Project 2: Number Conversion

Preface:

To be able to manipulate floating point numbers at a bitwise level we can start with manipulating scientific notations of binary integers. Recall from elementary school, we can represent any number in their respective scientific notation.

In decimal:

$15000 = 1.5 \times 10^4 = 15 \times 10^3$

with 1.5 being the mantissa; 10 being the base; and 4 being the exponent

In binary we can do the same

In binary:

$111000 = 1.11 \times 2^5 = 111 \times 2^3$

Tasks:

1. Design the following modules:
    a. A 32 to 5 priority encoder where the MSB has the highest priority.
        i. Module name: p_encoder_32_5
        ii. Input: [31:0] in
        iii. Output: [4:0] out
    b. A 5 to 32 decoder.
        i. Module name: decoder_5_32
        ii. Input: [4:0] in
        iii. Output: [31:0] out
2. Combine the two modules to design a module that will convert a 32-bit input to a 32-bit one hot encoded output.
    a. Module name: one_hot_32
    b. Input: [31:0] in
    c. Output [31:0] out

    Example:

    Input: 32'b 0011 1101 0010 0011 1100 0100 1000 0000

    Output: 32'b 0010 0000 0000 0000 0000 0000 0000 0000

3. Represent a 32-bit unsigned number in its scientific notation with 8-bit integer-value *mantissa* (discard any numbers after the binary point) and 5-bit *exponent* format.
    a. Module name: scientific
    b. Input: [31:0] in
    c. Output: [7:0] mantissa, [4:0] exponent

    Examples:

    i. Input: 32'b 1011 1101 1101 1110 0111 1101 1011 1011
    = 1011 1101 x 2^{24}
    Outputs: mantissa: 8'b 1011 1101, exponent: 5'b 11000 (5'd 24)

    ii. Input: 32'b 0000 0000 0010 0101 1011 0111 1011 1101
    = 1001 0110 x 2^{14}
    Outputs: mantissa: 8'b 1001 0110, exponent: 5'b 01110 (5'd 14)

    iii. Input: 32'b 0000 0000 0000 0000 0000 0000 0110 1000
    = 0110 1000 x 2^{0}
    Outputs: mantissa: 8'b 0110 1000, exponent: 5'b 00000 (5'd 0)

4. Write the corresponding test benches for task 2 and 3. The test benches should display the results for the examples given above with at least 4 additional test cases.

Submission:

1. 4 .sv file for the 4 SystemVerilog modules and two separate .sv files for the two test benches
2. Screenshot of the simulation output consol.