# Versatile Low Power Media Access for Wireless Sensor Networks

Joseph Polastre
Computer Science Department
University of California, Berkeley
Berkeley, CA 94720
polastre@cs.berkeley.edu

Jason Hill
JLH Labs
35231 Camino Capistrano
Capistrano Beach, CA 92624
jhill@jlhlabs.com

David Culler
Computer Science Department
University of California, Berkeley
Berkeley, CA 94720
culler@cs.berkeley.edu

## ABSTRACT

We propose *B-MAC*, a carrier sense media access protocol for wireless sensor networks. Motivated by environmental monitoring applications, B-MAC features ultra low power operation, effective collision avoidance, small code size, and predictable execution. To achieve low power operation, B-MAC employs an adaptive low power radio sampling scheme to reduce duty cycle, minimize idle listening, and eliminate the overhead of synchronization. B-MAC allows services to reconfigure the MAC protocol for optimal performance, whether it be for throughput, latency, or power conservation. MAC reconfiguration is necessary to meet the application's current (and often changing) demands. We build an analytical model of B-MAC's operation. By comparing B-MAC to conventional 802.11-inspired protocols, specifically S-MAC, we develop an experimental model over a wide range of network conditions. We show B-MAC outperforms S-MAC in packet delivery rates, throughput, latency, and energy consumption. By deploying a real world monitoring application with multihop networking, we validate our protocol design and model. Ultimately, B-MAC is simpler and smaller than existing protocols yet conserves more power enabling long term deployments.

## 1. INTRODUCTION

In wireless sensor network deployments, reliably reporting data while consuming the least amount of power possible is the ultimate goal. One such application that drives the design of low power media access control (MAC) protocols is environmental monitoring. Mainwaring et. al. [13] and the UCLA Center for Embedded Network Sensing [2, 6] have deployed wireless sensors for microclimate monitoring. These networks operate at low duty cycles with multihop networking and reliable data reporting. They have shown it is important that the media access mechanism support duty cycles of 1% but be flexible enough to efficiently transfer different workloads and adapt to changing networking conditions. These workloads include periodic data reporting, bulk log transfer, and wirelessly reprogramming a node. In this paper we discuss the design of a MAC protocol motivated by monitoring applications.

Nodes in wireless sensor networks do not exist in isolation; rather they are embedded into the environment and thereby affected by it [15]. Network links are unpredictable and change based on environmental conditions. Nodes are typically in fixed locations and the their surrounding environment changes. For example, RF performance may be hindered by a sudden rain storm or the opening and closing of doors in a building.

Woo [20] and Zhao [23] have studied the volatility in link quality in wireless sensor networks. Zhao shows the existence of "gray areas" where some nodes exceed 90% successful reception while neighboring nodes receive less than 50% of the packets. He also shows that the gray area is rather large–one-third of the total communication range. Woo independently verified Zhao's gray area findings. In designing a reliable multihop routing protocol, Woo shows that effectively estimating link qualities over time is essential. He shows that snooping on traffic over the broadcast medium is crucial for extracting information about the surrounding topology. Snooping can prevent cycles, notify neighboring nodes of unreachable routes, and provide link quality information. Since data must ultimately be reported out of the network through multihop routing, the media access protocol must be reconfigurable to meet changing demands.

Not only are the networking conditions different, applications for wireless sensor networks have different demands than those designed for traditional ad-hoc wireless networks. Intanagonwiwat et. al. [9] show how 802.11 is inappropriate for low duty cycle sensor network data delivery. Idle listening in 802.11 consumes as much energy when the protocol is idle as it does when receiving data. It is absolutely crucial that the MAC protocol support a duty cycling mechanism to eliminate idle listening.

For wireless sensor networks to gain acceptance in the scientific community, data must flow from the network predictably and reliably. Scientists determine the sample period and physical deployment of the nodes. The role of the network is to ensure that regardless of the scientist's settings, data is delivered as expected. A general rule for achieving predictable operation is to reduce complexity as much as possible from the application and its services. Since each node runs one application with a set of services, it is important to optimize communication performance for that application–not for a generic set of users.

To meet the requirements of wireless sensor network deployments and monitoring applications, we need a set of goals for the media access protocol that maps to the application's requirements. Our goals for a MAC protocol for wireless sensor network applications are:

- Low Power Operation
- Effective Collision Avoidance
- Simple and Predictable Operation
- Small Code Size and RAM Usage
- Tolerant to Changing RF/Networking Conditions
- Scalable to Large Numbers of Nodes

To meet the goals, we propose B-MAC, a configurable MAC protocol for wireless sensor networks. Our MAC protocol is simple in both design and implementation. It has a small core and factors out higher layer functionality. Factoring out some functionality and exposing control to services using the MAC protocol allows the protocol to support a wide variety of sensor network workloads. This minimalist model of MAC protocol design is in contrast to the classic monolithic MAC protocols optimized for a general set of workloads; however this paper shows the effectiveness of a small, configurable MAC protocol that supports low duty cycle applications.

The contributions of this paper are not only the design of a versatile MAC protocol for sensor networks. We propose an adaptive control interface for wireless sensor network applications. The interface allows middleware services to reconfigure the MAC protocol based on the current workload. We build a model of B-MAC's performance that maximizes a node's lifetime. Our model consists of a system of equations augmented with an experimental model of B-MAC's performance in a comprehensive set of microbenchmarks. To test the model, we built an environmental monitoring application and show how its performance matches our model's predictions. Our model can be used to identify the best parameters for an arbitrary low power wireless sensor network application and calculate the application's lifetime.

## 2. RELATED WORK

Most MAC protocols for wireless sensor networks have been based on conventional wireless protocols such as 802.11. These protocols typically provide a general purpose mechanism that works reasonably well for a large set of traffic workloads. The previous efforts serve as great building blocks for designing a MAC protocol that meets our goals.

The DARPA Packet Radio Network (PRNET) [11] was one of the first ad-hoc multihop wireless networks. PRNET had two media access protocols–Slotted ALOHA [1] and Carrier Sense Multiple Access [10]. Much of the standard MAC protocol functionality–including random delays, forwarding delays, link quality estimation, and low duty cycle through node synchronization–were first executed in PRNET. CSMA is validated as a way to efficiently use the majority of the channel's bandwidth while duty cycling nodes. TDMA and slotted ALOHA solutions in PRNET were ultimately dismissed due to their inability to scale.

Woo and Culler [19] illustrate the effect of changing the MAC protocol based on the workload. They show that the application scenario and network traffic characteristics differ significantly from conventional computer networks. In sensor networks, typically data is sent periodically in short packets. To achieve fairness and energy efficient transmission through a multihop network, they design an adaptive rate control protocol to transmit sensor network workloads efficiently–specifically optimized for $n$-to-1 data reporting and multihop networking. Existing MAC protocols are simply not suitable due to their failure to efficiently support sensor network workloads in low duty cycle conditions.

Hill and Culler [7] propose sampling the energy on the radio channel to duty cycle the radio. Hill's RF wakeup scheme samples the analog baseband of the radio every 4 seconds. By quickly evaluating the channel's energy, he reduced the duty cycle of the radio to below 1%. He demonstrated the use of low power RF wakeup on an 800 node multihop network.

Published concurrently with Hill's work, Aloha with preamble sampling [4] presents a low power technique similar to that used in paging systems [14]. To let the receiver sleep for most of the time when the channel is idle, nodes periodically wake up and check for activity on the channel. If the channel is idle, the receiver goes back to sleep. Otherwise, the receiver stays on and continues to listen until the packet is received. Packets are sent with long preambles to match the channel check period. El-Hoiydi [4] creates a model for Aloha with preamble sampling and presents the effect of delay due to long preambles. He proposes using the long preamble for initial synchronization of nodes; afterwards the nodes transmit and receive on a schedule with normal sized packets.

WiseMAC [5] is an iteration on Aloha with preamble sampling specifically designed for infrastructure wireless sensor networks. The main contribution of WiseMAC is an evaluation of the power consumption of WiseMAC, 802.11, and 802.15.4 under low traffic loads. They show that for the same delay, WiseMAC and preamble sampling lower power consumption by 57% over PSM used in 802.11 and 802.15.4. WiseMAC meets many of our goals except that it has no mechanism to reconfigure based on changes in network conditions or changing demands from services using the protocol.

S-MAC [21] is a low power RTS-CTS scheme for wireless sensor networks inspired by PAMAS and 802.11. S-MAC periodically sleeps, wakes up, listens to the channel, and then returns to sleep. Each active period is of fixed size, 115 ms, with a variable sleep period. The length of the sleep period dictates the duty cycle of S-MAC. At the beginning of each active period, nodes exchange synchronization information. Following the SYNC period, data may be transferred for the remainder of the active period using RTS-CTS. In a follow up paper [22], the authors add adaptive listening–when a node overhears a neighbor's RTS or CTS packets, it wakes up for a short period of time at the end of their neighbor's transmission to immediately transmit its own data. By changing the duty cycle, S-MAC can trade off energy for latency. S-MAC uses message fragmentation to deliver each piece of the message. Fragmentation uses the RTS-CTS scheme to reserve the channel, then transmit packets in a burst to save energy but reduce fairness. Although S-MAC achieves low power operation, it does not meet our goals of simple implementation, scalability, and tolerance to changing network conditions. As the size of the network increases, S-MAC must maintain an increasing number of schedules of surrounding nodes or incur additional overhead through repeated rounds of resynchronization.

T-MAC [18] improves on S-MAC by using a very short listening window at the beginning of each active period. After the SYNC part of the active period, there is a short window to send or receive RTS and CTS packets. If no activity occurs in that period, the node returns to sleep. By changing the protocol to have an adaptive duty cycle, T-MAC saves power at a cost of reduced throughput and additional latency. T-MAC, in variable workloads, uses 5 times less power than S-MAC. In homogeneous workloads, T-MAC and S-MAC perform equally well. T-MAC suffers from the same complexity and poor scaling of S-MAC. Shortening the active window in T-MAC reduces the ability to eavesdrop on surrounding traffic and adapt to changing network conditions.

Many of these protocols have only been evaluated in simulation. Not only must the protocol perform well in simulation, it must also integrate well with the implementation of any wireless sensor network application. Each of the protocols described in this section provide great solutions to meet a subset of the goals described in Section 1. Motivated by monitoring applications for wireless sensor networks, we leverage good ideas from previously published work to create a protocol that meets all of the goals from Section 1.

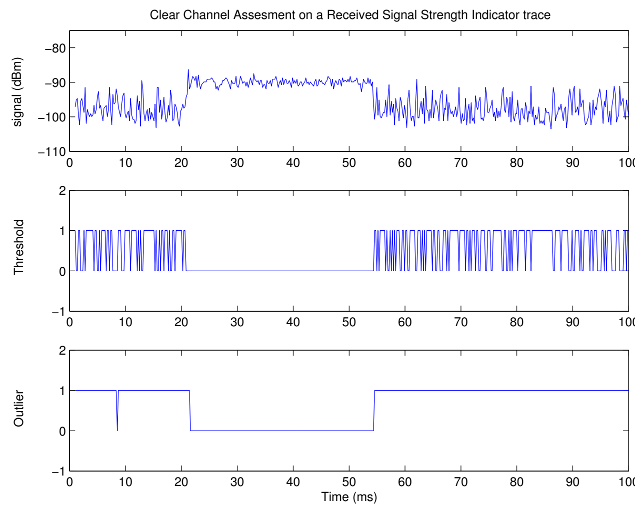


**Figure 1: Clear Channel Assessment (CCA) effectiveness for a typical wireless channel. The top graph is a trace of the received signal strength indicator (RSSI) from a CC1000 transceiver. A packet arrives between 22 and 54ms. The middle graph shows the output of a thresholding CCA algorithm based on the noise floor. A `1` indicates that the channel is clear, `0` indicates it is busy. The bottom graph shows the output of an outlier detection algorithm over a sliding window of 5 RSSI samples.**

## 3. DESIGN AND IMPLEMENTATION

To achieve the goals outlined in Section 1, we designed a CSMA protocol for wireless sensor networks called B-MAC. Although B-MAC is motivated by the needs of monitoring applications, the flexibility of our protocol allows allows other services and applications to be realized efficiently. These services include but are not limited to target tracking, localization, triggered event reporting, and multihop routing.

S-MAC is an example of a wireless sensor network protocol designed using a classical approach. The protocol provides low power operation for a set of workloads that the authors feel are representative of wireless sensor networks. Applications and services rely on S-MAC to adjust its operation as node and network conditions change. B-MAC was designed with a minimalist approach. B-MAC has a small core of media access functionality and factors out some logic and state maintenance to services using the protocol. With this design methodology, services using B-MAC can implement higher layer protocols without affecting other node services. We first describe the core media access functionality provided by the B-MAC protocol.

For effective collision avoidance, the MAC protocol must be able to accurately determine if the channel is clear, referred to as Clear Channel Assessment (CCA). Since the ambient noise changes depending on the environment, B-MAC employs software automatic gain control for estimating the noise floor. Signal strength samples are taken at times when

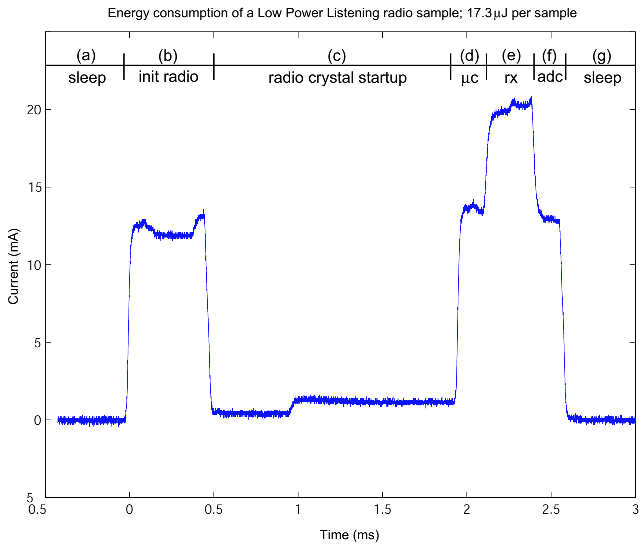


**Figure 2: When turning on the radio, the node must perform a sequence of operations. The node first starts in sleep state (a), then wakes up on a timer interrupt (b). The node initializes the radio configuration and tells the radio to begin its startup phase. The startup phase (c) waits for the radio's crystal oscillator to stabilize Upon stabilization, the microcontroller instructs the radio to enter receive mode (d). After the switch time, the radio enters receive mode (e) and a sample of the received signal energy may begin. After the ADC starts acquisition, the radio is turned off and the ADC value is analyzed (f). If there is no activity on the channel, the node returns to sleep (g).**

the channel is assumed to be free–such as immediately after transmitting a packet or when the data path of the radio stack is not receiving valid data. Samples are then entered into a FIFO queue effectively creating a sliding window. The median of the queue is added to an exponentially weighted moving average with decay $\alpha$. The median is used as a simple low pass filter to add robustness to the noise floor estimate. We found that using an $\alpha$ of $0.06$ and FIFO queue size of $10$ provided the best results for a typical wireless channel. Once a good estimate of the noise floor is established, a request to transmit a packet starts the process of monitoring the received signal from the radio. A common method used in a variety of protocols, including 802.15.4, takes a single sample and compares it to the noise floor. This thresholding method produces results with a large number of false negatives. Instead, B-MAC searches for outliers in the received signal such that the energy on the channel is significantly below the noise floor. If an outlier exists during the channel sampling period, B-MAC concludes that the channel must be clear. The effectiveness of outlier detection as compared to thresholding on a trace from a CC1000 [3] transceiver is shown in Figure 1.

B-MAC performs duty cycling through periodic channel sampling that we call Low Power Listening (LPL). Our technique is similar to the one used by Aloha [4]. Each time the node wakes up, it turns on the radio and checks for activity. If energy is detected, the node powers up and stays awake for the time required to receive the incoming packet. After reception, the node returns to sleep. If no packet is received (a false positive), a timeout forces the node back to sleep. Accurate energy detection is critical to achieving low power operation with this method. We use the noise floor estimation of B-MAC not only for finding a clear channel on transmission but also for determining if the channel is active during LPL. In order to receive data reliably, the length of the preamble is matched to interval that the channel is checked. For example, if the channel is checked every 100ms, the preamble must be at least 100ms long in order for a node to wake up, detect energy on the channel, receive the preamble and start of frame delimiter, and then receive the message. The effect of varying the preamble size and sleep time is discussed in more detail in Section 4.

A trace of the power consumption for sampling the channel on a Mica2 mote [16] is shown in Figure 2. The process shown in Figure 2 applies to any MAC protocol for sensor networks. It must perform initial configuration of the radio (b), start up the radio and its oscillator (c), switch the radio to receive mode (d), and then perform the actions of the protocol. As a result, the cost for powering up the radio is the same across all protocols. The difference between protocols is how long the radio is on after it has been started and how many times the radio is started.

In sensor networks, each node typically runs a single application. Since the RAM and ROM available on sensor nodes are extremely limited, keeping the size of the MAC implementation small is important. Reducing the complexity of the protocol reduces the likelihood of race conditions and reduces the state maintained by the protocol. Minimizing state eases protocol analysis and enables predictable operation. We implemented B-MAC in TinyOS [12] to evaluate its efficacy in meeting the goals outlined in this section. Since B-MAC does not have the RTS-CTS mechanism or synchronization requirements of S-MAC[1], the implementation is both simpler and smaller as shown in Table 1.

| **Protocol** | **ROM** | **RAM** |
|---|---|---|
| B-MAC | 3046 | 166 |
| B-MAC with ACK | 3340 | 168 |
| B-MAC with LPL | 4092 | 170 |
| B-MAC with LPL and ACK | 4386 | 172 |
| S-MAC | 6274 | 516 |

**Table 1: A comparison of the size of B-MAC and S-MAC in bytes. Each protocol was implemented in TinyOS.**

[1]All tests with S-MAC were performed with the implementation in `tinyos-1.x/contrib/s-mac/` in the TinyOS CVS repository [17] as of March 30, 2004.

| Operation | Time (s) | | I (mA) | |
|---|---|---|---|---|
| Initialize radio (b) | 350E-6 | $t_{r_{init}}$ | 6 | $c_{r_{init}}$ |
| Turn on radio (c) | 1.5E-3 | $t_{r_{on}}$ | 1 | $c_{r_{on}}$ |
| Switch to RX/TX (d) | 250E-6 | $t_{rx/tx}$ | 15 | $c_{rx/tx}$ |
| Time to sample radio (e) | 350E-6 | $t_{sr}$ | 15 | $c_{sr}$ |
| Evaluate radio sample (f) | 100E-6 | $t_{ev}$ | 6 | $c_{ev}$ |
| Receive 1 byte | 416E-6 | $t_{rxb}$ | 15 | $c_{rxb}$ |
| Transmit 1 byte | 416E-6 | $t_{txb}$ | 20 | $c_{txb}$ |
| Sample sensors | 1.1 | $t_{data}$ | 20 | $c_{data}$ |

**Table 2: Time and current consumption (I) for completing essential activities of a MAC protocol. Identifiers on each operation map back to the activities of acquiring a radio sample in Figure 2.**

| Notation | Parameter | Default |
|---|---|---|
| $c_{sleep}$ | Sleep Current (mA) | 0.030 |
| $L_{preamble}$ | Preamble Length (bytes) | 271 |
| $L_{packet}$ | Packet Length (bytes) | 36 |
| $n$ | Neighborhood Size (nodes) | 10 |
| $r$ | Sample Rate (packets/s) | 1/300 |
| $t_i$ | Radio Sampling Interval (s) | 100E-3 |
| $C_{batt}$ | Capacity of battery (mAh) | 2500 |
| $V$ | Voltage | 3 |
| $t_l$ | Expected Lifetime (s) | - |

**Table 3: Parameters for the B-MAC protocol. Changing each parameter will result in different energy consumption, $E$, and node lifetime, $t_l$.**

# 4. CONCEPTS AND TRADEOFFS

In this section we describe the fundamental tradeoffs of using LPL to duty cycle the radio. We build an analytical model from a system of equations. Using this system, we show how to compute B-MAC's parameters and maximize our system's lifetime. Using the model, we illustrate the tradeoffs of different parameters including duty cycle, network density, sampling rate. We present an adaptive control interface for wireless sensor network MAC protocols and discuss its operation.

## 4.1 Modeling Lifetime

To calculate node duty cycle and lifetime, we base our assumptions on the B-MAC implementation in TinyOS and the properties of the CC1000 radio. Table 2 lists the assumption we make about various operations performed by a low power MAC protocol. These assumptions describe a representative class of radios for wireless sensor networks. Radios with similar properties to those in Table 2 are manufactured by Chipcon, Infineon, and Motorola. New radios proposed for sensor networks, such as IEEE 802.15.4 compliant radios, also fall into the class described in Table 2. We use these assumptions throughout the remainder of this paper.

The node's lifetime is determined by its overall energy consumption. If the energy consumption is minimized, then the lifetime must be maximized. All of the energies, $E$, are defined in units of millijoules per second, or milliwatts. Calculating the total energy usage can be done by multiplying $E$ by the node lifetime $t_l$. For B-MAC, the energy used by a node consists of the energy consumed from receiving, transmitting, periodically sampling the radio channel with LPL, and sleeping.

$$E = E_{rx} + E_{tx} + E_{listen} + E_{sleep} \tag{1}$$

To receive a packet, B-MAC must sample the radio and detect activity. In order to reliably receive packets, the LPL check interval, $t_i$, must be less than the time of the preamble. Therefore we have the constraint:

$$L_{preamble} \geq \lceil t_i / t_{rxb} \rceil$$

Once we have chosen our check interval and calculated the length of the preamble, we can determine the power consumed by transmitting a packet with LPL. The energy consumed by transmitting, $E_{tx}$, is simply the length of the packet with the preamble times the rate packets are generated (assuming a periodic sensing application such as in [13]).

$$\begin{aligned} t_{tx} &= r \times (L_{preamble} + L_{packet}) t_{txb} \\ E_{tx} &= t_{tx} c_{txb} V \end{aligned} \tag{2}$$

Upon detecting energy on the channel, the node stays awake to receive the incoming data. The node may sample the channel at any point during the preamble, so the time to receive a packet is bounded by the length of the preamble, $L_{preamble}$, plus the length of the packet, $L_{packet}$. It is possible that a node will wake up during the preamble of a packet not destined for it. If the application is periodic with a uniform sampling rate, we can conclude that the node will detect and receive data from each of its $n$ neighbors, regardless of the packet's destination. We refer to the density of neighbors surrounding a node as the *neighborhood size* of the node.

Although receiving data from neighbors shortens a node's lifetime, it allows services using the MAC protocol to eavesdrop on the channel and make decisions based on channel activity–one such example is link reliability estimation and link failure detection for multihop routing.

Using this data, we can bound the total time the node will spend receiving and calculate an upper bound on the energy consumed by receiving, $E_{rx}$.

$$\begin{aligned} t_{rx} &\leq nr(L_{preamble} + L_{packet}) t_{rxb} \\ E_{rx} &= t_{rx} c_{rxb} V \end{aligned} \tag{3}$$

Our calculations are based on a single cell application. To change between single cell and multihop, we need to account for the routing traffic through each node due to its children and its neighbors' children. Instead of $r$ packets per second flowing through a particular node, the traffic through the node must also include all the packets routed by the node and its neighbors. The function children($i$) is defined by the multihop routing protocol.

$$r \times \sum_{i=0}^{n} (\text{children}(i) + 1)$$

Idle listening occurs when the node wakes up to sample the channel and there is no activity. This results in wasted energy; therefore we want to maximize the interval between LPL samples and minimize the time to sample the channel. Figure 2 shows our implementation of LPL sampling in B-MAC that has been tweaked to minimize the channel sampling time (stage (e) in Figure 2). Accordingly, we vary the LPL check interval, $t_i$, to match the traffic pattern. The energy consumed by each LPL sample, $E_{sample}$, is the aggregate of each operation depicted in Figure 2. The energy consumed by LPL sampling per second, $E_{listen}$, is inversely proportional to the check interval.

$$\begin{aligned} E_{sample} &= 17.3\mu\text{J} \\ t_{listen} &= (t_{r_{init}} + t_{r_{on}} + t_{rx/tx} + t_{sr}) \times \frac{1}{t_i} \\ E_{listen} &\leq E_{sample} \times \frac{1}{t_i} \end{aligned} \tag{4}$$

Sensors are an integral part of wireless sensor networks and must be considered when calculating a node's lifetime. Sampling sensors is often expensive and affects the lifetime of the node. The sampling parameters (shown in Table 2) are based on an application deployed by Mainwaring et. al. [13]. In their application, each node takes 1100 ms to start its sensors, sample, and collect data. The data is sampled every five minutes, or $r = 1/(5*60)$. The energy associated with sampling data, $E_d$, is

$$\begin{aligned} t_d &= t_{data} \times r \\ E_d &= t_d \times c_{data}[b] \end{aligned} \tag{5}$$

Finally, the node must sleep for the remainder of the time. The sleep time, $t_{sleep}$, is simply the time remaining each second that's not consumed by other operations.

$$\begin{aligned} t_{sleep} &= 1 - t_{rx} - t_{tx} - t_d - t_{listen} \\ E_{sleep} &= t_{sleep} \times c_{sleep} \end{aligned} \tag{6}$$

The lifetime of the node, $t_l$, is dependent on the total energy consumed, $E$, and the battery capacity, $C_{batt}$. We must bound the lifetime by the available capacity of the battery.

$$t_l = \frac{C_{batt} \times V}{E} \times 60 \times 60$$

By solving the system of equations (1 through 6) and entering the parameters in Table 3, we can find the minimum energy for a given network configuration. Let's start by looking at the effect sample rate has on how we configure B-MAC's LPL parameters.

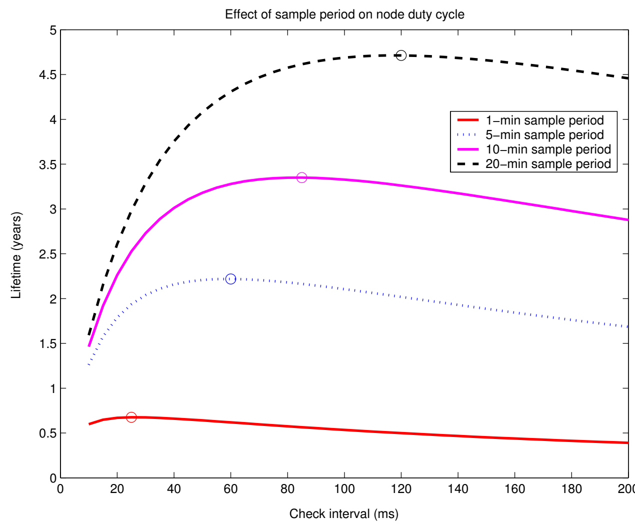


**Figure 3: The lifetime of each node is dependent on the check interval and the amount of traffic in the network cell. Each line shows the lifetime of the node at that sample rate and LPL check interval. The circles occur at the maximum lifetime (optimal check interval) for each sample rate.**

In a typical deployment, scientists will determine the physical location of the nodes (which affects each node's neighborhood size, $n$) and the ideal sampling rate, $r$. With this information, we can calculate the parameters to attain the best lifetime that B-MAC can achieve. This also provides the scientist with an estimate for how long the network will live. Assuming we have a network with approximately 10 neighbors per node, the optimum LPL check interval changes with sample rate. Increasing the sample rate increases the amount of traffic in the network. As a result, each node hears more packets. We must find the optimal amount of time to sleep $t_i$ such that we maximize the lifetime $t_l$. Lowering $t_i$ also lowers the preamble length. This means that the time to transmit and receive a packet is shorter and the radio is sampled more often. The tradeoff of more frequently checking the radio in order to shorten the packet transmission time is shown in Figure 3. Notice that the penalty for more idle listening than required by the traffic pattern, left of the maximum lifetime point in Figure 3, is much more severe than the penalty for

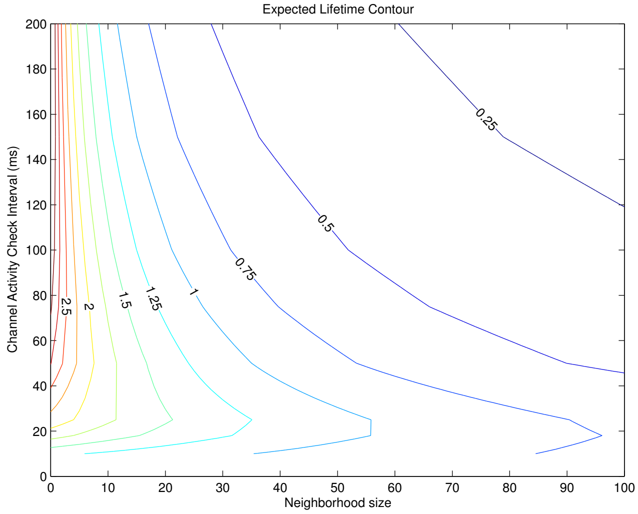


**Figure 4: Contour of node lifetime based on LPL check time and network density. If both parameters are known, their intersection is the expected lifetime using the optimal B-MAC parameters.**

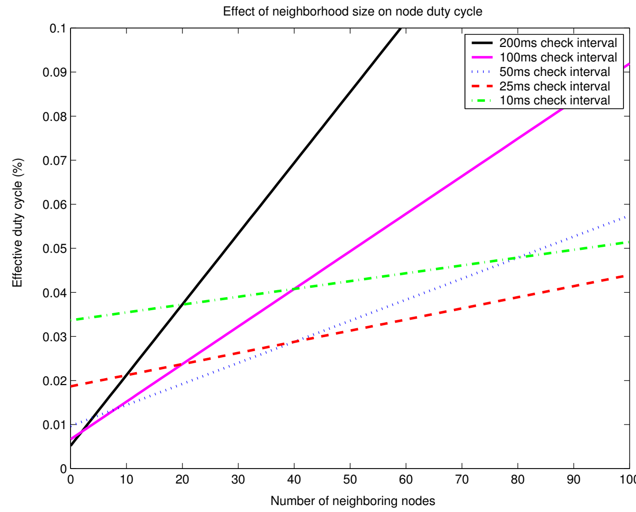


**Figure 5: The node's duty cycle is affected by the network density and LPL check interval. Typically used LPL check intervals in B-MAC's implementation are depicted. The best check interval is the lowest line at a given network density.**

sending packets that are longer than necessary.

If we fix the sample rate and vary the network density, $n$, instead, we can perform the same calculations to find the optimal parameters to maximize lifetime. Changing the neighborhood size in the real world can be done by altering the node's RF output power. Solving the same system of equations with the sample rate equal to once every 5 minutes yields Figure 4. Taking a few slices across the figure at realistic check intervals is shown in Figure 5. To find the best LPL check interval for an application, find the expected neighborhood size in Figure 5 and move up the y-axis to the lowest line. The check interval corresponding with this line will yield the maximum lifetime. For example, a check interval of 50 ms is optimal for a neighborhood size of 20, but if the neighborhood size is only 5, a check interval of 100 ms is optimal. As in Figure 3, the size of the neighborhood affects the amount of traffic flowing by each node. As a result one must trade off idle listening for a reduced time to transmit and receive.

## 4.2 Adaptive Control

Our model shows that it is advantageous to change the parameters of the MAC protocol based on changing network conditions. Since sensor networks are inherently volatile in terms of network stability, it is likely that links will appear or disappear over time [20, 23]. Nodes may join and leave the network, or the size of the neighborhood will change due to changes in the physical environment. The MAC protocol must be able to adjust for these changes and optimize its power consumption, latency, and throughput to support the services relying on it. The analytical model allows the node to recompute the check interval and preamble length. To address reconfiguration, we chose not to implement the functionality in the MAC protocol as Woo and Culler [19] have done. Instead, we created a `MacControl` interface that allows services to change the MAC protocol based on their current operating conditions.

Using the `MacControl` interface, services can tweak the MAC protocol's parameters. Although B-MAC supports a variety of operations within the protocol, of importance is the ability to turn them on or off and modify control parameters.

The most basic control functionality allows services to turn the clear channel assessment on or off. By disabling CCA, a scheduling protocol may use B-MAC in its implementation. If CCA is enabled, B-MAC will query services using it to determine the initial packet backoff and the packet backoff when the channel is busy. Enabling or disabling CCA and configuring the backoff allows services to change the fairness and available throughput provided by the protocol.

By factoring out more complex parts of conventional MAC protocols, services can decide which situations warrant the use of additional control. B-MAC provides optional link-layer acknowledgment support. If acknowledgments are enabled by a service using B-MAC, it receives notification of a successful acknowledgment when each transmission completes. The service may choose to retransmit the packet, change the packet's destination, or reconfigure the LPL parameters.

One scheme implemented above B-MAC is an RTS-CTS channel acquisition protocol. The RTS and CTS packets are sent using LPL. Once the channel is acquired, packets are sent with CCA and LPL disabled. After the acknowledgment, both nodes reenable LPL and return to sleep. In addition to our RTS-CTS implementation, WiseMAC [5], Flexible Power Scheduling [8], broadcast flood, and multihop routing can take advantage of the B-MAC flexibility.

## 5. EXPERIMENTAL METHOD

To evaluate the efficacy of B-MAC as a low power MAC protocol that meets the goals in Section 1, we devised microbenchmarks to strengthen our model and a macrobenchmark to evaluate the model. Our microbenchmarks show the effect of different network conditions on the performance of B-MAC and S-MAC. The microbenchmarks in Section 6 create an empirical model of B-MAC's behavior. The empirical model augments the system of equations presented in Section 4 such that one can predict the performance of B-MAC in any sensor network application. We use classical benchmarks such as total throughput to evaluate MAC protocol performance. But classical benchmarks are not sufficient for wireless sensor network protocols–just as conventional protocols are not suitable for this realm. To evaluate B-MAC and S-MAC in sensor network scenarios, we examine low duty cycle operation in terms of energy consumption as it relates to throughput, latency, and message fragmentation. The purpose of our microbenchmarks is the creation of an empirical model that shows how each protocol performs in a wide array of network conditions. To show the accuracy and usability of our model, we deployed B-MAC in a real world application called Surge. Surge is an environmental monitoring data collection application that runs at extremely low duty cycles. The results of the Surge deployment are described in Section 7

Both B-MAC and S-MAC were implemented in TinyOS. We used the Mica2 [16] wireless sensor nodes to perform our tests. All tests occurred in an unobstructed area with line of sight to every other node. To enable multihop networking, we reduced the RF output power of the node to its minimum value and placed the nodes with 1 meter spacing. Nodes were elevated 15 centimeters to reduce near-field effects.

To determine the power consumption of each protocol, we implemented counters in the MAC protocol that kept track of how many times various operations were performed. For B-MAC, this includes receiving a byte, transmitting a byte, and checking the channel for activity. For S-MAC, we counted the amount of time that the node was active, number of bytes transmitted and received, and the additional time the node spent awake due to adaptive listening. Since the S-MAC implementation does not actually put the node into sleep mode, we had to measure the power consumption indirectly by multiplying the cumulative time of each operation with the expected power to operate in that mode. All of our data assumes that S-MAC actually enters sleep mode even though the implementation does not. The power consumption of each operation is taken from Table 2.

In addition to tests on real sensor network hardware, we simulated T-MAC [18] in Matlab. We calculated the time of each operation and multiplied by the current consumption in Table 2 to get the overall expected power consumption of the T-MAC protocol. Since there is no T-MAC implementation (neither in embedded C nor TinyOS), we were unable to do a direct comparison between B-MAC, S-MAC, and T-MAC.

To measure latency, each node was connected to an oscilloscope. Upon submission of a packet to the MAC protocol, we toggled a hardware pin. When a packet was received, a different pin was toggled. Using an oscilloscope, we captured the time between each pin toggling to yield the total latency.

In some graphs we show the optimal solution. This solution is found by determining a perfect schedule such that the total transmit and receive costs are minimized. In this solution, all nodes are perfectly synchronized with no additional overhead. This metric serves to show the effect of overhead in our MAC protocols. To calculate the power consumption, the transmit and receive times are multiplied by the power to perform those operations.

In all cases, we measure the *data throughput* of the network. This factors out the protocol-specific overhead to evaluate the amount of data that can be delivered by each protocol and the cost of delivering that data. In all tests where we mention “packet size”, we are referring to the size of the data payload only, not the header information.

## 6. MICROBENCHMARK ANALYSIS

In this section we use a variety of microbenchmarks to evaluate B-MAC. These benchmarks show the effect of a wide array of network conditions on the energy consumption of B-MAC, S-MAC, and T-MAC. Using the results of the microbenchmarks, the behavior of B-MAC in a real world sensor network application can be predicted.

### 6.1 Channel Utilization

Channel utilization is a traditional benchmark for MAC protocols. It shows how efficiently a protocol can deliver large amounts of data. In sensor networks, high channel utilization is critical for delivering a large number of packets in a short amount of time. By minimizing the time to send packets, we can also reduce the network contention. In bulk data transfer protocols, such as wirelessly reprogramming a node (commonly referred to as *network reprogramming*), one wishes to wake up the network and reprogram as quickly as possible. To find the channel utilization under congestion, we placed $n$ nodes equidistant from the receiver. Each node transmitted as quickly as possible with the MAC protocol providing collision avoidance. There was no node or radio duty cycling in this test. The results are shown in Figure 6.

In general, better throughput is attained with fewer nodes trying to saturate the channel. With only a few transmitters, we can examine how much the protocol affects an application. In other words, if the protocol alone can saturate

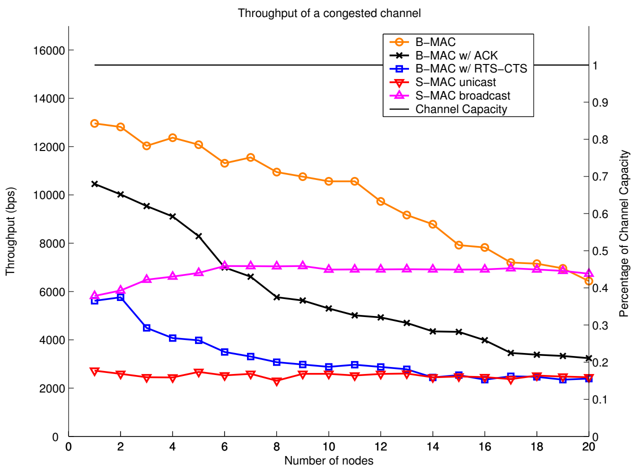


**Figure 6: Measured throughput of each protocol with no duty cycle under a contended channel. The throughput of each protocol is affected by the amount of nodes contending for the channel and the protocol's overhead. B-MAC achieves over 4.5 times the throughput of the standard S-MAC unicast protocol.**

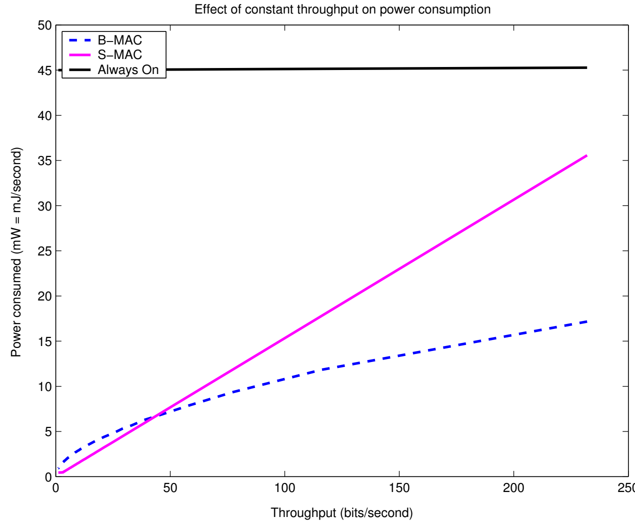


**Figure 7: The measured power consumption of maintaining a given throughput in a 10-node network. As the throughput increases, the overhead of S-MAC's SYNC period causes the power consumption to increase linearly. The throughput is the average node bitrate (number of data bytes sent in a 10 second time period) in the 10 node network.**

the channel, then the application can flexibly decide how it wants to use the channel. With one transmitter, B-MAC outperforms S-MAC broadcast mode by 2.5 times and S-MAC unicast mode by 4.5 times. The reduced performance in S-MAC means the protocol is hindering the performance of the application in scenarios such as bulk transfers. B-MAC exceeds the performance of S-MAC, but does not trade off fairness in the process. The test in Figure 6 uses a short initial random backoff proven in [19] to provide fair channel utilization. By analyzing our dataset, we found that each node in the test achieves no more than 15% more bandwidth than any other node. To yield even higher throughput with B-MAC, we can discard fairness as a requirement. Each transmitter can set its backoff to zero and take control of the channel.

In network reprogramming, minimizing the number of transmitters is valuable for a several reasons–less state must be maintained to choose a receiver to request pages and it reduces the chance of hidden terminals. Having fewer transmitters maximizes the bandwidth used as shown on the left side of Figure 6.

To illustrate the effectiveness of a configurable MAC protocol for sensor networks, we implemented an RTS-CTS scheme above the B-MAC protocol. The RTS-CTS scheme illustrates the effect of using control packets to remove the hidden node problem and increase fairness. On each packet transmission, an RTS packet was sent. A CTS response is sent if the destination node is idle and not delaying due to other transmissions. Upon a successful RTS-CTS exchange, the data packet is sent immediately with CCA turned off. Upon reception of the acknowledgment, CCA is reenabled. Since B-MAC can utilize 2.5 times more of the channel using the CCA algorithm from Section 3, the RTS-CTS implementation using B-MAC actually provides double the throughput of S-MAC. When many nodes compete for the channel, B-MAC with RTS-CTS support provides identical performance to S-MAC. From this test, we can conclude that implementing a CDMA scheme using B-MAC does not hinder performance.

## 6.2 Energy vs. Throughput

We designed B-MAC to run at both low and high data rates configured by the services relying on it. Low duty cycle applications such as environmental monitoring data collection have extremely low network throughput; however they often have other parts of the application such as bulk data transfer that stress the high throughput functionality of the MAC protocol. To evaluate how increased throughput affects power consumption, we vary the transmission rate of 10 nodes in a single cell. We bound the latency such that the data must arrive within 10 seconds. For B-MAC, the optimal check interval $t_i$ is calculated for the traffic pattern, the test is run, and the energy consumption is calculated. For S-MAC, the optimal duty cycle is calculated for the traffic pattern such that the data arrives within the 10 second period. The results are shown in Figure 7.

At low data rates, S-MAC can use an extremely low duty cycle to transmit and receive the data. As the amount of data increases, so does S-MAC's duty cycle. When the duty cycle increases, there are more active periods each with a dedicated SYNC period. Due to the overhead of the SYNC period at the beginning of each wakeup, the S-MAC energy consumption increases linearly. In B-MAC, at low throughput we send long preambles with a long check interval $t_i$. Because of the tradeoff between idle listening and packet length, the overhead dominates the energy consumption. The overhead of LPL is mitigated by a more frequent check interval and lower cost of transmission and reception when the throughput exceeds 45bits/second. Note that B-MAC's power consumption below 45bits/second is within 25% of S-MAC's power consumption; however, B-MAC has significantly less state and no synchronization requirements. B-MAC can easily change the check time based on the network bandwidth whereas S-MAC must create a new schedule and force surrounding nodes to resynchronize.

### 6.3 Fragmentation

Small periodic data packets are the most common workload in sensor networks, but certain cases arise where larger transfers are needed. S-MAC supports large message fragmentation within the MAC protocol. To compare B-MAC directly to S-MAC's design goal of efficient message fragmentation, we devised an experiment to match those done by the authors of S-MAC in [22]. Since S-MAC supports packet fragmentation to do bulk data transfer with a single RTS-CTS exchange, we chose to see how B-MAC compares since it has no fragmentation support. Using the network configuration in Figure 8, we transmitted packets from sources A and B to sinks D and E by routing through C.

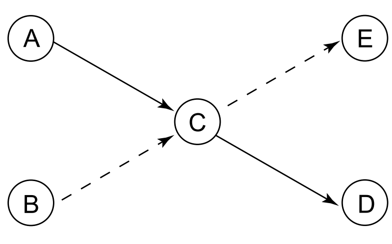


**Figure 8: X network configuration used for packet fragmentation tests.**

As in [22], our test sent 10 fragments per message. We varied the fragment size and measured the energy consumed for that transfer. C is the energy bottleneck node in the test network–C will cease operation before any other node since it must both receive and relay packets. We evaluate the energy required to deliver the fragments with S-MAC running at 10% duty cycle with adaptive listening. B-MAC is configured with the default parameters from Table 3. Figure 9 shows the energy cost per byte at node C when a message (consisting of 10 fragments) is sent every 10 seconds. Figure 10 show the energy cost per byte when the message generation interval is once every 100 seconds.

B-MAC without application control is simply the B-MAC protocol with the default parameters. Each fragment is considered a separate and independent packet. But, if middleware provided a bulk data transfer interface in addition to a single packet transmission interface, the middleware service could adjust the MAC protocol to minimize the energy con-

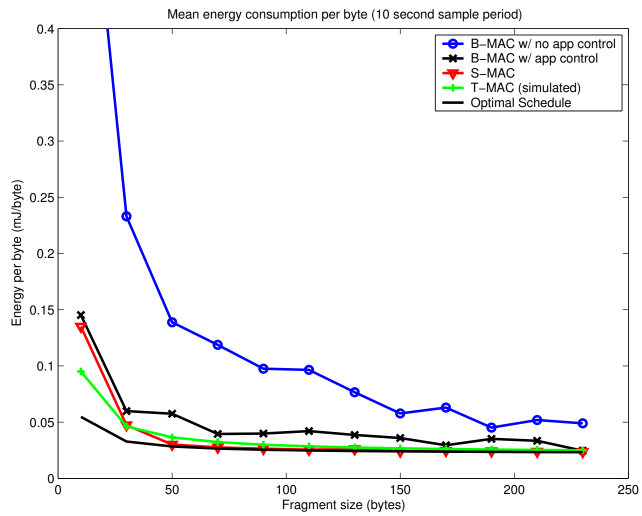


**Figure 9: The effective energy consumption per byte at node C for a network as shown in Figure 8. Each node generates a message every *10* seconds consisting of 10 fragments of the size given on the x-axis.**

sumed. In the application controlled measurements, the first fragment of the message was sent with LPL enabled and extra bytes to inform the receiver of the number of fragments. The remaining fragments are sent with LPL disabled. After the last fragment, the sender and receiver reenable LPL. With this flexibility, we achieve significant power savings and efficiency essentially identical to S-MAC without the additional protocol overhead including RAM and ROM usage. When the message generation time is large (as in Figure 10), the overhead of S-MAC and T-MAC are readily apparent as their energy consumption per byte is greater at all fragment sizes than that of B-MAC with assistance from the application. When there is no activity on the channel, T-MAC removes overhead by using adaptive active periods to return to sleep much quicker than S-MAC. Figure 9 and Figure 10 show that the energy cost of breaking up a short message into even shorter fragments is so high in all of our protocols that it is simply not a viable option in sensor networks.

### 6.4 Latency

The authors of S-MAC argue that when the MAC protocol is permitted to increase latency, S-MAC can reduce the node's duty cycle and conserve energy. A test for evaluating end-to-end latency was devised in [22]. We have reproduced their tests to provide a direct comparison between B-MAC and S-MAC. The test is run on a 10-hop network shown in Figure 11. In each test, the source node sends 20 messages that are 100-bytes long. There is no fragmentation on any messages.

The latency at each hop of the network was measured with the method described in Section 5. S-MAC was tested at a 10% duty cycle with adaptive listening. B-MAC was tested with the default parameters. The results are shown in Fig-

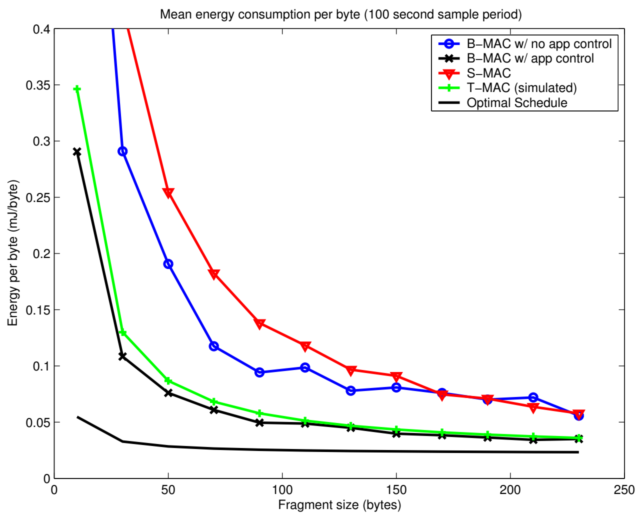


**Figure 10: The effective energy consumption per byte at node C for a network as shown in Figure 8. Each node generates a message every *100* seconds consisting of 10 fragments of the size given on the x-axis.**

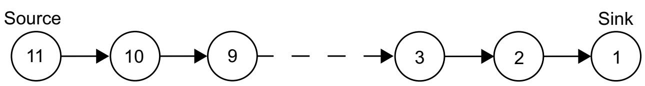


**Figure 11: 10 node configuration used for multihop end-to-end latency tests.**

ure 12. We were able to reproduce the latency data from [22] and it fits the previously published results.

The latency of B-MAC and S-MAC increase linearly. When duty cycling is disabled, the effect of RTS-CTS exchanges in S-MAC result in a much steeper slope. In low power communications, B-MAC has a slope almost identical to that of S-MAC with adaptive listening, however the y-intercept is much lower. Since the first packet cannot be sent until an S-MAC active period, it is delayed by at most 1150ms. Through adaptive listening, S-MAC does not incur an expected 1150ms additional delay at each hop. One protocol feature of B-MAC, link layer acknowledgments, increases latency by an insignificant amount.

To better evaluate the effect of increasing the latency to reduce power consumption, we fixed the throughput to one 100 byte packet per 10 second interval. We measured the end-to-end latency of the 10 hop network and varied the duty cycle of S-MAC. We also chose the optimal $t_i$ for B-MAC given the latency and throughput. The results are shown in Figure 13.

For latencies under 6 seconds, B-MAC performs significantly better than S-MAC. As S-MAC approaches the 10 second latency limit, its power dips below that of B-MAC. This phenomenon is similar to that seen when evaluating the energy consumed while varying the throughput in Figure 7. When

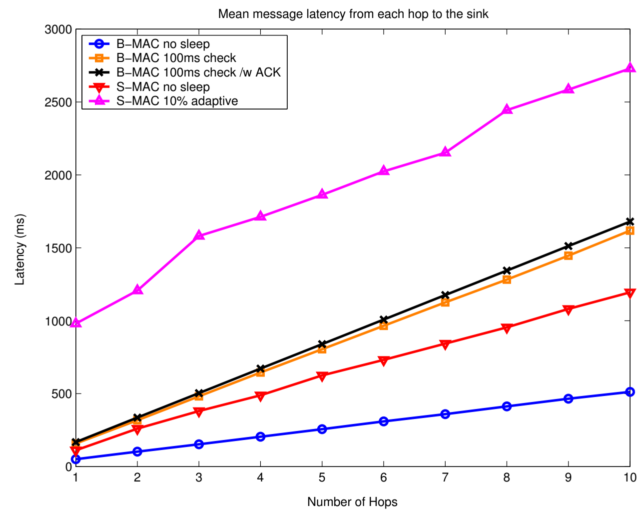


**Figure 12: The end-to-end latency is a linear function of the number of hops in the network. As the overhead of the MAC protocol increases, so does the slope of the latency.**

the latency exceeds 3 seconds, B-MAC's power consumption is bounded by the cost of idle listening. In contrast, the best case performance of S-MAC shown in Figure 13 relies on S-MAC synchronizing the entire 10-hop network and using adaptive listening to transmit the data through the network in one active period. In most cases, the network will not be completely synchronized and instead the mean network energy will be larger due to border nodes maintaining multiple schedules.

## 7. MACROBENCHMARK ANALYSIS

The results from the microbenchmarks presented in Section 6 combined with the analytical model in Section 4 creates a complete model of the B-MAC protocol under varying network conditions. This information can be used to predict the performance of an application using B-MAC. To evaluate the accuracy of our predictions, we implemented a real world application for environmental monitoring data collection.

Surge is our data collection application. Surge periodically reads data from the node's sensors, sends the readings, and sleeps. It was deployed by spreading a handful of nodes throughout the environment. The nodes automatically configured themselves into an ad-hoc routing network. They determined the initial network topology and continually monitor changes in network topology. Surge collects data from each node once every three minutes. Instead of collecting sensor data, we acquired statistical information from the B-MAC protocol about its performance (see Section 5 for the energy indicators implemented in B-MAC). For each node, Surge transmits the energy usage, battery voltage, link quality to surrounding nodes, and the parent selected. Data from Surge is used to verify that the network performance matches

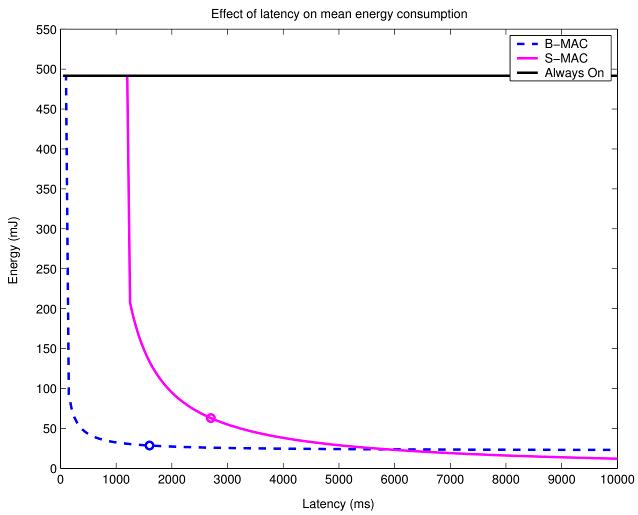


**Figure 13: As the latency increase, the energy consumed by both B-MAC and S-MAC decreases. The point illustrated on the B-MAC line is the default configuration as shown in Table 3. The point on the S-MAC line is the default S-MAC configuration at 10% duty cycle with adaptive listening.**

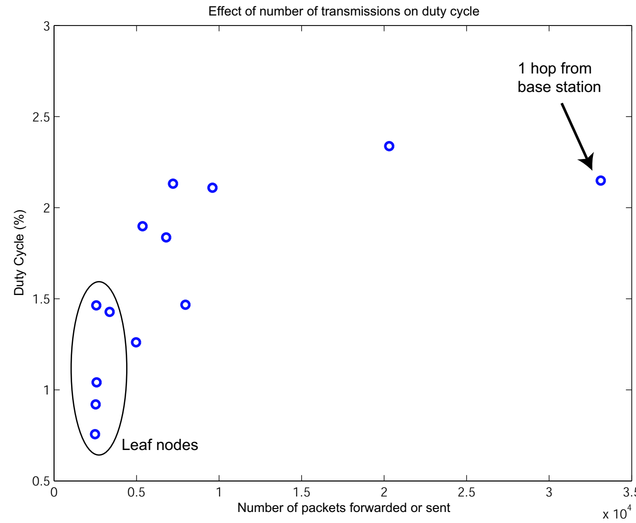


**Figure 14: As the traffic around a node increases, so does the duty cycle when using the B-MAC protocol with LPL.**

the prediction generated by our model.

Using our analytical estimate of lifetime versus check interval for a given sample period (see Figure 3), we selected a 100ms check interval $t_i$. By evaluating the node communication range and overall size of the network, we expected a maximum neighborhood size of 10 nodes. We chose to use the maximum neighborhood size instead of the average because the penalty for a longer check interval is less harsh than checking too often (as shown in Figure 5). Using parameters for a single cell network, the worst case node lifetime of 2 years on 2 AA batteries. Folding in the multihop forwarding analysis from the model, we anticipate the worst case lifetime is 0.75 years. In the worst case, one node routes data for all other nodes in the network. We expect that nodes on the edge of the network–that have a smaller neighborhood–will have a longer lifetime.

We deployed a 14-node Surge network. Since the most important thing in a monitoring network is reliably reporting the data, we must confirm that B-MAC successfully supports multihop data reporting. Our implementation uses the default multihop routing protocol from the TinyOS distribution, called MintRoute. After the trivial process of integrating B-MAC with MintRoute, the Surge application was run for a period of 8 days. We collected over 40,000 data packets profiling the performance of a real world wireless sensor network. Since MintRoute integrates seamlessly with B-MAC, it was a fair comparison to try to run S-MAC with this multihop routing implementation. At the time of this writing there is no power-aware multihop routing implementation that interfaces directly with S-MAC.

During the Surge deployment, the network yielded over 98.5% packet delivery while some nodes achieved an astounding 100% success rate. In our deployment, there were a total of 71 times where a node decided to change its parent–and consequently changed the routing tree–as a result of environmental changes altering the communication topology. In confirmation of Woo and Zhao's identification of gray areas, high-quality links that were stable for hours were intermittently broken due to environmental changes.

It is likely that nodes in the center of the network will communicate with a majority of the network. The base station was placed at a convenient location to install infrastructure. This location was in a corner of the network. From the data collected by the Surge application, we can determine the actual duty cycle of our deployed network. From the duty cycle, we can extrapolate the network lifetime. Figure 14 shows the measured duty cycles of each node in the network. The worst case duty cycle of 2.35% indicates that the first node should exhaust its battery supply approximately 1 year into the deployment. Our data shows that each node has an average of 5 neighbors, less than our maximum estimate of 10. Additionally, the nodes on the edge of the network have less than a 1% duty cycle. We attribute the lower duty cycle to the significant reduction in traffic at the edges of the network as compared to central nodes routing a large amount of data.

In addition to predicting the power consumption of the network, our empirical model also predicts the network latency that is introduced by the B-MAC protocol illustrated in Figure 12. The latencies measured experimentally in our Surge deployment along with the predicted values are plotted in Figure 15. The average latency is slightly higher than the prediction. However the minimum latency for each network level exactly matches the predicted value. The difference

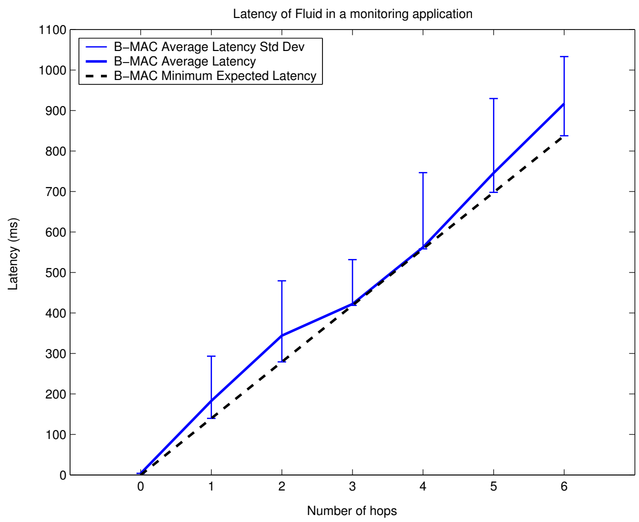


**Figure 15: The average latency of packets being delivered by the Surge application is dependent on the traffic in the network and reliability of each link. The average exceeds the expected latency because retransmissions add additional latency to each packet.**

between the average and minimum latencies is due to unexpected network congestion and packet loss. Surge instructs B-MAC to retransmit packets on failed acknowledgments to increase network reliability. Upon retransmission, the packet incurs additional latency.

In Section 6 we discussed how application control over MAC parameters can greatly improve overall performance. Although the network is homogeneous, we can exploit the fact that the base station runs with a different duty cycle (since it is always on) than the data collection network. Based on this assumption, nodes communicating directly with the base station–although they are homogeneous with respect to the rest of the network–can reconfigure B-MAC for packets sent to the heterogeneous node. Instead of sending packets with long preambles, the nodes one hop away can send packets with only an 8 byte preamble to the base station. Figure 14 shows the node duty cycle versus number of packets routed through the node. There was one node that despite forwarding almost 35,000 packets (about 85% of all data packets), had a duty cycle equal to nodes forwarding less than 10,000 packets. Figure 16 shows the duty cycle based on which level of the multihop routing tree a node resided. The node routing 35,000 packets one hop from the base station was able to alter its MAC behavior based on its position in the network in order to optimize performance. Without this optimization, the power consumption would have been 75% higher. The powered base station is a simple example of heterogeneity; one could imagine much more network heterogeneity with a hierarchy of devices. The differences between devices dictate other optimizations that could be made by notifying the MAC protocol.

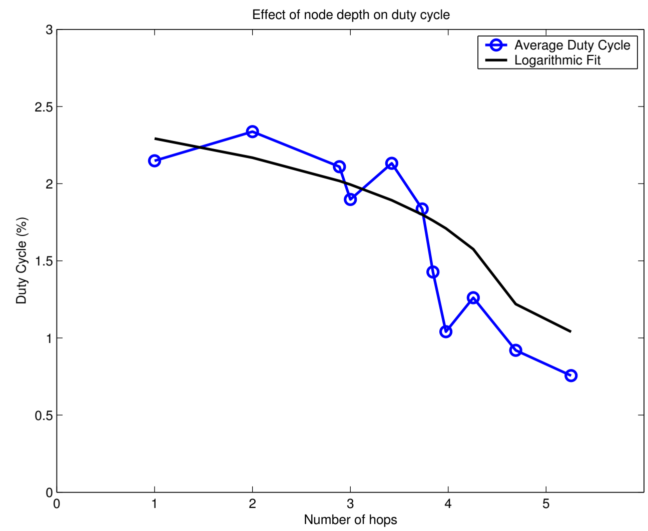


**Figure 16: The position in the network dictates the amount of traffic flowing through that level. As we approach the base station, nodes handle an increasing number of packets since the subtree below each node cannot decrease. Note that the nodes one-hop away from the base station can achieve a lower duty cycle because the base station is always on. The multihop routing layer reconfigures B-MAC to use short packets when sending to the base station.**

## 8. DISCUSSION

Our benchmarks have shown that a small amount of information from services using B-MAC can provide significant power savings. Additional power savings can be achieved through more information about the application and its operation. In this section, we discuss the implications of these ideas.

To mitigate the cost of reception incurred with B-MAC with LPL, a packet could be sent cyclically with a short preamble. Although this does not reduce the transmission cost, it reduces the time of receiving a packet to:

$$t_{rx} \leq 2 \times (L_{preamble} + L_{packet}) \times t_{rxb}$$

Note that $L_{preamble}$ is reduced from the long LPL preamble to only $8$ bytes. The node can return to sleep for the check interval after receiving a packet or can perform early rejection much quicker than packets sent with the long preambles.

We showed in Section 7 that using information about your parent can assist in reducing the duty cycle. Two optimizations are possible with more information. The first is the node can learn the offset of the check interval $t_i$ that their parent wakes up to sample the channel. Knowing the point at which the parent wakes up allows the node to create a communication schedule to its parent. By starting transmission at the parent's sample time, the preamble can be significantly reduced in size. This optimization reduces both the transmission and reception costs. A similar optimization is proposed by the authors of WiseMAC [5]. Broadcast packets would still be sent using the long preambles such that other nodes can snoop on a significant portion of the network traffic. If the schedule fails due to a change in link quality or parent node failure, retransmission of the data can fall back to the long preamble method.

As each node learns about the average size of its neighborhood and the amount of data flowing through it, it can use the model presented in this paper to recalculate the optimal parameters to maximize its duty cycle. Having a model of the protocol's operation is critical to predict the operation of any sensor network application. In this case, the model must exist to guide the network's adaptation and lower multihop network duty cycles. Children are notified of their parent's check interval and vice versa to optimize their transmission preamble length. If two nodes have conflicting values for the check interval, they can always fall back using the preamble length corresponding with the maximum check interval.

# 9. CONCLUSION

In this paper we presented a MAC protocol that features a simple, predictable, yet scalable implementation and is tolerant to network changes. B-MAC effectively performs clear channel estimation. It runs at extremely low duty cycles and permits services running on the node to reconfigure its operation. B-MAC does not incur the overhead of synchronization and state maintenance like other protocols proposed for wireless sensor networks.

We presented an analytical and empirical model for determining B-MAC's optimal parameters. The model also calculates the estimated node lifetime. Using microbenchmarks we showed that B-MAC can outperform existing wireless sensor network media access protocols with only a small amount of information from the services using it. With the default B-MAC parameters and no additional information, B-MAC still outperforms existing protocols in terms of throughput, latency, and often energy consumption. The performance of B-MAC with varying workloads gave us enough information to make accurate predictions of how a real world application runs. The Surge application operated within the realm of our model and reported a significant amount of data with over 98.5% packet throughput.

Media access reconfiguration is essential for dynamic systems like wireless sensor networks. Since each sensor node runs a single application, optimizing protocol performance for that application in a predictable manner proves the feasibility of this technology for long-term deployments.

## Acknowledgments

This work was supported by the Intel Research Laboratory at Berkeley, DARPA grant F33615-01-C1895 (Network Embedded Systems Technology "NEST"), the National Science Foundation, and the Center for Information Technology Research in the Interest of Society (CITRIS). Special thanks to Wei Ye for his help with S-MAC under TinyOS and also to the TinyOS community for their feedback during the development of B-MAC.

# 10. REFERENCES

[1] N. Abramson. The aloha system: Another alternative for computer communications. In *Proceedings of the Fall 1970 AFIPS Computer Conference*, pages 281–285, Nov. 1970.

[2] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao. Habitat monitoring: Application driver for wireless communications technology. In *2001 ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, San Jose, Costa Rica, Apr. 2001.

[3] Chipcon Corporation. CC1000 low power FSK transciever. `http://www.chipcon.com/files/CC1000_Data_Sheet_2_1.pdf`, Apr. 2002.

[4] A. El-Hoiydi. Aloha with preamble sampling for sporadic traffic in ad hoc wireless sensor networks. In *Proceedings of IEEE International Conference on Communications*, New York, NY, Apr. 2002.

[5] A. El-Hoiyi, J.-D. Decotignie, and J. Hernandez. Low power MAC protocols for infrastructure wireless sensor networks. In *Proceedings of the Fifth European Wireless Conference*, Barcelona, Spain, Feb. 2004.

[6] M. Hamilton, M. Allen, D. Estrin, J. Rottenberry, P. Rundel, M. Srivastava, and S. Soatto. Extensible sensing system: An advanced network design for microclimate sensing. `http://www.cens.ucla.edu`, June 2003.

[7] J. Hill and D. Culler. Mica: a wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, November/December 2002.

[8] B. Hohlt, L. Doherty, and E. Brewer. Flexible power scheduling for sensor networks. In *Proceedings of the Third International Symposium on Information Processing in Sensor Networks*, Berkeley, CA, Apr. 2004.

[9] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networks (MobiCOM 2000)*, Boston, MA, Aug. 200.

[10] J. Jubin and J. Tornow. The DARPA packet radio network protocols. *Proceedings of IEEE*, 75(1):21–32, Jan. 1987.

[11] R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kunzelman. Advances in packet radio technology. *Procedings of the IEEE*, 66(11):1468–1496, Nov. 1978.

[12] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler. TinyOS: An operating system for wireless sensor networks. In *Ambient Intelligence*, New York, NY, 2004. Springer-Verlag.

[13] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, pages 88–97, Atlanta, GA, USA, Sept. 2002. ACM Press.

[14] B. Mangione-Smith. Low power communications protocols: paging and beyond. In *Proceedings of the IEEE Symposium on Low Power Electronics*, 1995.

[15] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler. Lessons from a sensor network expedition. In *Proceedings of the First European Workshop on Sensor Networks (EWSN)*, Berlin, Germany, Jan. 2004.

[16] University of California, Berkeley. Mica2 schematics. `http://webs.cs.berkeley.edu/tos/hardware/design/ORCAD_FILES/MICA2/6310-0306-01ACLEAN.pdf`, Mar. 2003.

[17] University of California, Berkeley. TinyOS CVS Repository at SourceForge. `http://sf.net/projects/tinyos/`, 2004.

[18] T. van Dam and K. Langendoen. An adaptive energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems*, Los Angeles, CA, Nov. 2003.

[19] A. Woo and D. E. Culler. A transmission control scheme for media access in sensor networks. In *Proceedings of the seventh annual international conference on Mobile computing and networking*, Rome, Italy, July 2001.

[20] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of multihop routing in sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems*, Los Angeles, CA, Nov. 2003.

[21] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *In Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, New York, NY, June 2002.

[22] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated, adaptive sleeping for wireless sensor networks. In *IEEE Transactions on Networking*, Apr. 2004.

[23] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems*, Los Angeles, CA, Nov. 2003.