## The Digital Computer

- Electronic Machine to run Programs
  - Program: Instructions, Data
  - Sequence: Instructions are performed in the order in which they are written
- Machine-Level Instructions are atomic (simple)
  - Operate: Perform Arithmetic and Logic, ADD, AND
  - Data Move: Transfer Data between Memory Words and Processor Registers, LOAD, STORE
  - Control: Alter Execution Sequence, BRANCH, CALL

## Virtual Machines

- We seldom work on the *Actual Computer*.
- Our view of the Computer is determined largely by our Programming Language or IDE, a conceptualized or *Virtual Computer*.
- Programs developed for a *Virtual Computer* must be converted into equivalent Programs for the *Actual Computer*.
- Program conversion strategies:
  - Translation
  - Interpretation

## Translation *vs.* Interpretation

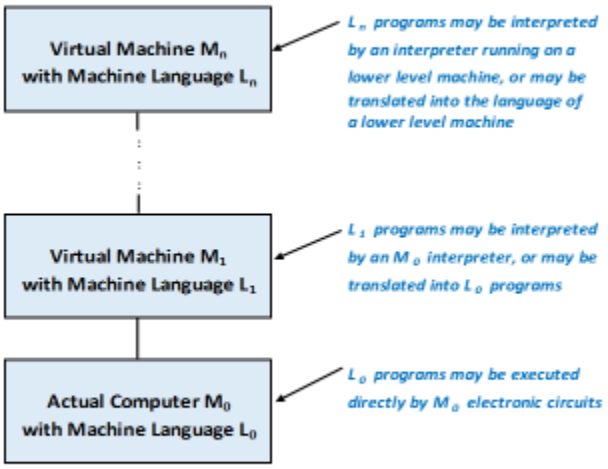
- Translation:

  ***Entire*** *Virtual Computer* program is translated into the equivalent *Actual Computer* program ***prior to execution***: Compilation, Assembly.

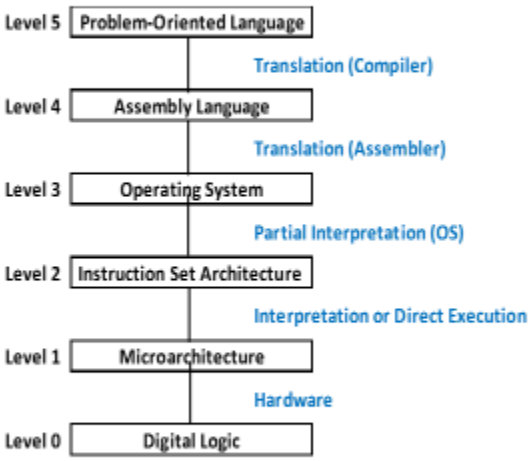
- Interpretation:

  Each *Virtual Computer* program instruction is ***translated individually*** into *Actual Computer* instructions and ***executed directly***, one by one: Interpreter: *interleaves conversion & execution one (virtual) instruction at a time*

## Virtual Machines



# Contemporary Multilevel Machines




# Virtual Machine Level Features

- Level 5: Problem-Oriented Language Level
  Compilers, IDE's
- Level 4: Assembly Language Level
  Assembler
- Level 3: Operating System Level
  Service Routines, Process Scheduling, Resource Management
- Level 2: Instruction Set Architecture Level
  Machine Language Instructions
- Level 1: Microarchitecture Level
  Data Path. May be hardwired or microprogrammed
- Level 0: Digital Logic Level
  Logic Gates, Registers, Memory, Combinational Circuits

## Evolution of Multilevel Machines

- The earliest digital computers had only Digital Logic and ISA Levels: COLOSSUS – 1943, EDSAC – 1945, ENIAC – 1946
- Introduction of the Assembly Language Level is associated with early stored program computers: 1949 - Initials Orders for the EDSAC used single-letter mnemonics.
- Introduction of the Problem-Oriented Language Level began with the first *Compilers* developed for the MARK 1 computer: 1952 – Autocode, 1954 – Mark1 Autocode.
- Early *Resident Monitor* programs automated job sequencing: GM-NAA-I/O - 1956 evolved into SHARE OS – 1959. As we know it, the Operating System Level probably began with the IBM OS 360 PCP, MFT and MVT families – circa 1966.
- The Microarchitecture Level – *following*

## Microinstructions

- A machine language instruction is performed as a series of data path operations; each transfers data between two registers.
  Example: LOAD REG, X_address

  ***MAR ← PC***
  ***PC ← PC + 1***
  ***MDR ← memory[MAR]***
  ***IR ← MDR***
  ***MAR ← X_address***
  ***MDR ← memory[MAR]***
  ***REG ← MDR***

- When coupled with control signals each data path operation is described as a microinstruction, an atomic data path operation.
- The job of the Control Unit is to sequence the microinstructions

# Microprogramming

- Microprogramming is a technique of implementing the functions of the control unit - proposed by Maurice Wilkes in 1951 as an alternative to hardwiring, and extended in collaboration in 1953.
- The microprogram comprises a sequence of microinstructions for each machine language instruction stored into a ROM. *It is firmware, software embedded in hardware*.
- The advantages of microprogramming include
  - ✓ Simplifying the hardware by reducing the number of vacuum tubes and other electronic circuitry
  - ✓ Improving reliability by reducing the potential for hardware failure
  - ✓ Providing flexibility – the ability to extend the ISA since a new machine language instruction could be programmed into the microprogram ROM
- The Microarchitecture Level was implemented for testing in 1957, and introduced commercially in the IBM 360's in 1964.

# Evolution of Microarchitecture

- Invention: Proposed by Wilkes in 1951, refined and extended in collaboration in 1953
- Adoption: Several Mainframe and Mini-Computer architectures in the mid 1960's, through the1970's:
  IBM System/360/370, DEC PDP-11 Series, DEC VAX 11 Series
- Impact: Migration of more hardware functionality to microcode leading to large CISC (next slide):
  VAX 11/780, MC 68020, Intel-based microprocessors
- Decline: Reduction in use due to several factors
  Bloating – ultimately, increase lead to slower processors
  VLSI – miniaturization made faster hardware more attractive
  CAD – greatly helped solve design & layout fifficulties
- https://people.cs.clemson.edu/~mark/uprog.html

## Increased Functionality in Microcode

- Instructions for integer multiplication and division
- Floating-point arithmetic instructions
- Instructions for calling and returning from procedures
- Instructions for speeding up looping
- Instructions for handling character strings
- Indexing and indirect addressing
- Relocation facilities
- Interrupt systems
- Process switching
- Processing audio, image, multimedia files

## Computer Generations

- Zeroth Generation Mechanical Computers (1642 – 1945)
  Pascal's Calculating Machine, Babbage's Analytical Engine
- First Generation Vacuum Tubes (1945 – 1955)
  COLOSSUS, ENIAC  von Neumann Architectures: EDVAC, IAS
- Second Generation Transistors (1955 – 1965)
  MIT: TX-0, TX-2  DEC: PDP-1, PDP-8  IBM: 7904, 1401  CDC: 6600
- Third Generation Integrated Circuits (1965 – 1980)
  IBM: System 360 Series  DEC: PDP-11 Series
- Fourth Generation VLSI Circuits (1980 – ?)
  IBM: PC, x86 family  Apple: Macintosh
- Fifth Generation Low-Power and Invisible Computers
  Tablets, Smartphones, Embedded Processors