K. J. Somaiya College of Engineering, Mumbai-77

(A Constituent College of Somaiya Vidyavihar University)

Department of Computer Engineering

Batch: D2	**Roll No.: 16010122323**
Experiment / assignment / tutorial No. 5	
Grade: AA / AB / BB / BC / CC / CD /DD	

TITLE: Implementation of IEEE-754 floating point representation

AIM: To demonstrate the single and double precision formats to represent floating point numbers.

Expected OUTCOME of Experiment: (Mention CO attained here)

Books/ Journals/ Websites referred:

1. Carl Hamacher, Zvonko Vranesic and Safwat Zaky, "Computer Organization", Fifth Edition, TataMcGraw-Hill.

2. William Stallings, "Computer Organization and Architecture: Designing for Performance", Eighth Edition, Pearson.

Pre Lab/ Prior Concepts:

The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is a technical standard for floating-point computation established in 1985 by the Institute of Electrical and Electronics Engineers (IEEE). The standard addressed many problems found in the diverse floating point implementations that made them difficult to use reliably and portably. Many hardware floating point units now use the IEEE 754 standard.

The standard defines:

- *arithmetic formats:* sets of binary and decimal floating-point data, which consist of finite numbers (including signed zeros and subnormal numbers), infinities, and special "not a number" values (NaNs)
- *interchange formats:* encodings (bit strings) that may be used to exchange floating-point data in an efficient and compact form
- *rounding rules:* properties to be satisfied when rounding numbers during arithmetic and conversions
- *operations:* arithmetic and other operations (such as trigonometric functions) on arithmetic formats
- *exception handling:* indications of exceptional conditions (such as division by zero, overflow, *etc*

Program Code

```
#include <stdio.h>
#include <math.h> // Include math.h for abs() function

int sign, bin_dec[50], bin_frac[50] = {0};

void binary_dec(int n, int r) {
    for (int i = r; i >= 0; i--) {
        bin_dec[i] = abs(n % 2);
        n = n / 2;
    }
    for (int i = 0; i <= r; i++)
        printf("%d", bin_dec[i]);
}

void binary_frac(float n, int r) {
    for (int i = 0; i <= r; i++) {
        n = n * 2;
        bin_frac[i] = (int)n;
        printf("%d", bin_frac[i]);
        if ((int)n != 0)
            break;
    }
}

int main() {
    printf("Name: Vedansh Savla\n");
    printf("Roll Number: 16010122323\nDivision: D2\n");
    printf("---------------------------------------\n");
    printf("COA exp 5: IEEE floating point representations\n");
    printf("Implementation details:\n");
    printf("---------------------------------------\n");
    printf(" IEEE floating point representations :\n");

    float num;
    int exp32, exp64, bt32 = 0, bt64 = 0;
    printf("Enter the number : ");
    scanf("%f", &num);

    printf("In floating point representation number is represented as:\nSign | 
Exponent | Mantissa\n");
```

```
    if (num > 0)
        sign = 0;
    else {
        sign = 1;
        num = -1 * num;
    }

    int int_num = (int)num;
    float frac_num = num - int_num;

    printf("\n\nSTEP : 1\nConverting number to Binary \n%d = ", int_num);
    binary_dec(int_num, 15);
    printf("\n%f = 0.", frac_num);
    binary_frac(frac_num, 15);
    printf("\nBinary Number = ");
    binary_dec(int_num, 15);
    printf(".");
    binary_frac(frac_num, 15);

    printf("\n\nSTEP : 2\nNormalizing the number\nAfter Normalization\nThe number
is\n1.");

    int p = 0, power = 0;
    while ((bin_dec[p]) != 1)
        p++;

    for (int i = p + 1; i <= 15; i++) {
        printf("%d", bin_dec[i]);
        power++;
    }

    for (int i = 0; i <= 15; i++)
        printf("%d", bin_frac[i]);

    printf(" x 2^%d", power);

    printf("\n\nSTEP : 3\nSingle Precision Representation\n");
    exp32 = power + 127;
    printf("Bias Exponent is : %d", exp32);
    printf("\nIn binary ");
    binary_dec(exp32, 7);
    printf("\nThe number is given as :-\n");
    printf("%d | ", sign);
    binary_dec(exp32, 7);
```

```
    printf(" | ");
    for (int i = p + 1; i <= 15; i++) {
        printf("%d", bin_dec[i]);
        bt32++;
    }

    for (int i = 0; i <= 15; i++) {
        printf("%d", bin_frac[i]);
        bt32++;
        if (bt32 == 23)
            break;
    }

    printf("\n\nSTEP : 4\nDouble Precision Representation\n");
    exp64 = power + 1023;
    printf("Bias Exponent is : %d", exp64);
    printf("\nIn binary ");
    binary_dec(exp64, 10);
    printf("\nThe number is given as :-\n");
    printf("%d | ", sign);
    binary_dec(exp64, 10);
    printf(" | ");
    for (int i = p + 1; i <= 15; i++) {
        printf("%d", bin_dec[i]);
        bt64++;
    }

    for (int i = 0; i <= 40; i++) {
        printf("%d", bin_frac[i]);
        bt64++;
        if (bt64 == 52)
            break;
    }

    printf("\n\n");

    return 0;
}
```

Output

```
Name: Vedansh Savla
Roll Number: 16010122323
Division: D2
---------------------------------------
COA exp 5: IEEE floating point representations
Implementation details:
---------------------------------------
 IEEE floating point representations :
Enter the number : 233.74
In floating point representation number is represented as:
Sign | Exponent | Mantissa


STEP : 1
Converting number to Binary
233 = 0000000011101001
0.740005 = 0.1
Binary Number = 0000000011101001.1

STEP : 2
Normalizing the number
After Normalization
The number is
1.11010011000000000000000 x 2^7
```

```
STEP : 3
Single Precision Representation
Bias Exponent is : 134
In binary 10000110
The number is given as :-
0 | 10000110 | 11010011000000000000000

STEP : 4
Double Precision Representation
Bias Exponent is : 1030
In binary 10000000110
The number is given as :-
0 | 10000000110 | 1001001100000000000000000000000000000000000000000000
```

Post Lab Descriptive Questions

1. Give the importance of IEEE-754 representation for floating point numbers?

The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is a technical standard for floating-point computation which was established in 1985 by the Institute of Electrical and Electronics Engineers (IEEE). The standard addressed many problems found in the diverse floating point implementations that made them difficult to use reliably and reduced their portability. IEEE Standard 754 floating point is the most common representation today for real numbers on computers, including Intel-based PC's, Macs, and most Unix platforms.

There are several ways to represent floating point number but IEEE 754 is the most efficient in most cases. IEEE 754 has 3 basic components:

The Sign of Mantissa –

This is as simple as the name. 0 represents a positive number while 1 represents a negative number.

The Biased exponent –

The exponent field needs to represent both positive and negative exponents. A bias is added to the actual exponent in order to get the stored exponent.

The Normalised Mantissa –

The mantissa is part of a number in scientific notation or a floating-point number, consisting of its significant digits. Here we have only 2 digits, i.e. O and 1. So a normalised mantissa is one with only one 1 to the left of the decimal.

Conclusion

The aim of the experiment is verified.