Karan Vora (kv2154)
Computer Systems Architecture Assignment 4

Problem 1):

For unsigned bits, the largest number that can be denoted is 2^N -1 where N is the number of bits so the range possible is 0 to $2^N - 1$ anything outside of this and overflow is raised
For signed bits, the first bit is dedicated for sign where 0 denotes a positive number and 1 denotes a negative number. The largest denomination possible is $2^{N-1} - 1$ so the range of possible number is $-2^{N-1} - 1$ to $2^{N-1} - 1$ anything outside of this range and overflow is raised

(A) 0110 1110 + 1001 1111

Unsigned Addition = 110 + 159 = 269 > $2^8 - 1$, overflow has occurred since 269 cannot be represented in 8 bits.
Signed Addition = 110 – 97 = 13 < $2^8 - 1$, no overflow has occurred since 13 can be represented in 8 bits in signed form.

(B) 1111 1111 + 0000 0001

Unsigned Addition = 255 + 1 = 256 > $2^8 - 1$, overflow has occurred since 256 cannot be represented in 8 bits.
Signed Addition = -1 + 1 = 0 < $2^8 - 1$, no overflow has occurred since 0 can be represented in 8 bits in signed form.

(C) 1000 0000 + 0111 1111

Unsigned Addition = 128 + 127 = 255 = $2^8 - 1$, no overflow has occurred since 255 can be represented with 8 bits.
Signed Addition = -128 + 127 = -1, can be represented in 2’s complement form in 8 bits, no overflow has occurred.

(D) 0111 0001 + 0000 1111

Unsigned Addition = 113 + 15 = 128 < $2^8 - 1$, no overflow has occurred since 128 can be represented with 8 bits.
Signed Addition = 113 + 15 = 128, cannot be represented in 2’s complement form, an overflow will occur.

==

Problem 2):

To represent the result, we will need 16 bits
$AB_{hex} = 171$
$Ef_{hex} = 239$

239 = 256 – 17, 17 = 16 + 1
$256 = 2^8$, $16 = 2^4$

Now,
Left shift 171 4 times and add 171 = (171 x 16) + 171 = 171 x 17 ---- (1)
Left shift 171 8 times = (171 x 256) ---- (2)
Perform (2) – (1) to get the required result

(171 x 256) – [(171 x 16) + 171] = 40869

In binary = 1001111110100101
In Hex = 9FA5

In total we get answer in 14 steps, 12 shift lefts, 1 addition and 1 subtraction

==

Problem 3):

32 bits is Single Precision Floating Point
0xDEADBEEF in binary is 1101 1110 1010 1101 1011 1110 1110 1111
sign bit = 1 => -1
exponent = 10111101 => Exponent bias = 189 – 127 = 62

Fraction = 01011011011111011101111 = $1 + 2^{-2} + 2^{-4} + 2^{-5} + 2^{-7} + 2^{-8} + 2^{-10} + 2^{-11} + 2^{-12} + 2^{-13} + 2^{-14} + 2^{-16} + 2^{-17} + 2^{-18} + 2^{-20} + 2^{-21} + 2^{-22} + 2^{-23}$ = 1.357389330863952629979 99

Decimal representation = $-1.35738933086395262997999 \times 2^{62}$ =
-6,259,853,398,707,797,984.9229547264623

==

Problem 4):

Because the number is positive, the sign bit is zero. 78 in binary is 1001110. 0.75 in binary is 0.110.

78.75 = 1001110.110 = 1.00111011×2^{6}

Exponent in 32 bits = 6 + 127 = 133 = 10000101
Exponent in 64 bits = 6 + 1023 = 1029 = 10000000101

32 bit = 01000010100111011000000000000000
64 bit = 0100000001011000010100000000000000000000000000000000000000000000

==

Problem 5):

Because the number is positive, the sign bit is zero. 78 in base 16 is 4E. 0.75 in base 16 is .C

78.75 in base16 representation 4E.C = 4.EC * 16

78.75 in binary representation 01001110.110

Shift right = 0.01001110110 x 16^2

bias of 64 added to exponent of 2, 66: 1000010 binary represent

32 bit rep with base 16: 01000010010011101100000000000000

==

Problem 6):

(A) Number = - 0.13625
Number is negative so the sign bit is 1, binary representation = 0.001000101110000101 = 1.000101110000101 * 2^{-3}.
Exponent = -3 + 15 = 12
16 bit representation = 101100000101110 000101, the bits that will be truncated due to only 10 bits of precision, it will need 32 bits to be represented perfectly.
Range of numbers in 16 bit = 2^{-14} to 2^{15}.
Range of numbers in single precision = 2^{-126} to 2^{127}

(B) 1.6125 x 10 = 16.125
Binary representation = 10000.001 = 1.0000001 x 2^4
Exponent with bias = 15 + 4 = 19, binary representation = 10011
16 bit representation = 0100110000001000

3.150390625 x 10^{-1} = 0.3150390625
Binary representation = 0.0101000010100110011 = 1.0100001010 0110011 x 2^{-2}
Exponent with bias = 15 – 2 = 13, binary representation = 1101
16 bit representation = 0011010100001010

We can’t directly add the binary representations because they don’t have the same exponents, we can shift (A) to left by 6 bits

1.0100001010
+ .00010000001 (Truncation error)
1.0101001010 x 2^4

Already normalized, no errors while adding the numbers. I do not know what to do with the Truncation and representation errors

==

Problem 7):

Single precision
Mantissa bits = 23
Exponent bits = 8
Exponent bias = 127

Smallest positive number = 2^{-126}

fp16
Mantissa bits = 10
Exponent bits = 5
Exponent bias = 15
Smallest positive number = 2^{-14}

bfloat16
Mantissa bits = 7
Exponent bits = 8
Exponent bias = 127
Smallest positive number = 2^{-126}

==

Problem 8):

Solution A):

Number	binary	Decimal
0	0 000 000	0.0
-0.125	1 000 100	-0.125
Smallest positive normalized number	0 001 000	0.25
Largest postive normalized number	0 110 111	15.0
Smallest postive denormalized number > 0	0 000 001	1/32 = 0.03125
largest postive denormalized number > 0	0 000 111	0.875

Solution B):

Let
a = 1 11110 1111111111
b = 0 11110 1111111111
Let c = 1.02 x 2-15
-15 = EXP – 15 => EXP = 0
c = 0 00000 0000000000
c is outside of the range that can be represented by a 16 bit FP standard
Now, a and b cancel each other (one is negative of the other)
Then (a + b) + c = c, because a and b cancel each other, however,
a + (b + c) = 0, because b + c = b (c is very small relative to b and is lost in underflow).