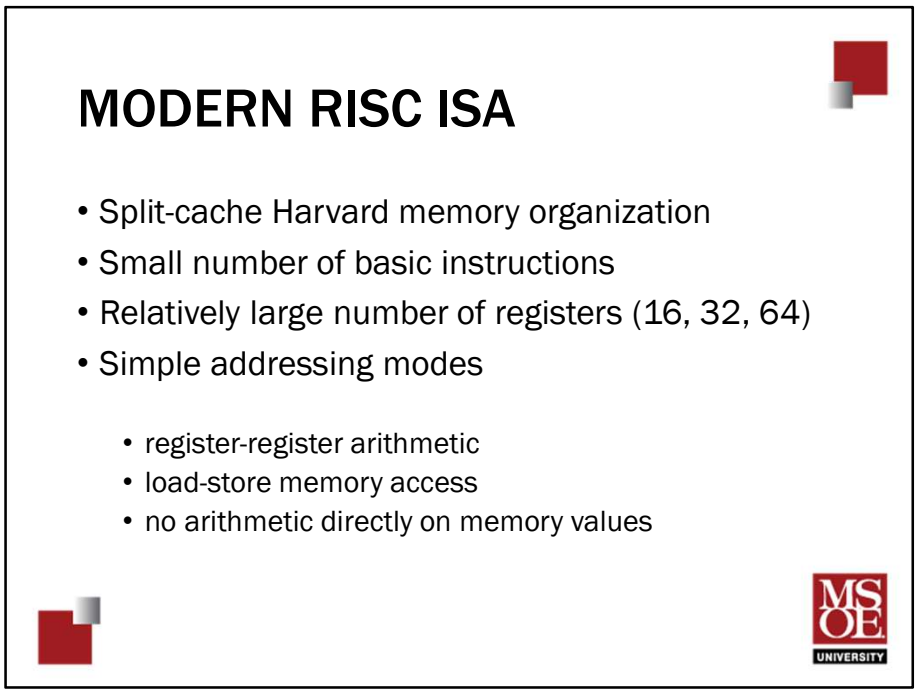


These researchers proposed a change in the way machine design should progress as companies moved forward. This is known as the RISC movement.

- A team of people at IBM (led by John Cocke) investigated reducing the complexity of the instruction set. This led to the IBM 801 in 1980. This was not well publicized as it was an internal company project.
- An academic team at Stanford (led by John Hennessey) began a project in 1981 that led to the founding in 1984 of MIPS Computer Systems – a fabless company that designed the MIPS ISA and microarchitectures it licensed to other manufacturers.
- An academic team at the University of California Berkeley (led by David Patterson) began a project in 1980. David Patterson defined the acronym RISC. His project results in the the RISC 1, which is later commercialized by Sun Microsystems as the SPARC ISA and microarchitectures.
- Cocke, Hennessey and Patterson are now considered the pioneers that encouraged the industry to rethink design.



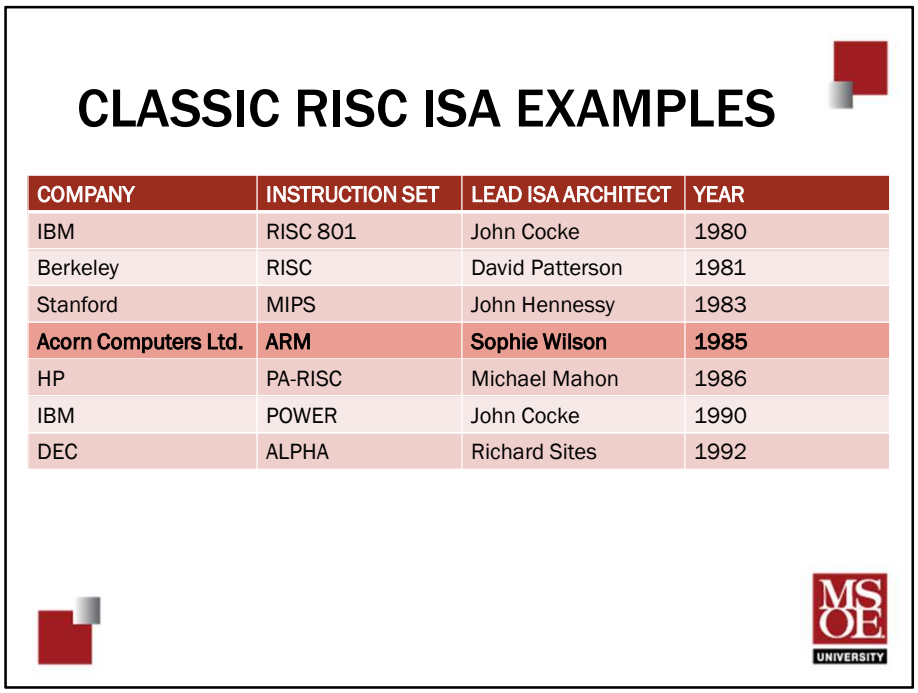
The work of the RISC pioneers led to the industry adopting four philosophical design rules that Patterson and Hennessey promote well in their textbooks on Computer Architecture. These rules are stated on this slide. Today, most modern processors are RISC designs built using these rules.



Like CISC machines, there are now **identifiable characteristics** that have evolved into the modern RISC instruction set architectures.

- A modified split-cache Harvard memory organization provides on-chip instruction and data caches. This allows the processor to avoid the memory bottleneck and achieve higher clock rates.
- A much smaller number of basic instructions are implemented in the circuitry.
- A larger number of registers are provided to the assembly language programmer.
- Simple addressing modes reduce the complexity of address calculation circuitry and increase speed.

    - All data in memory must be **loaded** into a register before used by the ALU. Any result from the ALU is put back in a register. It must then be **stored** to memory for long-term storage if needed. Arithmetic instructions **never** directly put values into memory. This is called the **load-store principle** or **load-store memory access**. Load-store memory access is the defining characteristic of a RISC machine.



This table provides some history RISC architectures for your exploration. There are many more, of course, but these have played important roles in computer history or are interesting architecturally because of unique features. We've already mentioned that Hennessey and Patterson both move their ISAs into successful commercial implementations in the form of MIPS microprocessors and SPARC microprocessors. So, let's comment a bit on the other ones.

- The IBM 801 project leads IBM to work with other companies in the Power PC consortium to create the POWER ISA. This ISA is implemented as microarchitecture in the PowerPC chips – and become the chip at the heart of the Macintoshes of the 1990s.
- The DEC Alpha ISA is one of the first 64-bit ISAs – and one of my personal favorites. Digital Equipment Corporation attempted to design a chip that would support its already popular operating system (VMS) from its earlier generation of CISC processors.
- The HP-PA RISC competed against SPARC in engineering workstations of the late 1980s and 1990s. Hewlett-Packard's experience with HP-PA RISC allows it to become a successful partner with Intel to develop the Itanium ISA (IA64) that Intel used in server-side machines through the first decades of the 21st century.

Finally, the ARM architecture, designed by Sophie Wilson, has become the most-used architecture in the world because of its widespread use in the cell-phone and embedded

systems market. Billions of ARM microchips are fabricated each year. ARM is a fabless company – designing the ISA and microarchitectures that is licenses to other companies to use.

KEY POINTS

- Instruction set architecture blueprints the programmer's model.
- Memory size once required complex instructions to ensure programs could be stored.
- CISC arithmetic instructions can directly access memory.
- RISC machines implement a load-store memory access model
- The RISC movement of the 1980s changed machine design.
- Modern processors are load-store and thus RISC machines.

This summary slide notes the key points of this presentation. Continue to review this video as needed. I also encourage you to explore computer history by reading about the people and ISAs noted in the presentation.