An Energy Efficient MAC in Wireless Sensor Networks to Provide Delay Guarantee

Anirudha Sahoo
Kanwal Rekhi School of Information Technology
Indian Institute of Technology, Bombay, Mumbai, India
Email: sahoo@it.iitb.ac.in

Prashant Baronia
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay, Mumbai, India
Email: prashant@cse.iitb.ac.in

Abstract— **This paper presents RTMAC, a realtime MAC protocol for wireless sensor network that can provide delay guarantee. RTMAC is based on TDMA protocol, but it is carefully designed to overcome the high latency of traditional TDMA protocols. It also conserves energy when a node may not be transmitting or receiving packets. We discuss the details of time slot assignment procedure of RTMAC and then present delay analysis of the protocol. We compare the performance of RTMAC with the well known energy efficient MAC protocol S-MAC using simulation. The simulation results show that RTMAC is better than S-MAC in terms of providing delay guarantee to packets.**

I. INTRODUCTION

Wireless sensor network is an emerging technology with wide range of potential applications such as environment monitoring, earthquake detection, patient monitoring systems etc. Sensor networks are also being deployed for many military applications, such as target tracking, surveillance and security management [1]. Such a network normally consists of a large number of distributed nodes that organize themselves into a multi-hop wireless network. Each node has one sensor, embedded processors and low-power radio and is normally battery operated. Sensors have limited energy resources and their functionality continues until their energy is drained. Therefore, energy for sensor networks should be managed carefully to extend the lifetime of sensors. The sensing element of a sensor probes the surrounding environment. After performing signal processing of the observed data, sensors communicate this data, typically using a radio-based short-haul links, to a cluster head or to a base station. For sending sensed data, a sensor node has to access the medium and then transmit the data. Thus, in a distributed system like wireless sensor network, medium access control (MAC) protocol plays an important role. There are primarily two types of MAC protocols for sensor network: contention based and TDMA based. In contention based MAC, the nodes can transmit without having any predetermined time assigned to them. Thus, this may result in collision. Hence, the protocol provides mechanism for resolving and avoiding collision. TDMA based protocols are collision free because each node has a designated time slot in which only that particular node transmits. MAC protocols for wireless sensor networks should be designed carefully so that they do not consume lot of energy. In addition, if the sensor network is to be used for real time applications, the MAC protocol should have low latency.

In this paper, we propose a Time Division Multiple Access (TDMA) based energy efficient MAC protocol, called RTMAC, that can support delay guaranteed communication. Traditional TDMA protocols suffer from high latency. But, RTMAC uses channel reutilization technique to reduce the latency between two successive channel access of a sensor node. Further, RTMAC allows sensors to go to sleep when they are not communicating (no transmission or reception) and hence it conserves energy. Since RTMAC can provide delay guarantee, it is suitable for realtime applications like detection of radioactive radiation, earthquake etc. We present the method by which time slots are assigned to sensor nodes. Then we present the delay analysis of RTMAC to show that the delay is bounded in RTMAC. We compare the performance of our protocol with S-MAC, a well known energy efficient MAC protocol for wireless sensor network, (more discussion on S-MAC in Section III). We choose latency and energy consumption as the performance parameters for the comparison. Although S-MAC performs better than RTMAC, we show that RTMAC provides delay guarantee, which cannot be provided by S-MAC.

The rest of the paper is organized as follows. Section II outlines some of the related work in this area. In Section III, we describe salient features of S-MAC protocol which is a well known energy efficient protocol for wireless sensor network. Section IV describes the deployment scenario for our work. In Section V, we explain the control plane of RTMAC in detail. Section VI deals with the data plane of RTMAC, where we explain the energy efficiency and delay analysis of RTMAC. We present performance analysis of RTMAC in Section VII. Finally, we conclude our paper in Section VIII.

II. RELATED WORK

Wireless sensor MAC protocols are broadly classified into two types: TDMA based, Contention based. TDMA based protocols are contention free protocols in which sensor nodes communicate in their assigned time slot [2]. Sohrabi et al. proposed a self organizing MAC for sensor network in which each node maintains a TDMA-like frame called superframe [3]. Interference between adjacent links is avoided by using FDMA and CDMA in potentially interfering links. In Contention based MAC protocols multiple nodes may access the medium simultaneously resulting in collision. The protocol provides mechanism to avoid collision. Standardized IEEE 802.11 distributed coordination function (DCF) [4] is one such protocol. Woo et al. [5] have studied different configuration of carrier sense multiple access (CSMA) and proposed an adaptive rate control mechanism to achieve fair bandwidth allocation to all nodes.

A survey of MAC protocols for wireless sensor networks has been reported in [6]. In [7], the authors discuss various MAC protocol design goals and the tradeoff associated with them. In [8], the authors pointed out that sensors spend lot of energy to perform idle listening. Hence they proposed a MAC protocol called *S-MAC* which conserve energy by having listen

[1]This research was partially supported by IRCC, IIT Bombay under grant number 03IR059 and by Ministry of Communications and Information Technology, Government of India, under grant number 1(1)/2004-E-Infra.

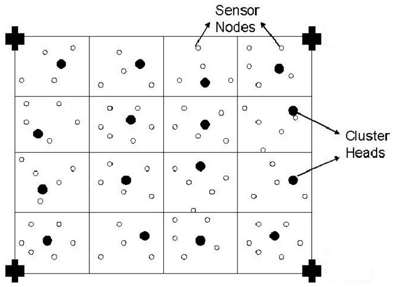

Fig. 1. Topology of Sensing Area

and sleep cycles. S-MAC has low energy consumption, but has higher latency. Several extensions and modifications to S-MAC like WiseMAC [9], TMAC [10] and DMAC [11] have been proposed to improve its performance. An important feature of wireless sensor networks is the in-network data processing, since data aggregation or other techniques can greatly reduce energy consumption by largely reducing the amount of data to be transmitted. A time-synchronized link protocol for real-time wireless sensor network is proposed in [12]. In [13] the authors discuss the impact of data aggregation in wireless sensor network.

III. S-MAC

S-MAC is a well known energy efficient MAC for wireless sensor network [8]. But latency in this protocol is a concern. We have compared our RTMAC protocol with S-MAC, hence we describe the salient features of operation of S-MAC in this section. S-MAC is contention-based protocol similar to 802.11, but has better energy efficiency than 802.11. S-MAC is designed to reduce energy consumption from all sources of energy wastage like idle listening, collision and overhearing. S-MAC uses a coarse-grained listen-sleep cycle. A complete listen-sleep cycle is called a *frame*. Each frame begins with a listen period for nodes that have data to send. A sleep period follows, during which nodes sleep if they have no data to send or receive. Nodes remain awake and exchange data if they are involved in communication. All the nodes are free to choose their own listen-sleep schedules. Nodes share their schedules with their neighbors so that communication between all nodes is possible. Nodes then schedule transmissions during the listen time of their neighboring nodes. To reduce control overhead, neighboring nodes adopt identical schedules. The collision avoidance mechanism in S-MAC is similar to that in the IEEE 802.11 DCF. For more details on S-MAC please refer to [8].

IV. Deployment of the Sensor Network

A. Topology

We assume that the sensing area is divided into rectangular grids as shown in Figure 1. We refer to each grid as a *cluster*. We further assume that each cluster has a cluster head which is a more powerful node than the sensors. The focus of this paper is to provide delay guaranteed communication in a cluster. Hence, RTMAC operates in each cluster to provide a delay guaranteed service from sensors in the cluster to its cluster head. Cluster heads should be approximately in the middle of the cluster. Although RTMAC does not have this requirement, having the cluster head in the middle increases the efficiency of the protocol, since this allows higher channel reutilization. The sensors should be deployed around the cluster somewhat uniformly distributed in the area.

B. Coordinates of the Sensors

One of the main assumptions made while designing RT-MAC is that we know the angle of each node with respect to the geographical north passing through the cluster head. We assume that any of the existing methods, some of which are described next, can be used to assign the angular position of sensors with respect to geographic north. There are many distributed algorithms reported in literature to find the coordinates of nodes in a sensor network. There are many different localization systems that depend on having direct distance estimates to globally accessible beacons such as the Global Positioning System. Recently there has been some research in localizing sensors in a wireless sensor networks when there are no globally accessible beacons. Most of the techniques make use of seed nodes to approximately calculate the coordinates of each node. Savvides et al. describe a distributed localization algorithm that recursively infers the positions of sensors with unknown position from the current set of sensors with known positions, using inter-sensor distance estimates [14]. In [15], the authors analyze accumulation of error with each inference and parameters affecting the error. Their algorithm does not rely on inter-sensor distance estimates and is fully distributed, and one can theoretically characterize how the density of the sensors affects the error.

V. Control Plane of RTMAC

When the sensors are deployed, the sensors will configure themselves and will be ready to run RTMAC in the data plane. The first task of each sensor node is to find out its distance from the cluster head in terms of hops. Once it is done, then the cluster would assign the time slots to each sensor.

A. Assumptions

RTMAC is designed under the following assumptions:

- The sensor nodes as well as the cluster head are stationary
- All the nodes are similar to each other in terms of battery power and transmission range.
- The interference range and the transmission range of the sensors are the same.
- Each node has a unique ID in the network.
- The event rate is low enough so that there will not be any queueing delay in the sensor nodes.
- The network carries constant size small packets (which represent the information about the event).
- The clocks of sensor nodes are synchronized by using out-of-band time synchronization sources in the cluster head. A similar mechanism as used in [12] may be used for this purpose.
- Propagation delay is neglible.

B. Hop Count Calculation

When the sensor nodes come up in the deployment area, they would run the following algorithm to find their distances from the cluster head.

- All the nodes broadcast a *HELLO* message which carries the node id, and a counter value. The initial value of the counter is set to 1.
- When an intermediate node receives this message, it increments the counter and transmits the message if this is the first time it received the message with the node id. If it had already transmitted a HELLO message of the node id, then it drops the message.
- When the cluster head receives HELLO message it sends an ACK message in which it copies the node id and counter value.

- Intermediate sensor nodes just retransmit the ACK messages.
- A sensor node whose node id matches with that in ACK message, notes down the counter value as its hop count from the cluster head and does not propagate the ACK any further.
- A sensor node may receive ACK message more than once with different counter values. It sets its hop count to the minimum of all the received counter values.
- A timer is set once a node sends out its HELLO message. If the node does not receive its ACK before the timer times out, then it would retransmit its HELLO message.
- This process continues till every node knows its hop count from the cluster head and there are no HELLO messages floating in the network.

We assume that in the control plane the sensors run a CSMA/CA protocol to get the hop count information. Figure 2 is a logical representation of the deployment. The concentric circles encompass sensor nodes at different hop counts. For example, sensors in circle $C1$, concentric ring $C2$ and $C3$ are one, two and three hops away from the cluster heads respectively.

C. Time Slot Assignment

Time slots are assigned to the sensor nodes such that they can be reused by sensors which do not interfere with each other. To understand the slot assignment, we use the following notation (see Figure 2).

Let us denote C_i ($i \geq 1$) as the concentric area between circle with radius $(i-1) \cdot r$ and circle with radius $i \cdot r$, where r is the transmission range of the sensors. Thus C_1 is area in the circle with radius r, C_2 is concentric area between circle with radius r and $2r$, and so on. We refer to a concentric area as a *ring*. Although C_1 is not a concentric area, we will still refer to it as a *ring*.

Let us denote S_i ($i \geq 1$) as the angular area between $(i-1) \cdot \theta$ and $i \cdot \theta$ with respect to geographical north (going clockwise). Thus S_1 is the angular area between 0 and θ, S_2 is the angular area between θ and 2θ. Later, we will explain how to get the value of θ. We refer to an angular area as a *sector*.

The intersection of a concentric area C_i with a angular area S_j is denoted as $B(C_i, S_j)$ and is referred to as *block*. Different Rings, sectors and the block $B(C_3, S_1)$ are depicted in Figure 2.

The slot assignment is done depending on the distance (or number of hops) of the node from the cluster head and its position in the sector. Slot assignment computation is performed by the cluster head and is communicated to the sensor nodes. Time slots are assigned from a *superframe*, which is the largest unit of time slots. Slot assignment happens in three levels of hierarchy. In the first level, the superframe is divided among the rings C_i. In the second level, in a given ring C_i, time slots are assigned to different sectors S_j. In the final level, a time slot is allotted to an individual sensor node belonging to a particular block.

1) Slot Assignment in the Rings: We consider a superframe of length T time units which repeats for ever in the system for slot allocation. Since the transmission and the interference range of the nodes are assumed to be same, the nodes in the ring C_i and C_{i+3} ($i > 0$) can communicate simultaneously (note that the consecutive rings are distance r apart from each other). Thus they can be assigned the same TDMA slots. But the nodes in the ring C_i, C_{i+1} and C_{i+2} must have different TDMA slots, since they would be in the interference range of each other. Hence, we divide the superframe into three equal

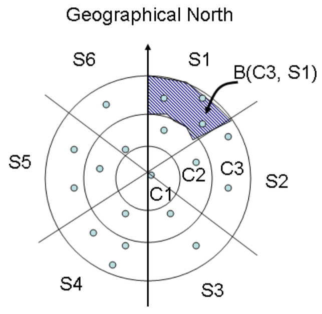

Fig. 2. Illustration of Different Terms Used

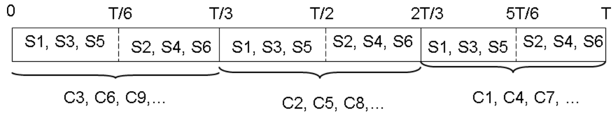

Fig. 3. Allocation of slots from SuperFrame (at block level)

parts and assign them to C_i, C_{i+1} and C_{i+2}. For example, ring C_1 is assigned slot $(\frac{2T}{3}, T]$, ring C_2 takes slot $(\frac{T}{3}, \frac{2T}{3}]$, ring C_3 take slot $[0, \frac{T}{3})$. The same assignment pattern is repeated in the farther rings, e.g., ring C_4 is assigned slot $(\frac{2T}{3}, T]$ (the same slot as C_1) and so on. The slot assignment to the rings can be generalized as follows. $C_{3i+1}(i \geq 0)$ is assigned slots $(\frac{2T}{3}, T]$, $C_{3i+2}(i \geq 0)$ is assigned slots $(\frac{T}{3}, \frac{2T}{3}]$ and $C_{3i+3}(i \geq 0)$ is assigned slots $[0, \frac{T}{3})$. This pattern of reallocation of slots makes sure that simultaneous communication happens across rings without collision. Each of these $\frac{1}{3}$ part of the superframe is referred to as a *ring subframe*.

2) Slot Assignment in Sectors of a Ring: We have designed the RTMAC such that time slots within a ring can also be reused. Sensor nodes which are within a particular ring but belong to different block can reuse the time slot if they are not within interference range of each other. This will be dependent on the angle θ at which the sectors are demarcated. This provides another level of reuse of time slots and hence further reduces overall latency. We will find the value of θ such that we can give same slots to the sectors 1, 3, 5 ... and similarly same slot to sectors 2, 4, 6 .... In order to find the value of θ we take into account the transmission range r of the node for the boundary nodes as shown in (Figure 4).

We will treat rings C_1 and C_2 as special case. Now consider the ring C_3 (see Figure 4). We draw the transmission range of a sensor node (circles with radius r) along the ring as shown in Figure 4. The Figure shows the extreme situation when a sensor in block $B(C_3, S_3)$ (marked as A) can barely interfere with a sensor in block $B(C_3, S_5)$ (marked as B). In such a situation

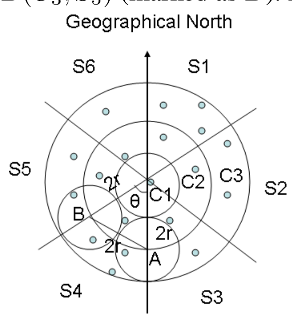

Fig. 4. Calculation of θ

it can easily be observed that θ is 60^0. Thus, the slots used in block $B(C_3, S_3)$ can be reused by sensors in block $B(C_3, S_5)$ and in $B(C_3, S_1)$. From our discussion on slot assignment of rings, we know that ring C_3 is assigned slot $[0, \frac{T}{3})$. Now, the nodes in the rings belong to two sets of non-interfering blocks. One set is $B(C_3, S_1), B(C_3, S_3), B(C_3, S_5)$ and the other set is $B(C_3, S_2), B(C_3, S_4), B(C_3, S_6)$. The first is assigned time slot $[0, \frac{T}{6})$ and the second set is assigned $[\frac{T}{6}, \frac{T}{3})$. Now we need to assign slots to individual sensor nodes in a block.

In order to assign slots to each node in a block, the cluster head need to know the number of nodes in a particular block. The cluster head sends a HELLO message to the nodes in a particular block. The message contains the ring ID and the sector ID. A HELLO message from the cluster head is only replied by the sensor nodes which belong to the block determined by the ring ID and sector ID. The reply contains the node ID. Once all the replies from a block are heard, the cluster head assigns time slot (at an individual sensor node level) from the block level time slot of length $\frac{T}{6}$. Since we assume that packet size in the cluster is constant, each sensor node is assigned a time slot which is long enough to transmit one packet. We denote node level time slot as t_{node}. If the cluster head hears n replies from a block, it assigns one time slot of size t_{node} to each of the n nodes beginning from the time slot of the block. For example, for block $B(C_3, S2)$ it will assign n node level slots starting from $T/6$. We assume that there is a maximum number of sensor nodes (M) allowed in a block such that $M \cdot t_{node} \leq \frac{T}{6}$. These time slots are conveyed by the cluster head to the sensor nodes in a *slot assignment* message. This message carries the node ID and the slot number.

For rings C_1 and C_2, no sectors will be considered for slot assignment. So the node level time slots will be assigned from the ring level time slots (e.g., $[\frac{2T}{3}, T)$ for C_1). Clearly, for C_1 and C_2, the total number of sensor nodes N_1 and N_2 should be such that $N_1 \cdot t_{node} \leq \frac{T}{3}$ and $N_2 \cdot t_{node} \leq \frac{T}{3}$. The node level time slots are then assigned starting from the boundary of the ring level time slots. The final slot allocation up to a block level in a superframe is shown in Figure 3.

VI. Data Plane of RTMAC

Since RTMAC is a TDMA based protocol, its operation in the data plane is quite simple. If a node has a packet to send, then it waits for its time slot to transmit the packet. But RTMAC allows nodes to go to sleep when they are neither sending nor receiving any packet. This causes the nodes to conserve energy. We explain the sleeping mechanism of RTMAC in the next section.

A. Energy Efficiency

RTMAC makes sure that when nodes are not transmitting or receiving, then they can go to sleep to save energy. For example, the ring subframe $[0, \frac{T}{3})$ is assigned to rings C_3, C_6 and so on. $[\frac{T}{3}, \frac{2T}{3})$ is assigned to C_2, C_5, and so on. $[\frac{2T}{3}, T)$ is assigned to C_1, C_4, and so on. The data path in this sensor network is from sensor nodes to the cluster head. So, when nodes in ring C_3 transmit to those in C_2, nodes in C_1 should not be transmitting or receiving. Because if nodes in C_1 transmit, then they will interfere with the nodes in C_2. When C_3 is transmitting to C_2, C_2 will be in receive mode and hence should not be transmitting anything to C_1. Hence, C_1 (which is the receiver ring of $C2$) should not be in receive mode during that time. Therefore, nodes in C_1 can sleep during the sub frame in which C_3 is transmitting. This means nodes which have slots assigned in ring subframe $[\frac{2T}{3}, T)$ (e.g., C_1) can sleep in sub frame $[0, \frac{T}{3})$ (assigned to C_3). Generalizing this rule, it is easy to verify that nodes assigned to particular ring subframe should sleep in the very next ring subframe and wake up at the end of the next sub frame. This sleep and wake up pattern enables RTMAC to save, on an average, $\frac{1}{3}$ energy as compared to a conventional TDMA system.

B. Delay Analysis

As mentioned earlier, we chose a TDMA based MAC because it can provide delay guarantee. We assume that the packet size in the network is same throughout the network. We also assume that events occur at a low rate. Hence, there will be no queueing delay of the packets in the network. We use the following notations for our delay analysis:

- T_r is the transmission time of a packet
- t_{node} is size of a node level time slot.
- H is the hop count of a sensor node (where the event was sensed) from the cluster head.
- N is the total number of nodes in the cluster.
- N_1 is the total number of nodes in the ring C_1.
- N_2 is the total number of nodes in the ring C_2.
- M is the maximum number of nodes allowed in a block.
- D is the worst case delay of an event, happening H hops away from the cluster head, to reach the the cluster head.

Consider the simple case, when the event is sensed by a sensor node inside ring C_1 ($H = 1$). In this case, the sensor node might just miss its time slot when the event occurred. Hence, it will have to wait for T time units to send the event to the cluster head. So the worst case delay in this case will be $T + T_r$. When the event is sensed inside ring C_2 ($H = 2$), in the worst case, sensor node in C_2 has to wait for $T + T_r$ to transmit. Then the sensor node in C_1 will have to wait for a maximum of $N_1 \cdot t_{node} + T_r$ to transmit the packet. Thus, the total worst case delay when $H = 2$ is $T + 2 \cdot T_r + N_1 \cdot t_{node}$. Now consider the general case when an event was sensed at a sensor node belonging to ring C_i ($i > 2$) at time $t = 0$. Similar to the previous case, the node has to wait for T unit of time to transmit the packet. Thus the packet will be out at $t = T + T_r$ unit of time. In the next ring C_{i-1}, the worst case will happen when the sender sensor node has a time slot at the end of the ring sub frame. In this case, the packet will experience a delay of $T/3 + T_r$ to reach the next ring C_{i-2}. This will be the delay for each of the rest $(H - 3)$ hops. In the penultimate ring C_2, the maximum delay will be $N_2 \cdot t_{node} + T_r$. In the last ring C_1, the maximum delay will be $N_1 \cdot t_{node} + T_r$. Hence, the worst case delay of an event to reach the cluster head is given by

$$D = \begin{cases} T + T_r + (H-3)(\frac{T}{3} + T_r) + \\ \quad N_2 \cdot t_{node} + T_r + N_1 \cdot t_{node} + T_r & for H \geq 3 \\ T + T_r + N_1 \cdot t_{node} + T_r & for H = 2 \\ T + T_r & for H = 1 \end{cases} \tag{1}$$

Simplifying the equations we get

$$D = \begin{cases} H \cdot T_r + H \cdot \frac{T}{3} + \\ \quad N_2 \cdot t_{node} + N_1 \cdot t_{node} & for H \geq 3 \\ T + 2T_r + N_1 \cdot t_{node} & for H = 2 \\ T + T_r & for H = 1 \end{cases} \tag{2}$$

If we limit the maximum number of nodes in a block to M and the maximum number of nodes in a cluster to N, then the value of superframe is given by $\frac{T}{6} = M \cdot T_r$ or $T = 6 \cdot M \cdot T_r$.

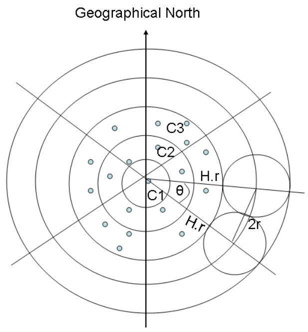

Fig. 5. Calculation of θ for Farther Rings

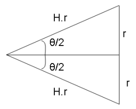

Fig. 6. Geometry in Calculating θ for Farther Rings

This is because each block is given a slot of $\frac{T}{6}$. But if a conventional TDMA system is implemented without any slot reuse, T is given by $T = N \cdot T_r$. Clearly, when $N \geq 6\ M$, the slot reuse due to rings and sectors will reduce latency.

The readers may have realized by now that it is quite easy to design a cluster for a given worst case delay. H is decided by the size of a cluster and the transmission range of a sensor node (r). M, N_1, N_2, and T should be chosen such that in addition to satisfying the delay requirement of (2), the following set of constraints should also be satisfied.

$$\begin{aligned} N_1 \cdot t_{node} &\leq \frac{T}{3} \\ N_2 \cdot t_{node} &\leq \frac{T}{3} \\ M \cdot t_{node} &\leq \frac{T}{6} \end{aligned} \quad (3)$$

C. Improving Delay

In Section V-C.2, we introduced angular sectors which are at an angle $\theta = 60^0$. If the angle θ is kept constant, then blocks belonging to rings, which are farther away from the cluster head, will have larger area. The maximum number of nodes in a block (M) is decided based on the largest block (in the outermost ring). Larger M means superframe T will be larger. Thus, a constant value of θ will increase the worst case delay of the system. Hence, as we go farther away from the cluster head, θ should decrease to a smaller value. Refer to Figure 5 to see how θ should be calculated for farther rings. Referring to Figure 6 we have

$$sin(\theta/2) = \frac{r}{H\ r} \quad (4)$$

Solving (4) for θ, we get

$$\theta = sin^{-1}[\frac{2 \cdot \sqrt{H^2 - 1}}{H^2}] \quad (5)$$

Note that for ring C_3, we get $\theta = 60^0$ by putting $H = 2$ in (5).

VII. Performance Evaluation

We have implemented RTMAC in ns2 [16]. Since we wanted to compare performance of RTMAC with S-MAC, we also obtained the S-MAC patch for ns2. Nodes were uniformly positioned around a cluster head. The area of deployment was 500×500 square meters. Each phenomenon or event is carried in a constant packet size of 48 bytes. Each sensor node had a transmission range of 100 meters. There were three rings in the experiment.

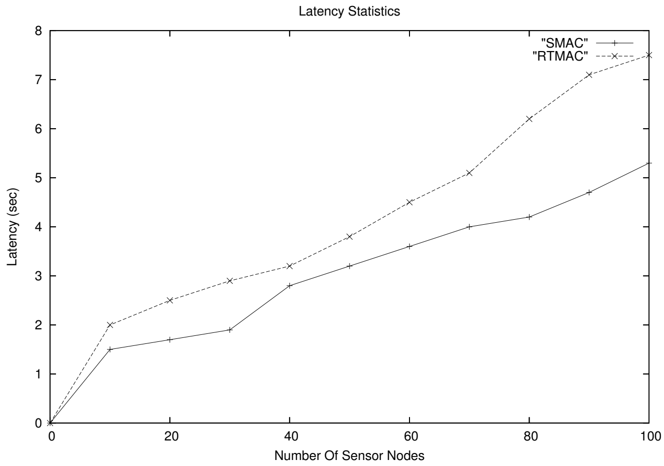

Fig. 7. Latency as a Function of Network Size

A. Latency Measurement

In this experiment, number of nodes in the network was varied from 10 to 100. Events (Phenomena) were generated at a constant rate of 3 broadcasts/sec. But the location of event is randomly chosen in the area. The latency of that event from the instance it was generated to the instant it was received by the cluster head was measured under RTMAC and S-MAC.

Figure 7 shows how average latency changes as the number of nodes in the network increase. As expected, RTMAC gives rise to higher latency than S-MAC. This is a well known drawback of TDMA protocol. However, the difference in latency between the two protocol is not much for small number of nodes. Although, S-MAC has lower average latency, it cannot guarantee that the events will meet their deadlines. But RTMAC, being a TDMA based protocol provides deadline guarantee to each event. We present this aspect of the protocol in Section VII-B. Hence, RTMAC is more suitable for a real time application which would need delay guarantee for every event.

B. Delay Guarantee

RTMAC is suitable for applications which requires that sensed events meet their deadlines. S-MAC, although an energy efficient MAC protocol, cannot guarantee that every packet (or event) meets its deadline.

To show that RTMAC can provide delay guarantee at an event or packet level, we ran our experiment by setting a deadline for the packets and noting the percentage of packet that missed their deadline when S-MAC is used. The deadline was normalized against the maximum delay of RTMAC protocol given by (2). This was repeated for different values of event (or phenomenon) generation rate. The simulation experiment was done using 30 nodes in the network with an area of $500 \times 500\ squaremeters$ area.

Figure 8 shows the result of this experiment. Since the deadline was normalized against maximum delay of a packet in RTMAC, all the packets in RTMAC meet their deadline. But when S-MAC is used, as event generation rate increases, more

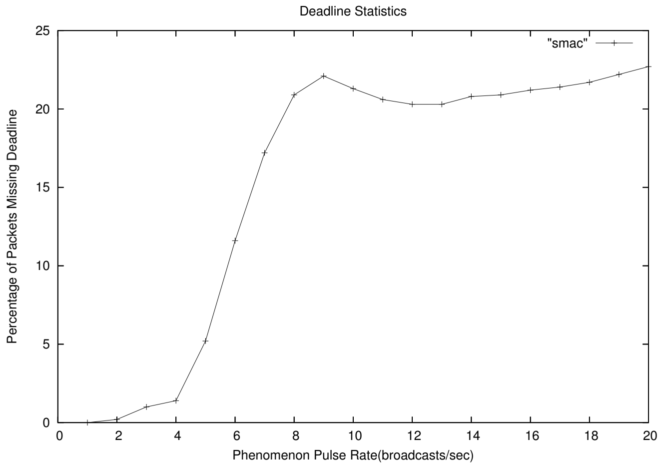

Fig. 8. Percentage of Packets Missing Deadline

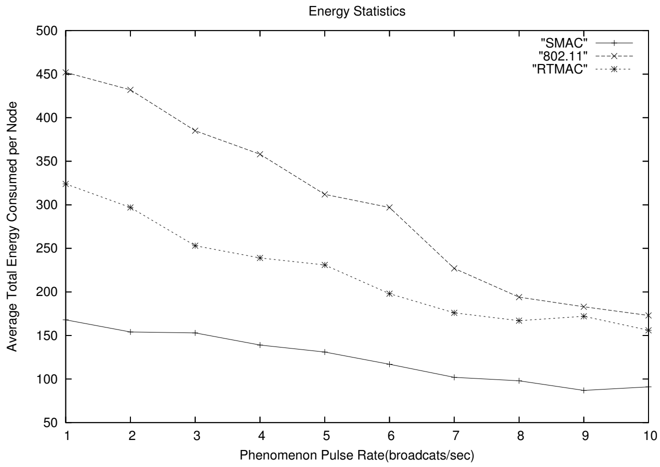

Fig. 9. Energy Consumption on Each Node

and more packets miss their deadline. The percentage packet drop for S-MAC was as high as about 20% at high event generation rate. Thus, S-MAC, although energy efficient, may not be suitable for real time application which require delay guarantee for each event.

C. Energy Consumption

Since energy is a very scarce resource in wireless sensor network, it was imperative that we analyzed the performance of RTMAC in terms of energy consumption. The sensor nodes were assumed to spend 1.0, 2.0, 0.001 and 4.0 units of energy for transmission, reception, sleeping and idle listening respectively. These parameters were chosen based on the data reported in [8]. All sensor nodes start with an initial energy of 1000 units. We compared RTMAC performance with IEEE 802.11 and S-MAC.

Figure 9 shows the graph of the measured average energy consumption per node as the phenomenon rate increases. During low phenomenon rate, 802.11 consumes far more energy than S-MAC since it performs idle listening very often. Performance of RTMAC protocol lies in between the two protocols in terms of energy efficiency. In RTMAC, the sensors sleep only for a third of the superframe which is not as efficient as the S-MAC which uses coordinated sleep and wakeup cycle. But RTMAC performs much better than 802.11 protocol when phenomenon rate is low. For high phenomenon rate the performance difference narrows. Since idle listening consumes more power than the other states, 802.11 and RTMAC perform poorly when large amount of time is spent in idle listening at low phenomenon rate (S-MAC reduces duration of idle listening by having periodic sleep and wake-up cycles). At high phenomenon rate, nodes are busy transmitting or receiving and spend less time in idle listening. Hence, performance of 802.11 and RTMAC comes closer to that of S-MAC.

VIII. Conclusion

In this paper, we have presented a new collision free TDMA based MAC protocol called RTMAC which uses channel reutilization techniques based on the topology of the network to reduce latency. Nodes go to sleep when they are not transmitting or receiving data. This sleep and wake up pattern saves energy for the sensor node. We have provided delay analysis of RTMAC to show that the worst case delay is bounded. Thus, RTMAC is suitable for real time applications like detection of radioactive radiation, earthquake which require that the sensed event (or packets) are delivered within a certain deadline. We have compared performance of RTMAC with S-MAC. Our performance analysis shows that although S-MAC performs better than RTMAC in terms of average latency and energy consumption, it fails to provide delay guarantee at the event level. Thus we submit that RTMAC should be preferred over S-MAC for real time applications having stringent delay guarantee requirement.

References

[1] I.F. Akyildiz and W.Su and Y. Sankarsubramaniam, "A Survey on Sensor Networks," in *IEEE Communications Magazine*, August 2004.

[2] S. Kulkarni and M. Arumuga, "TDMA Services for Sensor Networks," in *International Conference on Distributed Computing Systems Workshop*, 2004.

[3] K. Sohrabi and G. J. Pottie, "Performance of a novel self-organizationprotocol for wireless ad hoc sensor networks," in *Proc. IEEE 50th Vehicular Technology Conference, pp. 1222-1226*, 1999.

[4] "Wireless LAN Medium Access Control(MAC) and Physical Layer (PHY) Specifications." IEEE 802.11 Standard, 1997.

[5] A. Woo and D. Culler, "A Transmission Control Scheme for Medium Access in Sensor Networks," in *IEEE/ACM International Conference on Mobile Computing and Networking, pp. 221-235*, 2001.

[6] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC Protocols for Wireless Sensor networks: a Survey," *IEEE Communication Magazine*, vol. 44, no. 4, pp. 115–121, April 2006.

[7] W. Ye and J. Heideman, "Medium Access Control in Wireless Sensor Networks," in *USC/ISI Technical Report, ISI-TR-580*, 2003.

[8] W. Ye and J. Heidemann and D. Estrin, "Medium Access Control With Coordinated Adaptive Sleeping for Wireless Sensor Networks," *IEEE Transactions on Networks*, vol. 12, no. 3, pp. 493–506, 2004.

[9] A. El-Hoiydi, "Spatial TDMA and CSMA with Preamble Sampling for Low Power ad-hoc Networks," in *Seventh International Symposium on Computers and Communications*, July 2002.

[10] T.V Dam and K. Langendeon, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," in *The First ACM Conference on Embedded Networked Sensor Systems*, 2003.

[11] P. Lin and C. Qiao and X. Wang, "Medium Access Control with a Dynamic Duty Cycle for Sensor Networks," in *IEEE Wireless Communications and Networking Conference, Vol 3*, 2004.

[12] A. Rowe, R. Mangharam and R. Rajkumar, "RT-Link: A Time-Synchronized Link Protocol for Energy-Constrained Multihop Wireless Networks," in *Proceedings of IEEE SECON*, 2006.

[13] B. Krishnamachari and D. Estrin and S. Wicker, "The Impact of Data Aggregation in Wireless Sensor Networks," in *Proceedings of the 22nd International Conference on Distributed Computing Systems,Page 575 - 578*, 2003.

[14] A. Savvides and C.C Han and M.B. Srivastava, "Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors," in *MobiCom*, 2001.

[15] R. Nagpal and H. Shrobe and J. Bachrach, "Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network," in *2nd International Workshop on Information Processing in Sensor Networks(IPSN)*, April 2003.

[16] "NS2 simulator." `http://www.isi.edu/nsnam/ns/`.