RT-Link: A Time-Synchronized Link Protocol for Energy Constrained Multi-hop Wireless Networks

Anthony Rowe, Rahul Mangharam, Raj Rajkumar
Electrical and Computer Engineering Department, Carnegie Mellon University, Pittsburgh
Technical Report CMU-ECE-TR05-08

Abstract

Multi-hop wireless networks of embedded nodes facilitate applications in industrial control, surveillance and inventory tracking. Our focus is on low-cost large-scale deployments where nodes need to be battery-powered with predictable network lifetimes and applications require bounded end-to-end delay. An effective approach to such energy-constrained networks is to operate at low duty cycles and maximize the shutdown interval between packet exchanges. The primary challenge is in coordinating transmissions so they are collision-free while minimizing the duration the nodes are active. RT-Link is a time-synchronized link protocol for fixed and mobile embedded radios. We identify three key observations in the design and deployment of RT-Link: (a) RT-Link offers predictable network lifetime with bounded end-to-end delay. (b) Achieving global time synchronization is both economical and convenient for indoor and outdoor deployments. (c) Due to interference between nodes, our experiments confirm that nodes with the same schedule must be spaced by a minimum of 3 hops. Furthermore, to minimize end-to-end delay, it is more important to order time slots than to minimize the number of time slots. RT-Link has been deployed on networks with more than 30 custom embedded nodes and uses the IEEE 802.14.5 physical layer. It outperforms energy-efficient protocols such as B-MAC and S-MAC in throughput, energy consumption and end-to-end delay.

1. Introduction

Networks of embedded wireless nodes provide a versatile platform for applications in industrial control, surveillance and inventory tracking. The purpose of such networks is to gather data and deliver it across one or more hops to at least one gateway. The principal requirements are low-cost battery powered radios, minimal configuration on set up with simple and scalable energy-efficient protocols for predictable network lifetime and bounded end-to-end message delay. The following deployment scenarios motivate the need for networks of embedded nodes with such requirements:

- Industrial Control Networks: In chemical and automobile plants, remote control of machinery and access to performance data requires reliable real-time communication. In such environments, it is necessary that nodes do not require infrastructure for data and power as such provisioning may be both impractical and expensive.
- Surveillance and Monitoring Networks: Networks of embedded cameras for monitoring motion and intrusion require bounded end-to-end delay to the gateway and deterministic peak throughput for intermittent transfer of captured images.
- Inventory Tracking and Reporting: Networks to classify and locate assets need to be scalable and must operate in a variety of multi-hop wireless topologies.

An effective approach to energy-efficient service for applications with periodic and aperiodic flows is to operate all nodes at low duty cycles so as to maximize the shutdown intervals between packet exchanges. The two fundamental challenges in delivering delay-bounded service in such networks are (a) coordinating transmissions so that all active nodes communicate in a tightly synchronized manner and (b) ensuring all transmissions are collision-free. Time synchronization is important because it tightly packs the activity of all nodes so that they may maximize a common sleep interval between activities. Furthermore, it provides guarantees on timeliness, throughput and network lifetime for end-to-end communication. Such assurances are only possible when the link is reliable and collision-free. It is therefore the responsibility of the link layer protocol to provide exclusive and interference-free access to the shared wireless channel and a mechanism to coordinate sleep intervals of all nodes. The focus of this paper is on RT-Link, a time-synchronized link layer protocol for collision-free and energy-efficient real-time service over multi-hop wireless networks. RT-Link facilitates dynamic admission of both fixed and mobile nodes into a tightly synchronized regime. It schedules nodes in time slots such that concurrent transmitters do not interfere with each other and the activity of all nodes are coordinated to maximize the sleep duration. Finally, RT-Link maintains contention-free operation by employing an online and automatic link conflict detection and resolution scheme. Such a scheme is useful

when topology or environment changes cause interference to nodes between concurrent transmitters. RT-Link has been implemented as a link layer protocol in low-cost and low-power embedded nodes developed by us. Each node includes a short-range 2.4GHz IEEE 802.15.4 [1] physical layer for radio communication. Through the design and deployment of RT-Link, we identify the following four observations:

1. RT-Link offers predictable network lifetime with bounded end-to-end delay for packets between the gateway and every node.

2. Provision of global time synchronization for embedded multi-hop wireless networks is both economical and convenient. We achieve this by employing an Amplitude Modulation (AM) based carrier-current method indoors and with atomic clock receivers for the outdoors.

3. Our experiments show that due to interference across the shared channel, nodes with the same schedule (i.e. concurrent transmitters) must be spaced by a minimum 3-hop distance.

4. In high throughput networks, the scheduling objective is to maximize the number of concurrent transmitters [2]. In contrast, in energy-efficient sensor networks, the ordering of the time slots is more important than the number of time slots.

The paper is organized as follows: we address related work in the next section followed by a description of the RT-Link protocol in Section 3. We study the timeliness, robustness and efficiency of the protocol in Section 4. Section 5 provides an overview of our implementation platform and deployment experiences. This is followed by a comparative evaluation of RT-Link in Section 6 and our concluding remarks.

2. Related Work

Several MAC protocols have been proposed for low-power and distributed operation for single and multi-hop wireless mesh networks. Such protocols may be categorized by their use of time synchronization as asynchronous, loosely synchronous and fully synchronized protocols. In general, with a greater degree of synchronization between nodes, packet delivery is more energy-efficient due to the minimization of idle listening when there is no communication, better collision avoidance and elimination of overhearing of neighbor conversations. We briefly review key low-power link protocols based on their support for low-power listen, multi-hop operation with hidden terminal avoidance, scalability with node degree and offered load.

2.1. Asynchronous Link Protocols

The Berkeley MAC (B-MAC) [3] protocol performs the best in terms of energy conservation and simplicity in design. B-MAC supports Low Power Listening (LPL) where each node periodically wakes up after a sample interval and checks the channel for activity for a short duration of 2.5ms. If the channel is found to be active, the node stays awake to receive the payload following an extended preamble. Using this scheme, nodes may efficiently check for neighbor activity. The major drawback of B-MAC is that the transmitter must remain active for the duration of the sampling interval for each sent packet. For example, if receiver nodes periodically wake up every 800ms, then a transmitter would need to continuously transmit for 800ms to be detected by a neighbor. This coupling of the receiver's sampling interval and the duration of the transmitter's preamble severely restricts the scalability of B-MAC when operating in dense networks and across multiple hops. B-MAC does not inherently support collision avoidance due to the hidden terminal problem and the use of RTS-CTS handshaking is expensive and inefficient. In a multi-hop network, it is necessary to use topology-aware packet scheduling for collision avoidance. Furthermore, upon wake up, B-MAC employs Carrier Sense Multiple Access (CSMA) and is prone to wasting energy and adding non-deterministic latency due to packet collisions.

2.2. Loosely Synchronous Link Protocols

Protocols such as T-MAC [4], WiseMAC [5], and S-MAC [6] employ local sleep-wake schedules between node pairs to coordinate packet exchanges while reducing idle operation. All three schemes exchange synchronizing packets to inform their neighbors of the interval until their next activity and use CSMA prior to transmissions. Both T-MAC and WiseMAC use LPL to minimize energy consumption during channel sampling. WiseMAC, however, is designed for point-to-multipoint communication and does not cater to multi-hop networks. S-MAC is similar to but simpler than T-MAC, but does not implement LPL. Both schemes do not scale well because all the neighbors of a node cannot hear each other and this forces the node to set multiple wake up schedules for different groups of neighbors. Furthermore, the use of CSMA degrades performance severely with increasing node degree and traffic.

2.3. Fully Synchronized Link Protocols

With the provision of global time synchronization, TDMA protocols such as TRAMA [7] are able to communicate between node pairs in dedicated time slots. TRAMA supports both scheduled slots and CSMA-based contention slots for node admission and network management. RT-Link has similar support for contention slots but employs Slotted-ALOHA [8] rather than CSMA as it is more energy efficient with LPL. While TRAMA outlines a schedule exchange protocol, it does not explicitly specify a node scheduling scheme. The authors do not address an energy-efficient and practical time synchronization scheme. RT-Link has been inspired by systems such as [9, 10] which used dual-radio solutions for low power wake-up. However neither system has been used for time synchronized operation. RT-Link employs tight global time synchronization to

establish a common wake-up instance and duration for all nodes. The procedure after wake up is time synchronized into scheduled transmission slots and thereby eliminates any collisions in the network. To the best of our knowledge, RT-Link presents the first deployment of globally synchronized low-power sensor networks with energy-efficient and economical time synchronization

3. RT-Link Protocol Design

RT-Link is a Time Division Multiple Access (TDMA) based link layer protocol designed for networks that require predictability in throughput, latency and energy consumption. All packet exchanges occur in well-defined time slots. Global time synchronization is provided to all fixed nodes by a robust and low-cost out-of-band channel. We describe in detail the RT-Link protocol and its operation modes.

3.1. Protocol Overview

Each fixed (i.e. stationary) node has two radios which operate on separate channels. A low-power receiver (e.g. AM, FM, atomic clock) to detect a periodic synchronization pulse and an RF transceiver for data communication after the sync pulse is received. As shown in Figure 1, a periodic synchronization pulse is detected first and is followed by a finely slotted data communication period. This period is defined as a frame and the interval between time sync pulses is defined as a cycle. The sync pulse serves as an indicator of the beginning of the first frame. After a frame is complete, each node schedules a timer to wake up just before the expected time of the next sync pulse and promptly switches to sleep mode. One or more frames may be scheduled within a cycle. Each frame is divided into two regimes of time synchronized operation: (a) a series of Scheduled Slots (SS) within which nodes are assigned specific transmit and receive time slots and (b) a series of Unscheduled or Contention Slots (CS) where nodes, which are not assigned slots in the SS, select a transmit slot at random. Nodes operating in SS are provided timeliness guarantees as they are granted exclusive access of the shared channel and hence enjoy the privilege of interference-free and hence collision-free communication. Fixed nodes that have not yet been assigned a time slot in SS continue to operate in CS and are subject to a finite probability of packet collision. We assume all nodes are aware of the fixed number of slots within the SS and CS. The methods for assigning collision-free time slots are described in Section 4.

3.2. Node Types and Packet Types

RT-Link supports both fixed and mobile nodes. Only fixed nodes have a sync pulse receiver in addition to the RF data transceiver and are able to operate within SS. Fixed nodes maintain time synchronization by the global sync pulse and may be assigned specific transmit and receive time slots in the interval following the sync pulse. On the other hand, mobile nodes are not assigned specific scheduled time slots as their neighbors may change frequently and therefore operate

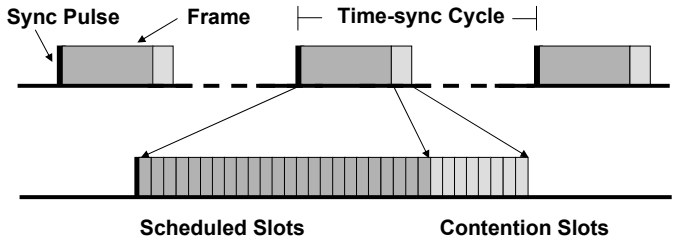


Figure 1. RT-Link time slot allocation with out-of-band synchronization pulses

solely in CS. Mobile nodes obtain time synchronization by listening to neighbors operating in SS. Upon detecting one such packet, the mobile node is informed of the current slot number and determines the start of the CS. It then randomly selects a transmit slot in the CS. Only nodes operating in the SS are permitted to route data to and from the gateway. Mobile nodes, typically broadcast data to the gateway via one or more fixed nodes. We now describe the six packet types shown in Figure 2 required for basic network operation.

1. RT-Link Packet Header
   Every packet includes the common link layer header. The common header, as shown in Figure 2(a), contains 16-bit Medium Access Control (MAC) addresses of the source, destination and current forwarding node. As fixed nodes operate either in the SS or CS, each packet is tagged explicitly with the transmit slot number. For example, in our network deployment, a cycle contains 32 slots including 24 (i.e. slots 0 to 23) slots within the SS and 8 slots within the CS. As guaranteed service is provided only to nodes assigned explicit slots within the SS, only these nodes are entitled to receive an acknowledgement. The receiver operating within SS implicitly acknowledges all packets received by setting a bit mask mapping the slots within which the packets were received. This provides an efficient mechanism to identify and acknowledge transmitters operating within the SS. Mobile nodes and nodes operating within the CS, only transmit broadcast packets destined for the gateway and are hence never acknowledged.

2. HELLO Packet
   The HELLO broadcast packet, as shown in Figure 2(b), advertises a node's list of neighbors. This packet serves as a keep-alive packet to inform neighbors of one's continued presence and also informs the gateway of the network topology. A node is considered a neighbor as long as at least one HELLO message is received from it within the past k (e.g. 5) cycles.

3. SCHEDULE Packet
   When the gateway receives multiple HELLO packets from a node and is satisfied about the stability of a

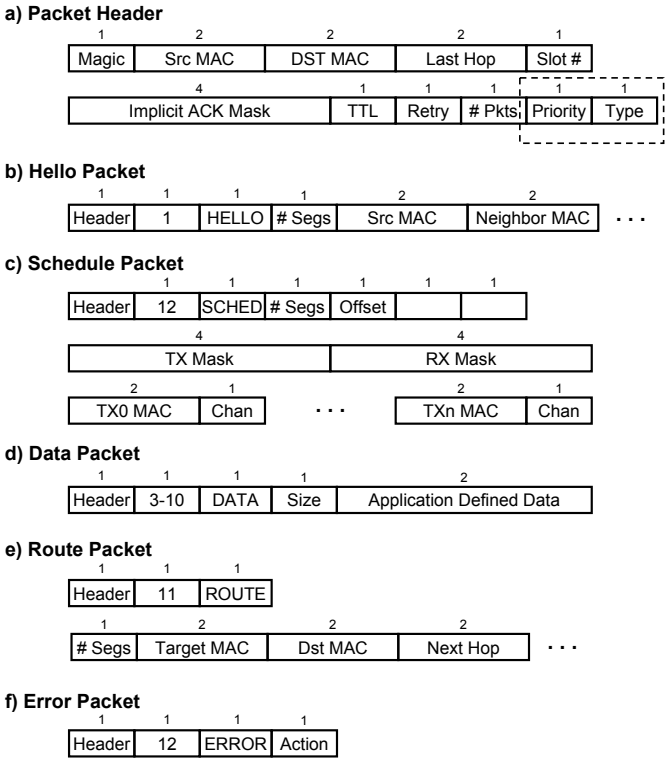


Figure 2. Six packet types required for basic operation.

node's neighborhood, it sends a unicast SCHEDULE packet Figure 2(c) to the node with its unique slot numbers. A node may be assigned one or more slots depending on a higher-level attribute such as application bandwidth requirements. The gateway executes a node coloring algorithm to determine the schedule for all known nodes in the network such that concurrently active nodes do not interfere with each other. The node scheduling algorithm is a network layer service and is out of the scope of this paper. However, in section 4 and 5 we provide insights and describe a basic algorithm for leaf to gateway tree data collection.

A node's schedule is described by transmit and receive bit-masks. A node is active in only those slots set in each mask. The radio is turned off in all other slots. This enables a node to avoid listening to neighboring nodes through which no common flow is routed. In our implementation, we use 32-bit masks for transmit and receive schedules.

4. DATA, ROUTE and ERROR Packets
The DATA packet (Figure 2(d)) can contain a maximum of 100 bytes of data as the maximum size of an IEEE 802.15.4 packet is 128 bytes. DATA packets support packet aggregation and forward aggregated data with the sender's address. The ROUTE packet Figure 2(e) is sent to a node via unicast from the gateway to explicitly set routes. We only provide primitive support for source routing as the focus is on link communication. The ERROR packet (Figure 2(f)) is sent to neighbors when a conflict, due to overlapping time slots among neighbors, is detected. A node experiencing the conflict informs its neighbors and the gateway of the error and may choose to relinquish its slots. A node relinquishing a slot needs to restart the association procedure (described below) to request one or more scheduled slots from the gateway. The ERROR packet informs the gateway of a scheduling conflict. Each packet is assigned a priority, which may be based on the last-hop address or packet type. When more packets have accumulated at a node than its available buffer space, packets are dropped in the ascending order of their priority. Packet priorities are useful in aggregation and the highest priority among the aggregated packets is assigned to the entire forwarded packet.

3.3. Network Operation Procedures

Given the general slot structure, we now describe the rules a node follows upon association, scheduled operation, during slot assignment conflicts and disassociation.

3.3.1. Network Association and Disassociation

RT-Link operates on a simple 3-state state machine as shown in Figure 3. In general, nodes operating in the CS are considered Guests, while nodes with scheduled slots are considered Members of the network. A fixed node that is currently a Guest becomes a Member once it is assigned one or more slots in the SS. On the other hand, mobile nodes are never assigned a scheduled slot and are considered Guests for the lifetime of their operation.

When a fixed node is powered on, it is first initialized as a Guest and operates in the CS. It initially keeps its sync radio receiver on until it receives a sync pulse. Following this, it waits for a set number of slots (spanning the SS) and then randomly selects a slot among the CS to send a HELLO message with its node ID. This message is then forwarded via flooding to the gateway and the node is eventually scheduled a slot in the SS. In a bootstrapping manner, Guest nodes closer to the gateway should be scheduled first in order to allow faster delivery of scheduling packets further away from the gateway.

During normal operation, Members and Guest wake up before the expected instance of the sync pulse and operate in the SS and CS. On the other hand, when a mobile node needs to transmit, it first stays on until it overhears a neighbor operate in its SS. The mobile node achieves synchronization by observing the Member's slot number and computes the time until the start of the CS. It then randomly selects a slot in the CS and transmits. The mobile node remains silent until a Member is identified. During normal operation, all nodes transmit on their assigned slots within the SS, and turn on their receiver during the receive slots from each neighbor.

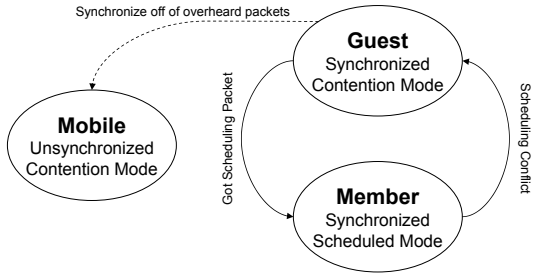


Figure 3. RT-Link node state machine.

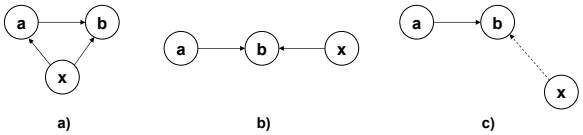


Figure 4. Scheduling Conflict Cases. Node a is trying to communicate with node b, while node x shares b's timeslot. The dotted line indicates an interfering link that cannot send valid data.

All nodes with scheduled slots listen on every slot in the CS. As described in Section 5, all nodes use a low power listen mechanism to quickly and economically detect if there is activity in the beginning of a slot and turn the receiver off if there is none.

When a node chooses to leave the network, it ceases broadcasting HELLO packets and is gracefully evicted from the neighbor list from each of its neighbors. The gateway eventually detects the absence of the departed node from each of the neighbors' HELLO updates and may reschedule the network if necessary.

3.3.2. Conflict Detection and Resolution

While all nodes are expected to operate in a tightly synchronized manner, there are several cases when the slot assignment results in conflicts. A synchronization conflict occurs when two neighbors within communication or interference range are assigned overlapping or partially overlapping time slots. Such cases may occur due to topology changes, some nodes experiencing extremely large clock drift or jitter, and incorrect slot assignment by the gateway, among other reasons. Figure 4 illustrates three potential conflict cases. In Figure 4(a), we observe that both node A and X are assigned the same slot resulting in collisions at node B. In this case, X is also within the communication range of A. In Figure 4(b), we see a similar case, except X is hidden from A. Finally, in Figure 4(c) while B is within A's communication range, it is also within X's interference range. B is unable to decode A's packets as X's interference raises the noise floor.

We present two mechanisms, Active Listener and Active Transmitter, to detect and correct the above timing errors. The Active Listener instructs nodes to actively listen on their transmit slot when they do not have any data to send. This way, a node is able to overhear a potential interferer and detect the conflict. For example, in the case of Figure 4(a), if node A were quiet in its transmission slot, it would overhear node X's transmission.

The Active Transmitter approach enables both conflict detection and resolution. During every kth cycle, e.g. k=5, a scheduled node broadcasts a packet with its neighbor list and slot assignment of each neighbor from a slot within CS. This informs all neighbors of timing conflicts with potential hidden terminals. For example, in the case of Figure 4(b), if B broadcasts its node list, then it would list both A and X as owners of the same receive slot. Either A or X may be nominated to relinquish their slot and re-associate with a new slot number.

In the final case, as B is able to only decode A's packets but suffers high packet loss due to interference from X, it requests A to relinquish its slot and re-associate by sending a HELLO packet to the gateway with its previous slot as a forbidden slot.

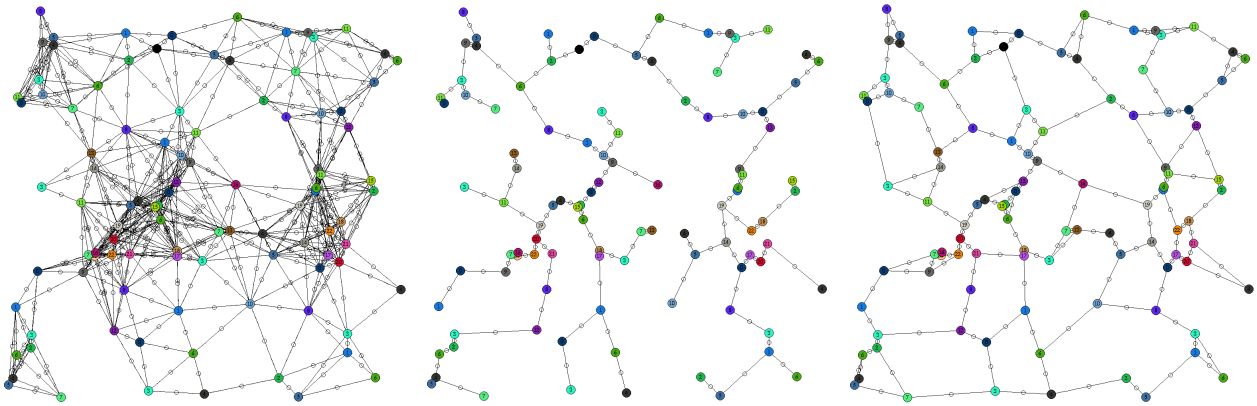
4. RT-Link Protocol Enhancements

In this section we briefly discuss enhancements that compliment the basic RT-Link protocol. These may be executed prior to or during network deployment to improve the overall throughput, end-to-end latency and network lifetime.

4.1. Topology-based Predictable Lifetime

We have developed a hybrid network simulator to predict network lifetime for a given topology and offered application traffic. The network topology may be generated by the simulator or be acquired from a deployed network. In order to acquire the network connectivity graph, we deploy an actual network and aggregate the neighbor lists from each node at the gateway. We then construct connectivity and interference graphs and schedule nodes based on k-hop coloring, such that two nodes with the same slot schedule are mutually separated by at least k+1 hops. The time-triggered simulator then simulates a given traffic workload during each cycle and calculates the worst -case network lifetime. The network lifetime is defined as the time until the first node completely consumes the available battery energy and disconnects the graph (i.e. a bottleneck node). Through this exercise, we are able to answer questions such as, "What is the expected lifetime for the given topology?" and alternatively, "What is the best topology for the given application demands?" we illustrate this point by calculating the expected lifetime for three different network topologies.

In Figure 5(a) we show the connectivity graph of a randomly generated topology with 100 nodes. The graph was colored based on the connectivity to ensure that it is free of collisions. Links can then be removed by instructing an adjacent node to no longer wakeup to listen on that particular timeslot. Using this principle, we generate a spanning tree starting from the gateway shown in Figure 5(b) that guarantees full connectivity. One could either measure network lifetime as the time before the first node runs out of power,



(a) Network Connectivity Graph (b) Spanning Tree Cover of Network (c) Network with Balanced Degree of 3

Figure 5. Network Graph Constructed For Energy Efficiency

or the time at which the network connectivity falls below a specific threshold. In either case, adding a higher degree of redundancy without increasing the maximum degree of the network will increase total lifetime. Figure 5(c) shows the spanning tree topology with links added back into the graph such that whenever possible nodes have a degree equal to the maximium degree of the original spanning tree. The original network has a simulated lifetime of 1.2 years, while the prunned networks do not experience a node loss for 2.2 years. We observe that well-balanced networks with uniform node degree outperform networks with asymmetric or highly dense sub graphs. Well-balanced graphs efficiently exploit spatial reuse and evenly spread the network load.

4.2. Interference-free Node Scheduling

In order to achieve high throughput in a multi-hop wireless network, it is necessary to minimize the number of collisions along each transmission hop. This problem has traditionally been solved as a distance-k node coloring (slot scheduling). To determine the interference range of a node, we placed a set of nodes along a line in an open field and first measured the packet loss between a transmitter and receiver as the transmitter's distance was varied. Once the stable communication distance between a transmitter and receiver was determined, we evaluated the effect of a constantly transmitting node (i.e. a jammer) on the receiver. Our experimental results for stable transmit power level 8 are shown in Figure 6. We notice that 100% or the packets are received up to a transmitter-receiver distance of 10m. Following this, we placed the transmitter at a distance of 2, 4, 6, 8, 10 and 12 meters and for each transmitter position, a jammer was placed at various distances. At each point, the transmitter sent one packet every cycle to the receiver for 2000 cycles. We measure the impact of the jammer by observing the percentage of successfully received packets. We observe two effects of the jammer: First, the effect of the jammer is largely a function of the distance of the jammer from the receiver and not of the transmitter from the receiver. Between 12-18 meters,

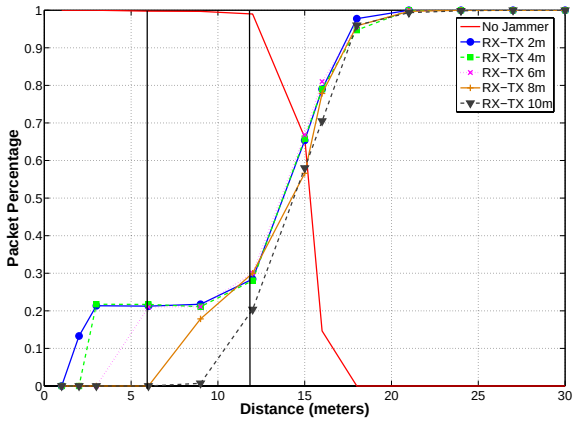


Figure 6. Packet Success rate while transmitting in a collision domain.

the impact of the jammer is similar across all transmitter distances. Second, when the transmitter and jammer are close to the receiver, (i.e. under 9m), the transmitter demonstrates a capture effect and maintains an approximately 20% packet reception rate. The above results show that the jammer has no impact beyond twice the stable reception distance (i.e. 20m) and a concurrent transmitter may be placed at thrice the stable reception distance (i.e. 30m). Such parameters are incorporated by the node coloring algorithm in the gateway to determine collision-free slot schedules. Results for a scheduled multi-hop network are presented in Section 6.

4.3. Coloring and Ordering

In multi-hop wireless networks, the goal for higher throughput has traditionally been approached from the perspective of maximizing the set of concurrent transmitters in the network [2]. This is achieved either by scheduling nodes or links such that they operate without any collisions. In the networks considered here, the applications generate steady

or low data rate flows but require low end-to-end delay.

In Figure 7 we see two schedules, one with the minimal number of timeslots, the other containing extra slots but provisioned such that leaf nodes deliver data to the gateway in a single TDMA cycle. The minimal timeslot schedule maximizes concurrent transmissions, but does not minimize the latency of all nodes. Instead, the maximal concurrency schedule will cause quequeing delays that will hurt overall network performance given all nodes equally contribute traffic.

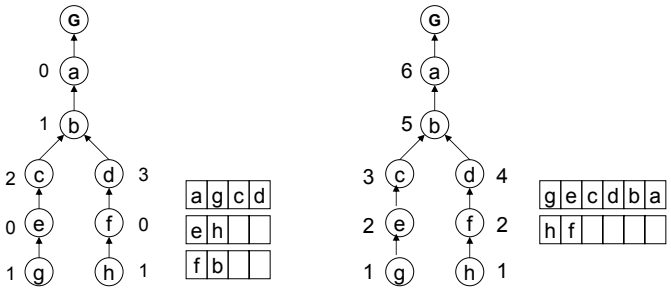


Figure 7. Maximal concurrency schedule (left) compared to a delay sensitive schedule (right). Note that the maximal concurrency schedule needs two frames to deliver all data. Even if the schedule is duplicated, it will still require two extra cycles compared to the delay sensitive schedule.

In Figure 8, we illustrate a set of nodes communicating to and from a gateway G. By assigning the time slots liberally and in preference to faster uplink and downlink routes, we show that for networks with delay-sensitive data, ordering of slots should take priority over maximizing spatial reuse. As nodes sleep between slots not assigned in their transmit or receive bit-masks, the energy saving in using fewer slots as compared to all available slots in a cycle is nominal. The best strategy is to use as many slots required to minimize the end-to-end delay along both the uplink and downlink.

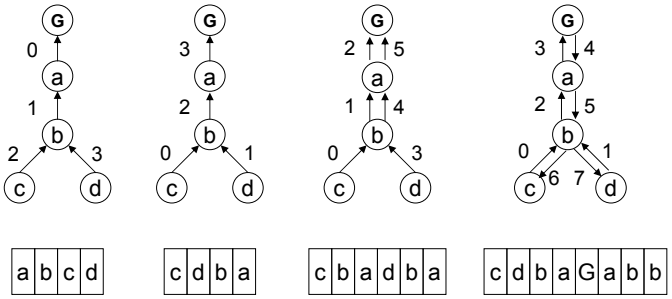


Figure 8. A set of nodes communicating with different schedules.

The generation of schedules is similar to the distance-two graph coloring problem that is known to be NP-complete [11]. In practice, many heuristics can work well given that deviations from the optimal color selection are typically overcome by excess time slots. Though not the focus of this paper, we will discuss one such heuristic that provides a basic scheduling ability for networks where the traffic consists of small packets being routed up a tree to a single gateway. The heuristic consists of four steps. The first step builds a spanning tree over the network rooted at the gateway. Using Dijkstra's shortest path algorithm any connected graph can be converted into a spanning tree. As can be seen in Figure 9(b), the spanning tree must maintain "hidden" links that are not used when iterating through the tree, but can be used to maintain that the two hop constraint is still satisfied in the original graph. Once a spanning tree is constructed, a breadth first search is performed starting from root of the tree. The heuristic begins with an initially empty set of colors. As each node is traversed by the breadth first search, it is assigned the lowest value in the color set that is unique from any single or two hop neighbors. If there are no free colors, a new color must be added into the current set. The next step in the heuristic tries to eliminate redundant slots that lie deeper in the tree by replacing them with larger valued slots. As will become apparent in the next step, this manipulation allows data from the leaves of the tree to move as far as possible towards the gateway in a single TDMA cycle. Figure 9(c) shows how the previous three nodes are given larger values in order to minimize packet latencies. The final step in the heuristic inverts all of the slot assignments such that lower slot values are towards the edge of the tree allowing information to be propagated and aggregated in a cascading manner towards the gateway.

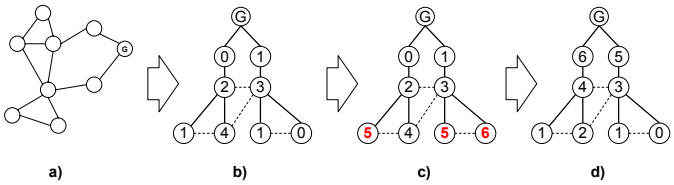


Figure 9. a) shows a mesh network topology. b) shows how this topology can be turned into a spanning tree and colored using Dijkstra's algorithm and a greedy breadth first search heuristic. c) Repeated colors that lie further down the tree are made larger so that data can flow in a single cycle towards the gateway. d) All slots are inverted such that earlier slots can cascade up the tree towards the gateway.

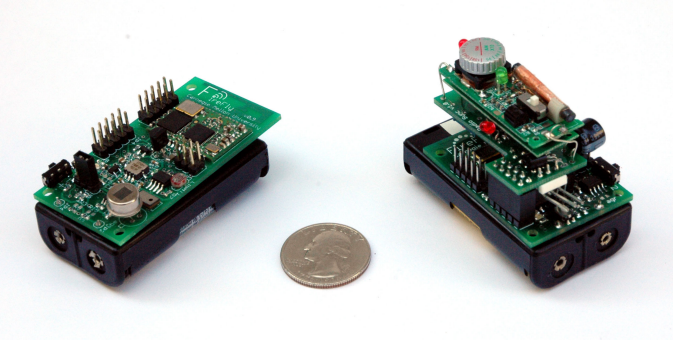


Figure 10. FireFly and FireFly Jr board with AM synchronization module

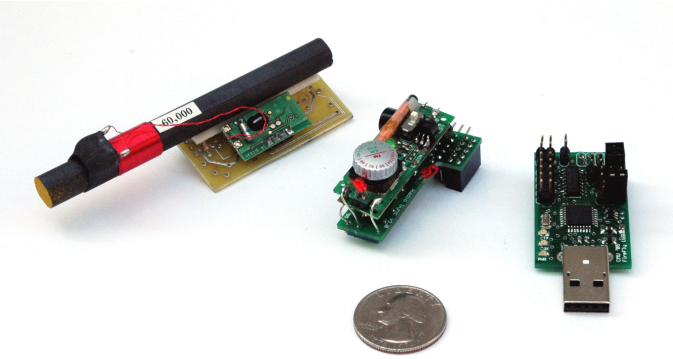


Figure 11. Left to Right: WWVB atomic clock receiver, AM receiver and USB interface board.

5. RT-Link Implementation

In the following section, we describe our hardware platform as well as two different hardware-aided out-of-band time synchronization solutions. First, we introduce FireFly, a custom 802.15.4 wireless sensor node. Following this, we describe an add-on board for receiving the atomic clock broadcast for outdoor synchronization and a board for receiving an AM broadcast synchronization pulse for indoors. We then evaluate the timing and energy impact of our synchronization hardware on the MAC protocol.

5.1. Hardware

Figure 10 shows our custom sensor node, FireFly. The board uses an Atmel Atmega32L [12] 8-bit microcontroller and a Chipcon CC2420 [13] IEEE 802.15.4 wireless transceiver. The microcontroller operates at 8Mhz and has 32KB of ROM and 2KB of RAM. The FireFly board includes light, temperature, audio, dual-axis acceleration and passive infrared motion sensors. We have also developed a lower-cost version of the board called the FireFly Jr. that does not include sensors, and is used to forward packets in the network. The FireFly board interfaces with a computer using an external USB dongle.

5.2. Time Synchronization

In order to achieve the highly accurate time synchronization required for TDMA at a packet level granularity, we use two out-of-band time synchronization sources. One uses the WWVB atomic clock broadcast, and the other relies on a carrier-current AM transmitter. In general, the synchronization device should be low power, inexpensive, and consist of a simple receiver. The time synchronization transmitter must be capable of covering a large area.

5.2.1. Implementation

The WWVB atomic broadcast is a pulse width modulated signal with a bit starting each second. Our system uses an off-the-shelf WWVB receiver (Figure 11) to detect these rising edges, and does not need to decode the entire time string. When active, the board draws 0.6mA at 3 volts and requires less than 5uA when powered off. Inside buildings, atomic clock and GPS receivers are typically unable to receive any signal, so we use a carrier-current AM broadcast. Carrier-current uses a building's power infrastructure as an antenna to radiate the time synchronization pulse. We used an off-the-shelf low-power AM transmitter and power coupler [14] that adhere to the FCC Part 15 regulations without requiring a license. The transmitter provides time synchronization to two 5-storey campus buildings which operate on 2 AC phases. Figure 11 shows an add-on AM receiver module capable of decoding our AM time sync pulse. We use a commercial AM receiver module and then designed a custom supporting-board which thresholds the demodulated signal to decode the pulse. The supporting AM board is capable of controlling the power to the AM receiver.

5.2.2. Energy Consumption

The energy required to activate the AM receiver module and to receive a pulse is equivalent to sending one and a half 802.15.4 packets. The use of a more advanced single chip AM radio [15] would bring these values lower and allow for a more compact design. We estimate that using a single chip AM radio receiver, the synchronization energy cost would be less than one tenth the energy of sending or receiving a single in band packet. We also investigated using a subcarrier FM transmission from a local radio station to transmit the synchronization pulse. Commercial FM radio stations are typically issued two subcarrier channels by the FCC for transmitting digital information such as song names, weather and traffic information. We have not yet pursued such technology since it would make control of the timing source more difficult during our early development phase since the transmitter would be physically located at a radio station.

5.2.3. Scalability and Performance

In order to maintain scalability across multiple buildings, our AM transmitter locally rebroadcasts the atomic clock time signal. The synchronization pulse for the AM transmitter is a line-balanced 50us square wave generated by a modified

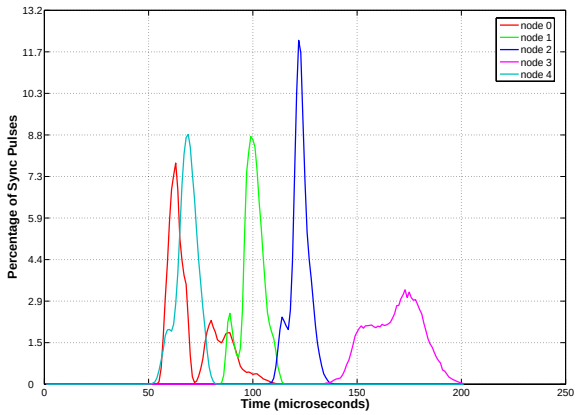


Figure 12. Distributions of AM carrier current time synchronization jitter over a 24 hour period.

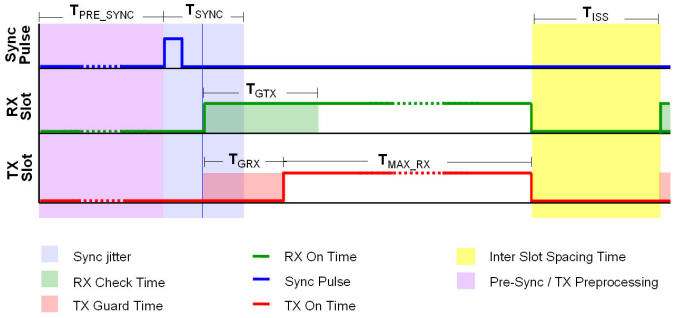


Figure 13. RT-Link operation and timing parameters.

FireFly node capable of atomic clock synchronization.

In order to evaluate the effectiveness of the synchronization, we placed five nodes at different points inside a five storey building. Each node was connected to a data collection board using several hundred feet of cables. The data collection board timed the difference between when the synchronization pulse was generated and when each node acknowledged the pulse. This test was performed while the MAC protocol was active in order to get an accurate idea of the possible jitter including MAC related processing overhead. Figure 12 shows a histogram with the distribution of each node's synchronization time jitter. An AM pulse was sent once per second for 24 hours during normal operation of a classroom building. The graph shows that the jitter is bounded to within 200us. 99.6% of the synchronization pulses were correctly detected. We found that with more refined tuning of the AM radios, the jitter could be bounded to well within 50us.

In order to maintain synchronization over an entire TDMA cycle duration, it is necessary to measure the drift associated with the clock crystal on the processor. We observed that the worst of our clocks was drifting by 10us/s giving it a drift rate of 10e-5. Our previous experiment illustrates that the jitter from AM radio was at worst 100us indicating that the drift would not become a problem for at least 10 seconds. The drift due to the clock crystal was also relatively consistent, and hence could be accounted for in software by timing the difference between synchronization pulses and performing a clock-rate adjustment. In our final implementation with a line-balanced input to the transmitter, we are able to maintain globally synchronization to within 20us.

5.3. TDMA Slot Mechanics

When a node is first powered on, it activates the AM receiver and waits for the first synchronization pulse. Figure 13 shows the actual timing associated with our TDMA frames. Once the node detects a pulse, it resets the TDMA frame counter maintained in the microcontroller which then powers down the AM receiver. When the node receives its synchronization pulse, it begins the active TDMA time cycle. After checking its receive and transmit masks, the node determines which slots it should transmit and receive on. During a receive timeslot, the node immediately turns on the receiver. The receiver will wait for a packet, or if no preamble is detected it will time out.

The received packet is read from the CC2420 chip into a memory address that was allocated to that particular slot. We employ a zero-copy buffer scheme to move packets from the receive to the transmit queue. In the case of automatic packet aggregation, the payload information from a packet is explicitly copied to the end of the transmit buffer. When the node reaches a transmit timeslot, it must wait for a guard time to elapse before sending data. Accounting for the possibility that the receiver has drifted ahead or behind the transmitter, the transmitter has a guard time before sending and the receiver preamble-check has a guard time extending beyond the expected packet. Table 2 in the next section shows the different timeout values that work well for our hardware configuration. Once the timeslot is complete, there needs to be an additional guard time before the next slot. We provide this guard time plus a configurable inter-slot processing time that allows the MAC to do the minimal processing required for inter-slot packet forwarding. This feature is motivated by memory limitations and reduction of network queue sizes. After the TDMA cycle has completed, the interrupt handler returns leaving the flow of execution to continue after the last processor sleep call. At this point the application can run a sensing task, schedule a packet for transmission, or return to sleep until the next interrupt is called. Figure 14 shows a sample trace of two nodes communicating with each other. The rapid receiver checks at the end of the cycle show the contention period with low-power listening for the duration of a preamble.

5.4. Modeling Lifetime

To calculate the node duty cycle and lifetime we sum the node's energy consumptions over a TDMA frame. Table 1 shows the power consumed by each component assuming

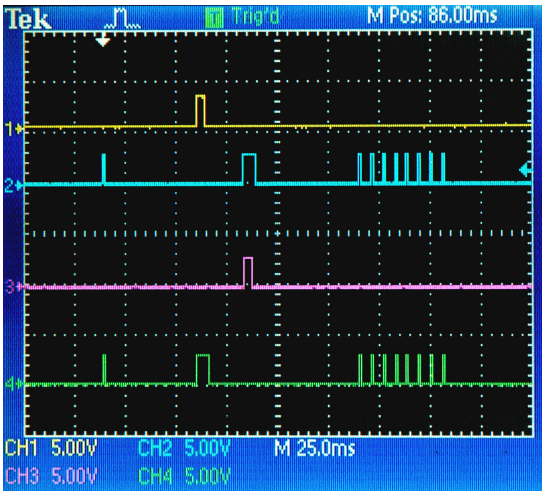


Figure 14. Channels 1 and 2 show transmit and receiver activity for one node. Channels 3 and 4 show radio activity for a second node that receives a packet from the first node and transmits a response a few slots later. The small pulses represent RX checks that timed out. Longer pulses show packets of data being transmitted. The group of pulses towards the right side show the contention slots.

operation at 3 volts. Table 2 and Table 3 show the timing parameters and the energy of each operation during the TDMA frame. The active time of each TDMA slot, T_{active}, is dependent on the total number of slots, N_{slots}, the maximum slots transmit time $T_{max\_payload}$, the AM synchronization setup T_{sync_setup} and capture T_{sync} as well as inter slot processing time T_{ISS}.

$$T_{active} = T_{sync\_setup} + T_{sync} + N_{slots} * (T_{max\_payload} + T_{ISS}) \quad (1)$$

The idle time, T_{idle}, between slots is the difference between the active time and the total frame time, T_{frame}. This is typically customized for the specific application since it has impact on both battery life and latency.

$$T_{idle} = T_{frame} - T_{active} \quad (2)$$

The three customizable parameters that define the lifetime of a node are the TDMA frame time, the number of TDMA slots (including the number of contention slots Ncontention) and the degree d of the node. As the degree increases, the node must check the start of additional time slots and may potentially have to receive packets from its neighbors. The minimum energy that the node will require during a single TDMA frame E_{min} is the sum of the different possible energy consumers assuming no packets are received and the node does not transmit packets:

$$E_{min} = E_{sync} + (d + N_{contention}) * E_{GRX} + E_{CPU\_active} + E_{CPU\_sleep} + E_{radio\_idle} + E_{radio\_sleep} \quad (3)$$

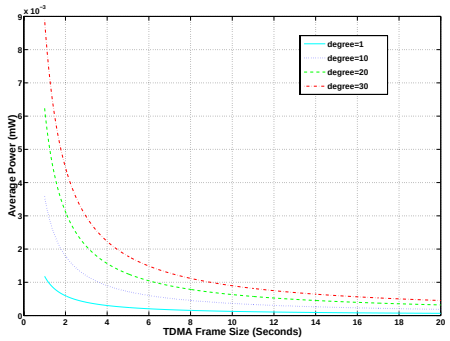


Figure 15. Avg. power consumed by a node with 32 time slots with respect to neighbor degree and TDMA frame size.

Power Parameters	Symbol	I(ma)	Power(mW)
Radio Transmitter	P_{radio_TX}	17.4	52.2
Radio Receiver	P_{radio_RX}	19.7	59.1
Radio Idle	P_{radio_idle}	0.426	1.28
Radio Sleep	P_{radio_sleep}	$1e^{-3}$	$3e^{-3}$
CPU Active	P_{CPU_active}	1.1	3.3
CPU Sleep	P_{CPU_sleep}	$1e^{-3}$	$3e^{-3}$
AM Sync Active	P_{sync}	5	15

Table 1. Power Consumption of the main components.

The maximum energy the node can consume during a single TDMA frame is the minimal energy consumed during that frame summed with the possible radio transmissions that can occur during a TDMA frame.

$$E_{max} = E_{min} + (d + N_{contention}) * E_{RX} + N_{TX\_slots} * E_{TX} \quad (4)$$

The maximum power consumed by a node over a TDMA frame can be computed as follows:

$$P_{avg} = E_{max}/T_{frame} \quad (5)$$

The lifetime of the node can be computed as follows:

$$Lifetime = (E_{capacity}/E_{max}) * T_{Frame} \quad (6)$$

Figure 15 shows the average power of a node with respect to TDMA frame size with different neighbor degrees using 32 time slots, 8 of which are used during the contention segment of the protocol. As the node degree increases, in order to maintain the same average power and hence lifetime, the TDMA frame size must be increased.

6. Performance Evaluation

In this section, we compare multi-hop performance of RT-Link with that of S-MAC and B-MAC. We first validated our implementation of RT-Link in a 10-node test-bed. Following this, we use simulation to compare latency and throughput.

Timing Parameters	Symbol	Time (ms)
Max Packet Transfer	$T_{max\_payload}$	4
Sync Pulse Jitter	T_{sync}	$100e^{-3}$
Sync Pulse Setup	T_{sync_setup}	$20 + (\rho * T_{frame})$
RX Timeout	T_{GRX}	$300e^{-3}$
TX Guard Time	T_{GTX}	$100e^{-3}$
Inter Slot Spacing	T_{ISS}	$500e^{-3}$
Clock Drift Rate	ρ	$10e^{-2}s/s$

Table 2. Timing Parameters for main components.

Energy Parameters	Symbol	Energy (mW)
Synchronization	E_{sync}	$P_{sync}*$ $(T_{sync} + T_{sync\_setup})$
Active CPU	E_{CPU_active}	$P_{CPU\_active} * T_{active}$
Sleep CPU	E_{CPU_sleep}	$P_{CPU\_sleep} * T_{idle}$
TX Radio	E_{radio_tx}	$P_{radio\_tx}*$ $(T_{max\_payload} + T_{GTX})$
RX Radio	E_{radio_rx}	$P_{radio\_rx} * T_{max\_payload}$
Idle Radio	E_{radio_idle}	$P_{radio\_idle} * T_{active}$
Sleep Radio	E_{radio_sleep}	$P_{radio\_sleep} * T_{idle}$
RX Radio Check	E_{GRX}	$P_{radio\_rx} * T_{GRX}$

Table 3. Energy of components with respect to power and time.

6.1. Multi-hop Network Performance

In order to determine the spatial separation between nodes for interference-free communication, we placed ten nodes in a line and fixed the power so that each node could only reliably communicate with its direct neighbors. The nodes generated a 50 byte packet of data each second for 1000 seconds. Each node transmitted data only to the neighbor closer to the gateway. In the first test, all nodes were assigned unique time-slots as shown in Figure 16(a) so that there were no concurrent transmitters. We repeated this test three times and observed that every node received 100% of its neighbor's packets. This provided a sanity check that the 500us inter-slot processing time provided the necessary temporal separation for both the synchronization jitter and the packet processing (i.e. packet reception and aggregation) between slots. We also observed at the gateway that the end-to-end delay for node 1's packets was consistently under 50ms. This confirmed our expectation of aggregating and forwarding packets on a slot-by-slot basis. In the second test, as shown in Figure 16b), two time slots (i.e. slots 0 and 1) were alternated across the nodes. This 1-hop coloring resulted in an average packet loss of 67% due to interference. Following this, three slots (i.e. 0, 1 and 2) were alternated across the nodes and each node received all transmitted packets. The tests were repeated thrice in two different outdoor locations (a football stadium and an open field in a park) and the results were consistent across all eighteen runs.

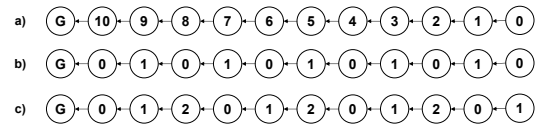


Figure 16. Multi-hop schedules in RT-Link test-bed.

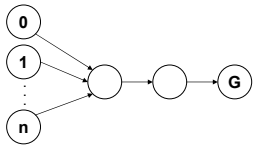


Figure 17. Multi-hop network topology with hidden terminal problem.

6.2. End-to-end Latency

In order to investigate the performance of RT-Link, we simulated its operation to compare the end-to-end latency and throughput with asynchronous and loosely synchronized protocols across various topologies. To study the latency in a multi-hop scenario we focused on the impact of the hidden terminal problem on the performance of B-MAC and S-MAC. All the tests in [3] were designed to avoid the hidden terminal problem and essentially focused on extremely low-load and one-hop scenarios. We simulated a network topology of two "backbone" nodes connected to a gateway. One or more leaf nodes were connected to the lower backbone node as shown in Figure 17. Only the leaf nodes generated traffic to the gateway. The total traffic issued by all nodes was fixed to 1000 1-byte packets. At each hop, if a node received multiple packets before its next transmission, it was able to aggregate them up to 100-byte fragments. The tested topology is the base case for the hidden terminal problem as the transmission opportunity of the backbone nodes is directly affected by the degree of the lower backbone node.

We compare the performance of RT-Link with a 100ms and 300ms cycle duration with RTS-CTS enabled B-MAC operating with 25ms and 100ms check times. The RTS-CTS capability was implemented as outlined in [3]. When a node wakes up and detects the channel to be clear, RTS and CTS with long preambles are exchanged followed by a data packet with a short preamble. We assume B-MAC is capable of perfect clear channel assessment, zero packet loss transmissions and zero cost acknowledgement of packet reception. We observe that as the node degree increases (Figure 18), B-MAC suffers a linear increase in collisions, leading to an exponential increase in latency. With a check time of 100ms, B-MAC saturates at a degree of 4. Increasing the check time to 25ms, pushes the saturation point out to a degree of 8. Using the schedule generate by the heuristic in Section 4, RT-Link demonstrates a flat end-to-end latency.

The clear drawback to a basic B-MAC with RTS-CTS is that upon hidden terminal collisions, the nodes immediately retry after a small random backoff. To alleviate problem, we

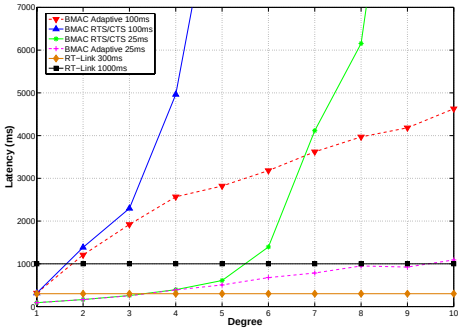


Figure 18. Impact of Latency with node degree

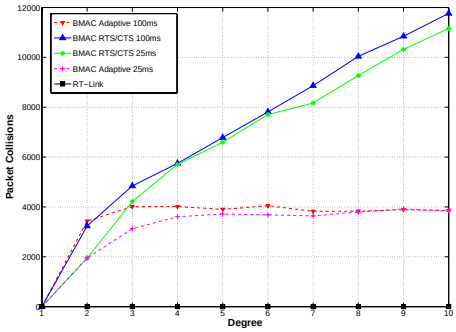


Figure 19. Effect of node degree on packet collisions for B-MAC

provided nodes with topology information such that a node's contention window size is proportion to the product of the degree and the time to transmit a packet. As can be seen in Figure 19, this allows for a relatively constant number of collisions since each node shares the channel more efficiently. This extra backoff, in turn increases latency linearly with the node degree. We see that RT-Link suffers zero collisions and maintains a constant latency.

6.3. Throughput

Figure 20 shows the effect of node degree on throughput. In this example, all nodes are within a single hop of the receiver and do not exhibit the hidden terminal problem. As in [3, 6], each node constantly transmits data to the one-hop away gateway. We observe that as the number of nodes communicating with the gateway increases, RT-Link is able to assign additional time slots and maintain a fixed throughput. In such a scenario, RT-Link can support approximately 2,500 unique time slots in one second while re-synchronizing every 10 seconds. RT-Link maintains a steady 80% throughput. The 20% loss in throughput (i.e. approximately 800us for every 4ms packet) is due to inter-slot spacing used for packet processing and aggregation. On the other hand, the throughput offered by B-MAC decreases as the channel contention around the gateway increases. The throughput offered by S-MAC is limited due to the fixed 115ms sleep duration enforced on each node.

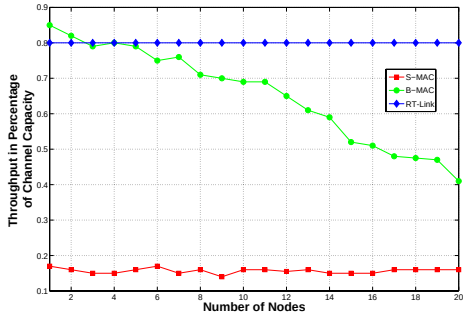


Figure 20. Effect of node degree on throughput for a single hop with no hidden terminals.

7. Conclusion

In this paper we explore the design, implementation and performance of a link layer protocol for energy-constrained multi-hop wireless networks with end-to-end delay constraints. We introduced RT-Link, a time-synchronized link protocol for fixed and mobile embedded radios. We identify three key observations in the design and deployment of RT-Link: (a) RT-Link offers predictable network lifetime with bounded end-to-end delay. (b) Achieving global time synchronization is both economical and convenient for indoor and outdoor deployments. (c) Due to interference between nodes, nodes with the same schedule must be spaced by a minimum of 3 hops. RT-Link has been implemented in FireFly, our sensor network platform, and has been deployed on networks with over 30 IEEE 802.14.5 nodes. It outperforms energy-efficient protocols such as B-MAC and S-MAC in throughput, energy consumption and end-to-end delay.

References

[1] IEEE Std 802.15.4, 2003.

[2] Randolph D. Nelson and Leonard Kleinrock. Maximum probability of successful transmission in a random planar packet radio network. *INFOCOM*, pages 365–370, 1983.

[3] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. *SenSys*, November 2005.

[4] T. Dam and K. Langendoen. An adaptive energy-efficient mac protocol for wireless sensor networks. *SenSys*, November 2003.

[5] A. El-Hoiydi and J. Decotignie. Wisemac: An ultra low power mac protocol for the downlink of infrastructure wireless sensor networks. *ISCC*, 2004.

[6] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. *INFOCOM*, June 2002.

[7] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves. Energy-efficient, collision-free medium access control for wireless sensor networks. *Sensys*, 2003.

[8] L. G. Roberts. Aloha packet system with and without slots and capture. *SIGCOMM*, 5(2):28–42, 1975.

[9] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava. Topology management for sensor networks: Exploiting latency and density. *MobiHoc*, 2002.

[10] C. Guo, L. C. Zhong, and J. Rabaey. Low power distributed mac for ad hoc sensor radio networks. *Globecom*, 2001.

[11] Hari Balakrishnan et al. The distance-2 matching problem and its relationship to the mac-layer capacity of ad hoc wireless networks. *IEEE Journal on Selected Areas in Comm.*, 22(6):1069–1079, August 2004.

[12] Atmel corporation, atmega32 data sheet, March 2005.

[13] Chipcon inc., chipcon cc2420 data sheet, October 2003.

[14] Radio systems 30w tr-6000 am transmitter data sheet, March 2001.

[15] Tea5551t 1-chip am radio philips semiconductors, October 1990.