PEARSON

Chapter 2 – Number Systems and Codes

ELEVENTH EDITION

Digital Systems

Principles and Applications

Ronald J. Tocci
Monroe Community College

Neal S. Widmer
Purdue University

Gregory L. Moss
Purdue University

PEARSON

Chapter 2 Objectives

- *Selected areas covered in this chapter*:
  - Converting between number systems.
    - Decimal, binary, hexadecimal.
  - Advantages of the hexadecimal number system.
    - Counting in hexadecimal.
  - Representing decimal numbers using the BCD code.
    - Pros and cons of using BCD.
    - Differences between BCD and straight binary.
  - Purpose of alphanumeric codes such as ASCII code.
  - Parity method for error detection.
    - Determine the parity bit to be attached to a digital data string.

2-1 Binary to Decimal Conversion

- Convert binary to decimal by summing the positions that contain a 1:

$$1 \quad 1 \quad 0 \quad 1 \quad 1_2$$

$$2^4 + 2^3 + 0 + 2^1 + 2^0 = 16 + 8 + 2 + 1$$
$$= 27_{10}$$

- An example with a greater number of bits:

$$1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1_2 =$$
$$2^7 + 0 + 2^5 + 2^4 + 0 + 2^2 + 0 + 2^0 = 181_{10}$$

2-1 Binary to Decimal Conversion

- The double-dabble method avoids addition of large numbers:
  - Write down the left-most 1 in the binary number.
  - Double it and add the next bit to the right.
  - Write down the result under the next bit.
  - Continue with steps 2 and 3 until finished with the binary number.

2-1 Binary to Decimal Conversion

- Binary numbers verify the double-dabble method:

Given: 1 1 0 1 1_2

Results:

$$1 \times 2 = 2$$
$$\underline{+\,1}$$
$$3 \times 2 = 6$$
$$\underline{+\,0}$$
$$6 \times 2 = 12$$
$$\underline{+1}$$
$$13 \times 2 = 26$$
$$\underline{+1}$$
$$27_{10}$$

2-1 Binary to Decimal Conversion

- Reverse process described in 2-1.
  - Note that all positions must be accounted for.

$$
\begin{aligned}
45_{10} = 32 + 8 + 4 + 1 &= 2^5 + 0 + 2^3 + 2^2 + 0 + 2^0 \\
&= 1 \quad\ 0 \quad\ 1 \quad\ 1 \quad\ 0 \quad\ 1_2
\end{aligned}
$$

- Another example:

$$
\begin{aligned}
76_{10} = 64 + 8 + 4 &= 2^6 + 0 + 0 + 2^3 + 2^2 + 0 + 0 \\
&= 1 \quad\ 0 \quad\ 0 \quad\ 1 \quad\ 1 \quad\ 0 \quad\ 0_2
\end{aligned}
$$

2-1 Binary to Decimal Conversion

Repeated Division

Divide the decimal number by 2.

Write the remainder after each division until a quotient of zero is obtained.

The first remainder is the LSB.

The last is the MSB.

$$\frac{25}{2} = \boxed{12} + \text{remainder of } 1 \quad \text{LSB}$$

$$\frac{\boxed{12}}{2} = 6 + \text{remainder of } 0$$

$$\frac{6}{2} = 3 + \text{remainder of } 0$$

$$\frac{3}{2} = 1 + \text{remainder of } 1$$

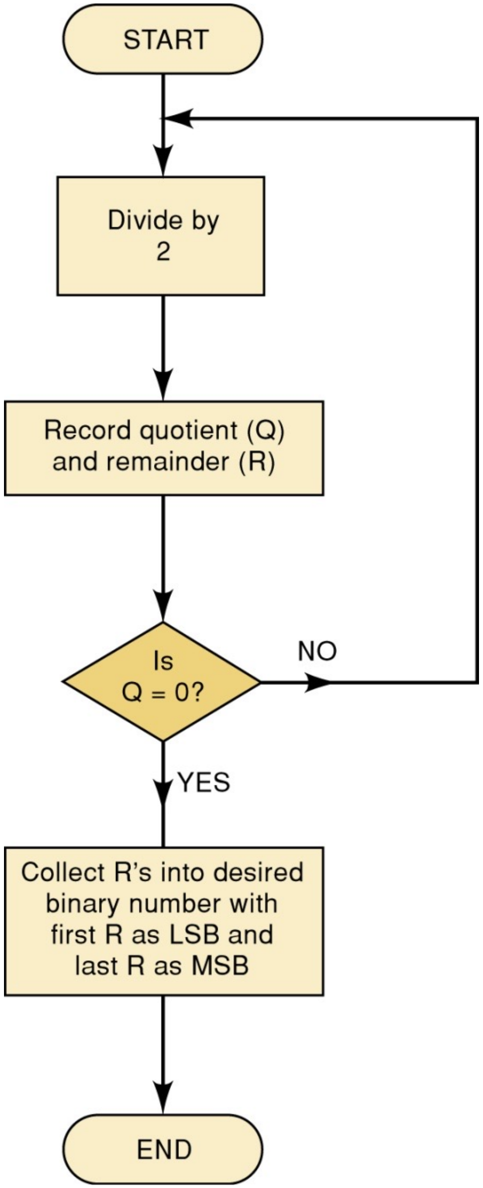
$$\frac{1}{2} = 0 + \text{remainder of } 1 \quad \text{MSB}$$

$$25_{10} = \boxed{1 \quad 1 \quad 0 \quad 0 \quad 1_2}$$

2-2 Decimal to Binary Conversion

Repeated Division

This flowchart describes the process and can be used to convert from decimal to any other number system.



2-3 Hexadecimal Number System

- Convert 37_{10} to binary:

$$\frac{37}{2} = \boxed{18.}5 \longrightarrow \text{remainder of 1 (LSB)}$$

$$\frac{\boxed{18}}{2} = 9.0 \longrightarrow \qquad 0$$

$$\frac{9}{2} = 4.5 \longrightarrow \qquad 1$$

$$\frac{4}{2} = 2.0 \longrightarrow \qquad 0$$

$$\frac{2}{2} = 1.0 \longrightarrow \qquad 0$$

$$\frac{1}{2} = 0.5 \longrightarrow \qquad 1 \text{ (MSB)}$$

2-3 Hexadecimal Number System

- Hexadecimal allows convenient handling of long binary strings, using groups of 4 bits—Base 16
  - 16 possible symbols: 0-9 and A-F

16^4	16^3	16^2	16^1	16^0	16^{-1}	16^{-2}	16^{-3}	16^{-4}

Hexadecimal point

2-3 Hexadecimal Number System

Relationships between hexadecimal, decimal, and binary numbers.

Hexadecimal	Decimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

2-3 Hexadecimal Number System – Hex to Decimal

- Convert from hex to decimal by multiplying each hex digit by its positional weight.

$$\begin{aligned} 356_{16} &= 3 \times 16^2 + 5 \times 16^1 + 6 \times 16^0 \\ &= 768 + 80 + 6 \\ &= 854_{10} \end{aligned}$$

- In a 2nd example, the value 10 was substituted for A and 15 substituted for F.

$$\begin{aligned} 2AF_{16} &= 2 \times 16^2 + 10 \times 16^1 + 15 \times 16^0 \\ &= 512 + 160 + 15 \\ &= 687_{10} \end{aligned}$$

For practice, verify that $1BC2_{16}$ is equal to 7106_{10}

2-3 Hexadecimal Number System – Decimal to Hex

- Convert from decimal to hex by using the repeated division method used for decimal to binary conversion.
- Divide the decimal number by 16
  - The first remainder is the LSB—the last is the MSB.

2-3 Hexadecimal Number System – Decimal to Hex

- Convert 423_{10} to hex:

$$\frac{423}{16} = 26 + \text{remainder of } 7$$

$$\frac{26}{16} = 1 + \text{remainder of } 10$$

$$\frac{1}{16} = 0 + \text{remainder of } 1$$

$$423_{10} = 1A7_{16}$$

- Convert 214_{10} to hex:

$$\frac{214}{16} = 13 + \text{remainder of } 6$$

$$\frac{13}{16} = 0 + \text{remainder of } 13$$

$$214_{10} = \boxed{D6_{16}}$$

2-3 Hexadecimal Number System – Hex to Binary

- Leading zeros can be added to the left of the MSB to fill out the last group.

$$
\begin{aligned}
9F2_{16} &= \quad 9 \qquad\qquad F \qquad\qquad 2 \\
&\qquad \downarrow \qquad\qquad \downarrow \qquad\qquad \downarrow \\
&= 1\ 0\ 0\ 1 \quad 1\ 1\ 1\ 1 \quad 0\ 0\ 1\ 0 \\
&= 100111110010_2
\end{aligned}
$$

For practice, verify that $BA6_{16} = 101110100110_2$

2-3 Hexadecimal Number System – Binary to Hex

- Convert from binary to hex by grouping bits in four starting with the LSB.
  - Each group is then converted to the hex equivalent
- The binary number is grouped into groups of four bits & each is converted to its equivalent hex digit.

$$1\,1\,1\,0\,1\,0\,0\,1\,1\,0_2 = \underbrace{0\;0\;1\;1}_{3}\;\underbrace{1\;0\;1\;0}_{A}\;\underbrace{0\;1\;1\;0}_{6}$$

$$= 3A6_{16}$$

For practice, verify that 101011111 2 = $15F_{16}$

2-3 Hexadecimal Number System – Decimal to Hex to Binary

- Convert decimal 378 to a 16-bit binary number by first converting to hexadecimal.

$$\frac{378}{16} = 23 + \text{remainder of } 10_{10} = A_{16}$$

$$\frac{23}{16} = 1 + \text{remainder of } 7$$

$$\frac{1}{16} = 0 + \text{remainder of } 1$$

2-3 Hexadecimal Number System

To perform conversions between hex & binary, it is necessary to know the four-bit binary numbers (0000 - 1111), and their equivalent hex digits.

Hex	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

2-3 Hexadecimal Number System – Counting in Hex

- When counting in hex, each digit position can be incremented (increased by 1) from 0 to F.
  - On reaching value F, it is reset to 0, and the next digit position is incremented.

Example:

38,39,3A,3B,3C,3D,3E,3F,40,41,42

When there is a 9 in a digit position, it becomes an A when it is incremented.

With three hex digits, we can count from 000_{16} to FFF_{16} which is 0_{10} to 4095_{10} — a total of 4096 = 16^3 values.

2-4 BCD Code

- Binary Coded Decimal (BCD) is a widely used way to present decimal numbers in binary form.
  - Combines features of both decimal and binary systems.
    - Each digit is converted to a binary equivalent.
- BCD is *not* a number system.
  - It is a decimal number with each digit encoded to its binary equivalent.
- A BCD number is *not* the same as a straight binary number.
  - The primary advantage of BCD is the relative ease of converting to and from decimal.

- Convert the number 874_{10} to BCD:
  - Each decimal digit is represented using 4 bits.
    - Each 4-bit group can never be greater than 9.

$$\begin{array}{cccc} 8 & 7 & 4 & \text{(decimal)} \\ \downarrow & \downarrow & \downarrow & \\ 1000 & 0111 & 0100 & \text{(BCD)} \end{array}$$

- Reverse the process to convert BCD to decimal.

$$\begin{array}{cccc} 9 & 4 & 3 & \text{(decimal)} \\ \downarrow & \downarrow & \downarrow & \\ 1001 & 0100 & 0011 & \text{(BCD)} \end{array}$$

2-4 BCD Code

- Convert 0110100000111001 (BCD) to its decimal equivalent.

$$\underbrace{0110}_{6}\ \underbrace{1000}_{8}\ \underbrace{0011}_{3}\ \underbrace{1001}_{9}$$

Divide the BCD number into four-bit groups and convert each to decimal.

2-4 BCD Code

- Convert 0110100000111001 (BCD) to its decimal equivalent.

$$\underbrace{0110}_{6}\ \underbrace{1000}_{8}\ \underbrace{0011}_{3}\ \underbrace{1001}_{9}$$

Divide the BCD number into four-bit groups and convert each to decimal.

2-4 BCD Code

- Convert BCD 011111000001 to its decimal equivalent.

$$\underbrace{0111}_{7}\ \underset{\downarrow}{1100}\ \underbrace{0001}_{1}$$

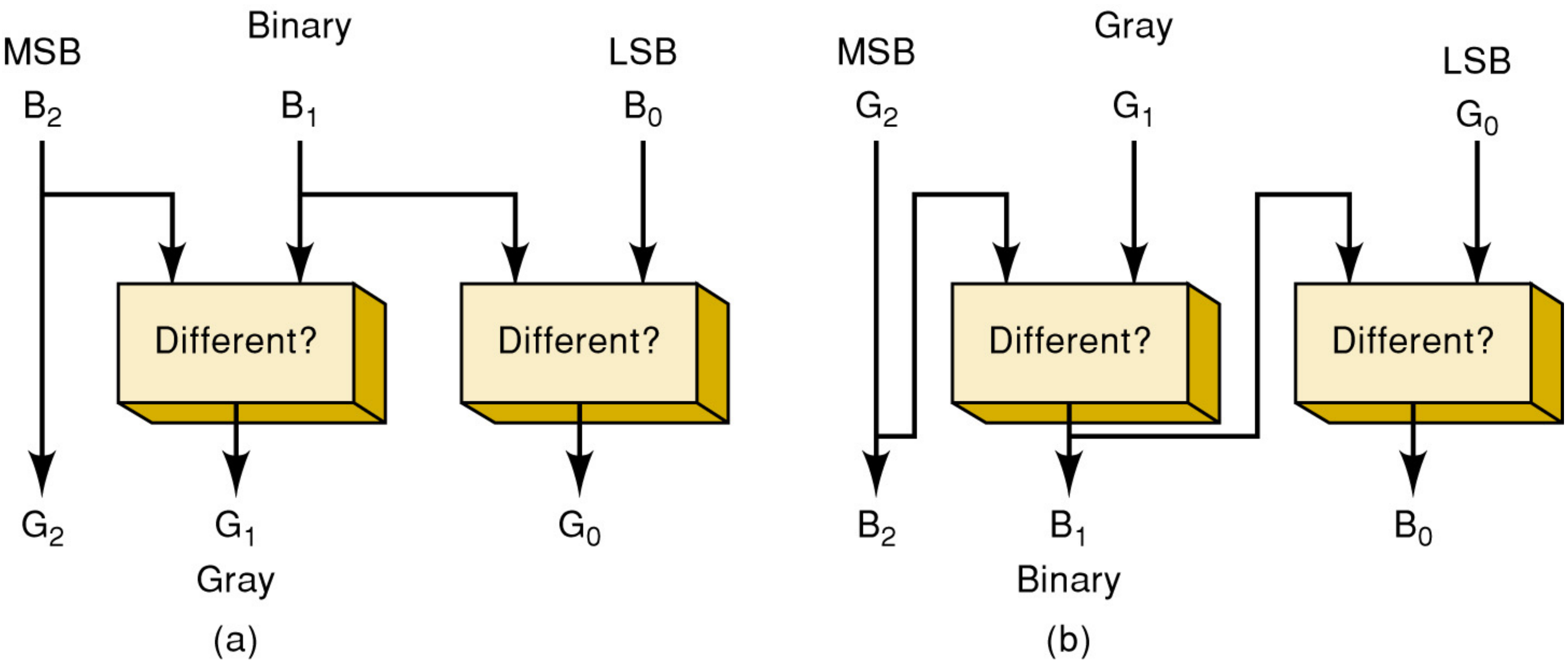
The forbidden group represents an error in the BCD number.

- The Gray code is used in applications where numbers change rapidly.
  - Only one bit changes from each value to the next.

Three bit binary and Gray code equivalents.

B_2	B_1	B_0	G_2	G_1	G_0
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

2-5 The Gray Code



Binary to Gray **Gray to Binary**

Conversion Algorithm

- From binary to Gray:

Let B[n:0] be the input array of bits in the usual binary representation, [0] being LSB
Let G[n:0] be the output array of bits in Gray code
G[n] = B[n]
for i = n-1 downto 0 G[i] = B[i+1] XOR B[i]

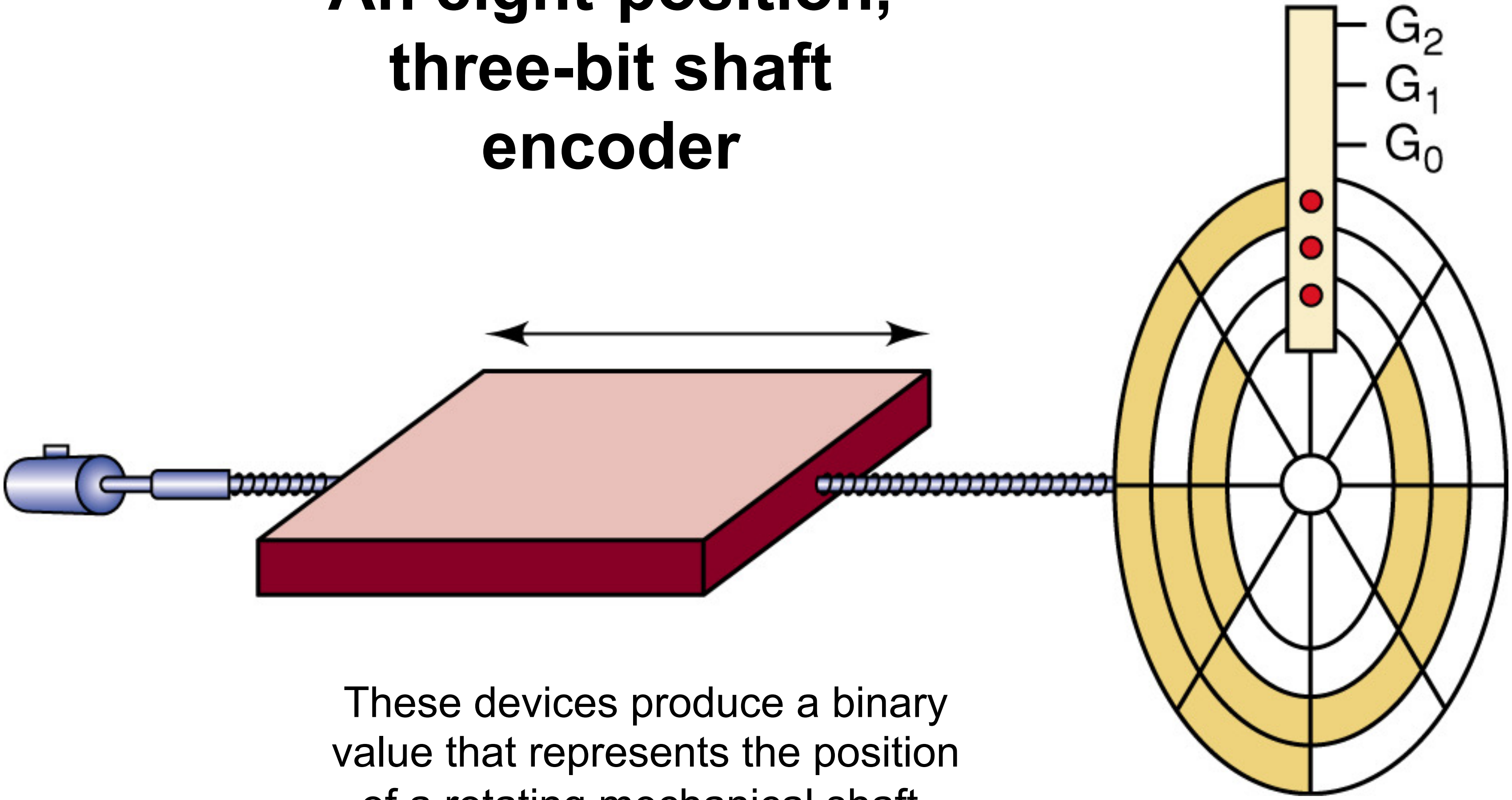
- From Gray to binary:

Let G[n:0] be the input array of bits in Gray code
Let B[n:0] be the output array of bits in the usual binary representation
B[n] = G[n]
for i = n-1 downto 0 B[i] = B[i+1] XOR G[i]

- Note: A XOR B = A'B+AB'

2-5 The Gray Code

An eight-position, three-bit shaft encoder



These devices produce a binary value that represents the position of a rotating mechanical shaft.

Decimal numbers 1 – 15 in binary, hex, BCD, Gray

Decimal	Binary	Hexadecimal	BCD	GRAY
0	0	0	0000	0000
1	1	1	0001	0001
2	10	2	0010	0011
3	11	3	0011	0010
4	100	4	0100	0110
5	101	5	0101	0111
6	110	6	0110	0101
7	111	7	0111	0100
8	1000	8	1000	1100
9	1001	9	1001	1101
10	1010	A	0001 0000	1111
11	1011	B	0001 0001	1110
12	1100	C	0001 0010	1010
13	1101	D	0001 0011	1011
14	1110	E	0001 0100	1001
15	1111	F	0001 0101	1000

2-7 The Byte, Nibble, and Word

- Most microcomputers handle and store binary data and information in groups of eight bits.
  - 8 bits = 1 byte.
    - A byte can represent numerous types of data/information.
- Binary numbers are often broken into groups of four bits.
  - Because a group of four bits is half as big as a byte, it was named a **nibble**.
- A **word** is a group of bits that represents a certain unit of information.
  - **Word size** can be defined as the number of bits in the binary word a digital system operates on.
    - PC word size is eight bytes (64 bits).

2-8 Alphanumeric Codes

- Represents characters and functions found on a computer keyboard.
  - 26 lowercase & 26 uppercase letters, 10 digits, 7 punctuation marks, 20 to 40 other characters.
- ASCII – American Standard Code for Information Interchange.
  - Seven bit code: $2^7 = 128$ possible code groups
  - Examples of use: transfer information between computers; computers & printers; internal storage.

2-8 Alphanumeric Codes

ASCII – American Standard Code for Information Interchange

Character	HEX	Decimal	Character	HEX	Decimal	Character	HEX	Decimal	Character	HEX	Decimal
NUL (null)	0	0	Space	20	32	@	40	64	.	60	96
Start Heading	1	1	!	21	33	A	41	65	a	61	97
Start Text	2	2	“	22	34	B	42	66	b	62	98
End Text	3	3	#	23	35	C	43	67	c	63	99
End Transmit.	4	4	$	24	36	D	44	68	d	64	100
Enquiry	5	5	%	25	37	E	45	69	e	65	101
Acknowlege	6	6	&	26	38	F	46	70	f	66	102
Bell	7	7	`	27	39	G	47	71	g	67	103
Backspace	8	8	(	28	40	H	48	72	h	68	104
Horiz. Tab	9	9	)	29	41	I	49	73	i	69	105
Line Feed	A	10	*	2A	42	J	4A	74	j	6A	106
Vert. Tab	B	11	+	2B	43	K	4B	75	k	6B	107
Form Feed	C	12	,	2C	44	L	4C	76	l	6C	108
Carriage Return	D	13	-	2D	45	M	4D	77	m	6D	109
Shift Out	E	14	.	2E	46	N	4E	78	n	6E	110

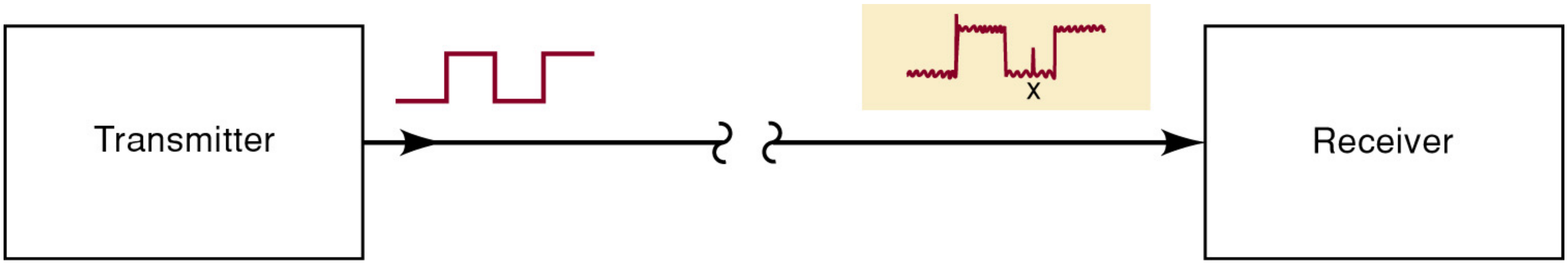
See the entire table on page 49 of your textbook.

2-9 Parity Method for Error Detection

- Binary data and codes are frequently moved between locations:
  - Digitized voice over a microwave link.
  - Storage/retrieval of data from magnetic/optical disks.
  - Communication between computer systems over telephone lines, using a modem.

2-9 Parity Method for Error Detection

- Electrical noise can cause errors during transmission.
  - Spurious fluctuations in voltage or current present in all electronic systems.



- Many digital systems employ methods for error detection—and sometimes correction.
  - One of the simplest and most widely used schemes for error detection is the **parity method**.

2-9 Parity Method for Error Detection

- The parity method of error detection requires the addition of an extra bit to a code group.
  - Called the parity bit, it can be either a 0 or 1, depending on the number of 1s in the code group.
- There are two parity methods, even and odd.
  - The transmitter and receiver must "agree" on the type of parity checking used.
    - Even seems to be used more often.

- Even parity method—the total number of bits in a group *including* the parity bit must add up to an *even* number.
  - The binary group **1 0 1 1** would require the addition of a parity bit **1**, making the group **1 1 0 1 1**.
    - The parity bit may be added at either end of a group.

$\boxed{1}$ 1 0 0 0 0 1 1

↑ added parity bit*

2-9 Parity Method for Error Detection

- Odd parity method—the total number of bits in a group *including* the parity bit must add up to an *odd* number.
  - The binary group **1 1 1 1** would require the addition of a parity bit **1**, making the group **1 1 1 1 1**.

The parity bit becomes a part of the code word.

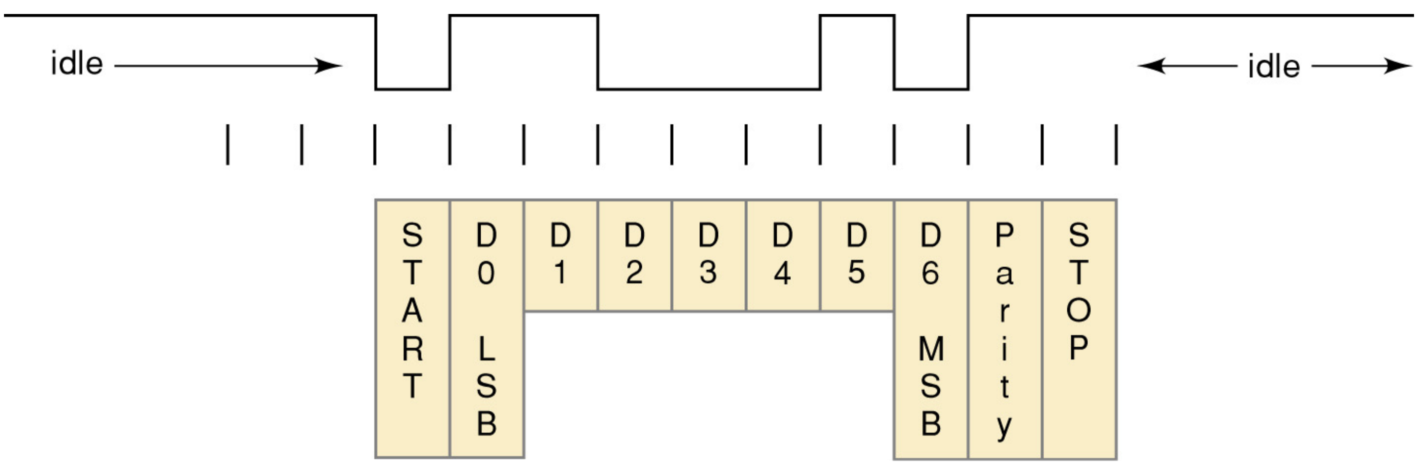
Adding a parity bit to the seven-bit ASCII code produces an eight-bit code.

2-10 Applications

- When ASCII characters are transmitted there must be a way to tell the receiver a new character is coming.
  - There is often a need to detect errors in the transmission as well.
- The method of transfer is called asynchronous data communication.

2-10 Applications

- An ASCII character must be "framed" so the receiver knows where the data begins and ends.
  - The first bit must always be a start bit (logic 0).
- ASCII code is sent LSB first and MSB last.
  - After the MSB, a parity bit is appended to check for transmission errors.
  - Transmission is ended by sending a stop bit (logic 1).



END

ELEVENTH EDITION

Digital Systems

Principles and Applications

Ronald J. Tocci
Monroe Community College

Neal S. Widmer
Purdue University

Gregory L. Moss
Purdue University

PEARSON