UNIVERSITY COLLEGE LONDON
DEPARTMENT OF COMPUTER SCIENCE
3035/GZ01: Networked Systems
Brad Karp
Individual Coursework 1
Distributed: 16th October 2014; Due: 23rd October 2014, 4:05 PM

Instructions: Answer all of the following problems. Either handwritten or typeset solutions are fine, but if you write by hand, please ensure your answers are legible. Please show all work! We cannot award credit for correct answers if their complete derivation isn't shown. Please state clearly all assumptions you make while solving a problem. This coursework is worth 8% of the total marks for 3035/GZ01.

Please monitor the 3035/GZ01 Piazza site during the period between now and the due date for the coursework. Any announcements (*e.g.,* helpful tips on how to work around unexpected problems encountered by others) will be posted there.

Hand-in instructions: Hand in hardcopy for your solutions at the start of lecture at 4:05 PM on the 23rd of October. There is no provision for electronic submission of this coursework. Any late submissions should be handed in at the 5th floor reception desk in the CS department (in MPEB).

Collaboration: Collaboration is *not permitted* on this problem set; you may not discuss the problems or their solutions with anyone else (whether or not the other person is taking the class), apart from the instructors and teaching assistants. All work you submit must be your own. You may of course refer to all lecture notes and readings, and any other materials you wish (textbooks, papers, or material found on the Internet).

Late days: If you wish to use any late days on this assignment, you *must* state how many days you wish to use clearly at the top of the first page. If your written assignment does not include a request to use late days, you will not be permitted to use date days on the assignment later. (Full late-day policies are stated on the 3035/GZ01 web site.)

1. **Catching Burst Errors with the CRC**

   In class we stated that the CRC will detect a burst error of length at most $g-1$ bits, provided that $G(x)$ contains the terms 1 and x^{g-1}.

   (a) Let's begin with the example case of a single burst error of two bits occuring at the end of a four-bit packet with generator $G(x) = 1 + x^2$.

      i. What are the resulting generator bits for $G(x)$?

      [1 mark]

      ii. What are the resulting bits of the error polynomial $E(x)$?

      [1 mark]

      iii. Show all steps of the long division to argue that the CRC will detect this particular burst error.

      [2 marks]

   (b) Now let's reason about any single burst error of length $k \leq g-1$ occuring i bits from the end of the frame. Recall from lecture that an error goes undetected whenever $E(x) = G(x) \cdot Z(x)$ for some polynomial $Z(x)$.

i. What is the error polynomial $E(x)$ for this error pattern? (Your answer will be a polynomial in x involving i and k).

[2 marks]

ii. Argue that in order for $E(x)$ to equal $G(x) \cdot Z(x)$, $Z(x)$ must contain the term x^i.

[2 marks]

iii. Why therefore can $E(x)$ never equal $G(x) \cdot Z(x)$? (Hint: Note that by your answer to the above question, $G(x) \cdot Z(x)$ must contain the term x^{i+g-1}.)

[2 marks]

2. **Implementing the CRC in Software (P & D 2.20)**

The CRC algorithm discussed in lecture requires lots of bit manipulations, for which a hardware design excels, but it is, however, possible to efficiently implement the CRC in software. Our approach will be to perform polynomial long division using a table-driven method, taking multiple bits at a time. We'll investigate the strategy here for long division three at a time, but in practice we would divide 8 or 16 bits at a time, and the table would have 256 or 65,536 entries.

p	$q = p.000 \div G$	$G \times q$
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 ___

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

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

[2 marks]

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

[3 marks, ½ mark per entry]

(c) Use the table to divide 101 001 011 001 100 by G.

Hint: the first three bits of the dividend are $p = 101$*, so from the table the corresponding first three bits of the quotient are* 110*. Write the* 110 *above the second three bits of the dividend, and subtract* $G \times q = 101110$*, again from the table, from the first six bits of the dividend. Keep going in groups of three bits. There should be no remainder.*

[3 marks]

3. **Odd or even?**

Argue convincingly that if the CRC generator polynomial $G(x)$ contains the factor $(1+x)$ then a CRC using $G(x)$ correctly flags packets containing *any* odd number of bit errors as incorrect.

Hint: Remember that the CRC misses an error whenever the error polynomial $E(x) = G(x) \cdot Z(x)$ for some $Z(x)$. What happens when you substitute 1 for x in this equation?

[5 marks]

4. **Reordering in rdt v3.0**

Networked Systems student Louis Reasoner has just studied `rdt3.0`, the stop-and-wait reliable transfer protocol with a one-bit sequence number described in lecture. Louis thinks that rdt3.0 would function even when the network reorders packets. Is he right? If so, argue informally why. If not, provide a counterexample timeline in the style of the timelines presented in lecture, showing sender and receiver states, and packet contents.

[5 marks]

5. **The Ethernet Capture Effect**

Suppose two stations, X and Y, are connected by an Ethernet, and each has an inexhaustible supply of frames ready to transmit. We denote the series of frames that X and Y wish to transmit as $X_1, X_2, \ldots$ and $Y_1, Y_2, \ldots$, respectively.

Consider the following example scenario, which illustrates a phenomenon known as the *Ethernet capture effect.* Suppose that X and Y initially attempt to transmit their first frames at exactly the same time. X and Y's transmissions collide. Let us assume that in this case X chooses a backoff interval of zero backoff slots while Y chooses a backoff interval of one backoff slot. Thus X "wins" and transmits X_1 first, and Y waits during X's transmission. Once X's transmission ends, Y will attempt to retransmit Y_1, but X will attempt to transmit X_2. Suppose that these two transmissions again collide, with the result that X chooses a backoff interval chosen uniformly in $[0,1]$ backoff slots, whereas Y chooses a backoff interval chosen uniformly in $[0,3]$.

(a) Derive the probability that X wins (*i.e.,* transmits first) this second backoff "race" between X and Y—that is, that X transmits before Y.

[2 marks]

(b) Assume now that X wins this second backoff race. Immediately thereafter, X transmits X_3 and Y retransmits Y_1. Derive the probability that X wins this *third* backoff race between X and Y.

[2 marks]

(c) Derive general expressions for the probability that X wins the ith backoff race and the probability that Y wins the ith backoff race.

[6 marks]

(d) Thus far, you've considered the Ethernet capture effect when there are only two nodes with frames to send. Does the Ethernet capture effect—broadly defined as one node repeatedly winning in contention to transmit during repeated collisions—worsen or ameliorate as the number of nodes with frames to send increases?

[2 marks]

[Problem set total: 40 marks]