SISTEMAS DIGITALES

UNIDAD 1- CÓDIGOS DIGITALES Y DETECCIÓN DE ERRORES

Departamento de Ingeniería Informática y Ciencias de la Computación

UNIVERSIDAD DE ATACAMA

Tabla de Contenidos

Ejercicios

El número binario 11011101 es igual al número decimal:

1. 121
2. 221
3. 441
4. 256

El complemento a 1 de 10111001 es:

1. 01000111
2. 01000110
3. 11000110
4. 10101010

El número binario 10001101010001101111 puede escribirse en hexadecimal como:

1. $AD467_{16}$
2. $8C46F_{16}$
3. $8D46F_{16}$
4. $AE46F_{16}$

Respuestas

1. El número binario 11011101 es igual al número decimal: **b) 221**
2. El complemento a 1 de 10111001 es: **b) 01000110**
3. El número binario 10001101010001101111 puede escribirse en hexadecimal como: **c)** $8D46F_{16}$

Tabla de Contenidos

Tabla de Contenidos

Código decimal binario (BCD)

El código decimal binario (BCD, Binary Coded Decimal) es una forma de expresar cada uno de los dígitos decimales con un código binario.

Código 8421

- La designación 8421 indica los pesos binarios de los cuatro bits (2^3, 2^2, 2^1, 2^0).
- La facilidad de conversión entre los números en código 8421 y los familiares números decimales es la principal ventaja de este código.
- Todo lo que tiene que recordar sobre las diez combinaciones binarias que representan los diez dígitos decimales

Dígito decimal	0	1	2	3	4	5	6	7	8	9
Binario	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Ejemplo

Convertir a BCD los siguientes números decimales: **(a)** 35 **(b)** 98 **(c)** 170 **(d)** 2469

Solución

(a) $\overbrace{3}^{\downarrow}\ \overbrace{5}^{\downarrow}$

$\overbrace{0011}\overbrace{0101}$

(b) $\overbrace{9}^{\downarrow}\ \overbrace{8}^{\downarrow}$

$\overbrace{1001}\overbrace{1000}$

(c) $\overbrace{1}^{\downarrow}\ \overbrace{7}^{\downarrow}\ \overbrace{0}^{\downarrow}$

$\overbrace{0001}\overbrace{0111}\overbrace{0000}$

(d) $\overbrace{2}^{\downarrow}\ \overbrace{4}^{\downarrow}\ \overbrace{6}^{\downarrow}\ \overbrace{9}^{\downarrow}$

$\overbrace{0010}\overbrace{0100}\overbrace{0110}\overbrace{1001}$

Ejemplo

Convertir a decimal cada uno los siguientes códigos BCD:

(a) 10000110 **(b)** 001101010001 **(c)** 1001010001110000

Solución

(a) $\underbrace{1000}_{\downarrow}\underbrace{0110}_{\downarrow}$ → **8 6**

(b) $\underbrace{0011}_{\downarrow}\underbrace{0101}_{\downarrow}\underbrace{0001}_{\downarrow}$ → **3 5 1**

(c) $\underbrace{1001}_{\downarrow}\underbrace{0100}_{\downarrow}\underbrace{0111}_{\downarrow}\underbrace{0000}_{\downarrow}$ → **9 4 7 0**

Tabla de Contenidos

Código Gray

- El código Gray es un código sin pesos y no aritmético; es decir, no existen pesos específicos asignados a las posiciones de los bits.
- La característica más importante del código Gray es que sólo varía un bit de un código al siguiente. Esta propiedad es importante en muchas aplicaciones, tales como los codificadores de eje de posición, en los que la susceptibilidad de error aumenta con el número de cambios de bit entre números adyacentes dentro de una secuencia.
- Como en los números binarios, el código Gray puede tener cualquier número de bits.

Código Gray

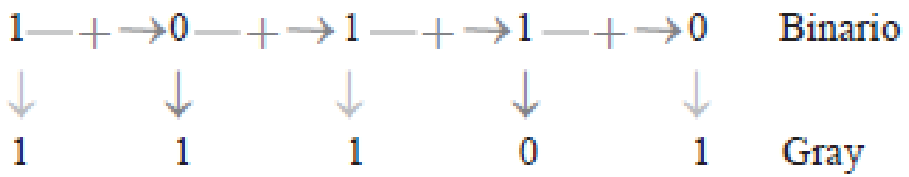
Decimal	Binario	Código Gray	Decimal	Binario	Código Gray
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

Conversión de código binario a código Gray

Las siguientes reglas explican cómo convertir un número binario en un número en código Gray:

1. El bit más significativo (el que está más a la izquierda, MSB) en el código Gray es el mismo que el correspondiente MSB del número binario.
2. Yendo de izquierda a derecha, sumar cada par adyacente de los bits en código binario para obtener el siguiente bit en código Gray. Los acarreos deben descartarse.

Por ejemplo, la conversión del número binario 10110 a código Gray se hace del siguiente modo:

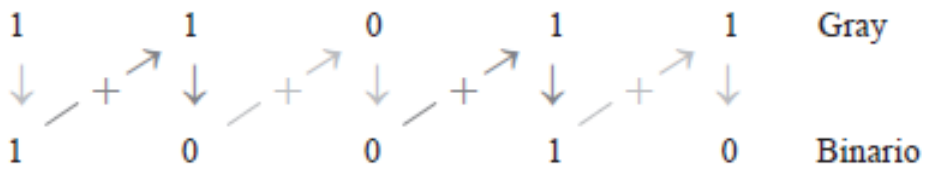


Conversión de código Gray a código binario

Para convertir de código Gray a binario, se utiliza un método similar, pero con algunas diferencias. Se aplican las siguientes reglas:

1. El bit más significativo (bit más a la izquierda) en el código binario es el mismo que el correspondiente bit en código Gray.
2. A cada bit del código binario generado se le suma el bit en código Gray de la siguiente posición adyacente. Los acarreos se descartan.

Por ejemplo, la conversión del número en código Gray 11011 a binario es como sigue

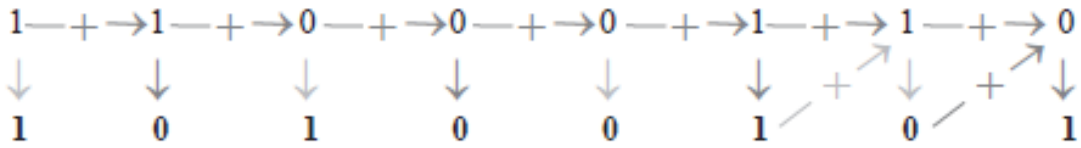


Ejemplo

(a) Convertir el número binario 11000110 a código Gray.
(b) Convertir a binario el código Gray 10101111.

Solución

(a) Binario a código Gray:



(b) Código Gray a binario:

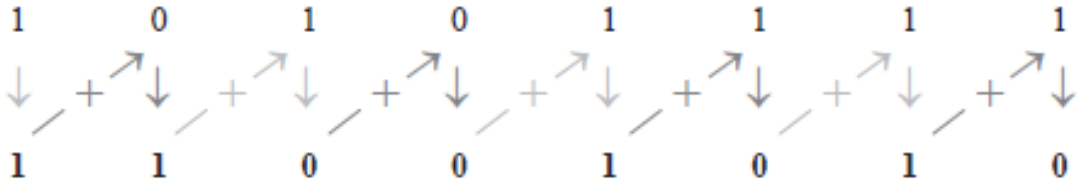


Tabla de Contenidos

ASCII

- El *American Standard Code for Information Interchange* (ASCII, código estándar americano para el intercambio de información) es un código alfanumérico universalmente aceptado, que se usa en la mayoría de las computadoras y otros equipos electrónicos.
- La mayor parte de los teclados de computadora se estandarizan de acuerdo con el código ASCII, y cuando se pulsa una letra, un número o un comando de control, es el código ASCII el que se introduce en la computadora.

Tabla ASCII

Caracteres de control				Símbolos gráficos											
Nombre	Dec	Binario	Hex	Símbolo	Dec	Binario	Hex	Nombre	Dec	Binario	Hex	Símbolo	Dec	Binario	Hex
NUL	0	0000000	00	space	32	0100000	20	@	64	1000000	40	`	96	1100000	60
SOH	1	0000001	01	!	33	0100001	21	A	65	1000001	41	a	97	1100001	61
STX	2	0000010	02	"	34	0100010	22	B	66	1000010	42	b	98	1100010	62
ETX	3	0000011	03	#	35	0100011	23	C	67	1000011	43	c	99	1100011	63
EOT	4	0000100	04	$	36	0100100	24	D	68	1000100	44	d	100	1100100	64
ENQ	5	0000101	05	%	37	0100101	25	E	69	1000101	45	e	101	1100101	65
ACK	6	0000110	06	&	38	0100110	26	F	70	1000110	46	f	102	1100110	66
BEL	7	0000111	07	'	39	0100111	27	G	71	1000111	47	g	103	1100111	67
BS	8	0001000	08	(	40	0101000	28	H	72	1001000	48	h	104	1101000	68
HT	9	0001001	09	)	41	0101001	29	I	73	1001001	49	i	105	1101001	69
LF	10	0001010	0A	*	42	0101010	2A	J	74	1001010	4A	j	106	1101010	6A
VT	11	0001011	0B	+	43	0101011	2B	K	75	1001011	4B	k	107	1101011	6B
FF	12	0001100	0C	,	44	0101100	2C	L	76	1001100	4C	l	108	1101100	6C
CR	13	0001101	0D	_	45	0101101	2D	M	77	1001101	4D	m	109	1101101	6D
SO	14	0001110	0E	.	46	0101110	2E	N	78	1001110	4E	n	110	1101110	6E
SI	15	0001111	0F	/	47	0101111	2F	O	79	1001111	4F	o	111	1101111	6F
DLE	16	0010000	10	0	48	0110000	30	P	80	1010000	50	p	112	1110000	70
DC1	17	0010001	11	1	49	0110001	31	Q	81	1010001	51	q	113	1110001	71
DC2	18	0010010	12	2	50	0110010	32	R	82	1010010	52	r	114	1110010	72
DC3	19	0010011	13	3	51	0110011	33	S	83	1010011	53	s	115	1110011	73
DC4	20	0010100	14	4	52	0110100	34	T	84	1010100	54	t	116	1110100	74
NAK	21	0010101	15	5	53	0110101	35	U	85	1010101	55	u	117	1110101	75
SYN	22	0010110	16	6	54	0110110	36	V	86	1010110	56	v	118	1110110	76
ETB	23	0010111	17	7	55	0110111	37	W	87	1010111	57	w	119	1110111	77
CAN	24	0011000	18	8	56	0111000	38	X	88	1011000	58	x	120	1111000	78
EM	25	0011001	19	9	57	0111001	39	Y	89	1011001	59	y	121	1111001	79
SUB	26	0011010	1A	:	58	0111010	3A	Z	90	1011010	5A	z	122	1111010	7A
ESC	27	0011011	1B	;	59	0111011	3B	[	91	1011011	5B	{	123	1111011	7B
FS	28	0011100	1C	<	60	0111100	3C	\	92	1011100	5C	|	124	1111100	7C
GS	29	0011101	1D	=	61	0111101	3D	]	93	1011101	5D	}	125	1111101	7D
RS	30	0011110	1E	>	62	0111110	3E	^	94	1011110	5E	~	126	1111110	7E
US	31	0011111	1F	?	63	0111111	3F	_	95	1011111	5F	Del	127	1111111	7F

Los caracteres de control ASCII.

- Los primeros treinta y dos códigos de la tabla ASCII (Tabla) representan los caracteres de control. Estos se utilizan para permitir a dispositivos, tales como una computadora o una impresora, que se comuniquen entre sí cuando transfieren información y datos.

Ejemplo

Determinar los códigos binarios ASCII que se han introducido a través del teclado de la computadora cuando se ha escrito la intrucción BASIC: 20 PRINT “A=”;X

Símbolo	Binario	Hexadecimal
2	0110010	32_{16}
0	0110000	30_{16}
Space	0100000	20_{16}
P	1010000	50_{16}
R	1010010	52_{16}
I	1001001	49_{16}
N	1001110	$4E_{16}$
T	1010100	54_{16}
Space	0100000	20_{16}
"	0100010	22_{16}
A	1000001	41_{16}
=	0111101	$3D_{16}$
"	0100010	22_{16}
;	0111011	$3B_{16}$
X	1011000	58_{16}

Tabla de Contenidos

Método de paridad para la detección de errores

- Muchos sistemas emplean un bit de paridad como medio para la detección de errores de bit. Cualquier grupo de bits contiene un número par o impar de 1s.
- Un bit de paridad se añade al grupo de bits para hacer que el número total de 1s en el grupo sea siempre par o siempre impar.
- Un bit de paridad par hace que el número total de 1s sea par, y un bit de paridad impar hace que el número total de 1s del grupo sea impar.
- Por ejemplo, si un sistema trabaja con paridad par, una comprobación que se realice en cada grupo de bits recibidos tiene que asegurar que el número total de 1s en ese grupo es par. Si hay un número impar de 1s, quiere decir que se ha producido un error.

- La Tabla enumera los bits de paridad de cada número BCD, tanto en el caso de paridad par como de paridad impar. El bit de paridad para cada número BCD se indica en la columna P.
- El bit de paridad se puede añadir al principio o al final del código, dependiendo del diseño del sistema. Observe que el número total de 1s, incluyendo el bit de paridad, siempre es par para paridad par, y siempre es impar para paridad impar.

Paridad par		Paridad impar	
P	BCD	*P*	BCD
0	0000	1	0000
1	0001	0	0001
1	0010	0	0010
0	0011	1	0011
1	0100	0	0100
0	0101	1	0101
0	0110	1	0110
1	0111	0	0111
1	1000	0	1000
0	1001	1	1001

Ejemplo

Un sistema de paridad impar recibe los siguientes grupos de códigos: 10110, 11010, 110011, 110101110100 y 1100010101010. Determinar qué grupos, si hay alguno, tienen error.

Solución Puesto que se requiere paridad impar, cualquier grupo con un número par de 1s es incorrecto. Los siguientes grupos contienen error:

110011 y 1100010101010.

Código Hamming de corrección de errores

- Un único bit de paridad puede indicar si existe un error en un determinado grupo de bits.
- Para corregir un error detectado, se necesita más información, ya que hay que identificar la posición del bit erróneo antes de poder corregirlo.
- Debe incluirse más de un bit de paridad en un grupo de bits para poder corregir el error detectado. En un código de 7 bits, existen siete posibles bits erróneos.
- En este caso, tres bits de paridad no sólo pueden detectar el error sino que también pueden especificar la posición del bit erróneo.
- El código Hamming proporciona un método de corrección de un único bit erróneo

Número de bits de paridad.

Si el número de bits de datos se designa por d, entonces el número de bits de paridad, p, se determina mediante la siguiente relación: $2^p \geq d + p + 1$

Example

- Si tenemos cuatro bits de datos, entonces p se calcula por el método de prueba y error Sea $p = 2$. Entonces: $2^p = 2^2 = 4$ y $d + p + 1 = 4 + 2 + 1 = 7$.
- Puesto que 2^p tiene que ser igual o mayor que $d + p + 1$, la relación de la Ecuación no se satisface.
- Probamos de nuevo, sea $p = 3$. Luego, $2^p = 2^3 = 8$ y $d + p + 1 = 4 + 3 + 1 = 8$

Example

- Este valor de p satisface la relación de la Ecuación anterior, por lo que se necesitan tres bits de paridad para poder corregir un único error en cuatro bits de datos.
- Debemos destacar que se proporciona la detección y corrección de errores para todos los bits, tanto de paridad como de datos, del grupo de códigos; es decir, los bits de paridad también se comprueban a sí mismos.

Colocación de los bits de paridad en el código.

- Ahora que ya sabemos cuál es el número necesario de bits de paridad en nuestro ejemplo, debemos colocar correctamente los bits dentro del código. Debe darse cuenta de que, en este ejemplo, el código está formado por cuatro bits de datos y tres bits de paridad.
- El bit más a la izquierda es el bit 1, el siguiente bit es el bit 2, y así sucesivamente, como se muestra a continuación:
  $bit1, bit2, bit3, bit4, bit5, bit6, bit7$
- Los bits de paridad se sitúan en las posiciones que se han numerado haciéndolas corresponder con las potencias de dos en sentido ascendente (1, 2, 4, 8, . . .), del modo siguiente:
  $P1, P2, D1, P3, D2, D3, D4$
- El símbolo Pn designa un determinado bit de paridad y Dn designa cada uno de los bits de datos.

Asignación de los valores de los bits de paridad.

- Para terminar, hay que asignar apropiadamente un valor de 1 o de 0 a cada uno de los bits de paridad. Puesto que cada bit de paridad proporciona una comprobación sobre los restantes bits del código total, tenemos que conocer el valor de dichos otros bits par asignar el valor del bit de paridad.
- Para hallar los valores de los bits, primero expresamos en binario el número correspondiente a cada posición de bit; es decir, escribimos el número binario correspondiente a cada número decimal de posición.

Designación de bit	P_1	P_2	D_1	P_3	D_2	D_3	D_4
Posición de bit	1	2	3	4	5	6	7
Número de posición en binario	001	010	011	100	101	110	111
Bits de datos (Dn)							
Bits de paridad (Pn)							

Observe que el número binario de posición del bit de paridad P1 tiene un 1 como su dígito más a la derecha. Este bit de paridad comprueba las posiciones de todos los bits, incluyéndose a sí mismo, que tienen 1s en la misma posición en el correspondiente número de posición en binario. Por tanto, el bit de paridad P1 comprueba las posiciones de bits 1, 3, 5 y 7.

- El número de posición en binario para el bit de paridad $P2$ tiene un 1 en su posición intermedia. Este bit comprueba entonces todas las posiciones de bit, incluyéndose a sí mismo, que tienen un 1 en esa misma posición. Por tanto, el bit de paridad $P2$ comprueba las posiciones de bit 2, 3, 6 y 7.
- El número de posición en binario para el bit de paridad $P3$ tiene un 1 como su bit más a la izquierda. Este bit comprueba entonces todas las posiciones de bit, incluyéndose a sí mismo, que tienen un 1 en esa misma posición. Por tanto, el bit de paridad $P3$ comprueba las posiciones de bit 4, 5, 6 y 7.
- En cada uno de los casos, se asigna un valor al bit de paridad de modo que la cantidad de 1s en el conjunto de bits que se desea comprobar sea impar o par, dependiendo de lo que se haya especificado.

Determinar el código Hamming para el número BCD 1001 (bits de datos), utilizando paridad par.

Solución

Paso 1. Hallar el número de bits de paridad requeridos. Sea $p = 3$. Entonces,

$$2^p = 2^3 = 8$$

$$d + p + 1 = 4 + 3 + 1 = 8$$

Tres bits de paridad son suficientes.

Número total de bits de código = 4 + 3 = 7

Paso 2. Construir la tabla de posiciones de los bits, como se muestra en la Tabla 2.12, e introducir los bits de datos. Los bits de paridad se determinan en los pasos siguientes.

Designación de bit	P_1	P_2	D_1	P_3	D_2	D_3	D_4
Posición de bit	1	2	3	4	5	6	7
Número de posición en binario	001	010	011	100	101	110	111
Bits de datos			1		0	0	1
Bits de paridad	0	0		1			

Paso 3. Determinar los bits de paridad como sigue:
El bit P_1 comprueba las posiciones de bit 1, 3, 5 y 7, y debe ser igual a 0 para que haya un número par de 1s (2) en este grupo.
El bit P_2 comprueba las posiciones de bit 2, 3, 6 y 7, y debe ser igual a 0 para que haya un número par de 1s (2) en este grupo.
El bit P_3 comprueba las posiciones de bit 4, 5, 6 y 7, y debe ser igual a 1 para que haya un número par de 1s (2) en este grupo.

Paso 4. Estos bits de paridad se anotan en la Tabla 2.12 y el código combinado resultante es 0011001.

Cómo detectar y corregir un error con el código Hamming

- Cada uno de los bits de paridad junto con su correspondiente grupo de bits debe comprobarse de acuerdo con la paridad que se vaya a utilizar.
- Si en una palabra de código hay tres bits de paridad, entonces se realizan tres comprobaciones de paridad. Si hay cuatro bits de paridad, deben realizarse cuatro comprobaciones, y así sucesivamente.
- Cada comprobación de paridad dará un resultado bueno o malo.
- El resultado total de todas las comprobaciones de paridad indica el bit, si existe, en el que se encuentra el error de la siguiente manera:

Implementación

1. Comience con el grupo comprobado por $P1$.
2. Compruebe si el grupo tiene la paridad correcta. Un 0 representa que la comprobación de paridad es correcta y un 1 que es incorrecta.
3. Repita el paso 2 para cada grupo de paridad.
4. El número binario formado por los resultados de todas las comprobaciones de paridad indica la posición del bit del código que es erróneo. Es el código de posición de error. La primera comprobación de paridad genera el bit menos significativo (LSB). **Si todas las comprobaciones son correctas, no habrá error.**

Ejemplo

Suponga que se transmite la palabra código 0011001 y que se recibe 0010001. El receptor no “sabe” lo que se ha transmitido y debe calcular las paridades apropiadas para determinar si el código es correcto. Indique cualquier error que se haya producido en la transmisión si se utiliza paridad par.

En primer lugar, construimos una tabla de posiciones de bits, como la mostrada en la Tabla 2.14.

Designación de bit	P_1	P_2	D_1	P_3	D_2	D_3	D_4
Posición de bit	1	2	3	4	5	6	7
Número de posición en binario	001	010	011	100	101	110	111
Código recibido	0	0	1	0	0	0	1

Tabla 2.14

Primera comprobación de paridad:
El bit P_1 comprueba las posiciones 1, 3, 5 y 7.
En este grupo hay dos 1s.
La comprobación de paridad es correcta. ⟶ 0 (LSB)

Segunda comprobación de paridad:
El bit P_2 comprueba las posiciones 2, 3, 6 y 7.
En este grupo hay dos 1s.
La comprobación de paridad es correcta. ⟶ 0

Tercera comprobación de paridad:
El bit P_3 comprueba las posiciones 4, 5, 6 y 7.
En este grupo hay un 1.
La comprobación de paridad es incorrecta. ⟶ 1 (MSB)

Resultado:
El código de posición del error es 100 (cuatro en binario). Esto quiere decir que el bit que se encuentra en la posición 4 es erróneo. Se ha recibido un 0 y tiene que ser un 1. El código corregido es 0011001, que es el mismo que el código transmitido.