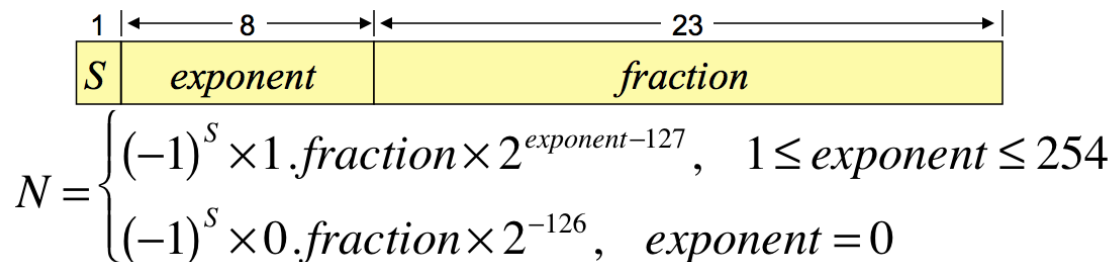


**Question 4 (30 points):** In decimal notation we can define the *decimal-point position* of an integer as the position of the most-significant digit of the integer (counting the position of the least-significant digit as position zero). This definition is convenient because the decimal-point position is equal the value of the exponent when the number is written into normalized scientific notation. Below are a few examples:

Decimal Integer Value	decimal-point position	normalized scientific notation
7	0	$7.0 \times 10^0$
31	1	$3.1 \times 10^1$
15624	4	$1.5634 \times 10^4$

Similarly, we can define the *binary-point position* for a positive integer value represented in two-complement notation as the position of the most-significant non-zero bit.

A *casting* operation is an operation that transforms the data type of a value. Assume that a floating-point number is represented in the IEEE 754 format shown below:



With this format the casting of a two-complement integer representation to a floating-point representation requires the computation of the sign, exponent and fraction and the assembling of the number into the binary floating point representation. Depending on the value of the integer this casting may incur in some loss of precision.

When writing the programs below you must follow all the MIPS function-calling and parameter-passing conventions. Also in your code you are not allowed to use pseudo instructions that manipulate constants that are larger than 16 bits.

- (10 points)** Write MIPS assembly code for a function called `BinPointPos` that receives in `$a0` the value of a non-zero positive integer represented in two-complement notation and returns in `$v0` the *binary-point position* of the number.
- (20 points)** Write a MIPS assembly code for a function called `IntToFloatCast` that receives in `$a0` a memory address. The 4-byte word at that address contains an integer value represented in two-complement notation. When `IntToFloatCast` returns, the same memory position will contain a 32-bit binary representation of the floating-point representation of the same value with as much precision as is possible for the casting to maintain. `IntToFloatCast` **must call** `BinPointPos` to compute the position of the binary point.