# L02 - Protocol Principles

## Protocol Principles

Layering, CL versus CO Service, ARQ Techniques,
Sequence Numbers, Windowing, Flow Control

## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview

© 2007, D.I. Manfred Lindner 

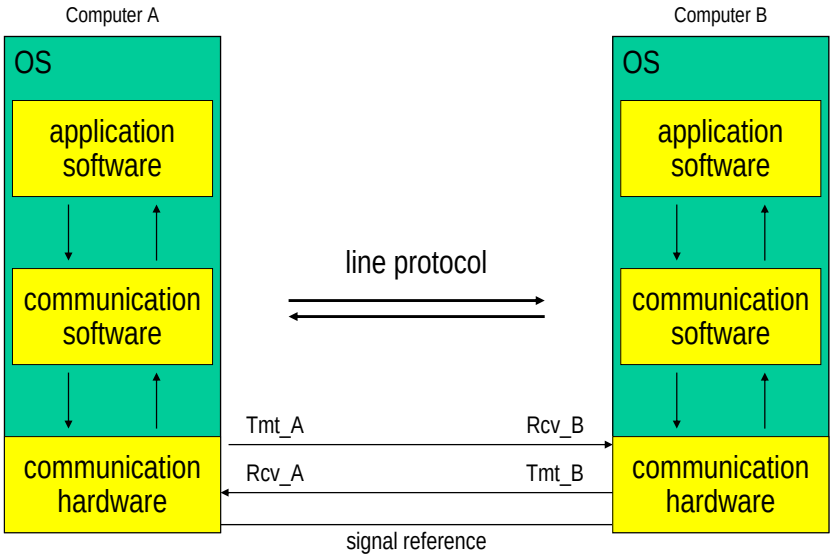
© 2007, D.I. Manfred Lindner

## Line Protocols

- **line protocols regulate and control communication between two devices over point-to-point line**
- **basic elements**
  - frame synchronization
  - frame protection
  - error detection
  - usually implemented in hardware
- **optional elements**
  - connection and line management
  - error recovery
  - flow control
  - usually implemented in software

© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 3

## 3 Layer Model




© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 4

© 2007, D.I. Manfred Lindner

## Software Aspects

- **application software uses**
  - the communication software (normally part of an operating system , OS) in order to exchange data
- **mailbox and queueing techniques**
  - allow cooperation of application and communication software within a computer system
- **the communication software**
  - uses a line protocol for peer to peer communication
    - virtual communication relationship on a given layer
  - hides the details of line protocols and other related tasks from the application software
- **procedural approach**



## Cooperation of Software Layers

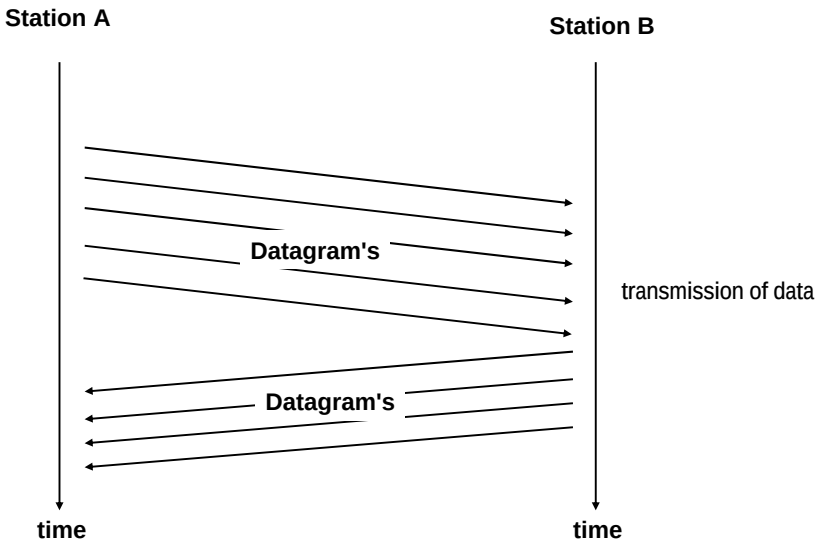
- **if information has to be transmitted from A to B**
  - the application SW of device A forwards some data blocks to the communication SW
  - the communication SW transmits the data using the communication hardware and the line protocol
  - the communication SW of device B receives the data and forwards it to the application SW
- **that means, the communication SW provides a service for the application SW**
  - this service can be

    connection-less or connection-oriented



© 2007, D.I. Manfred Lindner

## Connectionless Service





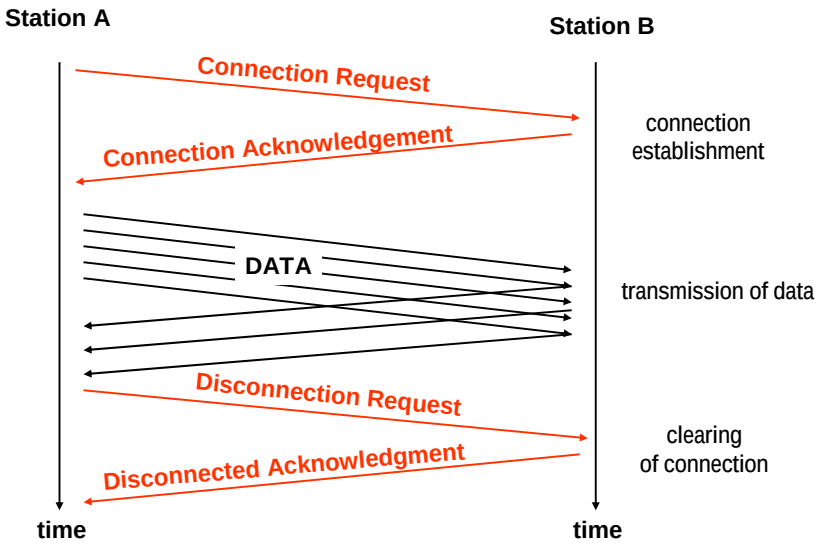

## Line Protocol Services - CL

- **Connection-Less (CL) - type of service**
  - communication SW uses only basic elements (frame synchronization, frame protection, error detection) to transmit data blocks
  - transmission errors causes receiver to discard data blocks
  - best effort service
  - no special frame types are necessary to implement this protocol strategy
  - low implementation requirements for communication SW
  - but error recovery (correction of errors) must be done by application



© 2007, D.I. Manfred Lindner

## Phases with Connection-Oriented Service






## Line Protocol Services - CO

- **Connection-Oriented (CO) - type of service**
  - a communication channel must be established before data blocks can be transmitted
    - logical connection
  - transmission errors will be detected and corrected by the communication SW using feedback error control
    - retransmission of corrupted data blocks
    - **Automatic Repeat reQuest (ARQ) method**
  - reliable transmission service for application SW
    - error recovery done by communication SW
  - special frame types are necessary (connect, disconnect)
  - more sophisticated communication SW is necessary in order to implement ARQ strategy



© 2007, D.I. Manfred Lindner

## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview



## Automatic Repeat Request (ARQ)

- **correct receipt of each transmitted data frame is acknowledged by the receiver**
  - special control message (ACK) in opposite direction
- **each data frame transmitted is stored in a retransmission buffer until receipt of corresponding acknowledgement**
- **if acknowledgement is not received, data frame will be retransmitted after a timeout**
- **identifiers (N, N+1, ...) are necessary to mark the sequence of data frames and to recognize duplicate frames**



© 2007, D.I. Manfred Lindner

## ARQ Variants

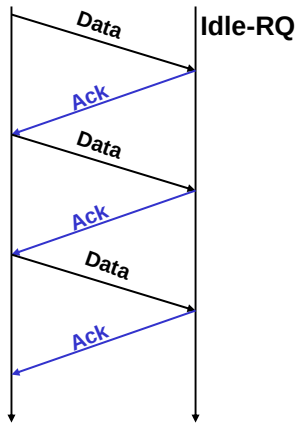
**Idle-RQ**

**Continuous-RQ**

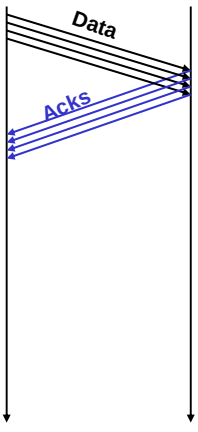
- **Selective ACK (SACK)**
- **GoBackN**
- **Positive ACK**
- **Selective Reject (SREJ)**



## Idle-RQ versus Continuous-RQ




half duplex protocol




**Continuous-RQ**

**Data and Acks are sent continuously !!**

full duplex protocol



© 2007, D.I. Manfred Lindner

## Idle-RQ and Continuous-RQ Facts

- **Idle-RQ**
  - old and slow method
    - but small code and only little resources necessary
    - even today used e.g. TFTP (Trivial File Transfer Protocol)
  - half duplex protocol
- **Continuous-RQ**
  - requires *dramatically* more resources than Idle-RQ or connectionless protocols !!!
    - Retransmission Timers
    - Retransmission Buffers
    - Receive Buffers
  - might result in high CPU loads !!!
  - full duplex protocol



## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview



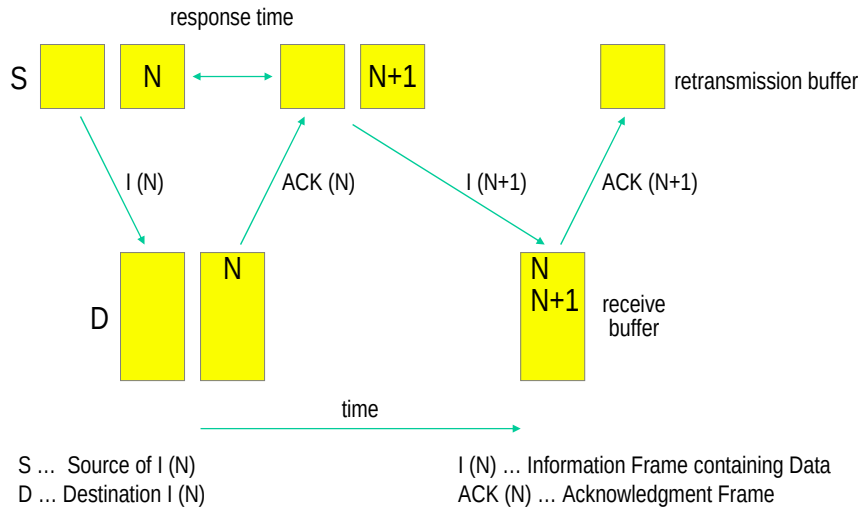
© 2007, D.I. Manfred Lindner

## Idle-RQ

- **simple ARQ implementation**
  - stop & wait protocol
  - device waits for the acknowledgement (ACK) before sending the next data frame
  - basic method can be improved by NACK
- **two identifiers are necessary (0, 1)**
  - distinction between new data frame or duplicate frame
- **numbering of data frames**
  - modulo 2
- **half duplex protocol**
- **full duplex lines can not be efficiently used**

© 2007, D.I. Manfred Lindner 

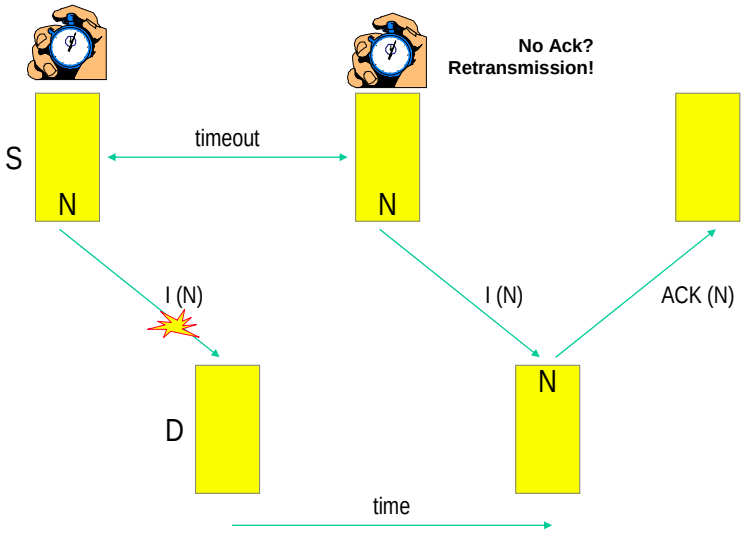
## Basic Sequence of Idle-RQ




© 2007, D.I. Manfred Lindner 

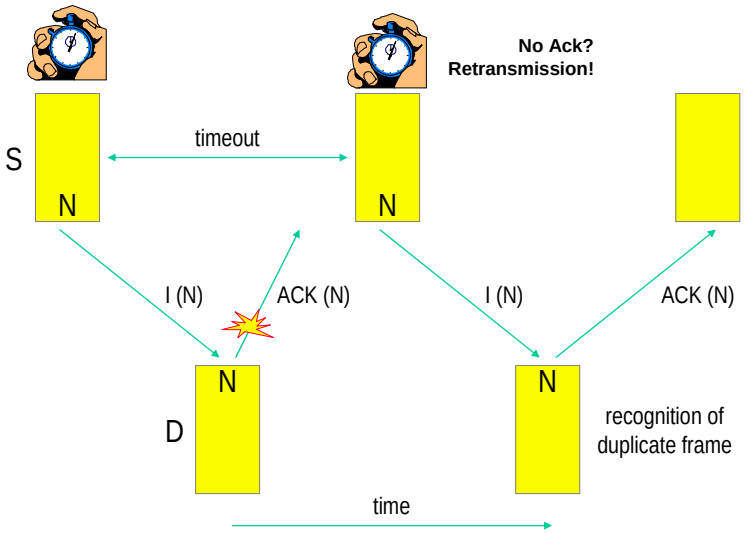
© 2007, D.I. Manfred Lindner

## Idle-RQ Retransmission 1





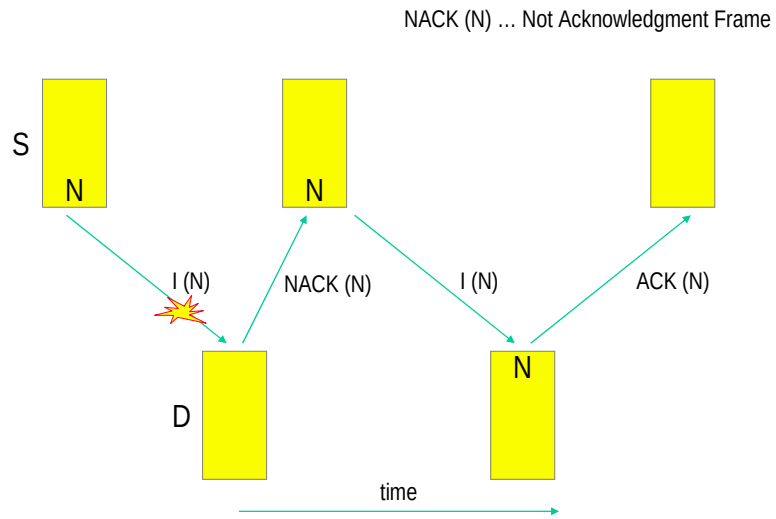

## Idle-RQ Retransmission 2






© 2007, D.I. Manfred Lindner

## Idle-RQ Retransmission with optional NACK






## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview



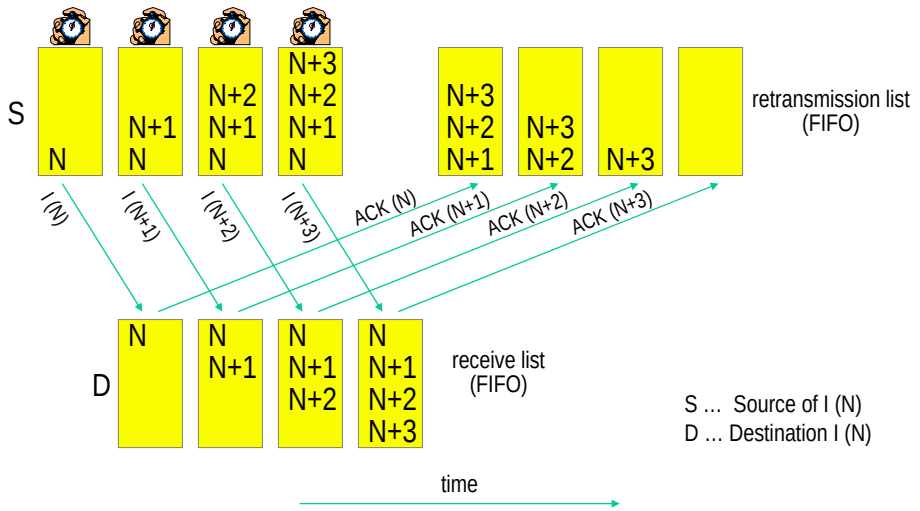
© 2007, D.I. Manfred Lindner

## Continuous-RQ

- **in order to use full duplex lines more efficiently, device does not wait for acknowledgements for frames already sent**
  - Continuous Repeat reQuest (C-RQ) protocols
- **full duplex protocol**
- **until receipt of acknowledgments, data frames are buffered in a retransmission list**
- **each incoming acknowledgment removes the corresponding data frame from that list**
- **receiver stores data frames in receive list**
  - to detect duplicates
  - to reorder the sequence

© 2007, D.I. Manfred Lindner 

## Sequence of Continuous-RQ




© 2007, D.I. Manfred Lindner 

© 2007, D.I. Manfred Lindner

## Error Control Variants with Continuous-RQ

- **several methods for error control**
  - based on selective acknowledgement
    - selective retransmission done implicitly
    - order of frames is not maintained by the procedure
    - e.g. TCP (Transmission Control Protocol) SACK option
  - based on multiple and negative acknowledgement
    - also known as GoBackN
    - order of frames is maintained by the procedure
    - e.g. HDLC (High Level Data Link Control) check pointing technique and REJ option
    - e.g. DDCMP (Digital Data Link Control Management Protocol)



## Error Control Variants with Continuous-RQ

- **several methods for error control (cont.)**
  - based on timeout and positive acknowledgement
    - order of frames is not maintained by the procedure
    - e.g. TCP´s original procedure (early TCP)
    - note: today's TCP uses additionally duplicate ACK´s to signal the rate of frames leaving the network to the sender
  - based on selective reject
    - selective retransmission done explicitly
    - order of frames is not maintained by the procedure
    - e.g. HDLC´s SREJ option



© 2007, D.I. Manfred Lindner

## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview



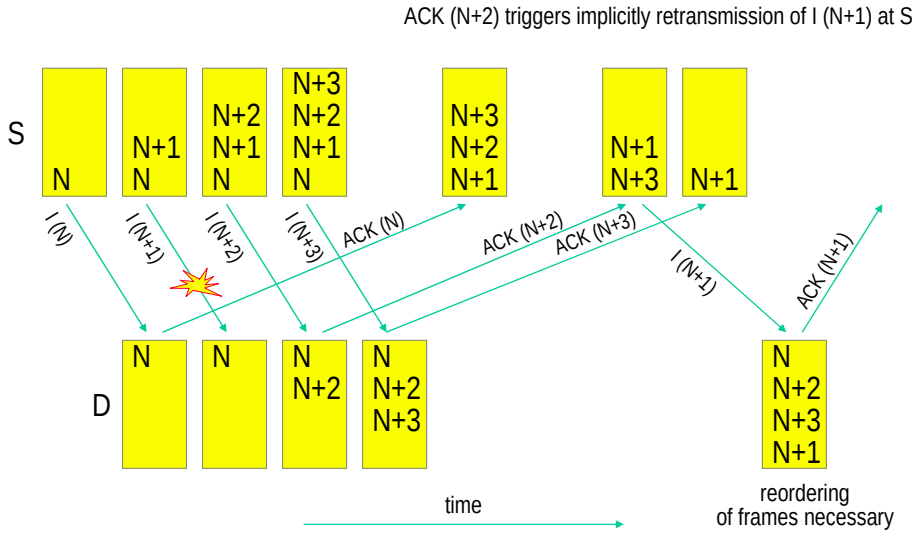
## Selective Acknowledgement

- **principle**
  - every data frame is exclusively confirmed
  - if acknowledgment is not received, corresponding data frame will be sent again and stored at the end of the retransmission list
  - in case of retransmission
    - data frames may not remain in sequence (scenario 1)
    - receiver must recognize duplicate frames and discard them (scenario 2)
  - each transmitted data frame starts an individual timer
    - which will be reset, if acknowledgement is received
    - if timeout occurs data frame is sent once again (scenario 3)



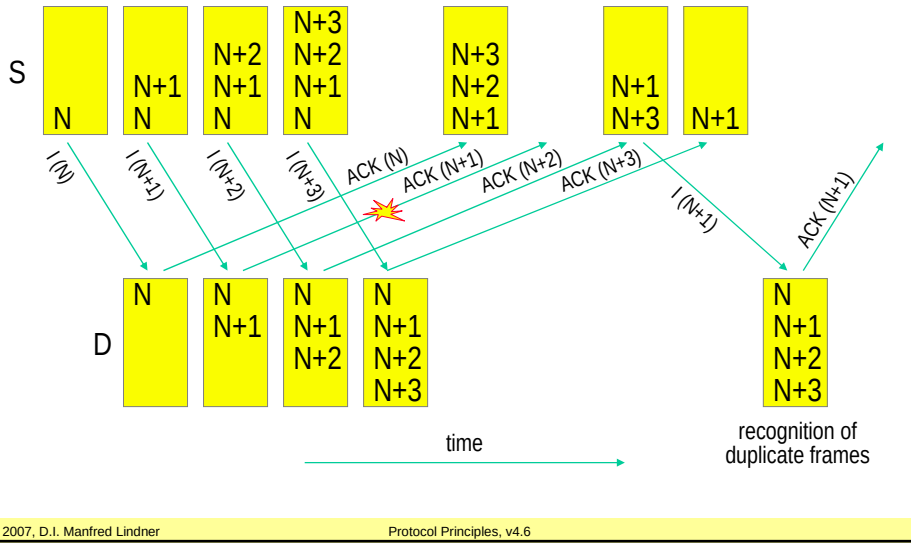
© 2007, D.I. Manfred Lindner

## Selective Acknowledgement 1





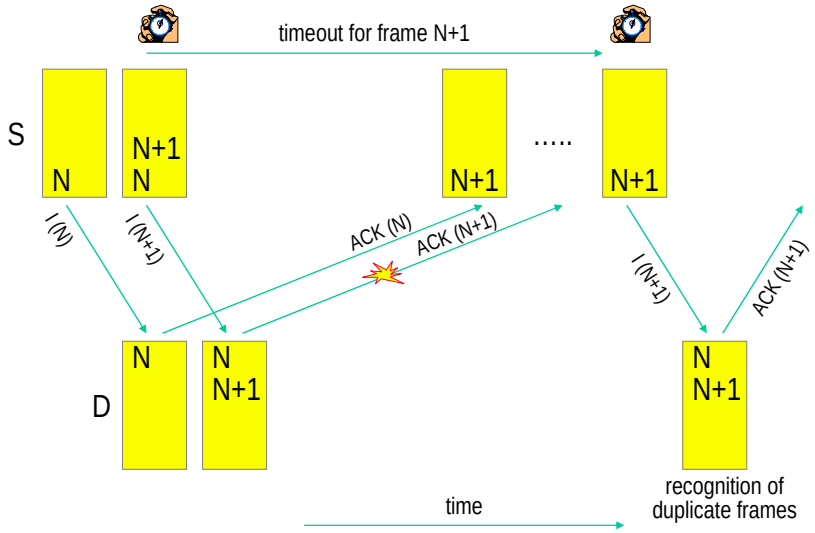

## Selective Acknowledgement 2






© 2007, D.I. Manfred Lindner

## Selective Acknowledgement 3






## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview



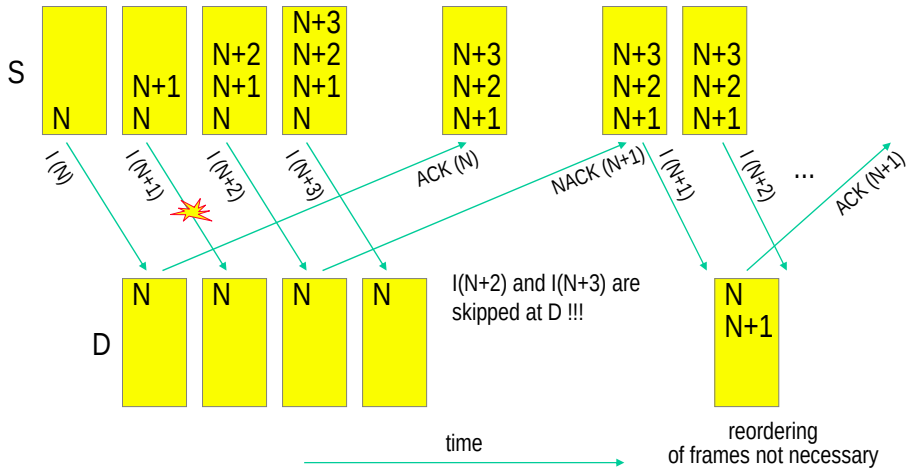
© 2007, D.I. Manfred Lindner

## GoBackN

- **principle**
  - in case of errors, all data frames since “N” will be requested again by NACK(N) (Negative Ack.)
  - all following frames are discarded by receiver until frame with correct sequence number arrives
    - reordering is not necessary in this case
    - SW at receiver could be kept more simple
  - a single acknowledgments could confirm multiple data frames (multiple acknowledgement)
    - often use to spare number of Ack´s in opposite direction
  - each transmitted data frame starts an individual timer
    - which will be reset, if acknowledgement is received
    - if timeout occurs data frame is sent once again (scenario 3)

© 2007, D.I. Manfred Lindner 

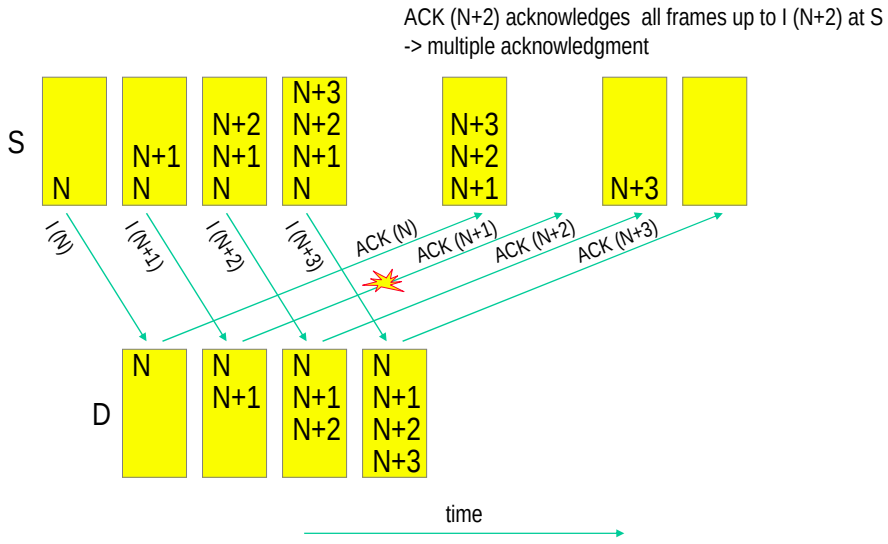
## GoBackN 1




© 2007, D.I. Manfred Lindner 

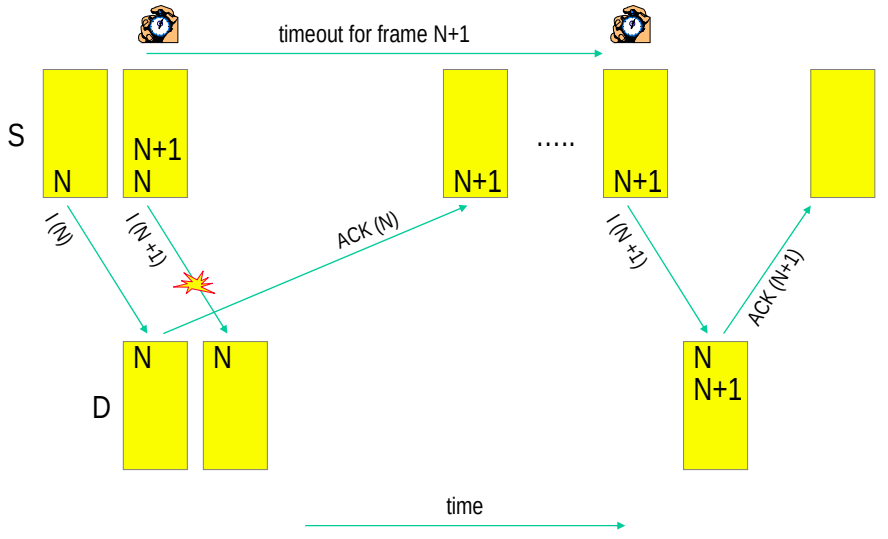
© 2007, D.I. Manfred Lindner

## GoBackN 2




© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 35

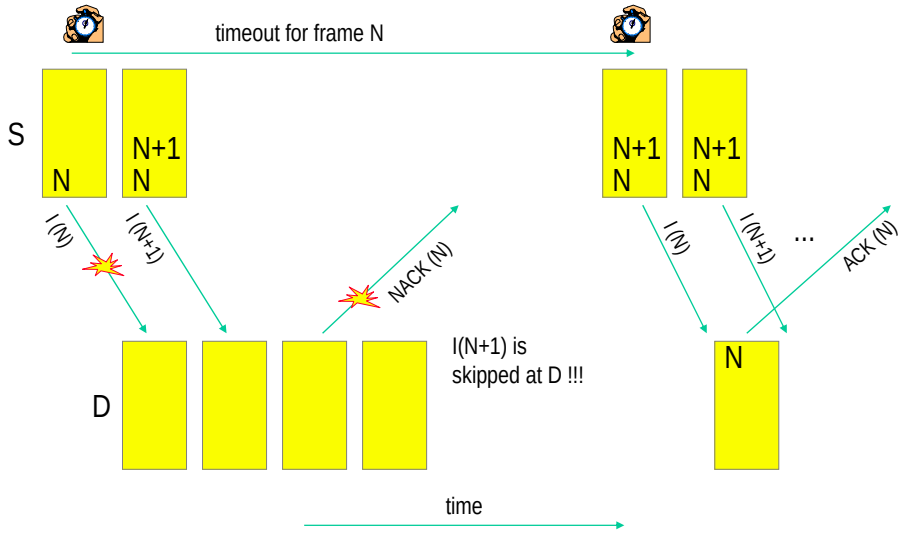
## GoBackN 3




© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 36

© 2007, D.I. Manfred Lindner

## GoBackN 4






## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview



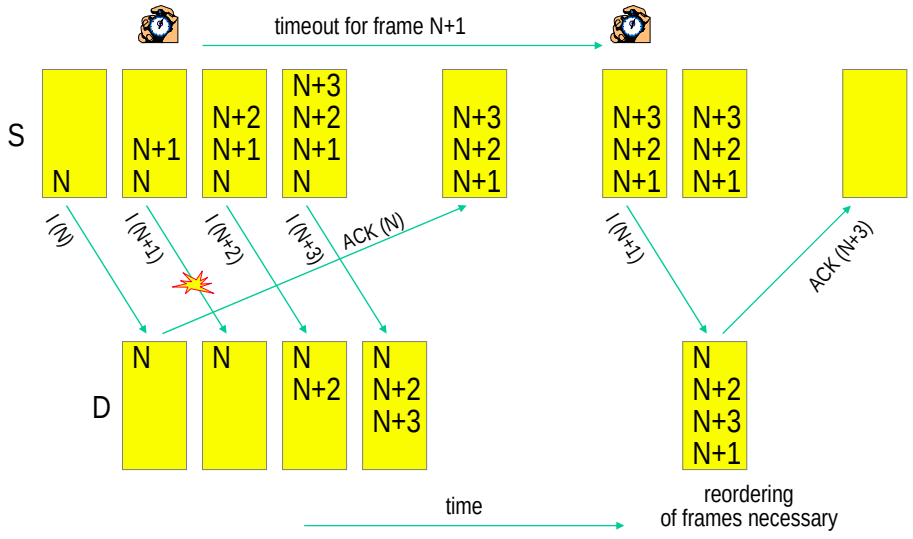
© 2007, D.I. Manfred Lindner

## Positive Acknowledgement

- **principle**
  - data frames will be confirmed as long as frames arrives in sequence
    - multiple acknowledgement can be used by receiver
  - if data frames get out of sequence, confirmation is stopped
  - nevertheless, all following data frames will be stored
  - each transmitted data frame starts an individual timer
    - which will be reset, if acknowledgement is received
    - if timeout occurs data frame is sent once again (scenario 1)
  - data frames which are already stored in the receiver
    - can be confirmed with multiple acknowledgements when missing data frame arrives (scenario 1)

© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 39

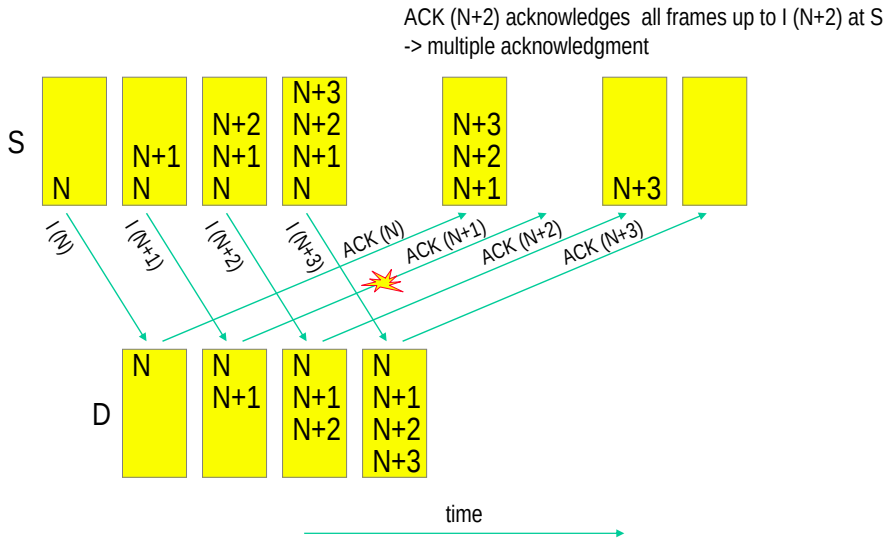
## Positive Acknowledgement 1




© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 40

© 2007, D.I. Manfred Lindner

## Positive Acknowledgement 2






## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview



© 2007, D.I. Manfred Lindner

## Sequence Number

- **identifiers of data frames are implemented by increasing numbers**
  - sequence numbers
  - the number used in I-frames
    - send sequence number N(S)
  - the number used in ACK/NACK/SREJ-frames
    - receive sequence number N(R)
  - register variables are necessary
    - V(S), V(R)
    - must be initialized (set to 0) by connection setup
  - handling of V(S), V(R), N(S), and N(R) will be explained in next slides for GoBackN

© 2007, D.I. Manfred Lindner 

## V(S), V(R) with GoBackN

- **V(S) indicates the sequence number of the next I-frame that will be sent**
- **V(R) indicates the expected sequence number of the next in-sequence I-frame to be received**
  - this value will be seen in N(R)
- **prior to sending an I-frame, the value of N(S) is set to the value of V(S)**
  - afterwards V(S) is increased by one

© 2007, D.I. Manfred Lindner 

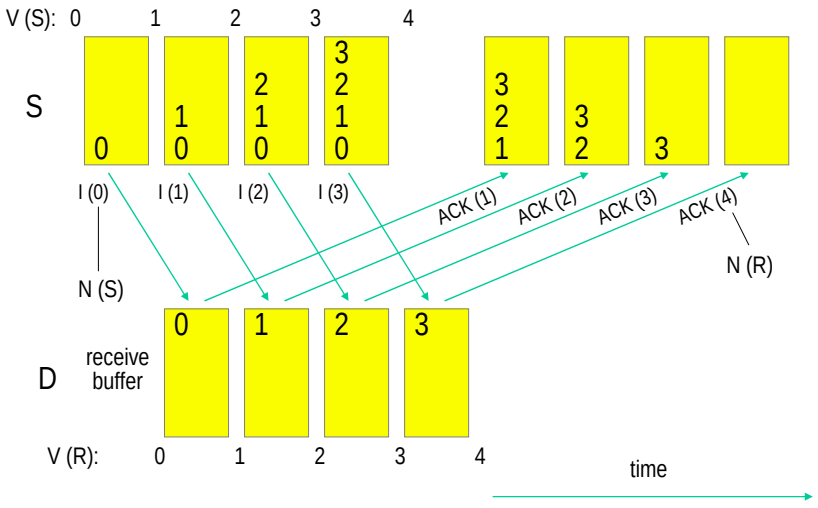
© 2007, D.I. Manfred Lindner

## V(S), V(R) with GoBackN

- **receiver only accepts I-frames with N(S) = V(R)**
  - after successful receipt of a frame V(R) will first be increased by one and then acknowledgment with N(R) = V(R) will be sent
- **therefore receipt of ACK with N(R) = x means**
  - that all I-frames until x-1 are confirmed



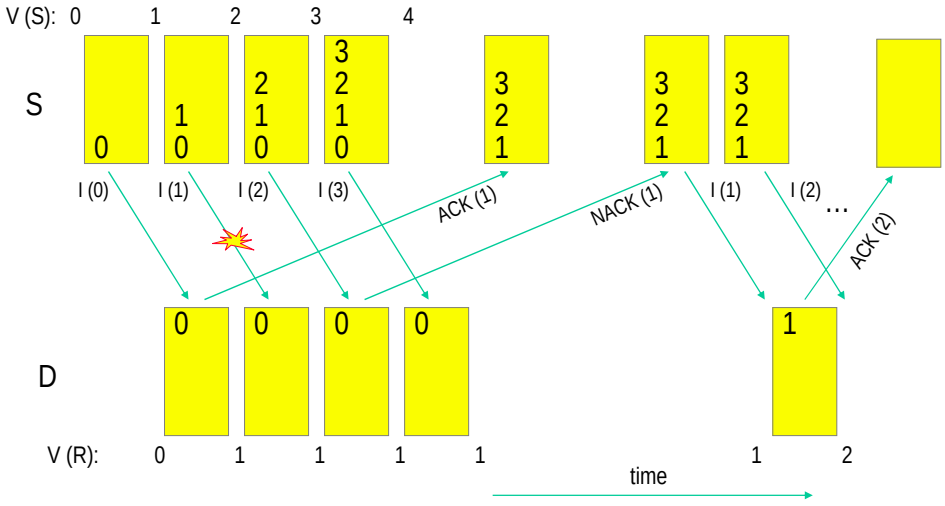
## GoBackN 1





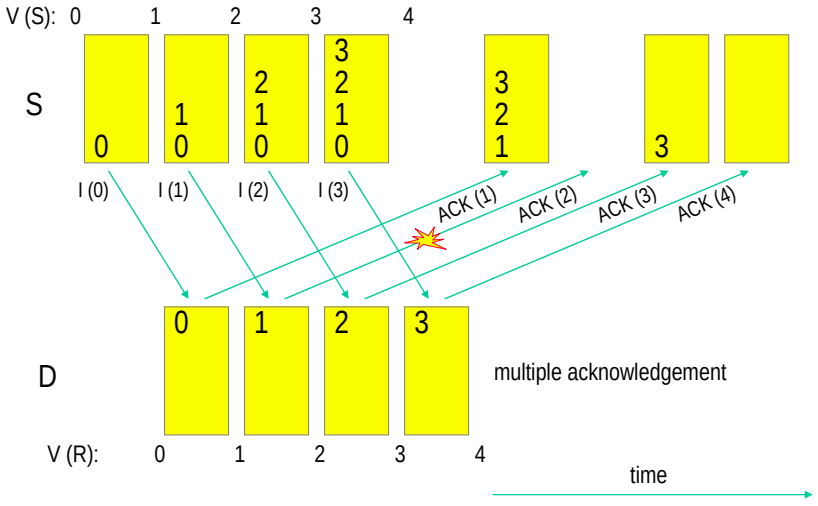

© 2007, D.I. Manfred Lindner

## GoBackN 2




© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 47

## GoBackN 3




© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 48

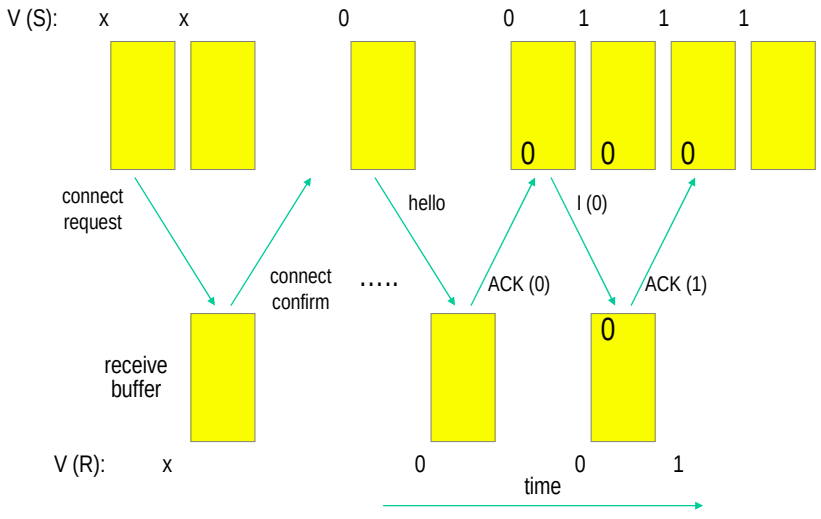
© 2007, D.I. Manfred Lindner

## Initializing / Transmission Pause

- **connection establishment initializes V(S) and V(R)**
- **after connection establishment**
  - connection is maintained by keepalive messages during transmission pauses
- **example for keepalive technique**
  - exchange of HELLO (request / poll) and ACK (response)
  - ACK is used for acknowledgment as well as for connection maintenance
  - because of this an ACK(N) only confirms all frames up to N-1 and not up to N
    - sender can distinguish between a keepalive response and acknowledgement of a new data frame

© 2007, D.I. Manfred Lindner 

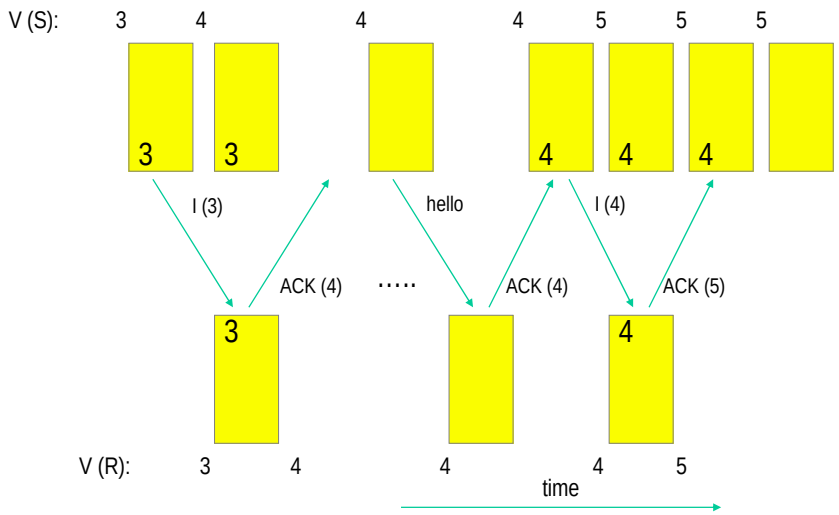
## Initializing / Keepalive




© 2007, D.I. Manfred Lindner 

© 2007, D.I. Manfred Lindner

## Keepalive




© 2007, D.I. Manfred Lindner 

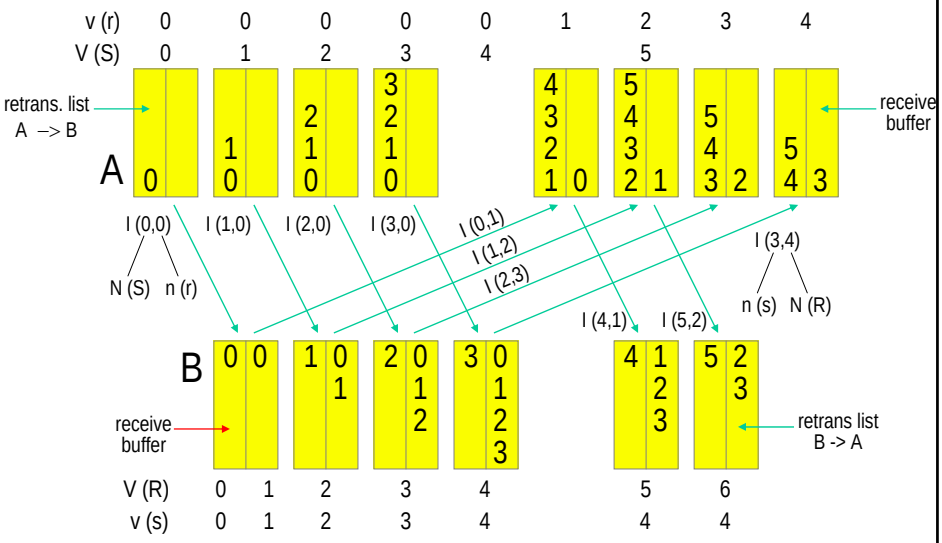
## Piggyback Acknowledgement

- confirmation of every data frame is only appropriate for data flow in one direction
- acknowledgment frames produce unnecessary overhead with full duplex data traffic
- acknowledgments contained in data frames in opposite direction can avoid that overhead
  - piggyback acknowledgement
- if no backward data frame is waiting for transmission
  - ACK frame will be sent still

© 2007, D.I. Manfred Lindner 

© 2007, D.I. Manfred Lindner

## Piggyback Acknowledgement






## Data Flow in both Directions

- data frames contain both send sequence number and receive sequence number of backward direction
- now I-frames and ACK/NACK-frames can arise in both directions
- communication devices must contain both V(S)- and V(R)-registers, retransmission and receive lists
- N(S), N(R), V(S) und V(R) control data transfer from A to B
- n(s), n(r), v(s) und v(r) control data transfer from B to A



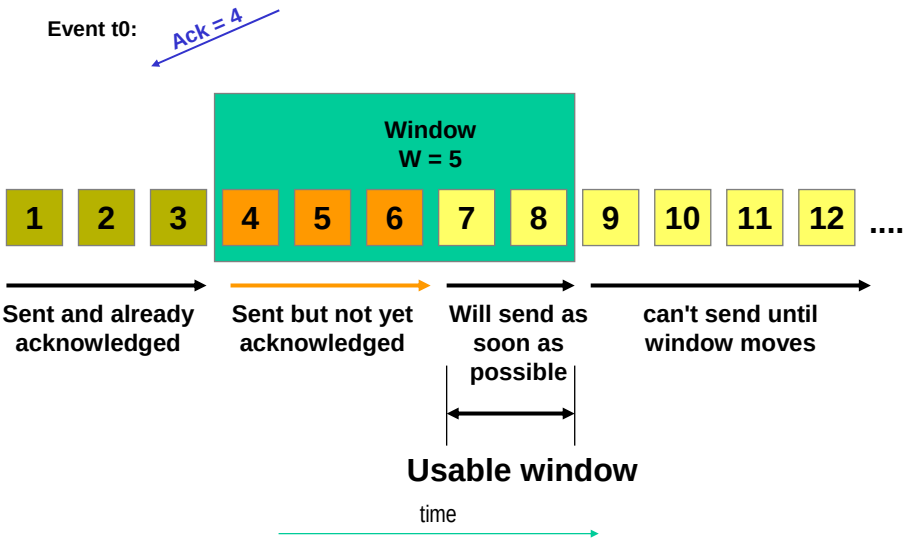
© 2007, D.I. Manfred Lindner

## Windowing

- without a restriction of the number of unconfirmed data frames, continuos-RQ would require infinite number of identifiers and buffer memory
- for that reason, the amount W of data frames stored for retransmission must be limited
  - W = send window
- if limit is reached, sending of additional data frames is stopped until receipt of acknowledgement indicates that window is opened again
  - windowing



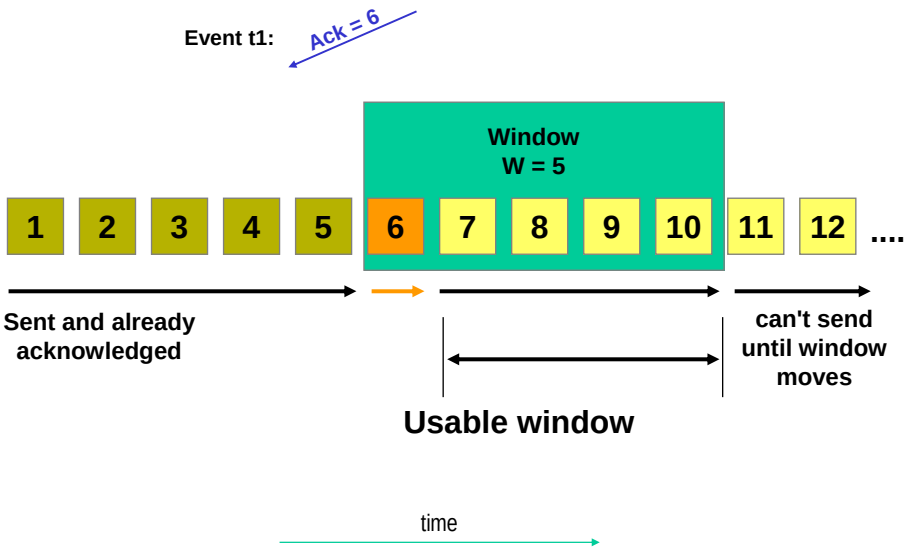
## Principle of Windowing






© 2007, D.I. Manfred Lindner

## Sliding Windowing






## Consequences of Windowing

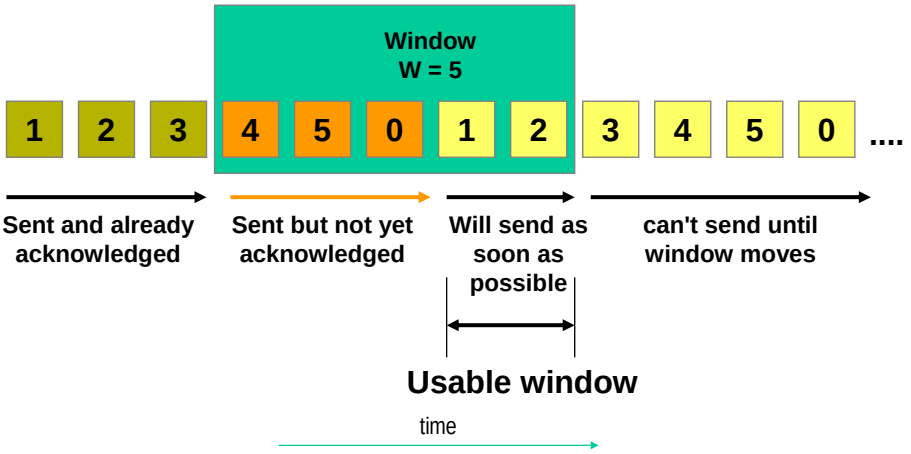
- **windowing reduces the buffer memory for retransmission buffer and receive-list**
  - buffer size of retransmission buffer = W * maximum frame size
- **windowing reduces the number of identifiers**
  - numbering of data frames can be done by a modulo operation
  - e.g. GoBackN with send window = W
    - needs W+1 identifiers because of worst case scenario next slide
    - data frames can be numbered modulo W+1
  - e.g. Selective Acknowledgement with send window = W
    - needs 2W identifiers because of worst case scenario
    - data frames can be numbered modulo 2W



© 2007, D.I. Manfred Lindner

## Windowing with Numbering Modulo W+1

**GoBackN and send-window W means W+1 Identifier and numbering with modulo W+1**




© 2007, D.I. Manfred Lindner 

## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview

© 2007, D.I. Manfred Lindner 

© 2007, D.I. Manfred Lindner

# Time to Transmit a given Number of Bytes

Serialization Delay (in ms) = [ ( Number of Bytes * 8 ) / ( Bitrate in sec ) ] * 1000

| | Bitrate | 9,6 kbit/s | 48 kbit/s | 128 kbit/s | 2,048 Mbit/s | 10 Mbit/s | 100 Mbit/s | 155 Mbit/s | 622 Mbit/s | 1 Gigabit/s |
|---|---|---|---|---|---|---|---|---|---|---|
| | Number of Byte | Delay in msec ($10^{-3}$) | Delay in msec ($10^{-3}$) | Delay in msec ($10^{-3}$) | Delay in msec ($10^{-3}$) | Delay in msec ($10^{-3}$) | Delay in msec ($10^{-3}$) | Delay in msec ($10^{-3}$) | Delay in msec ($10^{-3}$) | Delay in msec ($10^{-3}$) |
| Bit | 0,125 | 0,104167 | 0,020833 | 0,007813 | 0,000488 | 0,000100 | 0,000010 | 0,000006 | 0,000002 | 0,000001 |
| Byte | 1 | 0,833333 | 0,166667 | 0,062500 | 0,003906 | 0,000800 | 0,000080 | 0,000052 | 0,000013 | 0,000008 |
| PCM-30 | 32 | 26,666667 | 5,333333 | 2,000000 | 0,125000 | 0,025600 | 0,002560 | 0,001652 | 0,000412 | 0,000256 |
| ATM cell | 53 | 44,166667 | 8,833333 | 3,312500 | 0,207031 | 0,042400 | 0,004240 | 0,002735 | 0,000682 | 0,000424 |
| Ethernet | 64 | 53,333333 | 10,666667 | 4,000000 | 0,250000 | 0,051200 | 0,005120 | 0,003303 | 0,000823 | 0,000512 |
| X.25 | 256 | 213,333333 | 42,666667 | 16,000000 | 1,000000 | 0,204800 | 0,020480 | 0,013213 | 0,003293 | 0,002048 |
| IP | 576 | 480,000000 | 96,000000 | 36,000000 | 2,250000 | 0,460800 | 0,046080 | 0,029729 | 0,007408 | 0,004608 |
| Ethernet | 1.518 | 1.265,000000 | 253,000000 | 94,875000 | 5,929688 | 1,214400 | 0,121440 | 0,078348 | 0,019524 | 0,012144 |
| FR | 8.192 | 6.826,666667 | 1.365,333333 | 512,000000 | 32,000000 | 6,553600 | 0,655360 | 0,422813 | 0,105363 | 0,065536 |
| TCP | 65.534 | 54.611,666667 | 10.922,333333 | 4.095,875000 | 255,992188 | 52,427200 | 5,242720 | 3,382400 | 0,842881 | 0,524272 |

1kbit/s = 1000 bit/s !!!
1KByte = 1024 Byte !!!



# Propagation (Signal) Delay

Tp = Propagation Delay (in ms) = [ ( Distance in m ) / ( velocity in m/sec ) ] * 1000

| | | v=200.000km/s | v=300.000km/s |
|---|---|---|---|
| | Distance | Delay in msec ($10^{-3}$) | Delay in msec ($10^{-3}$) |
| CPU Bus | 10 cm | 0,0000005 | 0,0000003 |
| | 1 m | 0,0000050 | 0,0000033 |
| RS232, V24/V.28 | 15 m | 0,0000750 | 0,0000500 |
| LAN, Copper, RJ45 | 100 m | 0,0005000 | 0,0003333 |
| LAN, FO, X.21/V.11-V.10 | 1 km | 0,0050000 | 0,0033333 |
| Local Subscriber Line | 2,5 km | 0,0125000 | 0,0083333 |
| WAN Link Repeater | 10 km | 0,0500000 | 0,0333333 |
| WAN Link Repeater | 100 km | 0,5000000 | 0,3333333 |
| WAN FO Link Repeater | 1.000 km | 5,0000000 | 3,3333333 |
| WAN FO Link Repeater | 10.000 km | 50,0000000 | 33,3333333 |
| Satellite Link | 40.000 km | 200,0000000 | 133,3333333 |
| Satellite Link | 50.000 km | 250,0000000 | 166,6666667 |
| | 100.000 km | 500,0000000 | 333,3333333 |
| | 300.000 km | 1500,0000000 | 1000,0000000 |

Total Delay = Serialization Delay + Propagation Delay + (Switching Delay)



© 2007, D.I. Manfred Lindner

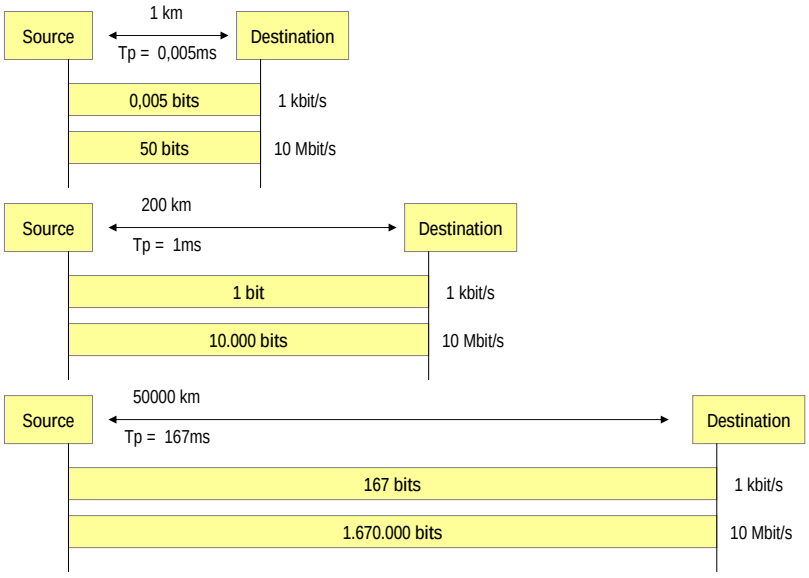
## How Long is a Bit?

Length (in m) = [ ( 1 / ( bitrate per sec) ] * [ ( velocity in m/sec ) ]

| | Bitrate | | Bit Length in meter | Bit Length in meter |
|---|---|---|---|---|
| | | | | |
| Analogue Modem | 9,6 kbit/s | | 20833,33 | 31250,00 |
| Analogue Modem | 48 kbit/s | | 4166,67 | 6250,00 |
| DS0 | 64 kbit/s | | 3125,00 | 4687,50 |
| ISDN (2B) | 128 kbit/s | | 1562,50 | 2343,75 |
| PCM-30, E1 | 2,048 Mbit/s | | 97,66 | 146,48 |
| Token Ring 4 | 4 Mbit/s | | 50,00 | 75,00 |
| Ethernet | 10 Mbit/s | | 20,00 | 30,00 |
| Token Ring16 | 16 Mbit/s | | 12,50 | 18,75 |
| Fast Ethernet, FDDI | 100 Mbit/s | | 2,00 | 3,00 |
| ATM STM1, OC-3 | 155 Mbit/s | | 1,29 | 1,94 |
| ATM STM4, OC-12 | 622 Mbit/s | | 0,32 | 0,48 |
| Gigabit Ethernet | 1 Gigabit/s | | 0,20 | 0,30 |
| OC-48 | 2,5 Gigabit/s | | 0,08 | 0,12 |
| 10 Gigabit Ethernet | 10 Gigabit/s | | 0,02 | 0,03 |
| | | | | |
| | | | Copper | LWL - Free Space |
| | | | 200.000 km /sec | 300.000 km / sec |

© 2007, D.I. Manfred Lindner 

## Propagation Delay and Number of Bits on a given Link




© 2007, D.I. Manfred Lindner 

© 2007, D.I. Manfred Lindner

## How Large should be the Window Size?

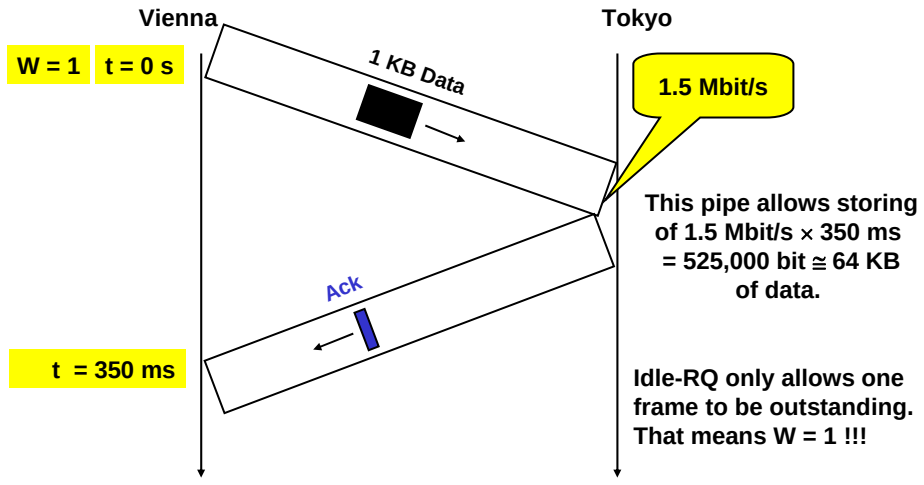
- **choice of window size is determined by**
  - response time ( = round trip time RTT)
    - 2 x (propagation + serialization) delay plus response delay of partner
  - bandwidth (bit rate) of communication channel
  - [available buffer size transmitter/receiver]
- **principle to achieve the optimum:**
  - the sender´s window must be big enough so that the sender can fully utilize the channel volume
  - the channel volume can be expressed by the

**Delay-Bandwidth Product**

  - window size W in bytes = RTT x BW

© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 65

## Nearly Empty Pipe with Idle-RQ and W = 1

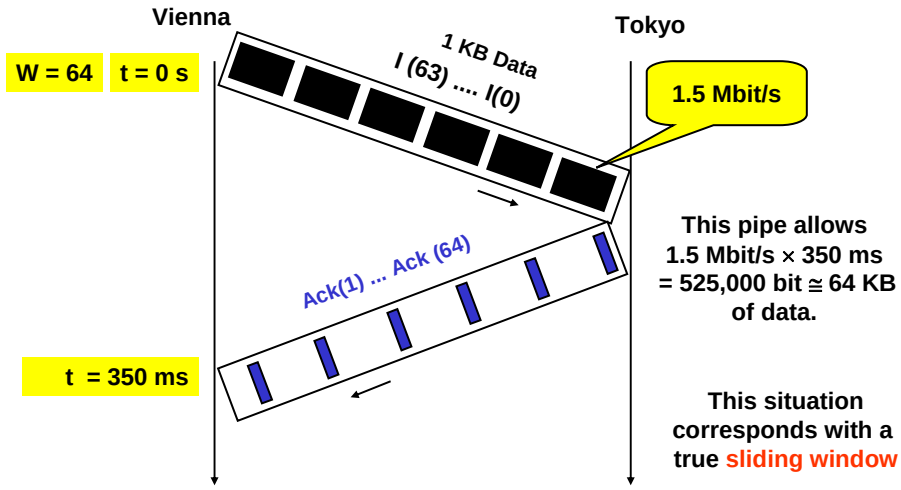



**Assume 1 KByte (1024 byte) maximum frame size, then the maximal achievable rate is (1024 × 8) bit / 0.35 s ≅ 23 kbit/s**

© 2007, D.I. Manfred Lindner Protocol Principles, v4.6 66

© 2007, D.I. Manfred Lindner

## Full Pipe with Continous-RQ and W = 64



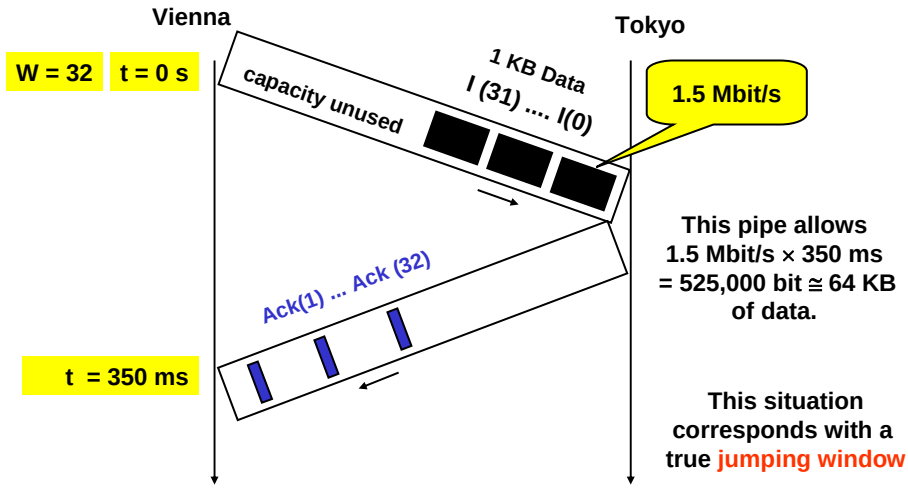

This pipe allows 1.5 Mbit/s × 350 ms = 525,000 bit ≅ 64 KB of data.

This situation corresponds with a true sliding window

Continous-RQ (C-RQ) allows several frames to be outstanding.
With 1 KByte maximum Frame Size optimum would be W = 64 !!!



## Only Half Pipe Used with C-RQ and W = 32




This pipe allows 1.5 Mbit/s × 350 ms = 525,000 bit ≅ 64 KB of data.

This situation corresponds with a true jumping window

With 1 KByte maximum Frame Size and W = 32
the pipe could be used only half the time !!!



© 2007, D.I. Manfred Lindner

## Optimal Window Size - Sliding Window

- **optimal window size (Continuous-RQ)**
  - acknowledgments arrive just in time to keep the window always open
    - sliding window
  - requirement for optimum
    - window size W in bytes in minimum equal to RTT x BW
- **if window size is smaller than RTT x BW**
  - transmission will be stopped until acknowledgments arrive
    - jumping window
  - Idle RQ behaviour in worst case with W = 1
- **if window size is too large**
  - in case of errors many good frames must be retransmitted (see Go Back N)



## Timers - Retransmission Timeout

- **the value for retransmission timeouts with line protocols can be easily calculated using the following parameters**
  - bitrate
  - maximum data frame size
  - worst case time at receiver to generate an acknowledgment
  - size of acknowledgment frame
- **calculation for network protocols with varying transmission delays is more complex**
  - adaptive process is necessary



© 2007, D.I. Manfred Lindner

## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview



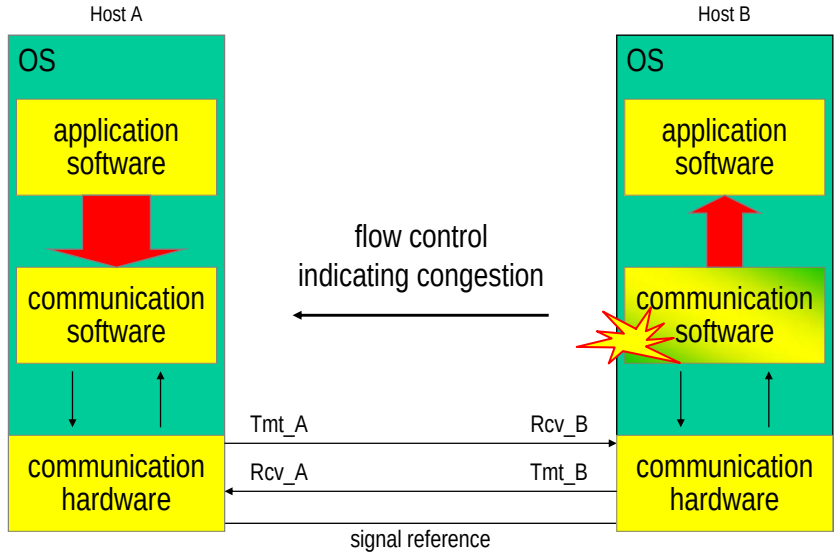
## Flow Control

- **if data frames arrive faster than application is able to process,**
  - receiver runs out of available buffer storage and good frames must be discarded by the receiver
  - discarded data frames will cause retransmission but they will be still discarded because of lack of buffers
- **therefore receiver should control the rate of transmission of data frames**
  - flow control
  - overload/congestion situation indicated to the sender using flow control messages
  - sender stops and waits until receiver is able to process frames again



© 2007, D.I. Manfred Lindner

## Flow Control






## Flow Control

- **windowing could be used to implement flow control**
  - receiver does not generate acknowledgements in case of congestion
  - sender will stop transmission if send window is closed
- **problem with windowing**
  - after timeout unconfirmed frames will be retransmitted
  - after a defined amount of unsuccessful retransmissions, the connection is considered to be broken
- **therefore flow control is based on**
  - separate flow control frames
  - and windowing



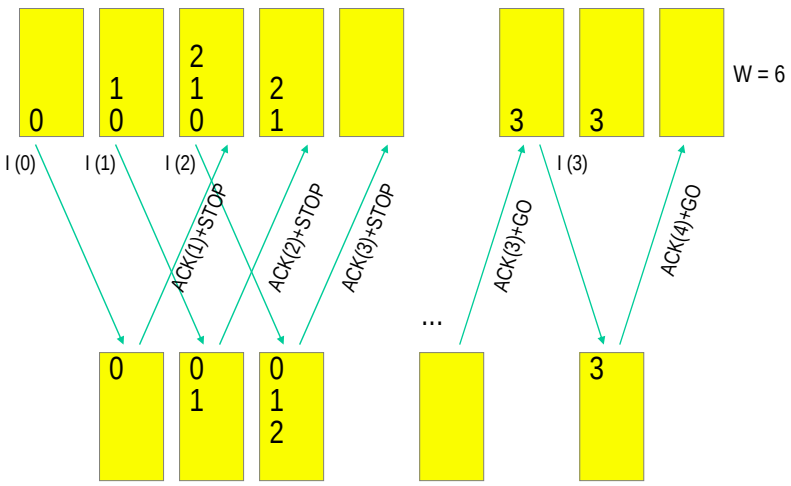
© 2007, D.I. Manfred Lindner

## Flow Control

- **typically used flow control messages**
  - STOP, e.g. HDLC´s Receiver Not Ready (RNR)
  - GO, e.g. HDLC´s Receiver Ready (RR)
- **in case of congestion**
  - receiver signals STOP
  - on receipt of STOP sender will suspend transmitting
  - in the worst case, sender will use the send window before stopping transmission
  - receiver signals GO when data flow can resume
  - STOP and GO may contain N(R) and hence be used for acknowledgement

© 2007, D.I. Manfred Lindner 

## Flow Control




© 2007, D.I. Manfred Lindner 

© 2007, D.I. Manfred Lindner

## Flow Control / Keepalive

- **in case of full duplex data communication**
  - STOP and GO control frames are used for flow control in both directions
- **in some cases STOP and GO frames are further used for connection management**
  - keepalive procedure
  - if no data frames are waiting for transmission a GO frame can signal keepalive to the partner
  - if traffic was suspended by STOP a periodic repetition of STOP can signal keepalive to the partner
  - in both cases keepalives maintain the connection



## Flow Control / Adaptive Windowing

- **window size could be**
  - constant or dynamic during lifetime of a connection
    - constant window size is used e.g. by HDLC, X.25
- **if window size is dynamic**
  - a start value is negotiated during connection establishment
  - actual window size will be dynamically adjusted to an optimal value
    - receiver continuously advertises optimal value (e.g. based on availability free buffer memory)
    - advertised window size = 0 -> STOP
    - advertised window size > 0 -> GO
  - adaptive windowing
    - e.g used by TCP



© 2007, D.I. Manfred Lindner

## Agenda

- **Introduction**
  - 3 Layer Model and Service Types
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
  - Sequence Numbers and Windowing
  - Delay Bandwidth Product
  - Flow Control
  - HDLC Overview



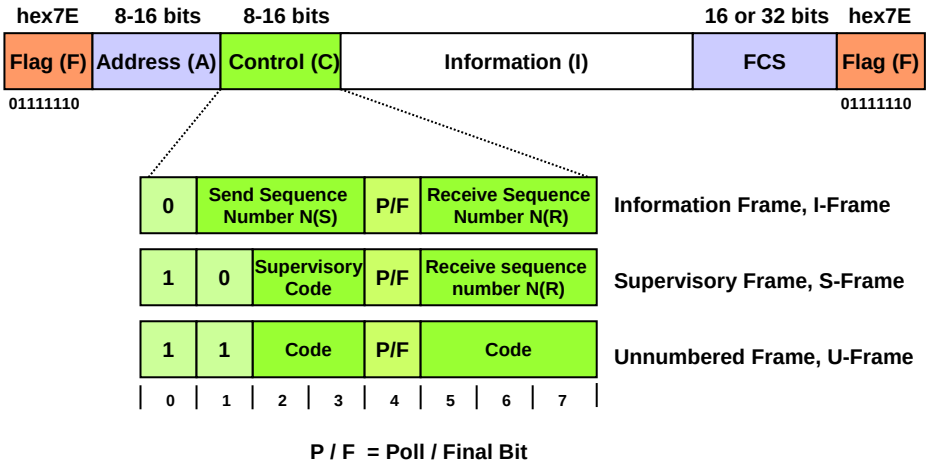
## HDLC

- **High-level Data Link Control**
- **most widely used data link control protocol based on building elements**
  - synchronous transmission
  - bit-oriented line protocol using bitstuffing
  - Continuos RQ with GoBackN, piggybacked ACK
  - P/F procedure (see appendix chapter for details)
- **provides many options**
  - half-duplex and full-duplex transmission (see appendix chapter for details)
  - point-to-point and multipoint configuration (see appendix chapter for details)
  - switched or non-switched channels



© 2007, D.I. Manfred Lindner

## HDLC Frame Format




P / F = Poll / Final Bit



## Supervisory Frames

| | |
|---|---|
| 0 0 | RR (Receiver Ready) = meaning of ACK plus GO |
| 0 1 | REJ (Reject) = meaning of NACK |
| 1 0 | RNR (Receiver Not Ready) = meaning ACK plus STOP |
| 1 1 | SREJ (Selective Reject) |



© 2007, D.I. Manfred Lindner

## HDLC Data Link Services

- **HDLC can provide connection-oriented service**
  - setup of connection done by U-frames
    - SNRM, SARM, SABM, UA
  - I-frames and S-frame can be used only after connection setup
    - I, RR, RNR, REJ, SREJ
  - clearing of a connection done by U-frames
    - DISC, UA
- **HDLC can provide connectionless service**
  - only U-frames can be used
    - UI for data transport

© 2007, D.I. Manfred Lindner 

## Frame-Types

**Connection-Oriented**

| | |
|---|---|
| **I** | **Information** |
| **RR** | **Receiver Ready** |
| **REJ** | **Reject** |
| **RNR** | **Receiver Not Ready** |
| **SREJ** | **Selective Reject** |
| **SNRM** | **Set Normal Response Mode** |
| **SABM** | **Set Async Balanced Mode** |
| **SARM** | **Set Async Response Mode** |
| **SNRME** | **Set NRM Extended Mode** |
| **SABME** | **Set ABM Extended Mode** |
| **SARME** | **Set ARM Extended Mode** |
| **DISC** | **Disconnect** |
| **UA** | **Unnumbered Acknowledge** |
| **RSET** | **Reset** |
| **FRMR** | **Frame Reject** |
| **RD** | **Request Disconnect** |
| **DM** | **Disconnect Mode** |

**Connection-Less**

| | |
|---|---|
| **UI** | **Unnumbered Information** |

**Miscellaneous**

| | |
|---|---|
| **XID** | **Exchange Identification** |
| **UP** | **Unnumbered Poll** |
| **SIM** | **Set Initialization Mode** |
| **RIM** | **Request Initialization Mode** |
| **NR0-3** | **Non-Reserved 0** |

© 2007, D.I. Manfred Lindner 

© 2007, D.I. Manfred Lindner

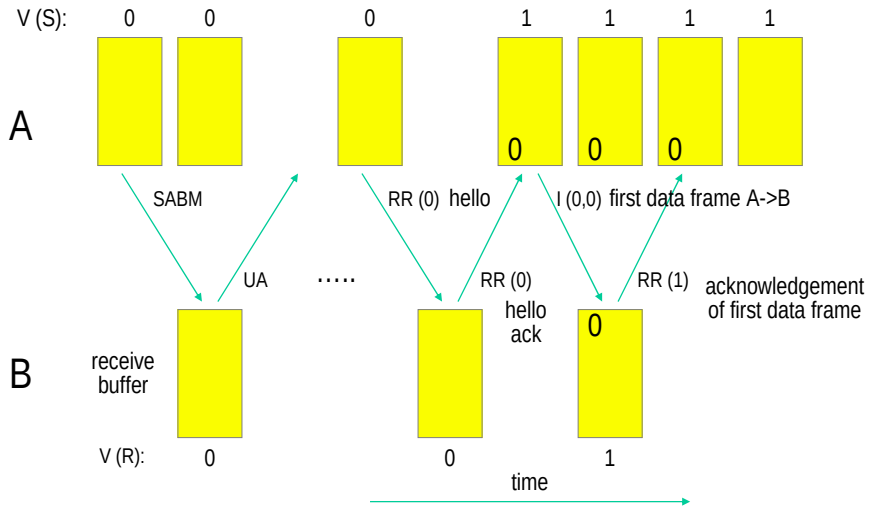
## Unnumbered Frames

| 1 | 1 | Code | P/F | Code |
|---|---|---|---|---|

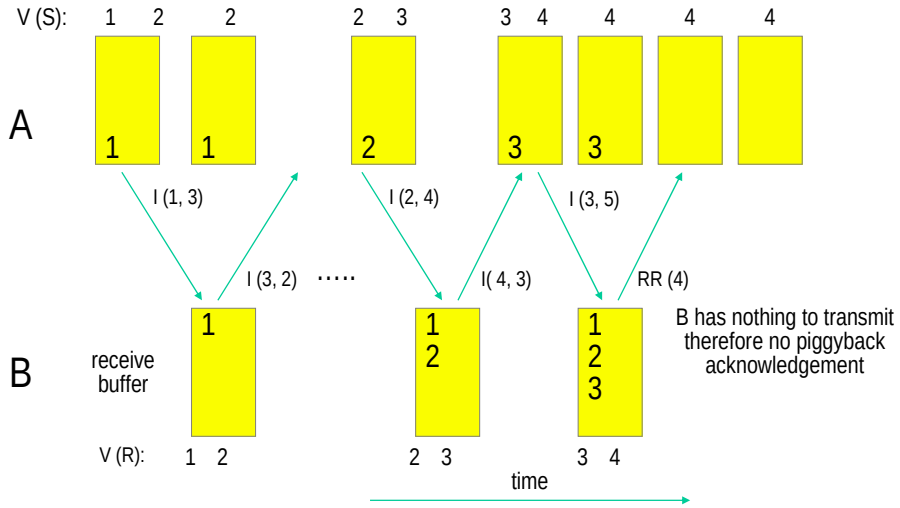
| | | | | | Command | Response |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | UI | UI |
| 0 | 0 | 0 | 0 | 1 | SNRM | |
| 0 | 0 | 0 | 1 | 0 | DISC | RD |
| 0 | 0 | 0 | 1 | 0 | UP | |
| 0 | 0 | 0 | 1 | 1 | | UA |
| 0 | 1 | 0 | 0 | 0 | NR0 | NR0 |
| 0 | 1 | 0 | 0 | 1 | NR1 | NR1 |
| 0 | 1 | 0 | 1 | 0 | NR2 | NR2 |
| 0 | 1 | 0 | 1 | 1 | NR3 | NR3 |
| 1 | 0 | 0 | 0 | 0 | SIM | RIM |
| 1 | 0 | 0 | 0 | 1 | | FRMR |
| 1 | 1 | 0 | 0 | 0 | SARM | DM |
| 1 | 1 | 0 | 0 | 1 | RSET | |
| 1 | 1 | 0 | 1 | 0 | SARME | |
| 1 | 1 | 0 | 1 | 1 | SNRME | |
| 1 | 1 | 1 | 0 | 0 | SABM | |
| 1 | 1 | 1 | 0 | 1 | XID | XID |
| 1 | 1 | 1 | 1 | 0 | SABME | |



## HDLC Example: Initializing / Keepalive / First Data Frame A -> B

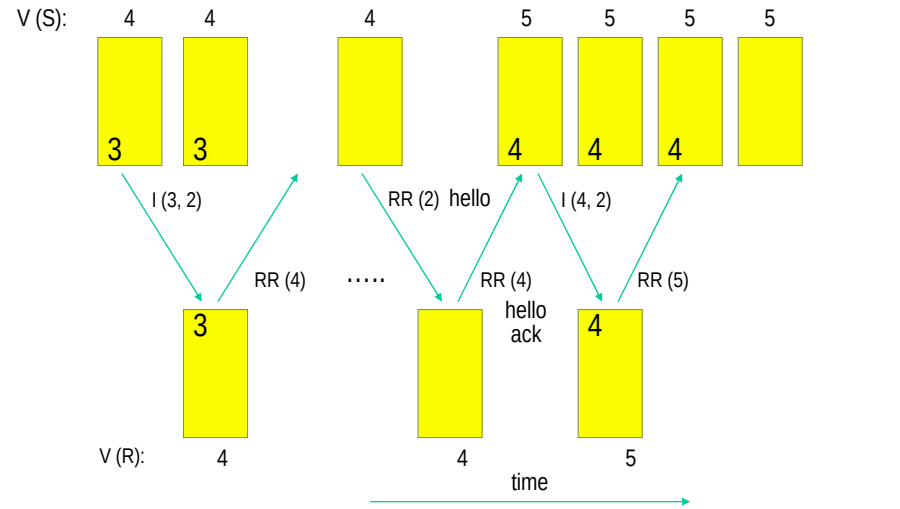


## HDLC Example: Data Frames A->B , B->A with piggyback ACK





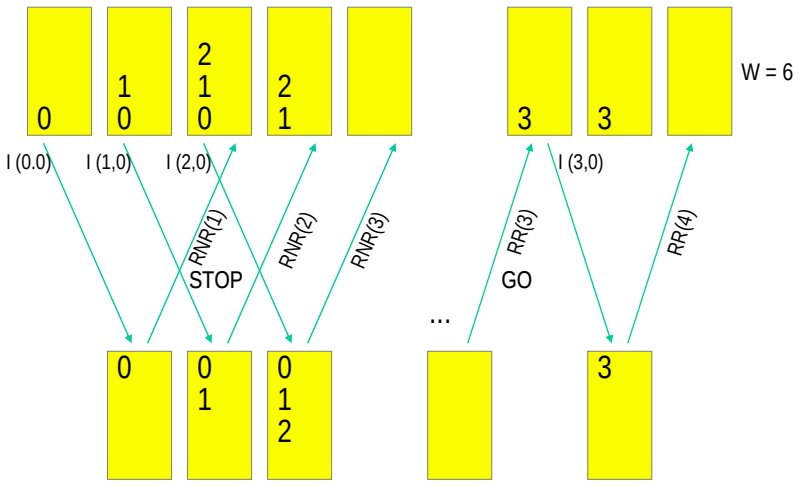

## HDLC Example: Keepalive






© 2007, D.I. Manfred Lindner

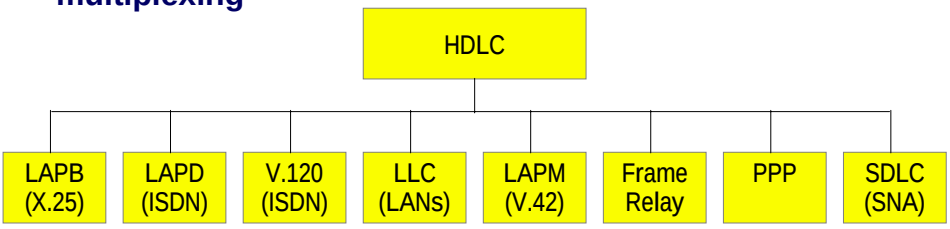
## HDLC Example: Flow Control






## HDLC Family

- **LAPB - Link Access Procedure Balanced**
  - link layer protocol for X.25
- **LAPD - Link Access Procedure D-Channel**
  - ISDN
- **V.120 - used on ISDN terminal adapters for multiplexing**
- **LAPM - Link Access Procedure for Modems**
- **PPP - Point-to-Point Protocol**
  - encapsulates network PDUs and identifies protocol type
- **SDLC - Synchronous Data Link Control (IBM)**






© 2007, D.I. Manfred Lindner