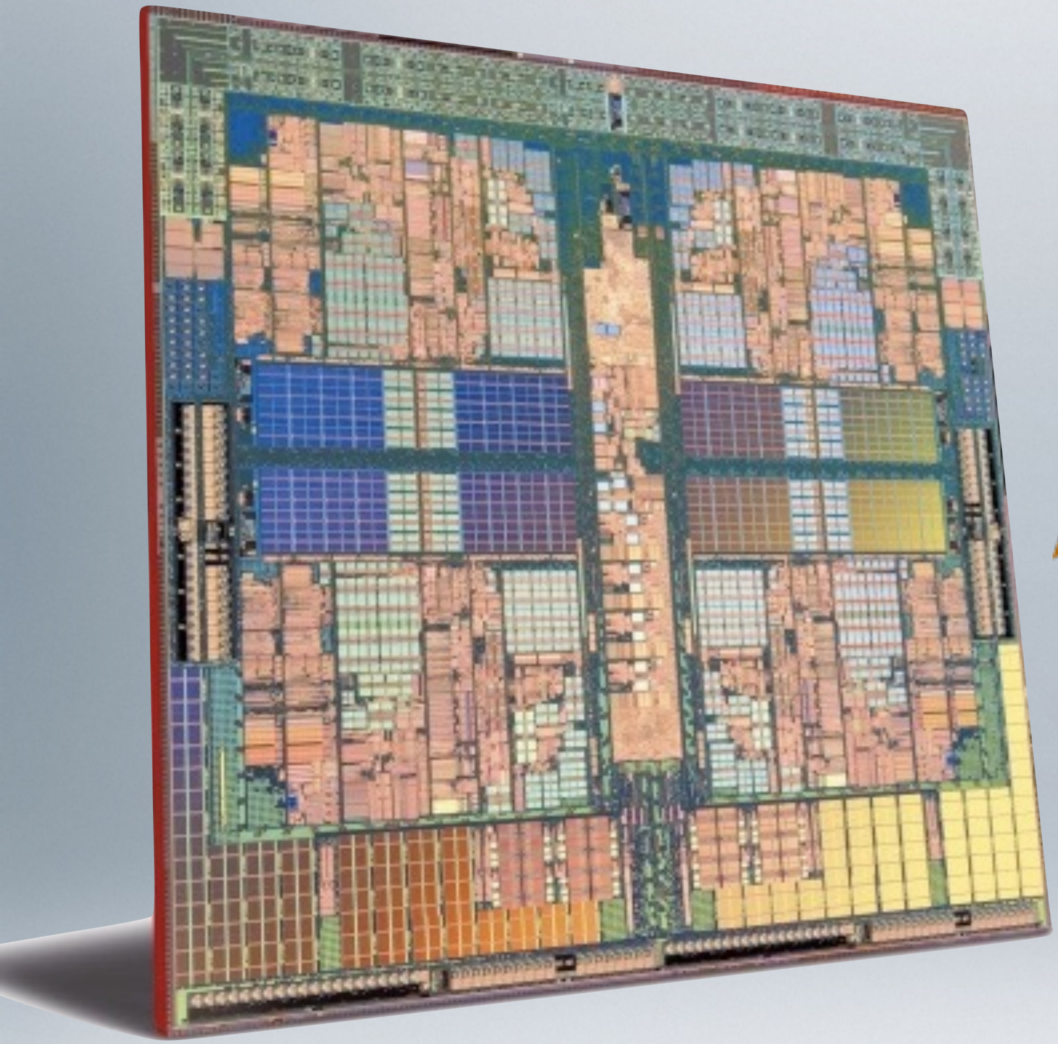


THE AMERICAN UNIVERSITY IN CAIRO
الجامعة الأمريكية بالقاهرة

CSCE 3301: COMPUTER ARCHITECTURE

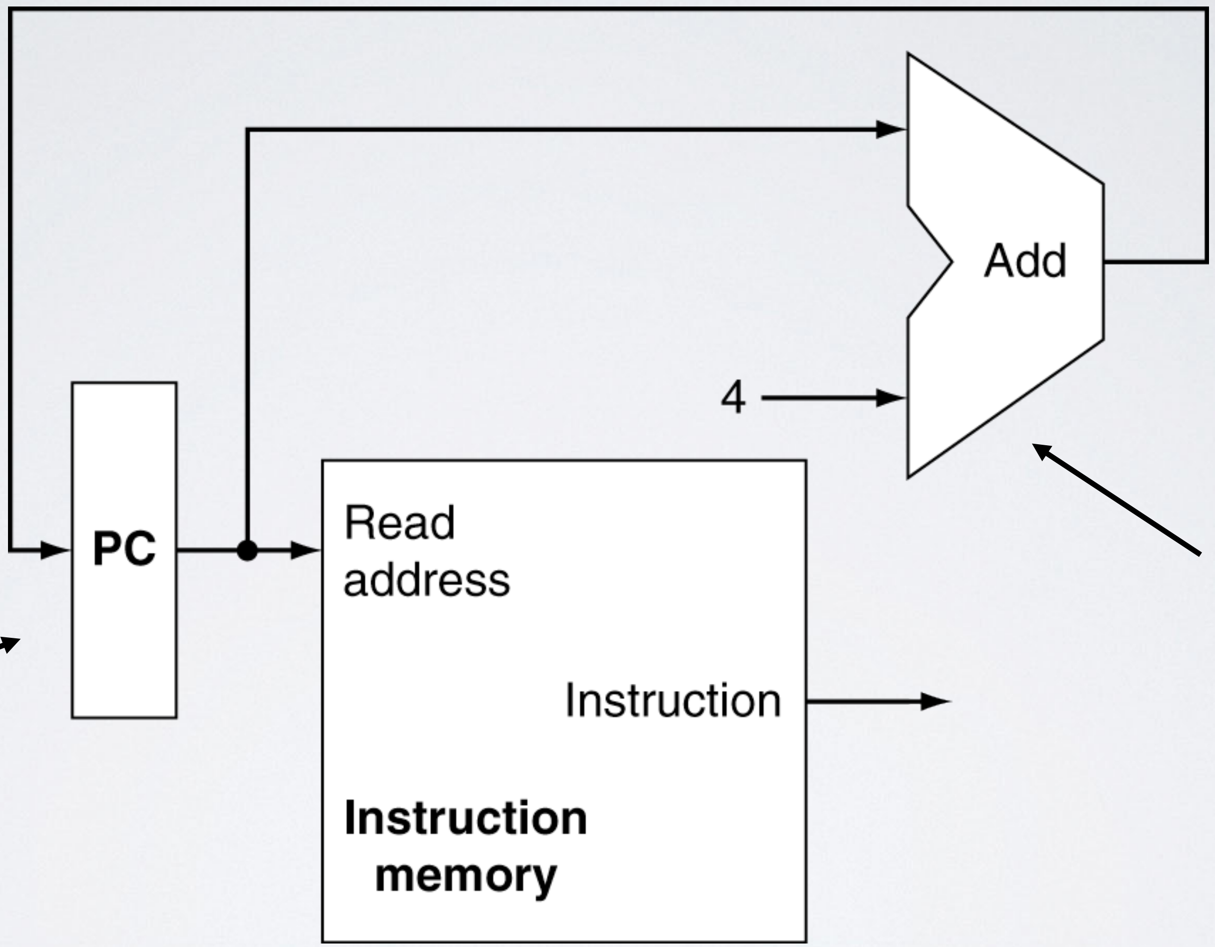
Lecture 6: RISC-V Single Cycle Implementation (Continued)

Dr. Cherif Salama

BUILDING A DATAPATH

- Datapath
  - Elements that process data and addresses in the CPU
    - Registers, ALUs, mux's, memories, …
- We will build a RISC-V datapath incrementally
  - Refining the overview design

INSTRUCTION FETCH

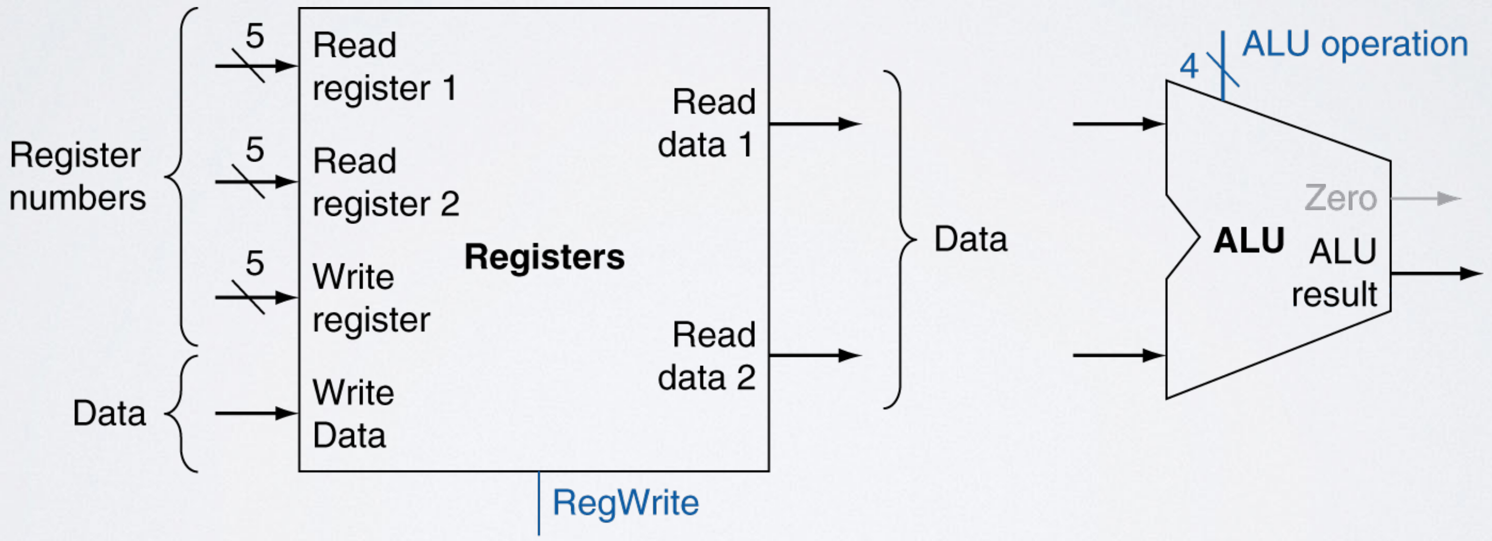


32 bit register

Increment by 4 for next instruction

R-FORMAT INSTRUCTIONS

- Read two register operands
- Perform arithmetic/logical operation
- Write register result

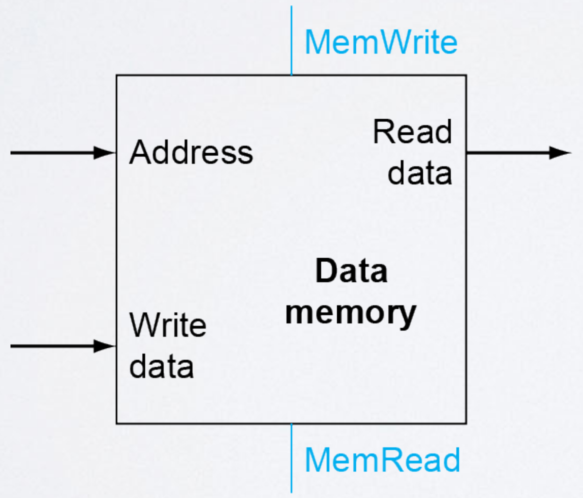


a. Registers

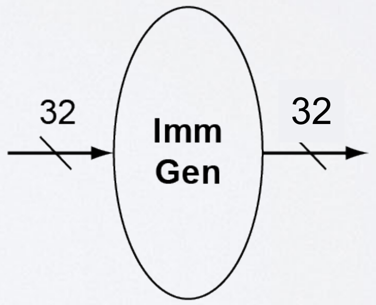
b. ALU

LOAD/STORE INSTRUCTIONS

- Read register operands
- Calculate address using 12-bit offset
  - Use ALU, but sign-extend offset
- Load: Read memory and update register
- Store: Write register value to memory



a. Data memory unit

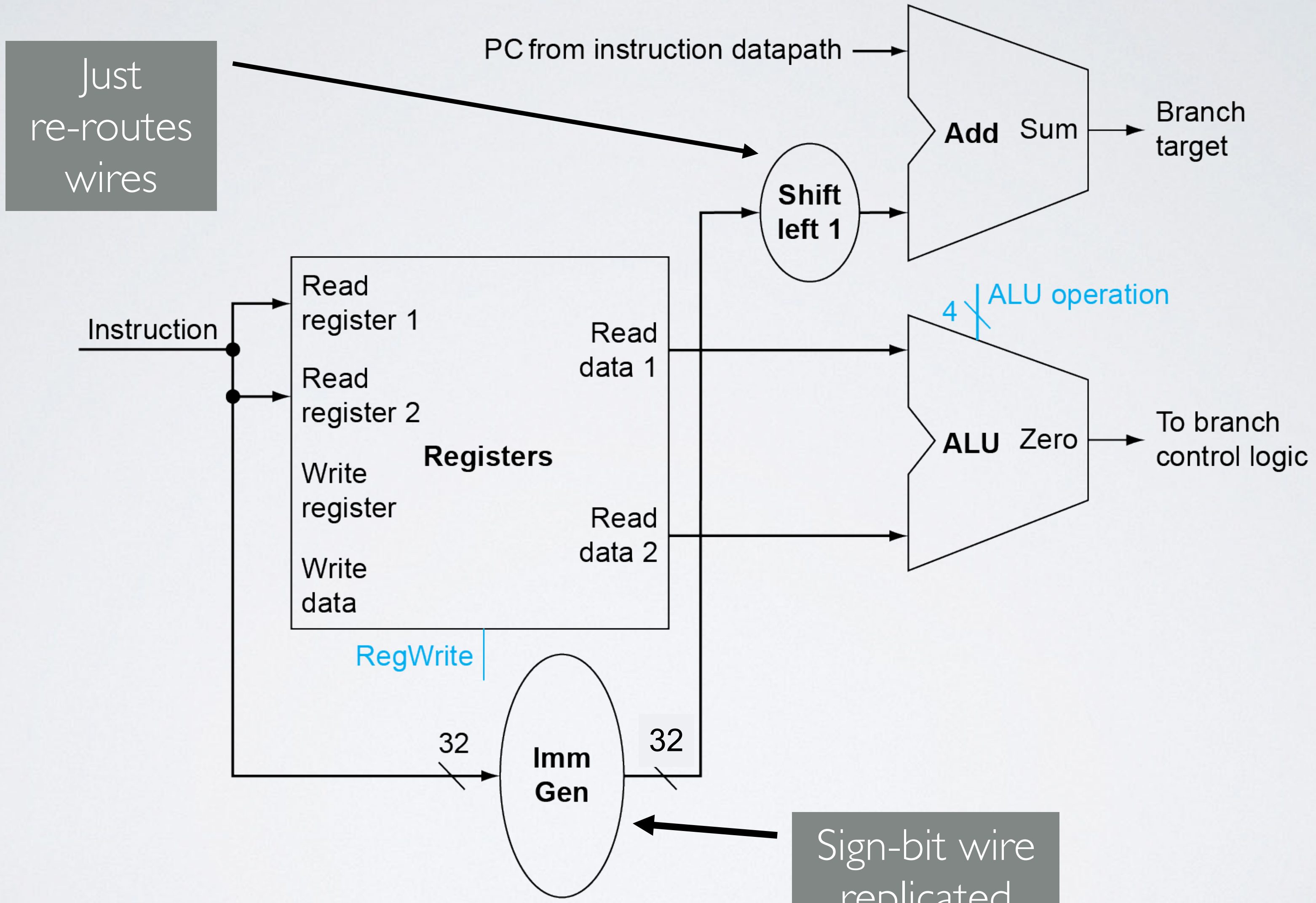


b. Immediate generation unit

BRANCH INSTRUCTIONS

- Read register operands
- Compare operands
  - Use ALU, subtract and check Zero output
- Calculate target address
  - Sign-extend displacement
  - Shift left 1 place (halfword displacement)
  - Add to PC value

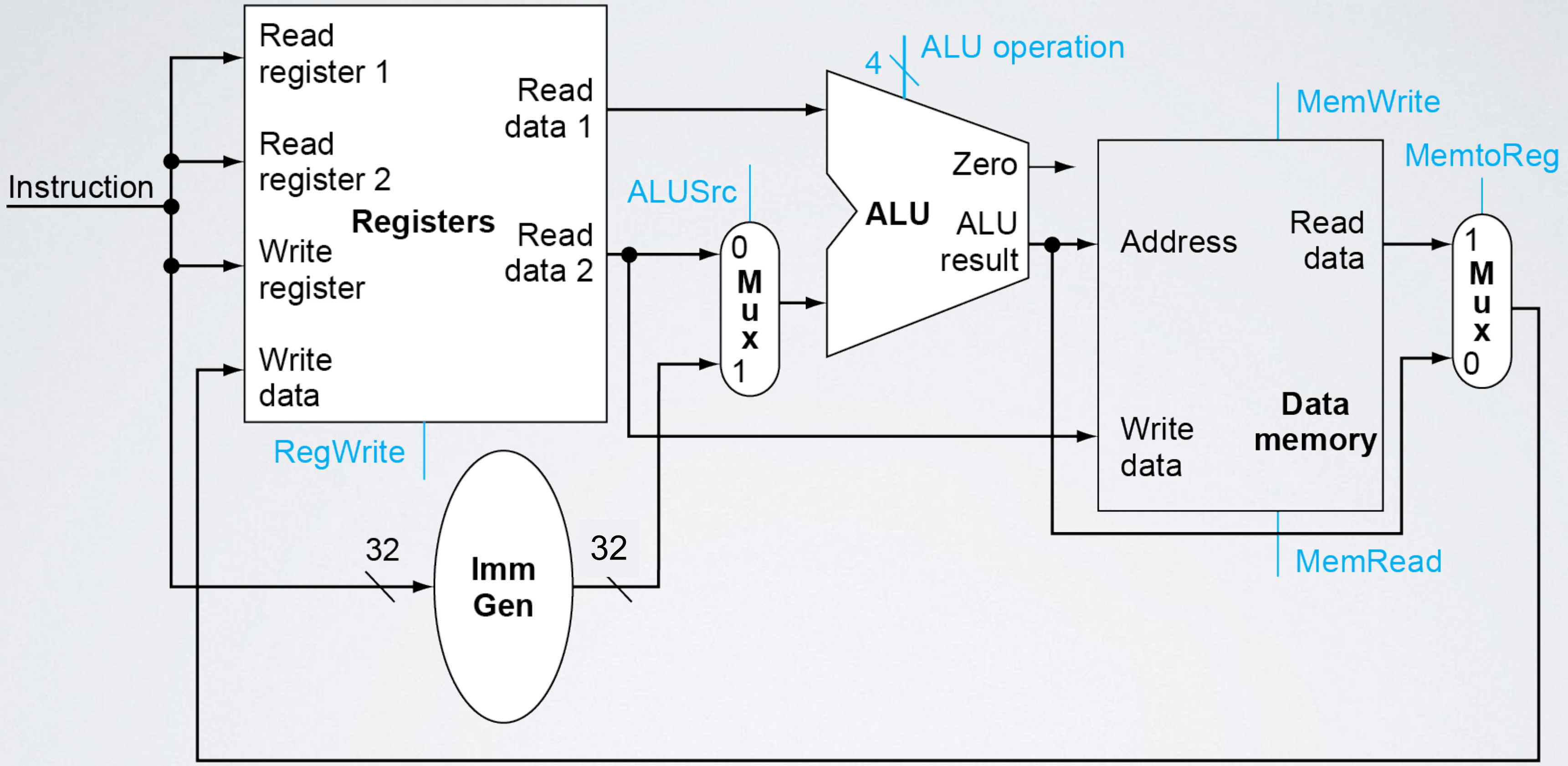
BRANCH INSTRUCTIONS



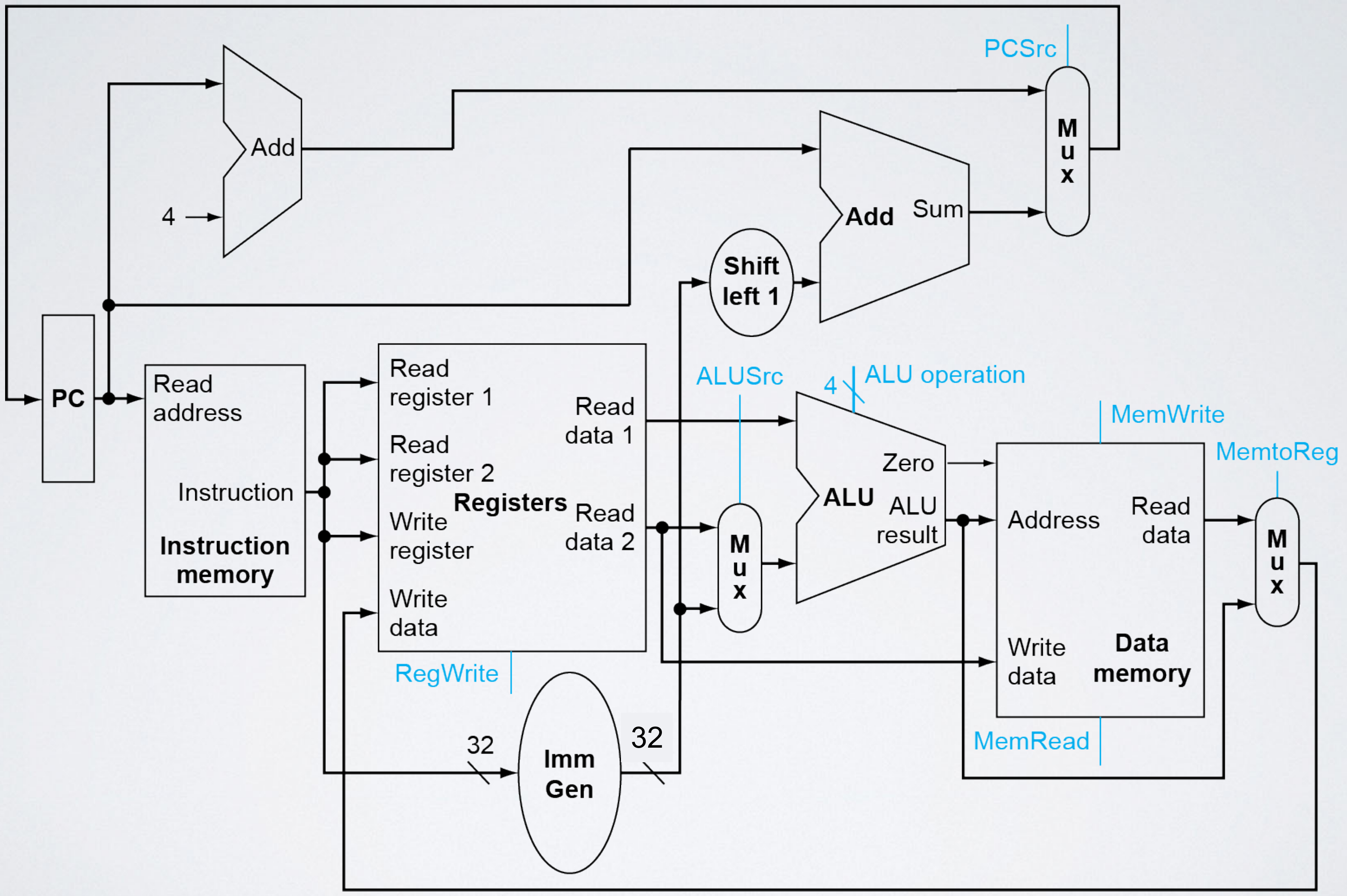
COMPOSING THE ELEMENTS

- First-cut data path does an instruction in one clock cycle
  - Each datapath element can only do one function at a time
  - Hence, we need separate instruction and data memories
- Use multiplexers where alternate data sources are used for different instructions

R-TYPE/LOAD/STORE DATAPATH



FULL DATAPATH



ALU CONTROL

- ALU used for
  - Load/Store: Function = add
  - Branch: Function = subtract
  - R-type: Function depends on opcode

ALU control	Function
0000	AND
0001	OR
0010	add
0110	subtract

ALU CONTROL

- Assume 2-bit ALUOp derived from opcode
  - Combinational logic derives ALU control

opcode	ALUOp	Operation	funct7	funct3	ALU function	ALU control
lw	00	load word	XXXXXXX	XXX	add	0010
sw	00	store word	XXXXXXX	XXX	add	0010
beq	01	branch equal	XXXXXXX	XXX	subtract	0110
R-type	10	add	0000000	000	add	0010
		subtract	0100000	000	subtract	0110
		AND	0000000	111	AND	0000
		OR	0000000	110	OR	0001

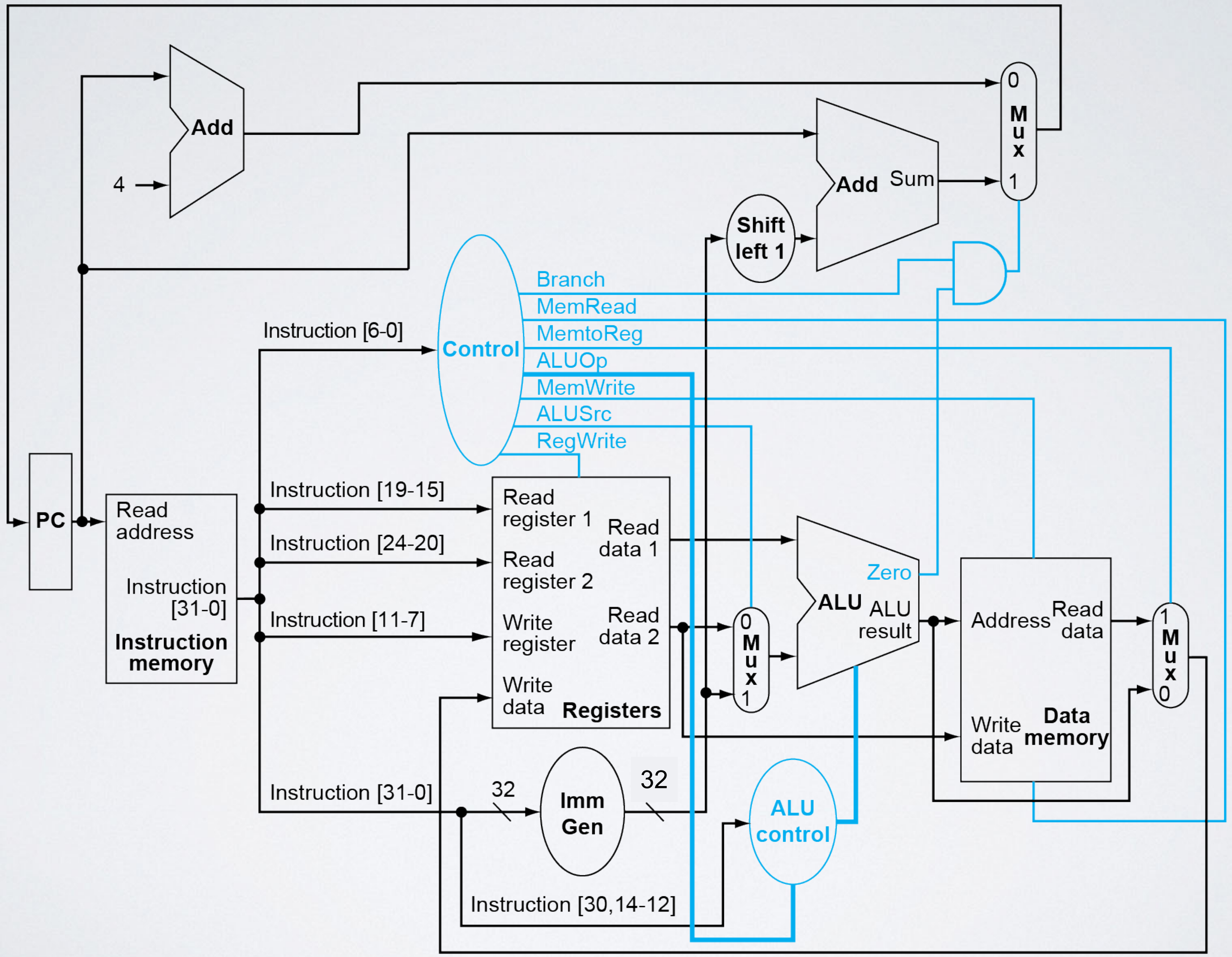
THE MAIN CONTROL UNIT

- Control signals derived from instruction

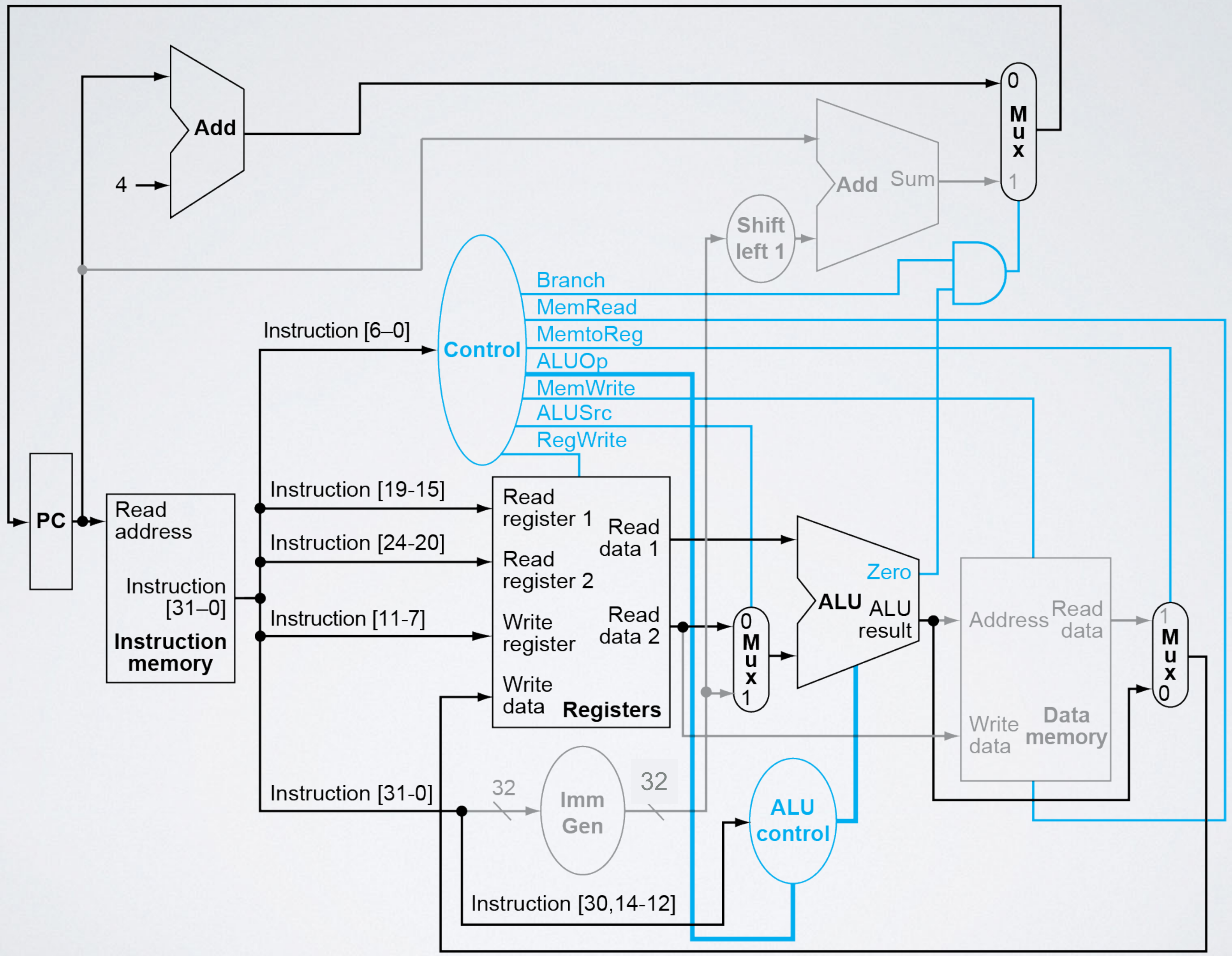
Name (Bit position)	31:25	24:20	19:15	14:12	11:7	6:0
(a) R-type	funct7	rs2	rs1	funct3	rd	opcode
(b) I-type	immediate[11:0]		rs1	funct3	rd	opcode
(c) S-type	immed[11:5]	rs2	rs1	funct3	immed[4:0]	opcode
(d) SB-type	immed[12,10:5]	rs2	rs1	funct3	immed[4:1,11]	opcode

ALUOp		Funct7 field							Funct3 field			
ALUOpl	ALUOp0	I[31]	I[30]	I[29]	I[28]	I[27]	I[26]	I[25]	I[14]	I[13]	I[12]	Operation
0	0	X	X	X	X	X	X	X	X	X	X	0010
X	1	X	X	X	X	X	X	X	X	X	X	0110
1	X	0	0	0	0	0	0	0	0	0	0	0010
1	X	0	1	0	0	0	0	0	0	0	0	0110
1	X	0	0	0	0	0	0	0	1	1	1	0000
1	X	0	0	0	0	0	0	0	1	1	0	0001

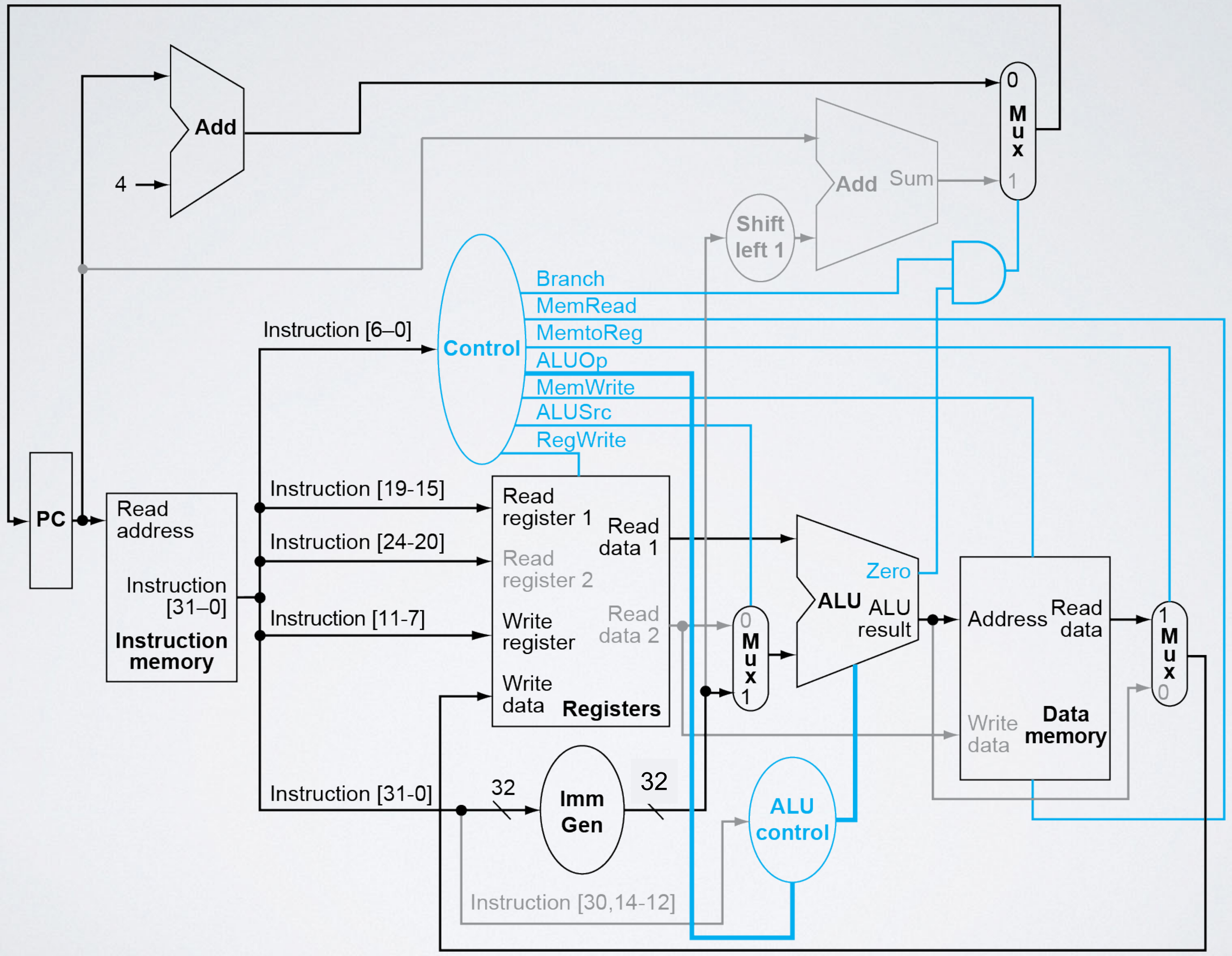
DATAPATH WITH CONTROL



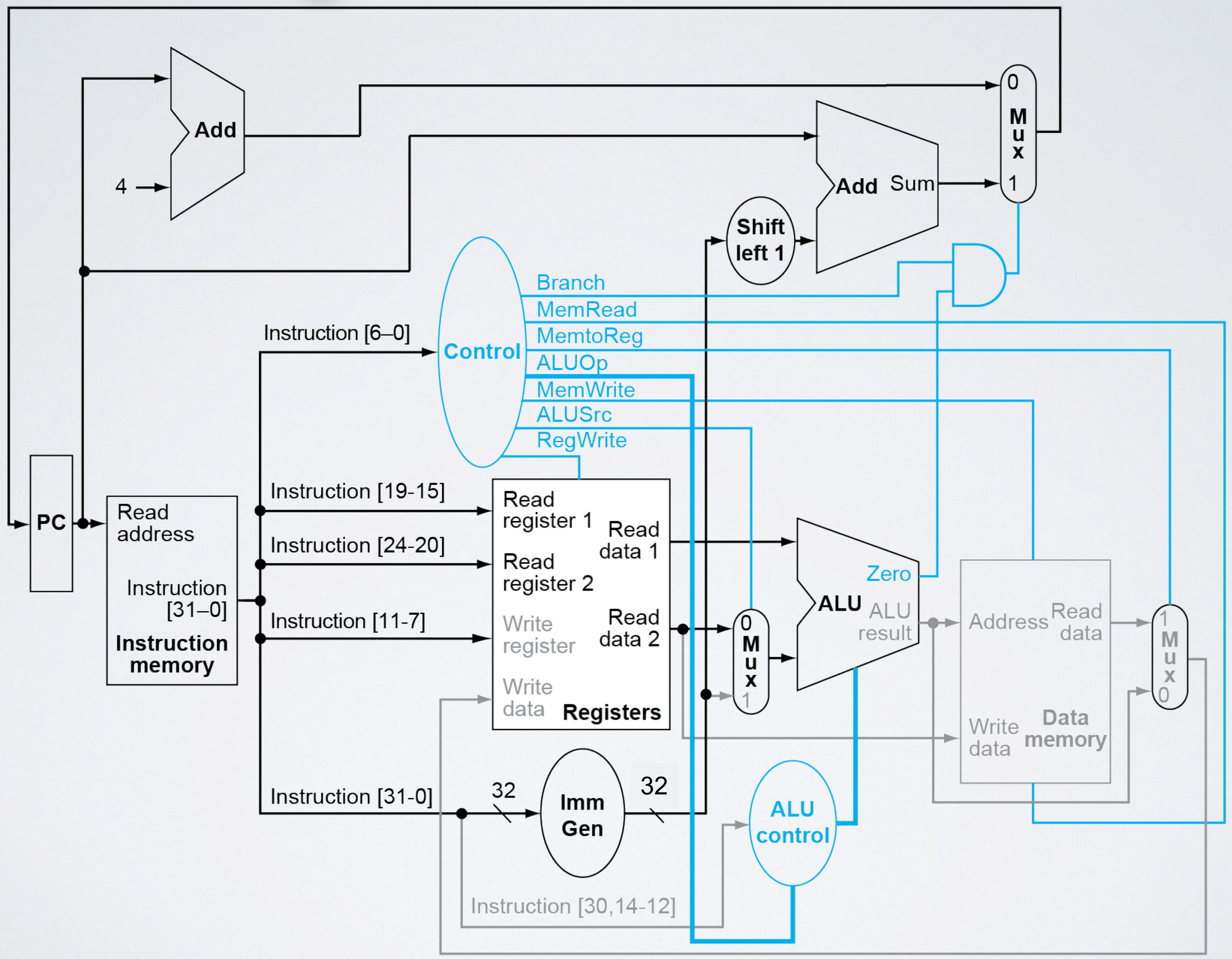
R-TYPE INSTRUCTION



LOAD INSTRUCTION



BEQ INSTRUCTION



PERFORMANCE ISSUES

- Longest delay determines clock period
  - Critical path: load instruction
  - Instruction memory → register file → ALU → data memory → register file
- Not feasible to vary period for different instructions
- Violates design principle
  - Making the common case fast
- We will improve performance by pipelining

REFERENCES

- David Patterson and John L. Hennessy, *Computer Organization and Design: RISC-V Edition*, Morgan Kaufmann, 2017