Hubmaster: Cluster-Based Access to High-Speed Networks

Glenn Elmore N6GN
Kevin Rowett N6RCE
Ed Satterthwaite N6PLO

Abstract

This paper describes the link-level design of Hubmaster, a packet radio network that we are currently building. Hubmaster is intended to offer medium-speed network access to collections of end users and to serve as a stepping-stone to higher-speed networks. Users are organized into local clusters. Each cluster consists of a centrally located hub and a dynamically formed group of secondaries. The hub time-multiplexes a single channel by polling the secondaries. Our initial design will provide 256 kbit/s links and operate on the 33 or 23 cm band.

1. Introduction and Summary

Amateur access to high-speed packet radio networks will enable a number of intriguing applications [1]. Several of us in northern California hope to learn how to organize and implement such networks by building prototypes and trying to use them. We are as much interested in overall system design as in the details of the particular components. Our goal is to accommodate varying levels of link performance and to provide cost-effective hardware at each level of performance. In particular, we believe that high data rates must be available to end users, not reserved for backbones and the like.

This paper addresses what we consider to be a medium-performance level, with maximum data rates of 100 kbit/s to 1 Mbit/s. It describes the link-level design of a local radio cluster intended to offer network access to end users. Clusters will be connected as transparently as possible by point-to-point links using, e.g., microwave technology that has already been demonstrated [3, 4]. The design of the clusters borrows ideas from local area network (LAN) technology, but users should not view a cluster as just a LAN. Intercluster links are essential parts of our overall architecture, but they are not discussed in detail here.

Each cluster consists of a hub and a dynamically formed collection of secondary stations. All packet exchanges go through the hub and take place on a single, shared RF channel. It is not necessary for secondaries to communicate directly. The hub communicates with one secondary at a time over a half-duplex link. A polling discipline administered by the hub is used to time-multiplex use of these links in a collision-free manner. The polling list is formed dynamically and the collection of active users can change over time. The hub also serves as a store-and-forward switch. It accepts packets from each secondary and subsequently delivers them to another secondary, which might represent a local user, a link to another cluster or a gateway to another network.

We believe that our design will provide a useful increment of performance in a way that is cost-effective, flexible and easy to manage We have not yet tested our ideas; implementation of the hardware and software pieces is still in progress, and no doubt some of the details will change as the design continues to evolve. Thus this paper should be considered an interim report and a request for comments.

2. What Are We Building?

Our initial implementation will provide 256 kbit/s links on the 33 cm band (904/916 MHz). Because of the somewhat dubious future of that

band, we are already planning 23 cm versions of the RF hardware.

We envision a typical cluster diameter of 10 to 20 miles. Coverage of a single cluster is intentionally limited so that bit error rates can be kept low and latencies due to propagation delays are small. The number of active users supported by a cluster at any instant depends upon the traffic patterns and level of service required. We estimate that a dozen or so active users can be accommodated easily. Adding users decreases the effective data rate for each user and increases the latency but has little effect on channel throughput. We have tried to keep the cost of the shared components of a cluster as low as possible, so that growth in the user community can be accommodated by forming new clusters, not by degrading service.

At the logical and RF center of each cluster is a hub. The hub is a shared resource, somewhat comparable to a shared FM repeater. The hub consists of some digital hardware for polling and switching, a radio with an integrated modem, and an omnidirectional antenna.

A dedicated PC with sufficiently fast serial ports would be adequate for the digital end, but we are building a leaner and more specialized board called Mundane I/O or MIO. MIO is a simplification of the previously reported Awesome I/O design [2]. It consists of an NEC V40 microprocessor, program and buffer memory, and a pair of Zilog 85C30 serial channel controllers (SCCs), both DMA-driven. The V40 is code-compatible with the 8086/8; thus inexpensive and widely available software can be used for program development. The V40 is also CMOS, for low power, and highly integrated, for a reduced parts count. MIO can be used either in an ISA expansion slot (IBM PC, XT or AT compatible) or in a stand-alone configuration. In the former, the PC and the V40 communicate through a shared window into MIO memory. In the latter, an MIO can function as a self-contained hub or it can be connected to a non-PC through another SCC port, e.g., to a Macintosh via LocalTalk.

The 904 MHz radios are crystal controlled and provide a maximum output power of about 10 watts. Modulation is FSK; at 256 kbaud, the RF channel bandwidth (sidebands down 70 dB or more) is 2 MHz. The receiver is able to recover data at signal levels as low as -90 dBm. The radios also feature a turnaround time of less than 40 μsec (transmit to receive). They are built on a pair of PC boards and fit into a 5" x 7" enclosure. Further details of the prototypes are reported in [6].

The hub must be able to communicate with all the secondaries and thus uses an omnidirectional antenna. This can be a gain antenna in the vertical plane, and we expect to use a collinear array to achieve a "pancake" pattern with 10 to 12 dB gain. The hub antenna should be elevated just enough to have paths that are line-of-sight or nearly so to all secondaries. Too little elevation produces path losses that reduce system margin; too much interferes with channel reuse among multiple clusters.

The hardware associated with a secondary can be more modest, although as an expedient we will use copies of the hub hardware in our initial tests. One critical difference is the antenna; a secondary is equipped with an antenna that is directional in both axes toward the hub. We plan to use 15 element Yagis with 4.2 λ booms, constructed to NBS designs [7], with a gain of approximately 14 dBd and a half-power beam width of about 30°. There is enough system margin in the design to allow secondaries to operate at lower power levels; we expect 100 mW to be adequate for line-of-sight paths. We have ideas for even simpler digital and RF hardware for the secondaries, but its development currently has low priority.

In operation, the hub controls use of the RF channel by roll-call polling. The polling protocol is loosely based on HDLC NRM (see, e.g., [10, pp. 254-257]). For each secondary in turn, the hub sends any accumulated traffic for that secondary, followed by an invitation to transmit. The secondary, and only the selected

secondary, responds with any waiting packet traffic. The packets are received by the hub, stored, and forwarded to the local destination the next time that destination is polled. Since a secondary may transmit only upon invitation, no packets are lost to collisions.

For greater efficiency, the polling exchanges are piggybacked on the data packets when possible. Also, to keep intracluster routing and switching overheads to a minimum, special short addresses are used within the cluster as described below. The polling list is formed dynamically. The hub periodically broadcasts an invitation for new stations to join; collisions and conflicts are resolved by the usual backoff algorithms. Secondaries are dropped from the polling list by a time-out mechanism.

The hub is linked to hubs in other clusters via one or more high-speed channels. Selected clusters can share a local address space. Traffic between such clusters is simply forwarded to the other cluster by the hub. For other traffic, software in the hub must provide bridging. Any secondary can also provide a bridge or gateway to other networks. In either case, links leading out of the cluster can operate in parallel with the local polling and forwarding as long as any RF channels are adequately isolated and the nodes have sufficient computing power.

3. Why Are We Doing It This Way?

Packet radio using CSMA protocols on the VHF and UHF bands is well-established. The data rates (1.2 to 100 kbit/s) are adequate for many purposes. We do not expect it to disappear, and we plan to offer access to and from the AX.25 network through gateways. One of the greatest strengths of the existing network is also one of its greatest weaknesses — each station is independent and very little coordination or centralized network management is required. Also, the entry cost is very low.

At the other end of the performance scale, existing technology can provide data rates of 1 to at least 10 Mbit/s. The bandwidths to support such data rates are available only at UHF and microwave frequencies. In a companion paper, one of us (N6GN) argues that point-to-point links using high gain antennas are not only necessary but also very attractive at these frequencies [5]. Another of us (N6PLO) has been involved in the development of a high-performance experimental LAN based on full-duplex point-to-point links [9].

We believe that an ultimate packet radio network will be constructed primarily from such links. In the short term, however, there seem to be some practical problems. Providing a digital interface capable of sustained operation at these rates is still expensive; indeed, many personal computers currently in use don't have the memory and I/O bandwidths to benefit significantly from such data rates. The RF circuitry to support full-duplex operation is also more complicated and expensive; in addition, more spectrum and coordination are required, since each such link requires a pair of RF channels.

Finally, there are a number of problems created by the use of point-to-point links and highly directional antennas. The network configuration is less flexible; dedicated antennas and radio hardware are needed for each link, and the end points of new links must be negotiated and coordinated. Also, configuring point-to-point links into a useful network is a challenging problem. Unless a special and highly regular topology such as a ring is used (probably impractical in amateur radio), there must be switching nodes within the network that can cooperate to forward a packet from any source to any destination. Such switches are endpoints of multiple, simultaneously active links, and their antenna systems thus can become unwieldy for amateur installations.

The design presented in this paper is a compromise. It offers an intermediate level of performance at a lower cost and in a more flexible way. Logically, the intracluster links are point-to-point (except during well defined

intervals when new stations are contending to join the cluster), so the problems of organizing such links into a useful network can be addressed and potential solutions evaluated. In addition, one end of each link, the secondary, does benefit from the use of gain antennas. Because the hub is omnidirectional, however, its antenna system is simple, and the set of possible secondaries is limited only by path length. The latter is an advantage for portable and emergency operation. Although resources are used less efficiently, nothing breaks even if it is necessary to operate the secondary with an omni antenna in such circumstances.

A companion paper [5] shows that path loss is independent of frequency in the point-to-omni configuration of a cluster if the physical antenna sizes at the secondaries remain constant. This might argue for using the lowest frequency at which the desired bandwidth is available, since the cost of generating and feeding RF power generally increases with frequency. Actually, by using a collinear array of constant aperture at the hub, we can make limited use of the decrease in path loss with increasing frequency. Construction of such antennas with satisfactory patterns becomes relatively more difficult above 1300 MHz, as does the generation of power at levels above 100 mW. Thus our initial experiments with a cluster configuration use the 33 and 23 cm bands.

We have been influenced in our choice of bands by other pragmatic considerations as well. Both bands are underutilized and need more activity. The membership of our group is currently too scattered to be connected effectively by short direct microwave links. The use of a few longer, non-line-of-sight paths helps considerably. Finally, we were able to obtain some relatively inexpensive surplus parts suitable for use at 900 MHz.

To control channel access, we have chosen to use polling. Since the secondaries cannot in general hear one another, conventional CSMA fails. Performance of pure Aloha, into which it would degenerate, is unacceptable. One alternative that retains CSMA operation would use a repeater at the hub, but the required RF hardware would be more difficult to design and, in our judgment, considerably more expensive. As a repeater, the hub must operate full-duplex. The secondaries must at least transmit and receive on separate frequencies, and they must also be full-duplex if CSMA/CD is desired.

Polling is not free. The polling exchange is an overhead that reduces effective channel capacity and, perhaps more seriously, increases latency. Performance is worst when there are large numbers of secondaries on the polling roll but few of them are offering traffic. This is one reason we believe that the size of the polling list must be limited. With a single channel and half-duplex links, the maximum throughput is half the channel data rate, even if there are no other contenders for the channel. On the other hand, polling has its charms. The effective data rate of the channel actually increases with load and, because of more effective piggybacking, with the number of stations offering traffic. We view polling as an expedient offering acceptable trade-offs at the level of cost and performance that we currently wish to explore, not as an ultimate solution.

4. How Does It Work?

This section explains part of our design in more detail. It focuses on the link level. Many of the design decisions and algorithms are arbitrary; it is easy to think of variants that might have more desirable behavior in one respect or another. We have tried to keep our initial design as simple as possible while providing adequate performance. One reason we want to build a network is to explore just such alternatives.

A. Packet Formats

Packets generally follow the HDLC (ANSI X3.66) frame format. We wanted to use an established framing convention that is compatible with commercial LSI circuits, such

as the popular Zilog 8530 SCC. This hardware provides satisfactory frame delimiting and data transparancy (bit-stuffing). It can also be programmed to do address-based packet filtering in a way that we exploit.

Our frames have the following format:

flag-DA-SA-C-DUA-SUA-I-FCS-flag

Flags (binary 01111110) delimit the packet. Zero stuffing is used to insure that data bytes will not be recognized as flags.

DA and SA, the so-called cluster addresses, are encoded as single bytes. Normally, these are the cluster-relative addresses of the destination and source secondaries respectively. The exact set of values and their intended uses are described in a following section. The SUA and DUA fields contain the call sign and chosen SSID of the source station and the destination station. Each field is eight bytes in length. These are included to satisfy FCC requirements. At the link level, they also uniquely identify the source and destination and serve as cluster-independent addresses.

The C field serves two purposes. It includes a Direction bit and a Poll/Final (P/F) bit. These are used to piggyback polling and completion messages onto other frames. The Direction bit is needed to resolve ambiguities that could otherwise arise when two secondaries can hear each other directly. It is set to one in packets from secondaries to the hub. The rest of the field is a packet type code. Some packet types are reserved for the protocol used to do dynamic cluster configuration; the others are not distinguished at the link level.

The I field is the information field. Its contents are arbitrary. The maximum length of an I field is 1500 bytes. This allows an Ethernet to Hubmaster gateway without packet fragmentation. The FCS field is the 16-bit CCITT V.41 CRC used to check for corrupted bits.

B. Polling Operation

The hub maintains a list of active stations and sequentially considers each one in turn. If the hub has any packets for a secondary, it transmits those packets first and sets the P/F bit of the last packet in the sequence. Otherwise, it sends a minimum length packet with the P/F bit set. This grants control of the RF channel to the secondary, which then sends any waiting traffic to the hub. The secondary marks the end of its traffic by setting the P/F bit in the last packet. The master stores any information frames that it receives and goes on to poll the next secondary. If the traffic is destined for another station in the same cluster, it is delivered as part of polling that secondary. Traffic for an intercluster link is delivered to a process within the hub for forwarding.

The protocol does not limit the number of frames sent in response to a poll. Since the hub can always be jammed by a malfunctioning or malicious station, there is no way to enforce such a limit. The hub can impose a certain amount of flow control by withholding the P/F bit when its buffers are too full. We expect protocols at higher levels to provide the primary flow control.

Similarly, our protocol makes no provision for packet acknowledgment. We believe that acknowledgment, if required at all, is an end-to-end function [8]. Since we are specifically designing the intracluster links for low bit error rates, we do not implement HDLC's sliding window protocol or anything similar at the link level. Packets that arrive with incorrect CRCs are tallied and discarded; the tallies are used to identify and report marginal links.

Delay	Time (μsec)	Notes
transmitter turn-on	10	
preamble transmission	250	8 bytes preceding the opening flag
packet transmission	719 + 31.25N	23 bytes of mandatory packet fields; N = data bytes in the I field
propagation	3.34R	R = cluster radius (km)

Table 1: Delay Times

Polling and packet overheads determine the performance of this protocol. Table 1 shows a breakdown of the time to transfer a single packet. The preamble is required for the clock in the receiver to lock to the received data stream. We have allowed 8 bytes for the 85C30 SCC (NRZI encoding), which we believe to be conservative. The propagation delay is not an important contributor at the data rates and distances being proposed here. We use the delay for a path of 15 km (9.4 miles) in subsequent computations. The time for a packet of minimum length is then 1029 μsec.

The poll of each secondary requires an exchange of minimum-length packets between it and the hub. The secondary must also wait at least 40 μsec before replying, to make sure that the hub has turned its receiver back on. Thus the time for a poll is 2106 μsec per secondary. Because of piggybacking, the first packet of data transmitted in each direction carries no additional overhead; only the I-field expands, and it adds 31.25 μsec per byte. With these observations, we can compute throughput under various assumptions.

Let the number of users on the polling list be S. The throughput is the number of information bits transferred per polling cycle, divided by the time to complete the cycle and further divided by 2, since two transfers are needed per packet delivered. If there are K packets transferred per cycle (all assumed to be piggybacked) and the average size of the I fields in these packets is N bytes, the channel throughput in kbit/s is given by the following formula:

$$\frac{256}{2} \cdot \frac{N \cdot K}{N \cdot K + 67.1 \cdot S}$$

where 67.1 is the polling overhead per secondary expressed in byte times. Some resulting throughputs for each of the K/2 multiplexed transfers are shown in Table 2 (in kbit/s). In entries of the form X/Y, X is the aggregate throughput and Y is the number of unidirectional packet streams.

The first line represents the best case for bulk transactions such as file transfers. The only offered traffic involves a single pair of secondaries, one sending and one receiving maximum length packets. The remaining lines describe what is likely to be more typical operation, with an average packet length of 100 bytes.

		S					
N	**K**	**2**	**4**	**8**	**16**	**32**	**64**
1500	2	122.5	117.5	108.6	94.3	74.6	52.6
1500	8	—	125.2/4	122.5/4	117.5/4	108.6/4	94.2/4
100	2	76.6	54.6	34.7	20.1	10.9	5.7
100	8	—	95.8/4	76.6/4	54.6/4	34.7/4	20.1/4
100	16	—	—	95.8/8	76.6/8	54.6/8	34.7/8

Table 2: Throughput (kbit/s)

	S					
K	**2**	**4**	**8**	**16**	**32**	**64**
0	15	25	46	88	172	340
8	—	—	108	150	234	402
16	—	—	—	213	297	465

Table 3: Latency (msec)

A reasonable measure of latency is the round trip time between two secondaries sending small packets (N = 32) in each direction. This determines how quickly the two can collaborate on demand-response style computations. The best case time is 2 polling cycles plus the final transfer time; the worst case is 3 cycles. Times for 2.5 cycles are shown in Table 3 (in msec). In this tabulation, K is the additional number of 100 byte packets transferred in each cycle.

These numbers and similar calculations show that aggregate throughput actually increases somewhat as the offered load increases, but the fraction available to each user drops. The average throughput per secondary cannot exceed 256/S kbit/s, but with large packets the unidirectional burst rate can approach 128 kbit/s.

Since our link-level protocol does not limit the amount of data exchanged per poll, there is no guaranteed upper bound on latency. As the figures above show, adding users drives up the latency even if they are offering no traffic. Most current applications have modest latency requirements, but low latency is critical to some proposed new ones. Keeping throughput per secondary high and latency low is the reason we hope that clusters will remain small and that overloading will be avoided by forming new ones.

C. Addressing and Routing

Two different kinds of addresses appear in each packet. The addresses SUA and DUA uniquely identify the originating station and the final destination. For our purposes, any identifier that is unique over a global address space consisting of the entire reachable network will do. Since call signs are required in any case, we propose to use them. Other alternatives such as IP addresses or Ethernet's 48-bit unique IDs would be equally suitable. Direct support of TCP/IP by using IP addresses and header formats would be very appealing. Since the performance of the polling protocol is very sensitive to minimum packet size, we have chosen instead to follow the Ethernet model, by requiring encapsulation of IP packets and eliminating the additional overhead at the link level.

Each packet also contains cluster-relative short addresses SA and DA. These cluster addresses are meaningful only within a single cluster (or collection of interconnected clusters administered as a single entity). They form a local address space and are attached to each packet as it enters the cluster. A possible source address (SA) is

- the unique cluster address assigned to an active secondary when it joins the cluster as described in Section D,
- an intercluster link, which is logically another kind of secondary and has a unique local address assigned or reserved by the hub,
- one (or perhaps several) control processes within the hub (for administrative and control packets),
- a null source (used by stations attempting to join the cluster).

A possible destination is any of the possible sources, plus reserved values for

- a broadcast destination ($ff, which is passed by all SCCs),
- a routing agent in the form of a process that runs within the hub.

In traffic sent by the hub (Direction = 0), DA is the selected secondary. SA identifies the secondary that originated the packet; it identifies the hub administrative port (poller) only if there is no traffic to forward. In a packet sent by a secondary (Direction = 1), SA is the cluster address assigned by the hub when the secondary joined the cluster. DA is the cluster address of the destination (local traffic) or of the agent managing the link to the destination (intercluster traffic) when such address is known to the source. Otherwise, it is the reserved address of the router port in the hub.

The cluster-relative address DA is important for two reasons. First, since the hub is broadcasting all traffic, the ability to do hardware-level filtering on DA takes a substantial load off the digital hardware in the secondaries. Secondly, DAs allow simple, fast intracluster switching by the hub. This is always desirable, and it will be essential in the higher performance switches we plan for the future. Since a DA is only 8 bits, forwarding decisions can be made by simple table lookup.

There are a number of ways that a secondary might supply the destination cluster addresses. One possibility is to defer to an intracluster router. To initiate an exchange with a station known only by its DUA, the unknown DA is replaced by a reserved DA that forwards the packet to a router. For intracell traffic, the router does simple lookup. If that fails, the router must choose a cluster address designating an intercell link or gateway (by a policy and mechanism not specified here). It then sets DA to that address and resends the packet. We expect the router to run on the hub hardware or on a computer tightly coupled to it; thus use of the router may add latency but does not generate extra RF traffic.

The SA field of the packet caters to another useful, low cost way of distributing cluster addresses: backward learning [10, pp. 297-298] by the secondaries. When a packet arrives, the secondary extracts the pair (SA, SUA) and stores that pair in some kind of associative memory (e.g., a hash table). Higher level software only understands UA addresses or their equivalents. For each transmitted packet, a low level of the packet driver does the lookup on DUA. If it succeeds (note that it always will after the first packet establishing a conversation), it inserts the correct DA; otherwise, it inserts the router DA.

Because of dynamic configuration, discussed in the next section, cluster addresses can be reused over time, and the (SA, SUA) pairs become stale. To deal with this, each pair is time-stamped when entered or re-entered; pairs with time stamps too old are periodically purged.

In this scheme, packets for the hub are normally not addressed to it, so its SCC must operate with address filtering disabled. Another consequence is this — a secondary that can hear another directly will get packets from it twice. The simplest way of dealing with this is for secondaries to discard all packets with the Direction bit set. We have some ideas for exploiting any direct paths that happen to exist, but they will not be part of our initial experiments.

D. Dynamic Configuration

The polling list is constructed and maintained dynamically. Unless the list has already reached its maximum permitted size, the hub periodically pauses in its polling of active stations and invites new stations to join the cluster. It does this by broadcasting a distinguished packet. Any station receiving such a packet can respond with another distinguished packet type that includes its

SUA. Multiple stations can respond. If the hub detects a collision, i.e., a carrier that cannot be demodulated or a packet with CRC error, it broadcasts that fact. Each station that responded is free to respond again if it first waits a random amount of time and defers to any other station that responds or is acknowledged first. Note that this is CSMA with a very long deaf period, i.e., close to Aloha. We expect cluster membership to change slowly relative to the rate of issuing invitations so that 0 or 1 response will be the norm. If this is not the case, we will use a protocol that converges to 1 response by successively narrowing the range of allowed SUA values.

When a hub hears a successful response to its solicitation, it allocates a cluster address for the new secondary, broadcasts an acknowledgment containing the SUA and SA, and adds the secondary to its polling list. Once a secondary hears such an acknowledgment of another station, it may not respond again until the initial invitation is repeated. If the selected secondary fails to hear the acknowledgment, it will continue to contend; the hub will either assign the same SA the next time it is heard or drop the entry by time-out.

Secondaries leave a cluster either by not responding to several successive polls, or by responding to a poll with an appropriate administrative packet.

Most of the states involved with joining or leaving a cluster have associated time-outs. Our experience indicates these time-outs must be carefully chosen [9]; we defer specifying the details pending some experimentation.

5. Where Are We?

Like many other projects done with spare-time labor and a very limited budget, progress has been erratic and is sometimes slower than we had wished. Our initial goal is to construct and operate 3 clusters in the San Francisco Bay area, one in Santa Rosa and the other two in the San Jose-Fremont-Palo Alto triangle. A prototype of the radio has been constructed and its performance has been demonstrated. We are currently trying to produce PC boards for the pilot run. The digital MIO board is a third generation descendent of the AIO design. At the time of writing, a wire-wrapped version is operational, PCB design is finished, and an initial batch of 3 boards is in fabrication. We also hope to borrow some hardware that will allow us to connect our network to an Ethernet and to experiment with digitized voice.

6. Acknowledgments

Conversations over the past several years with many amateurs helped to shape our ideas. Mike Chepponis K3MC designed the Awesome I/O board upon which the design of MIO is based and helped with the latter. Stu Phillips N6TTO provided much valuable advice on protocols and is developing some of the software.

7. References

[1] Chepponis, M., Elmore, G., Garbee, B., Karn, P. and Rowett, K., "The Implications of High-Speed RF Networking," *8th ARRL Computer Networking Conference Proceedings*, 19-29 [1989].

[2] Chepponis, M., and Mans, B., "A Totally Awesome High-Speed Packet Radio I/O Interface for the IBM PC XT/AT/386 and Macintosh II Computers," *7th ARRL Computer Networking Conference Proceedings*, 36-40 [1988].

[3] Elmore, G. and Rowett, K., "Implementation of a 1 Mbps Packet Data Link Using 10 GHz RF," *8th ARRL Computer Networking Conference Proceedings*, 38-42 [1989].

[4] Elmore, G. and Rowett, K., "Inexpensive Multi-Megabaud Microwave Data Link," *Ham Radio*, December 1989, 9-29 [1989].

[5] Elmore, G., "Physical Layer Considerations in Building a High Speed Amateur Radio Network," *9th ARRL Computer Networking Conference Proceedings* [1990].

[6] Elmore, G., "Prototype 905-MHz Digital Radio Completed," *QEX*, March 1990, 16 [1990].

[7] Reisert, J. H., "How to Design Yagi Antennas," *Ham Radio,* August 1987, 22-31 [1989].

[8] Saltzer, J. H., Reed, D. P., and Clark, D. D., "End-to-End Arguments in System Design," *ACM Transactions on Computer Systems*, November 1984, 277-288 [1984].

[9] Schroeder, M. D., Birrell, A. D., Burrows, M., Murray, H., Needham, R. M., Rodeheffer, T. L., Satterthwaite, E. H., and Thacker, C. P., *Autonet: a High-speed, Self-configuring Local Area Network Using Point-to-Point Links*, Research Report 59, DEC Systems Research Center, Palo Alto, CA [1990].

[10] Tanenbaum, A. S., *Computer Networks (2nd Edition)*, Prentice-Hall [1988].