Chapter-3

3.3 0101 1110 1101 0100

The attraction is that each hex degit contains one of 16 different characters. (0-9, A-F). Since with 4 binary bits you can represent 16 different patterns, in hex each digit requires exactly 4 binary bits.

3.12 62×12

Step	Action	Multiplier	Multiplicand	Product
0	Initial values	001 010	000 000 110 010	0000 0000 0000
1	lsb=0, no OP Left mcand Rshift Mplier	000 101	000 001 100 100	000 0000 0000
2.	Prod = Prod + Mcand Lshift mcand Rshift Mplier	000 010	000 011 001 000	000 001 100 100
3.	lsb =0, no op L shift mcand Rshift Mplier	000 001	000 110 010 000	000 001 100 100
4.	Prod = Prod + Mcand Lshift mcand Rshift Mplier	000 000	0001 100 100 000	000 111 110 100
5.	lsb = 0, no op Lshift mcand Rshift Mplier	000 000	011 001 000 000	000 111 110 100
6	Lsb=0, no op Lshift mcand R shift Mplier	000 000	110 010 000 000	000 111 110 100

$3 \times 7 \rightarrow (0011)_2 \times (0111)_2$

Iteration	Step	Multiplier	Multiplicand	Product
0	Initial values	0111	0000 0011	0000 0000
1.	1a: 1 ⇒ Prod = Prod + Mcand	0111	0000 0011	0000 0011
	2: Shift left multiplicand	0111	0000 0110	0000 0011
	3: Shift right multiplier	0011	0000 0110	0000 0011
2.	1a: 1 ⇒ Prod = Prod + Mcand	0011	0000 0110	0000 1001
	2: Shift left multiplicand	0011	0000 1100	0000 1001
	3: Shift right multiplier	0001	0000 1100	0000 1001
3.	1a: 1 ⇒ Prod = Prod + Mcand	0001	0000 1100	0000 0101
	2: Shift left multiplicand	0001	0000 1000	0001 0101
	3: Shift right multiplier	0000	0001 1000	0001 0101

3.15

$2 \times 4 \rightarrow (0010)_2 \times (0100)_2$

Iteration	Step	Multiplier	Multiplicand	Product
0	Initial values	0100	0000 0010	0000 0000
1	1a: 1 ⇒ Prod = Prod + Mcand	0100	0000 0010	0000 0010
	2: Shift left multiplicand	0100	0000 0100	0000 0010
	3: Shift right multiplier	0010	0000 0100	0000 0010
2.	1a: 1 ⇒ Prod = Prod + Mcand	0010	0000 0100	0000 0110
	2: Shift left multiplicand	0010	0000 1000	0000 0110
	3: Shift right multiplier	0001	0000 1000	0000 0110
3.	1a: 1 ⇒ Prod = Prod + Mcand	0001	0000 1000	0000 1110
	2: Shift left multiplicand	0001	0001 0000	0000 1110
	3: Shift right multiplier	0000	0001 0000	0000 1110
4.				

Devision

3.19

7 ÷ 2

Iteration	Step	Quotient	Divisor	Remainder
0	Initial value	0000	0010 0000	0000 0111
1	1a: Rem = Rem - Div	0000	0010 0000	① 1 1 0 1 1 1
	2b: Rem<0 ⇒ +Div, sll Q, Q0=0	0000	0010 0000	0000 0 1 1 1
	3: Shift Div Right	0000	0001 0000	0000 0 1 1 1
2.	1. Rem = Rem - Div	0000	0001 0000	① 1 1 1 0 1 1 1
	2b: Rem<0 ⇒ +Div, sll Q, Q0=0	0000	0001 0000	0000 0111
	3. Shift Div Right	0000	0000 1000	0000 0111
3	1: Rem = Rem - Div	0000	0000 1000	1111 1111
	2b: Rem<0 ⇒ +Div, sll Q, Q0=0	0000	0000 1000	0000 0111
	3: shift Div Right	0000	0000 0100	0000 0111
4.	1: Rem = Rem - Div	0000	0000 0100	0000 0011
	2: Rem ≥0, sll Q, Q0=1	0001	0000 0100	0000 0011
	3. shift Div Right	0001	0000 0010	0000 0011
5.	1: Rem = Rem - Div	0001	0000 0010	0000 0001
	2: Rem ≥0, sll Q, Q0=1	0011	0000 0010	0000 0001
	3: shift Div Right	0011	0000 0001	0000 0001

#3.22

Iteration	Step	Quotient	Divisor	Remainder
0	Initial value	0000	0010 0000	0000 0011
1	1: Rem = Rem - Div	0000	0010 0000	① 1110 0011
	2b: Rem < 0 ⇒ +Div, sll Q, Q=0	0000	0010 0000	0000 0011
	3: Shift div right	0000	0001 0000	0000 0011
2.	1: Rem = Rem - Div	0000	0001 0000	① 1111 0011
	2b: Rem < 0 ⇒ +Div, sll Q, Q=0	0000	0001 0000	0000 0011
	3. shift div right	0000	0000 1000	0000 0011
3.	1: Rem = Rem - Div	0000	0000 1000	① 1111 1011
	2b: Rem < 0 ⇒ +Div, sll Q, Q=0	0000	0000 1000	0000 0011
	3: shift div right	0000	0000 0100	0000 0011
4.	1: Rem = Rem - Div	0000	0000 0100	① 1111 1111
	2b: Rem < 0 ⇒ +Div, sll Q, Q=0	0000	0000 0100	0000 0011
	3. Shift div right	0000	0000 0010	0000 0011
5.	1: Rem = Rem - Div	0000	0000 0010	② 0000 0001
	2b: Rem > 0 ; sll Q, Q = 1	0001	0000 0010	0000 0001
	3: shift div right	0001	0000 0001	0000 0001

3.22

0x0C000000 = 0000 1100 0000 0000 0000 0000 0000 0000

= 0 0001 1000 0000 0000 0000 0000 0000 0000

sigh is positive

exp = 0x18

= 24 - 127

= -103

there is a hidden 1

mantisa = 0

answer = 1.0×2^{-103}