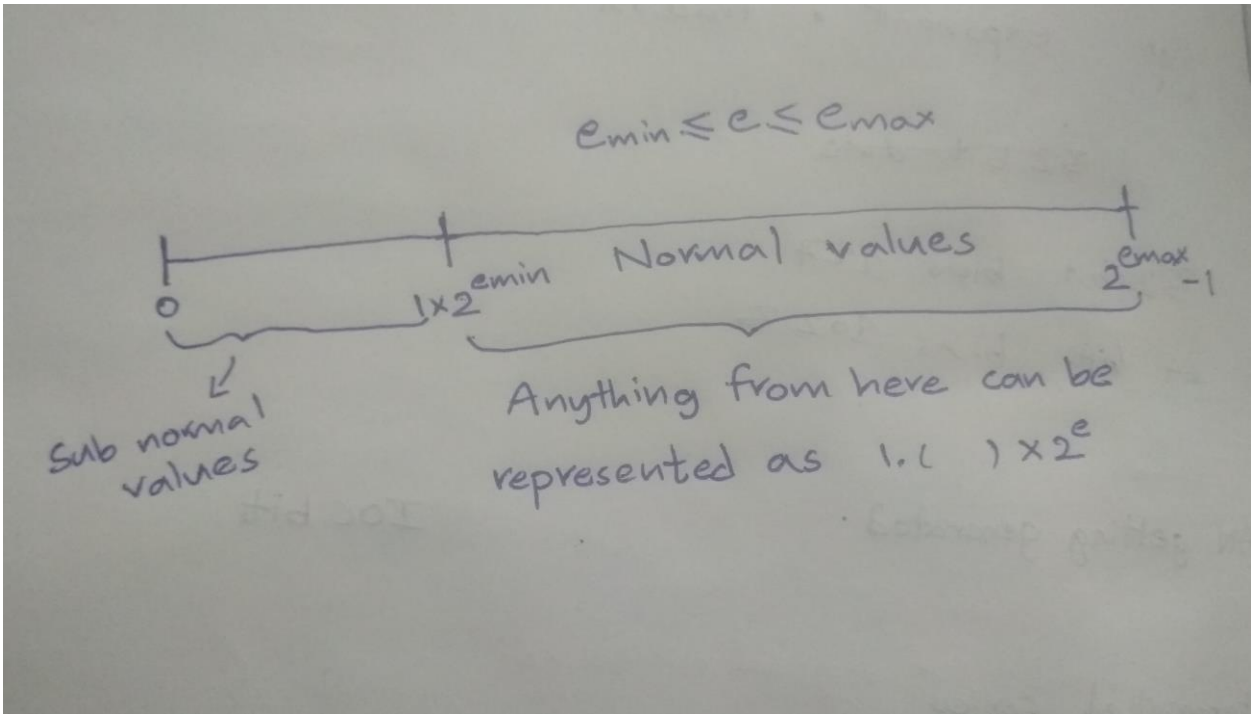
Advanced ARM architectures

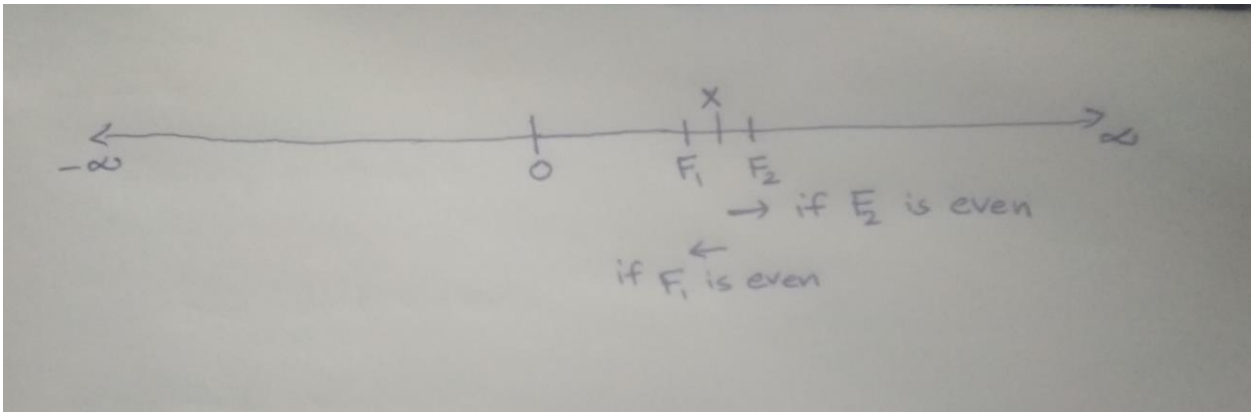
Assignment-3

IMT2015525 – Y V S Gnaneswara Reddy

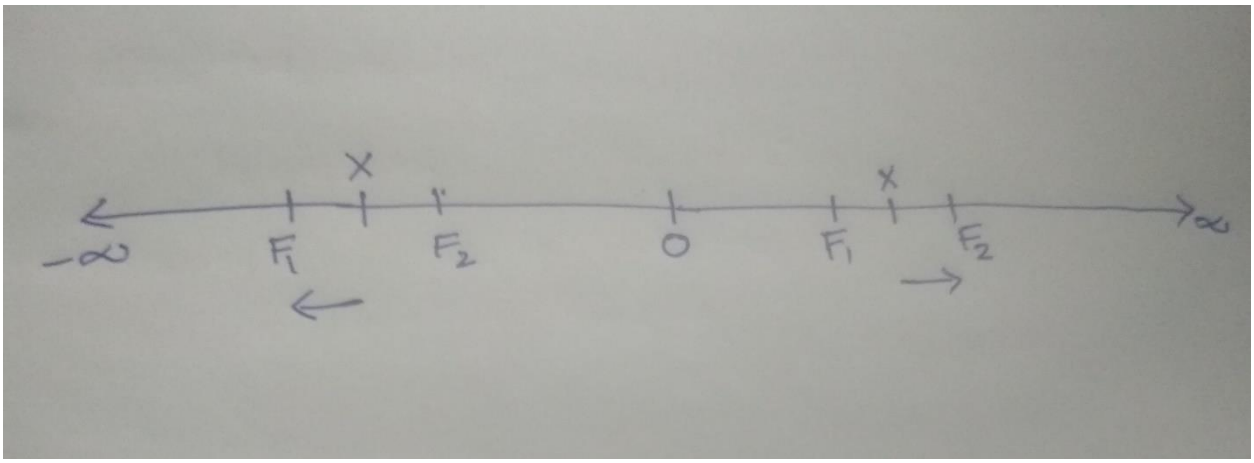
1. The smallest change that can be represented is called precision. The three components which we have in floating point representation are sign bit, exponential and fractional part. The sign bit doesn't play any role in defining the precision of the number. Fractional part defines the precision and precision is directly related to number of bits the fraction part takes to represent. The more the number of bits in fraction part, the smallest change can be represented. As the number of bits in fraction part increases, 1/(2 power the bit position from left) value will become smaller for the right end bits. Hence smallest change can be represented. But if we increase the bits for fractional part, we need more bits for exponent part to express the power. Therefore, precision is indirectly related to number of bits used to represent exponent.
   As we go from 32 bit to 64 bit, we increase the bits for fraction part and exponent to increase precision and range.
   For example, if we consider from half-precision (16-bit) to single precision (32-bit), the precision value increase from 4 bits in decimal to 7 bits.
2. Normally any number can be represented as $M \times 2^{e}$.
   For floating point representation, we make sure that M is always 1.something.
   The least number that can be represented like this is $X= 1 \times 2^{emin}$. Since we have always 1.something. We ignore one and store something as fraction part. These are called normal numbers.
   There will be values which are in between X and 0 (i.e. less than X and greater than Zero). They will be like 0.01×2^{emin} and there won't be 1 in M part. These are called subnormal numbers. These are represented with the exponent part as complete zeroes.



3. Not all decimal points can be represented as binary floating points. Therefore, for such numbers rounding is done to nearest precise floating point that can be represented.
   In case of the number is nearest to two floating points, then rounding is done using one of the following techniques:
   roundTiesToEven: it is rounded to nearest floating point. If there are two floating points nearer, then it is rounded to the point with least significant bit even.



   roundTiesToAway: it is rounded to nearest floating point. If there are two floating points nearer, then it is rounded to the point with highest magnitude i.e which is more away from zero.



roundTowardsPositive: it is rounded to floating point which is nearest and greater than its value.

roundTowardsNegative: it is rounded to floating point which is nearest and less than its value.

roundTowardsZero: it is rounded to floating point which is nearest and closer to zero i.e. in lesser magnitude.