DIGITAL LOGIC DESIGN
Course Code: 328356(28)

Mr. Manjeet Singh Sonwani
Assistant Professor
Department of Electronics & Telecomm.
Government Engineering College Raipur

UNIT-I NUMBER SYSTEMS, CODES AND BOOLEAN ALGEBRA

- Codes: Weighted and Non-Weighted Codes, Sequential Codes, Self-Complementing Codes, Cyclic Codes;
- The 8421 BCD Code: BCD Addition; Excess-3 Code;
- The Gray Code: Binary to Gray and Gray to Binary Code Conversion;
- Error Detecting Codes: Parity, Check Sums,
- Block Parity, Five-bit Codes, The Biquinary Code, The Ring Counter Code;
- Error Correcting Code:7-bit Hamming code;
- Alphanumeric Codes: The ASCII Code, The EBCDIC Code.

Binary Codes

- Need of Coding: Information sent over a noisy channel is likley to be distorted.
- Information is coded to facilitate-
- 1.Efficient transmission 2.Error detection
- 3. Error correction
- Coding/Encoding: coding is the process of altering the characteristics of information to make it more suitable for intended application.
- Code: A set of n-bit strings in which different bit strings represent different numbers or other things is called a code.
- Code Word: A particular combination of n bit-values is called a code word.

Binary Codes

- Decoding: decoding is the process of reconstructing source information from the received encoded information.
- Decoding can be more complex than coding if there is no prior knowledge of coding schemes.
- Bit Combinations
- Bit: A binary digit o or 1.
- Nibble: A group of four bits
- Byte: A group of 8 bits
- Word: A group of 16 bits

Return Next

Classification of Binary Codes

- Alphanumeric code : It represents the alphanumeric information which is the combination of letters of the alphabet and decimal numbers as a sequence of 0s and 1s .
- Numeric Codes- : It represents the numeric information i.e only numbers as a series of 0s and 1s . Numeric codes used to represent decimal digits are called BCD codes.
- Binary Coded Decimal Codes:In a BCD code the each digit of a decimal no. is encoded into groups of 4 binary digits.
- A sequence of binary bits which represent a decimal digit is called a code word

Binary Coded Decimal code

- Disadvantages :
- 1. BCD codes requires more hardware
- 2.It slows down the system
- 3. There can be **30×10^{10}($^{10}C_{10}$.10!)** possible codes.
- 4. Most of these codes will not have any special properties.
- Natural BCD : $(16.85)_{10}$
- It is used in calculators
- $(16.85)_{10}=(00010110.10000101)_{NBCD}$
- Based on these properties a coding scheme may be selected: It should have ease of coding.
- 2.Ease of arithmetic operations
- 3.Minimum use of hardware
- 4.Erroe detection property
- 5.Ability to prevent wrong output during transistors.

Return Next

weighted code

- BCD code may be weighted or non-weighted
- It obeys position weighting principles. In a weighted code the decimal value of a code is the algebraic sum of the weights of 1s appearing in the number.
- Let $(A)_{10}$ be a decimal number encoded in the binary form as $a_3a_2a_1a_0$. Then $(A)_{10} = w_3a_3 + w_2a_2 + w_1a_1 + w_0a_0$
- where w_3 ,w_2 ,w_1 and w_0 are the weights selected for a given code, and $a_{3,}a_{2,}a_1$ and a_0 are either 0s or 1s.
- The more popularly used codes have the weights as

w_3	w_2	w_1	w_0
8	4	2	1
2	4	2	1
8	4	-2	-1

- Positively Weighted Code: All weights assigned to the binary bits are positive.1st weight must be 1,2nd weight must be 1 or 2 and sum of all weight must be equal to or greater than 9

The decimal numbers in positively Weighted Codes are:

Decimal digit	**Weights 8 4 2 1**	**Weights 2 4 2 1**	**Weights 5 2 1 1**	**Weights 5 4 2 1**
0	0000	0000	0000	0000
1	0001	0001	0001	0001
2	0010	0010	0011	0010
3	0011	0011	0101	0011
4	0100	0100	0111	0100
5	0101	1011	1000	1000
6	0110	1100	1010	1001
7	0111	1101	1100	1010
8	1000	1110	1110	1011
9	1001	1111	1111	1100

Negatively Weighted Code: some of weights assigned to the binary bits must be negative.

Type	Negative Weighted Code		Non-Weighted Code
Decimal digit	Weights 6 4 2 -3	Weights 8 4 -2 -1	New code XS-3
0	0000	0000	0011
1	0101	0111	0100
2	0010	0110	0101
3	1001	0101	0011
4	0100	0100	0111
5	1011	1011	1000
6	0110	1010	1001
7	1101	1001	1010
8	1010	1000	1011
9	1111	1111	1100

- Non-Weighted Code: NWCs are the codes which do not obey the position-weighting principle .
- Example:Excess-3(XS-3) and Gray Code.
- In all the cases only ten combinations are utilized to represent the decimal digits. The remaining six combinations are illegal. However, they may be utilized for error detection purposes.
- Sequential Codes: each succeeding code word in one binary number is greater than its preceding code word.
- Example:Excess-3(XS-3) and 8421.
- Non-Sequential Codes : Ex: 5211,2421 and 642-3

1.Self-Complementing Codes

- There are many possible weights to write a number in BCD code. Some codes have desirable properties, which make them suitable for specific applications. Two such desirable properties are:
- 1. Self-complementing codes: Logical complement of a coded no. is also its arithmetic complement.
- A code is said to be self – complementing ,if code word of the 9's complement of N [i.e (9-N)] can be obtained from the code word of N by interchanging all the 0's to 1's .
- In self – complementing code ,the code for 9 is 0,8 for 1,7 for 1, 6 for 3,5 for 4.
- Example: a.There are 4 positively weighted self – complementing codes 2421,5211,3321,4311
- b.There are 13 negatively weighted self – complementing codes 642-3,84-2-1   c. XS-3 code
- Necessary condition: Sum of its weights should be 9.

SC codes

- When we perform arithmetic operations, it is often required to take the "complement" of a given number. If the logical complement of a coded number is also its arithmetic complement, it will be convenient from hardware point of view. In a self-complementing coded decimal number, (N)10, if the individual bits of a number are complemented it will result in (9 - N)10.
- Example: Consider the 2421 code.
- The 2421 code of (4)10 is 0100.
- Its complement is 1011 which is 2421 code for (5)10 = (9 - 4)10.
- Therefore, 2421 code may be considered as a self-complementing code. A necessary condition for a self-complimenting code is that the sum of its weights should be 9.
- A self-complementing code, which is not weighted, is excess-3 code. It is derived from 8421 code by adding 0011 to all the 8421 coded numbers.
- Another self-complementing code is 631-1 weighted code.

Cyclic codes/Unit Distance Code

- In this code each successive code word differ from the preceding one in only one bit position.
- Adjecent codes differ only one bit.
- They are also called unit distance codes.
- Ex:- Gray code-It is often used for translating an analog quantity such as shaft position into a digital form.
- Application:- For error detection and correction .
-

Cyclic codes/Unit Distance Code

Decimal Digit	UDC-1	UDC-2	UDC-3
0	0000	0000	0000
1	0100	0001	1000
2	1100	0011	1001
3	1000	0010	0001
4	1001	0110	0011
5	1011	1110	0111
6	1111	1111	1111
7	0111	1101	1011
8	0011	1100	1010
9	0001	0100	0010

2. Reflective codes

- Imaged about the center entries with one bit changed.
- Ex:- 9's complement of a reflected code word is formed by changing only one of its bits.

Decimal Digit	CODE-A	CODE-B
0	0000	0100
1	0001	1010
2	0010	1000
3	0011	1110
4	0100	0000
5	1100	0001
6	1011	1111
7	1010	1001
8	1001	1011
9	1000	0101

- BCD: Binary-coded decimal (BCD) which encodes the digits 0 through 9 by their 4 -bit unsigned binary representations

2.3 Binary Codes for Decimal Numbers

DD	8421	2421	Ex-3	Biquinary	1-out-of-10
0	0000	0000	0011	0100001	1000000000
1	0001	0001	0100	0100010	0100000000
2	0010	0010	0101	0100100	0010000000
3	0011	0011	0110	0101000	0001000000
4	0100	0100	0111	0110000	0000100000
5	0101	1011	1000	1000001	0000010000
6	0110	1100	1001	1000010	0000001000
7	0111	1101	1010	1000100	0000000100
8	1000	1110	1011	1001000	0000000010
9	1001	1111	1100	1010000	0000000001

Return Back Next

2.3 Binary Codes for Decimal Numbers

Unused code words (Pseudo Code)					
	1010	0101	0000	0000000	0000000000
	1011	0110	0001	0000001	0000000011
	1100	0111	0010	0000010	0000000101
	1101	1000	1101	0000011	0000000110
	1110	1001	1110	0000101	0000000111
	1111	1010	1111	……	……

2.3 Binary Codes for Decimal Numbers

- Weighted Code: 8421 code and 2421 code are weighted codes. The weights for the BCD bits are 8,4,2,1 and 2,4,2,1.

- Examples:

1. $(1001)_{8421BCD}=(\quad ? \quad)_{10}$

$(1001)_{8421BCD}=1\times8+0\times4+0\times2+1\times1=(9)_{10}$

2. $(1011)_{2421BCD}=(\quad ? \quad)_{10}$

$(1011)_{2421BCD}=1\times2+0\times4+1\times2+1\times1=(5)_{10}$

2.3 Binary Codes for Decimal Numbers

- Self-Complementing Code: The 2421 code and the excess-3 have the advantage that they are self-complementing. But excess-3 code is not weighted.

- Example:

  What is the 9s' complement code of $(1011)_{2421}$? Is the complement code a 2421 code word?

  $(1011)_{242\ \text{complement}} = (0100)$, it's a 2421 code word for decimal digit 4.

Return Back Next

2.3 Binary Codes for Decimal Numbers

- The biquinary code and 1-out-of-10 code use more than the minimum number of bits in a code. The potential advantage is an error-detecting property.

- Example:

  Any errors can be detected in the following codes?

  $(1011000)_{\text{biquinary}}$, $(0001000)_{\text{biquinary}}$, $(1010000000)_{\text{1-out-of-10}}$

Return Back Next

2.3 Binary Codes for Decimal Numbers

- Packed- BCD Representation: Each decimal digit is represented four bits. So one 8-bit byte may represent the value from 0~99; 12 bits may represent the value from 0~999; and so on.

- Example:

$(10010100)_{8421}=(\ ?\ )_{10}$

$(10010100)_{8421}=(94)_{10}$

Return Back

Gray Code

- It is non-weighted/cyclic /Reflective code.
- It is not suitable for arithmetic operations.
- It is not a BCD Code.
- In a Gray code only one bit changes between each pair of successive code words.
- Gray code is a ***reflected code***. It can be defined recursively using the following rules:

1. A 1-bit Gray code has two code words, 0 and 1.
2. The first 2^n code words of an (n+1)-bit Gray code equal the code words of an n-bit gray code,written in order with a leading 0 appended.

2.4 Gray Code

3. The last 2^n code words of an (n+1)-bit Gray code equal the code words of an n-bit gray code, but written in reverse order with a leading 1 appended.

- Example: For a 2-bit Gray code, n=1.

1-bit Gray code

0
1

2-bit Gray code

0	0	↓
0	1	↓
1	1	↑
1	0	↑

00
01
11
10

Return Back Next

2.4 Gray Code

- Example: For a 3-bit Gray code, n=2.

2-bit Gray code

00
01
11
10

3-bit Gray code

0	00	000
0	01	001
0	11	011
0	10	010
1	10	110
1	11	111
1	01	101
1	00	100

Return Back

Gray Code-Applications

1.Gray codes can easily converted to and from binary.

2.Gray codes are used in instrumentation and data acquisition system where linear or angular displacement is measured.

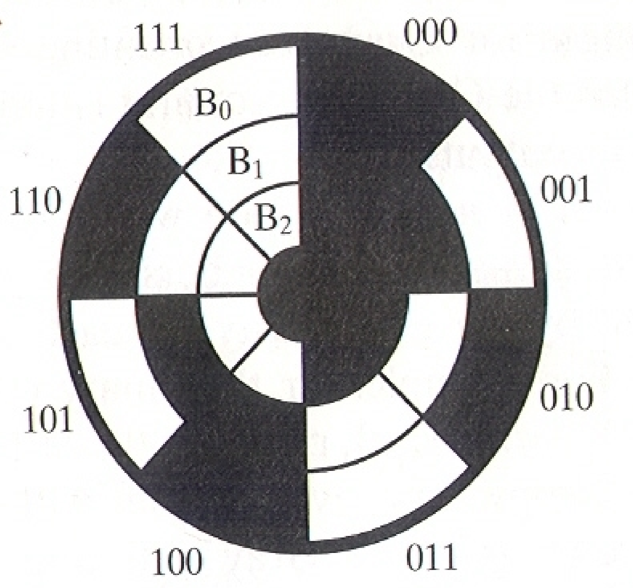
3.Gray codes are also used in shaft encoder, I/O devices,A/D converter and other peripheral equipments.

Reflection of Gray Codes

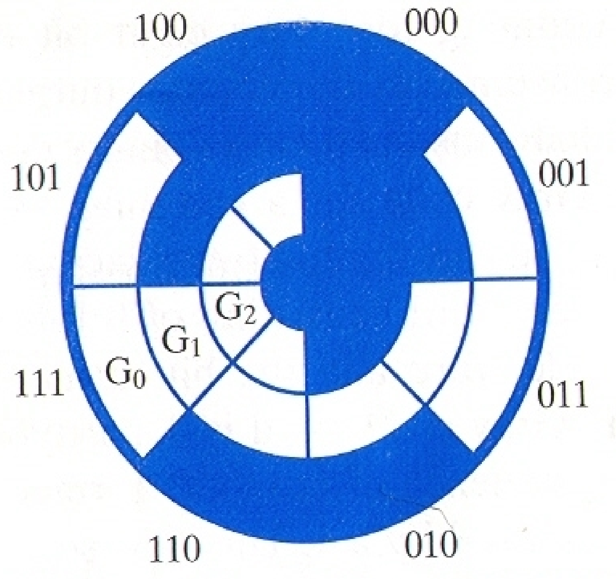
Digit	1 Bit Binary	1 Bit Gray	2 Bit Binary	2 Bit Gray	3 Bit Binary	3 Bit Gray	4 Bit Binary	4 Bit Gray
0	0	0	00	00	000	000	0000	0000
1	1	1	01	01	001	001	0001	0001
2			10	11	010	011	0010	0011
3			11	10	011	010	0011	0010
4					100	110	0100	0110
5					101	111	0101	0111
6					110	101	0110	0101
7					111	100	0111	0100
8							1000	1100
9							1001	1101
10							1010	1111
11							1011	1110
12							1100	1010
13							1101	1011

Position Indicator system: Shaft encoders

- A shaft encoder consists of a disk in which concentric circle have alternate sectors with reflective surfaces while the other sectors have non-reflective surfaces.



(a) Binary code for positions 0 through 7



(b) Gray code for positions 0 through 7

□ **FIGURE 1-5**
Optical Shaft-Angle Encoder

Binary to Gray Conversion

- If n bit binary number is represented by and it Gray code equivalent by where and are MSBs the gray code bits are obtained from the binary code as follows:

Binary to Gray Conversion Procedure:

- First Method:
- Write the MSB of binary number as the MSB of Gray code.
- Add the MSB of binary number to the next bit of binary,recording the sum and ignoring the carry,if any,i.e XOR the bits.This sum is the next bit of the Gray code.
- Add 2nd bit of binary number to the 3rd bit of binary, 3rd bit of binary number to the 4th bit of binary and so on.
- Write the successive sums as the successive bits of the Gray code untill all the bits of the binary no.s are exhaused/used.

Binary to Gray Conversion Procedure:

- Second Method:
- Exclusive OR the bits of binary number with those of the binary number shifted one position to the right.
- LSB of the shifted binary number is discarded and the MSB of Gray code no. is the same as the MSB of the original binary number.
- Ex:-Convert Binary 1001 to Gray code.
- Binary 1 0 0 1
- Gray 1 1 0 1

Gray to Binary Conversion

- If n bit Gray number is represented by and its binary code equivalent by where and are MSBs the binary code bits are obtained from the gray code as follows:

Gray to Binary Conversion Procedure:

- MSB of binary number is same as the MSB of Gray code number.
- Add the MSB of binary number to the next significant bit of Gray code,i.e XOR the bits.
- Write the sum and ignored the carry.
- Continue above steps till all the Gray bits are converted to the binary.

Gray to Binary Conversion Procedure:

- Ex:Convert Gray 1101 into Binary code.
- Gray 1 1 0 1
- Binary 1 0 0 1

Error Detecting Codes:

- When a binary data is transmitted and processed there is a possibility of distortion of its contents due to noise in channel.
- The 1s may get changed to 0s and 0s to 1s.
- In digital system error in transmitted data pose a serious problem ,So that whenever such an error occurs the concerned binary word can be corrected and retransmitted.
- Parity:-
- The simplest technique of error detecting is adding an extra bit ,known as the parity bit , to each word being transmitted.
- Types:-
- ODD PARITY: the parity bit is set to 0 or 1 at the transmitter such that the total no. of 1 bit in the word including the parity bit is an odd number.
- EVEN PARITY: the parity bit is set to 0 or 1 at the transmitter such that the total no. of 1 bit in the word including the parity bit is an even number.
- This parity check can always detect a single bit error but cannot detect two or more errors in the same word.

Error Detecting Codes:

- ODD and EVEN PARITY in 8421 BCD code:

Decimal Digit	8421 BCD	Odd Parity	Even Parity
0	0000	1	0
1	0001	0	1
2	0010	0	1
3	0011	1	0
4	0100	0	1
5	0101	1	0
6	0110	1	0
7	0111	0	1
8	1000	0	1
9	1001	1	0

Check sums:

- Simple parity check cannot detect two or more errors within the same word.
- As each word is transmitted , it is added to the sum of previously transmitted words and the sum retained at the transmitter end.
- At the end of the transmission,the sum (called check sum) up to that time is sent to the receiver.
- The receiver can check its sum with the transmitted sum.If two sums are same,then no errors were detected at the receiver end.
- If there is an errors ,the receiver can ask for retransmission of entire data.
- Application: Teleprocessing system

Block parity:

- When several binary words are transmitted or stored in succession,the resulting collection of bits can be regarded as a blocking of data,having rows and columns.
- Parity bits can then be assign to both rows and columns.This scheme makes it possible to correct any single bit error and to detect any two bit error in a word.
- This technique is widely used for data stored on magnetic taps.
- Example: six 8-bit succession can be formed into a 6x8 block for transmission.
- Parity bits are added to both row and column so that the block is transmitted as 7x9 block as shown in fig.A

Block parity:

	Table A Transmitted									
	0	1	0	1	1	0	1	1	0	Odd Parity
	1	0	0	1	0	1	0	1	1	
	0	1	1	0	1	1	1	0	0	
	1	1	0	1	0	0	1	1	0	
	1	0	0	0	1	1	0	1	1	
	0	1	1	1	0	1	1	1	1	
Parity Row	0	1	1	1	0	1	1	0	0	
									Parity Column	

Block parity:

	Table B Received					Error Bit			
	0	1	0	1	1	0	1	1	0	Odd Parity
	1	0	0	1	0	1	0	1	1	
3rd Row	0	1	1	0	0	1	1	0	0	
	1	1	0	1	0	0	1	1	0	
	1	0	0	0	1	1	0	1	1	
	0	1	1	1	0	1	1	1	1	
Parity Row	0	1	1	1	0	1	1	0	0	
					5th Column				Parity Column	

At the receiving end,parity is checked both row and column wise,suppose errors are detect and then corrected by complementing the error bit.

Block parity:

	Table C									
	0	1	0	1	1	0	1	1	0	Odd Parity
	1	0	0	1	0	1	0	1	1	
	0	1	1	0	1	1	1	0	0	
	1	0	0	0	0	0	1	1	0	
	1	0	0	0	1	1	0	1	1	
	0	1	1	1	0	1	1	1	1	
Parity Row	0	1	1	1	0	1	1	0	0	
		2nd Column		4th Column					Parity Column	

Here two errors are detected but not corrected,parity errors observed in both column 2 and 4,It indicate that in one row there are two error.

Five Bit Code:

- There are some 5-bit BCD code that have parity contained within each code for ease of error detection shown in table:

Decimal	63210	2 out of 5	Shift counter	51111
0	00110	00011	00000	00000
1	00011	00101	00001	00001
2	00101	00110	00011	00011
3	01001	01001	00111	00111
4	01010	01010	01111	01111
5	01100	01100	11111	10000
6	10001	10001	11110	11000
7	10010	10010	11100	11100
8	10100	10100	11000	11110
9	11000	11000	10000	11111

Biquinary Code:

- It is a 7-bit BCD code.It is a parity data code each code group regarded as consisting 2-bit subgroup and 5-bit subgroup .each of subgroups containes a single 1s so each code group has exactly two 1s .The weights of the bit positions are 5043210 .The biquinary code is used in Abacus,shown in table:

Decimal	5	0	4	3	2	1	0
0	0	1	0	0	0	0	1
1	0	1	0	0	0	1	0
2	0	1	0	0	1	0	0
3	0	1	0	1	0	0	0
4	0	1	1	0	0	0	0
5	1	0	0	0	0	0	1
6	1	0	0	0	0	1	0
7	1	0	0	0	1	0	0
8	1	0	0	1	0	0	0
9	1	0	1	0	0	0	0

Ring Counter Code:

- A 10-bit ring counter code produce a sequence of 10-bit group having the properly that each group has a single 1. It is a weighted because each bit weights equal to 1 of the 10 decimal digits. Although this code is inefficient (for 1024 encode in pure binary),it has excellent error detecting property to implement Ring counter, shown in table:

Decimal	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	0	1	0	0	0
4	0	0	0	0	0	1	0	0	0	0
5	0	0	0	0	1	0	0	0	0	0
6	0	0	0	1	0	0	0	0	0	0
7	0	0	1	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0	0	0

Error Correcting Code: Hamming Code

- Error Correcting Codes are used to convert the correct the errors present in the received data (bit stream). Parity code is also used for error Correction.
- To correct a single bit error the minimum distance of the code must be three.
- Error Correcting Codes must be locate the error bit(bit position) and by complementing it ,the message can be corrected.
- Hamming Code:It is useful for both detection and correction of error present in the received data.
- This code uses multiple parity bits **P_1, P_2 ,.......... P_k** located at positions **2^{k-1}**
- Such as 2^0, 2^1, 2^2.......So on.
- The minimum value of 'k' for which the following relation is correct(valid) is nothing but the required no. of parity bits.
- 2^{k-1}>=n+k+1
- Where 'n' is the no. of bits in the binary code "k" is the no. of parity bits.
- Therefore the no. of bits in the Hamming code is equal to (n+k) bit code word.

Hamming Code

- To Correct the Error ,k parity Check are performed on selected digits of each code word and position of error bit is located by farming an 'error word' and the error bit is then complemented.
- The k-bit error word is generated by putting 0 or 1in at $_2{}^{k-1}$ positions depending upon paroty type(Even or Odd parity).
- **For** even parity P=0 and for Odd P=1
- 7-bit Hamming Code
- All bit positions that are power of 2 are marked as parity bits
- ($_2{}^0$, $_2{}^1$, $_2{}^2$.......So on.) other bits are for data bits
- The word format of 7-bit HC would be as shown below:
- D_7 D_6 D_5 P_4 D_3 P_2 P_1
- For a (7,4) Hamming code,
- 1. the first even parity check should involve all the odd numbered locations
- 1,3,5,7 because these locations have a 1 in the least significant bit of their binary representations
- 2. the second even parity check has to involve locations 2,3,6,7 because these locations have a 1 in the next to least significant bit of their binary representations
- 3. the third even parity check has to involve locations 4,5,6,7
- - at least one of the bit in each set of locations is a parity bit which will be 0 or 1 in order to make the number of ones in the locations even.
- Here 1,2,4 are the parity bits and 3,5,6,7 are the information bits
- D_7 D_6 D_5 P_4 D_3 P_2 P_1

Alphanumeric Codes

- How do you handle alphanumeric data?
- Easy answer!
- Formulate a binary code to represent characters! ☺
- For the 26 letter of the alphabet would need 5 bit for representation.
- But what about the upper case and lower case, and the digits, and special characters

A code called ASCII

- ASCII stands for American Standard Code for Information Interchange
- The code uses 7 bits to encode 128 unique characters
- Reference the textbook, pg. 27, for a table of the ASCII code
- As a note, formally, work to create this code began in 1960. 1st standard in 1963. Last updated in 1986.

ASCII Code

- Represents the numbers
  - All start 011 xxxx and the xxxx is the BCD for the digit
- Represent the characters of the alphabet
  - Start with either 100, 101, 110, or 111
  - A few special characters are in this area
- Start with 010 – space and !”#$%&’()*+.-,/
- Start with 000 or 001 – control char like ESC

ASCII Example

- Encoding of 123
  - 011 0001   011 0010   011 0011
- Encoding of Joanne
  - 100 1010 110 1111   110 0001
  - 110 1110   110 1110   110 0101
- Note that these are 7 bit codes

What to do with the 8th Bit?

- In digital systems data is usually organized as bytes or 8 bit of data.
- How about using the 8th bit for an error coding. This would help during data transmission, etc.
- Parity bit – the extra bit included to make the total number of 1s in the byte either even or odd – called even parity and odd parity

Example of Parity

- Consider data 100 0001
  - Even Parity 0100 0001
  - Odd Parity 1100 0001
- Consider data 1010100
  - Even Parity 1101 0100
  - Odd Parity 0101 0100
- A parity code can be used for ASCII characters and any binary data.

Other Character Codes

- Once upon a time, a long, long time ago, there existed cards, called punch cards!
  - And a code for those cards called Hollerith code. (patented in 1889)
  - The code told you what character was being represented in a column when there was a punch out in various rows of that column.
- And another code for characters called EBCDIC (Extended Binary Coded Decimal Interchange Code) (1963, 1964 IBM) - similar to ASCII –
  - Digits are coded F0 through F9 in EBCDIC