Plancha de ejercicios 1

Leandro Spagnolo, Ignacio Ortego y Agustín López

Contents

1 Consigna:

Resolver los ejercicios 2, 3, 4, 6, 8, 12, 13, 15 y 16 de la Plancha 1.

2 Ejercicio 2

2.1 a)

Primero pasamos el 16 a binario:

$$16_{10} = 010000_2$$

Luego, siguiendo el metodo alternativo revisamos todos los dıgitos desde el menos significativo hacia el mas significativo y mientras se consiga un cero dejarlo igual. Al conseguir el primer numero 1, dejarlo igual para luego cambiar el resto de ellos hasta llegar al mas significativo.

$$-16_{10} = 110000_2$$

2.2 b)

$$13_{10} = 001101_2$$

2.3 c)

$$1_{10} = 000001_2$$

Siguiendo el metodo alternativo:

$$-1_{10} = 111111_2$$

2.4 d)

$$10_{10} = 001010_2$$

Siguiendo el metodo alternativo

$$-10_{10} = 110110_2$$

2.5 e)

$$16_{10} = 010000_2$$

2.6 f)

$$31_{10} = 011111_2$$

Siguiendo el metodo alternativo:

$$-31_{10} = 100001_2$$

2.7 Conclusion:

Lo que van a tener en comun todos los numeros negativos es que el primer bit siempre sera un 1 a diferencia de los numeros positivos que todos tendran su primer bit en 0.

Los números negativos se obtienen invirtiendo los bits del número positivo y sumando 1, mientras que los positivos se representan en binario natural.

3 Ejercicio 3

3.1 a)

$$16_{10} = 00010000_2$$

$$-16_{10} = 11110000_2$$

3.2 b)

$$13_{10} = 00001101_2$$

3.3 c)

$$1_{10} = 00000001_2$$

Siguiendo el metodo alternativo:

$$-1_{10} = 11111111_2$$

3.4 d)

$$10_{10} = 00001010_2$$

Siguiendo el metodo alternativo

$$-10_{10} = 11110110_2$$

3.5 e)

$$16_{10} = 00010000_2$$

3.6 f)

$$31_{10} = 00011111_2$$

Siguiendo el metodo alternativo:

$$-31_{10} = 11100001_2$$

3.7 Conclusion:

Al usar 8 bits en lugar de 6, obtenemos una mayor flexibilidad para representar números. Esto nos permite trabajar con un rango más amplio de valores. Además, facilita la manipulación de datos y reduce el riesgo de errores por desbordamiento. En resumen, usar 8 bits nos da más margen y seguridad al manejar números.

4 Ejercicio 4

4.1 a)

$$(00001101)_2 = 13$$

Para pasar de complemento a dos, a decimales hacemos la siguiente suma:

Sumamos todos los bits, menos el primero, en este caso:

$$(0001101)_2 = 0*2^6+0*2^5+0*2^4+1*2^3+1*2^2+0*2^1+1*2^0 = 13$$

Luego al numero obtenido le restamos al primer bit, o sea 128:

$$0 * 2^7 = 0$$

Entonces queda:

$$13 - 0 = 13$$

4.2 b)

$$(01001101)_2 = 77$$

4.3 c)

$$(11100001)_2 = -31$$

En este apartado se ve mas facil la cuenta que hicimos antes para calcular el complemento a dos, en este caso de un numero negativo:

$$(1100001)_2 = 97$$

y tenemos por otro lado:

$$1 * 2^7 = 128$$

Entonces queda:

$$128 - 97 = 31$$

4.4 d)

$$(11111001)_2 = -7$$

4.5 e)

$$(11111111)_2 = -1$$

4.6 f)

$$(00000000)_2 = 0$$

Binario	Octal	Hexadecimal	Decimal
1101100.110	154.46	6C.6	108.75
011110010.010011	362.23	F2.4C	242.296875
10100001.00000011	241.003	A1.03	161.01171875
1001010.010011001100...	112.2314631...	4A.4CCC...	74.3

5 Ejercicio 6

6 Ejercicio 8

6.1 a)

Primero representamos el 10 y el -3 en complemento a dos:

$$10_{10} = 00001010_2$$

$$3_{10} = 00000011_2$$

$$-3_{10} = 11111101_2$$

Luego hacemos la suma:

$$00001010_2 + 11111101_2 = 00000111_2$$

El resultado entonces es:

$$7_{10} = 00000111_2$$

6.2 b)

$$-39_{10} = 11011001_2$$

$$92_{10} = 01011100_2$$

$$11011001_2 + 01011100_2 = 100000101_2 = 5_{10}$$

6.3 c)

$$-19_{10} = 11101101_2$$

$$-7_{10} = 11111001_2$$

$$11101101_2 + 11111001_2 = 111010110_2 = -26_{10}$$

6.4 d)

$$44_{10} = 00101100_2$$

$$45_{10} = 00101101_2$$

$$00101100_2 + 00101101_2 = 111010110_2 = 89_{10}$$

6.5 e)

$$104_{10} = 01101000_2$$

$$45_{10} = 00101101_2$$

$$01101000_2 + 00101101_2 = 10010101_2 = -107_{10}$$

Incorrecto por desbordamiento: se enciende la Overflow Flag

6.6 f)

$$-75_{10} = 10110101_2$$

$$59_{10} = 00111011_2$$

$$10110101_2 + 00111011_2 = 11110000_2 = -16_{10}$$

6.7 g)

$$-103_{10} = 10011001_2$$

$$-69_{10} = 10111011_2$$

$$10011001_2 + 10111011_2 = 101110100_2 = 116_{10}$$

El resultado es 116, pero este resultado es incorrecto, ya que no da eso la suma de

$$-103 + -69 \neq 116$$

Lo que sucedio es que hubo un acarreo y desbordamiento.
Se encienden la carry flag y la overflow flag.

6.8 h)

$$127_{10} = 01111111_2$$

$$1_{10} = 00000001_2$$

$$01111111_2 + 00000001_2 = 10000000_2 = -128_{10}$$

En este caso, no hubo acarreo pero hay desbordamiento ya que la suma deberia dar un numero positivo y resulta en un numero negativo.

6.9 i)

$$-1_{10} = 11111111_2$$

$$1_{10} = 00000001_2$$

$$11111111_2 + 00000001_2 = 00000000_2 = 0_{10}$$

6.10 j)

$$-1_{10} = 11111111_2$$

$$11111111_2 - 00000001_2 = 11111110_2 = -210$$

6.11 k)

$$-8_{10} = 11111000_2$$

$$-127_{10} = 10000001_2$$

$$11111000_2 + 10000001_2 = 01111001_2 = 121_{10}$$

El resultado es incorrecto ya que hay desbordamiento.

6.12 l)

$$127_{10} = 01111111_2$$

$$01111111_2 + 01111111_2 = 11111110_2 = -2_{10}$$

El resultado es incorrecto, ya que hay desbordamiento y acarreo.

7 Ejercicio 12

Table 1: Tabla del servidor

Archivo	Unidades
Celda A	0345
Celda B	0672
Celda C	01250
Celda D	0507
Celda E	0710

7.1 a)

Table 2: Tabla con el equivalente en espacio en decimales

Archivo	Unidades	Decimal
Celda A	0345	229
Celda B	0672	442
Celda C	01250	680
Celda D	0507	327
Celda E	0710	456

7.2 b)

Total de espacio en el servidor en Decimal: 4096

7.3 c)

1. Total del servidor usado en Octal: 4126
2. Total del servidor usado en Decimal: 2134

7.4 d)

1. Total libre del servidor en Octal: 3652
2. Total libre del servidor en Decimal: 1962

7.5 e)

$$0500\,\text{Unidades} = 320\,\text{decimal}$$

Puede entrar en el servidor:

Table 3: Tabla del servidor con el nuevo archivo

Archivo	Unidades
Celda A	0345
Celda B	0672
Celda C	01250
Celda D	0507
Celda E	0710
Celda F	0500

1. Total del servidor usado en Octal: 4626
2. Total del servidor usado en Decimal: 2454

1. Total libre del servidor en Octal: 3152
2. Total libre del servidor en Decimal: 1642

Table 4: Tabla CrytopCoin

Transaccion	Identificador	Valor
A	0x1A2F	0x3B CC
B	0x2C4D	0x4E CC
C	0x3D5A	0x25 CC
D	0x4E7C	0x6A CC
E	0x5F8E	0x72 CC

Table 5: Tabla CrytopCoin en decimales

Transaccion	Identificador	Valor
A	6703	59
B	11341	78
C	15706	37
D	20092	106
E	24462	114

8 Ejercicio 13

8.1 a)

8.2 b)

- Total de las transacciones en decimal: 394
- Total de las transacciones en Hexa: 0x18A

8.3 c)

Table 6: Tabla CrytopCoin con transacciones agregadas

Transaccion	Identificador	Valor
A	0x1A2F	0x3B CC
B	0x2C4D	0x4E CC
C	0x3D5A	0x25 CC
D	0x4E7C	0x6A CC
E	0x5F8E	0x72 CC
F	0x6A7B	0x3C CC
G	0x7B9D	0x4A CC

- Total de las transacciones en decimal: 528
- Total de las transacciones en Hexa: 0x210

Table 7: Tabla CrytopCoin en decimales

Transaccion	Identificador	Valor
A	6703	59
B	11341	78
C	15706	37
D	20092	106
E	24462	114
F	27259	60
G	31645	74

9 Ejercicio 15 y 16

```c
#include <stdio.h>

int is_one(long n, int b) {
  // Desplaza n 'b' veces hacia la derecha
  return (n >> b) & 1;
}

void printbin(unsigned long n) {
  // Recorre cada bit y lo muestra
  for (int i = 31; i >= 0; i--)
    // Muestra un espacio cada 8 bits
    printf ((i%8 == 0) ? "%d " : "%d", is_one (n, i));
}

int main() {
  int nums[6] = { -16, 13, -1, -10, 16, -31 };

  for(int i = 0; i < 6; i++) {
    printbin(nums[i]);
    printf("%8d\n", nums[i]);
  }

  return 0;
}
```