**ICS 233 Homework Assignment 01 Key – Term 182**

1. Compute the final result of each of the following unsigned operations and for each determine whether there is unsigned overflow or not. Give reasons for each of your answer.
2. 21010 + 10910 (8-bit operands)

11

1101 0010 21010

+ 0110 1101 10910

0011 1111 6310

The carry bit of 1 indicates unsigned overflow

1. BFA6H + BFE7H (16-bit operands)

111

BFA6H

+ BFE7H

7F8DH

The carry bit of 1 indicates unsigned overflow

1. 2BD6H – FEA3H (16-bit operands)

2BD6H

- FEA3H

(b) 2D33H

The unsatisfied borrow of indicates unsigned underflow

1. 8210 - 23510 (8-bit operands)

82

- 235

-153

-153 < 0 🡺 unsigned underflow

1. 11010010B + 01101101B (8-bit operands)

11

11010010B

+ 01101101B

10111111B

The carry bit of 1 indicates unsigned overflow

1. Compute the final result of each of the following signed r’s complement operations and for each determine whether the operation will result in signed overflow or not. Show your steps, and give reason for your answer.
2. 00111111B + 11010101B (8-bit operands)

11111 111

0011 1111B

+ 1101 0101B

0001 0100B

Sign of addends are different 🡺 no signed overflow

Alternatively Carryin = Carryout 🡺 no signed overflow

1. 6310 + 10010 (8-bit operands)

01111 100

0011 1111B 6310

+ 0110 0100B 10010

1010 0011B -9310

The sign of the result is different to the addends 🡺 signed overflow

Alternatively Carryin ≠ Carryout 🡺 signed overflow

1. -6310 – 10010 (8-bit operands)

10000 000

1100 0001B -6310

+ 1001 1100B -10010

0101 1101B 9310

The sign of the result is different to the addends 🡺 signed overflow

Alternatively Carryin ≠ Carryout 🡺 signed overflow

1. 91H + 2BH (8-bit operands)

91H + 2BH = BCH

Sign of the addends are different 🡺 No signed overflow

1. B0H – 78H (8-bit operands)

B0H – 78H = B0H + (FFH – 78H + 1H)

= B0H + 88H

= 38H (discard final carry)

The sign of the result is different to the addends 🡺 signed overflow

1. Express the negative value -22 as a 2's complement integer, using eight bits. Repeat it for 16 bits and 32 bits. What does this illustrate with respect to the properties of sign extension as they pertain to 2's complement representation?

-2210 = 11101010B (8-bit)

= 11111111 11101010B (16-bit)

= 11111111 11111111 11101010B (32-bit)

Sign extending does not alter the magnitude of a value.

1. Provide the decimal representation of each of the following IEEE 32-bit floating point values. Show all steps.

* **3CC80000H**

3CC80000H = 0 011 1100 1 100 1000 0 ...0

Sign bit = 0

Exponent = 01111001 = 121

n = 121 – 127 = -6

F = 1001

00000000 < E < 11111111 🡺 Normal number

N = 1.10012 \* 2-6

= 1.5625 \* 2-6

Let 10x = 2-6

🡺x = -6 log 2

= -1.806179974

= -1 - 0.806179974

= (-1 – 1) + (1 - 0.806179974)

= -2 + 0.193820026

🡺 N = 1.5625 \* 100.193820026\* 10-2

= 2.44140625 \* 10-2

----------------------------------------------------------------------------------------

* **61440000H**

Exponent = 11000010 = 194

n = 194 – 127 = 67

F = 10001000000000000000000

00000000 < E < 11111111 🡺 Normal number

N = 1.100012 \* 267

= 1.53125 \* 267

Let 10x = 267

🡺x = 67 log 2

= 20.16900971

N = 1.53125 \* 100.1690071 \* 1020

=2.25972615 \* 1020

----------------------------------------------------------------------------------------

* **C0E00000H**

C0E00000H = 1100 0000 1110 0000 . . . 0000

Sign bit = 1

E = 10000001 = 129

n = 129 – 127 = 2

F = 11000000000000000000000

00000000 < E < 11111111 🡺 Normal number

N = -1.112 \* 22 = -111.02 = -7.0

-----------------------------------------------------------------------------------------

* **40F00000H**

40F00000H = 0100 0000 1111 0000 ... 0000

E= 10000001 = 129

n = 129 – 127 = 2

00000000 < E < 11111111 🡺 Normal number

F = 11100000000000000000000

N = 1.1112 \* 22 = 111.12 = 7.5

-----------------------------------------------------------------------------------------

* **FF800000H**

FF800000H = 1111 1111 1000 0000 ... 0000

Sign bit = 1 🡺 -ve number

E = 11111111 and F = 00000000000000000000000 🡺 -∞

----------------------------------------------------------------------------------------

* **00500000H**

00500000H = 0 000 0000 0101 0000 ...0000

E = 00000000

F = 1010 ...0

E = 00000000 and F ≠ 0 🡺 subnormal number

Subnormal numbers: N = (-1)s \* (0.Fraction) \* 2(1 – Bias)

N = (-1)0 \* 0.1012 \* 2-126

= 0.625 \* 2-126

Let 10x = 2-126

🡺 x = -126 \* log(2)

= -37.92977945

= -37 + -0.92977945

= (-37 – 2) + (2 - 0.92977945)

= -39 + 1.07022055

🡺2-126 = 101.07022055 \* 10-39

= 11.75494361 \* 10-39

🡺N = 0.625 \* 11.75494361 \* 10-39

= 7.346839755 \* 10-39

-----------------------------------------------------------------------------------------

* **80200000H**

80200000H = 1000 0000 0 010 0000 ... 0000

Sign bit = 1 🡺 -ve number

E = 00000000

F = 0100 ... 0

E = 0 and F ≠ 0 🡺 subnormal number

Subnormal numbers: N = (-1)s \* (0.Fraction) \* 2(1 – Bias)

N = - 0.012 \* 2-126

= - 0.2510 \* 2-126

Let 10x = 2-126

🡺 x = -126 \* log(2)

= -37.92977945

= -37 + -0.92977945

= (-37 – 2) + (2 - 0.92977945)

= -39 + 1.07022055

🡺2-126 = 101.07022055 \* 10-39

= 11.75494361 \* 10-39

🡺 N = - 0.25 \* 11.75494361 \* 10-39

= -2.938735903 \* 10-39

---------------------------------------------------------------------------------------

* FFAB0000H

FFAB0000H = 1111 1111 1010 1011 0000 ... 0000

Sign bit = 1 🡺 negative number

E = 11111111 and F ≠ 0 🡺 -NaN

----------------------------------------------------------------------------------------

1. What decimal number does the hexadecimal representation **C01D600000000000H** of a double value represent? Show all steps.

C01D600000000000H = 1100000000011101011000000000 . . . 0

🡺Sign bit = 1

Exponent = 100 0000 0001 = 1025

n = 1025 – 1023 = 2

F = 1101 0110 0000 0000 . . . 0

N = (-1)1 \* (1.Fraction) \* 22

N = -1. 1101 011 \* 22 = -111.010112 = -(7 + 2-2 + 2-4 + 2-5)

= -(7 + 0.25 + 0.0625 + 0.03125)

= -7.34375

1. Convert the IEEE 754 80-bit extended floating point representation **E003F900000000000000H** to decimal. Show all steps.

E003F900000000000000H = 1 110 0000 0000 0011 1111 1001 0000 … 0

Sign bit = 1

E = 110000000000011 = 24579

n = 24579 – 16383 = 8196

F = 111100100 . . . 0

N = -1.1111001 \* 28196

1.1111001 = 1 + ½ + ¼ + 1/8 + 1/16 + 1/64

= 1.953125

N = -1.953125 \* 28196

Let 10x = 28196

🡺 x = 8196 log 2

= 2467.241844

= 2467 + 0.241844

🡺10x = 100.241844 \* 102467

= 1.745195161 \* 102467

🡺N = -1.953125 \* 1.745195161 \* 102467

= -3.408584298 \* 102467

---------------------------------------------------------------------------------------------------------------------------------------------------

1. Provide the hexadecimal IEEE 32-bit , 64-bit and x86 extended 80-bit floating point representations of each of the following decimal numbers. Show all steps.

* **-125.9**

125 = 01111101

0.9 \* 2 = 1.8

0.8 \* 2 = 1.6

0.6 \* 2 = 1.2

0.2 \* 2 = 0.4

0.4 \* 2 = 0.8

Sign bit = 1

N = - 01111101.1

= - 1. 1111011 \* 26

Exponent = 6 + Bias

**Binary32**: Exponent = 6 + 127 = 133 = 10000101B

F = 1111011 1100 1100 1100 1100

N = 1 100 0010 1 1111011 1100 1100 1100 1100

= 1100 0010 1111 1011 1100 1100 1100 1100

= C 2 F B C C C C

N = C2FBCCCCH

**Binary64**: Exponent = 6 + 1023 = 1029 = 10000000101B

F = 1111011 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1

N=1100 0000 0101 1111 0111 1001 1001 1001 1001 1001 1001 1001 1001 1001 1001 1001

= C 0 5 F 7 9 9 9 9 9 9 9 9 9 9 9

= C05F799999999999H

**Extended**: Exponent = 6 + 16383 = 16389 = 100000000000101

F=1111011 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100

N=1100 0000 0000 0101 1111 1011 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100 1100

= C 0 0 9 F B C C C C C C C C C C C C C C

= C009FBCCCCCCCCCCCCCCH

--------------------------------------------------------------------------------------

* **47.523**

47 = 00101111

0.523 \* 2 = 1.046

0.046 \* 2 = 0.092

0.092 \* 2 = 0.184

0.184 \* 2 = 0.368

0.368 \* 2 = 0.736

0.736 \* 2 = 1.472

0.472 \* 2 = 0.944

0.944 \* 2 = 1.888

0.888 \* 2 = 1.776

0.776 \* 2 = 1.552

0.552 \* 2 = 1.104

0.104 \* 2 = 0.208

0.208 \* 2 = 0.416

0.416 \* 2 = 0.832

0.832 \* 2 = 1.664

0.664 \* 2 = 1.328

0.328 \* 2 = 0.656

0.656 \* 2 = 1.312

0.312 \* 2 = 0.624

0.624 \* 2 = 1.248

0.248 \* 2 = 0.496

0.496 \* 2 = 0.992

0.992 \* 2 = 1.984

0.984 \* 2 = 1.968

0.968 \* 2 = 1.936

0.936 \* 2 = 1.872

0.872 \* 2 = 1.744

0.744 \* 2 = 1.488

0.488 \* 2 = 0.976

0.976 \* 2 = 1.952

0.952 \* 2 = 1.904

0.904 \* 2 = 1.808

0.808 \* 2 = 1.616

0.616 \* 2 = 1.232

0.232 \* 2 = 0.464

0.464 \* 2 = 0.928

0.928 \* 2 = 1.856

0.856 \* 2 = 1.712

0.712 \* 2 = 1.424

0.424 \* 2 = 0.848

0.848 \* 2 = 1.696

0.696 \* 2 = 1.392

0.392 \* 2 = 0.784

0.784 \* 2 = 1.568

0.568 \* 2 = 1.136

0.136 \* 2 = 0.272

0.272 \* 2 = 0.544

0.544 \* 2 = 1.088

0.88 \* 2 = 0.176

0.176 \* 2 = 0.352

0.352 \* 2 = 0.704

0.704 \* 2 = 1.408

0.408 \* 2 = 0.816

0.816 \* 2 = 1.632

0.632 \* 2 = 1.264

0.264 \* 2 = 0.528

0.528 \* 2 = 1.056

0.056 \* 2 = 0.112

00101111.10000 1011110 001101

1.01111 100001011110 001101 \* 25

Sign bit = 0

Exponent = 5 + Bias

**Binary32**: Exponent = 5 + 127 = 132 = 10000100

F = 01111 10000 1011110 001101

N = 0 10000100 01111 10000 1011110 001101

= 0100 0010 0011 1110 0001 0111 1000 1101

= 4 2 3 E 1 7 8 D

= 423E178DH

**Binary64**: Exponent = 5 + 1023 = 1028 = 10000000100

F= 01111 10000 1011110 001101 010011 111101 1111001 1101101101

N=0100 0000 0100 0111 1100 0010 1111 0001 1010 1001 1111 1011 1110 0111 0110 1101

= 4 0 4 7 C 2 F 1 A 9 F B E 7 6 D

= 4047C2F1A9FBE76D

**Extended**: Exponent = 5 + 16383 = 16388 = 100000000000100

F= 01111 10000 1011110 001101 010011 111101 1111001 1101101100 10001011010

N = 0100 0000 0000 0100 1011 1110 0001 0111 1000 1101 0100 1111 1101 1111 0011 1011 0110 0100 0101 1010

= 4 0 0 4 B E 1 7 8 D 4 F D F 3 B 6 4 5 A

= 4004BE178D4FDF3B645AH

-----------------------------------------------------------------------------------------

**173.7**

173 = 10101101

0.7 \* 2 = 1.4

0.4 \* 2 = 0.8

0.8 \* 2 = 1.6

0.6 \* 2 = 1.2

0.2 \* 2 = 0.4

0.4 \* 2 = 0.8

10101101.1 0110 0110 0110 0110 0110

1. 0101101 1 0110 0110 0110 011 \* 27

Sign bit = 0

Exponent = 6 + Bias

**Binary32**: Exponent = 7 + 127 = 134 = 10000110

F = 01011011011001100110011

N = 0 10000110 01011011011001100110011

= 0100 0011 0010 1101 1011 0011 0011 0011

= 4 3 2 D B 3 3 3

= 432DB333H

**Binary64**: Exponent = 7 + 1023 = 1030 = 100000000000110

F = 0101101 1 0110 0110 0110 0110 0110 0110 0110 0110 0110 0110 0110

N = 0100 0000 0110 0101 1011 0110 0110 0110 0110 0110 0110 0110 0110 0110 0110 0110

= 4 0 6 5 B 6 6 6 6 6 6 6 6 6 6 6

= 4065B66666666666H

**Extended**: Exponent = 7 + 16383 = 16390 = 10000000110

F = 0101101 1 0110 0110 0110 0110 0110 0110 0110 0110 0110 0110 0110 0110 0110 011

N = 0100 0000 0000 0110 1010 1101 1011 0011 0011 0011 0011 0011 0011 0011 0011 0011 0011 0011 0011 0011

= 4 0 0 6 A D B 3 3 3 3 3 3 3 3 3 3 3 3 3

= 4006ADB3333333333333H