If $BW = 1\,GBps$ (B → Byte), then what is wrap around time

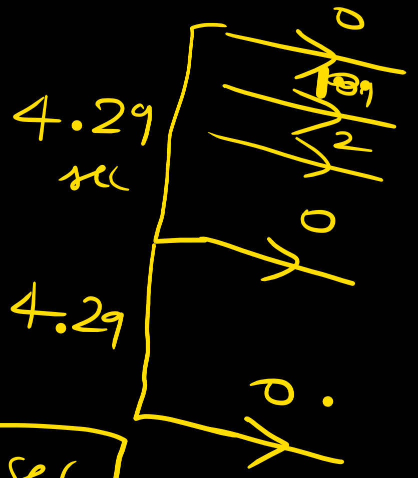
1 S — 1 GB

1 G Bytes → 1 sec.

1 G seq number → 1 sec.

$1 \text{ seq} \rightarrow \frac{1}{G} \text{ sec} = \frac{1}{10^9} \text{ sec}.$

$2^{32} \text{ seq} \rightarrow \frac{2^{32}}{10^9} \text{ sec} \cong \boxed{4.294 \text{ sec}}.$



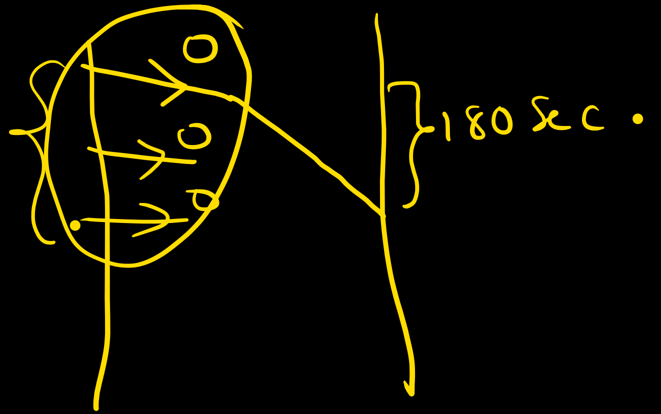
But lifetime = 3 min = 180 sec.

This is a problem

WAT < LT.

Always WAT > LT for proper operation.

Same seq numb is repeated even before old seq number is dead.



If BW = 1 GBbps and lifetime = 180 sec, then what is min seq number required.

WAT > LT ⇒ WAT > 180 sec which means we cannot repeat seq numbers within 180 sec. So we need distinct seq numbers for all bytes going out in 180 sec. So we have to find bytes going out in 180 sec.

$1\,GB \rightarrow 1\,sec$

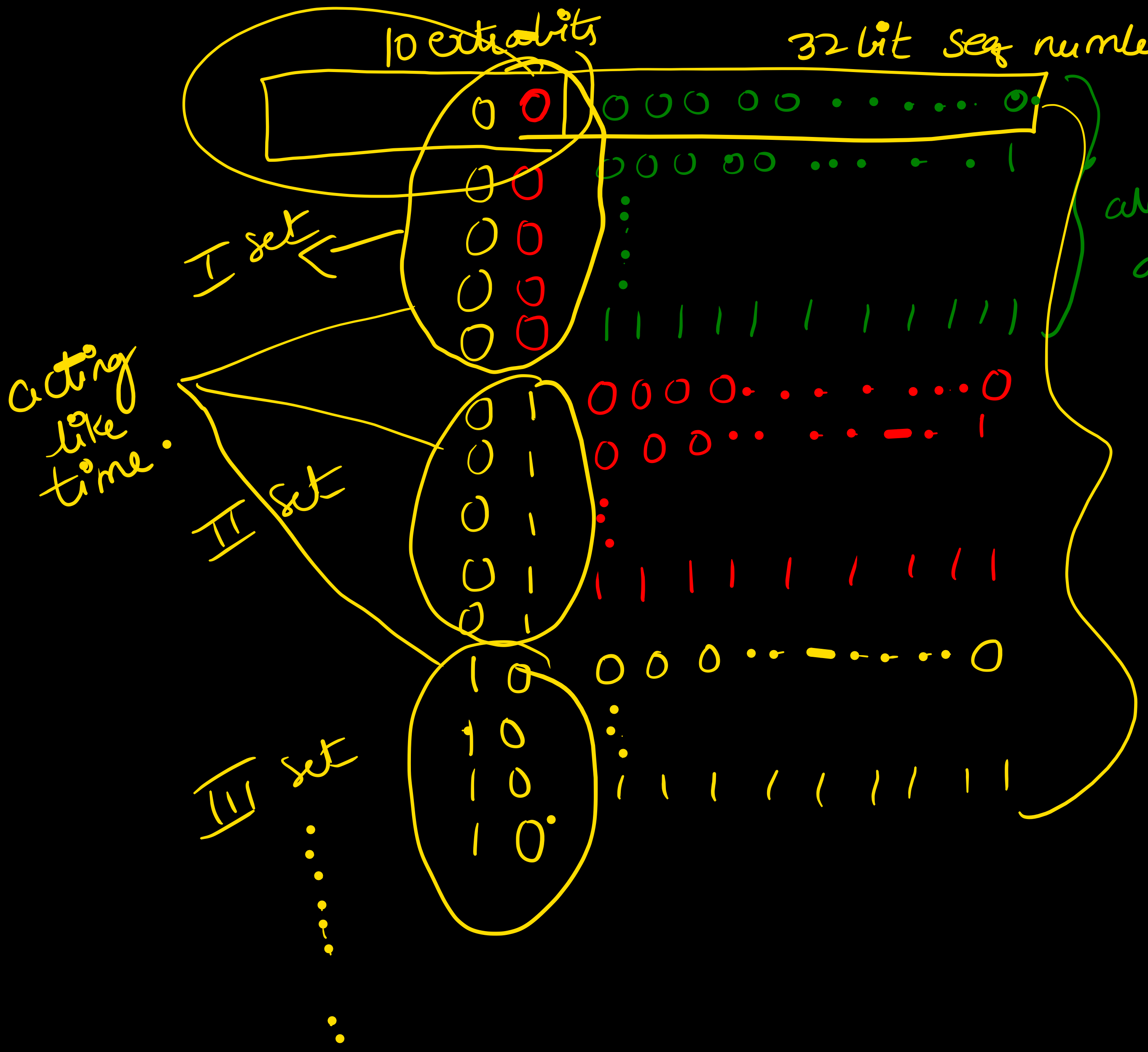
$1\,sec \rightarrow 10^9$ Bytes

$180\,sec \rightarrow 180 \times 10^9$ Bytes $\rightarrow 180 \times 10^9$ seq numbers.

∴ Bits in seq number field $= \lceil \log_2(180 \times 10^9) \rceil = 42$ bits.

But we have only 32 bits in SNF. ∴ 10 bits extra needed.

not time $\leftarrow$ Time stamp $\leftarrow$ 10 bits $\rightarrow$ options



all seq numbers are over.

2^{10} set

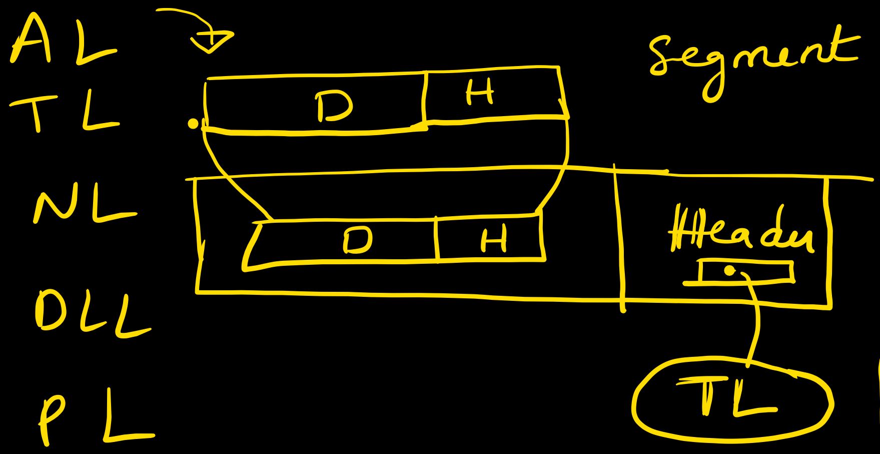
$\boxed{\text{if BW} = 1\text{GBps}}$

H/w: Bits
for 10GBps
100GBps

Acknowledgment number is always, seq number of next byte expected.

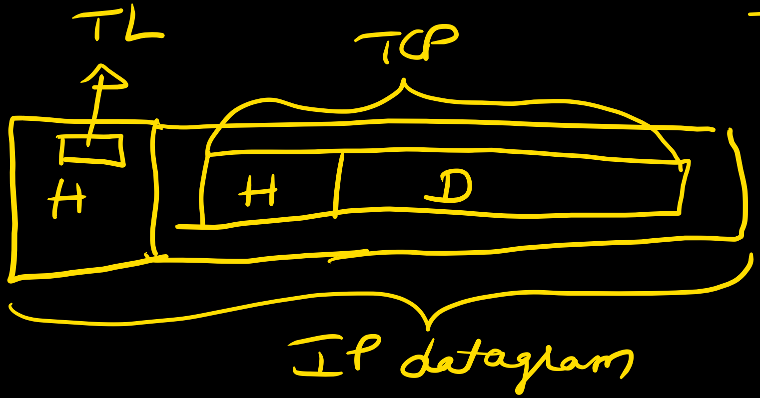
$\longrightarrow$ ack no = 200

Seq number on a segment is always 1 byte seq number.

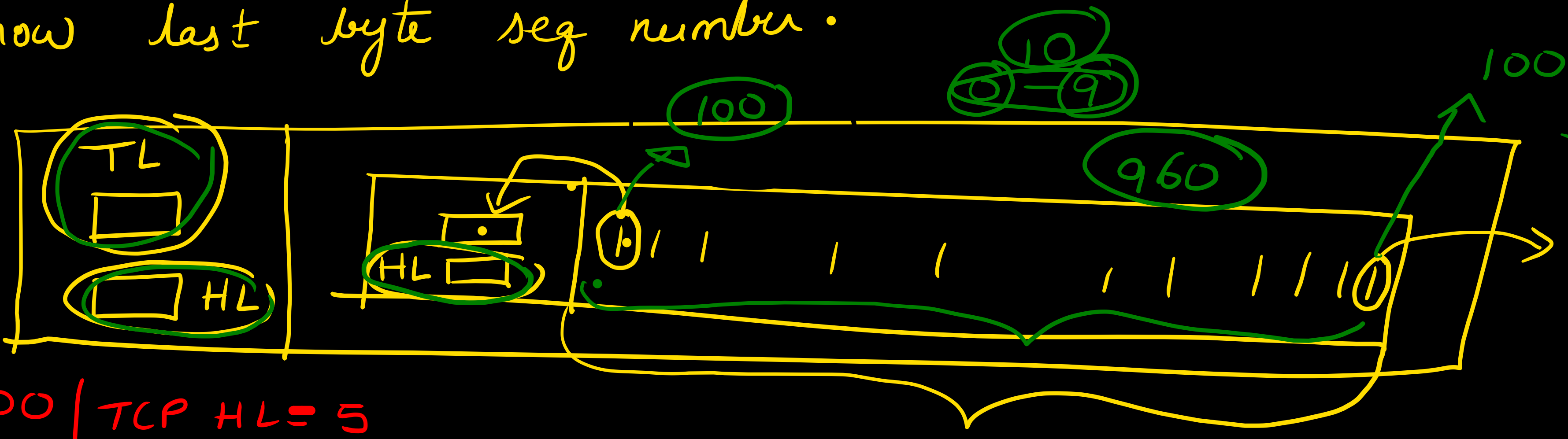


$\therefore$ Total length in IP datagram

= (IP header + TCP header + TCP data) in Bytes



I byte seq number is in the header of TCP segment. How will we know last byte seq number.



$100 + 960 - 1 = 1059$ ✓

Ex: TL = 1000 | TCP HL = 5

IP HL = 5 | $\underbrace{\text{Seq num} = 100}_{\text{I byte seq num.}}$

Data = TL − IPH − TCPH

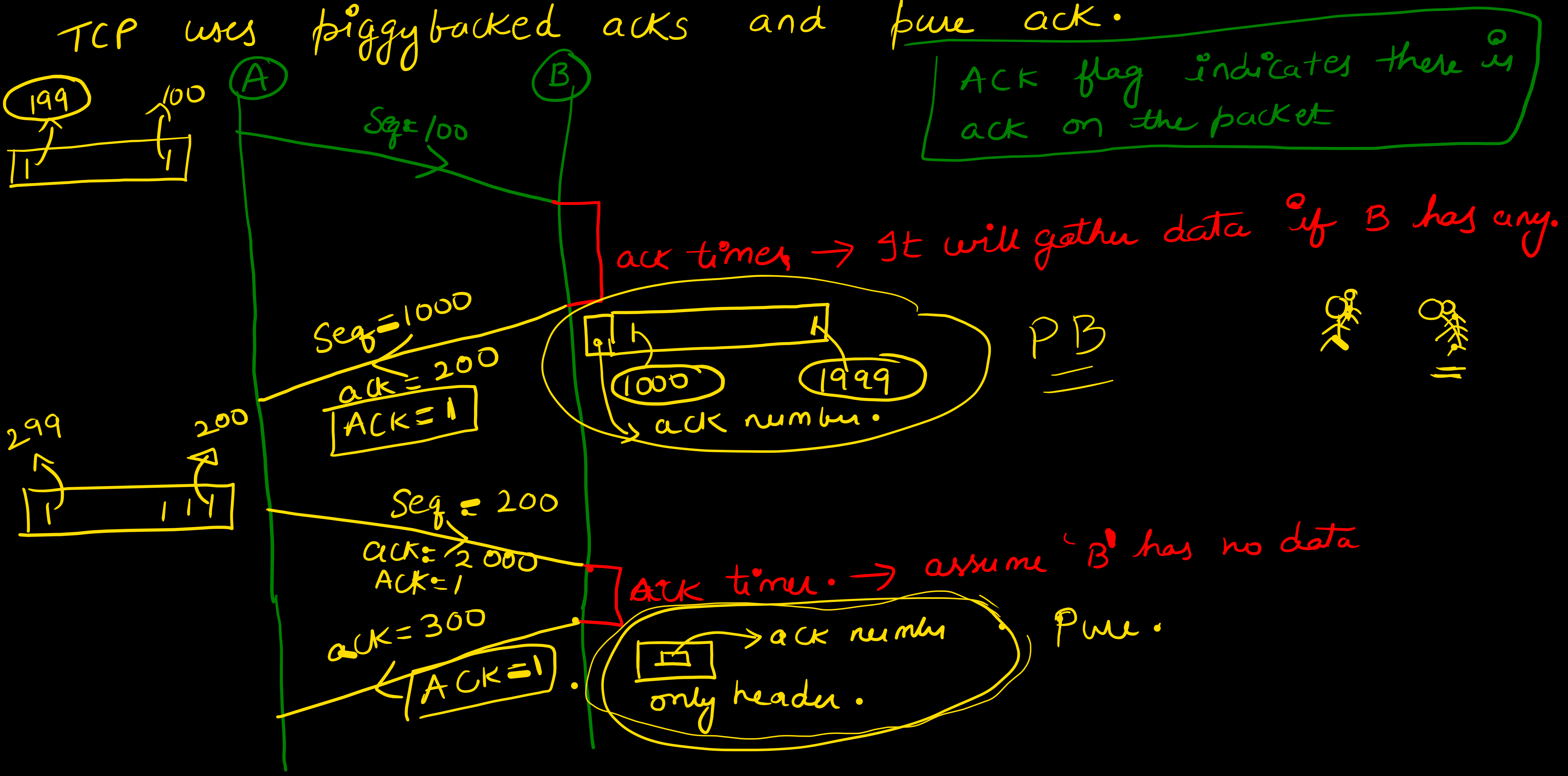
∴ Data = 1000 − 5×4 − 5×4 = 960 B.

Scaling

ack no = last byte seq number + 1

$= \boxed{1060}$

TCP uses piggybacked acks and pure ack.



TCP → Connection oriented protocol → Resources are reserved.

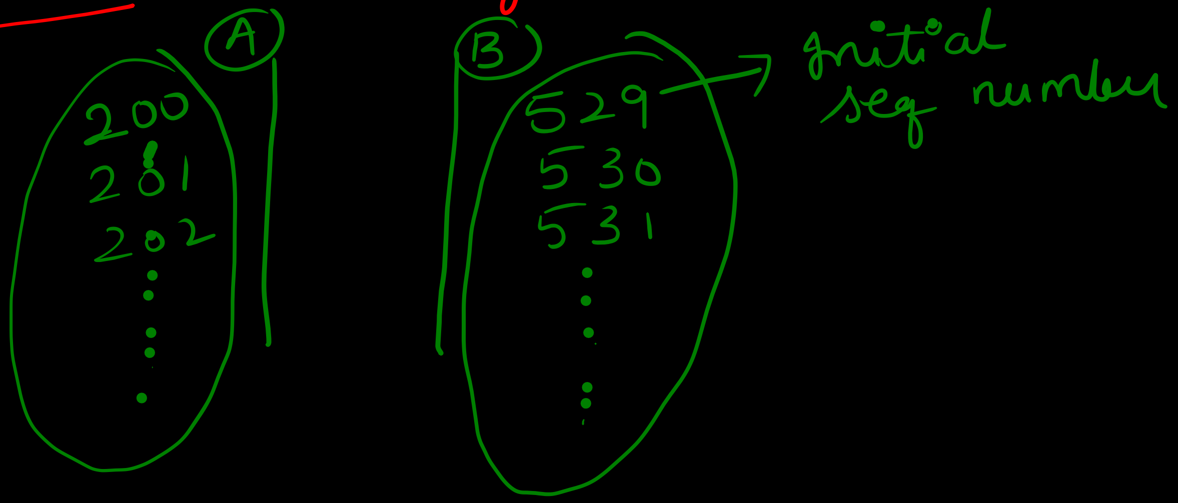
There are 3 phase

1) Connection establishment → we are saying we are sending data → be ready.

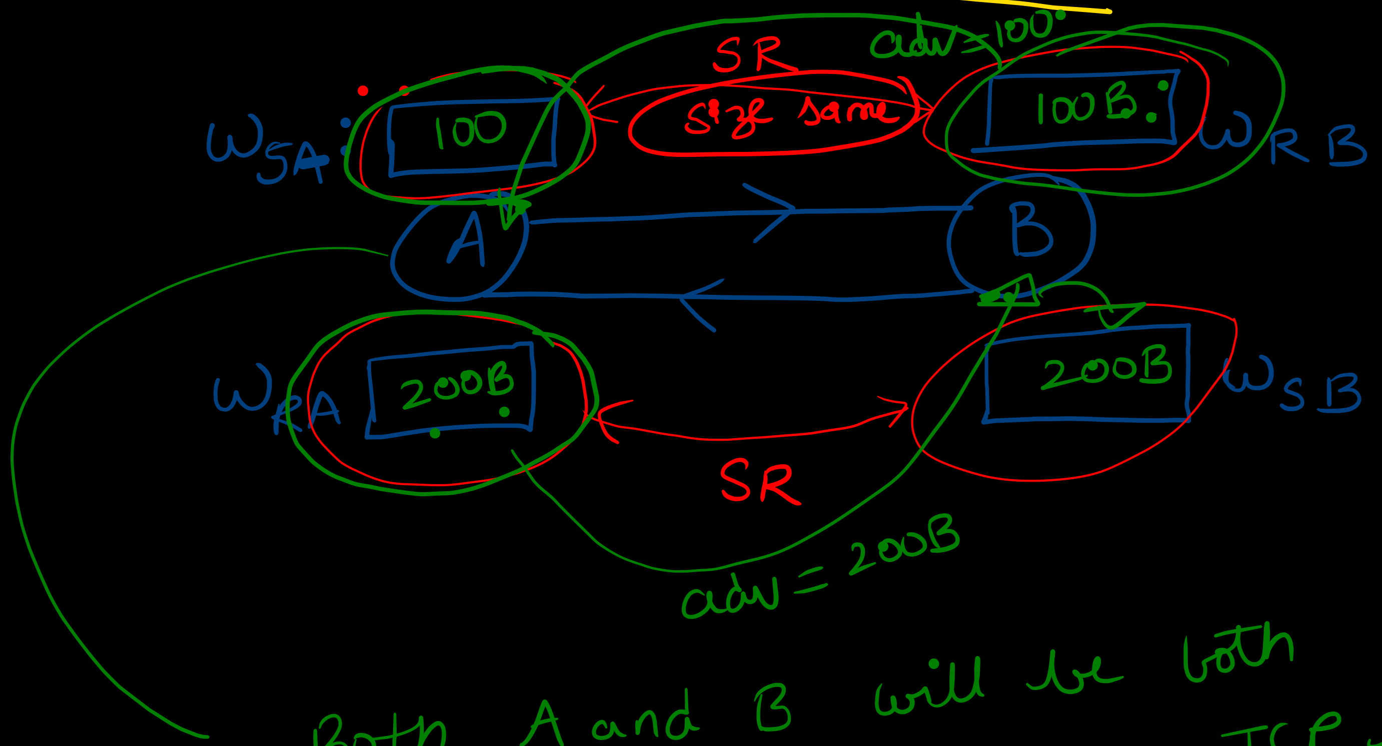
2) Data transfer →

3) Connection termination. → Transfer is over, so free up resources.

Revise: we always start with Random initial sequence numbers.



TCP is full duplex connection :

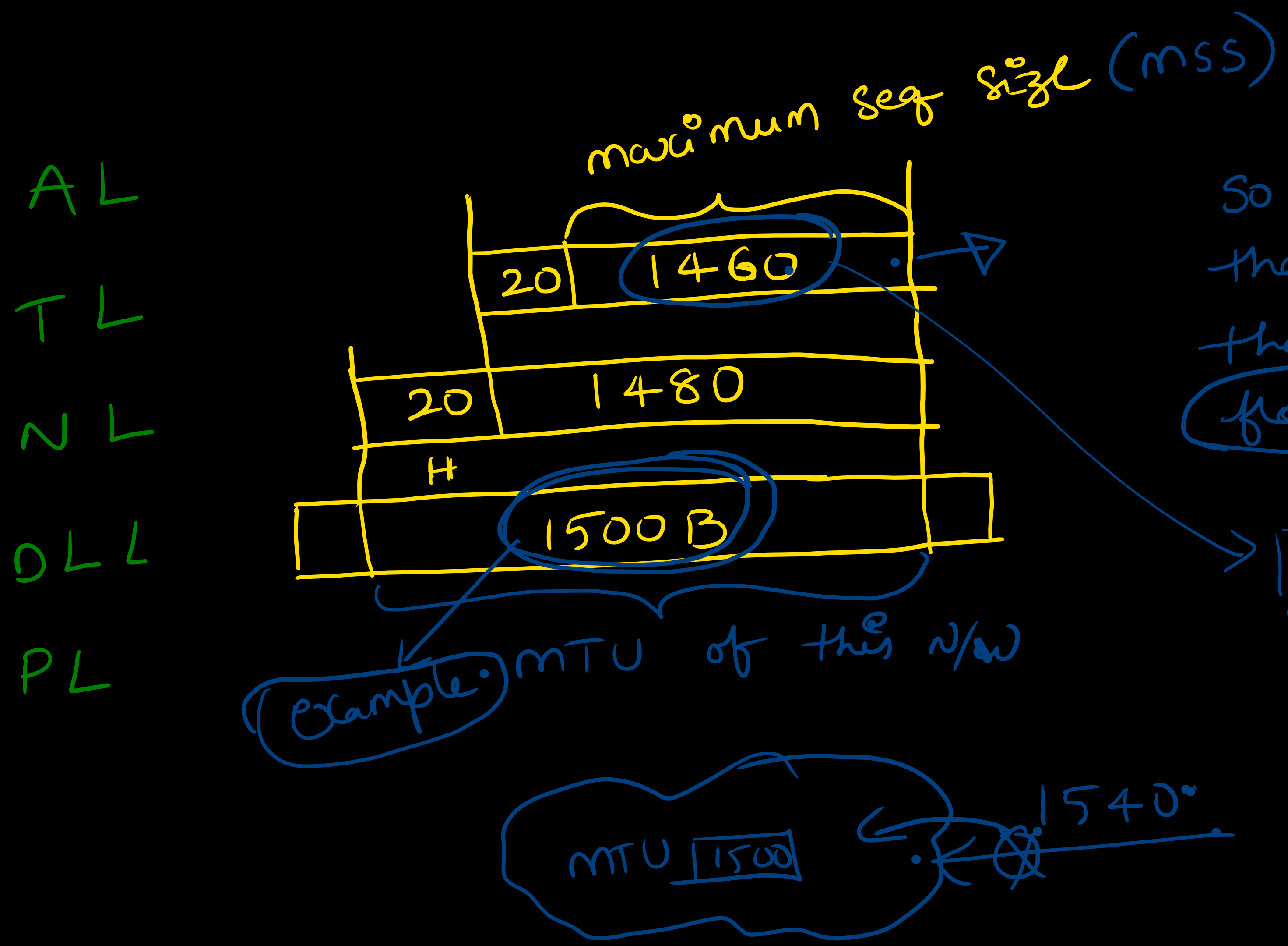


In SR → sender window = Rec window.

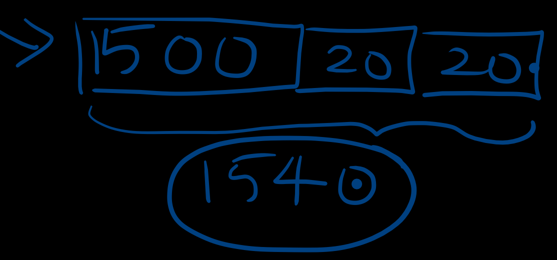
First Receiver window will be decided → and sent to the otherside.

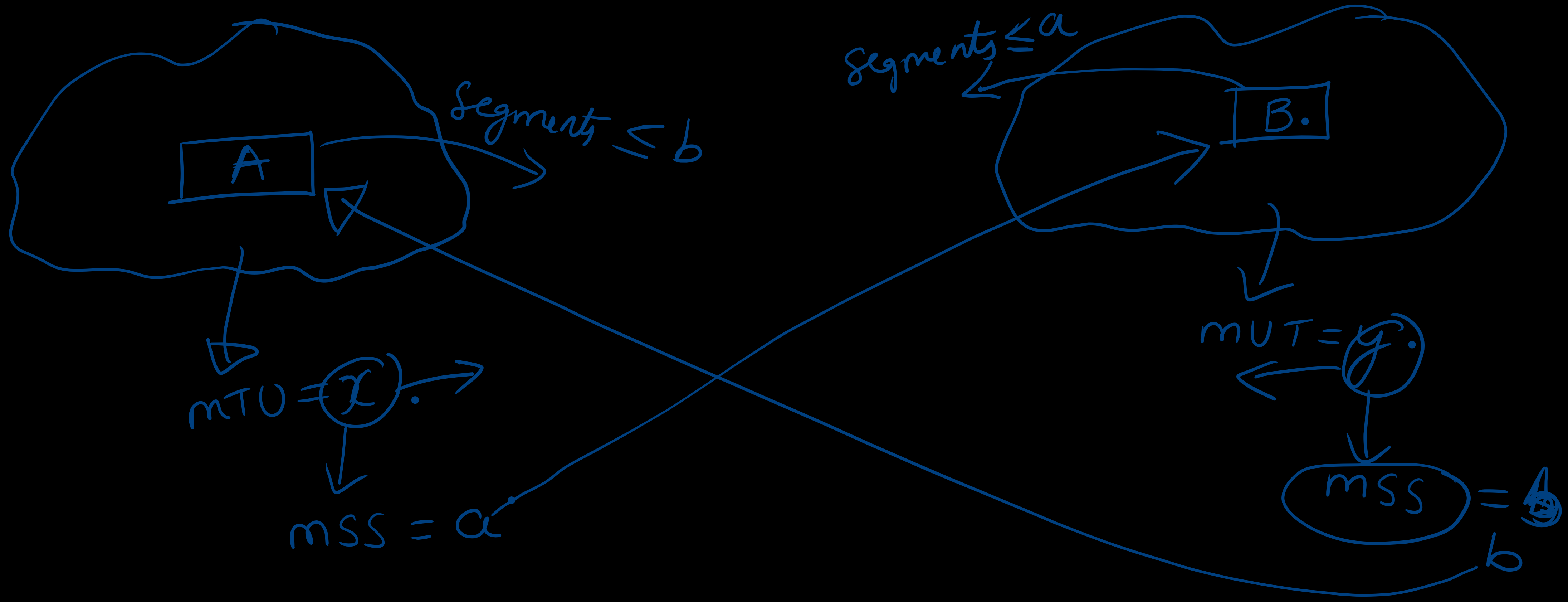
So sender will listen to receiver.

Both A and B will be both sender and rec. Because TCP is full duplex.



So if anyone sends more than 1460 B in one seg, then there will be fragmentation





A
segments ≤ b
MTU = x.
MSS = a
segments ≤ a
B.
MUT = y.
MSS = b

Connection establishment:

A

Random initial seq no = 521

W_{AR} 14600

B

Rand ini seq nu = 2000

Seq = 521

ACK = 0

SYN = 1

mss = 1460 B

Based on MTU etc

adv window = 14600

Request packet

W_{BR} 10000

Seq = 2000

SYN = 1, mss = 500 B

adv window = 10000.

ack = 522

ACK = 1

Reply

(iii)

→ 3 way handshake

ACK flag

Header

ack no

2001

Seq = 522, ack = 2001

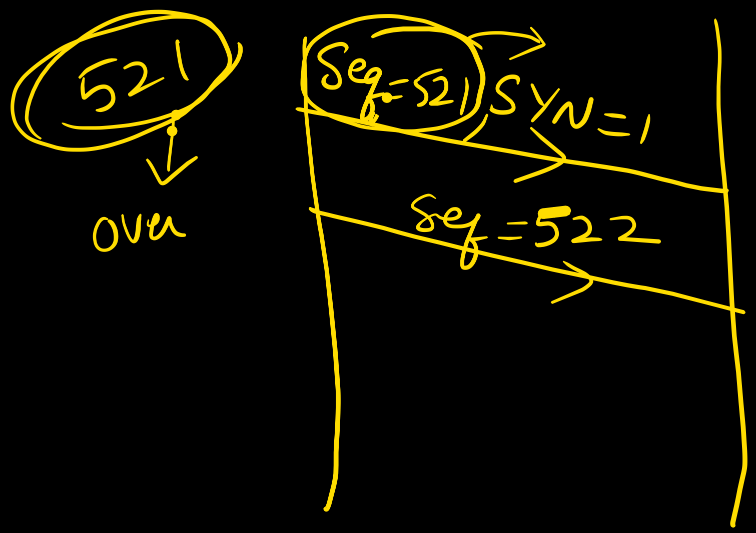
ACK = 1

ACK

adv win = every packet but I will not show you

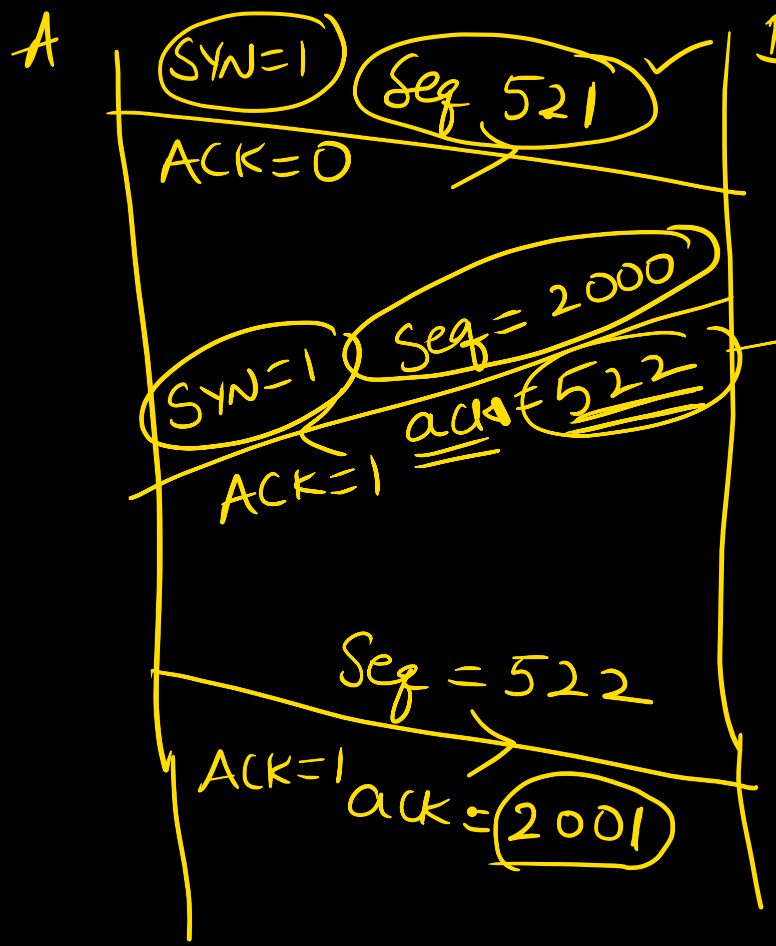
522

SYN → Synchronization flag, used to specify initial seq number.



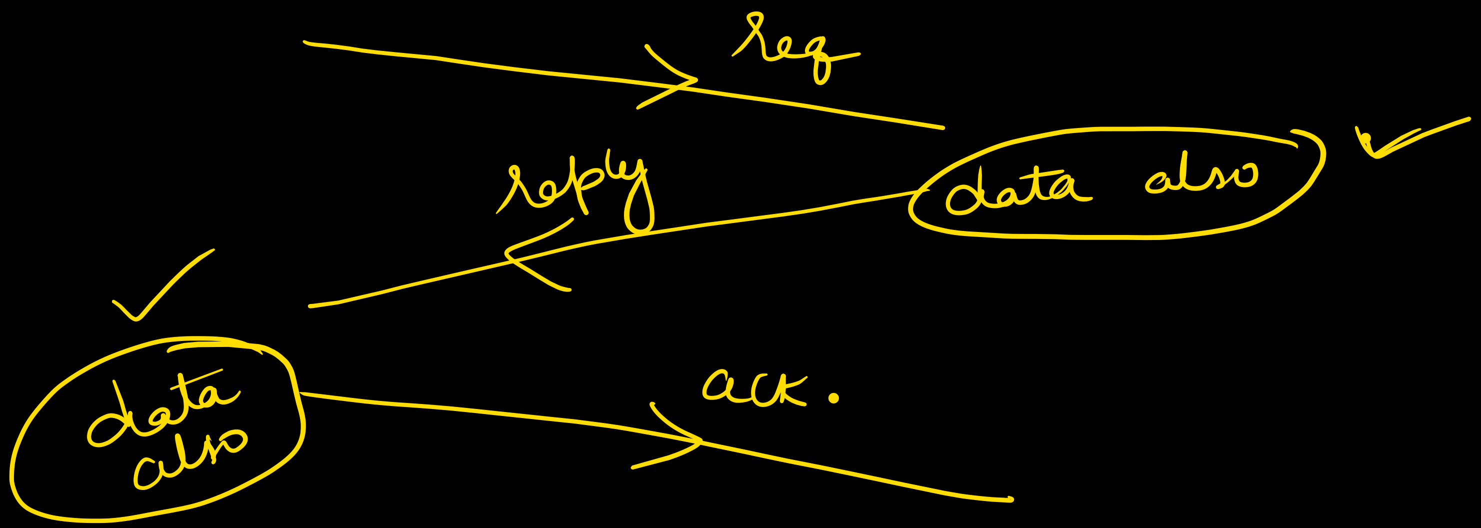
SYN packet will eat up one seq number.

RISN = ~~521~~ 522

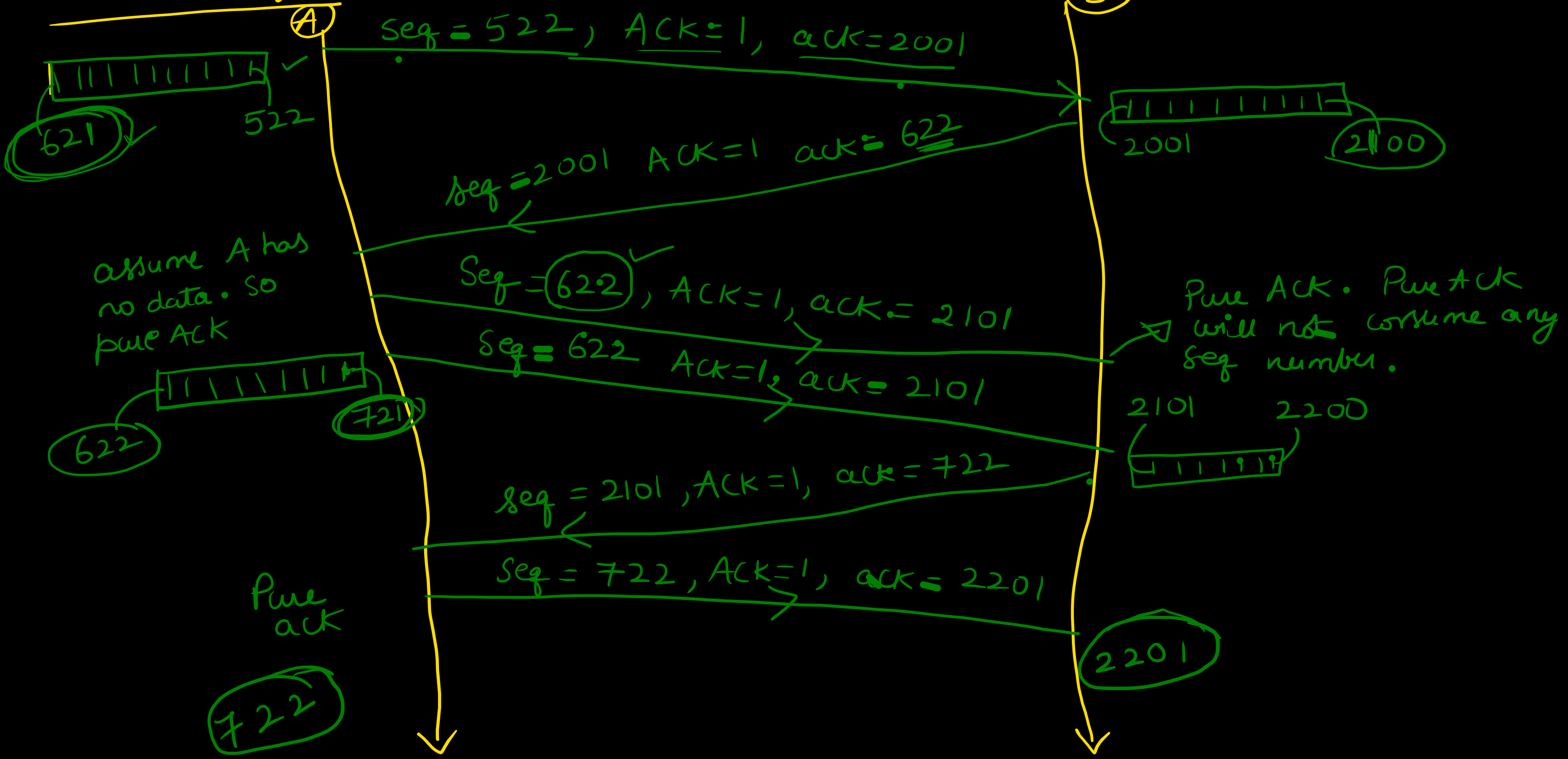


RISN = ~~2000~~

next expecting.


req
reply
data also
data also
ack.

Data transfer:

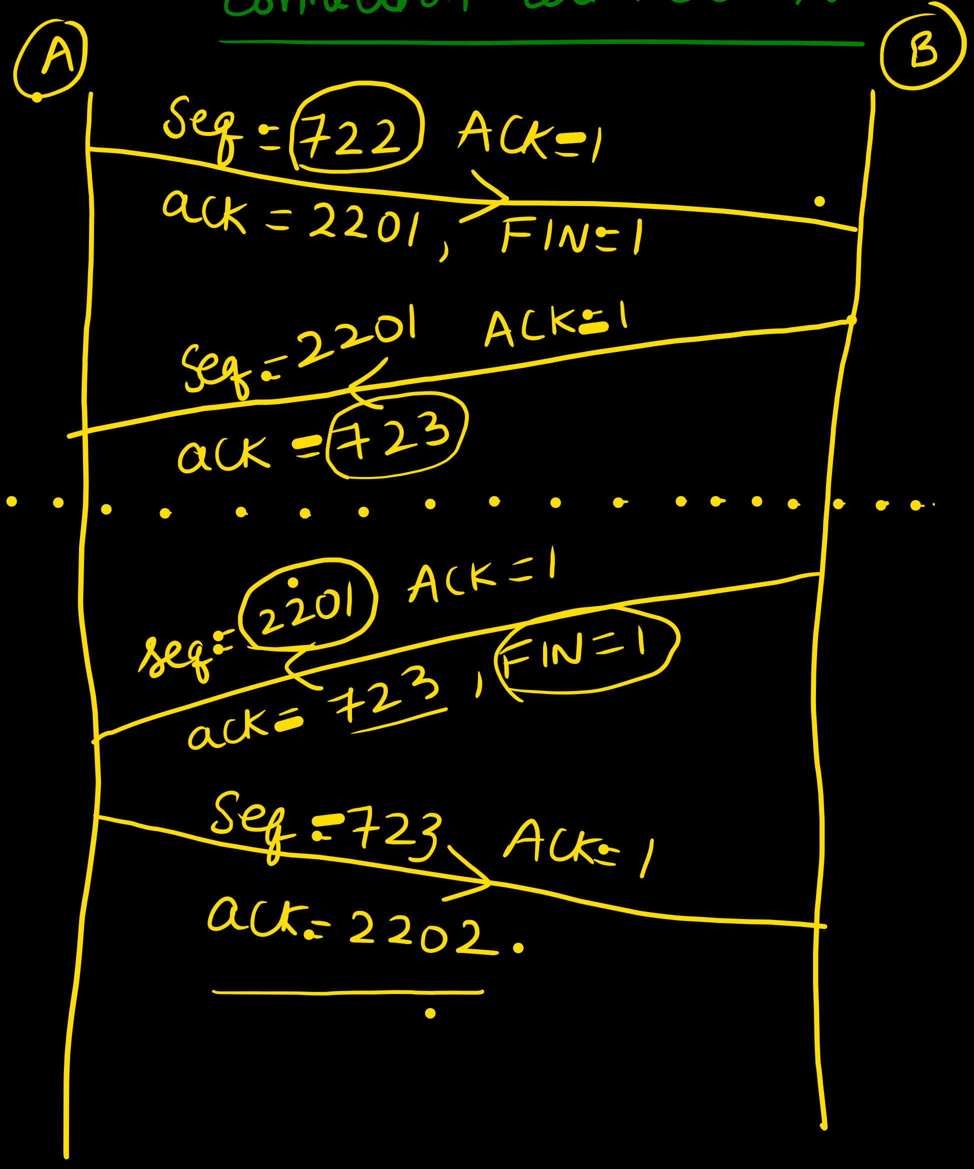


SYN	ACK	→ flags
1	0	→ I segment / request packet
1	1	→ II segment / Reply packet
0	1	→ Acknowledgment is present (PB or pure)
$\boxed{0}$	0	→ Not possible

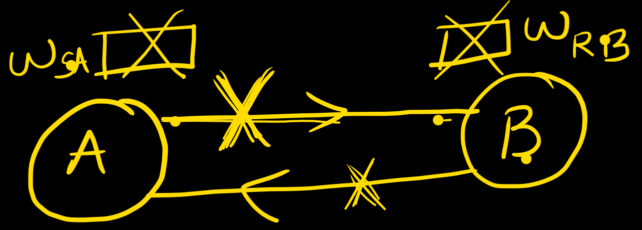
Every segment which is not first segment, will have ACK=1. Every segment except first will always carry acknowledgment.

Connection termination:

FIN → eats one seq number.



A → B is closed.



Data : A → B ✗

Data : B → A ✓

ACK : A → B ✓ → Pure ack.

ACK : B → A → PB ack