Chapter 1

Basic Structure of Computers

1.1. The steps needed to execute the machine instruction

```
Add     R4, R2, R3
```

are

- Send the address of the instruction word from register PC to the memory and issue a Read control command.
- Wait until the requested word has been retrieved from the memory, then load it into register IR, where it is interpreted (decoded) by the control circuitry to determine the operation to be performed.
- Increment the contents of register PC to point to the next instruction in memory.
- Send the contents of registers R2 and R3 to the ALU and issue an Add command to the ALU.
- Send the sum from the output of the ALU to register R4.

1.2. The steps needed to execute the machine instruction

```
Store     R4, LOC
```

are

- Send the address of the instruction word from register PC to the memory and issue a Read control command.
- Wait until the requested word has been retrieved from the memory, then load it into register IR, where it is interpreted (decoded) by the control circuitry to determine the operation to be performed.
- Increment the contents of register PC to point to the next instruction in memory.
- Send the address value LOC from the instruction in register IR, along with the contents of register R4, to the memory and issue a Write control command.
- Wait until the contents of register R4 have been written into the memory.

1.3 (*a*) A sequence of four instructions can be used.

```
Load    R2, A
Load    R3, B
Add     R4, R2, R3
Store   R4, C
```

(*b*) It is possible to use fewer instructions. The sequence is

```
Move    C, B
Add     C, A
```

1.4 (*a*) Execution time without the cache is

$$T = 250 \times 20 + 50 \times 20 \times 15 = 20,000$$

Execution time with the cache is

$$T_{cache} = 300 \times 20 + 50 \times 2 \times 14 = 7,400$$

Therefore, the speedup is

$$T/T_{cache} = 2.7$$

(*b*) Speedup is given by the expression

$$[(w - x) \times m + x \times m \times y]/[w \times m + x \times c \times (y - 1)]$$

which can be rearranged into the more useful form

$$[w \times m + x \times m \times (y - 1)]/[w \times m + x \times c \times (y - 1)]$$

(*c*) The expression involving *y* is

$$5 = [300 \times 20 + 50 \times 20 \times (y - 1)]/[300 \times 20 + 50 \times 2 \times (y - 1)]$$

which can be solved to yield

$$y = 49$$

(*d*) As the number of loop iterations, *y*, increases to become very large, the expression for speedup tends toward the value

$$[x \times m \times (y - 1)]/[x \times c \times (y - 1)]$$

which reduces to m/c as the upper limit for speedup.

1.5 (*a*) Instruction fetch time without the cache is 10 time units. Average fetch time with the cache is

$$0.96 \times 1 + (0.04 \times (10 + 1)) = 1.4$$

Therefore

$$\text{speedup} = 10/1.4 = 7.1$$

(*b*) With the size of the cache doubled, the average fetch time with the cache becomes

$$0.98 \times 1 + (0.02 \times (10 + 1)) = 1.2$$

Therefore

$$\text{speedup} = 10/1.2 = 8.3$$

1.6 When the carry-in bit is 0 for the four cases shown in Figure 1.4, the carry-out and sum bits are 00, 01, 01, and 10, respectively.

When the carry-in bit is 1, the carry-out and sum bits are 01, 10, 10, and 11, respectively.

1.7 Overflow cases are specifically indicated. In all other cases, no overflow occurs.

00100	(+4)	00110	(+6)	10011	(-13)
+ 01011	+ (+11)	+ 01110	+ (+14)	+ 01100	+ (+12)
01111	(+15)	10100	(+20)	11111	(-1)
		overflow			
11100	(-4)	11110	(-2)	10111	(-9)
+ 01000	+ (+8)	+ 10111	+ (-9)	+ 10010	+ (-14)
00100	(+4)	10101	(-11)	01001	(-23)
				overflow	

1.8 Overflow cases are specifically indicated. In all other cases, no overflow occurs.

00100	(+4)	00100
− 01011	− (+11)	+ 10101
	(-7)	11001
00110	(+6)	00110
− 01110	− (+14)	+ 10010
	(-8)	11000
10011	(-13)	10011
− 01100	− (+12)	+ 10100
	(-25)	00111
		overflow
11100	(-4)	11100
− 01000	− (+8)	+ 11000
	(-12)	10100
11110	(-2)	11110
− 10111	− (-9)	+ 01001
	(+7)	00111
10111	(-9)	10111
− 10010	− (-14)	+ 01110
	(+5)	00101

1.9 As a binary number, the pattern 01010011 represents the decimal value $83 = (64 + 16 + 2 + 1)$. As an ASCII code, it represents the capital letter S.

1.10 Suppose that the subtraction operation is

$$\mathrm{R} = \mathrm{A} - \mathrm{B}$$

Overflow may occur only if the signs of A and B are different. In that case, overflow occurs if the sign of the result, R, is different from the sign of A. An alternative criterion is: When the signs of A and B are different, overflow occurs if the sign of the result, R, is the same as the sign of B.