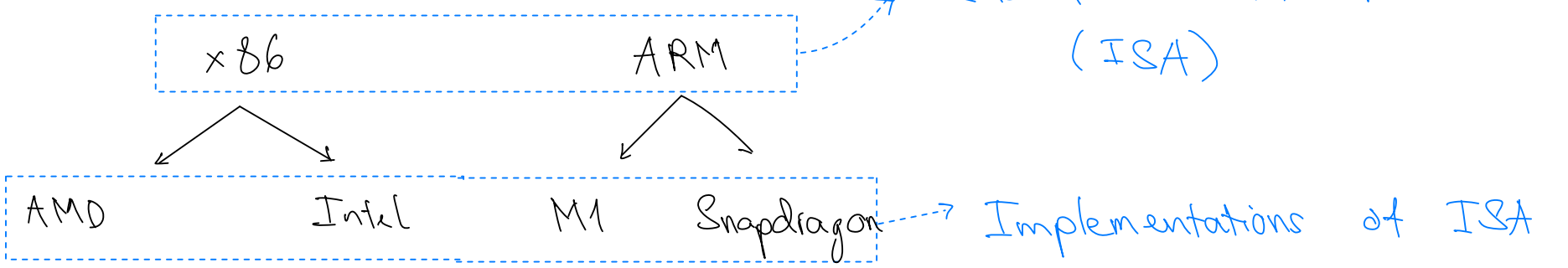
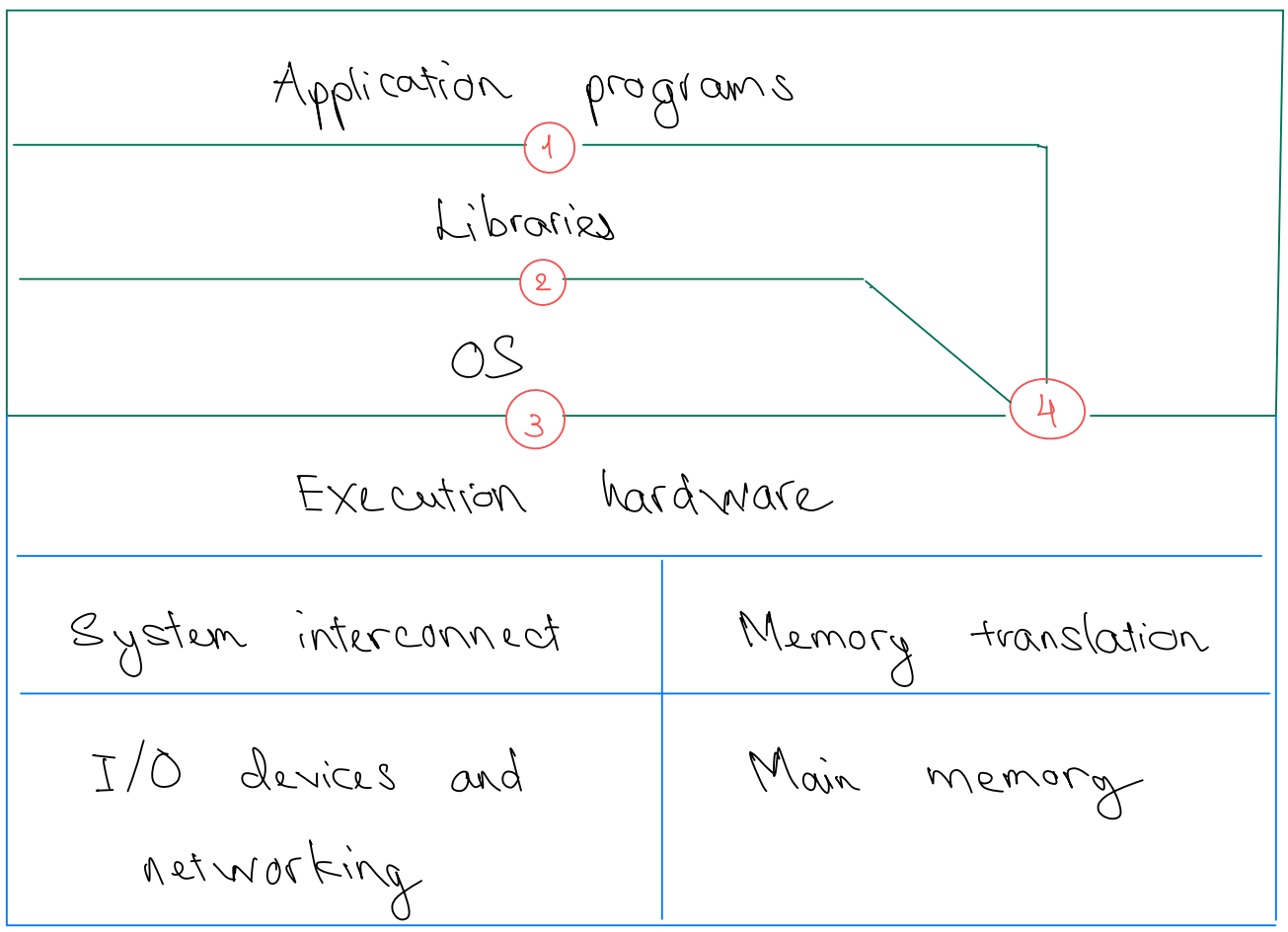
MACHINE ARCHITECTURE

Example:



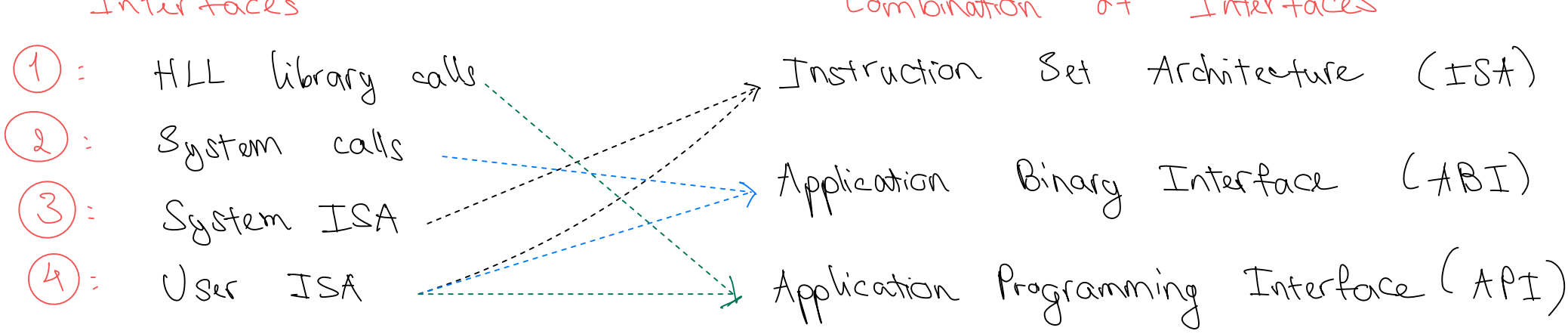
Deep dive into ISA:

Here is what a machine usually look like:



Interfaces

Combination of Interfaces



Focusing on (4) : User ISA

The formal contract between hardware and software

$A_4 = (\Sigma, \jmath)$ ⟶ Machine Architecture (A_4)

consists of

Machine States (Σ)

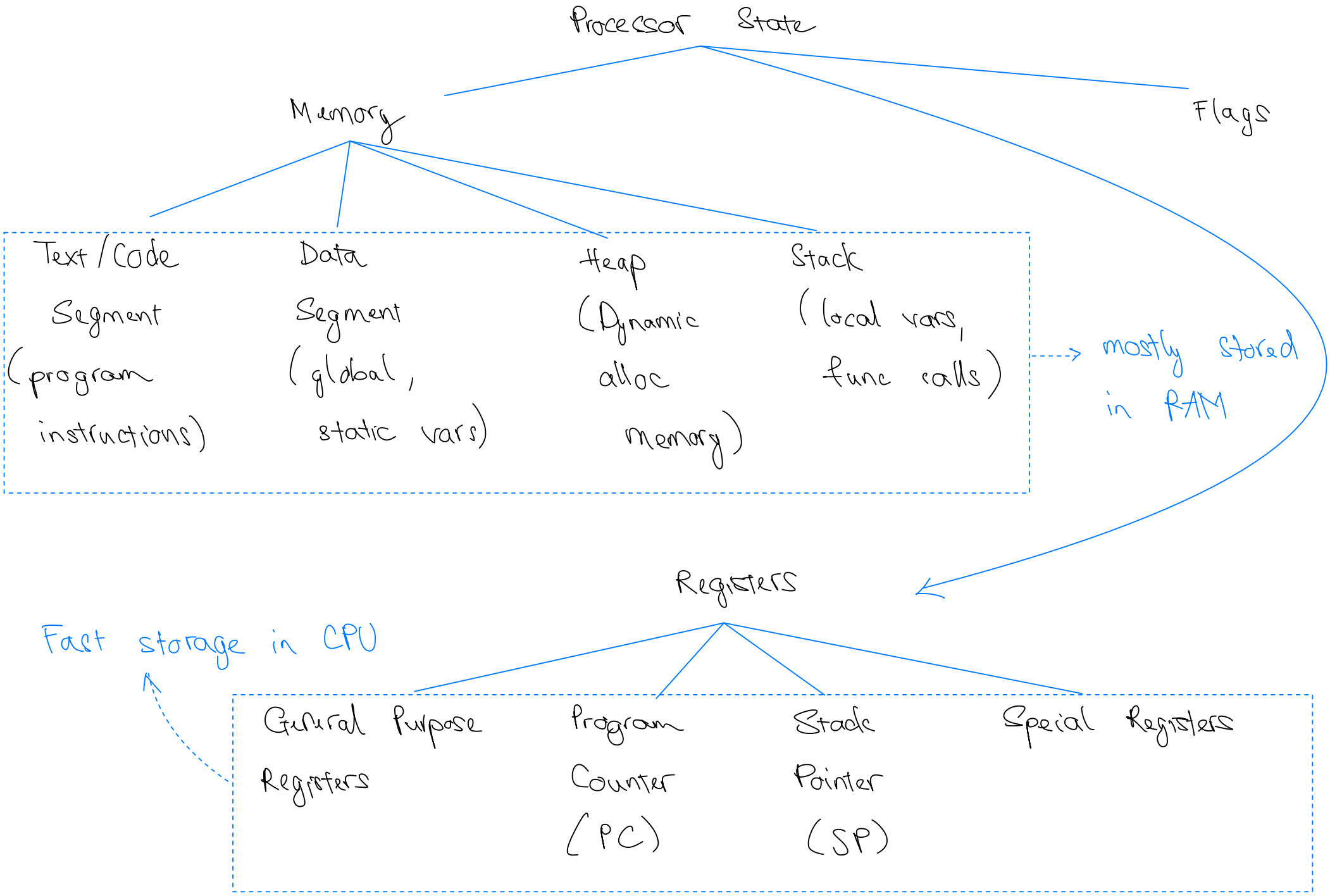
Memory State

Processor State

Machine Instructions (ISA, $\jmath$)

transform input state to output state

Processor State



How a program flows through Processor State:

Load program into memory → Initialization

- Load Instructions in Text Segment
- Set Program Counter to first instruction
- Init global vars
- Prepare stack for function call

Fetch Instruction

- CPU Fetch instruction from Text Segment pointed to by the program pointer (PC)

Decode Instruction

- CPU determines what operations to perform

Update Process State

- Update data/pointers in registers
- Update memory (heap, stack)
- Update flags

Interrupt Handling

(I/O operations, timers, ...)

Context Switching

where Processor State is saved, and other process is ran

ISA taxonomy

Based on maximum number of operands explicitly specified in instruction

- 0-operand machine "Stack Machine": (JVM)

  Example: PUSH (5) (push 5 to Stack, when PUSH is called, the PC pointed to the next item to get the value, 5 in this case)

  PUSH (3)
  ADD
  PRINT

- 1-operand machine "Accumulator Machine": (Small microcontrollers)

  Example:
  LOAD 5 (load 5 in accumulator)
  ADD 3 (add 3 to accumulator)
  MUL 2 (multiply accumulator by 2)
  STORE 100 (store result in memory location 100)

- 2-operand machine: (x86-64)

  Example:
  MOV R0, 5 (Move 5 into register R0)
  ADD R0, 3 (Add 3 to R0, R0 = 8)
  MUL R0, 2 (Multiply R0 by 2, R0 = 16)
  STORE R0, 100 (Store result of R0 to location 100)

- 3-operand machine: (ARM v8)

  Example:
  MOV R0, 5 (Move 5 into register R0)
  ADD R1, R0, 3 (Add R0(5) and 3, store in R1)
  MUL R2, R1, 2 (Multiply R1(8) by 2, store in R2)
  STORE R2, 100 (Store result of R2(16) to location 100)

How ISA taxonomy affects Processor State

It affects significantly how to updates Processor States. For example:

- 0-operand machine: Stack used heavily, PC increment each instruction, no general-purpose registers needed
- 3-operand machine: Multiple registers used, memory access less frequently, PC increments each iteration, status flags updated after ADD operation

- etc ...