Marina

The IP stack for Apple II

June 2015

by D. Finnigan

What is Marina?

Marina is the newest TCP/IP stack for the Apple II series, written in pure 6502 assembly language by D. Finnigan. Marina uses the Uthernet 10 Mb/s Ethernet card by A2RetroSystems to connect to an Ethernet LAN. With Marina, the Apple assembly language programmer has access to the UDP, DNS, ICMP, and IP protocols. Written with performance in mind, the send throughput of Marina on a standard Apple II can reach 15 packets per second.

The June 2015 edition of Marina adds a Berkeley sockets API. Though not fully complete, there is enough to construct UDP client applications that can easily send and receive UDP datagrams. Once fully completed and debugged, the sockets interface will make it extremely easy to write networking applications using Marina. This latest edition of Marina also includes several other changes and improvements, such as a more capable DNS resolver, which are noted at the end of this document.

Goals of Marina

From the beginning the author kept the following two goals in mind for Marina:

1. To learn more about Ethernet and TCP/IP networking
2. To create a new TCP/IP stack for the Apple II that had more robust RFC compliance than the existing implementations

Features of Marina

Marina has several features not present in other TCP/IP implementations for the Apple II, including:

1. A Berkeley sockets API
2. An automatic DHCP client more compliant with RFC 2131.
3. An ARP implementation fully compliant with RFC 826.
4. A Link-Local addressing implementation following RFC 3927.
5. Address conflict detection implementation following RFC 5227.
6. Separate send buffers for the ARP, DHCP, and DNS modules.
7. A split-buffer design for sending user datagrams to eliminate copying blocks of data.
8. A more robust Ethernet driver that verifies each Ethernet frame has been successfully transmitted.
9. A special-case checksum routine optimized for 20 byte IP header checksums.
10. Robust error and consistency checks for incoming data.

Hardware Requirements

Marina requires an Apple II with at least 48 kilobytes of RAM, capable of accepting an Uthernet card. The Uthernet card must be placed in slot 3. The network connection can be a direct link using a cross-over cable, or a hub, switch, router, or similar equipment that supports 10 Mb/s Ethernet. Marina assigns the Uthernet card the following MAC address: `00:80:10:6D:76:30`.

Marina has been tested to work successfully on the Integer Apple II, Apple II Plus, standard Apple IIe, enhanced Apple IIe, and ROM 3 Apple IIgs. Accelerator users: the Uthernet can be operated at accelerated speed. It does not need to be slowed to 1 MHz.

From testing on Cisco Gigabit switches it has been discovered that the switch won't always auto-negotiate link speed with the Uthernet card, meaning that the orange link light on the

Uthernet won't be lit. A work-around is to use a known-compatible hub or switch between the Apple II and the problem switch.

Starting Marina

Insert side A of the Marina program disk and reboot your Apple. After a few moments you should see the introductory screen. After reading the on-screen instructions, press Return to start the Marina program.

If your Apple does not have Applesoft BASIC, you will have to manually start Marina. From the BASIC prompt, type BRUN MARINA and press Return to start the Marina program.

At startup, Marina will look for an Uthernet card in slot 3. If the card is not found, Marina will exit to the BASIC prompt. Once the Uthernet is located, Marina will attempt to configure an IP address using DHCP. Marina will broadcast two DHCP discovery messages. If no DHCP server responds, then Marina will configure an RFC 3927 Link-Local address. Marina will continue to periodically send DHCP discovery messages. If at any time a DHCP server responds, Marina will attempt to configure the IP settings using the DHCP protocol. At present, Marina has no provision for manual IP address configuration. An address must either be assigned by a DHCP server, or randomly selected according to the procedures in RFC 3927.

Once Marina has configured an IP address, it will probe the local link to make sure that no other host is using that address. Marina sends 3 ARP probes. If no host responds, Marina will announce its address with 2 ARP announcements. If, however, there is an address conflict, Marina will reconfigure a new address, either by selecting a new Link-Local address, or by sending a DHCP decline message and restarting the DHCP discovery process.

Because Marina does not support multi-homing, once it configures a new IP address, it forgets about the old one. This means, for example, that Marina will no longer respond to its link-local address once it has configured an address using DHCP.

Operating Marina

In general, Marina runs like tcpdump because it is merely a technology demonstration at this point and has no real applications written for it. It reports incoming IP datagrams and ARP packets and prints some interesting information about them, such as source address, length, and port numbers. IP and UDP checksums are verified; if correct, a plus sign (+) is printed, if incorrect, a silent scream (@) will be emitted. Some applications will send UDP datagrams with no checksum; if so, a forward slash (/) is printed to indicate that there was no checksum to verify.

Marina will respond to certain types of incoming datagrams or ARP packets. Marina has a complete ARP module and will respond to requests for its IP address. Marina also has a full DHCP client and will respond to all DHCP server messages: OFFER, ACK, and NAK. Marina will acknowledge an ICMP Echo request by printing it, but will not send a response.

Incoming UDP datagrams are sent to the sockets subroutines to determine if the destination port number corresponds to any currently-open socket. Sockets may be bound on either port number alone, or the foreign address/port number tuple. If a relevant socket is located, the socket ID will be printed, followed by a $ (string). If no open socket matches, an * (asterisk) will be printed instead. See the next section for an explanation of the sockets commands.

The user may press any of the following keys to cause the following action:

A Send an RFC 3927 ARP Announcement.
D Send a DHCP discover message.
I Print the currently configured IP address.
L Lookup the IP address for a domain name.
O Open or close a UDP socket.
P Print contents of received UDP datagram from socket.
R Send an ARP Request.
S Send a UDP datagram to remote port 6502.
T Print the ARP table.
Esc Print the total number of received Ethernet frames, dropped Ethernet frames, then exit to the BASIC prompt. These figures are printed in hexadecimal.

The R and S commands accept an IP address in dotted decimal form, e.g. 128.174.180.122. The L command is used to resolve a domain name such as cites.uiuc.edu to an IP address using DNS. The resolver in Marina only queries for A records, following CNAME aliases as needed.

Using the Sockets Demo

The O, P, and S commands all make use of the Berkeley sockets API which is new in the June 2015 edition of Marina.

S will open a UDP socket and use the SENDTO function to send to destination port 6502. Marina allocates an ephemeral source port starting with 49152. The user is prompted to enter a message of up to 255 characters to send; Marina automatically strips the high-bit from all characters. After the message is sent, the socket is closed using SCLOSE. This means that each successive message will be sent with a successively higher-numbered source port.

The O command will open a UDP socket for receiving UDP datagrams using the SOCKET function, printing the resulting socket ID to the screen. If this socket is already open, then pressing O will call SCLOSE to close the socket. When opened, the socket is bound to local port 20000. Pressing P will call RECVFROM on this socket. If there is a new datagram, its contents will be printed in ASCII. Note that there is a race condition in that if a new IP datagram arrives, the old one will be lost. This defect will be corrected in a future edition of Marina. If there is no new datagram for the socket, then pressing P will have no effect.

Technical Overview

There were two major limitations that had to be accounted for when designing Marina: the first is that the network interface card, the Uthernet, does not support interrupt requests. It must be polled for every operation. The second limitation is that the Apple II does not readily support any sort of multitasking or multithreaded system (though the Multitasking Mouse Draw Demo may be adapted in future). Most TCP/IP stacks are designed for concurrency and use timers provided by the operating system to minimize the amount of state that the programmer must himself manage.

There is a third, minor limitation in that no model of Apple except for the IIgs has a built-in clock for timing. This deficiency will be addressed in a future release of Marina by using a TimeMaster II H.O. or similar clock card for timing.

The code layout of Marina is designed around a small initialization phase, followed by a main event loop that contains the majority of program control logic. The initialization procedures are merely to locate the Uthernet card, setup the Ethernet controller, clear some counters and flags, and to initialize the DHCP client by likewise setting counters and copying the Ethernet MAC address into the DHCP message templates.

The main event loop controls IP address configuration, processing of incoming Ethernet frames, sending Ethernet frames, and dealing with user input from the keyboard.

Briefly, the main event loop performs the following tasks upon each iteration:

1. Check if an IP address needs to be configured.
2. If using a Link-Local Address, periodically send a DHCP Discover message.
3. Send an ARP Probe if needed.
4. Send an ARP Announcement if needed.
5. Check for an incoming Ethernet frame, and process it if there is one.
6. Check for keyboard input and take appropriate action.

Future Challenges

The biggest challenge with implementing a practical TCP/IP stack on the Apple II is that the networking protocol is inherently asynchronous, and the standard Apple II is not well equipped to deal with asynchronous events. The challenge is exacerbated by the fact that the Uthernet does not support interrupt requests. The next-generation Uthernet II does indeed support interrupts, but it uses an Ethernet controller that features a built-in TCP/IP stack, so there is no point in writing a TCP/IP stack for it on the Apple II.

As the system presently stands, a program using Marina for communication would have to constantly poll the Uthernet for incoming IP datagrams. TCP segments should be acknowledged in a timely manner, and will be retransmitted as necessary, but it is possible to lose UDP datagrams.

One possible solution is to incorporate both Marina and the user program into a preemptive multitasking framework, using one or more threads/processes for Marina, and one or more threads/processes for the user program. Such a setup would closely mimic TCP/IP implementations on PC and Macintosh operating systems.

Changes to Marina

There have been the following changes since the May 2015 edition of Marina:

- Always ARP for router when a routable address is configured
- HANDLEUDP drops datagrams sent to port 0
- Properly store DNS answer count in NSREMAN
- Don't force domain names to upper case in DNSSPLIT
- UDPCHKSUM now works with UDP datagrams having a zero-byte payload
- IPCONFIG now clears router address

- DNSREPLY now follows CNAME aliases
- HANDLEICMP rejects messages less than 8 bytes
- HANDLEICMP verifies ICMP checksums
- UDPSENDTEST now accepts and sends a message from the keyboard
- Basic ICMP error checking and reporting added to sockets interface
- HANDLEIP copies incoming source address to IPSRC
- HANDLEUDP copies incoming source port to UDPFPORT
- Ethertypes that are not IP or ARP are silently discarded
- Sockets functions added: SOCKET, BIND, CONNECT, GETSOCKNAME, RECV, SEND, RECVFROM, SENDTO, SCLOSE

There have been the following changes since the April 2015 edition of Marina:

- Added checks on destination address for incoming datagrams
- Datagrams not addressed to our IP, or to broadcast, or DHCP replies, are dropped
- Broadcast flag is set for incoming broadcast datagrams
- IP datagrams with head longer than 20 bytes are checksummed and accepted, but options are not processed
- Fixed checksum routine to work with IP headers longer than 20 bytes
- Fixed the check for UDP datagrams sent with no checksum (offset was off by one)
- Added NETBCAST for storing network broadcast address, e.g. 169.254.255.255 or 192.168.2.255
- Incoming check for broadcasts now looks for "all ones" or the network broadcast address
- Updated IPSEND to use the new check on NETBCAST
- Wrote IPTOHEX subroutine to convert dotted decimal addresses to 4 hex digits
- ARPTEST and UDPSENDTEST now take a user-supplied IP address
- Bug fix: datagrams with Link-Local destination addresses were being forwarded to router when a routable address was configured
- intitial DNS resolver implementation that queries for A records
- IPSEND now uses a retry counter for ARP requests. It will not fail if the MAC address was not in the ARP table
- IPSEND forces TTL to 255 when destination is a Link-Local address
- Increased size of ARP table from 4 to 5
- display of incoming IP datagrams shows source address
- renamed UTHERDEMO file to MARINA