AMSC 460 - HW 3

Table of Contents

Problem 1

(a) Express x = (12.8)_10 as a double-precision IEEE float fl(x), using the round-tonearest rule.

(12.8)_10 = (12)_10 + (0.8)_10 Integer part: 12/2 = 6 r = 0 6/2 = 3 r = 0 3/2 = 1 r = 1 1/2 = 0 r = 1 -> 12_10 = 1100_2 Fractional part: 0.8 * 2 = 1.6 = 0.6 + 1 0.6 * 2 = 1.2 = 0.2 + 1 0.2 * 2 = 0.4 = 0.4 + 0 0.4 * 2 = 0.8 = 0.8 + 0 -> (0.8)_10 = (0.1100)_2 So (12.8)_10 = (1100.1100)_2, fl(x) = 1.[10011....01] 1001 x 2^3

Sine b_53 = 1 and the rest of bits are Not all zero, we lose R = (0.1001) x 2^(-52) x 2^3 = (0.1001) x 2^(-49) = 0.6 x 2^(-49) 2. By the round-tonearest rule, we add 1 to the b_52 bit to get an addition of 2^(-52) x 2^3 = 2^(-49) Thus we have fl(12.8) = 12.8 + 2^(-49) - 0.6 x 2^(-49) = 12.8 + 0.4 * 2^(-49)

(b) Compute the relative error d = `x # fl(x)`/|x| exactly as a base-10 number, and show that d satisfies the upper bound d ##_mach/2.

```
d = abs(0.4 * 2^(-49))/abs(12.8)
eps/2 - d
```

```
ans = 5.551115123125783e-17 > 0 so the d satisfies the upper bound d ##_
```

Problem 2

(a) Explain why between 2^53 and 2^54, the only double precision floating point numbers that exist are the even numbers.

```
syms a
a = 2^53
while a<=2^54
    if eps(a=1) ==2
        a = a+1
    else
        break
    end
end
```

```
So the smalles # for 2^53 is 2, which means we can add 2 to 2^53 to get
floating point, the distance between each floating point is 2. 2^53 is a
and even number plus 2 is also enen, thus the only double precision floa
numbers between 2^53 and 2^54 are the even numbers.
```

(b) Suppose we type the following into the MATLAB command prompt x = 2^{53}+1 What will MATLAB store in x? Explain.

```
syms x
x = 2^53+1
```

```
x = 2^53 since in matlab we can only rounded up to decimal point
15 digits. Therefore, 2^53
```

Problem 3

Express (12.8)_{10} as a computer word.

```
%2.810 in Decimal number system and want to translate it into Binary.
%Taking whole part of a number is obtained by dividing on the basis
 new
%We get 12 using 2 as a denominator we get 1100_2 as 12_10 in binary
%The fractional part will be rounded by multiplying the basis
%8/2=4......0
%6/2=3......1
%2/2=1......1
%4/2=2......0
%8/2=4......0
%6/2=3......1
%2/2=1......1
%4/2=2......0
%8/2=4......0
%6/2=3......1
%2/2=1......1
%4/2=2......0
%0.8_10 = 0.11001100110_2
%Adding two parts will be 1100.11001100110_2
%Done
```

Published with MATLAB® R2020b