**VECTOR NEGATE FUNCTION**

1. This function would provide a routine for negating a floating-point vector. The interpretation of a floating-point number has been derived from IEEE 754 standard, which specifies that
   1. Binary floating-point format (r=2)
   2. Single, double, extended and double extended precision.
   3. Representation for indefinite values (NaN) and infinity (INF)
   4. Signed zero and denormalized numbers
   5. Masked exceptions.
   6. Roundoff control.
2. The number representation is as depicted below. The most significant bit (MSB) is towards the left

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | - Sign |  | - Exponent |  | - Mantissa |

1. **The Sign**. The sign of the floating-point number is represented using a single bit. A 1 bit indicates a negative number and a 0 indicates a positive number.
2. **The Mantissa**. Taking the example of -6.521 x 105; the sign is negative; the mantissa is 6.5 and the exponent is 5. The fractional part of the mantissa is obtained by adding each digit in the mantissa multiplied by a power of 10, that is

0.521 = 5x10-1 + 2x10-2 + 1x10-3

A binary floating-point number follows a similar convention. Considering the example, +10.011 x 23 Here, the sign is positive, the mantissa is 10.011 and the exponent is 3. The fractional part can be represented as

0.011 = 0x2-1 +1x2-2 +1x2-3

1. **The Exponent**. The short real exponents are stored as 8-bit unsigned integers with a bias of 127, for example for the number 1.44 x 105 the exponent 5 is added to 127 and the sum 132 is stored in binary. Similarly, the exponent -7 will be saved as 127+(-7)=120 in binary.
2. **Summing Up – Floating Point Numbers with Base 2**. The floating-point number representation in binary can be generalized as

(-1)S x M x 2E

here S is sign, M is mantissa or significand and E is exponent.

1. **Vector Negate Function**. The vector negate function implementation takes a vector of floating-point numbers as input and produces a vector of same size as output with flipped sign bits of each member in the original vector. The function prototype is as listed below

void vector\_negate(const float\* src, float\* dst, int size)

where, src is the source vector, dst is the destination vector and size is the number of elements in the source vector.

1. **Methodology**.
   1. Consider the vector input of floating-point numbers in row/column major format.
   2. Iterate over each FP number keeping the size in view.
   3. Flip the sign bit of each FP number.