**ECE/CS 4434/6434  
Homework 5  
Due Date: Friday, October 11, 11:59 PM**

**Reading: Lectures 9 – 11 ⎯ Chapter 3 of I. Koren + Chapter 2 of R. K. Iyer**

**Problem 1 (10 pts)** –For each of the following codes, determine the code distance and the number of bit-errors they can detect or correct.

**Part A (5 pts)** Repetition code of length n = 4: C1 = {0000, 1111}

Code Distance = 4, Detect = 3, Correct = 2

**Part B (5 pts)** Single-bit even parity code: C2 = {000, 011, 101, 110}

Code Distance = 2, Detect = 1, Correct = 0

**Problem 2 (15 pts)** –Consider a (5, 4) cyclic code with the generator polynomial G(X) = X + 1. For each of the following parts, provide an answer in the space provided. Show your work and a justification for your answer to get partial credit.

**Part A (5 pts)** For data word D1(X) = (0110), we obtain the codeword C1(X) = **\_\_\_\_1010\_\_\_\_\_**

**Part B (5 pts)** The codeword C2(X) = X4+X is received with no error, the corresponding data word was D2(X) = \_\_**\_\_\_\_\_\_\_**

**Part C (5 pts)** The codeword C3(X) = X4+X3+X2+X+1 is received, is an error detected or not?

**Hint:** Remember that a cyclic code (n, k) encodes data into codewords of length *n*, by multiplying *k*-bit data word by a *n-k* degree polynomial.

**Problem 3 (30 pts) –** Given that X7 − 1 = (X+1) g1(X) g2(X), where g1(X) = X3+X+1,

**Part A (10 pts)** Calculateg2(X).

**Part B (10 pts)** Identify all the (7, *k*) cyclic codes that can be generated based on the factors of X7 − 1. How many different such cyclic codes exists?

**Part C (10 pts)** Show all the codewords generated by the polynomial G(X)=(X+1)g1(X) along with their corresponding data words. What is the distance for this code? What are the error detection capabilities of this code (how many errors it can detect or correct)?

**Hint:** Note thatsubtraction in modulo-2 arithmetic is identical to addition, thus Xn − 1 is identical to Xn +1.

**Problem 4 (10 pts)** – ***Checksums*** are commonly used in communication applications to detect errors in data transmission. The basic idea is to add up the block of data words to be transmitted and then transmit the sum (called the *checksum*) along with the data. The receiver adds up the data it receives and compares this sum with the checksum received. If the two do not match, an error is indicated. There are several variations of checksums. If the data words are ***d*** bits long, the *single-precision checksum* is calculated by performing **module-2d**addition of the words, ignoring any overflows in addition. The *double-precision checksum* is calculated by **module-22daddition** of the data words.

The following figure shows a scenario where four 4-bit words (*d3,d2,d1,d0*) are transmitted over a communication channel. Assuming that the line carrying bit *d3* is faulty with a stuck-at-1 value, fill in the empty boxes by showing the received data words and calculating the single-precision and double-precision checksums for both transmitted and received data. Then for each case indicate whether the error will be detected or not.

****

**Problem 5 (15 pts) –** Remember that an (*n*, *k*) cyclic code can detect single bit errors and multiple adjacent bit errors affecting fewer than (*n - k*) bits. Show all your work for the following parts to get partial credit.

**Part A (5 pts)** Show that if the generating polynomial G(X) of a cyclic code has more than one term, all single-bit errors will be detected.

**Part B (5 pts)** Show that if G(X) has a factor with three terms, all *adjacent* double-bit errors will be detected.

**Part C (5 pts)** Show that if G(X) has X + 1 as a factor, all odd numbers of bit errors will be detected. That is, if encoded word contains an odd number of terms (errors), it does not have X + 1 as a factor. **Hint:** Note that a single bit error in bit position *i* can be represented by Xi and the received codeword with such an error can be written as D(x)G(x) + Xi*.*

**Problem 6 (20 pts) –** A CRC code has the following generator polynomial:



Calculate the probability of detecting 17-bit burst error patterns and 18-bit burst error patterns.

**Hint:** Represent the data arriving as a sum *T(x)* = *D(x)g(x) + e(x)*, where *D(x)* is the original bit string, *e(x)* is a burst error vector, and + (plus) denotes the XOR operator. The error vector has 1's in the bit positions that are in error. A single burst error is characterized by an initial 1, a mixture of 0’s and 1’s, and a final 1, with all other bits being 0.

**Hint:** If the remainder of *T(x) / g(x)* is zero, then no error is detected.