Organización de Computadoras



Clase 1

# Bibliografía y web de cátedra

* ***Organización y Arquitectura de Computadoras*** ***– Diseño para optimizar prestaciones***, Stallings W., Editorial

Prentice Hall (5º edición).

* ***Organización de Computadoras****,* Tanenbaum A*.,* Editorial

Prentice Hall (4º edición).

* ***Estructura de Computadores y Periféricos***, Martinez

Durá R. et al., Editorial Alfaomega, 2001.

* ***Arquitectura de Computadores-Un enfoque cuantitativo*** Hennessy & Patterson, Editorial Mc Graw Hill (1º edición).

* **http://weblidi.info.unlp.edu.ar/catedras/organizacion/**

# Fechas importantes

REGIMEN INGRESANTES

**Promoción:** deben Aprobarse con el 70% o mas en la primera fecha.

* 27 de ABRIL - 1er PARCIAL (Prácticas 1 y 2)
* Para los que NO Aprobaron COC
* Único Recuperatorio 1er Parcial: 11 de MAYO
* 29 de MAYO - 2do PARCIAL (Prácticas 3 y 4)
* Para los que Aprobaron COC ó 1er parcial
* Único Recuperatorio 2do Parcial: 08 de JUNIO
* 29 de JUNIO - 3er PARCIAL (Prácticas 5 y 6)
* Para los que Aprobaron 1er y 2do parcial Único Recuperatorio 3º Parcial: 06 de JULIO Se tomarán en aula y horario de práctica.
* 13 de JULIO – Evaluación TOTAL
* En aulas y horarios a establecer

Promoción.

Condiciones y fechas.

* Aprobar Parcial 1 y Parcial 2 en primera fecha con el 70% de la nota máxima habilita para rendir
* Evaluación Corta de Teoría (ETC)
* MIÉRCOLES 06 de JUNIO
* Aprobar Parcial 3 en primera fecha con el 70% de la nota máxima y la ETC habilita para rendir
* Evaluación Teórica Promoción
* MIÉRCOLES 11 de JULIO
* Se tomarán en aula y horario de teoría.

# Repaso Curso de Ingreso

* Representación de Datos.
* Números sin signo. BCD.
* Lógica digital. Álgebra de Boole.

# Representación de datos

* Las computadoras almacenan datos e instrucciones en memoria
* Para ello utilizan el sistema binario Razones :
* el dispositivo se encuentra en uno de dos estados posibles (0 ó 1)
* identificar el estado es más fácil si sólo hay dos

# Representación de datos

* Ejemplo :
* lámpara encendida ó apagada
* lámpara encendida con 10 intensidades distintas

Es más fácil conocer el “estado” de la lámpara en el primer caso (encendida ó apagada), que determinar alguna de las 10

intensidades distintas

# Tipos de datos

Las computadoras manejan 4 tipos básicos de datos binarios

* Números enteros sin/con signo
* Números reales con signo
* Números decimales codificados en binario

(BCD)

* Caracteres

# Representación de números enteros

Sin signo

Módulo y signo

Complemento a uno ( Ca1 ) Complemento a la base reducida Complemento a dos ( Ca2 ) Complemento a la base

Exceso

Si el número tiene n bits, puedo representar

* 2n = números distintos

El rango va desde

* 0 a (2n – 1)

Ejemplo: n = 3 bits

Decimal Representación sin signo

0 000 1 001

2 010

.. .....

7 111

Ejemplo: n = 8 bits

|  |  |  |  |
| --- | --- | --- | --- |
| 0 |  |  | 00000000 |
| .. |  |  | .............. |
| 128 |  |  | 10000000 |
| .. |  |  | .............. |
| 254 |  |  | 11111110 |
| 255 |  |  | 11111111 |

 RECORDAR: la cantidad de representaciones distintas depende del número de bits

Nos distintos = 2n

# Sistemas Posicionales

Teorema Fundamental de la Numeración

*n*

*i*

*N* (*dígito*)*i* (*base*)

*i**m*

... *x*4*B*4 *x*3*B*3 *x*2*B*2 *x*1*B*1*x*0*B*0 *x*1*B*1*x*2*B*2 ...

N0 es el valor decimal de una cantidad expresada en

base B y con (n+1+m) dígitos en posiciones i.

# Sistema Decimal

* Base 10.
* Dígitos {0,1,2,3,4,5,6,7,8,9}

3574 = 3000 + 500 + 70 + 4

= 3 x 103 + 5 x 102 + 7 x 101 + 4 x 100

* 3 unidades de mil + 5 centenas + 7 decenas + 4 unidades

3.1416(103100 1101 4102 1103 6104

3 unidades + 1 décima + 4 centésimas + 1 milésima + 4 diezmilésimas

# Sistema Binario

* Base 2.
* Dígitos {0,1}

1001,12 = 1 x 23 + 0 x 22 + 0 x 21 + 1 x 20 + 1 x 2-1

= 8 + 0 + 0 + 1 + 0,5

= 9,510

# Números en punto fijo (1)

* Se considera que todos los números a representar tienen exactamente la misma cantidad de dígitos y la coma fraccionaria está siempre ubicada en el mismo lugar.  En sistema decimal: 0,23 ó 5,12 ó 9,11
* En los ejemplos cada número tiene tres dígitos, y la coma está a la derecha del mas significativo

# Números en punto fijo (2)

* En sistema binario:

11,10 (3,5)10 ó 01,10 (1,5)10 ó 00,11 (0,75)10

* Hay 4 dígitos y la coma está entre el 2do y 3er dígito.
* La diferencia principal entre la representación en el papel y su almacenamiento en computadora, es que no se guarda coma alguna, se supone que está en un lugar determinado.

Punto Fijo: Rango y Resolución

Rango: diferencia entre el número mayor y el menor

Resolución: diferencia entre dos números consecutivos

• Para el ejemplo anterior en sistema decimal

Rango es de 0,00 a 9,99 ó [0,00...9,99]

Resolución es 0,01

2,32 - 2,31 = 0,01 ó 9,99 - 9,98 = 0,01

Rango y Resolución(2)

* Notar que hay un compromiso entre rango y resolución.
* Si mantenemos tres dígitos y desplazamos la coma dos lugares a la derecha, el rango pasa a ser [0,..,999] y la resolución valdrá 1.

En cualquiera de los casos hay 103 números distintos

Ejemplo en binario con 4 bits

Binario Decimal

0000 0

4 parte ent. y 0 parte frac. 0001 1

0010 2

- - - - 00110100 34

0101 5

0110 6

Resolución 0111 7

1000 8

0001 – 0000 = 1001 9

1010 10

00012 = 110 1011 11

* 1. 12
  2. 13
  3. 14
  4. 15

# Ejemplo en ... (1)

Binario Decimal

000,0 0

3 parte ent. y 1 parte frac. 000,1 0,5

001,0 1

- - - , - 001,1010,0 12,5

010,1 2,5

011,0 3

Resolución 011,1 3,5

100,0 4

000,1 – 000,0 = 100,1 4,5

101,0 5

000,12 = 0,510 101,1 5,5

110,0 6

110,1 6,5

111,0 7

111,1 7,5

# Ejemplo en ... (2)

|  |  |  |
| --- | --- | --- |
| 2 parte ent. y 2 parte frac.  - - , - -  Resolución  00,01 – 00,00 =  00,012 = 0,2510 | Binario  00,00 00,01  00,10  00,11  01,00 01,01 01,10  01,11  10,00  10,01 10,10 10,11  11,00  11,01  11,10  11,11 | Decimal  0  0,25 0,5  0,75 1  1,25 1,5  1,75 2  2,25 2,5  2,75 3  3,25 3,5  3,75 |

# Ejemplo en ... (3)

|  |  |  |
| --- | --- | --- |
| 1 parte ent. y 3 parte frac.  - , - - -  Resolución 0,001 – 0,000 =  0,0012 = 0,12510 | Binario  0,000 0,001  0,010  0,011  0,100 0,101 0,110  0,111  1,000  1,001 1,010 1,011  1,100  1,101  1,110  1,111 | Decimal  0  0,125 0,25  0,375 0,5  0,625 0,75  0,875 1  1,125 1,25  1,375 1,5  1,625  1,75  1,875 |

# Ejemplo en ... (4)

|  |  |  |
| --- | --- | --- |
| parte ent. y 4 parte frac.  , - - - -  Resolución ,0001 – ,0000 =  ,00012 = 0,062510 | Binario  ,0000 ,0001  ,0010  ,0011  ,0100 ,0101 ,0110  ,0111  ,1000  ,1001 ,1010 ,1011  ,1100  ,1101  ,1110  ,1111 | Decimal  0  0,0625 0,125  0,1875 0,25  0,3125 0,375  0,4375 0,5  0,5625 0,625  0,6875 0,75  0,8125  0,875  0,9375 |

# Representación y error

* Al convertir un número decimal a sistema binario tendremos 2 casos:
* Sin restricción en la cantidad de bits a usar
* 3,12510 = 11,0012
* Con restricción, por ejemplo 3 bits para parte entera y 4 bits para parte fraccionaria
* 3,12510 = 011,00102

No cometemos error

Representación y error (2)

* Convertir 3,210 con distintas restricciones
* 3 bits para parte fraccionaria: 011,0012 = 3,12510
* Error = 3,2 – 3,125 = 0,075
* 4 bits para parte fraccionaria: 011,00112 = 3,187510
* Error = 3,2 – 3,1875 = 0,0125
* 5 bits para parte fraccionaria: 011,001112 = 3,2187510
* Error = 3,2 – 3,21875 = - 0,01875
* El error más pequeño es 0,0125 entonces 3,1875 es la representación más cercana a 3,2 y podría utilizar sólo 4 bits para la parte fraccionaria.

# Operaciones aritméticas

* Suma en binario
* Al ser un sistema posicional la suma es como en decimal con acarreos entre posiciones al superar el máximo valor representable con un dígito
* Ej: 1+1= 10 (ó ‘1 y me llevo 1’)
* Valores mas grandes requieren mas bits
* Hasta ahora sólo representamos valores en binario sin signo (que llamamos BSS).
* Las restas se podrán realizar si acomodamos los operandos de modo tal que resultado sea mayor que cero, sino deberemos ‘pedir prestado’.

Bits de condición (banderas)

Son bits que el procesador establece de modo automático acorde al resultado de cada operación realizada.

Sus valores permitirán tomar decisiones como:

Realizar o no una transferencia de control.

Determinar relaciones entre números (mayor, menor, igual).

# Banderas aritméticas

Z (cero): vale 1 si el resultado de la operación son todos bits 0.

C (carry): en la suma vale 1 si hay acarreo del bit más significativo; en la resta vale 1 si hay ‘borrow’ hacia el bit más significativo. Cuando la operación involucra números sin signo, C=1 indica una condición fuera de rango.

# Sistema Hexadecimal

* Base 16.
* Dígitos {0,1,2,3,4,5,6,7,8,9, A, B, C, D, E, F}

10,11,12,13,14,15

2CA,816 = 2 x 162 + C x 161 + A x 160 + 8 x 16-1

= 512 + 192 + 10 + 0,5

= 714,510

# Sistema hexadecimal codificado en binario (BCH)

Los dígitos hexadecimales se convierten uno a uno en binario

Para representar un dígito hexadecimal se utilizará siempre 4 bits

Se asocia cada dígito con su valor en binario puro

**Dígito hexadecimal Código BCH**

BCH  **0 0000**

1. **0001**
2. **0010 3 0011**
3. **0100**
4. **0101 6 0110 7 0111**

**8 1000 9 1001 A 1010 B 1011**

1. **1100**
2. **1101 E 1110 F 1111**

# Sistema decimal codificado en binario (BCD)

Los dígitos decimales se convierten uno a uno en binario Para representar un dígito decimal se requerirán 4 bits

Se asocia cada dígito con su valor en binario puro

**Dígito decimal Código BCD**

|  |  |  |
| --- | --- | --- |
| **0** |  | **0000** |
| **1** |  | **0001** |
| **2** |  | **0010** |
| **3** |  | **0011** |
| **4** |  | **0100** |
| **5** |  | **0101** |
| **6** |  | **0110** |
| **7** |  | **0111** |
| **8** |  | **1000** |
| **9** |  | **1001** |

* BCD tiene dos ámbitos de aplicación:

E/S y periféricos, los números se codifican usando un byte por dígito. Se dice que el número está desempaquetado.

En cálculo, se reservan 4 bits por dígito. Se dice que el número está empaquetado.

* Ejemplo: desempaquetado sin signo

834 = 11111000 11110011 11110100

= F8 F3 F4

* Por cada dígito se usan 8 bits, 4 para el binario puro y 4 se completan con “1”
* Desempaquetado con signo
* Con 4 bits hay 24=16 combinaciones posibles de unos y ceros :

Diez usamos para los dígitos 0 al 9

Nos quedan seis sin usar

C16= 1100 representa al signo +

D16 = 1101 representa al signo -

* Ejemplo: desempaquetado con signo

+ 834 = 11111000 11110011 11000100

= F8 F3 C4

* Los 4 bits que acompañan al último dígito son reemplazados por el signo.

Ejemplo:

* - 834 = 11111000 11110011 11010100

= F8 F3 D4

Ejemplo: empaquetado con signo

* + 834 = 10000011 01001100

= 83 4C

* - 34 = 00000011 01001101

= 03 4D

De las 16 representaciones posibles con 4 bits, usamos 10 para los dígitos 0 al 9

Nos sobran 6 combinaciones de 4 bits Al sumar dos dígitos BCD, se nos presentan dos casos :

la suma es ≤ 9 la suma es > 9

* En el primer caso no hay problema

|  |  |  |
| --- | --- | --- |
| 22 |  | 0010 0010 |
|  |  |  |
| 63 |  | 0110 0011 |

41 0100 0001

|  |  |  |
| --- | --- | --- |
| 1 |  | 111 |
| 15 |  | 0001 0101 |

|  |  |  |
| --- | --- | --- |
| 27 |  | 0010 0111 |
| 42 |  | 0011 1100 |

* En el segundo caso ¿Qué sucede ?

3 no válido Cuando la suma de los dos dígitos da >9 hay que generar el “acarreo” porque hay seis combinaciones no usadas

Entonces: cuando la suma de los dígitos es > 9 hay que sumar 6 en ese dígito

1 111

15 0001 0101 27 0010 0111

42 111 1

0011 1100

0110  6

Resultado correcto 0100 0010

* Ejemplo

1 111 1

476 0100 0111 0110

55 0101 0101

531 0100 1100 1011

0110 0110

0101 0011 0001

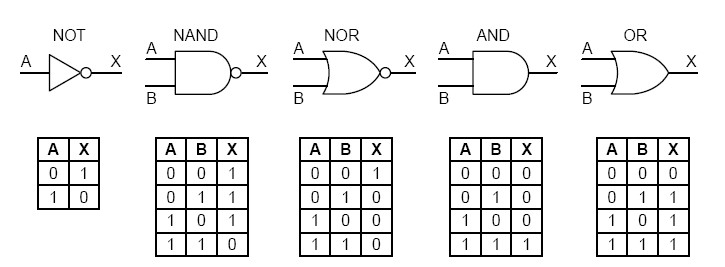
# El nivel de lógica digital

Un circuito digital es en el que están presentes dos valores lógicos

Compuertas son dispositivos electrónicos que pueden realizar distintas funciones con estos dos valores lógicos

Como vimos en el Ingreso las compuertas básicas son: AND, OR, NOT, NAND, NOR y XOR

## Compuertas: símbolo y descripción funcional



# Algebra Booleana

Para describir los circuitos que pueden construirse combinando compuertas, se requiere un nuevo tipo de álgebra, donde las variables y funciones sólo puedan adoptar valores 0 ó 1: álgebra booleana.

# Algebra Booleana

Puesto que una función booleana de n variables tiene 2n combinaciones de los valores de entrada, la función puede describirse totalmente con una tabla de 2n renglones, donde c/u indica un valor de la función (0 ó 1) para cada combinación distinta de las entradas:

=> **tabla de verdad**

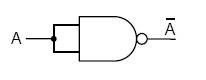
Recordemos algunas identidades del álgebra booleana

|  |  |  |
| --- | --- | --- |
| Identidad | 1.A=A | 0+A=A |
| Nula | 0.A=0 | 1+A=1 |
| Idempotencia | A.A=A | A+A=A |
| Inversa | A.A=0 | A+A=1 |
| Conmutativa | A.B=B.A | A+B=B+A |
| Asociativa | (AB).C=A(BC) | (A+B)+C=A+(B+C) |
| Distributiva | A+B.C=(A+B).(A+C) | A.(B+C)=AB+AC |
| Absorción | A.(A+B)=A | A+A.B=A |
| De Morgan | A.B=A+B | A+B=A.B |

# Leyes de De Morgan

Ejemplo: construir un NOT con NAND

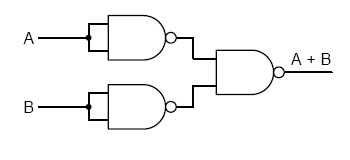
F=A.B=A.A=A



# Leyes de De Morgan

Ejemplo: construir un OR con NAND

F=A+B=A+B=A . B



# Implementación de funciones booleanas

Escribir la tabla de verdad para la función

Dibujar una AND para cada término que tiene un 1 en la columna de resultado

(con sus entradas apropiadas )

Invertir las entradas necesarias

Unir todas las AND a una OR

# Implementación

Ejemplo: construir la tabla de verdad e implementar el circuito de una función booleana M, de tres entradas A, B y C, tal que M=1 cuando la cantidad de ‘1’ en A, B y C es  2 y M=0 en otro caso.

|  |  |
| --- | --- |
|  | ? |
|  |
|  |

A

B  M

C

# Tabla de verdad

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | A | B | C | M | | 0 | 0 | 0 | 0 | | 0 | 0 | 1 | 0 | | 0 | 1 | 0 | 0 | | 0 | 1 | 1 | 1 | | 1 | 0 | 0 | 0 | | 1 | 0 | 1 | 1 | | 1 | 1 | 0 | 1 | | 1 | 1 | 1 | 1 | |  |
|  |
|  |
|  |

# Función M

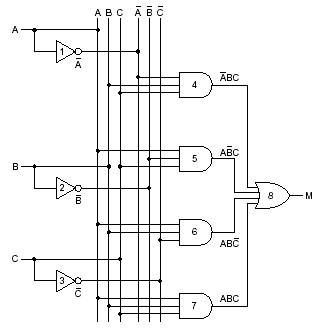
M=ABC + ABC + ABC + ABC

Hay tantos términos como 1s en la tabla

Cada término vale 1 para una única combinación de A, B y C

Las variables que valen 0 en la tabla aparecen aquí negadas

Función M (2)

M=ABC + ABC + ABC + ABC 

# Otro ejemplo

**Supongamos la siguiente Tabla de Verdad**

|  |  |  |
| --- | --- | --- |
| A | B | M |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

**Función M = AB + AB**  **M = A XOR B**

# Recordemos

En un AND, basta que una de sus entradas sea 0 para que la función valga 0. En un OR, basta que una de sus entradas sea 1 para que la función valga 1.

Hacer el XOR con 1 invierte el valor de la variable.

Hacer el XOR con 0 deja el valor de la variable como estaba.

# Circuitos combinatorios

Ejemplo A B S C

A

B

S

C

0

1

0

0

1

1

0

1

0 0

0

1

0 0 0

1. 1 1
2. 0 1

1 1 0

**S** representa la suma aritmética de 2 bits y **C** es el acarreo

**Semi-sumador ó Half adder**

# mayor información …

* Sistemas enteros y Punto fijo
* Apunte 1 de Cátedra
* Operaciones lógicas
* Apunte 3 de Cátedra
* Apéndice 8A: Sistemas de Numeración
* Stallings, 5º Ed.
* Apéndice A: Lógica digital (A.1., A.2.)
* Stallings, 5º Ed.
* Capítulo 3: Lógica digital y representación numérica
* Apuntes COC – Ingreso