**CE 4951 COURSE PROJECT**

**MESSAGE-EXCHANGING NODES ON CSMA/CD BUS USING MANCHESTER LINE CODING WITH BUS THAT IDLES HIGH**

**TEST PROCEDURE FOR FOURTH MILESTONE DEMONSTRATING ERROR CHECKING AND RANDOM BACK OFF**

**November 1, 2018 (Ver. 1.0)**

**Original by Charles Boyer and Kyle Krueger**

**SETUP TEST:**

The unit under test (UUT) is message-exchange node being tested and should be configured with the receiver, transmitter, and channel monitor software.

The test setup will be a loop-back system where the channel monitor, transmitter, and receiver are connected to the same node. Characters will be sent and received through a terminal service such as PuTTy. The following figure, Figure 1, shows the hardware setup of the unit under test.

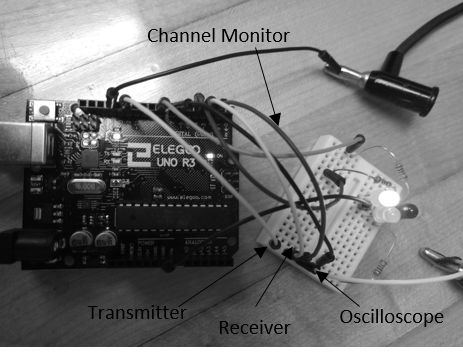


Figure 1 - Test Setup with transmitter, receiver, channel monitor, and oscilloscope connected to the same node.

**BEGIN TEST:**

Some setup tests like powering on the system and checking the state of the channel monitor on startup.

1. Upon startup of the UUT, ensure that the line is IDLE. ***Indicate Pass or Fail here: \_\_\_\_\_\_\_\_***
2. To test the random back off, wire the GND to the input of the receiver. The collision LED, red LED, should now be on. Now set the oscilloscope to Single capture-mode. Now type “test” into the putty console. At this point no message should have been sent, meaning that the oscilloscope shouldn’t be triggered. ***Indicate Pass or Fail here: \_\_\_\_\_\_\_\_***
3. Now remove the GND connection to the input. After a few seconds the system should return to idle. Now the oscilloscope should have triggered. After a gap of IDLE-line voltage the message is sent. **Measure the width of the gap: \_\_\_\_\_\_\_\_\_\_\_\_\_.**
4. Now repeat steps 2 and 3 two more times. **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.** To prove the Random Back off to is operating randomly, you should notice that all three widths are different. ***Indicate Pass or Fail here: \_\_\_\_\_\_\_\_***
5. Now to test the timeout of retransmission, repeat step 2 and wait roughly 20 seconds and then remove the GND connection. At this point no transmission should be sent because the system went into timeout mode because of the number of collisions before timeout was reached. ***Indicate Pass or Fail here: \_\_\_\_\_\_\_\_***
6. Lastly is to check the CRC, first step is not enabling the console to print out the CRC bits for transmission and the receiver.
7. Now to test a single bit, in the putty console send the letter “A”. The transmission CRC bits should be **11000000** in binary, which should read outas **192** in decimal**. *Indicate Pass or Fail here: \_\_\_\_\_\_\_\_***
8. On the receiving end, the CRC bits calculated for the receiver should be **000000000.** Demonstrating error-free transmission. ***Indicate Pass or Fail here: \_\_\_\_\_\_\_\_***
9. Now to test 2-Bytes. in the putty console, send the letters “Az”. The transmission CRC bits should print out **00101111,** which is **47** in decimal**. *Indicate Pass or Fail here: \_\_\_\_\_\_\_\_***
10. On the receiving end, the CRC bits calculated for the receiver should be **00000000.** Demonstrating Error free transmission. ***Indicate Pass or Fail here: \_\_\_\_\_\_\_\_***
11. In the code settings, set the CRC flag to send to not send a CRC. This will force the FCS field to be sent as 0xAA, which is 10101010 or 170 in decimal. Send the character, “B”, through the Putty interface, and check the CRC value sent and received and ensure it is 0xAA. **Indicate Pass or Fail here: \_\_\_\_\_\_\_\_**

The CRC flag should be set to off, or 0x00. **Indicate Pass or Fail here: \_\_\_\_\_\_\_\_**

**Witnesses**

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Student Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Student Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_