Chapter 3

"Arithmetic for Computers"

→ 3.1:

Ans: $5ED4_{16} - 07A4_{16}$

$$\begin{array}{r} 5ED4 \\ -\ 07A4 \\ \hline 5730 \end{array}$$

→ 3.2:

Ans: The given values are,

$(5ED4)_{16} = (0101\ 1110\ 1101\ 0100)_2$

$(07A4)_{16} = (0000\ 0111\ 1010\ 0100)_2$

for signed numbers in sign magnitute format, those two number are positive. That's why 5ED4 - 07A4 = 5730.

So, the result in signed 16-bit hexadecimal numbers in sign-magnitute format is 5730.

→ 3.3:

Ans: The given value is 5EDA where represent the hexadecimal

$(5EDA)_{16} = (0101\ 1110\ 1101\ 01000)_2$

The attraction of hexadecimal is that the numbering system contains 16 different character (0-9, A-E)

It allows representing large binary numbers in more compactly where each hexadecimal represents 4bit binary number and make it easier for readable.

→ 3.4:

Ans:

$$(4365)_8 \rightarrow 1000\ \ 1111\ \ 0101$$
$$(3412)_8 \rightarrow 0111\ \ 0000\ \ 1010$$

$$(753)_8 \rightarrow 0001\ \ 1110\ \ 1011$$

So, in unsign 12bit octal numbers is $(753)_8$

→ 3.5:

Ans: Magnitude of 4365 → $(365)_8$ or $(245)_{10}$

Magnitude of 3412 → $(412)_8$ or $(266)_{10}$

To subtract 3412 from 4365 in sign magnitude format, $(365)_8 - (412)_8 = -3777_8$

→ 3.6:

Ans: $185_{10} - 122_{10} = (63)_{10}$

Indicate neither.

→ 3.7:

Ans: The given numbers in binary,

$(185)_{10} = 1011\ 1001$, $(122)_{10} = 0111\ 1010$

Assume that, these numbers are sign then we get,

$(-57)_{10} = (1010\ 1001)_2$ from 185, as the MSB indicates the sign bit.

Now, $(-57)_{10} + (122)_{10} = 65$ (neither)

→ 3.8:

Ans:

$(-57)_{10} - (122)_{10} = (-179)_{10}$

This is not fit with 8 bit sign magnitute format.

→ 3.9:

Ans:

$(151)_{10} = 1001\ 0111$

$(214)_{10} = 1101\ 0110$

from these two value we get (-105), and (-42)

$(-105) + (-42) = (-147)_{10}$. As the range is -128.

→ 3.10:

Ans: here,

$(-105)-(-42)=-63.$

→ 3.11:

Ans: $151+214=365$ where 8 bit unsigned range is 255.

→ 3.12:

Ans: Given that 6 bit unsigned integer 62 and 12 multiplication:

Iteration	Step	Multiplier	Multiplicand	Product
0	initial	001 010	000 000 110 010	000 000 000 000
1	lsb=0, no op Lshift Mcand R shift Mer	000 101	000 001 100 100	000 000 000 000
2	lsb=1 Prod = Prod+ Mcand lshift Mcand Rshift Mper	000 010	000 011 001 000	000 001 100 100
3	lsb=0, no op lshift Mcand Rshift Mper	000 001	000 110 010 000	000 001 100 100
4	lsb = 1 product = prod + mcand lshift mcand Rshift Mper	000 000	001 100 100 000	000 111 110 100
5	lsb=0, op	000 000	011 001 000 000	000 111 110 100
6	lsb = 0, no op	000 000	110 010 000 000	000 111 110 100

So the product is $(000\ 111\ 110\ 100)_2 = (764)_{18}$

→ 3.13:

Ans: Hexadecimal multiplication of 62×12;

Iteration	Step	Multiplier	Multiplicand	Product
0	initial	0110 0010	0000 0000 0001 0010	0000 0000 0000 0000
1	lsb = 0, no op Lshift Mcand Rshift Mper	0011 0001	0000 0000 0010 0100	0000 0000 0000 0000
2	lsb = 1, Prod = Prod + Mcand Lshift Mcand Rshift Mper	0001 1000	0000 0000 0100 1000	0000 0000 0010 0100
3	lsb = 0 Lshift Mcand Rshift Mer	0000 1100	0000 0000 1001 0000	0000 0000 0010 0100
4	lsb=0 Lshift Mcand Rshift Mer	0000 0110	0000 0001 0010 0000	0000 0000 0010 0100
5	lsb = 0 Lshift Mcand Rshift Mer	0000 0011	0000 0010 0100 0000	0000 0000 0010 0100
6	lsb = 1 Prod = Prod+Mcand Lshift Mcand Rshift Mer	0000 0001	0000 0100 1000 0000	0000 0010 0110 0100
7	lsb = 1 Prod = Prod+Mcad Lshift Mcand Rshift Mer	0000 0000	0000 0100 1000 0000	0000 0110 1110 0100
8	lsb = 0	0000 0000	0001 0010 0000 0000	0000 0110 1110 0100

So the product is $(0000\ 0110\ 1110\ 0100)_2 = (6E4)_{16}$

→ 3.14:

Ans: For hardware, loop takes (3×A) cycles. for software, loop takes (5×A) cycles,

So,

(3×8)×4 tu = 96 times units for hardware.

and (5×8) × 4 tu = 160 times units for hardware.

→ 3.15:

Ans: It takes B times units to get through an adder and there will be A-1 adders where word is 8 bits requiring 7 adders (8-1)

7 × 4 tu = 28 times.

→ 3.16:

Ans: It takes B time units to get through an adder and the adder are arranged in a tree structure as it works for faster multiplication,

It required $\log_2(A)$ levels which is 3 levels

So, 8 bit wite word required 7 adders in 3 levels (3 × 4 tu) = 12 tu

→ 3.18:

Ans: Given that 74/21 = 3 and remainder 9.

Iteration	Steps	Quotient	Divisor	Remainder
0	Initial	000 000	010 001 000 000	000 000 111 100
1	Rem=Rem-Div Rem<0, R+D, Q<< Rshift Div	000 000	001 000 100 000	000 000 111 100
2	Rem = Rem - Div Rem <0, R+D Q << Rshift	000 000	000 100 010 000	000 000 111 100
3	Rem=Rem-Div Rem<0, R+D Q << Rshift Div	000 000	000 010 001000	000 000 111 100
4	Rem=Rem-Div Rem <0, R+D Q<< Rshift Div	000 000	000 001 000 100	000 000 111 100
5	Rem = Rem - D Rem<0, R+D Q<< Rshift Div	000 000	000 000 100 010	000 000 111 100
6	Rem= Rem -D Rem>0, Q<<1 Rshift div	000 001	000 000 010 001	000 000 011 010
7	Rem = Rem-Div Rem>0, Q<<1 Rshift Div	000 0011	000 000 001 000	000 000 001 001

→ 3.19:

Ans:

Iteration	step	Divisor	Remainder / Quotient
0	initial	010 001	000 000 111 100
1	R << Rem = Rem - Div Rem <0, R+D	010 001	000 001 111 000
2	R << Rem = Rem - D R<0, R+D	010 001	000 011 110 000
3	R << Rem = Rem - Div R<0, R+D	010 001	000 111 100 000
4	R << Rem = Rem - Div R<0, R+D	010 001	001 111 000 000
5	R << Rem = Rem - Div Rem ≥0, R_0=1	010 001	001 101 000 001
6	R << Rem = Rem - Div Rem ≥0, R_0=1	010 001	001 001 000 011

→ 3.20:

Ans: Given 0x0C000000 = 0000 1100 0000 0000 0000 0000 0000 0000

the decimal number and the unsigned integer, it does represent is 201326592.

→ 3.22:

Ans: Given 0x 0C000000 = 0000 1100 0000 0000 0000 0000 0000 0000

here, the sign bit = 0

the exponent = 24 - 127 = - 103

and significand = 0

We know that, $(1.0+0) \times 2^{-103} = 1.0 \times 2^{-103}$

→ 3.23:

Ans: The binary representation of 63.25 is,

$111111.01 = 1.1 \times 2^5$

Sign positive, exp = 127 + 5 = 132

final pattern = 0 1000 0100 1111 1010 0000 0000 0000 0000

→ 3.24:

Ans: $63.24 \times 10^0 = 111111.01 \times 2^0$

Normalizing, 1.1111101×2^5

sign = 0 and exponent = 1023 + 5 = 1028

where 1 bit for sign, 11 bits for exponent and 52 bit for significand.

→ 3.25:

Ans: $63.25 \times 10^{10} = 3F.4 \times 16^{0}$

for normalized, 0.340×16^{2}

sign = 0, As single precision, exp = 64+2 = 66
where 1 bit for sign, 8 bits for exponent and 23 bit for significand.

→ 3.26:

Ans: $-1.5625 \times 10^{-1} = -0.15625 \times 10^{0}$

$= -0.00101 \times 2^{0}$

Normalizing, -0.101×2^{-2}

exponent = -2 fraction = $-0.101 \ldots 0$

Here, ex = $2047 + (-2) = 2045$

→ 3.27:

Ans: $-1.5625 \times 10^{-1} = -0.15625 \times 10^{0}$

$= 1.01 \times 10^{-3}$

exponent = $-3 + 15 = 12$

→ 3.30:

Ans: Given that,

$-8.0546875 \times 10^{0} \times (-1.79931640625 \times 10^{-1})$

$= 1.449293137$

$-8.0546875 \times 10^0 = -1.0000000111 \times 2^1$

$-1.79931640625 \times 10^1 = -1.0111000010 \times 2^1$

Exp: $-3+3=0$; $0+16=16$ (10000)

both has negative sign

$$\begin{array}{r} 1.0000000111 \\ \times\ 1.0111000010 \\ \hline 1.01110011000000100 1110 \end{array}$$

Guard = 0, Round = 0, Sticky = 1.

→ 3.32:

Ans: Given that,

$(3.984375 \times 10^{-1} + 3.4375 \times 10) + 1.771 \times 10^3$

here, $0.3984375 = 0.0110011000000$

$\therefore A = 1.1001100000 \times 2^{-2}$

$0.34375 = 0.010110000000$

$\therefore B = 1.0110000000 \times 2^{-2}$

$1.771 \times 10^3 = 1771 = 0110\ 1110\ 1011$

$= 1.1011101011 \times 2^{10}$

$$
\begin{array}{rr}
(A) & 1.1001100000000 \\
(B)\ + & 1.0110000000000 \\
\hline
 & 10.111110000 0
\end{array}
$$

Normalize, $(A+B) = 1.0111110000000 \times 2^{-1}$

$(A+B)+C = 0110\,1010\,1110\,1100 = 1772$

→ 3.33:

Ans: $3.984375 \times 10^{-1} + (3.4375 \times 10^{-1} + 1.771 \times 10^{3})$

A, $(0.3984375) = 1.1001100000 \times 2^{-2}$

B, $(0.34375) = 1.011000000 \times 10^{-2}$

C, $1.771 \times 10^{3} = 1771 = 1.10111\ 01011 \times 2^{10}$

$(B+C) = +1.1011101011$

$A + (B+C) = +1.1011101011 \times 2^{10}$

$= 0110\,1010\,1110\,1011 = 1771$