Fall, 2026

Week 2

2026.09.14

組別：_____ 簽名：_____

[group 4]

1. A semiconductor company wants to produce 1,000 good dies for sale.
   Wafer area = 100 cm^2; cost per wafer = $500.
   Die area = 1 cm^2; defect rate = 0.2 defects/cm^2.
   Calculate the estimated manufacturing cost for 1,000 good dies using the yield model. Ignore wafer-edge losses and use average cost per good die (no whole-wafer rounding).

Ans:
Dies per wafer = Wafer area / Die area = 100 / 1 = 100 dies
Yield = $1 / (1 + (\text{Defect rate} \times \text{Die area}) / 2)^2 = 1 / (1 + (0.2 \times 1) / 2)^2 = 1 / (1.1)^2 = 0.8264$
Cost per die = Cost per wafer / (Dies per wafer × Yield) = 500 / (100 × 0.8264) = $6.05
Cost for 1,000 good dies = 6.05 × 1,000 = $6,050

[group 8]

2. What are the advantages and disadvantages of using a multiprocessor system?

Ans:
Advantage
Breaks the power wall — can't lower voltage or remove more heat, so add cores to keep improving performance.
Disadvantages
Requires explicitly parallel programming (unlike ILP, which is hidden from the programmer).
Hard to do: load balancing, and optimizing communication and synchronization.

[group 3]

3. MIPS (millions of instructions per second) is not always an accurate metric for evaluating processor performance. Consider two processors executing the same task:
   P1: clock rate = 4 GHz, average CPI = 0.8, instruction count = 2×10^9.
   P2: clock rate = 3 GHz, average CPI = 0.75, instruction count = 1×10^9.
   Which processor has the higher MIPS? Which has better overall performance?

Ans:
MIPS=(Instrct. count) / (Execution time$\times 10^6$)=(Instrct. count) / ((Instrct. count×CPI) / (Clock rate)$\times 10^6$)=(Clock rate) / (CPI$\times 10^6$)
P1: MIPS=(4×10^9) / (0.8×10^6)=5×10^3
P2: MIPS=(3×10^9) / (0.75×10^6)=4×10^3
P1 has the higher MIPS.
Execution time=(Instrct. count×CPI) / (Clock rate)
P1: Execution time=($2\times 10^9\times 0.8$) / (4×10^9)=0.4
P2: Execution time=($1\times 10^9\times 0.75$) / (3×10^9)=0.25
P2 has the better overall performance.

[group 5]

4. According to Amdahl's Law, suppose 80% of a program's execution time can be improved. How much must the optimizable portion be sped up to achieve an overall speedup of 2 times?

Ans:
According to Amdahl's Law,
Speedup_overall=(1) / ((1-P)+(P) / (S))
where P=0.8, Speedup_overall=2, S is the speedup of the improved portion.
Therefore,
2=(1) / (0.2+(0.8) / (S))
S=(0.8) / (0.3)=(8) / (3)≈2.67
Answer: The optimizable portion must be sped up by approximately 2.67×.

[group 5]

5. What are three fundamental design principles of instruction set architecture (ISA) discussed in class? Which principle explains why MIPS uses a relatively small number of general-purpose registers (32 registers)?

Ans:
Three fundamental design principles are:
I. Simplicity favors regularity
II. Smaller is faster
III. Make the common case fast
MIPS uses only 32 general-purpose registers mainly because of:
Principle II: Smaller is faster.
A smaller register file generally allows faster access and can simplify hardware implementation.

[group 4]

6. Why does MIPS use registers for arithmetic operations instead of operating directly on memory?

Ans:
Because registers are faster than memory and allow arithmetic instructions to execute more efficiently.
Memory data must first be transferred using instructions like lw and sw.

[group 8]

7. Why do we need an immediate operand in ISA? What are its advantages?

Ans:

Why

Small constants are used frequently — 50% of operands.

Avoids loading the constant from memory.

Design Principle 3: make the common case fast.

Advantages

No extra load instruction — fewer instructions.

No memory access, so faster.

[group 1]

8. Translate the following C statement into MIPS assembly:
   f = (g - h) + (i - j) + (k - 10);
   Assume f, g, h, i, j, k are in $s0, $s1, $s2, $s3, $s4, $s5, respectively.

Ans:

Sol(1):

```
sub $t0, $s1, $s2 # $t0 = g - h
sub $t1, $s3, $s4 # $t1 = i - j
add $s0, $t0, $t1 # $s0 = (g - h) + (i - j)
addi $t2, $s5, -10 # $t2 = k - 10
add $s0, $s0, $t2 # $s0 = (g - h) + (i - j) + (k - 10)
```

Sol(2)

(Use as less temporary registers as possible)

```
sub $s0, $s1, $s2 # f = g - h
sub $t0, $s3, $s4 # $t0 = i - j
add $s0, $s0, $t0 # f = (g - h) + (i - j)
addi $t0, $s5, -10 # $t0 = k - 10
add $s0, $s0, $t0 # f = ((g - h) + (i - j)) + (k - 10)
```

[group 2]

9. 已知 C 語言交換兩個整數的程式為 temp = a; a = b; b = temp;
   (1) 若 a、b 分別在 $t0、$t1，使用 add 與 $zero，至少需要幾條指令完成交換？請寫出指令。
   (2) 若交換的是 A[7]、A[8]，陣列 A 的起始位址在 $s0，且每個整數佔 4 bytes，請用 lw、sw 寫出指令。

Ans:

```
add $t2, $t0, $zero
add $t0, $t1, $zero
add $t1, $t2, $zero
------------------
lw $t0, 28($s0)
lw $t1, 32($s0)
sw $t1, 28($s0)
sw $t0, 32($s0)
```

[group 12]

10. C code: A[12] = h + A[8];
    A is an integer array with 4 bytes per element. h is in $s2, and the base address of A is in $s3.
    Compiled MIPS code:
    lw $t0, X($s3)
    add $t0, $s2, $t0
    sw $t0, Y($s3)
    為什麼 X、Y 分別等於 32、48，而不是 8、12？

Ans:
因為lw跟sw的offset是位移了「幾個 bytes」而非陣列的index，又因為一個int是佔4 bytes，所以才需將8跟12都乘上4。

[group 10]

11. Consider the following C program:
    A[5] = A[2] + B[3] - h;
    B[6] = A[5] + B[1];
    The base addresses of A and B are in $s0 and $s1, respectively; h is in $s2.
    $s0 = 1000, $s1 = 2000, h = 7. Each integer occupies 4 bytes.
    Initial memory contents (address: value):
    1008: 20; 1020: 0; 2004: 6; 2012: 15; 2024: 0.
    What value is stored at memory address 2024 after execution?

Ans:
Since one integer occupies 4 bytes: Address(A[i])=1000+4i and Address(B[i])=2000+4i
First, Address(A[2])=1000+2(4)=1008 From the given memory contents: A[2]=20
Also, Address(B[3])=2000+3(4)=2012 so: B[3]=15
The first statement is: A[5] = A[2] + B[3] - h;
Therefore: A[5]=20+15-7=28
So the value stored at A[5], whose address is 1000+5(4)=1020 becomes 28.
Next: B[6] = A[5] + B[1];
The address of B[1] is: 2000+1(4)=2004 and therefore: B[1]=6
Thus: B[6]=28+6=34
The address of B[6] is: 2000+6(4)=2024
Therefore: The value stored at memory address 2024 is 34