ChatGPT

Number System Conversions Cheat Sheet

Converting between Binary, Decimal, and Hexadecimal number systems.

Binary to Decimal

Steps:

1. Starting from the rightmost bit of the binary number, assign positional values as powers of 2. The rightmost bit has value 2^0, the next bit 2^1, then 2^2, and so on.
2. Multiply each binary digit by 2 raised to the power of its position value.
3. Add up all these products to get the decimal equivalent of the binary number.

Examples:

- `1011` (binary) → `11` (decimal) – because $1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 = 8 + 0 + 2 + 1 =$ **11**.
- `110101` (binary) → `53` (decimal) – $1\times2^5 + 1\times2^4 + 0\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0 = 32 + 16 + 0 + 4 + 0 + 1 =$ **53**.
- `10011010010` (binary) → `1234` (decimal).

Decimal to Binary

Steps:

1. Divide the decimal number by 2 and record the remainder (0 or 1).
2. Use the quotient from each division to continue dividing by 2, each time recording the next remainder.
3. Repeat this process until the quotient becomes 0.
4. The binary result is the sequence of remainders read in reverse order (with the final remainder as the leftmost bit of the binary number).

Examples:

- `13` (decimal) → `1101` (binary) – 13 ÷ 2 = 6 R1; 6 ÷ 2 = 3 R0; 3 ÷ 2 = 1 R1; 1 ÷ 2 = 0 R1; reading remainders from last to first gives **1101**.
- `156` (decimal) → `10011100` (binary).
- `1000` (decimal) → `1111101000` (binary).

Hexadecimal to Decimal

Steps:

1. Assign positional values for the hex digits, starting with 16^0 for the rightmost digit, 16^1 for the next, 16^2 for the next, and so on.
2. Convert each hexadecimal digit to its decimal value: (0–9 as themselves, A=10, B=11, C=12, D=13, E=14, F=15).
3. Multiply each digit's decimal value by 16 raised to the power of its position value.
4. Sum all these products to obtain the decimal equivalent.

Examples:

- `1A` (hex) → `26` (decimal) – $1×16^1 + 10×16^0 = 16 + 10 =$ **26**.
- `3E8` (hex) → `1000` (decimal) – $3×16^2 + 14×16^1 + 8×16^0 = 768 + 224 + 8 =$ **1000**.
- `BEEF` (hex) → `48879` (decimal).

Decimal to Hexadecimal

Steps:

1. Divide the decimal number by 16.
2. Note the remainder for each division. If a remainder is 10 or greater, convert it to the corresponding hex digit (10→A, 11→B, 12→C, 13→D, 14→E, 15→F).
3. Continue dividing the quotient by 16 until the quotient is 0, recording each remainder.
4. The hexadecimal result is the sequence of remainders read in reverse order (the final remainder gives the leftmost hex digit).

Examples:

- `10` (decimal) → `A` (hex) – 10 ÷ 16 = 0 remainder 10, which is **A** in hex.
- `26` (decimal) → `1A` (hex) – 26 ÷ 16 = 1 remainder 10 (A); 1 ÷ 16 = 0 remainder 1, so reading remainders upward gives **1A**.
- `255` (decimal) → `FF` (hex).
- `2748` (decimal) → `ABC` (hex).

Hexadecimal to Binary

Steps:

1. Replace each hex digit with its 4-bit binary equivalent. (0 → 0000, 1 → 0001, ... 9 → 1001, A → 1010, B → 1011, C → 1100, D → 1101, E → 1110, F → 1111).
2. Concatenate all the 4-bit groups in the same order to get the full binary representation. *(If the leftmost group starts with 0s, you can drop those leading zeros for the final binary number.)*

Examples:

- A5 (hex) → 10100101 (binary) – A = 1010, 5 = 0101, so combined **10100101**.
- 2F (hex) → 101111 (binary) – 2 = 0010, F = 1111, combined 0010 1111. Dropping leading zeros yields **101111**.
- ABC (hex) → 101010111100 (binary) – A = 1010, B = 1011, C = 1100, so **101010111100**.

Binary to Hexadecimal

Steps:

1. Starting from the rightmost end of the binary number, group the bits into chunks of four. (Pad the leftmost group with leading 0s if needed to form a 4-bit group.)
2. Convert each 4-bit group into its hexadecimal equivalent (0000→0, 0001→1, ... 1001→9, 1010→A, ... 1111→F).
3. Write down the hex digits for each 4-bit group in the same order to get the hexadecimal value.

Examples:

- 1010 (binary) → A (hex).
- 11011110 (binary) → DE (hex) – group as 1101 and 1110: 1101 = D, 1110 = E, giving **DE**.
- 101011110001 (binary) → AF1 (hex) – group as 1010, 1111, 0001: 1010 = A, 1111 = F, 0001 = 1, giving **AF1**.
