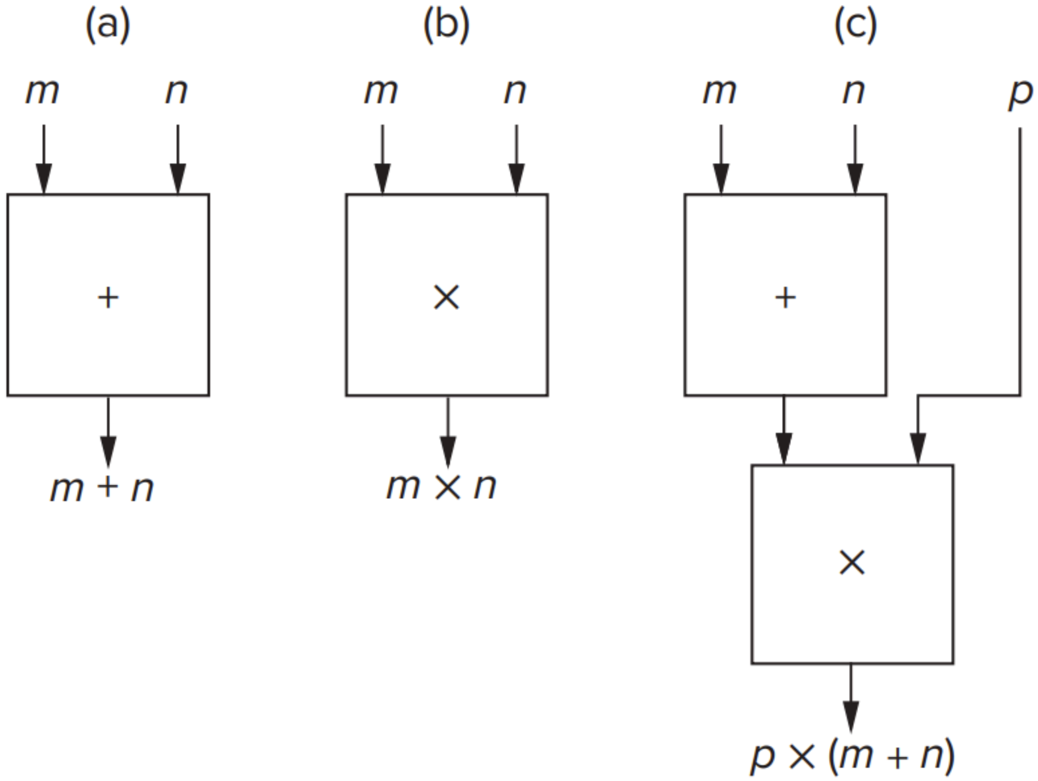
Homework 1

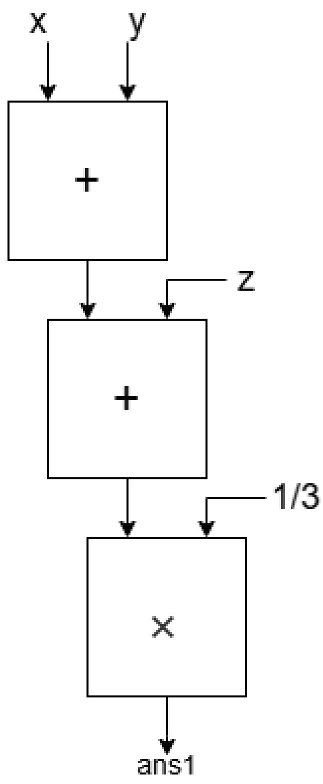
T1

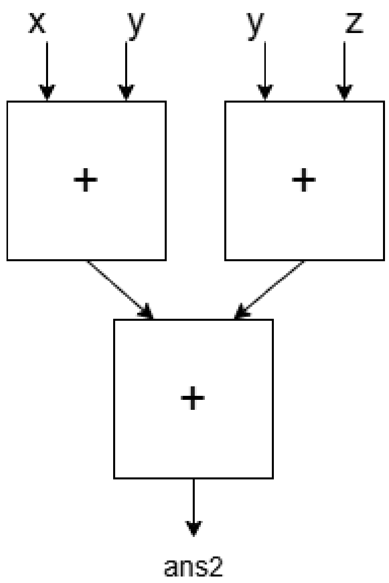
Say we had a “black box,” which takes two numbers as input and outputs their sum. See Figure 1.10a.
Say we had another box capable of multiplying two numbers together. See Figure 1.10b.
We can connect these boxes together to calculate p × (m + n). See Figure 1.10c.

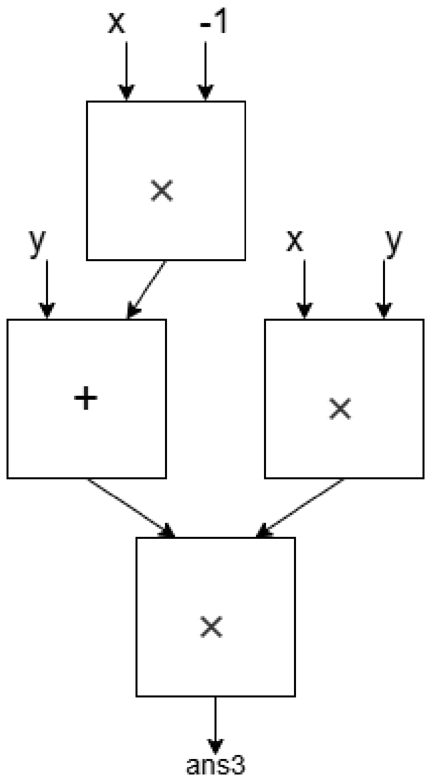


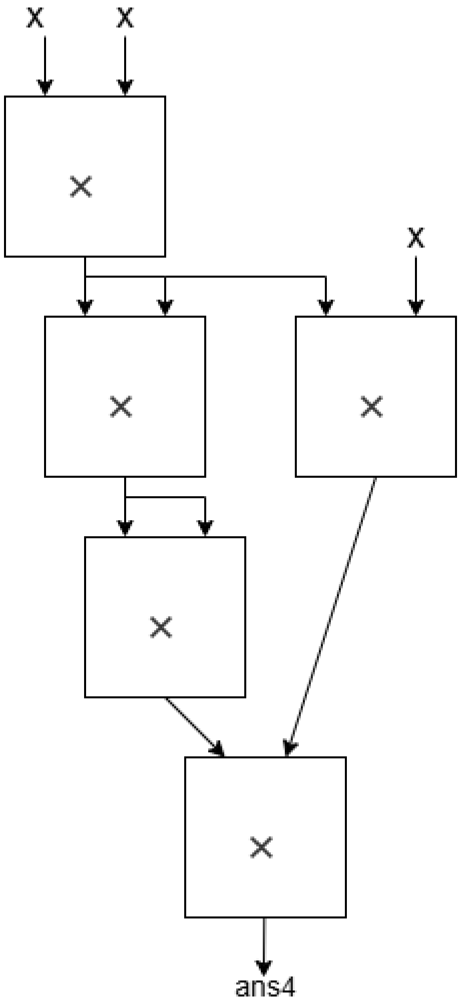
Assume we have an unlimited number of these boxes. Show how to connect them together to calculate:

1. The average of the three input numbers x, y, and z.
2. x+2y+z
3. $xy^2 - x^2y$
4. How many boxes do you need at least to calculate x^{11}?


x
y
+
z
+
1/3
×
ans1


x
y
y
z
+
+
+
ans2


x
-1
×
y
x
y
+
×
×
ans3


x
x
×
x
×
×
×
×
ans4

T2

Convert these decimal numbers to 8-bit 2's complement numbers:

1. 73
2. 46
3. -115

Convert the following 8-bit 2's complement numbers to decimal:

1. 0101 1010
2. 1111 0110
3. 1000 0110

> 0100 1001
>
> 0010 1110
>
> 1000 1101
>
> 90   -10   -122

T3

Compute the following and answer it in decimal. Assume each operand is a 2's complement binary number.

1. 11 + 01010101
2. 01001 - 111010
3. 1010 − 01011

> 84
>
> 15
>
> -17

T4

Write your result in binary and decimal.

1. What is the largest positive number one can represent in an eight-bit 2’s complement code?
2. What is the greatest magnitude negative number one can represent in an eight-bit 2’s complement code?
3. What is the largest positive number one can represent in n-bit 2’s complement code?
4. What is the negative number with the largest absolute value one can represent in n-bit 2’s complement code?

> 1. 0111 1111   127
> 2. 1000 0000   -128

> 3. 011...1111 $2^{n-1}-1$
> 4. 100...0000 -2^{n-1}

T5

Describe what conditions indicate overflow has occurred when two 2’s complement numbers are added?
Describe what conditions indicate overflow has occurred when two unsigned numbers are added?

> 对于n bits 的情况，两个正数相加或两个负数相加时，且绝对值之和不小于2^n，会发生溢出；
>
> 正数和负数相加永远不可能溢出。
>
> 例如8-bit的两个数相加，如果发生溢出，是指后7位都归零，且溢出1：
>
> 对于正负数相加，不可能同时满足归零和溢出：
>
> 0111 1111 (127) + 0000 0001(1) = 1000 0000 (-128)
>
> 00001010 (10) + 11110110 (-10) = 0000 0000 (0)

T6

Write the decimal equivalents for the following IEEE floating point numbers:

1. `0 10010010 01110000000000000000000`
2. `1 00001110 10011000000000000000000`

Write IEEE floating point representation of the following decimal numbers:

1. 5.375
2. $-10\frac{9}{32}$

> $(1+0.4375)\times 2^{19} = 1.4375\times 2^{19}$
>
> $-1\times(1+0.59375)\times 2^{-113} = -1.59375\times 2^{-113}$
>
> 0 10000001 01011000000000000000000
>
> 1 10000010 01001001000000000000000

T7

What are the largest and smallest exponents the IEEE standard allows for a 32-bit floating point number? (Answer in decimal)

What about the smallest number regardless of infinity? And the smallest positive number?
(Answer in binary)

> - **1 bit** for the sign

> - **8 bits** for the exponent　**注意全0和全1特殊表示**
> - **23 bits** for the fraction
>
> 最大指数部分：11111110　254-127=127
>
> 最小指数部分：00000001　1-127=-126
>
> 最小数字：1 11111110 11111111111111111111111 $-(2-2^{-23})\times 2^{127}$
>
> **Smallest Positive Normalized Number**: 0 00000001 00000000000000000000000 1.0×2^{-126}
>
> **Smallest Positive Subnormal Number**: 0 00000000 00000000000000000000001 $1.0\times 2^{-126}\times 2^{-23}$

T8

Compute the following and answer in **hexadecimal**:

1. (0011 AND 0110) AND 1101
2. 0101 0111 OR NOT(1101 0111)
3. (1101 0010 OR 0001 1001) OR NOT(0110 1101 AND 1010 1110)

What strategy would you use to design a program that can quickly compute the result of a long series of n-bit AND operations?

> 0x0
>
> 0x7F
>
> 0xDB
>
> 每一位可并行计算；对于遇到的操作数，如果对应位为0，则可终止迭代。
>
> 合理即可。

T9

Refer to Example 2.11(Page 43) for the following questions.

1. What mask value and what operation would one use to indicate that machine 2 is busy?
2. What mask value and what operation would one use to indicate that machines 2 and 6 are no longer busy?
3. What mask value and what operation would one use to indicate that all machines are busy?
4. What mask value and what operation would one use to indicate that all machines are idle?
5. Using the operations discussed in this chapter, develop a procedure to isolate the status bit of **machine 5** as the sign bit. For example, if the BUSYNESS pattern is 01**0**11100, then the output of this procedure is 00000000. If the BUSYNESS pattern is 01**1**10011, then the output is 10000000. ***Hint:*** What happens when you ADD a bit pattern to itself?

> Current BUSYNESS Vector AND 11111011 (mask)
>
> Current BUSYNESS Vector OR 0100 0100 (mask)
>
> Current BUSYNESS Vector AND 00000000 (mask)
>
> Current BUSYNESS Vector OR 11111111 (mask)
>
> 第一步：Current BUSYNESS Vector AND 00100000 第二步：执行两次加法 00100000 + 00100000 + 00100000 = 10000000 即左移两位

T10

Fill in the truth table for the equations given.

`Q1 = NOT(X AND Z) AND (X AND Y OR Z)`
`Q2 = NOT(Y OR Z) AND NOT(X AND Y AND Z)`

X	Y	Z	Q1	Q2
0	0	0	0	1
0	0	1	1	0
0	1	0	0	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	1	0
1	1	1	0	0

> 注意优先级