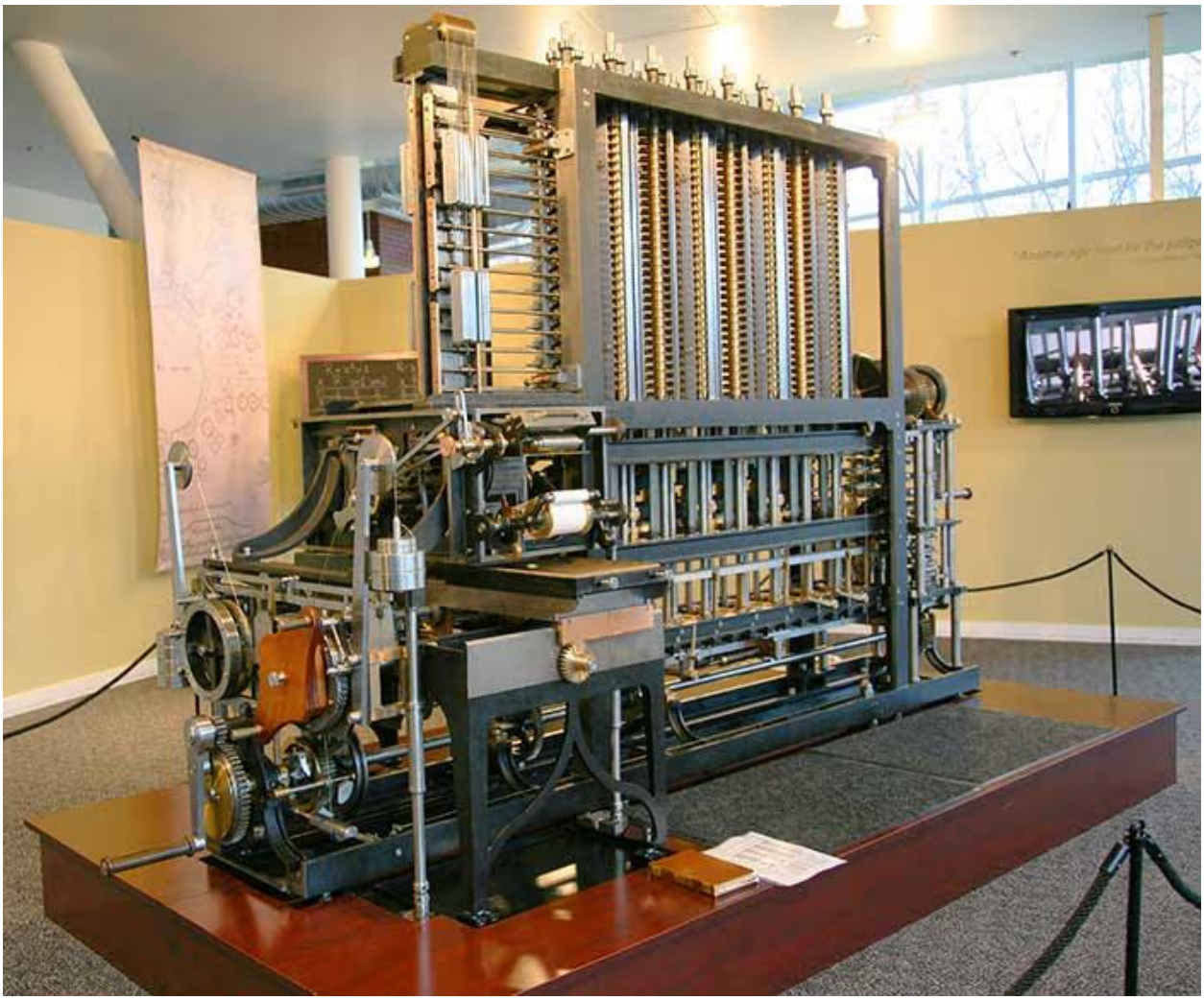
1. Fundamentals of Computers Architecture

The first practical computers were developed in 1950s and 60s along with the invention of semiconductors. By 1990s, by the help of hardware company IBM and software company Microsoft, computing started becoming more personal. The very first computers were bulky, huge, and power hungry machines.



Courtesy of Jitze Couperus. Copyright: CC-Att-SA-2 (Creative Commons Attribution-ShareAlike 2.0 Unported).

The first computer was invented by Charles Babbage (1791-1871) but couldn't built it. 153 years after the design the first complete Babbage Engine was completed in London in 2002. Charles Babbage's:

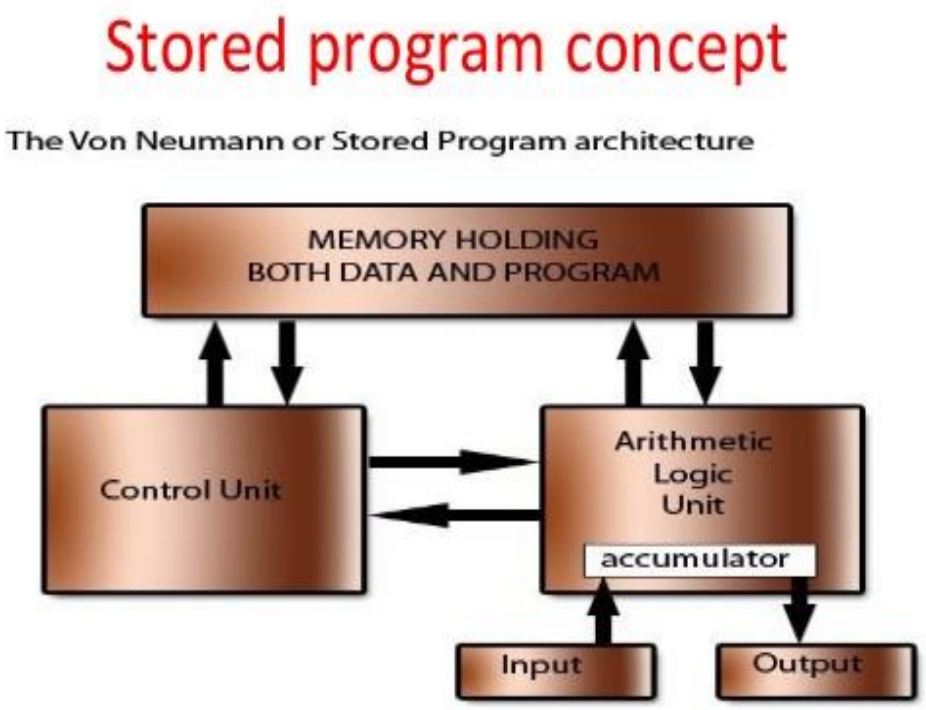
• Difference Engine in 1823

• Analytic Engine in 1833

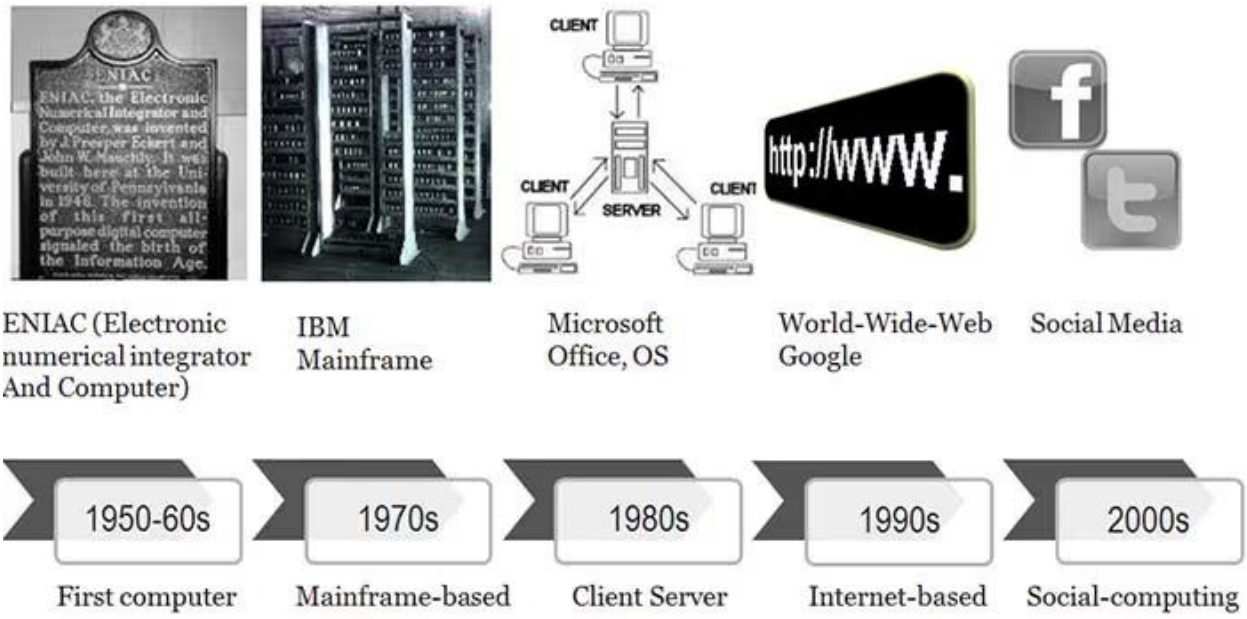
The first computers run on punch cards,

- The set of cards with fixed punched holes dictated the pattern of weave ⇒ program
- The same set of cards could be used with different colored threads ⇒ numbers

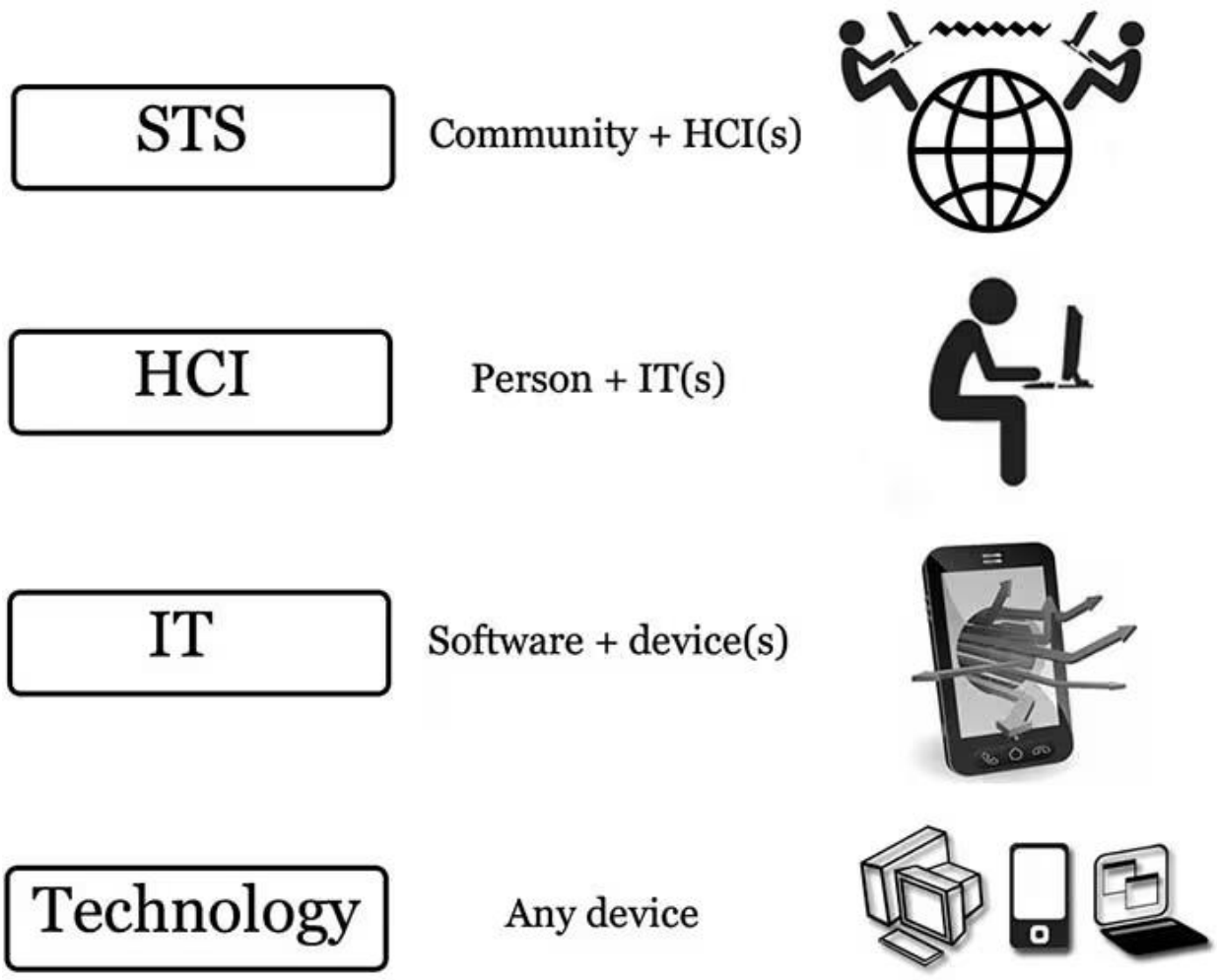
But later on along with Von Neumann architecture and a concept popularly known as “the stored program concept” changed how we process instructions. The stored program concept introduced how we can store program data and instructions in same memory. Computers that stored program data and instructions in same memory is called stored program computer.



Along with stored program concept, there were significant advancements in computer architectures, like pipelining, parallelism, etc. From vacuum tubes to transistors, computers are becoming more complex, fast, and highly efficient.



Courtesy of Brian Whitworth and Adnan Ahmad. Copyright: CC-Att-SA-3 (Creative Common Attribution-ShareAlike 3.0). The computing evolution



Courtesy of Brian Whitworth and Adnan Ahmad. Copyright: CC-Att-SA-3 (Creative Commons Attribution-ShareAlike 3.0). Computer system levels

Size of microprocessor – 16 bit

Name	Year Of Invention	Clock Speed	Number Of Transistors	Inst. Per Sec
8086	1978 (multiply and divide instruction, 16 bit data bus and 20 bit address bus)	4.77 MHz, 8MHz, 10MHz	29000	2.5 Million
8088	1979 (cheaper version of 8086 and 8 bit external bus)			2.5 Million
80186/ 80188	1982 (80188 cheaper version of 80186, and addtional components like interuppt controller, clock generator, local bus controller,counters)	6 MHz		
80286	1982 (data bus 16bit and address bus 24 bit)	8 MHz	134000	4 Million

Size of microprocessor – 32 bit

Name	Year Of Invention	Clock Speed	Number Of Transistors	Inst. Per Sec
INTEL 80386	1986 (other versions 80386DX, 80386SX, 80386SL and data bus 32 bit address bus 32 bit)	16 MHz – 33 MHz	275000	
INTEL 80486	1986 (other versions 80486DX, 80486SX, 80486DX2, 80486DX4)	16 MHz – 100 MHz	1.2 Million transistors	8 KB of cache memory
PENTIUM	1993	66 MHz		Cache memory 8 bit for instructions 8 bit for data

Size of microprocessor – 64 bit

Name	Year Of Invention	Clock Speed	Number Of Transistors	Inst. Per Sec
INTEL core 2	2006 (other versions core2 duo, core2 quad, core2 extreme)	1.2 GHz to 3 GHz	291 Million transistors	64 KB of L1 cache per core 4 MB of L2 cache
i3, i5, i7	2007, 2009, 2010	2.2GHz – 3.3GHz, 2.4GHz – 3.6GHz, 2.93GHz – 3.33GHz		

Instruction set Architecture

Instruction set architecture is a crucial aspect of the design and operation computers. It defines the instruction set that the computer needs to execute by the processor. It serves as the interface between software and hardware, that enables the programs to communicate effectively with the underlying hardware components.

Key Characteristics of ISA

1. **Instruction Set:** An extensive collection of operations that a processor can execute, including arithmetic, logical, control, and memory tasks.
2. **Registers:** Quick, small storage areas within the CPU utilized for temporarily holding data and instructions.
3. **Addressing Modes:** Methods for identifying the operands of instructions, enabling versatile data manipulation.
4. **Data Types:** Specifies the kinds of data that the processor can process, such as integers, floating-point values, and characters.

Key advancement is Instruction set Architecture

1. Complex Instruction Set Computing (CISC) (Early ISA Approach – 1970s-1980s)

- Goal: Reduce the number of instructions in a program by making each instruction do more.
- Key Features:
  1. Multi-step instructions (e.g., "MULT" performs load, multiply, and store in one step).
  2. Larger instruction set, variable-length instructions.
  3. Easier for programmers but required complex decoding hardware.
- Example Architectures: x86 (Intel, AMD), VAX

2. Reduced Instruction Set Computing (RISC)

(Alternative to CISC – 1980s-Present)

- Goal: Optimize execution speed by using simple, fast instructions that take one clock cycle.
- Key Features:
  a. Fixed-length, simple instructions (easier to pipeline).
  b. Load/Store architecture (memory access is separate from computation).
  c. Fewer instructions but higher execution speed.
- **Example Architectures:** ARM, RISC-V, PowerPC, MIPS

Impact: RISC became dominant in mobile computing and embedded systems (e.g., Apple M1, Qualcomm Snapdragon, Raspberry Pi).

3. 64-bit Architecture

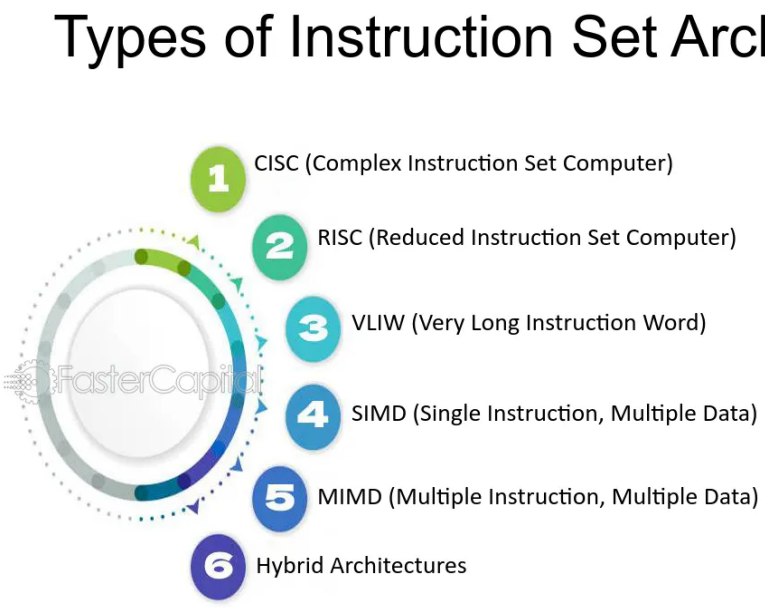
- Transition from 32-bit to 64-bit ISAs (x86-64, ARM64, RISC-V 64-bit).
- Allows access to more memory (beyond 4GB) and increases processing power.
- **Example:** AMD introduced x86-64 (AMD64) in 2003, which Intel later adopted.

4. **SIMD (Single Instruction, Multiple Data) – Vector Processing**

- **Goal**: Enable parallel execution of the same operation on multiple data points (e.g., multimedia, AI, physics simulations).
- **Examples of SIMD Instructions:**
  1. Intel MMX, SSE, AVX (x86)
  2. ARM NEON (ARM)
  3. RISC-V Vector Extensions
- **Impact**: Boosts graphics, gaming, AI, and scientific computing

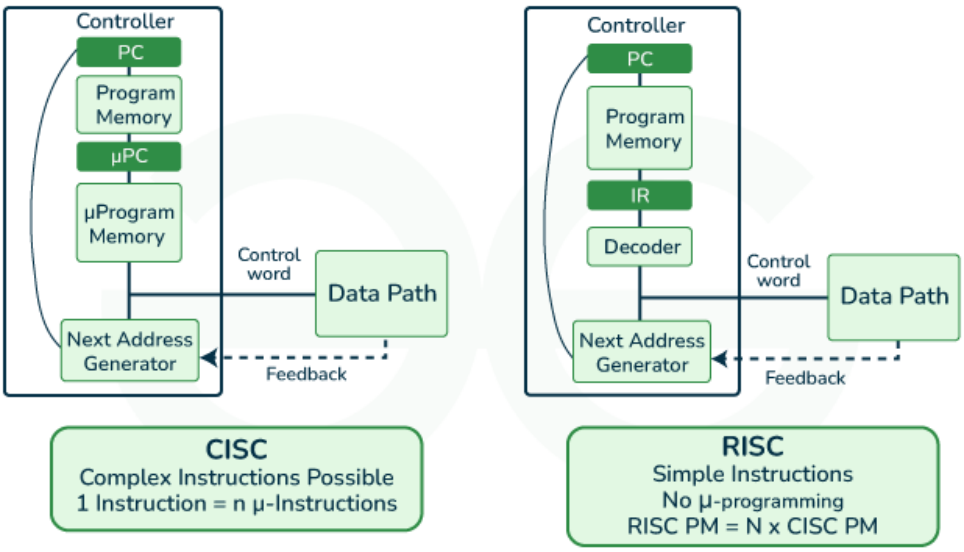
5. **Multi-threading & Parallel Execution (SIMT, SMT, HT)**

- Added instructions for parallel workloads.
- **Example ISAs:**
  1. Intel Hyper-Threading (HT) – Allows a single core to run multiple threads.
  2. ARM big. LITTLE – Dynamically switches between high-power and low-power cores.
  3. SIMT (Single Instruction, Multiple Threads) – Used in GPUs (e.g., NVIDIA CUDA).



CISC and RISC comparison

CISC focusses on performing instructions at once, making the instruction more complex whereas RISC tries to make instructions simple. Here are their characteristics and features in details.



GeeksforGeeks. (2024, December 27). RISC and CISC in computer organization.
Types of instruction set Architectures - FasterCapital. (n.d.). FasterCapital.

CISC, Features:

- CISC features complex instructions, leading to intricate instruction decoding.
- Instructions are typically larger than a single word in size.
- Execution of an instruction may require multiple clock cycles.
- There are fewer general-purpose registers because operations are conducted directly in memory.
- Addressing modes are more complicated.
- There is a wider variety of data types.

Benefits of CISC:

1. **Decreased code length:** CISC processors utilize intricate instructions that can execute several operations, leading to a reduction in the code required to complete a task.
2. **Increased memory efficiency:** Due to the complexity of CISC instructions, fewer commands are needed to carry out intricate tasks, which can lead to code that is more efficient in terms of memory usage.
3. **Extensively utilized:** CISC processors have been around longer than RISC processors, resulting in a larger user community and a greater availability of software. Benefits of CISC Decreased code length: CISC processors utilize intricate instructions that can execute several operations, leading to a reduction in the code required to complete a task.

4. **More efficient in terms of memory:** Due to the complexity of CISC instructions, fewer instructions are needed to carry out intricate tasks, leading to code that is often more memory-efficient.
5. **Commonly utilized:** CISC processors have been around longer than RISC processors, resulting in a larger user community and a greater selection of available software.

Disadvantages of CISC:

- **Slower performance:** CISC processors require more time to execute instructions due to their intricate instruction sets and the extended time needed for decoding.
- **Increased design complexity:** CISC processors possess more complicated instruction sets, leading to greater challenges in their design and manufacturing processes.
- **Greater power usage:** CISC processors have higher power consumption compared to RISC processors because of their complicated instruction sets.

RISC, features:

- More straightforward instructions lead to easier instruction decoding.
- Instructions consist of only one word.
- Each instruction requires just one clock cycle for execution.
- A greater number of general-purpose registers are available.
- Addressing modes are uncomplicated.
- There are fewer data types utilized.
- A pipeline implementation is possible.

Advantages of RISC Architecture

- RISC processors employ a streamlined set of straightforward instructions, which enhances their decoding and execution speed. This leads to quicker processing times.
- Due to their simpler instruction set, RISC processors achieve faster instruction execution compared to CISC processors.
- RISC processors use less power than CISC processors, making them well-suited for portable devices.

Disadvantages of RISC

- RISC processors need a greater number of instructions to accomplish intricate tasks compared to CISC processors.
- RISC processors necessitate more memory to accommodate the extra instructions required for executing complex tasks.

- The development and production of RISC processors can incur higher costs than those associated with CISC processors.