DATA TRANSMISSION

1

Introduction

- Data into a computer (via a keyboard) is encoded by the electronics within the keyboard into an equivalent binary-coded pattern using standard coding schemes:
- Two most common schemes:
  - EBCDIC (Extended Binary Coded Decimal Interchange Code): 8-bit code
  - ASCII ((American Standards Committee for Information Interchange): 7-bit code

2

EBCDIC

- Extended Binary Coded Decimal Interchange Code
- 8-bit character code developed by IBM
  - used for data communication, processing and storage
  - extended earlier proprietary 6-bit BCD code
  - designed for backward compatibility or marketing?
  - still in use today on some mainframes and legacy systems.
- Allows for 256 different character representations (2^8)
  - includes upper and lower case
  - lots of special characters (non-printable)
  - lots of blank (non-used codes)
    - assigned to international characters in various versions
  - used with/without parity (block transmissions)

3

EBCDIC Code

Bit 4567 / 0123	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0000	NUL	SOH	STX	ETX	PF	HT	LC	DEL				VT	FF	CR	SO	SI
0001	DLE	DC1	DC2	DC3	RES	NL	BS	IL	CAN	EM			IFS	IGS	IRS	IUS
0010			FS		BYP	LF	EOB	PRE			SM			ENQ	ACK	BEL
0011			SYN		PN	RS	UC	EOT					DC4	NAK		SUB
0100	SP										¢	.	<	(	+	|
0101	&										!	$	*	)	;	¬
0110	-	/									¦	,	%	_	>	?
0111										\	:	#	@	'	=	"
1000		a	b	c	d	e	f	g	h	i						
1001		j	k	l	m	n	o	p	q	r						
1010		~	s	t	u	v	w	x	y	z						
1011																
1100	{	A	B	C	D	E	F	G	H	I						
1101	}	J	K	L	M	N	O	P	Q	R						
1110			S	T	U	V	W	X	Y	Z						
1111	0	1	2	3	4	5	6	7	8	9						□

Note: To read this chart, simply find the character on the chart, then look to the left side of the row for bits 0, 1, 2, and 3, and to the top of the column for bits 4, 5, 6, and 7. This is only one of many possible implementations of EBCDIC.

EBCDIC special characters

ACK	Acknowledgement	EOT	End of Transmission	PF	Punch Off
BEL	Bell	ETX	End of Text	PN	Punch On
BS	Backspace	FF	Form Feed	PRE	Prefix
BYP	Bypass	FS	File Separator	RES	Restore
CAN	Cancel	HT	Horizontal Tab	RS	Reader Stop
CR	Carriage Return	IFS	Information File Separator	SI	Shift In
DC1	Device Control 1	IGS	Information Group Separator	SM	Start Message
DC2	Device Control 2	IL	Idle	SO	Shift Out
DC3	Device Control 3	IRS	Information Record Separator	SOH	Start of Heading
DC4	Device Control 4	IUS	Information Unit Separator	SP	Space
DEL	Delete	LC	Lower Case	STX	Start of Text
DLE	Data Link Escape	LF	Line Feed	SUB	Substitute
EM	End of Medium	NAK	Negative Acknowledgement	SYN	Synchronous Idle
ENQ	Enquiry	NL	New Line	UC	Upper Case
EOB	End of Block	NUL	Null	VT	Vertical Tab

4

ASCII CODE

- 7-bit code developed by the American National Standards Institute (ANSI)
  - most popular data communication character code today
- Allows for 128 different character representations (2^7)
  - includes upper and lower case
  - lots of special characters (non-printable)
  - generally used with an added parity bit
  - better binary ordering of characters than EBCDIC
- Extended ASCII uses 8 data bits and no parity
  - Used for processing and storage of data
  - Allows for international characters
  - 8th bit stripped of for transmission of standard character set

5

ASCII Code

Bits 7654321	Character	Bits 7654321	Character	Bits 7654321	Character	Bits 7654321	Character
0000000	NUL	0100000	SP	1000000	@	1100000	`
0000001	SOH	0100001	!	1000001	A	1100001	a
0000010	STX	0100010	"	1000010	B	1100010	b
0000011	ETX	0100011	#	1000011	C	1100011	c
0000100	EOT	0100100	$	1000100	D	1100100	d
0000101	ENQ	0100101	%	1000101	E	1100101	e
0000110	ACK	0100110	&	1000110	F	1100110	f
0000111	BEL	0100111	'	1000111	G	1100111	g
0001000	BS	0101000	(	1001000	H	1101000	h
0001001	HT	0101001	)	1001001	I	1101001	i
0001010	LF	0101010	*	1001010	J	1101010	j
0001011	VT	0101011	+	1001011	K	1101011	k
0001100	FF	0101100	,	1001100	L	1101100	l
0001101	CR	0101101	-	1001101	M	1101101	m
0001110	SO	0101110	.	1001110	N	1101110	n
0001111	SI	0101111	/	1001111	O	1101111	o
0010000	DLE	0110000	0	1010000	P	1110000	p
0010001	DC1	0110001	1	1010001	Q	1110001	q
0010010	DC2	0110010	2	1010010	R	1110010	r
0010011	DC3	0110011	3	1010011	S	1110011	s
0010100	DC4	0110100	4	1010100	T	1110100	t
0010101	NAK	0110101	5	1010101	U	1110101	u
0010110	SYN	0110110	6	1010110	V	1110110	v
0010111	ETB	0110111	7	1010111	W	1110111	w
0011000	CAN	0111000	8	1011000	X	1111000	x
0011001	EM	0111001	9	1011001	Y	1111001	y
0011010	SUB	0111010	:	1011010	Z	1111010	z
0011011	ESC	0111011	;	1011011	[	1111011	{
0011100	FS	0111100	<	1011100	\	1111100	|
0011101	GS	0111101	=	1011101	]	1111101	}
0011110	RS	0111110	>	1011110	^	1111110	~
0011111	US	0111111	?	1011111	—	1111111	DEL

6

ASCII Non-Printable Codes

ASCII control characters			
BEL	Bell	EM	End of Medium
CAN	Cancel	ESC	Escape
DC1	Device Control 1	NUL	Null
DC2	Device Control 2	SI	Shift In
DC3	Device Control 3	SO	Shift Out
DC4	Device Control 4	SUB	Substitute
DEL	Delete		
Control codes			
ACK	Acknowledge	ETX	End of Text
DLE	Data Link Escape	NAK	Negative Acknowledge
ENQ	Enquiry	SOH	Start of Heading
EOT	End of Transmission	STX	Start of Text
ETB	End of Transmission Block	SYN	Synchronous Idle
Format effectors			
BS	Backspace	HT	Horizontal Tabulation
CR	Carriage Return	LF	Line Feed
FF	Form Feed	VT	Vertical Tabulation
Information separators			
FS	File Separator	RS	Record Separator
GS	Group Separator	US	Unit Separator

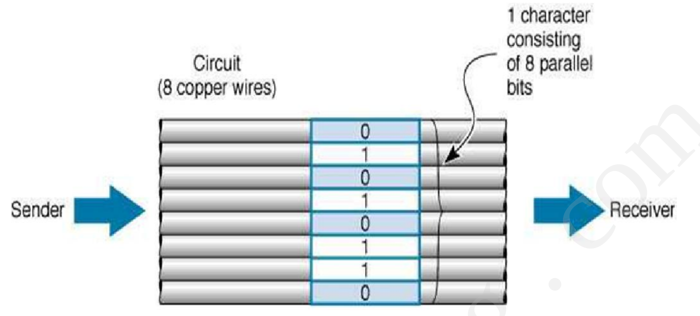
7

Bit Transmission

- Parallel mode: uses several wires, each wire sending one bit at the same time as the others.
- Serial mode: sends bit by bit over a single line
- Serial mode is slower than parallel, but can be used over longer distances because the bits stay in the order they were sent, while bits sent in parallel mode tend to spread out over long distances.

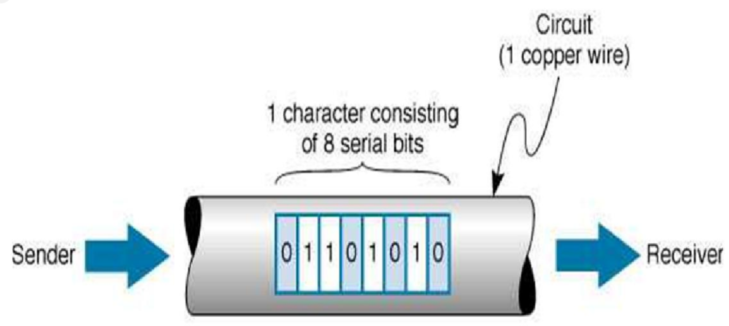
8

Parallel Transmission



9

Serial Transmission



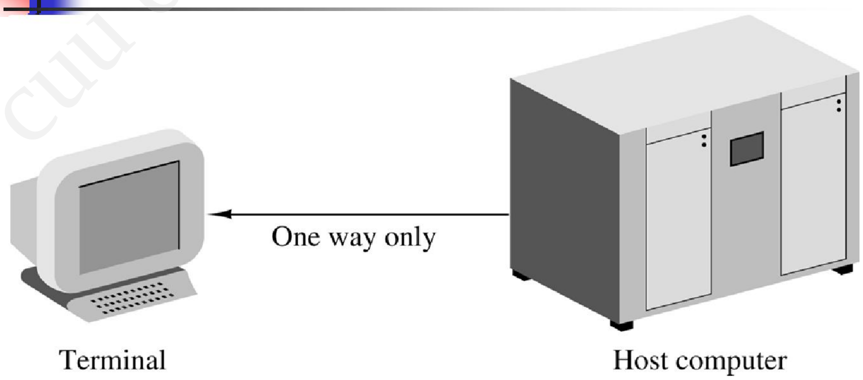
10

Communication Modes

- Simplex: Used when data is to be transmitted in one direction only
- Half-Duplex: Used when the two interconnected devices wish to exchange information alternately (only one device can transmit at a time)
- Full-Duplex: Used when data is to be exchanged between the two connected devices in both directions simultaneously

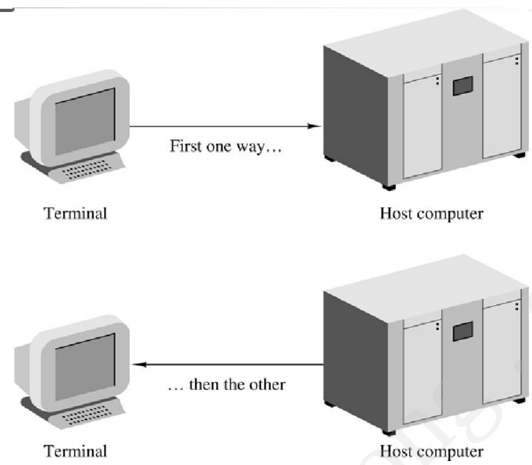
11

Simplex Communications



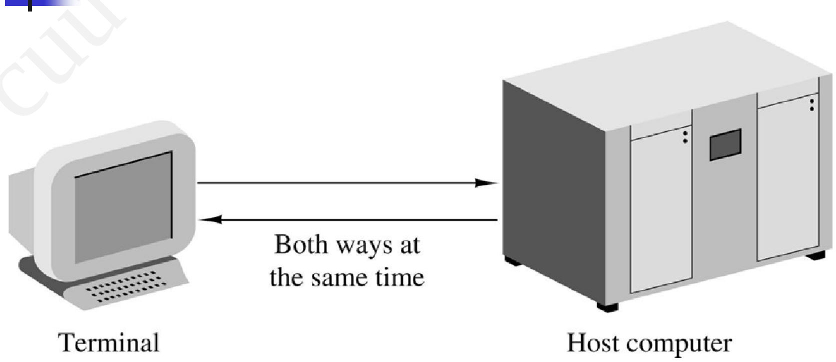
12

Half-Duplex Communications



13

Full-Duplex Communications



14

Transmission Modes

- For the receiver to decode and interpret bit pattern correctly, it must be able to determine
  - The start of each bit cell period → bit / clock synchronization
  - The start and end of each elements (character / byte) → character / byte synchronization
  - The start and end of each complete message block → frame synchronization

15

Transmission Modes

- Timing problems require a mechanism to synchronize the transmitter and receiver
- Two solutions
  - Asynchronous transmission: Transmitter and receiver clocks are independent
  - Synchronous transmission: Transmitter and receiver clocks are synchronized

16

Asynchronous Transmission

- Data transmitted on character at a time
  - 5 to 8 bits
- Timing only needs maintaining within each character
- Resynchronize with each character

17

Asynchronous Transmission

- In a steady stream, interval between characters is uniform (length of stop element)
- In idle state, receiver looks for transition 1 to 0
- Then samples next seven intervals (char length)
- Then looks for next 1 to 0 for next char

- Simple
- Cheap
- Overhead of 2 or 3 bits per char (~20%)
- Good for data with large gaps (keyboard)

18

Synchronous Transmission

- Block of data transmitted without start or stop bits
- Clocks must be synchronized
- Can use separate clock line
  - Good over short distances
  - Subject to impairments

19

Error Control and Flow Control

- Error control: schemes and procedures to detect and correct the occurrence of transmission errors
- Flow control: The control of the flow of information between two DTEs

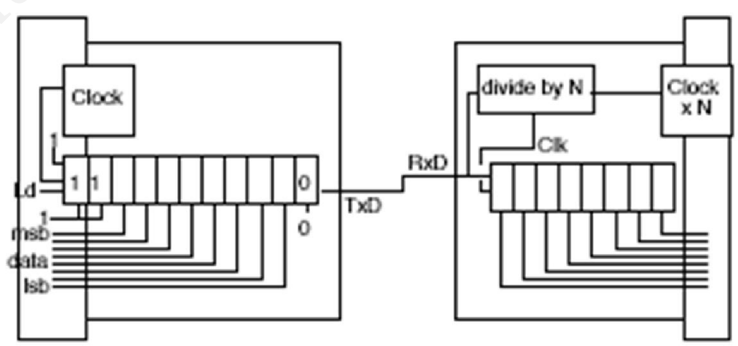
20

Asynchronous Transmission

- Transmission control circuits within DTE must perform the following functions:
  - Parallel-to-serial conversion (PISO) of each character / byte in preparation for its transmission
  - Serial-to-parallel conversion (SIPO) of each received character / byte in preparation for its storage and processing
  - A means for receiver to achieve bit, character and frame synchronization
  - The generation of suitable error check digits for error detection

21

Asynchronous Transmission



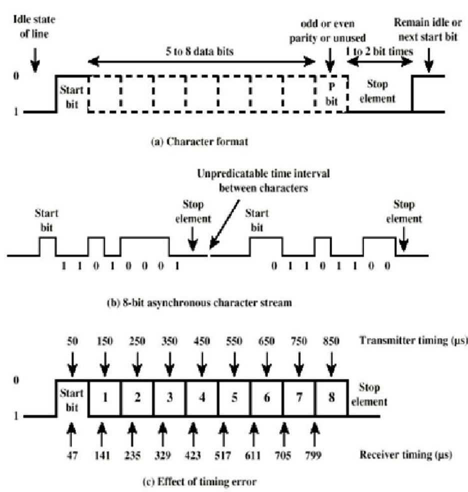
22

Asynchronous Transmission

- Bit synchronization
  - Start bit & stop bit
  - Receiver clock samples the incoming signal as near to the center of the bit cell as possible
- Character synchronization
  - Number of bits / character
- Frame synchronization
  - Control character: STX & EXT

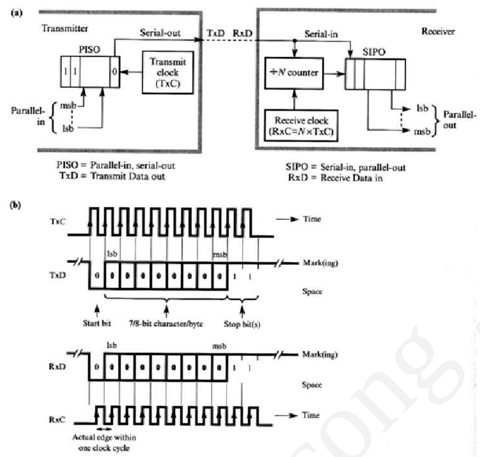
23

Asynchronous (Diagram)

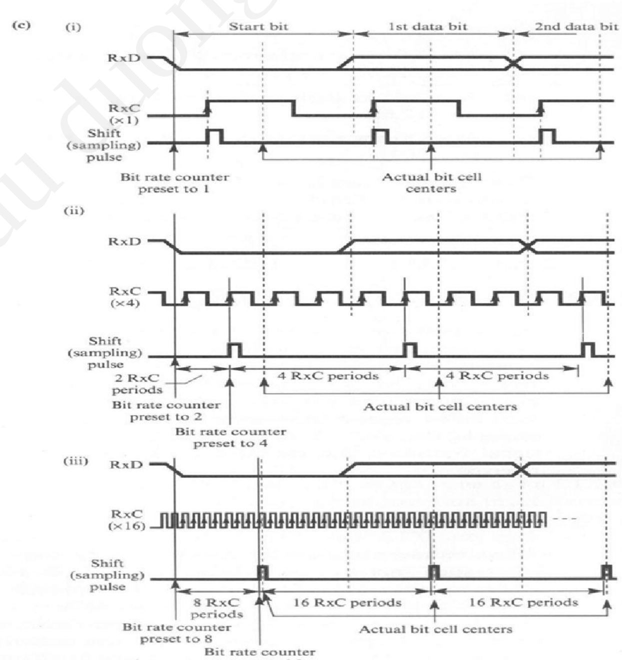


24

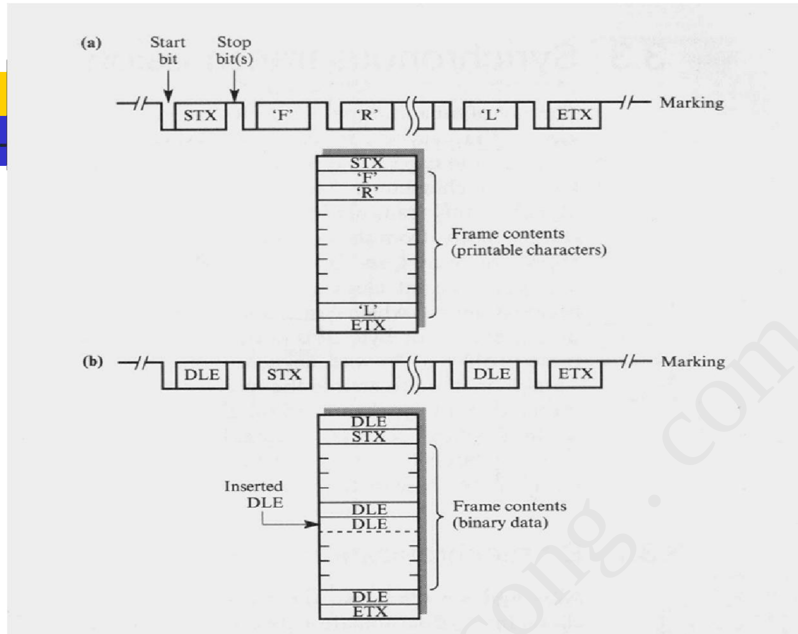
Asynchronous Transmission



25



26



27

Synchronous Transmission

- Block of data transmitted without start or stop bits
- Each frame is transmitted as a contiguous stream of binary digits
- Clocks must be synchronized

28

Synchronous Transmission

- Bit synchronization
  - Clock encoding and extraction
  - Digital Phase-lock-loop (DPLL)
  - Hybrid
- Frame synchronization
  - Character-oriented
  - Bit-oriented

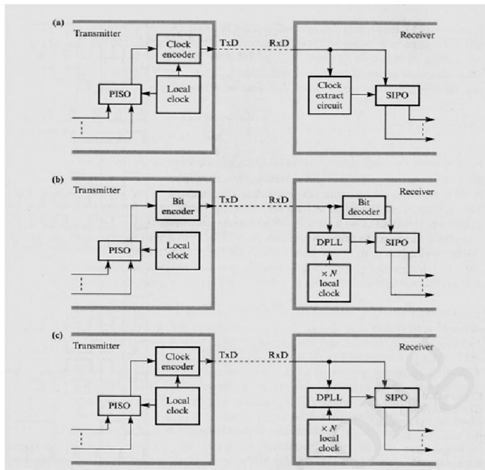
29

Bit Synchronization

- The receiver obtains (and maintains) bit synchronization in one of two ways
  - The clock (timing) information is embedded into the transmitted signal and subsequently extracted by the receiver → clock encoding and extraction
  - Receiver has a local lock (as with asynchronous transmission) but this time is kept in synchronism with the receiver signal by digital phase-lock-loop (DPLL) → DPLL
  - Hybrid schemes that exploit both methods → Hybrid

30

Synchronous Transmission

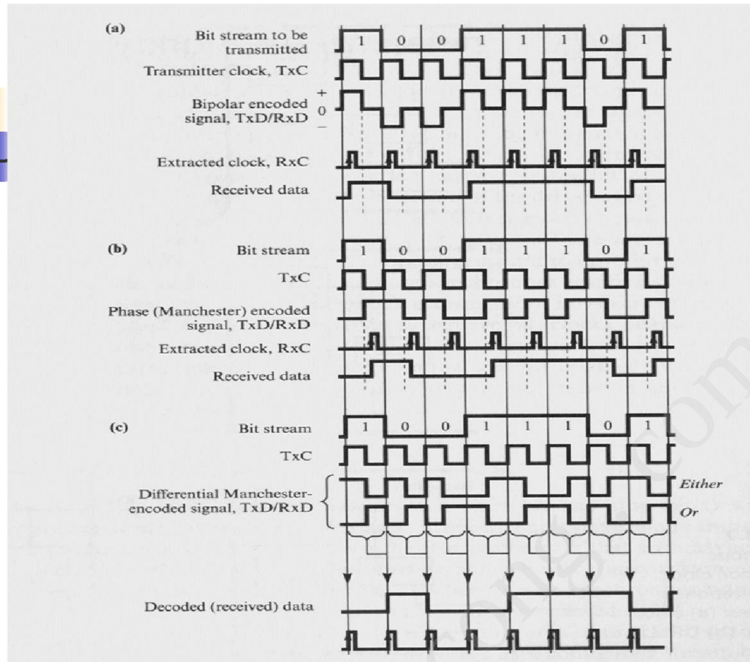


31

Bit Synchronization

- Bit synchronization
  - Clock encoding and extraction: embbeding timing (clock) information into a transmitted bit stream → RZ, Manchester, differential Manchester encoding

32



33

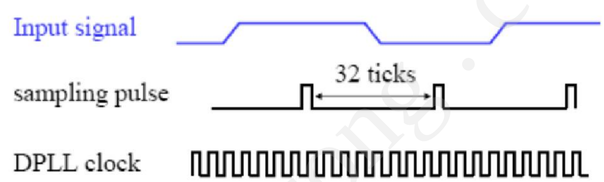
Synchronous Transmission

- Bit synchronization
  - DPLL: encoding the information in such a way that there are always sufficient bit transition (1 → 0 or 0 → 1) in transmitted waveform to enable the receiver clock to be resynchronized at frequent intervals
    - passing the data to be transmitted through a scrambler which randomizes the transmitted bit stream, removing contiguous strings of 1s or 0s
    - Encoding data in such a way that suitable transitions will always be present

34

Synchronous Transmission

- Bit synchronization
  - DPLL
    - DPLL' clock is 32 times the bit rate used on the data link
    - If there is no transition on the line, DPLL simply generates a sampling pulse every 32 clock periods after the previous one



35

Synchronous Transmission

- Bit synchronization
  - DPLL
    - When a transition is detected, the time interval between the previously generated sampling pulse and the next is determined according to the position of the transition relative to where DPLL thought it should be occur

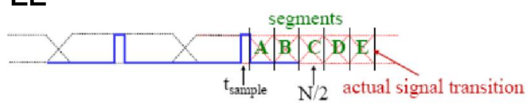
$$\text{Drift} = t_{actual\text{-}transition} - (t_{sample} + N/2)$$

    - Adjust to the next sampling pulse depending on the amount of drift

36

Synchronous Transmission

- Bit synchronization
  - DPLL



t_{sample}	t_{actual}	Δ (N/2 = 16)	drift	Segment	sample	adjustment
40	60	20	+4	E	too early	+2
40	56	16	0	C	OK	0
40	50	10	-6	A	too late	-2

segment of actual signal transition	next anticipated sampling pulse	sampling adjustment	actual next sample
A	too late	- 2 ticks	30
B	late	-1 tick	31
C	OK	none	32
D	early	+1 tick	33
E	too early	+2 ticks	34

37

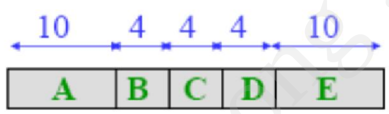
Synchronous Transmission

- Bit synchronization
  - DPLL
    - To achieve this, each bit period is divided into five segments, shown as A, B, C, D and E.
      - A transition occurring in segment A and E indicates that the previous sampling pulse was too early relative to the transition → typical division A = E = 10
      - Transitions in segments B and E are clearly nearer to the assumed transition and hence the relative adjustment are less → typical division B = D = 4
      - A transition in segment C is deemed to be close enough to the assumed transition to warranty no adjusment → typical division C = 4

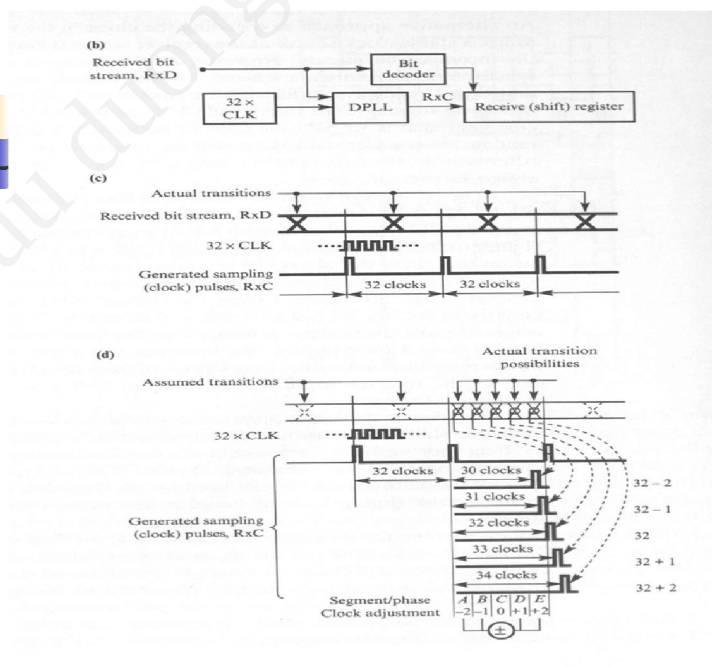
38

Synchronous Transmission

- Bit synchronization
  - DPLL
    - In the worst case, DPLL requires 10 bit transitions to converge to the nominal bit center of a waveform: 5 bit periods of coarse adjustment (±2), 5 bit periods of fine adjustment (±1)



39



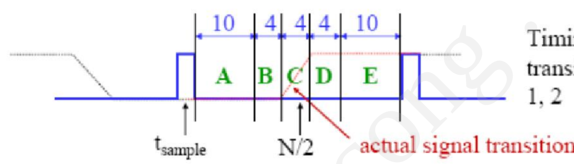
40

Synchronous Transmission

- Bit synchronization
  - DPLL

Examples of synchronization on signal transitions

	t_{sample}	t_{actual}	Δ (N/2 = 16)	drift	Segment	Sample	Adjustment
1.	0	16	16	0	C	OK	0
2.	32	47	15	-1	C	OK	0
3.	64	78	14	-2	B	too late	-1



Timing of transitions 1, 2

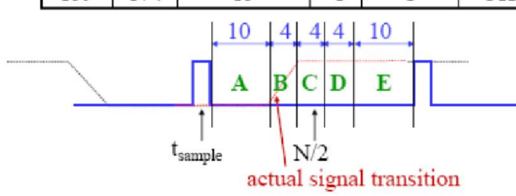
7 41

Synchronous Transmission

- Bit synchronization
  - DPLL

Examples of synchronization on signal transitions

	t_{sample}	t_{actual}	Δ (N/2 = 16)	drift	Segment	Sample	Adjustment
3.	64	78	14	-2	B	too late	-1
4.	95	110	15	-1	C	OK	0
5.	127	142	15	-1	C	OK	0
6.	159	174	15	-1	C	OK	0



Timing of transition 3

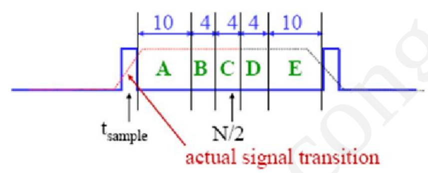
42

Synchronous Transmission

- Bit synchronization
  - DPLL

Starting from worst possible situation:
drift = ±16 DPLL clock cycles

	t_{sample}	t_{actual}	Δ (N/2 = 16)	drift	Segment	Sample	Adjustment
1.	0	0	0	-16	A	too late	-2
2.	30	32	2	-14	A	too late	-2
3.	60	64	4	-12	A	too late	-2



Timing of transition 1

9

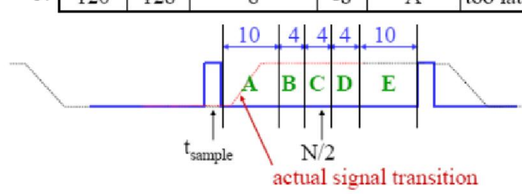
43

Synchronous Transmission

- Bit synchronization
  - DPLL

Started from worst possible situation:
drift = ±16 DPLL clock cycles

	t_{sample}	t_{actual}	Δ (N/2 = 16)	drift	Segment	Sample	Adjustment
2.	30	32	2	-14	A	too late	-2
3.	60	64	4	-12	A	too late	-2
4.	90	96	6	-10	A	too late	-2
5.	120	128	8	-8	A	too late	-2



Timing of transitions 2, 3, 4, 5

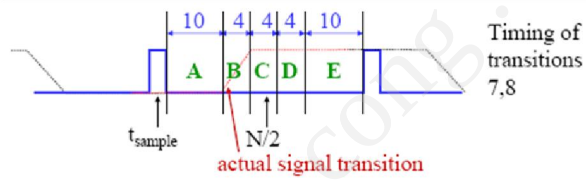
44

Synchronous Transmission

- Bit synchronization
  - DPLL

Started from worst possible situation:
drift = ±16 DPLL clock cycles

	t_{sample}	t_{actual}	Δ (N/2 = 16)	drift	Segment	Sample	Adjustment
5.	120	128	8	-8	A	too late	-2
6.	150	160	10	-6	A	too late	-2
7.	180	192	12	-4	B	too late	-1
8.	211	224	13	-3	B	too late	-1

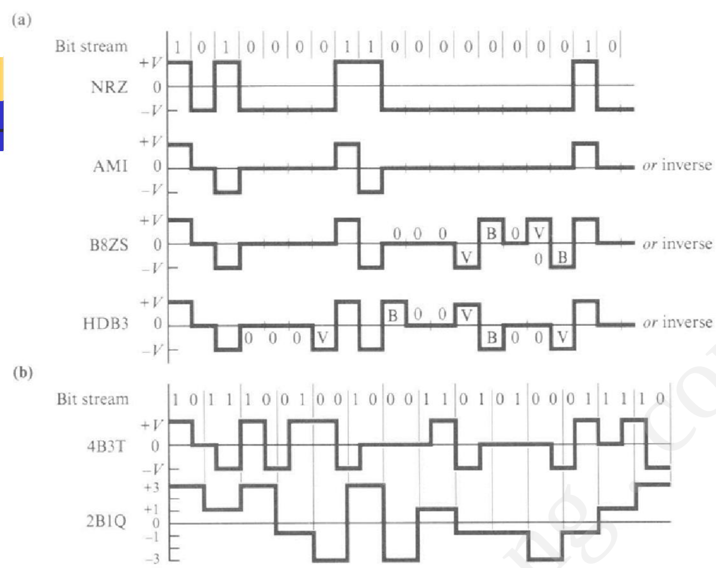


45

Synchronous Transmision

- Bit synchronization
  - DPLL
    - Line Coding: NRZ, AMI, HDB3, B3ZS, B6ZS, B8ZS, 4B3T, 2B1Q
      - Mean (DC) level of the transmitted signal is zero
      - Bandwidth
      - Bit rate
      - Error detection

46



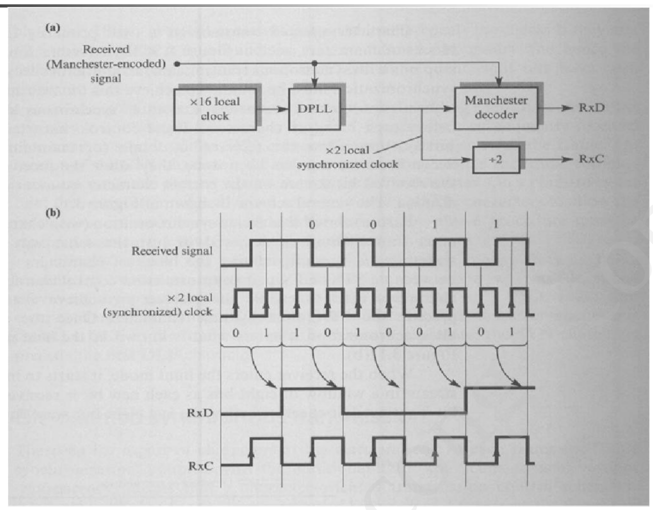
47

Synchronous Transmission

- Bit synchronization
  - Hybrid schemes: is used as the bit rate increases
    - Uses a combination of Manchester encoding and a DPLL

48

Bit Synchronization



49

Frame Synchronization

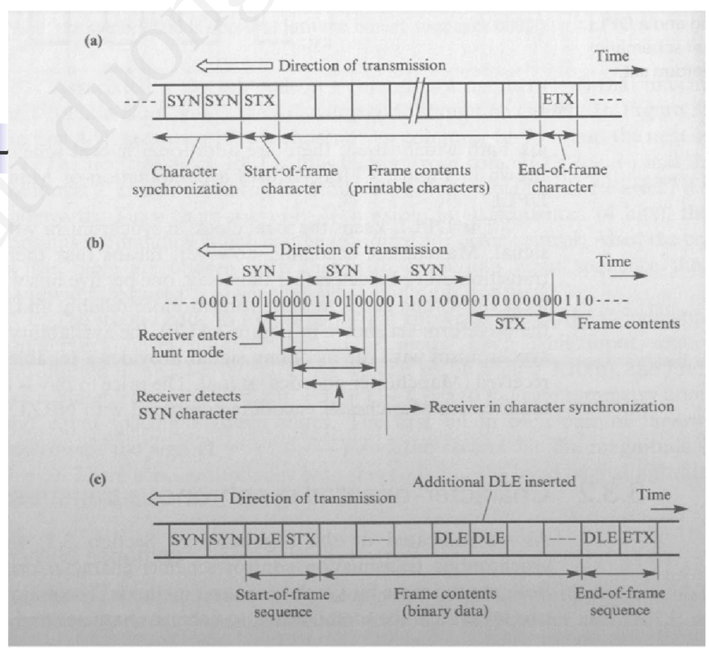
- 2 types of synchronous transmission control scheme:
  - Character-oriented: is used primarily for the transmission of blocks of characters
  - Bit-oriented: is used for the transmission of frames comprising either printable characters or binary data

50

Frame Synchronization

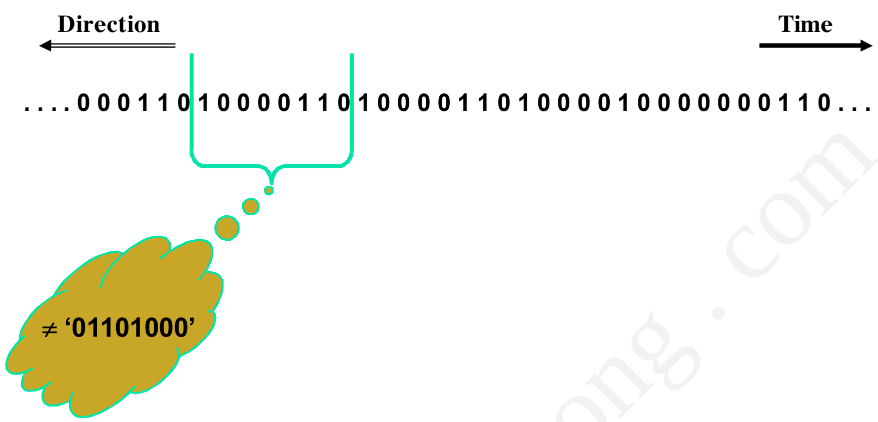
- Character-oriented:
  - Transmitter adds two or more transmission control characters, known as synchronous idle or SYN characters before block of characters
  - SYN characters have 2 functions:
    - Bit synchronization
    - Character synchronization

51



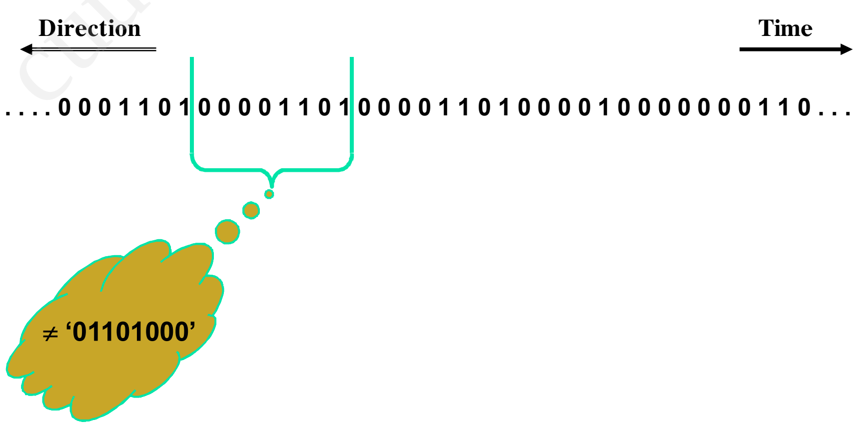
52

Character Oriented - Example



53

Character Oriented - Example

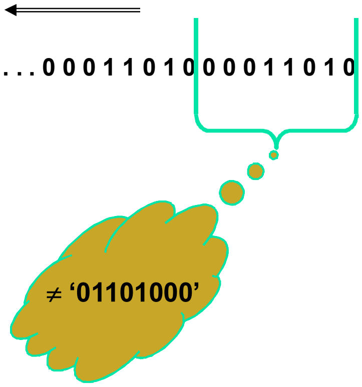


54

Character Oriented - Example

Direction ←   Time →

. . . . 0 0 0 1 1 0 1 0 0 0 0 1 1 0 1 0 0 0 0 1 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 . . . .

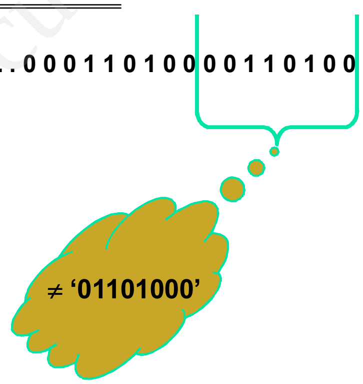


55

Character Oriented - Example

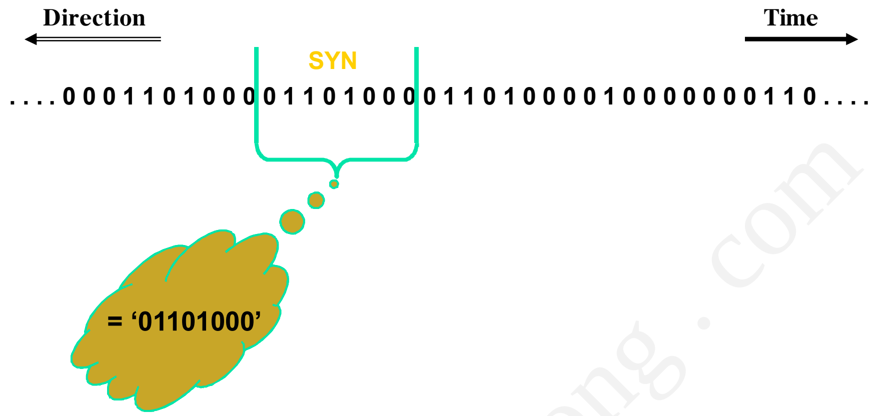
Direction ←   Time →

. . . . 0 0 0 1 1 0 1 0 0 0 0 1 1 0 1 0 0 0 0 1 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 . . . .



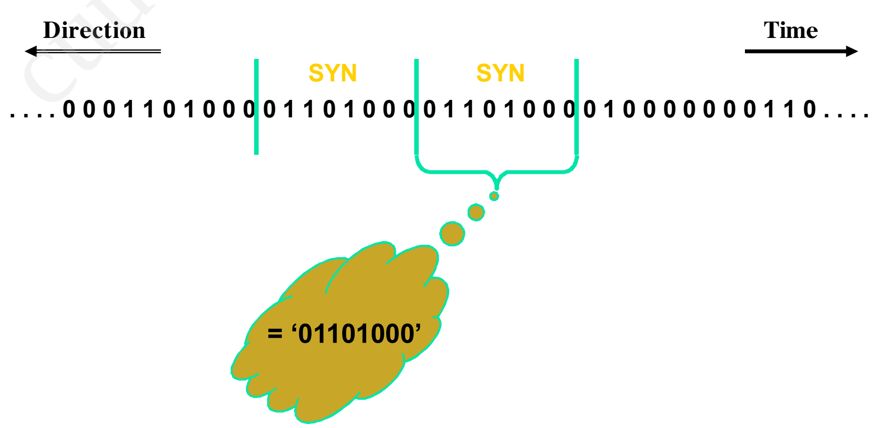
56

Character Oriented - Example



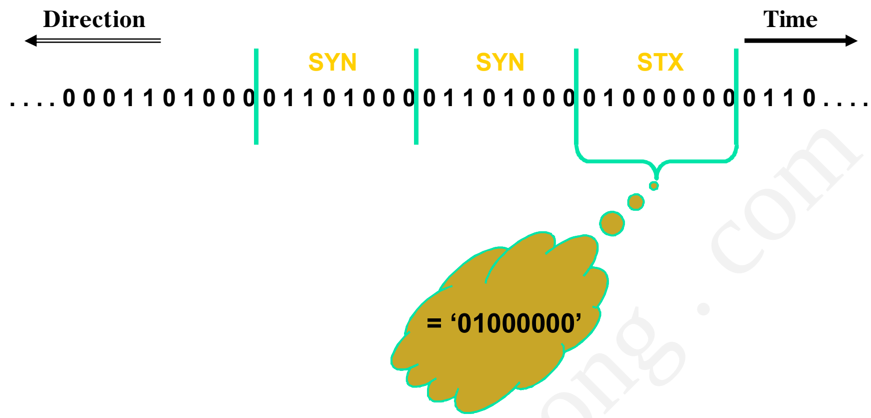
57

Character Oriented - Example



58

Character Oriented - Example



59

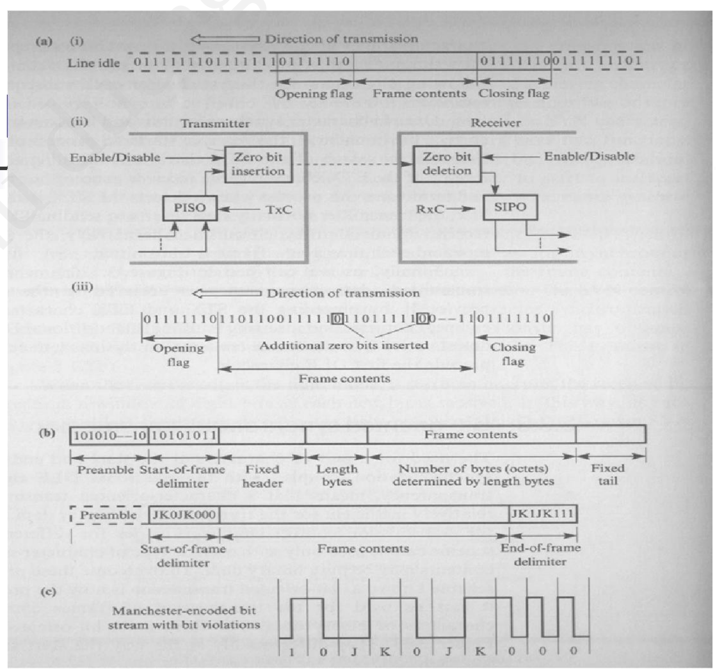
Character Oriented - Example

60

Frame Synchronization

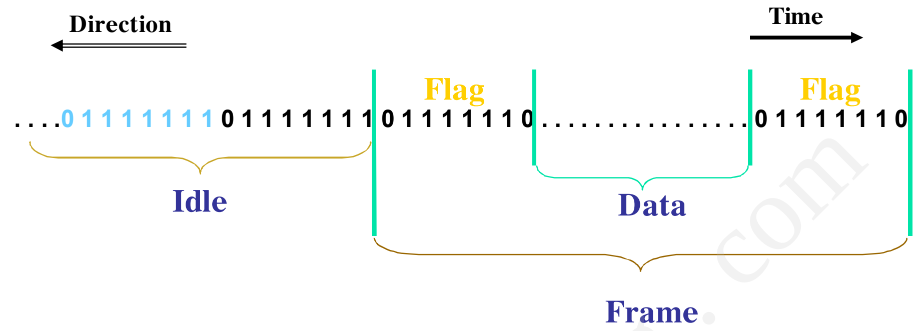
- Bit-oriented
  - The start and end of a frame are both signaled by the same unique 8-bit pattern 01111110, known as the flag byte or flag pattern
  - To enable the receiver to obtain and maintain bit synchronism, the transmitter sends a string of idle bytes (each comprising 01111111) preceding the start-of-frame flag
  - To achieve data transparency, zero bit insertion or bit stuffing is used

61



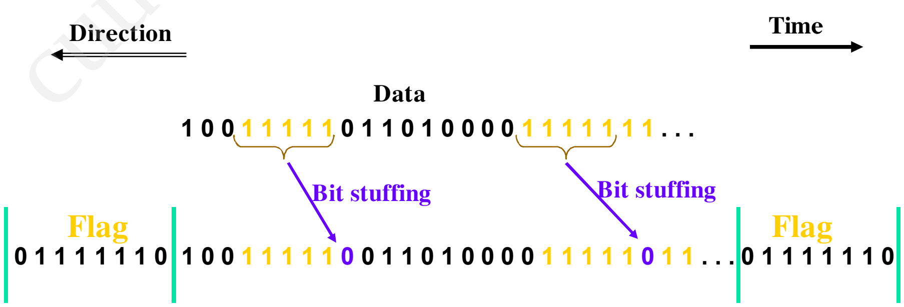
62

Bit Oriented - Example



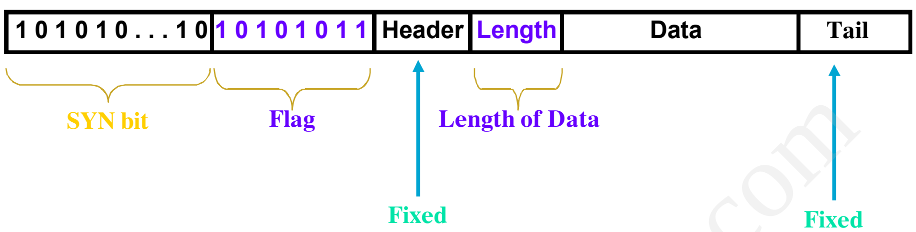
63

Bit Oriented - Example



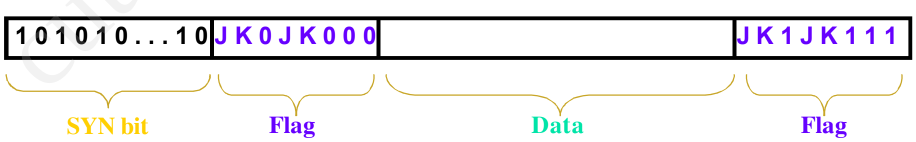
64

Bit Oriented - Example

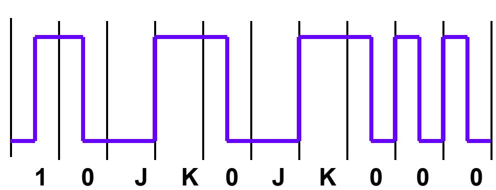


65

Bit Oriented - Example



Manchester encoding for flag



66

Error Detection Methods

- Forward error control
  - Detect errors when they are present and determine (if possible) where in the received bit stream the errors are
- Feedback (backward) error control
  - Retransmission control scheme is used to request that another, hopefully correct, copy of the information to be sent

→ Feedback error control is the predominant method used in the types of data communications

67

Error Detection Methods

- Feedback error control can be divided into two parts:
  - The techniques that are used to achieve reliable error detection (FEC)
  - The control algorithm that are available to perform the associated retransmission control schemes

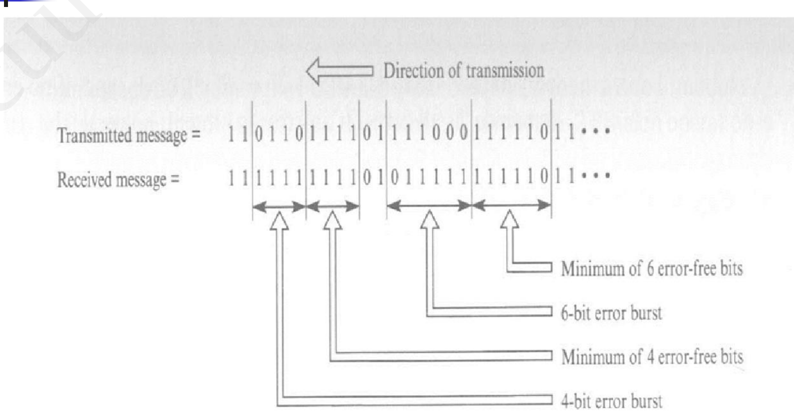
68

Error Detection Methods

- Two factors that determine the type of error detection scheme used:
  - Bit Error Rate (BER)
  - Type of errors
    - Single-bit error: errors occur as random single-bit errors, adjacent bits not affected
    - Burst errors: group of contiguous string of bit errors
      - Length B
      - Contiguous sequence of B bits in which first last and any number of intermediate bits in error

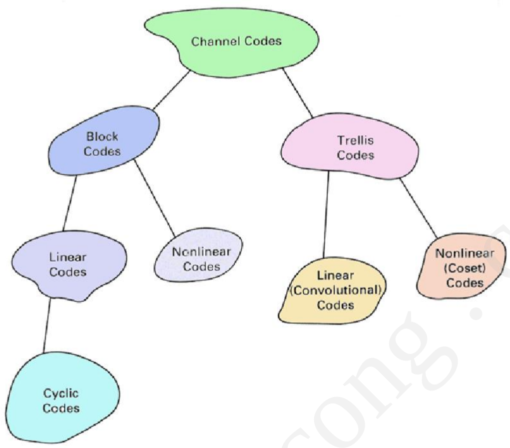
69

Error Detection Methods



70

Code Taxonomy



71

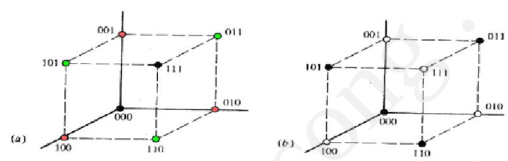
What is Coding?

- Coding is mapping of binary source (usually) output sequences of length *k* into binary channel input sequences *n* *(>k)*
- A block code is denoted by *(n,k)*
- Binary coding produces 2^k codewords of length *n.* Extra bits in codewords are used for error detection/correction

72

Representing Codes by Vector

- Code strength is measured by Hamming distance that tells how different code words are:
  - Codes are more powerful when their minimum Hamming distance d_{min} (over all codes in the code family) is large
- Hamming distance d(X,Y) is the number of bits that are different between code words
- (*n,k*) codes can be mapped into *n*-dimensional grid:

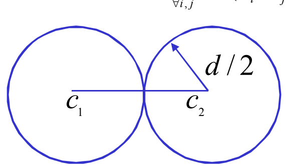


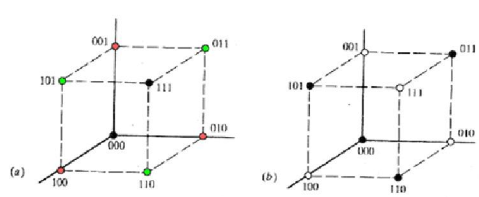
73

Hamming distance: The decision sphere interpretation

- Consider two block code (*n,k*) words c_1 and c_2 at the Hamming distance in the *n*-dimensional code space:

$$d = \min_{\forall i,j} d(c_i, c_j)$$





74

Error Detection Methods

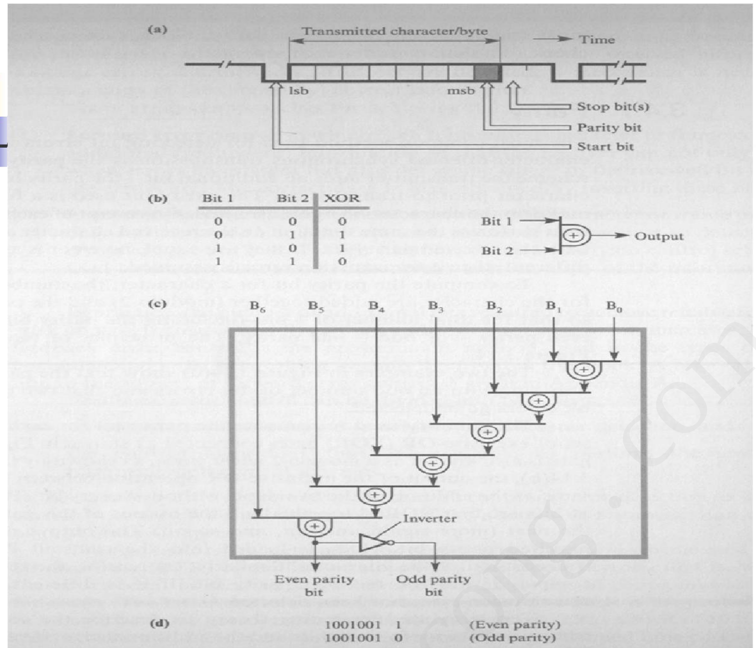
- Parity check
- Block sum check
- Cyclic Redundant Check

75

Parity Checking

- Used with asynchronous and character-oriented synchronous transmission
- Transmitter adds an additional bit – the parity bit – to each transmission character prior to transmission
- Two parity checking methods:
  - Even parity: The total number of 1 bits is even
  - Odd parity: The total number of 0 bits is odd

76



77

Parity Checking

- Detect all single-bit errors
- Does not detect 2-bit errors

78

Block Sum Check

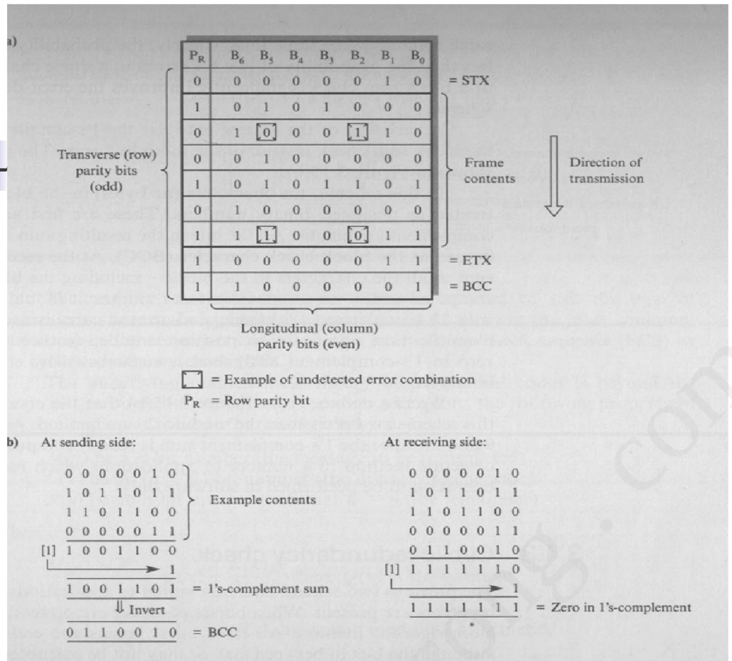
- When blocks of characters (frames) being transmitted, we can achieve an extension to the error-detecting capabilities obtained from a single parity bit per character (byte) by using an addition set of parity bits computed from the complete block of characters (byte) in frame.

79

Block Sum Check

- In this method:
  - Each character (byte) in the frame is assigned a parity bit as before (transverse or row parity)
  - An extra bit is computed for each bit position (longitudinal or column parity) in the complete frame.
- The set of parity bits for each column is referred to as the block (sum) check character

80



Block Sum Check

- Detect 2 bit errors in a character only if 2 bit errors occur in the same column at the same time.

Block Sum Check - Example

P_R	B_6	B_5	B_4	B_3	B_2	B_1	B_0	
0	0	0	0	0	0	1	0	STX
1	0	1	0	1	0	0	0	
0	1	0	0	0	1	1	0	
0	0	1	0	0	0	0	0	
1	0	1	0	1	1	0	1	
0	1	0	0	0	0	0	0	
1	1	1	0	0	0	1	1	
1	0	0	0	0	0	1	1	ETX
1	1	0	0	0	0	0	1	BCC (Even)

Odd Parity

83

1 Bit Error

P_R	B_6	B_5	B_4	B_3	B_2	B_1	B_0	
0	0	0	0	0	0	1	0	STX
1	0	1	0	1	0	0	0	
0	1	0	0	0	0	1	0	
0	0	1	0	0	0	0	0	
1	0	1	0	1	1	0	1	
0	1	0	0	0	0	0	0	
1	1	1	0	0	0	1	1	
1	0	0	0	0	0	1	1	ETX
1	1	0	0	0	0	0	1	BCC (even)

Odd Parity

84

2 Bit Error

Odd Parity

P_R	B_6	B_5	B_4	B_3	B_2	B_1	B_0	
0	0	0	0	0	0	1	0	STX
1	0	1	0	1	0	0	0	
0	1	1	0	0	0	1	0	
0	0	1	0	0	0	0	0	
1	0	1	0	1	1	0	1	
0	1	0	0	0	0	0	0	
1	1	1	0	0	0	1	1	
1	0	0	0	0	0	1	1	ETX
1	1	0	0	0	0	0	1	BCC (even)

85

Not Detected Cases

Odd Parity

P_R	B_6	B_5	B_4	B_3	B_2	B_1	B_0	
0	0	0	0	0	0	1	0	STX
1	0	1	0	1	0	0	0	
0	1	1	0	0	0	1	0	
0	0	1	0	0	0	0	0	
1	0	1	0	1	1	0	1	
0	1	0	0	0	0	0	0	
1	1	0	0	0	1	1	1	
1	0	0	0	0	0	1	1	ETX
1	1	0	0	0	0	0	1	BCC (even)

86

Cyclic Redundant Check

- Parity or block sum check does not provide a reliable detection schemes against error bursts → the most common alternative is based on the us of polynomial codes
- Polynomial codes are used with frame (or block) transmission scheme
- A single set of check digits is generated (computed) for each frame transmitted based on the content of the frame, and is appended by the transmitter to the tail of the frame
- The computed check digits are referred to as the Frame Check Sequence (FCS) or Cyclic Redundant Check (CRC)

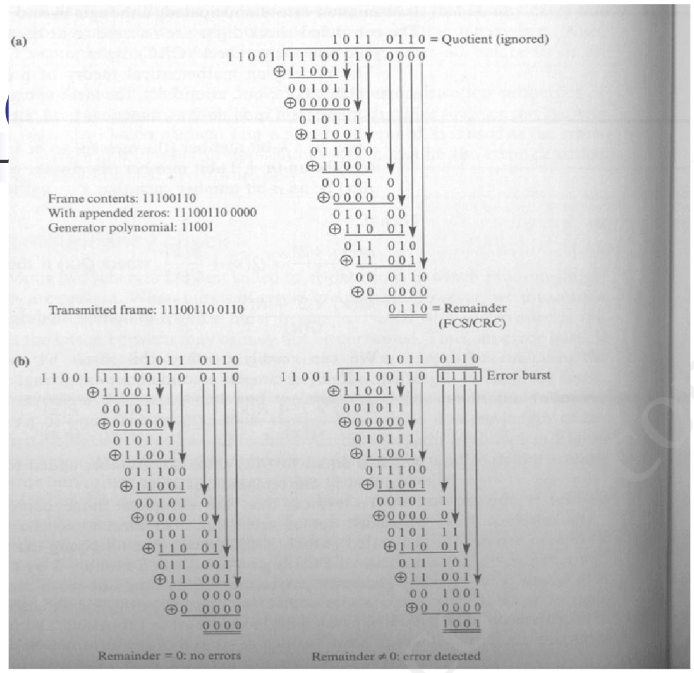
87

Cyclic Redundant Check

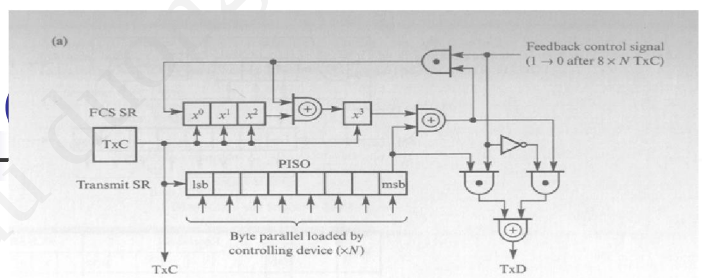
- M(x) be a k-bit number (the message to be transmitted)
- G(x) be an (n+1)-bit number (the divisor or generator)
- R(x) be an n-bit number (the remainder)
- T(x) be transmitted bit

→ $$T(x) = x^n . M(x) + R(x)$$

88



89



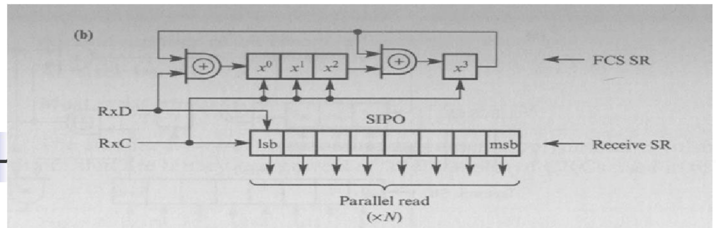
TxC pulses	Transmit SR lsb							msb	FCS SR x^0	x^1	x^2	x^3
0	0	1	1	0	0	1	1	1	0	0	0	0
1	0	0	1	1	0	0	1	1	1	0	0	1
2	0	0	0	1	1	0	0	1	0	1	0	0
3	0	0	0	0	1	1	0	0	1	0	1	1
4	0	0	0	0	0	1	1	0	1	1	0	0
5		0	0	0	0	0	1	1	0	1	1	0
6			0	0	0	0	0	1	1	0	1	0
7				0	0	0	0	0	1	1	0	0
8					0	0	0	0	0	1	1	0
9						0	0	0		0	1	1
10							0	0			0	1
11								0				0

Time

SR = Shift register

TxD

90



RxC	RxD	Receive SR								FCS SR			
		lsb							msb	x^0	x^1	x^2	x^3
0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1								1	0	0	0
2	1	1	1							1	1	0	0
3	0	1	1	1						1	1	1	0
4	0	0	1	1	1					0	1	1	1
5	1	0	0	1	1	1				1	0	1	0
6	1	1	0	0	1	1	1			1	1	0	1
7	0	1	1	0	0	1	1	1		0	1	1	1
8	0	0	1	1	0	0	1	1	1	1	0	1	0
		Byte read by controlling device											
9	1									0	1	0	1
10	1									0	0	1	1
11	0									0	0	0	0
12										0	0	0	0

Remainder = 0

91

Cyclic Redundant Check

- Detect
  - All single-bit errors
  - All double-bit errors
  - All odd number of bit errors
  - All error bursts < R
  - Most error burst ≥ R

92

Data Compression

- Packed decimal
- Relative encoding
- Character suppression
- Huffman coding
- Run-length coding

93

Huffman Coding

- Huffman coding exploits the property that not all symbols in a transmitted frame occur with the same frequency.
- Encoding scheme: most common characters are encoded using fewer bits than less frequent characters → statistical encoding.
- Variable number of bits per character → used in bit-oriented transmission.

94

Huffman Coding

- The character string to be transmitted is analyzed and the character types and their relative frequency determined.
- The coding operation involves creating an unbalanced tree with some branches → Huffman code tree (binary tree)

95

Huffman coding

- Binary tree with branches assigned the value 0 or 1
  - The base of the tree, the geometric top, is known as the root node
  - The point at which a branch divides is known as the branch node
  - The termination point of a branch is known as a leaf node

96

Huffman Coding

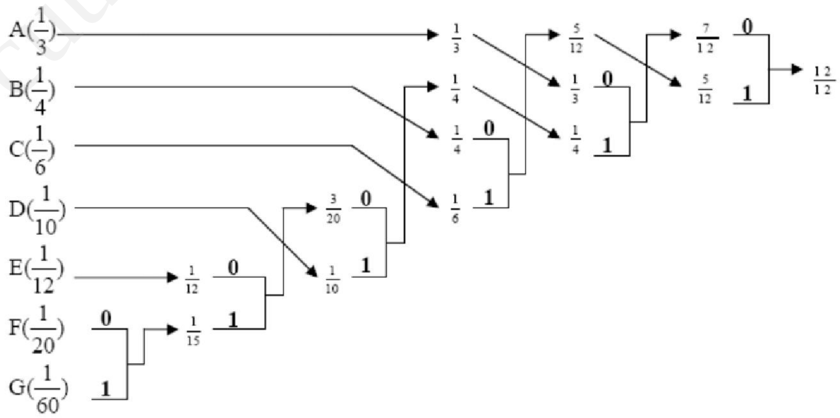
- The theoretical minimum average number of bits per codeword to transfer the message string is known as the entropy, *H*, of the message.
- Entropy *H* (bits per codeword) can be calculated:

$$H = -\sum_{i=1}^{n} P_i \log_2 P_i$$

  - N is the number of characters
  - P_i is the probability of character i

97

Huffman Coding

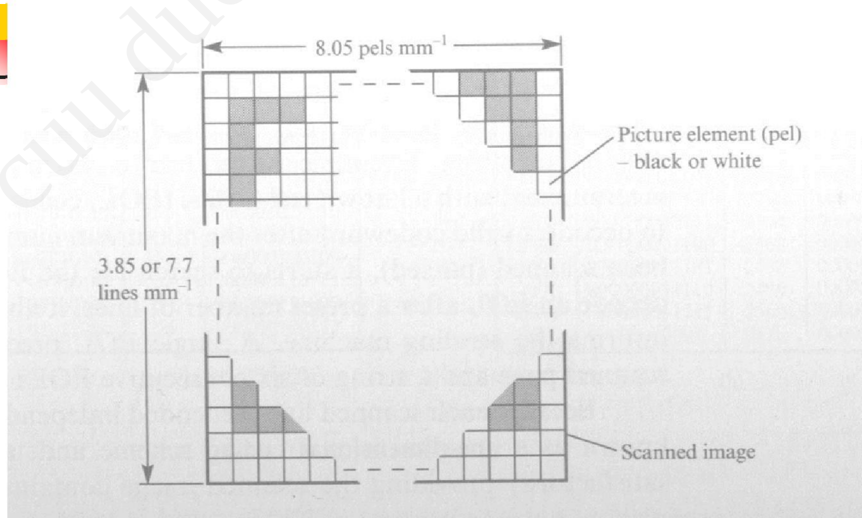


98

Run-length Coding

- Used in black and white Facsimile machine
- It scans a page with a vertical resolution of 3.85 or 7.7 lines per millimeter – approximately 100 or 200 lines per inche.
- Each scan line is digitized at the rate of 8.05 picture element or pels per millimeter – a 0 for white and 1 for black
- A typical scanned page produces about 2 million binary digits

99



100

Run-length Coding

- The codewords are fixed and grouped into 2 separate groups: the termination-codes and the make-up codes
- To enable the receiver to regain synchronism, each scanned line is terminated with a known end-of-line (EOL) code.
- String of six EOLs indicate the end of a page

101

White run length	Code-word	Black run length	Code-word
0	00110101	0	0000110111
1	000111	1	010
2	0111	2	11
3	1000	3	10
4	1011	4	011
5	1100	5	0011
6	1110	6	0010
7	1111	7	00011
8	10011	8	000101
9	10100	9	000100
10	00111	10	0000100
11	01000	11	0000101
12	001000	12	0000111
13	000011	13	00000100
14	110100	14	00000111
15	110101	15	000011000
16	101010	16	0000010111
17	101011	17	0000011000
18	0100111	18	0000001000
19	0001100	19	00001100111
20	0001000	20	00001101000
21	0010111	21	00001101100
22	0000011	22	00000110111
23	0000100	23	00000101000
24	0101000	24	00000010111
25	0101011	25	00000011000
26	0010011	26	000011001010
27	0100100	27	000011001011
28	0011000	28	000011001100
29	00000010	29	000011001101
30	00000011	30	000001101000
31	00011010	31	000001101001
32	00011011	32	000001101010
33	0010010	33	000001101011
34	00010011	34	000011010010
35	00010100	35	000011010011
36	00010101	36	000011010100
37	00010110	37	000011010101
38	00010111	38	000011010110
39	00101000	39	000011010111
40	00101001	40	000001101100
41	00101011	41	000001101101
42	00101011	42	000011011010
43	00101100	43	000011011011
44	00101101	44	000001010100
45	00000100	45	000001010101
46	00000101	46	000001010110
47	00001010	47	000001010111
48	00001011	48	000001100100
49	01010010	49	000001100101
50	01010011	50	000001010010
51	01010100	51	000001010011
52	01010101	52	000000100100
53	00100100	53	000000110111
54	00100101	54	000000111000
55	01011000	55	000000100111

(a)

White run length	Code-word	Black run length	Code-word
56	01011001	56	000000101000
57	01011010	57	000001011000
58	01011011	58	000001011001
59	01001010	59	000000101011
60	01001011	60	000000101100
61	00110010	61	000001011010
62	00110011	62	000001100110
63	00110100	63	000001100111

(a) cont.

White run length	Code-word	Black run length	Code-word
64	11011	64	0000001111
128	10010	128	000011001000
192	010111	192	000011001001
256	0110111	256	000001011011
320	00110110	320	000000110011
384	00110111	384	000000110100
448	01100100	448	000000110101
512	01100101	512	0000001101100
576	01101000	576	0000001101101
640	01100111	640	0000001001010
704	011001100	704	0000001001011
768	011001101	768	0000001001100
832	011010010	832	0000001001101
896	011010011	896	0000001110010
960	011010100	960	0000001110011
1024	011010101	1024	0000001110100
1088	011010110	1088	0000001110101
1152	011010111	1152	0000001110110
1216	011011000	1216	0000001110111
1280	011011001	1280	0000001010010
1344	011011010	1344	0000001010011
1408	011011011	1408	0000001010100
1472	010011000	1472	0000001010101
1536	010011001	1536	0000001011010
1600	010011010	1600	0000001011011
1664	011000	1664	0000001100100
1728	010011011	1728	0000001100101
1792	00000001000	1792	00000001000
1856	00000001100	1856	00000001100
1920	00000001101	1920	00000001101
1984	000000010010	1984	000000010010
2048	000000010011	2048	000000010011
2112	000000010100	2112	000000010100
2176	000000010101	2176	000000010101
2240	000000010110	2240	000000010110
2304	000000010111	2304	000000010111
2368	000000011100	2368	000000011100
2432	000000011101	2432	000000011101
2496	000000011110	2496	000000011110
2560	000000011111	2560	000000011111
EOL	00000000001	EOL	00000000001

(b)

102

Transmission Control Circuits

- Universal asynchronous receiver transmitter (UART)
  - Start and stop bit insertion and deletion
  - Bit (clock synchronization)
  - Character synchronization
  - Parity bit generation and checking per character (BCC computed by controlling device)

103

Transmission Control Circuits

- Universal synchronous receiver transmitter
  - Low bit rate DPLL clock synchronization
  - Character synchronization
  - Synchronous idle character generation
  - Parity generation and checking per character (BCC computed by controlling device)

104

Transmission Control Circuits

- Universal synchronous / asynchronous receiver transmitter (USART)
  - Can be programmed to operate as either a UART or USRT
  - Has all programmable features of both devices

105

Transmission Control Circuits

- Bit-oriented protocol circuits (BOPs)
  - Opening and closing flag insertion and deletion
  - Zero bit insertion and deletion
  - CRC generation and checking
  - Idle pattern generation

106

Transmission Control Circuits

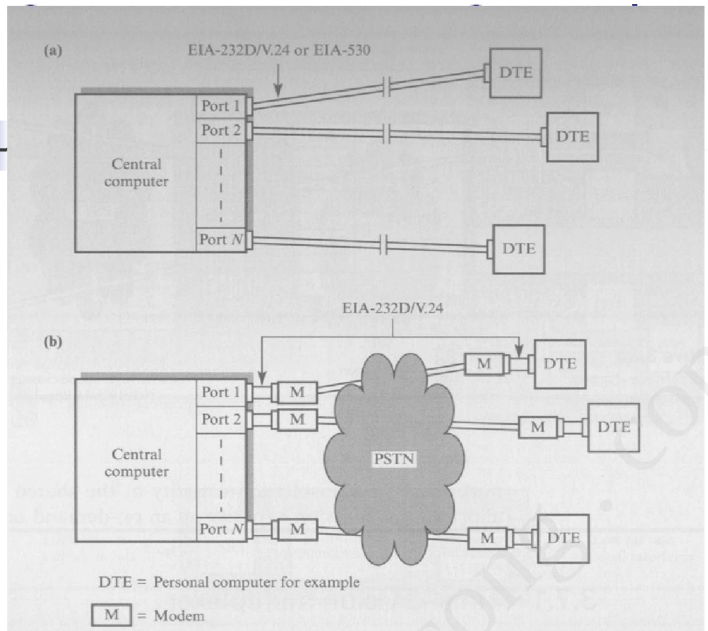
- Universal communication control circuits
  - Can be programmed to operate either UART, a USRT or a BOP
  - Has all the programmable features of each circuit

107

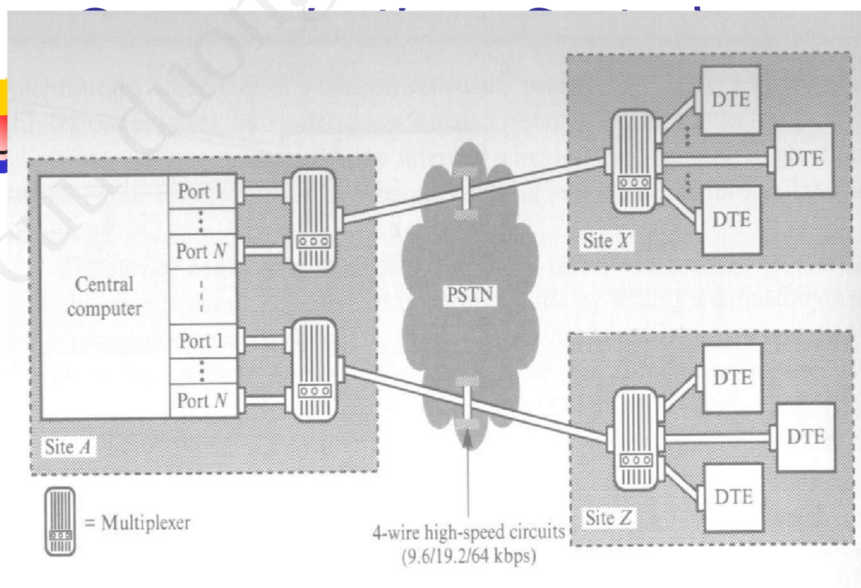
Communications Control Devices

- Simple terminal networks
- Multiplexer-based networks

108

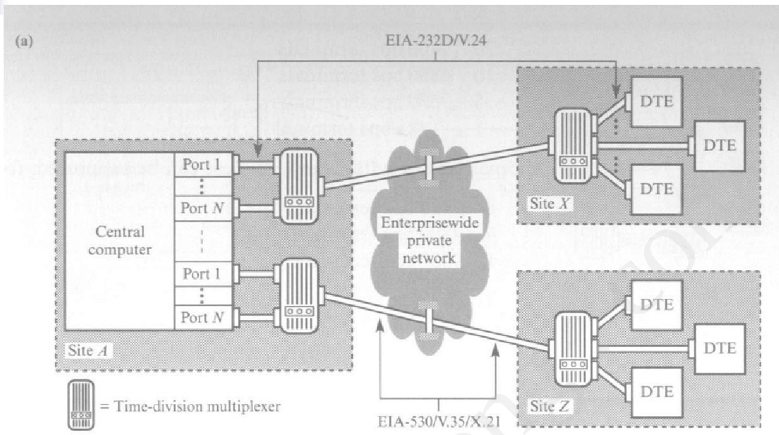

(a)
EIA-232D/V.24 or EIA-530
Central computer
Port 1
Port 2
Port N
DTE
DTE
DTE
(b)
EIA-232D/V.24
Central computer
Port 1
Port 2
Port N
M
PSTN
DTE
DTE = Personal computer for example
M = Modem

109

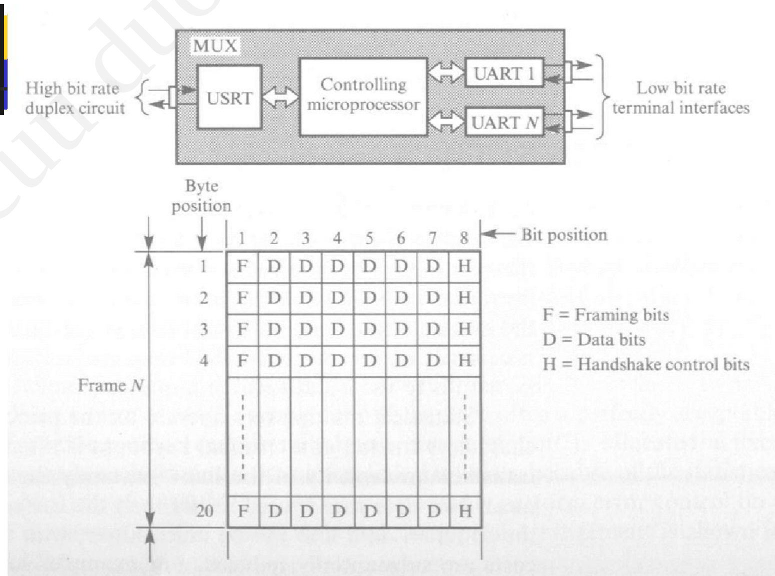

Central computer
Port 1
Port N
Site A
PSTN
Site X
Site Z
DTE
= Multiplexer
4-wire high-speed circuits (9.6/19.2/64 kbps)

110

Time-division Multiplexer

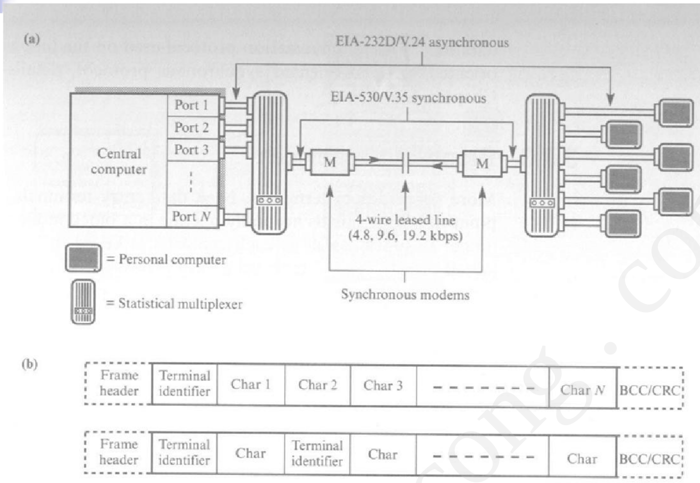


111



112

Statistical Multiplexer



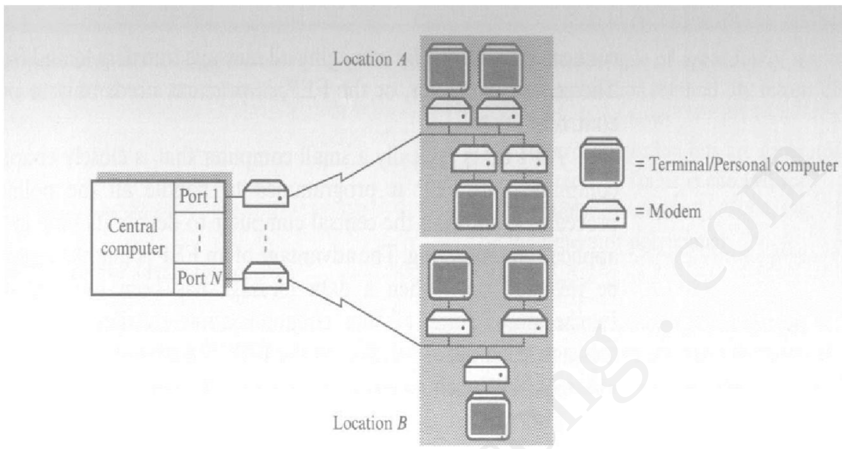
113

Block-mode Devices

- Multidrop (multipoint) lines
- Cluster controller
- Hub-polling

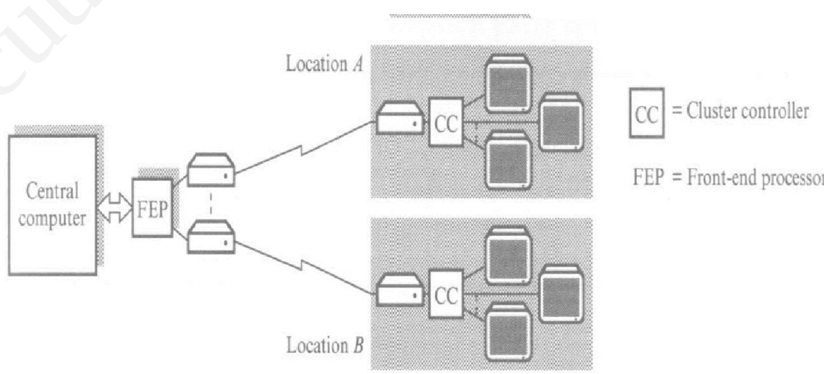
114

Block-mode Devices



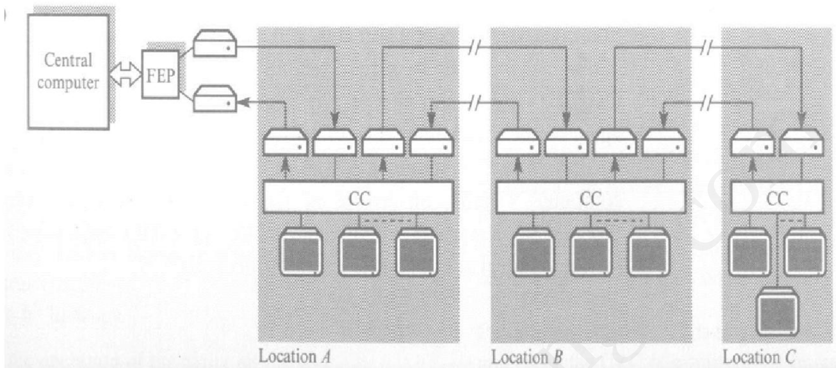
115

Block-mode Devices

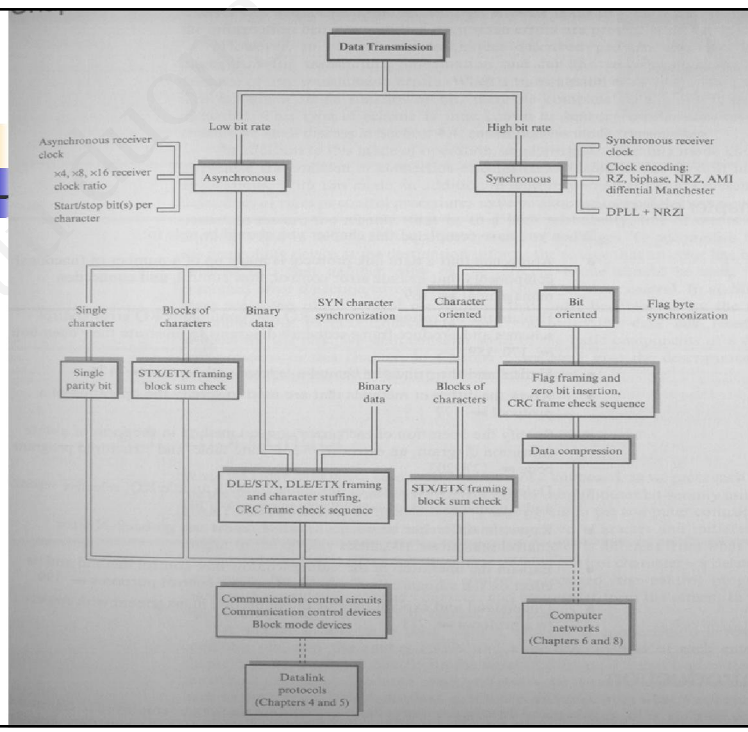


116

Block-mode Devices



117



118