

# Data Link Layer

”Data is being packaged into frames”

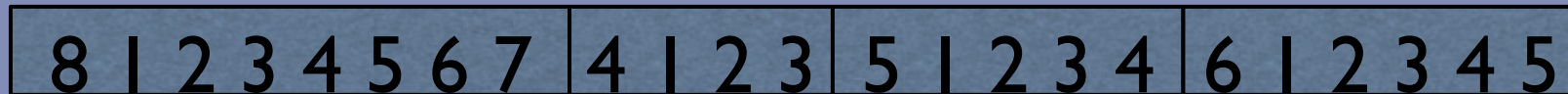
- **Frame Synchronization:** Begin & End of Frame must be detectable
- **Flow Control:** Sender should not send frames faster than the Receiver can handle
- **Error Control:** Error detection and correction
- **Addressing:** Source and Destination address
- **Control & Data Integration:** Control and Data have to be packed and unpacked in the same frame
- **Link Management:** Procedures for initiation, maintenance and termination

# FRAMING

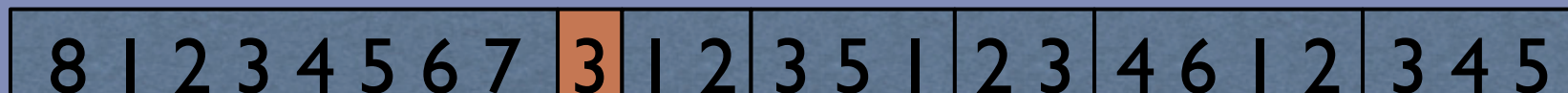
## Frame Synchronization

1. Character Count
2. Starting & Ending Characters with Character Stuffing
3. Starting & Ending Flags with Bit Stuffing
4. Physical Layer Coding Violations

# I. Character Count

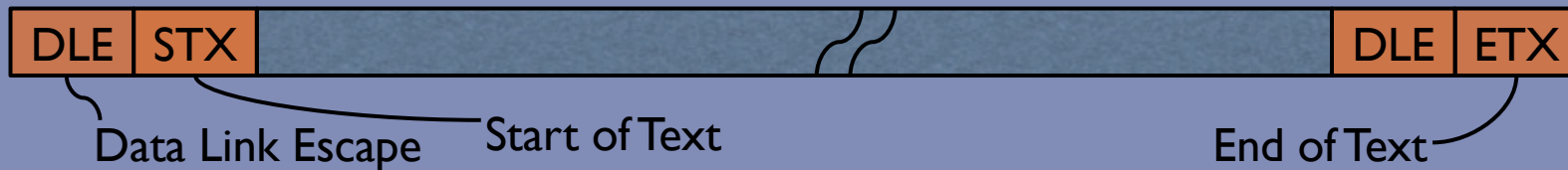


What if through a bit error character field gets a different number?

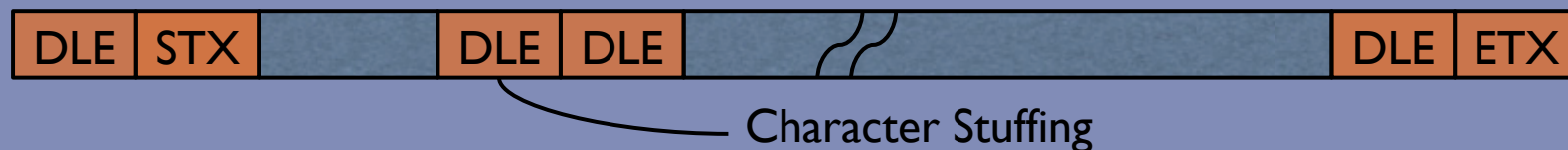


A MESS !!!!!!! Recovery not possible

# 2. Character Stuffing



What if by accident DLE is also part of the payload?



On receiving site Character Destuffing is required!!



Disadvantage: Character (ASCII) bound

# 3. Bit Stuffing

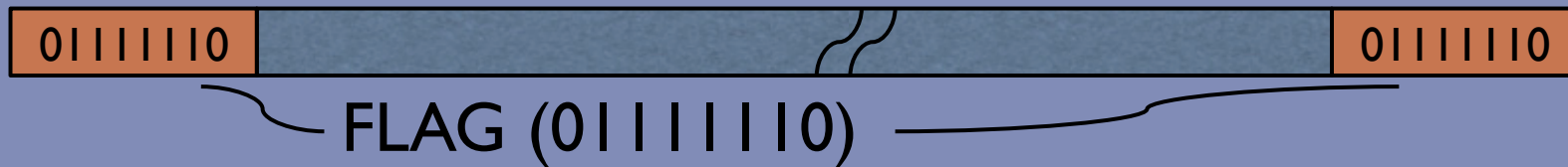


Now every sequence of 5 one's in the payload is replaced by 11110 (bit stuffing)

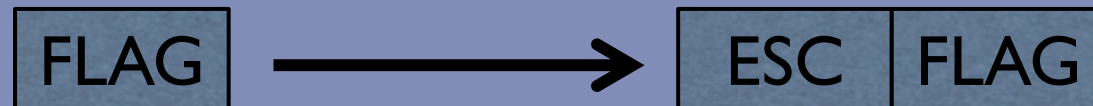
So: 011111111110010  
      ↓ stuffing  
      01111101111100010  
      ↓ destuffing  
      011111111110010

# The “ppp” Protocol

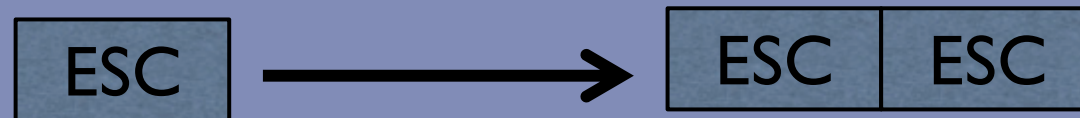
Point-to-Point Protocol “defacto” protocol for ISP’s



What if FLAG occurs within the payload?



What if ESC occurs within the payload?



Etc. etc.... So a mix of bit and character stuffing

# 4. Physical Layer Code Violations

Using redundancy in physical layer encoding

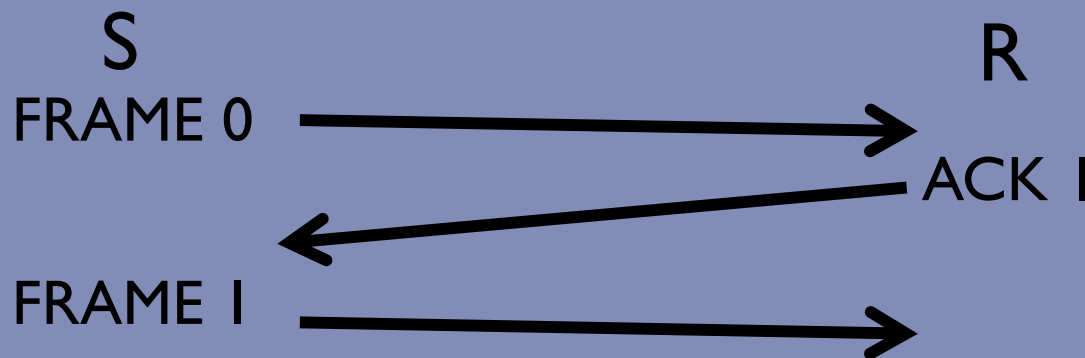
Example:	00	error	} 802 LAN standard
	01	0	
	10	1	
	11	error	

So error code can be used as escapes.

# FLOW CONTROL

## I. Stop-and-Wait Flow Control

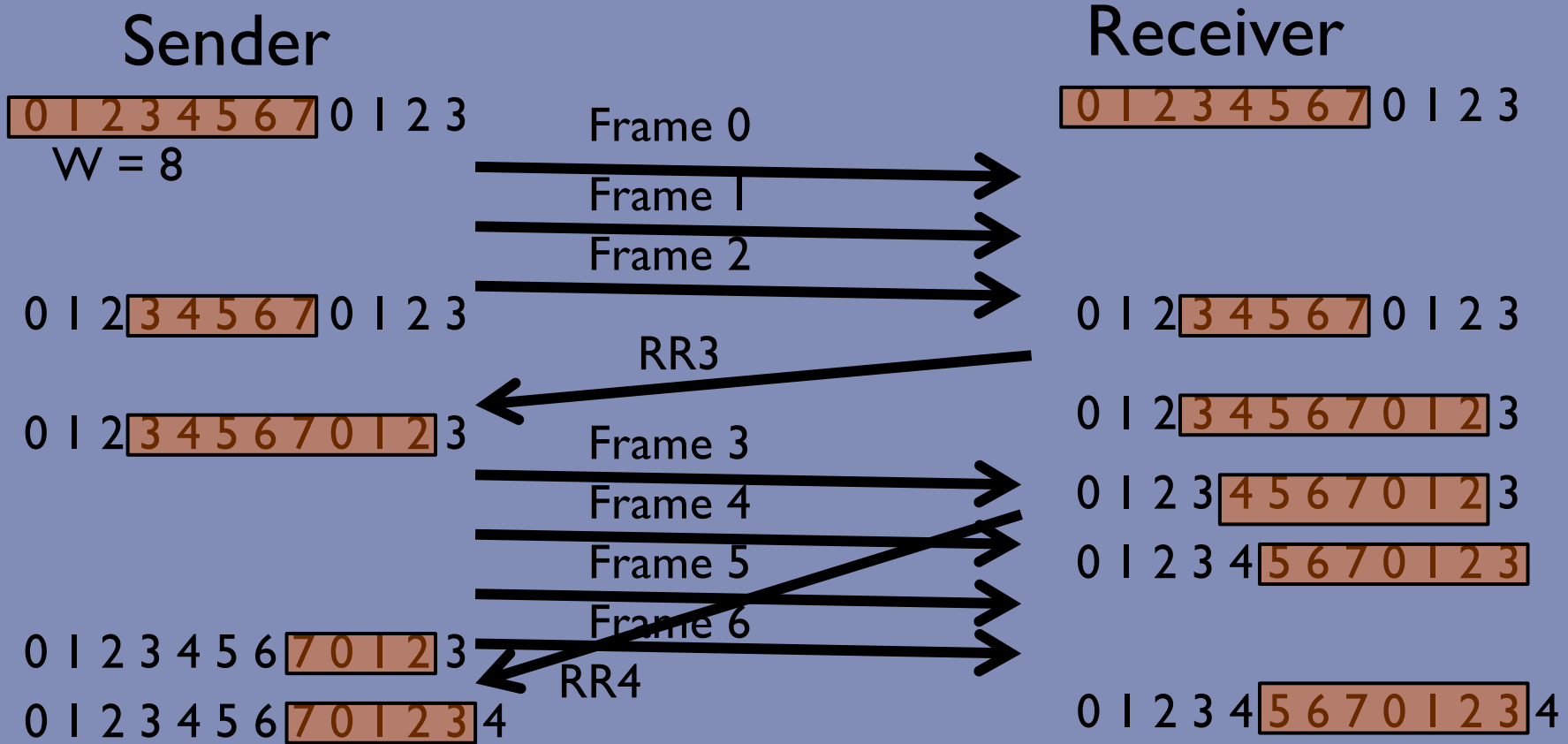
Only send the next packet (frame), when an explicit ACK of the previous packet has been received.



Transmission time per frame:  $t + 2a$  instead of  $t + a$   
So if  $a$  (latency) is comparable to  $t$  (frame processing time),  
then 30% efficiency loss !!!!!



# 2. Sliding Window Protocol



**RRn (Received & Ready):**

**Received frames up till n-1 and ready to receive frame n**

Often Combined with:

**RNRn (Received but Not Ready):**

**Received frames up till n-1, but not ready to receive frame n**

# Piggy Backing

Normally data communication is bi-directional:  
So communication is between S/R and R/S

In this case, pack both **Frame Id** as well as **Frame Ack Id** in the same Frame.

If there is no data to be send: send separate ACK

If there is no ACK to be send: repeat previous ACK

# Flow Control & Error Control

## Two types of Errors

- LOST FRAMES
- DAMAGED FRAMES (as detected by CRC, for instance)

## In general solved by a combination of:

1. Error detection
2. Positive ACK (for error free frames)
3. Retransmission after Time
4. Negative ACK & Retransmission

These four mechanisms together form  
an **Automatic Repeat ReQuest (ARQ)** protocol

# Stop\_and\_Wait ARQ

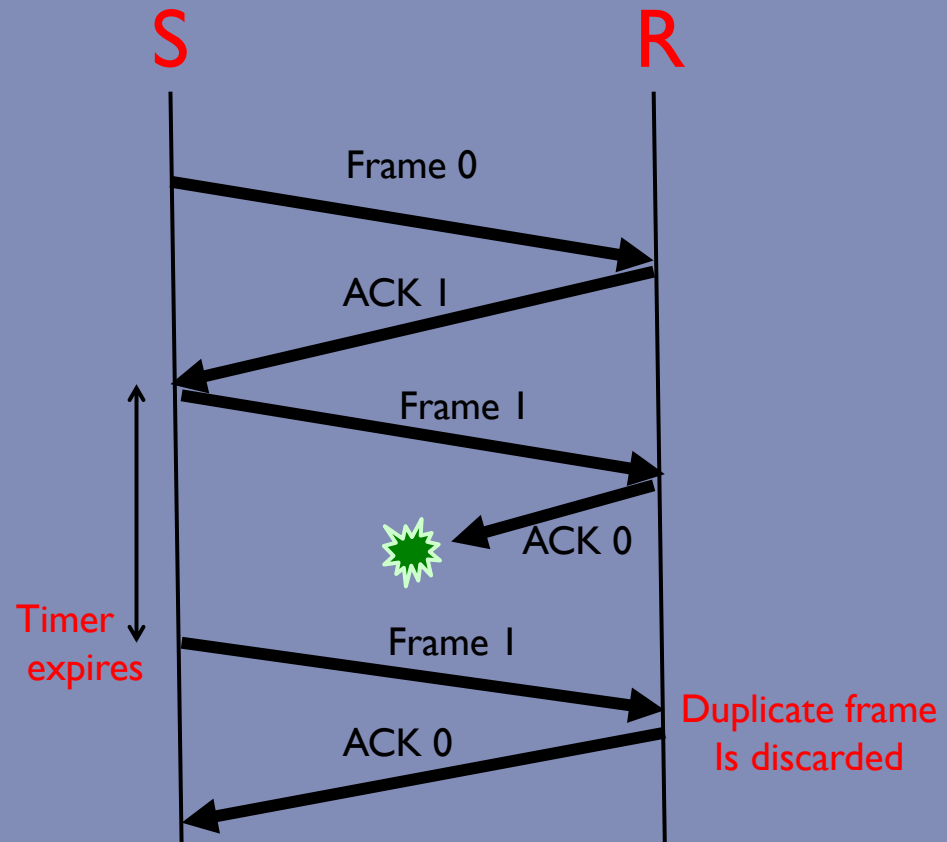
## Principle:

- After sending a frame, sender starts a timer
- If timer expires and sender has not received a positive ACK, then sender does a RETRANSMIT of the frame

2 Cases:

- Receiver gets a damaged frame, then simply discard
- Sender gets a damaged ACK: PROBLEM

This is solved by numbering the frames & ACK 0, 1, 0, 1,.... (ACK 0 acknowledges correct receipt of frame 1 and vice versa. Or ACK 0 tells sender that a frame with number 0 can be send.



# Go\_back\_N ARQ

## Principle: Pipelined version of Stop\_and\_Wait ARQ

Frames are numbered sequentially modulo a number N

Two additional messages:

- **RR** Ready to Receive
- **REJ** Reject, all remaining frames in the pipeline are discarded

### 1 Damaged (Lost) Frame

S → Frame i

R detects error on Frame i and Frame i-1 was received correctly

Then R discards Frame i, now **two cases**:

**a)** Within a certain amount of time

S → Frame i+1

R receives Frame i+1 out of order

R → REJ i

S retransmits Frame i and following Fr.

**b)** Within a certain amount of time

S → nothing; R → nothing; etc.

Timer S expires

S → RR Frame with poll-bit (P) = 1

R → RR i

S → retransmits Frame i

### 2 Damaged (Lost) RR

S → Frame i

R → RR i+1

RR i+1 gets lost

#### Two cases

**a)** Before Timer S expires:

R → RR j, with  $j > i+1$

Everything is OK

**b)** Timer S expires

S → RR with P-bit = 1

S turns on P-bit timer

if P-bit timer expires,

retry, retry, .....

if not successful after

a number of times

S → RESET

### 3 Damaged REJ

Equivalent to **1b**: If S is out of its window or will run out of its window and then S will expect a RR message in order to proceed. R is waiting for Frame i so will not send an RR. So nothing will happen and timer S expires Then the same actions are taken as under **1b**.

# Selective Reject ARQ

## Principle:

- Only retransmit those frames who actually went lost, and who caused a SREJ message to be sent.

*Seems more efficient, BUT buffering is required BOTH at Sender and Receiver next to (re)ordering logic*

*→ Extra LOGIC*

*→ More costly*

# Putting it Together

## “The HDLC protocol”

- High Level Data Link Protocol (ISO 3009, ISO 4335)
- IBM: SDLC (Synchronous DLC) → ANSI/ISO (1975)
- CCITT → LAP (Link Access Procedure) for X.25 in 1976 (Orange Book for WAN)
- ANSI → ADCCP (Advanced Data Communication Control Procedure)  
ISO → HDLC, both in 1979
- CCITT → LAPD as part of ISDN to make it more compatible with HDLC in 1993

CCITT: Comité Consultatif International Telegraphique et Telephonique  
predecessor of the ITU: International Telecommunication Union

# The HDLC protocol

## ◆ 3 TYPES STATIONS:

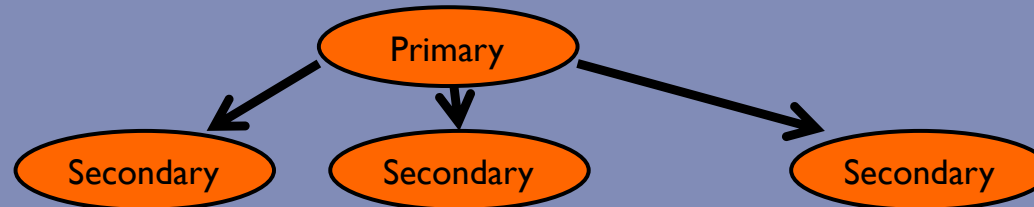
**Primary Stations issue COMMAND frames**

**Secondary Stations issue RESPONSE frames**

**Combined Stations issue both frames**

## ◆ 2 CONFIGURATIONS:

**Unbalanced Configuration:**



**Balanced Configuration:**





# The HDLC protocol

## ◆ 3 Data Transfer Modes:

### **Normal Response Mode (NRM)**

#### **Unbalanced Configuration**

Used on “multi-drop” lines (host-terminals)

### **Asynchronous Balanced Mode (ABM)**

#### **Balanced Configuration**

Both stations can initiate communication; no permission is required

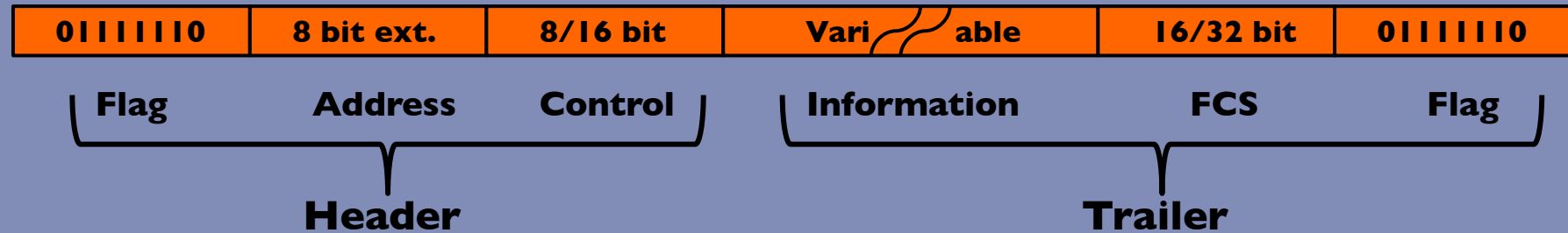
Used for point to point connections

### **Asynchronous Response Mode (ARM)**

#### **Unbalanced Configuration**

Secondary station can initiate transmission without explicit permission of the primary, primary stays responsible for error recovery etc.

# The HDLC protocol Frame Structure



## Flag

Bit stuffing is: after every five 1's insert a 0.

→ Still “strange” things can happen with single bit errors

01011110 → 01111110 in the information field

## Address

Address of the secondary station.

11111111 means broadcast from primary to all secondaries

Extendible: 

01110010	01001110	111011010
----------	----------	-----------

# The HDLC protocol

## Frame Structure

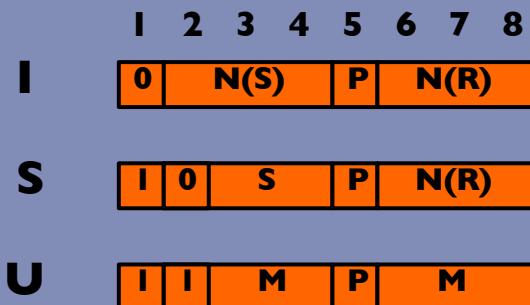
### Control

**I-frames** (Information frames): Data+Control data (piggyback.).

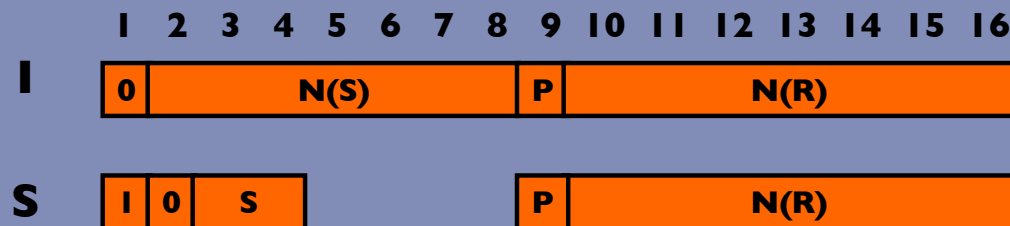
**S-frames** (Supervisory frames): ARQ mechanism if no piggyback.

**U-frames** (Unnumbered frames): Additional link control funct.

### 8-bit Control



### 16-bit Control



# The HDLC protocol

## Frame Structure

### **Information Field**

A variable number of octets (8 bits, byte). Maximum number determined by the system parameters

### **Frame Check Sequence Field (FCS)**

16-bit CRC-CCITT on all bits except Flags and FCS bits  
possibly 32-bit CRC for large frames or high reliability

# The HDLC protocol

## Commands & Responses

Type	C/R	Description
Information (I)	C/R	Exchange User Data
Supervisory (S)		
Receive Ready (RR)	C/R	Ready to receive I-frame
Received Not Ready (RNR)	C/R	Ack. But not ready to receive
Reject (REJ)	C/R	Go back N
Selective Reject (SREJ)	C/R	Selective Reject
Unnumbered (U)		
Set Normal Response Mode / Extended (SNRM/SNRME)	C	Set mode, extended: 7 bit sequence number
Set Asynchronous Response Mode / Extended (SARM/SARME)	C	Set mode, extended: 7 bit sequence number
Set Asynchronous Balanced Mode / Extended (SABM/SABME)	C	Set mode, extended: 7 bit sequence number
Set Initialization Mode (SIM)	C	Initialize Link Control function
Disconnect (DISC)	C	Terminate
Unnumbered Acknowledgement (UA)	R	Acknowledgement of the set mode commands
Disconnect Mode (DM)	C	Terminate
Request Disconnect (RD)	R	Request for DISC
Request Initialization Mode (RIM)	R	Request for SIM
Unnumbered Information (UI)	C/R	Exchange Control Information
Unnumbered Poll (UP)	C	Ask Control Information

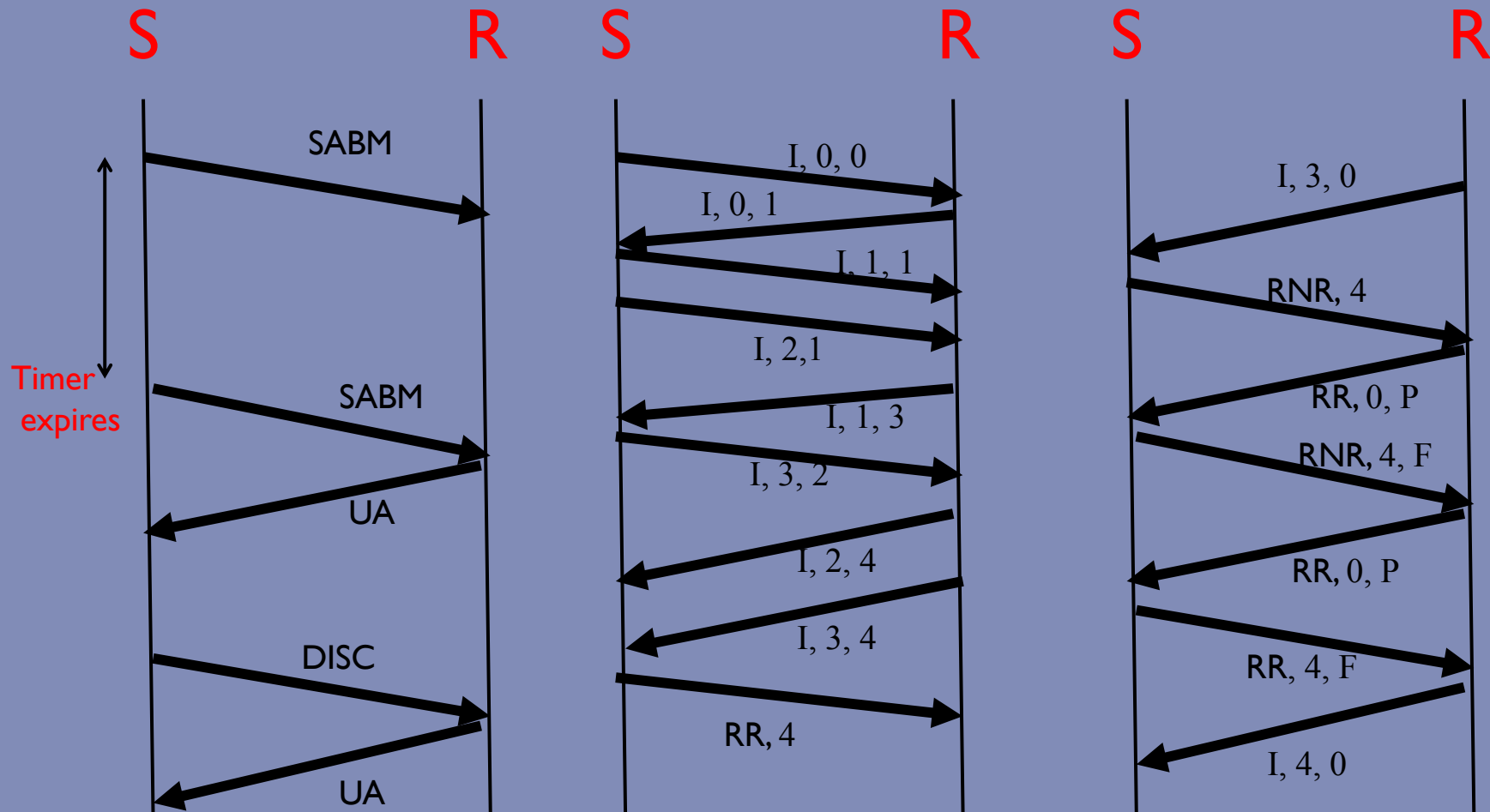
# The HDLC protocol

## Commands & Responses (cont.)

Type	C/R	Description
Reset (RSET)	C	Reset N(S) and N(R)
Exchange Identification (XID)	C/R	Request/Report Status
Test (TEST)	C/R	Exchange Identical Info Fields for checking
Frame Reject (FRMR)	R	Unacceptable Frame

# The HDLC protocol

## Examples



# Why Error Detection

Even with a bit error rate of  $10^{-9}$ , so 1 bit out of 1000 000 000 is wrong, then

With a line of 1 Mbps and 1000 bit frames:

$(1000\ 000 \times 3600 \times 24 \times 10^{-9})$  bit errors =  
86.4 bit errors per day  $\approx$

➔ 75\* wrong frames a day!!!!!!!

With 100 Mbps: 7500 wrong frames a day.

\* K balls in N buckets. The expected number of buckets with at least one ball =  $N * (1 - ((N-1)/N)^K)$  ( $N = 1000000 \times 3600 \times 24 / 1000$ ,  $K = 86.4$ )



# A Simple Scheme

## Odd/Even Parity

Odd parity: 0110 → 01101

Even parity: 0110 → 01100

# Theory

In general: If data to be send consists of  $m$  bits, then **add  $r$  redundant bits**.

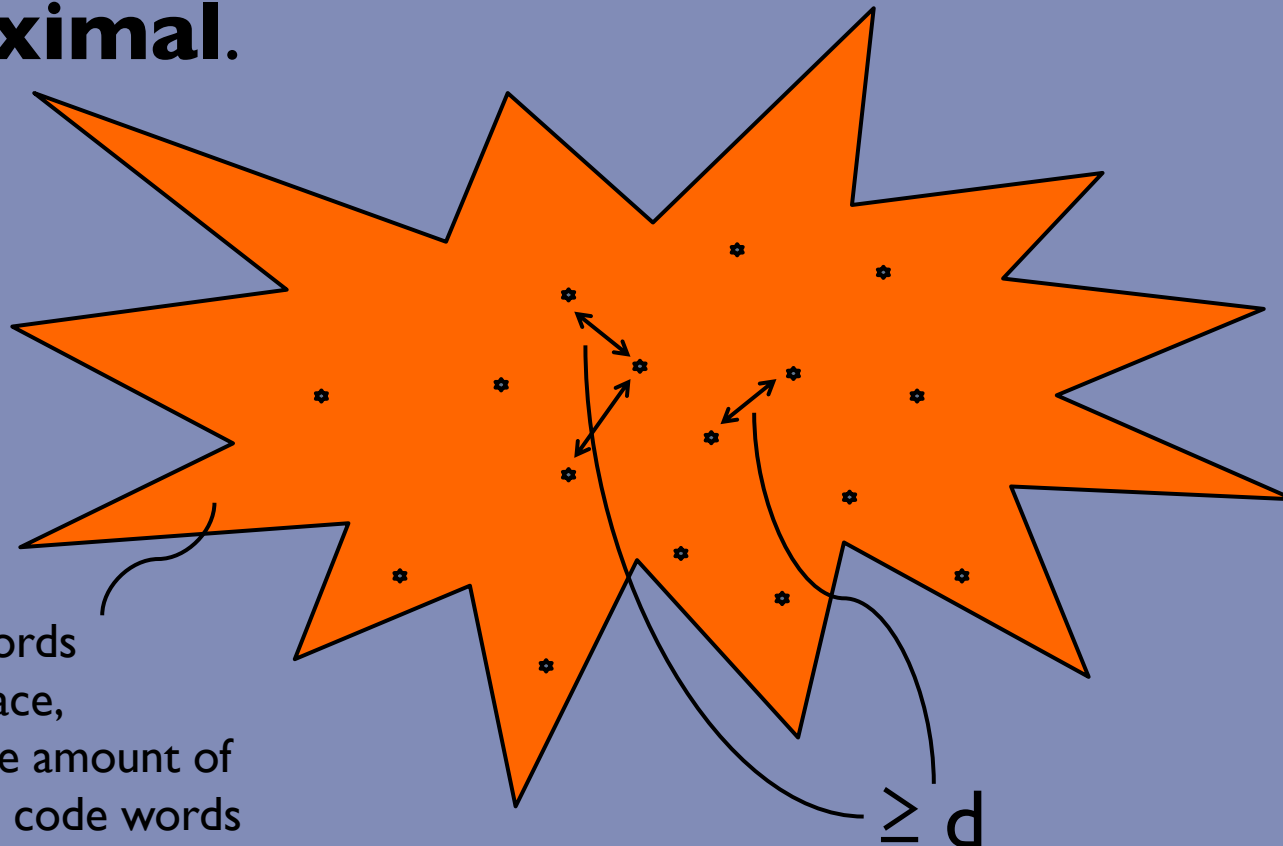
So  $m$  bits are being packed into  $m + r (= n)$  bits words called “code words”, think of encrypting signals.

**Hamming Distance:** number of bits in which two code words differ.

So, Hamming distance  $H(101010, 110010) = 2$ , and can be computed by taking a bit-wise XOR and counting the number of 1's.

# Theory (2)

The TRICK, encode  $m$ -bit words in  $n$ -bit code words, so that the **minimal** Hamming distance between any two code words ( $d$ ) is **maximal**.



$2^m$  code words  
in a  $2^n$  space,  
So  $r$  determines the amount of  
Space between the code words

# Theory (3)

**Theorem:** If the minimal Hamming distance between any two code words is  $d$ , then all  $(d-1)$  bit errors can be detected and any  $\text{ceiling}(d/2) - 1$  bit errors can be corrected.

Examples:

Odd/Even parity: Hamming distance is 2. So 1 bit error detection possible but NO correction.

00 → 00000 00000

01 → 00000 11111

10 → 11111 00000

11 → 11111 11111

Hamming distance = 5, so  
4-bit error detection and 2-bit error correction

# Cyclic Redundance Check (CRC)

“a Practical Error Detection Test”

Makes use of **polynomial codes**:

$$101001 \rightarrow X^5 + X^3 + X^0 = X^5 + X^3 + 1$$

Arithmetic is performed MOD 2 or by XOR

E.g.

$$\begin{array}{r} X^5 + X^3 + 1 \\ X^4 + X^3 + 1 \quad + \\ \hline X^5 + X^4 \end{array}$$

So, addition and subtraction are the same !!!!!!!!!!!

# Cyclic Redundance Check (CRC)

## The algorithm

Sender and Receiver agree on a Generator Polynomial:

$$G[X], \text{ eg. } X^4 + X + 1$$

For sending  $M[X]$ :

1. If  $r$  is the degree of  $G[X]$  ( $r = 4$ ). Add  $r$  (low order) bits to  $M[X]$ , in other words:  $X^r.M[X]$
2. Divide  $X^r.M[X]$  by  $G[X]$  using MOD 2 arithmetic
3. Subtract the remainder ( $\leq r$  bits) from  $X^r.M[X] = T[X]$
4. Transmit  $T[X]$
5. Receiver checks whether  $T[X]$  is divisible by  $G[X]$

# Cyclic Redundance Check (CRC)

## Detection of single bit errors

**WHAT IF RECEIVED  $T[X]$  IS NOT DIVISIBLE BY  $G[X]$**

Then  $T[X] + E[X]$  is received with

$E[X]$  having 1's where a bit error occurred

And the remainder of  $(T[X] + E[X])/G[X] = E[X]/G[X]$

**Lemma 1:** If  $E[X] = X^i$  and  $G[X]$  has two or more components, then  $G[X] \nmid E[X]$ .

**Proof:** Suppose  $G[X] \mid E[X]$ , then  $E[X] = G[X] \cdot P[X] = (X^m + X^n + \dots) \cdot P[X]$ , with  $X^m$  representing the largest term in  $G[X]$  and  $X^n$  representing the smallest term in  $G[X]$ . Then  $m \neq n$ ,  $m > n$ . Let  $X^k$  be the largest component in  $P[X]$  and  $X^l$  the smallest component in  $P[X]$  (so  $k \geq l$ ). Then  $X^{m+k}$  as well as  $X^{n+l}$  belongs to  $E[X]$ , with  $m + k \neq n + l$ . Contradiction !!!!!  
QED

So if  $|G[X]| \geq 2$ , all single bit errors are detected.

# Cyclic Redundance Check (CRC)

## Detection of double bit errors

**Lemma 2:** If  $E[X] = X^i + X^j$  with  $i > j$ ,  $X \nmid G[X]$ , and  $G[X] \nmid X^k + 1$  for all  $k \leq M$  with  $M$  the maximum possible difference between  $i$  and  $j$ , then  $G[X] \nmid E[X]$

**Proof:** Write  $E[X] = X^j (X^{i-j} + 1)$ . Assume  $G[X] \mid E[X]$ , then  $G[X] = P[X].Q[X]$  with  $P[X] \neq 1$ ,  $P[X] \mid X^j$  and  $Q[X] \mid (X^{i-j} + 1)$ . So,  $P[X] = X^m$  for some  $m \neq 0$ . Thus  $X \mid P[X]$ . Contradiction. QED

So if  $X \nmid G[X]$ , and  $G[X] \nmid X^k + 1$  for all  $k \leq M$ , then all double bit errors are detected.

Example:  $X^{15} + X^{14} + 1$  is not a divider of  $X^k + 1$ , for all  $k < 32768$



# Intermezzo

Why was in the previous proof:

$$G[X] \mid E[X] \Rightarrow G[X] = P[X].Q[X] \text{ with} \\ P[X] \mid X^j \text{ and } Q[X] \mid (X^{i-j} + 1)$$

Polynomial Rings and therefore also Polynomial Rings over a Field  $GF(2)$  (“Galois Field”) also called Polynomial Codes or Binary Polynomials, have the property that:

**any  $P[X]$  can be uniquely factored into prime polynomials.**

Binary Prime Polynomials:  $X$   $X+1$   $X^2+X+1$   $X^3+X+1$   $X^3+X^2+1$   
 $X^4+X^3+1$   $X^4+X+1$   $X^4+X^3+X^2+X+1$   $X^5$ .....

# Cyclic Redundance Check (CRC)

## Detection of odd number of bit errors

**Lemma 3:** If  $|E[X]|$  is odd and  $(X + 1) \mid G[X]$ , then  
 $G[X] \nmid E[X]$

**Proof:** Assume  $G[X] \mid E[X]$ . Then, because  $(X+1) \mid G[X]$ ,  $(X+1) \mid E[X]$ .  
So  $E[X] = (X + 1) \cdot Q[X]$ . So  $E[1] = 0$ . However,  $|E[X]|$  is odd, so  $E[1] = 1$ .  
Contradiction. QED

So, if  $(X + 1) \mid G[X]$ , then any odd number of bit errors is detected

# Cyclic Redundance Check (CRC)

## Burst Errors

**Lemma 4:** If there is a burst error of length  $k$  and  $1$  is part of  $G[X]$  and  $k-1 < \text{degree}(G[x])$ , then  $G[X] \nmid E[X]$

**Proof:** Write  $E[X] = X^i(X^{k-1} + X^{k-2} + \dots + 1)$ . So a burst error starting at bit  $i$  and of length  $k$ . Assume  $G[X] \mid E[X]$ . Then, because  $1$  is part of  $G[X]$ , there is no  $P[X] \mid G[X]$  such that  $P[X] \mid X^i$ .

So  $G[X] \mid (X^{k-1} + X^{k-2} + \dots + 1)$ , but this is in contradiction with the fact that  $\text{degree}(G[X]) > k-1$ . QED

So, if  $1$  is part of  $G[X]$  and  $\text{degree}(G[X]) > k-1$ , then any burst of length  $\leq k$  is detected.

# Cyclic Redundance Check (CRC)

## Summary

### Favorable Conditions

$$|G[X]| \geq 2$$

$X \nmid G[X]$ , and  $G[X] \nmid X^k + 1$  for all  $k \leq M$

$$(X + 1) \mid G[X]$$

$1$  is part of  $G[X]$  and  $\text{degree}(G[X]) > k-1$

### Some Standards

$$\text{CRC-12: } X^{12} + X^{11} + X^3 + X^2 + X + 1$$

$$\text{CRC-16: } X^{16} + X^{15} + X^2 + 1$$

$$\text{CRC-CCITT: } X^{16} + X^{12} + X^5 + 1$$

$$\text{CRC-32: } X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1. \text{ Used in ppp !!!!!!!!}$$