Data Link Control Protocols

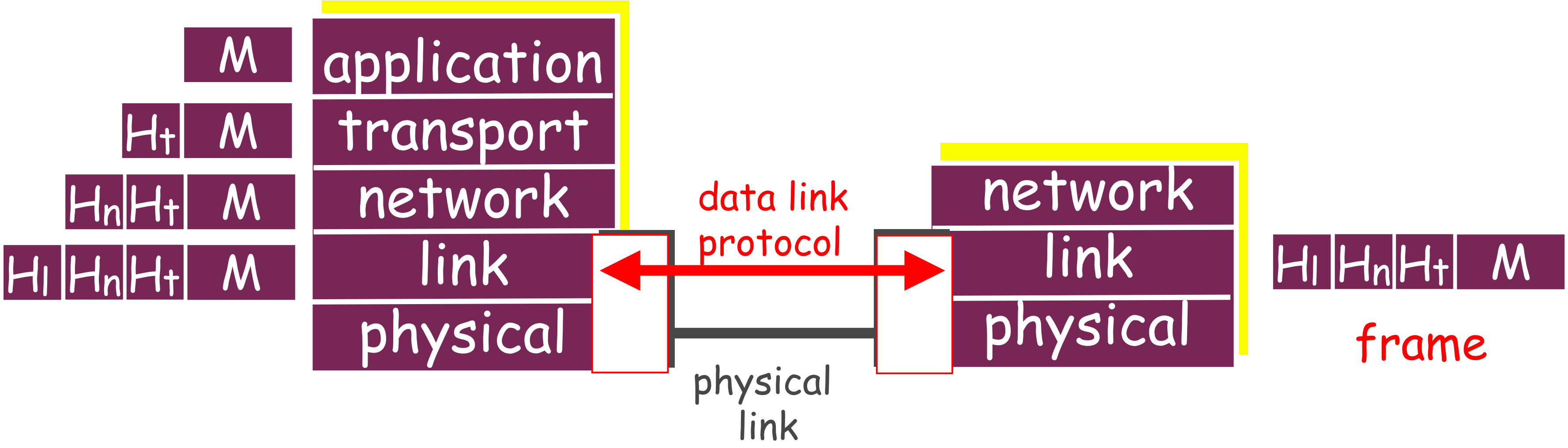
EE450: Introduction to Computer Networks

Professor A. Zahid

Data Link Layer

- Two physically connected devices:
  - Host-Router, Router-Router, Router-Host

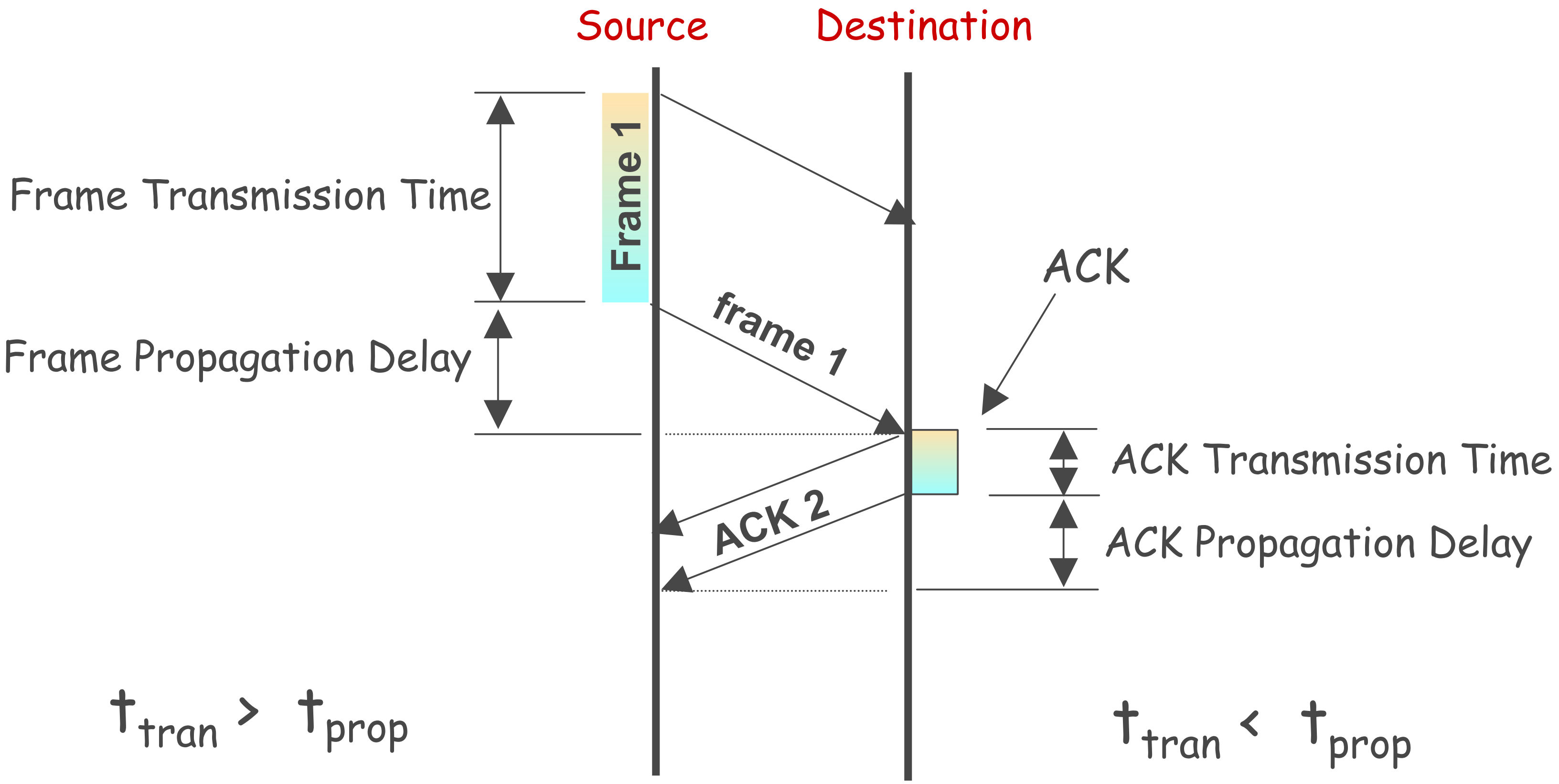
- Unit of data: frame



Data Link Layer Services

- Framing
  - Encapsulate packet into frame, adding header/trailer
  - Establish frame synchronization
- Error Detection & Control
  - Errors caused by signal attenuation, noise.
  - Receiver detects presence of errors:
    - Receiver drops frame
    - Receiver requests retransmission (ARQ)
    - Receiver corrects errors (discussed in EE568)
- Flow Control
  - Ensuring the sender does not overwhelm the receiver (i.e., preventing buffer overflow)

Frame Transmission Model



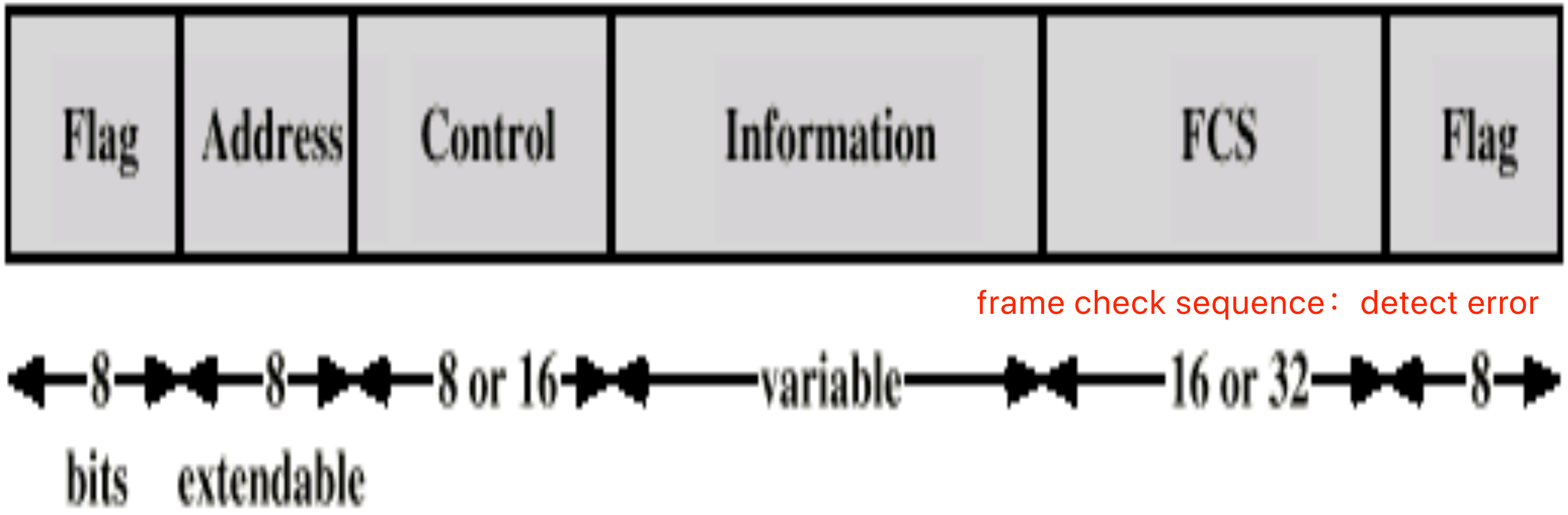
$t_{tran} > t_{prop}$

$t_{tran} < t_{prop}$

Typical Frame Structure

这里是point to point link 所以address只有一个而且一般是11111111

flag：标志frame两端位置



Flag	Address	Control	Information	FCS	Flag
8	8	8 or 16	variable	16 or 32	8
bits	extendable				

frame check sequence： detect error

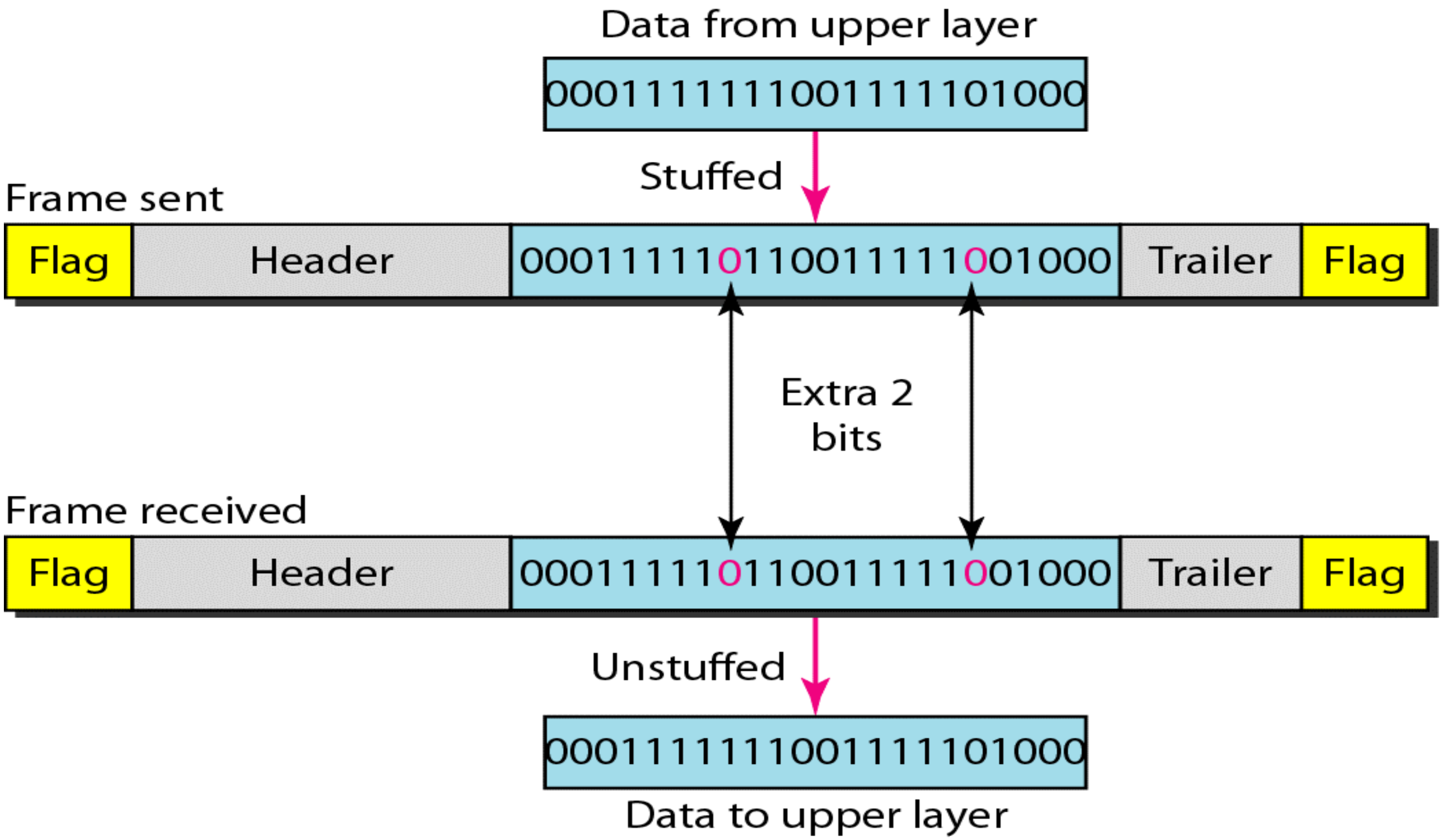
We shall see that the above structure does not work in a Multipoint link (like in LANs)

Frame Synchronization

- A special pattern, called a Flag (01111110) appears at the beginning and the end of the frame
- Receiver hunts for flag sequence to synchronize
- **Bit stuffing** used to avoid confusion with data containing 01111110
  - 0 inserted after every sequence of five 1s
  - If receiver detects five 1s it checks next bit
  - If 0, it is deleted
  - If 1 and seventh bit is 0, accept as flag
  - If sixth and seventh bits 1, sender is indicating abort

每隔5个1sender会在后边插一个0， 为了跟flag区别开
receiver会detect连续的5个1，要是后边是0就去掉，要是1的话就标志为flag

Bit Stuffing and un-Stuffing



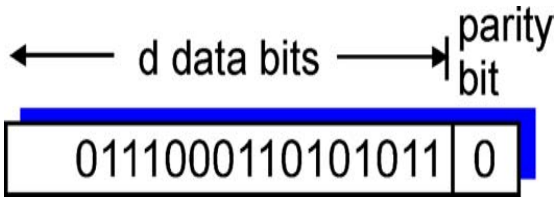
Error Detection

- Additional bits added by transmitter for error detection purposes
- Single Parity
  - Value of parity bit is such that character has even (even parity) or odd (odd parity) number of ones
  - Even number of bit errors goes undetected
- Two-Dimensional Parity

Parity Checking

Single Bit Parity:

Detect single bit errors

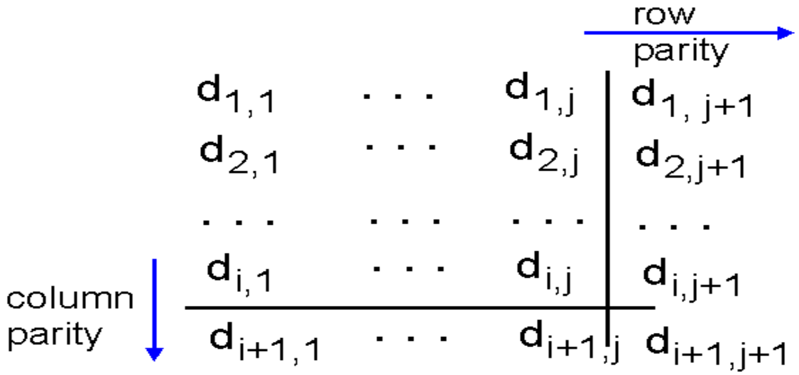


Odd parity scheme

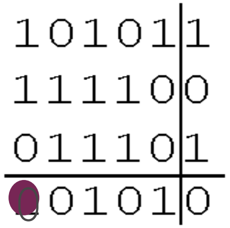
Parity bit value is chosen such that number of 1's send is odd.
Ex. 9 1's in the data, so the parity bit is '0'.

Two Dimensional Bit Parity:

Detect *and correct* single bit errors

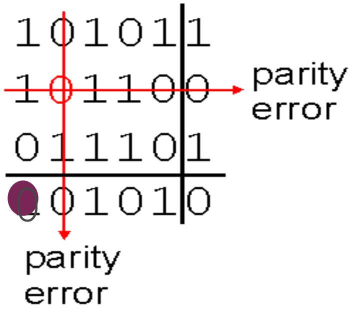


row parity
column parity
$d_{1,1}$ . . . $d_{1,j}$ $d_{1,\ j+1}$
$d_{2,1}$ . . . $d_{2,j}$ $d_{2,j+1}$
. . . .
$d_{i,1}$ . . . $d_{i,j}$ $d_{i,j+1}$
$d_{i+1,1}$ . . . $d_{i+1,j}$ $d_{i+1,j+1}$



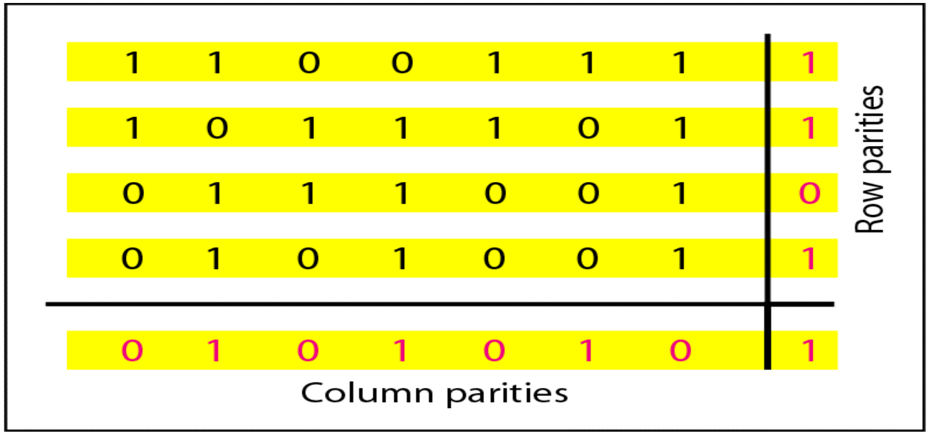
no errors

(even parity)



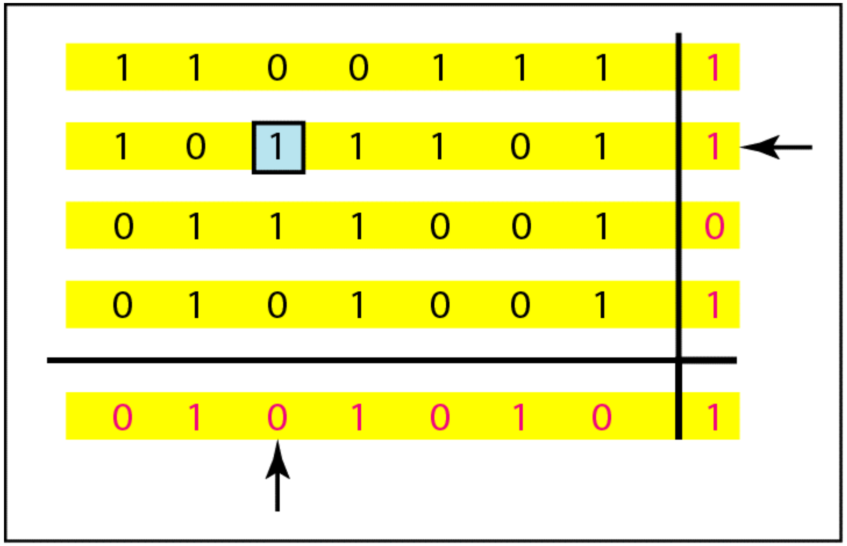
correctable single bit error

Example: Two-Dimensional Parity



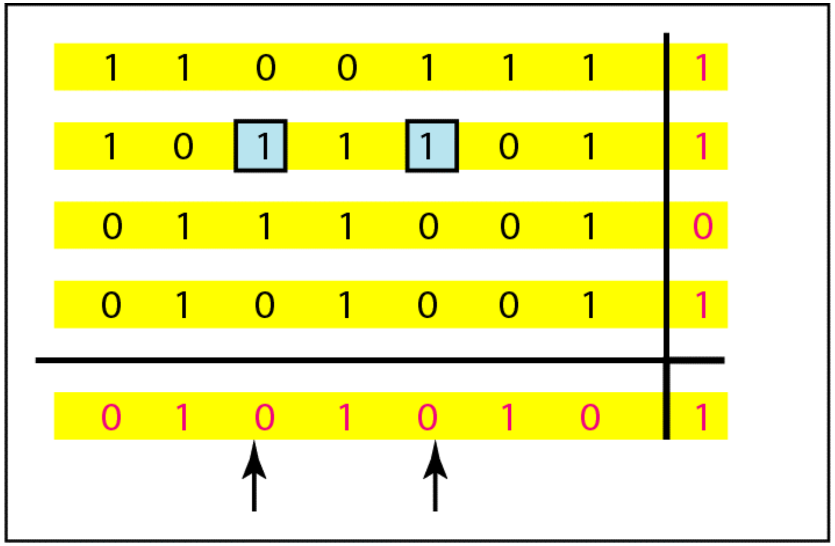
1	1	0	0	1	1	1	1
1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

a. Design of row and column parities



1	1	0	0	1	1	1	1
1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

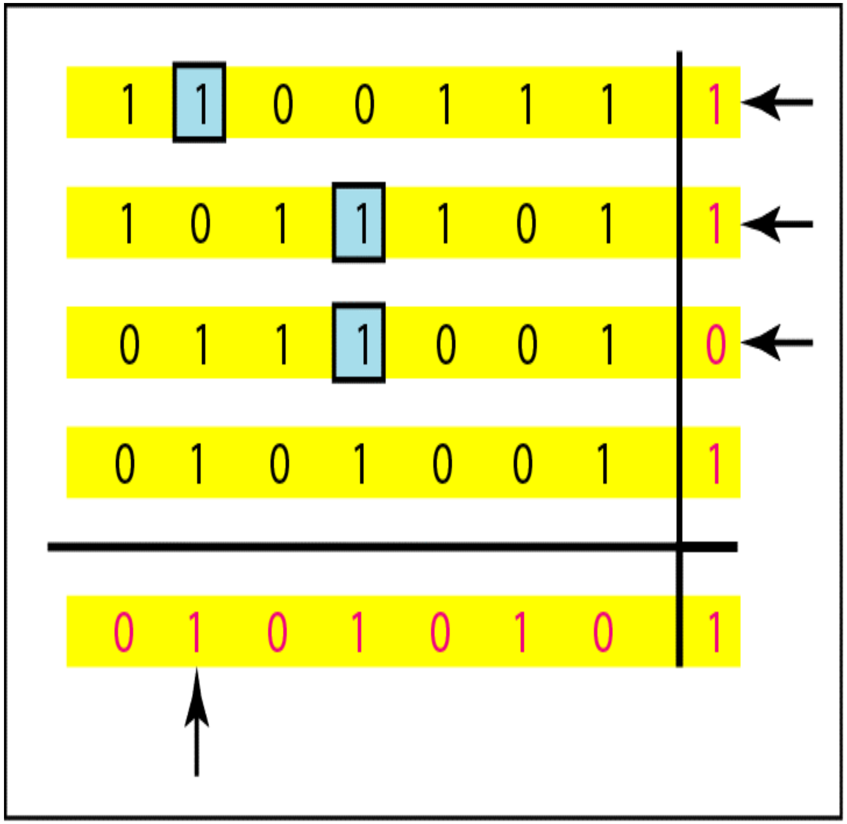
b. One error affects two parities



1	1	0	0	1	1	1	1
1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

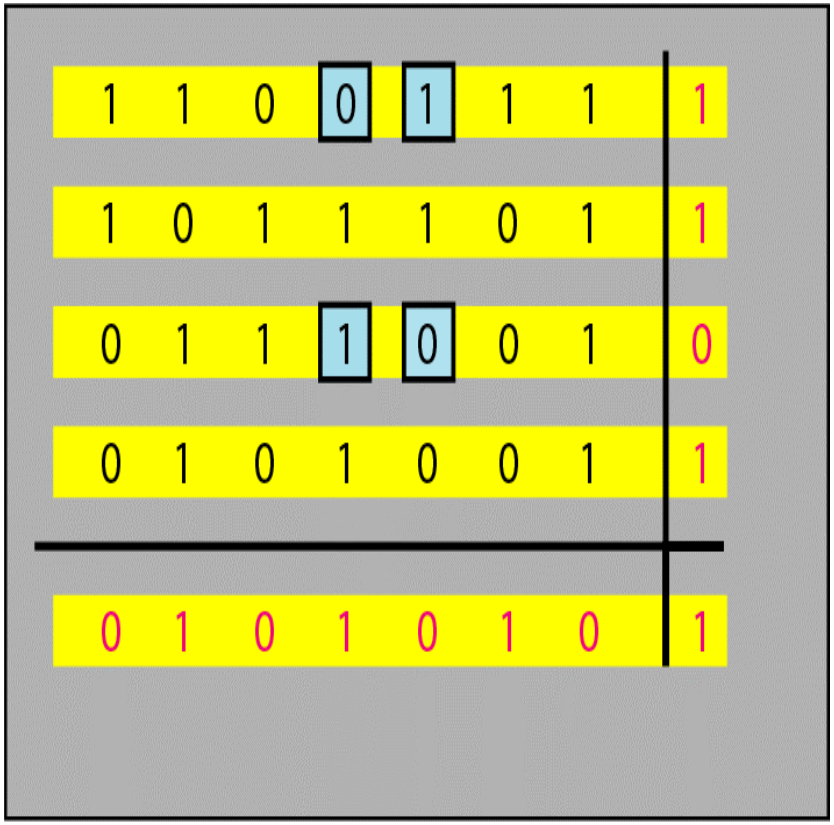
c. Two errors affect two parities

Example (Continued)



1	1	0	0	1	1	1	1
1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

d. Three errors affect four parities



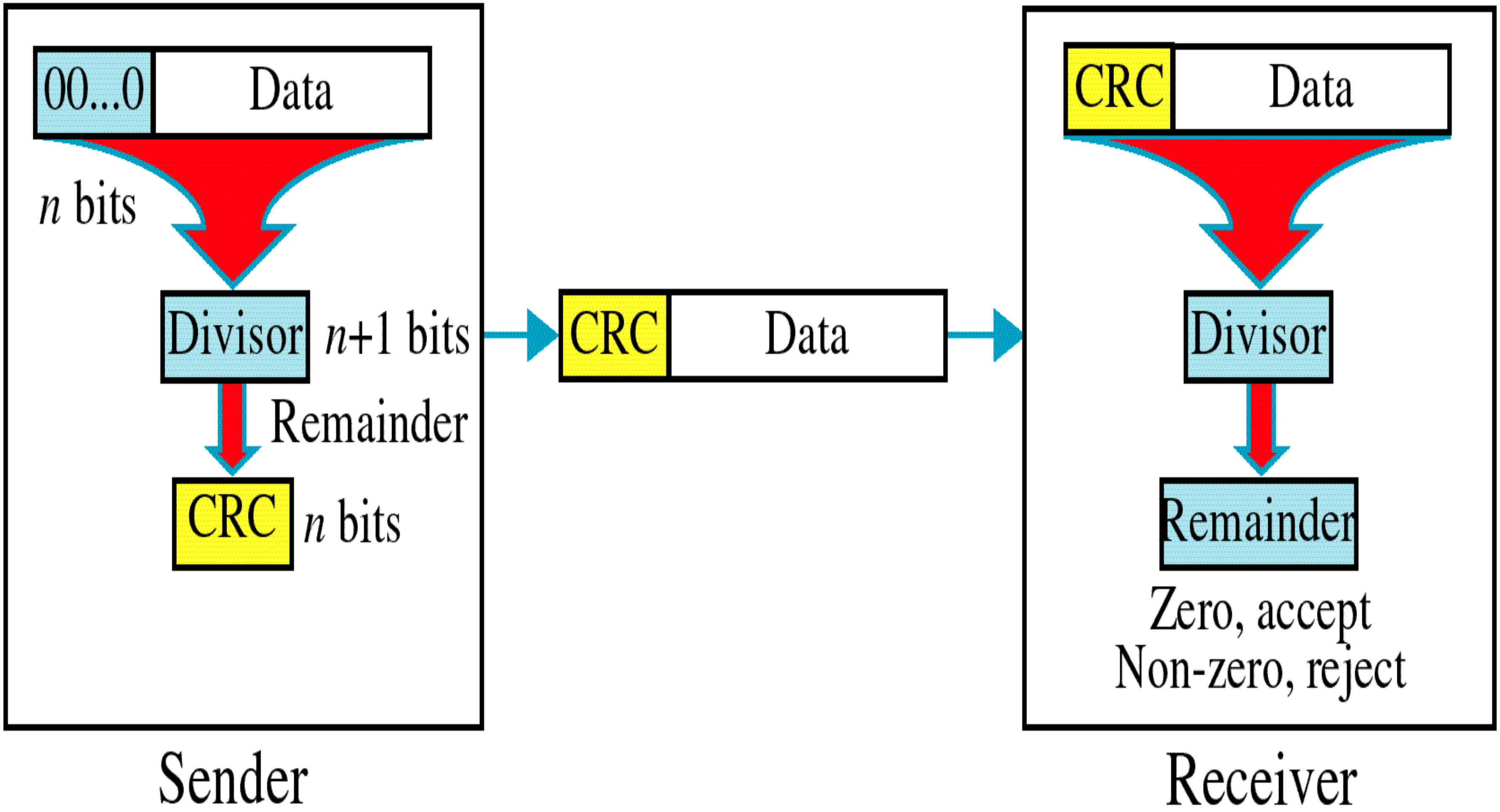
1	1	0	0	1	1	1	1
1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

e. Four errors cannot be detected

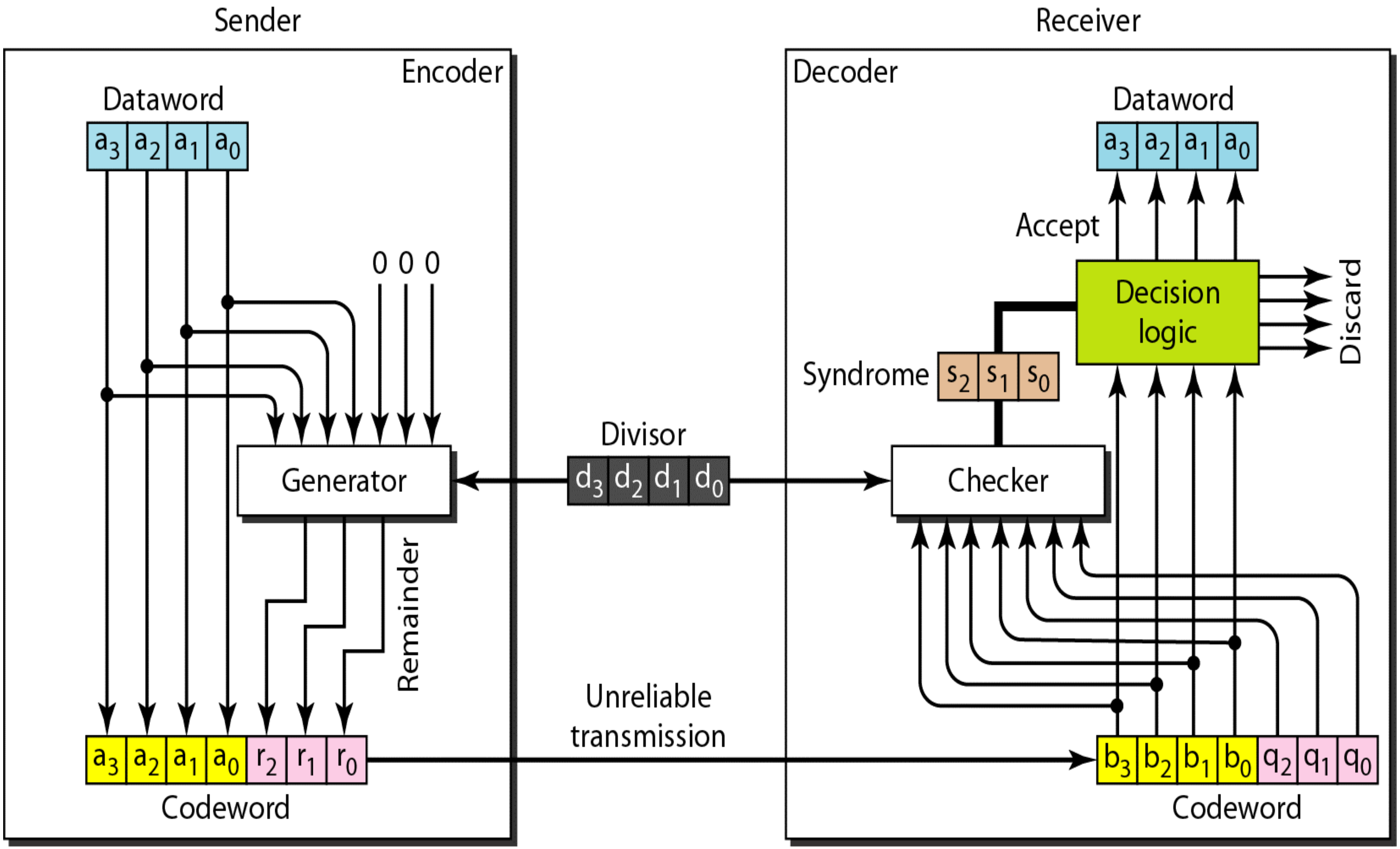
Frame Check Sequence (FCS)

- For every block of k bits, transmitter generates n bit sequence
- Transmit k+n bits which is exactly divisible by some number
- Receive divides frame by that number
  - If no remainder, assume no error
  - If reminder, an error is detected

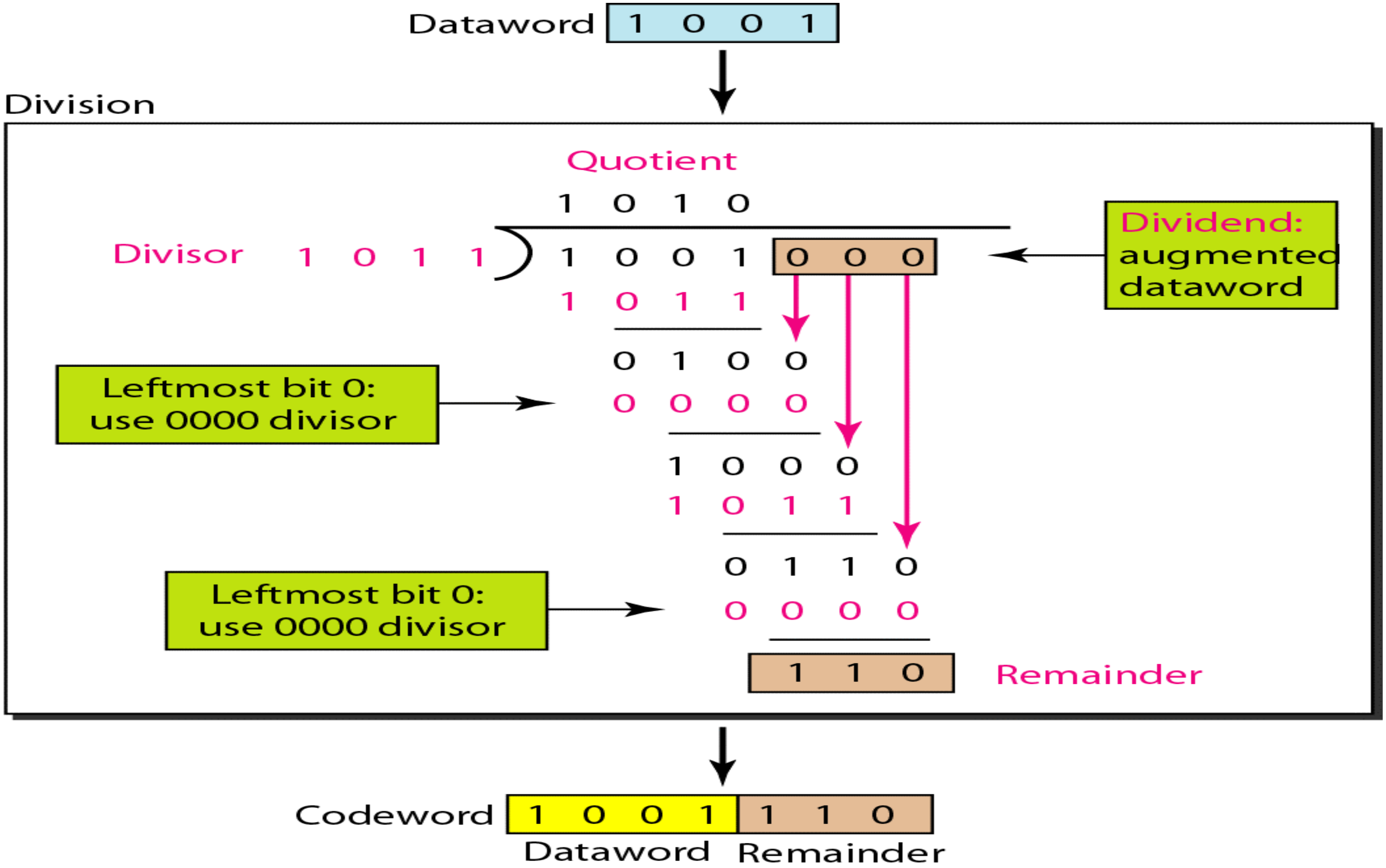
FCS (CRC) Structure



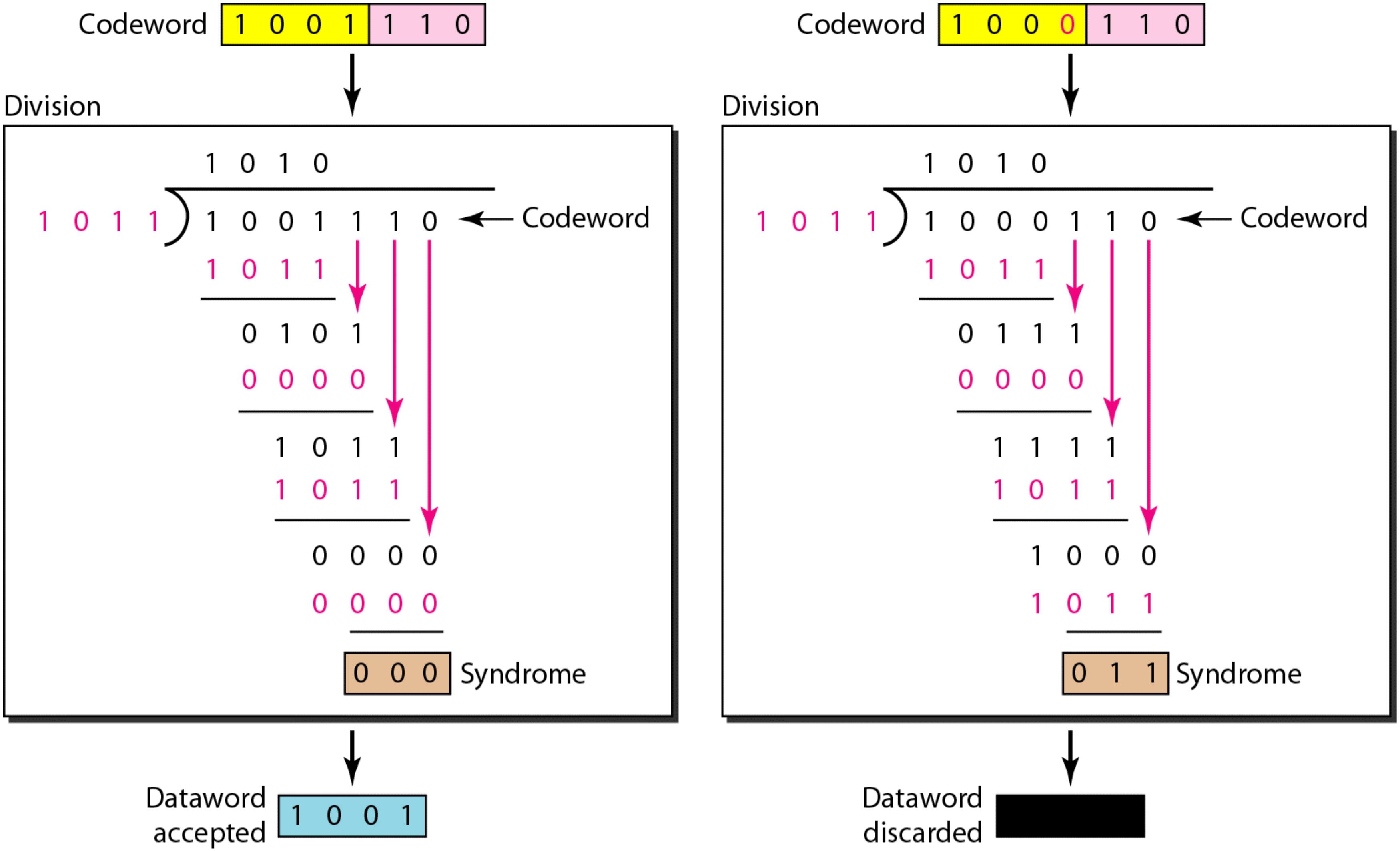
FCS Structure (Cont.)



Example of FCS (CRC)



FCS (error-free and w/errors)



Codeword 1 0 0 1 | 1 1 0

Division

```
        1 0 1 0
       ________________
1 0 1 1 ) 1 0 0 1 1 1 0   ← Codeword
          1 0 1 1
          -------
            0 1 0 1
            0 0 0 0
            -------
              1 0 1 1
              1 0 1 1
              -------
                0 0 0 0
                0 0 0 0
                -------
                  0 0 0   Syndrome
```

Dataword accepted 1 0 0 1

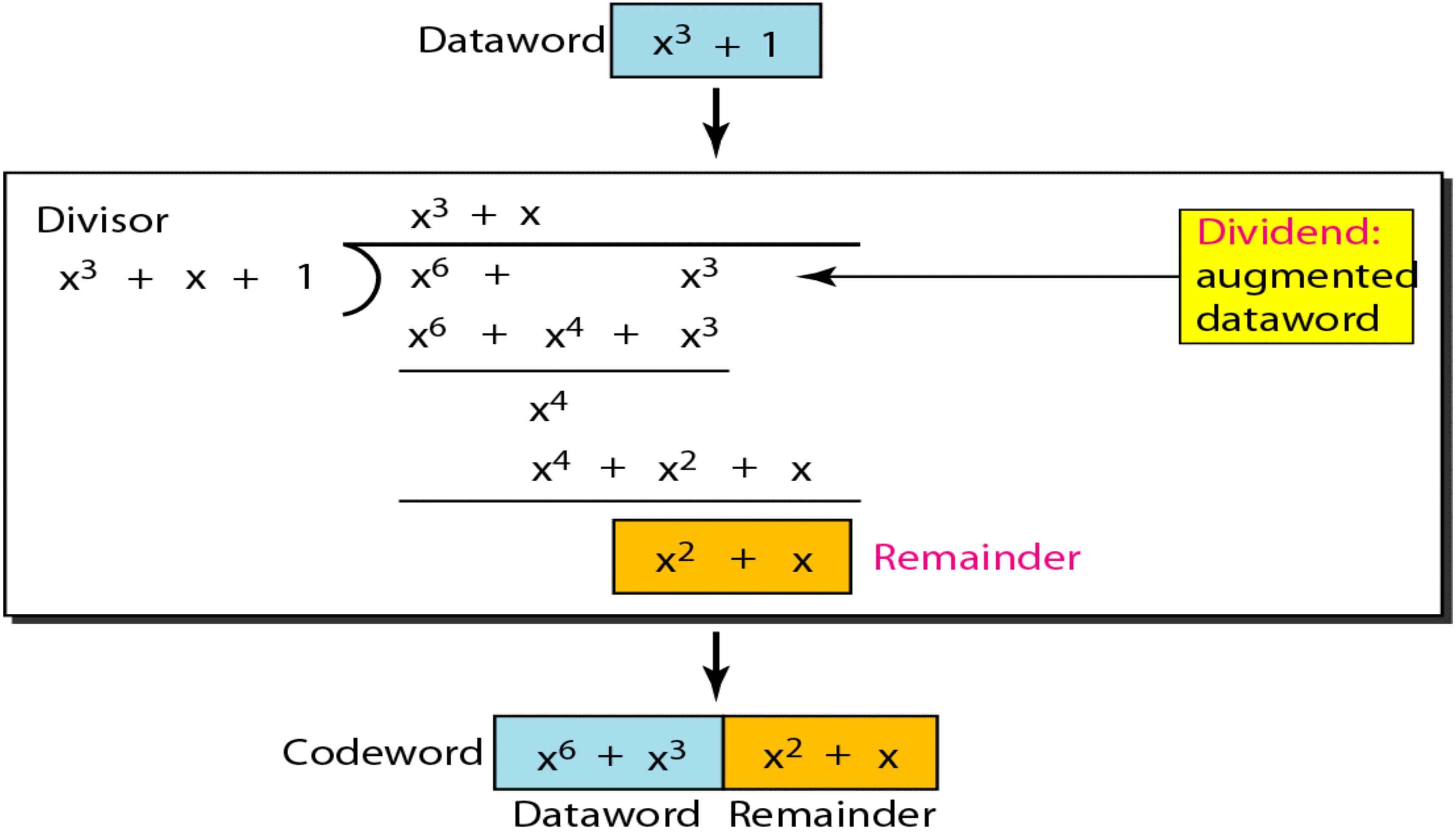
Codeword 1 0 0 0 | 1 1 0

Division

```
        1 0 1 0
       ________________
1 0 1 1 ) 1 0 0 0 1 1 0   ← Codeword
          1 0 1 1
          -------
            0 1 1 1
            0 0 0 0
            -------
              1 1 1 1
              1 0 1 1
              -------
                1 0 0 0
                1 0 1 1
                -------
                  0 1 1   Syndrome
```

Dataword discarded

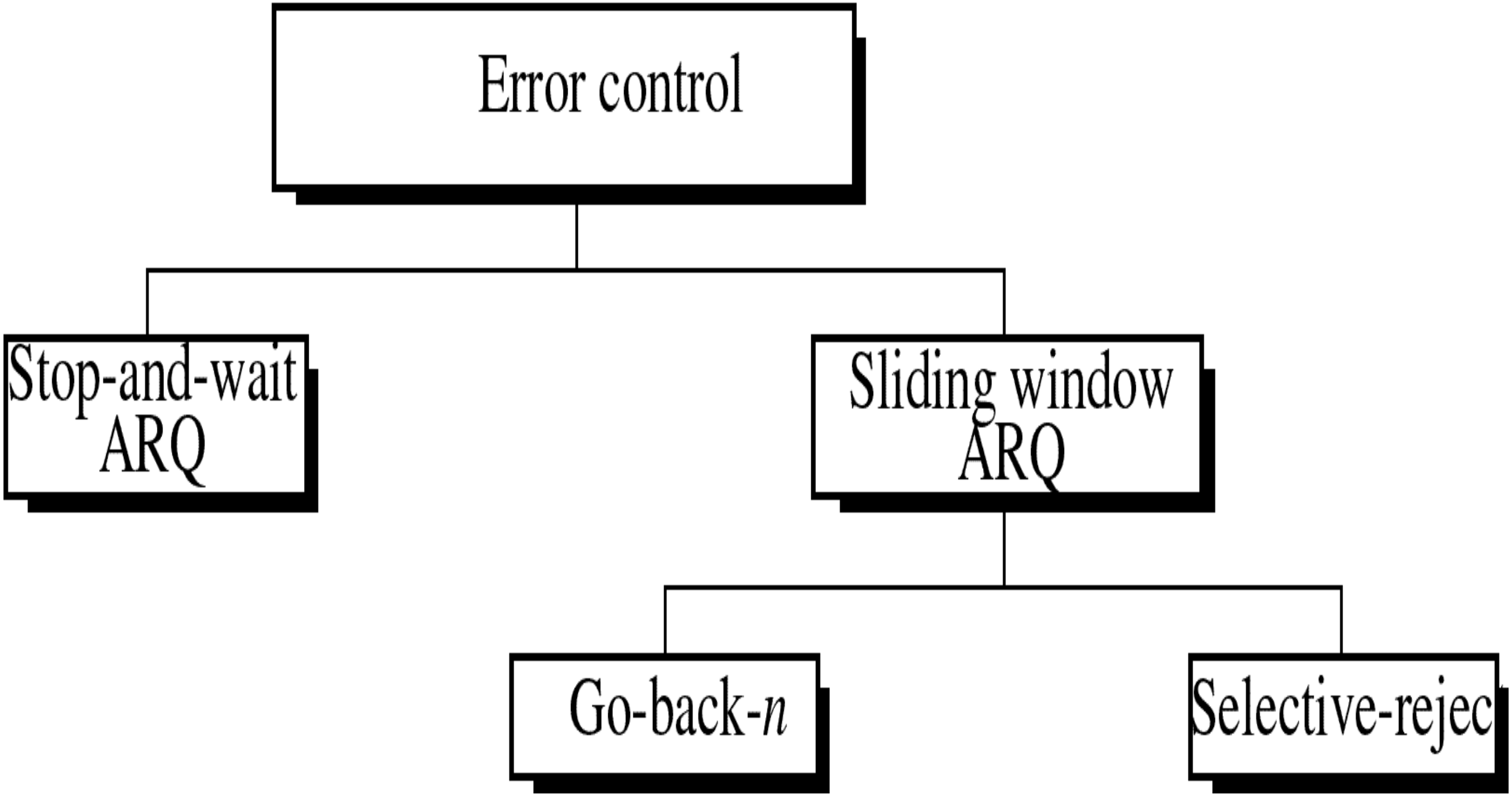
FCS using Polynomials



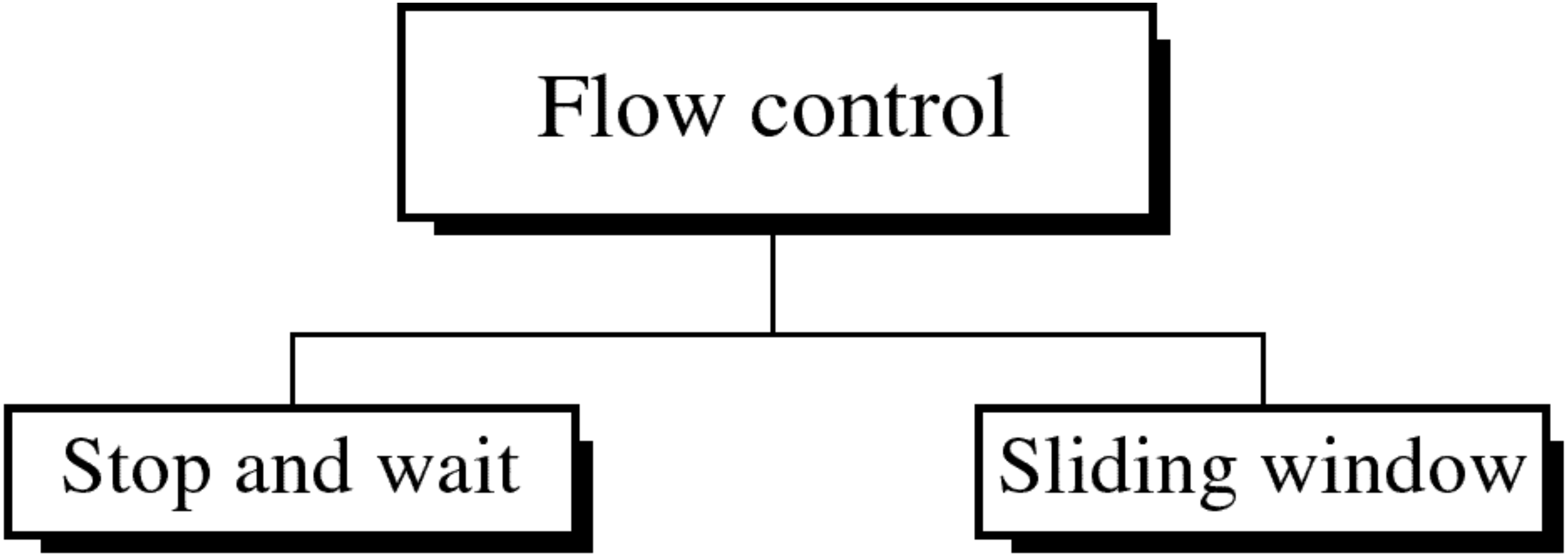
Receiver Rules

- If the reminder is not zero, then one or more bits are corrupted and the frame is rejected
- If the remainder is 0, then
  - No bits are corrupted or
  - Some bits are corrupted but the FCS decoder failed to detect them

Error Control Procedures



Flow Control Procedures



Flow control

Stop and wait

Sliding window

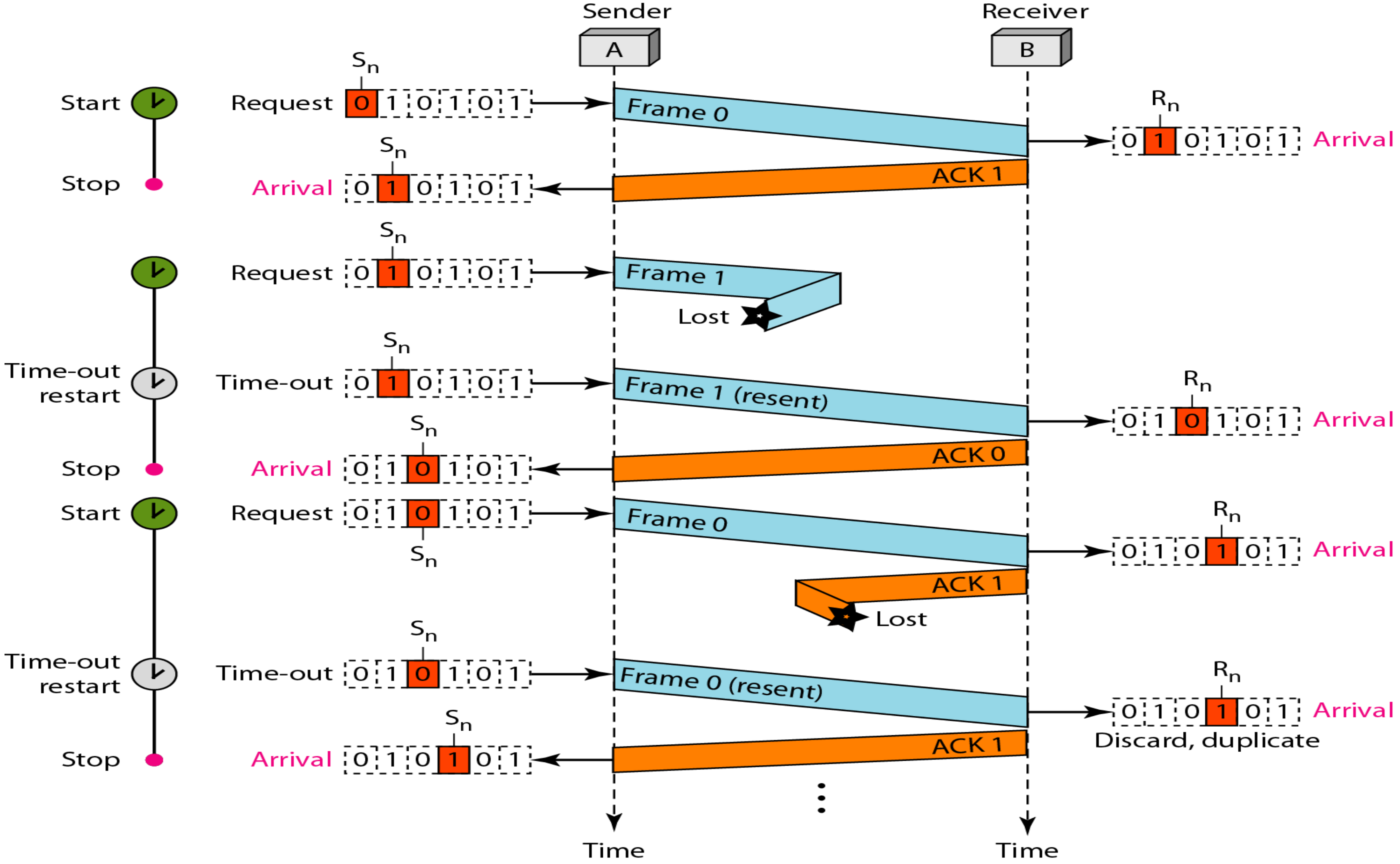
Send one frame at a time

Send several frames at a time

Stop and Wait ARQ

- Source transmits a single frame at a time
- Wait for ACK
- If received frame damaged, discard it
  - Transmitter has timeout timer
  - If no ACK within t_{out} = timeout, retransmit frame
  - Transmitter buffers copy of frame until ACK is received
- If ACK damaged, transmitter will not recognize it
  - Transmitter will retransmit
  - Receiver gets two copies of frame and discards one.
  - Use ACK_0 (recv'd frame 1) and ACK_1 (recv'd frame 0)

Stop & Wait ARQ



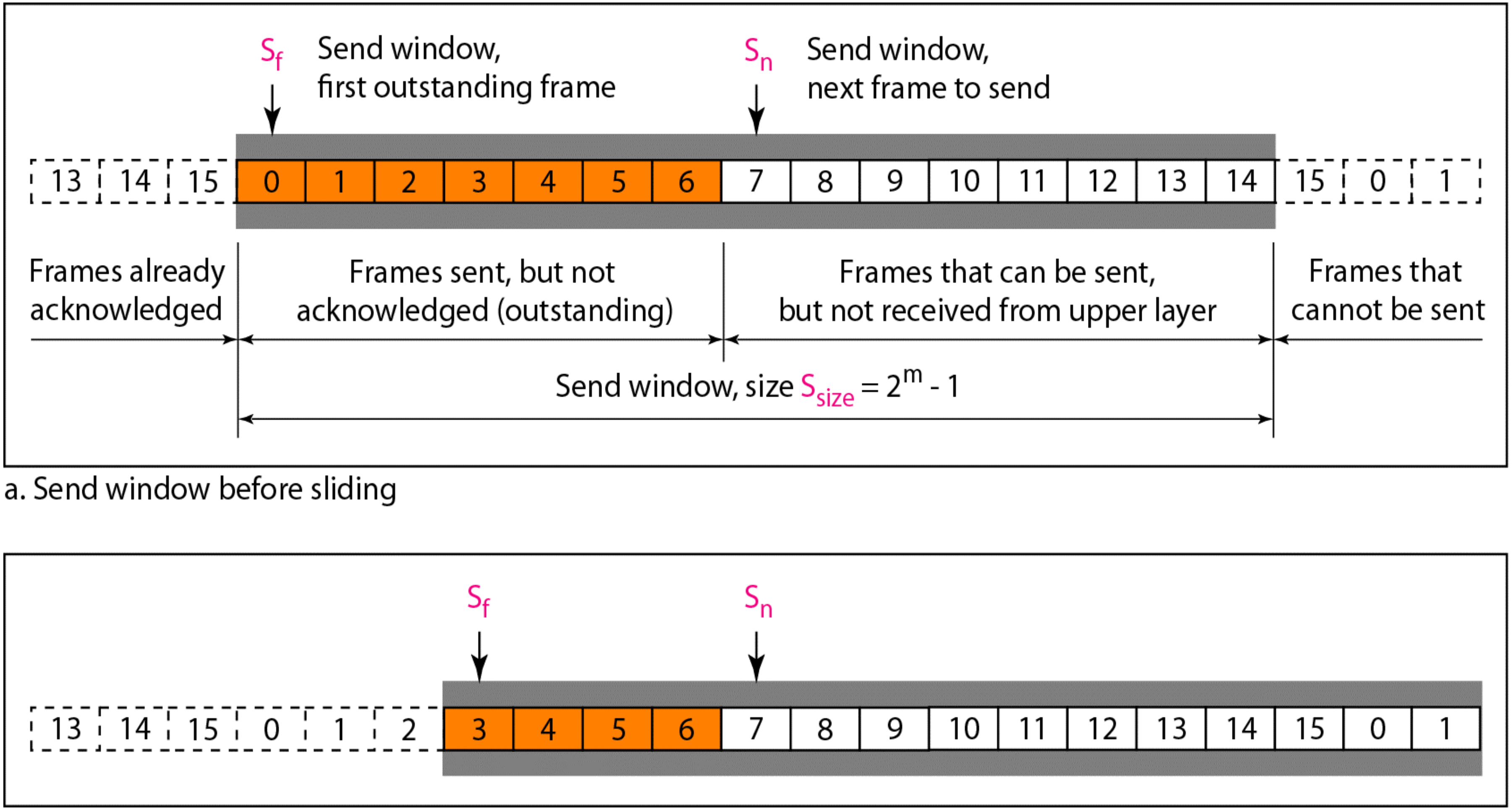
Link Utilization is Stop & Wait ARQ

- Link Bandwidth: 1 Mbps
- RTT: 20 msec
- Frame Length: 1000 bits
- BW x Delay Product = 20000 bits = 20 frames
- Sender can ONLY send 1 frame during RTT
- Hence Link Utilization is 5%
- Really Bad!

Go-Back-N ARQ

- Based on sliding window Protocol
- If no error, ACK as usual with next frame expected
- Use window to control number of outstanding frames
- If error, reply with rejection
  - Discard that frame and all future frames until error frame received correctly
  - Transmitter must go back and retransmit that frame and all subsequent frames

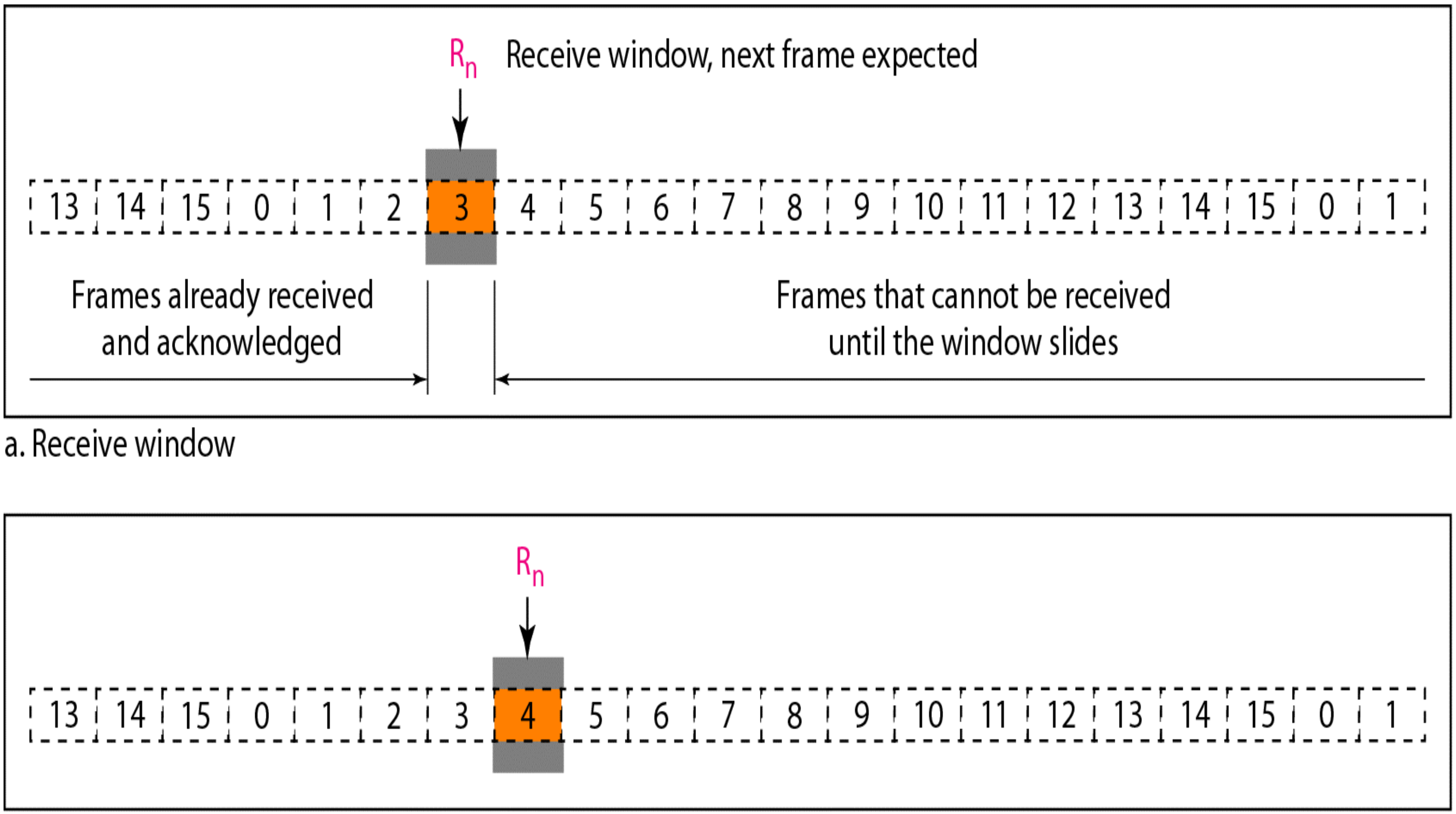
Sending Window in Go-Back-N ARQ


S_f
Send window, first outstanding frame
S_n
Send window, next frame to send
13
14
15
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
0
1
Frames already acknowledged
Frames sent, but not acknowledged (outstanding)
Frames that can be sent, but not received from upper layer
Frames that cannot be sent
Send window, size $S_{size} = 2^m - 1$
S_f
S_n
13
14
15
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
0
1

a. Send window before sliding

b. Send window after sliding

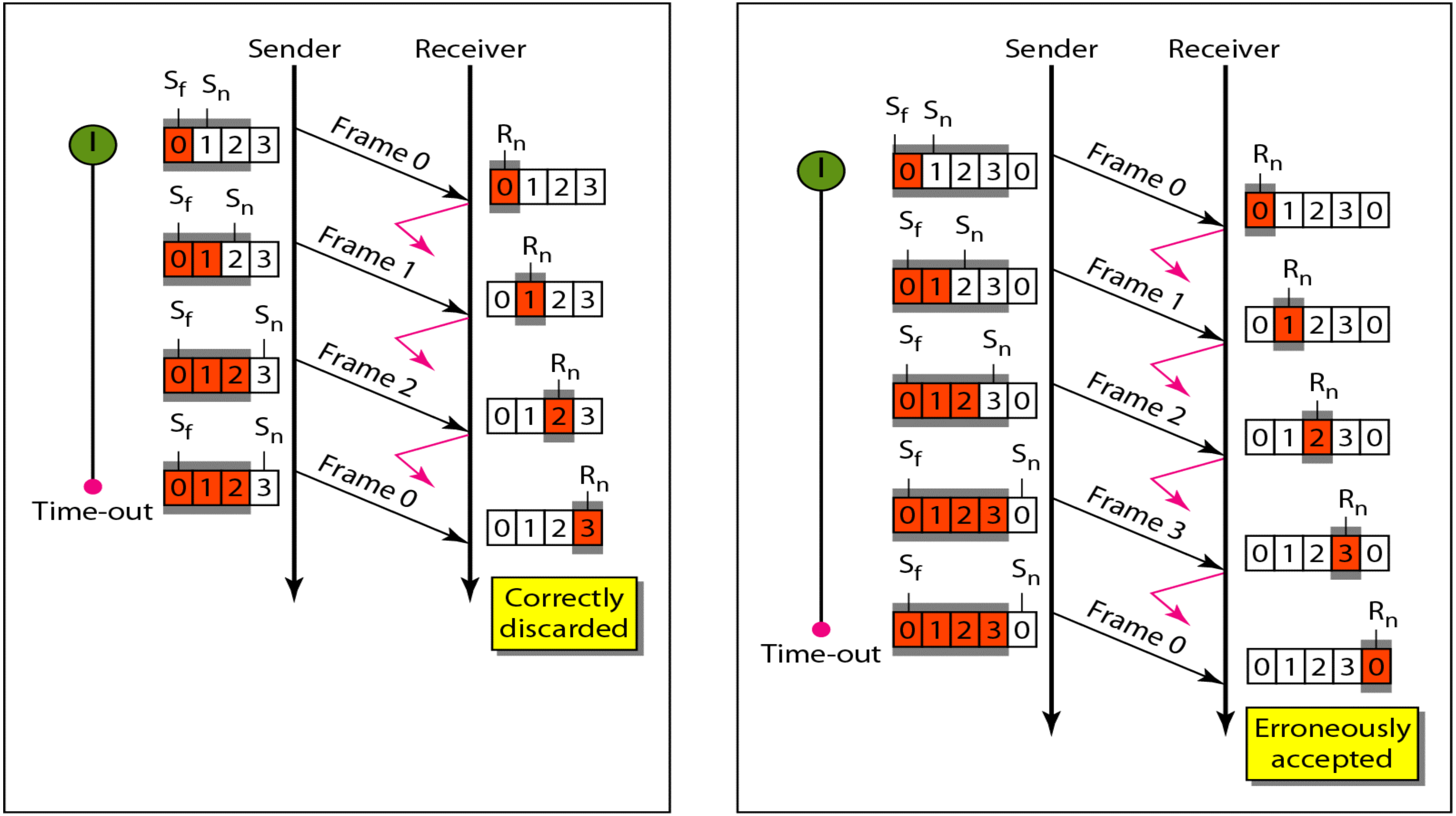
Receiver Window in Go-Back-N ARQ


R_n
Receive window, next frame expected
13 14 15 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 0 1
Frames already received and acknowledged
Frames that cannot be received until the window slides
R_n
13 14 15 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 0 1

a. Receive window

b. Window after sliding

Window Size in Go-Back-N ARQ



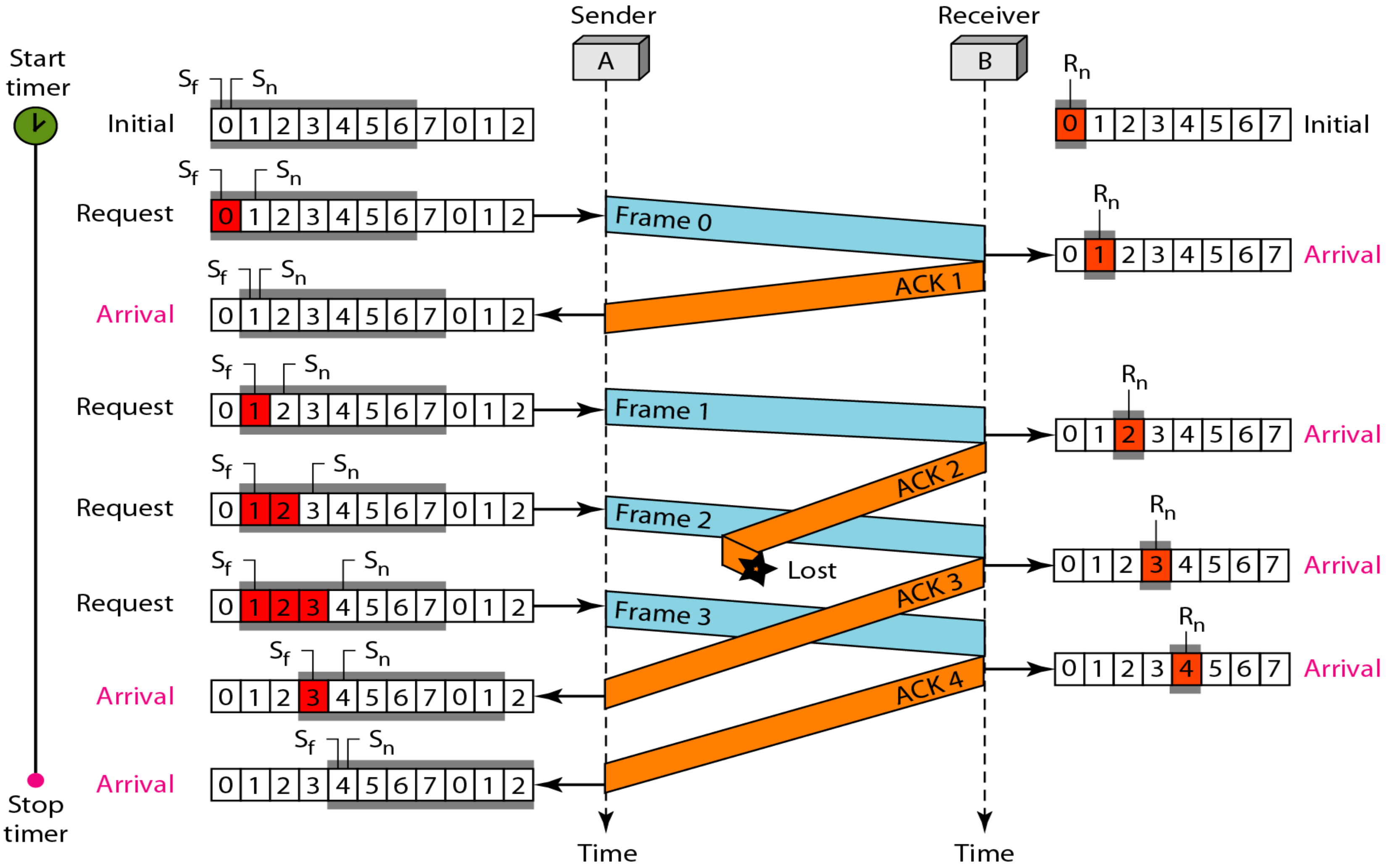
a. Window size < 2^m

b. Window size = 2^m

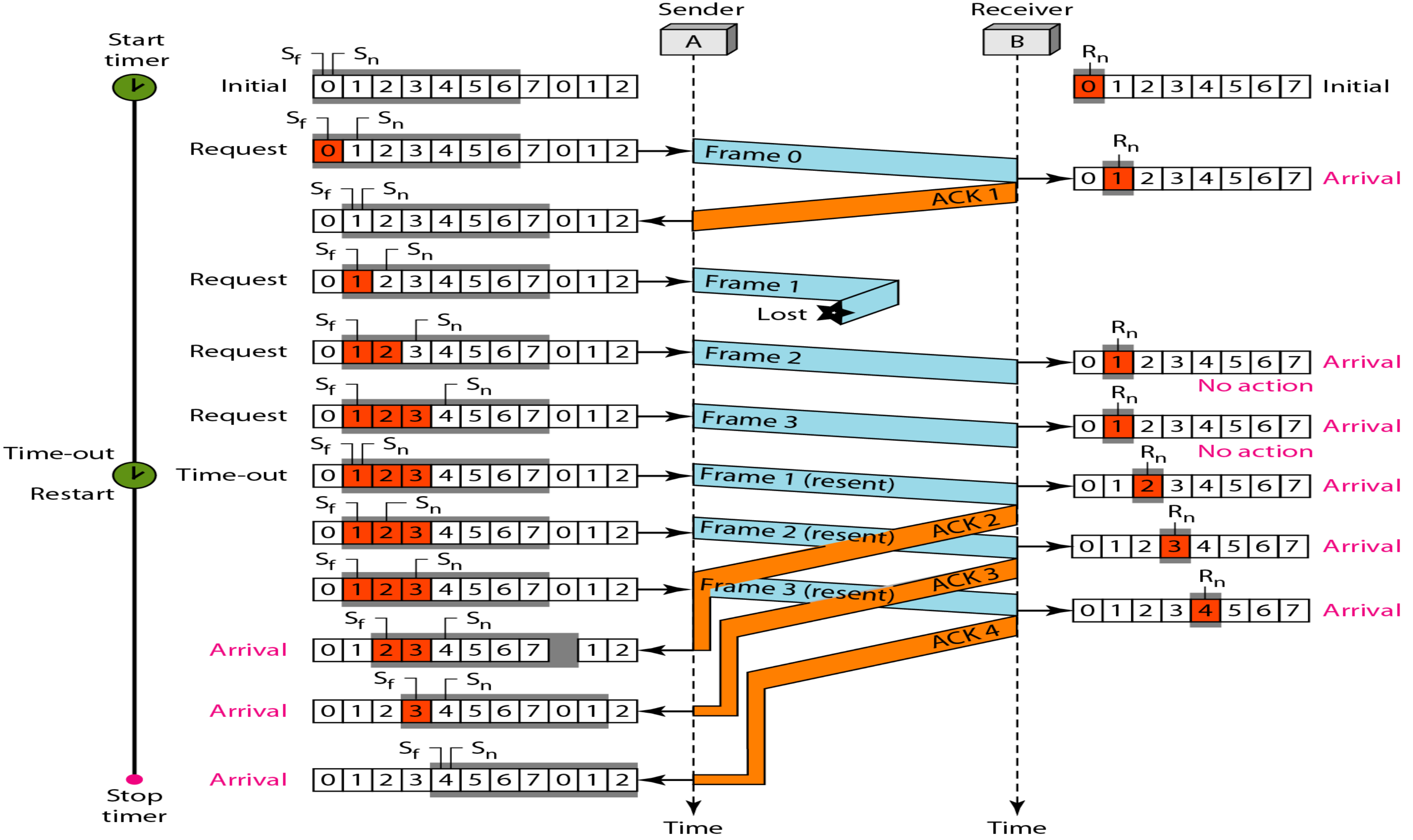
Summary Notes for Go-Back-N ARQ

- In the Go-Back-N Protocol, the sequence numbers are modulo 2^m, where m is the size of the sequence number field in bits.
- The send window can slide one or more slots when a valid acknowledgment arrives.
- In Go-Back-N ARQ, the size of the send window must be less than 2^m; the size of the receiver window is always 1.
- The receive window of size 1. The window slides when a frame with no detected errors arrive; i.e. sliding occurs one slot at a time. Receiver will drop any out-of-order frames

Example: Reliable Forward Channel



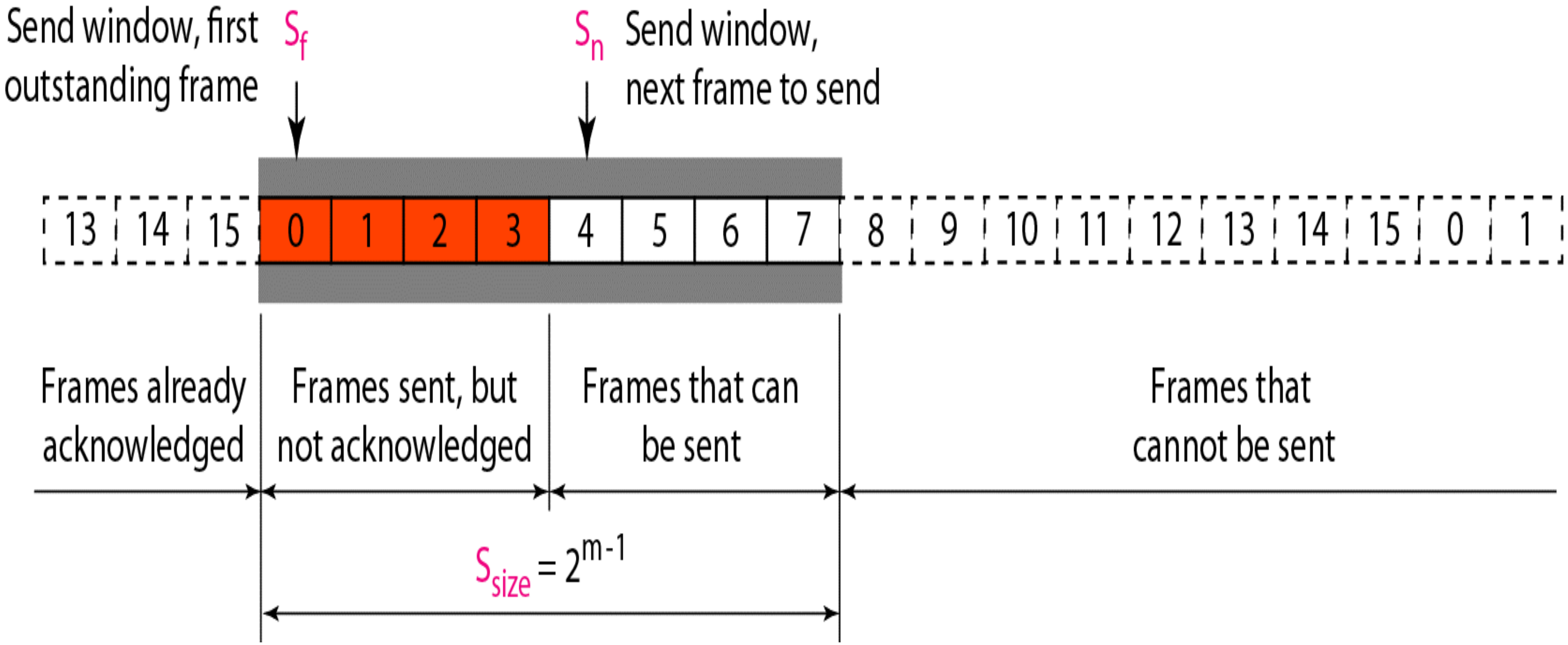
Example: Un-reliable Forward Channel



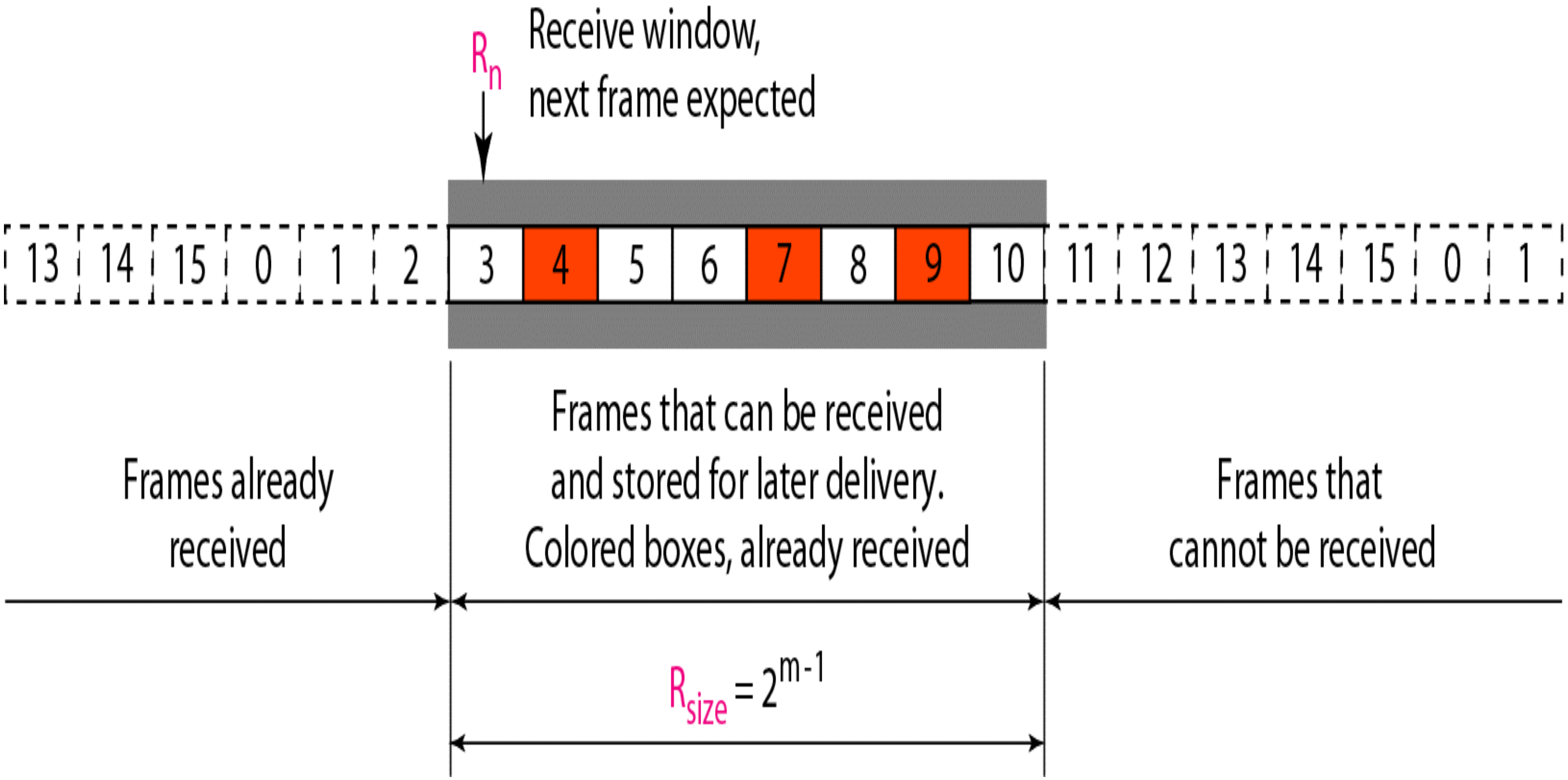
Selective Repeat (Reject) ARQ

- Only rejected frames are retransmitted
- Subsequent frames are accepted by the receiver and buffered
- Minimizes retransmission
- Receiver must maintain large enough buffer
- More complex transmitter

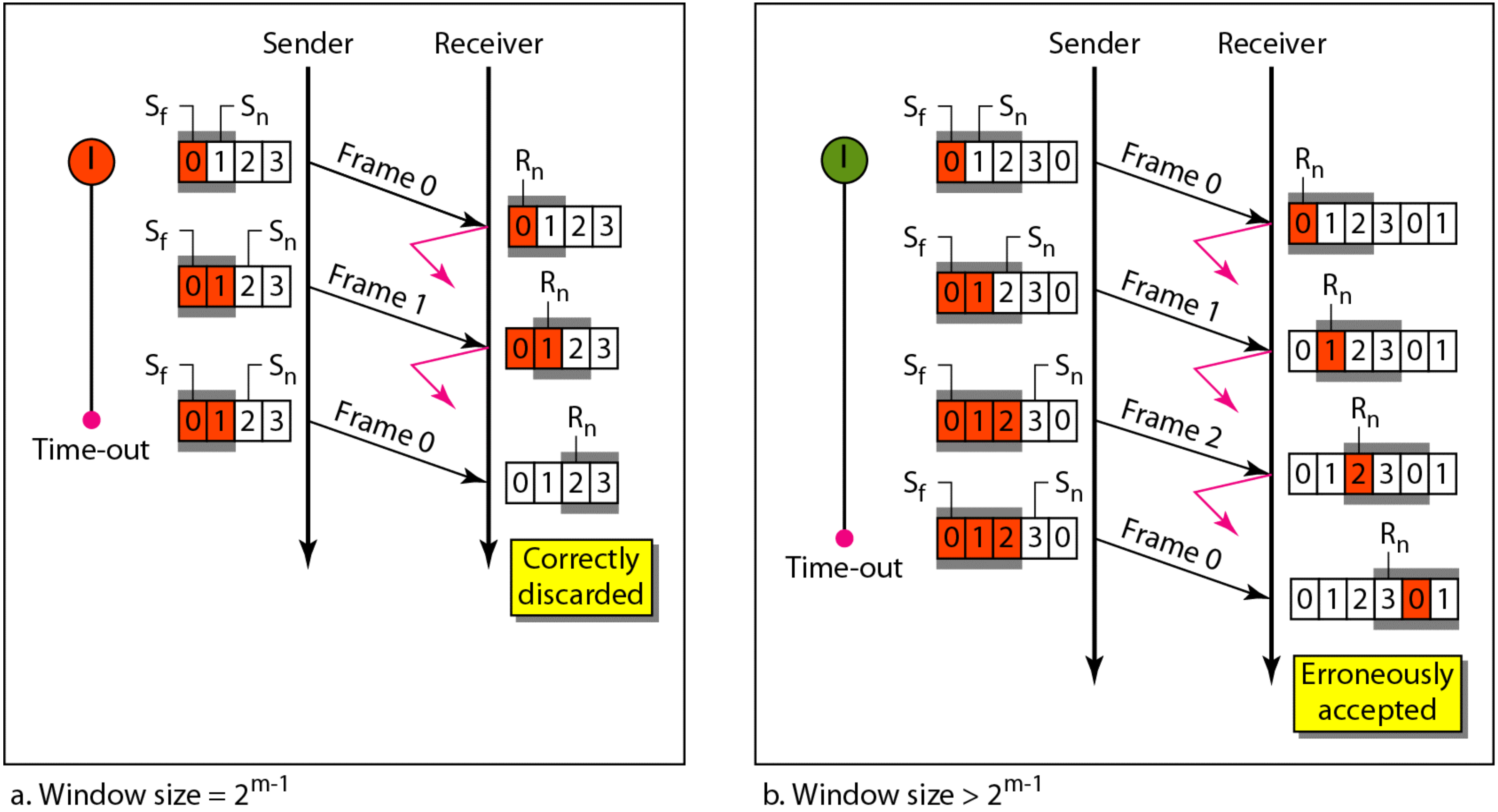
Sender Window fo SR ARQ


Send window, first outstanding frame
S_f
S_n
Send window, next frame to send
13
14
15
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
0
1
Frames already acknowledged
Frames sent, but not acknowledged
Frames that can be sent
Frames that cannot be sent
$S_{size} = 2^{m-1}$

Receive Window for SR ARQ


R_n
Receive window, next frame expected
13 14 15 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 0 1
Frames already received
Frames that can be received and stored for later delivery. Colored boxes, already received
Frames that cannot be received
$R_{size} = 2^{m-1}$

SR ARQ Window Size



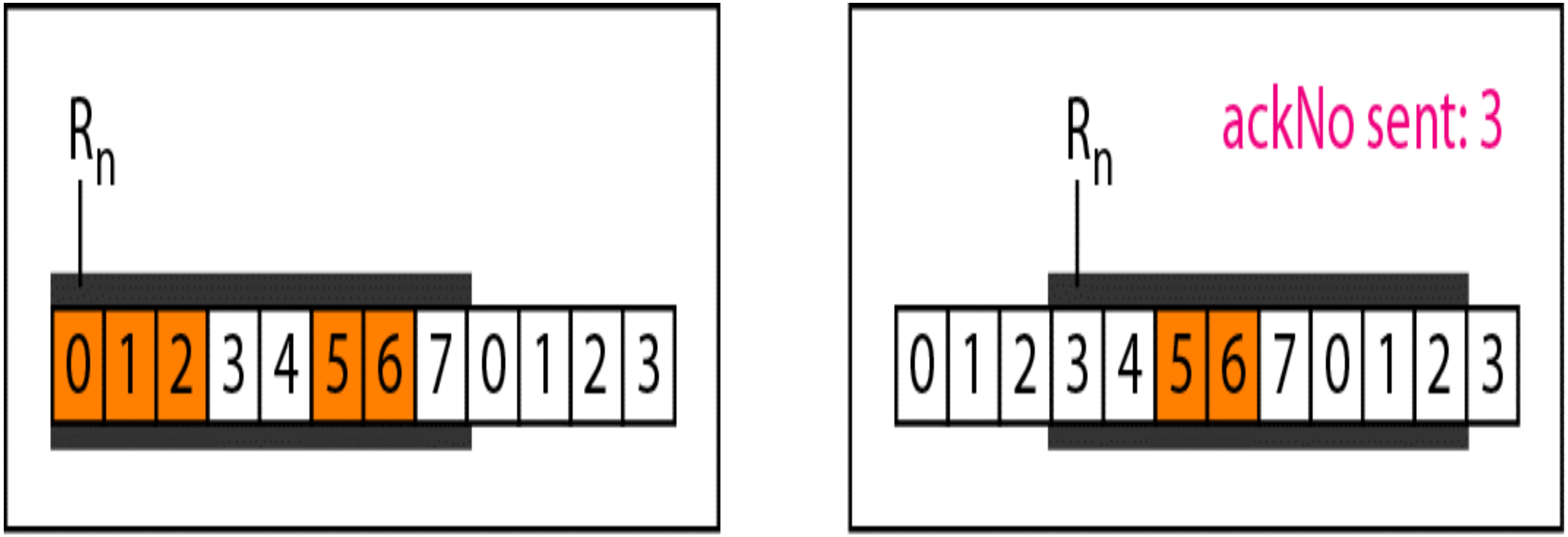
a. Window size = 2^{m-1}

b. Window size > 2^{m-1}

Summary Notes for SR ARQ

- In Selective Repeat ARQ, the size of the send window must be at most 2^{m-1}; the size of the receiver window is always 1.
- The receive window of size is usually (but not necessarily) the same as that of the sender window. Receiver will buffer any out-of-order frames
- Receive can acknowledge only frames that are in sequence

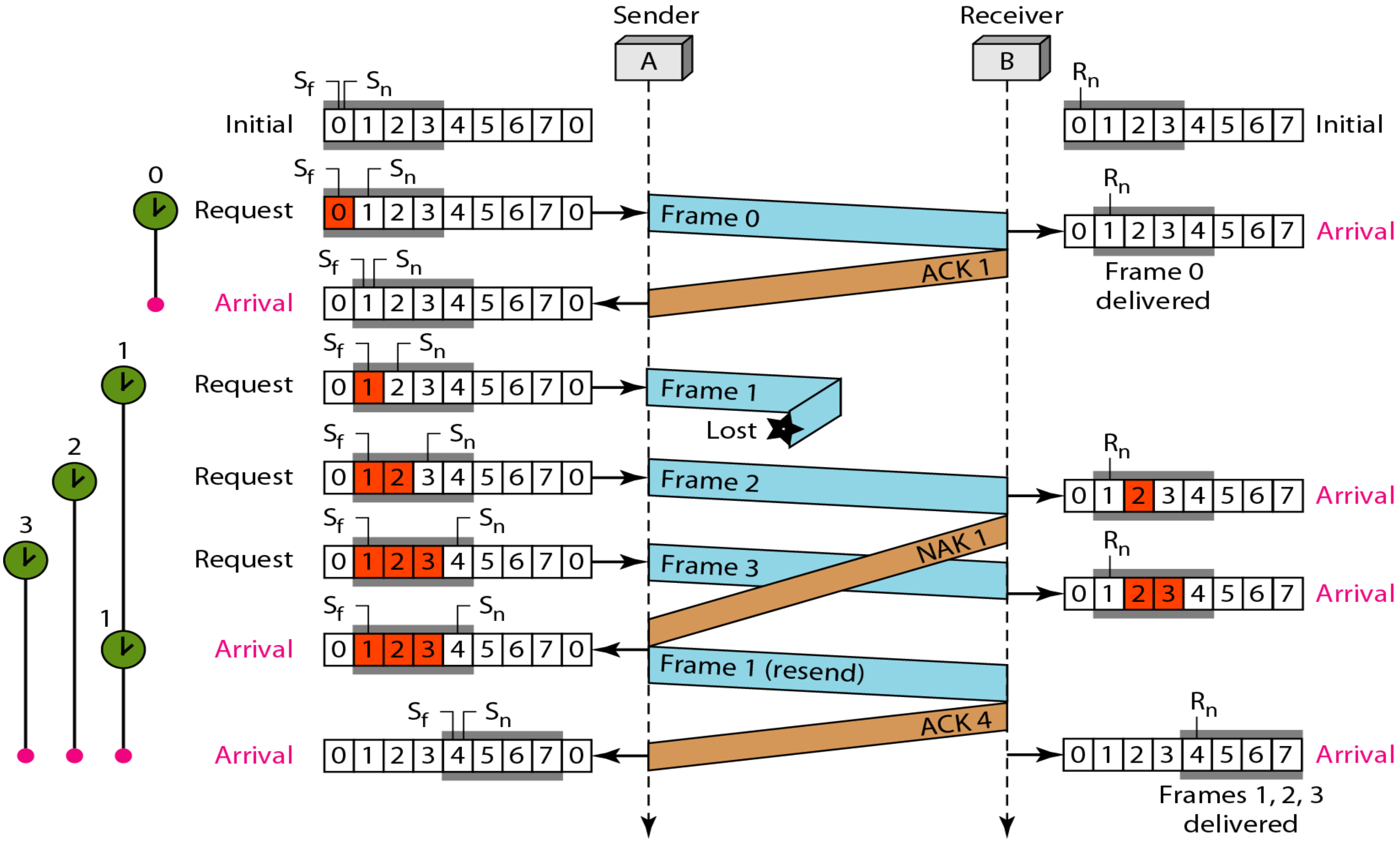
Delivery of Data in SR ARQ



a. Before delivery

b. After delivery

Example of SR ARQ


Sender
Receiver
A
B
Sf
Sn
Initial
0 1 2 3 4 5 6 7 0
Rn
0 1 2 3 4 5 6 7
Initial
0
Request
Frame 0
Arrival
Frame 0 delivered
ACK 1
Arrival
1
Request
Frame 1
Lost
2
Request
Frame 2
Arrival
3
Request
Frame 3
NAK 1
Arrival
1
Arrival
Frame 1 (resend)
ACK 4
Arrival
Arrival
Frames 1, 2, 3 delivered