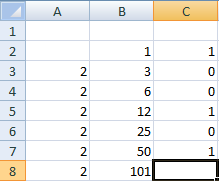
CISP 440

Austin Smothers

Homework 0

1. What is the range of unsigned 16 bit numbers in decimal and in binary?
   1. 216 – 1 = 65,535 **Range: 0 to 65,535**
2. Convert decimal 101 to an 8-bit binary number.

 🡪 **011001012 = 10110**

1. Convert decimal -101 to an 8 bit 2’s compliment number.
   1. 011001012 (10110) 🡪 **100110112 = -10110**
2. What is the decimal value of the 2’s complement number 11011001?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |

-128 + 64 + 0 + 16 + 8 + 0 + 0 + 1 =

**-3910 = 1101 10012**

1. What are the largest and smallest decimal values representable in 2’s complement using 8 bits?
   1. 28-1 – 1 = 127 = Largest 127 \* -1 – 1 = -128 = Smallest

**Largest: 127 Smallest: -128**

1. What are the largest and smallest decimal values representable in 2’s complement using 16 bits?
   1. 216-1 – 1 = 32,767 = Largest 32,767 \* -1 – 1 = -32,768 = Smallest

**Largest: 32,767 Smallest: -32,768**

1. How many values can be represented with 42 bits using 2’s complement?
   1. 242 = **4,398,046,511,104 values**
2. Convert decimal 48 to an 8 bit binary number
   1. 48 – 32 = 16 – 16 = 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |

**0011 00002 = 4810**

1. Convert decimal -48 to an 8 bit 2’s complement number
   1. -128 + 64 = -64 -64 + 16 = -48

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |

**1101 00002 = -4810**

1. Convert decimal 38 to an 8 bit binary number
   1. 38 – 32 = 6 6 – 4 = 2 2 – 2 = 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |

**0010 01102 = 3810**

1. Convert decimal -38 to a 2’s complement number
   1. -128 + 64 = -64 -64 + 16 = -48 + 8 = -40 -40 + 2 = -38

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |

**1101 10102 = -3810**

1. What is the decimal value of the 8 bit 2’s complement number 1101 1001?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |

-128 + 64 + 0 + 16 + 8 + 0 + 0 + 1 =

**-3910 = 1101 10012**

1. What is the decimal value of the 8 bit 2’s complement number 1001 1000?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |

-128 + 0 + 0 + 16 + 8 + 0 + 0 + 0 =

**-10410 = 1001 10002**

1. What is the decimal value of the 16 bit 2’s complement number 1101 1001 1001 1000?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -32768 | 16384 | 8192 | 4096 | 2048 | 1024 | 512 | 256 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| -32,768 + 16,384 + 0 + 4,096 + 2,048 + 0 + 0 + 256 | | | | | | | |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |

+ 128 + 0 + 0 + 16 + 8 + 0 + 0 + 0 =

**-983210 = 1101 1001 1001 10002**

1. What is the decimal value of the 16 bit 2’s complement number 0101 1001 1001 1000?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -32768 | 16384 | 8192 | 4096 | 2048 | 1024 | 512 | 256 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 + 16,384 + 0 + 4,096 + 2,048 + 0 + 0 + 256 | | | | | | | |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |

+ 128 + 0 + 0 + 16 + 8 + 0 + 0 + 0 =

**22,93610 = 0101 1001 1001 10002**

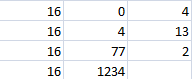
1. What is the decimal value of the 8 bit 2’s complement number 0000 0101?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

0 + 0 + 0 + 0 + 0 + 4 + 0 + 1 =

**510 = 0000 01012**

1. Convert 1234 base 10 to base 16

 🡪 **04D216 = 123410**

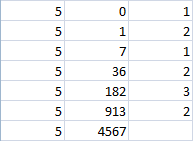
1. Convert 1234 base 16 to base 10

|  |  |  |  |
| --- | --- | --- | --- |
| 4096 | 256 | 16 | 1 |
| 1 | 2 | 3 | 4 |

4096 + 512 + 48 + 4 =

**466010 = 123416**

1. Convert 4567 base 10 to base 5

 🡪 **001212325 = 456710 This answer assumes an 8 bit or**

**larger system**

1. Convert 1234 base 5 to base 10

|  |  |  |  |
| --- | --- | --- | --- |
| 125 | 25 | 5 | 1 |
| 1 | 2 | 3 | 4 |

125 + 25 + 15 + 4 =

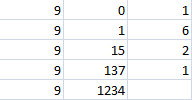
**016910 = 12345**

1. Convert 1234 base 7 to base 9

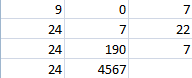
|  |  |  |  |
| --- | --- | --- | --- |
| 343 | 49 | 7 | 1 |
| 1 | 2 | 3 | 4 |

343 + 98 + 21 + 4 =

46610 = 12347

 🡪 **16219 = 123410**

1. Convert 4567 base 10 to base 24

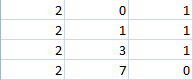
 🡪 **07M724 = 456710**

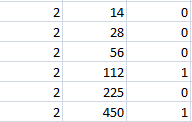
1. Convert 4567 base 24 to base 2

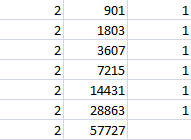
|  |  |  |  |
| --- | --- | --- | --- |
| 13,824 | 576 | 24 | 1 |
| 4 | 5 | 6 | 7 |

55,296 + 2,280 + 144 + 7 =

5772710 = 456724 **This answer assumes a 32 bit or larger system.**





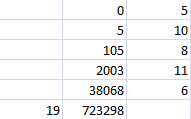
 🡪 **0 1110 0001 0111 11112 = 456724**

1. Convert FIVE base 36 to base 19 (FIVE != 5)

|  |  |  |  |
| --- | --- | --- | --- |
| 46,656 | 1296 | 36 | 1 |
| F(15) | I(18) | V(31) | E(14) |

699,840 + 23,328 + 1,116 + 14 =

72329810 = FIVE36 **This answer assumes a 16 bit or larger system**

 🡪 **05A8B619 = FIVE36**

1. Convert THREE base 28 to base 10 (THREE != 3)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 17,210,368 | 614,656 | 21,952 | 784 | 28 | 1 |
| 2 (carried) | T(29) 🡪 27 | H(17) | R(27) | E(14) | E(14) |

34420736 + 16595712 + 373184 + 21168 + 392 + 14 =

**5141120610 = THREE28**

1. Convert 1234 base 10 to base 36

Problem26.png 🡪 **YA36 = 123410**

1. The IEEE standard uses excess 127 format for exponents. What would be the decimal exponent value of 0110 1001?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |

0 + 64 + 32 + 0 + 8 + 0 + 0 + 1 =

105 – 127 (excess) = **-2210 = 0110 10012 (excess 127)**

1. The IEEE standard uses excess 127. Would excess 128 be better? Explain.
   1. **Excess 128 isn’t any better because your range has changed from (-127 to 128) to (-128 to 127) which, while not a large difference, means you no longer have access to the largest positive number that a signed 8 bit system could produce. I guess the only true answer is: it depends on whether or not you REALLY need access to -128 instead of 128. General short answer: no, it doesn’t help.**
2. What is the range of exponent values possible using 6 exponent bits? What is the value of k in this “Excess k” format using the IEEE excess mapping scheme?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # of Bits | **Number of Values** | Min Value | Max Value | **Value of K** |
| 6 | **64** | -32 | 31 | **32** |

1. See assignment for question; Problems done in excess 7

|  |  |  |
| --- | --- | --- |
| Sign | Mantissa | Exponent |
| 1 | 11100 | 1011 |

1.11100\*21 + 2 + 0 + 8 - 7 = 1.11100 \* 24 = 111102 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |

0 + 0 + 0 + 16 + 8 + 4 + 2 =

**-3010 = 1 1011 11100**

|  |  |  |
| --- | --- | --- |
| Sign | Mantissa | Exponent |
| 0 | 01011 | 0100 |

1.01011 \* 20 + 0 + 4 - 7 = 1.01011 \* 2-3 = 0.001010112 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0.5 | 0.25 | 0.125 | 0.0625 | 0.03125 | 0.015625 | 0.0078125 | 0.00390625 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

0 + 0 + 0.125 + 0 + 0.03125 + 0 + 0.0078125 +0.00390625

**0.1679687510 = 0 0100 01011**

|  |  |  |
| --- | --- | --- |
| Sign | Mantissa | Exponent |
| 1 | 11000 | 1111 |

1.11000 \* 21 + 2 + 4 + 8 - 7 = 1.11000 \* 28 = 1110000002 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

256 + 128 + 64 =

**-44810 = 1 111 11000**

|  |  |  |
| --- | --- | --- |
| Sign | Mantissa | Exponent |
| 0 | 01001 | 0110 |

1.01001 \* 20 + 2 + 4 + 0 – 7 = 1.01001 \* 2-1 = 0.1010012 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0.5 | 0.25 | 0.125 | 0.0625 | 0.03125 | 0.015625 | 0.0078125 | 0.00390625 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

0.5 + 0 + 0.125 + 0 + 0 + 0.015625 =

**0.64062510 = 0 0110 01001**

1. Convert decimal 85.0 to this format (excess 7, 1 significand, 5 mantissa, 4 exponent)

85.0 – 64 = 21.0 21.0 – 16 = 5.0 5.0 – 4 = 1.0 1.0 – 1 = 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

0101 01012 = 1.01010*1* \* 26+7 = 1.01010 \* 213 🡪 1310 = 11012

*Excess place value lost to stack overflow*

|  |  |  |
| --- | --- | --- |
| **Sign** | **Mantissa** | **Exponent** |
| **0** | **01010** | **1101** |

**0 1101 010102 = 84.010 (error due to stack overflow)**

1. Convert decimal 25.0 to this format (excess 7, 1 significand, 5 mantissa, 4 exponent)

25.0 – 16 = 9.0 – 8 = 1.0 1.0 – 1 = 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |

0001 10012 = 1.10010 \* 24 + 7 = 1.10010 \* 211 🡪 1110 = 10112

|  |  |  |
| --- | --- | --- |
| **Sign** | **Mantissa** | **Exponent** |
| **0** | **10010** | **1011** |

**0 1011 100102 = 25.010**

1. Convert decimal 2.25 to this format (excess 7, 1 significand, 5 mantissa, 4 exponent)

2.25 – 2 = 0.25 0.25 – 0.25 = 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 4 | 2 | 1 | 0.5 | 0.25 | 0.125 | 0.0625 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

0010.01002 = 1.00100 \* 21 + 7 = 1.00100 \* 28 🡪 810 = 10002

|  |  |  |
| --- | --- | --- |
| **Sign** | **Mantissa** | **Exponent** |
| **0** | **00100** | **1000** |

**0 1000 001002 = 2.2510**

1. Convert decimal 7.125 to this format (excess 7, 1 significand, 5 mantissa, 4 exponent)

7.125 – 4 = 3.125 –2 = 1.125 –1 = 0.125 0.125 – 0.125 = 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 4 | 2 | 1 | 0.5 | 0.25 | 0.125 | 0.0625 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |

0111.00102 = 1.11001 \* 22 + 7 = 1.11001 \* 29 🡪 910 = 10012

|  |  |  |
| --- | --- | --- |
| **Sign** | **Mantissa** | **Exponent** |
| **0** | **11001** | **1001** |

**0 1001 110012 = 7.12510**

1. See assignment for question (Problems done in excess 4)

|  |  |  |
| --- | --- | --- |
| Sign | Mantissa | Exponent |
| 1 | 1011 | 111 |

1.1011 \* 21 + 2 + 4 – 4 = 1.1011 \* 23 = 1101.12 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 4 | 2 | 1 | 0.5 | 0.25 | 0.125 | 0.0625 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |

8 + 4 + 0 + 1 + 0.5 =

**-13.510 = 1 1011 1112**

|  |  |  |
| --- | --- | --- |
| Sign | Mantissa | Exponent |
| 0 | 0100 | 010 |

1.0100 \* 20 + 2 + 0 – 4 = 1.0100 \* 2-2 = 0.01012 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 4 | 2 | 1 | 0.5 | 0.25 | 0.125 | 0.0625 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

**0.312510 = 0 0100 0102**

|  |  |  |
| --- | --- | --- |
| Sign | Mantissa | Exponent |
| 1 | 1111 | 110 |

1.1111 \* 20 + 2 + 4 – 4 = 1.1111 \* 22 = 111.112 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 4 | 2 | 1 | 0.5 | 0.25 | 0.125 | 0.0625 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |

0 + 4 + 2 + 1 + 0.5 + 0.25 =

-**7.7510 = 1 111 1102**

|  |  |  |
| --- | --- | --- |
| Sign | Mantissa | Exponent |
| 0 | 0110 | 010 |

1.0110 \* 20 + 2 + 0 – 4 = 1.0110 \* 2-2 = 0.0101102 =

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0.5 | 0.25 | 0.125 | 0.0625 | 0.03125 | 0.015625 | 0.0078125 | 0.00390625 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |

0 + 0.25 + 0 + 0.0625 + 0.03125 =

**0.3437510 = 0 0110 0102**

1. Convert decimal 2.25 to this format (excess 4, 1 significand, 4 mantissa, 3 exponent)

2.25 – 2 = 0.25 0.25 – 0.25 = 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 4 | 2 | 1 | 0.5 | 0.25 | 0.125 | 0.0625 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

0010.01002 = 1.0100 \* 21 + 4 = 1.0100 \* 25 🡪 510 = 1012

|  |  |  |
| --- | --- | --- |
| **Sign** | **Mantissa** | **Exponent** |
| **0** | **0100** | **101** |

**0 0100 1012 = 2.2510**

1. Convert decimal 0.625 to this format (excess 4, 1 significand, 4 mantissa, 3 exponent)

0.625 – 0.5 = 0.125 0.125 – 0.125 = 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 4 | 2 | 1 | 0.5 | 0.25 | 0.125 | 0.0625 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

0000.10102 = 0.1010 \* 20 + 4 = 1.0100 \* 23 🡪 310 = 0112

|  |  |  |
| --- | --- | --- |
| **Sign** | **Mantissa** | **Exponent** |
| **0** | **0100** | **011** |

**0 0100 0112 = 0.62510**

1. Convert decimal 0.06640625 to this format (excess 4, 1 significand, 4 mantissa, 3 exponent)

0.06640625 – 0.0625 = 0.00390625 0.00390625 – 0.00390625

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0.5 | 0.25 | 0.125 | 0.0625 | 0.03125 | 0.015625 | 0.0078125 | 0.00390625 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

0.000100012 = 1.0001 \* 2-3 + 4 = 1.0001 \* 21 🡪 110 = 0012

|  |  |  |
| --- | --- | --- |
| **Sign** | **Mantissa** | **Exponent** |
| **0** | **0001** | **001** |

**0 0001 0012 = 0.0664062510**

1. Convert decimal 5.125 to this format (excess 4, 1 significand, 4 mantissa, 3 exponent)

5.125 – 4 = 1.125 1.125 – 1 = 0.125 0.125 – 0.125 = 0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 4 | 2 | 1 | 0.5 | 0.25 | 0.125 | 0.0625 |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |

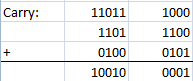
101.0012 = 1.0100*1* \* 22 + 4 = 1.01001 \* 26 🡪 610 = 1102

*Excess place value lost to stack overflow*

|  |  |  |
| --- | --- | --- |
| **Sign** | **Mantissa** | **Exponent** |
| **0** | **0100** | **110** |

**0 0100 1102 = 5.010 (error due to stack overflow)**

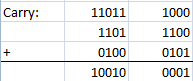
1. Add 1101 1100 + 0100 0101. Interpret the answer as a signed 2’s complement value and as an unsigned value. Indicate if the answer overflows for signed as well as unsigned interpretations.

 🡪 Answer assumes “2’s complement” implies 8 bit system; therefore discard as an extra bit

Signed (*Doesn’t Overflow*) = 33 Unsigned (*Overflows*) = 33

Correct answers: 33 289

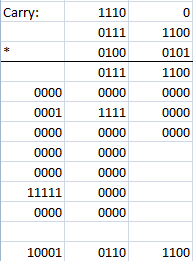
1. Add 1101 1100 + 0100 0101. Indicate if the answer overflows for signed as well as unsigned interpretations.

🡪 Answer assumes a system which can handle 9 bits; therefore no bits are discarded

Signed: (*Overflows*) = -223 Unsigned (*Doesn’t Overflow*) = 289

Correct answers: 33 289

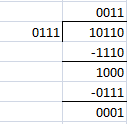
1. Multiply 0111 1100 \* 0100 0101. Indicate if the answer overflows

124 \* 69 = 8556

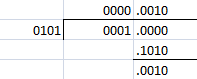
8192+0+0+0+0+256+0+64+32+0+8+4+0+0 = 8556

**Answer: 1 0001 0110 1100 with no overflow, because multiplication problems don’t overflow; only addition and subtraction do**

1. Divide 22 binary by 7 binary using integer binary long division.

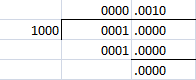
 It does not terminate, just keeps on going. It’s Pi.

1. Divide 1 binary by 5 binary using integer binary long division. Indicate if the division terminates.

 **The division does not terminate. Answer is: \_\_**

**0.001**

1. Divide 1 binary by 8 binary using integer binary long division. Indicate if the division terminates.

**The division terminates. Answer: 0.0010**

1. See assignment for question
2. See assignment for question

For 46 and 47, I’ll take the point loss in exchange for sleep and make it up with coding later. It’s not worth losing 2 hours of sleep just to properly format the problems on a computer.