Organización de Computadoras - Primer parcial 2013

Ejercicio 1. Dado el número decimal negativo -125.6302, llevar adelante los siguientes cambios de base:

- **a)** Convertirlo a **octal**, empleando el método de la división tanto para la parte entera como para la parte fraccionaria, expresando el resultado en **complemento a la base**, con cinco dígitos octales para la parte entera y cuatro para la parte fraccionaria
- **b)** Convertirlo a **binario** utilizando el método de la multiplicación tanto para la parte entera como para la parte fraccionaria, expresando el resultado en **complemento a la base disminuida**, con 10 bits para la parte entera y 6 bits para la parte fraccionaria.

Ejercicio 2. Considerando los números octales X = 7521 e Y = -4367, llevar adelante las siguientes operaciones con una precisión de cinco dígitos (incluido el signo) indicando claramente el resultado obtenido y la existencia o no de *overflow*:

1. Calcular $Y - X$, trabajando en octal en complemento a la base.
2. Calcular $X + Y$, trabajando en octal en complemento a la base disminuida.
3. Convertir X e Y a decimal, y calcular $X + Y$ haciendo uso de un hardware que opera en una codificación BCD Exceso-3 y complemento a la base, indicando claramente qué operación se está realizando en cada uno de los pasos intermedios.

Ejercicio 3. Considerando el código Hamming mínima distancia 4 (Hamming extendido) empleando paridad par, ubicando los bits de código de izquierda a derecha y el bit de paridad al final.

- **a)** Calcular los bits de código asociados al dato 110 1100 1010 y armar el codeword correspondiente que integra el dato y los bits calculados

Considerando que el receptor recibe los siguientes codewords, que contienen los bits de dato y de código (C_i y paridad):

- **1)** 0011 0111 1001 0101
- **2)** 0010 1010 1101 0100

Determinar para cada uno de los casos:

- **b)** Cómo trabaja el mecanismo de detección/corrección ante una política $c = 1$, $d = 2$. Justificar apropiadamente la respuesta.
- **c)** Cómo opera el mecanismo de detección/corrección ante una política $c = 0$, $d = 3$. Justificar apropiadamente la respuesta.

Ejercicio 4. Considerando el Código Cíclico Redundante (CRC):

- **a)** Construir el mensaje *T(x)* a transmitir asociado al siguiente mensaje de dato: *M(x)* = 1100 0110 1110 y empleando el polinomio generador $G(x) = x^4 + x^3 + 1$.
- **b)** Suponiendo que el mensaje es modificado durante la transmisión y el receptor recibe un *T(x)* modificado tal que: *T(x)* + *E(x)* = 110 0011 1010 0110, determinar cómo opera el mecanismo de detección de errores y cuál es la conclusión que se alcanza.
- **c)** Comparando el mensaje transmitido *T(x)* y el mensaje recibido *M(x)* = *T(x)* + *E(x)*, ¿cuál es el valor del polinomio *E(x)*? ¿Cuál es la lonfitud de la ráfaga en error?

Ejercicio 5. Pasaje de parámetros por referencia

Dado un arreglo de n pares de enteros:

(a_0,b_0)	(a_1,b_1)	(a_2,b_2)	…	(a_{n-1},b_{n-1})

Declarado como se muestra a continuación:

```
struct registro {
      int a;
      int b;
};
typedef struct registro tReg;
```

- **a)** Implementar en C una *única función* void sumareincrementar() que dado un arreglo R de tipo tReg y un entero **n** que denota su longitud, recursivamente, sume todas las componentes *a* del arreglo e incremente todas las componentes *b* en la suma obtenida. Por ejemplo, si el arreglo de entrada es

(1,2)	(3,4)	(5,6)	(6,7)

la suma de las componentes *a* es 15, y el arreglo debe modificarse de la siguiente forma:

(1,17)	(3,19)	(5,21)	(6,22)

- **b)** Completar en la siguiente función *cáscara* el código necesario para llamar a la función del inciso anterior. Esta función *cáscara* recibe como parámetro, al menos, el arreglo de registros **A** inicializado.

```
void modificar(tReg A[], …) {
      …
      Sumareincrementar(…)
}
```