Plancha 2
Representación Computacional de Números Reales

Arquitectura del Computador

Licenciatura en Ciencias de la Computación

1. Introducción

Esta plancha trata los sistemas de representación de números enteros en el lenguaje de programación C y los operadores de bits para manipular estos números a bajo nivel.

2. Procedimiento

Resuelva cada ejercicio en computadora. Cree un subdirectorio dedicado para cada ejercicio, que contenga todos los archivos del mismo. Para todo ejercicio que pida escribir código, genere un programa completo, con su función `main` correspondiente; evite dejar fragmentos sueltos de programas.

Asegúrese de que todos los programas que escriba compilen correctamente con `gcc`. Se recomienda además pasar a este las opciones `-Wall` y `-Wextra` para habilitar advertencias sobre construcciones cuestionables en el código.

3. Ejercicios

1) Expresar en norma IEEE 754 simple precisión los siguientes números:

a) 29

b) 0.625

c) 0.1

d) 5.75

e) -138

f) -15.125

g) 0.1

2) Haga dos funciones o macros de C para extraer la fracción y el exponente de un `float` sin usar variables auxiliares.

Sugerencia: utilice corrimientos de bits y máscaras. Luego use los tipos definidos en la cabecera `ieee754.h` para corroborar.

3) Convierta a `double` y `float` norma *IEEE 754* el número: $N = 6.225$. Realice el cálculo de manera explícita y luego corrobore el resultado mediante un programa que aproveche las herramientas provistas en el ejercicio anterior. Analizar en cada caso si se ha cometido error de representación y en caso afirmativo la magnitud del mismo.

4) Dados los números $N_1 = (100000)_{10}$, $N_2 = (0.2)_{10}$ y $N_3 = (0.1)_{10}$:

a) Realizar la operación $N_1 \otimes (N_2 \oplus N_3)$ en simple precisión *IEEE 754*.

b) Realizar la operación $(N_1 \otimes N_2) \oplus (N_1 \otimes N_3)$ en simple precisión *IEEE 754*.

c) Repetir para doble precisión *IEEE 754*.

d) Comparar los resultados.

5) Realice el procedimiento de suma (simple precisión) del número $(1.75)_{10} \times 2^{-79}$ con el número `0x19d00000` expresado en IEEE 754 simple precisión.

6) El siguiente programa muestra algunas cualidades de los números `NaN` (*Not A Number*) y la función `isnan` de C, que indica si un flotante es *NaN*.

```
#include <stdio.h>
#include <math.h>

int main(void)
{
    float g = 0.0;
    float f = 0.0 / g;
    printf("f: %f\n", f);
    // ADVERTENCIA: 'NAN' es una extensión de GCC.
    if (f == NAN) {
        printf("Es NAN\n");
    }
    if (isnan(f)) {
        printf("isNaN dice que sí\n");
    }
    return 0;
}
```

a) El programa muestra que comparar con `NAN` retorna siempre falso y para saber si una operación dio *NaN* se puede usar `isnan`. Utilizando las funciones del ejercicio anterior, implemente una función `myisnan` que haga lo mismo que la función `isnan` de C.

b) Implemente otra función, `myisnan2`, que haga lo mismo pero utilizando solo una comparación y sin operaciones de bits.

c) ¿Ocurre lo mismo con $+\infty$?

d) ¿Qué pasa si se suma un valor a $+\infty$?

7) Dados los siguientes números representados en punto flotante IEEE 754 simple precisión, indicar a qué número en formato decimal corresponden y analizar si son números normalizados:

a) $N_1 = (11000010111011010100000000000000)_2$

b) $N_2 = (40600000)_{16}$

c) $N_3 = (00600000)_{16}$

8) Determinar la representación de punto flotante de π utilizando 4 dígitos. Utilizar la norma IEEE 754 simple precisión.

9) Realizar la suma $0.1 + 0.2$ utilizando aritmética en punto flotante con norma IEEE 754 simple precisión. Luego realizar la suma $0.1 + 0.4$. ¿Qué se puede observar?

Ayuda: Al realizar las conversiones se puede reducir la cantidad de operaciones observando la periodicidad de los resultados.

10) Efectuar los siguientes cálculos utilizando aritmética en punto flotante con norma IEEE 754 simple precisión, siendo $a = 12345$, $b = 0.0001$, $c = 45.5$:

a) $(a \oplus b) \oplus c$

b) $a \oplus (b \oplus c)$

Al comparar los resultados en cada uno de las operaciones, ¿qué puede concluirse?

Aclaración: El símbolo $\oplus$ se usa para notar la operación de suma dentro del conjunto de números con parte fraccionaria en el sistema de numeración norma IEEE 754 a diferencia del símbolo $+$ que se usa para notar la suma dentro del conjunto de los números reales.

11) Repetir el ejercicio anterior pero ahora utilizando doble precisión.

Ayuda: No es necesario volver a hacer todo el procedimiento para hacer las conversiones.

12) Dados los números $A = (24)_{10}$, $B = (30)_{10}$ y $C = (15.75)_{10}$:

a) Realizar la suma $S = A \oplus B \oplus C$ en norma IEEE 754 simple precisión.

b) Expresar el resultado en hexadecimal.

c) Convertir el resultado a doble precisión y expresar el resultado en hexadecimal.

13) Considere el conjunto de números de punto flotante $\mathbb{F}(2, 3, -1, 2)$:

a) Determinar el mayor número representable (N_{min}), el menor número representable (N_{max}), el Épsilon de máquina (ϵ_M) y la cantidad de elementos del conjunto $\mathbb{F}$, usando la siguiente representación:

$$N = (-1)^s \times (1.f) \times 2^e,$$

donde s es el bit de signo del número con la convención $s = 0$ para números positivos y $s = 1$ para números negativos, f es la mantisa ($f < 1$) y e es el exponente.

b) Graficar sobre la recta real los números del conjunto $\mathbb{F}$.

Ayuda: Recordar la notación $\mathbb{F}(\beta, t, L, U)$, donde β es la base, t es la cantidad de dígitos significativos de la mantisa, L es el valor mínimo del exponente y U es el valor máximo del exponente.

14) (Opcional) Determinar si el número $(2.89)_{10} \times 10^{10}$ es representable en un formato de coma flotante de 16 bits, con significante normalizado de la forma $(1.b_{-1}b_{-2}\cdots)_2$ con bit implícito para la parte entera y 7 bits para el exponente.