IoT and Networking

1 Overview
The goal of this project is build the essential components of a basic embedded Ethernet appliance. Building on a simple Ethernet framework provided in class, an DHCP client and a TCP application will be added.
The solution must be implemented on a TM4C123GXL board using an ENC28J60 ethernet interface.
As the intent of this project is to understand the details of these simple elements, your code solution can be based only on provided class code and code you write. You should not be incorporating more than a few lines of code from other sources.

2 Command-line Interface Requirements
The solution must provide a command-line interface using UART0 and configuring the device and reading out the status. The command-line interface should support the following commands at a minimum:
dhcp ON | OFF
This command enables and disables DHCP mode and stores the mode persistently in EEPROM.
dhcp dhcp REFRESH | RELEASE
This command refreshes a current IP address (if in DHCP mode) or releases the current IP address (if in DHCP mode).
set IP | GW | DNS | SN w.x.y.z
This command sets the IP address, gateway address, DNS address, and subnet mask (used when DHCP is disabled) and stores these values persistently in EEPROM.
ifconfig
This command at a minimum dumps the current IP address, DHCP mode, subnet mask, gateway address, and DNS address.
reboot:
This command restarts the microcontroller.

3 Power-Up Requirements
On power-up, the solution must initialize the clocks, UART, ethernet controller, and timer service.
The DHCP mode must be retrieved from EEPROM on power-up. If the EEPROM has never been written, it will read 0xFFFFFFFF, so this can be used strategically to determine if the value is valid or the default value. If DHCP mode is disable, then static values for the IP address, subnet mask, gateway address, and DNS address should be retrieved from EEPROM.
If DHCP mode is enabled, then a DHCP discovery process should start anew.

4 Timer Service Requirements
For server parts of the design, a timer service is needed. It can be implemented with systick or a generalpurpose
timer. This should provide the ability to request a timer with a function callback. Applications of this include renew (T1) and rebind (T2) timers for DHCP, gratuitous ARP window after the ARP request send to test the DHCP offer, and DHCP broadcast repeat attempt timer for DHCP.

5 DHCP Client Requirements
When DHCP is enabled, the solution must implement an DHCP client. The behavior of DHCP is detailed in RFC2131, will be discussed in class, and is summarized below.
Initial request:
A DHCP discovery message is broadcast to request an offer. In response, one or more DHCP servers will send a DHCP offer message with a proposed IP address. After selecting an DHCP offer, a DHCP request message is sent to the DHCP server. The DHCP server then sends a DHCP ACK message.
To verify the IP is not used by another device, an ARP request is sent to the proposed address. A timer is started to allow time for ARP responses.
If an ARP response is recelved in the timer period, then the IP address is in use and a DHCP decline is sent to the DHCP server. The DHCP server then try again with a different IP address.
If no ARP response is not received in timer period, the client records the IP address, subnet mask, gateway server address, DNS server address, lease time, and DHCP server address (important later for

renewal). These values are not stored in the EEPROM as they are not persistent over a power cycle. A timer is started with a period of T1 (usually 50% of the lease time).
When a power cycle occurs, this process must be repeated.
Renewal:
If a T1 timeout occurs, a new timer with a period of T2 (87.5% of lease time) is started. the client should start to send DHCP request to the DCHP server that granted the request above in an attempt to renew the DHCP assignment. It should repeat this regularly until the T2 timeout occurs.
Rebinding:
When a T2 timeout occurs, the client will attempt to rebind by sending a broadcast to all DHCP servers as in initialization.

6 TCP
The solution must implement a TCP state machine that implements server functionality (will be in passive open state) with support for at least one socket. Support for either a simple web server with HTTP (port 80) or a Telnet server (port 21) must be provided.
If in CSE6359, you should extend support to SSH (port 22) or HTTPS (port 80) using the WolfSSL libraries. The security part of your course requirement can extend into the later part of the course.