LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-CONF-216013

A Token Ring Protocol for Dynamic Ad-hoc Wireless Environments

P. Top, V. Kohlhepp, F. Dowla

October 10, 2005

LLNL Internship Program
Livermore, CA, United States
July 1, 2005 through September 15, 2005

A Token Ring Protocol for Dynamic Ad-hoc Wireless Environments

Philip Top, *Member, IEEE,* Virgil Kohlhepp, *Member, IEEE,* Farid Dowla, *Member, IEEE,*

***Abstract*— A wireless ad-hoc networking protocol is presented. The protocol is designed to be flexible, easy to use and adaptable to a wide variety of potential applications. The primary considerations in design are small code size, guaranteed bandwidth access, limited delay, and error resilience in a highly dynamic ad-hoc environment. These considerations are achieved through the use of token ring protocol.**

***Index Terms*— Token Ring, wireless, ad-hoc.**

I. Introduction

WIRELESS networking has become an increasingly popular area for commercial and academic purposes. The existence of wireless networks in our society has become commonplace with the advent of the cellular networks and multitudes of 802.11 access points. Primarily, these networks consist of a base station and an array of mobile devices that typically communicate solely with the base station. Various access control protocols have been proposed [1], nearly all involving some sort of contention scheme for multiple access control. The more recent protocols, as used in cellular networks, rely on a reservation scheme to obtain bandwidth.

Ad-hoc networks, in contrast, do not rely on a base station. Instead of a centralized architecture they use identical nodes to form a network without a fixed layout or infrastructure. The transmission of data is most often accomplished through the use of routing and inquiry messages to establish the structure of the network and create routing tables. This approach works well for situations in which the environment is reasonably static and not changing rapidly, so the routing tables can be assumed accurate without significant energy expenditure for maintenance.

Dynamic, ad-hoc wireless networks present a further challenge in that routing tables are no longer static and increasing amounts of bandwidth are required to maintain the framework. The simplest way to overcome this difficulty is through the use of message flooding. Further enhancements to the flooding protocol can be made to improve it using probabilistic and deterministic approaches. [2] Flooding, though somewhat inefficient, is fairly reliable and simple.

Sensor networks and general communication networks in a highly mobile environment often require a communication system and protocol that is specifically tuned to the application and environment; many existing wireless protocols are standardized and are very difficult or expensive to adapt quickly to specific applications. Thus, the solution is often to utilize a protocol that can be designed quickly or simply patch together components that were optimized for a different task. It is a simple principle that more specifically an engineering solution is to a problem the better it will solve it.

A networking protocol is required that is simple, compact, and easily tunable for specific applications. The protocol must also be capable of delivering guaranteed consistency of access, limited delay, and rapid connectivity under a wide variety of highly dynamic conditions.

Manuscript received month XX, 2005; revised Month XX, 200X. This work was performed under the auspices of the U.S. Department of Energy by UC, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48. The authors are with the Electrical and Electronics Technologies Division Lawrence Livermore National Laboratory, University of California, Livermore, CA 94550 USA (e-mail: top1@llnl.gov; virgilk@llnl.gov;dowla1@llnl.gov).

II. Token Ring Concept

The concept of a token ring protocol for networking was formalized in the wired IEEE 802.4 standard [3]. This protocol grants only the token holder the right to transmit; the token can be kept for a limited amount of time after which it must be passed on to the next node in the ring. The token ring network has to deal with situations in which the token becomes lost or multiple tokens are generated as well as nodes occasionally dropping out or joining the ring. A wireless token ring network operates on similar concepts; the key differences being that the ring is now be a purely virtual concept. Not all nodes are be able to directly communicate with each other which leads to a more complex communication system.

The original wireless token ring protocol was developed by a research team at UC Berkeley [4]. This development was aimed at intelligent Transportation systems and networking moving vehicles with specific data throughput needs. Furthermore, this protocol had more general applicability to ad-hoc networks [5], [6]. Their efforts demonstrated the efficiency of token ring protocol as compared with a 802.11 network, even with the fact that the the token ring protocol was overlayed on top of 802.11 [7]. The details of the protocol operation are available in Ergen's Masters Thesis [8]. This work indicates the potential of the wireless token ring concept to provide the consistent access and limited delay as well as rapid connectivity. Further enhancements were suggested by Deng et. al. [9], [10]. These included variable token holding times, a hibernation mechanism, and a contention period. These improvements were designed to improve power usage and throughput in small scale networks where all nodes can hear each other.

The focus of our work has been to further develop the wireless token ring concept for use in a wide assortment of contexts while expanding its capabilities. Out intent was to

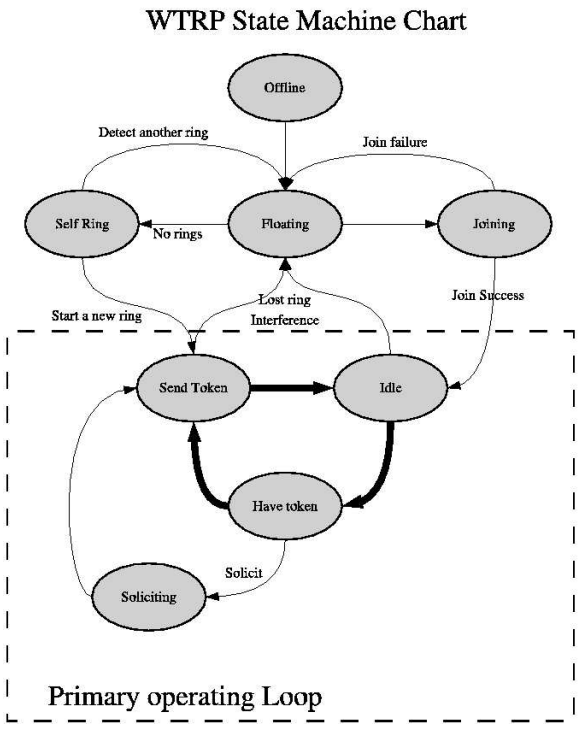

Fig. 1. Flow chart for the wireless token ring protocol

Control Packet Fields

field:	Sync	length	Error Check	Packet Type
size(bytes)	1	1	1	1

Frame Control	NON	RA	DA	SA
2	2	4	4	4

NON: Number of Nodes
RA: Ring Address
DA: Destination Address
SA: Source Address

Fig. 2. Packet format for control packets

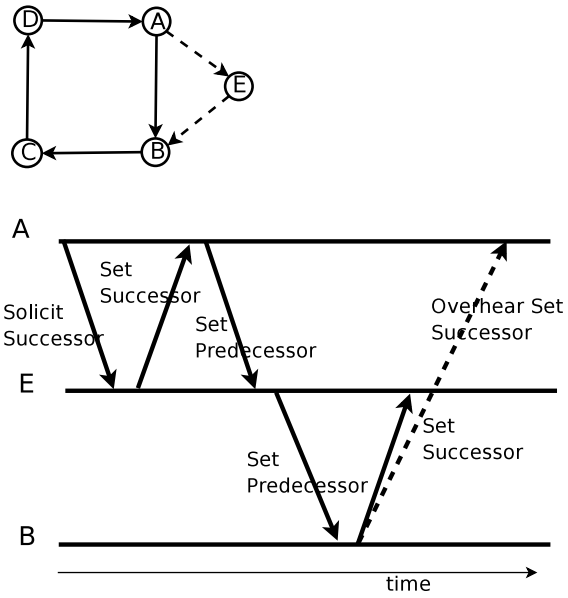

Fig. 3. Joining Procedure

establish a compact, flexible code base, so the protocol could be used easily in a wide assortment of environments including simulation and applications.

III. Protocol operation

At the core of the protocol is a finite state machine in which the function of each state is independent of any previous states. The machine operates in one of eight independent states: offline, floating, joining, self ring, idle, have token, send token, and soliciting. Figure 1 is a flow chart diagram of the state machine operation. Send token, have token, idle and soliciting states comprise the primary operating loop. The names are descriptive of the purposes for these states. The offline state is an initialization or shutdown mode, a node enters the floating state if it loses a ring connection or has just been activated. The joining state is as its name implies and self-ring is the state a node will enter if it has not received any invitations in a set period of time. From the self-ring state a node may start a new ring and invite others to join.

A. control messages

The protocol makes uses eight distinct control messages corresponding to various control operations. The first control message, the token, is used to pass on the right to transmit data. A claim token message is generated by a node accepting ownership of the token. A solicit successor is a broadcast inquiry to any nodes wishing to join the ring. Set successor and set predecessor are used by the joining process and in the event of any change in ring structure. The token deleted message is used to inform a node that it has previously broadcast an invalid token. The channel switch message is used in resolving overlapping ring issues. The ring switch is used for handling multiple simultaneous ring connections. Each one of the control messages has a uniform format shown in figure 2. Beyond this, the token contains a generation sequence field, a sequence field and an attempt number. The claim token contains a field for the new ring address, the set successor and set predecessor contain the address of the node which is the new successor or predecessor. The ring switch and channel switch contain channel number or ring number in place of the number of nodes field. Following these fields all tokens contain a termination byte to mark the end of a packet.

B. joining procedure

In order for a new node to join an existing ring it must be able to connect with the soliciting node and its successor. To ensure this connectivity in a continuously changing environment a sequence of control messages from all three involved nodes is used. The sequence is illustrated in figure 3 There exists a possibility of eliminating some steps from this sequence if some connectivity information is stored. However, in dynamic environments, this is not reliable and would require assumptions and storage for information concerning the validity of the information, or some other method for removing outdated entries. The three node procedure removes the need

to store information while a node is not connected thus ensuring continuity of the ring, even in cases of asymmetric transmission and reception.

The frequency with which a node will solicit for new nodes is determined by how recently the ring structure around a node has changed as well as the data load. During ring formation there are many solicits, whereas in a well established ring solicitation is minimized to better facilitate rapid ring rotations and data transmissions while still allowing rapid reaction to dynamic ring conditions. This feature of the protocol is controllable as to the minimum frequency via compilation variables.

C. Fault tolerance

If the protocol is operating in a dynamic environment it must be able to continue reliable data transmission even with nodes dropping out on a regular basis. This resilience is accomplished through the use of a connection list; all the other nodes that a node is aware of are listed in a ring structure. When any node becomes aware of a new node joining the ring, the node's information is inserted into the virtual ring of the connection table. The procedure for passing the token involves transmitting the token message and waiting for the transmission of any message be it data, a solicit, or passing on the token. This is known as an implicit acknowledgment. The node attempting to pass the token waits a certain amount of time for the implicit acknowledgment; if it does not hear such it will repeat the token transmission a limited number of times. If this limit is exceeded, the node will proceed to try the next node in its list. The above procedure is then repeated for the new node.

In the process of finding the next available node it is possible a large segment of the ring could be cut out of the communication sequence. Thus, the protocol must have a method of very rapidly detecting if a node has been kicked out of the ring. This information is derived by monitoring token transmissions. For example, if node1 notices that its predecessor is transmitting a token that is not destined for node1, or if its successor is transmitting a token that node1 did not previously receive, or if any node is transmitting a token with node1's ring address and a generation sequence number that is at least two higher than the node1's current generation sequence, node1 assumes that it has been dropped from a ring and immediately starts to listen for a solicit to rejoin a ring again. With these procedures the ring is capable of very rapidly reforming itself even during highly dynamic conditions.

D. Multiple ring interference

In partially connected dynamic environments it is probable that multiple small rings will form and then start to interfere with each other. The channel switch control message was conceived to deal with this situation. When a node in a ring detects a token from a ring that it does not recognize it sets a flag. If, after the node has received the token it again detects the interference, it will start a channel switch procedure. The procedure involves first determining which of the rings has a higher priority. Priority is determined by the number of nodes in a ring. The ring with the higher number of nodes has the higher priority. If the node determines that its ring has the higher priority it will transmit a channel switch to the interfering node. Upon reception of a channel switch, and if the receiving node is the ring owner, the node will select a new channel and forward the channel switch message around its ring. The channel switch message is immediately forwarded around the ring without waiting for the token transmission; channel switch takes priority over other transmissions. If multiple channels are available and the interfering ring is larger than a minimum size the ring will then simultaneously switch to a new channel where it does not interfere. If these conditions are not met the ring will simply collapse and nodes that were in the ring will attempt to join the other ring. If the node that detected the interference determined its own ring had the lower priority it will start forwarding the channel switch to its own ring and the same procedure as stated above will occur.

E. Token Faults

In dynamic situations it is possible for the node holding the token to go out of range of the other nodes or to die which effectively loses the token. In this case the nodes who are in the ring will wait for the ring rotation time. If they have not received the token then they will listen for any transmissions; if one is detected, they will assume they got kicked out the ring somehow and try to rejoin. If no transmissions are detected a node will transmit a claim token message. This message is then forwarded around the ring in similar fashion to the channel switch message. The claim token indicates the previous ring address and the new ring address, thus a node will only accept claim tokens from its current ring address. Upon reception of a claim token a node will reset its timers and change its ring address to the new address. If for some reason two tokens are generated on the ring one will be eliminated either by a claim token wiping out the first token or being received by a node that has already received the first token and incremented its generation sequence number. In which case the receiving node will reply to the sending node with a delete token message and the token will then be deleted.

F. Error detection and correction

Although error correction is generally relegated to higher layers in the protocol stack, the token ring layer is designed to detect errors in the important control information. This is accomplished through the use of a checksum that is placed in the error check field of the control message. The error check is computed through by simple addition of the address fields, and sequence numbering, as these are necessary for proper operation of the ring. However, as the number of these fields is dependent on which control message is being sent, the frame control and packet type fields are error corrected to ensure proper processing by the protocol. This provides some additional insurance against corrupted messages causing improper operation of the protocol. The error checksum can be disabled in which case null values are transmitted in the error check field and no comparison is performed on the receiving

Data Header Fields

field:	Sync	length	Error Check	Packet Type
size(bytes)	1	1	1	1

Frame Control	Pass #	RA	DA	SA	OSA	DID
2	2	4	4	4	4	4

RA: Ring Address
DA: Destination Address
SA: Source Address
OSA: Original Source Address
DID: Data Identification

Fig. 4. Packet Format for data packets

end. The token control message itself is sent multiple times if needed to ensure continued operation of the ring. This manner of an error corrected frame control and a checksum in the control messages and data headers allows the protocol to provide functionality in environments with error rates up to 10^{-3}. No error correction or detection is performed on transmitted data, thus if error free data transmission is desired additional error correction would need to be performed on the data itself.

Frame Control Specification for Data Header

FFF MMM PPP PPP XXXX
FFF= forwarding control
010 Forward
101 No Forward
MMM=Acknowledge required
010 No Ack
101 Ack Required
PPP PPP = priority specification
010 010 Low
010 101 Medium
101 010 High
101 101 Urgent
XXXX = Unspecified bits

Fig. 5. Frame Control Specification for Data and Acknowledge Packets

IV. Data Transmission

Data Transmission in a stable, error free environment, in which all nodes are within range of each other presents no difficulty. The data is simply transmitted while the node has the token. However, in partially connected dynamic environments some mechanism must be present for forwarding the data. The protocol implements two types of data headers as shown in figure 4. Both are identical in the header portion. The data packets then further contain an additional synchronization field and a length, followed by the actual data, and terminated by a termination byte. Figure 5 shows an expansion of the frame control format for data transmission. The forwarding portion of the frame control is used in determining whether or not a message should forwarded if needed. This determination is made at transmission time by examining the connection list to see if the recipient node was in range during the last token pass. If the recipient node was in range the field was set in the frame control and the data will not be forwarded. If the recipient node is not within transmission range, the data is forwarded around the ring until it arrives at its destination or it has traversed the entire ring. In the case of asymmetric transmission and reception this feature would be disabled and all data would require forwarding.

The acknowledge message is used to verify reception of the data. This also alerts other nodes to stop forwarding the data packet and remove it from their queues. The token ring protocol allows for a priority specification on the data. The order of transmission is determined by priority; within each priority the order is determined by time of arrival although acknowledgment messages always have a higher transmission priority than data packets. The priority field is used to rapidly ascertain where the data might be listed in the transmission queues so it can be removed if needed. Data packets are also removed from the queues if the packet has been transmitted by the node's successor, as the objective is to propagate the transmission of the data around the ring to the destination a transmission by the successor removes the need for a node to retransmit the data. To accommodate this each data packet generated is given a data identification number. This number is used as an identification in all forwarding situations.

V. Implementation

The protocol was written in the C language for speed and portability. The code is divided up into several sections each corresponding to different operational stages in the code. Each state consists of a function call which processes the current situation and loops until the state changes. Each state function returns a number corresponding to the state that is to follow. The primary loop then simply calls the function corresponding to the next state. In addition to the state functions there are functions for dealing with each of the possible control message types, one for each control message and one for the data and acknowledge packets. The purpose of these functions is to examine the message and alert the state machine to any conditions or situations it needs to know for operation. This communication is done through the use of flag variables. Other blocks of code deal with the data and acknowledge handling, the data structure initialization, and the connection list manager.

The token ring protocol is designed to be operated with minimal external dependencies and links into the outside world via a small set of interaction functions. These simple functions are customized for each different environment in which the protocol operates thereby minimizing deployment costs. All timer lengths, buffer sizes, and optional enhancements are specified in header files so they can easily be modified in order to tune the protocol to a specific application.

This protocol software architecture facilitates optimization and testing of various features and operational parameters. The entire protocol consists of around 4000 lines of code and when

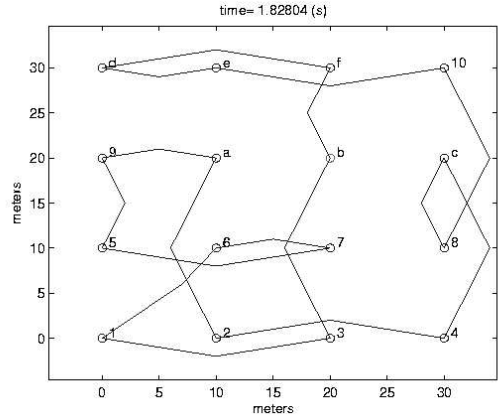

Fig. 6. Ring connections in a partially connected environment

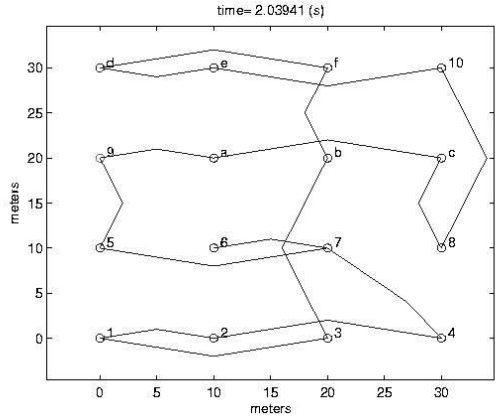

Fig. 8. Restored Ring connection after recovery

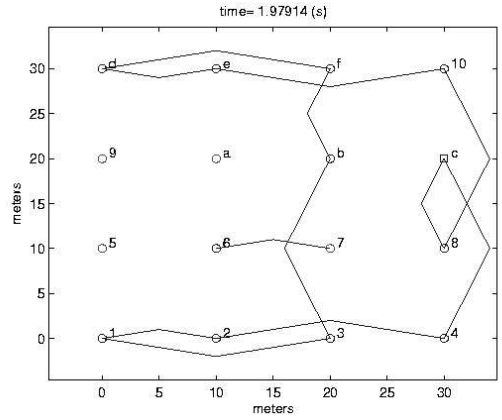

Fig. 7. Ring connections after node 6 has been disabled

compiled is under 30 kilobytes in size when the debugging system(s) are disabled. The compactness and efficiency of the core code allows it to fit and run on a wide variety of platforms from small microcontrollers to pc's.

VI. Simulation

In addition to the core protocol, a simulator was built to allow extensive evaluation and verification. The same code base used for normal operation is also used by the simulator. All interface link files simply operate through the simulator instead of through hardware. The simulator allows the specification of the networking situation through a text file, in which numerous physical parameters for each node and the environment can be controlled. Each node specified is generated as a separate thread and allowed to operate independently of the other nodes only interacting through the simulation communication channels as it would be if they were transmitting over the air. The simulator provides a testing environment through which the protocol can be evaluated over a wide variety of possible situations. Also included with the simulator is a control program through which the user may issue commands that alter the environment or the nodes during the simulation. The control allows such operations as moving a node in and out of range as well as disabling it completely. This facilitates testing of fault recovery mechanisms in the protocol.

The simulator operates through a text display and, at specified intervals, records state files which may be used for visualization. The simulator is not locked to a real-time clock so the time shown in the figure is arbitrary and can be setup to emulate many different operational speeds.

Figure 6 shows a partially connected ring and how it formed to include all the available nodes. The predecessor node of the node that was removed simply attempts to find the path that will minimize the number of the nodes that get cut out yet still maintain the ring. The nodes that do get cut off, detect this and immediately attempt to rejoin the ring. Figure 7 shows the nodes that have been cut out of the loop. Node 6, the one that was cut off from communication, is unable to send or receive information so it assumes it still is connected to the ring, as shown by the connecting line. Figure 8 shows how the protocol reformed after losing a member and reestablished the ring. The scenario depicted here shows statically placed nodes positions in a grid 10 meters apart. The radio transmission range was specified at 21 meters, meaning nodes farther than this distance apart cannot communicate directly.

Various types of movement are possible including linear, circular, and Brownian. Node movement and positioning can be performed in three dimensions, thus allowing simulations involving flying nodes linked with ground nodes or networking in multi-level structures. The transmission power may be changed dynamically along with many other operational parameters. The multi-threaded nature of the simulator depends on the operating system for controlling the threads so exact operation is dependent on the computer operation, which has the effect of randomizing ring layouts and situations each time the simulator is run. This simulation mechanism provides an simple debugging environment through which the protocol can be tested quickly over a wide range of conditions.

VII. Hardware tests

Once the protocol code was debugged, it was quickly ported to other environments. Initial testing was done over a serial link using spread spectrum radios. The protocol was also tested under a variety of conditions using a group of infrared serial links based on the Air-Byte Serial Infrared Transceiver manufactured by Reynolds Electronics [11]. The IR serial

links have no low level multiple access control thus allowing full testing of the protocol. The infrared communication link also facilitates testing of node failure and various startup and operating conditions including partially connected networks and asymmetric transmission situations by simply blocking the line of sight of various nodes from each other.

The test network was set up in an office environment that included people intermittently blocking signals and various transmission impediments that often disrupted the link(s). In addition, the orientation of the IR transceivers tended to create random transmission errors when poorly aligned. This provided a relatively thorough test of the limits of the error resiliency of the protocol.

Our serial links operate at 2400 baud; with stop, start and parity bits this translates to peak data speed of roughly 1600 bits per second. The slow speed allows manual intervention and detailed display of the progress of the token and data transmission. Several tests were run over this network. Typically, a test was set up with a specific number of nodes; each node configured to generate 40 bytes of data for transmission at each token rotation. The data was transmitted on a general broadcast address. The protocol's forwarding mechanism was enabled, so the data was repeated around the ring. The number of transmissions depends on the number of nodes in the ring. In a fully connected network the number of transmissions of single data packet is equal to the number of nodes minus one, as nodes will not retransmit data which was transmitted by their successor. Figure 9 shows the rotation times with increasing number of nodes. The cases with two, three, and four nodes represent fully connected networks. The five node case is partially connected case in which two nodes could hear all other nodes, two nodes could hear three others, and the fifth node could only hear two others. Figure 9 also shows the comparison between the IR testbed results and the simulation results. The slightly slower rotation times of the IR testbed are attributed to computer switching times and packet errors resulting in repeat token transmissions. Figure 10 shows the distributions of the rotations times from the simulator and IR testbed. There are several distinct times, the slower times are the result of going through a solicit process, the faster periods are the result of a missed data transmission resulting in not forwarding the data as would otherwise be the case.

VIII. PROTOCOL TIMING

The time which a node can keep the token is determined by two factors. The first is that time must be allowed to complete the joining procedure of Figure 3. This is determined by the transmission times of the messages and the necessary waiting periods for each step. The second factor is determined by the desired data packet transmission time, including the length of each packet and how many data packets any node should be allowed to send during the token holding time. The actual longest token hold time would be determined by the maximum of these two factors. Table I shows the maximum holding times for needed for soliciting by each node with various transmission speeds. The maximum ring rotation time is then computed by multiplying the token block time by the maximum number of nodes. The numbers in the table represent a maximum and would only be reached in cases of soliciting and peak data loads, they also would not be the case for all nodes in a ring, except possibly during initial ring formation.

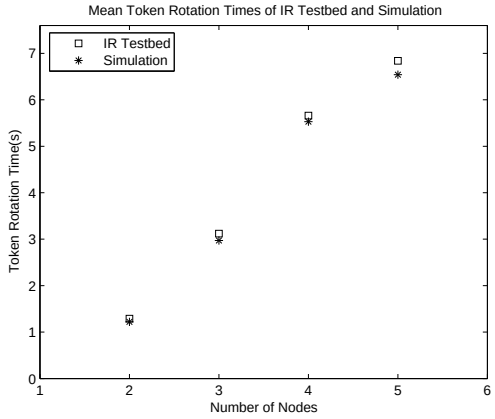

Fig. 9. Token Rotation times for various number of nodes in a ring

Fig. 10. Individual token Rotation times for IR Testbed and Simulator

IX. CONCLUSION

The wireless token ring protocol developed here provides a mechanism for establishing wireless networks in a fast efficient manner. The token ring concept inherently provides guaranteed access by all members once a ring has been established, and by the same mechanism provides a limited throughput delay.

Speed	Token Block Time
2000 bps	1.8 s
10 Kbps	360 ms
50 Kbps	72 ms
100 Kbps	36 ms
250 Kbps	14.4 ms
1 Mbps	3.6 ms
8 Mbps	450 μs
50 Mbps	72 μs

TABLE I

TOKEN BLOCK TIMES FOR VARIOUS TRANSMISSION SPEEDS

The protocol is designed to be compact, fast, and quickly adapted to operate on a wide assortment of platforms and applications. Testing of the protocol with a network of IR transceivers and a network simulator demonstrated the ring formation and operation as well as the error resiliency of the protocol.

REFERENCES

[1] I. F. Akyildiz, J. McNair, L. C. Martorell, R. Puigjaner, and Y. Yesha, "Medium access control protocols for multimedia traffi c in wireless networks," *IEEE Network*, vol. 13, no. 4, pp. 39–47, July.

[2] J. Wu and F. Dai, "A generic distributed broadcast scheme in ad hoc wireless networks," *IEEE transactions on Computers*, vol. 53, no. 10, pp. 1343–1354, Oct.

[3] *International Standard ISO IEC8802-4:1990*, ANSI/IEEE Std. 802.4, 1990.

[4] D. Lee, R. Attias, A. Puri, R. Sengupta, S. Triptikas, and P. Varaiya, "A wireless token ring protocol for intelligent transportation systems," in *Proceeding of the IEEE 4th International Conference on Intelligent Transportation Systems*, Aug. 2001, pp. 1152–1157.

[5] M. Ergen, D. Lee, A. Puri, P. Varaiya, R. Sengupta, R. Attias, and S. Tripakis, "Wireless token ring protocol," *IEEE Transactions on Wireless Technology*, vol. 53, no. 6, Nov. 2004.

[6] D. Lee, A. Puri, P. Varaiya, R. Sengupta, R. Attias, and S. Tripakis, "Wireless token ring protocol for ad-hoc networks," in *IEEE Aerospace Conference Proceedings*, vol. 3, Mar. 2002, pp. 1219–1228.

[7] M. Ergen, D. Lee, R. Sengupta, and P. Varaiya, "Wireless token ring protocol-performance comparison with ieee 802.11," in *Proceedings of the Eigth IEEE International Symposium on Computers and Communication*, vol. 3, July 2003, pp. 710–715.

[8] M. Ergen, "WTRP-Wireless token ring protocol," 2002. [Online]. Available: citeseer.ist.psu.edu/ergen02wtrpwireless.html

[9] Z. Deng, Y. Lu, C. Wang, and W. Wang, "EWTRP: enhanced wireless token ring protocol for small-scale wireless ad hoc networks," in *Proceeding of the International Conference on Communications, Circuits, and Systems*, vol. 1, June 2004, pp. 398–401.

[10] ——, "E^2WTRP: An energy-effi cient wireless token ring protocol," in *Proceeding of the IEEE conference on Personal, Indoor, and Mobile Radio Communications*, vol. 1, Sept. 2004, pp. 574–578.

[11] [Online]. Available: http://www.rentron.com/Products/Air-Byte.htm

PLACE PHOTO HERE

Philip Top Philip Top received his BS in Engineering from Dordt College in Sioux Center, IA in 2002, his MS in Engineering from Purdue University in 2004, and is currently working on Ph.D. at Purdue University. He has been working with Lawrence Livermore Laboratory since 2002 on project relating to Wireless Networking and Ultrawideband communications. Research interests include Wireless Networking, Power Systems monitoring and control, and Genomic Signal Processing.

Virgil Kohlhepp Biography text here.

PLACE PHOTO HERE

Farid Dowla