Instruction Decode

MO601

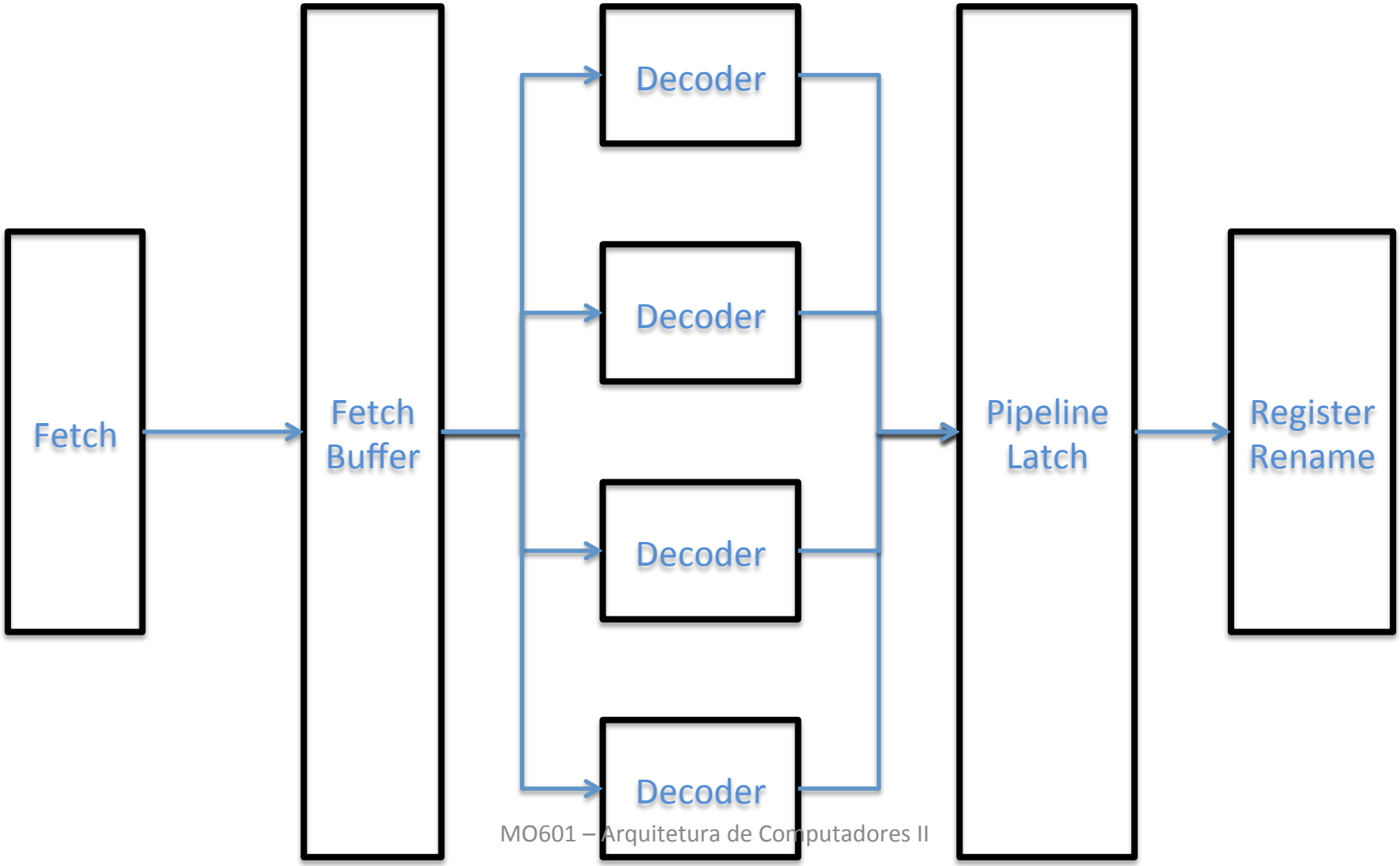
What do we need to decode?

- What type of instruction
  - Control, memory, arithmetic, etc
- What operation the instruction should perform
  - ALU operation
  - Branch condition to check
- What resources are needed
  - Input registers
  - Output registers

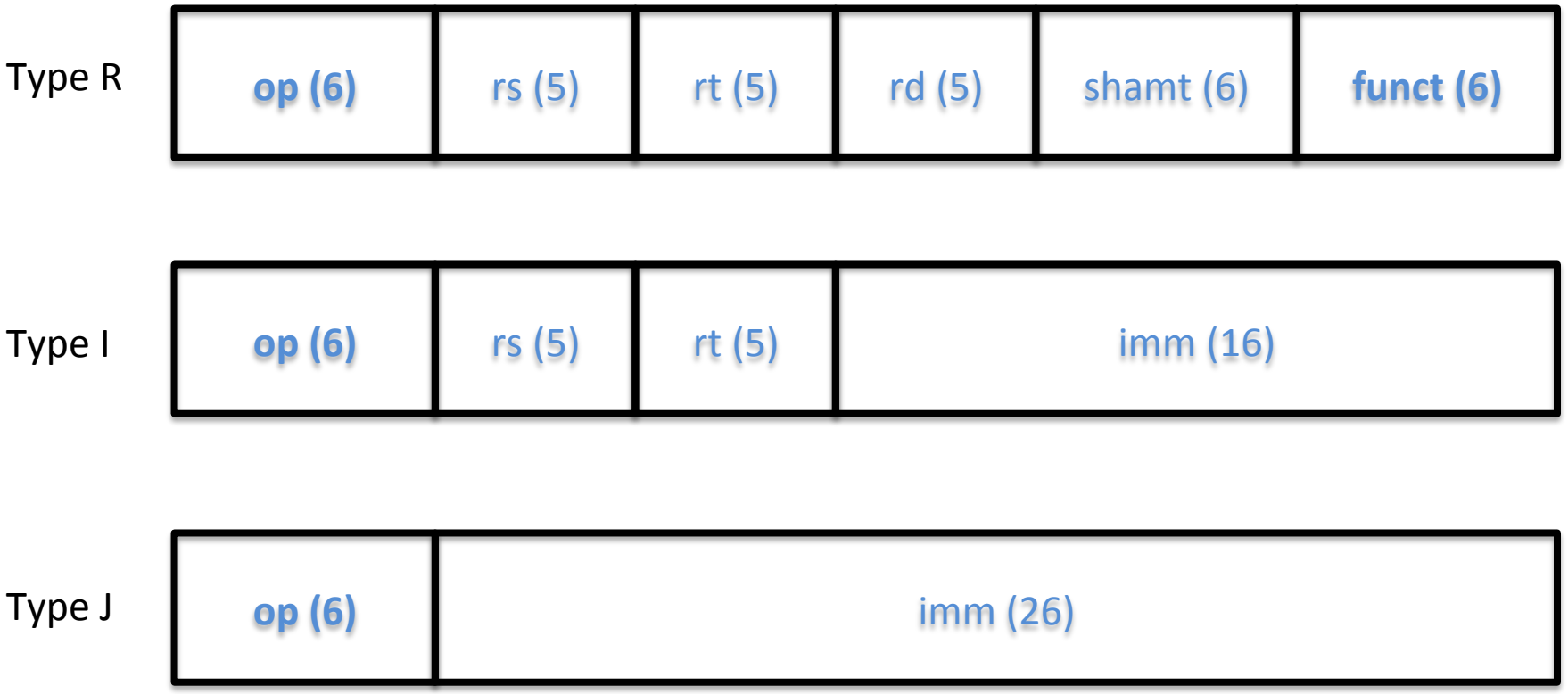
RISC vs CISC

RISC		CISC
Fixed size	Instruction size	Variable size
Few	Encoding formats	Several
Low	Instruction complexity	High
Simple	Cycles to decode	Multiple
Few	Fields to decode	Multiple

Decoding multiple instructions



MIPS Instruction Formats

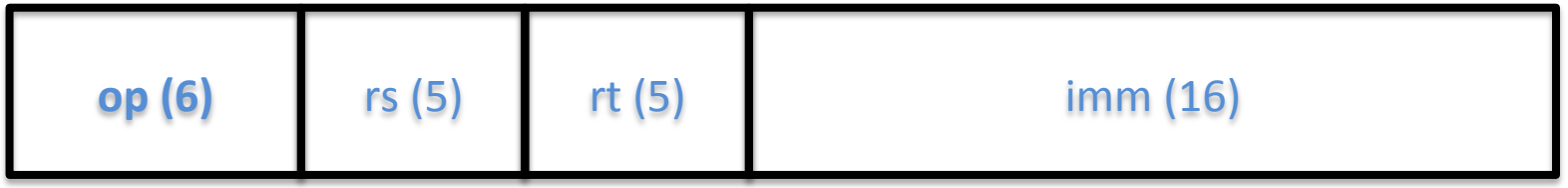


Exercise

Instruction	Type	Fields
add	R	op = 0; funct = 32
sub	R	op = 0; func = 34
addi	I	op = 8
bne	I	op = 5
j	J	op = 2

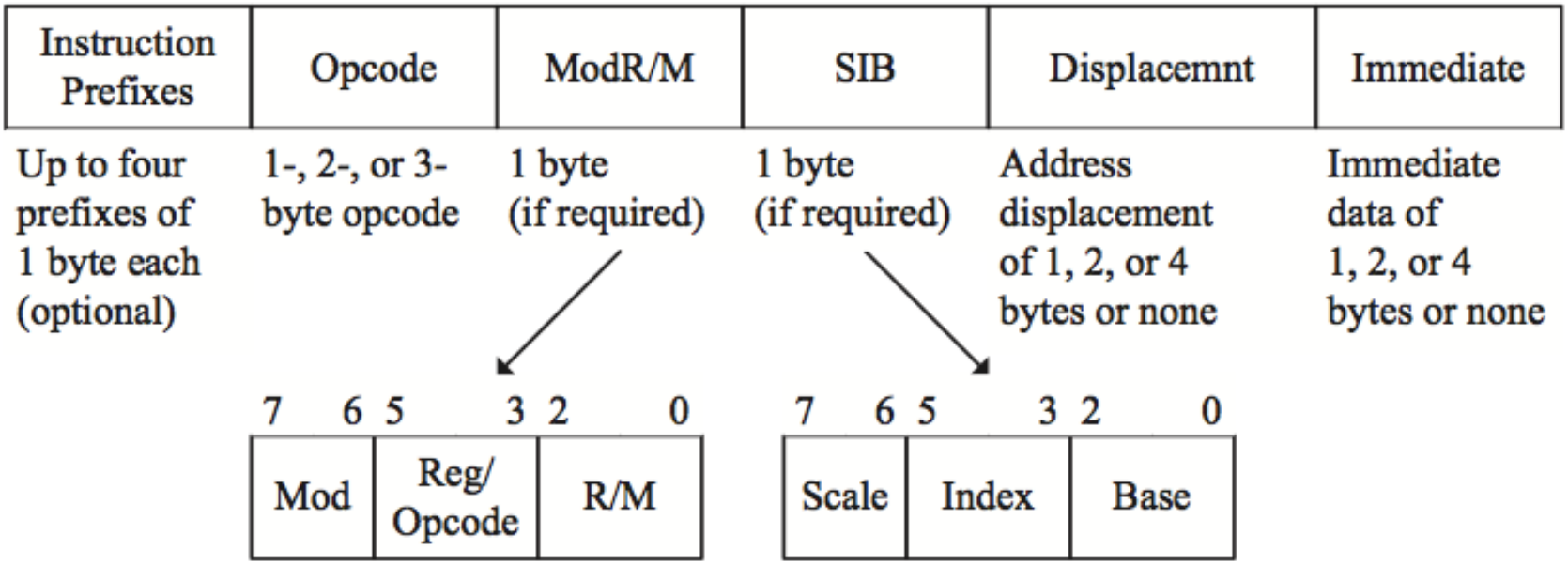
Type R

Type I



Type J

x86 instruction format



How to decode x86 instruction?

- Decode instruction length
- Look for the first 8 bytes
  - Up to 4 prefix byte before it
  - Up to 3 bytes of opcode
  - Sometimes look for ModR/M
- Look for operands
  - Encoded in the opcode
  - Extra bytes
  - Look for ModR/M
  - Prefixes may change the opcode size

How to speedup decoding?

- Store the instruction size in the cache together with the instruction
- Have multiple simple decoders and one complex decoder
- Convert instruction to micro-codes (uop) and execute uops instead of instructions

Intel Nehalem decoding pipeline