Directions: Write your name on the exam and on every page you submit. Write something for every question. Students who do not write something for everything lose out over students who write down wild guesses. You will get some points if you attempt a solution but nothing for a blank sheet of paper. Write something down, even wild guesses. Problems take long to read but can be answered concisely.

<table>
<thead>
<tr>
<th>Question</th>
<th>Maximum</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Overview, 40 points: Give the most important answer. 4 points apiece.

- **a) Spam and Layering:** Some ISPs use anti-spam devices at the network that check for the content of every packet to see if it is likely spam (i.e., junk mail) and drop those packets. The ISP claims that looking at email contents (not headers) is not a layer violation. More specifically, they claim that if the routing or transport layer changes, they would not have to reprogram their anti-spam device. Explain what is wrong with this argument.

- **b) Media Impairments:** Cat 5 cable has a Nyquist Bandwidth of 100 Mhz. In order to transmit at 100 Mbps over Cat 5 cable why can you not use Manchester encoding?

- **c) Clock Recovery:** Why phase-locked loops are preferable to simply resetting the receiver clock when a transition is detected.

- **d) Preambles and Transitions:** Why is 01011 01011 01011 a good preamble for 4-5 encoding whereas 0101010101 is a good preamble for Manchester encoding (Hint: in 4-5 encoding besides bit synchronization what other kind of synchronization is needed).

- **e) Media:** Most laptops use 802.11 (WiFi) which uses radio transmission. Why would go wrong (or right) if the 802.11 standard decided to use infrared instead of radio?

- **f) Framing:** Why is 11111111 a bad flag for framing regardless of the stuffing rule used?

- **g) Latency vs Throughput:** In the old days of batch processing, a number of jobs were run at the same time on an IBM 360. Explain in terms of latency and throughput what batch processing is trading off. Be precise (say latency and throughput of what, and whether each measure increases or decreases).

- **h) Restarting Data Link Protocols:** Suppose after a crash, the sender sends a RESTART message using a pseudo-random number generator that uses the sender ID as a seed. What could go wrong?

- **i) Multiplexing:** When a voice call is made, 64kbps of data bandwidth is reserved across any digital line in the path of the call. Why is strict multiplexing reasonable for a voice call instead of statistical multiplexing?

- **j) End-to-end argument:** In the Internet, vendors sometimes require that their packets be encrypted (i.e., coded such that the packet cannot be read by intruders listening on the wire). Why is it better to have the packet encrypted by the source and decrypted at the destination (i.e., end-to-end) instead of being encrypted and decrypted by each router in the path (i.e., hop-by-hop)?
2. Framing with a flag and a length: There is a protocol called DDCMP that does framing with a start flag followed by a frame length. Thus there is no need to do bit stuffing because flags in the data are skipped over by the receiver because it knows the frame length. A frame format is shown below. Thus the receiver looks for a start flag, reads the length of the Data and expects to find an end flag at the end. Notice that a false flag in between is skipped over. If it does not find a end flag, the receiver assumes the end flag is corrupted and the receiver looks for a start flag immediately after the corrupted end flag. If the start flag is also corrupted, the receiver assumes that the next field is the length field, and continues as before.

![Flag Length False Flag](image)

Figure 1:

- (2 points) a) If a few errors occur in the flag bits and they stop, argue that the receiver stays in synchronization with the sender.

- (3 points) b) Despite your answer in a), explain how the receiver may get out of synchronization with the sender because of even a single error.

- (5 points) c) Describe an example where the receiver gets out of synchronization with the receiver and stays out of synchronization for ever even after all errors stop. This may require some data that is very unlikely to occur in practice; don’t worry if you think it is unlikely, just state your example.

- (7 points) d) Examples like the one you created above in c) should be rare. Try to invent a simple strategy to get back into synchronization after you have lost frame synchronization.

- (3 points) e) By contrast, if you did bit stuffing (and there were no false flags in the data), describe a simple strategy to get back in synchronization after errors. Argue that an example like you gave in c) can never occur and the receiver is guaranteed to get back in synchronization after errors stop regardless of the data.
2. **CRCs** Peter Protocol only knows how to compute 8 bit CRCs efficiently (CRC-8) and cannot figure out how to implement CRC-16 efficiently. No worries, he thinks, I can just use two CRC-8’s in the same frame as shown in the figure. He uses a standard CRC (shown as CRC-1). He then adds a second CRC, shown as CRC-2, that applies to the original data plus CRC-1.

![Diagram of CRC-1 and CRC-2](image)

- (5 points) Suppose there is an error in positions i, j, . . k in the Data. Assume that positions i, j, . . k are numbered starting with the LSB of CRC-1 (in other words, the first bit of the data will be Bit 8.) Write down the error polynomial with respect to CRC-1

- (5 points) Now write down the error polynomial with respect to CRC-2. Note that the numbering of bits will change with respect to CRC-2 because with respect to CRC-2, bit 0 is the LSB of CRC-2. So the same error will result in a different error polynomial for CRC-2

- (6 points) Using your understanding of polynomial division, show that if CRC-1 fails to detect an error in the data, then CRC-2 will also not detect the error.

- (4 points) Based on your last answer, how does Peter Protocol’s idea of using two CRC-8’s compare with using a single CRC-8? How does it compare to using CRC-16?
3. **SMART Retransmission Strategy, 20 points:** The SMART retransmission strategy is a retransmission strategy that tries to do as well as selective reject without the overhead of adding a bitmap (or a list of sequence numbers). The idea is that the receiver sends the same cumulative ack as before but also sends one extra number — the number of the received frame that caused the ack to be sent. To see how this works, imagine the sender sends frames (see figure below) with sequence numbers 0, 1, 2, 3, 4, 5 but frame 1 and 4 get lost. On receiving 0, the receiver will send back Ack (1,0); on receiving 2, the receiver will send back Ack (1, 2); on receiving 3, the receiver sends back (1, 3); and on receiving 5, the receiver sends back (1, 5).

- When the sender receives Ack(1,2) why does the sender know enough to retransmit D(1)? (2 points)
- When sender receivers Ack (1,3) why does the sender know enough not to retransmit D(2)? What information does the sender need to maintain to not take any action at this point? (6 points)
- Write down on the picture (in place of the question number) the acks that are sent after receiver gets D(1) and D(4). (4 points)
- When the ack to the retransmitted D(1) arrives, how does the sender know not to retransmit D(4)? (4 points)
- Why is this scheme better than the fast retransmit scheme (done by TCP) where 3 duplicate acks trigger a retransmission. (4 points)

![Figure 3:](image-url)