**Question 1**. n Floating point representation we have three components , 1.The Sign Bit ;2.Exponent ;3.Fractional Part. Precession is one the prime attribute of any Floating point Representation,

1. Does any of the above three components play a role in the defining the Precession of the number ?

Answer: Yes esp Fraction part

1. If so which are the component or Components which play the  role in defining precession  and how ?

Answer: To represent right bit we need 3.33 bits per decimal ( Log 210 =3.32) . Based on experimental data it is found that at least 7 Digits are needed to reproduce right results, which means, needed bits are at least 23 bits. Therefore if 1 bit used for Sign bit & 8 for Exponent, out of 32 bit 23bits still can be used for faithful Fraction part reproduction. But if 64 bits format is been used fraction precision depth goes to 52 bits deep can represent up to 15 bits data.

1. Explain this with example in your own words

Answer:

Example :

Decimal number 201.7

**Step 1** :

1. DEC 201 = HEX C9 = 1100 1001
2. Decimal Part =10110011001100110….
3. Total : 1100 1001 (DOT) 10110 0110 0110 0110..

**Step 2 :** Scientific rep : 1.100 100110110011001100110.. X 2 7

**Step 3**

IEEE Rep

|  |  |  |
| --- | --- | --- |
| Sign BIT ( 1 bit) | Exponential ( 8 bits) | mantissa |
| 0 | =127+7 =134 | 23 bits can rep 7 digits |
| 0 | 1000 0110 | 1001 0011 0110 0110 0110 011 |

Final number is 01000 01101001 0011 0110 0110 0110 011

If I use 64 bit representation, Exponential bits will be used 11 instead of 8 & The Mantissa can go up to 52 bits representing up 15 digit depth instead of current 7 bits. Thus 64 bits reparse also called double precision.

**Question 2**. What is Normal and Subnormal  Values as per IEEE754  standards  explain this  with the  help of number line

Answer:

To increase the precision of the significand, the IEEE 754 Standard uses a normalized significand which implies that its most significant bit is always 1. As this is implied, it is assumed to be on the left of the (virtual) decimal point of the significand. Thus in the IEEE Standard, the significand is 24 bits long – 23 bits of the significand which is stored in the memory and an implied 1 as the most significant 24th bit. The extra bit increases the number of significant digits in the significand. Thus, a floating point number in the IEEE Standard is

(– 1) s × (1. f )2 × 2exponent – 127  
where s is the sign bit; s = 0 is used for positive numbers and s = 1 for representing negative numbers. f represents the bits in the significand. Observe that the implied 1 as the most significant bit of the significand is explicitly shown for clarity.For a 32-bit word machine, the allocation of bits for a floating point number are given in Figure 2. Observe that the exponent is  
placed before the significand

Subnormal Numbers: When all the exponent bits are 0 and the leading hidden bit of the siginificand is 0, then the floating point number is called a subnormal number. Thus, one logical representation of a subnormal number is

(–1)s × 0.f × 2–127 (all 0s for the exponent) , where f has at least one 1 (otherwise the number will be taken as 0).

However, the standard uses –126, i.e., bias +1 for the exponent rather than –127 which is the bias for some not so obvious reason, possibly because by using –126 instead of –127, the gap between the largest subnormal number and the smallest normalized number is smaller

**Question 3** IEEE 754vv defines standards for rounding floating points numbers to a represent able value. There are five methods defines by IEEE for this – Take time and  understand what these five methods and explain it in your words using diagrams, illustrations of your own.

Refer to the slides presented in the class, go the last slide which says Implementation of Exponential Series in Cortex –M4  and do this on KEIL Simulator

Answer:

When a mathematical operation is performed with two floating point numbers, the destination register may have been representing say 32 bits format, the significand of the result may exceed 23 bits.. In such a case, there are two alternatives. One is to truncate the result, i.e., ignore all bits beyond the 32nd bit. The other is to round the result to the nearest significand.

**Rounding upwards :** if the first bit which is truncated is 1, add 1 to the least significant bit,. This is called Rounding upwards. Ex: 4.78 is rounded off to 4.8

**Rounding downwards** If the extra bits are ignored, it is called rounding downwards.

Ex: 4.78 is rounded off to 4.7

The IEEE Standard suggests to hardware designers to get the best possible result while performing arithmetic that are reproducible across different manufacturer’s computers using the standard. The Standard suggests that in a given equation,

Say c = a <op> b,  
where a and b are operands and <op> an arithmetic operation, the result c should be as if it was computed exactly and then rounded. This is called correct rounding

The IEEE 754 floating point standard for 64-bit (called double precision) numbers is very similar to the 32-bit standard but since number of bits are more, gets freedom to have more precision