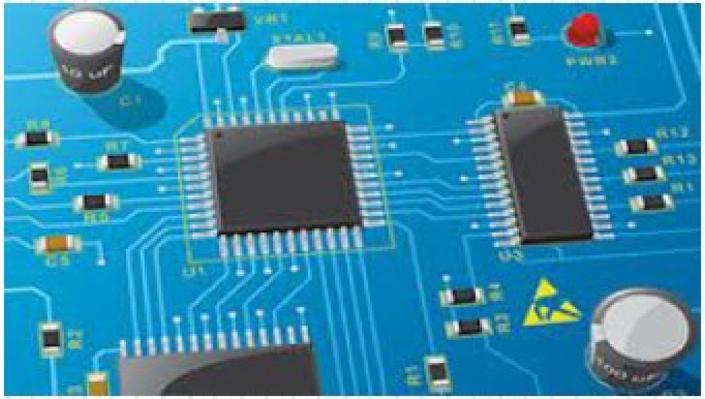
Universidad de Jaén

Compromiso con la sociedad 20 años 1993-2013

T1. Representación de Información
1.1 Introducción. Aritmética Coma Fija

José Santamaría López

Fundamentos de Arquitectura de Computadores



Departamento de Informática

Contenido del capítulo

- Sistema de numeración
- Sistema binario
- Aritmética simple
- Códigos intermedios
- Concepto de palabra
- Aritmética de computadores: coma fija
- Bibliografía
- Actividades

Contenido del capítulo

- Sistema de numeración
- Sistema binario
- Aritmética simple
- Códigos intermedios
- Concepto de palabra
- Aritmética de computadores: coma fija
- Bibliografía
- Actividades

Introducción

- Un computador es una máquina que procesa un conjunto de instrucciones que se ejecutan sobre un conjunto de datos.
- El hombre suministra información a la máquina mediante símbolos (**caracteres**):
  - Caracteres alfabéticos: { a,b,…,z,A,B,…,Z }.
  - Caracteres numéricos: { 0,1,…,9 }.
  - Caracteres especiales y gráficos: { (,) ,*,+,-, ?, …, €, ,♣,♠,β, … }.
  - Caracteres de control: { fin de línea , carácter de sincronización, avance página, pitido, … }.

Introducción (2)

- El ordenador trabaja con sistema binario {0, 1}, en función de si recibe corriente: (5V, 0) ~ (0V, -3.3V).
- Es necesario transformar internamente los datos a una representación de tipo binaria que la máquina sea capaz de procesar.
- Es **muy** importante el correcto almacenamiento y cómputo de los valores numéricos
- No es casualidad que el sistema se denomine como **COMPUTADOR**.

Sistema de Numeración

- Formas de representar información numérica.
- Referencia de *base*: número de dígitos diferentes para representar todos los números.
- El sistema habitual de numeración para las personas es el Decimal (base 10).
- El método de los sistemas electrónicos digitales es el **Binario** (base 2): **{0, 1}.**
- Otros sistemas como el **Octal** (base 8) y el **Hexadecimal** (base 16) facilitan la representación:
  - Mismo valor en menos espacio: Más legible por el ser humano

Sistema de Numeración (2)

- Sistema de representación decimal
- Sistema en base 10 → b=10.

- Cada posición tiene un nombre y peso específicos:
  - Posición 0, peso $b^0=1$, unidades.
  - Posición 1, peso $b^1=10$, decenas.
  - Posición 2, peso $b^2=100$, centenas.
  - Posición 3, peso $b^3=1000$, millares.
  - …

  - Así, $427 = 4\cdot 10^2+2\cdot 10^1+7\cdot 10^0 = 400 + 20 + 7$.

Contenido del capítulo

- Sistema de numeración
- Sistema binario
- Aritmética simple
- Códigos intermedios
- Concepto de palabra
- Aritmética de computadores: coma fija
- Bibliografía
- Actividades

Sistema Binario

- Cada dígito binario usado por el computador se denomina **bit** (contracción de binary digit).
- Múltiplos: se utiliza el factor multiplicador **1024** en lugar de 1000 ($\mathbf{2^{10} = 1024}$).
- Para diferenciarlos del SI, definimos los nombres Kibi (Ki), Mebi (Mi), Gibi (Gi)… para referirse a este tipo de múltiplos en base 2.

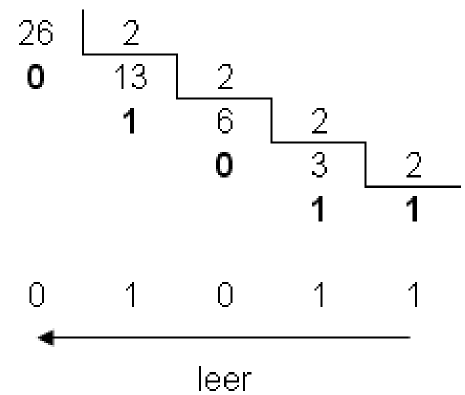
Tabla de múltiplos

El byte es otra de las unidades básicas de medida de la información representada mediante este sistema

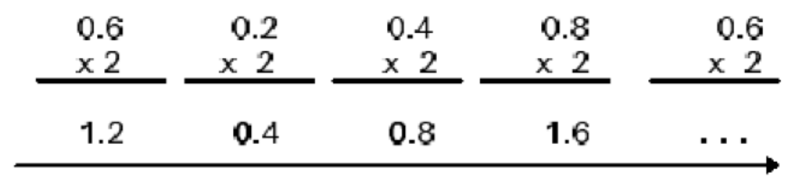
Múltiplo	**Representa**	
Nibble	Conjunto de 4 bits	1001
Byte	Conjunto de 8 bits	10101010
Kibibyte (KiB)	Conjunto de 1024 bytes	2^{10} * 8 bits
Mebibyte (MiB)	Conjunto de 1024 KiB	2^{20} * 8 bits
Gibibyte (GiB)	Conjunto de 1024 MiB	2^{30} * 8 bits
Tebibyte (TiB)	Conjunto de 1024 GiB	2^{40} * 8 bits
Pebibyte (PiB)	Conjunto de 1024 TiB	2^{50} * 8 bits
Exbibyte (EiB)	Conjunto de 1024 PiB	2^{60} * 8 bits
Zetbibyte (ZiB)	Conjunto de 1024 EiB	2^{70} * 8 bits
Yotbibyte (YiB)	Conjunto de 1024 ZiB	2^{80} * 8 bits

Transformación de una base a otra

- Decimal a Binario:
  - Parte entera: debemos dividir el primero por 2 siendo el resto de cada una de las divisiones leído de derecha a izquierda los que compondrán el número binario.
  - Parte fraccionaria: se multiplica por la base siempre con la parte fraccionaria resultante
  - 26.6 10) → 11010.1001… 2)



$26_{10} = 11010_2$



Transformación de una base a otra (2)

- Binario a Decimal: multiplicamos cada cifra del binario por 2 elevado a una potencia que ira disminuyendo hasta llegar a cero.
- Para determinar la primera potencia contamos las cifras del binario (5 en este caso) y disminuimos dicho número en 1 unidad (4 en el ejemplo).

$$11010_2 = 1*2^4+1*2^3+0*2^2+1*2^1+0*2^0$$

$$11010_2 = 1*16+1*8+0*4+1*2+0*1$$

$$11010_2 = 16+8+0+2+0$$

$$11010_2 = 26_{10}$$

Contenido del capítulo

- Sistema de numeración
- Sistema binario
- Aritmética simple
- Códigos intermedios
- Concepto de palabra
- Aritmética de computadores: coma fija
- Bibliografía
- Actividades

Suma de Números Binarios

- Es similar a la suma decimal excepto que se manejan sólo dos dígitos (0 y 1).
- Las sumas básicas son:
  - 0 + 0 = 0
  - 0 + 1 = 1
  - 1 + 0 = 1
  - 1 + 1 = 10 (número 2 en binario)
- Ejemplo: 100110101 + 11010101 =

		1	1	1	1		1		1	
		1	0	0	1	1	0	1	0	1
+										
			1	1	0	1	0	1	0	1
	1	0	0	0	0	0	1	0	1	0

Resta de Números Binarios

- Utilizar el complemento a 2 (ver más adelante):
  - Se toma el número binario de derecha a izquierda y se copia hasta encontrar el primer 1; a continuación se invierten el resto de dígitos.
  - Ejemplo: 1001100 → 0110100; 0101110 → 1010010
  - También sumar "1" al complemento a 1 (bits negados)
- Se suma el minuendo al C2 del sustraendo:
  - La siguiente resta, 91 - 46 = 45, en binario es:
    - 91: 1011011
    - 46: 0101110 → 1010010 (C2)
    - Se desprecia el bit más significativo

```
 1011011
+1010010
 ----------
10101101
```

Antes de hacer el C2 se iguala el nº de cifras!

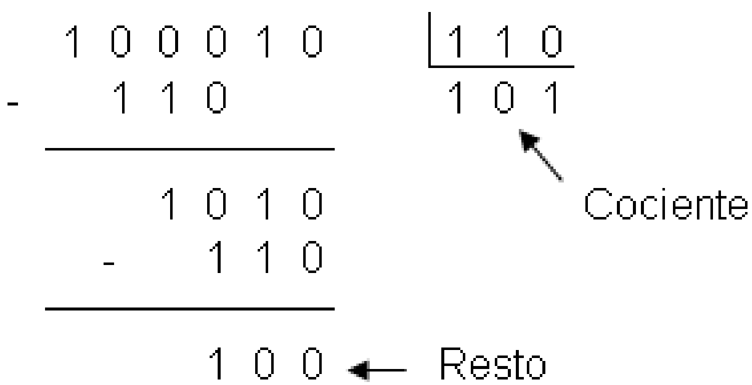
Producto de Números Binarios

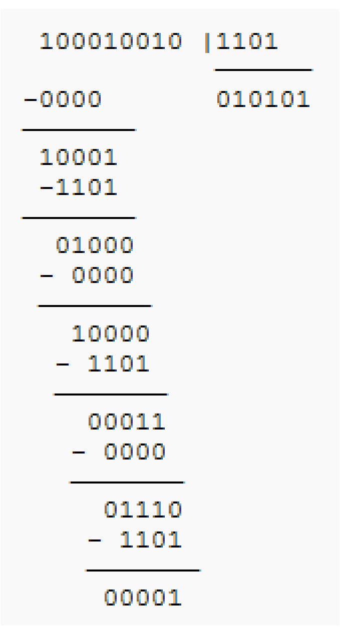
- El producto de números binarios es semejante al decimal, ya que el 0 multiplicado por cualquier otro da 0, y el 1 es el elemento neutro del producto.
- Los productos básicos son:
  - 0 * 0 = 0
  - 0 * 1 = 0
  - 1 * 0 = 0
  - 1 * 1 = 1
- Ejemplo: 10110 * 1001 =

```
      1 0 1 1 0
        1 0 0 1
---------------
      1 0 1 1 0
    0 0 0 0 0
  0 0 0 0 0
1 0 1 1 0
---------------
1 1 0 0 0 1 1 0
```

Cociente de Números Binarios

- La división se realiza en forma semejante al decimal, con la salvedad que las multiplicaciones y restas internas del proceso de la división se realizan en binario.
  - Ejemplo: 100010 / 110 =





Contenido del capítulo

- Sistema de numeración
- Sistema binario
- Aritmética simple
- Códigos intermedios
- Concepto de palabra
- Aritmética de computadores: coma fija
- Bibliografía
- Actividades

Códigos Intermedios

- Facilitan la labor de programación:
  - Trabajar en binario es muy laborioso.
  - Puede llevar a errores por cualquier cambio en la secuencia de ceros y unos.
- Se basan en la facilidad de transformar un número en base dos a otra base mayor que es potencia de dos.
- Existen dos tipos de códigos intermedios:
  - Octal (Base 8)
  - **Hexadecimal (Base 16)**.

Códigos Intermedios (2)

- **Base Hexadecimal**: dígitos {0, …, 9, A,…, F}
- Conversión de hexadecimal a binario:
  - Se hace en sentido inverso: Cada dígito hexadecimal se sustituye por cuatro binarios (tabla).
  - Ejemplo. N: 0x3A6.D = 0011 1010 0110 . 1101 $_{(2}$
- Conversión de hexadecimal a decimal:
  - Utilizando la fórmula de conversión
  - N: 0x3A6.D = $3 \cdot 16^2 + 10 \cdot 16^1 + 6 \cdot 16^0 + 13 \cdot 16^{-1} = 3 \cdot 256\ (768) + 10 \cdot 16\ (160) + 6 \cdot 1\ (6) + 13 \cdot 0.0625\ (0.8125) = 934{,}8125_{(10}$
- Conversión de decimal a hexadecimal:
  - Dividiendo y multiplicando por la base B=16

Decimal	Binario	Hex
0	0000	0x0
1	0001	0x1
2	0010	0x2
3	0011	0x3
4	0100	0x4
5	0101	0x5
6	0110	0x6
7	0111	0x7
8	1000	0x8
9	1001	0x9
10	1010	0xA
11	1011	0xB
12	1100	0xC
13	1101	0xD
14	1110	0xE
15	1111	0xF

Contenido del capítulo

- Sistema de numeración
- Sistema binario
- Aritmética simple
- Códigos intermedios
- Concepto de palabra
- Aritmética de computadores: coma fija
- Bibliografía
- Actividades

Concepto de Palabra

- Una palabra es una cadena finita de bits que son manejados como un **conjunto** por la máquina.
- El tamaño o longitud de una palabra hace referencia al número de bits contenidos en ella.
- Es un aspecto muy importante al momento de diseñar una arquitectura de computadores.
- Los ordenadores modernos normalmente tienen un tamaño de palabra de 16, 32 ó 64 bits.
- El tamaño se define por compatibilidad hacia atrás con los ordenadores anteriores:
  - En arquitectura x86-64 una "palabra" son 16 bits, aunque los operandos de 64-bit ("cuádruple" palabra) son más comunes.

Concepto de Palabra (2)

- **Representación numérica:**
  - Aritmética de coma fija: Números enteros
  - Aritmética de coma flotante: Números reales.
- **Direcciones:**
  - Los contenedores para direcciones de memoria tienen que ser capaces de expresar el rango necesario de valores.
  - A menudo el tamaño utilizado es el de la palabra pero *puede ser un múltiplo o una fracción*.
- **Registros:**
  - Los registros son diseñados con un tamaño apropiado para el tipo de dato que almacenan: enteros, números reales o direcciones.
  - Las arquitecturas de computadores usan registros de “propósito general”. Estos registros se dimensionan para permitir los más tipos de valores más grandes, i.e. el tamaño de palabra de la arquitectura.

Concepto de Palabra (3)

- **Transferencia memoria-procesador:**
  - Cuando el procesador lee o escribe del subsistema de memoria a un registro, la cantidad de datos transferidos es una palabra.
  - Obviamente depende del subsistema de memoria en el que nos encontremos (niveles).
- **Instrucciones:**
  - Las instrucciones máquina normalmente son fracciones o múltiplos de la longitud de palabra de la arquitectura.
  - Es una elección natural: instrucciones y datos comparten el mismo subsistema de memoria.
- **¡Al final todo se trata con palabras (cadenas de bits) en el computador!**

Contenido del capítulo

- Sistema de numeración
- Sistema binario
- Aritmética simple
- Códigos intermedios
- Concepto de palabra
- Aritmética de computadores: coma fija
- Bibliografía
- Actividades

Aritmética de Computadores

- Los computadores no almacenan los números con precisión infinita sino *aproximada* empleando un número fijo de bits.
- El programador puede elegir varias representaciones o 'tipos de datos' (short, char, int, float, …)
- Los tipos de datos difieren en el número de bits usados y cómo se almacena el número representado:
  - Coma fija o punto fijo (también denominado 'entero')
  - Coma flotante (denominado 'real') [inglés "floating point"]

Aritmética en coma fija

- Un entero se puede representar empleando todos los bits de una palabra de computadora.
- Existen principalmente 4 tipos de representación:
  - Signo y magnitud
  - Datos en complemento a 2.
  - Datos sesgados o en exceso
  - Codificación BCD (Decimal Codificado en Binario)

Signo y Magnitud -.- Comp. 2

- En **Signo y Magnitud**, un bit se reserva para el signo.
  - En una máquina con longitud de palabra de 32 bits los enteros están comprendidos entre $(-2^{31} + 1,\ 2^{31} - 1)$.
  - En este caso +0 y -0 representan el mismo número.

- El **Complemento a 2** es una forma de "invertir" un número binario.
  - <u>Solo aplicable</u> a los números negativos (los positivos se mantienen igual).
  - El MSB será siempre 1 para los números negativos.
  - Se calcula alterando todos los dígitos binarios (C1) y sumando 1 (1111…) al valor resultante.
  - De forma más sencilla:
    - Mantener bits de derecha a izquierda hasta el primer bit = 1
    - El resto de bits se alteran (flip).
    - En caso de número fraccionario, se ejecuta igual, antes de la coma.

Sesgo y BCD

- Para los números sesgados (en exceso o desplazados) se los trata como "sin signo", pero se los "desplaza" sumándoles un "sesgo".
- Se altera el rango de los valores representados, de modo que los negativos empiecen en 0, y los positivos de la mitad hasta el final.
- Ejemplo:
  - Con 8 bits en signo/mag. represento números del -127 (-2^7+1) al 127 (2^7-1)
  - En repr. sesgo elimino el bit de signo: desplazo según número inferior ("127"): [0, 255]
  - Para obtener el valor original, vuelvo a restar el sesgo original (2^7-1).

- La representación BCD se refiere al uso de 4 bits para definir un ÚNICO dígito decimal.
- La codificación resulta equivalente a la representación sin signo. Sin embargo, es poco eficiente pues se pierden 6 valores
- Comodidad por proximidad al sistema decimal: Números de dos o más cifras se representan yuxtaponiendo los valores (conjuntos de 4 bits).

Coma Fija: Resumen

Decimal	Sin signo	Signo y mag.	Compl. 2	Sesgo 4
7	111	-	-	-
6	110	-	-	-
5	101	-	-	-
4	100	-	-	-
3	011	011	011	111
2	010	010	010	110
1	001	001	001	101
+0	000	000	000	100
-0	-	100	000	100
-1	-	101	111	011
-2	-	110	110	010
-3	-	111	101	001
-4	-	-	100	000

Aritmética en coma fija (2)

- Un número representado en formato entero es 'exacto'.
- Las operaciones aritméticas entre números enteros son también 'exactas' siempre que:
  - La solución no esté fuera del rango que se puede representar (generalmente con signo).
  - En estos casos se dice que se comete un **error de desbordamiento** por exceso o por defecto (en inglés: *Overflow y Underflow*).
  - En la división se desprecia cualquier resto.
- La aritmética de coma fija se emplea muy raramente en cálculos no triviales.

Contenido del capítulo

- Sistema de numeración
- Sistema binario
- Aritmética simple
- Códigos intermedios
- Concepto de palabra
- Aritmética de computadores: coma fija
- Bibliografía
- Actividades

Bibliografía

- Patterson y Hennessy: Estructura y Diseño de Computadores: Capítulo 3.

- Prieto, Lloris, Torres: Introducción a la Informática: Capítulo 3.

- Murdocca y Heuring: Principios de Arquitectura de Computadoras: Capítulos 2 y 3.

Contenido del capítulo

- Sistema de numeración
- Sistema binario
- Aritmética simple
- Códigos intermedios
- Concepto de palabra,
- Aritmética de computadores: coma fija
- Bibliografía
- Actividades

Ejercicios

- Ejercicio 1:
  - Expresa, en código binario, los números decimales siguientes: 191, 25, 67, 99, 135, 276
  - Realiza la misma operación pasando por código hexadecimal
- Ejercicio 2:
  - Averigua cuántos números pueden representarse con 8, 10, 16 y 32 bits y cuál es el número más grande que puede escribirse en cada caso. Utiliza la representación sin signo.
- Ejercicio 3:
  - Expresa, en el sistema decimal, los siguientes números binarios:
    - 110111, 111000, 010101, 101010, 10111110, 01011101
  - Repite el ejercicio pasando el número a código hexadecimal

Ejercicios (2)

- Ejercicio 4:
  - Dados dos números binarios: 01001000 y 01000100
  - ¿Cuál de ellos es el mayor? ¿Podrías compararlos sin necesidad de convertirlos al sistema decimal?
- Ejercicio 5:
  - Expresa en el sistema decimal las siguientes cifras hexadecimales: 0x2BC5, 0x100, 0x1FF
- Ejercicio 7:
  - Convierte al sistema hexadecimal los siguientes números decimales: 3519, 1024, 4095

Ejercicios (3)

- Ejercicio 8:
  - Convierte a hexadecimales los siguientes números binarios:
    - 1010100101011101010, 111000011110000, 1010000111010111, 1000111101010011010
- Ejercicio 9:
  - Convierte a binario los números hexadecimales siguientes: 0x7A5D, 0x135C, 0x8F8F
- Ejercicio 10:
  - Representa en todos los formatos de coma fija los números siguientes, indicando la longitud de palabra máxima necesaria en cada caso: 22, 255, -31, 0, -3675

Ejercicios (4)

- Ejercicio 11:
  - Cuáles serían los números decimales enteros correspondientes a los números: 1010 1111, 0111 1011, 1000 0000
  - Con las representaciones:
    - Sin signo
    - Signo y magnitud
    - Complemento a 1
    - Complemento a 2
    - Sesgada
    - BCD

Ejercicios (5)

- Realiza las siguientes operaciones en aritmética binaria:
  - 0001 1000 + 0111 1001
  - 0110 1110 – 0110 0011
  - 0100 0000 * 1100 0011
  - 0101 1010 / 0001 0001
  - 0xBE – 0xA8
  - 0x8A4 + 0xFE0
- Siguiendo el formato de coma fija en complemento a 2, con 12 bits, 7 para la parte entera y 5 para la decimal:
  - Representa los números A = 31,72 y B = -0,35
  - Realiza la operación C = A + B
  - Determina los errores absoluto y relativo cometidos en la operación