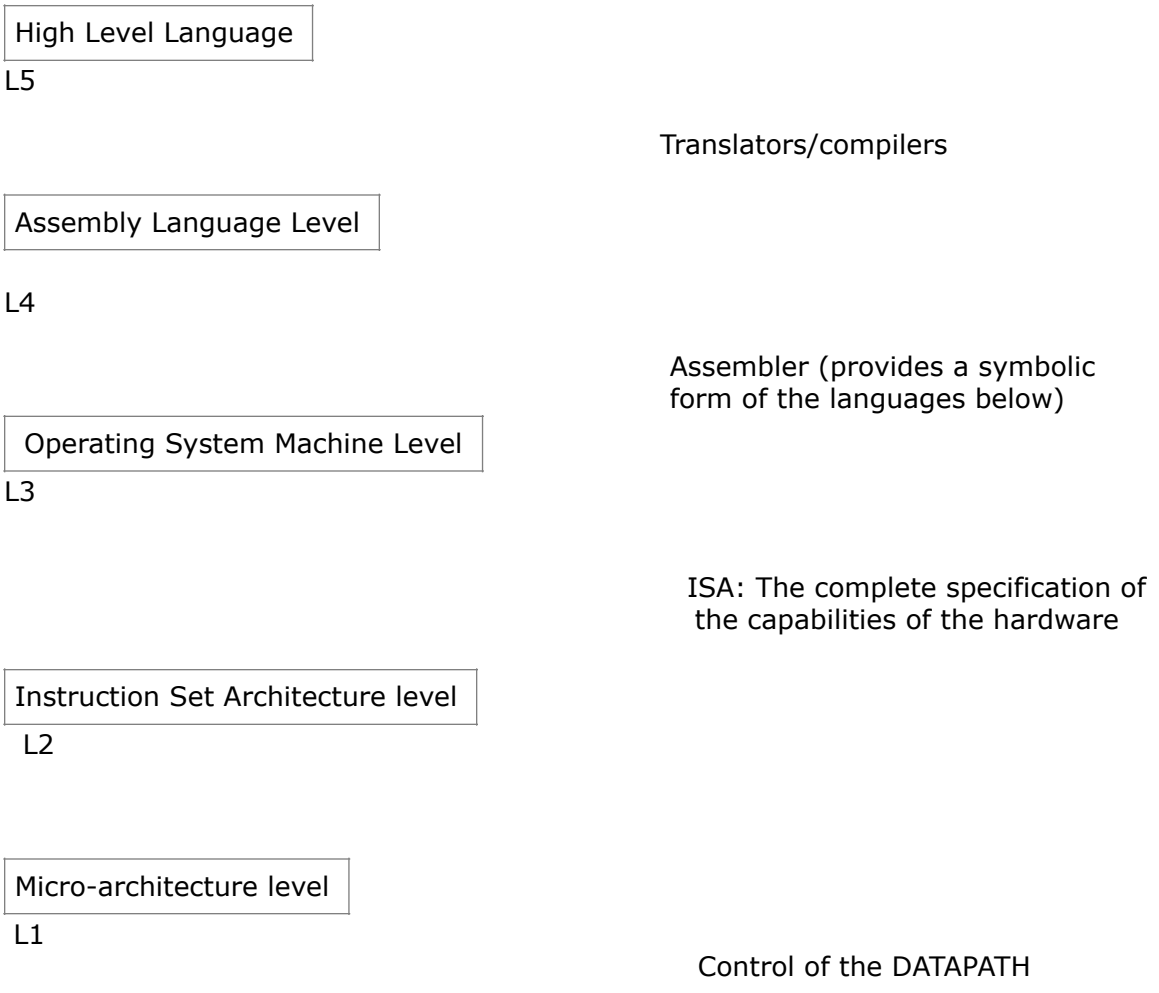
Translating a problem stated in a natural language to voltages and currents on a set of "wires" requires a sequence of transformations:
Problem → Algorithm → Program → ISA → Microarchitecture → Logic Circuits
At each level, the problem is stated in the "language" of the level to the right, till the problem could be communicated to the hardware.
Imagine for now that the built-in machine (at the lowest level) understands a language **L0**. At the next level above that, we could create a new set of instructions that are more convenient to use. Together, these form a language **L1**.
One way to execute a program in L1 is to expand each L1 instruction into a sequence of L0 instructions (TRANSLATION), create an equivalent program in L0, load and run it. Another way is to write a program in L0 that takes a program in L1 as input and expands each instruction and directly executes it (INTERPRETATION). The process could be continued: a new language L2, each of whose instructions are made up of a seq. of L1 instructions could be created and so on.

A way to abstractly view this process is to view each level j, with its language Lj as a VIRTUAL MACHINE Mj with Lj as its MACHINE LANGUAGE (all the instructions Mj can execute). Highlighting an important relation between a (virtual) machine and its language:
The language defines the machine and the machine defines the language
Contemporary machines have 2 to 6 levels:



(ALU, registers and the Buses)
micro-program/hardwired

Digital Logic Gate level

L0

Gates: AND/OR/NOT &REGISTERS