2

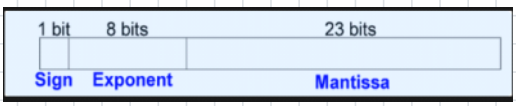
Monday, May 6, 2024 5:04 PM

- **Logical Right shift (A>>N)** - move A to right by N position, extension with 0
  - 10101 >> 3 = 00010
- **Logical left shift {A<<N)** - move A to left by N position, extension with 0
  - 00101 << 3 = 01000
    Is a multiplier A<<N = A * 2**N
- **arithmetic shift (A>>>N)** - move A to right by N position, extension with sign
  - **10101 >>> 3 = 11110**
  - is a divider A>>>N = A / 2**N
- **Fractional Binary Numbers**
  - Unsigned - N bits can represent [2**N] values, so to represent a value of a we need [log2A + 1] bits
  - Two's complement - N bits can represent [2**N-1] values, so to represent value A we need [log2A + 2] (1 bit less if A=-2N-1] bits
  - **Fixed Point Fractions**
    - base 10 → binary
      - if it is a signed number first convert it into sign/magnitude then to two's complement and use the following following same process
        # Eg - 7.510 to binary (4 integer bits and 4 fraction bits)

        1.Separate integer from fractional part
        # 7 and 510
        2. Use known algorithms for integer part
        # 0111
        3. Multiply fractional part p=2p
        # 2 * 510 = 1.20
        4. If p<1 the next digit is 0
        Otherwise, the next digit is 1 and the fractional part p=p-1
        # the fraction part .1 -> .10 -> .100 -> .1001 -> .1001
        If p ≠ 0, go back to point 3, otherwise we are done.

        Finally write the result
        # 01111001

- **Floating Point Fractions**
  - **IEEE 754 Floating-point standard**
    - Uses 32 bits to represent a number
    - Base 10 → ISEE 754 floating point
      # Eg - convert 228 to a binary floating point
      1. Convert it to base 2(unsigned putting the sign in the sign bit position in the 32nd bit)
      # 228 = 11100100
      2. Write the number in "binary scientific notation":
      # 111001002 = 1.110012 × 2**7
      3. Fill in each field of the 32-bit floating point number(That IEEE 754 diagram)
      - In the mantisa part store just the fraction part. b/c the first 1 is known implicitly
      - Change the exponent to biased exponent. biased exponent = 127 + exponent
      5. write the 32 bit together



- largest biased exponent - 254
- smallest biased exponent - 1