Computer Architecture
(ENE1004)

Lec – 12: The Processor (Chapter 4) - 3

Upcoming Schedule

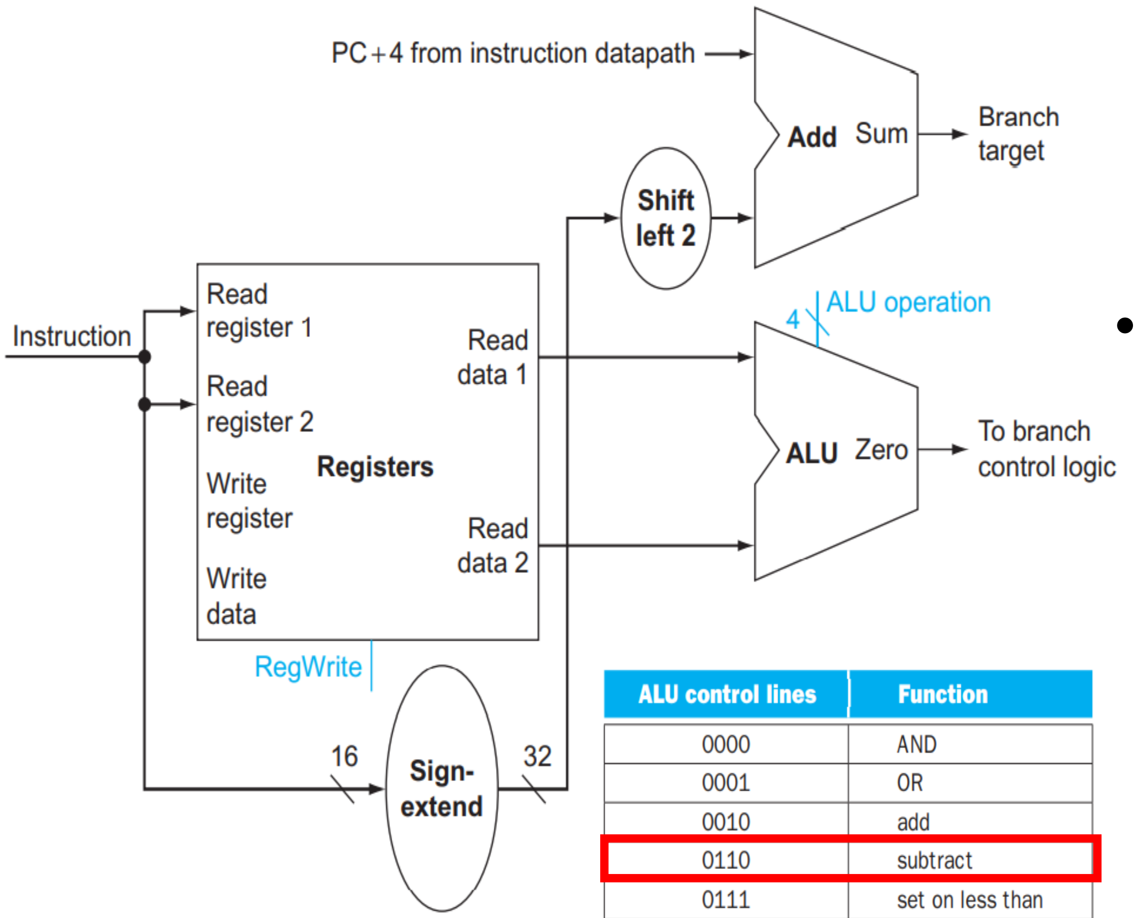
- Midterm exam
  - Apr. 24 (Mon), regular class hour
  - 60-70 minutes
- No class on Apr. 20 (Thur)
- Sample questions are provided on Apr. 21 (Fri)
- Assignment #1
  - By May 14 (Sun) at midnight
  - Your code will be compared with others (your classmates, previous semesters)

Datapath Elements for Branch

- Branch instructions compute their target addresses using PC-relative addressing
  - **beq $t1, $t2, offset_value**
  - Target address = ① address of the subsequent instruction + ② branch offset in bytes
  - Target address = ① (PC + 4) + ② (offset_value * 4)
- Let us consider what datapath elements branch instructions need
  - ① (PC + 4) can be obtained from the datapath (I) for fetching an instruction
  - ② (offset_value * 4) can be done by shifting left the offset_value by 2
- Branch instructions can have two different scenarios depending on the condition
- (1) If it is true, the next instruction is the instruction at the target address
  - We say that the branch is “taken”
  - PC ← target address
- (2) If it is not true, the next instruction is the instruction that follows sequentially
  - We say that the branch is “not taken”
  - PC ← PC + 4

Datapath Elements for Branch

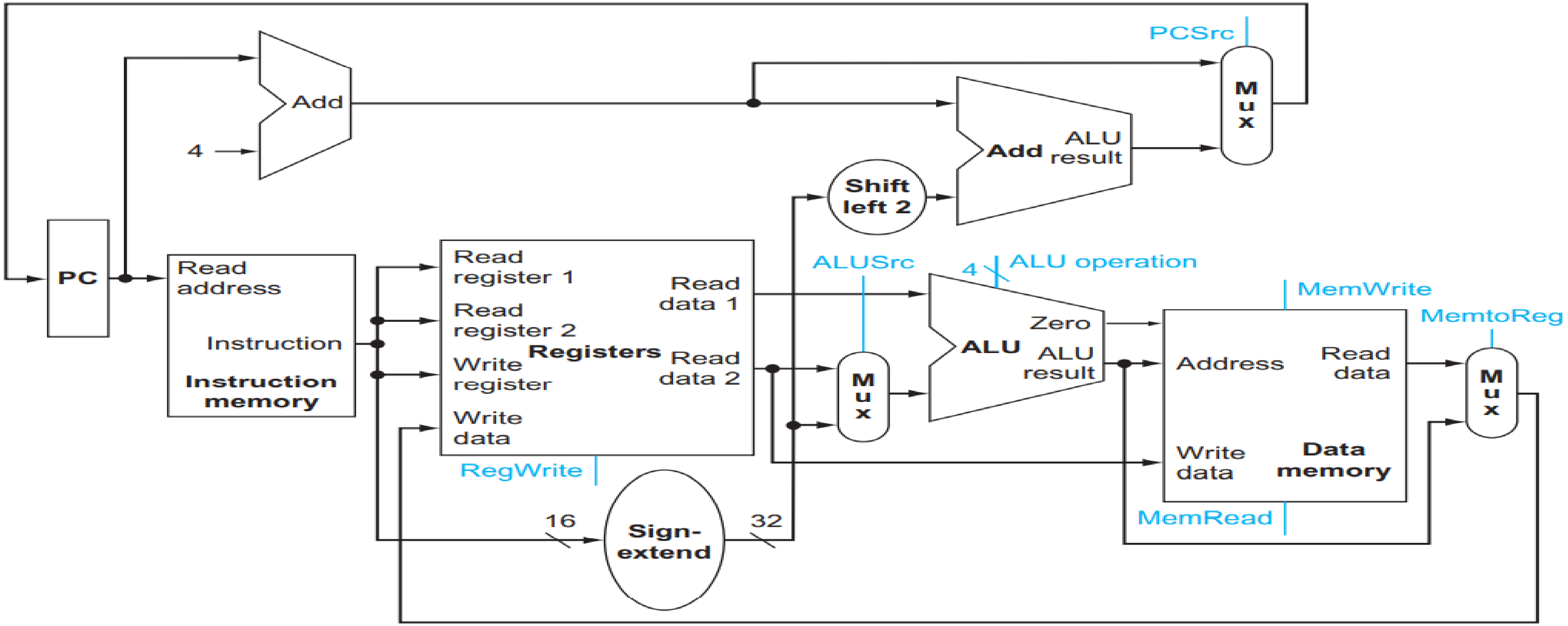
beq $t1, $t2, offset_value



ALU control lines	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set on less than
1100	NOR

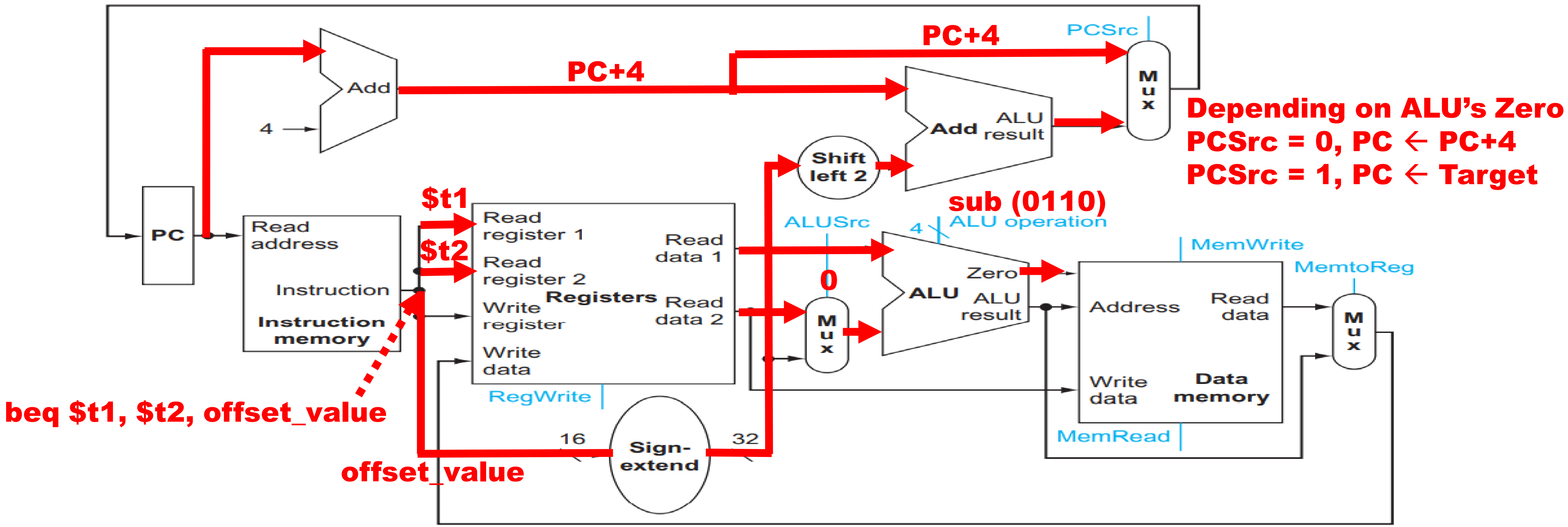
- (1) Computing the target address
  - Target address = (PC + 4) + (offset_value * 4)
  - **Sign extension unit** for (offset_value)
  - **Shift left 2 unit** for (offset_value * 4)
  - **Adder** for (PC + 4) + (offset_value * 4)
- (2) Comparing the register contents
  - **Register file** to supply two operands (**$t1** & **$t2**)
  - **ALU** for comparing the two operands (**$t1** vs **$t2**)
  - **ALU operation = 0110** for a subtract
  - If **Zero = 1** (two values are equal), the instruction in the target address should be executed next
  - If **Zero = 0** (two values are not equal), the instruction that follows sequentially should be executed next

A Single Datapath for Fetching + R + Load/Store + Branch



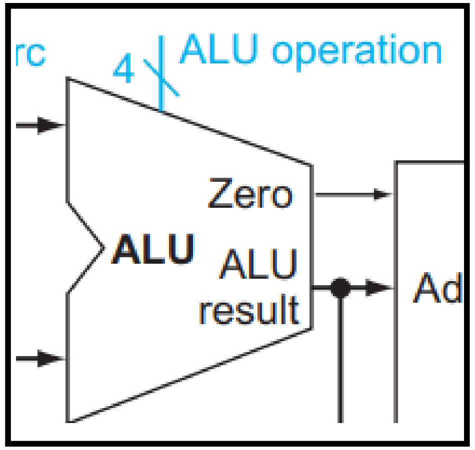
- (I) For fetching instructions, we need PC, Instruction memory, Adder
- (II) For R-type instructions, we need Register file and ALU
- (III) For load/store instructions, we need Register file, ALU, Sign extension unit, Data memory
- (IV) For branch instructions, we need Register file, ALU, Sign extension unit, Shift left 2 unit, Adder

Execution of **Branch** Instruction on the Datapath



- (1) Computing the target address: (PC+4) + (**offset_value***4) using sign extension/shift left 2/adder
- (2) At the same time, comparing the two register operands: (**$t1** vs **$t2**) using register file/ALU
- The result of (2) is evaluated whether it is zero or not using **ALU's Zero**
- Based on the evaluation, either (PC+4) or {(PC+4)+(offset_value*4)} is selected using **PCSrc = 0/1**

ALU Control Unit



ALU control lines	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set on less than
1100	NOR

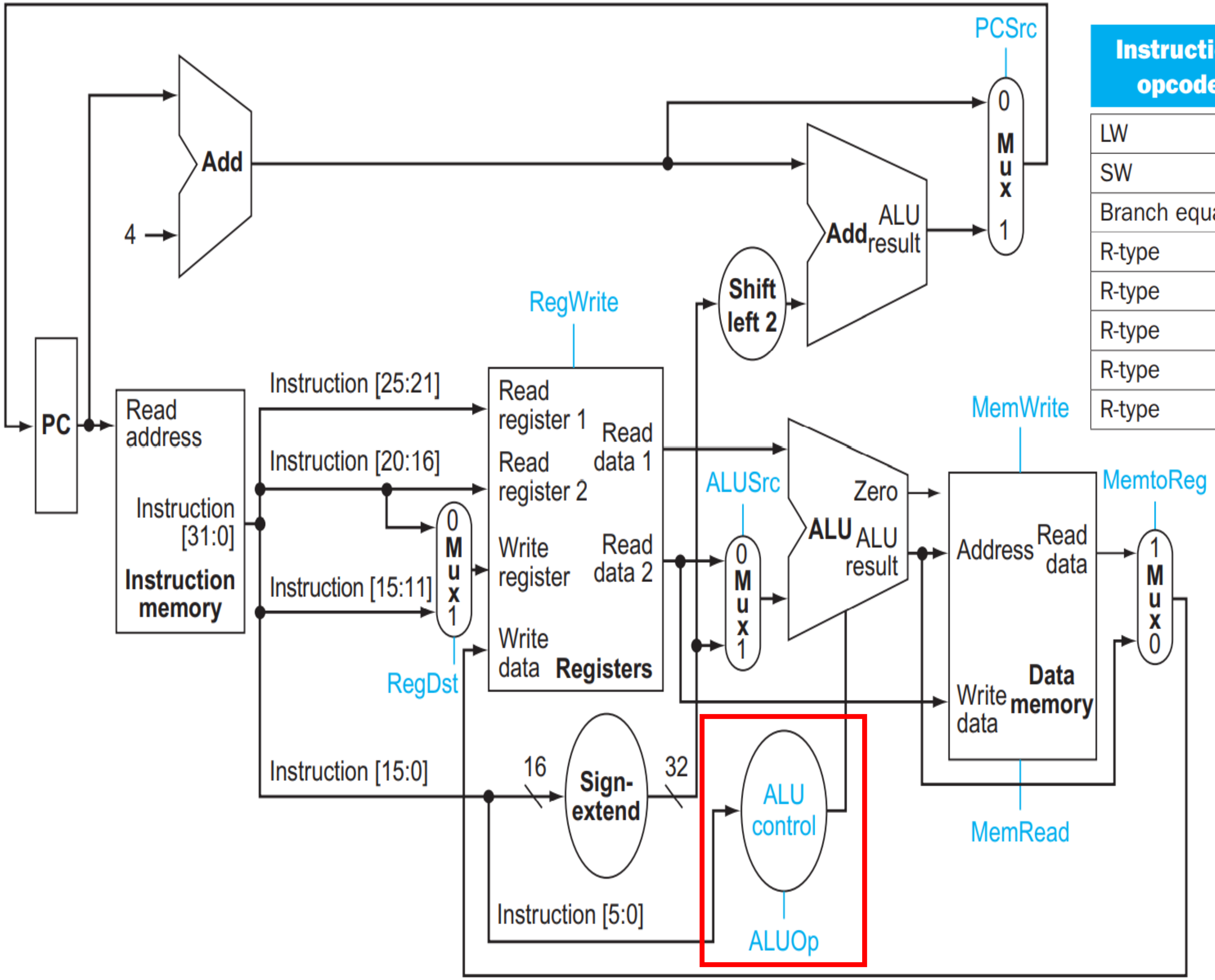
000000	rs	rt	rd	shamt	funct
31:26	25:21	20:16	15:11	10:6	5:0

op(31:26)=000000 (R-format), funct(5:0)								
2–0 5–3	0(000)	1(001)	2(010)	3(011)	4(100)	5(101)	6(110)	7(111)
0(000)	shift left logical		shift right logical	sra	sllv		srlv	srav
1(001)	jump register	jalr			syscall	break		
2(010)	mfhi	mthi	mflo	mtlo				
3(011)	mult	multu	div	divu				
4(100)	add	addu	subtract	subu	and	or	xor	not or (nor)
5(101)			set l.t.	set l.t. unsigned				
6(110)								
7(111)								

[R-type encoding]

- ALU performs one of these functions
  - Load/store instructions use “addition” to compute the memory address
  - Branch-equal instruction uses “subtraction” to compare two register values
  - R-type instructions selects “actions”, depending on the value of the 6-bit funct field
- We need a “control unit” that determines what function ALU performs in datapath
  - Input (2 bits) – which type is this instruction, load/store, branch-equal, or R-type?
  - Input (6 bits) – what is the value of the funct filed if it is R-type?
  - Output (4 bits) – what function the ALU performs?

ALU Control Unit



Input(2-bit) Input(6-bit) Output(4-bit)

Instruction opcode	ALUOp	Instruction operation	Funct field	Desired ALU action	ALU control input
LW	00	load word	XXXXXX	add	0010
SW	00	store word	XXXXXX	add	0010
Branch equal	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
R-type	10	subtract	100010	subtract	0110
R-type	10	AND	100100	AND	0000
R-type	10	OR	100101	OR	0001
R-type	10	set on less than	101010	set on less than	0111

- ALUOp is determined by instruction types
  - 00 for load/store instructions
  - 01 for branch-equal instruction
  - 10 for R-type instructions
- Funct field is extracted from instruction
- Based on **ALUOp** and **Instruction[5:0]**, ALU control unit determines the action to be performed by the ALU

Datapath for Formats of Instructions

Field	0	rs	rt	rd	shamt	funct
Bit positions	31:26	25:21	20:16	15:11	10:6	5:0

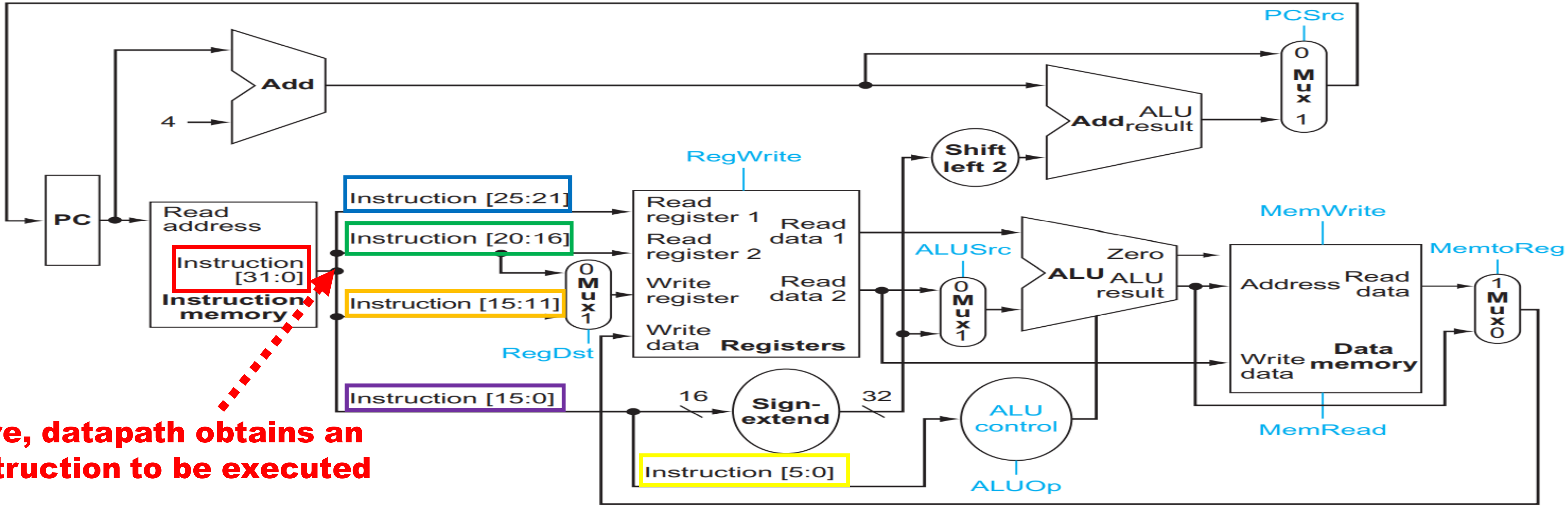
a. R-type instruction

Field	35 or 43	rs	rt	address
Bit positions	31:26	25:21	20:16	15:0

b. Load or store instruction

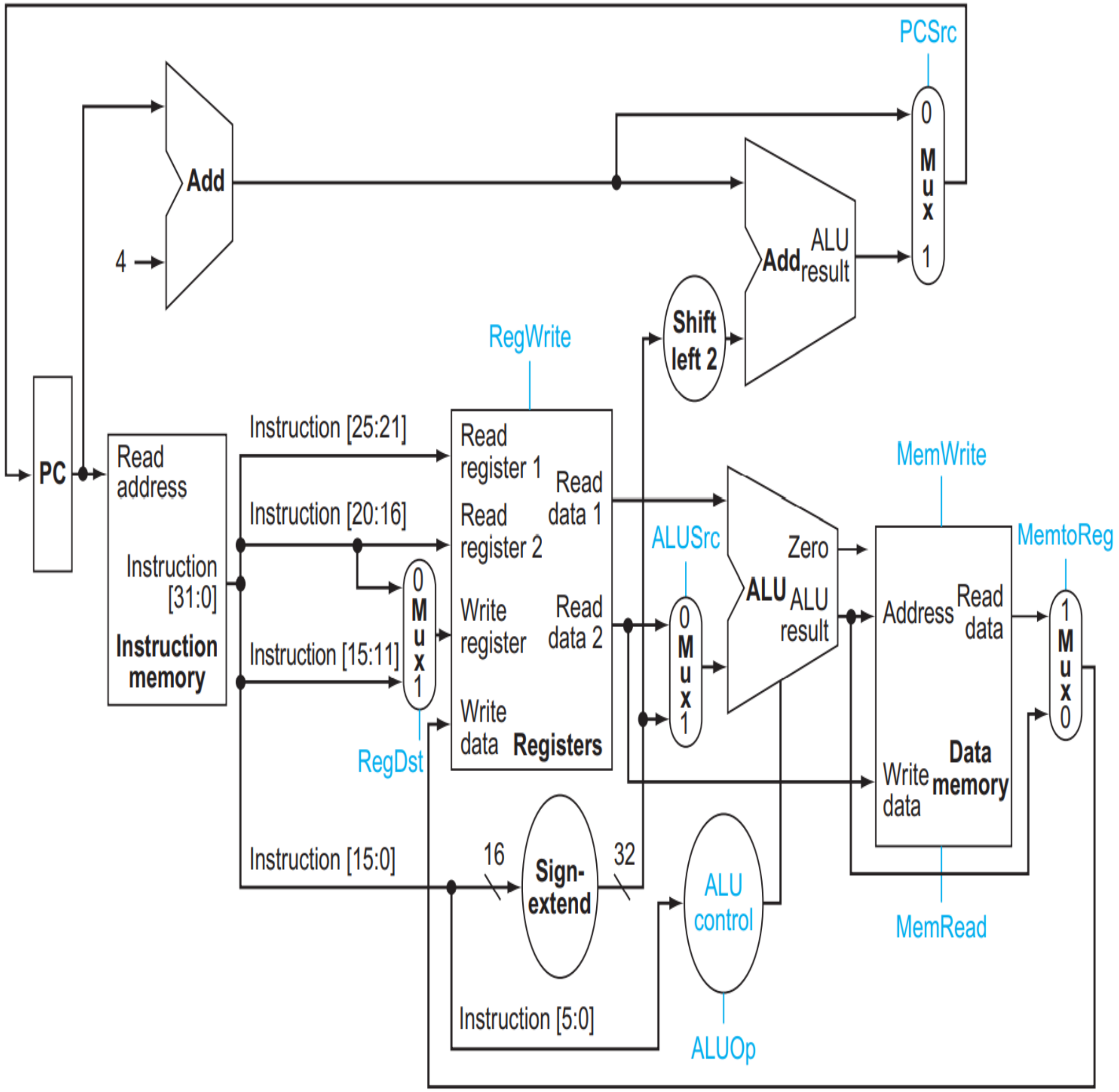
Field	4	rs	rt	address
Bit positions	31:26	25:21	20:16	15:0

c. Branch instruction



Here, datapath obtains an instruction to be executed

Control Signals



Signal name	Effect when deasserted	Effect when asserted
RegDst	The register destination number for the Write register comes from the rt field (bits 20:16).	The register destination number for the Write register comes from the rd field (bits 15:11).
RegWrite	None.	The register on the Write register input is written with the value on the Write data input.
ALUSrc	The second ALU operand comes from the second register file output (Read data 2).	The second ALU operand is the sign-extended, lower 16 bits of the instruction.
PCSrc	The PC is replaced by the output of the adder that computes the value of PC + 4.	The PC is replaced by the output of the adder that computes the branch target.
MemRead	None.	Data memory contents designated by the address input are put on the Read data output.
MemWrite	None.	Data memory contents designated by the address input are replaced by the value on the Write data input.
MemtoReg	The value fed to the register Write data input comes from the ALU.	The value fed to the register Write data input comes from the data memory.