La Fuente

Organización de Computadoras – Recuperación 2do Parcial - Tema B - 2012

1_ Dado un sistema de punto flotante con mantisa fraccionaria normalizada con bit implícito en BCS de 6 bits, y exponente en CA2 de 4 bits (orden de izq. a der.). ¿Qué número representa 1100001111?

1100001111= _____-0,375_____

Mantisa: 110000. Es fraccionaria con Bit Implícito (BI) y es BCS, por lo tanto el bit más significativo es el signo (negativo) y el segundo equivale a 2^{-2}, ya que no hay que olvidarse del BI que vale 2^{-1} aunque no se vea. Por lo tanto el valor de la mantisa es $-(2^{-1} + 2^{-2})$

Exponente: 1111. Se encuentra en CA2 y es negativo (comienza con 1), por lo tanto hay que convertirlo a BSS: 0001, sin olvidar el signo. Es el -1.

El número entonces es: $-(2^{-1} + 2^{-2}) \times 2^{-1} = -(2^{-2} + 2^{-3}) = -(0{,}25 + 0{,}125) = -0{,}375$

2_ ¿Cuál es el mayor positivo representable (en decimal) con el sistema del punto 1?

Mayor positivo= _____126_____

Mayor número positivo representable: mantisa máxima positiva x $2^{\text{exponente máximo positivo}}$

Mantisa máxima positiva 011111 = $(2^{-1}+2^{-2}+2^{-3}+2^{-4}+2^{-5}+2^{-6})$ – No olvidar el BI –

Exponente máximo positivo: 0111 = 7.

Resultado: $(2^{-1}+2^{-2}+2^{-3}+2^{-4}+2^{-5}+2^{-6}) \times 2^{7} = (2^{6}+2^{5}+2^{4}+2^{3}+2^{2}+2^{1}) = 64+32+16+8+4+2 = 126$

3_ ¿Cuál es la resolución (en decimal) en el extremo inferior positivo del sistema del punto 1?

Resolución= _____2^{-13}_____

La resolución es la diferencia entre dos números consecutivos.

Extremo inferior positivo es: mantisa mínima positiva x $2^{\text{exponente máximo negativo}}$

Mantisa mínima positiva: 2^{-1} (el bit implícito)

Exponente máximo negativo: 1001 = -7

000001 1001

- 000000 1001 (no escribo el BI, pero sí los resto)

000001 1001 = 2^{-6} x 2^{-7} = 2^{-13} (el "espacio" del BI sigue estando, aunque vale 0)

4_ En el sistema del punto 1, si el exponente fuera en BSS ¿Qué número representa 0111001100?

0111001100= _____2944_____

Mantisa 011100. Vale (2^{-1} + 2^{-3} +2^{-4} + 2^{-5})

Exponente: 1100 Vale (8+4) = 12.

Resultado: (2^{-1} + 2^{-3} +2^{-4} + 2^{-5}) x 2^{12}= (2^{11} + 2^{9} +2^{8} + 2^{7})= 2048 + 512 + 256 + 128 = 2944

5_ En IEEE 754 ¿Qué valor representa la cadena 1 01111111 00000000000000000000000? _-1_

Este es el caso donde 00000000<Exponente<11111111 y mantisa cualquiera.

El valor será ±1,M x $2^{(E(BSS)\ -127)}$, por lo tanto será – 1,0 (en binario) x $2(^{127-127})$= - 2^{0} x 2^{0} = -2^{0}=-1

6_ Escriba la cadena que representa al número 1024,125 en el sistema IEE 754 de simple precisión:

_____0__10001010__00000000000010000000000_____

Lo primero que se hace es convertir el número a BSS:

1024,125 = 010000000000,001, multiplicándolo por 2 elevado al exponente 0, que da como resultado el mismo número

Luego, agregamos los ceros a la izquierda de la parte entera, y/o a la derecha de la fraccionaria (ya que no valen nada) para completar los 23 bits del formato IEEE 754 simple precisión:

Mantisa: 0000010000000000,0010000

Exponente: 00000000

Luego, modificamos la mantisa para que nos quede de la forma "1,…", para esto hay que achicarla, por lo tanto, para que quede el mismo número, hay que agrandar el exponente (tantas unidades como lugares se corrió la coma).

Se mueve 10 lugares, por lo tanto el exponente queda +10, que en BSS es 1010

Entonces:

Mantisa: 1,00000000000010000000000

Exponente: 00001010 (BSS)

Por último, hay que ajustar los números en base al estándar (La mantisa es fraccionaria normalizada, con la coma después del primer bit (que es siempre 1) y el exponente está representado en exceso (2^{n-1}-1), o sea, exceso 2^{8-1}-1, exceso 127.
Por lo tanto, al exponente hay que sumarle el exceso, 127 (01111111).

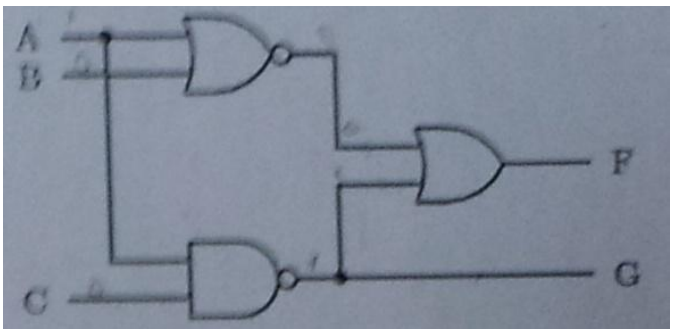
```
  00001010
+ 01111111
  10001001 (Exponente final)
```

En el caso de la mantisa, quitar el 1 delante de la coma (pasa a ser un bit implícito) y completar los 23 bits agregando un cero a la derecha.
Resultado final:
Mantisa: 00000000000010000000000

Exponente: 10001001

7_ Dado el siguiente circuito, si A= 1, B= 0 y C= 0

¿Cuál será el valor de las salidas F y G?



F=____1___ G= ____1_____

En este ejercicio hay que hacer la tabla de verdad para 3 entradas, A, B, C. 2^3=8 bits por cada entrada:

A	B	C	D= A NOR B	F= D OR C	G= A NAND C
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	0	1	1
0	1	1	0	1	1
1	**0**	**0**	**0**	**1**	**1**
1	0	1	0	0	0
1	1	0	0	1	1
1	1	1	0	0	0

Y luego verificamos las salidas en los casos que da el enunciado.

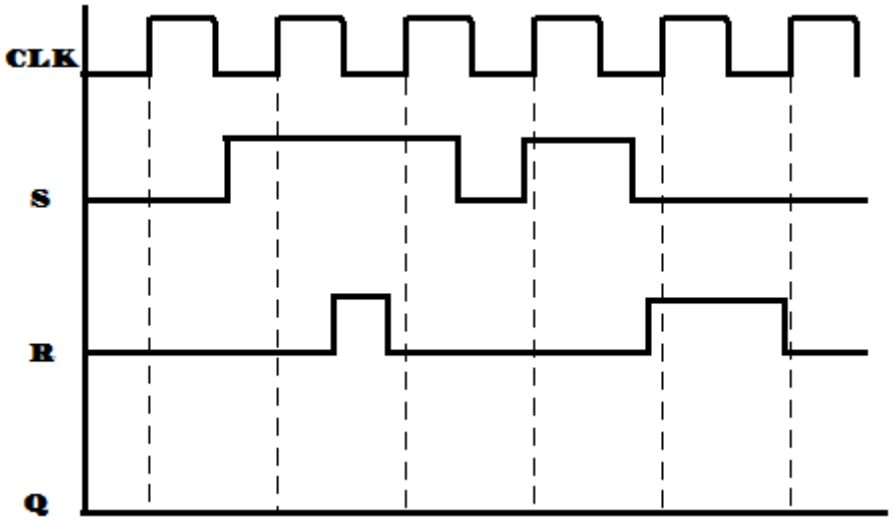
8_ Escriba la ecuación que relaciona las entradas del circuito del punto 8 con la salida F

F= _____(~A.~B.~C)+ (~A.~B.C)+(~A.B.~C)+(~A.B.C)+(A.~B.~C)+(A.B.~C)_____

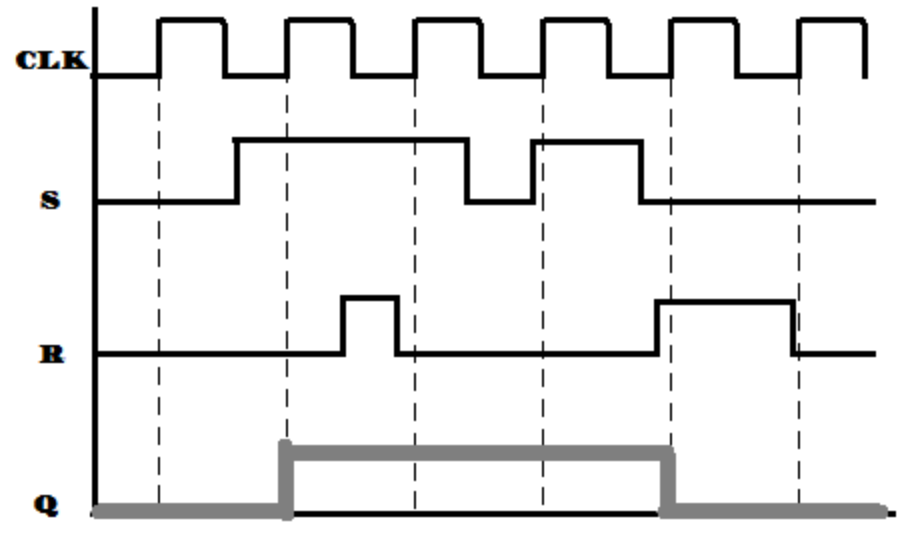
En este caso hay que usar el método de la suma de productos, donde por cada salida 1 en F, se sumará un término al resultado. En cada término, se niegan las entradas que valen 0.

(~A.~B.~C)+ (~A.~B.C)+(~A.B.~C)+(~A.B.C)+(A.~B.~C)+(A.B.~C)

9) Complete el siguiente diagrama de tiempo de un flip flop S-R sincrónico activo por flanco ascendente de CLK. Suponga que inicialmente Q =0.



En un flip flop S-R, Q tomará el valor de S cuando S y R sean distintos, Q quedará con el valor anterior cuando S=R=0 y Q es inválido cuando S=R=1, estos valores se mantienen hasta el siguiente flanco de subida del Clock.



Las preguntas 10 a 15 están referidas al siguiente programa:

```
        ORG     1000H
TAB1    DB      DUP (13, 40, 39, 11, 8, 15, 5, 12)
TAB2    DB      DUP (0, 0, 0, 0, 0, 0, 0, 0)
        ORG     2000H
        MOV     CH, 0
        MOV     AH, 1
        MOV     AL, 0
        MOV     DX, 0
        MOV     CL, OFFSET TAB2 – OFFSET TAB1
VUELTA: MOV     BX, OFFSET TAB1
        ADD     BX, DX
        AND     [BX], AH
        JZ      SALTO
        MOV     BX, OFFSET TAB2
        ADD     BX, DX
        MOV     [BX], AH
        INC     AL
SALTO:  INC     DX
        DEC     CL
        JNZ     VUELTA
        HLT
        END
```

10) ¿Cuál es la dirección de memoria a la que hace referencia la etiqueta TAB2?

1008H

Para calcularlo, hay que tener en cuenta todos los datos anteriores a TAB2. En este caso está:

```
        ORG     1000H
TAB1    DB      DUP (13, 40, 39, 11, 8, 15, 5, 12)
```

TAB1 es DB, o sea que sólo va a ocupar un espacio de memoria por cada dato que tenga en ella.

Por otra parte, es un arreglo duplicado (lo dice DUP) pero, como antes de DUP no hay ningún número, entonces no está duplicado (si hubiera un 2, entonces sería dos veces la tabla).

Ahora se cuentan los datos.

1000H = 13 1001H = 40 1002H = 39 1003H = 11 1004H = 8 1005H = 15

1006H = 5 1007H = 12

Luego comienza TAB2, que por lo tanto está en la dirección 1008H.

11) ¿Cuántos elementos de TAB2 permanecen con el valor 0 al finalizar el programa? 3.

Se guarda un 1 cada vez que el número de TAB1 es impar, por lo tanto quedarán 5 unos y 3 ceros.

Nos damos cuenta de ello en las siguientes instrucciones:

```
VUELTA: MOV   BX, OFFSET TAB1 ; guarda el comienzo de la tabla 1 en BX
        ADD   BX, DX ; incrementa el valor de BX con DX (cada vez que da una vuelta, para moverse en la tabla)
        AND   [BX], AH ; compara el valor de AH (que siempre es 1) con el valor de la tabla (el contenido de BX es el valor de la tabla) haciendo un AND. Como AH es 1, si el valor de TAB1 es impar, terminará en 1 (1 AND 1 = 1); en cambio, si es par, terminará en 0 (0 AND 1 = 0). Todos los demás bits del número que tiene TAB1 darán 0 en el resultado. Entonces este AND determinará que el número es par cuando da 0, e impar cuando da 1.
        JZ    SALTO ;esta instrucción saltará a la etiqueta SALTO cuando el número es par
```

; si el número es impar, el programa continúa

MOV BX, OFFSET TAB2 ;guarda en BX la dirección de TAB2 -1008H

ADD BX, DX ;Le suma lo que tiene DX (que al principio es un 0 y se incrementa cada vez que se guarda algo en TAB2, esto sirve para recorrer la tabla y no pisar cosas almacenadas)

MOV [BX], AH ; AH siempre vale 1, se va a almacenar en el contenido de BX, o sea en la dirección apuntada por él. Pone un 1 en la tabla.

INC AL; AL se incrementa, contando la cantidad de veces que se guardó un 1 en TAB2

12) ¿Qué valor contiene AL al finalizar el programa?

AL se incrementa cada vez que el número de TAB1 es impar. Por lo tanto AL = 5 al finalizar el programa.

13) Si la instrucción "AND [BX], AH" fuera reemplazada por "OR [BX], CH" ¿Qué valor contendría AL al finalizar el programa?

AND [BX], AH, determina si el número es par cuando el resultado da 0 e impar, si el resultado da 1, ya que AH es 1 y el último bit de un byte o cualquier binario determinará si el número es par o impar.

OR [BX], CH, determina lo mismo. Porque CH es 0 y el OR trabaja de la manera inversa a AND. Si un número es par, termina en 0 y 0 OR 0 = 0. (al igual que 0 AND 1 es igual a 0). Si el número es impar, 1 OR 0 = 1 (al igual que 1 AND 1 = 1).

Por lo tanto, el valor de AL sería 5 también.

14) ¿Cuántas veces se produce el salto con la instrucción JNZ VUELTA?

Tantas veces como incrementé DX o decrementé CL, por lo tanto, 8 veces.

15) ¿Cuál es el valor de DX al finalizar el programa?

DX vale 8 porque se incrementa tantas veces como valores tiene la TAB1, ya que esto es controlado por el decremento de CL, que guarda la cantidad de datos que contiene TAB1.