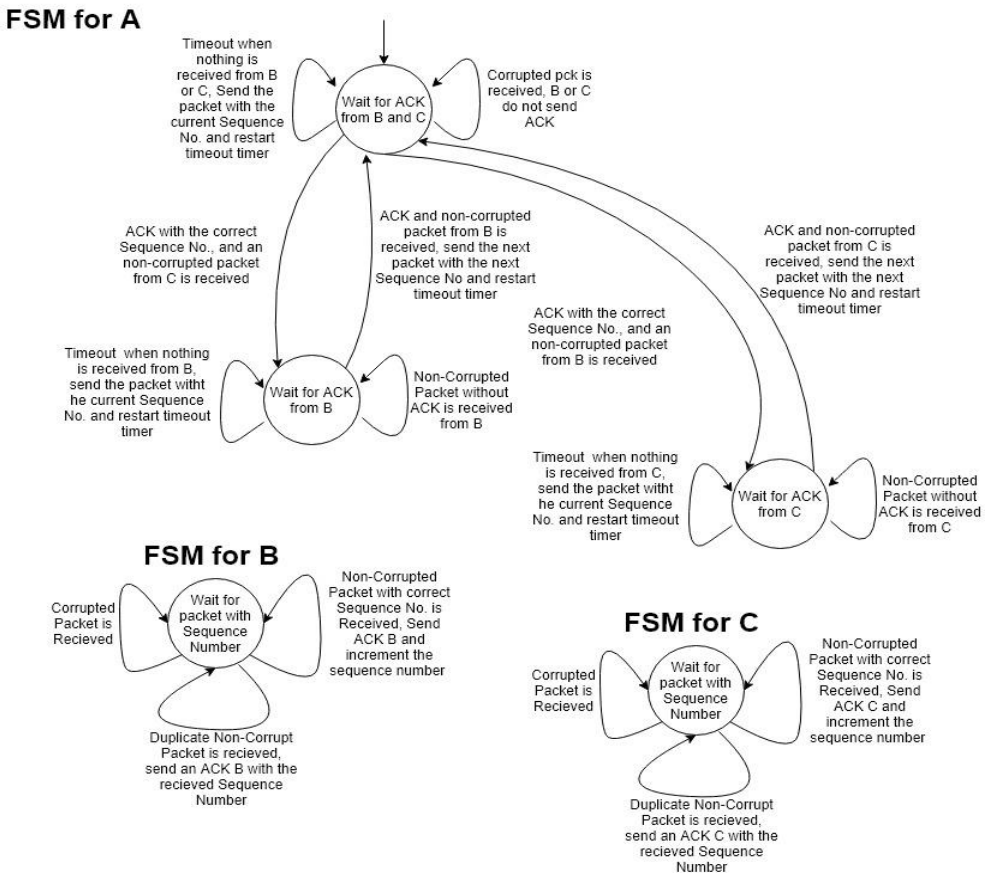
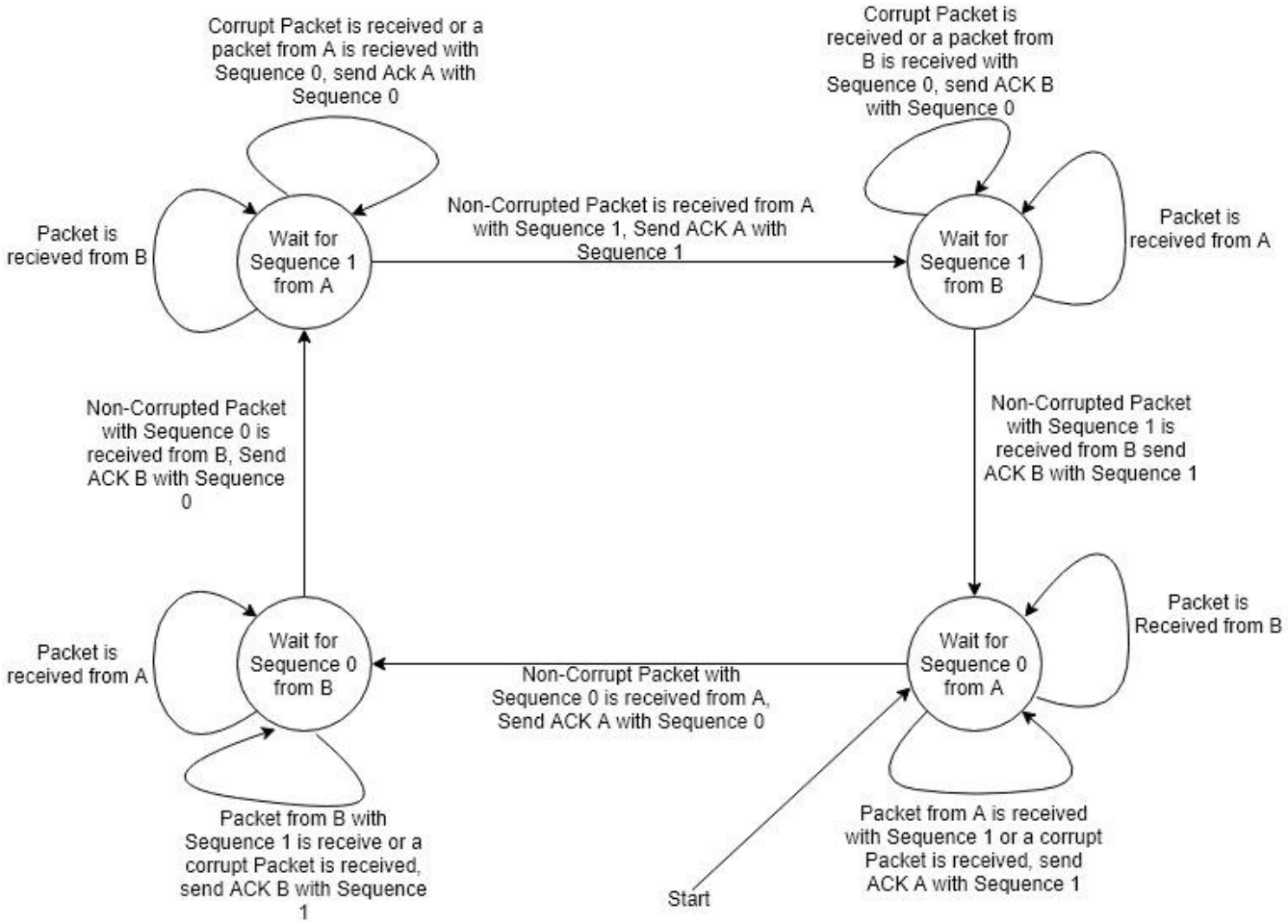
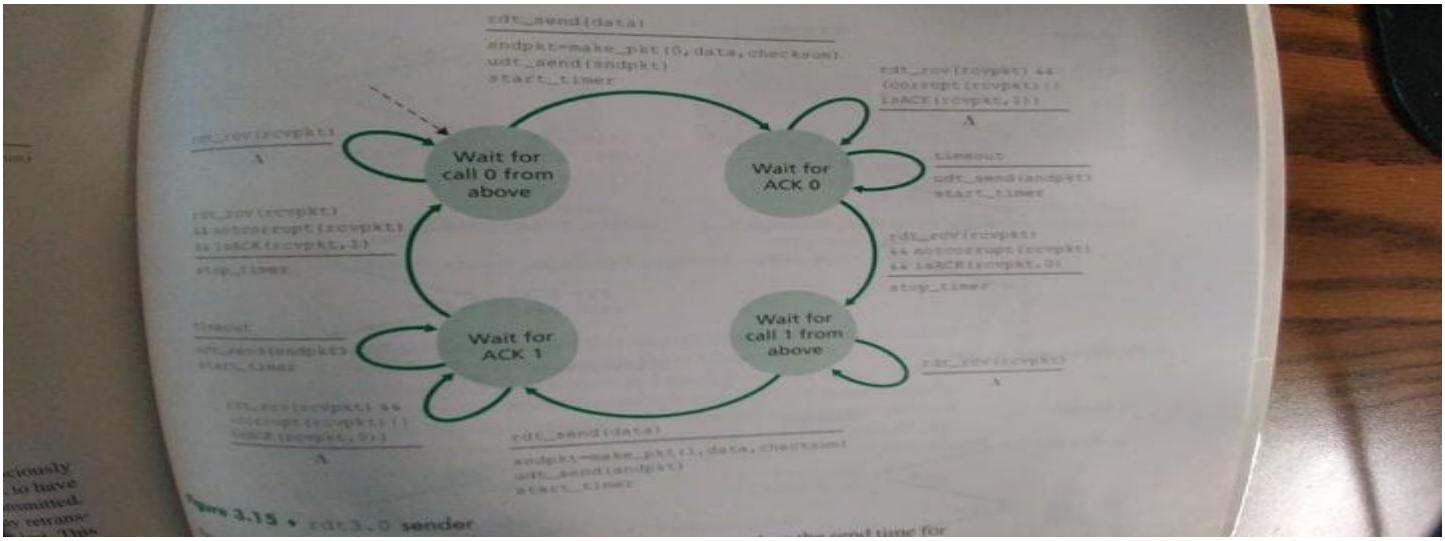
1. P11:
   a. **No,** the protocol would not work if we were to remove $sndpkt = make\_pkt(ACK, 0, checksum)$, because a deadlock would occur with the sender and receiver. If the sender sends a corrupted pck1 to the receiver, it waits to receive a packet from the receiver. In this case if we remove the $sndpkt = make\_pkt(ACK, 0, checksum)$ when the receiver receives the corrupted pkt1 the receiver will just enter into Wait-for-1-from-below state without sending anything back to the sender. Thus, a dead lock occurs where the sender is waiting for an ACK but it will never get one because the receiver is waiting for an uncorrupt pkt. The same scenario would occur if we were to remove the Waitfor-0-from-below state yielding a deadlock.
2. P14:
   a. Use a NAK only protocol would yield a lot of delay in detecting packet loss if the sender sends data infrequently. If the sender were to send pck0, then after some time (due infrequency) the sender sends pck1 then pck2. The receiver receives pck0 then after a significant amount of time it receives pc2, after it receives pck2 instead of pck1 the receiver realizes that pck1 has been lost.
   b. In the case where the sender sends data more frequently a NAK only protocol would be preferable because the sender will be able to send more data and not have to worry about getting an ACK from the receiver the only time the receiver needs to send anything to the client is when a packet is lost.
3. P16:
   a. The alternating ACK0 and ACK1 would increase channel utilization as the receiver upon getting packets from the sender will need to send ACKs to the sender allowing the sender to pipeline data through the channel.
   b. A potential problem with this would be that because not all the data may get to the receiver, there is no way for the receiver to communicate with the sender that there has been data loss. Meaning that the receiver may not receive all the data the sender sent.
4. P19:



a. From A to B and C:
    i. Use sequence number to keep track of which packets have been received, if the correct ACKs with the correct sequence number is received then increment the sequence number and send the next packet with the updated sequence number
b. From B and C to A
    i. Use an ACK number to keep track of which packet are received, if a non-corrupted packet containing the correct sequence number is received send the ACK with the correct sequence number and increment the sequence number to send with the next packet.

5. P20:





a. Sender FSM:
    i. The sender is the same as the alternating bit protocol on Figure 3.15 in the Textbook from Chapter 3 pg. 217

b. Formats:
    i. A and B to C, and C to A and B: use an alternating bit to keep track of alternating packets from A and B
    ii. Receiver: if at any state a non-corrupt packet or one with the correct sequence number is received then an ACK with that sequence number and the sender (A or B) is sent
    iii. Sender: Use alternating bits (0 or 1) for the packets that the sender sends, a timer is started once the packet is sent with its checksum, data, a bit.

6. P24:
    a. True: here's an example:
        i. At T1 packets 1-5 are sent by the sender and then the sender starts it's timeout timer. At T2 packets 1-5 are received by the receiver and then it send ACKs 1-5 to the sender, the sender times out because it doesn't receive the ACKs 1-5 from the receiver and then resends packets 1-5. At T3 the receiver receives packets 1-5 again and then resends the ACKs 1-5 to the sender. At T4 the sender receives the ACKs 1-5 and then sends packets 6-10. At T5 the sender receives the ACKs for the packets it sent in T2. So it is possible for a the sender to receive ACK for a packet outside of its current window with SR.
    b. True: same as the example above.
    c.
    d. True: the difference between alternating-bit protocol and the SR protocol is that the window size for the alternating-bit protocol is 1, because for every packet sent the sender waits for an ACK to send to the next.
    e. True, the explanation for C is the same for D
7. P27:
    a. $SequenceNo = 127 + 80 = 207, Source\ Port: 302, Destination\ Port: 80$
    b. $ACK\ No: 207, Source\ Port: 80, Destination\ Port: 302$
    c. $ACK\ No: 127$, the receiver is waiting for 127 bytes and beyond

d.