- Six levels of computer include:
    - **Digital logic**: physical implantation of computer circuits (hardware)
    - **Microarchitecture level:** decide which components we use and how we use it
    - **Instructional set architecture level:**

*Binary operations have 2 inputs unary 1 input. 3 inputs =

- **Truth Table:** describes how logic's circuit output depends on logic level input. Table for all possible input. The convention is exhaust 0 first in input A. Then once all possible combination for input 0. Then raise input

- **Logic gates**: are used to build logic circuits. There are five different types:
    - Or
    - And
    - Not (inverter)
    - Nand
    - Nor
    - Xor
    - Xnor

- **Or gate: + (A + B)**
    - Boolean expression is x= A + B
    - Do first or circuit

- **And Gate: * (A*B or AB or Ab)**
    - X = AB

- **Not gate: - OR ‘ (NOTE IF NO A NOT GATE) *BEFORE TEST RECHECK**
  - Unary only taking one input aka inversion or complementation.
  - Boolean expression: X = A

- **Nand gate: negative “AND”** (AB with Hat or (AB)’)
  - **2 inputs**

- **Nor gate: inversion of “or”** Bubble represents (A + B with Hat) or (A+B)’

- **Xor gate: A(+)B**

    - If both input are 1 output is 0
    - Mutually exclusive so if both 1 it's not exclusive

- **Xnor gate:** inverted form of xor
    - Opposite of xor

- Boolean is best written as sum of multiple and ie y = AB + AB
- More efficient is using one type in parrel or series.

- **Multiplexer:** lots of input but only one will be outputted.

- **Decoder:** coded message and you wanted to decode (3 input). All 7 will not be fired only 1.

- **Comparators:** If you have two numbers we use to compare. Someone better (or exlcusive or input + output or). If two numbers are equal. If end thing is not 1 then its not equal

- **Shifter:** shift every byte left. Empty spots is zero

- **Half adders:** NOTE: 1 1 = 0, 0 1 = 1 (sum) (carry 0)

- **Adders:** add cirut or numbers. Carry in is 0/1/0/1

- **Latches (SR):** devices that act as memory cell retain values/write./read. So we can later use it or feedback A gate only uses present input. Store outputs. Based of nor principal. If input s is 1 then output is 0 (opposite of D-latch)
-
- **Flip Flop:** is latch but are clocked

- **D latch:** difference is that: D (is an input on diagram) = s = R inverse. So if D = 1 then S = 1 R = 0. But if D (input on diagram) = 1 then Q = 1 [Reverse for 0]

- 1 ram = 1 byte

- 5ghz over 2,4ghz is due to carry more bandwidth but attenuated (bounce off walls)
- **Programmable logic arrays:** reduces gate delays keyboard. Contains all gates and combines them.

- **Gate delay:** one is going to turn up before other

- **Clockedspeed** Can't just increase clock speed to increase speed It because you have gate delays to worry about and other factors.