205388820
Chang

2.72

1) **Because word is defined as an unsigned integer, all integers in this line will be casted to unsigned making it impossible to return a signed integer. In addition, because we are doing a logical right shift and then & 0xFF, this implementation is unable to account for negative numbers.**

2) int xbyte (packed_t word, int bytenum)
{
    return (int)word << ((3-bytenum) << 3) >> 24;
}

2.81

a. (x < y) == (-x > -y)
Can yield 0 because if x is -2147483648 and y is -2, -x would still be -2147483648 and -y would be 2. In this case, (x < y) is true but (-x > -y) is not true.

b. ((x + y) << 4) + y - x == 17*y + 15*x
Always yields 1 because the left side of this expression, (x+y) << 4 is equivalent to 16(x +y). If we distribute, add y, and subtract x, we will be left with 15*x + 17*y which is equal to the right side of the equation, 17*y+15*x.

c. ~x + ~y + 1 == ~(x + y):
Always yields 1 because when we take the ~ of a two's complement number, we are actually switching it's sign and subtracting one (~x = -x - 1). In the context of this expression, ~x+~y+1 = -x - 1 + -y - 1 + 1 = -x + -y - 1. On the other side of the expression ~(x+y) = -x + -y - 1. The left and right side of this expression are equal, so this test evaluates to true.

d. (ux - uy) == -(unsigned)(y - x)
Always yields 1 because taking the negative of any unsigned integer equals 2^32 minus itself (-ux = 2^32 - ux). Knowing this, the left side of this expression can be rewritten as (ux + 2^32 - uy). Likewise, the right side can be written as 2^32 - (uy - ux) → (2^32 - uy + ux). Comparing the left and right sides, we should get (2^32 + ux - uy) == (2^32 + ux - uy) which evaluates to true.

e. ((x >> 2) << 2) <= x:
Always yields 1 because when we shift x to the right, we perform an arithmetic shift which preserves the sign of the integer (01111. . . >> 2 → 00011. . . or 11010. . . >> 2 → 11110. . .) . However, when we shift back to the left, we fill the lowest significant bits with 0s making the new x ultimately smaller than the original x.

Ex. x = 10101010 = -86
x >> 2 = 11101010 = -22
→ << 2 = 10101000 = -88