**Floating Point Storage**

**Storage Format for Real Data**

The SPIM assembler, for the SPIM simulator, encodes real numbers in floating point format.

A real number, **N**, is expressed in binary normalized form as follows:

**N** = (+/-) **1.xxxxxxxxxxxxxxxxxxxxxxx \* 2Power**

The format for storage is given in the figure.

**31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0**

sign

mantissa

biased exponent

First, the sign of **N** is stored in the sign bit (bit 31) as a 1 if N is negative, and 0 otherwise.

Second, the mantissa as given as the series of **x**’s in the representation above is stored in 23 bits (bits 0 through 22). The leading 1 **bit** is simply not stored!

Third, the exponent, given as **Power** in the expression above, is stored in 8 bits (bits 23 through 30).

The actual storage is the calculated sum of the bias constant **0111 1111** and the 8 bit value for the exponent.

See the table for some examples.

|  |  |
| --- | --- |
| exponent | biased exponent |
| -2 | 0111 1101 |
| -1 | 0111 1110 |
| 0 | 0111 1111 |
| 1 | 1000 0000 |
| 2 | 1000 0001 |

0. Include the assembler directive **.float 4.125** in an SPIM program, then load and execute.

1. Give the floating-point storage for the decimal number, **4.125** .

a) Express the number in **binary**.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

b) Give the number in the **normalized form**.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

c) Give the single bit to encode the **sign**.

**\_\_\_**

d) Give the bit string for the **biased exponent**.

**\_ \_ \_ \_ \_ \_ \_ \_**

e) Give the **bit string** encoding for the stored real number.

**\_** || **\_ \_ \_ \_ \_ \_ \_ \_** || **\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_**

f) Give the **hexadecimal** description for the stored real number.

**\_\_ \_\_ \_\_ \_\_ \_\_ \_\_ \_\_ \_\_**

2. What **decimal real number** is stored by the bit string given in hexadecimal as **BFB0 0000** ?

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Consider floating-point storage for real numbers in a 6 bit unit (different from SPIM), where

bit 5is the sign bit (**S**),

bits 2 through 4 stores a biased exponent (3 bits) (**BBB**), and

bits 0 through 1 stores the mantissa (2 bits) (**MM**).

|  |
| --- |
| **5 432 10** |
| **S BBB MM** |

Suppose also that real numbers are stored, similar to that for the SPIM simulator. First convert the number to binary and normalize. Then store the sign, mantissa, and the actual exponent with a bias of 011.

For example, the storage for

**-1.0 = - (b) 1.00 \* 20**

would be

**1 011 00**

and the storage for

**1.0 = (b) 1.00 \* 20**

would be

**0 011 00**.

1. Complete the table below for the column of Decimal values?

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S** | **BBB** | **MM** | **Binary Normal Form** | **Rational** | **Decimal** |
| **0** | **000** | **00** | **(b)1.00 \* 2-3** | **1/8** | **0.125** |
| **0** | **000** | **01** | **(b)1.01 \* 2-3** | **5/32** |  |
| **0** | **000** | **10** | **(b)1.10 \* 2-3** | **3/16** |  |
| **0** | **000** | **11** | **(b)1.11 \* 2-3** | **7/32** |  |
| **0** | **001** | **00** | **(b)1.00 \* 2-2** | **1/4** |  |
| **0** | **001** | **01** | **(b)1.01 \* 2-2** | **5/16** |  |
| **0** | **001** | **10** | **(b)1.10 \* 2-2** | **3/8** |  |
| **0** | **001** | **11** | **(b)1.11 \* 2-2** | **7/16** |  |
| **0** | **010** | **00** | **(b)1.00 \* 2-1** | **1/2** |  |
| **0** | **010** | **01** | **(b)1.01 \* 2-1** | **5/8** |  |
| **0** | **010** | **10** | **(b)1.10 \* 2-1** | **3/4** |  |
| **0** | **010** | **11** | **(b)1.11 \* 2-1** | **7/8** |  |
| **0** | **011** | **00** | **(b)1.00 \* 2 0** | **1** |  |
| **0** | **011** | **01** | **(b)1.01 \* 2 0** | **5/4** |  |
| **0** | **011** | **10** | **(b)1.10 \* 2 0** | **3/2** |  |
| **0** | **011** | **11** | **(b)1.11 \* 2 0** | **7/4** |  |

4) Complete the table below for the column of Rational values and the column of Decimal values?

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S** | **BBB** | **MM** | **Binary Normal Form** | **Rational** | **Decimal** |
| **0** | **100** | **00** | **(b)1.00 \* 21** | **2** | **2** |
| **0** | **100** | **01** | **(b)1.01 \* 21** | **5/2** | **2.5** |
| **0** | **100** | **10** | **(b)1.10 \* 21** | **3** |  |
| **0** | **100** | **11** | **(b)1.11 \* 21** | **7/2** |  |
| **0** | **101** | **00** | **(b)1.00 \* 22** | **4** |  |
| **0** | **101** | **01** | **(b)1.01 \* 22** | **5** |  |
| **0** | **101** | **10** | **(b)1.10 \* 22** | **6** |  |
| **0** | **101** | **11** | **(b)1.11 \* 22** | **7** |  |
| **0** | **110** | **00** | **(b)1.00 \* 23** |  |  |
| **0** | **110** | **01** | **(b)1.01 \* 23** |  |  |
| **0** | **110** | **10** | **(b)1.10 \* 23** |  |  |
| **0** | **110** | **11** | **(b)1.11 \* 23** |  |  |
| **0** | **111** | **00** | **(b)1.00 \* 24** |  |  |
| **0** | **111** | **01** | **(b)1.01 \* 24** |  |  |
| **0** | **111** | **10** | **(b)1.10 \* 24** |  |  |
| **0** | **111** | **11** | **(b)1.11 \* 24** |  |  |

5) How many of these real number encodings are strictly between 0.0 and 1.0? **????**

6) How many of these real number encodings are strictly greater than 1.0? **????**