YIngqing Huang
yh3ra
floatingpoint.pdf
09/19

Your magic (32 bit) floating point number is -4.890625

- Sign bit = 1 (negative)
- Whole number = 4 = 2^2
    - Exponent = 2 + 127 = 129
    - 129 in binary is 1000 0001
- Decimals = 0.890625; Mantissa
    - Convert the number to absolute rational form: 4 57/64 = 313/64
    - Divide by 2^2 = 4
    - 313/(64*4) = 313/256
    - Subtract 1: 313/256 - 1 = 57/256
        - From 57/256, I cannot subtract ½
        - From 57/256, I cannot subtract ¼
        - From 57/256, I can subtract ⅛ (32/256), yielding 25/256
        - From 25/256, I can subtract 1/16 (16/256), yielding 9/256
        - From 9/256, I can subtract 1/32 (8/256), yielding 1/256
        - From 1/256, I cannot subtract 1/64
        - From 1/256, I cannot subtract 1/128
        - From 1/256, I can subtract 1/256, yielding zero
    - Thus the parts of the mantissa are 1/8 + 1/16 + 1/32 + 1/256
    - The decimal part should be 0011 1001
    - Putting into the 32bit one will be 001 1100 1000 0000 0000 0000
- In all, the binary number should be 1100 0000 1001 1100 1000 0000 0000 0000
- The hex representation should be 0xc09c8000 (little-endian)
- Convert to big-endian: 0x809cc0

Your other magic floating point number is, in hex, 0x0000a03f

- Convert to big-endian is: 0x3fa00000
- Convert to binary is: 0011 1111 1010 0000 0000 0000 0000
- Sign bit = 0
    - Positive
- 8-bit exponent = 0111 1111
    - Convert to decimal: 127
    - 127 - 127 = 0
- 23-bit mantissa = 0100 0000 0000 0000 000
    - Denormalize: 1.01 * 2^0 = 1.01
    - Convert .01 to decimal = 0.25
- Result: 1.25