Set I. Conversions (Number storage)

- Numbering Systems
  - Binary (or any numbering system) into Decimal
    - Binary is base 2. A binary number may be written with a subscript 2 such as 101110001_2 to designate that this is binary number.
    - Generate the place values based on exponents of the base. See example below.
    - Multiply each digit times its associated place value and add the products together. For binary, this effectively means add the place values with 1's.
    - In the example below, the binary value 10000001 is shown be equivalent to 129.

128 2^7	64 2^6	32 2^5	16 2^4	8 2^3	4 2^2	2 2^1	1 2^0
1	0	0	0	0	0	0	1

  - Decimal into Binary
    - Remember the decimal numbering system is base 10, which is the numbering system we use every day.
    - Use a 1 in the largest binary place value that is not larger that the number needed, subtract off, and repeat using the remainder. The remaining bits are zeros.
    - For example, to write the decimal value 56 in 8-bit binary, you would first use a 1 in the place value 32. Now there are 24 left. Use the 16 and there are 8 left. Use the 8 and that makes 56. 56-32=24, 24-16=8; 8-8=0.

128 2^7	64 2^6	32 2^5	16 2^4	8 2^3	4 2^2	2 2^1	1 2^0
0	0	1	1	1	0	0	0

  - Hexadecimal (hex) into binary
    - Hex is based 16, therefore there are 16 digits with a value of 0 to 15. For the 10 to 15 you cannot use two digits so A- F represent 10- 15, respectively.
    - Each hex digit can be written in 4 bits. (Bits are binary digits). To translate a hex value into binary, simply translate the hex into 4-bit binary
    - Example- F3 = 1111 0011, because first you would write the 15 in 4-bit binary, then the 2 in 4-bit binary.

  - Binary into Hex
    - Simply do the shortcut in reverse, write every 4 binary digits in its hex form, remember that A= 10, B=11, C=12, D=13, E=14, F=15.
    - For example, the binary value 0100 1100 is equivalent to the hex value 4C.

Set II. Characters and Signed Numbers

- Character Storage
    - Characters are stored using a binary code such as ASCII or Unicode
        - ASCII - American Standard Code for Information Interchange
            - 8-bit code (1 byte per character)
            - How many bytes would ASCII use to store the word “PAGE”? (4 bytes)
            - Example conversion from hex 3F = 0011 1111
            - The lowercase letter “m” is stored in the binary code equivalent to “6D” in hex. What is the binary code for the lowercase letter “n”? Answer 6E = 0110 1110
        - Unicode – Universal Code
            - 16-bits (2 bytes per character)
            - How many bytes would Unicode use to store the word “PAGE”? (8 bytes)
            - Example conversion from hex 01E9 = 0000 0001 1110 1001
    - Parity bits – check bit (1’s on bit, 0’s off bits)
        - Example: ODD parity scheme
            - 1 0011 1111
            - 0 0011 1110
        - Example: EVEN parity scheme
            - 0 0011 1100
            - 1 0011 1110
        - Example
            - 0 0100 1111 EVEN Not correct
            - 0 0100 1111 ODD Correct
- Twos Complement - Signed Number (negative/positive)
    - Twos complement is the way computers use signed values
    - 2* is the subscript used to designate signed values.
    - The range of values for a given bit length changes (but keeps the same number of values). For example, in a 4-bit twos complement value, instead of the values 0 to 15, twos complement would have the range -8 to positive 7. These ranges both are 16 values.
    - The first bit on the left is the sign bit. 1’s mean the number is negative and 0’s mean the number is positive.
    - To convert a decimal value into the twos complement
        - First determine if it is in range. If it is not in range you need more bits
        - If it is in range, ask if it is positive of negative.
            - Positive number are written as they would normally be written in binary with the specified number of bits.
            - To convert negative values write the one less than the absolute value using the specified number of bits, then flip the bits (1s become 0s and 0s become 1s). For example, to express a -2 in 4-bit twos complement, write a binary 1 in 4 bits -> 0001; then flip the bits -> 1110

Set III.

Facts to know

2^N = # of items; Where N is the number of bits

4 bits = 1 hex digit = 1 nibble

8 bits = 1 byte

Memorize the top row and the right column of this chart.

512 2^9	256 2^8	128 2^7	64 2^6	32 2^5	16 2^4	8 2^3	4 2^2	2 2^1	1 2^0
									1K 2^{10}
									1M 2^{20}
									1G 2^{30}
									1T 2^{40}

Now you have an easy way to express powers of 2.

Examples:

How many XXXXXX would there be with a 32-bit address? Answer 2 to the 32nd power number of XXXXXXX's. XXXXXXX could be ports, devices, colors, printers, etc. Don't worry about what XXXXXXX is – just know that each individual XXXXXX has to have a unique values. $2^{32} = 2^2 * 2^{30} =$ **4G**

How many colors could be supported if the address is 6 hex digits in length? The maximum value would be FFFFFF. Since each hex digit is 4 bits, this is a 24-bit address. $2^{24} = 2^4 * 2^{20} =$ **16M**

How many ports would be supported with a 2 byte port address? Since there are 8 bits in a byte, this is a 16-bit address. $2^{16} = 2^6 * 2^{10} =$ **64K**

Now, looking for N instead of the number of items, here is another example. How many bits would the address need to be to support 128 devices. Answer: **7 bits** because $2^7 = 128$