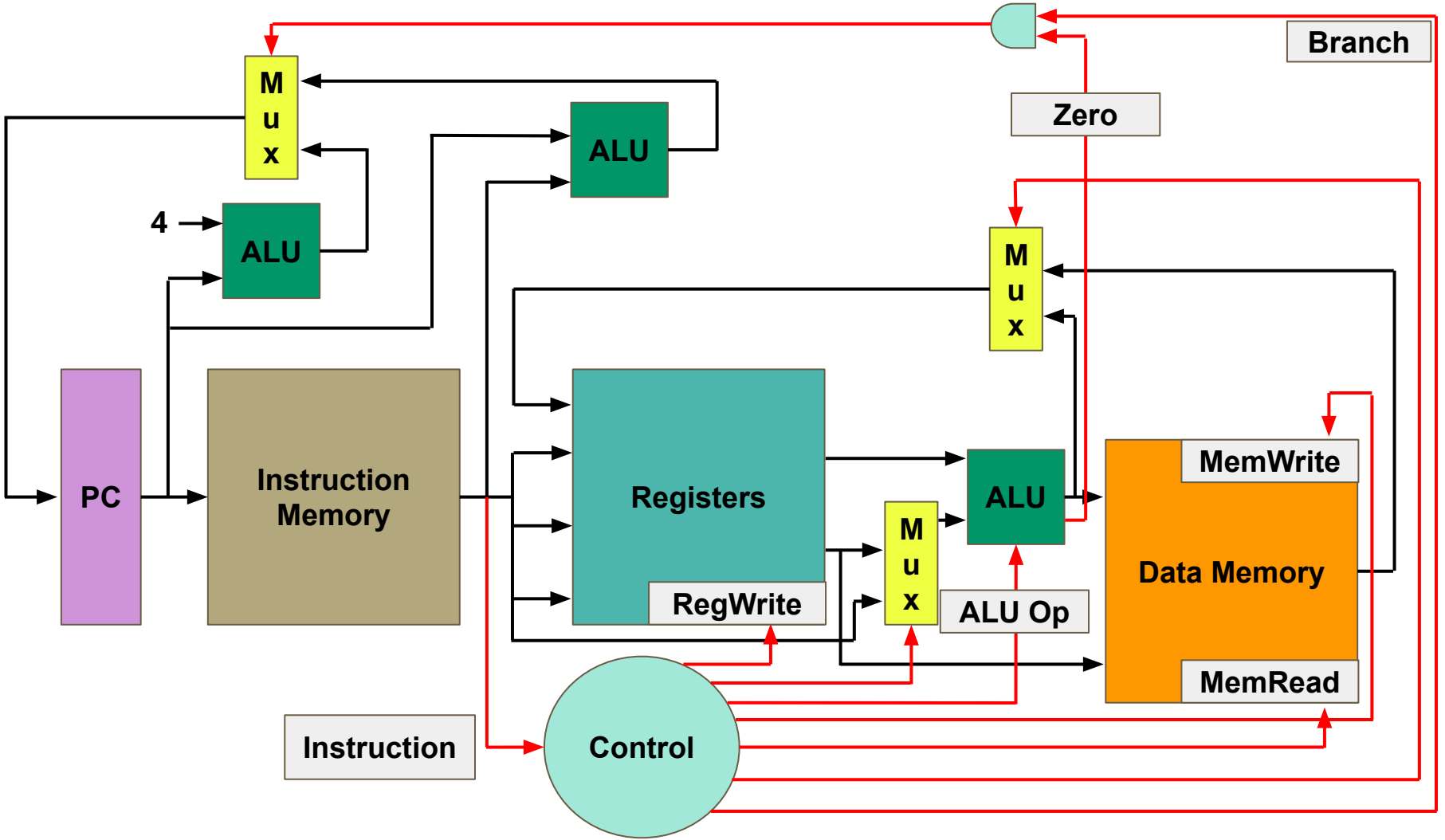
Bits of Architecture

RISC-V Processor
Implementation Details

Our Processor

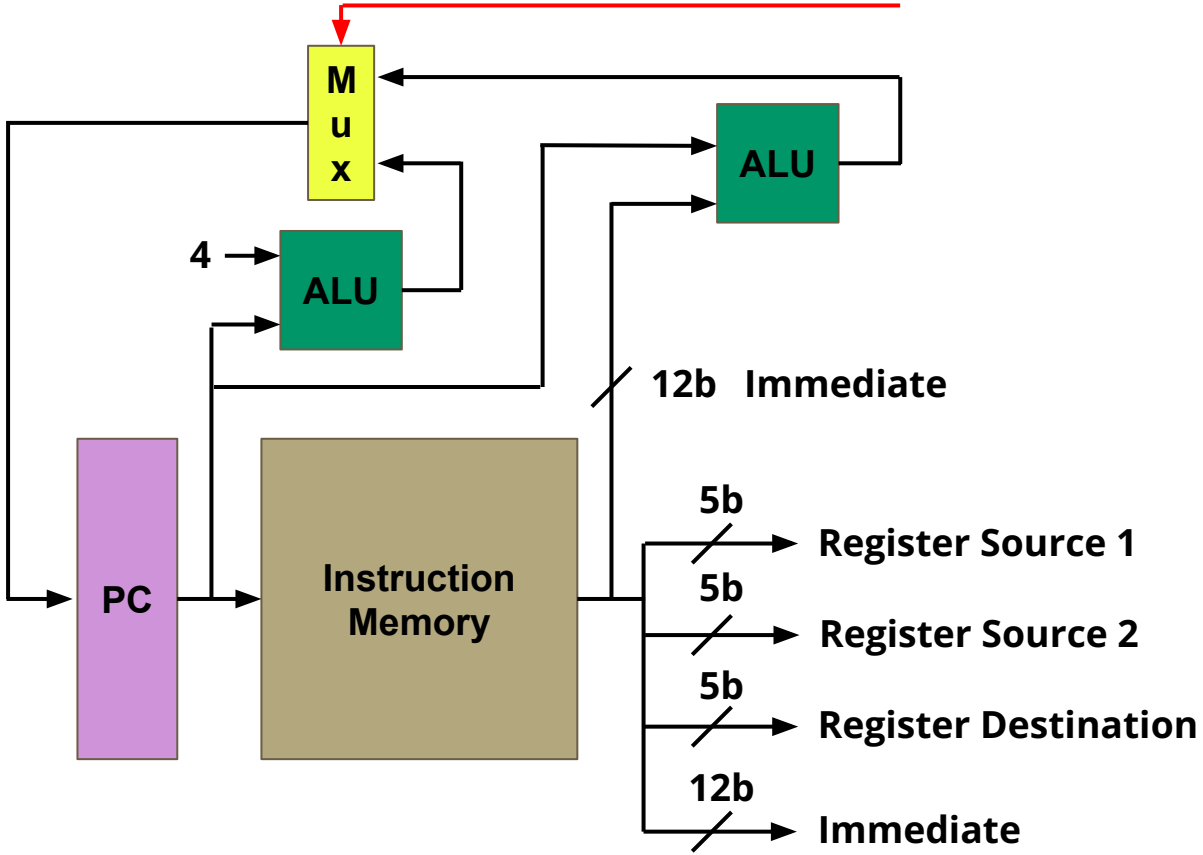

Branch
M
u
x
Zero
ALU
4
ALU
M
u
x
PC
Instruction
Memory
Registers
RegWrite
M
u
x
ALU
MemWrite
Data Memory
MemRead
ALU Op
Instruction
Control

We're Still Missing Components...

Handling Immediates

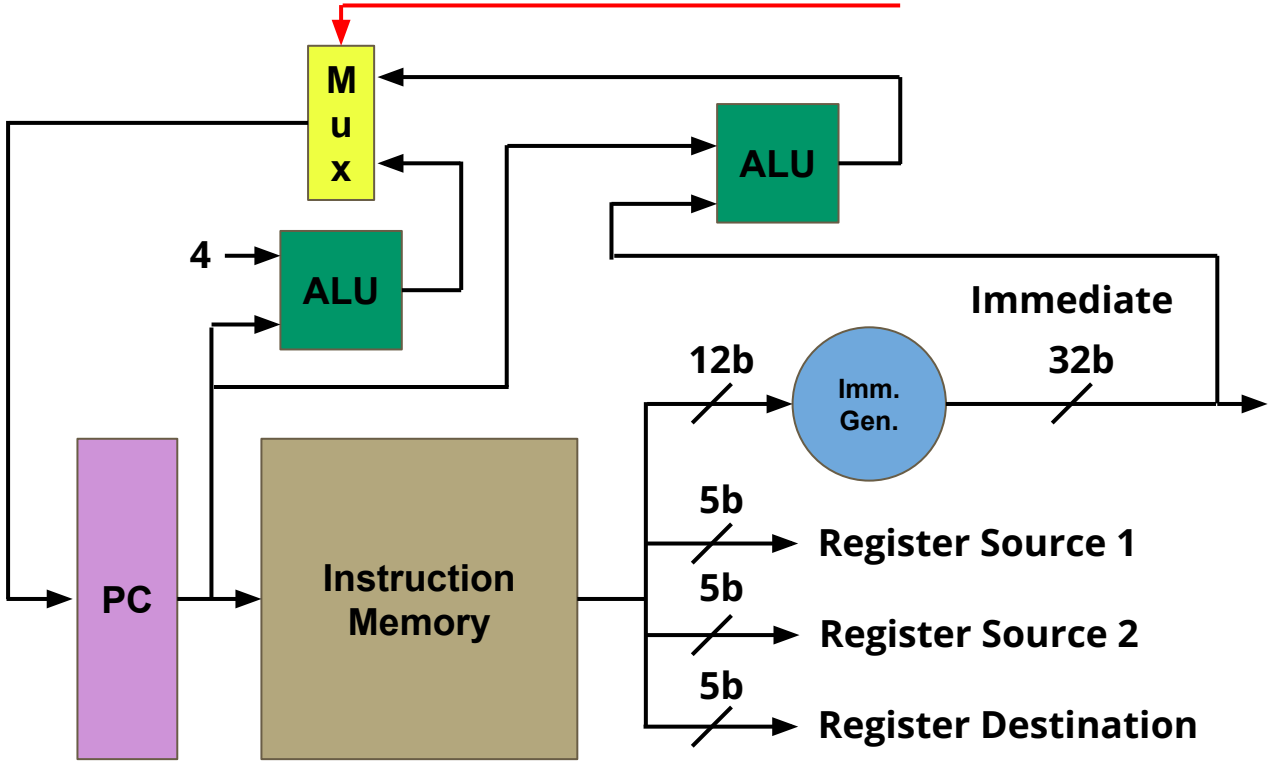
Handling Immediates

- Our immediate value not 32b
- Our other inputs are
  - Register Data
  - PC Value
- How do we add a 12b value with a 32b one?



Immediate Generation

- Create 32b values from our 12b immediates
  - Sign extension
- Rely on **imm. gen.** to choose the right bits from the instruction

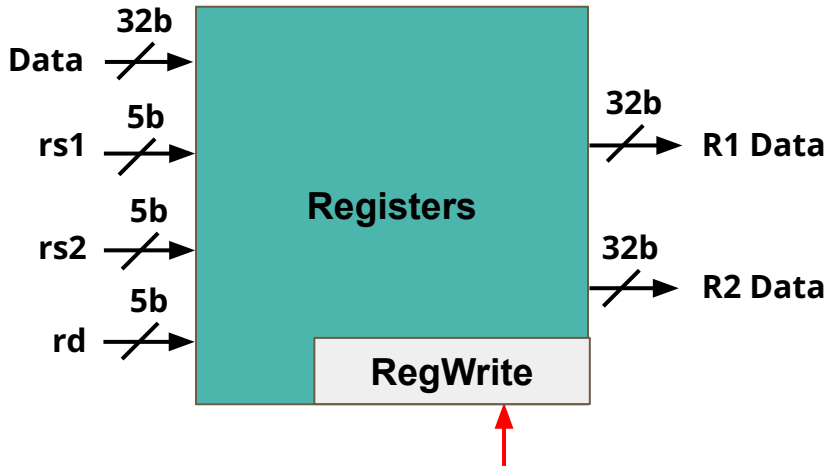


Configuring our Registers

Configuring our Registers

- How do we select which register(s) to use?
- 5b fields from instruction
- Not all instructions use all 3 registers
  - Output will be ignored (multiplexers)

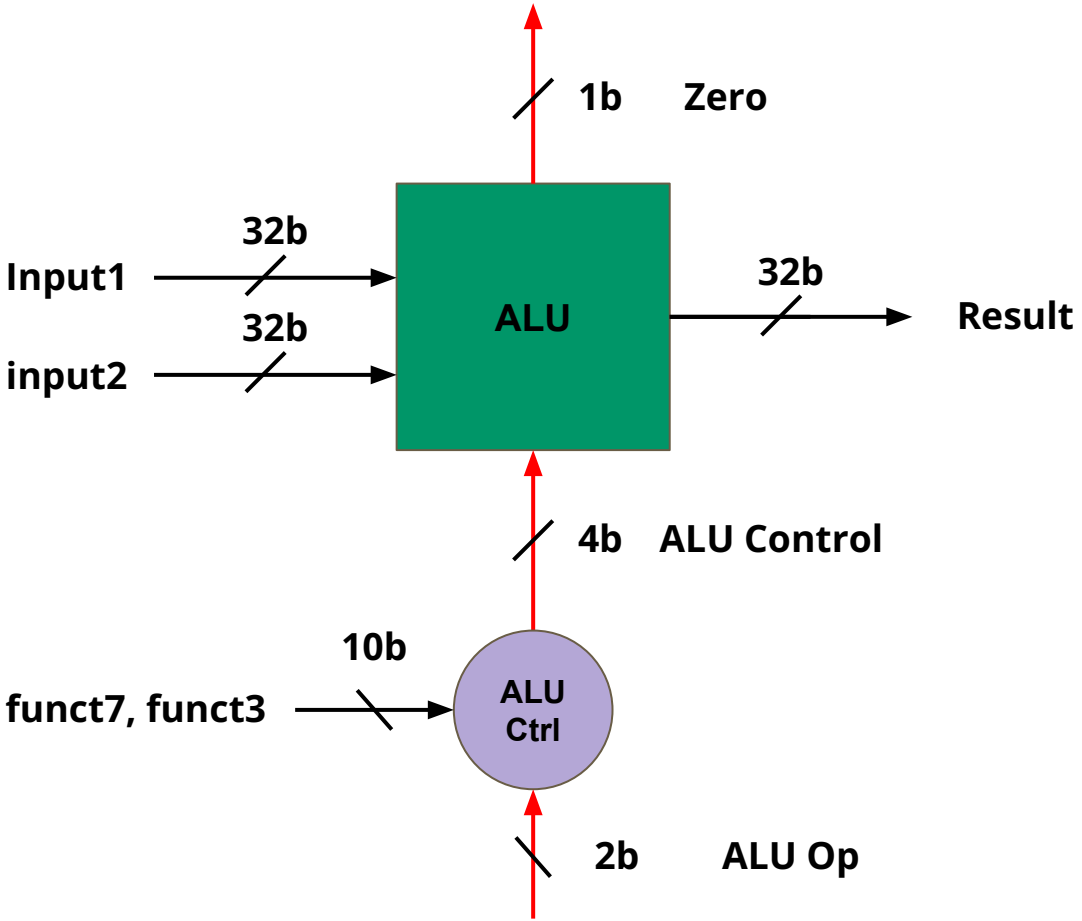
funct7	rs2	rs1	funct3	rd	opcode
31:25	24:20	19:15	14:12	11:7	6:0



Configuring Our ALU

ALU Control

- We need to configure our ALU to perform our operations
  - 0000 - AND
  - 0001 - OR
  - 0010 - Add
  - 0110 - Subtract
- Why 4 bits?
  - A Invert
  - B Invert
  - Op selection



What About the Remaining Signals?

Back to Logic Design...

- Take the input bits
- Create a truth table
  - Mapping of inputs to outputs
- Optimize and turn into gates
- Where do we use it?
  - Instruction -> Control Signals
  - Control Signal -> ALU Control Signal