CS455 HW 4 Solutions

(2018)

Question 4

- If you look at Stop-and-Wait carefully, you will see that:

Stop-and-Wait = Go-back-N using:

send window size = 1
recv window size = 1

We have proved that for Go-Back-N that:

$$W_{send} + W_{recv} \leq N$$ ← N = # seq. numbers

The protocol will guarantee reliable communication

Therefore:

$$W_{send} + W_{recv} = 1 + 1 = 2.$$

If we have 2 seq. numbers in Stop-and-wait, Stop-and-wait will guarantee reliable communication.

□

Question 2:

Go-back-N : Recv. Window Size = 1.

Given: Send-Window Size = 4.

Initial state:

SW = {3, 0, 1, 2}

Sender

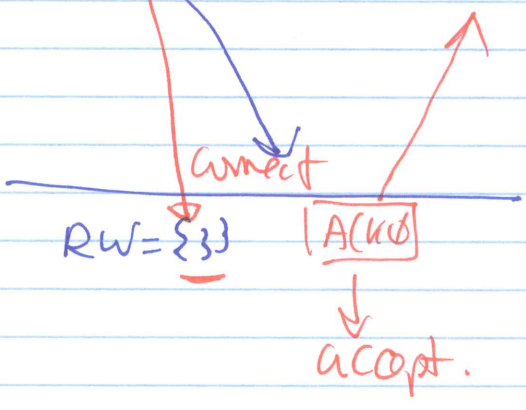
Receiver

RW = {0}

Exchange:

{3, 0, 1, 2}

(1) Sender [3]

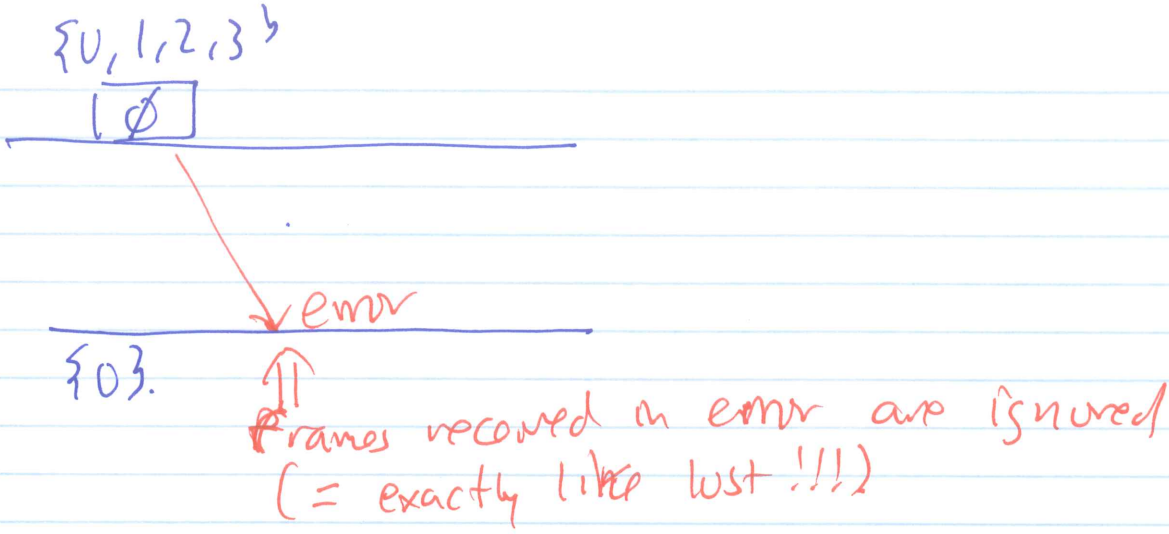


Receiver sends ACK 0

New state: SW = {0, 1, 2, 3}

RW = {0}.

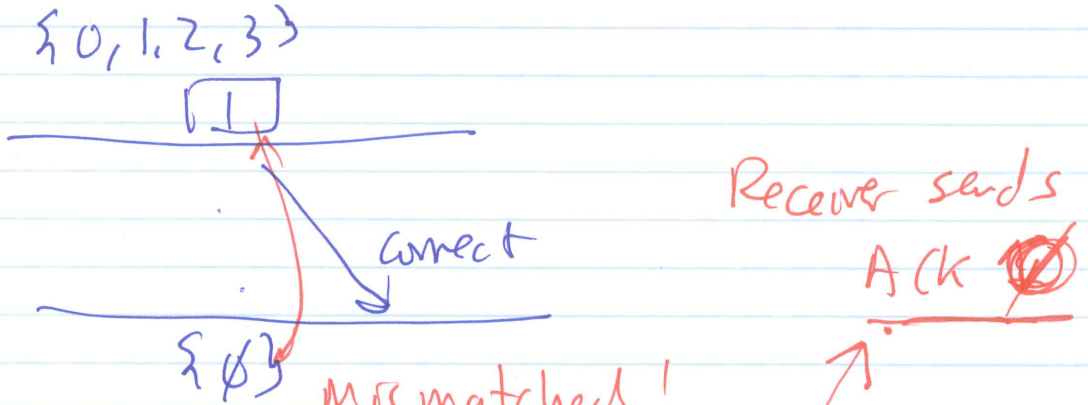
(2)



Frames received in error are ignored
(= exactly like lost !!!)

⇒ Receiver does nothing!

(3)

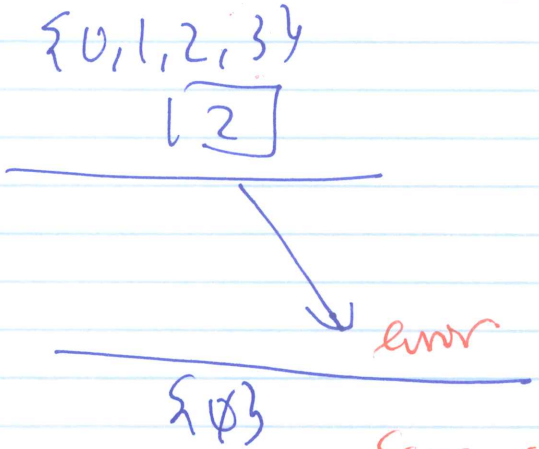


Mismatched!

Receiver sends ACK ∅

triggers a response.

(4)

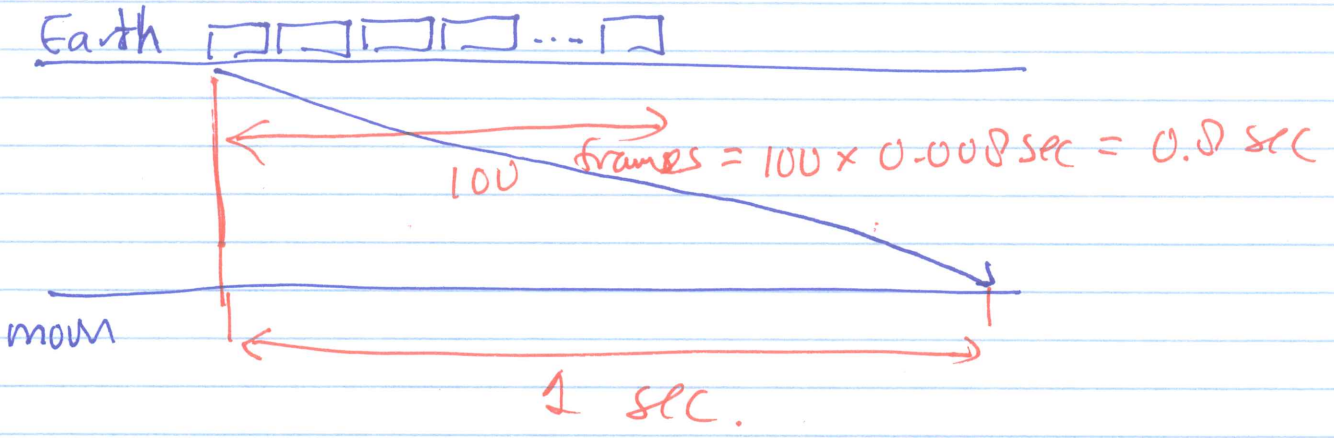


Same as (2) above

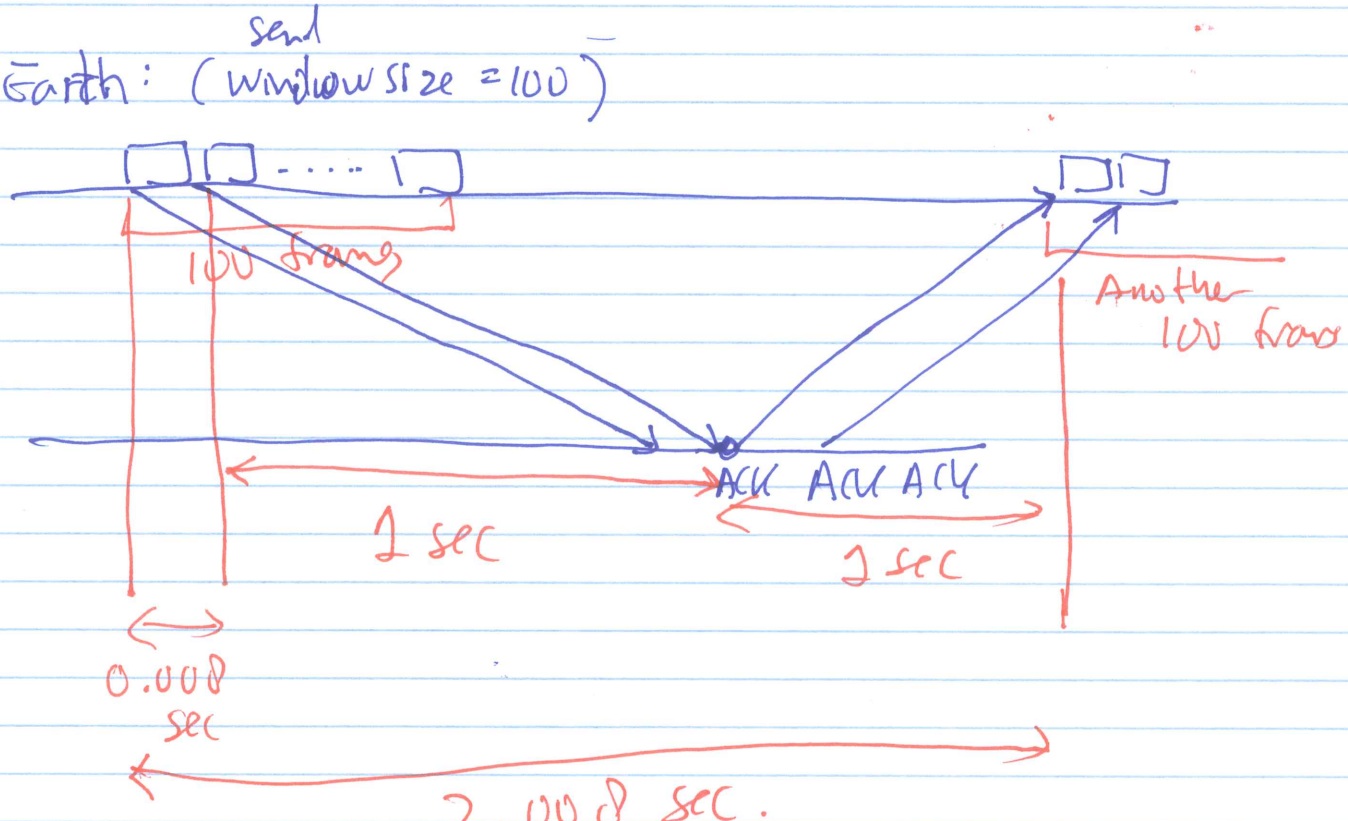
Receiver does nothing.

Question 3

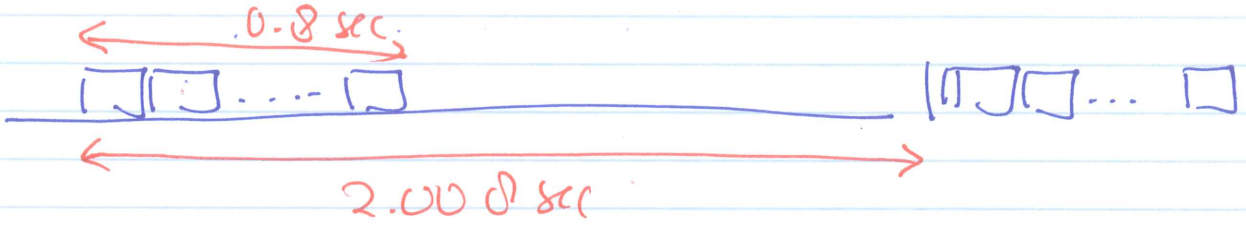
8000 bits = $\frac{8000}{1M}$ sec = 0.008 sec.



When there are no errors, sender and receiver in sliding window will exchange frames / ACKs as follows:



If you look at the sender:

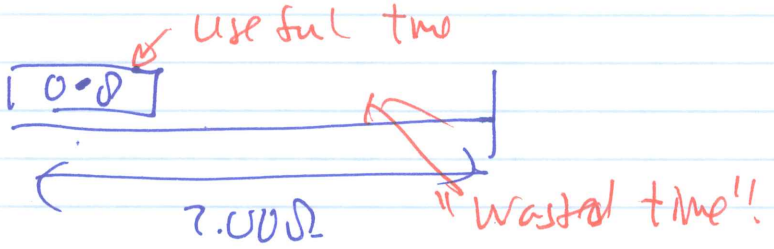


Sender goes like this:

transmit data frames for 0.8 sec
Then idle (wait for ACKs) until time 2.008

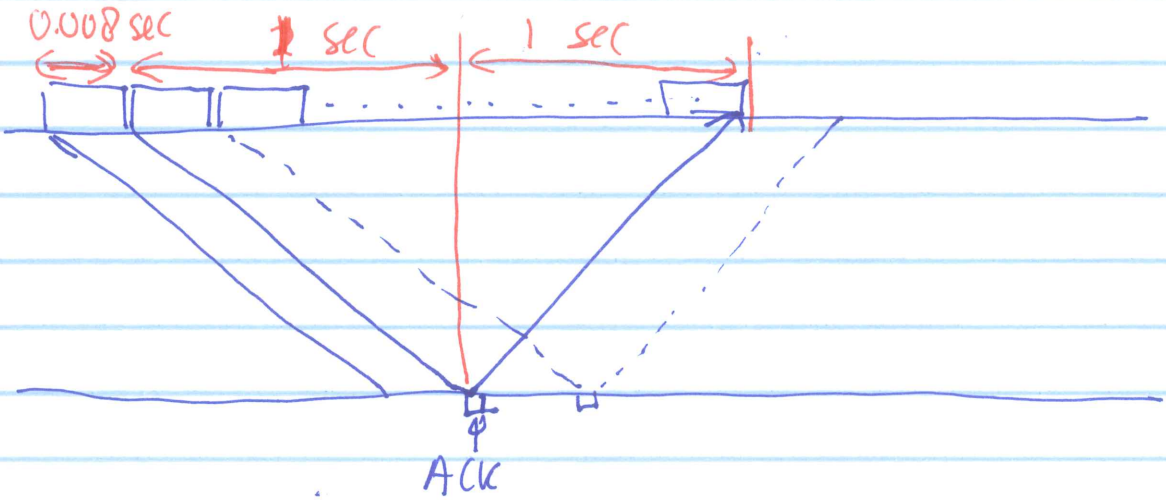
Then repeat the same cycle.

So: Utilization (fraction of time used in transmit)



$$\text{Utilization} = \frac{0.8}{2.008} = 39.84\%$$

(B) In order for the sender to transmit continuously, the ACK for the first frame must return in time:



(Note: the ACK for the 2nd frame will be on time (assuming no error) if the first ACK is on time !!!).

The sender must transmit this number of frames:

$$\#\text{ frames} = \frac{2.008\text{ sec}}{0.008\text{ sec}} = 251$$

$$\#\text{ bits} = 8$$

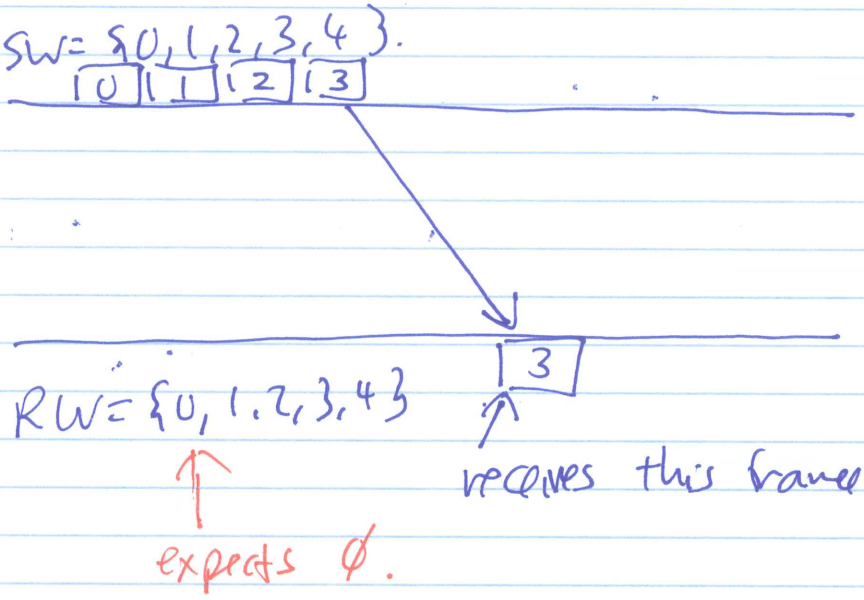
Question 4

Given: $W_{send} = 5$ $\quad$ $W_{recv} = 5$ $\quad$ 3 bit seq #: 0, 1, 2 ..., 7. (wraps around!!)

Initially: $SW = \{0, 1, 2, 3, 4\}$

$RW = \{0, 1, 2, 3, 4\}$.

(1)



Receiver will ⇒ (1) buffer [3] (because it's in the RW range)

(2) send ACK 7 (0 − 1 = 7 in 3 bits)

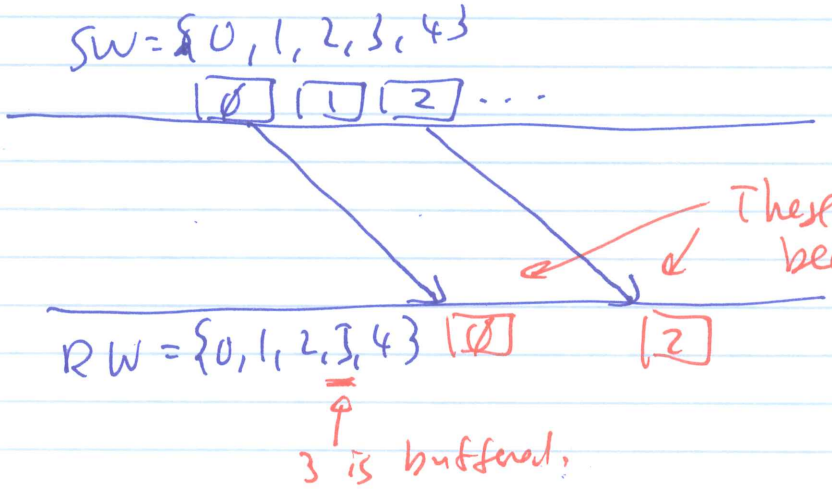
Answer:

sends ACK 7

(2) Fact: because receiver did not send an ACK that allow the sender to shift its send window forward, we know:

$SW = \{0, 1, 2, 3, 4\}$ (unchanged)

Su:



Receiver will: (1) buffer 0 and 2

(2) deliver 0, shift RW

$RW = \{1, 2, 3, 4, 5\}$.

(3) send ACK 0

Answer: sends ACK 0

New state is used to answer the next question!!!

(3) Fact: Because receiver sent: ACK 0

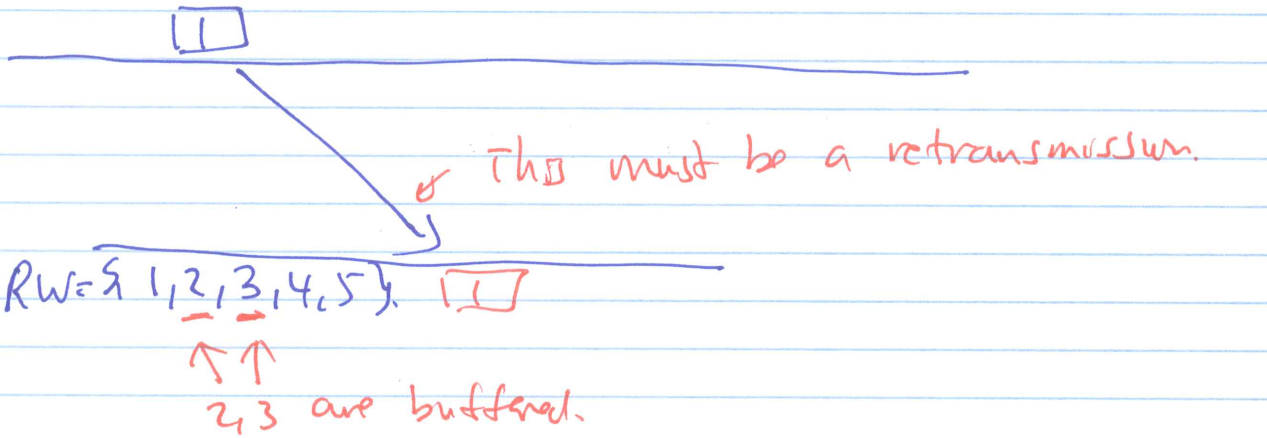
Sender's Window can be:

SW = {0, 1, 2, 3, 4} ← ACK 0 lost

or SW = {1, 2, 3, 4, 5} ← ACK 0 received

In both cases: #1 denotes an "old" frame.

SW:



Receiver will: (1) buffer [1]

(2) Deliver: [1] [2] [3]

(3) Shift RW = {4, 5, 6, 7, 0}

(4) send ACK 3

Answer: send ACK 3

(4) Fact: Because receiver sent:

ACK ∅ (part 2)

and ACK 3 (part 3)

The sender window can be one of:

SW = {0,1,2,3,4} ← ACK ∅, ACK 3 lost (old frame ∅ !!!, ACK ∅)

or: SW = {1,2,3,4,5} ← ACK ∅ recv, ACK 3 lost

or: SW = {4,5,6,7,0} ← ACK 3 recv'd. (new frame ∅ !!!)

Therefore:

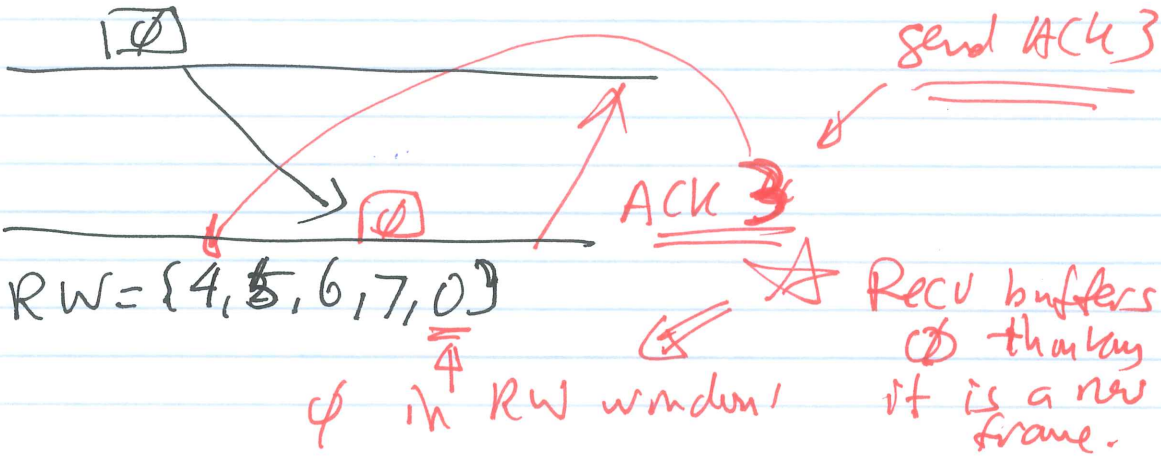
sender can send an old frame ∅

(when SW = {0,1,2,3,4}) ← old frame

or a new frame ∅

(when SW = {4,5,6,7,0}). ← new frame.

Receiver action when [∅] is received:



(5) The sender window can be one of:

ACK 0

SW = {0,1,2,3,4} ← ACK 0, ACK 3 lost

or: SW = {1,2,3,4,5} ← ACK 0 recv'd, ACK 3 lost

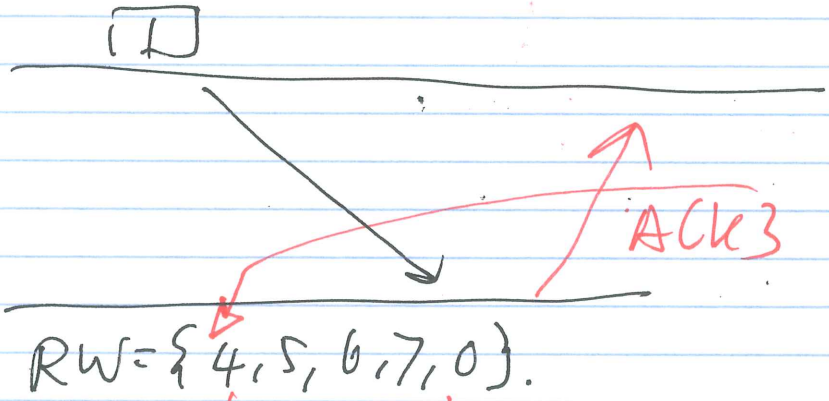
or: SW = {4,5,6,7,0} ← ACK 3 recv'd.

new frame number #1 is not possible !!!

The seq. # 1 has NOT wrapped around !!!

→ So frame 1 must be an old frame !!!

Receiver action when [1] is received:



1 is not in RW.

⇒ discard [1] (dup. ⇒ buffer)

⇒ send ACK 3

Question 5

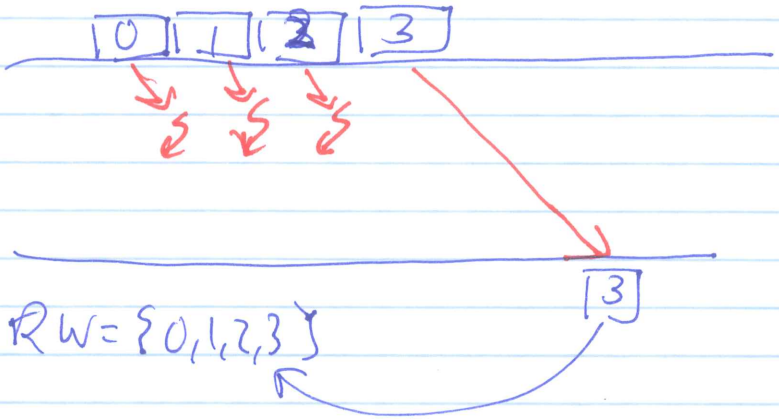
Given: $W_{send} = 4$ 3 bit seq#: 0, 1, 2, ..., 7

$W_{recv} = 4$

Initially: $SW = \{0, 1, 2, 3\}$.

$RW = \{0, 1, 2, 3\}$.

(1) $SW = \{0, 1, 2, 3\}$

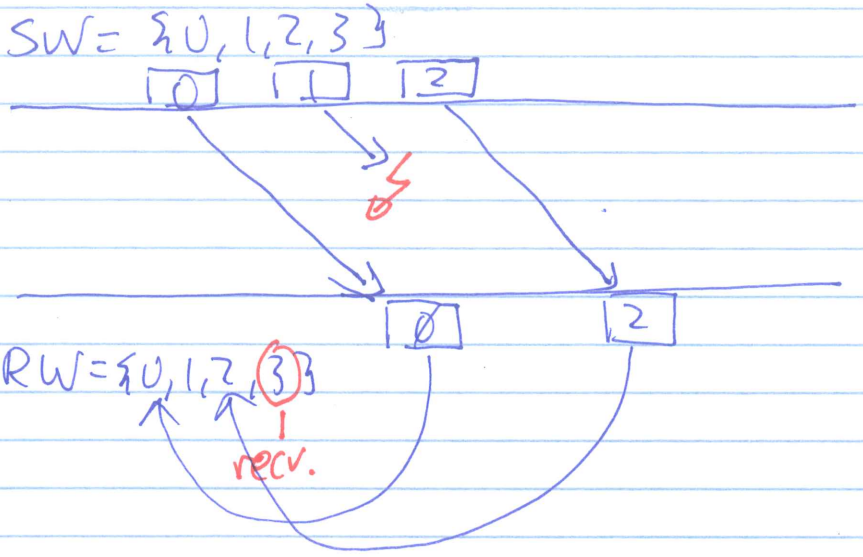


Receiver will: ① buffer [3]

② send ACK 7

Answer: sends ACK 7

(2) Next:



Receiver will: (1) buffer [0] and [2]

(2) deliver [0] !!!

RW = {1, 2, 3, 4}

(3) Send ACK 0

(to acknoledge [0]).

Answer:

sends ACK 0

(3) Pre-analysis:

ACK 0 can be received or lost

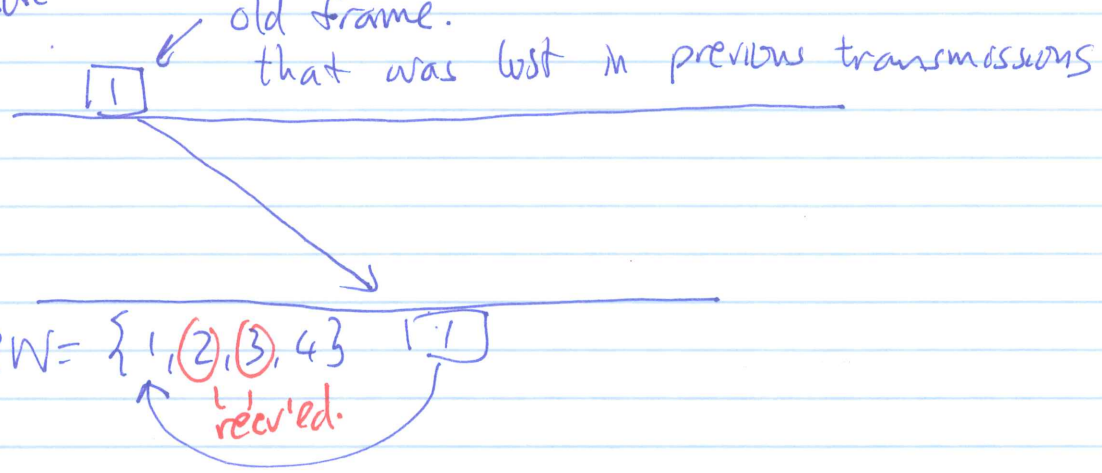
Sender window can be:

$SW = \{0, 1, 2, 3\}$

or $SW = \{1, 2, 3, 4\}$

Seq. no 1 is used to label an OLD frame in BOTH SW window !!

Therefore:



Receiver will:

① buffer 1

② deliver: 1 2 3

RW = {4, 5, 6, 7}.

③ Send ACK 3

Answer:

Send ACK 3

(4) Pre-analysis:

Sender window can be one of these:

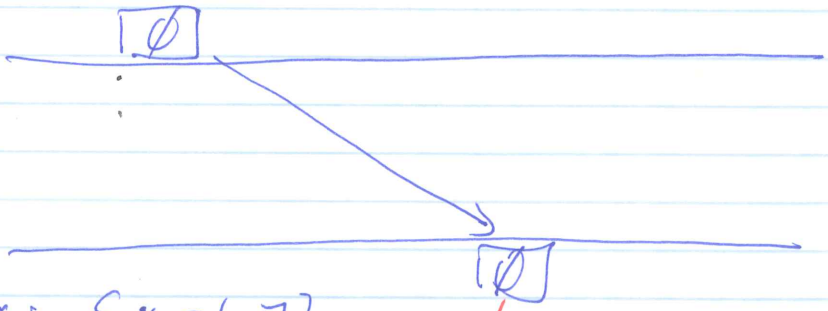
SW = {0, 1, 2, 3}. ← Ack 0, Ack 3 both lost

or SW = {1, 2, 3, 4}. ← Ack 3 lost.

or SW = {4, 5, 6, 7} ← Ack 3 received.

0 will ALWAYS used for a re-transmission of an old frame (frame 0 was not Acked !!!).

Receiver action:



RW = {4, 5, 6, 7}

Not in RW. → Discard!

Receiver will: (1) discard 0

(2) send Ack 3

Answer: send Ack 3 ⟷ send Ack 3

This frame 0 is ALWAYS a retransmission, cannot be a new frame

(5) same pre-analysis:

Sender window can be one of these:

SW = {0, 1, 2, 3} ← Ack 0, Ack 3, Ack 3 lost

or SW = {1, 2, 3, 4} ← Ack 3, Ack 3 lost

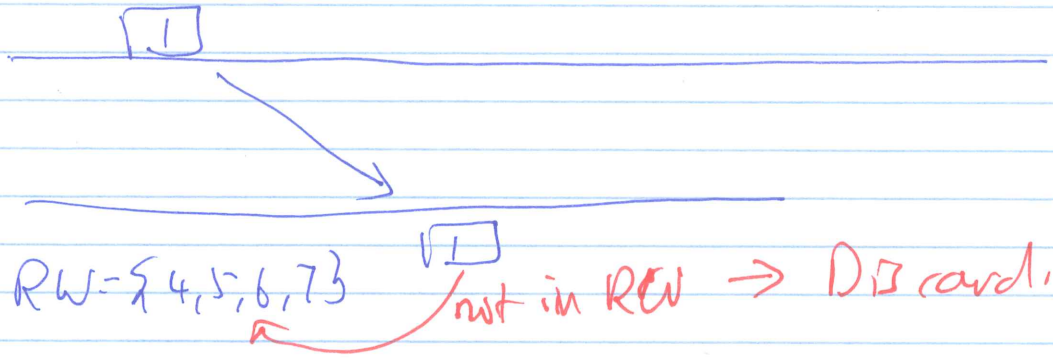
or SW = {4, 5, 6, 7}. ← one of Ack 3 recv'd.

1 is used when Ack 3's were lost

→ 1 must be a retransmission

(frame 1 was not acknowledged yet !!!)

Receiver action:



Receiver will: (1) Discard 1

(2) send Ack 3

Answer: send Ack 3

This frame 1 is ALWAYS a retransmission (because sender did not receive any Ack 3 (which would ack frame 1)).