
PO-JEN CHUANG AND CHIH-SHIN LIN
Department of Electrical Engineering
Tamkang University
Tamsui, 251 Taiwan
E-mail: pjchuang@ee.tku.edu.tw

This paper presents a new MAC protocol to achieve energy efficiency and low transmission latency for the static wireless sensor networks. The proposed new protocol is called A-MAC, an Alternative MAC, which assigns modified alternative wakeup schedules to different sensor nodes to minimize the probability of transmission collision. In A-MAC, after all sensor nodes are deployed, the base station will start an initial process to set a height for each node. When the height of each node (= the node’s hop counts to the BS) is decided, the active interval of nodes whose height difference = 1 will be set to be continuous to reduce transmission latency, whereas the active interval of nodes with the same height will stagger to avoid collision (i.e., to reduce the probability of simultaneous transmission). Operating by such alternative wakeup schedules, the new protocol is able to smooth data transmission and meanwhile conserve energy consumption for the energy-constrained wireless sensor networks.

Keywords: alternative wakeup schedules, collision avoidance, energy efficiency, latency reduction, medium access control (MAC) protocols, static wireless sensor networks

1. INTRODUCTION

A wireless sensor network (WSN) is usually composed of hundreds/thousands of sensor nodes and deployed in an unprotected environment to collect the needed information [1]. As it is difficult for the battery-powered sensor nodes to get recharged after distribution, energy efficiency becomes a basic and critical issue in the design and application of WSNs. This energy issue makes traditional medium access control (MAC) protocols unsuitable for WSNs – a number of new MAC protocols ([2-11]) able to meet WSNs’ urgent demand for energy efficiency are hence introduced. In practice, the limited energy resources of sensor nodes are usually wasted through overhearing, idle listening and collision.

• Overhearing: When a node sends data, all neighbor nodes within the transmission range (destination nodes or not) will receive the data – unnecessarily wasting energy.
• Idle listening: Each node sends or receives data only occasionally but needs to constantly “listen” to the channels for possible transmissions, consuming additional energy.
• **Collision:** When more than two nodes attempt to send data to the same destination at the same time, the destination node may fail to receive the data. When the source nodes are forced to resend the data, extra energy will be consumed.

Energy efficiency is important and so is **transmission efficiency** which can be attained through improved latency, throughput and load fairness.

• **Latency:** In particularly time-critical WSN applications, transmission latency should be shortened to the least so that a node can send the collected data to the base station (BS) as soon as possible to facilitate instant and appropriate actions.

• **Throughput:** Desirable throughput is especially important when a WSN needs to transmit all of the data collected in a certain period of time.

• **Fairness:** If routing channels are constantly occupied by a certain number of nodes, the other nodes will be forced to hold and wait. To attain desirable network performance, all nodes should be given fair chance to use transmission channels.

Based on the above observation, this paper presents a new MAC protocol, an Alternative MAC (A-MAC), to pursue power efficiency and low latency for static WSNs. The proposed A-MAC will assign modified alternative wakeup schedules to different nodes to minimize the chance of transmission collision. Under our protocol, right after all sensor nodes are deployed, the BS will start an initial process to set a height for each node (= a node’s hop counts to the BS). When the height for each node is decided, the active interval of nodes whose height difference = 1 will be set to be continuous to reduce transmission latency, whereas the active interval of nodes with the same height will stagger to avoid collision (i.e., to reduce the probability of simultaneous transmission). By such alternative wakeup schedules, our new protocol is able to smooth data transmission and meanwhile conserve energy consumption for the resource-constrained WSNs.

### 2. RELATED WORKS

**S-MAC** [2] puts sensor nodes into periodic sleep to reduce power consumption due to idle listening and to decrease the probability of collision. It also aims to solve the overhearing problem: When receiving a packet, a non-destination node will be forced into sleep. Putting a sensor node into sleep may solve the overhearing problem; it may also increase transmission latency (due to transmission interruption) and reduce throughput (because the interrupted data must wait until the node wakes up to resume transmission).

**D-MAC** [3] attempts to improve the transmission interruption problem in S-MAC by alternating the wakeup period of sensor nodes. Employing the data gathering tree (e.g. GIT [12]), D-MAC allows nodes on different tree levels to have different but continuous wakeup periods to solve the problem of transmission interruption.

**T-MAC** [4] is proposed to reduce energy consumption due to idle listening. It puts nodes into periodic sleep like S-MAC but with a different design – the dynamic duty cycle. In T-MAC, when a node is in the wakeup mode but sends or receives no data for a certain period of time, it will switch to the sleep mode to avoid wasting energy.
Z-MAC [5] combines the concept of CSMA and TDMA. The protocol allows each node to have its own time slot. Nodes in the owner time slot have the shortest carrier sense time—i.e., guaranteed with the highest priority to transmit data, while nodes in the non-owner time slot have longer carrier sense time—i.e., given with lower priority to transmit data. Z-MAC not only allows a node to transmit data at any time to reduce the delay time; it also assures that each node can receive data at any time (by letting nodes keep LPL—low power listening—in the sleep mode to sense if any data is received).

ED-TDMA [6] is an energy-efficient MAC protocol for event driven application. Its operation is divided into rounds, each with two phases—the set-up phase and the steady phase. The set-up phase executes clustering and time synchronization; the steady phase is divided into more variable-length frames. When a node intends to transmit data, it will send a request to the cluster head at the beginning of a frame and wait for the cluster head to allocate a channel. That is, ED-TDMA will assign time slots to nodes only when they need to transmit data—such a design will increase not only utilization but also overhead.

SESAM [7] uses the information of the application level to predict future transmission. It sets up a node’s wakeup schedule based on the data rate of application and is able to adjust a node’s duty cycle accordingly. SESAM thus allows nodes to work under a low duty cycle to save energy (only for low-bit-rate applications).

3. THE PROPOSED ALTERNATIVE-MAC PROTOCOL DESIGNS

As mentioned, the proposed A-MAC is an energy-efficient MAC protocol for static WSNs, designed mainly to minimize the probability of transmission collision (by assigning modified alternative wakeup schedules to the sensor nodes). It can meanwhile reduce idle listening and transmission latency in a wireless environment.

- **To reduce idle listening:** As nodes are not constantly sending/receiving data, it is unnecessary for them to listen to routing channels at all times. To reduce idle listening, A-MAC puts sensor nodes into sleep periodically (as most established protocols) but adopts the dynamic (instead of the fixed) duty cycle to reserve the energy of the nodes.

- **To reduce latency:** Putting sensor nodes into the periodical sleep mode may interrupt data forwarding and thus increase transmission latency. For improvement, A-MAC employs the alternative wakeup schedule in D-MAC to let sensor nodes on different levels wake up sequentially (Fig. 1). Such a design helps avoid transmission latency—because when data are being transmitted to the BS, packet routing on all levels will not be interrupted by a sleeping node (to be demonstrated by the transmission process in Fig. 2).

![Fig. 1. The alternative wakeup schedule (Rx/Tx indicates the sensor node is ready to receive/transmit data, SL = the sensor node is put into the sleep mode).](image1)

![Fig. 2. The transmission process.](image2)
To avoid collision: A-MAC adopts the alternative wakeup schedule to reduce not only transmission latency but also the probability of collision. Collision happens when several neighbor nodes attempt to transmit data simultaneously. To reduce the probability of collision then means to keep neighbor nodes from waking up at the same time. Besides reducing the probability of transmission collision, interchanging the wakeup schedules of neighbor nodes can also reduce energy consumption due to overhearing. To alternate the wakeup schedule of neighbor nodes, our algorithm (to appear in later sections) will assign a group ID – 0 or 1 – to each node of the same height and the assignment would vary as much as possible to pursue the best of collision avoidance. Fig. 3 shows our modified alternative wakeup schedule for nodes with the same height but different group IDs (0 or 1). Based on such a schedule, transmission collision can be avoided because the wakeup time of nodes at the same height will stagger under different group IDs.

The above discussion shows that A-MAC needs a mechanism to attain time synchronization. As existing synchronizing mechanisms can synchronize nodes within the accuracy of less than 10 microseconds ([13-15]), we believe our A-MAC – whose slot length is far greater than 10 microseconds – can achieve perfect time synchronization.

4. OUR ALGORITHM FOR CALCULATING THE ALTERNATIVE WAKEUP SCHEDULE

In A-MAC, each node is first assigned a height $h$ (the tree level) and a group ID $gid$ – 0 or 1 – decided by the coordination between neighbor nodes to interchange their wakeup time. Each node will also record an update time $t$ to indicate its most recent update time. After distribution, sensor nodes first go through the initial process to initialize $h$ and $gid$ and to calculate the wakeup schedule. (For easier reference, the parameters of A-MAC – with definitions – are listed in Table 1.)

The focal point will be: How to assign different group IDs to neighbor nodes of the same height? As stated, our original algorithm includes 3 steps: (1) Initializing the height, (2) initializing the group ID (0 or 1) and (3) calculating the wakeup schedule – of each node. It faces a problem: A number of nodes will find no downlink nodes to transmit packets back to the BS – because the algorithm divides nodes of the same height into two groups according to the assigned group IDs and will thus make nodes of different groups unable to communicate with each other. To solve the problem, i.e., to locate downlink nodes for each node in the network, we try to initialize group IDs by different conditions to achieve the desired uniform ID distribution and finally come up with a new algorithm. The new algorithm neatly complies with our need: To distribute the group IDs of nodes uniformly and to provide each node with downlink nodes for packet transmission.
Table 1. The parameter list of A-MAC.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>The latest parameter updating time of a node.</td>
</tr>
<tr>
<td>gid</td>
<td>The group ID of a node.</td>
</tr>
<tr>
<td>h</td>
<td>The height of a node.</td>
</tr>
<tr>
<td>t_{update}</td>
<td>The update time recorded in an update packet = the latest updating time of the node which broadcasts the update packet.</td>
</tr>
<tr>
<td>gid_{update}</td>
<td>The group ID recorded in the update packet = the group ID of the node which broadcasts the update packet.</td>
</tr>
<tr>
<td>h_{update}</td>
<td>The height recorded in the update packet = the height of the node which broadcasts the update packet.</td>
</tr>
<tr>
<td>m</td>
<td>The number of time slots.</td>
</tr>
<tr>
<td>μ</td>
<td>The slot length.</td>
</tr>
<tr>
<td>τ(m, h, gid)</td>
<td>The function used to calculate delay time of each node. The delay time of each node will be set according to m, h, and gid, so this function has three parameters.</td>
</tr>
</tbody>
</table>

The New Algorithm

The new algorithm has a major difference. It first initializes the group ID gid and then the height h = i.e., before initializing h, it first divides the nodes of the entire network into two groups by the assigned group IDs. Details are as follows.

To initialize the group ID (gid)

Before node distribution, h and gid for each node is set as infinite and t is set = 0. After node distribution, the BS will broadcast an update packet whose group ID (gid_{update}) = 0 and whose update time (t_{update}) = the time to send the packet.

Fig. 4. The algorithm for updating group IDs of sensor nodes.

Algorithm 1:

1. if (Group ID of the sensor node is not yet changed at this time of update) 
2. 
3.  Change the Group ID of the node to let the node has a different Group ID from the update packet sender’s ID. 
4.  Broadcast the changed Group ID to update the Group IDs of neighbor nodes. 
5. 
6. 
7. 
8. 
9. if (Group ID of the sensor node is already changed at this time of update) 
10.  
11.  Accumulate the number of the Group ID recorded in the update packet. 
12.  Choose a less employed Group ID to update the Group ID of the sensor node. 
13.  
14. 

Algorithm 2:

1. if (receive the initial packet from the node has the lower Height) 
2. 
3.  Change the Height of received node to let received node becomes the next level of the update packet sender. 
4.  Broadcast the changed Height to update the Heights of neighbor nodes. 
5. 

Algorithm 3:

1. if (the node is in the initial state) 
2. 
3.  if (receive Group ID initializing packet) 
4.  
5.  Update Group ID gid according to the information recorded in received packet. 
6.  
7.  if (receive Height initializing packet from the same group) 
8.  
9.  Update Height h according to the information recorded in the received packet. 
10.  
11.  if (receive packet asking the node to calculate wakeup schedule) 
12.  
13.  Calculate the number of slots during a cycle m according to duty cycle. 
14.  
15.  Calculate the delay time according to the parameters m, h, and gid. 
16.  
17.  Wait until the cycle start time, and then start to countdown the delay time. 
18.  
19.  Change the node to the normal state. 
20.  
21. 
22. 
23. 
24. 
25. if (the node is in the normal state) 
26. 
27.  if (receive the rescheduled packet) 
28.  
29.  Set the Group ID and Height of node to the initial value. 
30.  
31.  Change the node state to the initial state. 
32.  

Fig. 5. The algorithm for updating heights of sensor nodes.

Fig. 6. The working algorithm for sensor nodes.
Receiving such an update packet, a sensor node starts to initialize its group ID \((gid)\) according to Algorithm 1 in Fig. 4. The Algorithm shows that when receiving an update packet, a node faces two situations: not yet updating its \(gid\) \((t < t_{\text{update}})\) or already updating its \(gid\) \((t = t_{\text{update}})\) by the moment.

\(t < t_{\text{update}}\): indicating the node is not yet updated and hence needs to update its \(gid\) into \((gid_{\text{update}} + 1) \mod 2\) (lines 3-4 of Algorithm 1). After updating its \(gid\), the node will set its update time \(t\) to \(t_{\text{update}}\) (to show its present status) and then broadcast the newly updated \(gid\) \((gid_{\text{update}})\) and the most recent update time (lines 5-6).

\(t = t_{\text{update}}\): indicating the node is already updated. To prevent neighbor nodes from attaining the same group ID, the node needs to record \(gid_{\text{update}}\) and follows the recorded \(gid_{\text{update}}\) to choose a less employed group ID (lines 11-13).

Under such a limited ID choice (0 or 1), Algorithm 1 may not achieve the optimized distribution of group IDs (i.e., assign neighbor nodes with totally different group IDs) but is able to turn out very favorable result. The favorable group ID distribution produced by Algorithm 1 has indeed significantly reduced transmission collision and conserved energy. We thus consider it impractical to further pursue the little gain in ID distribution at the cost of increasing energy consumption (to be proved later by our simulation results).

**To initialize the height \((h)\)**

When the group ID initialization for all neighbor nodes of the BS is finished, the BS once again broadcasts an update packet with height \((h_{\text{update}}) = 0\). A node receiving the update packet will go on to update its height \(h\) by Algorithm 2 (in Fig. 5) which divides the update packets according to the heights of their sending nodes:

\(h - h_{\text{update}} \leq 1\): The node receives an update packet from a node with bigger or the same height – ignore this update packet and do nothing.

\(h - h_{\text{update}} > 1\): The node receives an update packet from a node with lower height – update its own height \(h\) into \(h_{\text{update}} + 1\) and broadcast the updated data (lines 3-5).

**To calculate the schedule**

When the height initialization for all nodes neighboring the BS is complete, the BS will again broadcast a packet asking sensor nodes to calculate their wakeup schedules (lines 15-22, Algorithm 3, Fig. 6). After receiving the asking packet, nodes move on to calculate the wakeup time according to their heights and group IDs. A node will first divide the cycle length into \(m\) parts – each part is taken as a time slot. As the schedule in Fig. 7 shows, the node wakes up only at two time slots and is in the sleep mode at the other time slots. \(m\) can be calculated by the duty cycle:

\[
\text{Duty Cycle} = \frac{\text{ActiveInterval}}{\text{CycleLength}} \times 100\% = \frac{2\mu}{m\mu} \times 100\% = \frac{2}{m} \times 100\%.
\]

\(\mu\): the slot length

The cycle length = \(m\mu (m > 5)\)

The active interval = \(2\mu\) (RX and TX)

The sleep interval = \((m - 2)\mu\)
Calculating the time for each node to start working: Assume each node has different delay time \( \tau \). In calculating the delay time, we must take the height and group ID of a node into consideration to reduce both the transmission latency and probability of transmission collision. Thus, we can set the formula for calculating the delay time as follows.

\[
\tau(m, h, gid) = [m - (h \mod m)] + [3 \times gid]
\]

Duty cycle = \( \frac{\text{active time}}{\text{cycle time}} \times 100\% = \frac{2\mu}{m\mu} \times 100\% = \frac{2}{m} \times 100\% = 10\% \)

\[
\Rightarrow \frac{200\%}{m} = 10 \Rightarrow m = 20
\]

After \( m \) is attained, a node can calculate the delay time by its \( h \) and \( gid \), and start its wakeup schedule. The operating steps of sensor nodes are given in Fig. 8. The gray zone indicates the delay time during which the nodes are kept under the sleep mode; after the delay time, each node wakes up periodically according to the wakeup schedule. As observed, nodes with \( gid = 1 \) can construct a route from a higher level to a lower level, and so can nodes with \( gid = 0 \). Transmission latency is thus reduced. Meanwhile, as nodes with the same height but different group IDs will not translate into the wakeup mode at the same time, the probability of collision can be substantially decreased.

If a node cannot complete initialization in time due to unexpected factors (such as collision, packet loss, etc.), it will stand silent by the low duty cycle until next initialization to avoid influencing the normal nodes. As the amount of non-initialized nodes is usually very small in practice, it will not affect the coverage rate of a sensor network.

If a sensor network is to work normally, argument \( m \) must be set bigger than 5 to prevent possible collision and interference. If \( m \leq 5 \), packet transmissions for nodes whose height difference = 1 may interfere with each other. For instance, in Fig. 9 when a node with \( h = 6, gid = 0 \) transmits packets to a node with \( h = 5, gid = 0 \), we can see that another node with \( h = 4, gid = 1 \) is also in the transmission mode. Collision and interference are likely to happen.
To reschedule

After a static sensor network performs for a period of time, the network topology may change due to such factors as the death of nodes (running out of battery power), the earthquake attack in the observed zone or the distribution of newly added nodes. Sensor nodes affected by these factors must be rescheduled to maintain good function. The BS is in charge of such periodical rescheduling. The rescheduling, whose cycles vary with different applications, runs as follows. The BS first broadcasts a rescheduling packet to each node which then resets its $h$ and $gid$ to the initial state and broadcasts the rescheduling packet out (lines 25-32, Fig. 6). After all nodes receive the packet and reset their $h$ and $gid$ to the initial state, the whole network gets re-initialized. To complete rescheduling, simply repeat the above steps (initializing $gid$ and $h$ of nodes) and calculate the schedule.

Note that A-MAC is proposed to work under a static sensor network. When put to work under a mobile sensor network, the protocol needs to increase the frequency of rescheduling – based on the speed of sensor nodes – to guarantee favorable transmission. Rescheduling will cause certain degrees of latency because the BS needs to flood 4 times to indicate the reschedule, the initial $gid$, the initial $h$ and the calculation of the wakeup schedule. The time length of flooding is subject to network sizes: Assuming the network size is $n$ hops, one flooding will take $n\mu s$ second while 4 floodings will take $4n\mu s$. To reduce such rescheduling latency, our A-MAC conducts the 4 floodings by the pipeline concept which helps reduce the flooding interval into just 3 time slots to avoid the interference between different floodings. Rescheduling latency for A-MAC is hence reduced to $(n + 9)\mu s$ second.

5. PERFORMANCE EVALUATION

Experimental evaluation is conducted to compare the performance of different MAC protocols, including S-MAC, T-MAC, D-MAC, Z-MAC and our A-MAC. Among these protocols, S-MAC and T-MAC use the same approach as A-MAC to save energy consumption due to idle listening, while D-MAC, like A-MAC, attempts to reduce transmission latency by assigning different delay time to the nodes. Z-MAC, on the other side, offers to enhance transmission throughput at the cost of acceptable energy consumption – performance comparison between Z-MAC and A-MAC indeed gives a good chance to evaluate the tradeoff between the throughput gain and the needed energy consumption.
The above protocols are implemented in the ns-2 network simulator. Table 2 lists all the adopted parameters. The slot length of each protocol is 10ms and the duty cycle is 10%. Both the sending and receiving intervals for D-MAC and A-MAC are 10ms, the active interval for S-MAC, T-MAC, and Z-MAC is 20ms, and the sleep interval for all protocols is 180ms. The performance of these target protocols will be measured and compared in terms of their ability to reduce latency, to avoid collision and to reduce energy consumption. Latency refers to the end-to-end delay of a packet, collision is the total number of packets with delivery failure due to transmission collision, and energy consumption is the total energy cost for delivering packets. In our simulation, the collected result of latency = the average delay of each transmission, collision = the total number of collisions after certain packet transmissions, and energy consumption = the average energy consumption of each sensor node.

5.1 Average Transmission Latency under the Straight-Line Transmission

A simple straight-line transmission with 11 nodes is simulated to evaluate the transmission latency of different protocols. The distance between 2 nodes is 50 meters, and the first node will send out a packet per 0.5 sec to the last node. In the simple straight-line topology, sensor nodes can transmit data without any interference – transmission interruption happens only when the receiver is under the sleep mode (which is not likely for our A-MAC).

As the result in Fig. 10 shows, both S-MAC and T-MAC generate jumping latency patterns which we believe are caused by interrupted data transmission – because neither of them provides any interruption-handling mechanisms. The result meanwhile shows that both D-MAC and A-MAC are free of the transmission delay problem because their

---

Table 2. The simulation parameters

<table>
<thead>
<tr>
<th>Argument</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>The cycle length</td>
<td>200ms</td>
</tr>
<tr>
<td>The slot length</td>
<td>10ms</td>
</tr>
<tr>
<td>The active interval</td>
<td>20ms</td>
</tr>
<tr>
<td>The sleep interval</td>
<td>180ms</td>
</tr>
<tr>
<td>The source transmission rate</td>
<td>2 packets/second</td>
</tr>
<tr>
<td>The packet size</td>
<td>1Mb</td>
</tr>
</tbody>
</table>

---

Fig. 10. Average packet transmission latency.  
Fig. 11. Average energy consumption.
alternative wakeup schedules are able to assure uninterrupted data transmission. Z-MAC is also free of this problem as it allows nodes to receive and transmit data at any time by LPL, TDMA and CSMA.

5.2 Average Energy Consumption under the Straight-Line Transmission

Fig. 11 shows the average energy consumption of each packet transmission vs. the chain length. The result exhibits that Full_Active takes the least time but the most energy consumption to complete a packet transmission – because it keeps all nodes in the wakeup mode throughout the entire transmission period. T-MAC, by contrast, consumes the least energy as it applies the dynamic duty cycle to reduce the active time of nodes. Among the other protocols, transmission interruption has increased the transmission time of S-MAC, and the ability to solve the interruption problem has reduced the transmission time and energy consumption of A-MAC and D-MAC. As for Z-MAC which allows nodes to keep LPL in the sleep mode, the required energy consumption is higher than that of our A-MAC. The result also reveals quite close performance for A-MAC and D-MAC. The reason for A-MAC which adopts the energy-efficient dynamic wakeup schedule not to outperform D-MAC in conserving energy is simple: All nodes in the straight-line topology need to transmit – no node can switch to the sleep mode early.

5.3 Numbers of Collisions under the Random Distribution Topology

As Fig. 12 shows, 100 sensor nodes are randomly distributed over a 100m × 500m area, with the center node – the triangle – acting as the BS of the topology. To evaluate the number of transmission collisions, this simulation is conducted under such a topology with different numbers of source nodes. (Note that increasing source nodes will increase both the number of ongoing packets and the probability of transmission collision.) As the result in Fig. 13 shows, S-MAC and T-MAC generate the most collisions. This is understandable as both protocols yield longer transmission latency which will increase the amount of data transmission when more source nodes simultaneously send out packets.

![Fig. 12. The random distribution topology.](image)

![Fig. 13. Numbers of collisions.](image)

![Fig. 14. Average transmission latency of each packet under the random distribution topology with low traffic.](image)
By contrast, A-MAC outperforms even D-MAC by producing the least collisions – because it is able to alternate the wakeup schedule of neighbor nodes by assigning different group IDs. Note that Z-MAC is excluded from the collision evaluation here because it will always make transmission attempts to increase the throughput whenever nodes have data to transmit – disregarding the rising probability of collisions.

5.4 Average Transmission Latency under the Random Distribution Topology with Low Traffic

This simulation randomly chooses 5 sensor nodes (the dark dots) at the outskirt of the topology (in Fig. 12) as the sources to transmit data to the BS by fixed rates. The average transmission latency of each packet for the protocols is plotted in Fig. 14. From Figs. 7 and 14, we can see that S-MAC and T-MAC both depict bigger latency under the random distribution topology than under the straight-line transmission. This is because the random distribution topology allows more than one source to transmit data at a time, causing channel competition.

The performance of D-MAC and A-MAC, on the other hand, remains relatively stable because both protocols can transmit data to the BS without interruption – even when the nodes contend for routing channels. Z-MAC achieves lower delay as Full_Active; the slight delay increase in transmissions around the BS is caused by channel competition and collision.

5.5 Average Energy Consumption under the Random Distribution Topology with Low Traffic

Fig. 15 gives the average energy consumption of each node vs. the operation time under the random distribution topology with low traffic. As can be observed, Full_Active which constantly keeps nodes in the waking mode consumes energy at the fastest rate. S-MAC and T-MAC consume more energy than D-MAC due to interrupted data transmission which increases the probability of transmission failure as well as re-transmission. Although T-MAC also adopts the dynamic wakeup schedule, interrupted data transmission has cost it more energy consumption than our A-MAC. Without the dynamic wakeup schedule, D-MAC consumes nearly a double amount of energy than A-MAC. In total, our A-MAC saves respectively 97.2%, 77.3%, 65.2%, 51.1% and 45.3% energy consumption over Full_Active, S-MAC, T-MAC, Z-MAC and D-MAC.
5.6 Average Transmission Latency under the Random Distribution Topology with High Traffic

To generate high traffic in the topology, we randomly choose 15 sensor nodes at the outskirt of the topology as the sources to transmit data to the BS by fixed rates. Fig. 16 shows the average transmission latency of each packet under high network loads for the protocols. The result reveals almost the same trend as what is collected under low network loads (Fig. 14) – except that the latency of all protocols increases more quickly with longer transmission distances (i.e., closer to the BS). A-MAC displays the least latency increase thanks again to its alternative wakeup schedule which allows nodes to transmit data at different time schedules. Such an alternative schedule is especially effective in facilitating the high-traffic network transmission.

5.7 Average Energy Consumption under the Random Distribution Topology with High Traffic

Fig. 17 displays the average energy consumption for each protocol under the random distribution topology with high traffic. The result reveals a similar trend as what is obtained under the same topology but low traffic in Fig. 15. A-MAC, as expected, stands as the most energy-efficient protocol because its dynamic wakeup schedule helps nodes reduce the probability to remain in the wakeup mode. In total, A-MAC saves respectively 96.7%, 79.2%, 73.3%, 45.3% and 39.3% energy consumption over Full_Active, S-MAC, T-MAC, Z-MAC and D-MAC.
5.8 A-MAC vs. D-MAC

Both protocols employ the alternative wakeup schedules to reduce latency and yield quite similar performance—but with some differences. A-MAC performs better in reducing idle listening because it adopts the dynamic duty cycle instead of the fixed duty cycle of D-MAC. It also outperforms D-MAC in reducing collision by the improved alternative wakeup schedule which alternates the wakeup time of neighbor nodes to avoid collision.

5.9 Pursuing the Optimal Efficiency of A-MAC

The proposed A-MAC protocol, as mentioned, divides nodes into two groups—0 and 1. To evaluate how different distributions of the two groups will affect the efficiency of the protocol, we try to pursue the optimized distribution of the two groups on known topologies and allow A-MAC to perform on both normal and optimal distributions to see the difference. Fig. 18 shows the average transmission latency of each packet under the random distribution topology with heavy traffic. A-MAC and A-MAC_optimum indicate the protocol works respectively on the normal and optimal distributions. As the result exhibits, when the chain length grows, A-MAC_optimum can reduce more transmission latency than A-MAC—but the difference is not extensive. Fig. 19 records the average energy consumption of each node under the same condition for A-MAC and A-MAC_optimum. The results are almost the same, i.e., nodes consume almost the same amount of energy under either the normal or optimal distribution. The above results pinpoint out a fact: A-MAC performs nearly as well as A-MAC_optimum. Based on this fact, we thus consider it impractical to pursue the potential small performance gain at substantial cost, i.e., to consume more energy to optimize the distribution of the two groups.

5.10 Other Discussions

Protocols, such as A-MAC, S-MAC, D-MAC, T-MAC and Z-MAC, are all developed based on CSMA. They allow sensor nodes to contend channels with the same probability to avoid the fairness problem. As specified previously, Z-MAC gives all sensor nodes the chance to transmit data at any time—to achieve low latency, high throughput and high channel utilization—at the expense of increased energy consumption. A-MAC, D-MAC, S-MAC and T-MAC, on the other hand, let nodes wake up according to special schedules, i.e., nodes will be allowed to transmit data only at their specified active time. Such an arrangement may have drawbacks—increased latency, reduced throughput and lowered channel utilization—but is superior in conserving energy (in contrast to Z-MAC). Besides conserving energy, A-MAC and D-MAC are able to reduce transmission latency by using special wakeup schedules to solve the transmission interruption problem. The proposed A-MAC embraces even one more advantage that further distinguishes itself from the other protocols: Dividing sensor nodes into different groups to effectuate channel utilization and as a result to increase throughput and reduce transmission latency.

6. CONCLUSION

As the nodes in a WSN are battery-powered and are difficult to get recharged after
distribution, energy efficiency becomes a critical issue. Based on the special features of a static WSN, this paper presents A-MAC, a new and energy-efficient MAC protocol, to conserve the limited energy of such a network and also to reduce transmission latency and collision. When a sensor network is deployed, the proposed A-MAC starts an initial process to assign a group ID (0 or 1) and a height (hop counts to the BS) to each node. With such a specific design and function, A-MAC is able to arrange a favorable alternative wakeup schedule for all nodes in the network: The active interval of nodes whose height difference = 1 is set as continuous to reduce transmission latency, while the active interval of nodes with the same height will stagger to reduce the probability of simultaneous transmission, i.e., collisions. Operating based on such a dynamic job schedule which facilitates data transmission by reduced latency and collisions, the proposed A-MAC outperforms existing S-MAC, T-MAC, D-MAC and Z-MAC in most situations to prove itself an energy-efficient protocol for the wireless environment.

REFERENCES


Po-Jen Chuang received the B.S. degree from National Chiao Tung University, Taiwan, R.O.C., in 1978, the M.S. degree in Computer Science from the University of Missouri at Columbia, U.S.A., in 1988, and the Ph.D. degree in Computer Science from the Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette, U.S.A. (now the University of Louisiana at Lafayette), in 1992. Since 1992, he has been with the Department of Electrical Engineering, Tamkang University, Taiwan, where he is currently a Professor. He was the department chairman from 1996 to 2000. His main areas of interest include parallel and distributed processing, fault-tolerant computing, computer architecture, mobile computing and network security.

Chih-Shin Lin received his B.S. and M.S. degrees in Electrical Engineering in 2005 and 2007 from Tamkang University, Taiwan, where he is currently pursuing the Ph.D. degree. His research interests include parallel and distributed processing, network security and mobile computing.