3 Solutions

Solution 3.1

3.1.1

a.	7620
b.	3626

3.1.2

a.	752
b.	3626

3.1.3

a.	2771	–723
b.	103	103

3.1.4

a.	730
b.	1560

3.1.5

a.	730
b.	1560

3.1.6

a.	010111110010
b.	010110100110

The attraction is that each octal digit contains one of 8 different characters (0–7). Since with 3 binary bits you can represent 8 different patterns, in octal each digit requires exactly 3 binary bits. You can write down the conversion directly.

Solution 3.2

3.2.1

a.	EA4B
b.	F034

3.2.2

a.	CFE3
b.	8406

3.2.3

a.	3380	3380
b.	47645	–14877

3.2.4

a.	9662
b.	6321

3.2.5

a.	DE96
b.	F29D

3.2.6

a.	1011101001111100
b.	1010101011011111

The attraction is that each hex digit contains one of 16 different characters (0–9, A–E). Since with 4 binary bits you can represent 16 different patterns, in hex each digit requires exactly 4 binary bits. And bytes are by definition 8 bits long, so two hex digits are all that are required to represent the contents of 1 byte.

Solution 3.3

3.3.1

a.	Underflow (–21)
b.	Neither (58)

3.3.2

a.	Overflow (result = 159, which does not fit into an 8-bit SM format)
b.	Overflow (result = 146, which does not fit into an SM 8-bit format)

3.3.3

a.	Neither (–21)
b.	Neither (58)

3.3.4

a.	–56 + 103 = 47
b.	–9 – 19 = –28

3.3.5

a.	–56 – 103 = –128 (–159)
b.	–9 + 19 = 10

3.3.6

a.	200 + 103 = 255 (303)
b.	247 + 237 = 255 (484)

Solution 3.4

3.4.1

a. 50×23

Step	Action	Multiplier	Multiplicand	Product
0	Initial Vals	010 011	000 000 101 000	000 000 000 000
1	Prod = Prod + Mcand	010 011	000 000 101 000	000 000 101 000
	Lshift Mcand	010 011	000 001 010 000	000 000 101 000
	Rshift Mplier	001 001	000 001 010 000	000 000 101 000
2	Prod = Prod + Mcand	001 001	000 001 010 000	000 001 111 000
	Lshift Mcand	001 001	000 010 100 000	000 001 111 000
	Rshift Mplier	000 100	000 010 100 000	000 001 111 000
3	lsb = 0, no op	000 100	000 010 100 000	000 001 111 000
	Lshift Mcand	000 100	000 101 000 000	000 001 111 000
	Rshift Mplier	000 010	000 101 000 000	000 001 111 000
4	lsb = 0, no op	000 010	000 101 000 000	000 001 111 000
	Lshift Mcand	000 010	001 010 000 000	000 001 111 000
	Rshift Mplier	000 001	001 010 000 000	000 001 111 000
5	Prod = Prod + Mcand	000 001	001 010 000 000	001 011 111 000
	Lshift Mcand	000 001	010 100 000 000	001 011 111 000
	Rshift Mplier	000 000	010 100 000 000	001 011 111 000
6	lsb = 0, no op	000 000	001 010 000 000	001 011 111 000
	Lshift Mcand	000 000	101 000 000 000	001 011 111 000
	Rshift Mplier	000 000	101 000 000 000	001 011 111 000

b. 66×04

Step	Action	Multiplier	Multiplicand	Product
0	Initial Vals	000 100	000 000 110 110	000 000 000 000
1	lsb = 0, no op	000 100	000 000 110 110	000 000 000 000
	Lshift Mcand	000 100	000 001 101 100	000 000 000 000
	Rshift Mplier	000 010	000 001 101 100	000 000 000 000
2	lsb = 0, no op	000 010	000 001 101 100	000 000 000 000
	Lshift Mcand	000 010	000 011 011 000	000 000 000 000
	Rshift Mplier	000 001	000 011 011 000	000 000 000 000

Step	Action	Multiplier	Multiplicand	Product
3	Prod = Prod + Mcand	000 001	000 011 011 000	000 011 011 000
	Lshift Mcand	000 001	000 110 110 000	000 011 011 000
	Rshift Mplier	000 000	000 110 110 000	000 011 011 000
4	lsb = 0, no op	000 000	000 110 110 000	000 011 011 000
	Lshift Mcand	000 000	001 101 100 000	000 011 011 000
	Rshift Mplier	000 000	001 101 100 000	000 011 011 000
5	lsb = 0, no op	000 000	001 101 100 000	000 011 011 000
	Lshift Mcand	000 000	011 011 000 000	000 011 011 000
	Rshift Mplier	000 000	011 011 000 000	000 011 011 000
6	lsb = 0, no op	000 000	011 011 000 000	000 011 011 000
	Lshift Mcand	000 000	110 110 000 000	000 011 011 000
	Rshift Mplier	000 000	110 110 000 000	000 011 011 000

3.4.2

a. 50×23

Step	Action	Multiplicand	Product/Multiplier
0	Initial Vals	101 000	000 000 010 011
1	Prod = Prod + Mcand	101 000	101 000 010 011
	Rshift Product	101 000	010 100 001 001
2	Prod = Prod + Mcand	101 000	111 100 001 001
	Rshift Mplier	101 000	011 110 000 100
3	lsb = 0, no op	101 000	011 110 000 100
	Rshift Mplier	101 000	001 111 000 010
4	lsb = 0, no op	101 000	001 111 000 010
	Rshift Mplier	101 000	000 111 100 001
5	Prod = Prod + Mcand	101 000	101 111 100 001
	Rshift Mplier	101 000	010 111 110 000
6	lsb = 0, no op	101 000	010 111 110 000
	Rshift Mplier	101 000	001 011 111 000

b. 66×04

Step	Action	Multiplicand	Product/Multiplier
0	Initial Vals	110 110	000 000 000 100
1	lsb = 0, no op	110 110	000 000 000 100
	Rshift Mplier	110 110	000 000 000 010
2	lsb = 0, no op	110 110	000 000 000 010
	Rshift Mplier	110 110	000 000 000 001
3	Prod = Prod + Mcand	110 110	110 110 000 001
	Rshift Product	110 110	011 011 000 000
4	lsb = 0, no op	110 110	011 011 000 000
	Rshift Mplier	110 110	001 101 100 000
5	lsb = 0, no op	110 110	001 101 100 000
	Rshift Mplier	110 110	000 110 110 000
6	lsb = 0, no op	110 110	000 110 110 000
	Rshift Mplier	110 110	000 011 011 000

3.4.3 No solution provided

3.4.4

a. $54 \times 67 = 424$

Step	Action	Mplier	Multiplicand	Product	Sign
0	Initial Values	110 111	000 000 101 100	000 000 000 000	0
	Multiplier.sign XOR Multiplicand.sign (1 XOR 1)				0
	Make positive	010 111	000 000 001 100	000 000 000 000	0
1	Prod = Prod + Mcand	010 111	000 000 001 100	000 000 001 100	0
	Lshift Mcand	010 111	000 000 011 000	000 000 001 100	0
	Rshift Mplier	001 011	000 000 011 000	000 000 001 100	0
2	Prod = Prod + Mcand	001 011	000 000 011 000	000 000 100 100	0
	Lshift Mcand	001 011	000 000 110 000	000 000 100 100	0
	Rshift Mplier	000 101	000 000 110 000	000 000 100 100	0
3	Prod = Prod + Mcand	000 101	000 000 110 000	000 001 010 100	0
	Lshift Mcand	000 101	000 001 100 000	000 001 010 100	0
	Rshift Mplier	000 010	000 001 100 000	000 001 010 100	0

Step	Action	Mplier	Multiplicand	Product	Sign
4	lsb = 0, no op	000 010	000 001 100 000	000 001 010 100	0
	Lshift Mcand	000 010	000 011 000 000	000 001 010 100	0
	Rshift Mplier	000 001	000 011 000 000	000 001 010 100	0
5	Prod = Prod + Mcand	000 001	000 011 000 000	000 100 010 100	0
	Lshift Mcand	000 001	000 110 000 000	000 100 010 100	0
	Rshift Mplier	000 000	000 110 000 000	000 100 010 100	0
6	lsb = 0, no op	000 000	000 110 000 000	000 100 010 100	0
	Lshift Mcand	000 000	001 100 000 000	000 100 010 100	0
	Rshift Mplier	000 000	001 100 000 000	000 100 010 100	0
7	Prod msb = sign	000 000	001 100 000 000	000 100 010 100	0

b. $30 \times 7 = 250$

Step	Action	Mplier	Multiplicand	Product	Sign
0	Initial Values	000 111	000 000 011 000	000 000 000 000	0
	Multiplier.sign XOR Multiplicand.sign (0 XOR 0)				0
	Make positive	000 111	000 000 011 000	000 000 000 000	0
1	Prod = Prod + Mcand	000 111	000 000 011 000	000 000 011 000	0
	Lshift Mcand	000 111	000 000 110 000	000 000 011 000	0
	Rshift Mplier	000 011	000 000 110 000	000 000 011 000	0
2	Prod = Prod + Mcand	000 011	000 000 110 000	000 001 001 000	0
	Lshift Mcand	000 011	000 001 100 000	000 001 001 000	0
	Rshift Mplier	000 001	000 001 100 000	000 001 001 000	0
3	Prod = Prod + Mcand	000 001	000 001 100 000	000 010 101 000	0
	Lshift Mcand	000 001	000 011 000 000	000 010 101 000	0
	Rshift Mplier	000 000	000 011 000 000	000 010 101 000	0
4	lsb = 0, no op	000 000	000 011 000 000	000 010 101 000	0
	Lshift Mcand	000 000	000 110 000 000	000 010 101 000	0
	Rshift Mplier	000 000	000 110 000 000	000 010 101 000	0
5	lsb = 0, no op	000 000	000 110 000 000	000 010 101 000	0
	Lshift Mcand	000 000	001 100 000 000	000 010 101 000	0
	Rshift Mplier	000 000	001 100 000 000	000 010 101 000	0

Step	Action	Mplier	Multiplicand	Product	Sign
6	lsb = 0, no op	000 000	001 100 000 000	000 010 101 000	0
	Lshift Mcand	000 000	011 000 000 000	000 010 101 000	0
	Rshift Mplier	000 000	011 000 000 000	000 010 101 000	0
7	Prod msb = sign	000 000	011 000 000 000	000 010 101 000	0

3.4.5

a. $54 \times 67 = (-24 \times -11 = 264)$

Step	Action	Multiplicand	Product/Multiplier
0	Initial Vals	101 100	0 000 000 110 111
1	Prod = Prod + Mcand	101 100	1 101 100 110 111
	ARshift Mplier	101 100	1 110 110 011 011
2	Prod = Prod + Mcand	101 100	1 100 010 011 011
	Rshift Product	101 100	1 110 001 001 101
3	Prod = Prod + Mcand	101 100	1 011 101 001 101
	Rshift Mplier	101 100	1 101 110 100 110
4	lsb = 0, no op	101 100	1 101 110 100 110
	Rshift Mplier	101 100	1 110 111 010 011
5	Prod = Prod + Mcand	101 100	1 100 011 010 011
	Rshift Mplier	101 100	1 110 001 101 001
6	Prod = Prod – Mcand	101 100	0 000 101 101 001
	Rshift Mplier	101 100	0 000 010 110 100

b. $30 \times 7 = 250$

Step	Action	Multiplicand	Product/Multiplier
0	Initial Vals	011 000	0 000 000 000 111
1	Prod = Prod + Mcand	011 000	0 011 000 000 111
	Rshift Mplier	011 000	0 001 100 000 011
2	Prod = Prod + Mcand	011 000	0 100 100 000 011
	Rshift Product	011 000	0 010 010 000 001
3	Prod = Prod + Mcand	011 000	0 101 010 000 001
	Rshift Mplier	011 000	0 010 101 000 000
4	lsb = 0, no op	011 000	0 010 101 000 000
	Rshift Mplier	011 000	0 001 010 100 000

Step	Action	Multiplicand	Product/Multiplier
5	lsb = 0, no op	011 000	0 001 010 100 000
	Rshift Mplier	011 000	0 000 101 010 000
6	lsb = 0, no op	011 000	0 000 101 010 000
	Rshift Mplier	011 000	0 000 010 101 000

3.4.6 No solution provided

Solution 3.5

3.5.1 For hardware, it takes 1 cycle to do the add, 1 cycle to do the shift, and 1 cycle to decide if we are done. So the loop takes $(3 \times A)$ cycles, with each cycle being B time units long.

For a software implementation, it takes 1 cycle to do the add, 1 cycle to do each shift, and 1 cycle to decide if we are done. So the loop takes $(4 \times A)$ cycles, with each cycle being B time units long.

a.	(3 × 4) × 3tu = 36 time units for hardware (4 × 4) × 3tu = 48 time units for software
b.	(3 × 32) × 7tu = 672 time units for hardware (4 × 32) × 7tu = 896 time units for software

3.5.2 It takes B time units to get through an adder, and there will be A – 1 adders.

a.	Word is 4 bits wide, requiring 3 adders. 3 × 3tu = 9 time units.
b.	Word is 32 bits wide, requiring 31 adders. 31 × 7tu = 217 time units.

3.5.3 It takes B time units to get through an adder, and the adders are arranged in a tree structure. It will require log2(A) levels.

a.	4 bits wide word requires 3 adders in 2 levels. 2 × 3tu = 6 time units.
b.	32 bits word requires 31 adders in 5 levels. 5 × 7tu = 35 time units.

Solution 3.6

3.6.1

a.	0x24 × 0xC9 = 0x1C44. 0x24 = 36, and 36 = 32 + 4, so we can shift 0xC9 left 5 places, then add to that value (0x1920) 0xC9 shifted left 2 places (0x324) = 0x1C44. Total 2 shifts, 1 add.
b.	0x41 × 0x18 = 0x618 0x41 = 64 + 1, 0x18 = 16 + 2. Best way would be to shift 0x18 left 6 places, and then add 0x18. 1 shift, 1 add.

3.6.2

a.	0x24 × 0xC9 = 0x24 × –0x49 = –0xA44 = 8A44 0x24 = 36, and 36 = 32 + 4, so we can shift 0x49 left 5 places (0x920), then add to that value 0x49 shifted left 2 places (0x124) = 0xA44. We need to keep track of the sign ... one of the two is negative, so the result will be negative. Total 2 shifts, 1 add.
b.	0x41 × 0x18 = 0x618 0x41 = 64 + 1, 0x18 = 16 + 2. Best way would be to shift 0x18 left 6 places, and then add 0x18. 1 shift, 1 add.

3.6.3 No solution provided

3.6.4 Quoting the wikipedia entry directly:

Booth's algorithm involves repeatedly adding one of two predetermined values A and S to a product P, then performing a rightward arithmetic shift on P. Let x and y be the multiplicand and multiplier, respectively; and let x and y represent the number of bits in x and y.

1. Determine the values of A and S, and the initial value of P. All of these numbers should have a length equal to $(x + y + 1)$.
   a. A: Fill the most significant (leftmost) bits with the value of x. Fill the remaining $(y + 1)$ bits with zeros.
   b. S: Fill the most significant bits with the value of (–x) in two's complement notation. Fill the remaining $(y + 1)$ bits with zeros.
   c. P: Fill the most significant x bits with zeros. To the right of this, append the value of y. Fill the least significant (rightmost) bit with a zero.
2. Determine the two least significant (rightmost) bits of P.
   a. If they are 01, find the value of $P + A$. Ignore any overflow.
   b. If they are 10, find the value of $P + S$. Ignore any overflow.
   c. If they are 00 or 11, do nothing. Use P directly in the next step.
3. Arithmetically shift the value obtained in the previous step by a single place to the right. Let P now equal this new value.
4. Repeat steps 2 and 3 until they have been done y times.
5. Drop the least significant (rightmost) bit from P. This is the product of x and y.

3.6.5

a. $0x42 \times 0x36 = 0x0DEC$

Action	Multiplicand	Product/Multiplier
Initial Vals	0100 0010	0000 0000 0011 0110 0
00, nop shift	0100 0010 0100 0010	0000 0000 0011 0110 0 0000 0000 0001 1011 0
10, subtract shift	0100 0010 0100 0010	1011 1110 0001 1011 0 1101 1111 0000 1101 1
11, nop shift	0100 0010 0100 0010	1101 1111 0000 1101 1 1110 1111 1000 0110 1
01, add shift	0100 0010 0100 0010	0011 0001 1000 0110 1 0001 1000 1100 0011 0
10, subtract shift	0100 0010 0100 0010	1101 0110 1100 0011 0 1110 1011 0110 0001 1
11, nop shift	0100 0010 0100 0010	1110 1011 0110 0001 1 1111 0101 1011 0000 1
01, add shift	0100 0010 0100 0010	0011 0111 1011 0000 1 0001 1011 1101 1000 0
00, nop shift	0100 0010 0100 0010	0001 1011 1101 1000 0 0000 1101 1110 1100 0

b. $0x9F \times 0x8E = -0x61 \times -0x72 = 2B32$

Action	Multiplicand	Product/Multiplier
Initial Vals	1001 1111	0000 0000 1000 1110 0
00, nop shift	1001 1111 1001 1111	0000 0000 1000 1110 0 0000 0000 0100 0111 0
10, subtract shift	1001 1111 1001 1111	0110 0001 0100 0111 0 0011 0000 1010 0011 1
11, nop shift	1001 1111 1001 1111	0011 0000 1010 0011 1 0001 1000 0101 0001 1
11, nop shift	1001 1111 1001 1111	0001 1000 0101 0001 1 0000 1100 0010 1000 1
01, add shift	1001 1111 1001 1111	1010 1011 0010 1000 1 1101 0101 1001 0100 0
00, nop shift	1001 1111 1001 1111	1101 0101 1001 0100 0 1110 1010 1100 1010 0

Action	Multiplicand	Product/Multiplier
00, nop shift	1001 1111 1001 1111	1110 1010 1100 1010 0 1111 0101 0110 0101 0
10, subtract shift	1001 1111 1001 1111	0101 0110 0110 0101 0 0010 1011 0011 0010 1

3.6.6 No solution provided

Solution 3.7

3.7.1

a. 50/23 = 2 remainder 2

Step	Action	Quotient	Divisor	Remainder
0	Initial Vals	000 000	010 011 000 000	000 000 101 000
1	Rem = Rem – Div	000 000	010 011 000 000	101 101 101 000
	Rem < 0, R + D, Q<<	000 000	010 011 000 000	000 000 101 000
	Rshift Div	000 000	001 001 100 000	000 000 101 000
2	Rem = Rem – Div	000 000	001 001 100 000	110 111 001 000
	Rem < 0, R + D, Q<<	000 000	001 001 100 000	000 000 101 000
	Rshift Div	000 000	000 100 110 000	000 000 101 000
3	Rem = Rem – Div	000 000	000 100 110 000	111 011 111 000
	Rem < 0, R + D, Q<<	000 000	000 100 110 000	000 000 101 000
	Rshift Div	000 000	000 010 011 000	000 000 101 000
4	Rem = Rem – Div	000 000	000 010 011 000	111 110 010 000
	Rem < 0, R + D, Q<<	000 000	000 010 011 000	000 000 101 000
	Rshift Div	000 000	000 001 001 100	000 000 101 000
5	Rem = Rem – Div	000 000	000 001 001 100	111 110 111 100
	Rem < 0, R + D, Q<<	000 000	000 001 001 100	000 000 101 000
	Rshift Div	000 000	000 000 100 110	000 000 101 000
6	Rem = Rem – Div	000 000	000 000 100 110	000 000 000 010
	Rem > 0, Q << 1	000 001	000 000 100 110	000 000 000 010
	Rshift Div	000 001	000 000 010 011	000 000 000 010
7	Rem = Rem – Div	000 000	000 000 010 011	111 111 101 111
	Rem < 0, R + D, Q<<	000 010	000 000 010 011	000 000 000 010
	Rshift Div	000 010	000 000 001 101	000 000 000 010

b. 25/44 = 0 remainder 25

Step	Action	Quotient	Divisor	Remainder
0	Initial Vals	000 000	100 100 000 000	000 000 010 101
1	Rem = Rem – Div	000 000	100 100 000 000	100 011 101 011
	Rem < 0, R + D, Q<<	000 000	100 100 000 000	000 000 010 101
	Rshift Div	000 000	010 010 000 000	000 000 010 101
2	Rem = Rem – Div	000 000	010 010 000 000	101 110 010 101
	Rem < 0, R + D, Q<<	000 000	010 010 000 000	000 000 010 101
	Rshift Div	000 000	001 001 000 000	000 000 010 101
3	Rem = Rem – Div	000 000	001 001 000 000	110 111 010 101
	Rem < 0, R + D, Q<<	000 000	001 001 000 000	000 000 010 101
	Rshift Div	000 000	000 100 100 000	000 000 010 101
4	Rem = Rem – Div	000 000	000 100 100 000	111 011 110 101
	Rem < 0, R + D, Q<<	000 000	000 100 100 000	000 000 010 101
	Rshift Div	000 000	000 010 010 000	000 000 010 101
5	Rem = Rem – Div	000 000	000 010 010 000	111 110 000 101
	Rem < 0, R + D, Q<<	000 000	000 010 010 000	000 000 010 101
	Rshift Div	000 000	000 001 001 000	000 000 010 101
6	Rem = Rem – Div	000 000	000 001 001 000	111 111 001 101
	Rem > 0, R + D, Q<<	000 000	000 001 001 000	000 000 010 101
	Rshift Div	000 000	000 000 100 100	000 000 010 101
7	Rem = Rem – Div	000 000	000 000 100 100	111 111 110 001
	Rem < 0, R + D, Q<<	000 000	000 000 100 100	000 000 010 101
	Rshift Div	000 000	000 000 010 010	000 000 010 101

3.7.2 In these solutions a 1 or a 0 was added to the Quotient if the remainder was greater than or equal to 0. However, an equally valid solution is to shift in a 1 or 0, but if you do this you must do a compensating right shift of the remainder (only the remainder, not the entire remainder/quotient combination) after the last step.

a. 50/23 = 2 remainder 2

Step	Action	Divisor	Remainder/Quotient
0	Initial Vals	010 011	000 000 101 000
1	R<<	010 011	000 001 010 000
	Rem = Rem – Div	010 011	111 110 010 000
	Rem < 0, R + D	010 011	000 001 010 000

Step	Action	Divisor	Remainder/Quotient
2	R<<	010 011	000 010 100 000
	Rem = Rem – Div	010 011	101 111 100 000
	Rem < 0, R + D	010 011	000 010 100 000
3	R<<	010 011	000 101 000 000
	Rem = Rem – Div	010 011	110 010 000 000
	Rem < 0, R + D	010 011	000 101 000 000
4	R<<	010 011	001 010 000 000
	Rem = Rem – Div	010 011	111 001 000 000
	Rem < 0, R + D	010 011	001 010 000 000
5	R<<	010 011	010 100 000 000
	Rem = Rem – Div	010 011	000 001 000 000
	Rem > 0, R0 = 1	010 011	000 001 000 001
6	R<<	010 011	000 010 000 010
	Rem = Rem – Div	010 011	101 111 000 010
	Rem < 0, R + D	010 011	000 010 000 010

b. 25/44 = 0 remainder 25

Step	Action	Divisor	Remainder/Quotient
0	Initial Vals	100 100	000 000 010 101
1	R<<	100 100	000 000 101 010
	Rem = Rem – Div	100 100	100 100 101 010
	Rem < 0, R + D	100 100	000 000 101 010
2	R<<	100 100	000 001 010 100
	Rem = Rem – Div	100 100	100 011 010 100
	Rem < 0, R + D	100 100	000 001 010 100
3	R<<	100 100	000 010 101 000
	Rem = Rem – Div	100 100	100 010 101 000
	Rem < 0, R + D	100 100	000 010 101 000
4	R<<	100 100	000 101 010 000
	Rem = Rem – Div	100 100	100 001 010 000
	Rem < 0, R + D	100 100	000 101 010 000

Step	Action	Divisor	Remainder/Quotient
5	R<<	100 100	001 010 100 000
	Rem = Rem – Div	100 100	100 110 100 000
	Rem < 0, R + D	100 100	001 010 100 000
6	R<<	100 100	010 101 000 000
	Rem = Rem – Div	100 100	110 001 000 000
	Rem > 0, R0 = 1	100 100	010 101 000 000

3.7.3 No solution provided

3.7.4

a. 55/24 = 0 remainder 15: Dividend negative

Sign of Quotient = (Sign bit of Divisor) XOR (Sign bit of Dividend) = negative
Sign of Remainder = Sign of Dividend = negative

Step	Action	Quotient	Divisor	Remainder
0	Initial Vals	000 000	010 100 000 000	000 000 001 101
1	Rem = Rem – Div	000 000	010 100 000 000	101 100 001 101
	Rem < 0, R + D, Q<<	000 000	010 100 000 000	000 000 001 101
	Rshift Div	000 000	001 010 000 000	000 000 001 101
2	Rem = Rem – Div	000 000	001 010 000 000	110 110 001 101
	Rem < 0, R + D, Q<<	000 000	001 010 000 000	000 000 001 101
	Rshift Div	000 000	000 101 000 000	000 000 001 101
3	Rem = Rem – Div	000 000	000 101 000 000	111 011 001 101
	Rem < 0, R + D, Q<<	000 000	000 101 000 000	000 000 001 101
	Rshift Div	000 000	000 010 100 000	000 000 001 101
4	Rem = Rem – Div	000 000	000 010 100 000	111 101 101 101
	Rem < 0, R + D, Q<<	000 000	000 010 100 000	000 000 001 101
	Rshift Div	000 000	000 001 010 000	000 000 001 101
5	Rem = Rem – Div	000 000	000 001 010 000	111 110 111 101
	Rem < 0, R + D, Q<<	000 000	000 001 010 000	000 000 001 101
	Rshift Div	000 000	000 000 101 000	000 000 001 101
6	Rem = Rem – Div	000 000	000 000 101 000	111 111 100 101
	Rem < 0, R + D, Q<<	000 000	000 000 101 000	000 000 001 101
	Rshift Div	000 000	000 000 010 100	000 000 001 101

Step	Action	Quotient	Divisor	Remainder
7	Rem = Rem – Div	000 000	000 000 010 100	111 111 111 001
	Rem < 0, R + D, Q<<	000 000	000 000 010 100	000 000 001 101
	Rshift Div	000 000	000 000 001 010	000 000 001 101
8	Set sign bits	100 000	000 000 001 010	100 000 001 101

b. 36/51 = 3 remainder 3: Dividend positive

Sign of Quotient = (Sign bit of Divisor) XOR (Sign bit of Dividend) = negative
Sign of Remainder = Sign of Dividend = positive

Step	Action	Quotient	Divisor	Remainder
0	Initial Vals	000 000	001 001 000 000	000 000 011 110
1	Rem = Rem – Div	000 000	001 001 000 000	110 111 011 110
	Rem < 0, R + D, Q<<	000 000	001 001 000 000	000 000 011 110
	Rshift Div	000 000	000 100 100 000	000 000 011 110
2	Rem = Rem – Div	000 000	000 100 100 000	111 110 111 110
	Rem < 0, R + D, Q<<	000 000	000 100 100 000	000 000 011 110
	Rshift Div	000 000	000 010 010 000	000 000 011 110
3	Rem = Rem – Div	000 000	000 010 010 000	111 110 001 110
	Rem < 0, R + D, Q<<	000 000	000 010 010 000	000 000 011 110
	Rshift Div	000 000	000 001 001 000	000 000 011 110
4	Rem = Rem – Div	000 000	000 001 001 000	111 111 010 110
	Rem < 0, R + D, Q<<	000 000	000 001 001 000	000 000 011 110
	Rshift Div	000 000	000 000 100 100	000 000 011 110
5	Rem = Rem – Div	000 000	000 000 100 100	111 111 111 010
	Rem < 0, R + D, Q<<	000 000	000 000 100 100	000 000 011 110
	Rshift Div	000 000	000 000 010 010	000 000 011 110
6	Rem = Rem – Div	000 000	000 000 010 010	000 000 001 100
	Rem > 0, Q << 1	000 001	000 000 010 010	000 000 001 100
	Rshift Div	000 001	000 000 001 001	000 000 001 100
7	Rem = Rem – Div	000 010	000 000 001 001	000 000 000 011
	Rem > 0, Q << 1	000 011	000 000 001 001	000 000 000 011
	Rshift Div	000 011	000 000 000 100	000 000 000 011
8	Set sign bits	100 011	000 000 000 100	000 000 000 011

3.7.5

a. 55/24 = 0 remainder 15: Dividend negative

Sign of Quotient = (Sign bit of Divisor) XOR (Sign bit of Dividend) = negative
Sign of Remainder = Sign of Dividend = negative

Step	Action	Divisor	Remainder/Quotient
0	Initial Vals	010 100	000 000 001 101
1	R<<	010 100	000 000 011 010
	Rem = Rem – Div	010 100	101 100 011 010
	Rem < 0, R + D	010 100	000 000 011 010
2	R<<	010 100	000 000 110 100
	Rem = Rem – Div	010 100	101 100 110 100
	Rem < 0, R + D	010 100	000 000 110 100
3	R<<	010 100	000 001 101 000
	Rem = Rem – Div	010 100	101 101 110 100
	Rem < 0, R + D	010 100	000 001 101 000
4	R<<	010 100	000 011 010 000
	Rem = Rem – Div	010 100	101 111 010 000
	Rem < 0, R + D	010 100	000 011 010 000
5	R<<	010 100	000 110 100 000
	Rem = Rem – Div	010 100	110 010 100 000
	Rem < 0, R + D	010 100	000 110 100 000
6	R<<	010 100	001 101 000 000
	Rem = Rem – Div	010 100	111 001 000 000
	Rem > 0, R0 = 1	010 100	001 101 000 000
7	Adjust signs	010 100	101 101 100 000 (Q = –0, Rem = –15)

b. 36/51 = 3 remainder 3: Dividend positive

Sign of Quotient = (Sign bit of Divisor) XOR (Sign bit of Dividend) = negative
Sign of Remainder = Sign of Dividend = positive

Step	Action	Divisor	Remainder/Quotient
0	Initial Vals	001 001	000 000 011 110
1	R<<	001 001	000 000 111 100
	Rem = Rem – Div	001 001	110 111 111 100
	Rem < 0, R + D	001 001	000 000 111 100

Step	Action	Divisor	Remainder/Quotient
2	R<<	001 001	000 001 111 000
	Rem = Rem – Div	001 001	111 000 111 000
	Rem < 0, R + D	001 001	000 001 111 000
3	R<<	001 001	000 011 110 000
	Rem = Rem – Div	001 001	111 010 110 000
	Rem < 0, R + D	001 001	000 011 110 000
4	R<<	001 001	000 111 100 000
	Rem = Rem – Div	001 001	111 110 100 000
	Rem < 0, R + D	001 001	000 111 100 000
5	R<<	001 001	001 111 000 000
	Rem = Rem – Div	001 001	000 110 000 000
	Rem > 0, R0 = 1	001 001	000 110 000 001
6	R<<	001 001	001 100 000 010
	Rem = Rem – Div	001 001	000 011 000 010
	Rem > 0, R0 = 1	001 001	000 011 000 011
7	Adjust signs	001 001	000 011 100 011 (Q = –3, Rem = 3)

3.7.6 No solution provided

Solution 3.8

3.8.1 In these solutions a 1 will be shifted into the quotient and a compensating right shift of the remainder will be performed. This is the alternate approach mentioned in Solution 3.7.2.

a. 75/12 = 6 remainder 1

Step	Action	Divisor	Remainder/Quotient
0	Initial Vals	001 010	000 000 111 101
	R<<	001 010	000 001 111 010
	Rem = Rem – Div	001 010	110 111 111 010
1	Rem < 0, Q << 0, Addnext	001 010	101 111 110 100
	Rem = Rem + Div	001 010	111 001 110 100
2	Rem < 0, Q << 0, Addnext	001 010	110 011 101 000
	Rem = Rem + Div	001 010	111 101 101 000

Step	Action	Divisor	Remainder/Quotient
3	Rem < 0, Q << 0, Addnext	001 010	111 011 010 000
	Rem = Rem + Div	001 010	000 101 010 000
4	Rem > 0, Q << 1, Subnext	001 010	001 010 100 001
	Rem = Rem – Div	001 010	000 000 100 001
5	Rem > 0, Q << 1, Subnext	001 010	000 001 000 011
	Rem = Rem – Div	001 010	110 111 000 011
6	Rem < 0, Q << 0, Addnext	001 010	101 110 000 110
	Rem = Rem + Div	001 010	111 000 000 110
7	Rem < 0, Rem = Rem + Div	001 010	000 010 000 110
	Shift Rem >> 1	001 010	000 001 000 110 (Q = 6, Rem = 1)

b. 52/41 = 1, remainder 11

Step	Action	Divisor	Remainder/Quotient
0	Initial Vals	100 001	000 000 101 010
	R<<	100 001	000 001 010 100
	Rem = Rem – Div	100 001	100 000 010 100
1	Rem < 0, Q << 0, Addnext	100 001	000 000 101 000
	Rem = Rem + Div	100 001	100 001 101 000
2	Rem < 0, Q << 0, Addnext	100 001	000 011 010 000
	Rem = Rem + Div	100 001	100 100 010 000
3	Rem < 0, Q << 0, Addnext	100 001	001 000 100 000
	Rem = Rem + Div	100 001	101 001 100 000
4	Rem < 0, Q << 0, Addnext	100 001	010 011 000 000
	Rem = Rem + Div	100 001	110 100 000 000
5	Rem < 0, Q << 0, Addnext	100 001	101 000 000 000
	Rem = Rem + Div	100 001	001 001 000 000
6	Rem > 0, Q << 1, Subnext	100 001	010 010 000 001
	Rem = Rem – Div	100 001	110 001 000 001
7	Rem < 0, Rem = Rem + Div	100 001	010 010 000 001
	Shift Rem >> 1	100 001	001 001 000 001 (Q = 1, Rem = 11)

3.8.2 No solution provided

3.8.3 No solution provided

3.8.4

a. 17/14 = 1 remainder 3

Step	Action	Quotient	Temp	Divisor	Remainder
0	Initial Vals	000000	000000 000000	001100 000000	000000 001111
1	Temp = Rem – Div	000000	110100 000111	001100 000000	000000 001111
	Temp < 0, Q << 0	000000	110100 000111	001100 000000	000000 001111
	Rshift Div	000000	110100 000111	000110 000000	000000 001111
2	Temp = Rem – Div	000000	111010 001111	000110 000000	000000 001111
	Temp < 0, Q << 0	000000	111010 001111	000110 000000	000000 001111
	Rshift Div	000000	111010 001111	000011 000000	000000 001111
3	Temp = Rem – Div	000000	111101 001111	000011 000000	000000 001111
	Temp < 0, Q << 0	000000	111101 001111	000011 000000	000000 001111
	Rshift Div	000000	111101 001111	000001 100000	000000 001111
4	Temp = Rem – Div	000000	111110 101111	000001 100000	000000 001111
	Temp < 0, Q << 0	000000	111110 101111	000001 100000	000000 001111
	Rshift Div	000000	111110 101111	000000 110000	000000 001111
5	Temp = Rem – Div	000000	111111 010111	000000 110000	000000 001111
	Temp < 0, Q << 0	000000	111111 010111	000000 110000	000000 001111
	Rshift Div	000000	111111 010111	000000 011000	000000 001111
6	Temp = Rem – Div	000000	111111 110111	000000 011000	000000 001111
	Temp < 0, Q<<	000000	111111 110111	000000 011000	000000 001111
	Rshift Div	000000	111111 110111	000000 001100	000000 001111
7	Temp = Rem – Div	000000	000000 000011	000000 001100	000000 001111
	T > 0, Q << 1, R = T	000001	000000 000011	000000 001100	000000 000011
	Rshift Div	000001	000000 000011	000000 000110	000000 000011

b. 70/23 = 2 remainder 22

Step	Action	Quotient	Temp	Divisor	Remainder
0	Initial Vals	000000	000000 000000	010011 000000	000000 111000
1	Temp = Rem – Div	000000	101101 111000	010011 000000	000000 111000
	Temp < 0, Q << 0	000000	101101 111000	010011 000000	000000 111000
	Rshift Div	000000	101101 111000	001001 100000	000000 111000
2	Temp = Rem – Div	000000	110111 011000	001001 100000	000000 111000
	Temp < 0, Q << 0	000000	110111 011000	001001 100000	000000 111000
	Rshift Div	000000	110111 011000	000100 110000	000000 111000

Step	Action	Quotient	Temp	Divisor	Remainder
3	Temp = Rem – Div	000000	111100 001000	000100 110000	000000 111000
	Temp < 0, Q << 0	000000	111100 001000	000100 110000	000000 111000
	Rshift Div	000000	111100 001000	000010 011000	000000 111000
4	Temp = Rem – Div	000000	111110 001000	000010 011000	000000 111000
	Temp < 0, Q << 0	000000	111110 001000	000010 011000	000000 111000
	Rshift Div	000000	111110 001000	000001 001100	000000 111000
5	Temp = Rem – Div	000000	111110 110100	000001 001100	000000 111000
	Temp < 0, Q << 0	000000	111110 110100	000001 001100	000000 111000
	Rshift Div	000000	111110 110100	000000 100110	000000 111000
6	Temp = Rem – Div	000000	000000 010010	000000 100110	000000 111000
	T > 0, Q << 1, R = T	000001	000000 010010	000000 100110	000000 010010
	Rshift Div	000001	000000 010010	000000 010011	000000 010010
7	Temp = Rem – Div	000001	111111 111111	000000 010011	000000 010010
	Temp < 0, Q << 0	000010	111111 111111	000000 010011	000000 010010
	Rshift Div	000010	111111 111111	000000 001001	000000 010010

3.8.5 No solution provided

3.8.6 No solution provided

Solution 3.9

3.9.1 No solution provided

3.9.2 No solution provided

3.9.3 No solution provided

Solution 3.10

3.10.1

a.	614858756	614858756
b.	–1346437120	2948530176

3.10.2

a.	`addiu $6,$5,4`
b.	`sw $31, 0($29)`

3.10.3

a.	sign is positive exp = 0x49 = –0xb7 = 73 – 128 = –55 there is a hidden 1 mantissa = 0x580008 = 5 × 16^{-1} + 8 × 16^{-2} + 8 × 16^{-6} = (.25) + (.0625) + (.03125) + (.0000002384185791) answer = +1.343750476837158203125 × 2^{-55}
b.	sign is negative exp = 0x5F = 95 – 128 = –33 there is a hidden 1 mantissa = 0x7D0000 = 7 × 16^{-1} + 13 × 16^{-2} = .4375 + .05078125 answer = –1.48828125 × 2^{-33}

3.10.4

a.	1609.5 × 10^0 = 110 0100 1001.10 × 2^0 normalize, move binary point 10 to the left 110 0100 1001.10 × 2^0 = 1.10010010011 × 2^{10} sign = negative, exp = 128 + 10 = 138 Final bit pattern: 11000101010010010011000000000000
b.	938.8125 × 10^0 = 1110101010.1101 × 2^0 normalize, move binary point 9 to the left 1.1101010101101 × 2^9 sign = negative, exp = 128 + 9 = 137 Final bit pattern: 11000100111010101011010000000000

3.10.5

a.	1609.5 × 10^0 = 110 0100 1001.10 × 2^0 normalize, move binary point 10 to the left 110 0100 1001.10 × 2^0 = 1.10010010011 × 2^{10} sign = negative, exp = 1024 + 10 = 1034 Final bit pattern: 11000000101010010010011000000000000000000000000000000000 00000000
b.	938.8125 × 10^0 = 1110101010.1101 × 2^0 normalize, move binary point 9 to the left 1.1101010101101 × 2^9 sign = negative, exp = 1024 + 9 = 1033 Final bit pattern: 11000000100111010101011010000000000000000000000000000000 00000000

3.10.6

a.	$1609.5 \times 10^0 = 011001001001.10 \times 2^0 = 649.8 \times 16^0$ move hex point 3 hex digits to the left $0110\ 0100\ 1001.10 \times 2^0 = .0110010010011 \times 16^3$ sign = negative, exp = 64 + 3 = 67 Final bit pattern: 11000011011001001001100000000000
b.	$-938.8125 \times 10^0 = 1110101010.1101 \times 2^0 = 3AA.B \times 16^0$ normalize, move hex point 3 to the left $.0011\ 1010\ 1010\ 1101 \times 16^3$ sign = negative, exp = 64 + 3 = 67 Final bit pattern: 11000011001110101010110100000000

Solution 3.11

3.11.1

a.	$5.00736125 \times 10^5 = 500736.125 \times 10^0 = 0x7A400.2 \times 16^0 =$ $1111010010000000000.0010 \times 2^0$ move the binary point 19 to the left = $.1111010010000000000001 \times 2^{10011}$ exponent = +19, mantissa = +.1111010010000000000000100000 answer: 000000010011011110100100000000000010
b.	$-2.691650390625 \times 10^{-2} = -.02691650390625 \times 10^0 = -.00000110111001 \times 2^0$ move the binary point 5 to the right = $-.110111001 \times 2^{-5}$ exponent = –5, mantissa = –.110111001 answer: 1111111110111001000111000000000000

3.11.2

a.	$5.00736125 \times 10^5 = 500736.125 \times 10^0 = 0x7A400.2 \times 16^0 =$ $1111010010000000000.0010 \times 2^0$ move the binary point 18 to the left = $1.1110100100000000000010 \times 2^{10010}$ exponent = +18, mantissa = +1110100100000000000010 answer: Cannot represent +18, use biggest possible (11111) answer: 0111111110100100
b.	$-2.691650390625 \times 10^{-2} = -.02691650390625 \times 10^0 = -.00000110111001 \times 2^0$ move the binary point 6 to the right = $-1.10111001 \times 2^{-6}$ exponent = –6 = –6 + 16 = 10, mantissa = –.10111001 answer: 1010101011100100

3.11.3

a.	$5.00736125 \times 10^5 = 500736.125 \times 10^0 = 0x7A400.2 \times 16^0 =$ $1111010010000000000.0010 \times 2^0$ move the binary point 19 to the left = $.11110100100000000000010 \times 2^{10011}$ exponent = +19, mantissa = +.1111010010000000000000010 answer: 01111010010000000000000100010110

b.	$-2.691650390625 \times 10^{-2} = -.02691650390625 \times 10^{0} = -.00000110111001 \times 2^{0}$ move the binary point 5 to the right $= -.110111001 \times 2^{-5}$ exponent = –5, mantissa = –.110111001 answer: 10010001110000000000000000001011

3.11.4

a.	$-1.278 \times 10^{3} + -3.90625 \times 10^{-1}$
	$-1.278 \times 10^{3} = -1278 = -10011111110 = -1.0011111110 \times 2^{10}$
	$-3.90625 \times 10^{-1} = -.390625 = -1.1001000000 \times 2^{-2}$
	Shift binary point 12 to the left to align exponents,
	$-1.1001000000 \times 2^{-2} \text{ –> } -0.0000000000011 \times 2^{12}$
	`              GR`
	`-1.0011111110 00`
	`-0.0000000000 01 1 (Guard = 0, Round = 1, Sticky = 1)`
	`----------------`
	`-1.0011111101 11    Guard = 1, Round = 1, Round up.`
	$-1.0011111110 \times 2^{10} = -1.278 \times 10^{3}$

b.	$2.3109375 \times 10^{1} + 6.391601562 \times 10^{-1}$
	$2.3109375 \times 10^{1} = 23.109375 = 1.0111000111 \times 2^{4}$
	$6.391601562 \times 10^{1} = .6391601562 = 1.0100011101 \times 2^{-1}$
	Shift binary point 5 to the left and align exponents,
	`             GR`
	`1.0111000111 00`
	`0.0000101000 11 101  (Guard = 1, Round = 1, Sticky = 1)`
	`------------------`
	`1.0111101111 11`
	In this case Guard and Round are both 1, so we round up.
	$1.0111110000 \times 2^{4} = 10111.110000 \times 2^{0} = 23.75 = 2.375 \times 10^{1}$

3.11.5 No solution provided

3.11.6 No solution provided

Solution 3.12

3.12.1

a.

$5.66015625 \times 8.59375$

$5.66015625 = 1.0110101001 \times 2^2$
$8.59375 = 1.0001001100 \times 2^3$

Exp: $2 + 3 = 5$, $5 + 16 = 21$ (10101)
Signs: both positive, result positive

Mantissa:

```
                        1.0110101001
                      × 1.0001001100
                        ------------
                         00000000000
                        00000000000
                       10110101001
                      10110101001
                     00000000000
                    00000000000
                   10110101001
                  00000000000
                 00000000000
                00000000000
               10110101001
              1.100001010010001011000
```

1.1000010100 10 00101100 Guard = 1, Round = 0, Sticky = 1: Round up

1.1000010101×2^5 = 011010100010101 (110000.10101 = 48.65625)

$5.66015625 \times 8.59375 = 48.6419677734375$

Some information was lost because the result did not fit into the available 10-bit field. Answer off by .0142822265625

b.

$6.18 \times 10^2 \times 5.796875 \times 10^1$

$6.18 \times 10^2 = 618 = 1.0011010100 \times 2^9$
$5.796875 \times 10^1 = 57.96875 = 1.1100111111 \times 2^5$

Exp: 9 + 5 = 14, 16 + 14 = 30 (11110)
Signs: both positive, result positive

Mantissa:

```
           1.0011010100
         × 1.1100111111
           ------------
            10011010100
           10011010100
          10011010100
         10011010100
        10011010100
       10011010100
      00000000000
     00000000000
    10011010100
   10011010100
  10011010100
 10001011111110000101100 Must Normalize, add one to exponent
```

1.0001011111 10 000101100 Guard = 1, Round = 0, Sticky = 1: round

$1.0001100000 \times 2^{15}$ = 0111110001100000(1000110000000000 = 35840)

$618 \times 57.96875 = 35824.6875$

Some information was lost because the result did not fit into the available 10-bit field. Answer off by 15.3125

3.12.2 No solution provided

3.12.3 No solution provided

3.12.4

a.

$3.264 \times 10^3 / 6.525 \times 10^2$

$3.264 \times 10^3 = 3264 = 1.1001100000 \times 2^{11}$
$6.525 \times 10^2 = 652.5 = 1.0100011001 \times 2^9$

Exponent = 11 – 9 = 2, 2 + 16 = 18 (10010)
Signs: both positive, result positive

Mantissa:

```
                                  1.0100000000100101
         10100011001. | 11001100000.0000000000000000
                       -10100011001.
                       ------------
                          101000111.0
                          101000111.00
                         -101000110.01
                         -------------
                                   .11000000000
                                  -.10100011001
                                   -------------
                                   .00011100111000
                                  -.00010100011001
                                   --------------
                                   .0000100001111100
                                  -.0000010100011001
                                   -----------------
                                   .0000001101100011
```

1.0100000000 10 0101 Guard = 1, Round = 0, Sticky = 1: Round up

1.0100000001×2^2 = 0100100100000001 = 101.00000001 = 5.00390625

3264/652.5 = 5.002298850575

Some information was lost because the result did not fit into the available 10-bit field. Answer off by .001607399425

b. $-2.27734375 \times 10^0/1.154375 \times 10^2$

$-2.27734375 \times 10^0 = -2.27734375 = -1.0010001110 \times 2^1$
$1.154375 \times 10^2 = 115.4375 = 1.1100110111 \times 2^6$

Exponent = 1 – 5 = –5, –5 + 16 = 11 (01011)
Signs: one negative, one positive, result negative

Mantissa:

```
                                  0.1010000110011101
         11100110111. | 10010001110.0000000000000000
                      -  1110011011.1
                       ------------
                           11110010.100
                         - 11100110.111
                           ------------
                               1011.10100000
                             - 111.00110111
                               ------------
                                100.011010010
                             -   11.100110111
                                ------------
                                   .110011011001
                                  -.011100110111
                                   ------------
                                   .0101101000010
                                  -.0011100110111
                                    ------------
                                   .00100000010110
                                  -.00011100110111
                                    ------------
                                   .0000101111110101
                                  -.0000011100110111
```

0.1010000110011101 need to normalize, decrement exponent, fix sign

–1.0100001100 11 101 Guard = 1, Round = 1, Sticky = 1: Round up

$-1.0100001101 \times 2^{-6}$ = 1010100100001101 = .0000010100001101 =
–.0197296142578125

–2.27724375/115.4375 = –.01972844990015983 08997743

Some information was lost because the result did not fit into the available 10-bit field. Answer off by .0000011643576527

3.12.5 No solution provided

3.12.6 No solution provided

Solution 3.13

3.13.1

a. $(-1.6360 \times 10^4 + 1.6360 \times 10^4) + 1.0 \times 10^0$

$-1.6360 \times 10^4 = -1.1111111010 \times 2^{13} = -11111111010000.$
$1.6360 \times 10^4 = 1.1111111010 \times 2^{13} = 11111111010000.$
$1.0 \times 10^0 = 1.0 = 1.0000000000 \times 2^0 = 1.0000000000$

```
(A)      -1.1111111010
(B)      +1.1111111010
          -------------
(A+B)     0.0000000000
(C)      +1.0000000000
          -------------
(A+B)+C   1.0000000000 = 0100000000000000 = 1
```

b. $(2.865625 \times 10^1 + 4.140625 \times 10^{-1}) + 1.2140625 \times 10^1$

$-2.865625 \times 10^1 = 1.1100101010 \times 2^4$
$4.140625 \times 10^{-1} = 1.1010100000 \times 2^{-2}$
$1.2140625 \times 10^1 = 1.1000010010 \times 2^3$

shift binary point of smaller left 6 so exponents match

```
(A)       1.1100101010
(B)        .0000011010 10 0000 Guard=1, Round=0, Sticky=0
          ------------
(A+B)     1.1101000100 No round
(A+B)     1.1101000100
(C)      + .1100001001 00 Guard=0, Round=0, Sticky=0
          -------------
(A+B)+C  10.1001001101 Normalize, add 1 to exponent
(A+B)+C = 1.0100100110 × 2^5 = 0101010100100110 = 41.1875
```

3.13.2

a. $-1.6360 \times 10^4 + (1.6360 \times 10^4 + 1.0 \times 10^0)$

$-1.6360 \times 10^4 = -1.1111111010 \times 2^{13} = -11111111010000.$
$1.6360 \times 10^4 = 1.1111111010 \times 2^{13} = 11111111010000.$
$1.0 \times 10^0 = 1.0 = 1.0000000000 \times 2^0 = 1.0000000000$

```
(B)       1.1111111010
(C)      + .0000000000 00 10000000000 Guard=0, Round=0, Sticky=1
          -------------
(B+C)     1.1111111010 Do not round
(A)      -1.1111111010
          -------------
A+(B+C)   0.0000000000
A+(B+C)   0.0000000000 = 0000000000000000 = 0
```

b.	$2.865625 \times 10^{1} + (4.140625 \times 10^{-1} + 1.2140625 \times 10^{1})$
	$-2.865625 \times 10^{1} = 1.1100101010 \times 2^{4}$
	$4.140625 \times 10^{-1} = 1.1010100000 \times 2^{-2}$
	$1.2140625 \times 10^{1} = 1.1000010010 \times 2^{3}$
	shift binary point of smaller left 6 so exponents match
	`(C)       1.1000010010`
	`(B)        .0000110101 00 000 Guard=0, Round=0, Sticky=0`
	`         -------------`
	`(C+B)     1.1001000111 No round`
	`(A)       1.1100101010`
	`(C+B)      .1100100011 10 Guard=1, Round=0, Sticky=0`
	`         -------------`
	`A+(B+C)  10.1001001101 10 Normalize, add 1 to exponent`
	`          1.0100100110 11 0 Guard=1, Round=1, Sticky=0, Round up`
	A+(B+C) = $1.0100100111 \times 2^{5}$ = 0101010100100111 = 41.21875

3.13.3

a.	No, they are not equal: (A + B) + C = 1, A + (B + C) = 0 (steps shown above). Exact: –16360 + 16360 + 1 = 1
b.	No, they are not equal: (A + B) + C = 41.1875, A + (B + C) = 41.21875 (steps shown above). Exact answer is 41.2109375

3.13.4

a.	$(4.8828125 \times 10^{-4} \times 1.768 \times 10^{3}) \times 2.50125 \times 10^{2}$
	(A) $4.8828125 \times 10^{-4} = 1.0000000000 \times 2^{-11}$
	(B) $1.768 \times 10^{3} = 1.1011101000 \times 2^{10}$
	(C) $2.50125 \times 10^{2} = 1.1111010001 \times 2^{7}$
	Exp: $-11 + 10 = -1$
	Signs: both positive, result positive
	Mantissa:
	`(A)                 1.0000000000`
	`(B)               × 1.1011101000`
	`                    ------------`
	`                   10000000000`
	`                 10000000000`
	`                10000000000`
	`               10000000000`
	`             10000000000`
	`            10000000000`
	`           ----------------------`
	`           1.10111010000000000000`
	A × B 1.1011101000 00 00000000 Guard = 0, Round = 0, Sticky = 0: No Round
	A × B $1.1011101000 \times 2^{-1}$
	Exp: 1 + 7 = 8
	Signs: both positive, result positive
	Mantissa:
	`(A×B)               1.1011101000`
	`(C)               × 1.1111010001`
	`                    ------------`
	`                     11011101000`
	`                 11011101000`
	`               11011101000`
	`              11011101000`
	`             11011101000`
	`            11011101000`
	`           11011101000`
	`          -----------------------`
	`          11.0101111110110110100 Normalize, add 1 to exponent`
	(A × B) × C 1.1010111111 01 101101000 Guard = 0, Round = 1, Sticky = 1: No Round
	(A × B) × C $1.1010111111 \times 2^{8} = 431.75$

b. $(4.721875 \times 10^1 \times 2.809375 \times 10^1) \times 3.575 \times 10^1$

(A) $4.721875 \times 10^1 = 1.0111100111 \times 2^5$
(B) $2.809375 \times 10^1 = 1.1100000110 \times 2^4$
(C) $3.575 \times 10^1 = 1.0001111000 \times 2^5$

Exp: 5 + 4 = 9
Signs: both positive, result positive

Mantissa:

```
(A)                    1.0111100111
(B)                  × 1.1100000110
                       ------------
                       10111100111
                      10111100111
                10111100111
               10111100111
              10111100111
             ----------------------
            10.10010111010001101010 Normalize, add 1 to exponent
             1.0100101110 10 001101011  Guard=1, Round=0, Sticky=1:
Round up
```

A × B $1.0100101111 \times 2^{10}$

Exp: 10 + 5 = 15
Signs: both positive, result positive

Mantissa:

```
(A × B)                  1.0100101111
(C)                    × 1.0001111000
                         ------------
                       10100101111
                      10100101111
                     10100101111
                    10100101111
                10100101111
               ----------------------
               1.01110010101000001000
               1.0111001010 10 000010000 Guard=1, Round=0, Sticky=1:
Round up
```

(A × B) × C $1.0111001011 \times 2^{15}$

3.13.5

a.	$4.8828125 \times 10^{-4} \times (1.768 \times 10^3 \times 2.50125 \times 10^2)$
	(A) $4.8828125 \times 10^{-4} = 1.0000000000 \times 2^{-11}$
	(B) $1.768 \times 10^3 = 1.1011101000 \times 2^{10}$
	(C) $2.50125 \times 10^2 = 1.1111010001 \times 2^7$
	Exp: 10 + 7 = 17
	Signs: both positive, result positive
	Mantissa:
	`(B)               1.1011101000`
	`(C)             × 1.1111010001`
	`                  ------------`
	`                   11011101000`
	`               11011101000`
	`             11011101000`
	`            11011101000`
	`           11011101000`
	`          11011101000`
	`         11011101000`
	`        ----------------------`
	`      11.010111111101101101000 Normalize, add 1 to exponent`
	`      1.1010111111 01 101101000 Guard=0, Round=1, Sticky=1: No Round`
	B × C $1.1010111111 \times 2^{18}$ OVERFLOW: Cannot be represented

b. $4.721875 \times 10^1 \times (2.809375 \times 10^1 \times 3.575 \times 10^1)$

(A) $4.721875 \times 10^1 = 1.0111100111 \times 2^5$
(B) $2.809375 \times 10^1 = 1.1100000110 \times 2^4$
(C) $3.575 \times 10^1 = 1.0001111000 \times 2^5$

Exp: 4 + 5 = 9
Signs: both positive, result positive

Mantissa:

```
(B)                        1.1100000110
(C)                      × 1.0001111000
                           ------------
                         11100000110
                        11100000110
                       11100000110
                      11100000110
                  11100000110
                 ----------------------
                 1.1111011000101101
                 1.1111011000 10 11010000 Guard=1, Round=0, Sticky=1:
Round up
```

B × C 1.1111011001×2^9

Exp: 5 + 9 = 14
Signs: both positive, result positive

Mantissa:

```
(A)                        1.0111100111
(B × C)                  × 1.1111011001
                           ------------
                            10111100111
                         10111100111
                        10111100111
                       10111100111
                      10111100111
                     10111100111
                    10111100111
                   10111100111
                  ---------------------
                 10.11100101000111001111 Normalize, add 1 to exponent
                  1.0111001010 00 111001111  Guard=0, Round=0, Sticky=1:
No Round
```

A × (B × C) $1.0111001010 \times 2^{15}$

3.13.6

a.	a) No: $(A \times B) \times C = 1.1010111111 \times 2^8 = 431.75$ $B \times C = 1.1010111111 \times 2^{18}$ OVERFLOW: Cannot be represented B and C are both large, so their product does not fit into the 16-bit floating point format being used.
b.	d) No: $(A \times B) \times C = 1.0111001011 \times 2^{15} = 47456$ $A \times (B \times C) = 1.0111001010 \times 2^{15} = 47424$ "Exact": $47.21875 \times 28.09375 \times 35.75 = 47424.225341796875$

Solution 3.14

3.14.1

a.

$1.5234375 \times 10^{-1} \times (2.0703125 \times 10^{-1} + 9.96875 \times 10^{1})$

(A) $1.5234375 \times 10^{-1} = 1.0011100000 \times 2^{-3}$
(B) $2.0703125 \times 10^{-1} = 1.1010100000 \times 2^{-3}$
(C) $9.96875 \times 10^{1} = 1.1000111011 \times 2^{6}$

Shift binary point 9 to the left, match exponents

```
(C)    1.1000111011
(B)     .0000000011  01  0100000 Guard=0,  Round=1,  Sticky=1
       ---------------
(B+C)  1.1000111110 × 2^6
```

Exp: $-3 + 6 = 3$
Signs: both positive, result positive

Mantissa:

```
(A)                      1.0011100000
(B+C)                  × 1.1000111110
                         ------------
                         10011100000
                        10011100000
                       10011100000
                      10011100000
                     10011100000
                 10011100000
                10011100000
A × (B+C)       1.1110011011 10 01000000 Guard=1, Round=0, Sticky=1:
Round up
```

$A \times (B + C)$ 1.1110011100×2^3

b.	$-2.7890625 \times 10^1 \times (-8.088 \times 10^3 + 1.0216 \times 10^4)$
	(A) $-2.7890625 \times 10^1 = -1.1011111001 \times 2^4$
	(B) $-8.088 \times 10^3 = -1.1111100110 \times 2^{12}$
	(C) $1.0216 \times 10^4 = 1.0011111101 \times 2^{13} = 10216$
	Shift binary point 4 to the left, match exponents
	`(C)     1.0011111101`
	`(B)     -.1111110011 0  Guard = 0,  Round = 0,  Sticky = 0:  no round`
	`        ---------------`
	`(B+C)   0.0100001010  Normalize, subtract 2 from exponent`
	(B + C) $1.0000101000 \times 2^{11}$
	Exp: 4 + 11 = 15
	Signs: one negative, one positive — sign negative
	Mantissa:
	`(A)                      1.1011111001`
	`(B+C)                  × 1.0000101000`
	`                         ------------`
	`              11011111001`
	`            11011111001`
	`        11011111001`
	`       -----------------------`
	`       1.1100111110 10 11101000 Guard=1, Round=0, Sticky=1: Round up`
	A × (B + C) –1.1100111111
	A × (B + C) $-1.1100111111 \times 2^{15}$

3.14.2

a.	$1.5234375 \times 10^{-1} \times (2.0703125 \times 10^{-1} + 9.96875 \times 10^{1})$
	(A) $1.5234375 \times 10^{-1} = 1.0011100000 \times 2^{-3}$
	(B) $2.0703125 \times 10^{-1} = 1.1010100000 \times 2^{-3}$
	(C) $9.96875 \times 10^{1} = 1.1000111011 \times 2^{6}$
	Exp: $-3 - 3 = -6$
	Signs: both positive, result positive
	Mantissa:
	`(A)                1.0011100000`
	`(B)              × 1.1010100000`
	`                   ------------`
	`                10011100000`
	`              10011100000`
	`            10011100000`
	`           10011100000`
	`A×B        10.0000010011 00 00000000 Normalize, add 1 to exponent`
	`A×B 1.0000001001 10 0 0...0 Guard=1, Round=0, Sticky=0: Round to even`
	A × B $1.0000001010 \times 2^{-5}$
	Exp: $-3 + 6 = 3$
	Signs: both positive, result positive
	Mantissa:
	`(A)                1.0011100000`
	`(C)              × 1.1000111011`
	`                   ------------`
	`                    10011100000`
	`                   10011100000`
	`                 10011100000`
	`                10011100000`
	`               10011100000`
	`           10011100000`
	`          10011100000`
	`A×C        1.1110010111 11 10100000 Guard=1, Round=1, Sticky=1: round up`
	A × C $1.1110011000 \times 2^{3}$
	Shift binary point 8 to the left, match exponents
	`A×C  +1.1110011000`
	`A×B    .0000000100 00 001010 Guard=0, Round=0, Sticky=1: No Round`
	`     -------------`
	`      1.1110011100`
	$(A \times B) + (A \times C) = 1.1110011100 \times 2^{3}$

b. $-2.7890625 \times 10^1 \times (-8.088 \times 10^3 + 1.0216 \times 10^4)$

(A) $-2.7890625 \times 10^1 = -1.1011111001 \times 2^4$
(B) $-8.088 \times 10^3 = -1.1111100110 \times 2^{12}$
(C) $1.0216 \times 10^4 = 1.0011111101 \times 2^{13} = 10216$

Exp: 4 + 12 = 16 OVERFLOW: Cannot Represent
Signs: both negative, result positive

Mantissa:

```
(A)                1.1011111001
(B)              × 1.1111100110
                   ------------
                   11011111001
                  11011111001
               11011111001
              11011111001
             11011111001
            11011111001
           11011111001
          11011111001
         ----------------------
        11.01110001001010110110 Normalize, add 1 to exponent
A×B     1.1011100010 01 010110110 Guard=0, Round=1, Sticky=1: No Round
```

A × B $1.1011100010 \times 2^{17}$ OVERFLOW: Cannot Represent

Exp: 4 + 13 = 17 OVERFLOW: Cannot Represent
Signs: one negative, one positive, result negative

Mantissa:

```
(A)                1.1011111001
(C)              × 1.0011111101
                   -----------
                    11011111001
                  11011111001
                 11011111001
                11011111001
               11011111001
              11011111001
             11011111001
         11011111001
        ------------------------
        10.00101100100000010101 Normalize, add 1 to exponent
A×C    -1.0001011001 00 000010101 Guard=0, Round=0, Sticky=1: No Round
```

A × C $-1.0001011001 \times 2^{18}$ OVERFLOW: Cannot Represent

```
A×C   -1.1011011111 × 2^18  OVERFLOW:  Cannot Represent
A×B     .1101110001 × 2^18  OVERFLOW:  Cannot Represent
       -------------
      -0.1101101110 × 2^18  OVERFLOW:  Cannot Represent
      -1.1011011100 × 2^17  OVERFLOW:  Cannot Represent
```

A × B + A × C $-1.1011011100 \times 2^{17}$ OVERFLOW: Cannot Represent

3.14.3

a.	a) Yes: A × (B + C) = 1.1110011100 × 2^3 = 15.21875, and (A × B) + (A × C) = 1.1110011100 × 2^3 = 15.21875 Exact: .15234375 × (.20703125 + 99.6875) = 15.2183074951171875
b.	d) No: While it is possible to calculate A × (B + C), it is not possible to calculate A × B or A × C—the intermediate steps are not representable in this FP format. A × (B + C) = –1.1100111111 × 2^{15} = 59360 A × B + A × C = –1.1011011100 × 2^{17} OVERFLOW: Cannot Represent Exact: –27.890625 × (–8088 + 10216) = 59351.25

3.14.4

	Answer	Sign	Exp	Exact?
a.	0 01111110 01010101010101010101011	+	–2	No
b.	1 01111101 00100100100100100100101	–	–3	No

3.14.5

a.	a + a + a = 1.0000000000000000000001 a × 3 = 1.0000000000000000000001 They are the same, but they should be 1.0000000000000000000000
b.	d + d + d + d + d + d + d = 1.0000000000000000000000011 d × 7 = 1.0000000000000000000000011

3.14.6 No solution provided

Solution 3.15

3.15.1

a.	1000 0000 0000 0000 0000 0000	0x.800000	Yes
b.	0001 1100 0111 0001 1100 0111	.1C71C7	No

3.15.2

a.	0101 0000 0000 0000 0000 0000	.50000	Yes
b.	0001 0001 0001 0001 0001 0001	.111111	No

3.15.3

a.	0111 0111 0111 0111 0111 0111	.777777	No
b.	0001 1010 0000 0000 0000 0000	.1A0000	Yes

3.15.4

a.	01111 00000 00000 00000	.F000	Yes
b.	00011 01010 00000 00000	.3A00	Yes