Computer Architecture

Fall, 2026

Week 9

2026.11.02

組別：_____ 簽名：_____

[group 1]

1. How many registers does a MIPS register file contain, and what are the functions of the RA, RB, and RW inputs?

Ans:

A MIPS register file contains 32 registers.

RA selects the register whose data is output on busA.

RB selects the register whose data is output on busB.

RW selects the register to be written through busW when Write Enable is 1.

[group 1]

2. What is the primary function of the Sign-Extend unit in the MIPS datapath, and what are its input and output bit widths?

Ans:

The Sign-Extend unit converts a 16-bit signed immediate value into a 32-bit signed integer by replicating the sign bit (bit 15) across the upper 16 bits.

Input width: 16 bits (taken from instruction bits [15:0]).

Output width: 32 bits (fed into the second input of the ALU multiplexer).

[group 14]

3. Explain how the instruction lw $t0, 8($s1) is executed in a MIPS single-cycle processor. Describe the datapath components involved.

Ans:

執行流程：

1. Instruction Memory：根據 PC 取得 lw 指令。
2. Register File：讀取 $s1 的值。
3. Sign Extend：將 16-bit immediate 8 擴展成 32 bits。
4. ALU：計算記憶體位址 R[$s1] + 8。
5. Data Memory：從計算出的位址讀取資料。
6. Register File：將讀取的資料寫入 $t0。

[group 8]

4. In a single-cycle MIPS processor, both lw and sw use the ALU to calculate a memory address. Explain the difference between lw and sw, and identify the values of MemRead, MemWrite, and RegWrite for each instruction.

Ans:

Both lw and sw use the ALU to calculate the memory address (Base Register + Offset).

- lw: MemRead = 1, MemWrite = 0, RegWrite = 1
- sw: MemRead = 0, MemWrite = 1, RegWrite = 0

lw loads data from memory into a register, while sw stores data from a register into memory.

[group 12]

5. Consider a MIPS single-cycle processor executing:
   sw $t2, 12($s1)
   Assume $s1 = 1000 and $t2 = 42.
   (a) What memory address does the ALU calculate?
   (b) What are the values of ALUSrc, MemWrite, MemRead, and RegWrite?
   (c) Why is the ALU needed for a store instruction?

Ans:

(a) Address = 1000 + 12 = 1012. The value 42 is stored at memory address 1012

(b)

- ALUSrc = 1
- MemWrite = 1
- MemRead = 0
- RegWrite = 0

(c) The ALU calculates the effective memory address by adding the base register and the sign-extended offset. It does not perform the memory write itself.

[group 7]

6. Consider the following three MIPS instructions executed separately on a single-cycle processor:
   add $t0, $t1, $t2
   lw $t0, 8($t1)
   sw $t0, 8($t1)
   (a) For each instruction, identify the two inputs to the ALU and explain the operation performed by the ALU and its purpose.
   (b) For the sw instruction, which of the following combinations of control signals is correct?
   The control signals are listed in the following order: (RegDst, ALUSrc, MemtoReg, RegWrite, MemWrite)
   (X represents Don't Care.)
   A. (0, 1, 1, 1, 0)
   B. (X, 1, X, 0, 1)
   C. (1, 0, 0, 1, 0)
   D. (X, 0, X, 0, 1)

Ans:

(a)

Instruction	ALU Input A	ALU Input B	Operation and Purpose
add	R[$t1]	R[$t2]	Addition; calculates the sum of two registers
lw	R[$t1]	SignExt(8)	Addition; calculates the memory address for loading
sw	R[$t1]	SignExt(8)	Addition; calculates the memory address for storing

(b) Correct answer: B. (X, 1, X, 0, 1)

The sw instruction uses the sign-extended immediate to calculate the memory address (ALUSrc = 1) and writes data to memory (MemWrite = 1). It does not write to the Register File (RegWrite = 0), so RegDst and MemtoReg are Don’t Care.

[group 4]

7. How does the processor check whether to take a beq branch?

Ans:

The ALU subtracts the values of the two source registers. If the result is zero, the registers are equal and the branch is taken. The next PC becomes PC = PC + 4 + (sign extended immediate << 2). Otherwise, the next PC is PC+4.

[group 6]

8. Why does the lw instruction have a longer critical path than an R-type instruction in a single-cycle processor?

Ans:

The lw instruction requires more steps because it must access data memory. It needs to fetch the instruction, read the register, calculate the memory address using the ALU, access data memory, and write the result back to a register.

An R-type instruction only needs the ALU result and does not access data memory.

[group 8]

9. A single-cycle processor has the following instruction execution delays:
   - add: 300 ps
   - sw: 550 ps
   - lw: 750 ps

   What is the minimum clock cycle time required for this processor? Explain why an add instruction cannot use a shorter clock cycle even though its execution delay is only 300 ps.

Ans:

In a single-cycle processor, every instruction must complete within one clock cycle. Therefore, the clock period must accommodate the slowest instruction.

Clock Cycle Time = max(300, 550, 750) = 750 ps

Even though add only requires 300 ps, it still occupies one full clock cycle of 750 ps because all instructions share the same clock period.

The slowest instruction determines the minimum clock period in a single-cycle processor, which can result in wasted time for faster instructions.

[group 5]

10. A single-cycle processor supports three instruction types: arithmetic instructions take 5 ns, load instructions take 8 ns, and branch instructions take 6 ns.
    (a) What should the minimum clock period be for the processor to operate correctly?
    (b) If a program executes 20 arithmetic instructions, 10 load instructions, and 5 branch instructions, how long does the whole program take?
    (c) Suppose the designer changes the clock period to 7 ns to improve performance. Will the processor still work correctly? Explain which instruction causes the problem.

Ans:

(a) The clock period must be at least as long as the slowest instruction. The maximum delay is 8 ns, so the minimum clock period is 8 ns.

(b) Total instructions = 20 + 10 + 5 = 35. Since this is a single-cycle processor, execution time = 35 × 8 ns = 280 ns.

(c) No. The load instruction requires 8 ns, but the new clock period is only 7 ns. The load instruction may not finish before the next clock edge, causing incorrect results.

[group 11]

11. In a single-cycle MIPS CPU, one control signal in the control unit is broken and stuck at a fixed value. Before the program below runs, $t3 and $t4 are both 0,
    memory 0($s0) = 5, and 4($s0) = 7:
    lw $t1, 0($s0)
    lw $t2, 4($s0)
    add $t3, $t1, $t2
    sub $t4, $t3, $t1
    After the program finishes, you observe: $t1 = -5, $t2 = 12, $t3 = 0, $t4 = 0
    (a) What should the correct results be?
    (b) Which control signal is broken? Is it stuck at 0 or 1?
    (c) With this fault, will sw and beq still work correctly? Why?

Ans:

- (a) The correct results are $t1 = 5, $t2 = 7, $t3 = 12, $t4 = 7

- (b) RegDst is stuck at 0. With RegDst = 0, the destination register is rt instead of rd, so R-type instructions write their results to the wrong register:

- add $t3,$t1,$t2 has rt = $t2, so $t2 = 5 + 7 = 12

- sub $t4,$t3,$t1 has rt = $t1, so $t1 = 0 − 5 = −5 ($t3 is never written, so it stays 0)

- Both lw instructions write to rt anyway, so they are not affected

- (c) Yes, they still work. sw and beq have RegWrite = 0, so they never write back to a register, and RegDst is a don't-care for them.