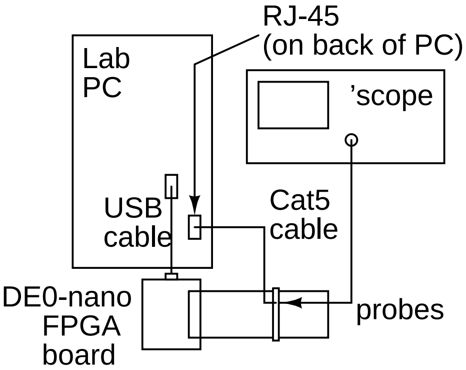
10 Mb/s Ethernet Transmitter

Introduction

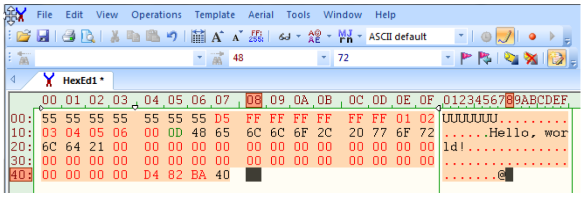
In this lab you will create an Ethernet frame using a hex editor and implement a simple interface to transmit that frame using the 10 Mb/s Ethernet PHY to a PC. You will use a 'scope to measure the interface voltages and signal timing. You will use the Wireshark protocol analyzer to display the frame contents.



Create An Ethernet Frame

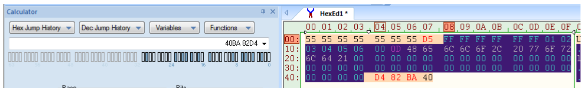
Run HexEdit and use menu item File⇒New to create a new file. Right-click in the data area and select "Allow Changes". Use the Insert key to switch between inserting and over-writing existing values. Use the Tab key to switch between entering values as hex or ASCII characters.

Enter the bytes of an Ethernet packet that contains your full name as the data. Include the preamble and SFD. The destination address should be the broadcast address (all 0xff). The source address should be `1:2:3:4:5:6`. The length/type field should be the length of the packet (not including any padding). The data field should contain your name padded with zeros if necessary to reach the minimum frame length.



```
    00 01 02 03  04 05 06 07  08 09 0A 0B  0C 0D 0E 0F  0123456789ABCDEF
00: 55 55 55 55  55 55 55 D5  FF FF FF FF  FF FF 01 02  UUUUUUU.........
10: 03 04 05 06  00 0D 48 65  6C 6C 6F 2C  20 77 6F 72  .......Hello, wor
20: 6C 64 21 00  00 00 00 00  00 00 00 00  00 00 00 00  ld!.............
30: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
40: 00 00 00 00  D4 82 BA 40                            .......@
```

To compute the FCS, select the bytes *following the preamble* and compute the 32-bit CRC using the menu item Operation⇒Checksum⇒CRC 32. Add this 4-byte value to the end of the packet. Note that the CRC value computed by HexEdit is a 32-bit value that must be transmitted in order from LS to MS bit (i.e. you will have to reverse the byte order).



```
40BA 82D4

    00 01 02 03  04 05 06 07  08 09 0A 0B  0C 0D 0E 0F
00: 55 55 55 55  55 55 55 D5  FF FF FF FF  FF FF 01 02
10: 03 04 05 06  00 0D 48 65  6C 6C 6F 2C  20 77 6F 72
20: 6C 64 21 00  00 00 00 00  00 00 00 00  00 00 00 00
30: 00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00
40: 00 00 00 00  D4 82 BA 40
```

Select all of the data (control-A) and export the packet contents to a file (e.g. `packet.hex`) in Intel hex format (File⇒Export⇒Intel Hex Records).

A Simple 10 Mb/s Ethernet Transmitter

A block diagram of the transmit side of a very simple 10 Mb/s Ethernet transmitter is shown in Figure 1. The components of the transmitter are as follows:

- a 20 MHz clock is used to generate two pulses for each bit to enable Manchester line coding at a bit rate of 10 Mb/s
- a 12-bit counter divided into three fields:
  - the least significant bit selects the pulse number (0 or 1) of the Manchester symbol
  - the next 3 bits select the bit being transmitted (0 to 7 with 0 being the LS bit)
  - the most significant 8 bits select the byte to be read out of a 256-byte memory
- a byte-wide memory that stores the data to be transmitted
- a multiplexer that selects the bit to be transmitted
- an exclusive-or gate that inverts the value being transmitted during the second half of each bit. This produces the Manchester encoding.

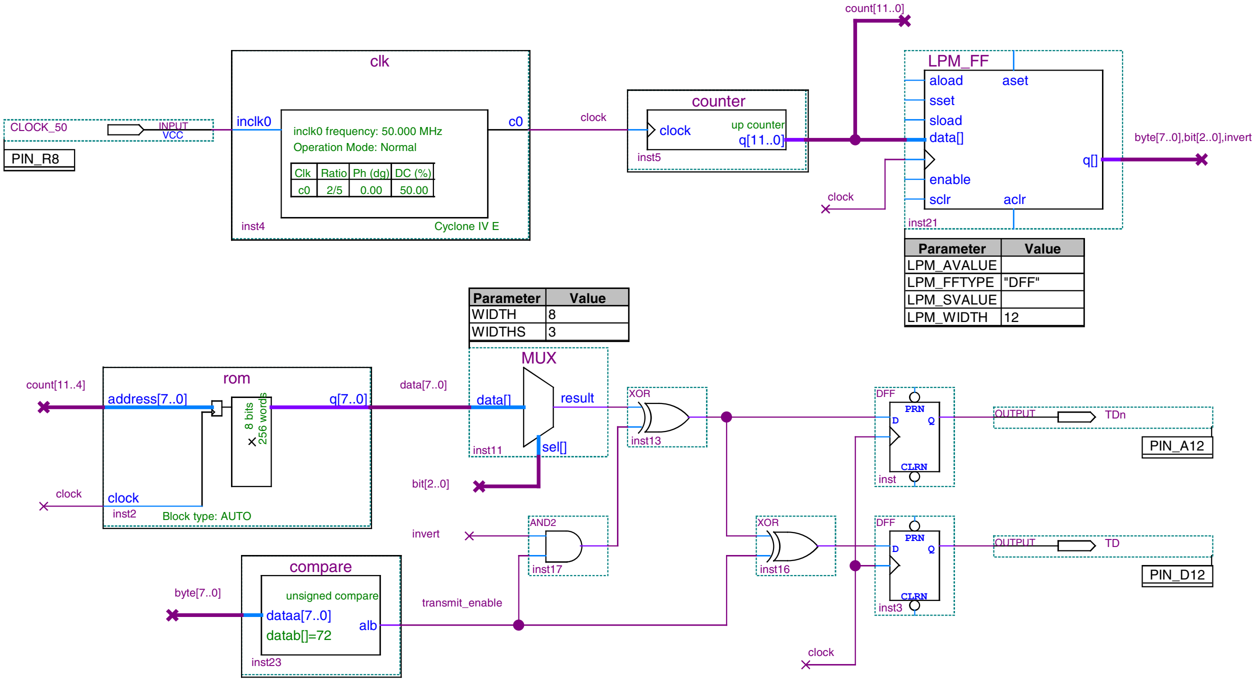


Figure 1: 10 Mb/s Ethernet Transmitter.

- an inverter and two flip-flops that hold the differential outputs fixed during each half-bit duration

- a comparator and an 'and' gate that disables Manchester coding and differential outputs when the byte index exceeds the length of the frame (the length is a constant that you must configure in the comparator block); this creates an idle period between frames

- the 12-bit count register provides a one-clock delay to match the delay through the ROM's address latch.

Configure the Transmitter

You can build the transmitter yourself or use the supplied archived project files.

To use the supplied design files, download the Quartus II project archive file from the course web site and open it with Quartus II. Extract it to convenient folder (see warning above). The project will then be opened. In the Project Navigator window's Files tab you can double-click the `.bdf` file to edit the schematic.

If you used the supplied design you will need to change parameters in the `rom` and `compare` blocks. Right-click on each of these blocks and run the IP Parameter editor to modify the block parameters:

- change the data stored in the memory by specifying the `.hex` file you prepared above.

  Move the hex file you created previously into the Quartus II project directory and specify this file in the "Mem Init" tab of the `rom` component. The memory in your FPGA design will then be initialized with the contents of this file.

- change the constant comparison value (b) to the length of your packet.

  Specify the packet length as the constant value of 'b' in the "General 2" tab of the `compare` component. This disables the transmitter after the end of the packet.

A 50 MHz clock input (`CLOCK_50`) comes from the on-board oscillator connected to `PIN_R8`. The two transmit data outputs (`TD` and `TDn`) should be configured as pins `PIN_D12` and `PIN_A12` of the FPGA. These pins are connected to pins labelled `GPIO_032` and `GPIO_030` on the expansion connector[1], the low-

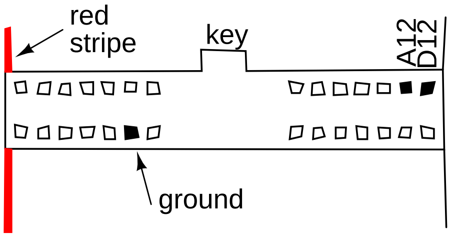
[1] A previous lab has all of the pin names and locations.

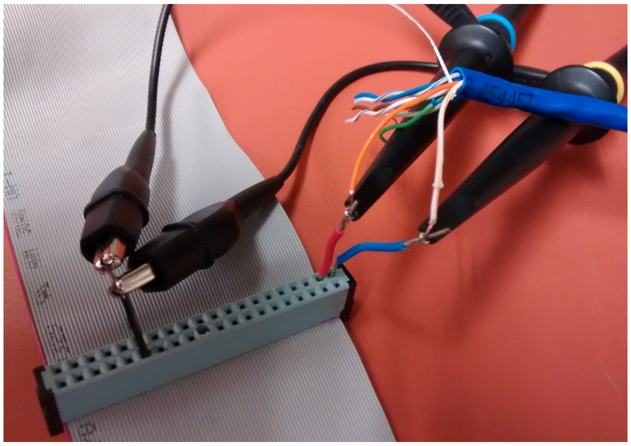
est two pins of on the side with the “key.”

Compile the design and program the FPGA.

Connect one side of the supplied ribbon cable to the header pins on the side of the FPGA board that has the USB connecctor.

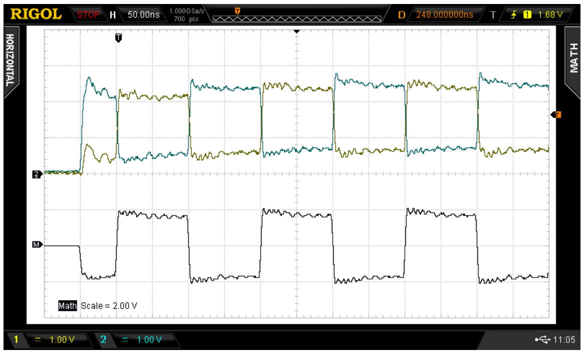
Use short wires to connect the two ’scope probes to the appropriate sockets on the other end of the ribbon cable and ground to the ground pin as shown below (you will connect the Cat-5 cable later):





Use the oscilloscope to check that the voltage levels, signal period and the preamble waveform are as expected. Do not connect either signal to ground! Use MATH mode (A-B) to measure the differential signal.

Set the ’scope for single capture with the trigger threshold set at about 1.5 V (press RUN/STOP and press SINGLE until you capture the start of a frame). Take a screen capture showing the superimposed ground-referenced signals and the differential voltage at the start of the packet:



Strip and connect the white/orange and orange (`RX+` and `RX-`) wires at the other end of the cable and use the ’scope probes to connect these to the FPGA outputs as shown above.

Connect the Ethernet cable to the unused (upper) RJ-45 connector on the back of the PC. If necessary, ask the instructor for help in locating the right port.

Verify that the signal is not distorted. Unterminated cables reflect the signal back with a delay of ≈ 10 ns per meter of cable. This will cause ISI.

Configure the PC’s Interface

You will need to configure the network interface for 10 Mb/s full duplex operation. From the Start menu search for “View Network Connections”. Right-click on the appropriate Local Area Connection (it should be the Intel Ethernet Connection 1217-LM; ask the instructor if unsure) select “Properties”. Click on “Configure...” and under the “Advanced” tab set:

- “Legacy Switch Compatibility Mode” to “Enabled,” and
- “Link Speed & Duplex” to “10 Mbps Full Duplex.”

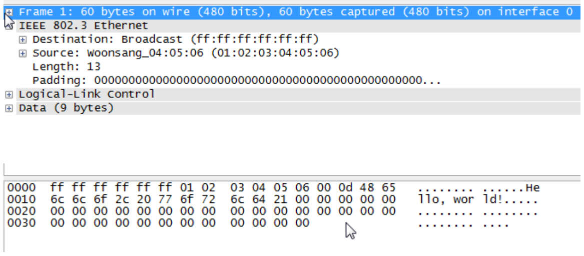
Click OK. The interface should now say “Identifying” or “Unidentified Network”. If it says “Network cable unplugged” then check the waveform and connections.

View Packets Using Wireshark

Run the Wireshark protocol analyzer. Select the network interface you configured above and click on “Start.” You should now see your packets being displayed with your name appearing in the data field.

Show the instructor your packets being decoded to get marks for completing the lab.

Take a screen capture (or two) showing the contents of the 802.3 header and the complete contents of the packet in both hex and ASCII for your report. For example:



```
⊞ Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
  IEEE 802.3 Ethernet
   ⊞ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
   ⊞ Source: Woonsang_04:05:06 (01:02:03:04:05:06)
     Length: 13
     Padding: 000000000000000000000000000000000000000000000000...
⊞ Logical-Link Control
⊞ Data (9 bytes)

0000  ff ff ff ff ff ff 01 02  03 04 05 06 00 0d 48 65   ........ ......He
0010  6c 6c 6f 2c 20 77 6f 72  6c 64 21 00 00 00 00 00   llo, wor ld!.....
0020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0030  00 00 00 00 00 00 00 00  00 00 00 00               ........ ....
```

Pre-Lab Report

Create the `.hex` file according to the instructions above. Submit a PDF file containing the usual identification information and a screen capture showing the contents of your packet including the CRC.

Bring the `.hex` file when you come to the lab.

Report

Submit a report including the following:

- the usual identification information
- a printout from HexEdit showing your packet's contents in hex and ASCII (including any corrections you made in the lab)
- a 'scope screen capture showing the single-ended and differential voltages at the start of the packet showing the preamble waveform
- document any changes to the transmitter design other than memory contents and frame length
- screen captures from Wireshark showing the received Ethernet header and data fields.
- Answers to the following questions:

  (1) What is the minimum length of an 802.3 frame, not including the preamble? Does this include the header and FCS?

  (2) What is the frequency of the preamble waveform[2]? Why?

  (3) Assuming the minimum frame size, how many bytes will be transmitted per frame, including preamble and FCS?

  (4) The design above uses a counter to continuously cycle through the 256-byte memory. Packets are continuously transmitted with a gap between them. If you include the overhead caused by the gap, what is the net data throughput (in Mbps)?

  (5) What parts of the frame you created does Wireshark not show?

[2] Read the square wave's period from the 'scope screen capture and compute the frequency. The answer may not be what you expect.