Ejercicios resueltos de Organización de Computadoras

2017

Información del instructor

Instructor
Ing. Dario Kiryczun

Correo electrónico

Ubicación y horarios

Información general

Descripción

La siguiente guía tiene por objetivo poder ayudar al alumno a resolver las prácticas de la materia, por medio de la resolución de ejercicios similares.

Expectativas y objetivos

Se espera que el alumno pueda comprender los temas descriptos en forma clara y sea capaz de completar las prácticas relacionadas de la materia.

Ejercicio

Conversiones de números de diferentes sistemas

Realizar las siguientes sumas y expresar el resultado en hexadecimal:

$$30015|o + 9823|d + 1011011|b + 5293|h + 13560|o$$

Las letras detrás del "|" representan la sigla de la base en la que están expresados.

Inicialmente hay que realizar las conversiones de cada término a decimal. Se toma el dígito de más a la derecha del número a convertir y se lo va multiplicando por la base elevando a una potencia, donde esta potencia es el número de la posición del dígito comenzando desde la derecha menos 1. Por ejemplo, para el primer número de la derecha del valor octal 30015, es el 5, y su posición es la 1 (que menos 1 da 0). Al ser octal, la base es "8" (16 para hexadecimal, 2 para binario, etc.).

Esto sería: (8 elevado a la 0) * 5. Repetirlo con cada dígito y sumar los resultados parciales. Así obtenemos su valor en decimal.

```
30015|o

  (8 elevado a la 0) * 5 = 5

+ (8 elevado a la 1) * 1 = 8

+ (8 elevado a la 2) * 0 = 0

+ (8 elevado a la 3) * 0 = 0

+ (8 elevado a la 4) * 3 = 4096 * 3 = 12288

-----------------------------------------

= 12301|d


9823|d (ya está en decimal)
```

```
1011011|b
  (2 elevado a la 0) * 1 = 1
+ (2 elevado a la 1) * 1 = 2
+ (2 elevado a la 2) * 0 = 0
+ (2 elevado a la 3) * 1 = 8
+ (2 elevado a la 4) * 1 = 16
+ (2 elevado a la 5) * 0 = 0
+ (2 elevado a la 6) * 1 = 64
-----------------------------
= 91|d

5293|h
  (16 elevado a la 0) * 3 = 3
+ (16 elevado a la 1) * 9 = 144
+ (16 elevado a la 2) * 2 = 512
+ (16 elevado a la 3) * 5 = 20480
---------------------------------
= 21139|d

13560|o
  (8 elevado a la 0) * 0 = 0
+ (8 elevado a la 1) * 6 = 8 * 6    = 48
+ (8 elevado a la 2) * 5 = 64 * 5   = 320
+ (8 elevado a la 3) * 3 = 512 * 3  = 1536
+ (8 elevado a la 4) * 1 = 4096 * 1 = 4096
-----------------------------------------
= 6000|d
```

Sumar los resultados de cada conversión:

12301|d + 9823|d + 91|d + 21139|d + 6000|d = 49354|d

Pasar el resultado final a una expresión en Hexadecimal. Para esto se deberá ir dividiendo el valor decimal por la base del sistema resultante, en este caso es dividido por 16 porque es Hexadecimal (8 para octal, 2 para binario, etc.). Al ir dividiendo, se generará un resto que deberá volver a ser dividido hasta que el resto sea menor a 16:

49354 / 16 = 3084 con un resto de 10

3084 / 16 = 192 con un resto de 12

192 / 16 = 12 con un resto de 0

Luego se toma el último resultado que es 12 (de dividir 192 y 16) y se une con los restos conformando una cadena de números, tomándolos de abajo hacia arriba:

12 0 12 10

Por último, se transforman los valores mayores a 9 a las siguientes letras: 10=A ; 11=B ; 12=C ; 13=D ; 14=E ; 15=F

En este caso, siendo que 12 es la C y 10 es la A, el resultado final del ejercicio es:

C0CA

Ejercicio 2

Complemento a 1

Cuando sumamos o restamos números binarios, se nos puede dar la situación donde los signos sean iguales, y en dicho caso no hay inconvenientes mayores debido a que el resultado de la operación tendrá el mismo signo que los números que la componen. Pero en el caso de que los signos sean distintos, por ejemplo que un valor sea negativo y el otro positivo entonces hay que realizar la operación utilizando los complementos.

A continuación resolvemos entonces una situación de operación con signos distintos utilizando el Complemento a 1, para este ejemplo expresado en 8 bits:

Realizar la operación de los siguientes números decimales:

```
- 27

+ 122
```

Expresarlos en binario para 8 bits:

```
- 27 es 00011011

 122 es 01111010
```

El Complemento a 1 (Ca1) se hace invirtiendo cada dígito binario. Para poder hacer una operación donde hay un valor negativo, lo que quiere decir que implica una resta, habrá que aplicar el Ca1 al valor negativo, en este caso el -27:

```
-27 es en binario     00011011

Su Complemento a 1 es 11100100
```

Luego se puede realizar la operación de manera convencional en binario como si fuese una suma:

```
Ca1 de -27:     11100100

 valor 122:   + 01111010

-------------- ---------

               101011110
```

Nótese el 1 que aparece a la izquierda, denominado ***acarreo***. La aparición del acarreo indica que el resultado será positivo.

Se debe tomar entonces el 1 del acarreo y eliminarlo de esa posición, y después sumarlo al resultado:

```
Ca1 de -27:     11100100

 valor 122: +   01111010

-------------- ---------

                01011110

            +          1

-------------- ---------

Resultado final 01011111
```

El resultado final expresado en 8 bits es 01011111 con signo positivo.

En este caso, al haber acarreo significó que el resultado es positivo. Veamos un caso donde el resultado es negativo:

```
Resolver con Ca1 en 8 bits:

+  3  - 14
```

Expresados en binario para 8 bits:

```
3 es   00000011

-14 es 00001110
```

Tomamos el valor con signo negativo y le aplicamos el Ca1:

```
Ca1 de -14 es 11110001
```

Realizamos la operación convencional de suma en binario:

```
   Valor 3:     00000011

Ca1 de -14:   + 11110001

-------------- ---------

               11110100
```

Nótese que en este caso no apareció un 1 como valor de ***acarreo***. Al no haber acarreo, para llegar al resultado final hay que aplicarle al resultado anterior el Ca1, ya que esto significa que estamos ante la presencia de un resultado negativo:

```
Ca1 de 11110100

   es: 00001011
```

El resultado final expresado en 8 bits es 00001011 con signo negativo.

Hay un caso particular dado por la siguiente operación:

```
+5  -5  = ???
```

Entonces:

```
Valor 5:     00000101

Ca1 de -5  + 11111010

------------ --------

             11111111
```

Como no hubo acarreo, aplicamos el complemento a 1 al resultado:

```
El Ca1 de 11111111 es 00000000.
```

El resultado final es 00000000, entonces el signo es irrelevante.

Ejercicio 3

Representación Ca2

Realizar la operación de resta siguiente representando los números en Ca2 en palabras de 8 bits:
116 - 38

Al sumar dos números en Complemento a 2, (Ca2) se está incluyendo también en la operación el bit de signo.
El acarreo producido por el bit de signo se desprecia (no se tiene en cuenta).
Para representar un valor en Ca2, se debe realizar Ca1 y luego sumar 1:

116 es en binario 01110100
38 en binario es 00100110
El complemento a 1 de 38 en 8 bits es 11011001
El complemento a 2 de 38 en o bits es 11011010 (se le sumó 1 al Ca1 de 38)

Se hace la suma convencional de ambos valores:

```
 01110100
 11011010
------------
101001110
```

Como el acarreo se desprecia el resultado es **01001110**, que convirtiéndolo a decimal nos da 78.

Ejercicio adicional:

Sumar los números -115 y + 86 representándolos en Ca2 en palabras de 8 bits.
El número 115 en binario es 01110011. Su complemento a 1 es 10001100. Su complemento a 2 (sumar 1) es 10001101

```
   10001101   representación en Ca2 de -115
 + 01010110   representación en Ca2 de   86
   --------
   11100101
```

Como el resultado es un número negativo (el bit de signo es 1) para leerlo primero hay que recomplementarlo:
De 11100101 su Ca1 es 00011010. Su Ca2 es 00011011. Este valor es -29.

Ejercicio 4

Representación de números en sistemas “Binarios Sin Signo” (BSS) y “Binario Con Signo” (BCS)

Indicar cuál es la capacidad de representación, la resolución y el rango de un sistema BSS de 6 bits.

- El número mínimo representable es 000000 que es el 0 decimal.
- El número máximo representable es 111111 que es el 63 decimal.

  Por lo tanto, el rango es [0, 63]

- Si decimos que “n” es la cantidad de bits a representar, podemos indicar que el rango en un sistema BSS es:

  [0 , (2^n)-1]

- La capacidad de representación es 2^n, es decir 2^6 que da 64 números.
- La resolución es 1, lo cual obtenemos al hacer la resta entre 000000 y su próximo número representable, es decir, 000001.

Indicar cuál es la capacidad de representación, la resolución y el rango de un sistema BCS de 6 bits.

- El número máximo representable es 011111 que es el 31 decimal, que sale de calcular (2^5)-1
- El número mínimo representable es 111111 que es el -31 decimal.

  Por lo tanto, el rango es [-31, 31]

- Si decimos que “n” es la cantidad de bits a representar, podemos indicar que el rango en un sistema BSS es:

  [-(2^(n-1))-1, (2^(n-1))-1]

- La capacidad de representación es 2^n, es decir 2^6 que da 64 números.
- La resolución sigue siendo la misma que para el ejercicio anterior.