Lecture 4
Peer-to-Peer Protocols and Data Link Layer

ARQ Protocols and Reliable Data Transfer

Flow Control

Lecture 4
Peer-to-Peer Protocols and Data Link Layer

ARQ Protocols and Reliable Data Transfer

Peer-to-Peer Protocols

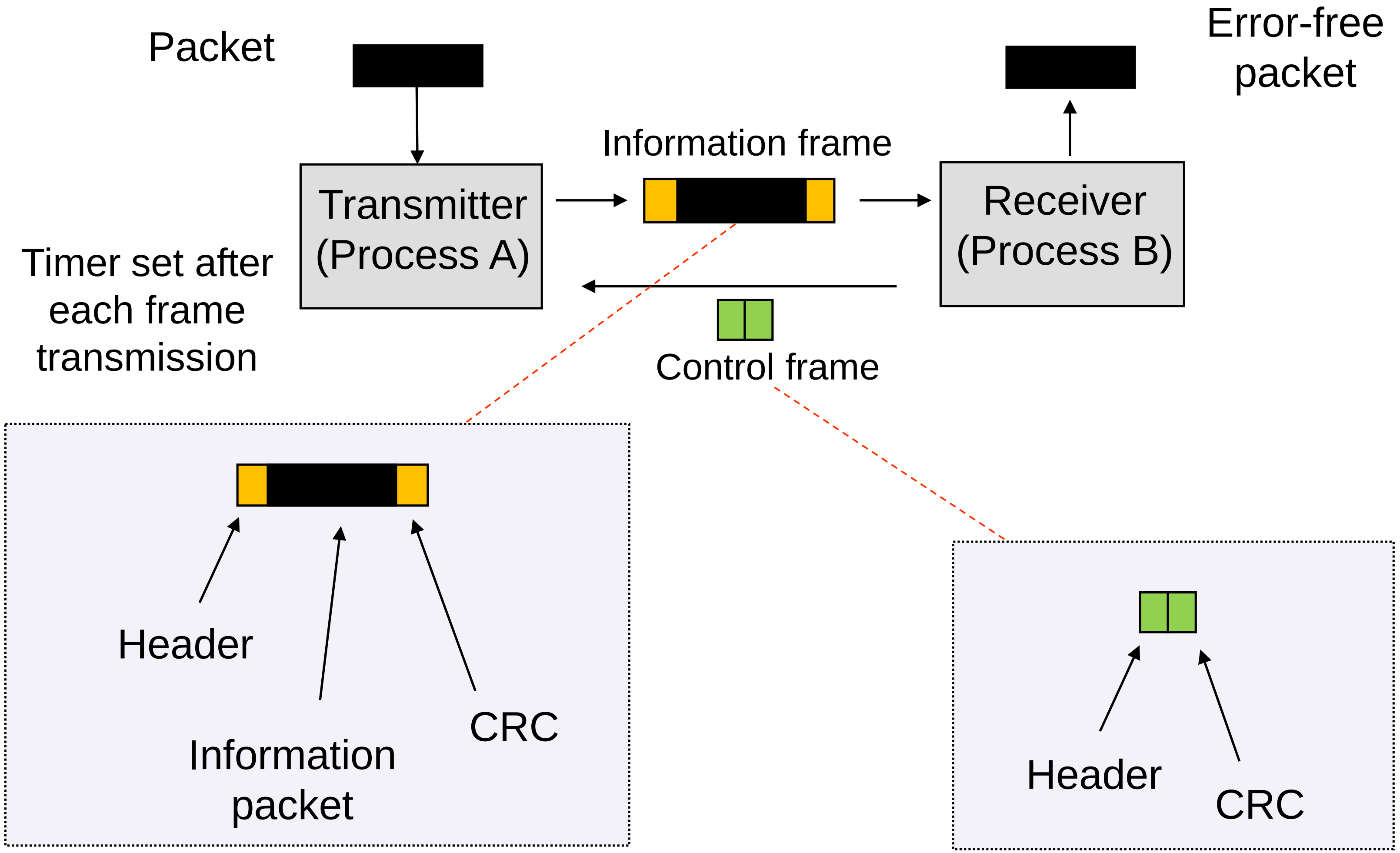
- many protocols involve the interaction between two peers
  - Service Models are discussed & examples given
  - Detailed discussion of ARQ provides example of development of peer-to-peer protocols
  - Flow control

Automatic Repeat Request (ARQ)

- *Purpose*: to ensure a sequence of information packets is delivered in order and without errors or duplications despite transmission errors & losses
- We will look at:
  - Stop-and-Wait ARQ
  - Go-Back N ARQ
  - Selective Repeat ARQ
- Basic elements of ARQ:
  - *Error-detecting code* with high error coverage
  - *ACKs* (positive acknowledgments)
  - *NAKs* (negative acknowlegments)
  - *Timeout mechanism*

Stop-and-Wait ARQ

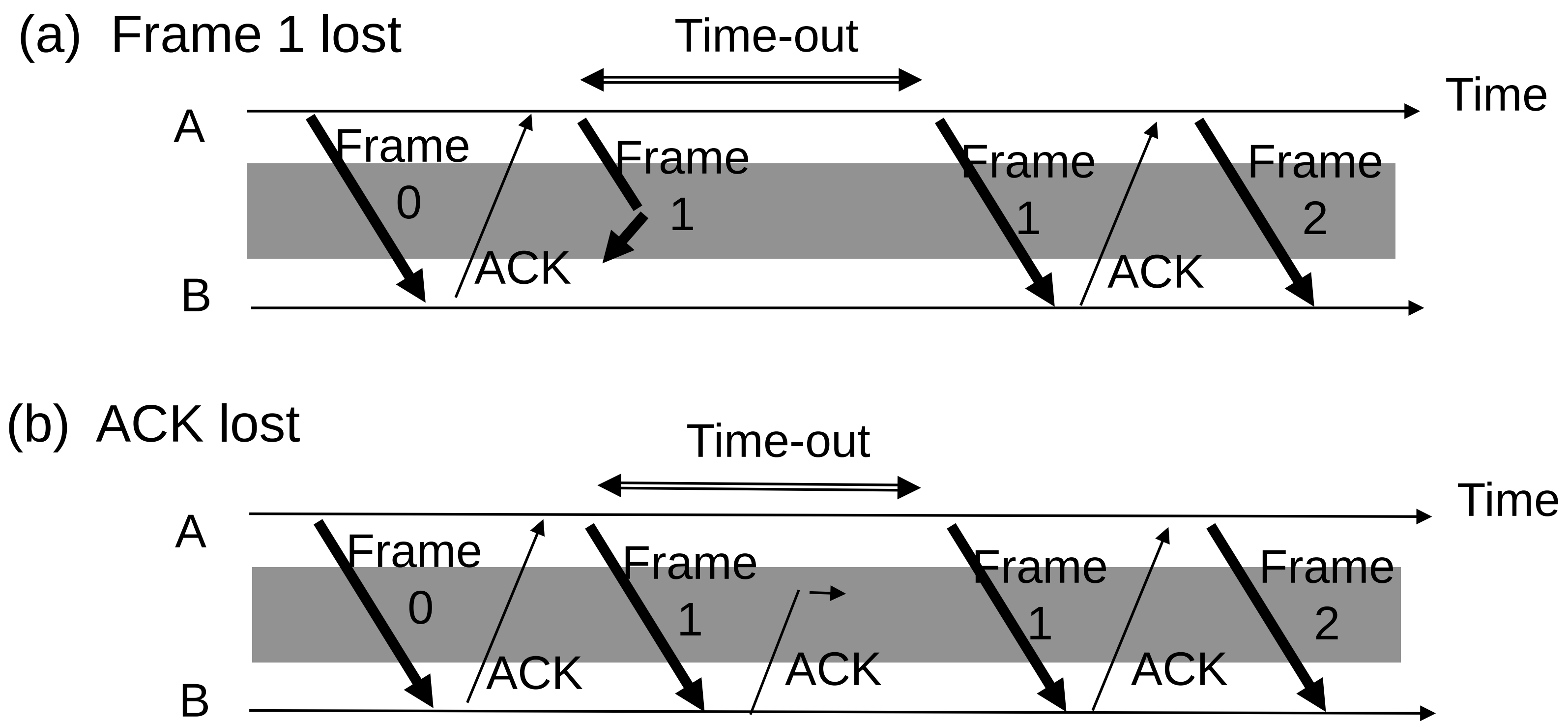
Transmit a frame, wait for ACK



Information frame

Control frame: ACKs

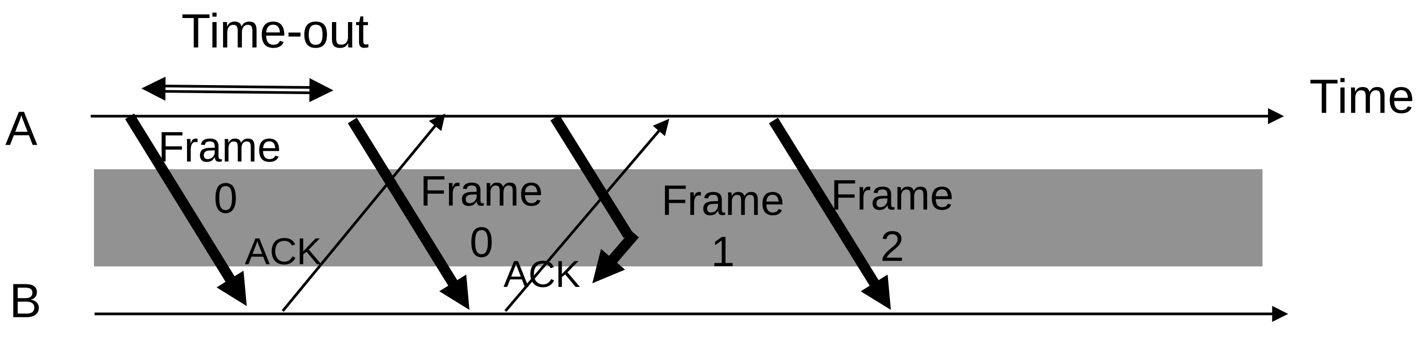
Need for Sequence Numbers



- In cases (a) & (b) the transmitting station A acts the same way
- But in case (b) the receiving station B accepts frame 1 twice
- Question: How is the receiver to know the second frame is also frame 1?
- Answer: *Add frame sequence number in header*
- S_{last} is sequence number of most recent transmitted frame

Sequence Numbers

(c) Premature Time-out



- The transmitting station A misinterprets duplicate ACKs
- Incorrectly assumes second ACK acknowledges Frame 1
- Question: How is the receiver to know second ACK is for frame 0?
- Answer: *Add frame sequence number in ACK header*
- R_{next} is sequence number of next frame expected by the receiver
- Implicitly acknowledges receipt of all prior frames

Stop-and-Wait ARQ

Transmitter

Ready state

- Await request from higher layer for packet transfer
- When request arrives, transmit frame with updated S_{last} and CRC
- Go to Wait State

Wait state

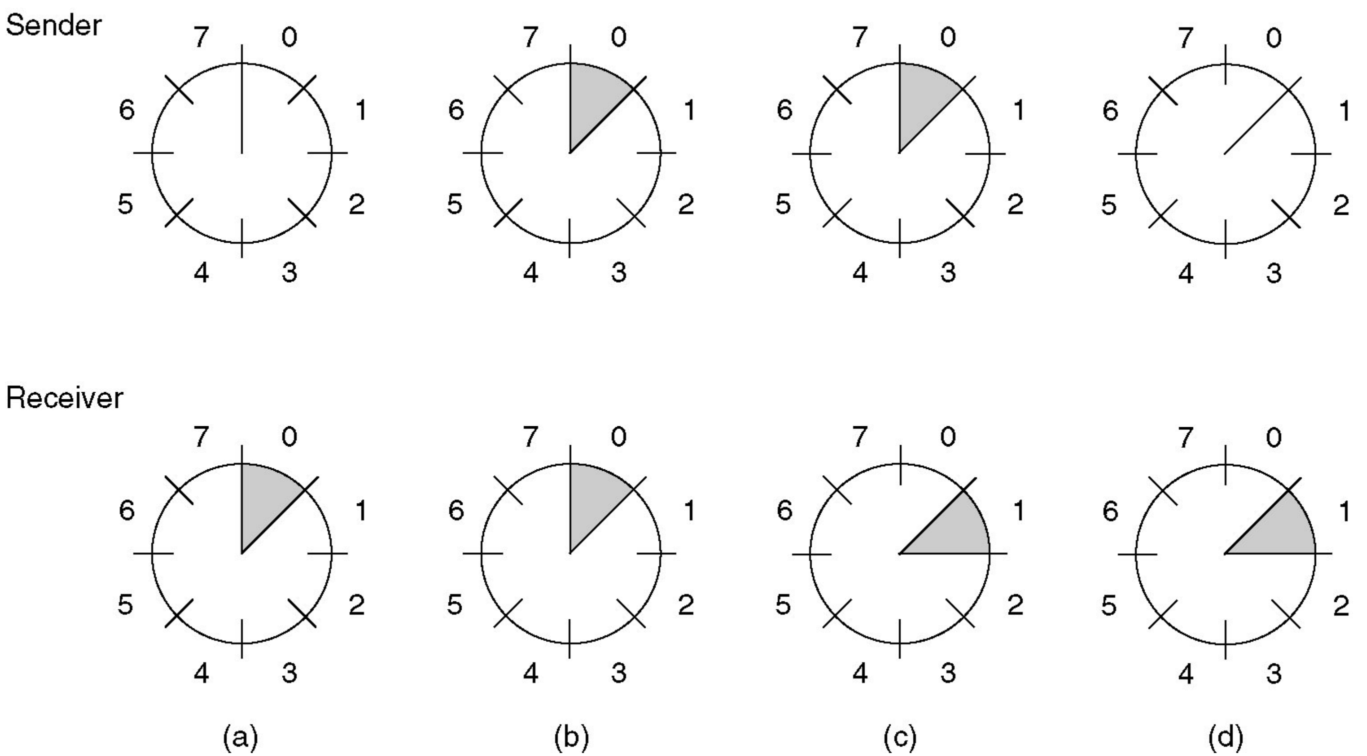
- Wait for ACK or timer to expire; block requests from higher layer
- If timeout expires
  - retransmit frame and reset timer
- If ACK received:
  - If sequence number is incorrect or if errors detected: ignore ACK
  - If sequence number is correct ($R_{next} = S_{last} + 1$): accept frame, go to Ready state

Receiver

Always in Ready State

- Wait for arrival of new frame
- When frame arrives, check for errors
- If no errors detected and sequence number is correct ($S_{last}=R_{next}$), then
  - accept frame,
  - update R_{next},
  - send ACK frame with R_{next},
  - deliver packet to higher layer
- If no errors detected and wrong sequence number
  - accept frame
  - send ACK frame with R_{next}
- If errors detected
  - discard frame

Sliding Window Protocols



Suppose that a sliding window of size is 1, with a 3-bit sequence number.

(a) Initially.

(b) After the first frame has been sent.

(c) After the first frame has been received.

(d) After the first acknowledgement has been received.

Practice on Slide Window Protocols

- Draw a series of buffers at a sender and a receiver which illustrates the following (a), (b), (c), and (d).

  Suppose that a sliding window of size is has a 3-bit sequence number. Thus, each of them has 8 buffers.

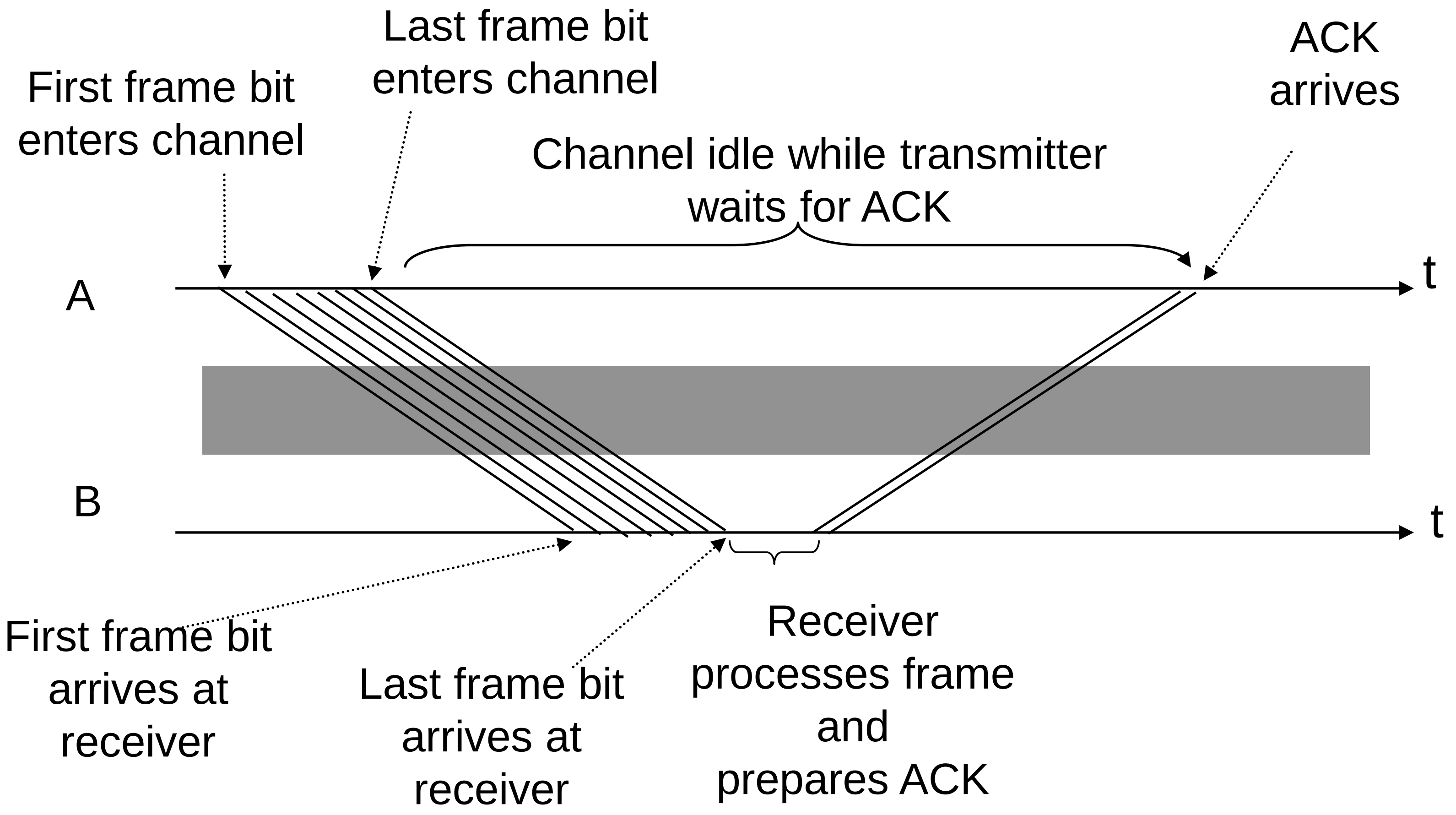
  (a) Initially. The receiver knows that the sender will send a data.

  (b) After the first frame has been sent.

  (c) After the first frame has been received & the ACK has been sent by the receiver.

  (d) After the first acknowledgement has been received.

Stop–and–Wait Efficiency



- **10000 bit frame @ 1 Mbps takes 10 ms to transmit**
- **If wait for ACK = 1 ms, then efficiency = 10/11= 91%**
- **If wait for ACK = 20 ms, then efficiency =10/30 = 33%**

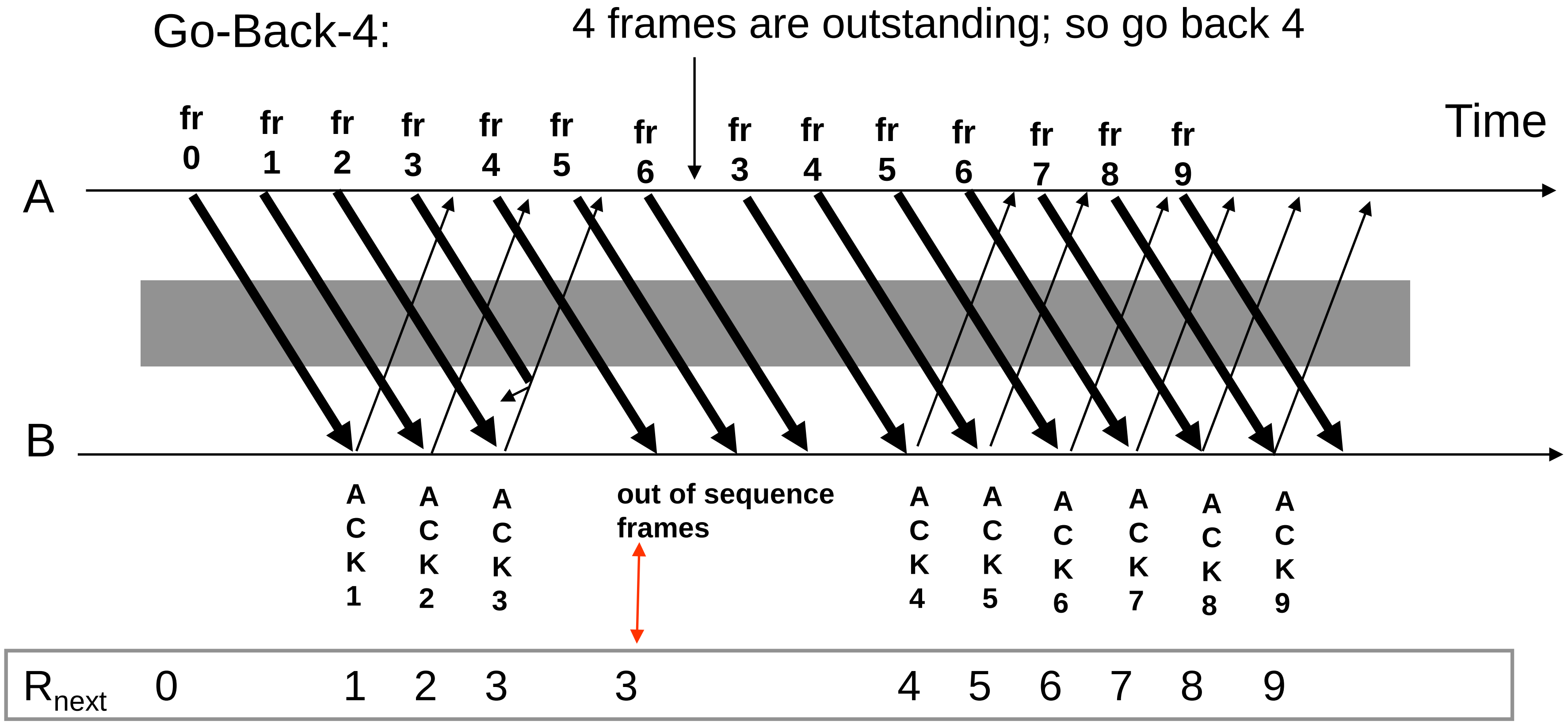
Applications of Stop-and-Wait ARQ

- IBM *Binary Synchronous Communications protocol* (Bisync): character-oriented data link control
- *Xmodem*: modem file transfer protocol
- *Trivial File Transfer Protocol* (RFC 1350): simple protocol for file transfer over UDP

Go-Back-N

- Improve Stop-and-Wait by not waiting!
- Keep channel busy by continuing to send frames
- Allow a window of up to W_s outstanding frames
- Use *m*-bit sequence numbering
- If ACK for oldest frame arrives before window is exhausted, we can continue transmitting
- If window is exhausted, pull back and retransmit all outstanding frames
- Alternative: Use timeout

Go–Back–N ARQ



- Frame transmission are *pipelined* to keep the channel busy
- Frame with errors and subsequent out-of-sequence frames are ignored
- Transmitter is forced to go back when window of 4 is exhausted

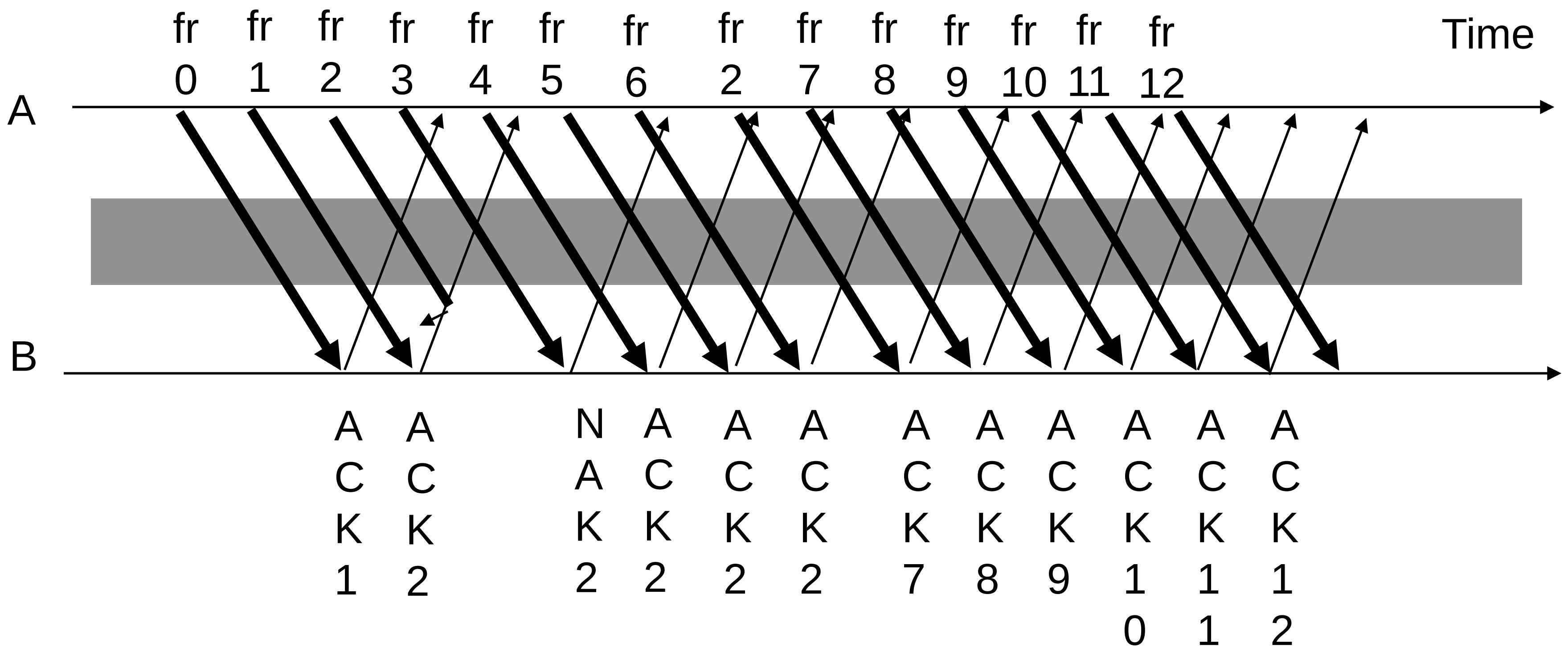
Applications of Go-Back-N ARQ

- *HDLC* (High-Level Data Link Control): bit-oriented data link control
- *V.42 modem*: error control over telephone modem links

Selective Repeat ARQ

- Go-Back-N ARQ inefficient because *multiple* frames are resent when errors or losses occur
- Selective Repeat retransmits *only an individual frame*
  - Timeout causes individual corresponding frame to be resent
  - NAK causes retransmission of oldest un-acked frame
- Receiver maintains a *receive window* of sequence numbers that can be accepted
  - Error-free, but out-of-sequence frames with sequence numbers within the receive window are buffered
  - Arrival of frame with R_{next} causes window to slide forward by 1 or more

Selective Repeat ARQ



Applications of Selective Repeat ARQ

- *TCP* (Transmission Control Protocol): transport layer protocol uses variation of selective repeat to provide reliable stream service
- *Service Specific Connection Oriented Protocol*: error control for signaling messages in ATM networks

Lecture 4
Peer-to-Peer Protocols and Data Link Layer

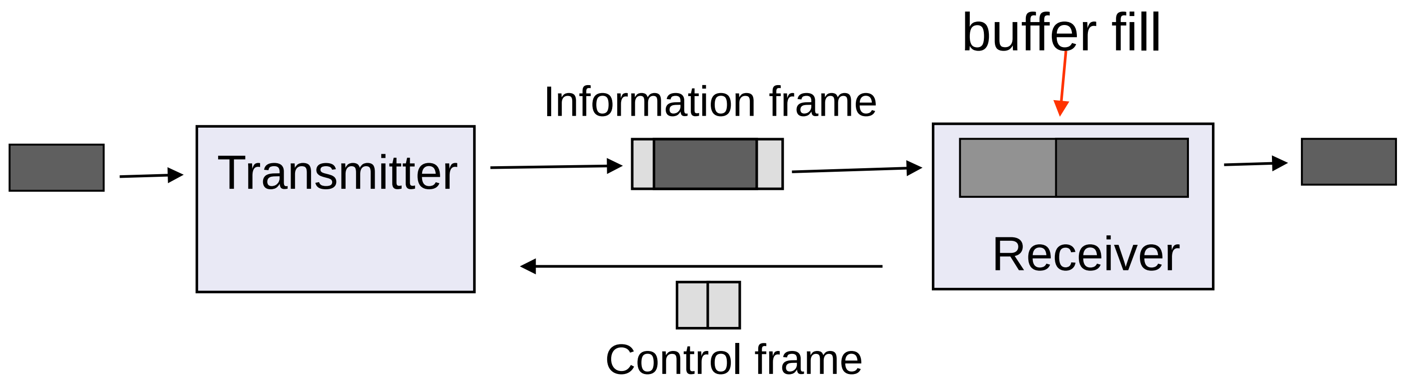
ARQ Protocols and Reliable Data Transfer

Flow Control

Lecture 4
Peer-to-Peer Protocols and Data Link Layer

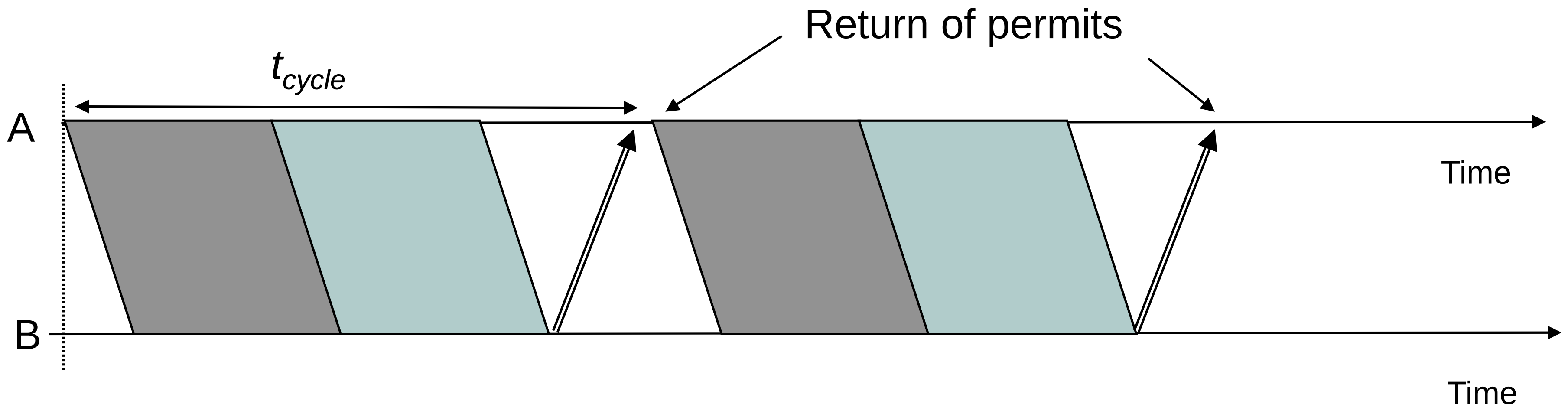
Flow Control

Flow Control



- Receiver has limited buffering to store arriving frames
- Several situations cause buffer overflow
  - Mismatch between sending rate & rate at which user can retrieve data
  - Surges in frame arrivals
- *Flow control* prevents buffer overflow by regulating rate at which source is allowed to send information

Window Flow Control



- Sliding Window ARQ method with W_s equal to buffer available
  - Transmitter can never send more than W_s frames
- ACKs that slide window forward can be viewed as permits to transmit more
- Can also pace ACKs as shown above
  - Return permits (ACKs) at end of cycle regulates transmission rate
- Problems using sliding window for both error & flow control
  - Interplay between transmission rate & retransmissions