#Multani Aman (040877727)

Lab08

Due date: Wednesday, October 26, 2016

Objectives: converting fractional number to binary, use of one’s, two’s complement and IEEE floating point in computer.

Note: please print this lab. I will not accept any hand writing.

**Fractional number**

1. Convert these fractions numbers to binary number (multiplication method)

Example: .75= .75X 2=1.50 Read top to bottom

.50X2=1.00

(up to 4 bit)

1- .1=.**0001**

0.1 x 2 = 0.2 = 0

0.2 x 2 = 0.4 = 0

0.4 x 2 = 0.8 = 0

0.8 x 2 = 1.6 = 1

2- .25= .**01**

0.25 x 2 = 0.5 = 0

0.5 x 2 = 1.0 = 1

3- .625= .**101**

0.625 x 2 = 1.250 = 1

0.250 = 0.500 = 0

0.500 x 2 = 1.0 = 1

4- .345= .**0101**

0.345 x 2 = 0.690 = 0

0.690 x 2 = 1.380 = 1

0.380 x 2 = .760 = 0

0.760 x 2 = 1.520 = 1

5- .8765= .**1110**

0.8765 x 2 = 1.7530 = 1

0.7530 x 2 = 1.5060 = 1

0.5060 x 2 = 1.0120 = 1

0.0120 x 2 = 0.024 = 0

1. Convert these fractions numbers to binary

1- **1=0001.01**

1= 0001

0.25 x 2 = 0.5 = 0

0.5 x 2 = 1 = 1

2- 15.25= **1111.01**

8 + 4 + 2 + 1 = 15

1000 + 0100 + 0010 + 0001 = 1111

0.25 x 2 = 0.5 = 0

0.5 x 2 = 1 = 1

3- 25.625= **1101. 101**

16 + 8 + 1 = 25

10000 + 1000 + 0001 = 11001

0.623 x 2 = 1.250 = 1

0.250 = 0.500 = 0

0.500 x 2 = 1.0 = 1

1. Convert these fractions binary to decimal numbers.
2. 101.11= **5.75**

1 + 4 = 5

001 +100 = 101

2 + 1 = 3

010 + 001 = 11

0.5 + 0.25 = 0.75

1. 1001.1111= **9. 9375**

8 + 1 = 9

1000 = 0001

8 + 4 + 2 + 1 = 15

1000 + 0100 + 0010 + 0001 = 1111

0.5 + 0.25+ 0.125 + .0625

= 0.9375

C- .1101**= 0.8125**

0.5 +0.25 + 0.0625

= 0.8125

**Addition and Subtraction**

1-Add these two binary numbers

11111+

00110

**100101**

2- Add these two binary numbers

10011 +

011101

**110000**

3- Subtract these binary numbers

101-

100

**001**

4- Subtract these binary numbers

11000-

01101

**01011**

5- Subtract these binary numbers

011100000-

010001011

**001010101**

+010001011

101110100

101110101

+011100000

101110101

1001010101

**Sign Magnitude , One’s and Two’s Complements**

1. **Sign Magnitude**

**Example: What is the sign magnitude of 12 in 8 bit**

12=00001100 sign Magnitude: 10001100

1-What is the Sign Magnitude of 35 in 8 bit?

**35 = 00100011**

**Sign magnitude: 10100011**

0 0 1 0 0 0 1 1

32 16 8 4 2 1

32 + 2 + 1 = 35

2-What is the sign magnitude of 47 in 8 bits?

**47** = **00101111**

**Sign magnitude: 10101111**

0 0 1 0 1 1 1 1

32 8 4 2 1

32 + 8 + 4 + 2 + 1 = 47

3-What is the sign magnitude of 110 in 8 bits?

**110**=**01101110**

**Sign magnitude:11101110**

0 1 1 0 1 1 1 0

64 32 8 4 2

64 + 32 + 8 + 4 + 2= 110

1. **One’s complement**

Example: What is the one’s complement of -45 in 8 bits binary?

45=00101101 one’s complement: 11010010

1-What is the one’s complement of 7 in 8 bits binary?

**7 = 00000111**

0 0 0 0 0 1 1 1

4+2+1 = 7

2-What is the one’s complement of 70 in 8 bits binary**?**

**70= 01000110**

0 1 0 0 0 1 1 0

64 + 4+ 2

3-What is the one’s complement of -49 in 8 bit binary?

-49 = -00110001 one complements: 11001110

0 0 1 1 0 0 0 1

32 +16 + 1 32+16+1 = 49

4-What is the one’s complement of -56 in 8 bit binary?

**-56= -00111000 one complement: 11000111**

0 0 1 1 1 0 0 0

32+16+8

1. **Two’s complement**

Example: what is the two’s complement of -45 in 8 bit binary?

45= 00101101

11010010 one’s complement

1+

11010011 two’s complement

What is the two’s complement of 23 in8 bit binary?

**23 = 00010111**

0 0 0 1 0 1 1 1

16 + 4+2 +1

What is the two’s complement of -23 in 8 bit binary?

**-23 = -00010111 one complement: 11101000**

**two complement: 11110101**

0 0 0 1 0 1 1 1

16 + 4+2 +1

What is the two’s complement of -35 in 8 bit binary?

-35= -**00100011 one complement:11011100**

**two complement:11011101**

0 0 1 0 0 0 1 1

32 + 2+ 1

**1 1 0 1 1 1 0 0**

What is the two’s complement of -45 in 8 bit binary?

-45=-**00101101**  **one complement:11010010**

**two complement:11010011**

0 0 1 0 1 1 0 1

32 + 8+4+ 1

**1 1 0 1 0 0 1 0**

**+1**

1. **One’s complement subtraction**:
2. Convert the negative number (-3) to one’s complement
3. Add the positive number to the one’s complement number.
4. If you have extra bit (carry over bit) adds to the result.

Example: 7-3=4 7= 111 3=011

111 (7)

-011 (3)

100 one’s complement of 3

111 (7)

Extra bit 1 0 11

Add 1

0 100

1-Using the one’s complement method subtracts these two binary numbers:

11001 – 01100 = **01101**

+01100

10011

+1

10100

+11001

10100

101101

**2-** Using the one’s complement method, subtract these two binary numbers:

1100–11011= **0001**

+11011

00100

+1

00101

+1100

00101

10001

1. Two’s complement subtraction

Using the two’s complement method, subtrac**t** these two binary numbers 1000-010

Example: 8-2=6 8=1000 2=0010

1. Convert the two numbers to binary
2. Find the negative number of (-2) in two’s complement

0010 (2)

1101 One’s complement of 2

add 1

1110 Two’s complement of 2

1. Add the two numbers together

1000 (8)

add 1110 + two’s complement

0110

Important note: if you have an extra bit in two’s complement method, discard it.

**1-** Using the two’s complement method, subtrac**t** these two binary numbers:

10110-01011 = **01011**

+01011

10100

+1

10101

+10110

10101

101011

2-Using the two’s complement method, subtrac**t** these two binary numbers

10001- 01001= **01000**

+01001

10110

+1

10111

+10001

10111

101000

3-Using the two’s complement method, subtrac**t** these two binary numbers

-0110-0101= **0101**

+0101

1010

+1

1011

+0110

1001

+1

1010

+1010

1011

0101

4-Using the two’s complement method, subtrac**t** these two binary numbers

-0110 +0101= **1111**

+0110

1001

+1

1010

+0101

1010

1111

**Decimal to Floating Point**

**The Conversion Procedure**

The rules for converting a decimal number into floating point are as follows:

1. Convert the integer part to binary (division/subtraction method)
2. Normalize the number. Move the binary point so that it is 1bit from the left. Adjust the exponent.
3. Place the mantissa into the mantissa field of the number. Omit the leading 1, and fill with zeros on the right up to 23 bits.
4. Add the bias (magic number) to the exponent of n , and place it in the exponent field. Set the sign bit, 1 for negative, 0 for positive, according to the sign of the original number.
5. Put all binary number together and convert them to **Hexadecimal.**

**Using The Conversion Procedure**

* Convert 2.625 to IEEE 32 bit floating point format.
  1. The integral part is easy, 210 = 102. For the fractional part:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0.625 | × 2 = | 1.25 | 1 | *Generate 1 and continue with the rest.* |
| 0.25 | × 2 = | 0.5 | 0 | *Generate 0 and continue.* |
| 0.5 | × 2 = | 1.0 | 1 | *Generate 1 and nothing remains.* |

* 1. So 0.62510 = 0.1012, and 2.62510 = 10.1012.
  2. Add an exponent part: 10.1012 = 10.1012 × 20.
  3. Normalize: 10.1012 × 20 = 1.01012 × 21.
  4. Mantissa: 0101
  5. Add the magic number 1ca27 to the exponents number 1and convert the result to binary 127+1=128 128=10000000 Exponent in binary
  6. Now put all binary numbers together including, sign bit, exponent, mantissa up to 32 bits
  7. 0 100 0000 0 010 1 000 0000 0000 0000 0000

I . 4 0 2 8 0 0 0 0 (hexadecimal)

* 1. Convert all the binary numbers to Hexadecimal 40280000

1. Convert -4.75 to IEEE 32 bit floating point format.

**- 4 =- 100** 0.75 x 2= 1.5 :> 1 **0.75 = 0.11**  
 0.5 x 2 = 1 :> 1

**Normalization = -4.75 = -100.11 = -1.0011 x 10^2**

**Mantissa:0011**

**127 + 2 =129 = 0 1 0 0 0 0 0 0 1**

**HexaDecimal = 1 100 0000 1 001 1 000 0000 0000 0000 0000 = C 0980000**

2- Convert 15.8125 to IEEE 32 bit floating point format.

**15 = 1111 0.8125x2=1.625 :> 1 0.8125=0.1101**

**.625x2=1.25 :> 1**

**0.25 x2= 0.5 :> 0**

**0.5x2=1 :> 1**

**Normalization =15.8125= 1111.1101 =1.1111101 x 10^3**

**Mantissa =1101**

**127+3 =130 =0 1 0 0 0 0 0 1 0**

**HexaDecimal = 0 100 0001 0 111 1101 0000 0000 0000 0000 = 417D0000**

3. Convert 0.40625 to IEEE 32 bit floating point format.

**0.40625x2=0.8 =0.8125 :>0**

**0.8125x2=1.625:>1**

**0.625x2=1.25 :> 1**

**0.25x2=0.5 :> 0**

**0.5x2=1 :> 1**

**Normalization = 0.40625 =0.01101=1.101 x 10^-2**

**Mantissa = 101**

**127-2=125=0 1 1 1 1 1 0 1**

**HexaDecimal = 0 111 1101 1010 0000 0000 0000 0000 0000 = 7DA00000**

4. Convert -1313.3125 to IEEE 32-bit floating point format.

**-1313=-11011101 0.3125x2= 0.625 :> 0**

**0.625x2=1.25 :> 1**

**0.25x2 =0.5 :> 0**

**0.5x2=1 :> 1**

**0.3125 = 0.0101**

**Normalization = -1313.3125 = -11011101.0101 = -1.10111010101 x 10^7**

**Mantissa= 10111010101**

**127+7=134 = 10000110**

**HexaDecimal = 1 100 0011 0101 1101 0101 0000 0000 0000 = C35D5000**

5. Convert 1.75 to 8 bit floating point format.

**1=0001 0.75x2=1.5 :>1 0.75=0.11**

**0.5x2=1 :> 1**

**Normalization = 1.75=1.11 x 10^0**

**Mantissa = 11**

**3+0=3=11**

**HexaDecimal = 0 111 1000 0000 0000 0000 0000 0000 0000= 78000000**

Note: for this question the magic number is 3. So add the exponent number to 3.

Sign bit=one bit Exponent=3 bit mantissa= 4

One for sign bit 3 bits for exponent and the rest for mantissa. Example 0 101 1010