Network Homework 2

Du Ruofei (5090109228)

3. UDP and TCP use 1s complement for their checksums. Suppose you have the following three 8-bit bytes: 01010011, 01010100, and 01110100. What is the 1s complement of the sum of these 8-bit bytes? Show all work. Why it that UDP takes the 1s is complement of the sum; that is, why not just use the sum? With the 1s complement scheme, how does the receiver detect errors? Is it possible that a 1-bit error will go undetected? How about a 2-bit error?

Solution:

(1)Compute the sum:

$$01010011 + 01010100 + 01110100 = 00010001$$

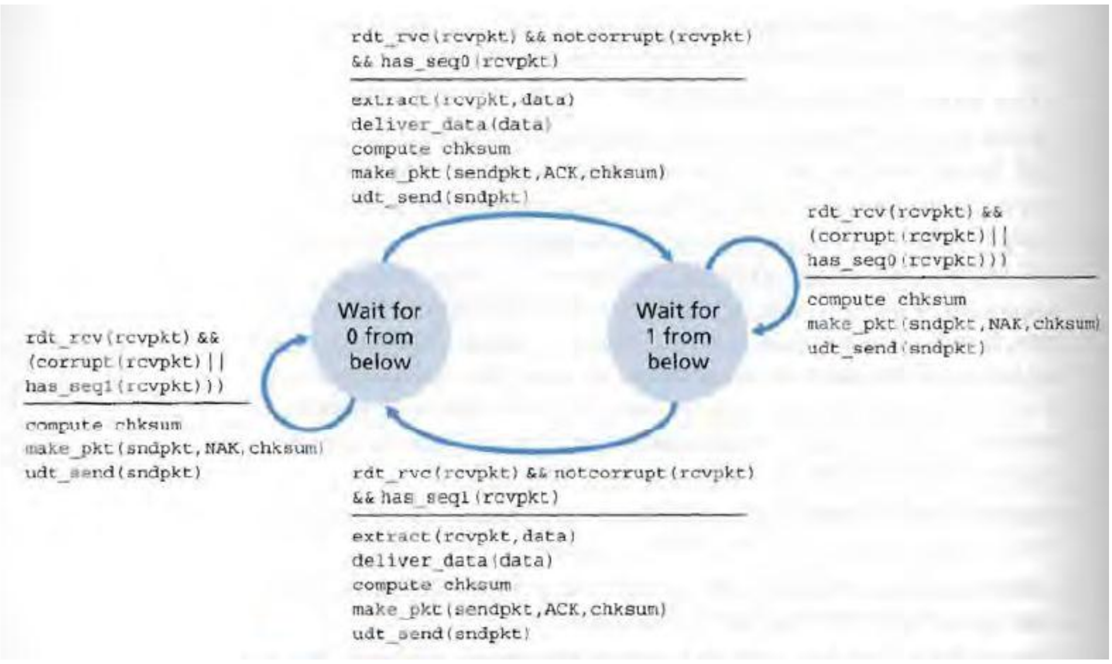
Compute the 1s complement:

$$(00010001)_{\text{complement}} = 11101110$$

(2)To detect errors, the receiver should add the three 8-bit bytes together with the checksum. If the sum contains a zero, there must be an error.

(3)All 1-bit error will be detected while 2-bit error can go undetected. For instance, if two bits on the same position of two different words change at the same time, the checksum remains.

6. Consider our motivation for correcting protocol rdt 2.1. Show that the receiver, shown in Figure 3.57, when operating with the sender shown in Figure 3.11, can lead the sender and receiver to enter into a deadlock state, where each is waiting for an event that will never occur?

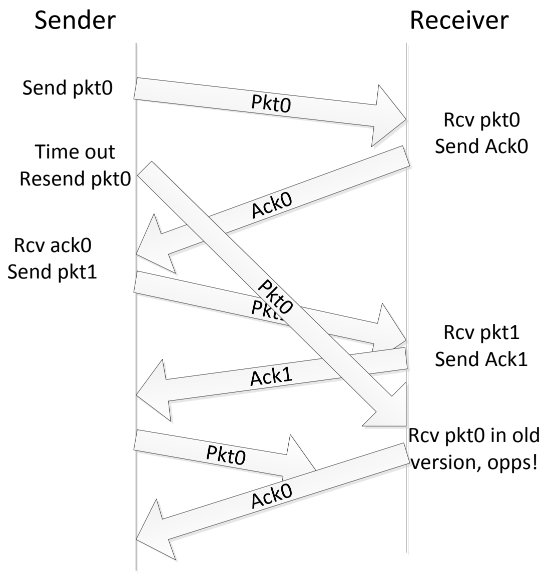


Solution:

The deadlock state would come as follows: Assume that the sender is in the state “Wait for 0 from below” and the receiver is in the state “Wait for 1 from below”. The sender sends a packet with sequence number 1, and transitions to the state “Wait for ACK or NAK 1”. Assume now the receiver receives the packet thus coming to the state “Wait for 0 from below” and sending a ACK to the sender. Nevertheless, if the ACK is corrupted, the sender would resend the packet with sequence number 1. However, the receiver would refuse to receive sequence 0 packet and send a NAK. Here comes a deadlock state.

12 Consider the rdt 3.0 protocol. Draw a diagram showing that if the network connection between the sender and receiver can reorder messages (that is, that two messages propagating in the medium between the sender and receiver can be reordered), then the alternating-bit protocol will not work correctly (make sure you clearly identify the sense in which it will not work correctly). Your diagram should have the sender on the left and the receiver on the right, with the time axis running down the page, showing data (D) and acknowledgment (A) message exchange. Make sure you indicate the sequence number associated with any data or acknowledgment segment.

Solution:



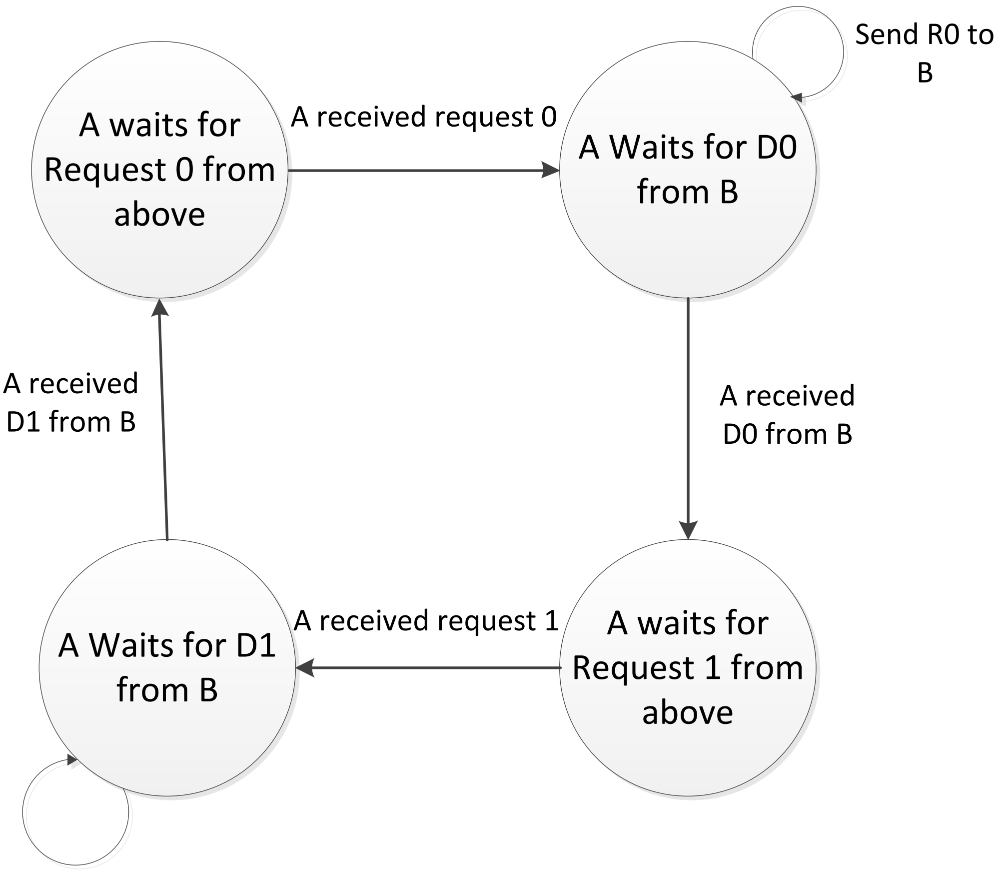
19 Suppose we have two network entities, A and B. B has a supply of data messages that will be sent to A according to the following conventions. When A gets a request from the layer above to get the next data (D) message from B, A must send a request (R) message to B on the A-to-B channel. Only when B receives an R message can it send a data (D) message back to A on the B-to-A channel. A should deli ver exactly one copy of each D message to the layer above. R messages can be lost (but not corrupted) in the A-to-B channel; D messages, once sent, are always delivered correctly. The delay along both channels is unknown and variable.

Design (give an FSM description of) a protocol that incorporates the appropriate mechanisms to compensate for the loss-prone A-to-B channel and implements message passing to the layer above at entity A, as discussed above. Use only those mechanisms that are absolutely necessary.

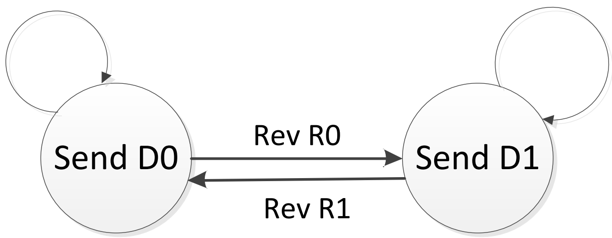
Solution:

We can ensure that D messages can always be delivered correctly but R message cannot be. Thus A must be designed to retransmit the message after the time out. To avoid duplicated R

message, we can use a 1-bit sequence number for the protocol.
For A:



For B:



21 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.
b. With GBN, it is possible for the sender to receive an ACK for a packet that falls outside of its current window.
c. The alternating-bit protocol is the same as the SR protocol with a sender and receiver window size of 1.
d. The alternating-bit protocol is the same as the GBN protocol with a sender and receiver window size of 1.

Solution:

a) True. Suppose the sender has a window size of 2 and sends packet 1,2 at 0ms. At 10ms, the receiver acknowledges packet 1,2. At 20ms, the sender times out and resends packet 1,2, At 30ms, the receiver receives packet 1,2 again and re-acknowledges 1,2. At 40ms the sender receives acknowledge of 1,2 and move the windows towards 3,4. At 60ms the sender receive the re-acknowledge of 1,2 again, but 1,2 are out of its current window.

b) True. The same argument as a)

c) True. When window size is 1, there won't be out-of-order packets within one window, a cumulative ACK is just an ordinary ACK in this situation.

d) True. The same argument as c)

27 Consider the TCP procedure for estimating RTT. Suppose that $\alpha = 0.1$. Let $\mathrm{SampleRTT_1}$ be the most recent sample RTT. Let $\mathrm{SampleRTT_2}$ be the next most recent sample RTT, and so on.

a. For a given TCP connection, suppose four acknowledgments have been returned with corresponding sample RTTs $\mathrm{SampleRTT_4}$, $\mathrm{SampleRTT_3}$, $\mathrm{SampleRTT_2}$ and $\mathrm{SarnpleRTT_1}$. Express $\mathrm{EstimatedRTT}$ in terms of the four sample RTTs.

b. Generalize your formula for n sample RTTs.

c. For the formula in part (b) let n approach infinity. Comment on why this averaging procedure is called an exponential moving average.

Solution:

a) Denote $\mathrm{EstimatedRTT_k}$ for the estimate RTT after the $\mathrm{k_{th}}$ sample.
Then

$$\mathrm{EstimatedRTT_1} = SampleRTT_1$$
$$\mathrm{EstimatedRTT_2} = \alpha SampleRTT_1 + (1-\alpha)SampleRTT_2$$
$$\mathrm{EstimatedRTT_3} = \alpha SampleRTT_1 + (1-\alpha)[\alpha SampleRTT_2 + (1-\alpha)SampleRTT_3]$$
$$\begin{aligned}\mathrm{EstimatedRTT_4} &\\ &= \alpha SampleRTT_1 + (1-\alpha)\alpha SampleRTT_2 \\ &+ (1-\alpha)^2\alpha SampleRTT_3 + (1-\alpha)^3\alpha SampleRTT_3\end{aligned}$$

b) By induction, we arrive at

$$\mathrm{EstimatedRTT_n} = \alpha\sum_{i=1}^{n-1}(1-\alpha)^i\, SampleRTT_i + (1-\alpha)^n SampleRTT_n$$

Take $\alpha = 0.1$

$$\mathrm{EstimatedRTT_n} = \frac{1}{9}\sum_{i=1}^{n-1}0.9^i\, SampleRTT_i + 0.1^n SampleRTT_n$$

c) As n goes to infinity,

$$\mathrm{EstimatedRTT_\infty} = \frac{1}{9}\sum_{i=1}^{\infty}0.9^i\, SampleRTT_i$$

The coefficient 0.9^i decease exponentially as time goes by which lead the method to be called an exponential moving average.