**Control y Sistemas**

**Trabajo práctico: Programación usando números en punto fijo**

**Ejercicio 1**

Compile el siguiente código en C en su PC:

#include <math.h>

#include <stdio.h>

void main(void)

{

signed char a, b, c, d, s1, s2;

a = 127;

b = 127;

c = a + b;

d = a \* b;

s1 = (-8) >> 2;

s2 = (-1) >> 5;

printf("c = %d \n", c );

printf("d = %d \n", d );

printf("s1 = %d \n", s1 );

printf("s2 = %d \n", s2 );

}

1. Verifique el valor de las variables c y d.
2. ¿Son los valores correctos?
3. De no ser así, ¿qué soluciones propone?

**Ejercicio 2**

Cree 2 funciones:

1. Una función para pasar de punto fijo a punto flotante, fx2fp( ).
2. Una función para pasar de punto flotante a punto fijo, fp2fx( ).
3. Verifique el correcto funcionamiento para Q23.8 haciendo

b = fx2fp( fp2fx( 2.4515) )

1. Compare b con 2.4515 para Q23.8 y Q21.10.

**Ejercicio 3**

Cree las siguientes funciones:

1. Una función que implemente redondeo por truncación, truncation().
2. Una función que implemente redondeo al valor más cercano, rounding().
3. Una función que implemente aritmética de saturación, saturation().
4. Luego, multiplique dos números en Q21.10.
5. Compare el resultado de cada esquema de redondeo con el resultado que obtendría usando números en formato double.
6. Opere con una serie de número para verificar que saturation() funciona correctamente.

**Ejercicio 4**

1. Escriba un programa en C que multiplique los números 62.4 y 41.2.
2. Verifique si la representación Q21.10 es adecuada.
3. Compare el resultado en punto fijo con el que se obtiene al operar en formato double.

**Ejercicio 5**

1. Escriba un programa en C que implemente la operación MAC en punto fijo para los siguientes vectores:

double X[5] = {1.1, 2.2, 3.3, 4.4, 5.5 } ;

double Y[5] = {6.6, 7.7, 8.8, 9.9, 10.10 };

1. Represente los números en formato Q21.10.
2. Implemente la operación MAC con dos enfoques:
   1. Multiplique 2 números de 32 bits y redondee antes de sumar cada resultados.
   2. Multiplique 2 números de 64 bits y solo redondee luego de sumar todos los números.

for(i=0; i < 5; i++)

{

acum\_32 += (int32\_t) ( ( ( A[i] \* B[i] ) >> FRACTIONAL\_BITS ) );

acum\_64 += (int64\_t) ( (int64\_t) A[i] \* (int64\_t) B[i] );

}

1. Compare ambos resultados en punto fijo con el que se obtiene al operar en formato double.

**Ejemplo de Resolución**

* 1. c = -2

d = -1