Solution: CS120 Homework Assignment #1

Name: ______________ ID: ___________________

Please read the following instructions carefully before answering the questions:

- This assignment is to be completed by each student **individually**.
- There are a total of **6** questions.
- When you write your answers, please try to be precise and concise.
- Fill your name and student ID at the first page.
- Please typeset the file name and format of your submission to the following one: YourID_CS120_HW1.pdf (Replace "YourID" with your student ID). Submissions with wrong file name or format will **NOT** be graded.
- Submit your homework through Blackboard.

1. (10 points) How "wide" is a bit on a 10-Gbps link? (5 points) How long is a bit in copper wire, where the speed of propagation is 2.3×10^8m/s? (5 points)

   10 Gbps = 10^{10} bps, meaning each bit is 10^{-10} sec (0.1 ns) wide. The length in the wire of such a bit is 0.1 ns × 2.3 × 10^8 m/sec = 0.023 m or 23mm.

2. (20 points) Calculate the total time required to transfer a 1000-KB file in the following cases, assuming an RTT of 50 ms, a packet size of 1 KB data, and an initial 2 ×RTT of "handshaking" before data is sent:

   (a) The bandwidth is 1.5 Mbps, and data packets can be sent continuously. (5 points)

   (b) The bandwidth is 1.5 Mbps, but after we finish sending each data packet we must wait one. RTT before sending the next. (5 points)

   (c) The bandwidth is "infinite", meaning that we take transmit time to be zero, and up to 20. packets can be sent per RTT. (5 points)

   (d) The bandwidth is infinite, and during the first RTT we can send one packet (2^{1-1}), during the second RTT we can send two packets (2^{2-1}), during the third we can send four (2^{3-1}), and so on. (5 points)

   (a) 2 initial RTT's (100ms) + 1000KB/1.5Mbps (transmit) + RTT/2 (propagation = 25ms) ≈ 0.125 + 8Mbit/1.5Mbps = 0.125 + 5.333 sec = 5.458 sec.

   If we pay more careful attention to when a mega is 10^6 versus 2^{20}, we get 8,192,000 bits/1,500,000bps = 5.461 sec, for a total delay of 5.586 sec.

   (b) To the above we add the time for 999 RTTs (the number of RTTs between when packet 1 arrives and packet 1000 arrives), for a total of 5.586 + 49.95 = 55.536.

   (c) This is 49.5 RTTs, plus the initial 2, for 2.575 seconds.

   (d) Right after the handshaking is done we send one packet. One RTT after the handshaking we send two packets. At n RTTs past the initial handshaking we have sent $1 + 2 + 4 + \cdots + 2^n = 2^{n+1} - 1$ packets. At n = 9 we have thus been able to send all 1,000 packets; the last batch arrives 0.5 RTT later. Total time is 2+9.5 RTTs, or 2.575 sec.

3. (5 points) Suppose that a certain communications protocol involves a per-packet overhead of 50 bytes for headers and framing. We send 1 million bytes of data using this protocol; however, one data byte is corrupted and the entire packet containing it is thus lost. Give the total number of overhead + loss bytes for packet data sizes of 1000, 10,000, and 20,000 bytes. Which size is optimal?

The number of packets needed, N, is $\lceil 10^6/D \rceil$, where D is the packet data size. Given that overhead = $50 \times N$ and loss = D (we have already counted the lost packet's header in the overhead), we have overhead+loss = $50 \times \lceil 10^6/D \rceil + D$.

D	overhead+loss
1000	51000
10000	15000
20000	22500

The optimal size is 10,000 bytes which minimizes the above function.

4. (25 points) The CRC algorithm as presented in this chapter requires lots of bit manipulations. It is, however, possible to do polynomial long division taking multiple bits at a time, via a table-driven method, that enables efficient software implementations of CRC. We outline the strategy here for long division 3 bits at a time (see Table 1); in practice, we would divide 8 bits at a time, and the table would have 256 entries.

   Let the divisor polynomial $C = C(x)$ be $x^3 + x^2 + 1$, or 1101. To build the table for C, we take each 3-bit sequence, p, append three trailing 0s, and then find the quotient $q = p \frown 000 \div C$, ignoring the remainder. The third column is the product $C \times q$, the first 3 bits of which should equal p.

Table 1 Table-Driven CRC Calculation

p	$q = p \frown 000 \div C$	$C \times q$
000	000	000 000
001	001	001 101
010	011	010 ___
011	0__	011 ___
100	111	100 011
101	110	101 110
110	100	110 ___
111	___	111 ___

(a) Verify, for $p = 110$, that the quotients $p \frown 000 \div C$ and $p \frown 111 \div C$ are the same; that is, it doesn't matter what the trailing bits are. (5 points)

(b) Fill in the missing entries in the table. (10 points)

(c) Use the table to divide 101 001 011 001 100 by C. Hint: The first 3 bits of the dividend are $p = 101$, so from the table the corresponding first 3 bits of the quotient are $q = 110$. Write the 110 above the second 3 bits of the dividend, and subtract $C \times q$ = 101 110, again from the table, from the first 6 bits of the dividend. Keep going in groups of 3 bits. There should be no remainder. (10 points)

(a)

Solution 1

$$p \frown 000 \div C = 100$$

$$\begin{array}{r}
100 \\
1101\overline{)\,110000} \\
\underline{1101} \\
100
\end{array}$$

$$p \frown 111 \div C = 100$$

$$\begin{array}{r}
100 \\
1101\overline{)\,110111} \\
\underline{1101} \\
011
\end{array}$$

Solution 2

$$p \frown \mathrm{XXX} \div C = 100$$

$$\begin{array}{r}
100 \\
1101\overline{)\,110\mathrm{XXX}} \\
\underline{1101} \\
\mathrm{YXX}
\end{array}$$

(b)

Table 1 Table-Driven CRC Calculation		
p	q = p ⌢ 000 ÷ C	C × q
000	000	000 000
001	001	001 101
010	011	010 111
011	010	011 010
100	111	100 011
101	110	101 110
110	100	110 100
111	101	111 001

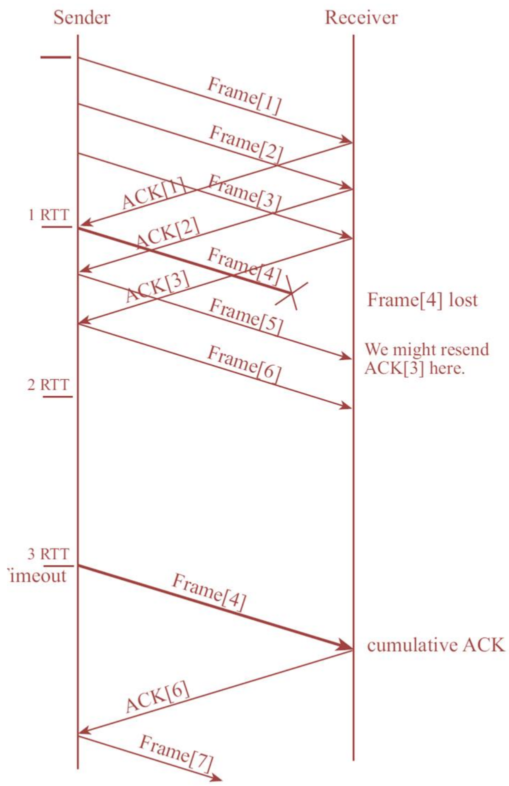
(c)

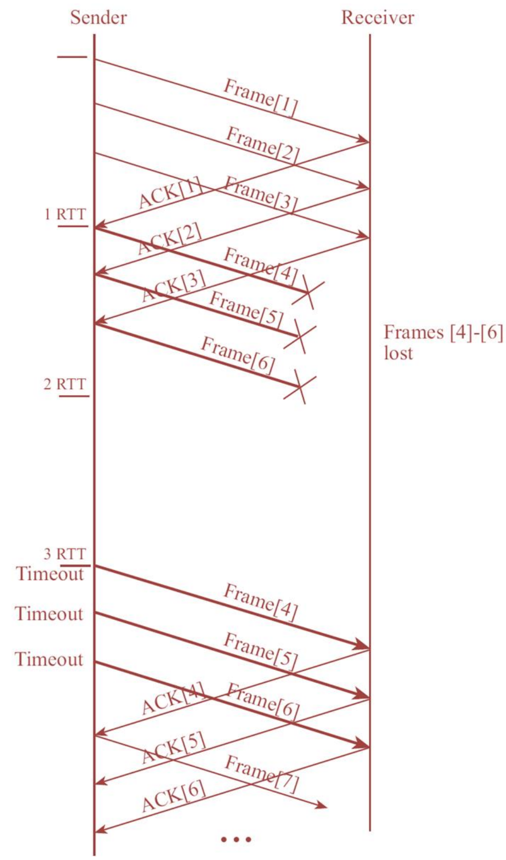
$$\begin{array}{r|ccccc}
 & & \mathbf{110} & 101 & 011 & 100 \\ \hline
1101 & \mathbf{101} & 001 & 011 & 001 & 100 \\
 & \mathbf{101} & \mathbf{110} & & & \\ \cline{2-3}
 & & 111 & 011 & & \\
 & & 111 & 001 & & \\ \cline{3-4}
 & & & 010 & 001 & \\
 & & & 010 & 111 & \\ \cline{4-5}
 & & & & 110 & 100 \\
 & & & & 110 & 100 \\ \cline{5-6}
 & & & & & 0
\end{array}$$

5. (20 points) Draw a time line diagram for the sliding window algorithm with SWS = RWS = 3 frames, for the following two situations. Use a timeout interval of about 2 × RTT.

(a) Frame 4 is lost. (10 points)

(b) Frames 4 to 6 are lost. (10 points)





6. (20 points) Let A and B be two stations attempting to transmit on an Ethernet. Each has a steady queue of frames ready to send; A's frames will be numbered A_1, A_2, and so on, and B's similarly. Let $T = 51.2$ µs be the exponential backoff base unit.

Suppose A and B simultaneously attempt to send frame 1, collide, and happen to choose backoff times of $0 \times T$ and $1 \times T$, respectively, meaning A wins the race and transmits A_1 while B waits. At the end of this transmission, B will attempt to retransmit B_1 while A will attempt to transmit A_2. These first attempts will collide, but now A backs off for either $0 \times T$ or $1 \times T$, while B backs off for time equal to one of $0 \times T, \cdots, 3 \times T$.

(a) Give the probability that A wins this second backoff race immediately after this first collision; that is, A's first choice of backoff time $k \times 51.2$ is less than B's. (5 points)

(b) Suppose A wins this second backoff race. A transmits A_3, and when it is finished, A and B collide again as A tries to transmit A_4 and B tries once more to transmit B_1. Give the probability that A wins this third backoff race immediately after the first collision. (5 points)

(c) Give a reasonable lower bound for the probability that A wins all the remaining backoff races. (5 points)

(d) What then happens to the frame B_1? (5 points)

This scenario is known as the Ethernet *capture effect*.

(a) A can choose k_A=0 or 1; B can choose k_B=0,1,2,3. A wins outright if (k_A, k_B) is among (0,1), (0,2), (0,3), (1,2), (1,3); there is a 5/8 chance of this.

(b) Now we have k_B among 0...7. If k_A=0, there are 7 choices for k_B that have A win; if k_A=1 then there are 6 choices. All told the probability of A's winning outright is 13/16.

(c) P(winning race 1) = 5/8 > 1/2 and P(winning race 2) = 13/16>3/4; generalizing, we assume the odds of A winning the ith race exceed $(1 - 1/2^{i-1})$.

We now have that:

P(A wins every race given that it wins races 1-3) $\geqslant$ (1 −1/8)(1 − 1/16)(1 − 1/32)(1 − 1/64).... $\approx$ 3/4

(d) B gives up on it, and starts over with B_2.

Another solution of (c):

P(A winning race i | A has won race 1~(i-1)) = $\frac{2^{i+1}-3}{2^{i+1}}$

P(A winning race 4~n | A has won race 1~3) = $\prod_{i=4}^{n} \frac{2^{i+1}-3}{2^{i+1}} \approx$ 0.8239, $n \to \infty$

or according to Exponential Backoff scheme:

P(A winning race 4~15 | A has won race 1~3)

= P(A winning race 4~10 | A has won race 1~3)

* P(A winning race 11~15 | A has won race 1~10)

= ($\prod_{i=4}^{10} \frac{2^{i+1}-3}{2^{i+1}}$) * $(\frac{2^{11}-3}{2^{11}})^5 \approx$ 0.8189