Unsigned integer representation
- With n bits, max value that can be represented: \(2^n - 1\)

Binary to Decimal conversion
\[
\begin{array}{ccccccc}
\phantom{1} & \phantom{1} & \phantom{1} & \phantom{1} & 1 & 0 & 1 \\
2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
\hline
32 & 16 & 8 & 4 & 2 & 1 & 0 \\
\end{array}
\]
\[= 52\]

6 bits, so max number possible is \(2^6 - 1 = 63\)

Decimal to Binary (unsigned)
1. Find number of bits required \(\lfloor \log_2(n) \rfloor + 1\)
2. For each bit-position, starting from highest, Repeatedly check if number greater or equal to \(2^n\) position, and set bit to 0 or 1 accordingly

Number of bits required (unsigned)
- \(52: \log_2(52) = 5.7; \lfloor \log_2(5.7) \rfloor = 5; \# \text{ bits} = 6\)
  - Check \(2^5 < 1 \leq 52 < 64\) \(\text{YES}\)
  - Check \(2^6 < 1 \leq 52 < 64\) \(\text{YES}\)
- \(102: \log_2(102) = 6.67; \lfloor \log_2(6.67) \rfloor = 6; \# \text{ bits} = 7\)
  - Check \(2^5 < 1 \leq 102 < 128\) \(\text{YES}\)
  - Check \(2^6 < 1 \leq 102.127 < 128\) \(\text{YES}\)
- \(276: \log_2(276) = 8.10; \lfloor \log_2(8.10) \rfloor = 8; \# \text{ bits} = 9\)
  - Check \(2^7 < 1 \leq 276 < 512\) \(\text{YES}\)
  - Check \(2^8 < 1 \leq 276.545 < 512\) \(\text{YES}\)
  - Check \(2^9 < 1 \leq 276.255 < 512\) \(\text{YES}\)

Decimal to Binary (2’s comp)
- First get number of bits \(\lfloor \log_2(\text{abs}(\text{number})) \rfloor + 2\)
- If positive number, then use process we developed before and you are done
- If negative number,
  - First get representation of the absolute value
  - Then invert all bits
  - Then add +1 to the inverted bits

Decimal to Binary 2’s complement
- 132
  1. \# bits = 7
  2. Negative number
     a. Representation of -132 = 1011100
     b. Invert all bits:
        \[01001010\]
        \[+0000001\]
        \[= 01001011\]
  3. Add +1:
        \[01001011\]
        \[+1011100\]
        \[= 11001100\]

2’s complement binary to decimal
- If MSB is 0, same as unsigned
- If MSB is 1, reverse steps:
  a. Invert all bits
  b. Add +1
  c. Now determine magnitude
     Remember it is a negative number

2’s Complement Binary to decimal
- \(1001100\)
- MSB is 1
  a. Invert all bits:
     \[0110011\]
  b. Add +1:
     \[0110011\]
     \[+0000001\]
     \[= 0110010\]
     \[2^6\] \(\text{YES}\)
     \[2^5\] \(\text{NO}\)
     \[2^4\] \(\text{NO}\)
     \[2^3\] \(\text{NO}\)
     \[2^2\] \(\text{NO}\)
     \[2^1\] \(\text{NO}\)
     \[2^0\] \(\text{NO}\)
     \[\text{ Remainder}\]
     \[= 52\]
  c. Go back to 2’s complement range and check
  d. \(2^6 + 0 = 64\)
  e. \(2^6 - 1 = 63\)
  f. \(2^5 + 0 = 32\)

2’s complement arithmetic
It’s bitwise addition!
- \(52 + (-01) = -49\)
  a. \[0011100\]
  b. \[+1011001\]
  c. \[= 11001100\]

Conversion from binary to decimal
- \(1.101010\)
  1. \# bits = 7
  2. Negative number
     a. Representation of -1.101010 = 10101011
     b. Invert all bits:
        \[0101011\]
        \[+0000001\]
        \[= 0101010\]
  3. Add +1:
        \[0101010\]
        \[+1010100\]
        \[= 1111110\]

Floating Point Standard
IEEE-754 Standard
Single-Precision Representation
- \(S\) \(8\) \(23\)

Decimal to Binary 2’s complement
- 132
  1. \# bits = 7
  2. Negative number
     a. Representation of -132 = 1011100
     b. Invert all bits:
        \[01001010\]
        \[+0000001\]
        \[= 01001011\]
  3. Add +1:
        \[01001011\]
        \[+1011100\]
        \[= 11001100\]

Extension rule: 2s complement
- \(1001100\)
  7 bits
  - To take a number represented in \(X\) bits can get its representation in \(Y\) bits, \(Y > X\), copy the MSB into the "new" bit positions

3’s complement
- \(0101111\)
  1. \# bits = 8
  2. Add +1:
     \[0101111\]
     \[+0000100\]
     \[= 0101101\]

4’s complement
- \(0001011\)
  1. \# bits = 6
  2. Add +1:
     \[0001011\]
     \[+0000100\]
     \[= 0001111\]

5’s complement
- \(0000110\)
  1. \# bits = 6
  2. Add +1:
     \[0000110\]
     \[+0000100\]
     \[= 0001010\]