Charles Zhang
UID: 305413659

UCLA CS 118 Computer Networks

Midterm Exam

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.

Question	Maximum	Score
1	40	32
2	20	20
3	20	17
4	20	20
Total		89

1, Overview, 40 points: Circle **exactly one** alternative, the best you can think of. 4 points apiece. Do not give your reasoning! Partial credit is available for some alternatives that are closer to the right answer.

1. **Layering and Layer Violations:** Many clouds today use intrusion detection systems (IDSs) that watch every packet and look for suspicious strings in the content that indicate a possible attack.
   ↳ routing

   Packet Routing, not transport →

   −2

   - ~~i)~~ Ben Bitdiddle argues this is not a layer violation because an application level device (the IDS) is not looking at TCP, IP, or Data Link. headers.
   - ~~ii)~~ Ben Bitdiddle argues that content is not really a header so it does not matter
   - iii) Sam Spade argues this is a layer violation because the IDS will be have be reprogrammed if we replace TCP with a different transport that has different header lengths.
   - iv) Sam Spade argues this is a layer violation because application data is not being read through an interface.

2. **Fourier Analyis:** In the homework we saw that the Fourier coefficients of the square wave went down linearly with the frequency. This is because:
   - i) When you integrate $sin(2\pi ft)$ there is a multiplier that is equal to $1/f$
   - ~~ii)~~ When you differentiate $sin(2\pi ft)$ there is a multiplier that is equal to $1/f$
   - ~~iii)~~ When you integrate $sin(2\pi ft)$ there is a multiplier that is equal to f
   - ~~iv)~~ When you differentiate $sin(2\pi ft)$ there is a multiplier that is equal to f.

3. **Media Impairments, Shannon Limit intuition:** Suppose you have a source that can send at any voltage from 0 to 4V and noise can be at most 0.5 V. The source would like to send multiple bits at a time by using one signal with different voltages. What is the maximum number of bits it can send at a time to avoid the possibility of two signal levels being confused with each other.

   $0 \to -0.5 \to 0.5 \to 00$
   $1 \to 0.5 \to 1.5 \to 01$
   $2 \to 1.5 \to 2.5 \to 10$
   $3 \to 2.5 \to 3.5 \to 11$
   $\to$ 2 bits

   - i) 8
   - ii) 4
   - iii) 2
   - iv) 1

4. **Media in the news:** Why is Ukraine relying so heavily on StarLink which is a system of Low Orbiting (not geosynchronous) Satellites. Give the best answer you can think of (there will be partial credit for some less important reasons)

   −2

   - ~~i)~~ Because it is cheap
   - ~~ii)~~ Because it is wireless
   - iii) Because it has lower latency than a geosynchronous satellite
   - ~~iv)~~ Because it has higher latency than a geosynchronous satellite

5. **End-to-end argument:** Why does FiberChannel (the Data Link used to connect computers to disks) uses hop-by-hop retransmission despite its expense? Give the best answer you can think of (there will be partial credit for some less important reasons)

- i) Reduced latency in case of loss
- ii) Increased thoughput because bit errors are very common in data centers
- iii) Can do hop-by-hop retransmission in hardware
- iv) ~~More~~ secure

6. **Bit Stuffing Rules:** Suppose we choose the flag 10101010 (instead of 01111110 for HDLC) and try to avoid false flags by stuffing a 1 after 101. 10① 1011 10

false
10 10101010
1011 01010
can only create a false end flag

- i) The stuffing rule is correct and can never create a false flag.
- ii) It fails but the only way to produce a false flag is because data at the end of a frame can be mistaken for the start of the end flag.
- iii) It fails but the only way to produce a false flag is because a stuffed bit could be followed by data to create a false end flag.
- iv) Both cases in ii) iiii are possible and neither is the *only* way.

7. **Latency vs Throughput:** One can connect two Data Links using either a bit-by-bit repeater or a router that only sends the right packets onwards after looking at the destination address in the packet.

−2

- i) The router solution has higher latency and lower throughput than the repeater.
- ii) The router solution has lower latency and lower throughput compared to the repeater.
- iii) The router solution has higher latency but higher throughput than the repeater.
- iv) The router solution has lower latency but higher throughput than the repeater.

8. **Multiplexing:** Wavelength division multiplexing is often used on fiber links to create more bandwidth by sending lights of different colors. Suppose company A is assigned color Blue and company B is assigned color Red. What is one advantage and one disadvantage of this scheme compared to multiplexing as in Ethernet.

- i) Advantage: Company A is guaranteed a fixed bandwidth; Disadvantage: Company A cannot use Company B's bandwidth.
- ii) ~~Advantage: Company A can use Company B's bandwidth;~~ Disadvantage: Company A is guaranteed a fixed bandwidth.
- iii) Advantage: Company A is guaranteed a fixed bandwidth; Disadvantage: ~~There can be chromatic dispersion caused by the red and blue wavelengths~~.
- iv) Advantage: ~~Company A can use Company B's bandwidth~~; Disadvantage: There can be chromatic dispersion caused by the red and blue wavelengths.

9. **Quasi-reliability in Data Links:** There are many reasons why Data Link CRCs designed to detect errors with very low probabilities (say once a year) when data frames are lost much more often. Which one of these is one reason for using such high quality CRCs?

?

−2

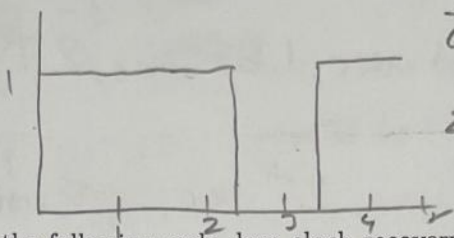
- i) Corrupted frames can cause routers to crash.

- ii) Hop by hop error recovery can fail if the sequence number is corrupted.
- iii) Hop by hop error recovery can fail if the destination address is corrupted.
- iv) Forwarding can fail if the sequence number is corrupted.

10. j) **Restarting Data Link Protocols:** One way to fix RESTART protocols is to rely on a maximum frame lifetime. What is the disadvantage of this method?

    - ~~i)~~ Because frame lifetimes are hard to estimate accurately on most links
    - ~~ii)~~ Because it is difficult to synchronize clocks at the sender and receiver
    - iii) Because non-volatile memory is commonly available in routers
    - iv) Because it increases the time to reboot a router

#	Clk	P	lag
0	0.5	1	0
1	1.5	1.6	-0.4
2	2.1	2.2	-0.4
3	2.7	3.2	0
4	3.7		



#	C	P	L
0	0.5	1	0
1	1.5	2	0
2	2.5		6,2

2, Clock Recovery, 20 points: As in HW 1, the following code does clock recovery for 4-5 but this time with an optional defense against noise. We have the usual code except for the last line which ignores the measured lag if its absolute value is too high. Recall —x— is the absolute value of x, so whether x is +5 or -5 it returns 5.

```
Receiver Code
Data Structures and Initialization:
 T: real constant; (* nominal time to send a bit, input to program *) -> 1 usec
 P = 0: real; (* predicted next time for transition*)
 A: real; (* actual real time at which a transition occurs *)
 lag = 0: real; (* difference between actual and predicted *)
 StartTimer (T/2);
 Wait (TimerExpiry); -> place at half bit

 Do until end of frame
   Output (SampleSignal); (* output sampled value when timer expires *)
   P = P + T + lag;
   StartTimer (T + lag);
   Wait (Timer Expiry);
   In parallel with Wait look for a Transition if any
     If Transition is detected at actual time A
        lag = A - P; (* difference between real and predicted *)
        if | lag |> T/3 then lag = 0 (* OPTIONAL LAG ADJUSTMENT IF ABSOLUTE
                                                  VALUE OF LAG IS TOO HIGH *_
 End
```

#	0	Clk	P	lag
0	1	0.5	1	0
1	1	1.5	2	0
2	0	2.5	3.2	0.2
3		3.7		

You are going to run this code assuming a nomimal bit time of 1 usec (e.g., T = 1 usec) and a sender who is sending 10% slower than the receiver. Thus the sender sends his first bit from 0 to 1.1, second bit from 1.1 to 2,2,usec etc. The sender sends 1101

- **a, 2 points)** What are the ideal mid-bit transition times? $\boxed{\text{~~0.85usec, 1.65usec, 2.75usec, 3.85usec~~}}$
- **b, 4 points)** Write down the sampling times assuming no noise. applying the pesuducode above (4 points) $\boxed{\text{~~0.5usec, 1.5usec, 2.5usec, 3.7usec~~}}$
- **c, 4 points)** Now assume there is a noise pulse that causes a transition at t = 0.6. Write down the sampling times assuming **we do not do the optional lag adjustment**. We want all sampling instants until the time of the final, ideal midbit transition which may be more sampling instants than in the other parts of the problem. (4 points)
- **d, 4 points)** Now assume there is an instantaneous noise pulse that causes a transition at t = 0.6 (the noise pulls down the high level briefly to a 0). Write down the sampling times assuming we do the optional lag adjustment .
- **e, 3 points)** What is one possible advantage of lag adjustment? How does it compare to phase locked loops? (2 points)
- **f, 3 points)** What is one disadvantage of lag adjustment? Why might it not be such a big deal in practice? (2 points)

a) 0.55 usec, 1.65 usec, 2.75 usec, 3.85 usec +2

b) 0.5 usec, 1.5 usec, 2.5 usec, 3.7 usec +4

c) 0.5 usec, 1.5 usec, 2.1 usec, 2.7 usec, 3.7 usec +4

d) 0.5 usec, 1.5 usec, 2.5 usec, 3.7 usec +4

e) Lag adjustment makes our sampling times more resilient to noise, reducing errors like those we saw in HW1. Phase locked loops slowly adjust sampling times, so they dampen the effect of noise while lag adjustment can completely ignore noise. Phase locked loops also adjust slower to real transitions, while lag reduction (assuming the predicted transitions are relatively accurate) adjusts to real transitions as quickly as normal. +3

f) For lag adjustment, if we are not precise about the cutoff for when we set lag=0, we may end up ignoring actual transitions that just happened to be far off from our prediction. +3
In practice, the sender/receiver clocks should never be so out of sync that this occurs, so a reasonable boundary should be able to be determined quite easily.

ABCO -> 4ABC
ABC -> 3ABCO

1 1111111 2YZLO
127

3, Alyssa's Framing Scheme, 20 points: Recall Alyssa's framing scheme from the Homework. The idea was that if you want to send say XOLO, where O represents the zero byte we first always add a zero "padding" byte to get XOLOO. We then encode this (to remove all zero byte characters O by putting in length fields where the zero byte characters were) to get say 2X2L1. We finally add the framing characters to send O2X2L1O. In addition if there is a block that is 254 bytes without a zero we encode it with a length of 255. We want to consider some variants below.

- (5 points) **a)** Suppose we change Alyssa's scheme to add a padding byte of O only if the frame does not already end with a O. So in the example above we would not pad, and encode XOLO as 2X2L. What goes wrong?
- (5 points) **b)** Suppose we do not do the change in **a)** but move all length fields to the start of the frame. For example instead of 2X2L1 we encode the data as 221XL. The same information is being preserved; we are only reordering bytes. However, we do need to know when the length fields are finished and the data (without the zero bytes) begins. For this example, you need to know that the data fields begin at the fourth byte with X. What is a simple rule you can use to know this?
- (5 points) **c)** Cathy Careful, Alyssa's manager decides we wish to remove not just the zero byte O but also the byte of all 1's (say I) so as to use the framing character O for start of frame and I for end of frame. So the frame above would not be sent on the wire as O2X2L1O but as O2X2L1I. So now suppose we need to remove two bytes, I or O from within a frame. To do so, Alyssa, changes her rules as follows. If the first bit in a length byte is a 0, that block ends with a O. If the first bit in a length byte is a 1, that block ends with an I. The remaining 7 bits encodes the length of the block. What other rule does Alyssa need to change to make this change?
- (5 points) **d)** Cathy Careful claims that the new framing scheme with frames starting with O and ending with I is more resilient to errors. Cathy says that even if the physical layer corrupts bytes and so that framing is lost, it is easy to get back into correct frame synchronization once all errors stop. Explain why Cathy thinks so.

a) If we do this, we have no way to tell if the trailing O is padding or data. If it's padding, it must be removed from the data, but we can no longer check if this is the case. +5

b) Add an O byte after the data fields. It's impossible to have a length of O and all O should be removed from the data. During decoding, we know there will be a O byte between the length bytes and the data bytes, so we treat it as a delimeter, not a framing character. +2

c) As is, Alyssa has no way to encode blocks that end in neither O nor I. Alyssa must reserve 01111111 to represent these blocks and force all blocks to have a maximum length of 126, like how the 255 case was handled.

+5

d) Before, since both the begin-frame and end-frame bytes were O, it was impossible to tell if a O was beginning a frame or ending a frame, unless you were already synchronized. Now, differentiating between I and O makes this issue much easier to resolve.

+5

Scanned with CamScanner

4. Error Recovery, State View, Selective Reject Modulus, 20 points: Consider a sender and receiver doing the Selective Reject code over a FIFO link in both directions. Suppose the window size is 3. The sender and receiver both start with sequence number 0 and the link is empty. Assume initially the sequence numbers are infinite integers. So the initial state is:

```
 ----------                                    ---------
 | s = 0 | ------------------------------------| r = 0 |
 |       |-------------------------------------|       |
 ---------                                     --------
```

Suppose the sender now sends data packets with all the outstanding packets in its window before the receiver receives any. But also suppose the sender has a bad timer and retransmits its entire window before the receiver receives any data packet, The state becomes:

```
 ----------          2  1  0   2  1  0         ---------
 | s = 0 | ------------------------------------| r = 0 |
 |       |-------------------------------------|       |
 ---------                                     --------
```

- **a, 4 points)** Suppose the receiver receives the first three data packets numbered 0, 1, and 2 and sends an ACK back but the ACK is not received. What is the new state of the system? Mark it on the picture where the question marks are:

```
            ----------          ? D(2) D(1) D(0) ---------
 0|1|2      | s = ? | ------------------------------| r = ? |   0|1|2|3|4|5
   ↓        |     0 |-------------------------------|     3 |        ↓
 0|0|0      ---------           ? A(3)               --------   1|1|1|0|0|0
```

(The problem continues on the next page)

-4

- **b, 4 points)** Suppose the ACK is received but no more data packets are received (in other words, the second copy of the first window 0, 1, 2 is still on the link). Suppose the sender now sends the next window 3, 4, 5 after receiving the ACK. What is the new state of the system? Mark it on the picture below where the question marks are:

```
0|1|2|3|4|5   ---------   D(5) D(4) D(3)? D(2) D(1) D(0)   ---------   0|1|2|3|4|5
     ↓        | s = ? | -----------------------------------| r = ? |        ↓
1|1|1|0|0|0   |   3   |------------------------------------|   3   |   1|1|1|0|0|0
              ---------               ?          +4         ---------
```

- **c, 4 points)** Suppose now all except data packets 0 and 5 are dropped because of error. Suppose 0 and 5 are received. What should the receiver do with 0 and 5 if it is using Go Back 3. What if it is using Selective Reject?
- **d, 4 points)** Suppose now instead of using infinite sequence numbers we use a Modulus of 5 in the same scenario still using Selective Reject. What goes wrong?
- **e, 4 points)** What is the minimum size of the Modulus to avoid this bug for Selective Reject using a window size of 3?

c) In Go Back 3, the receiver must reject both packets since it expects packet 3 at that time.
In Selective Reject, the receiver rejects 0 because R>0 and buffers 5, since it is within the window but is out of order. +4

d) Selective Reject cannot differentiate between 0 and 5, as both values mod 5 equal 0. As a result, the receiver would accidentally treat packet 0 as packet 5, accepting it ~~and~~ buffering it. Meanwhile, packet 5 would be seen as a re-transmission. +4

e) Selective Reject requires a minimum modulus of 2w, or 6 in this case. +4