

- a) What is the CPI for each machine?

The CPI for each machine is given by:

$$\text{CPI}_{\text{Mbase}} = (40\% \times 2) + (25\% \times 3) + (25\% \times 3) + (10\% \times 5) = 2.8$$

$$\text{CPI}_{\text{Mopt}} = (40\% \times 2) + (25\% \times 2) + (25\% \times 3) + (10\% \times 4) = 2.45$$

- b) What are the native MIPS rating for *Mbase* and *Mopt*?

The MIPS ratings for each machine is given by:

$$\text{MIPS} = \frac{\text{Instruction Count}}{\text{Execution Time} \times 10^6} = \frac{\text{Clock Rate}}{\text{CPI} \times 10^6}$$

$$\text{MIPS} = \frac{500 \times 10^6}{2.8 \times 10^6} = 178.6 \text{ MIPS}$$

$$\text{MIPS} = \frac{600 \times 10^6}{2.45 \times 10^6} = 244.9 \text{ MIPS}$$

- c) How much faster is *Mopt* than *Mbase*?

The execution times can be calculated for both machines:

$$\begin{aligned} \text{Execution Time} &= \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}} \\ \text{Execution Time}_{\text{Mbase}} &= \frac{1 \times 2.8}{500 \times 10^6} = 1 \times 5.6 \times 10^{-9} \text{ s} \\ \text{Execution Time}_{\text{Mopt}} &= \frac{1 \times 2.45}{600 \times 10^6} = 1 \times 4.08 \times 10^{-9} \text{ s} \end{aligned}$$

And finally, the performance can be compared:

$$N = \frac{\text{Execution Time}_{M_{\text{base}}}}{\text{Execution Time}_{M_{\text{opt}}}} = \frac{1 \times 5.6 \times 10^{-9} \text{ s}}{1 \times 4.08 \times 10^{-9} \text{ s}} = 1.37$$

The machine M_{opt} is 1.37 times faster than M_{base} .

3. (points 30) You are to improve cost/performance on an existing system based on a single-chip microprocessor having the following parameters.

Base Machine	Clock Frequency	100MHz	
	Die Size	10 mm x 10 mm	
	Instruction Mix	int	62
		FP	38
	CPI	Int	1.6
		FP	4.2

By using an optimizing compiler, the number of FP instructions is reduced by 20% and int instructions by 10% for the same application program.

- a) What is the new MIPS number of the base machine when using the optimizing compiler? (Instruction mix is changed!)

Instruction mix has changed, so that there is a different proportion of FP and int instructions. Therefore, we need to recalculate CPI:

$$\text{CPI} = (0.9 \times 0.62 \times 1.6 + 0.8 \times 0.38 \times 4.2) / (0.9 \times 0.62 + 0.8 \times 0.38) = 2.5169$$

The denominator represents the decrease in Instruction Count, due to the optimizing compiler.

$$\text{MIPS}_{\text{comp}} = \text{Cik freq}/\text{CPI} = 100\text{MHz}/2.5169 = 39.73 \text{ MIPS}$$

- b) When running the same application, how much performance gain do you expect from using such an optimizing compiler?

MIPS is about the same, but IC has reduced. The relative performance is given by:

$$\frac{\text{Performance_comp}}{\text{Performance_base}} = \frac{38.64 \text{ MIPS}}{39.73 \text{ MIPS} / (0.9 \times 0.62 + 0.8 \times 0.38)} = 1.128$$

The optimizing compiler improves performance by 12.8%.

By re-designing an FP hardware, die size will be increased by 20%, its FP CPI will be reduced to 2.8, clock frequency will remain as the same.

- c) What is the new MIPS number of this improved machine with an ordinary compiler?

$$\text{CPI} = 1.6 \times 0.62 + 0.38 \times 2.8 = 2.056$$

$$\text{MIPS} = 100 \text{ MHz} / 2.056 = 48.64 \text{ MIPS}$$

- d) Assuming die yield is proportional to the inverse of the cube of the die size, how much additional cost do you expect? Assume that the number of dies per wafer is inversely proportional to the die size and that all the cost is proportional to the die.

Cost is inversely proportional to yield—as yield goes down, cost goes up. The yield goes down by the cube of the area: if the area goes up by 1.2x, the yield goes down by $(1.2)^3$ x. Also, we can fit less dies on a wafer as the area goes up. If the area increases by 1.2x, we can fit 1.2x less dies per wafer. Since the cost per wafer stays the same, the cost per die goes up.

$$\frac{\text{COST_FP}}{\text{COST_BASE}} = \frac{\frac{\text{Cost / Wafer}}{\text{die / wafer} \times \text{dies yields}}}{\frac{\text{Cost / Wafer}}{\text{die / wafer} \times \text{dies yields}}} = \frac{1}{1/1.2 \times 1/(1.2)^3} = 2.037$$

The cost went up by 107%.

As an alternative, you are going use a new, scaled CMOS technology without re-designing.

In that case, the clock frequency changes to 125 MHz, its wafer cost doubles, and its die size is reduced by 15%. Use the same yield rule.

e) How much performance improvement over the base machine do you expect?

The CPI and IC is the same as the base machine. The only difference is the clock rate.

Performance_scaled = 125 MHz = 1.25 improvement.

Performance_base 100 MHz

The scaled CMOS is 25% faster.

f) How much additional cost over the base machine do you expect?

$$\frac{\text{COST_scaled}}{\text{COST_base}} = \frac{\frac{\text{Cost / Wafer}}{\text{die / wafer x dies yields}}}{\frac{\text{Cost / Wafer}}{\text{die / wafer x dies yields}}} = \frac{2}{1} = \frac{1/0.85 \cdot (1/0.85)^3}{1} = 1.044$$

The scaled CMOS costs 1.044 times as much or 4.4% more.

4. (points 20) The program below executes on the implementation also show below

```

add      r1, r2, r3
and      r4, r1, r5
sw       0(r4), r1
lw       r1, 8(r4)
xori    r5, r1, #1
beqz    r5, TARGET
sub     r5, r5, r5
.....

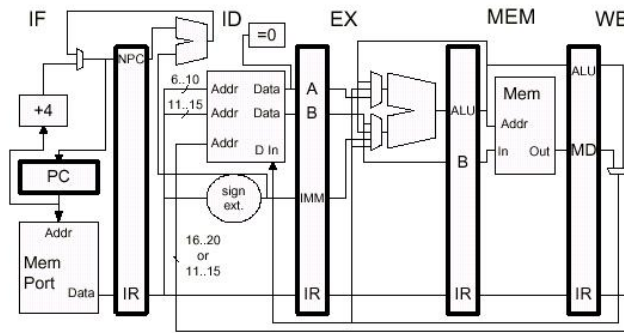
```

TARGET:

```

or       r10, r5, r1

```



The implementation includes only the forwarding paths that are shown in the figure. A new register value can be read in the same cycle it is written. Show a pipeline execution diagram for an execution of the code in which the branch is taken.

Cycle	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
add r1, r2, r3	IF	ID	EX	MEM	WB											
and r4, r1, r5		IF	ID	EX	MEM	WB										
sw 0(r4), r1			IF	ID		EX	MEM	WB								
lw r1, 8(r4)				IF		ID	EX	MEM	WB							
xori r5, r1, #1						IF	ID		EX	MEM	WB					
beqz 5, TARGET							IF		ID			EX	MEM	WB		
sub r5, r5, r5									IF			X				
.....																
TARGET:																
or r10, r5, r1												IF	ID	EX	MEM	WB