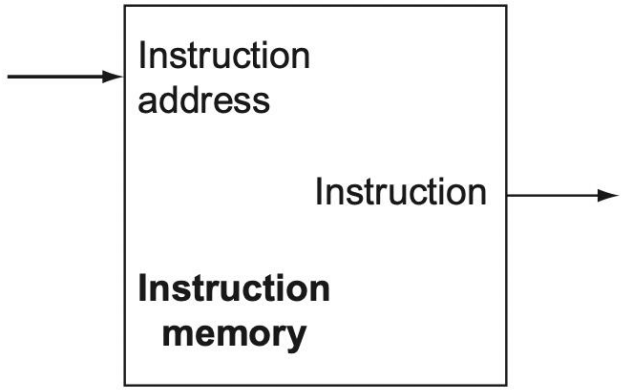
Computer Architecture
(ENE1004)

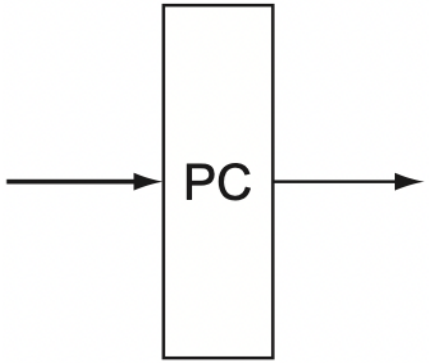
Lec – 10: The Processor (Chapter 4) - 1

(I) Datapath Elements for an Instruction

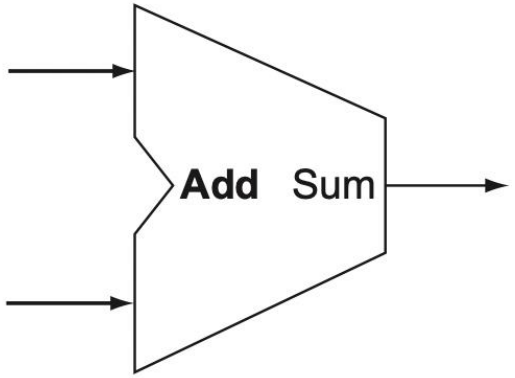
First, processors should know which instruction they will execute now



[Instruction memory]



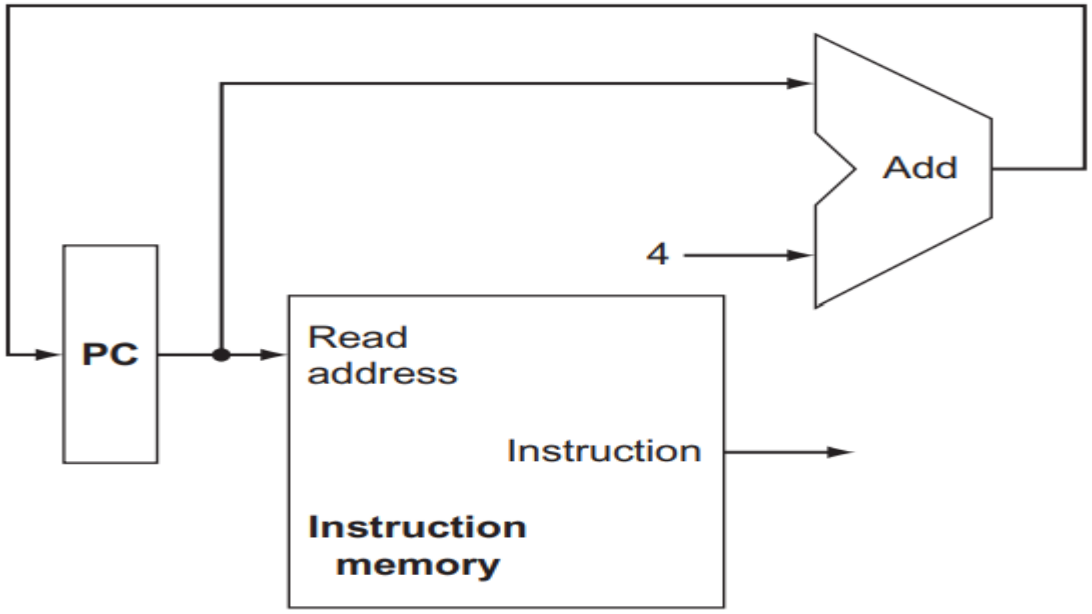
[Program counter]



[Adder]

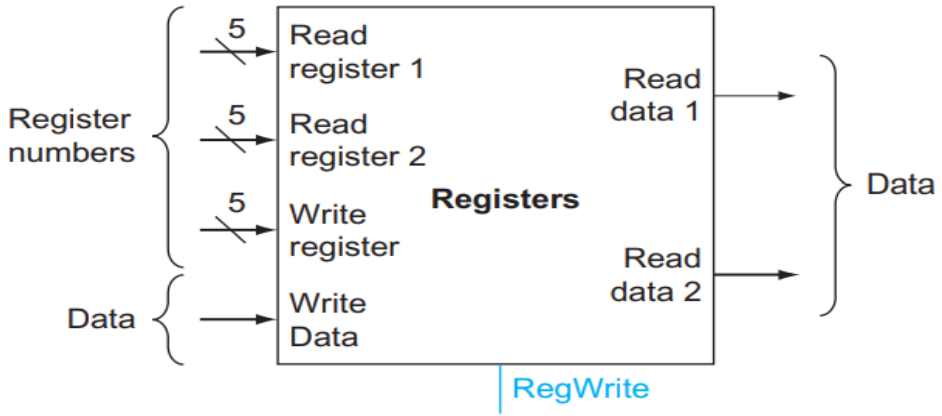
- **Instruction memory**: a memory unit to store the instructions of a program
  - Given an address (input), it supplies the corresponding instruction (output)
  - This is actually part of the memory (text segment)
- **Program counter (PC)**: a register that holds the address of the current instruction
- **Adder**: a logic that increments the PC for the address of the next instruction
  - This is actually built from the arithmetic logic unit (ALU) that performs arithmetic/bitwise ops

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)

(II) Datapath Elements for R-Type



[Register file]

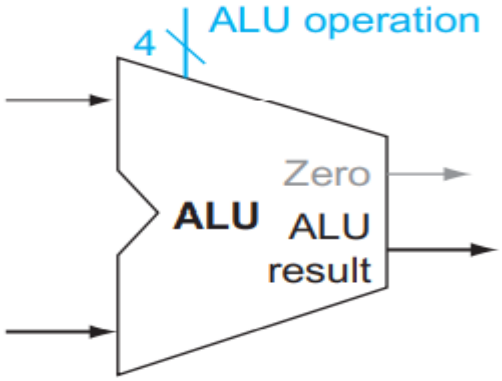
* Source 1 register (read):
Input (Read register 1) → Output (Read data 1)

* Source 2 register (read):
Input (Read register 2) → Output (Read data 2)

* Destination register (write):
Inputs (Write register & Write data)
& signal (RegWrite) = 1 → No output

- R-type instructions read two registers, perform an ALU operation on the contents of the registers, and write the result to a register
  - E.g., **add**, **sub**, **and**, **or**, **slt**
- **Register file**: a collection of the processor's 32 general-purpose registers
  - For two data words to be read, it provides two inputs that specify the register numbers (Read registers 1 & 2) and two outputs that carry the values of the registers (Read data 1 & 2)
  - To write a data word, it provides two inputs: one to specify the register number to be written (Write register) and one to supply the data to be written (Write data)
  - A write is controlled by a control signal (RegWrite), which must be asserted for a write, while reads are always serviced on the Read register inputs
  - Register inputs are 5 bits wide; data input/output buses are each 32 bits wide

(II) Datapath Elements for R-Type

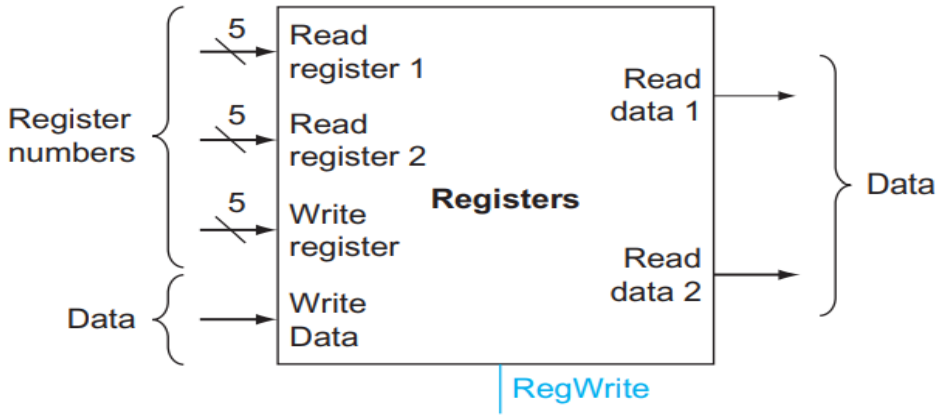


[Arithmetic Logic Unit (ALU)]

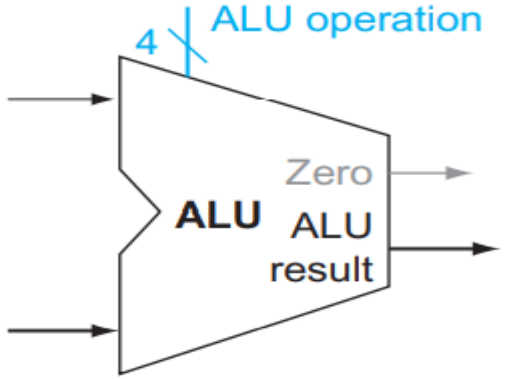
ALU control lines	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set on less than
1100	NOR

- **ALU**: a unit that can perform various arithmetic, logical operations on given inputs
  - It takes two 32-bit operands as inputs
  - It produces a 32-bit result (**ALU result**) as an output
  - It produces a 1-bit signal (**Zero**) if the result is 0
  - The 4-bit control signal (**ALU operation**) determines what operation the ALU performs

(III) Datapath Elements for Load/Store



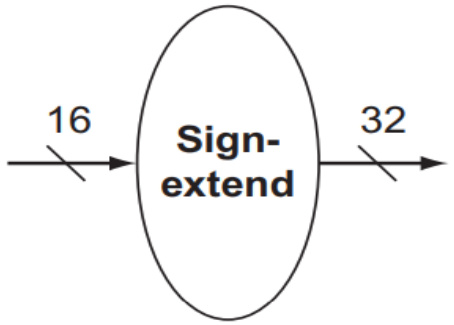
[Register file]



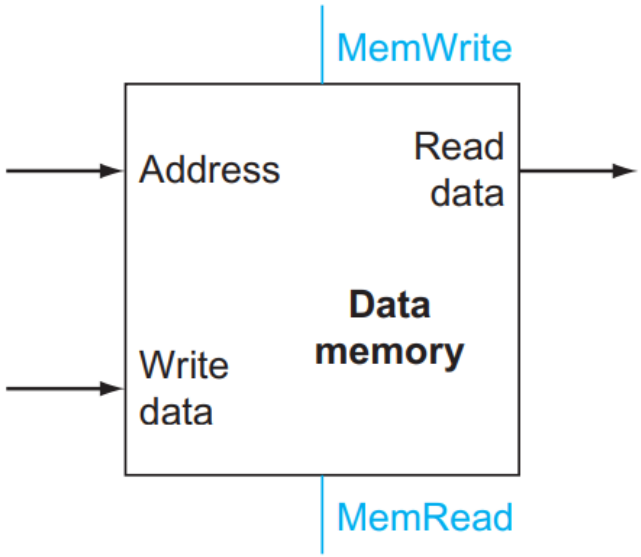
[Arithmetic Logic Unit (ALU)]

- A load/store instruction computes a memory address by adding the base register to the 16-bit signed offset field contained in the instructions
  - E.g., **lw $t1, offset_value($t2)**
  - E.g., **sw $t1, offset_value($t2)**
  - The value of **$t2** is read from the register file
  - **$t2 + offset_value** can be done by the ALU
- For a load instruction (**lw**), the value read from memory must be written into the register file (**$t1**)
  - Write register, write data, & write signal should be set
- For a store instruction (**sw**),the value to be stored must be read from the register file (**$t1**)
  - Note that **$t2** is read for computing the memory address

(III) Datapath Elements for Load/Store



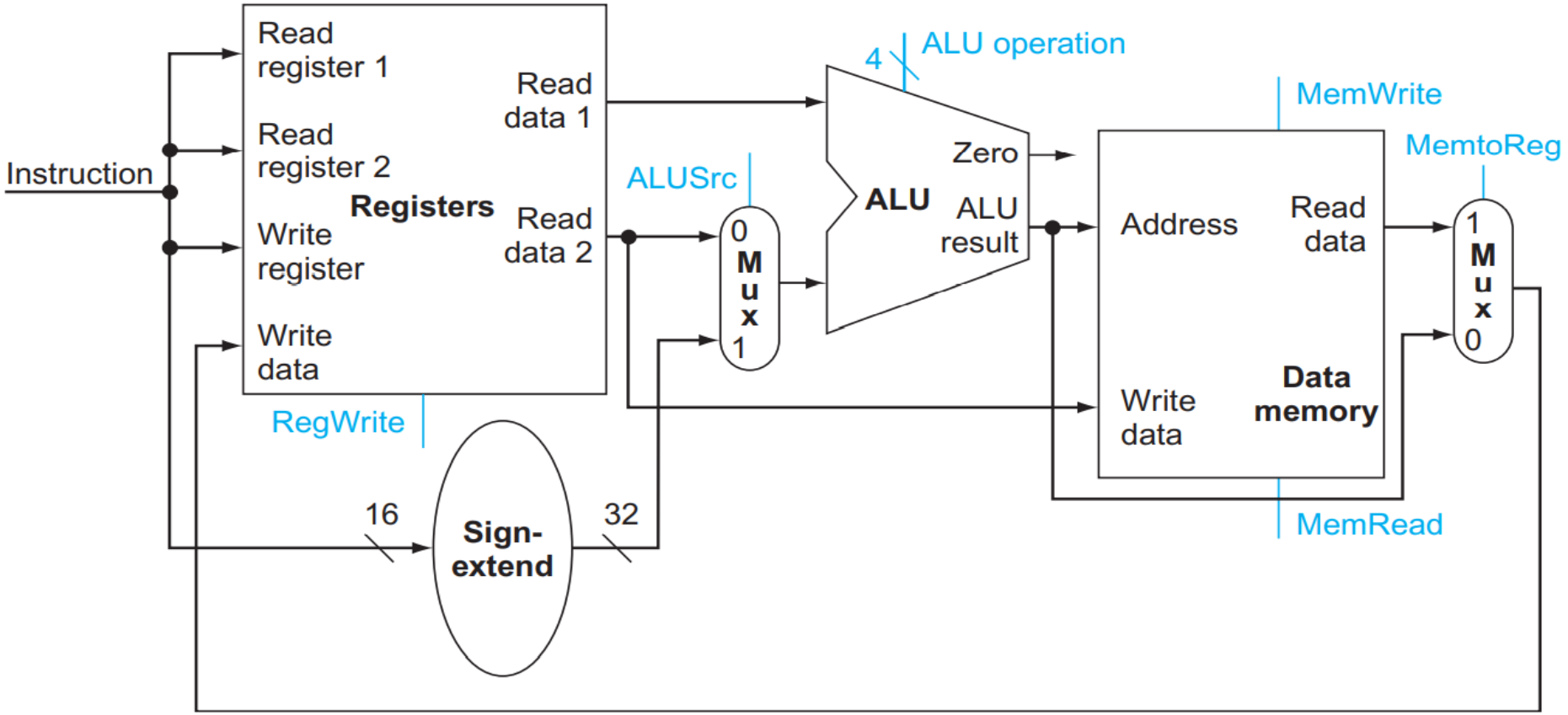
[Sign extension unit]



[Data memory]

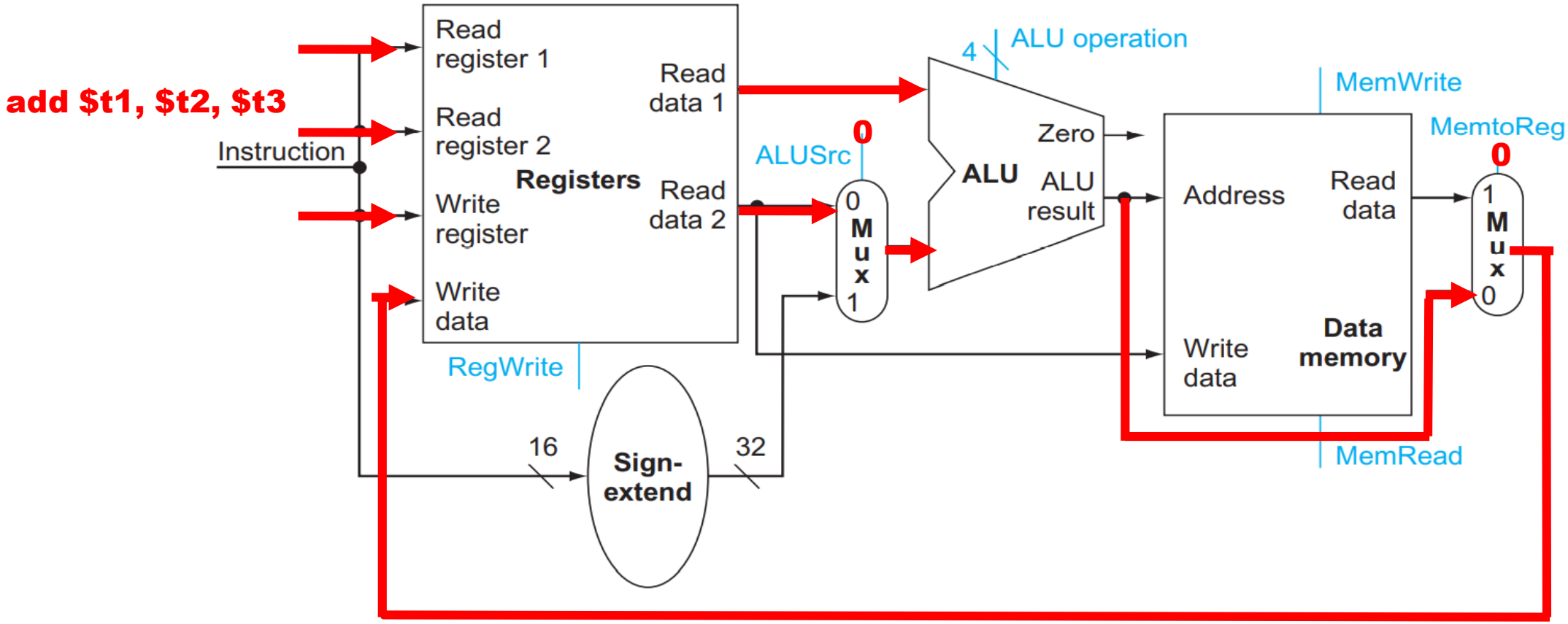
- In addition, we need a unit to sign-extend the 16-bit offset field in the instruction to a 32-bit signed value
  - **lw $t1, offset_value($t2) / sw $t1, offset_value($t2)**
  - **Sign extension unit**: takes a 16-bit signed value as an input and produces a 32-bit sign-extended value as an output
  - 0XXXXXXXXXXXXXXX → 0000000000000000 0XXXXXXXXXXXXXXX
  - 1XXXXXXXXXXXXXXX → 1111111111111111 1XXXXXXXXXXXXXXX
- **Data memory**: a mem unit where data is read/stored
  - It offers separate control signals (**MemWrite** and **MemRead**), each of which is asserted for a write and a read, respectively
  - For a memory read (**lw**), the computed address is given (**Address**) and the corresponding data is provided (**Read data**)
  - For a memory write (**sw**), the data to be written is given (**Write data**) wile the computed address is given as well

A Datapath for R-Type and Load/Store (II) + (III)



- 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

Execution of R-Type on the Datapath



- **ALUSrc = 0** to select **Read data 2** of the register file to an input of ALU
- **MemtoReg = 0** to select the data value of **ALU result** to supply **Write data** of the register file
- For R-type instructions, sign extension unit and data memory are not used