Part I Syllabus

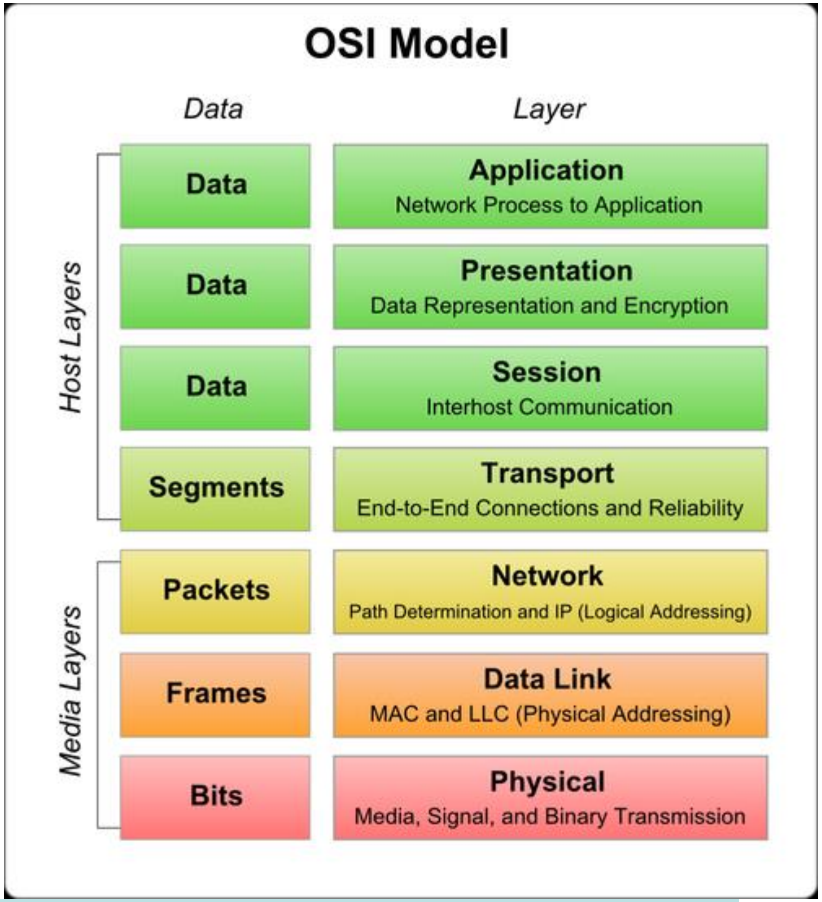
Lecture	Date	Subject
1	10/08/2016	Introduction
2	10/08/2016	Layered network architecture & Physical resilience
3	**17/08/2016**	**Data link layer – flow control**
4	17/08/2016	Data link layer – error control
5	24/08/2016	Data link layer – HDLC
6	24/08/2016	Local area network – introduction
7	31/08/2016	Local area network – MAC
8	31/08/2016	Local area network – Ethernet
9	07/09/2016	Local area network – WLAN
10	07/09/2016	Packet switch network - Introduction
11	14/09/2016	Packet switch network – queue analysis
12	14/09/2016	Review and examples

Drinking from Fire Hose

Lecture 3
Data Link Layer (DLL): Flow Control

Contents

- **Data Link Layer Fundamentals**
  - DLL Services
  - Framing mechanisms
  - Link configuration
- **Flow Control in DLL**
  - Main purpose of flow control
  - Stop-and-wait mechanism
  - Sliding window mechanism

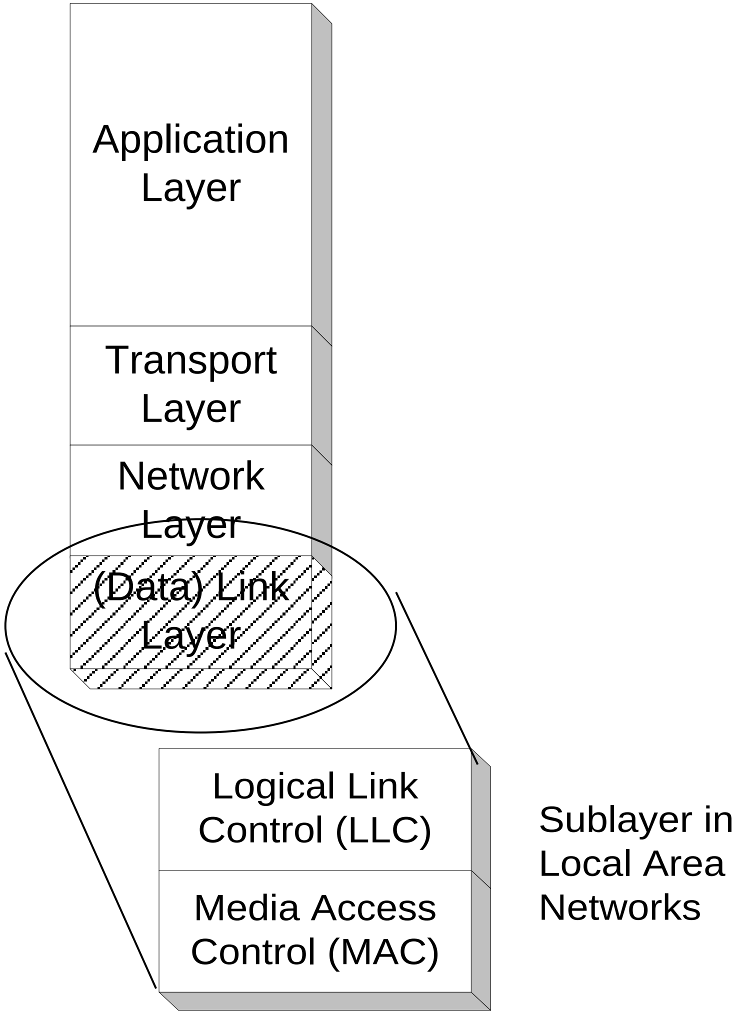


NANYANG TECHNOLOGICAL UNIVERSITY

Data Link Layer Fundamentals

Data Link Layer: Roles

- **DLL Services**
  - Framing: encapsulate each network-layer datagram within a link-layer frame before transmission over the link
  - Link Access: MAC protocol specifying the rules by which a frame is transmitted onto the link
  - Flow Control: control of data flow to ensure sender not overwhelm the receiver with data
  - Reliable Delivery: move each network-layer datagram across the link without error



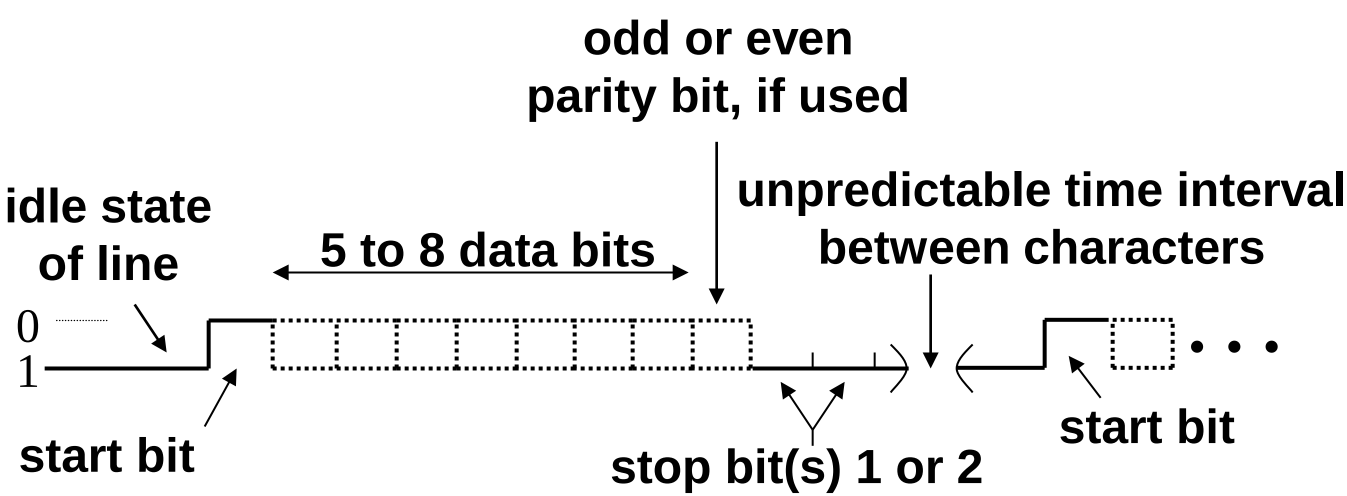
NANYANG TECHNOLOGICAL UNIVERSITY

Framing

- **Byte Oriented (Character Oriented):**
  - Information is framed into a fixed 8-bit basic unit.
  - Some of these basic units are used for signaling (protocol control).
  - Good solution when digital technology was in its primitive age (late 60s).
- **Bit Oriented (HDLC)**
  - A flag is used to frame the bits sent.
  - Header/Trailer are used to describe the content of a frame. Frames may be used for control.
  - Used by all modern protocols (eg HDLC, PPP, Ethernet, etc).

Byte-Oriented Async. Transmission

- **Pre-determined frame format**
  - Start/stop bit
  - Parity check bit
  - Data bits

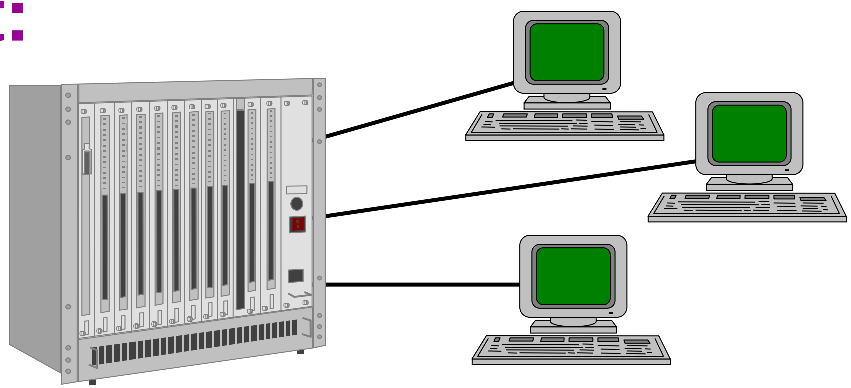


Link Configuration/Access

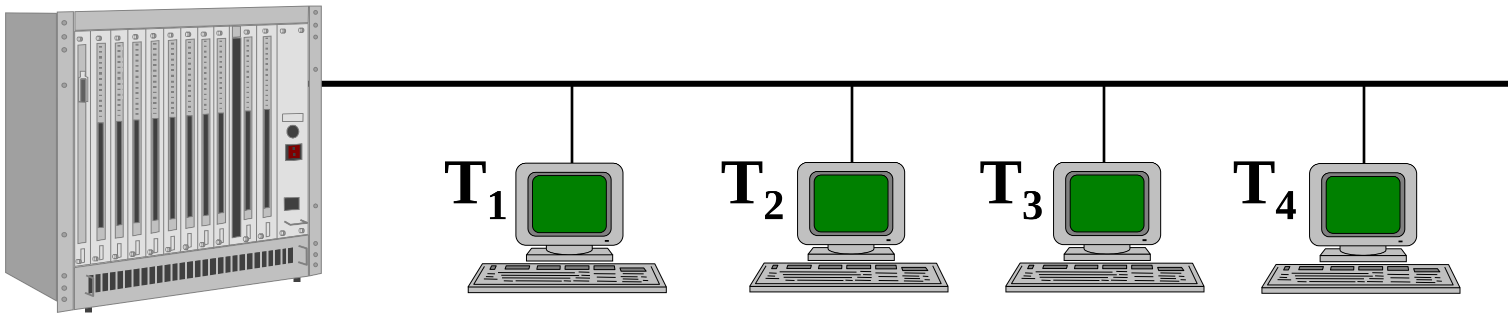
- **Objective: determine *who* gets to transmit at *when* on a link**
- **Topology: physical arrangement of stations**
  - Point-to-Point: pairs of hosts are directly connected
  - Broadcast: all stations share a single channel
- **Duplexity**
  - Half Duplex: Only one party may transmit at a time.
  - Full Duplex: Allows simultaneous transmission and reception between two parties (eg two logical half-duplex channels on a single physical channel).

Topology

Point-to-point:



Point-to-Multipoint (Broadcast):

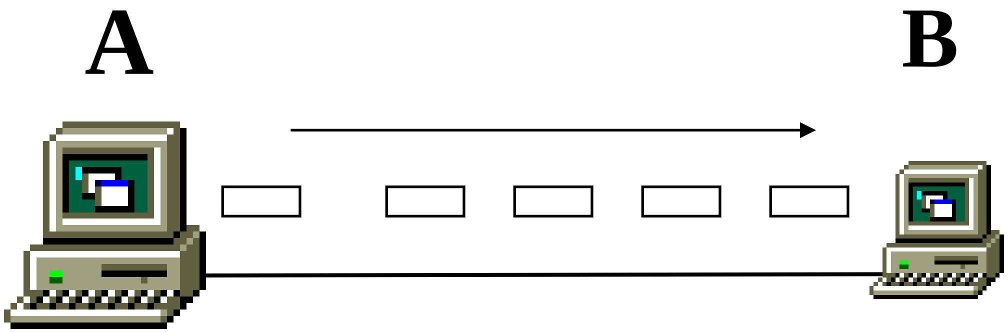


All terminals share the same medium controlled by the primary station (mainframe)

Flow Control

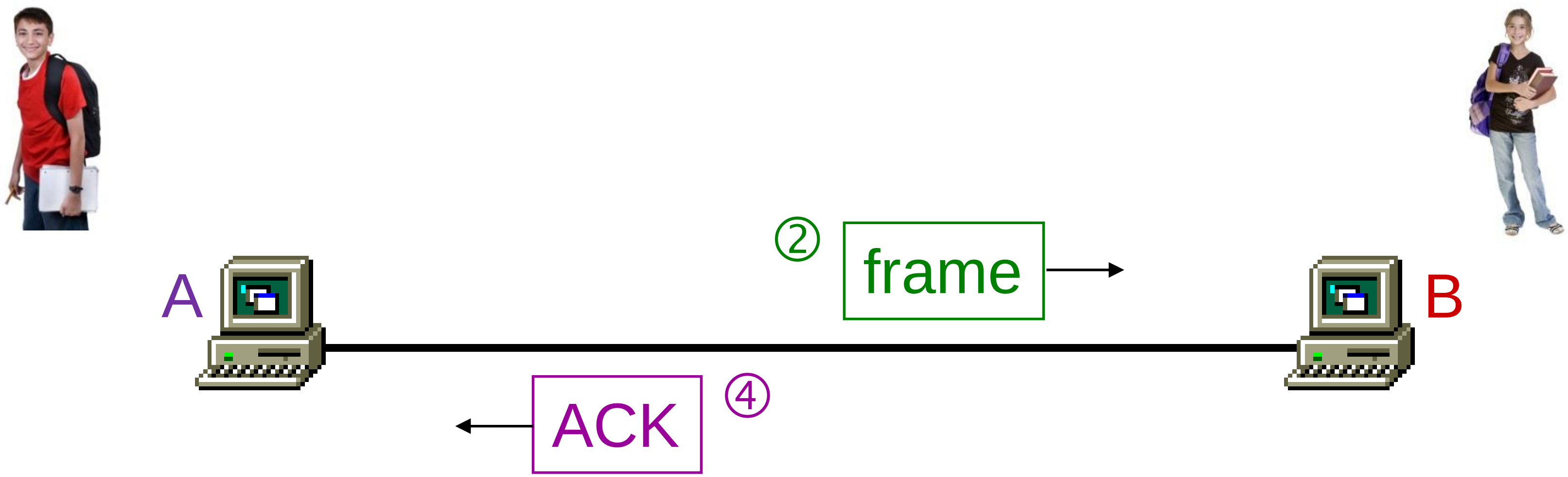
Functions and Mechanisms

- **Flow control**
  - Ensuring that a transmitting station does not overwhelm a receiving station with data, i.e., buffers at the receiver do not get overflow.
  - No frame error



- **Two Flow-Control Mechanisms**
  - Stop-and-Wait
  - Sliding Window

Stop-and-Wait Flow Control



<u>Operations:</u>

① *A* packs binary information into a frame

② *A* sends the frame to *B*

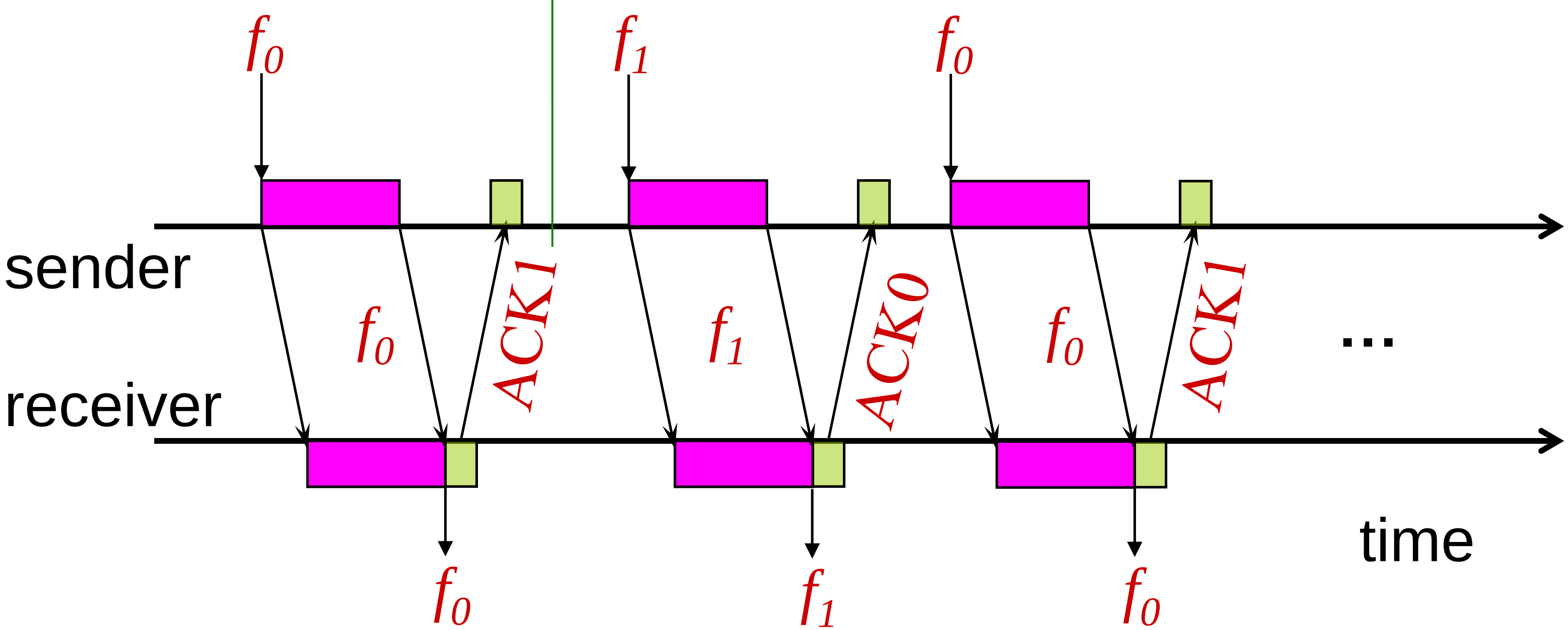
③ *A* waits for an ACK

④ When *B* has received the frame, *B* sends an ACK

⑤ When *A* has received the ACK, *A* repeats ①

Frame Flow in Stop-and-Wait

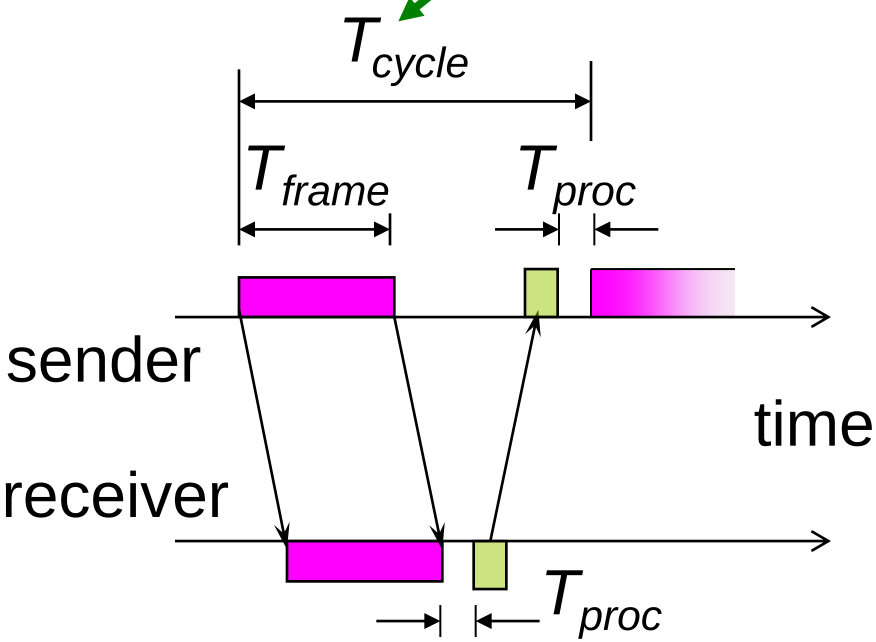
ACK1 means the receiver expects f_1, implying f_0 is received successfully



Flow-Control Performance: Throughput

$$\text{Throughput (U) (Link Utilization)} = \frac{\text{the time that the link carries useful information}}{\text{the total time}} = \frac{T_{frame}}{T_{cycle}}$$

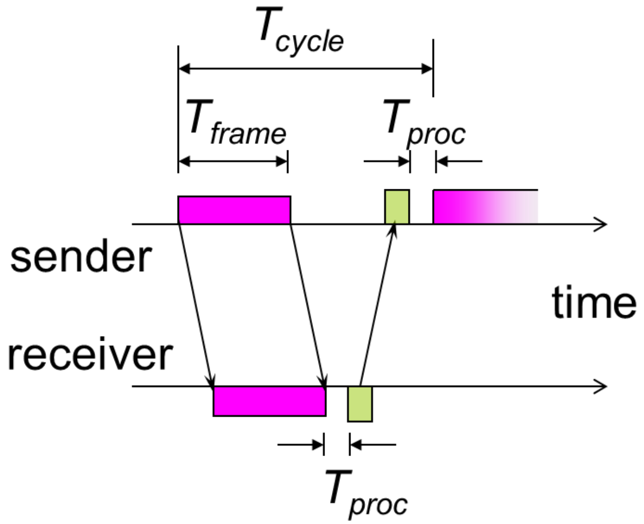
$$T_{cycle} = T_{frame} + T_{prop} + T_{proc} + T_{ack} + T_{prop} + T_{proc}$$



- T_{cycle}: Time needed to send a frame
- T_{proc}: Processing time
- T_{ack}: ACK Tx time
- T_{frame}: Frame Tx time
- T_{prop}: Signal Propagation delay

Throughput for Stop-and-Wait

- **Assumptions**
  - Input is saturated
  - No error
  - Ignoring T_{ack} & T_{proc}



We get:

$$T_{cycle} = T_{frame} + 2\,T_{prop}$$

Then:

$$U = T_{frame} / (T_{frame} + 2\,T_{prop})$$
$$= 1 / (1+2a)$$

Where:

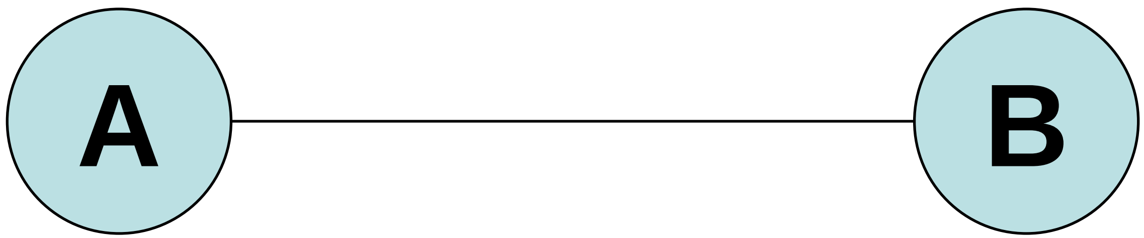
$$a = T_{prop} / T_{frame}$$

$$\boxed{U = \frac{1}{1+2a}}$$

Parameter *a* is also as Normalized Propagation Delay

Example

A communication link exists between two nodes A and B. The transmission rate on the link is 2.4 Mbps. The distance between A and B is 50 *km* and the signal velocity is 2×10^8 *m/s*. The frame length is 300 bytes. No frame error. Calculate the link unitization for the stop-&-wait flow control mechanism.



R= 2.4 Mbps, L=300 bytes =2400bits

H=50km, v = 2×10^8 *m/s*

$U=1/(1+2\mathbf{a}) \longrightarrow a = \mathbf{T_p/T_f} \longrightarrow T_p=H/V=5\times10^4/2\times10^8 = 250\ \mu s$

$U=1/(1+2*0.25) \longleftarrow a = \mathbf{0.25} \longleftarrow T_f=L/R=2400/2.4\times10^6 = 1000\ \mu s$

$= \mathbf{2/3}$

Stop-and-Wait: Disadvantages

- **If frame or ACK is lost, long waiting time is expected**
  - Using a TIMEOUT control in the sender
- **If the propagation time is long, the sender must wait a long time before it can perform the next transmission**
  - Use **Buffers** at the sender/receiver (sliding window operation)

$$\boxed{U = \frac{1}{1+2a}}$$

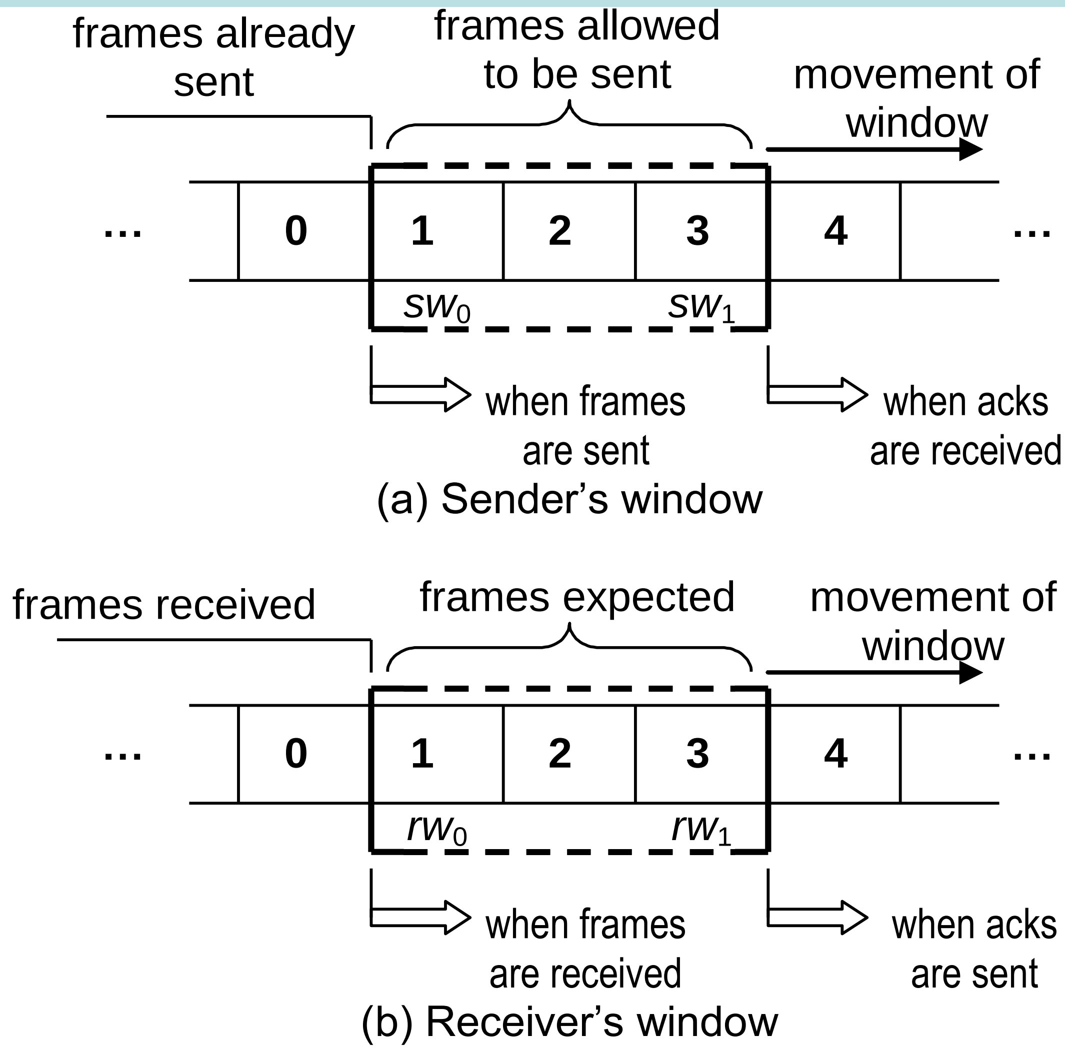
Sliding Window Flow Control

- **Allows multiple frames to be in transit.**
- **Sender and Receiver have buffer *N* long.**
- **Sender can send up to *N* frames without receiving ACKs.**
- **Each frame is labeled.**
- **ACK includes number of next expected frame.**
- **Sequence No. bounded by field size (*k* bits)**
  - **Frames are numbered modulo 2^k**
  - ***Sequence number $[0, 2^k-1]$***

NANYANG TECHNOLOGICAL UNIVERSITY

Sliding Window Operations

- **Sender**
  - Move lower bound when frames sent
  - Move upper bound when acks received
- **Receiver**
  - Move lower bound when frames received
  - Move upper bound when acks sent

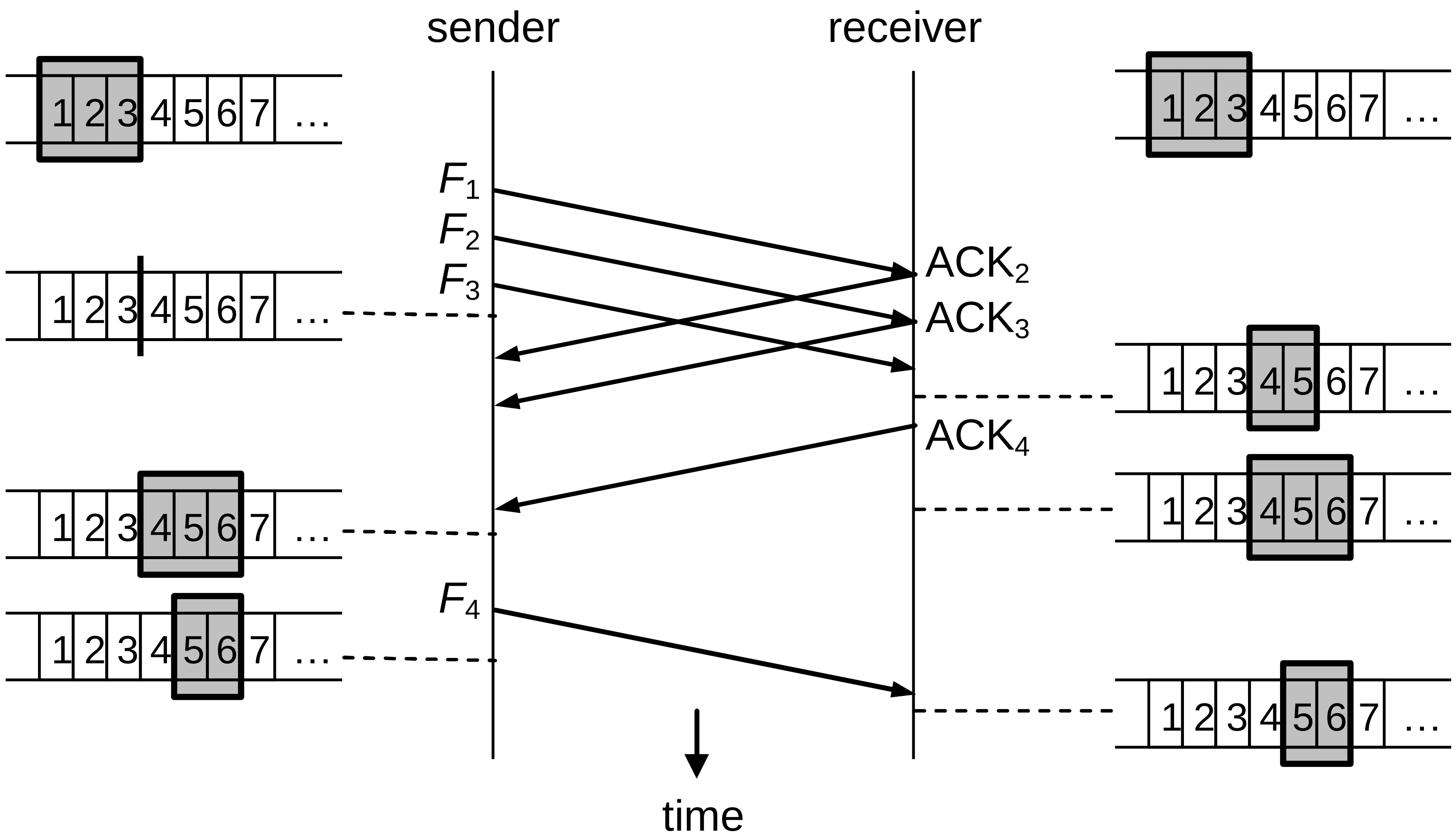


NANYANG TECHNOLOGICAL UNIVERSITY

Sliding Window Operations

- Sender maintains a window, containing frame numbers that can be transmitted.
- Sender window shrinks from trailing edge (left side) as frames are sent.
- Frames are buffered at the sender until acknowledged.
- Receiver maintains a window as well, its window shrinks from trailing edge as frames are received.
- Receiver’s window expands from the leading edge (right side) as ACKs are sent.
- Sender’s window expands from the leading edge as ACKs are received.

Sliding Window: Example



Sliding Window Algorithm

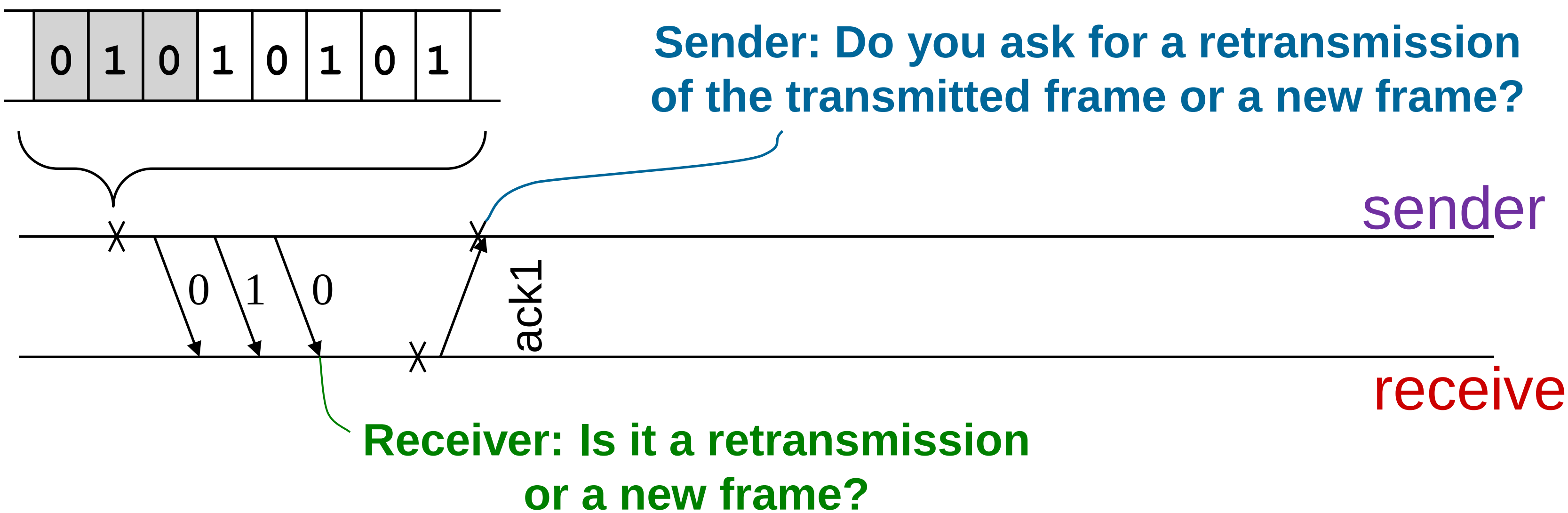
```
/* Protocol 4 (sliding window) is bidirectional. */

#define MAX_SEQ 1                         /* must be 1 for protocol 4 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"
void protocol4 (void)
{
  seq_nr next_frame_to_send;              /* 0 or 1 only */
  seq_nr frame_expected;                  /* 0 or 1 only */
  frame r, s;                             /* scratch variables */
  packet buffer;                          /* current packet being sent */
  event_type event;
  next_frame_to_send = 0;                 /* next frame on the outbound stream */
  frame_expected = 0;                     /* frame expected next */
  from_network_layer(&buffer);            /* fetch a packet from the network layer */
  s.info = buffer;                        /* prepare to send the initial frame */
  s.seq = next_frame_to_send;             /* insert sequence number into frame */
  s.ack = 1 – frame_expected;             /* piggybacked ack */
  to_physical_layer(&s);                  /* transmit the frame */
  start_timer(s.seq);                     /* start the timer running */
  while (true) {
     wait_for_event(&event);              /* frame_arrival, cksum_err, or timeout */
     if (event == frame_arrival) {        /* a frame has arrived undamaged. */
          from_physical_layer(&r);        /* go get it */
          if (r.seq == frame_expected) {  /* handle inbound frame stream. */
               to_network_layer(&r.info);    /* pass packet to network layer */
               inc(frame_expected);       /* invert seq number expected next */
          }
          if (r.ack == next_frame_to_send) {         /* handle outbound frame stream. */
               stop_timer(r.ack);         /* turn the timer off */
               from_network_layer(&buffer);          /* fetch new pkt from network layer */
               inc(next_frame_to_send);/* invert sender's sequence number */
          }
     }
     s.info = buffer;                     /* construct outbound frame */
     s.seq = next_frame_to_send;          /* insert sequence number into it */
     s.ack = 1 – frame_expected;          /* seq number of last received frame */
     to_physical_layer(&s);               /* transmit a frame */
     start_timer(s.seq);                  /* start the timer running */
  }
}
```

NANYANG TECHNOLOGICAL UNIVERSITY

Window Size Consideration

Say, Window Size, N = 3
with k=1 bit Sequence number

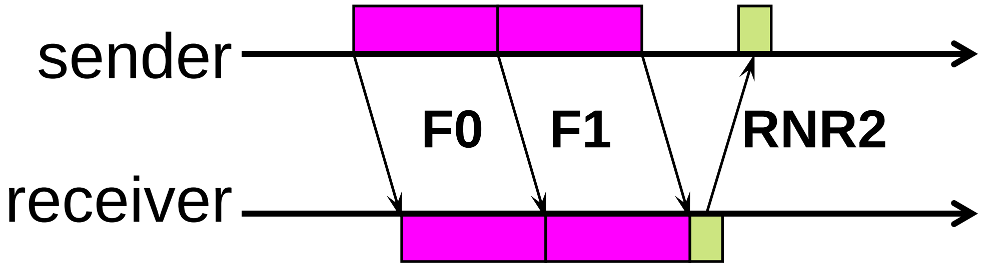
$N<=2^k$



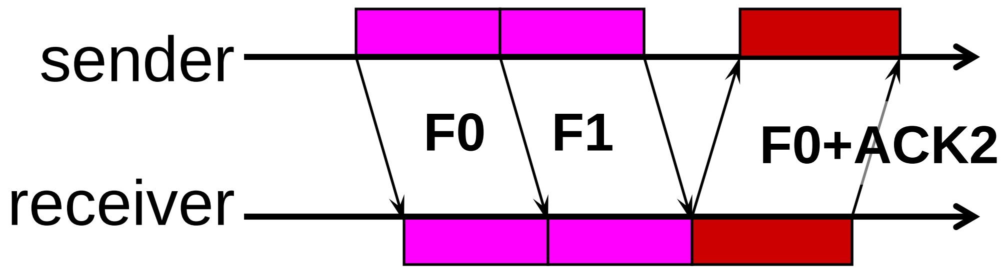
- Is the second $\boxed{0}$ a new frame or the retransmitted frame?
- Which frame is to be transmitted next after receiving ack1?

Sliding Window: Other Features

- **Receiver can acknowledge frames without permitting further transmission (by sending '*Receive Not Ready*', RNR frame). Receiver must send a normal acknowledgement to resume.**



- **ACK can be *piggybacked* on the data frames in the reverse direction.**

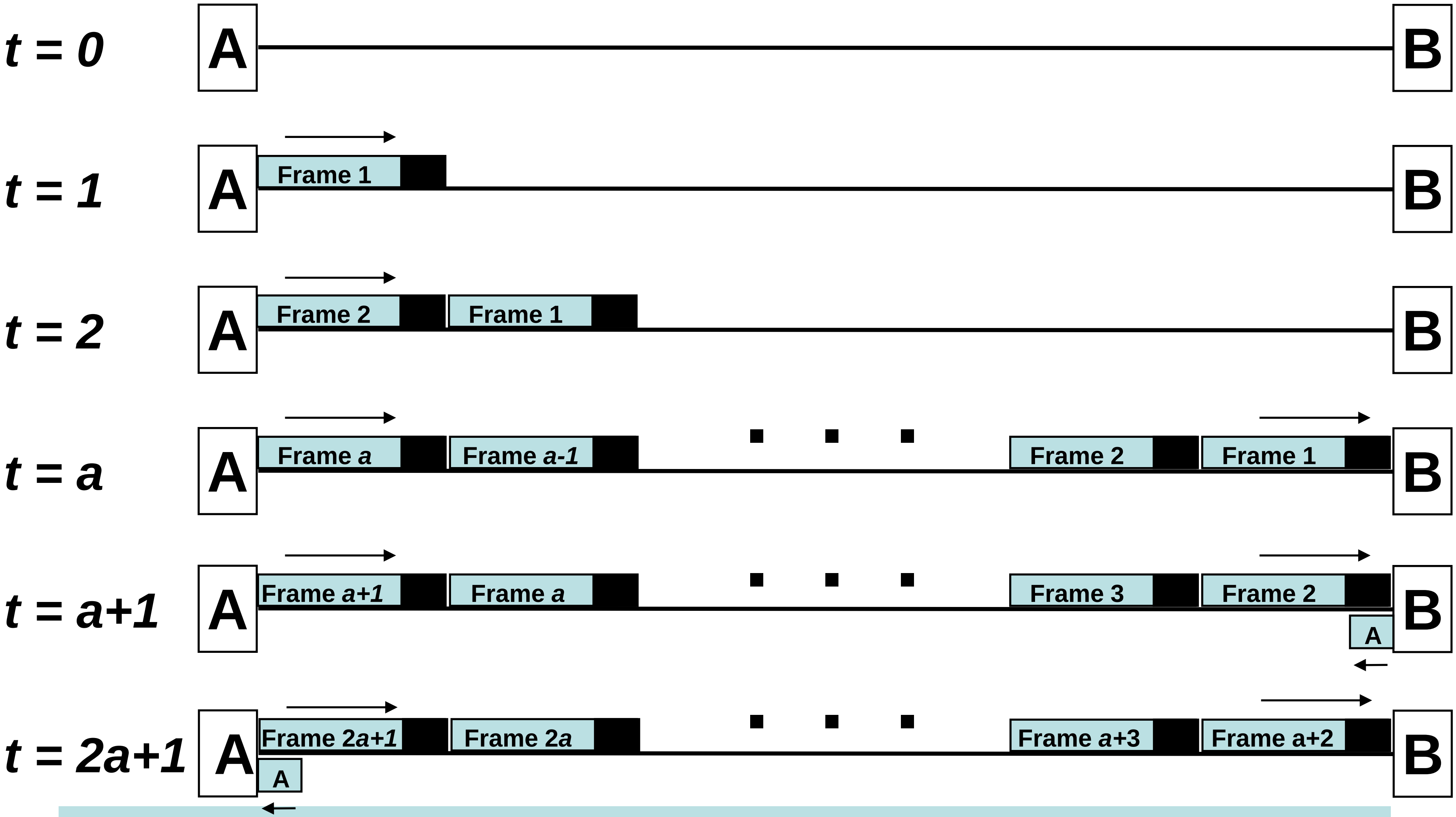


Sliding Window: Performance

- Performance depends upon (error-free operation):
  - Parameter ***a***, and
  - Window size, ***N.***
- Assumption: T_{ack} and T_{proc} are negligible.
- Frame transmission time = 1 (normalized to itself)
- Normalized propagation delay (one-way) = *a*
- **We need to consider two cases:**
  - $N \geq 2a + 1$: Station can transmit continuously without exhausting its window → $U = 1.0$
  - $N < 2a + 1$: Station's window is exhausted at $t = N$, and the station cannot send additional frames until $t = 2a + 1$, → $U = N/(1 + 2a)$

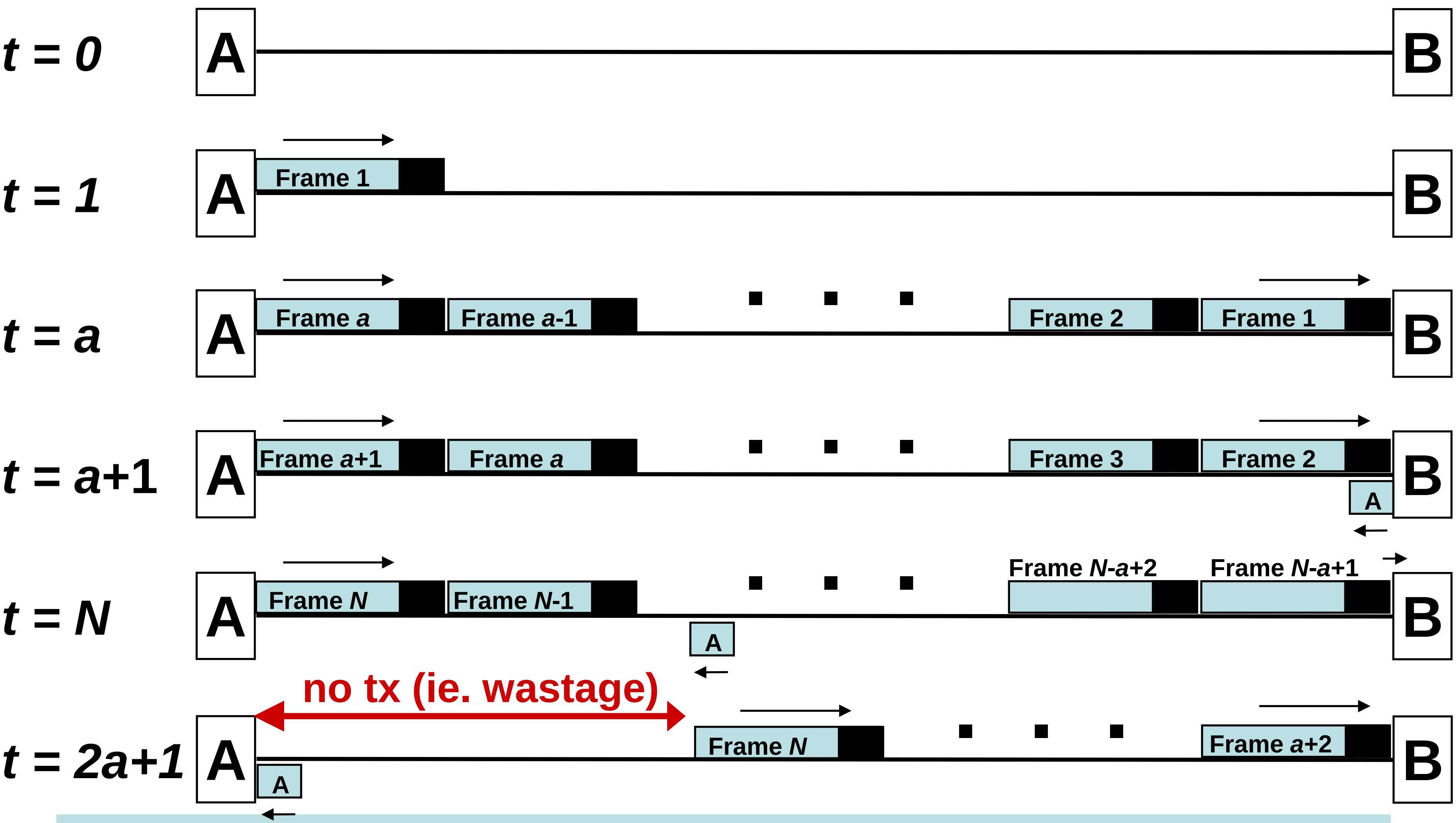
NANYANG TECHNOLOGICAL UNIVERSITY

Case I: $N \geq 2a + 1$ (U=1)



NANYANG TECHNOLOGICAL UNIVERSITY

Case II: $N < 2a + 1$ (U=N/2a)



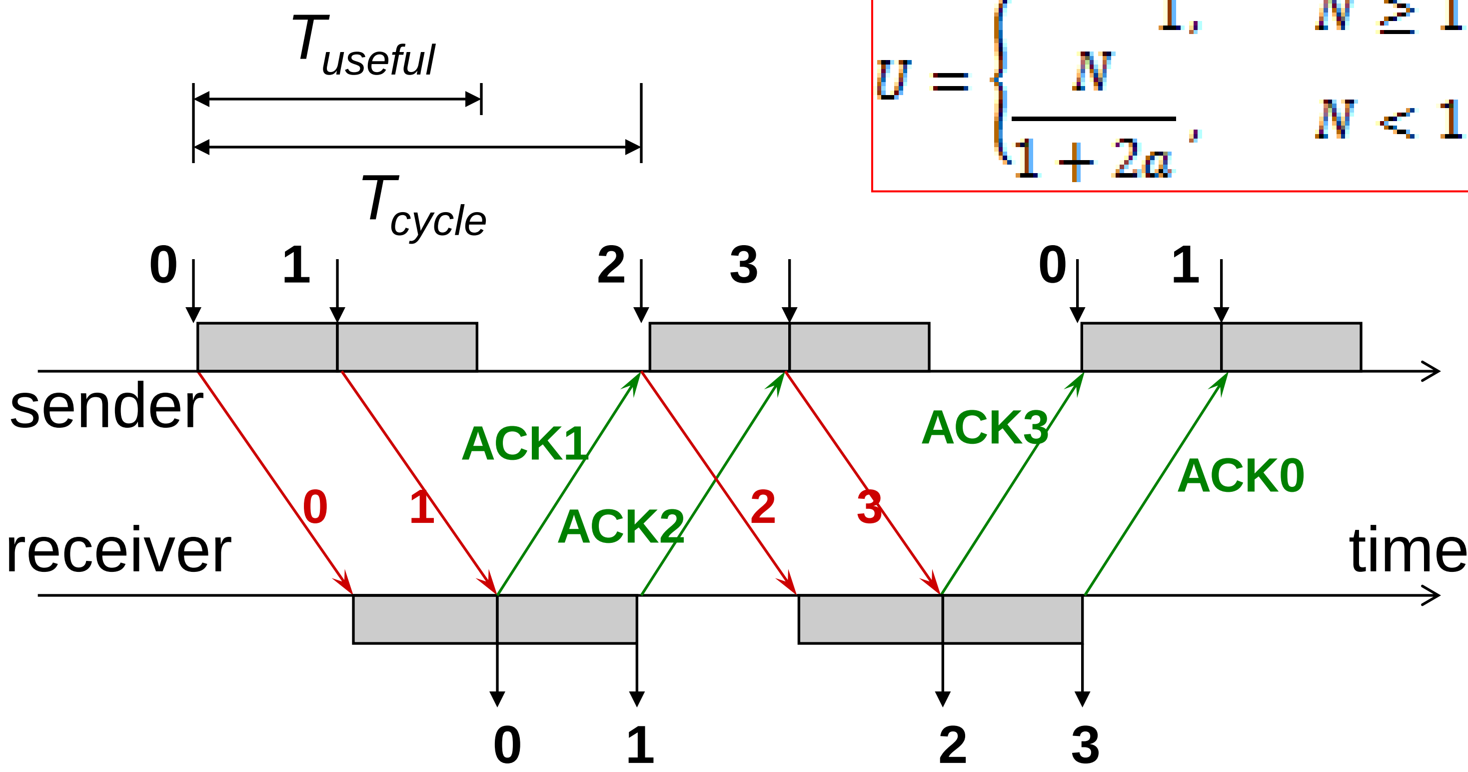
Sliding Window: Performance

Window Size = N

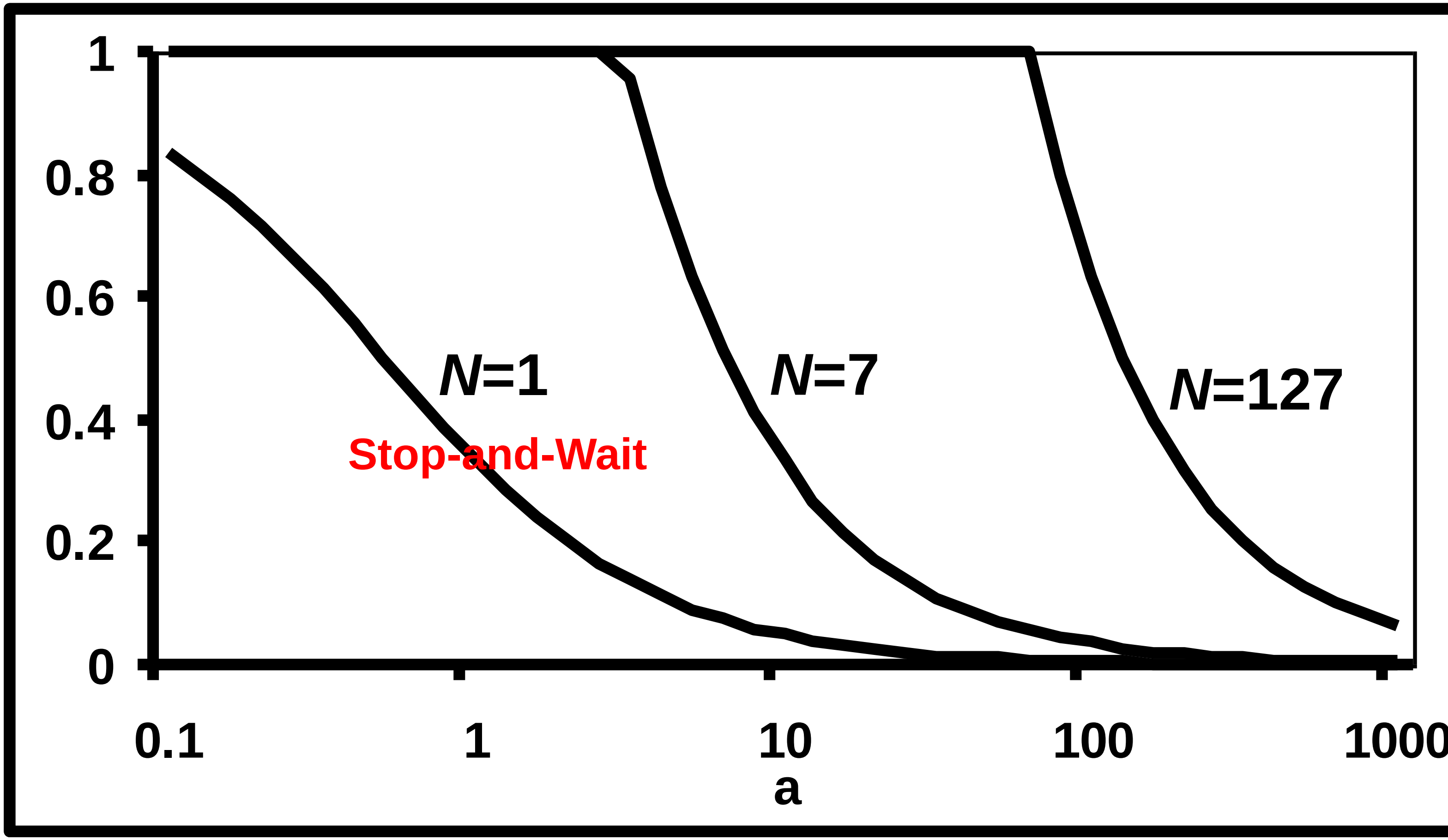
$T_{useful} = N * T_{frame}$

$T_{cycle} = T_{frame} + 2 * T_{prop}$

$$U = \begin{cases} 1, & N \geq 1 + 2a \\ \dfrac{N}{1+2a}, & N < 1 + 2a \end{cases}$$



Flow Control: Link Utilization



Link Utilization versus *a*

NANYANG TECHNOLOGICAL UNIVERSITY

Learning Objectives

- **Data Link Layer Fundamentals**
  - To understand its (four) main functions
- **Flow Control**
  - To understand its main purpose
  - Stop-and-Wait Flow-Control Mechanism
    - Operational protocol
    - Link utilization calculation
  - Sliding Window Flow-Control Mechanism
    - Operational protocol
    - Window size determination
    - Link utilization calculation (two cases)