Protocol Principles

Layering, CL vs. CO Protocols, Best-Effort vs. Reliable Services
ARQ Techniques, Sequence Numbers, Windowing,
Bandwidth-Delay Product, Flow Control, HDLC

Agenda

- **<u>Introduction</u>**
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - Flow Control
  - HDLC Overview

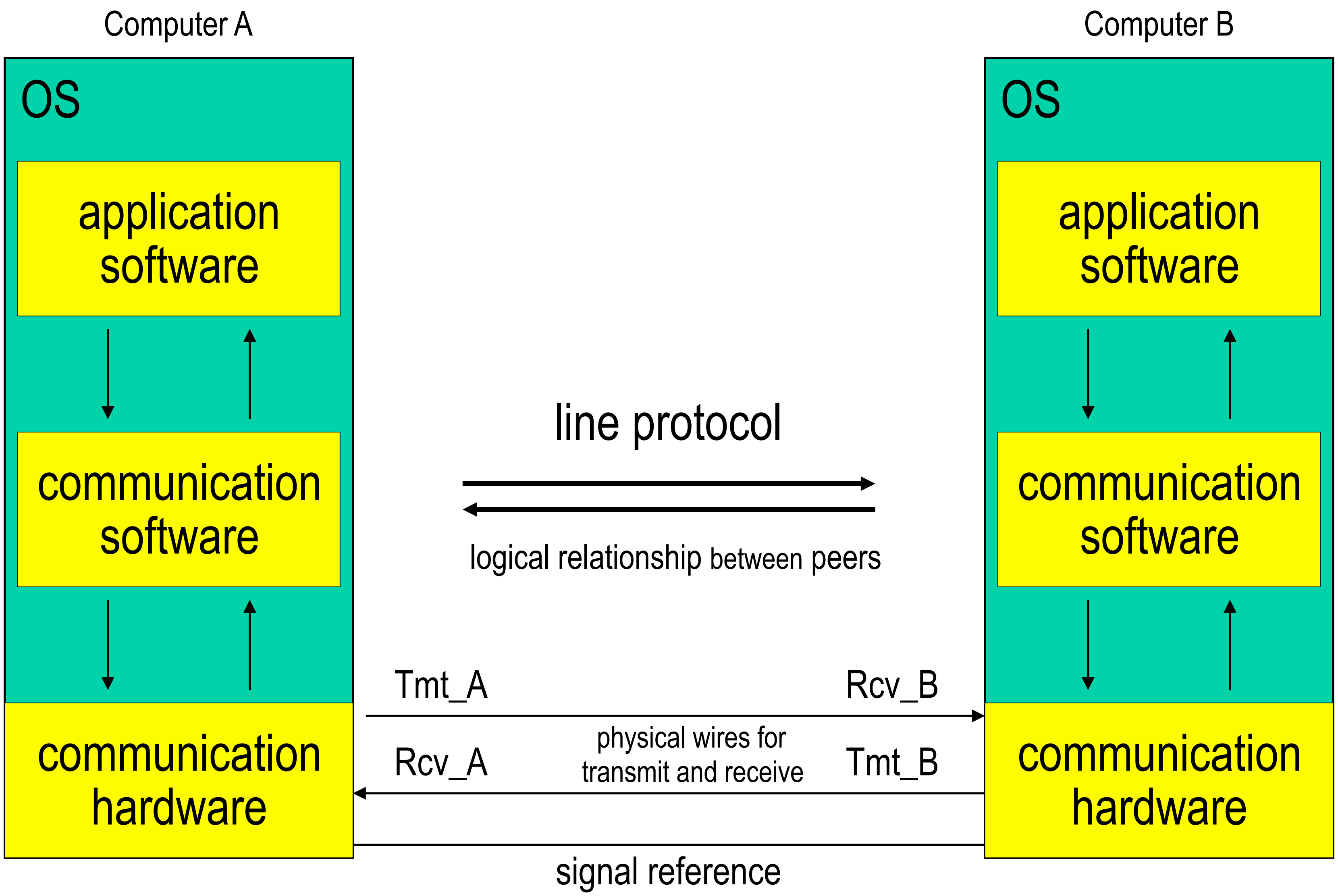
Line Protocols

- **Line protocols regulate and control communication between two devices over a point-to-point line**
- **Basic elements**
  - Frame synchronization
  - Frame protection
  - Error detection
  - Usually implemented in hardware
- **Optional elements**
  - Connection and line management
  - Addressing
  - Error recovery
  - Flow control
  - Usually implemented in software

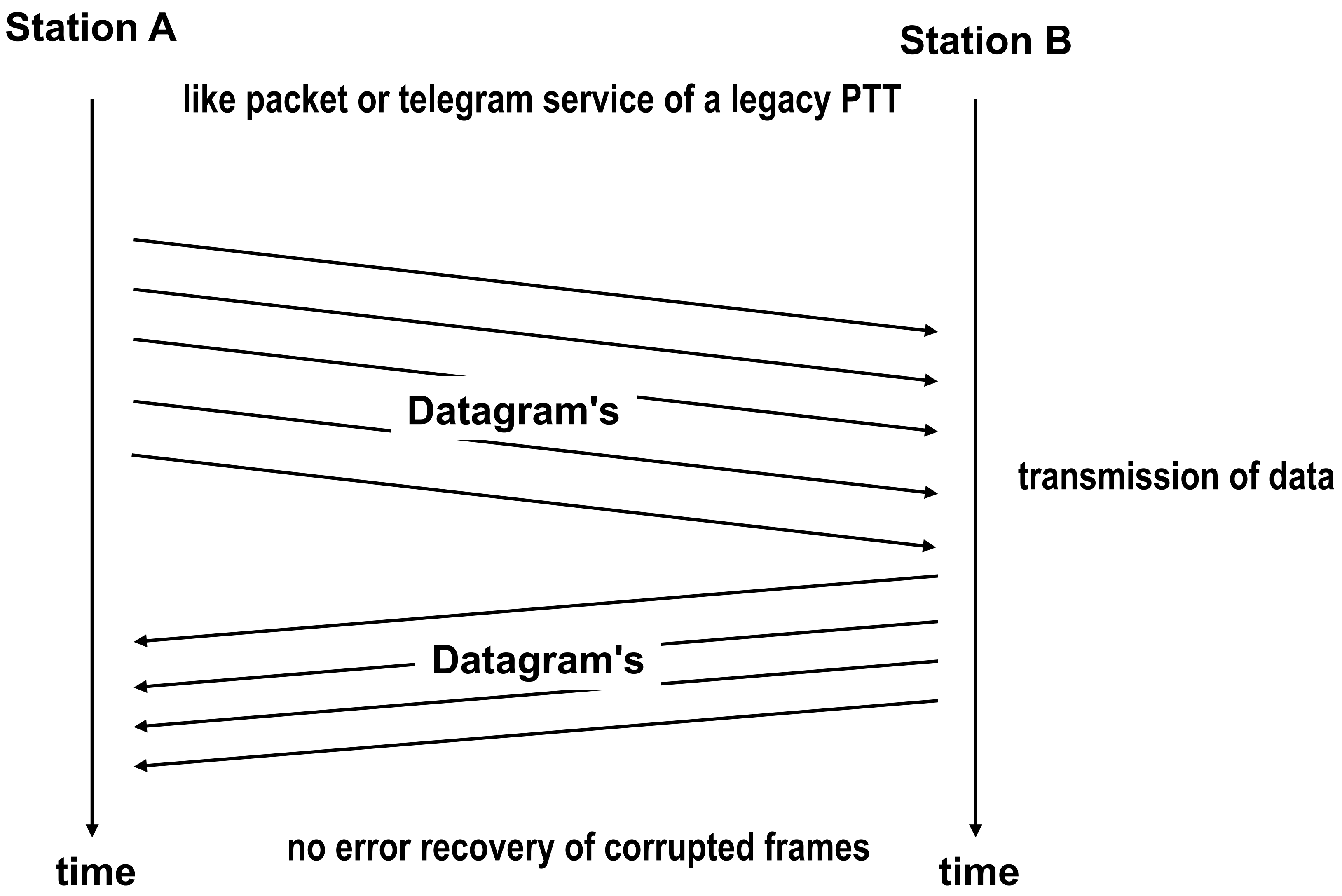
Three Important Principles for Data Communication

- **Layering**
  - Structuring the complex task of data communication into smaller pieces by usage of "layers"
- **Services**
  - Are provided by a layer to the upper layer and describe what is exchanged between layers within a system
- **Protocols**
  - Are used for communication between the systems (peers) within a layer

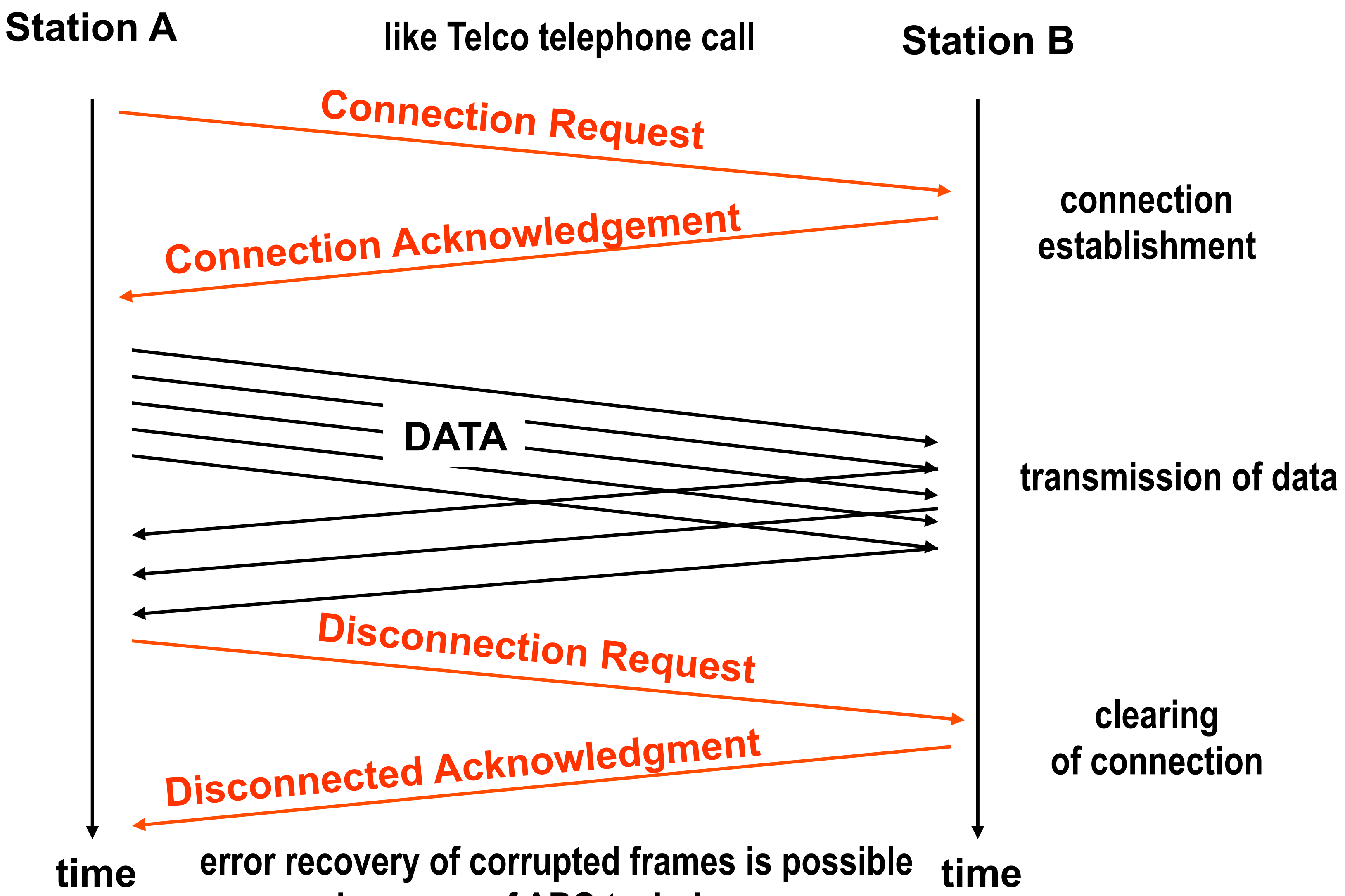
Point-to-Point Communication
3 Layer Model



Best-Effort Service: Connection-less Protocol



Reliable Service: Connection-oriented Protocol



error recovery of corrupted frames is possible by usage of ARQ techniques

Connection-oriented Protocols

- **Different definitions**
  - Some say *"protocols without addressing information"* and think of circuit-switched technologies
  - Some say *"protocols that do error recovery"*

– **Correct: "protocols that require a connection establishment before sending data and a disconnection procedure when finished"**

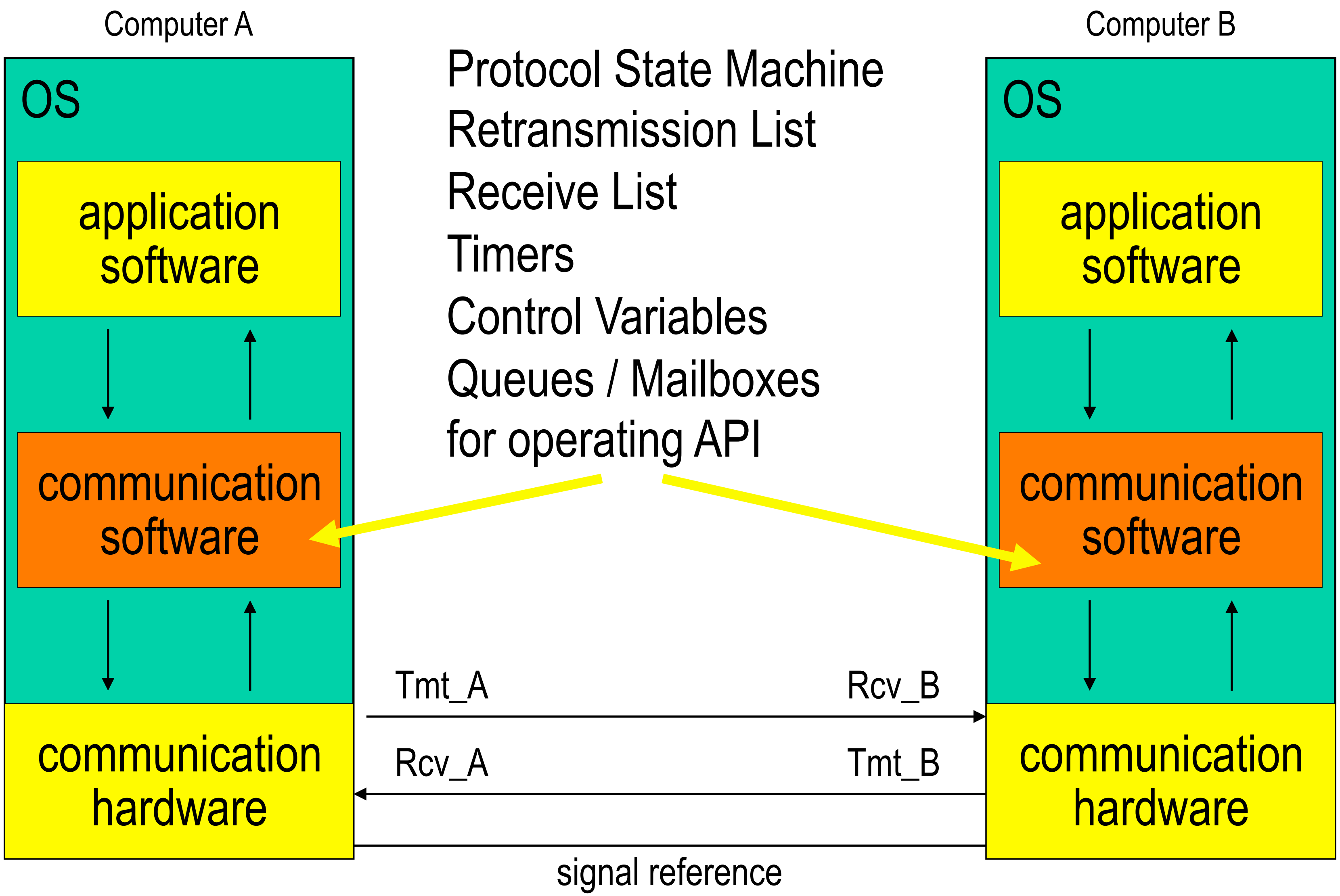
Agenda

- **Introduction**
- **<u>ARQ Techniques</u>**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth Delay Product
  - Flow Control
  - HDLC Overview

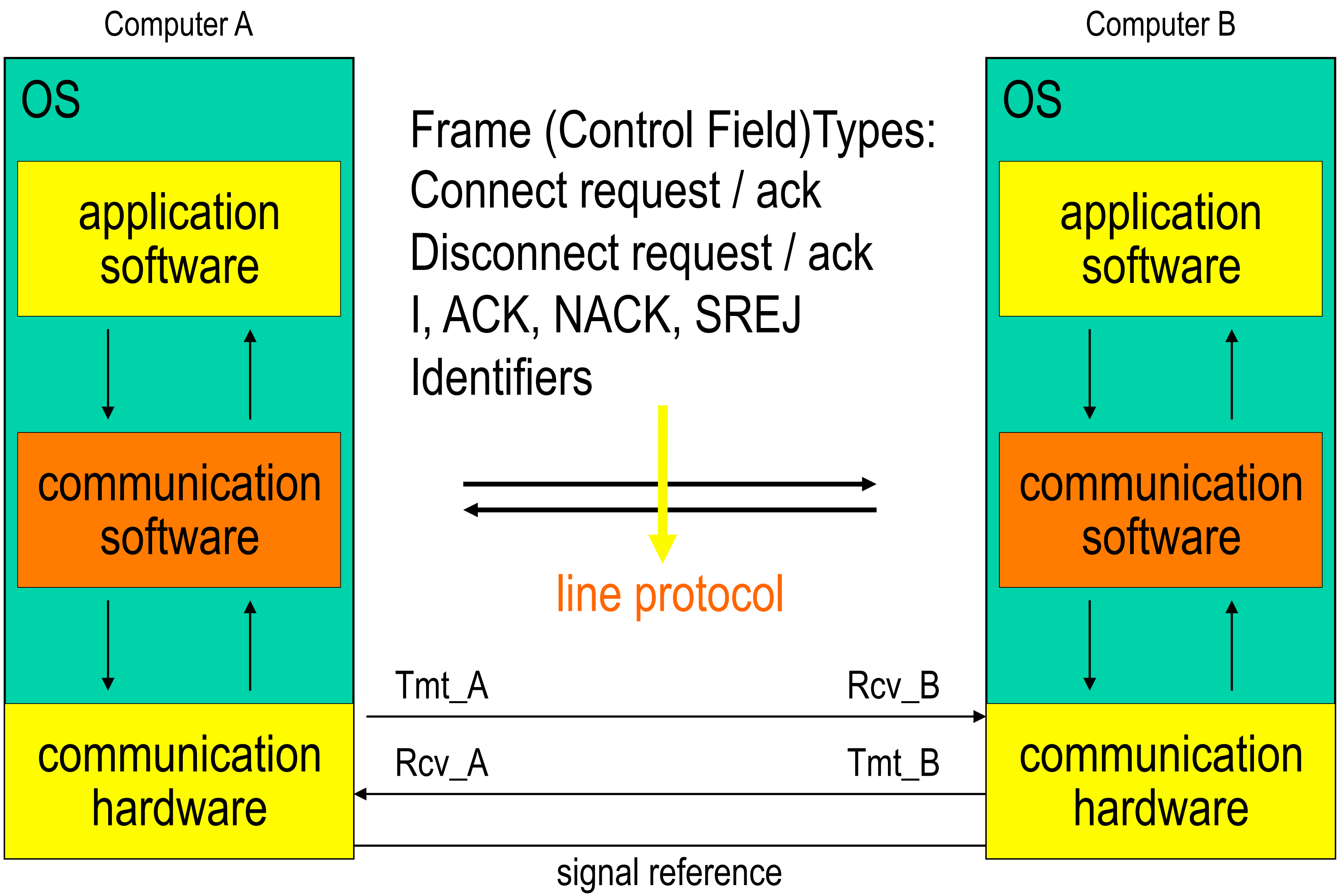
ARQ Techniques Overview

- **ARQ protocols guarantee correct delivery of data**
  - Error recovery by usage of feedback error control
  - Retransmission of data (information) frames after errors are detected by the receiver
- **Basic Method:**
  - Receiver acknowledges correct receipt of data frame by sending special control frames (ACK) in opposite direction
  - Acknowledgements refer to identifiers (sequence numbers) carried in the protocol header of the original data frame (I)

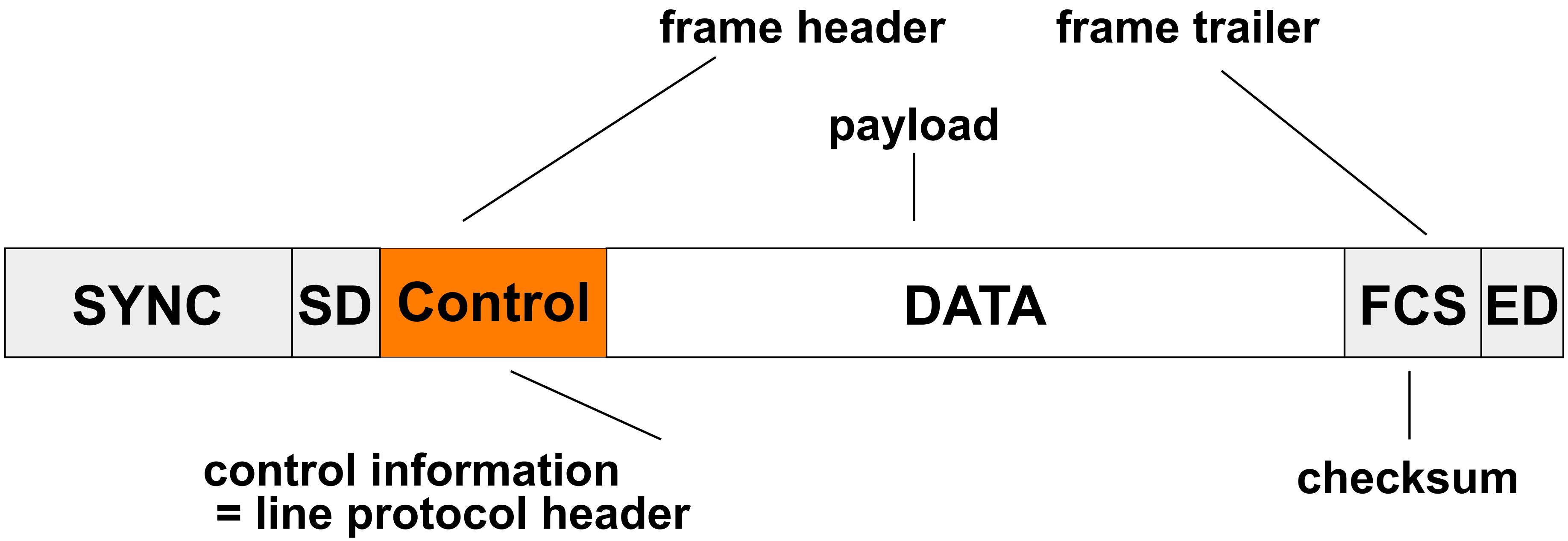
Com-SW Layer Implementation
Necessary Resource Elements



Com-SW Layer Implementation
Necessary Protocol Elements



Remember Control Field (Generic Frame Format)



Frame Type Field: Connect Request, I, ACK, NACK, …
Identifier Field: N, N+1, …

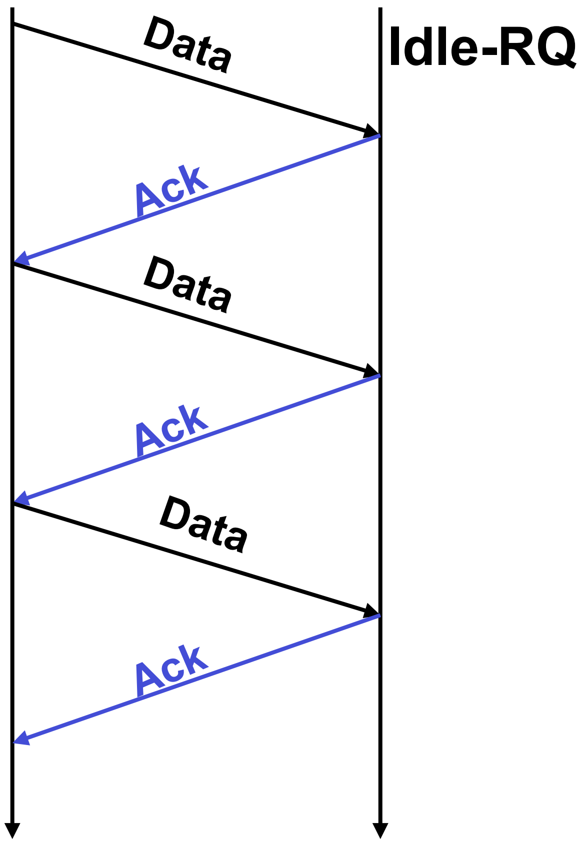
ARQ Variants

Idle-RQ

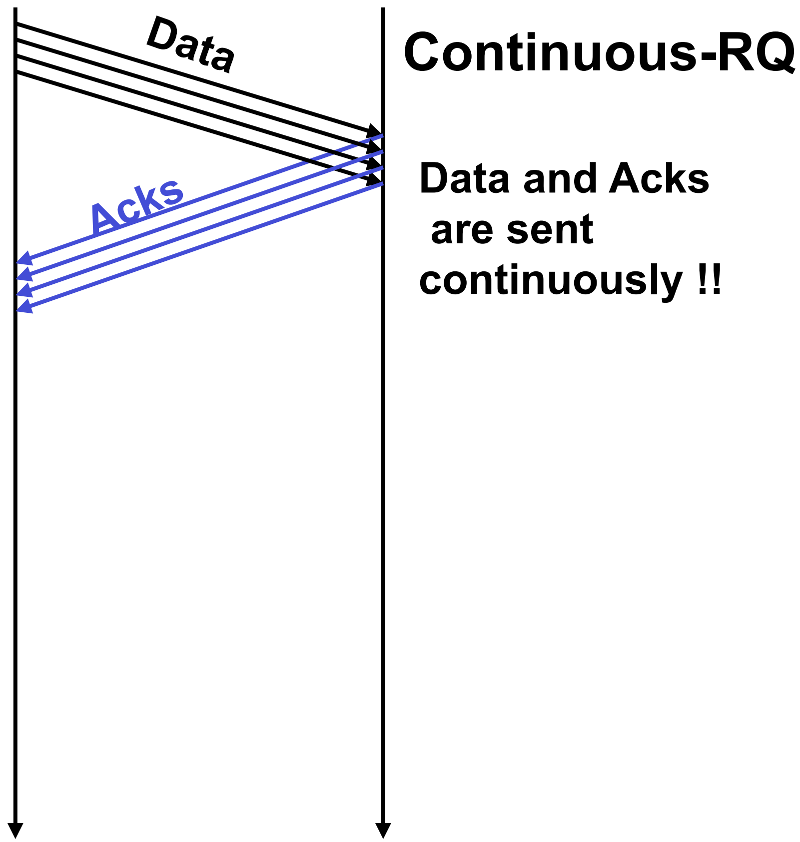
Continuous-RQ

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

Idle-RQ versus Continuous-RQ



half duplex
protocol

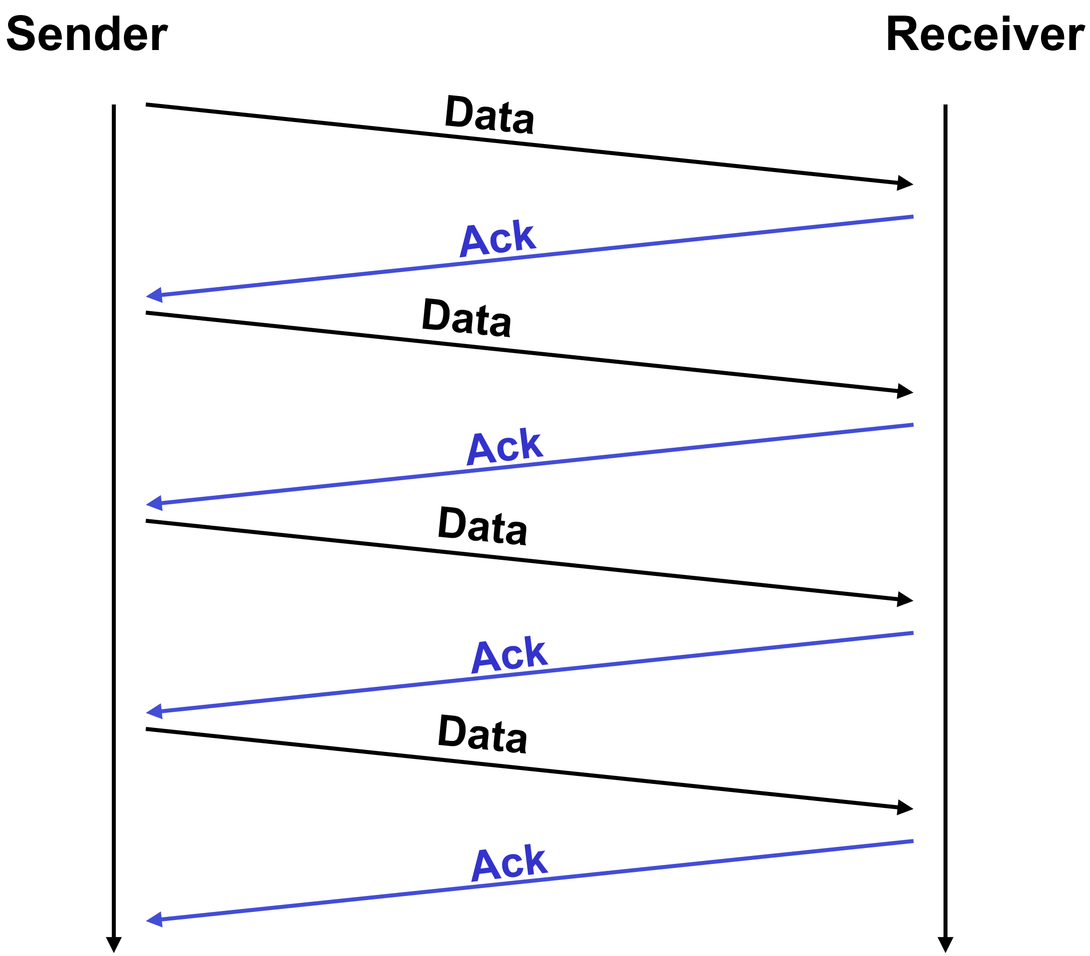


full duplex
protocol

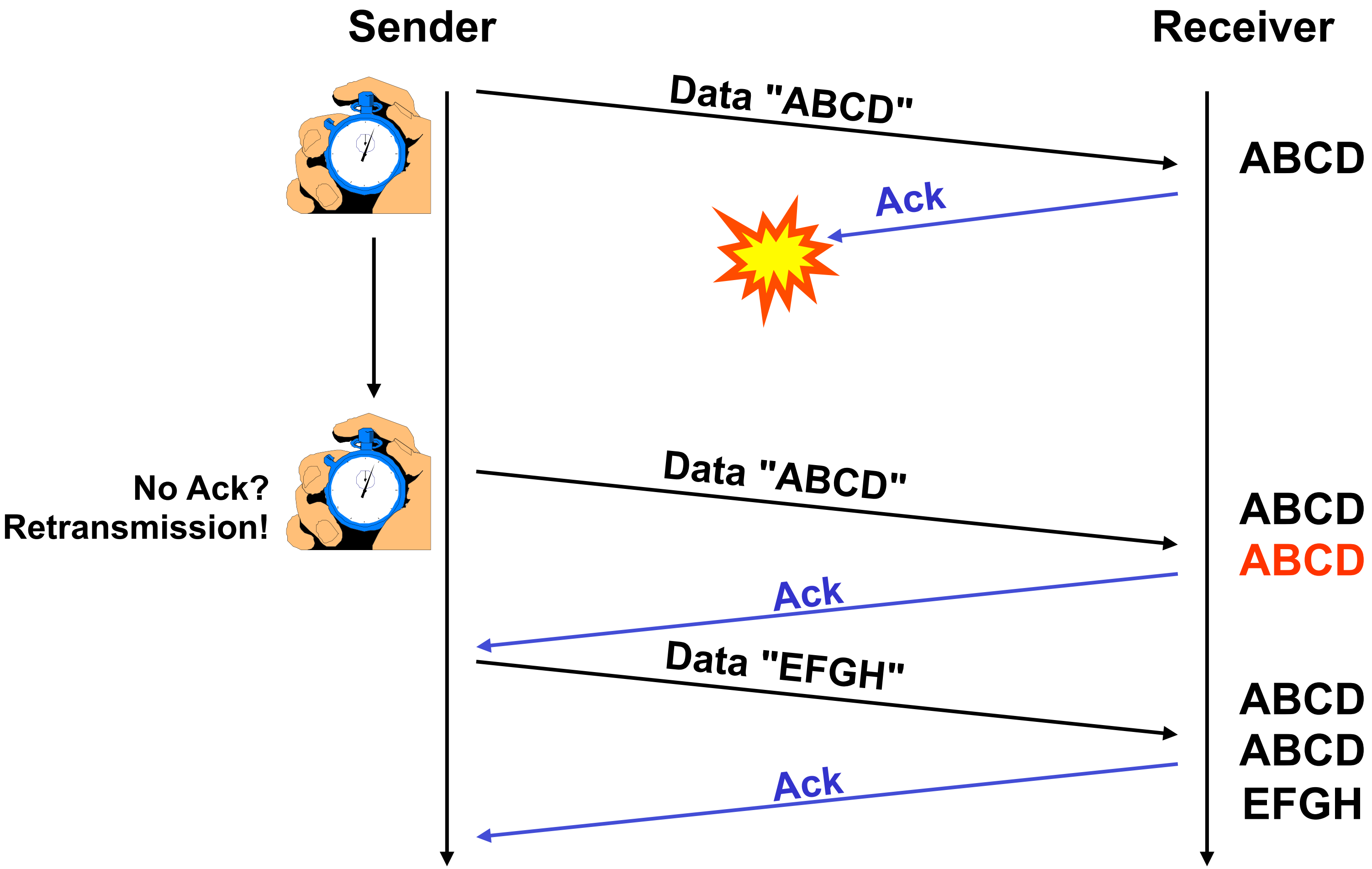
Agenda

- **Introduction**
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - Flow Control
  - HDLC Overview

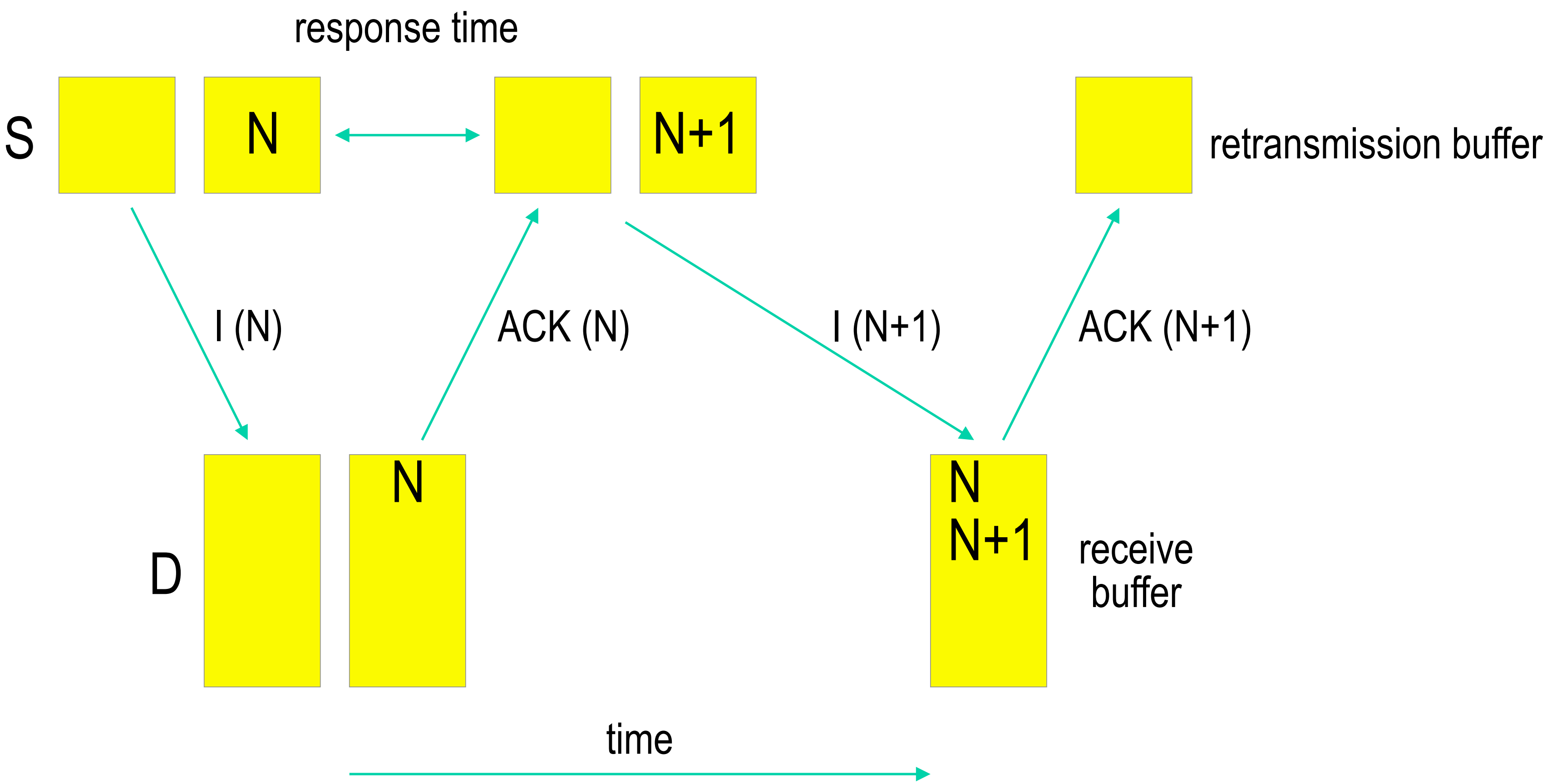
Idle-RQ



Why Identifiers or Sequence Numbers?



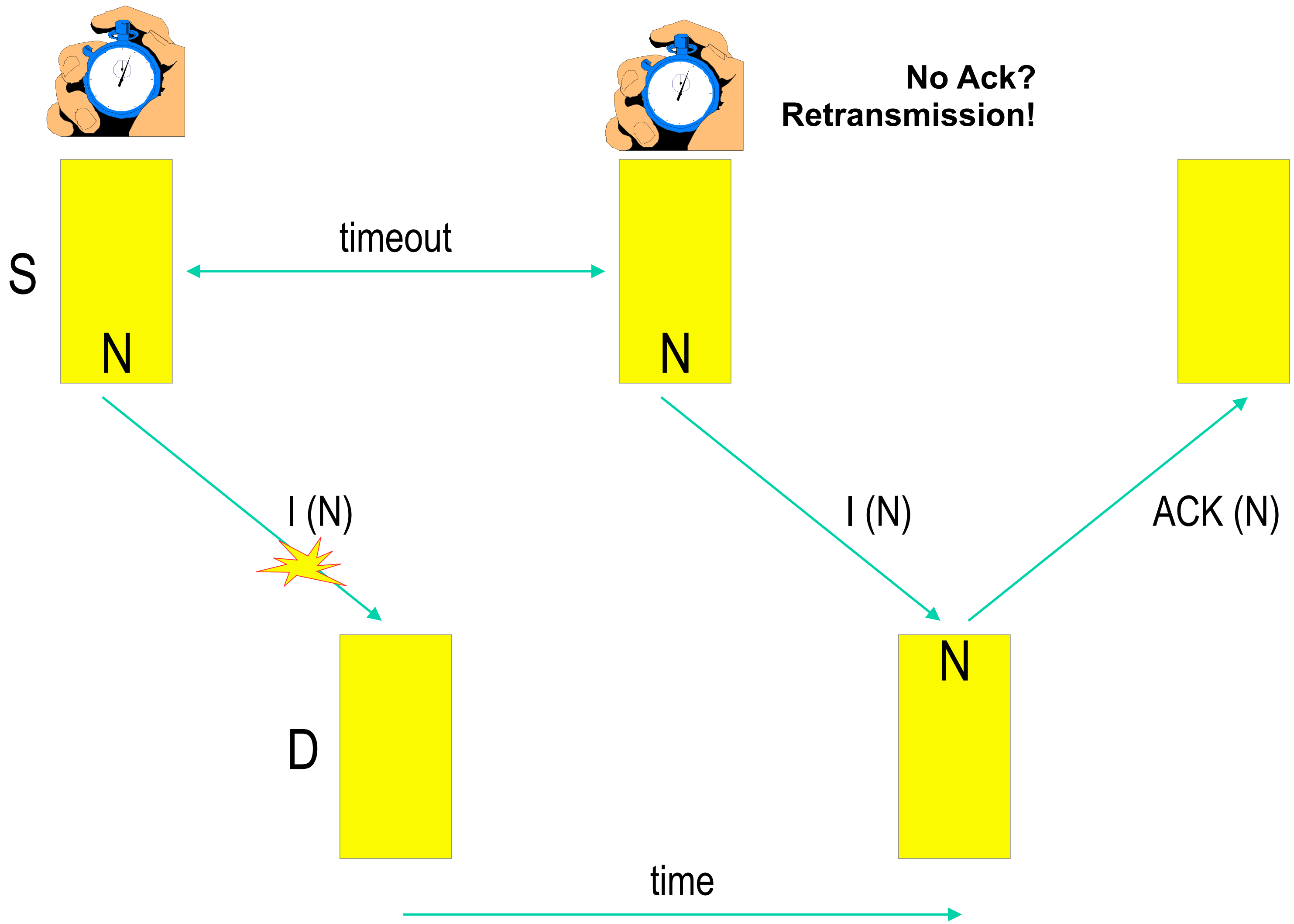
Basic Sequence of Idle-RQ



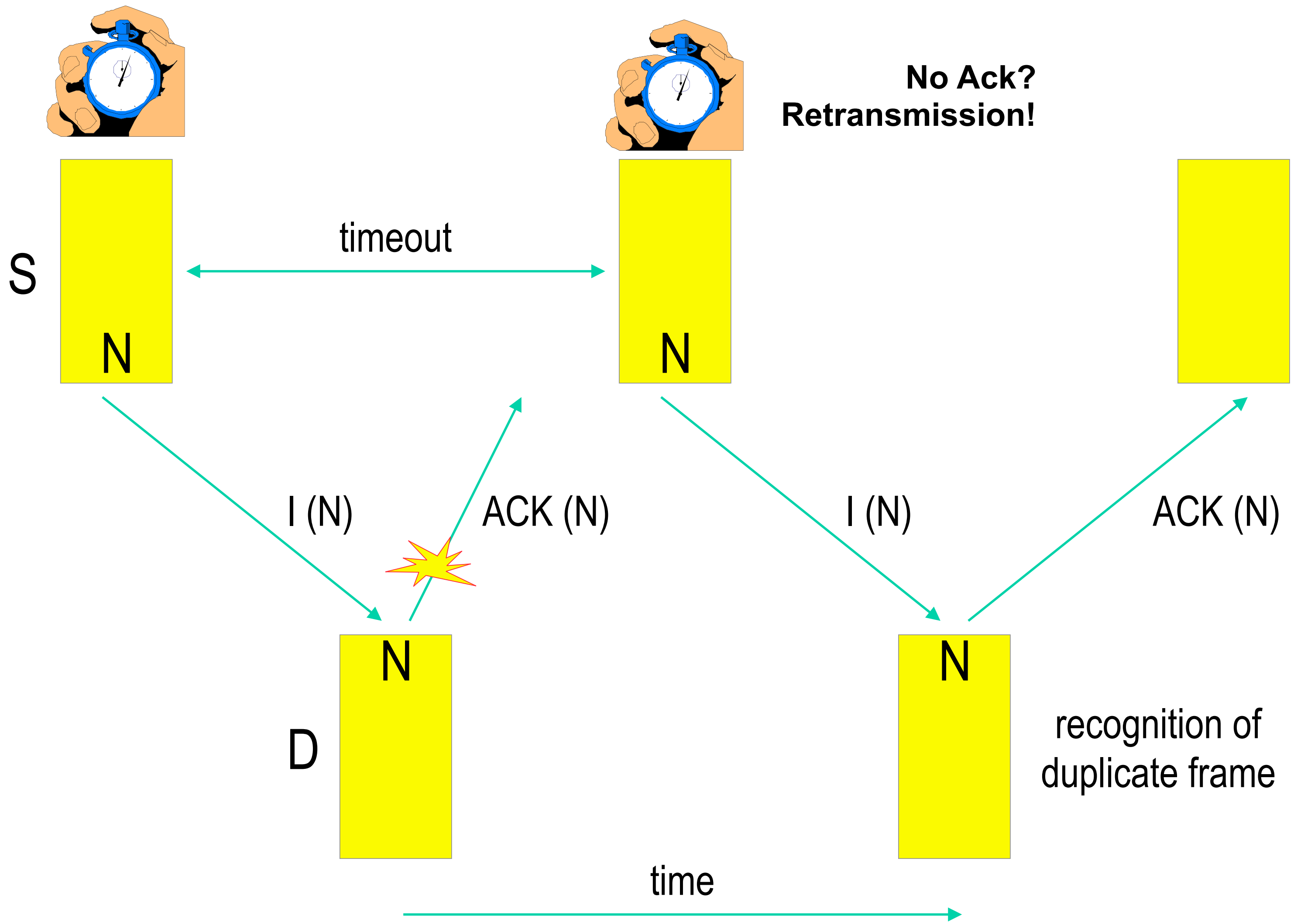
S … Source of I (N)
D … Destination I (N)

I (N) … Information Frame containing Data
ACK (N) … Acknowledgment Frame

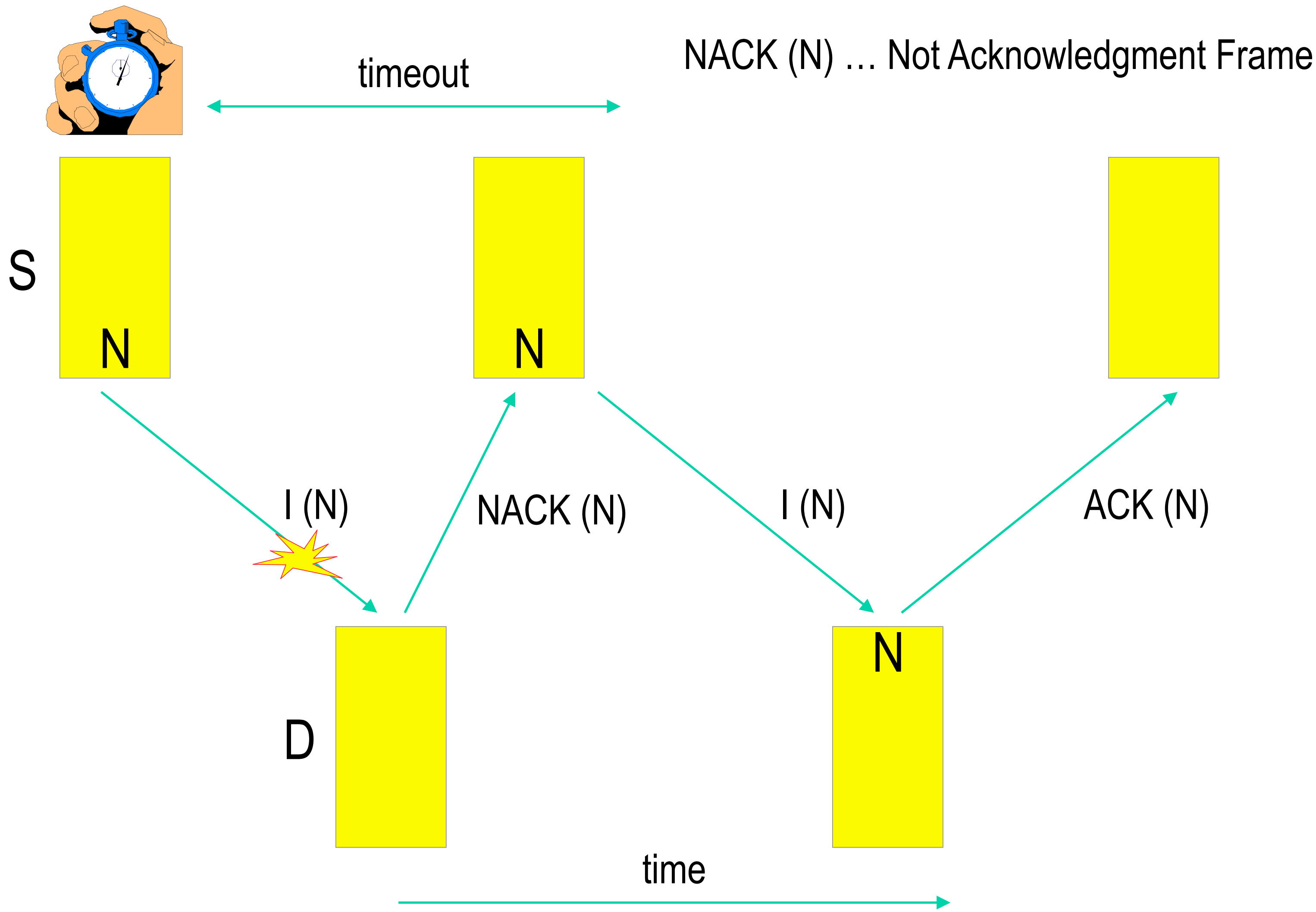
Idle-RQ Retransmission 1



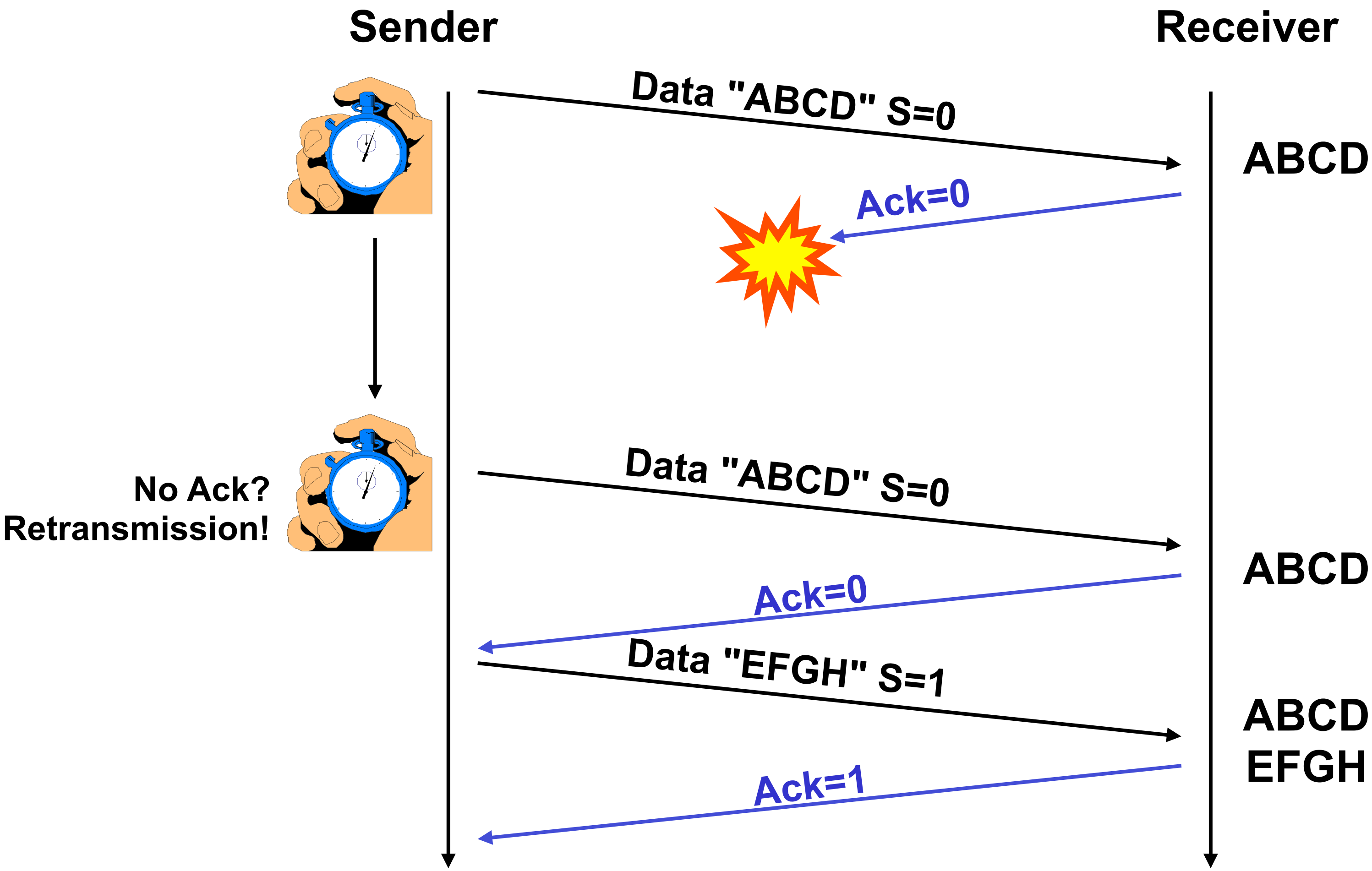
Idle-RQ Retransmission 2



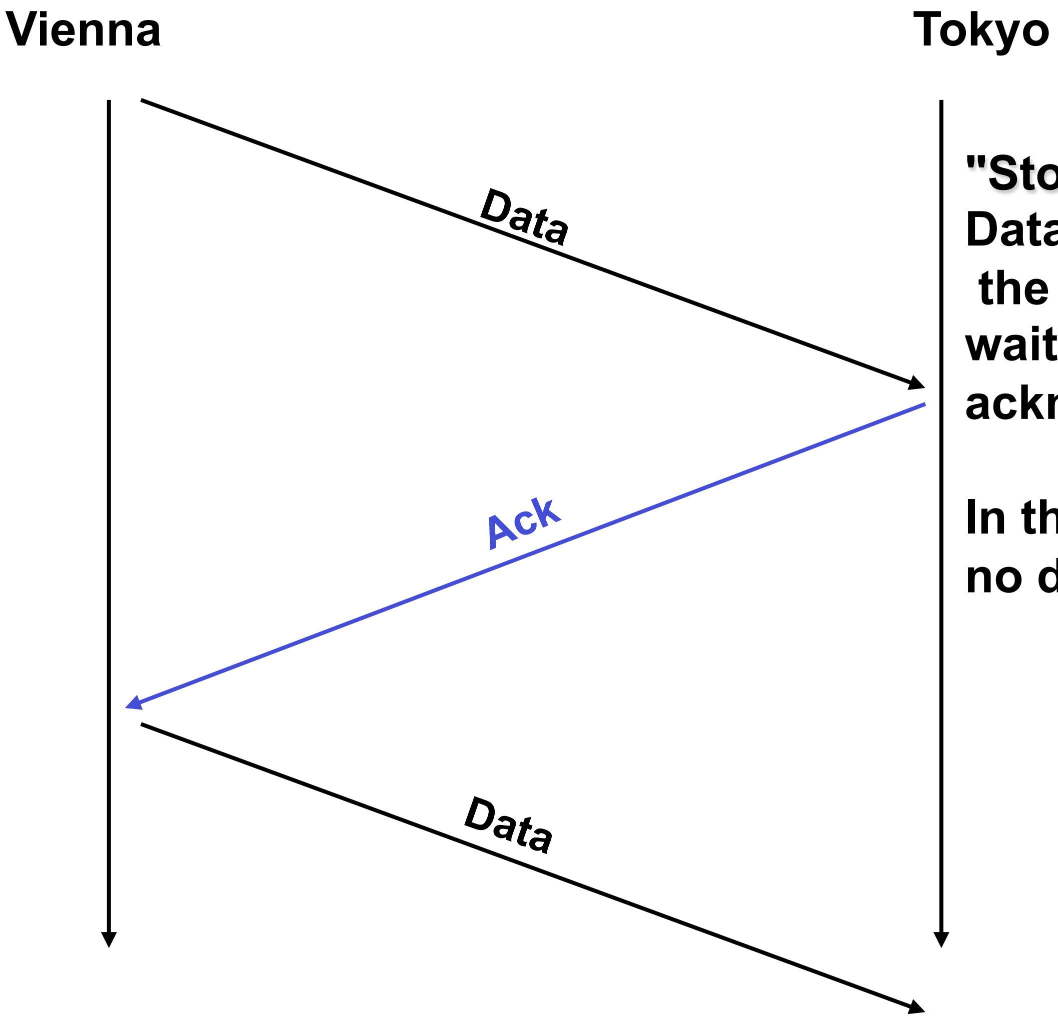
Idle-RQ Retransmission with optional NACK



Modulo 2 Numbering for Identifiers Sequence Numbers 0 and 1



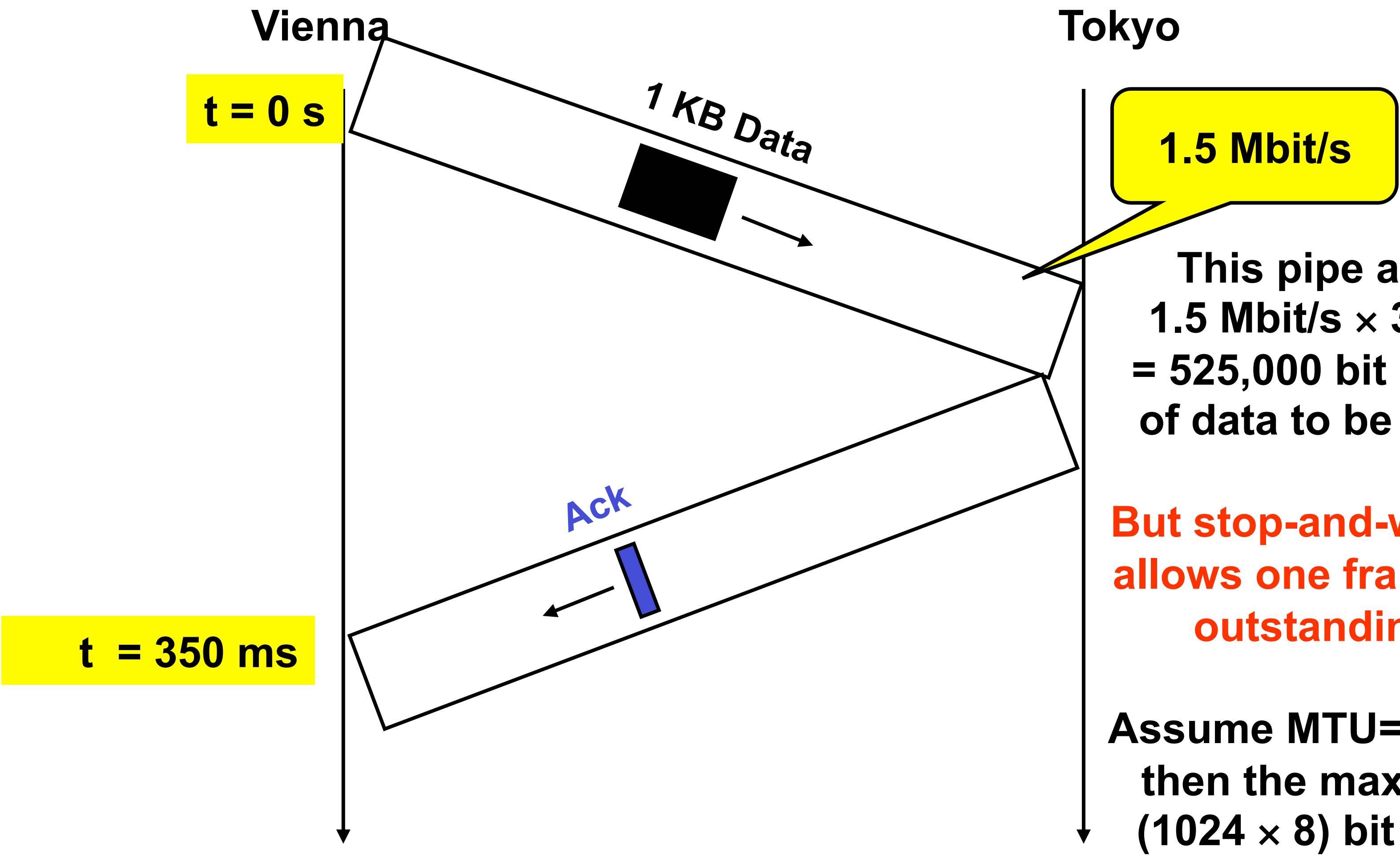
Slow !



"Stop and Wait":
Data is traveling round the earth while sender waits for the acknowledgement.

In the meantime no data can be sent !!

Nearly Empty Pipe !



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

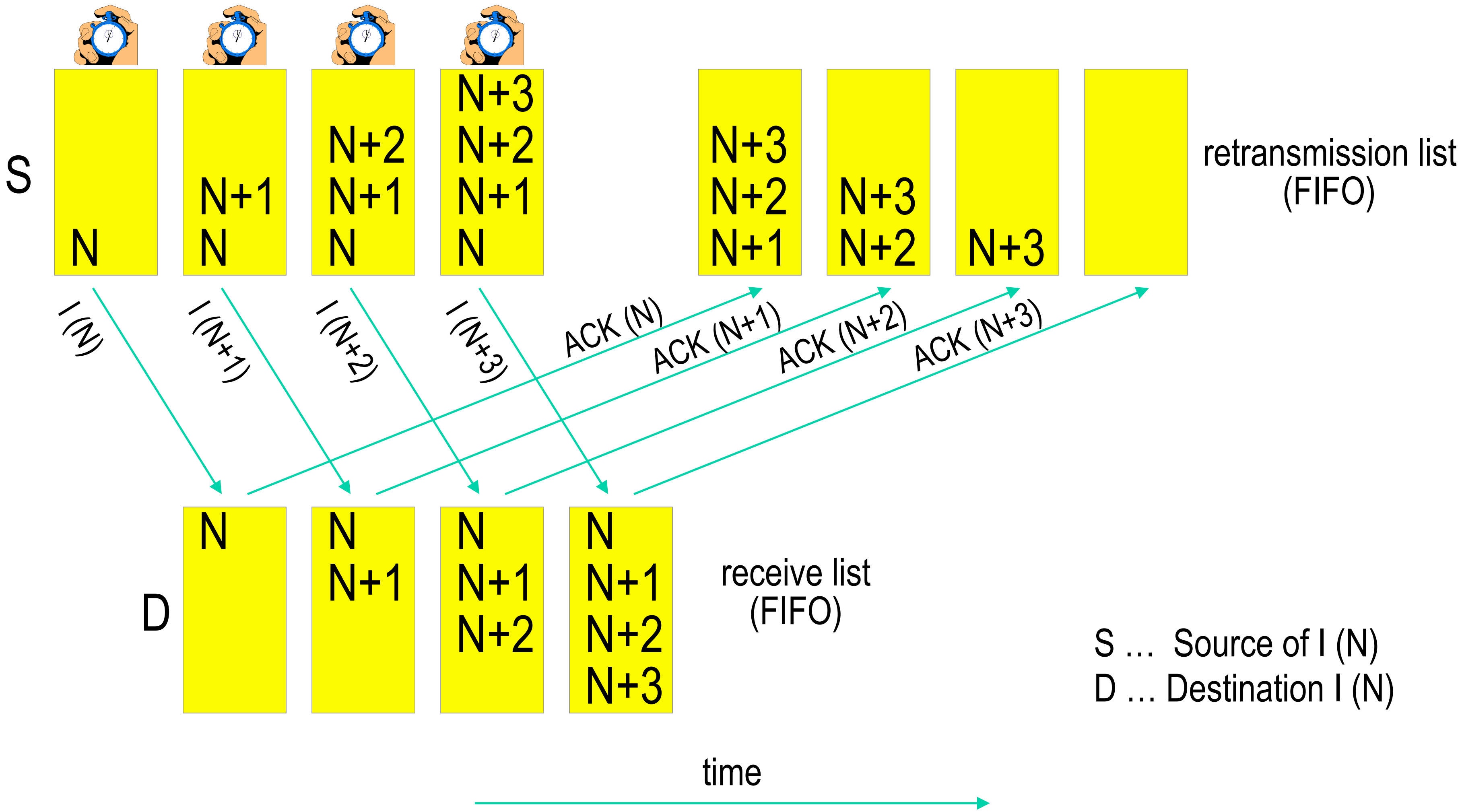
But stop-and-wait only allows one frame to be outstanding !!!

Assume MTU=1 KByte, then the max rate is
(1024 × 8) bit / 0.35 s
≅ 23 Kbit/s

Agenda

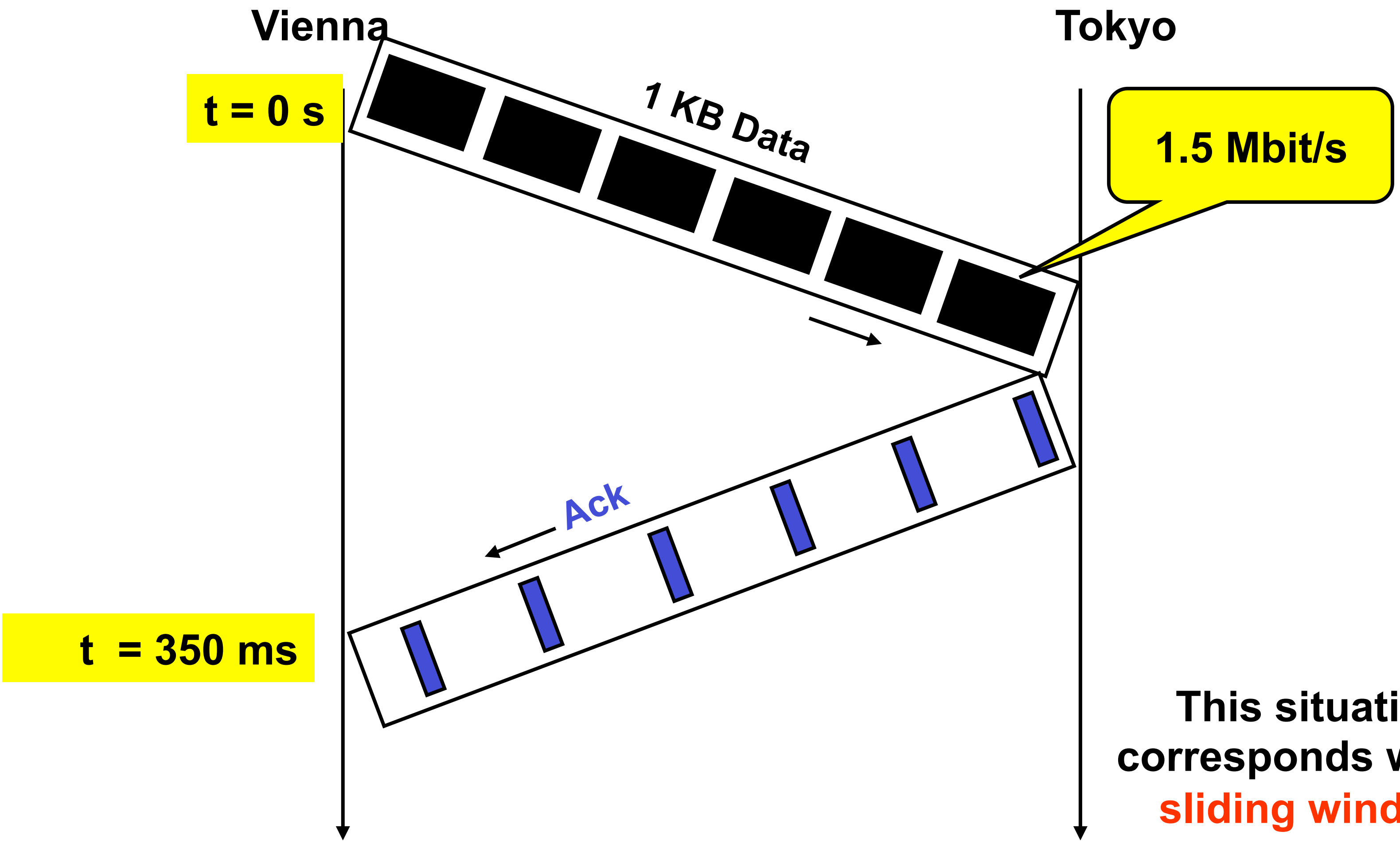
- **Introduction**
- **<u>ARQ Techniques</u>**
  - Introduction
  - Idle RQ
  - <u>Continuous RQ</u>
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - Flow Control
  - HDLC Overview

Continuous-RQ Principle



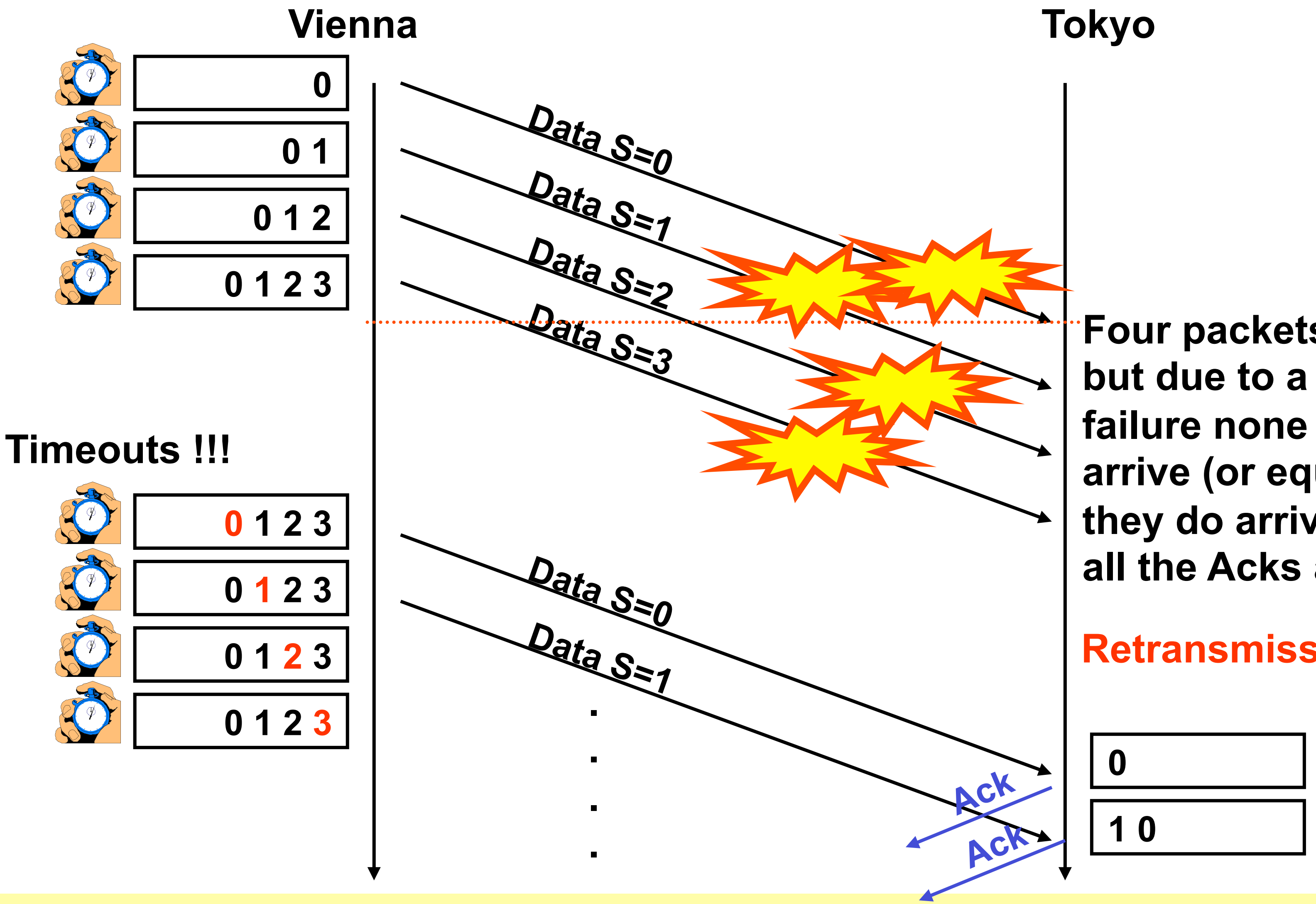
S … Source of I (N)
D … Destination I (N)

Full Pipe !



This situation corresponds with a **sliding window**

Why we need a Retransmission Buffer?



Four packets are sent, but due to a network failure none of them arrive (or equivalently they do arrive but all the Acks are lost)

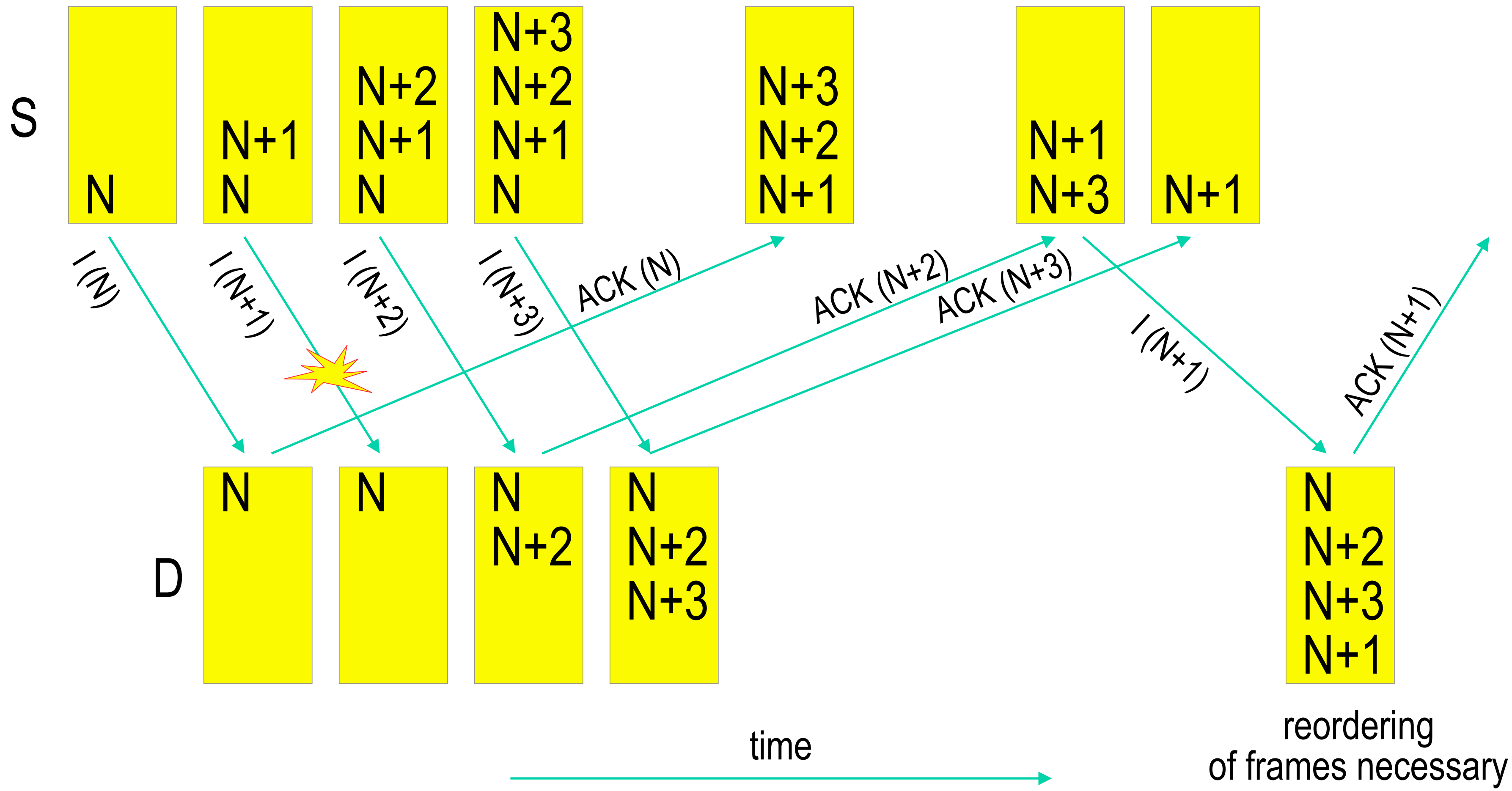
Retransmissions

Agenda

- **Introduction**
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - Flow Control
  - HDLC Overview

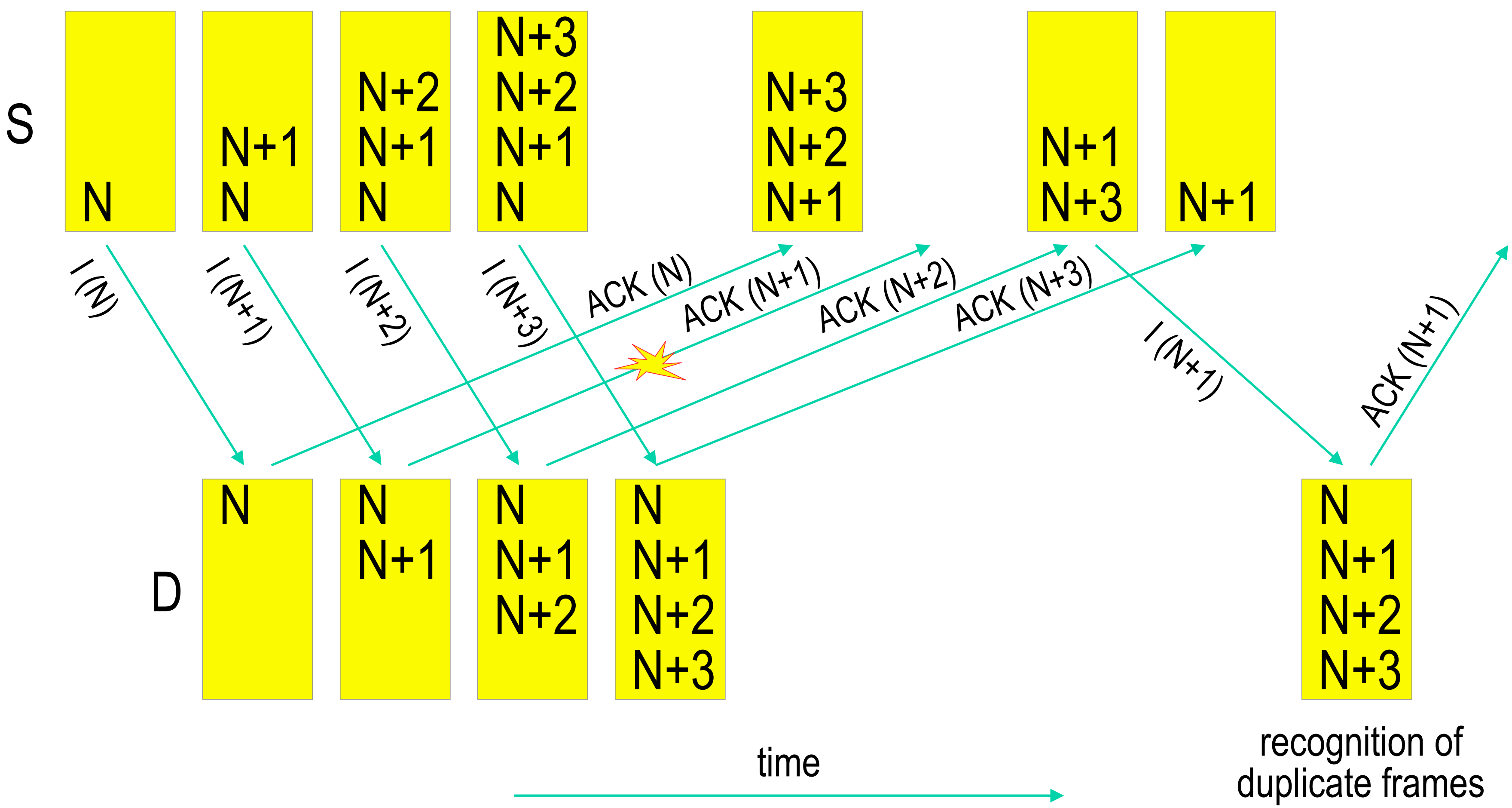
Selective Acknowledgement (Scenario 1)

ACK (N+2) triggers implicitly retransmission of I (N+1) at S

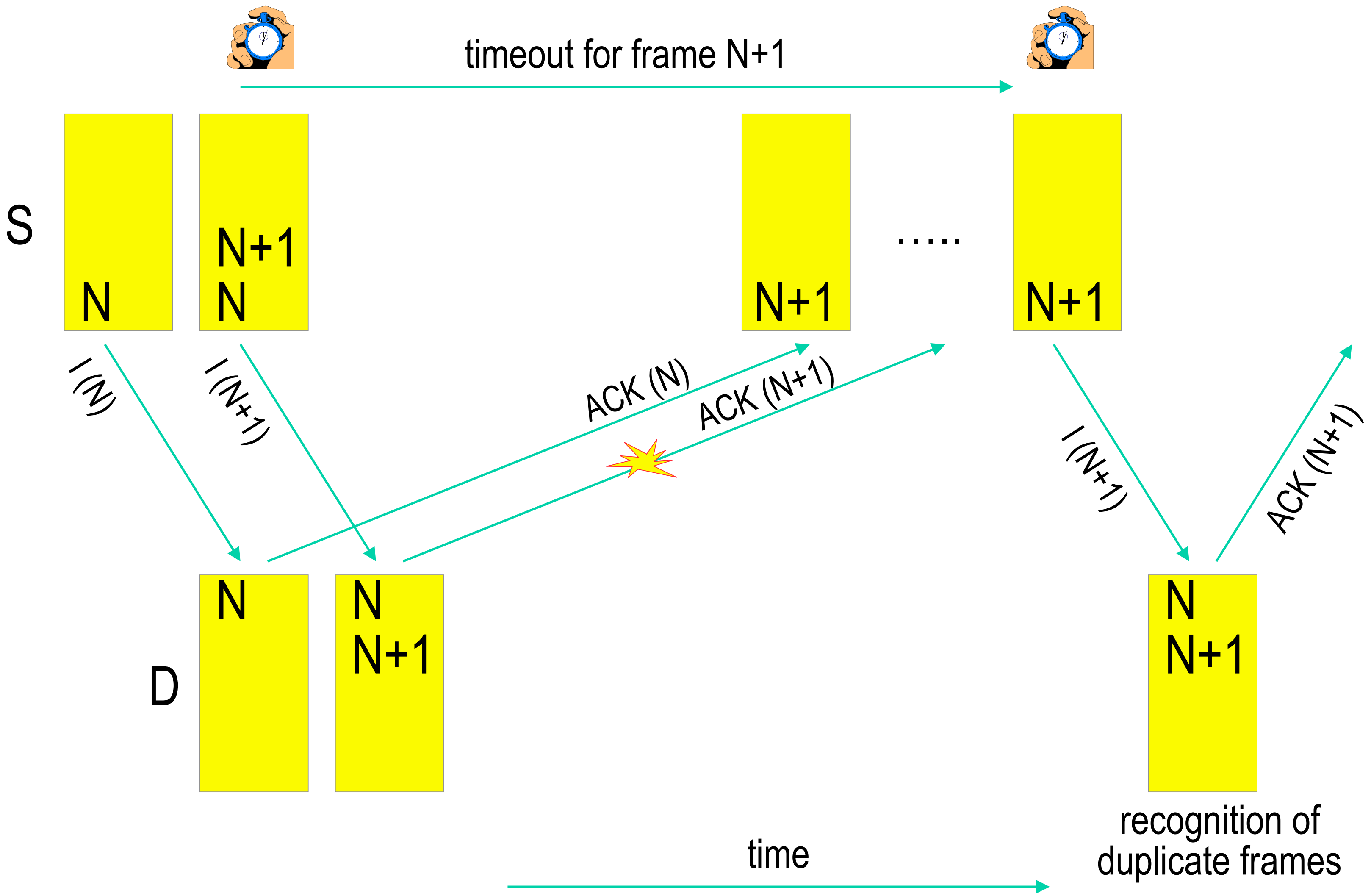


Selective Acknowledgement (Scenario 2)

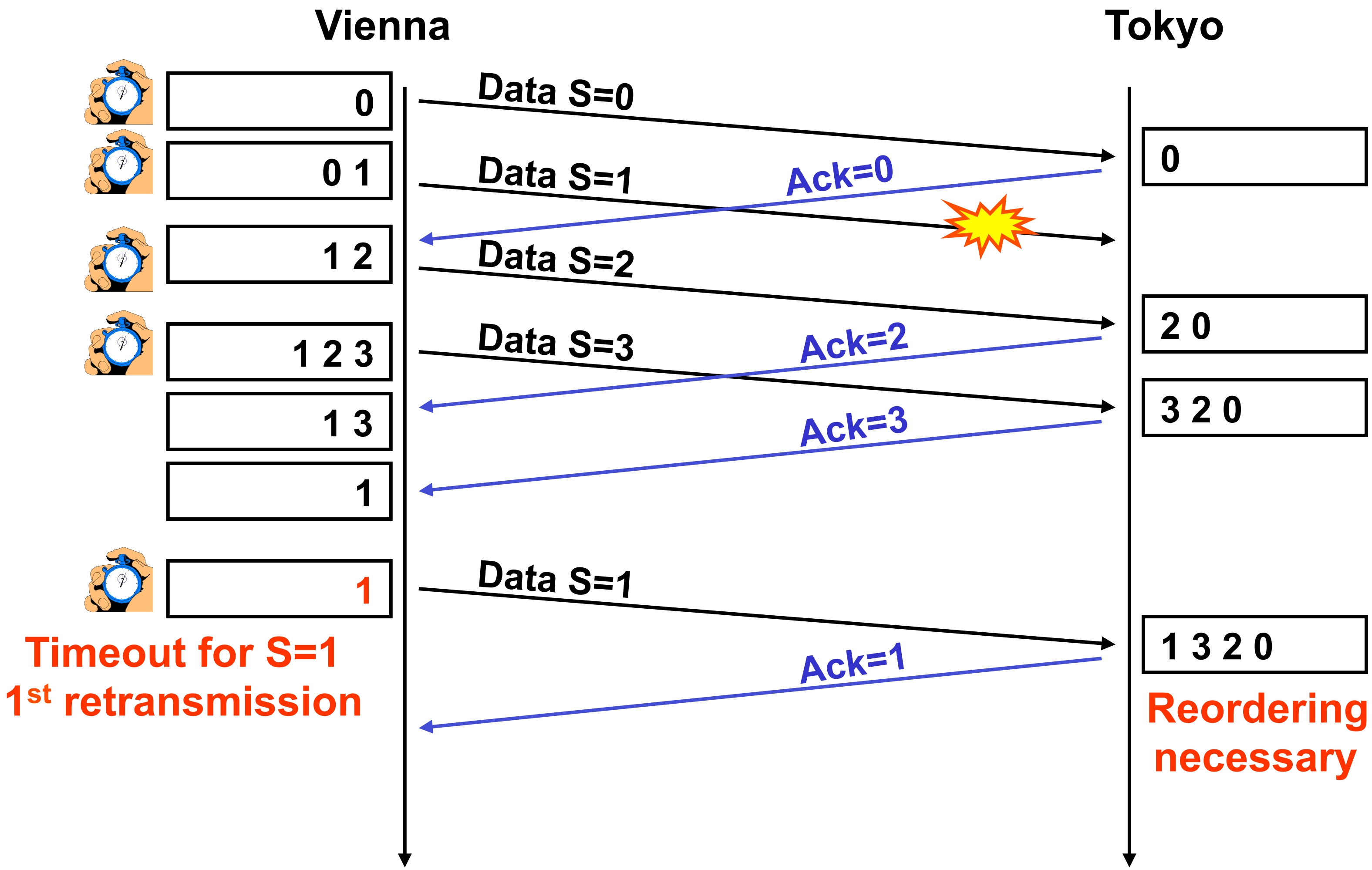
ACK (N+2) triggers implicitly retransmission of I (N+1) at S



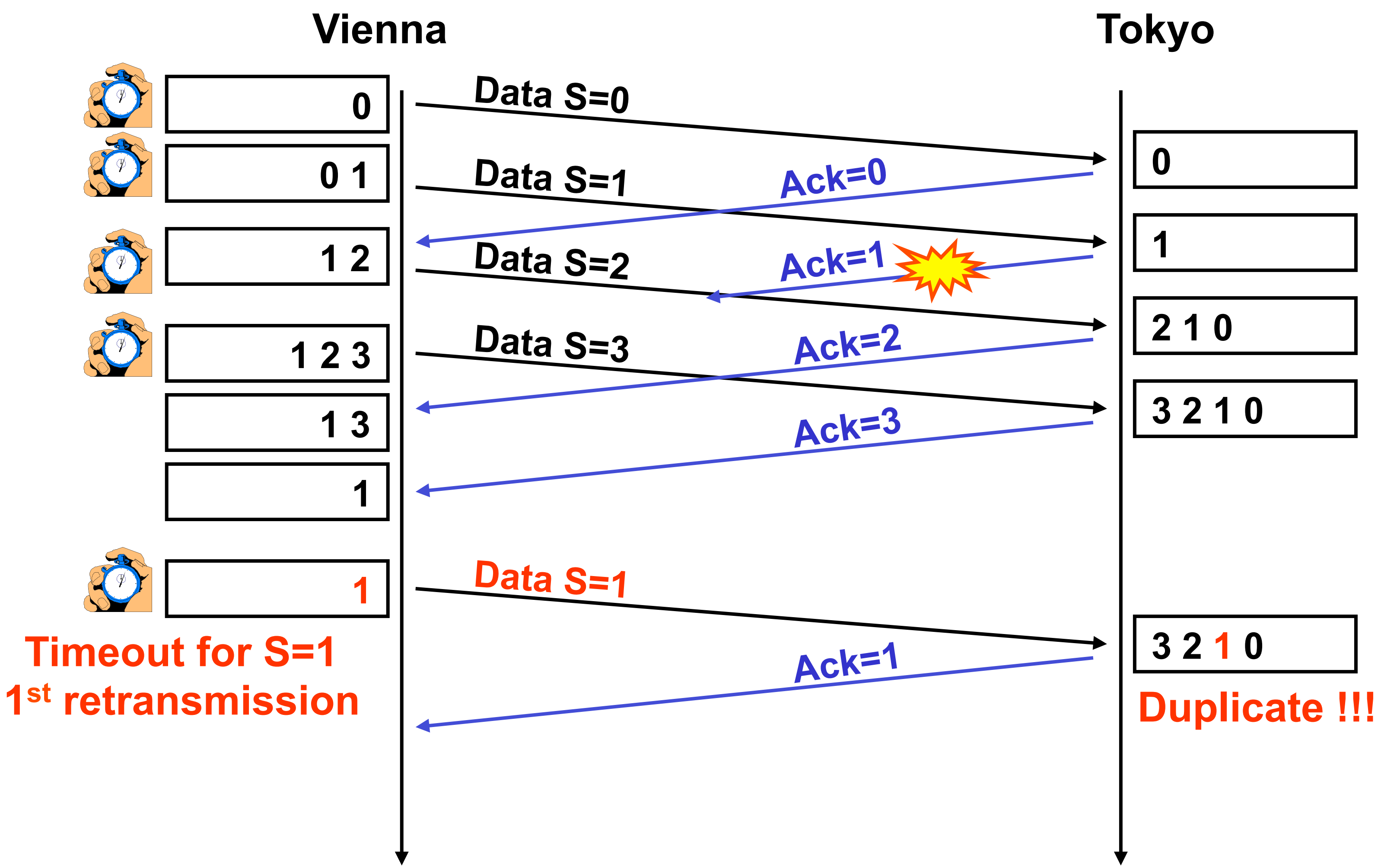
Selective Acknowledgement (Scenario 3)



Selective Acknowledgement (Scenario 4)



Selective Acknowledgement (Scenario 5)



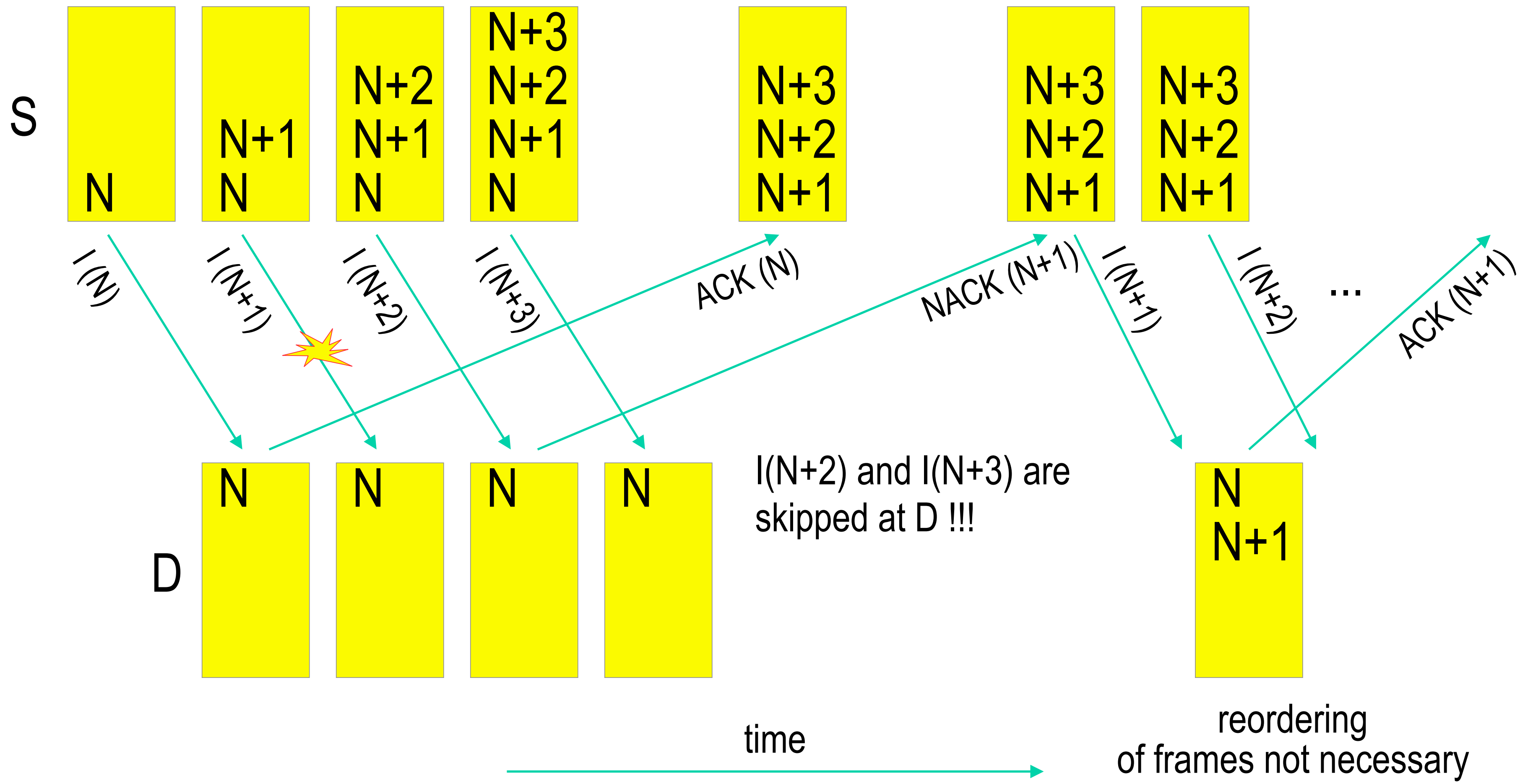
SACK in TCP

- **Application:**
  - New option for TCP to accommodate to long fat pipes with high BER (Bit Error Rate)
  - Part of modern TCP
- **Optionally, retransmissions might be sent immediately when unexpected (the next but one) ACK occurs**
- **Opposite idea: Cumulative ACK**

Agenda

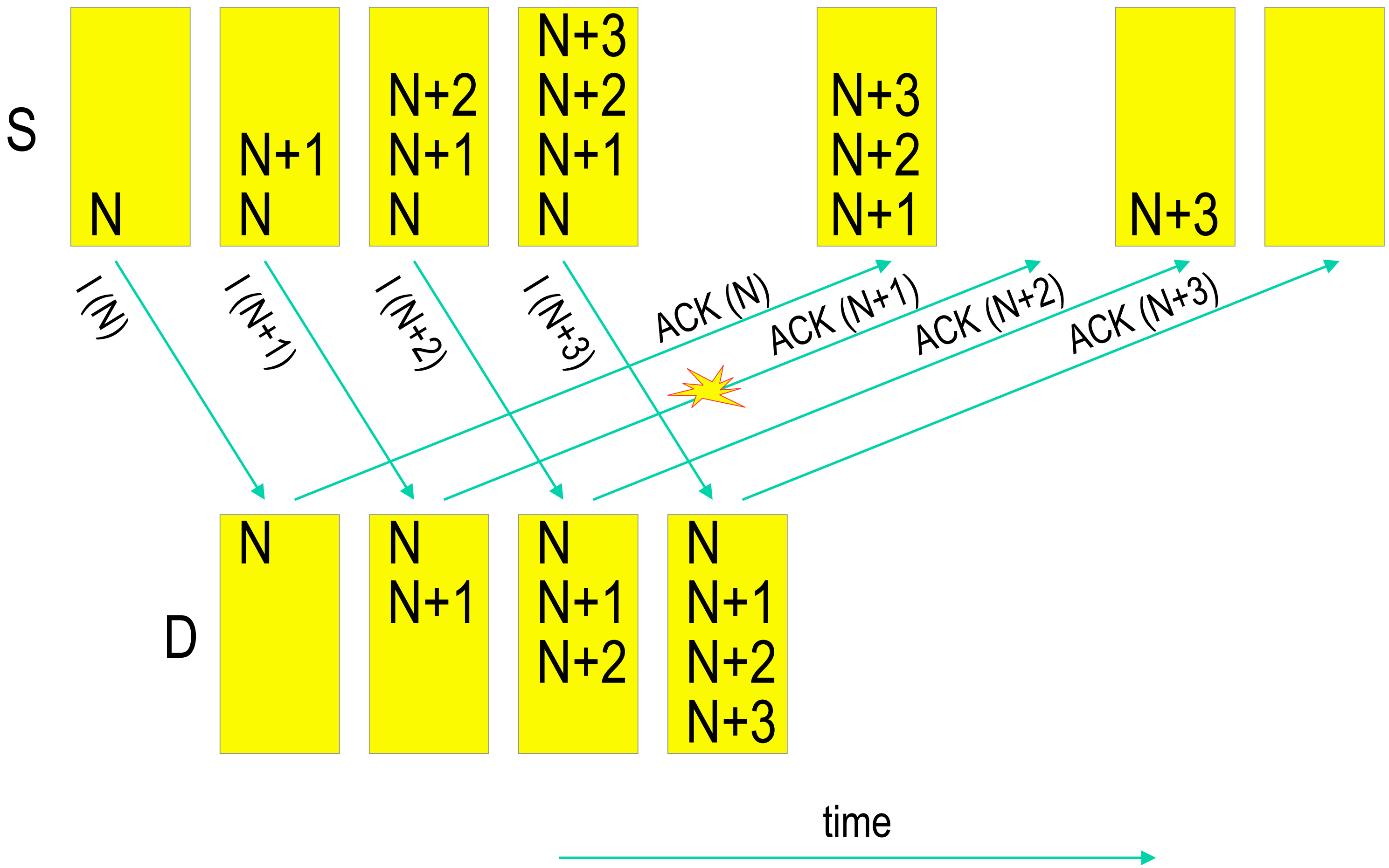
- **Introduction**
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - Flow Control
  - HDLC Overview

GoBackN (Scenario 1)

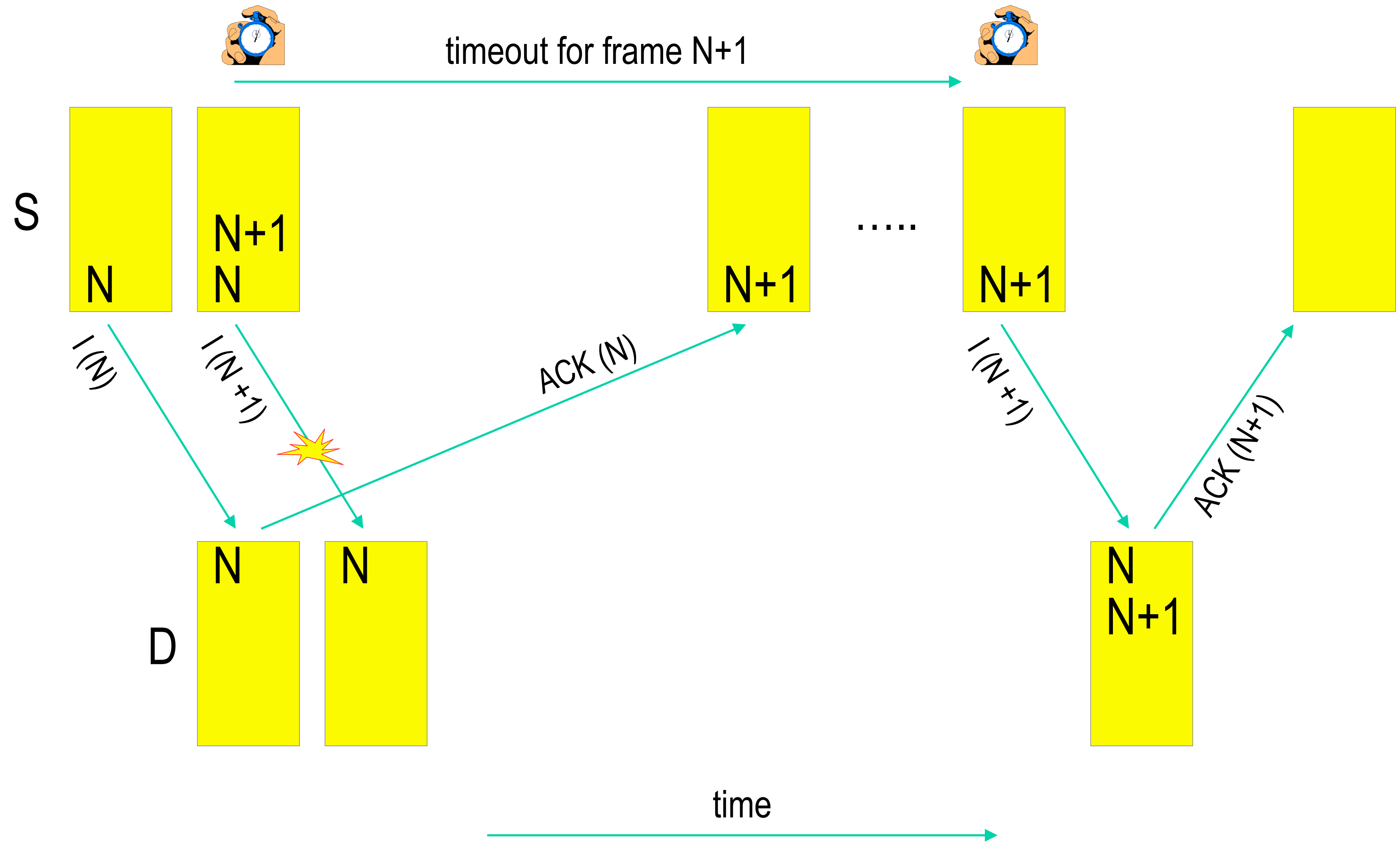


GoBackN (Scenario 2)

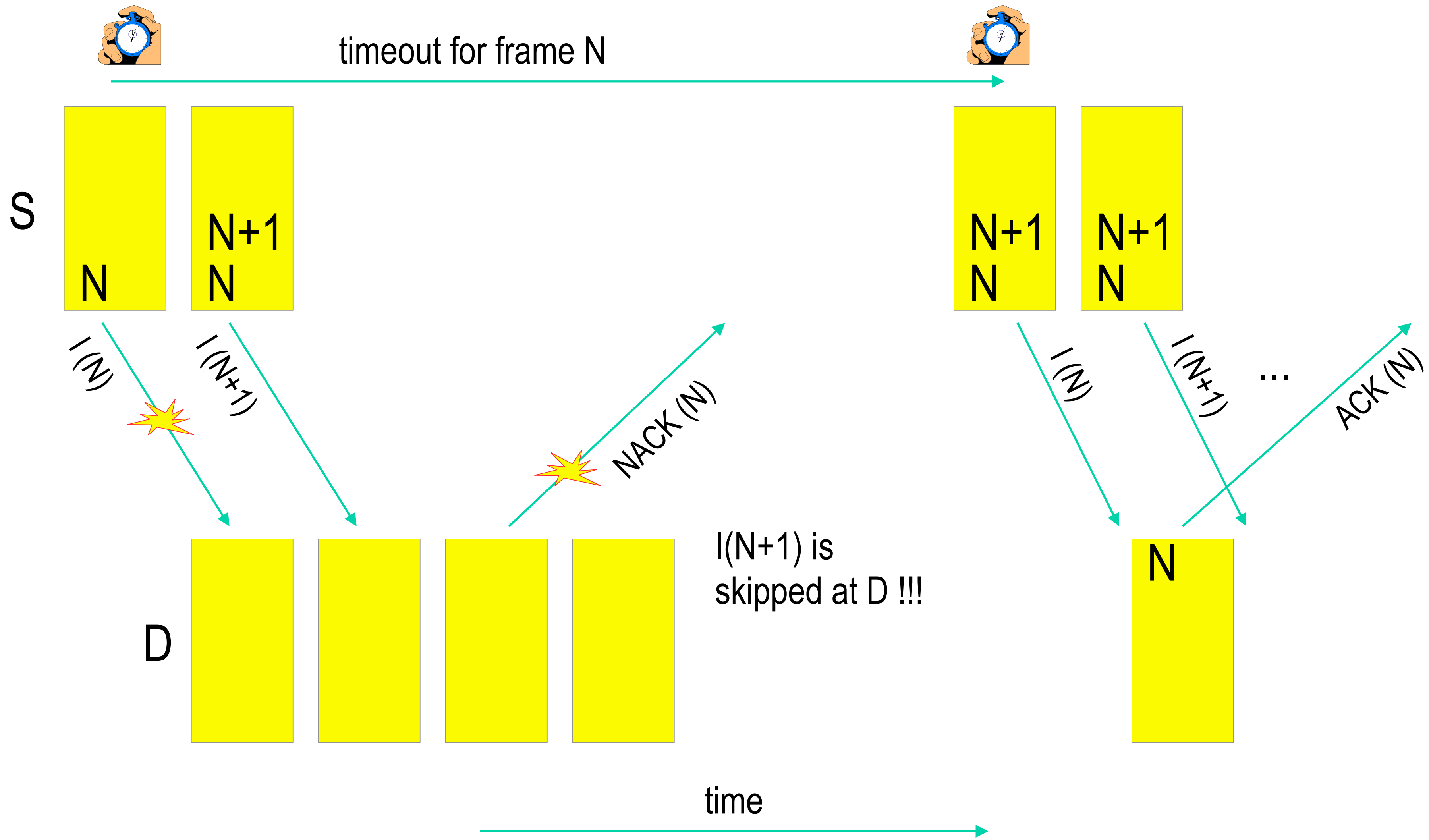
ACK (N+2) acknowledges all frames up to I (N+2) at S
-> multiple acknowledgment



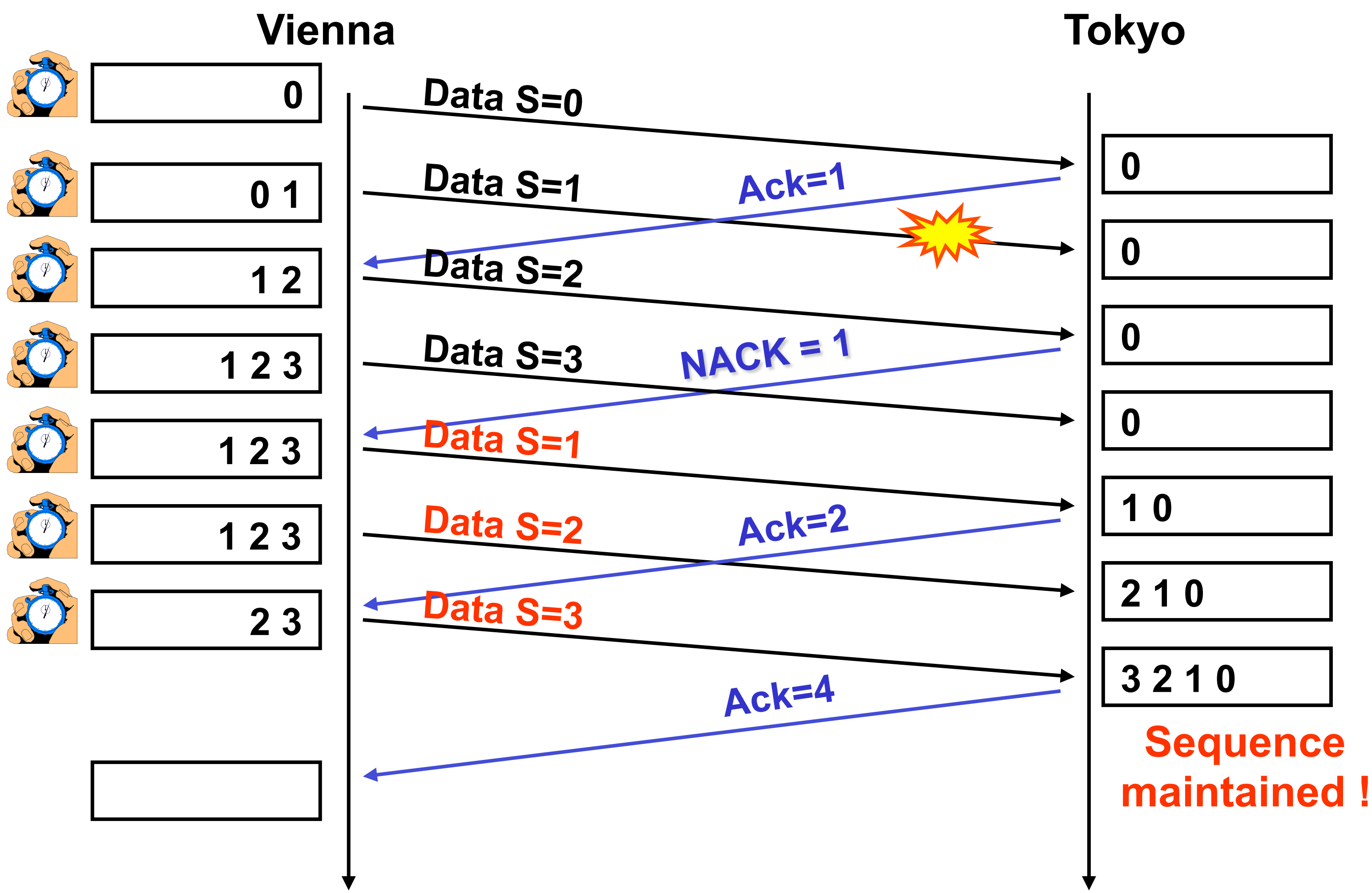
GoBackN (Scenario 3)



GoBackN (Scenario 4)



GoBackN with Sequence Numbers



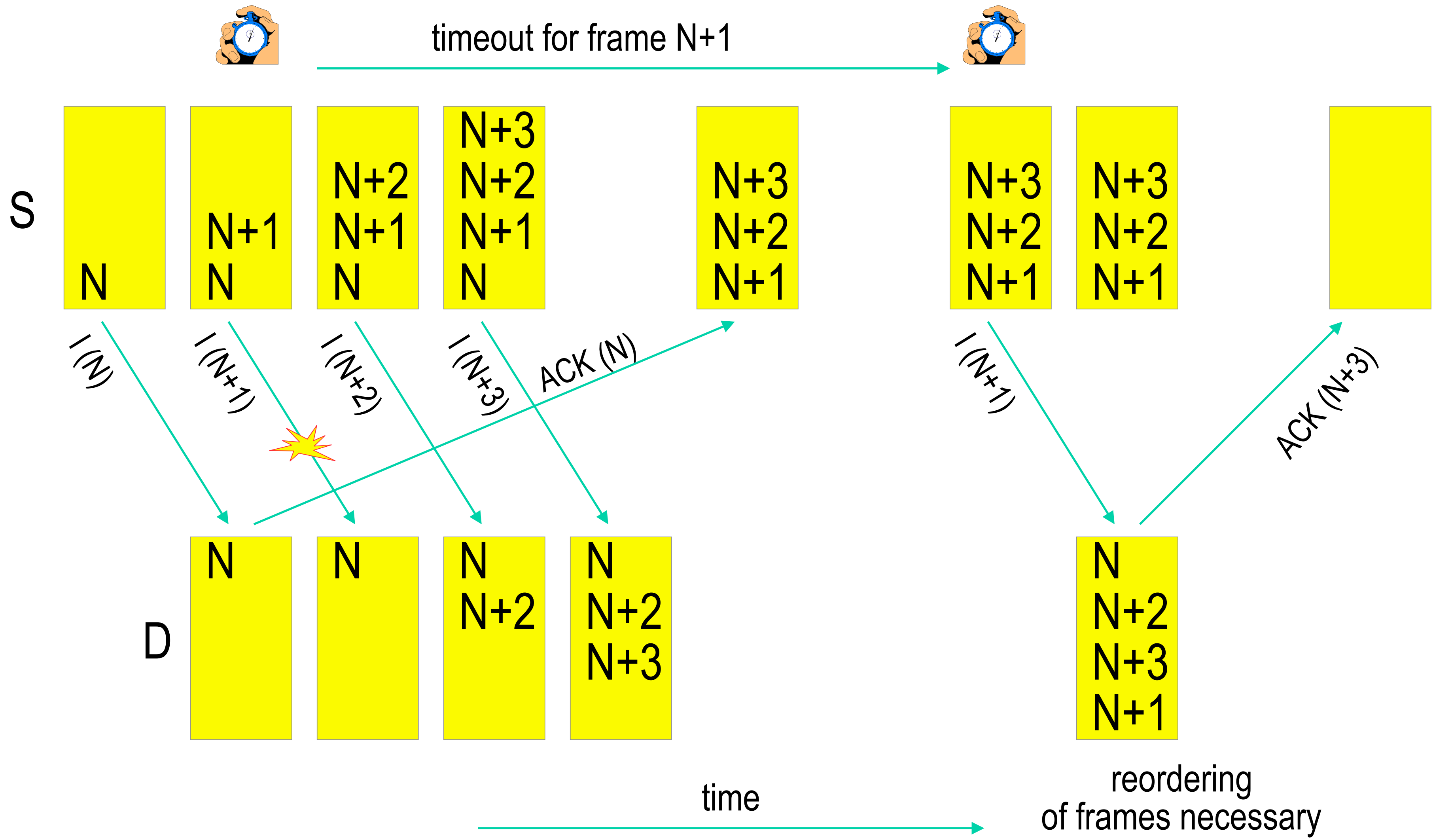
GoBackN - Facts

- **Maintains order at receiver-buffer**
  – Reordering was too much time-consuming in earlier days
- **Still used by**
  – HDLC and clones ("REJECT")
  – TCP
    - Variant known as "fast retransmit"
    - Uses duplicate Acks as NACK

Agenda

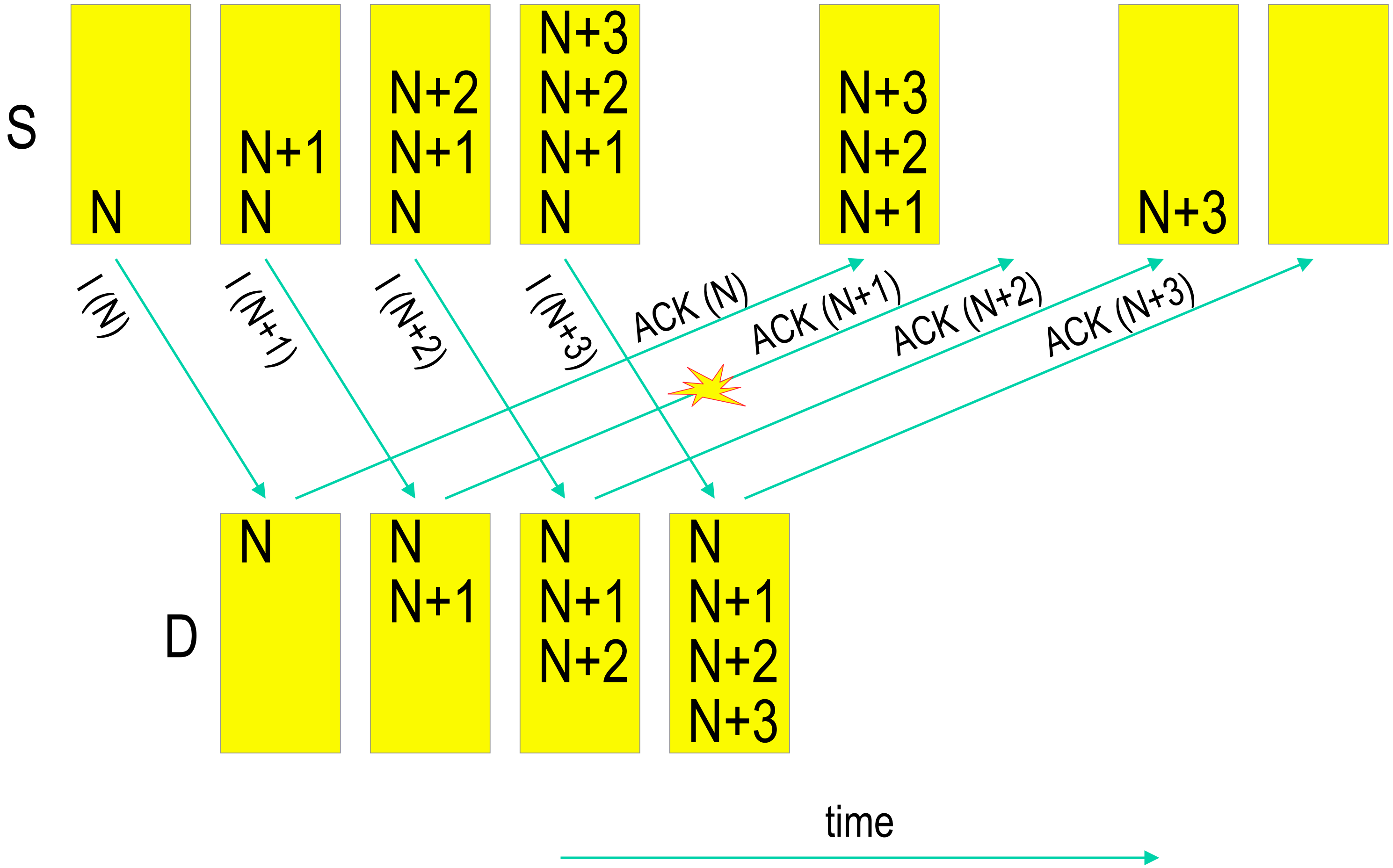
- **Introduction**
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - Flow Control
  - HDLC Overview

Positive Acknowledgement (Scenario 1)

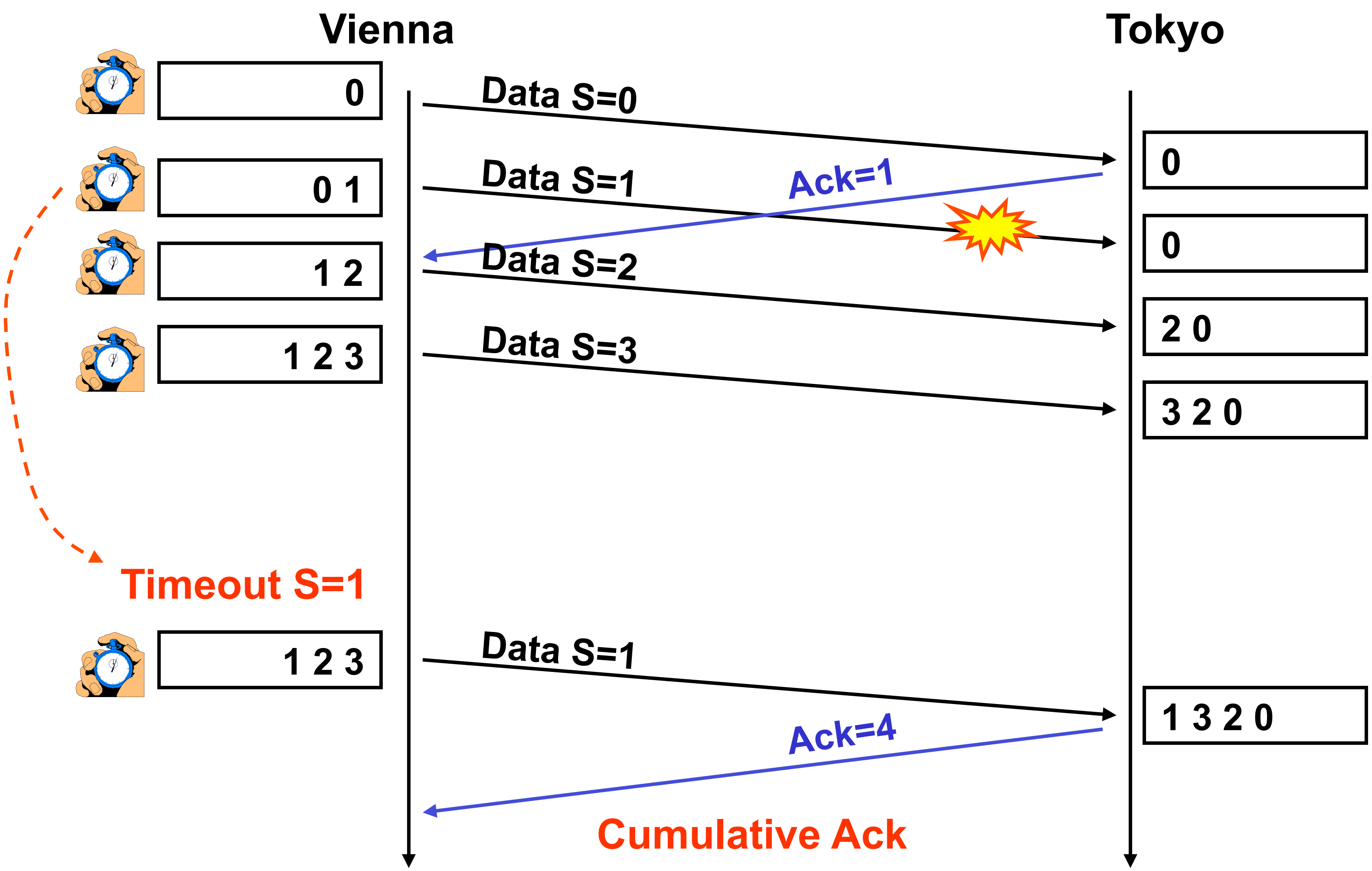


Positive Acknowledgement (Scenario 2)

ACK (N+2) acknowledges all frames up to I (N+2) at S
-> multiple acknowledgment



Positive ACK with Sequence Numbers



Positive ACK - Facts

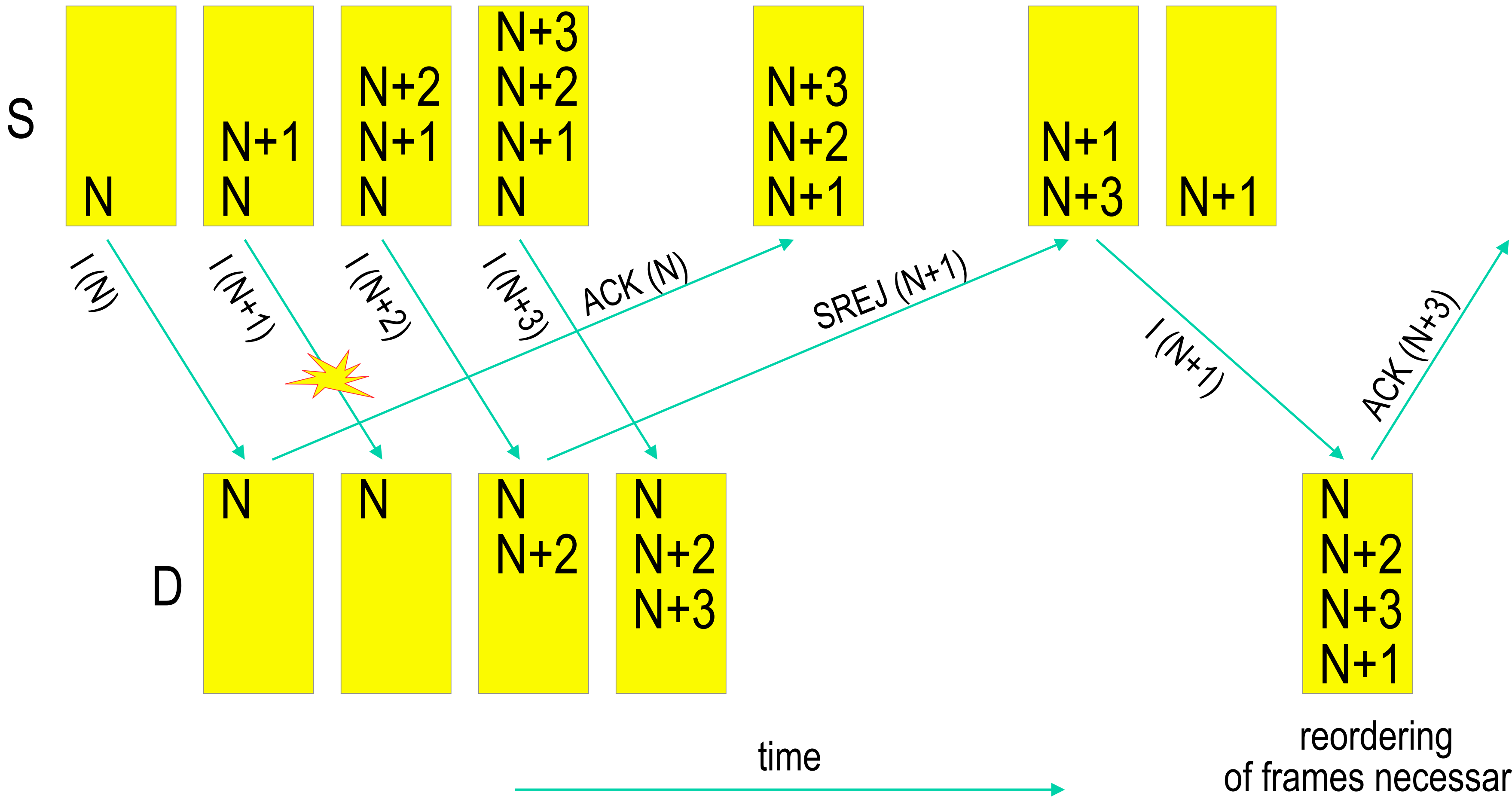
- **Always together with cumulative ACKs**
  – Any frame received is buffered
  – Receiver must be able to reorder
- **Problem:**
  – Only timeouts trigger retransmission
- **Application:**
  – Early (Original) TCP

Agenda

- **Introduction**
- **<u>ARQ Techniques</u>**
  - Introduction
  - Idle RQ
  - <u>Continuous RQ</u>
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - <u>Selective Reject</u>
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - Flow Control
  - HDLC Overview

Selective Reject (Scenario 1)

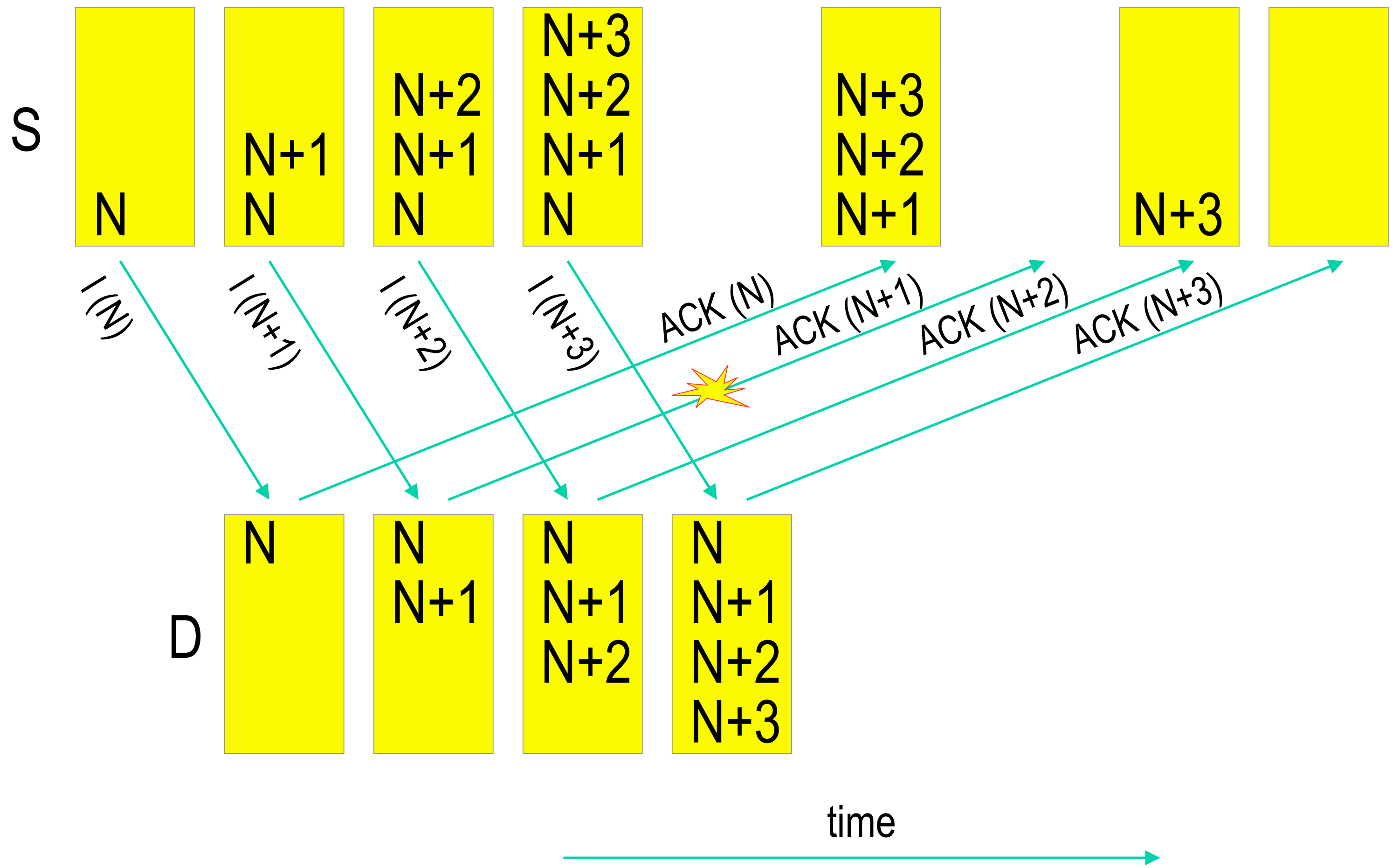
SREJ (N+1) triggers explicitly retransmission of I (N+1) at S,
ACK (N+3) acknowledges all frames up to I (N+3) at S
-> multiple acknowledgment



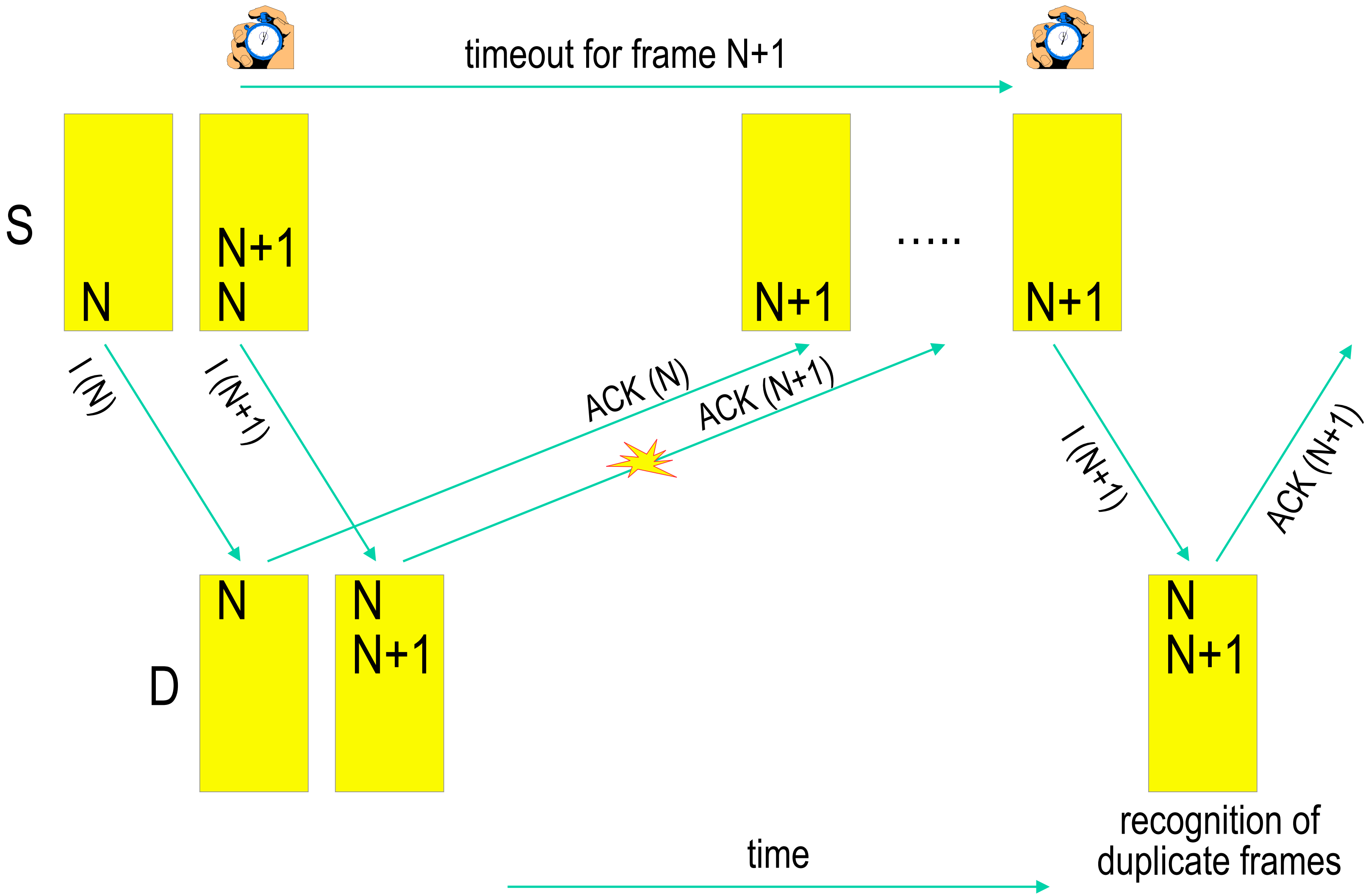
reordering
of frames necessary

Selective Reject (Scenario 2)

ACK (N+2) acknowledges all frames up to I (N+2) at S
-> multiple acknowledgment



Selective Reject (Scenario 3)



Selective Reject - Facts

- **Modern modification of GoBackN**
- **Only those frames are retransmitted that receive a SREJ**
  - Or those that time out
- **Receiver must be able to reorder frames**
- **Application:**
  - Optional for modern HDLC clones
  - SREJ control frame

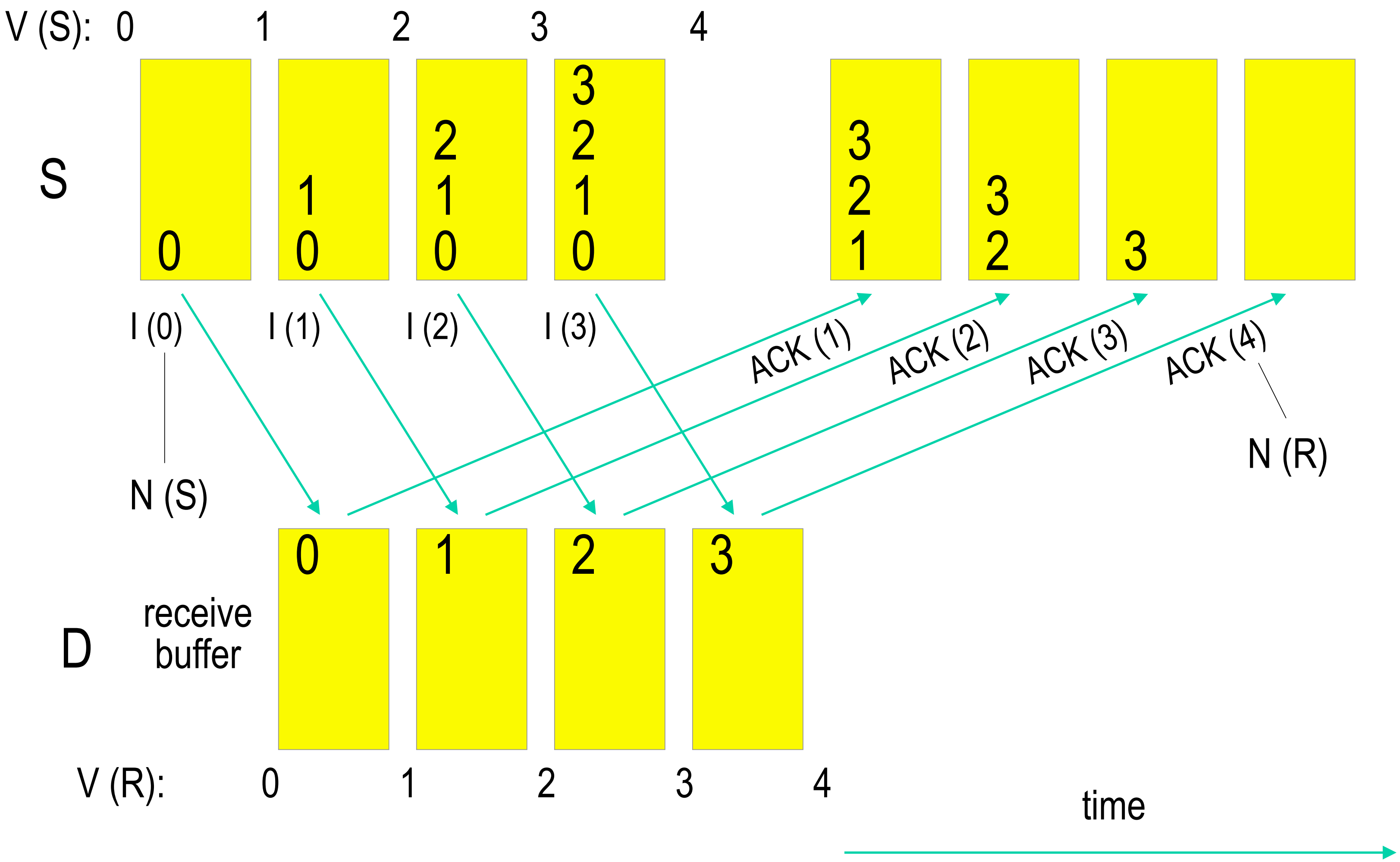
Agenda

- **Introduction**
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - Flow Control
  - HDLC Overview

Sequence Number

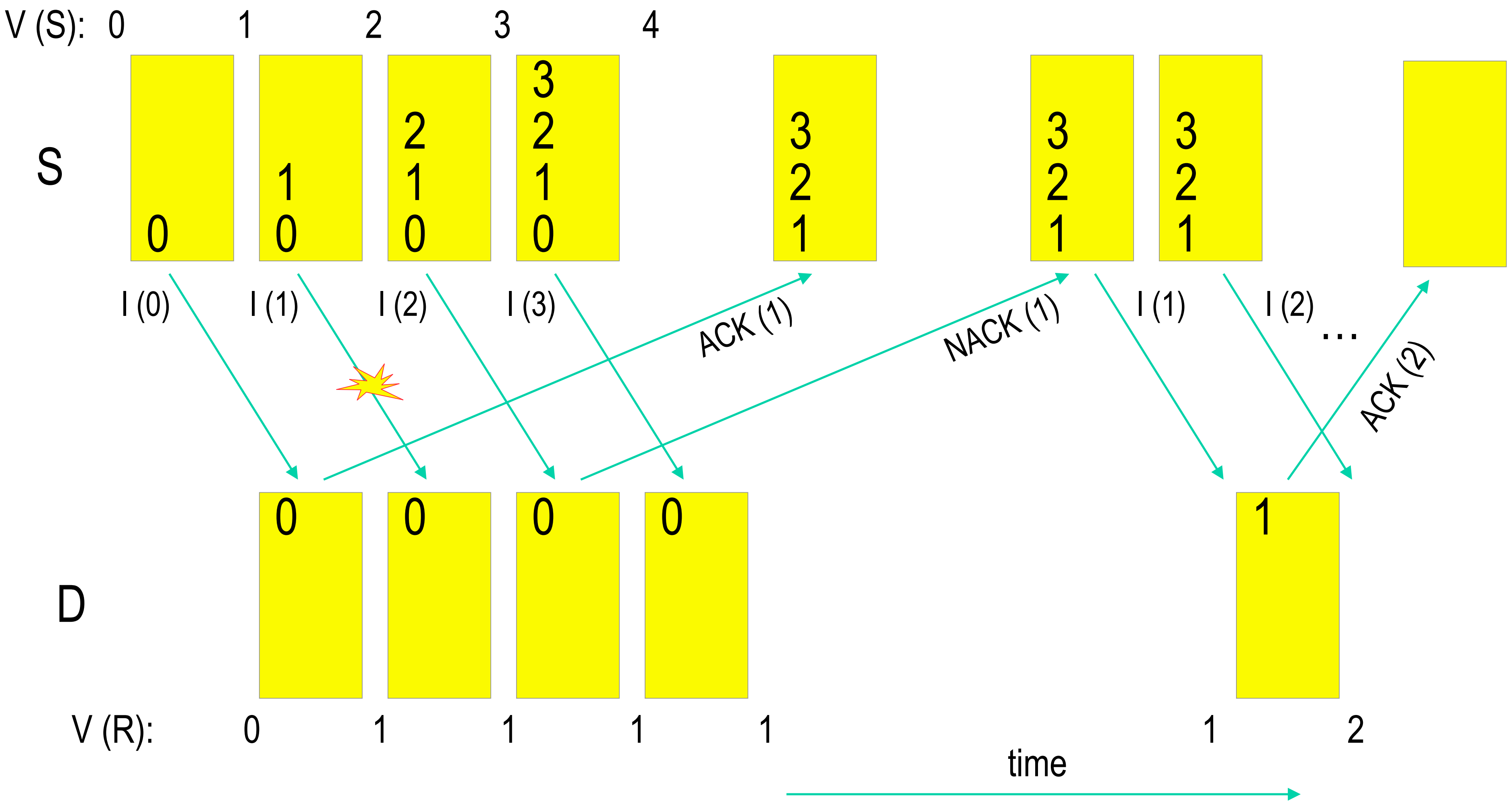
- **Identifiers are implemented by increasing numbers**
- **The number used in I-frames**
  - Send sequence number N(S)
- **The number used in ACK/NACK/SREJ-frames**
  - Receive sequence number N(R)
  - Shows number of the frame which is expected next by the receiver !!!
- **Register variables are necessary for protocol implementation in the finite state machine of sender and receiver**
  - V(S), V(R)
  - Must be initialized (set to 0) by connection setup

GoBackN 1



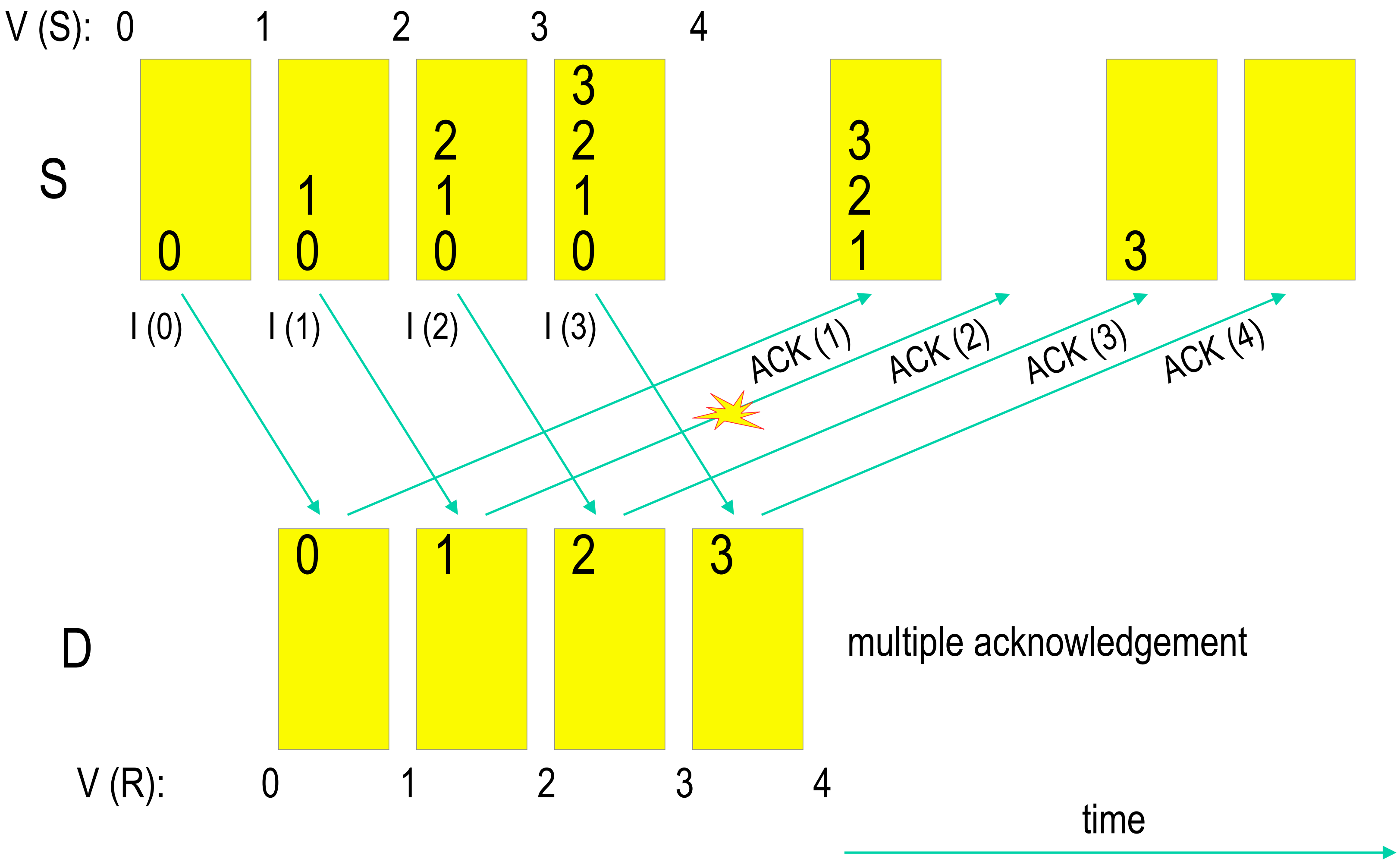
V (S): 0 1 2 3 4
S
0
1 0
2 1 0
3 2 1 0
3 2 1
3 2
3
I (0)
I (1)
I (2)
I (3)
N (S)
ACK (1)
ACK (2)
ACK (3)
ACK (4)
N (R)
D
receive buffer
0
1
2
3
V (R): 0 1 2 3 4
time

GoBackN 2



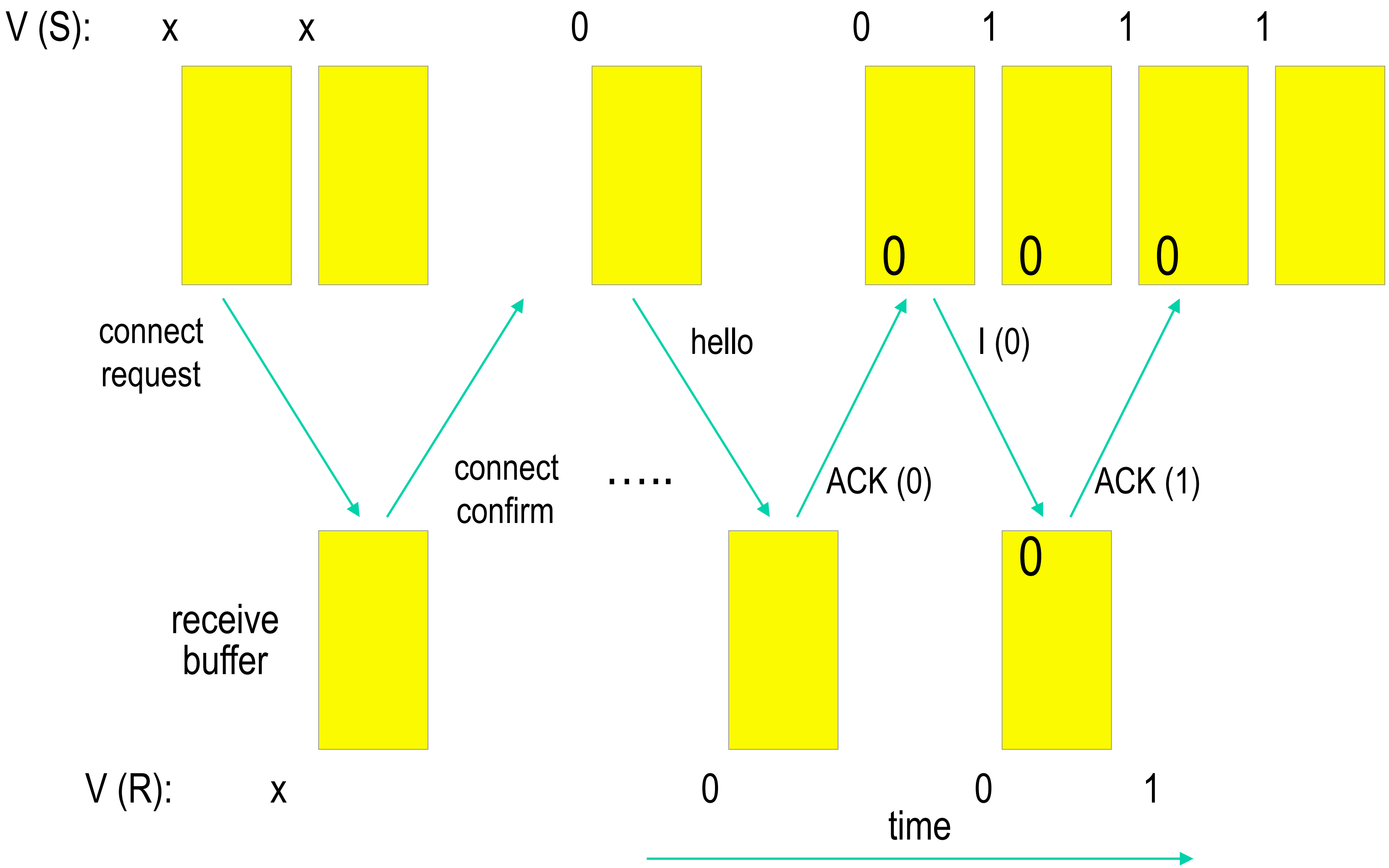
V (S): 0 1 2 3 4
S
0
1
0
2
1
0
3
2
1
0
3
2
1
3
2
1
3
2
1
I (0)
I (1)
I (2)
I (3)
ACK (1)
NACK (1)
I (1)
I (2)
...
ACK (2)
0
0
0
0
1
D
V (R): 0 1 1 1 1 1 2
time

GoBackN 3

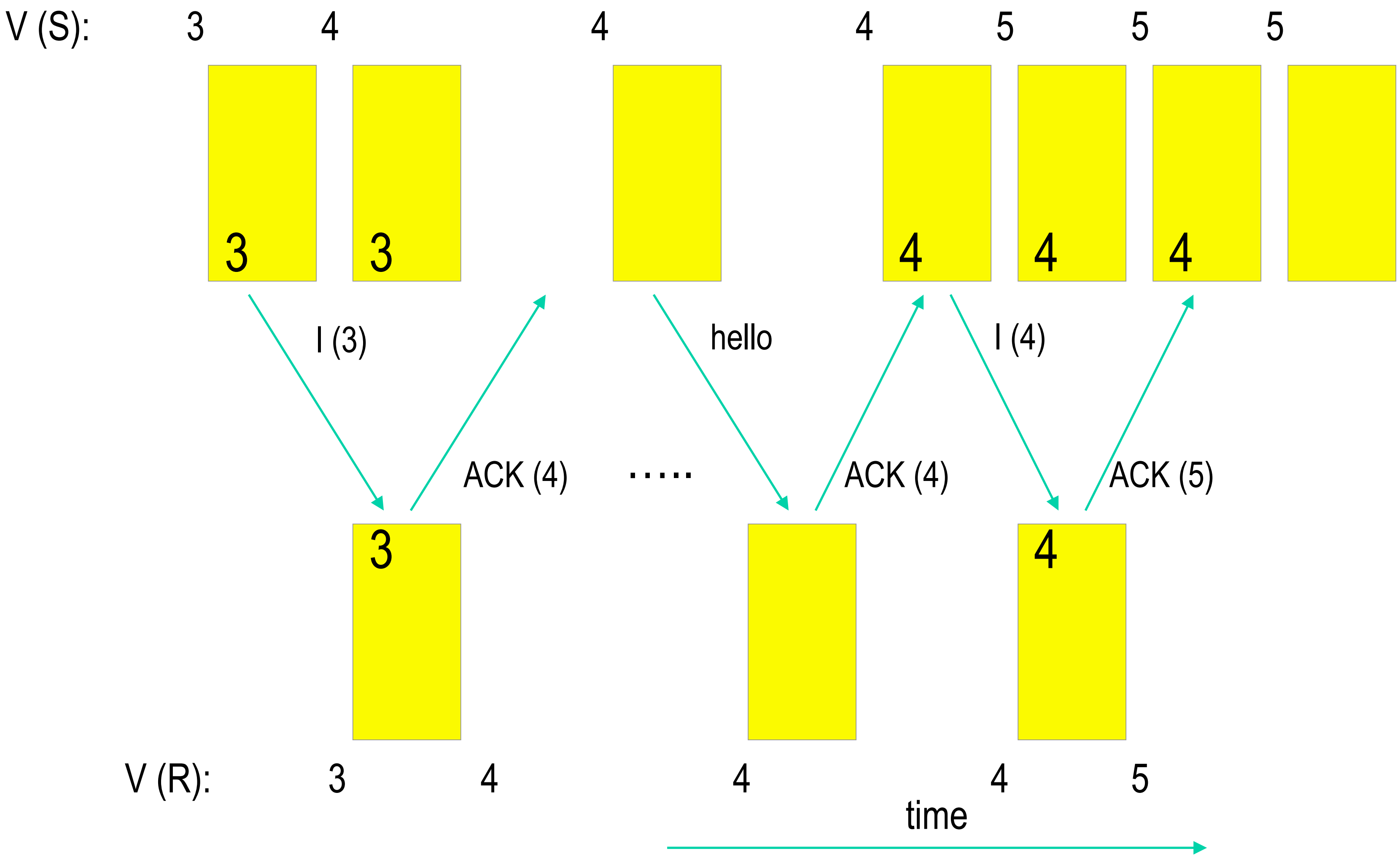


V (S): 0 1 2 3 4
S
0
1
0
2
1
0
3
2
1
0
3
2
1
3
I (0)
I (1)
I (2)
I (3)
ACK (1)
ACK (2)
ACK (3)
ACK (4)
0
1
2
3
D
multiple acknowledgement
V (R): 0 1 2 3 4
time

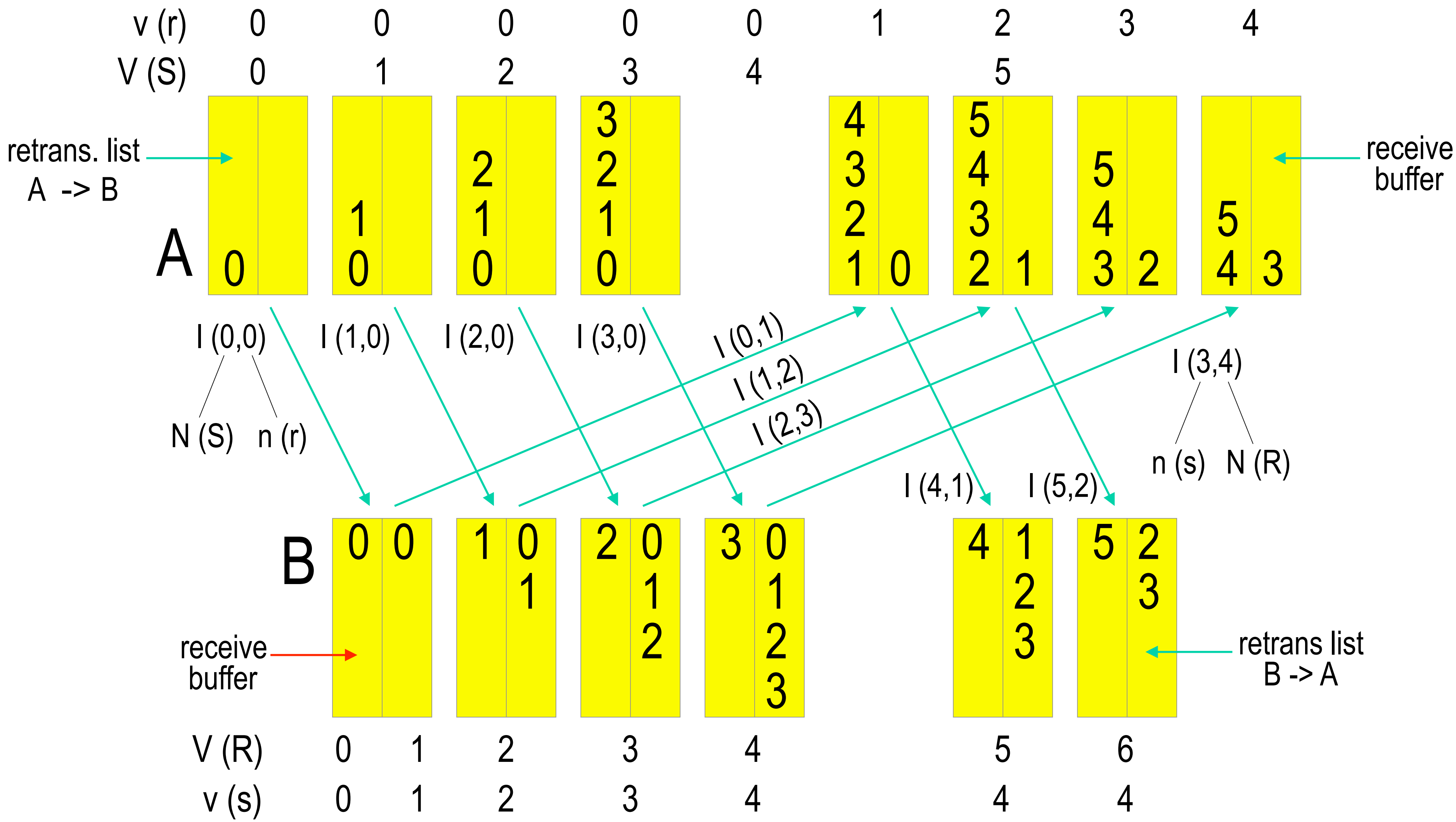
Connection Initializing / Keepalive at the Beginning
Reason why Nr(1) acknowledges frame I(0) !



Keepalive in Between



Piggyback Acknowledgement



Timers - Retransmission Timeout

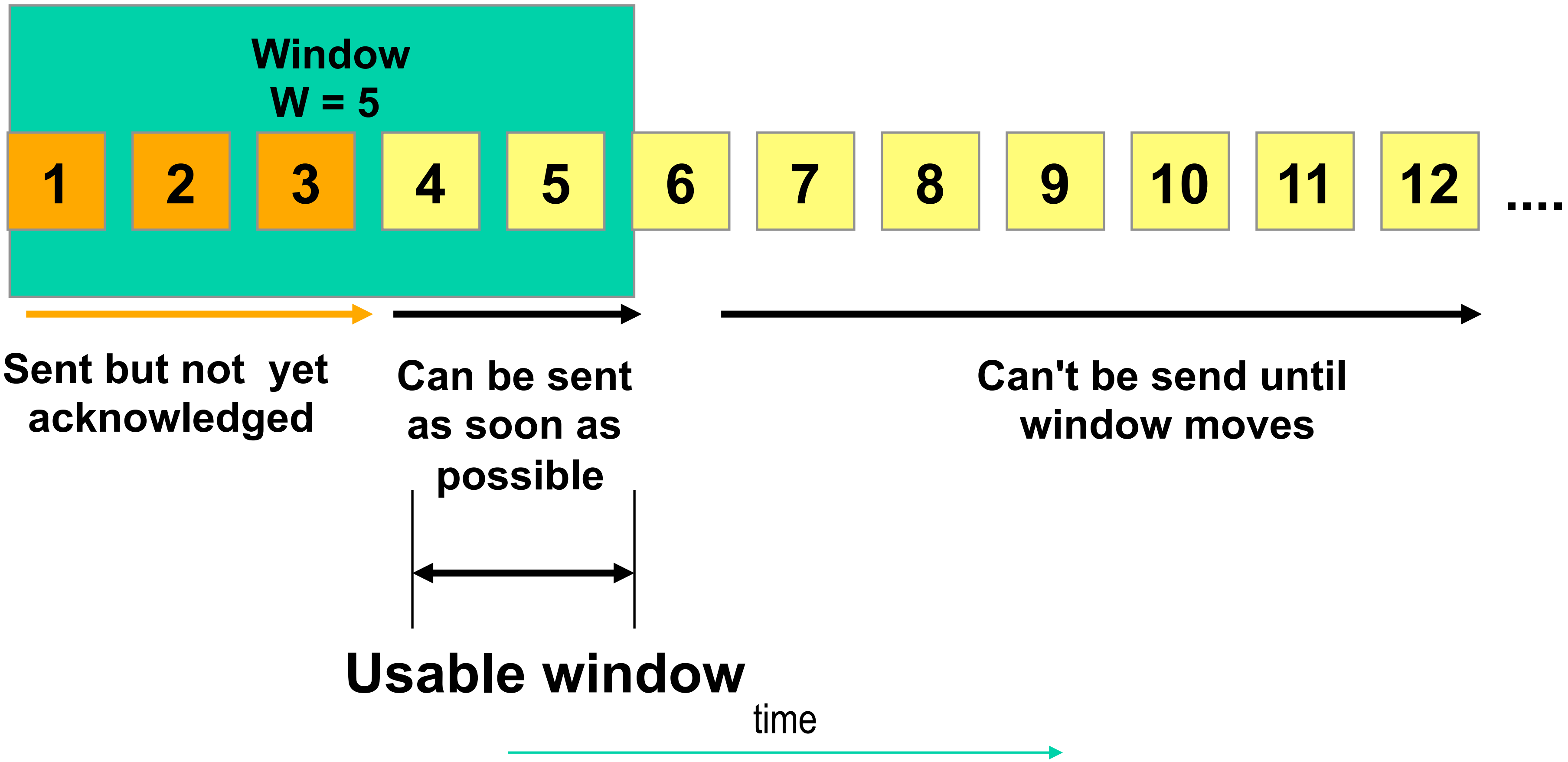
- **The value for retransmission timeouts for line protocols can be easily calculated using the following parameters**
  - Bitrate
  - Maximum data frame size
  - Worst case time at receiver to generate an acknowledgment
  - Size of acknowledgment frame
- **Calculation for network protocols with varying transmission delays is more complex**
  - Adaptive process is necessary

Send Window and Sliding Window

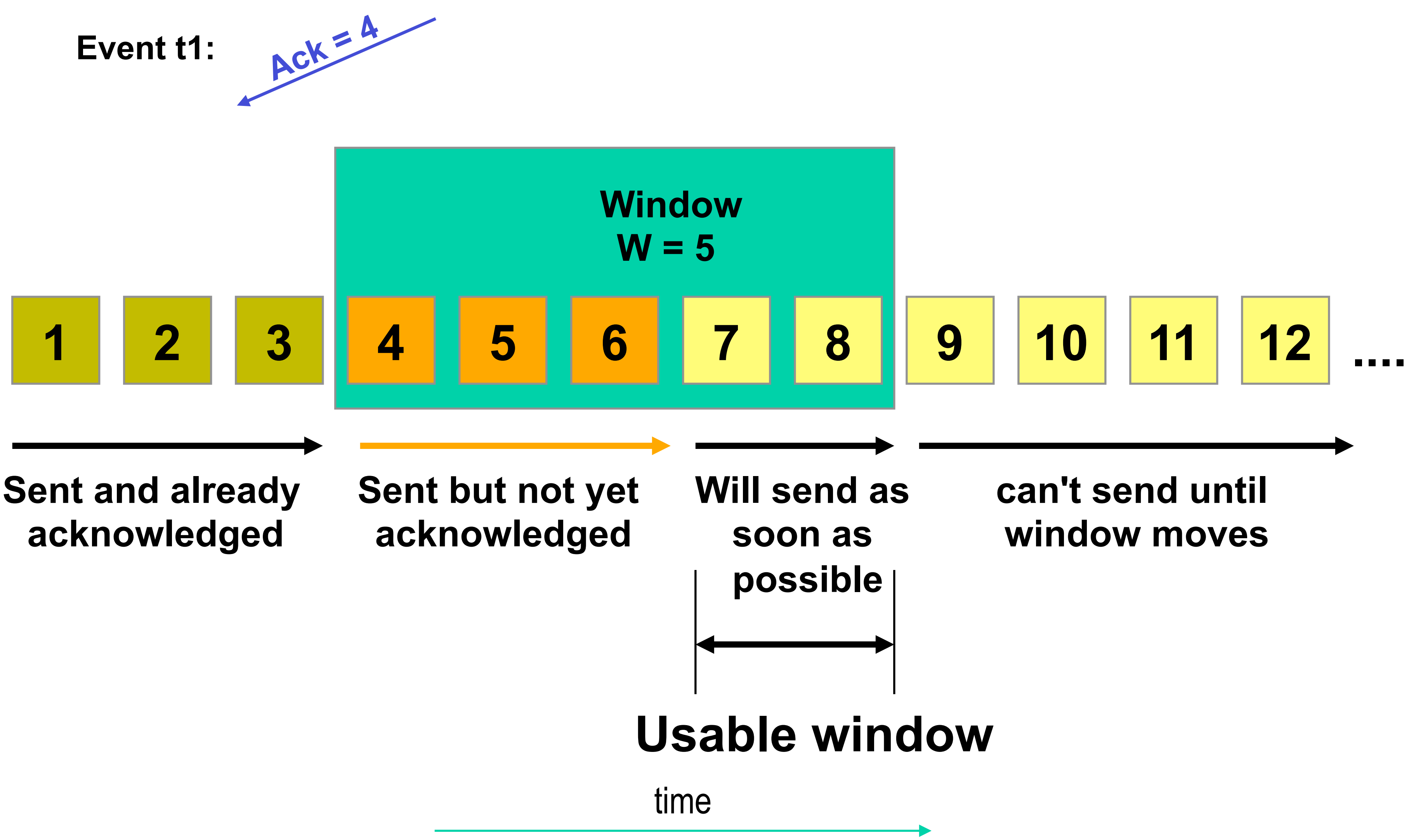
- **Continuous-RQ techniques would require infinite sender/receiver buffer size (also infinite number of identifiers)**
  - If we do not restrict the number of unacknowledged frames
- **Send Window W**
  - Is the maximum allowed number of unacknowledged frames in the retransmission list
- **Necessary sender-buffer size is W * maximum frame size**
  - Also called the **Window Size**
- **Handling procedure**
  - Is called **(Sliding) Windowing**

Sliding Window Basics (1)

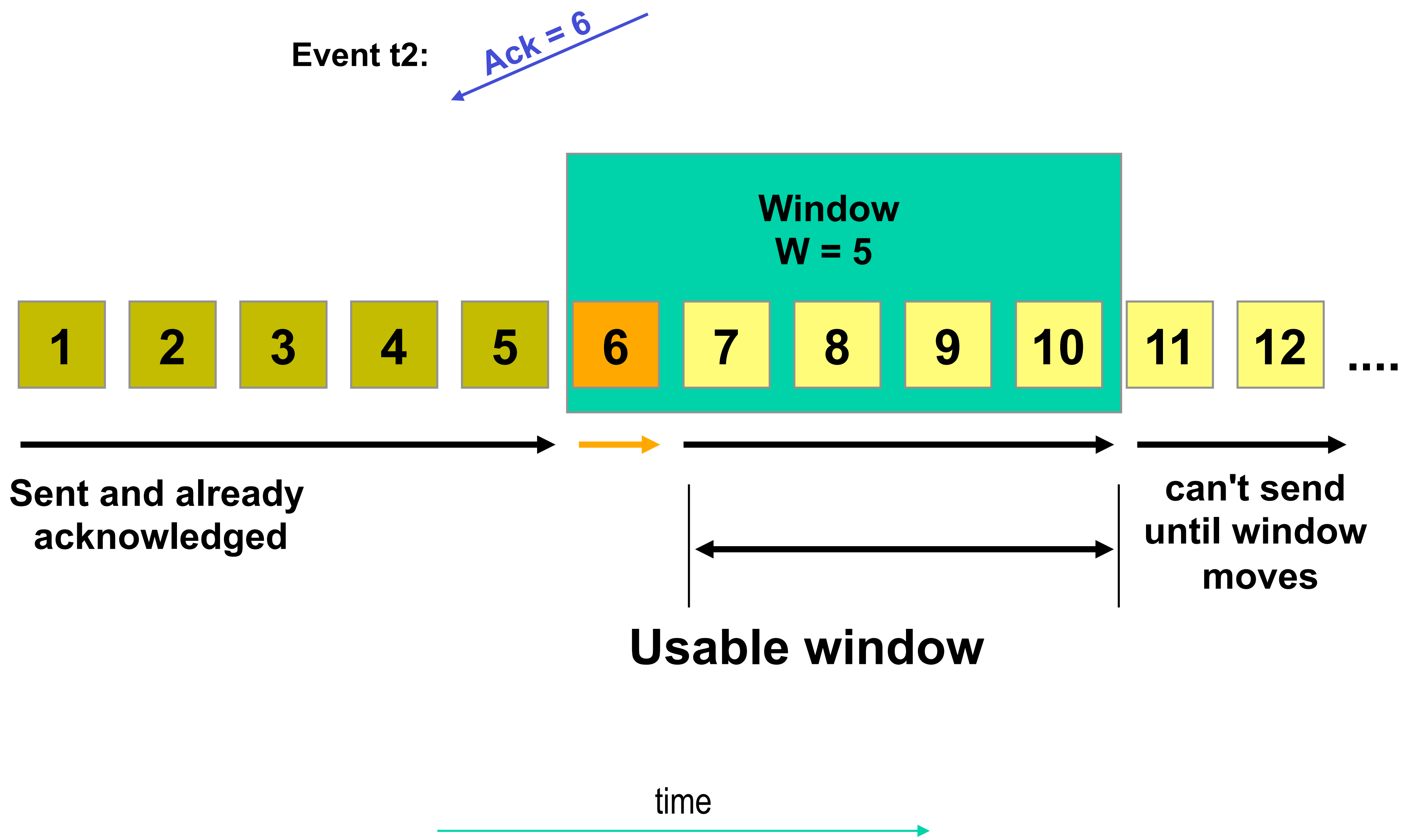
Event t0:



Sliding Window Basics (2)

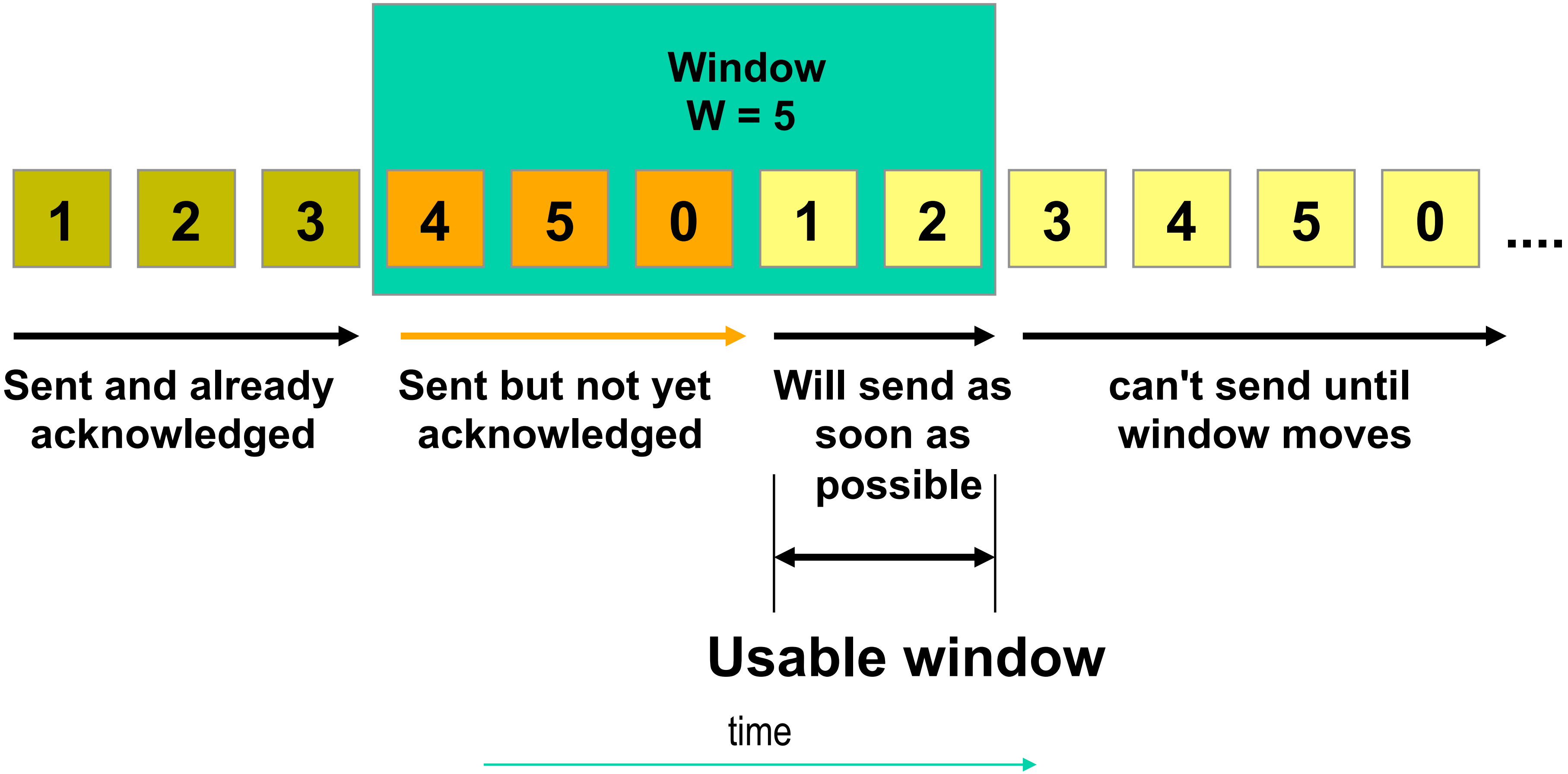


Sliding Window Basics (3)

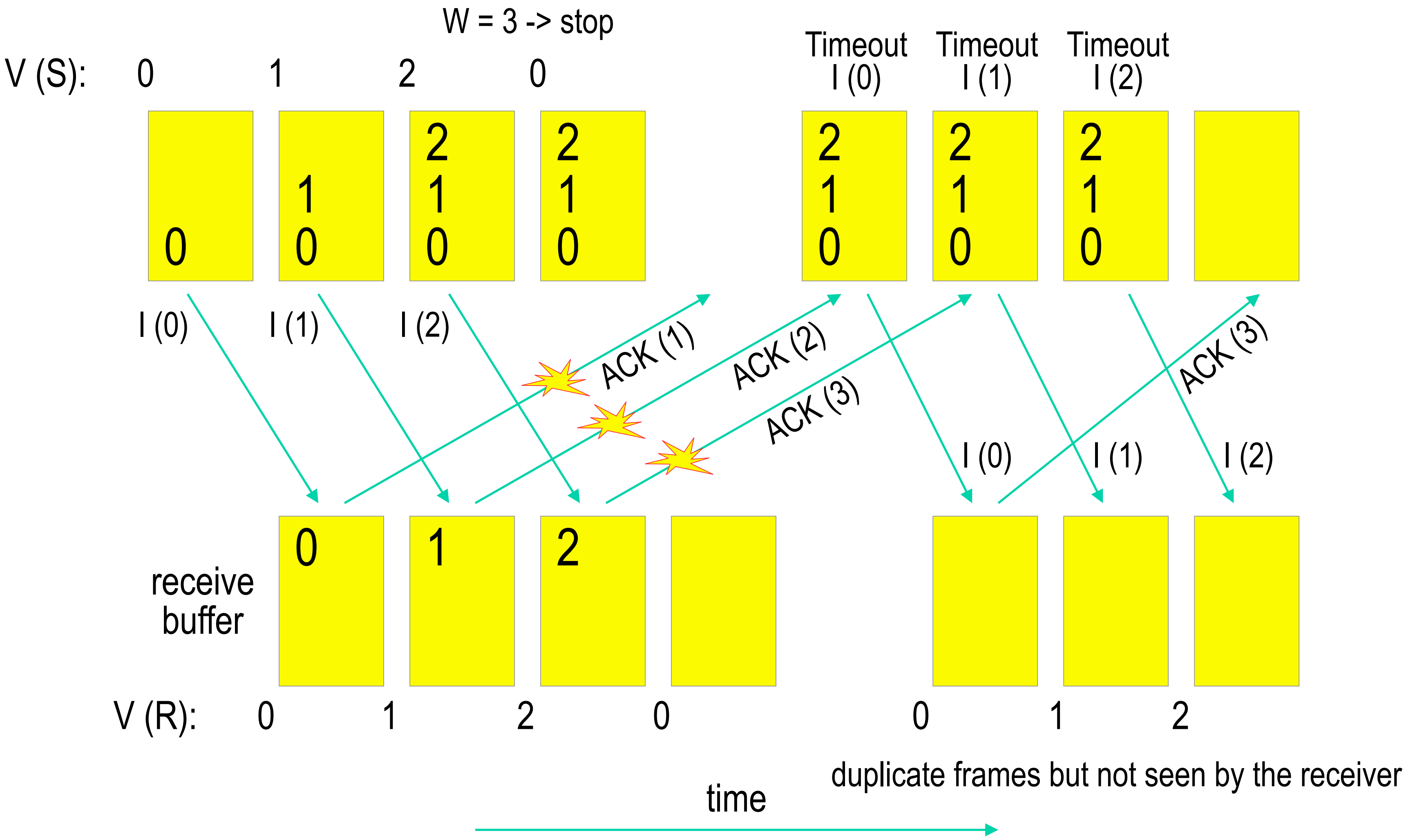


Windowing with Numbering Modulo W+1

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



Worst Case Scenario with GoBackN and W=3



Agenda

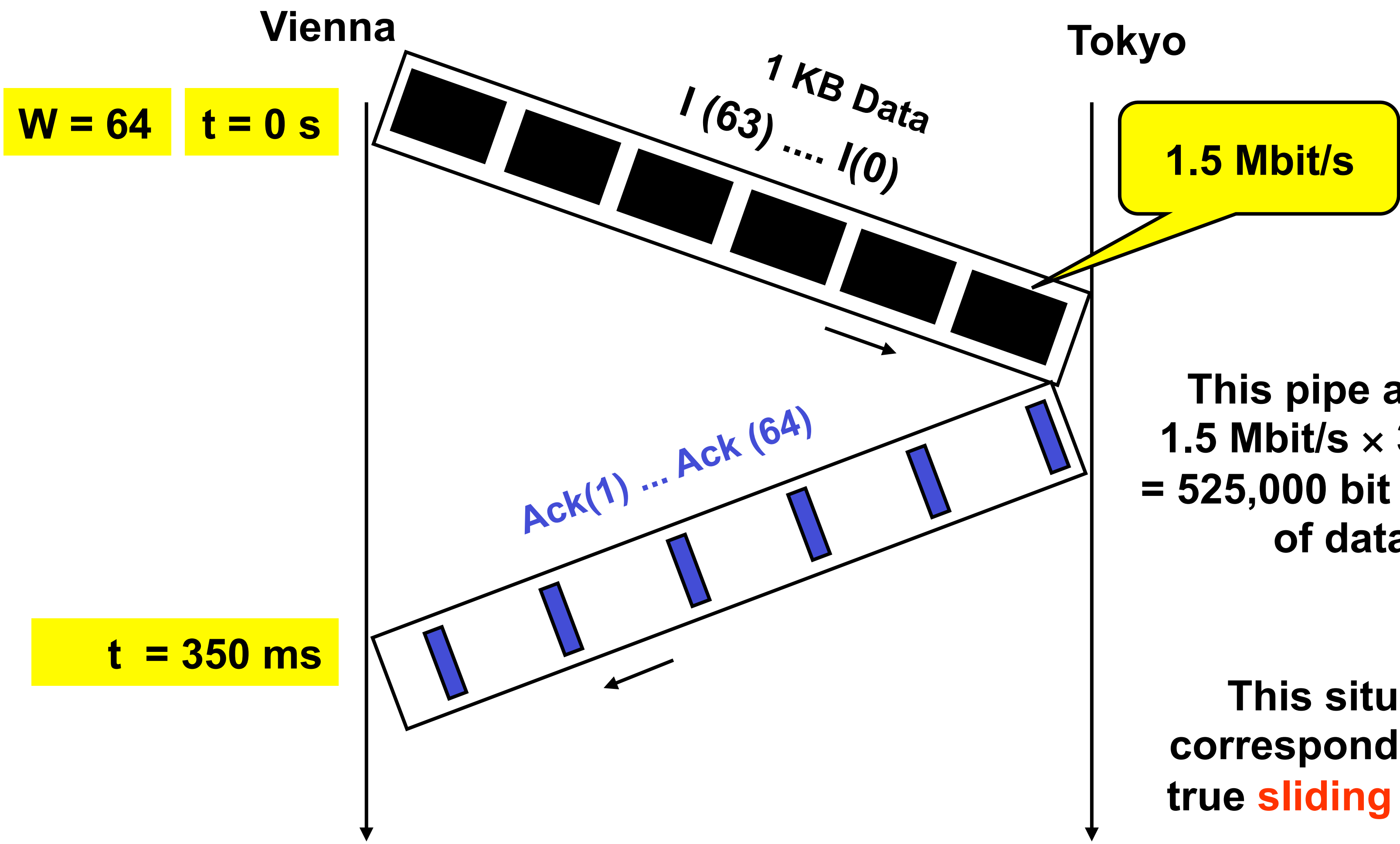
- **Introduction**
- **<u>ARQ Techniques</u>**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - <u>Bandwidth-Delay Product</u>
  - Flow Control
  - HDLC Overview

How Large should be the Window Size?

- **Window size depends on**
  - Bandwidth (bit rate) of the communication channel
  - Round-Trip-Time (RTT)
    - = 2 x (propagation + serialization) delay plus response delay of partner
  - (Available buffer size transmitter/receiver)
- **The optimum window size**
  - must be big enough so that the sender can fully utilize the channel volume which is given by the

    Bandwidth-Delay Product
  - Window size in Bytes >= BW × RTT
  - If smaller: jumping window
  - Extreme case: Idle-RQ with W=1

Full Pipe with Continuous-RQ and W = 64

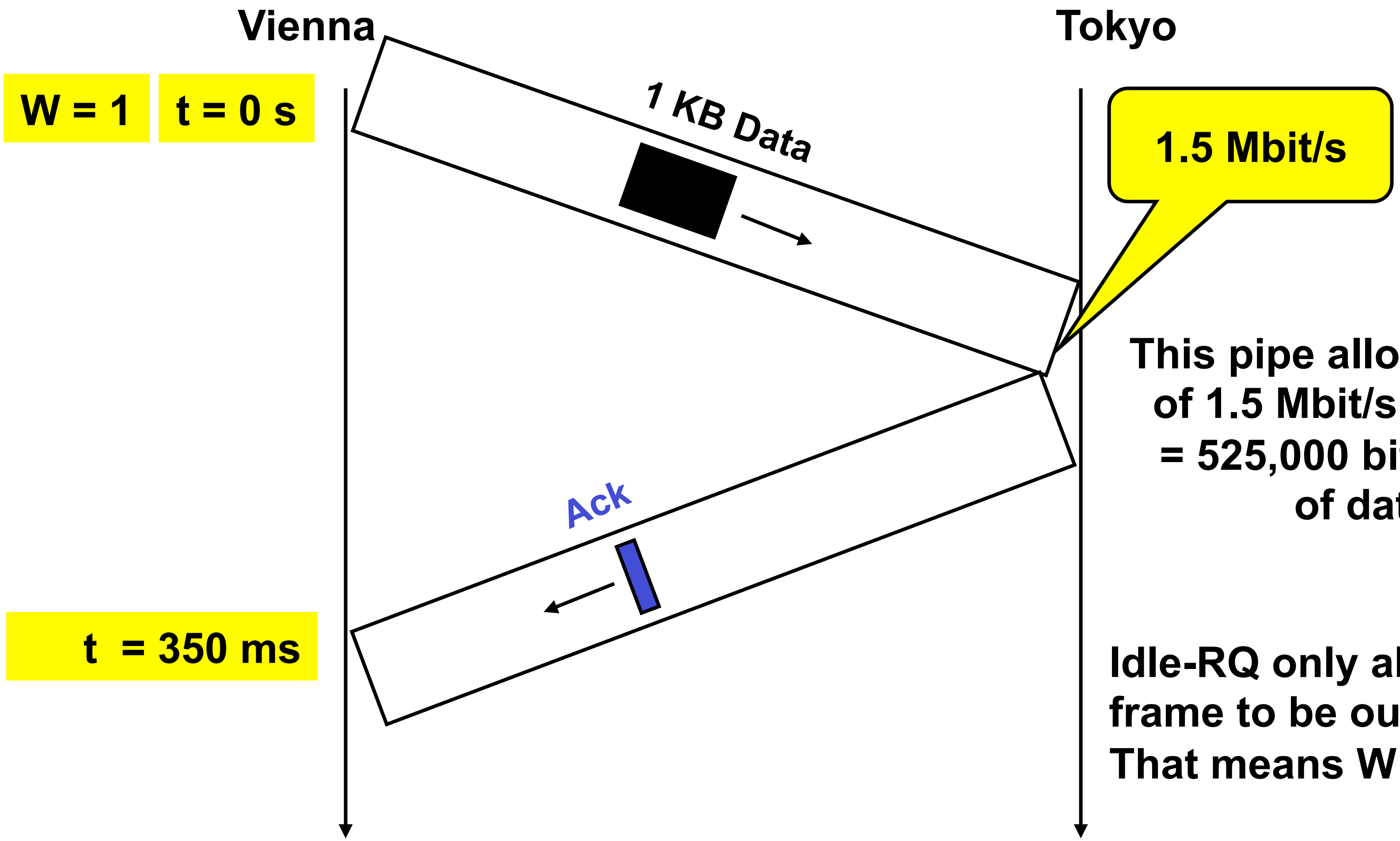


This pipe allows 1.5 Mbit/s × 350 ms = 525,000 bit ≅ 64 KB of data.

This situation corresponds with a true sliding window

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

Nearly Empty Pipe with Idle-RQ and W = 1

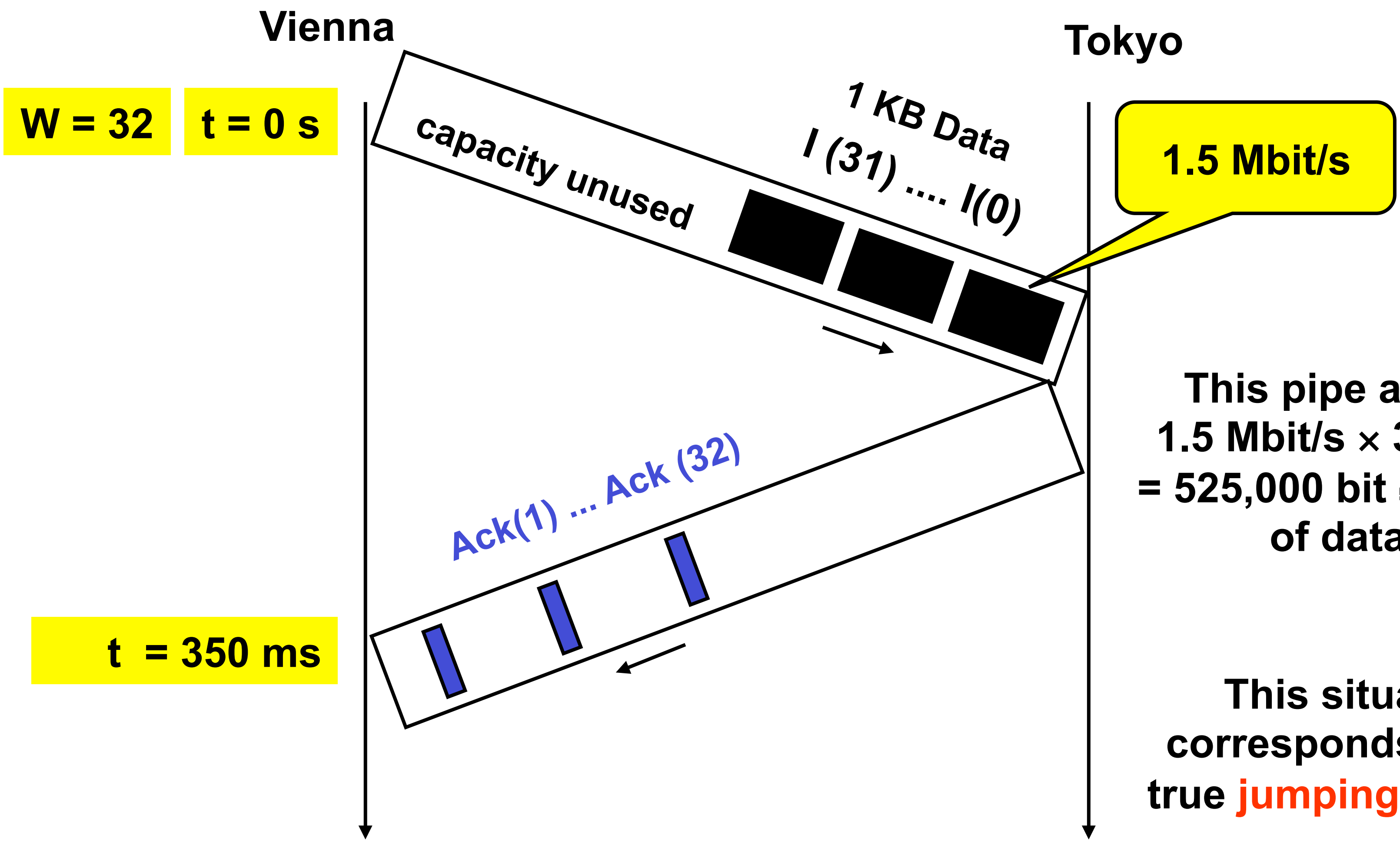


This pipe allows storing of 1.5 Mbit/s × 350 ms = 525,000 bit ≅ 64 KB of data.

Idle-RQ only allows one frame to be outstanding. That means 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

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

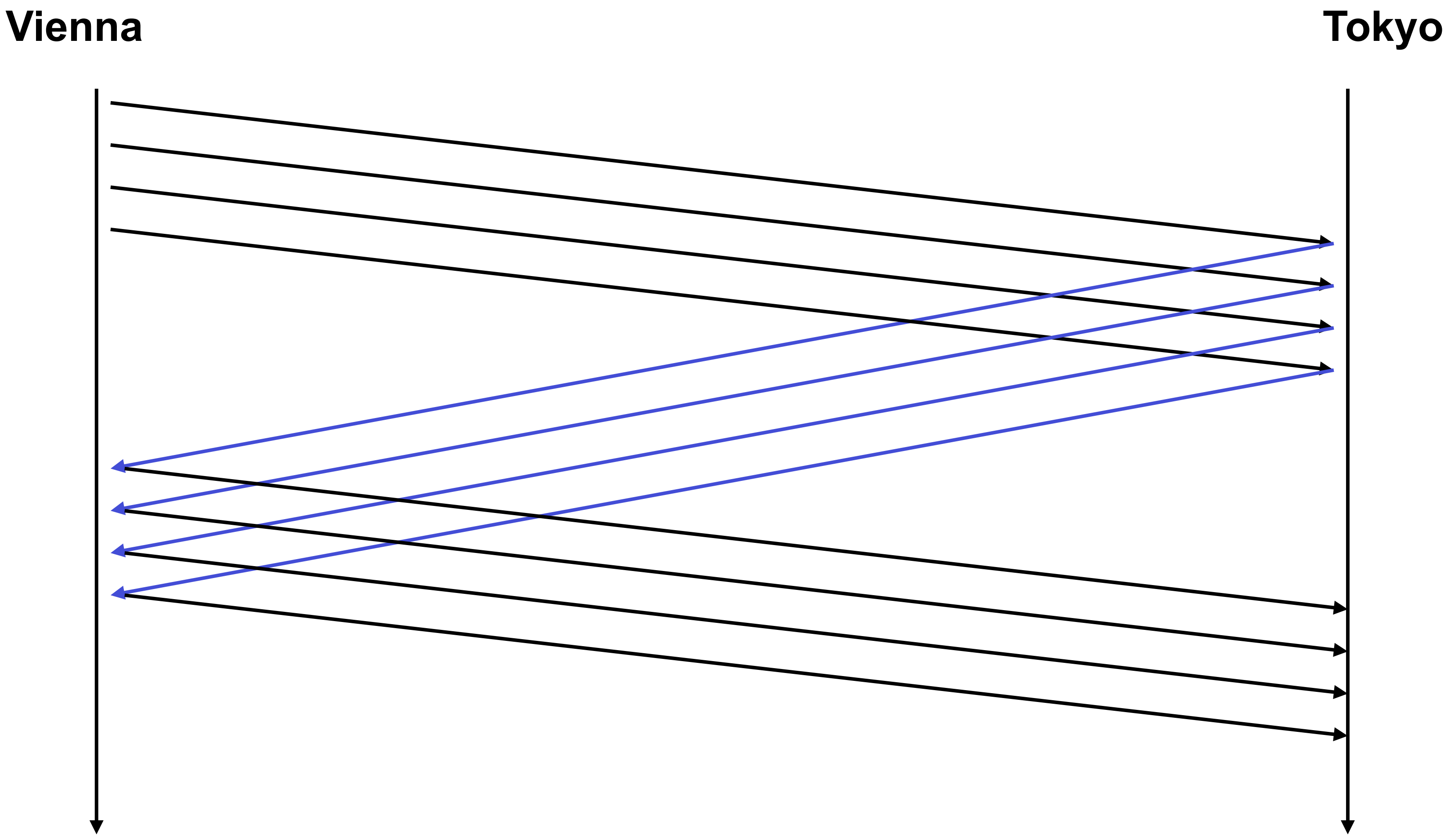


This pipe allows 1.5 Mbit/s × 350 ms = 525,000 bit ≅ 64 KB of data.

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 !!!

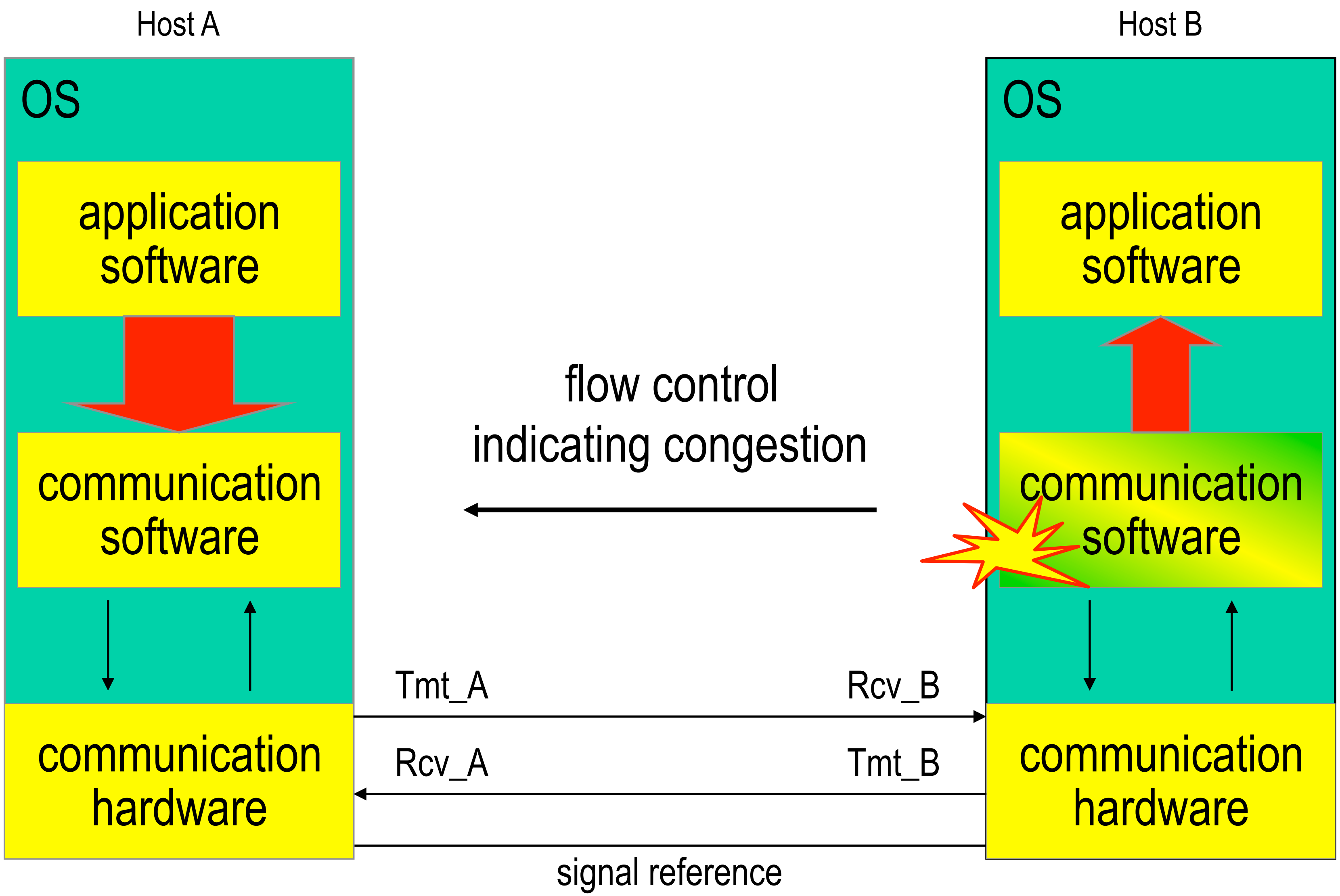
Jumping Window



Agenda

- **Introduction**
- **<u>ARQ Techniques</u>**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - <u>Flow Control</u>
  - HDLC Overview

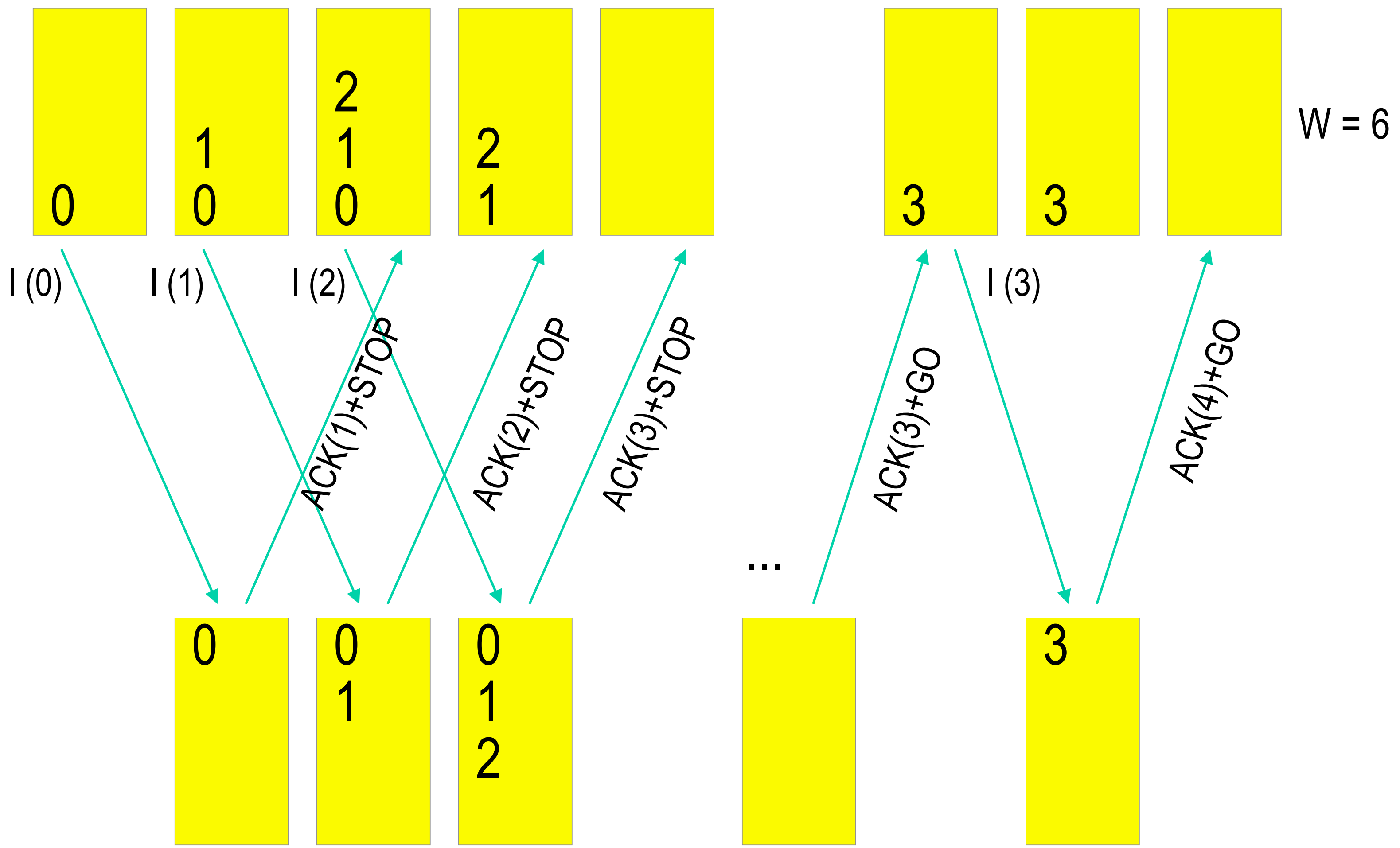
Flow Control



Flow Control Possibilities

- **Misuse of traditional windowing**
  - Constant send window during session time
  - Receiver stops acknowledgements in case of congestion
  - Not a good approach
- **Separate flow control frames**
  - Stop and Go indications
- **Adaptive windowing**
  - Variable send window during session time
  - Implicitly contains Stop and Go indication

Flow Control: Stop and Go



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 of free buffer memory)
    - Advertised window size = 0 -> STOP
    - Advertised window size > 0 -> GO
  - Adaptive windowing (e.g used by TCP)

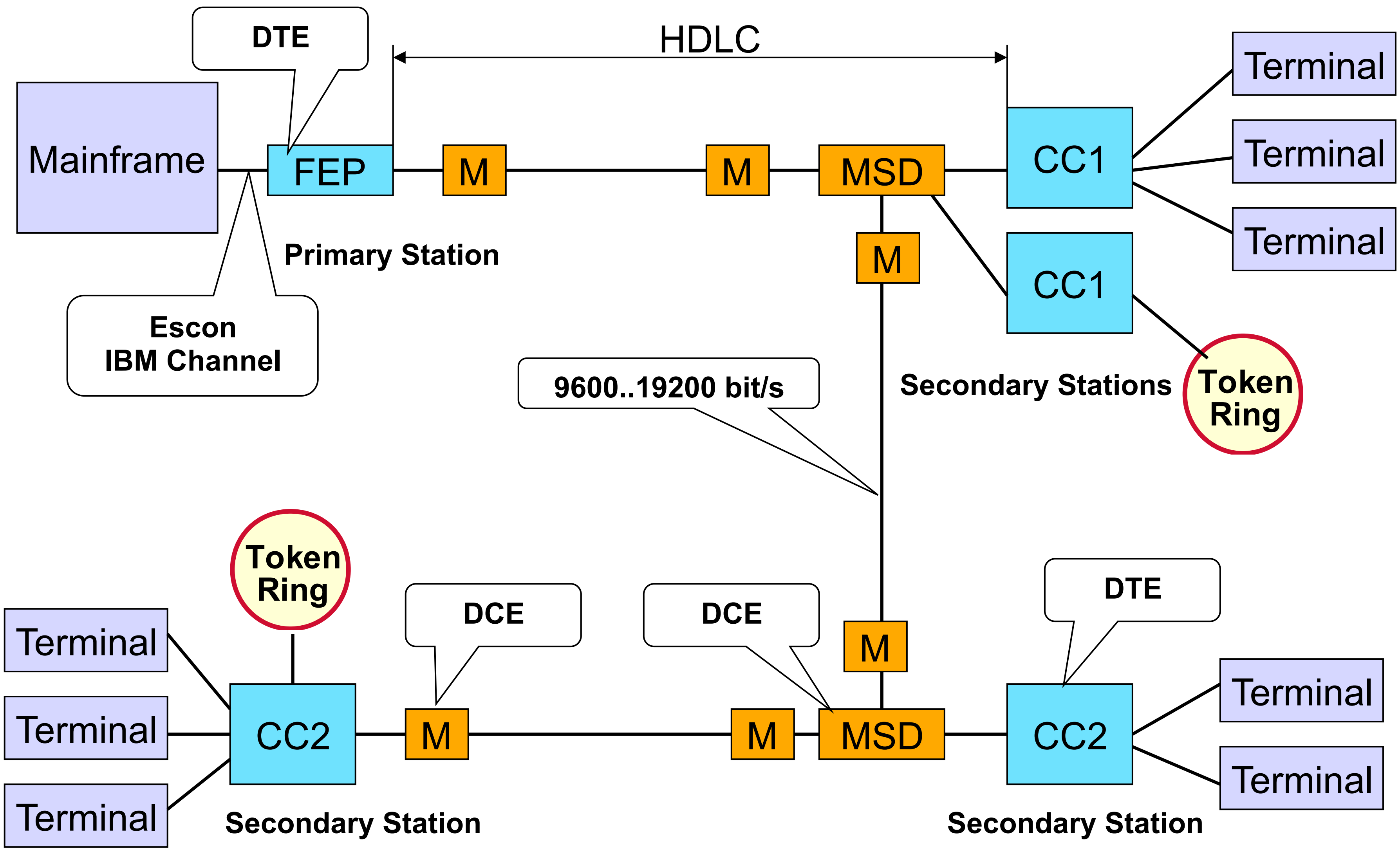
Agenda

- **Introduction**
- **ARQ Techniques**
  - Introduction
  - Idle RQ
  - Continuous RQ
    - Selective Acknowledgement
    - GoBackN
    - Positive Acknowledgement
    - Selective Reject
  - Sequence Numbers and Windowing
  - Bandwidth-Delay Product
  - Flow Control
  - HDLC Overview

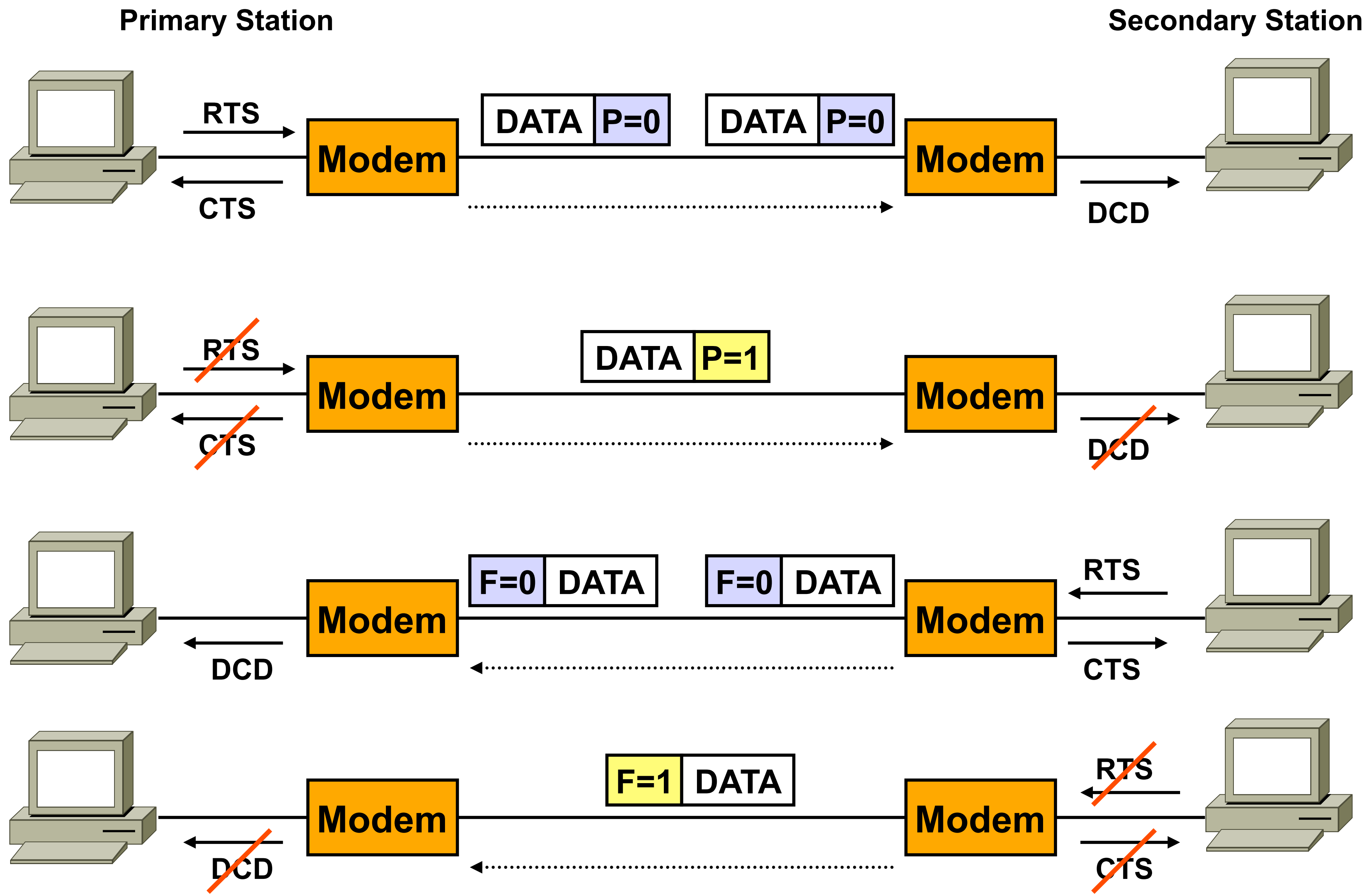
What is HDLC ?

- **High-Level Data Link Control**
- **Early link layer protocol**
- **Based on SDLC (Synchronous-DLC, IBM)**
  - Access control on half-duplex modem-lines
  - Connection-oriented or connectionless
  - Framing
  - Frame Protection
- **Building elements**
  - Synchronous transmission
  - Bit-oriented line protocol using bit-stuffing
  - Continuous RQ with GoBackN, piggybacked ACK
  - P/F procedure for access control and check-pointing
  - Flow Control based on STOP and GO
- **Mother of many LAN and WAN protocols**

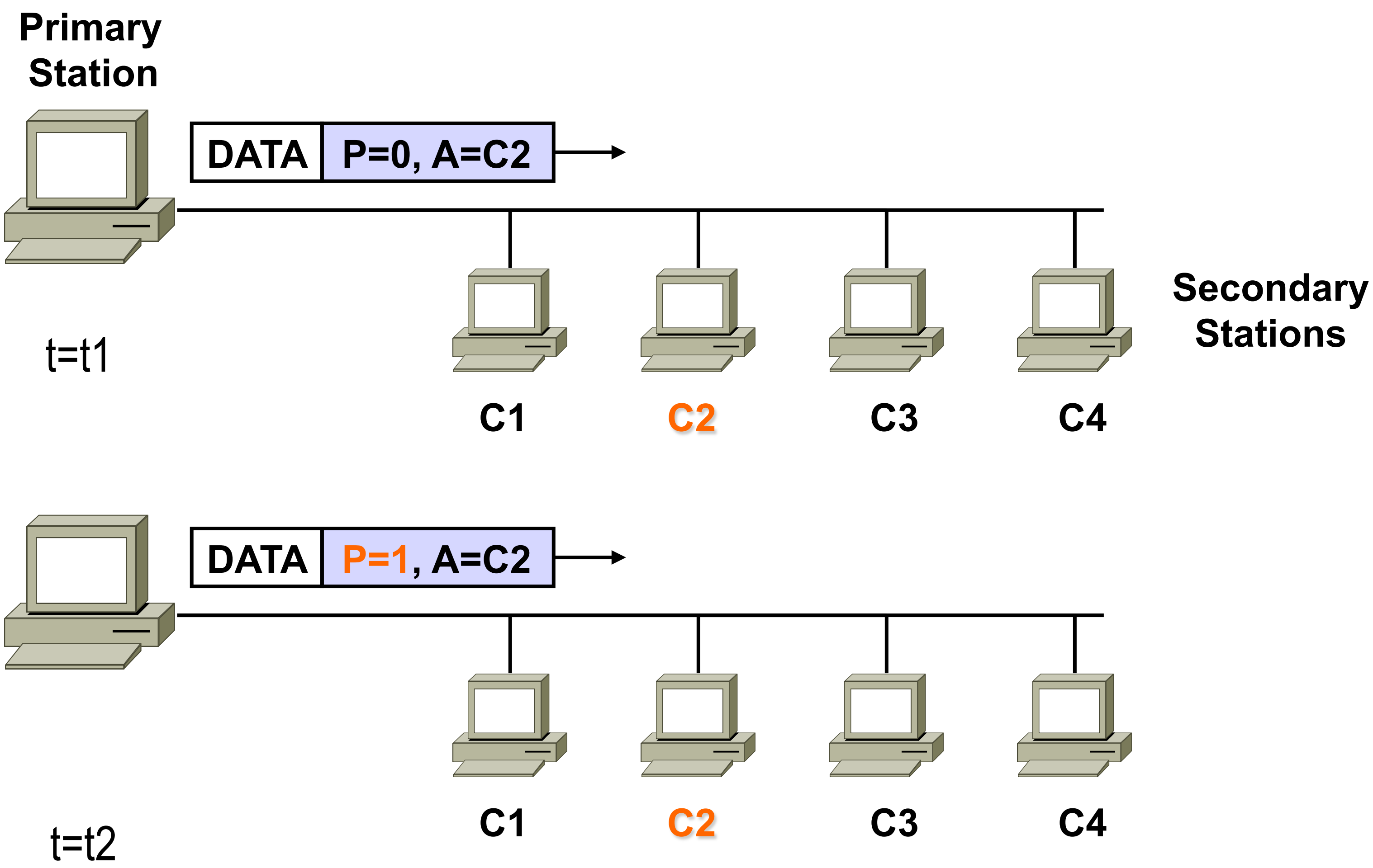
Early HDLC Example



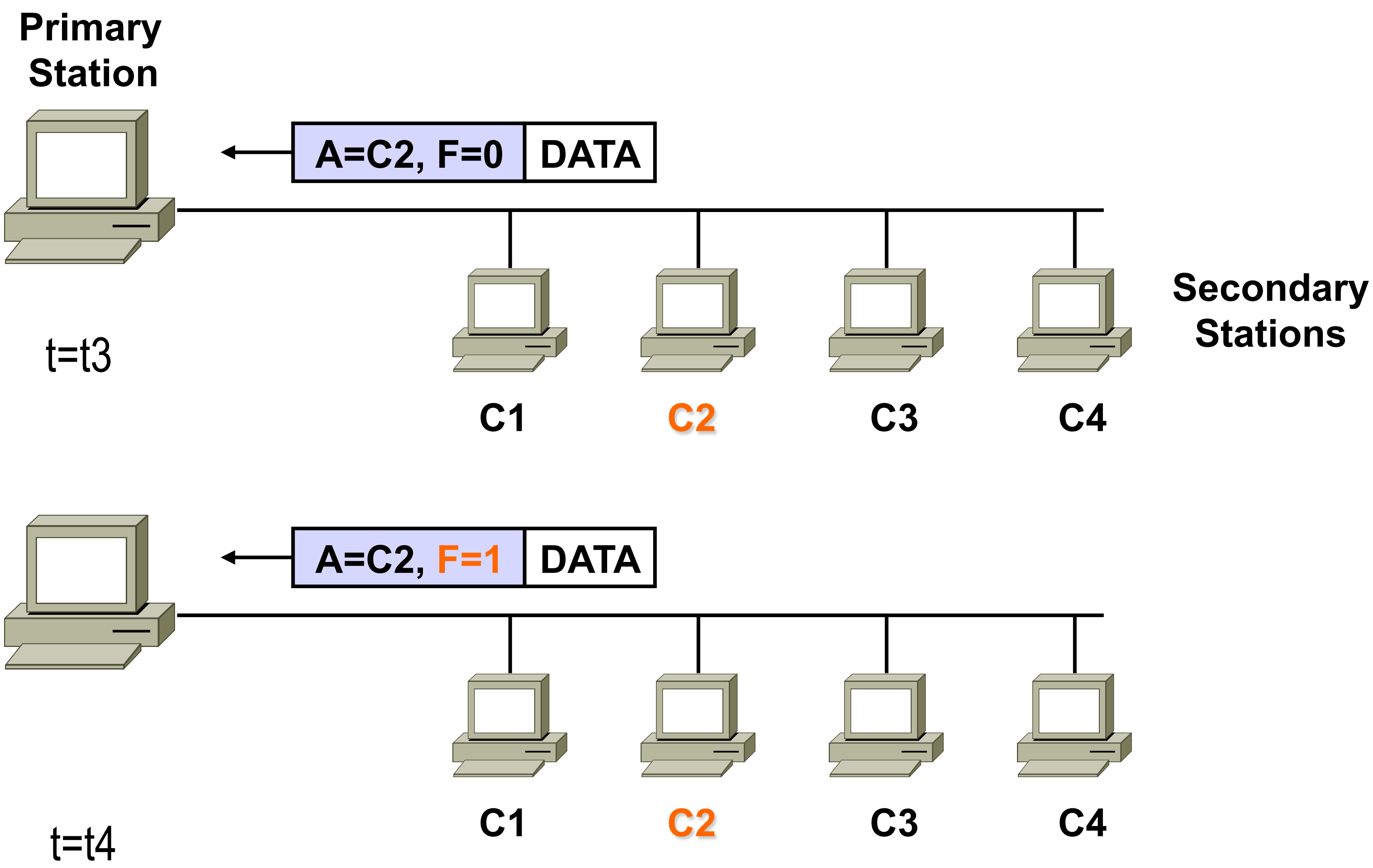
Half-Duplex Management



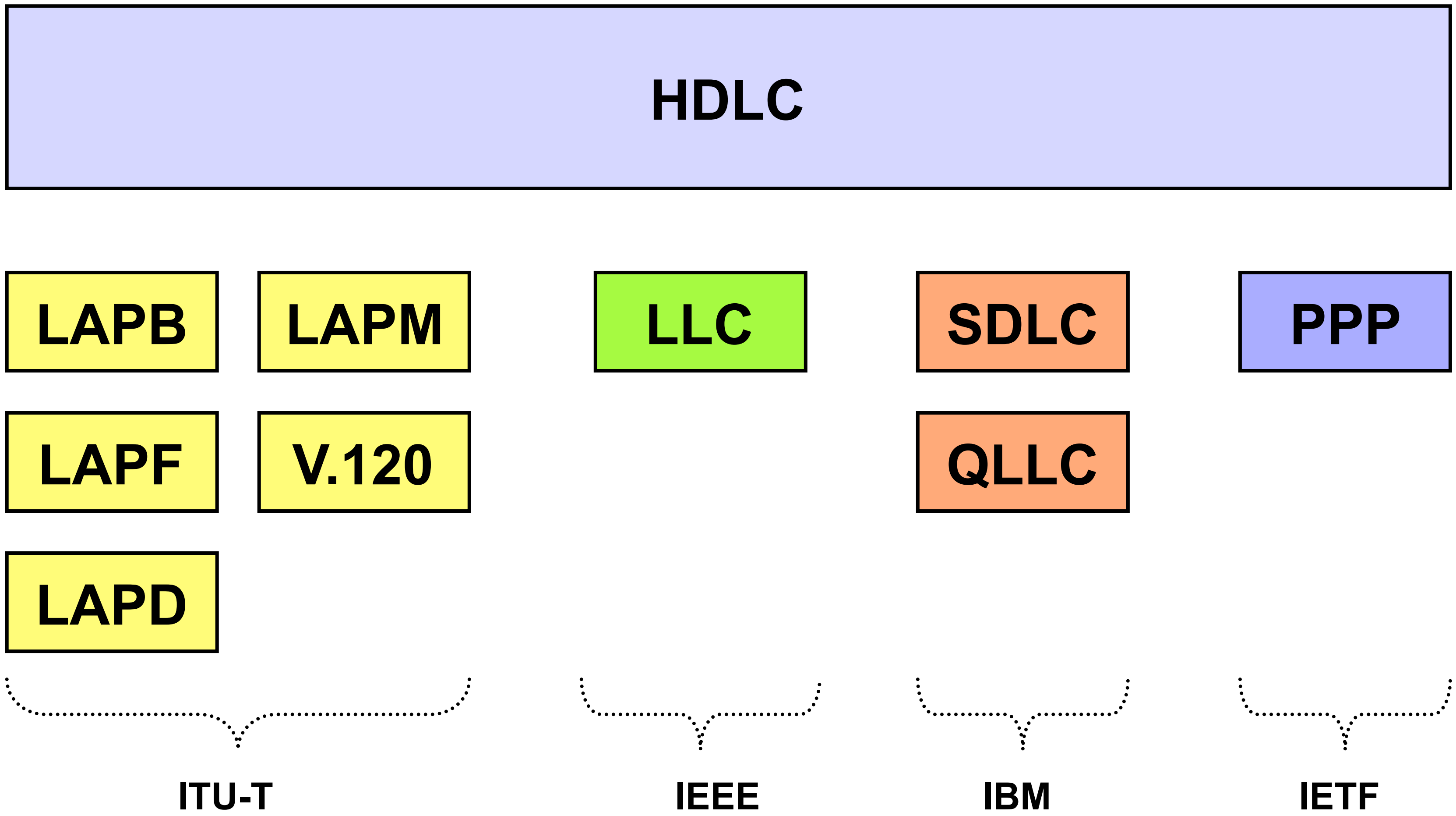
Same on Multipoint Lines (1)



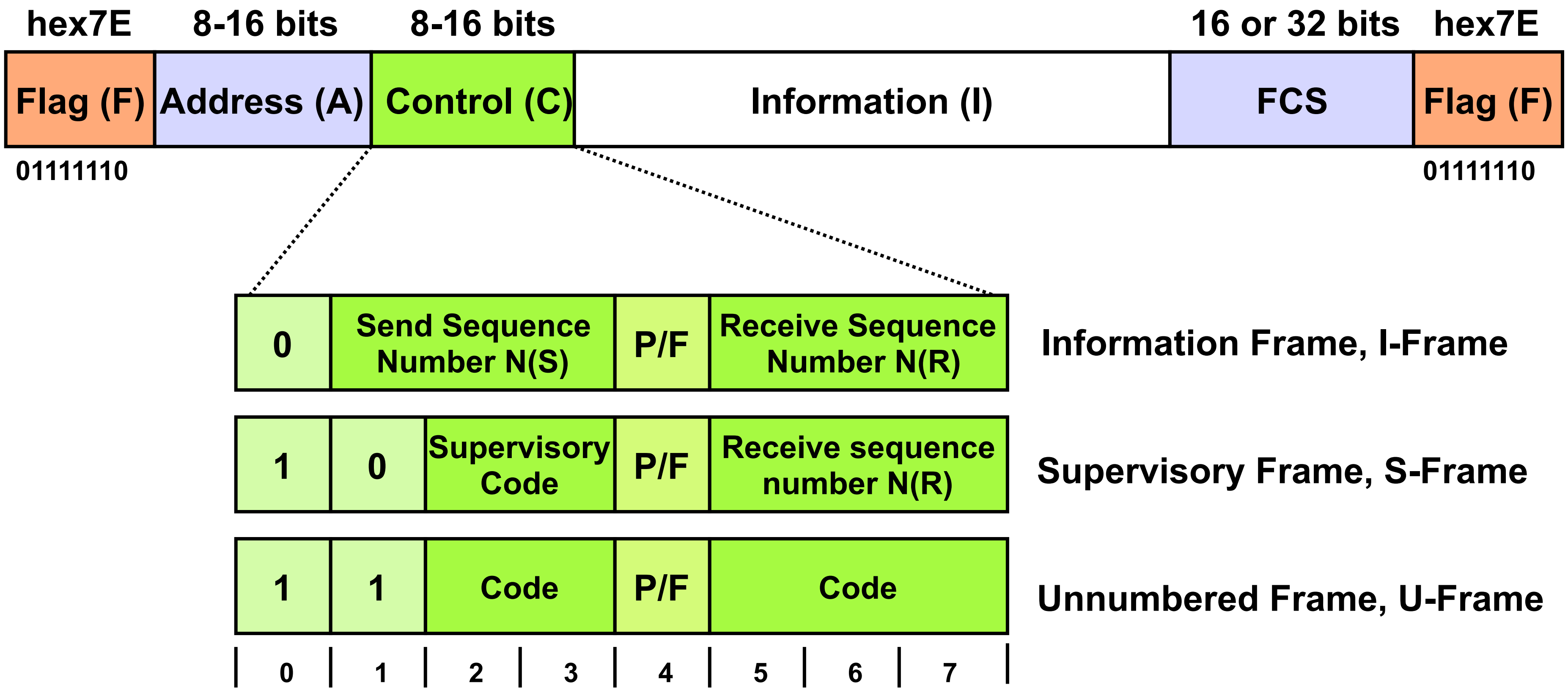
Same on Multipoint Lines (2)



HDLC Family


HDLC
LAPB
LAPM
LLC
SDLC
PPP
LAPF
V.120
QLLC
LAPD
ITU-T
IEEE
IBM
IETF

HDLC Frame Format



P / F = Poll / Final Bit

Supervisory Frames

1	0	Supervisory Code	P/F	Receive Sequence Number N(R)

Supervisory Code		Meaning
0	0	RR (Receiver Ready) = ACK plus GO
0	1	REJ (Reject) = NACK
1	0	RNR (Receiver Not Ready) = ACK plus STOP
1	1	SREJ (Selective Reject) = SREJ

Unnumbered Frames

1	1	Code	P/F	Code

Code		Code			Primary Station Commands	Secondary Station Responses
0	0	0	0	0	UI	UI
0	0	0	0	1	SNRM	
0	0	0	1	0	DISC	RD
0	0	1	0	0	UP	
0	0	1	1	0		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	

ARQ Principles in HDLC

- **Default: GoBackN without dedicated NACK frame (!)**
  - Receive-Sequence Number indicates next frame expected
- **"Check-pointing"**
  - Sender triggers (N)ACK information with P/F bit
- **Optional: Reject (REJ)**
  - Dedicated NACK frame
  - Can be sent at any time (no check-pointing)
- **Optional: Selective Reject (SREJ)**
  - Requests retransmission of single frame
- **Flow control with RR and RNR**

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

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

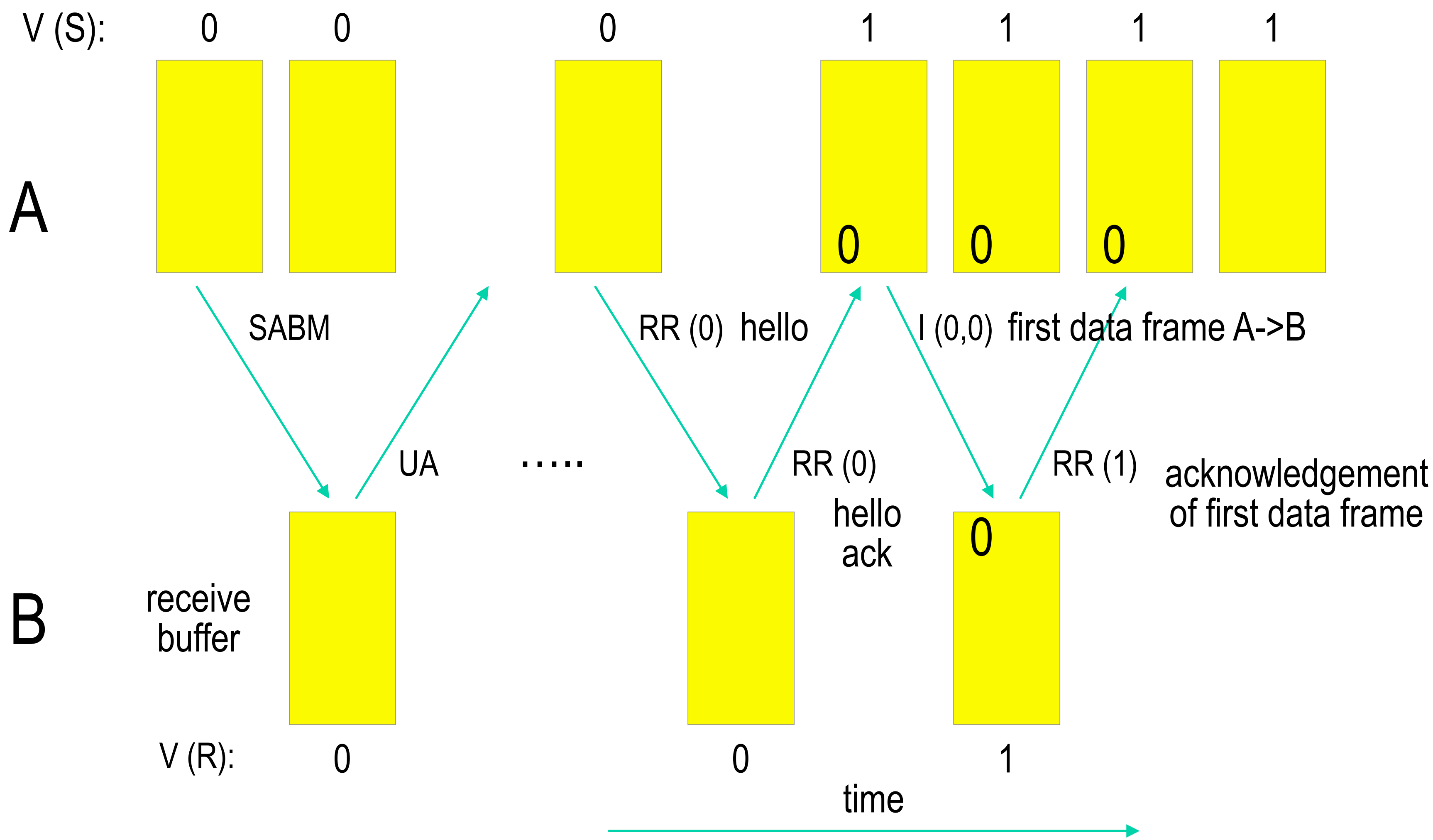
Connection-Less

UI	Unnumbered Information

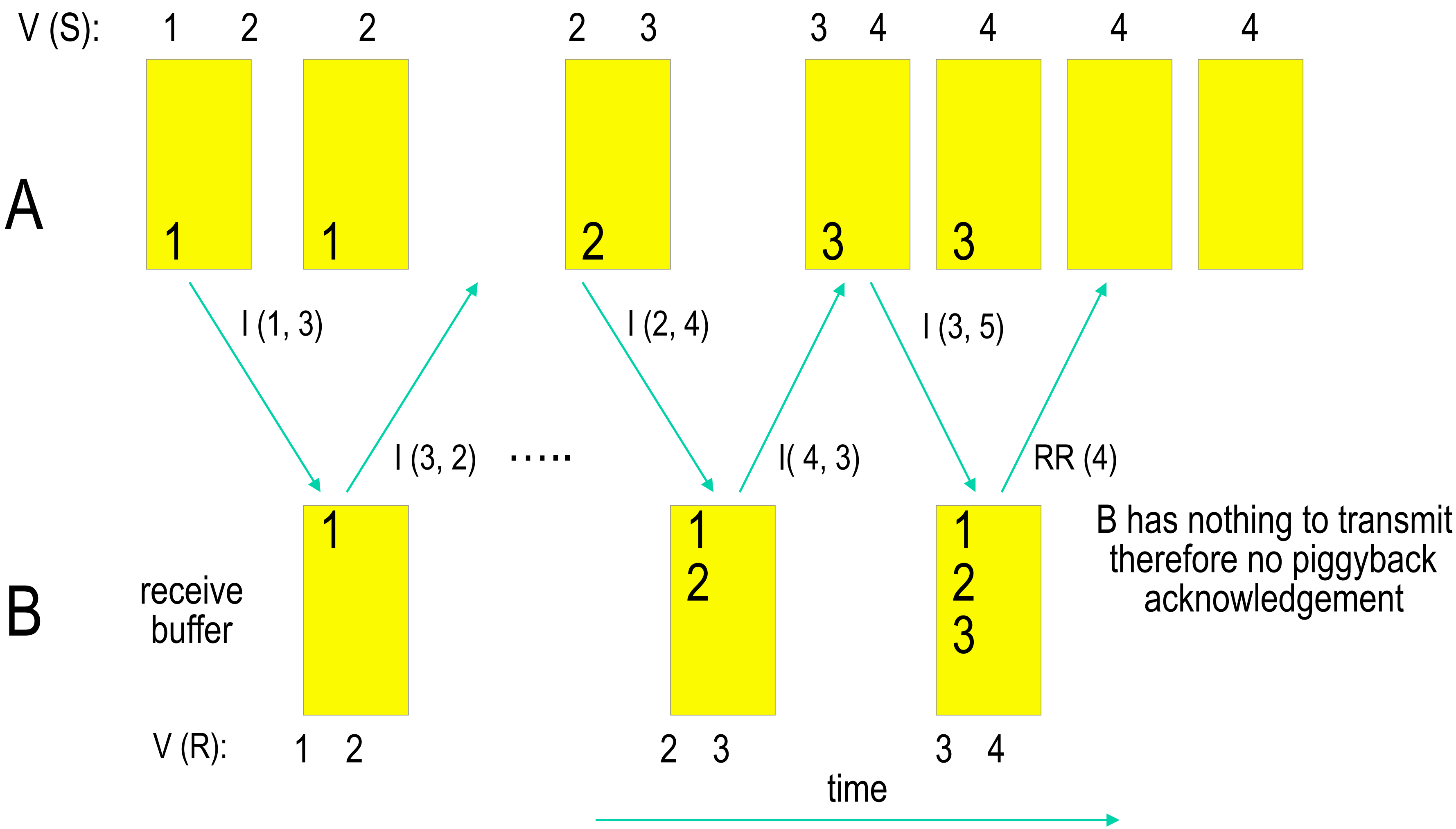
Miscellaneous

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

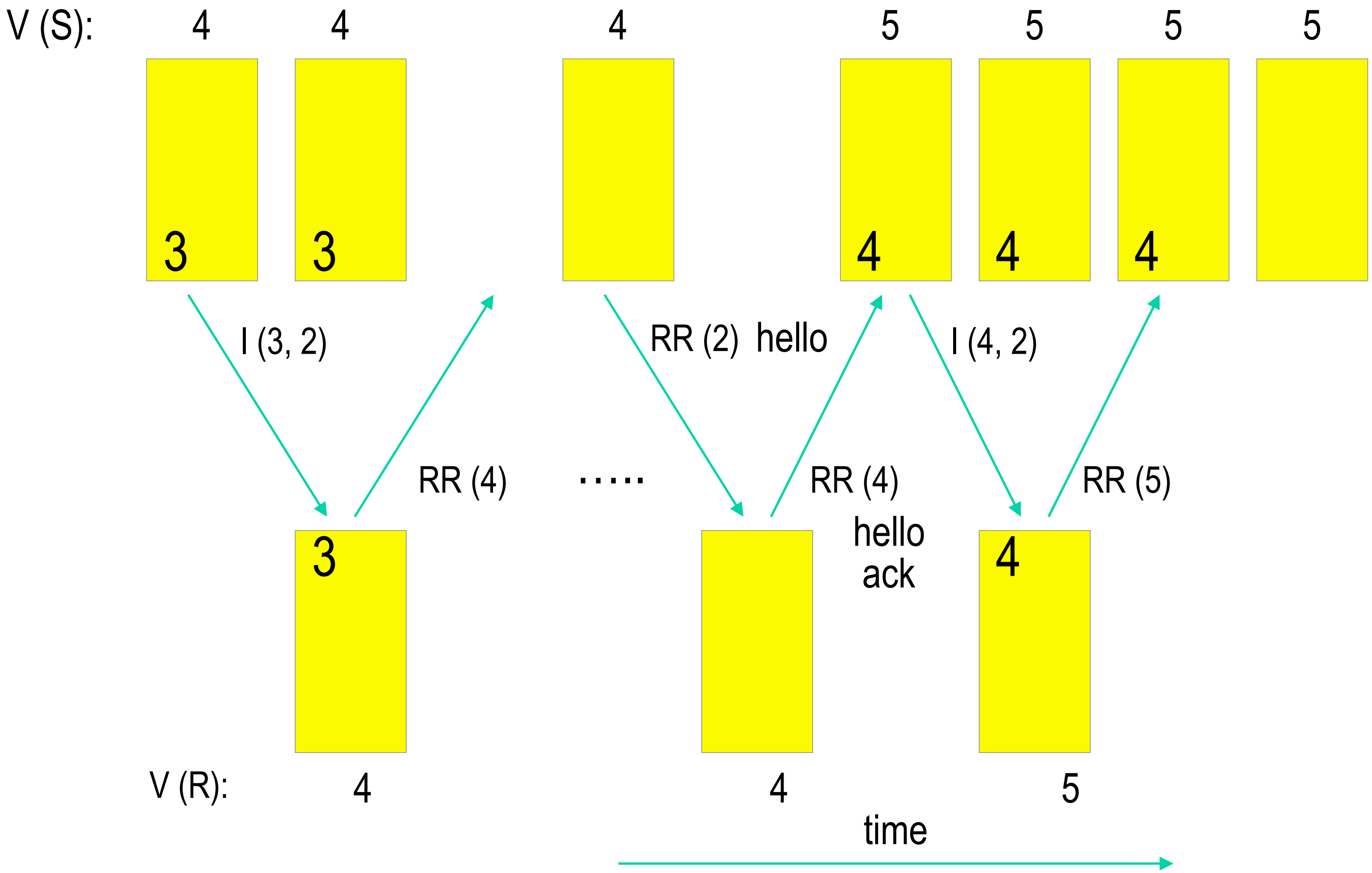
HDLC Example: Initializing / Keepalive / First Data Frame A -> B



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



HDLC Example: Keepalive



HDLC Example: Flow Control