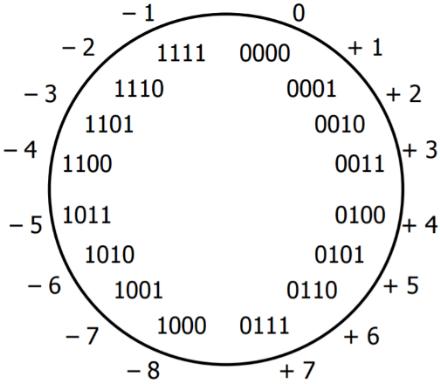
**Integers and Floating Point**

**Signed Integers with Two’s Complement**

Two’s complement is the standard for representing signed integers:

* The most significant bit (MSB) has a negative value; all others have positive values (same as unsigned)
* Binary addition is performed the same way for signed and unsigned
* The bit representation for the negative (additive inverse) of a two’s complement number can be found by:

flipping all the bits and adding 1 (i.e. − = ~ + ).

The “number wheel” showing the relationship between 4 -bit numerals and their Two’s Complement interpretations is shown on the right:

* The largest number is 7 whereas the smallest number is -8
* There is a nice symmetry between numbers and their negative counterparts except for -8

**Exercises:** (assume 8-bit integers)

1)What is the **largest integer**? The **largest integer + 1**?

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | Unsigned: | | | |  | Two’s Complement: | | | | | |
|  |  | | 1111 1111 -> 0000 0000 | | | | | 0111 1111 -> 1000 | | | | 0000 | |
|  | | 2)How do you represent (if possible) the following numbers:39, -39, 127? | | | | | | | | | | | |
|  | |  | | Unsigned: | | |  |  | Two’s Complement: | | | | |
|  | |  | |  |  |  |  |  |  |  |  | |  |
|  | |  | | 39: | 0010 | | 0111 | 39: | | 0010 | 0111 | | |
|  | | -39: | | | Impossible | | | -39: 1101 | | | 1001 | | |
|  | | 127: | | | 0111 | | 1111 | 127: | | 0111 | 1111 | | |



3) Compute the following sums in binary using your Two’s Complement answers from above. *Answer in hex*.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **a.** 39 | -> 0b **0 0 1 0 0 1 1 1** | **b**. 127 ->0b **0 1 1 1 1 1 1 1** |
|  | +(-39) -> 0b **1 1 0 1 1 0 0 1** | | + (-39)-> 0b **1 1 0 1 1 0 0 1** |
|  | 0x **0 0** <- 0b **0 0 0 0 0 0 0 0** | | 0x **5 8** <- 0b **0 1 0 1 1 0 0 0** |
|  | **c**. 39 | -> 0b **0 0 1 0 0 1 1 1** | **d**. 127 ->0b **0 1 1 1 1 1 1 1** |
|  | +(-127)-> 0b **1 0 0 0 0 0 0 1** | | + 39 -> 0b **0 0 1 0 0 1 1 1** |
|  | 0x **A 8** <- 0b **1 0 1 0 1 0 0 0** | | 0x **A 6** <- 0b **1 0 1 0 0 1 1 0** |
| 4) Interpret each of your answers above and indicate whether or not overflow has occurred. | | |
|  | **a.** 39 + (-39) | | **b.** 127 + (-39) |
|  | Unsigned: 0 overflow | | Unsigned: 88 overflow |
|  | Two’s Complement: 0 no overflow | | Two’s Complement: 88 no overflow |
|  | **c.** 39 + (-127) | | **d.** 127 + 39 |
|  | Unsigned: 168 overflow | | Unsigned: 166 no overflow |
|  | Two’s Complement: -88 no overflow | | Two’s Complement: -90 overflow |

1

**Goals of Floating Point**

Representation should include: [1] a large range of values (both very small and very large numbers), [2] a high amount of precision, and [3] real arithmetic results (*e.g.* ∞ and NaN) .

**IEEE 754 Floating Point Standard**

The value of a real number can be represented in scientific binary notation as:

**Value = (-1)sign** × **Mantissa2** × **2Exponent = (-1)S** × **1.M2** × **2E-bias**

The binary representation for floating point values uses three fields:

* **S**: encodes the*sign*of the number (0 for positive, 1 for negative)
* **E**: encodes the*exponent*in **biased notation** with a bias of 2w-1-1
* **M**: encodes the*mantissa*(or*significand*, or*fraction*) – stores the fractional portion, but does not includethe implicit leading 1.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| float |  | **S** |  | **E** |  | **M** |  |
|  |  |  |  |
| double |  | 1 bit |  | 8 bits |  | 23 bits | |
|  |  | 1 bit |  | 11 bits |  | 52 bits | |



How a float is interpreted depends on the values in the exponent and mantissa fields:

|  |  |  |
| --- | --- | --- |
| **E** | **M** | **Meaning** |
| 0 | anything | denormalized number (denorm) |
| 1-254 | anything | ∞ |
| normalized number |
| 255 | zero | infinity ( ) |
| 255 | nonzero | not-a-number (NaN) |



**Exercises:**

5) What is the largest, finite, positive value that can be stored using a float?

S = 0, E = 254, M = 0b1…1. Value is then 1.1 … 1×2127 = (2 – 2-23 )×2127 = 2128 – 2104.

6) What is the smallest, positive, normalized value that can be stored using float?

**S = 0, E = 1, M = 0b0…0. Value is then 1.0×2-126 = 2-126.**

7) Convert the decimal number 1.25 into single precision floating point representation:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |