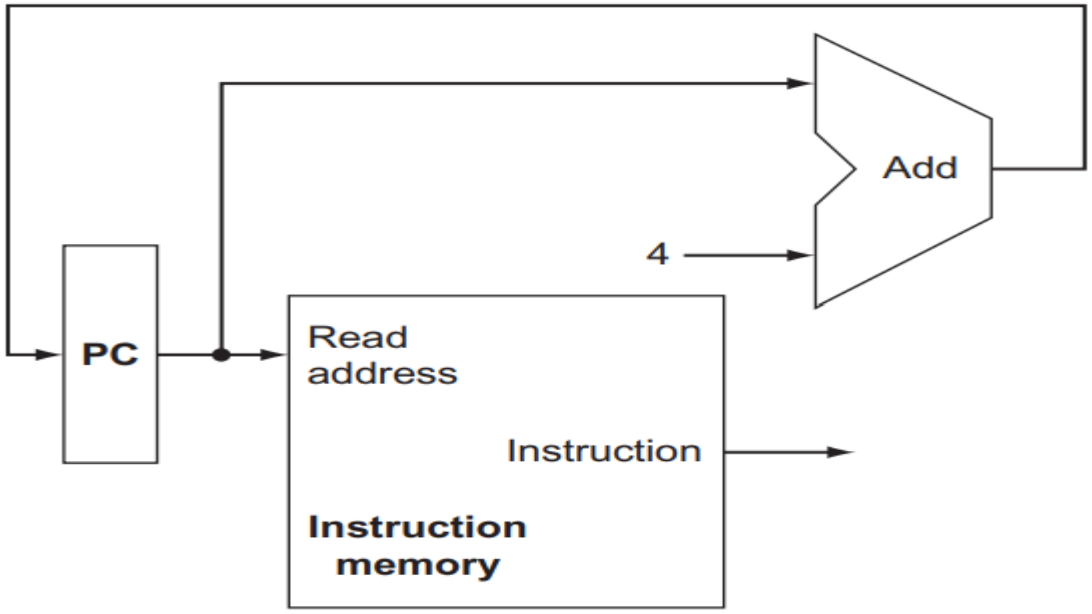
Computer Architecture
(ENE1004)

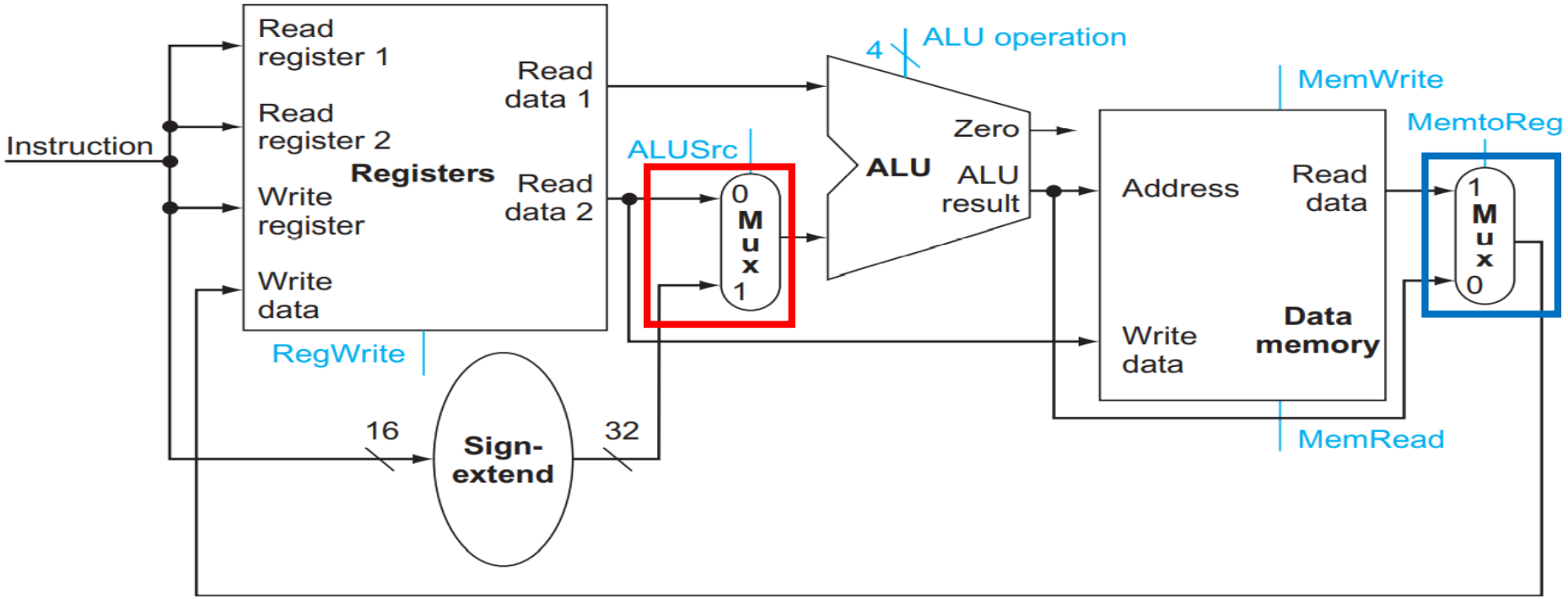
Lec – 11: The Processor (Chapter 4) - 2

A Datapath for Fetching Instructions



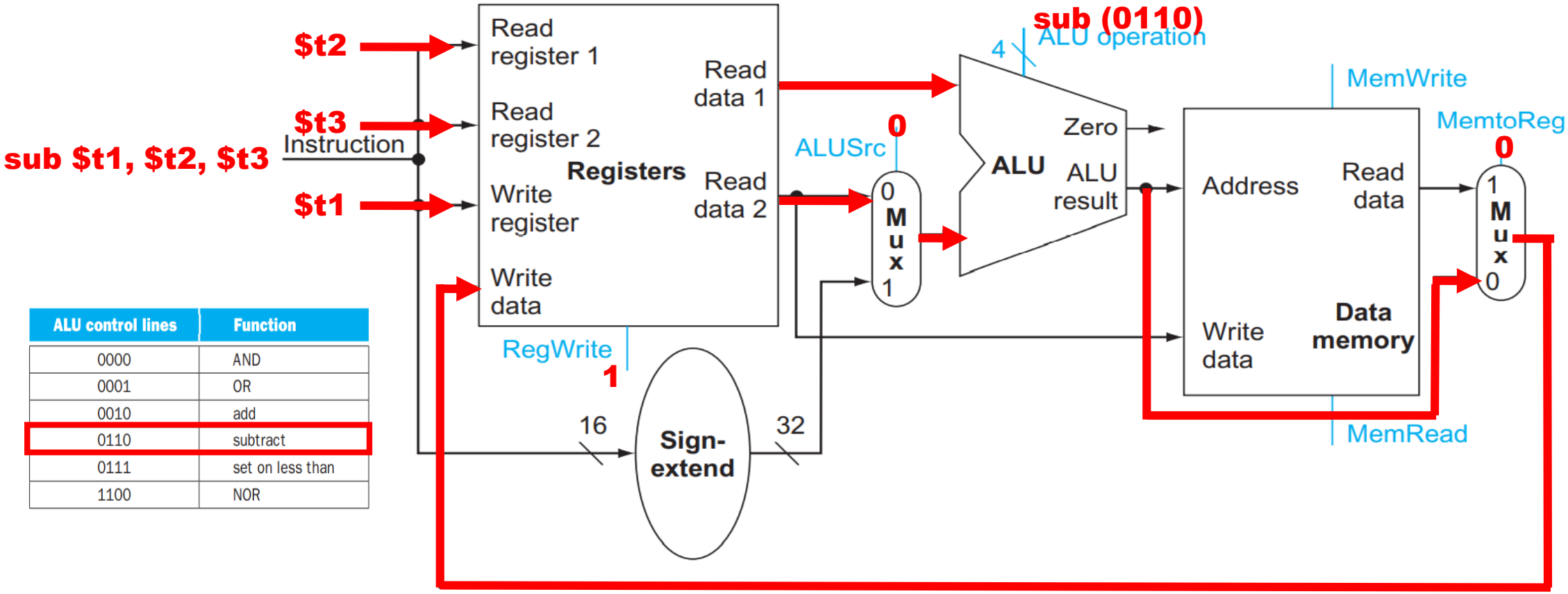
- (1) To execute an instruction, we must “fetch” the instruction from memory
  - The processor obtains an instruction when the address of PC is given to the instruction memory
- (2) To prepare for the execution of the next instruction, we must also “update (increment) the PC” so that it points to the next (sequential) instruction
  - The size of an instruction is 4 bytes
  - For jump and branch instructions, PC will point at the target address (instruction)

A Datapath for R-Type and Load/Store



- We combine (II) + (III) into a single datapath to execute any of the two types
  - R-type (e.g., **add $t1, $t2, $t3**): register file, ALU
  - Load/store (e.g., **lw $t1, 100($t2)**): register file, ALU, sign extension unit, data memory
- We add multiplexer (**Mux**) to select the required path for a given instruction
  - **Mux** for an input of the ALU: data from a register? or immediate from the instruction?
  - **Mux** for the data input to the register file: data from the data memory? or result of the ALU?

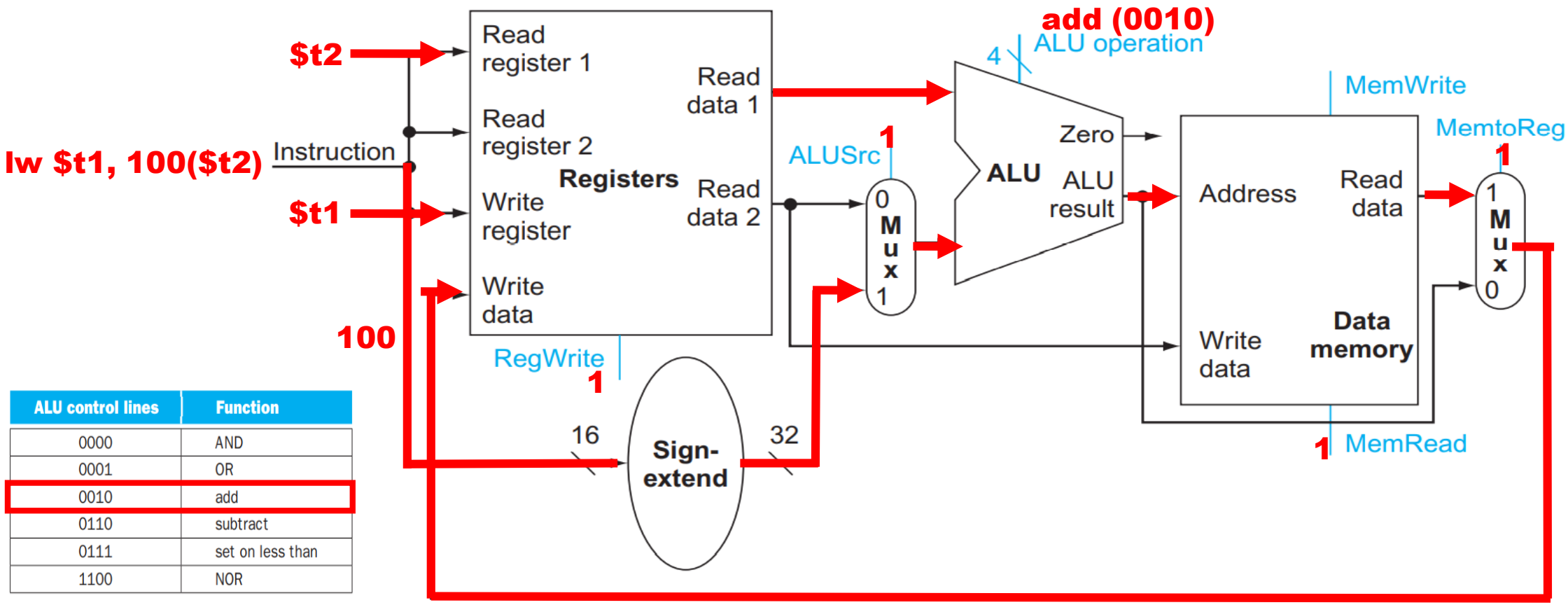
Execution of **R-Type** on the Datapath



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

- **ALUSrc = 0** to select **Read data 2** of the register file to an input of ALU
- **ALU operation = 0110** to perform **sub** instruction on ALU
- **MemtoReg = 0** to select the data value of **ALU result** to supply **Write data** of the register file
- When writing into a register, **RegWrite = 1**

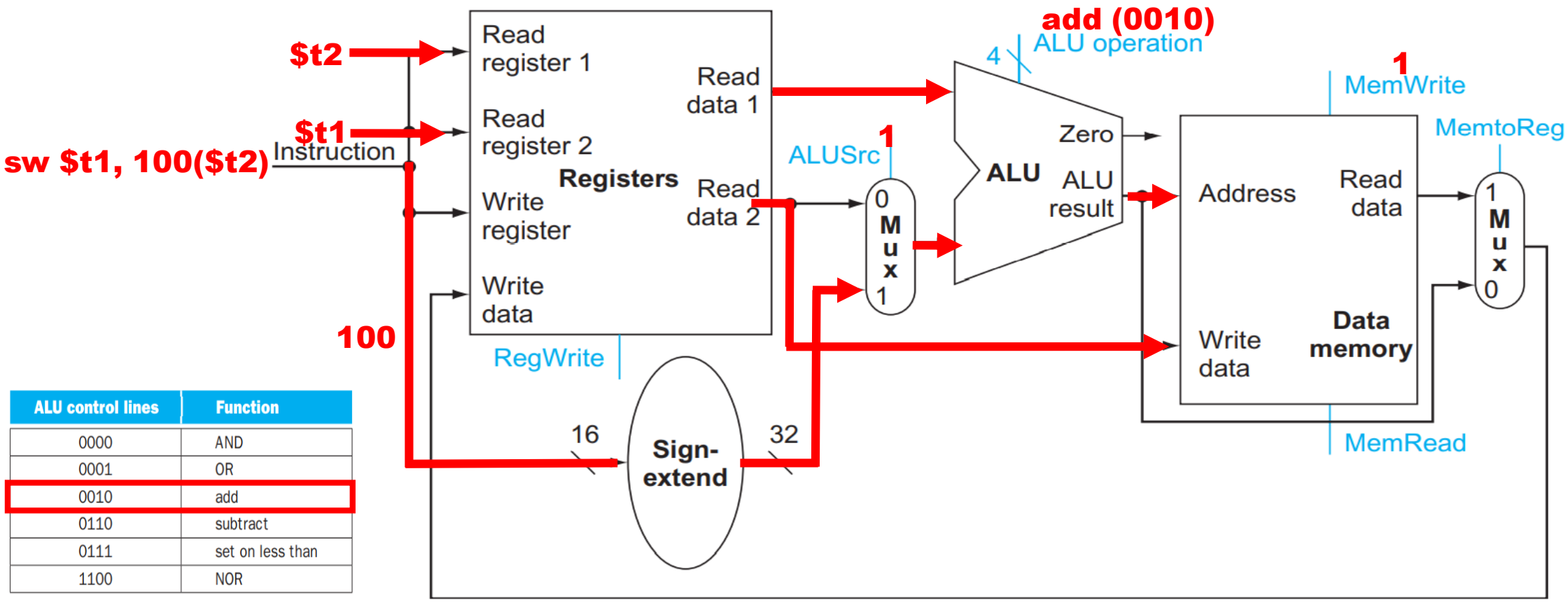
Execution of **Load** Instruction on the Datapath



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

- **ALUSrc = 1** to select **a sign-extended immediate value** to an input of ALU
- **ALU operation = 0010** to perform **add** instruction on ALU
- **MemRead = 1** to read data from the data memory
- **MemtoReg = 1** to select the data read from the data memory to supply **Write data** of the register file
- When writing a register, **RegWrite = 1**

Execution of **Store** Instruction on the Datapath



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

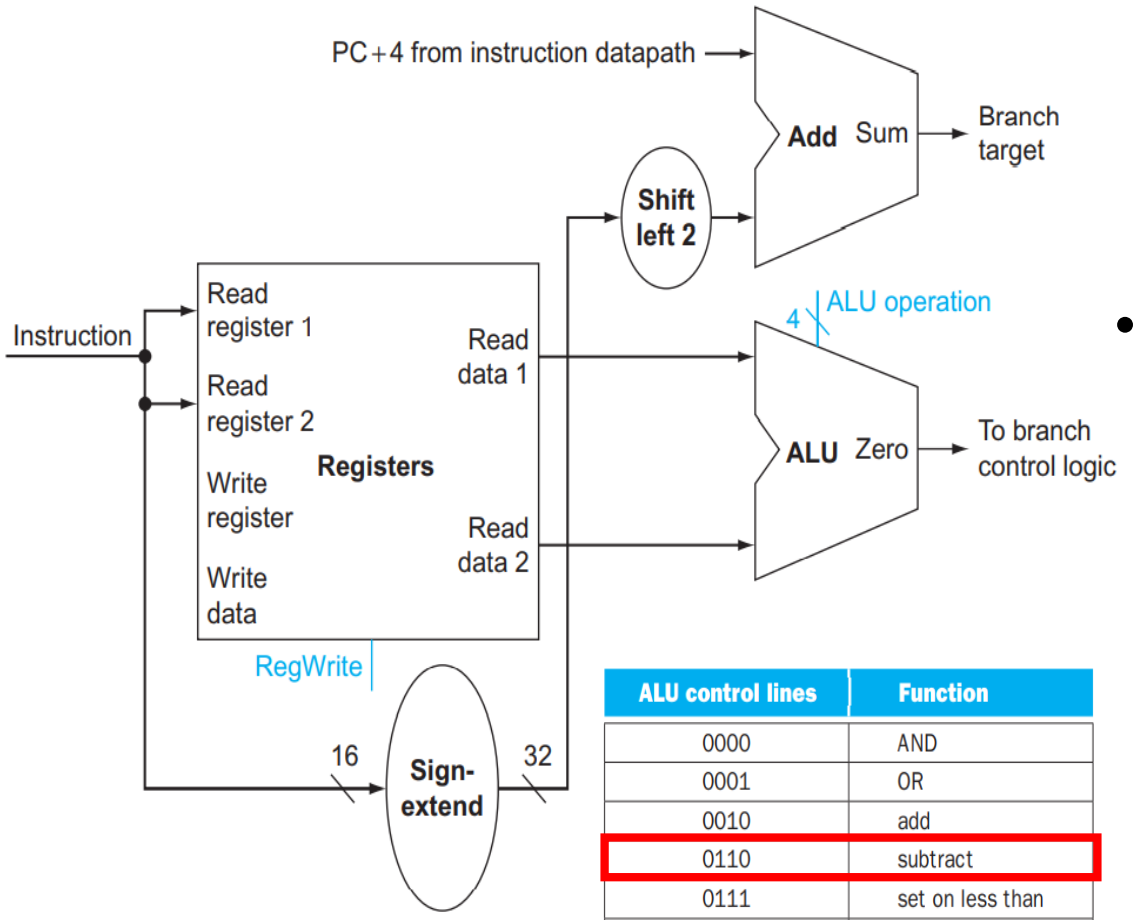
- **ALUSrc = 1** to select **a sign-extended immediate value** to an input of ALU
- **ALU operation = 0010** to perform **add** instruction on ALU
- **MemWrite = 1** to write data into the data memory

Datapath Elements for Branch

- A branch instruction computes its 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 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

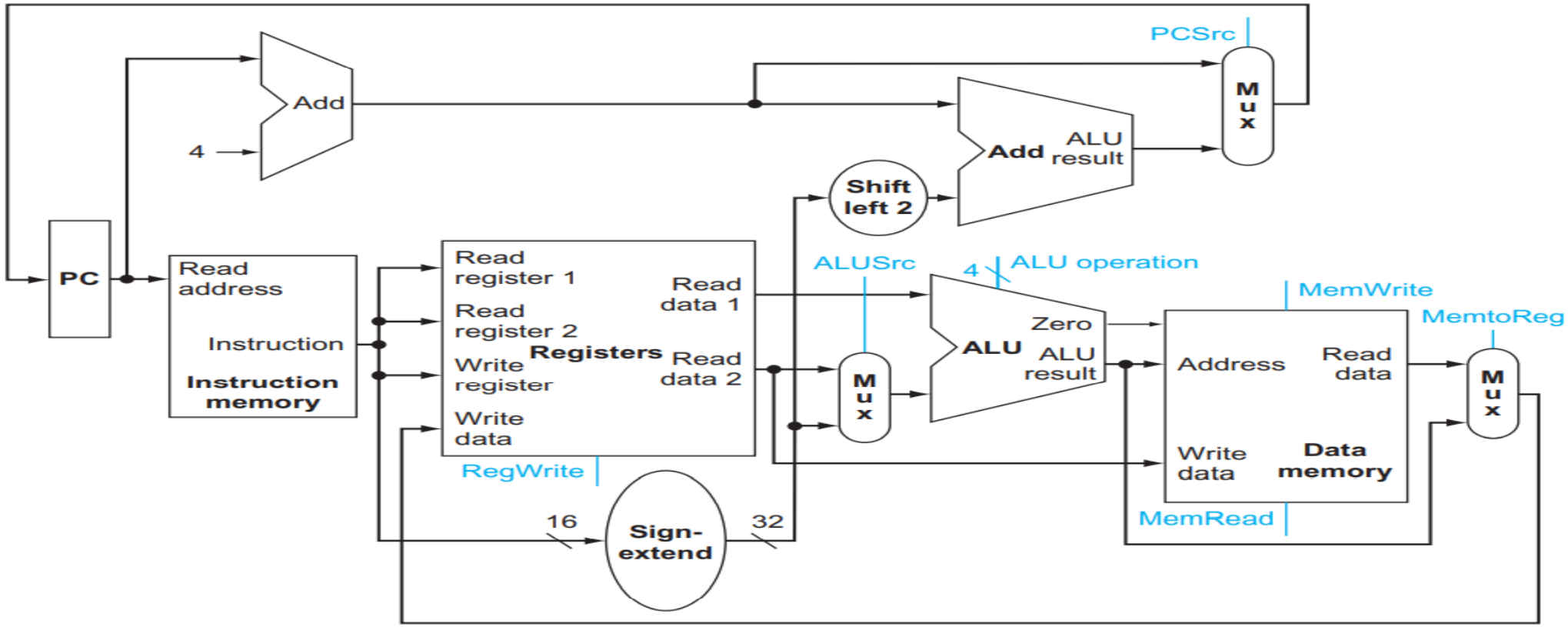
beq $t1, $t2, offset_value



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

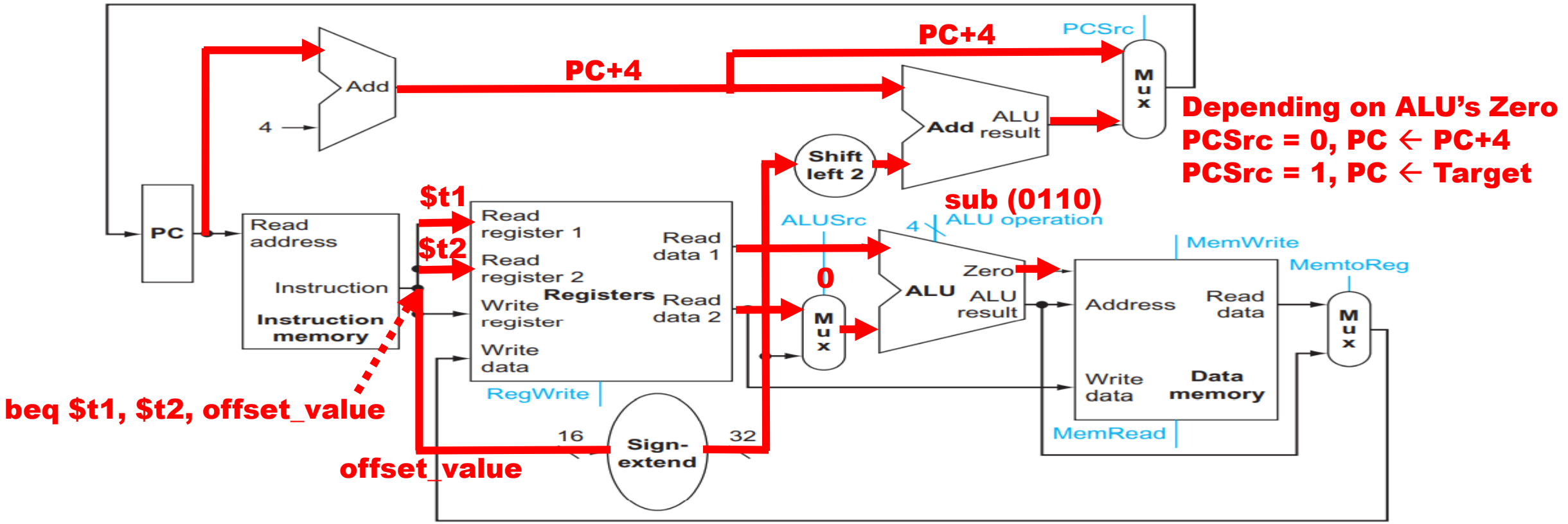
- (1) Computing the target address
  - Target address = (PC + 4) + (offset_value * 4)
  - **Sign extension unit** for (offset_value)
  - **Shift left 2 unit** for (offset_value * 4)
  - **Adder** for (PC + 4) + (offset_value * 4)
- (2) Comparing the register contents
  - **Register file** to supply two 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 that Combines R, I, and Branch



- For fetching instructions, we need PC, Instruction memory, Adder
- For R-type instructions, we need Register file and ALU
- For load/store instructions, we need Register file, ALU, Sign extension unit, Data memory
- 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**