1. 简述多路复用和多路分解。

答：从下到上，根据报文段中的端口号将传输层报文段中的数据分发到接受主机正确的套接字的工作称为多路分解。

从上到下，从源主机的不同套接字中收集数据块，并为某个数据块封装上首部信息从而生成报文段，然后将报文段传递到网络层的工作称为多路复用。

2. 计算下列两个 16 位字的校验和。

```
01111001  10111001          this binary number is 31161 decimal (base 10)

11101010  00001100          this binary number is 59916 decimal (base 10)
```

答：将这两个 16 位数相加结果如下：

$$\begin{array}{r} 0111100110111001 \\ \underline{1110101000001100} \\ 0110001111000101 \end{array}$$

结果有溢出，需要进行回卷，回卷得到的结果为 01100011 11000110，取反码结果为 10011100 00111001，所以校验和为 10011100 00111001。

1. (P24) Answer true or false to the following questions and briefly justify your answer:

   a. With the SR protocol, it is possible for the sender to receive an ACK for a packet that falls outside of its current window. (T) 因为存在超时的包延迟到达。

   b. With GBN, it is possible for the sender to receive an ACK for a packet that falls outside of its current window. (T) 因为存在超时的包延迟到达。

   c. The alternating-bit protocol is the same as the SR protocol with a sender and receiver window size of 1.

   (T) 当窗口大小为 1 时，相当于停等协议。

   d. The alternating-bit protocol is the same as the GBN protocol with a sender and receiver window size of 1 (T)

3. 可靠传输有哪些策略？

答：1. 校验和检测比特位错误
2. 接收者反馈 ACK，NAK
3. 超时重传
4. 数据报添加序号
5. 定时器 timer

（建议进行适当解释）

4. 回退 N 步（GBN）和选择重传（SR）有什么相同和不同点？

答：答: 相同点是都使用了流水线的滑动窗口机制，使用序号标明数据报，以及超时重传，

都采用了校验和。

不同点是 GBN 采用了累积确认，以及定时器只对最老的一个未被确认的数据报进行计时，如果接收者收到的序号不连续，就会丢弃该报文，不返回确认报文。一旦超时，就会对窗口中的 N 个包一起重发。而 SR 采用了单个确认的方式，每一个发送的包都会有一个定时器。接收端对单个数据包进行确认。超时只会重传未被确认的包。

5. UDP 和 TCP 报文头部有什么区别？为什么有这些区别？

答：答：UDP 报文头部长度是固定的八个字节，包括源端口，目的端口，报文长度以及校验和。而 TCP 报文头部长度是可变的，头部字段以 32 位为单位给出头部大小。包括源端口，目的端口，校验和，序列号（此次发送的数据在整个报文段中的起始字节数），确认号（下一个期望收到的字节，只有当 ack 标志为 1 时有效），6 位标志位，窗口大小，紧急指针，选项字段。产生这些区别的原因是 UDP 是一种不可靠的，无连接的传输，报文头需要的信息很少。而 TCP 是一种可靠的，面向连接的字节流服务，为了保证连接以及数据传输的可靠性，增加了很多必要的字段。

6. Consider the rdt2.2 protocol from the text (pages 209-212). The sender and receiver FSMs for the sender and receiver are shown below:

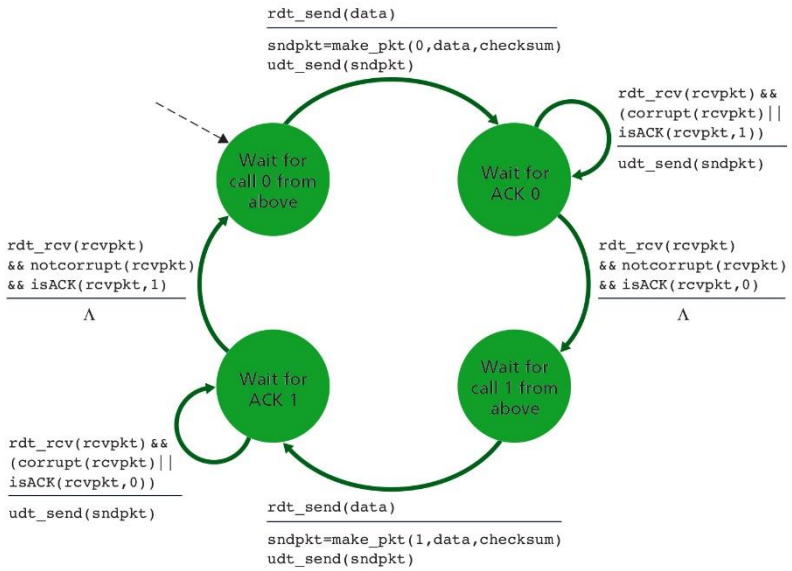


Figure 3.13 ♦ rdt2.2 sender

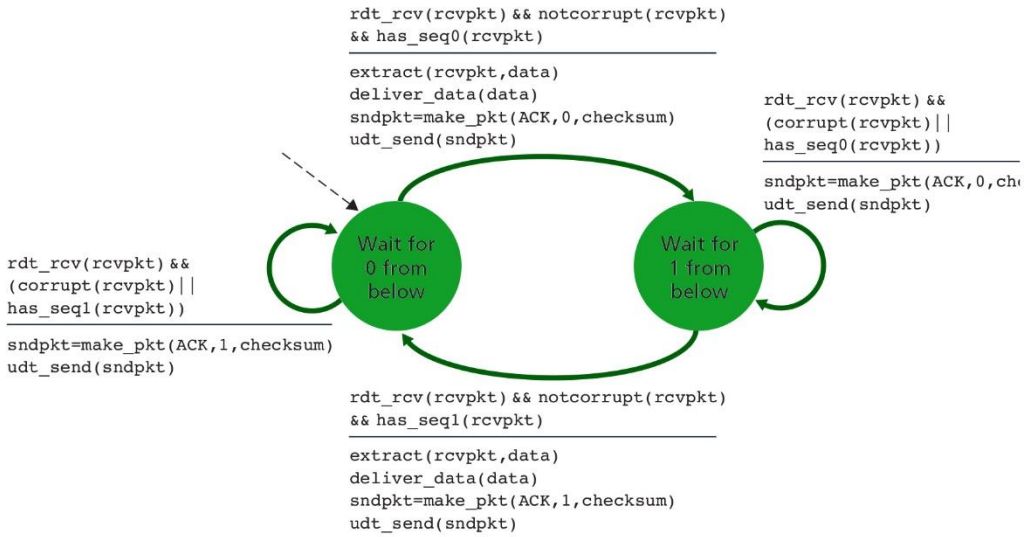
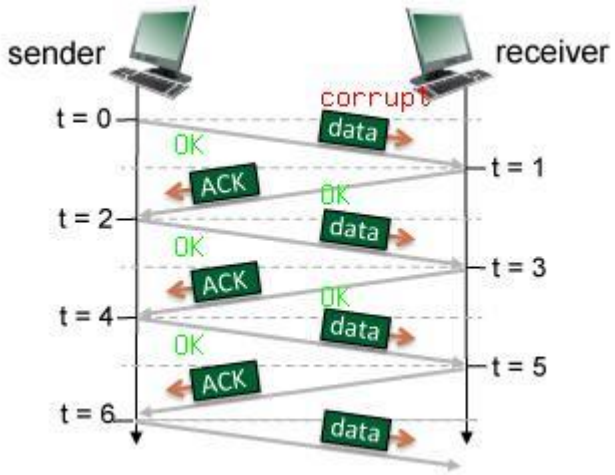


Figure 3.14 ♦ rdt2.2 receiver

Suppose that the channel connecting the sender and receiver can corrupt but not lose or reorder packets. Now consider the figure below, which shows four data packets and three corresponding ACKs being exchanged between an rdt 2.2 sender and receiver. The actual corruption or successful transmission/reception of a packet is indicated by the corrupt and OK labels, respectively, shown above the packets in the figure below.



Fill out the table below indicating *(i)* the state of the sender and the receiver just *after* the the transmission of a new packet in response to the received packet at time *t* , *(ii)* the sequence number associated with the data packet or the ACK number associated with the ACK packet sent at time *t*.

How many times is the payload of the received packet passed up to the higher layer at the receiver in the above example? At what times is the payload data passed up?

t	sender state	receiver state	packet type sent	seq. # or ACK # sent
0	Wait ACK0	Wait0 from below	data	0
1	Wait ACK0	Wait0 from below	ACK	1
2	Wait ACK0	Wait1 from below	data	0
3	Wait ACK0	Wait1 from below	ACK	0
4	Wait ACK1	Wait1 from below	data	1
5	Wait ACK1	Wait0 from below	ACK	1
6	Wait ACK0	Wait0 from below	data	0

2 packets were passed up to the higer layer at the receiver, at times t = 3, 5.

1.（P27）主机 A 和 B 经一条 TCP 连接通信，并且主机 B 已经收到了来自 A 的最长为 126 字节的所有字节。假定主机 A 随后向主机 B 发送两个紧接着的报文段，第一个和第二个报文段分别包含了 80 字节和 40 字节的数据。在第一个报文段中，序号是 127，源端口号是 302，目的地端口号是 80。无论何时主机 B 接受到来自主机 A 的报文段，它都会确认发送。

a. 在从主机 A 发往 B 的第二个报文段中，序号、源端口号和目的端口号各是什么？

答：序号：207，源端口：302，目的端口：80

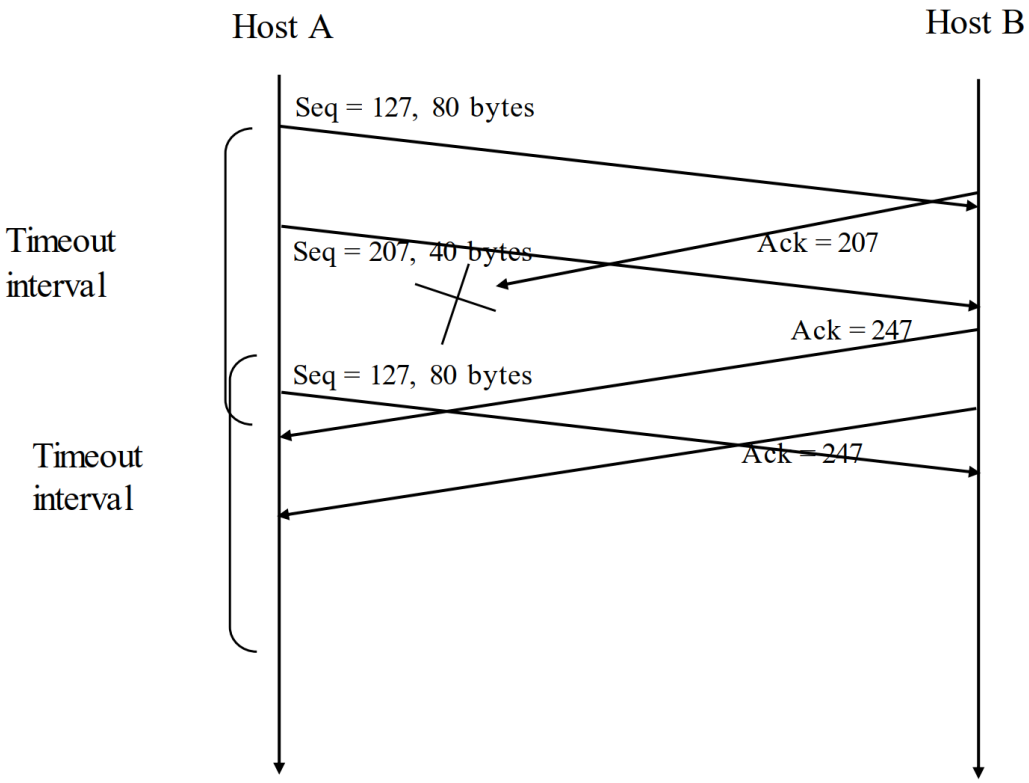
b. 如果第一个报文段在第二个报文段之前到达，在第一个到达报文段的确认中，确认号、源端口号和目的端口号各是什么？

答：确认号：207，源端口：80，目的端口：302

c. 如果第二个报文段在第一个报文段之前到达，在第一个到达报文段的确认中，确认号是什么？

答：确认号：127。

d. 假定由 A 发送的两个报文段按序到达 B。第一个确认丢失了而第二个确认在第一个超时间隔之后到达。画出时序图，显示这些报文段和发送的所有其他报文段和确认。（假设没有其他分组丢失。）对于图上每个报文段，标出序号和数据的字节数量；对于你增加的每个应答，标出确认号。



2. 流量控制和拥塞控制有什么区别？

答：流量控制是为了解决发送方和接收方速度不匹配而导致的数据丢失问题；拥塞控制是为了处理网络出现拥塞的现象，例如丢包，长延迟等，在出现拥塞现象时遏制发送方，以减少分组的丢失。

TCP 拥塞控制：

使用控制变量—拥塞窗口来表征网络的拥塞情况，对发送方能发送的速率进行限制。TCP 发送方通过丢包事件感知它在与目的地之间路径上出现的拥塞。TCP 拥塞算法包括三个主要部分：加性增、乘性减，慢启动，对超时时间的反应。算法对拥塞窗口的调整是以最大报文段长度字节为单位进行的。TCP 的拥塞控制包括三种状态：慢启动状态、拥塞避免状态和快速恢复状态。TCP 拥塞控制的实现 ：

1. TCP 拥塞控制采用端到端的控制，即没有网络层的反馈，而是通过当前网络状况进行动态控制(timeout, 冗余包的数目)

2. TCP 中有一个参数 cwnd 用来控制所有未收到回应的包的数目。

3. 当没有发生 timeout 事件以及没有发生 收到 3 次冗余包的时候，每一次收到 Ack, TCP 就逐渐增大 cwnd 的值，提升发送速率（先指数上升，到达某个阈值的时候再线性上升）。

4. 当发生 loss event 的时候(time out 或者连续收到 3 个冗余包) TCP 就减少 cwnd 的值来降低 TCP 的发送速率。

10. SOLUTION

1. The times where TCP is in slow start are: 1,2,26,27,28,29,34,35,36
2. The times where TCP is in congestion avoidance are: 3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,30,32,33,37,38,39,40
3. The times where TCP is in fast recovery are: 31
4. The times where TCP has a loss by timeout are: 1,25,33
5. The times where TCP has a loss by triple duplicate ACK are: 30
6. The times where the *ssthresh* changes are: 2,26,31,34

The complete solution is shown in the figure below:

For intervals of time when TCP is in slow start, the plotted value of *cwnd* is shown as a green square

For intervals of time when TCP is in congestion avoidance, the plotted value of *cwnd* is shown as a yellow square

For intervals of time when TCP is in fast reccovery, the plotted value of *cwnd* is shown as an orange square

The values for *ssthresh* are shown following a change as a red plus sign

A flight of packets experiencing a loss has the loss type (which determines the next value of *cwnd*) labeled above