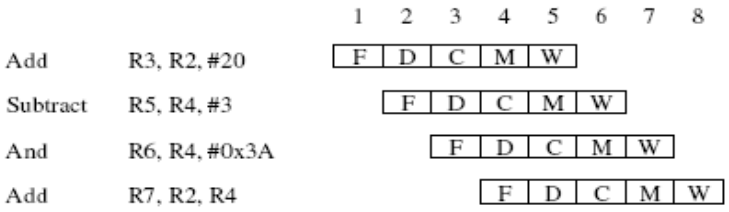
Solved Exercises 6

1. Solve Problem **6.1** from the textbook.

(*a*) The pipeline execution diagram for the given code is shown below.



The description of activity in each stage during each cycle is given below. Activity for instructions prior to the first Add instruction and instructions after the second Add instruction is not described, but would nonetheless occur as determined by the type of instruction and the stage of the pipeline.

Cycle	Stage	Activity
1	F	fetching Add instruction
2	F	fetching Subtract instruction
	D	decoding Add instruction, reading register R2 (value 2000)
3	F	fetching And instruction
	D	decoding Subtract instruction, reading register R4 (value 50)
	C	performing arithmetic $2000 + 20 = 2020$ for Add instruction
4	F	fetching Add instruction
	D	decoding And instruction, reading register R4 (value 50)
	C	performing arithmetic $50 - 3 = 47$ for Subtract instruction
	M	no operation for Add instruction
5	D	decoding Add instruction, reading register R2 (value 200) and register R4 (value 50)
	C	performing logic operation $50 \wedge 3A_{16} = 50$ for And instruction
	M	no operation for Subtract instruction
	W	write result of 2020 for Add instruction to register R3
6	C	performing arithmetic for Add instruction
	M	no operation for And instruction
	W	write result of 47 for Subtract instruction to register R5
7	M	no operation for Add instruction
	W	write result of 50 for And instruction to register R6
8	W	write result of 2050 for Add instruction to register R7

(*b*) The contents of each register during each cycle are described in the table below. The notation '–' is used to indicate that there is insufficient information to determine the register contents in that cycle. The instruction occupying the IR in each cycle is identified symbolically with an abbreviation.

	2	3	4	5	6	7	8
IR	Add	Sub	And	Add	–	–	–
PC	1004	1008	1012	1016	1020	1024	1028
RA	–	2000	50	50	2000	–	–
RB	–	–	–	–	50	–	–
RZ	–	–	2020	47	50	2050	–
RY	–	–	–	2020	47	50	2050

2. Consider a pipelined datapath consisting of five stages:

- **F** – fetch the instruction from the memory,
- **D** – decode the instruction and read the source register(s),
- **C** – execute the ALU operation specified by the instruction,
- **M** – execute the memory operation specified by the instruction,
- **W** – write the result in the destination register.

Identify data hazards in the code below and insert **NOP** instructions where necessary.

```
ADD    R2, R4, R1        // R1 = R2 + R4
ADD    R4, R6, R5        // R5 = R4 + R6
ADD    R0, R2, R3        // R3 = R0 + R2
MOV    R6, (R1)          // MEMORY[R1] = R6
MOV    (R3), R6          // R6 = MEMORY[R3]
MOV    R4, R2            // R2 = R4
ADD    #4, R4, R4        // R4 = R4 + 4
ADD    R0, R2, R1        // R1 = R0 + R2
MOV    R2, R0            // R0 = R2
```

Solution:

```
ADD    R2, R4, R1        // R1 = R2 + R4
ADD    R4, R6, R5        // R5 = R4 + R6
ADD    R0, R2, R3        // R3 = R0 + R2
NOP                      // Waiting for R1
MOV    R6, (R1)          // MEMORY[R1] = R6
NOP                      // Waiting for R3
MOV    (R3), R6          // R6 = MEMORY[R3]
MOV    R4, R2            // R2 = R4
ADD    #4, R4, R4        // R4 = R4 + 4
NOP                      // Waiting for R2
NOP                      // Waiting for R2
ADD    R0, R2, R1        // R1 = R0 + R2
MOV    R2, R0            // R0 = R2
```

2. Solve Problem **6.9** from the textbook.

Branches constitute 20 percent of the total of N instructions that are executed. They are taken 75 percent of the time.

With static prediction (where branches are assumed to be not-taken), the execution time is $N + (0.2 \times 0.75)N = 1.15N$.

With dynamic prediction where f_{wrong} is the fraction of branches that are mispredicted, the execution time is $N + (0.2 \times f_{\text{wrong}})N = (1 + 0.2 \times f_{\text{wrong}})N$.

For the execution time with dynamic prediction to be no worse than the execution time with static prediction, it must be the case that $(1 + 0.2 \times f_{\text{wrong}}) \leq 1.15$. After rearrangment of the equation, the solution is $f_{\text{wrong}} \leq 0.75$. This result is equivalent to the requirement that the dynamic prediction must be correct at least 25 percent of the time.

If branches are correctly predicted 90 percent of the time, then $f_{\text{wrong}} = 0.1$ and the execution time with dynamic prediction is $1.02N$. The speedup of dynamic prediction in this case over static prediction is

$$\left(\frac{1.15}{1.02} - 1\right) \times 100 = 13 \text{ percent}$$

3. Solve Problem **6.14** from the textbook.

The sequence of N instructions consists of 75 percent arithmetic instructions and 25 percent memory-access (Load/Store) instructions. There are no branch instructions.

On a basic five-stage pipelined processor, the execution time is N cycles (neglecting the final four cycles for the last instructions).

On a superscalar processor, the arithmetic instructions (the largest fraction) pass one at a time through the corresponding execution unit and therefore dictate the execution time as $0.75N$ cycles.

The speedup with superscalar execution is

$$\left(\frac{1}{0.75} - 1\right) \times 100 = 33 \text{ percent}$$

4. Solve Problem **9.21** from the textbook.

(a)

+1.7	0	01111	101101
−0.012	1	01000	100010
+19	0	10011	001100
1/8	0	01100	000000

"Rounding" has been used as the truncation method in these answers.

(b) Other than exact 0 and ±infinity, the smallest numbers are $\pm 1.000000 \times 2^{-14}$ and the largest numbers are $\pm 1.111111 \times 2^{15}$.

(c) Assuming sign-and-magnitude format, the smallest and largest integers (other than 0) are ± 1 and $\pm(2^{11} - 1)$; and the smallest and largest fractions (other than 0) are $\pm 2^{-11}$ and approximately ± 1.

(d)

$$\begin{aligned} A + B &= 0\ 10000\ 000000 \\ A - B &= 0\ 10000\ 110110 \\ A \times B &= 1\ 10000\ 001011 \\ A/B &= 1\ 10000\ 101110 \end{aligned}$$

"Rounding" has been used as the truncation method in these answers.