Research Article

pQueue-MAC: An Energy Efficient Hybrid MAC Protocol for Event-Driven Sensor Networks

Liantao Wu, Shuguo Zhuo, Zhibo Wang, and Zhi Wang

State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China

Correspondence should be addressed to Zhi Wang; wangzhizju@gmail.com

Received 11 March 2014; Accepted 22 July 2014

Academic Editor: Shaojie Tang

Copyright © 2015 Liantao Wu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Although a lot of MAC protocols have been proposed to deal with burst traffic in wireless sensor networks, most of them do not optimize the energy efficiency of low and nonperiodic traffic, which actually wastes a lot of energy. To solve the problem, we propose a novel MAC protocol, called pQueue-MAC, to achieve high energy efficiency when no event is detected and still guarantee immediate yet energy efficient data transmission when events are detected. pQueue-MAC uses preamble sampling to check potential communications in low traffic condition. Specifically, through short channel listening, nodes can decide whether there is data destined for them, based on which nodes choose to sleep for energy conservation or keep active for data exchange. Moreover, adaptive channel listening scheme is adopted for dynamic traffic load. When load increases, based on the load information collected from son nodes by piggybacking, cluster heads adaptively allocate TDMAslots and embed the new schedule information into the beacon frame broadcasted to son nodes. We have implemented pQueue-MAC on STM32W108 chips that offer IEEE 802.1.5.4 standard communication and conducted extensive experiments to evaluate its performance. Experimental results show that pQueue-MAC outperforms other typical MAC protocols in hybrid traffic load situation, since it achieves higher energy efficiency and lower transmission latency.

1. Introduction

Wireless sensor networks (WSNs) are envisioned as consisting of a number of sensor nodes that are deployed over a region of interest to monitor a physical phenomenon. Cooperative effort of sensor nodes and other merits ensure a wide range of applications for WSNs, such as building automation, environmental monitoring, target tracking, and industrial process control. As one kind of the most primary applications, event-driven applications feature in dynamic traffic load which is composed of low, nonperiodic traffic load for a long time and burst traffic load for very short time. With regard to intrusion detection, when there is no target in monitoring area, sensor node has only a few data to send for system maintenance; thus the network traffic is quite low; once a target appears, many sensor nodes detecting the target generate lots of data and report them to the base station; therefore, the network load increases significantly during a short period. From this example, we can learn that event-driven sensor network applications require communication protocols to have the ability that not only keeps sensor nodes operating in low power mode to conserve energy when no event happens but also deals with the burst data conveniently to guarantee fast data transmission.

It is well known that sensor networks are power limited as sensor nodes are powered by batteries and it is usually impossible to replace or recharge sensor nodes. Considering radio communication is the most energy consuming part for sensor networks, which consists of four states as transmitting, receiving, listening, and sleeping. Idle listening for the former three states is the main source of energy waste. To reduce the energy consumption for idle listening, duty cycle scheme, alternating sensor nodes between listening and sleeping periodically, is widely used in MAC protocols.

Until now, a lot of duty cycle MAC protocols have been proposed. As one of the earliest MAC protocols adopting duty cycle, S-MAC [1] adopts fixed duty cycle scheme, using CSMA/CA in DATA period to send or receive data packets. With fixed duty cycle scheme, it is only suitable for low load or fixed load occasions. Later, [2–4] are proposed to enhance
the ability to adapt to the varying traffic load. However, they can only deal with the collision due to varying traffic load in a certain extent. When the load in network bursts, they cannot provide sufficient bandwidth for data transmission, leading to data collisions, thus resulting in energy waste and big delay. Our former work Queue-MAC [5] and some other solutions [6–8] can deal with burst traffic, but they do not optimize the energy usage for low traffic. It is necessary to improve the energy efficiency for event-driven sensor networks.

In this paper, we propose an adaptive duty cycle MAC protocol, called pQueue-MAC, which achieves high energy efficiency and in the meantime guarantees immediate data transmission. When no event happens, pQueue-MAC uses the synchronization based variable preamble (SVP) sampling to handle the low traffic situation. When traffic load increases, by inheriting the merits for dealing with burst traffic of Queue-MAC, the senders’ queue length will be used to inform the receiver for dynamically allocating time slots to the senders (TDMA) for packet transmission. The SVP sampling achieves high energy efficiency through short channel listening period and the dynamically allocating TDMA slots make full use of the communication bandwidth according to the traffic load, achieving low transmission delay. pQueue-MAC also has good scalability by using the synchronization based preamble sampling, which is convenient for new nodes to join the network.

The contributions of this work are summarized as follows.

(i) We present a SVP sampling based hybrid TDMA/CSMA MAC protocol for event-driven sensor networks. It achieves high energy efficiency in low traffic situation and still guarantees immediate yet energy efficient data transmission when events are detected.

(ii) We derive the lower bound of energy consumption when sensor networks have low traffic and figure out the optimal parameters for pQueue-MAC to achieve its best performance.

(iii) We have implemented pQueue-MAC on STM32W108 chips and conducted extensive experiments to evaluate its performance. The results show that our protocol outperforms other typical MAC protocols.

The rest of this paper is organized as follows. Section 2 discusses the related MAC protocols. We present the pQueue-MAC in Section 3 and derive the lower bound of energy consumption when no event happens. Experimental results are shown in Section 4. Finally, Section 5 concludes this paper.

2. Related Works

To conserve energy, most of the MAC protocols in wireless sensor networks adopt duty cycle technique.

Early MAC protocols in WSNs were designed for low traffic load, under the assumption that the network has long-term low rate periodic traffic. S-MAC adopts fixed duty cycle scheme. It schedules communication windows through synchronization and then uses CSMA/CA in DATA period to send or receive data packets. Thus when the traffic in network varies, it cannot provide appropriate communication bandwidth for data transmission. B-MAC [9] operates in the asynchronous duty cycle mode, using LPL (low power listening) technique to wake up receiver node. But long preamble stream introduces extra heavy overheads and severe wireless medium occupation, which makes B-MAC still need to be improved. X-MAC [10] improves the B-MAC by dividing preamble frame into small pieces and each piece is attached with destination address. In this way, X-MAC prevents long preamble from occupying the channel and reduces overhearing. SCP-MAC [11] is another LPL based protocol mainly designed for ultralow traffic load, using synchronization to reduce the length of preamble. LPL based MAC protocols are suitable for low traffic load applications, where occasional data transmissions do not incur too much overhead and channel contention is not severe. However, the above protocols are suitable for low traffic load situations, since all of the former MAC protocols adopt CSMA as their main scheme, which will lead to heavy collision when the traffic load increases sharply.

To adapt to the traffic load, many traffic adaptive MAC protocols have been proposed. Considering the asymmetric loads among sender and receiver, receiver-initiated probing based asynchronous MAC protocols are adopted to release more room for data transmission and reduce wireless medium occupation. In RI-MAC [12], upon receiving the beacon, sender forwards its data requirement packet to the receiver and will be acknowledged by the next beacon, which also initiates following transmissions if the sender still has pending packets. However, concurrent packet transmission to the same receiver will lead to packets collision, resulting in retransmissions and extra overhead under heavy load. Reference [7] adopts a receiver-initiated transmission protocol which is implemented for continuous multiple data relay. T-MAC [2] and MaxMAC [3] adapt to traffic load by tuning the wake-up periods when the estimated traffic rate exceeds predefined thresholds. Inaccurate traffic estimation makes them less efficient. Then some online identification based MAC protocols were proposed. Breath [13] and pTunes [14] build up the model of the network system and self-adapt MAC parameters for handling varying situations. However, finding an accurate model itself for the network is not realistic considering constrained energy and latency requirement. Strawman MAC [15] solves collision through sampling the received signal strength, judging the length of collision packet to decide the node with the longest collision packet to be the channel winner. Though Strawman MAC had made the improvement, collision packets introduce a considerable amount of overheads. A+MAC [16] and Z-MAC [4] adapt to traffic by adopting hybrid TDMA/LPL or hybrid TDMA/CSMA. In Z-MAC, slots can be stolen by nonowners when the owner has no packets to transmit (slot stealing). However, since Z-MAC follows a static TDMA scheme, only the transmissions of the slots owners can be guaranteed. IEEE802.15.4 standard [17] also uses hybrid CSMA/TDMA based on the superframe structure. But it only provides a limited number of slots in the CFP period, which cannot be extended to deal with burst traffic. Above several protocols, performance for handling burst traffic load still needs improvement.
Under burst traffic load circumstances, [8] embeds CCA slots into the shared transaction slot to perform the channel sense operation. Alert [18], a MAC protocol is designed for traffic bursts, aiming to minimize the delay of the first message. It reduces contention by a combination of time and frequency multiplexing. CoSenS [19], a collecting then sending burst protocol, uses traffic estimation algorithm to dynamically adjust its active period. Queue-MAC in [5] is another online load identification based MAC protocol; it piggybacks the queue length information onto every data packet to inform the routers for dynamic TDMA slots allocation. However, those proposed MACs still need to be improved since they still have high overheads in low traffic conditions.

In this paper, we propose pQueue-MAC for event-driven sensor networks. Using the main mechanism of Queue-MAC while integrating the SVP sampling to deal with low traffic load, pQueue-MAC has high energy efficiency when no event happens and achieves immediate yet energy efficient data transmission when events are detected, as we will further detail in the following sections.

3. Design of pQueue-MAC

Generally speaking, pQueue-MAC is designed for cluster based topology where each cluster has a cluster head and many son nodes. In this section, we first give out the basic idea of pQueue-MAC and then present different components of the design.

3.1. Preliminaries and Basic Idea

pQueue-MAC is based on superframe structure, as shown in Figure 1. Cluster head periodically broadcasts beacon frames to divide time into repeating superframe that contains active period and inactive period. All data exchanges will be completed in the active period and all nodes including cluster heads will turn off their radio to conserve energy in the inactive period. The superframe is composed of four parts: beacon period, variable TDMA period, synchronization based variable preamble sampling period, and inactive period.

Considering low traffic load occupies most of the time in event-driven applications, we adopt sender-initiated scheme, which is more suitable for low traffic load applications [20], to transmit these scattered data. Specifically, pQueue-MAC improves its energy efficiency through using SVP sampling, rather than adopting fixed CSMA scheme when dealing with low traffic, as shown in Figure 2. SVP sampling technique is much more efficient through short channel listening period, and beacon synchronization further reduces the preamble length. pQueue-MAC also adaptively adjusts its channel listening based on traffic load, avoiding both extra idle listening and insufficient bandwidth, which is brought by fixed duty cycle scheme when there are many nodes contending to send.

When traffic load increases, cluster heads in pQueue-MAC learn the traffic load situation in a quick manner. Without modifying the existing IEEE 802.15.4 MAC header, we use the first byte of payload field in data frame as the queue indicator to state the buffered packet number of the sensor nodes, as shown in Figure 3. When son nodes send their data to cluster head, it will learn the exact buffered packet numbers of the senders by extracting the queue indicator from the received packet. The collected queue length information will be further used to allocate TDMA slots to specific son nodes.

After cluster head node learns the load condition in its cluster by extracting the queue information from received packets, it accordingly allocates TDMA slots to son nodes in the next superframe period. The cluster head node embeds the allocation information into broadcasted beacon frame through which son nodes can learn the schedule by receiving the beacon. Beacon frame structure of cluster head is shown in Figure 4.

3.2. Traffic Transmission Control

3.2.1. Dealing with Low Traffic in Long Period

Son nodes, who have packet to transmit, learn the regular channel listening time of the cluster head by receiving and analyzing the beacon and then wake up just before the start of channel listening period. When no event happens, the network has very low traffic load; son nodes’ queue indicators will usually indicate zero. In this case, cluster head does not allocate any TDMA slots for son nodes to transmit data. A son node first contends to send a short preamble to the cluster head; once it gets acknowledged by the cluster head, it sends the data packet to the cluster head immediately. To realize high energy efficiency, we adopt the SVP sampling technique into our MAC protocol, since it features in very short time for channel listening to check potential communications.

When there is a successful transmission during the first channel listening period, the cluster head adaptively adds another channel listening period for potential coming packet. By piggybacking the queuing information, the successfully transmitted packets in the polling (the term polling is also sometimes used to indicate a base station interrogating each client to see if it has data to send; in this paper, polling refers only to each node sampling the channel to check for activity) period notify the cluster head about the son nodes’ load situation and implicitly apply for scheduled TDMA transmission slots. If cluster head finds no son node transmitting data during the regular channel listening period, it considers no packet generation in this superframe period and will turn off the radio to conserve energy. In this way, the cluster head can keep a very short listening period, say 2 ms, in each superframe period when the network is under low traffic.
Channel listening time (also the preamble length of sender node) is dependent on clock difference between sender node and receiver node. Based on the superframe structure, we add some synchronization information in beacon frame to do a rough synchronization. After receiving the beacon and getting synchronized, nodes can use synchronization based preamble sampling to reduce the length of preamble and thus avoid long time occupation on medium. The following shows the analysis of channel listening time.

Denote the synchronization period by $T_{\text{sync}}$, which is a configurable parameter, and the clock drift rate by $r_{\text{clk}}$. The maximum clock difference between two nodes is

$$t_{\text{diff}} = 2T_{\text{sync}}r_{\text{clk}}.$$  

This will happen when each node’s clock drifts in opposite direction. Since a sender does not know which direction its clock drifts compared with a receiver, it needs to put the guard time in both directions, making the total guard time be $2t_{\text{diff}}$. Therefore, the duration of the wake-up tone (preamble length) is

$$t_{\text{tone}} = 2t_{\text{diff}} = 4T_{\text{sync}}r_{\text{clk}}.$$  

Considering the time for cluster head polls the channel is short, accessing to cluster head through this short polling time may fail due to contentions which causes large back off time. But when the network is under low traffic which is the main application scenario for synchronization based preamble sampling, we will show next that the probability of preamble packets collision is very low.

Let $T_{\text{superframe}}$ denote the superframe period, and let $T_{\text{period_report}}$ be the average interval for nodes reporting to cluster heads when the system is in steady state (low traffic condition). $N$ is the number of son nodes in a cluster, and every son node generates data packet independently. So a son node’s probability of generating a data packet in one superframe period is

$$p = \frac{T_{\text{superframe}}}{T_{\text{period_report}}}.$$  

Considering that all son nodes generate data packets in Poisson distribution, we have

$$\lambda = N \cdot p.$$  

The probability of more than one son nodes generating data packet in the same superframe period is

$$P = \sum_{k=2}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!}.$$  

In our experimental setting, the superframe period is 1 s; the average interval nodes report to cluster heads in steady state is 100 s. Let us pick a large number of son nodes in one cluster as 30 to make the result more trustworthy; we have $\lambda = N \cdot p = 0.33$. Then, according to (5), the probability of more than one son node generating data packet in the same superframe period is

$$P = \sum_{k=2}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} = 0.0407.$$  

and probability of more than two son nodes generating data packet in the same superframe period is

$$P = \sum_{k=3}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} = 0.003.$$  

We can see that the probability of more than one son node transmitting in the same superframe is quite small, which
is only 4 percent. Also, through experiments, we validate that pQueue-MAC can totally handle the situation that two son nodes contend to transmit in the same channel listening time, leading to the fact that pQueue-MAC can successfully transmit data packets under low traffic load with a probability of almost 99.9%.

The reporting interval and the number of son nodes also have influence on the probability of preamble packets collisions (more than two concurrent transmissions in the same channel listening period). The relationships are shown in Figure 5. Smaller reporting interval and more contending nodes lead to larger probability. System parameters can be chosen appropriately according to Figure 5.

3.2.2. Dealing with Burst Traffic. When events occur, mass of data packets will be generated by son nodes, leading to a traffic burst. pQueue-MAC handles burst traffic mainly by using the variable TDMA period, in a way similar to [5]. Firstly, son nodes send packet in the cluster head’s polling period; then the cluster head learns the traffic load of every specific son node by extracting the queue length indicators which are piggybacked on the received packets. The cluster head accordingly allocates requested number of TDMA slots to son nodes for packet transmission in the next superframe. In this way, transmissions are arranged in a TDMA way which reduces collision and is energy efficient. High throughput will be achieved by allocating slots when nodes queues are building up in the network.

For son node with pending packets without allocated time slots in this superframe round, it tries to transmit in the SVP sampling period. Once one son node succeeds in transmitting a packet in the polling period, its queue information will be extracted by the cluster head, and it will be allocated TDMA slots in next round. Knowing their pending packets will be granted for TDMA slots in the next superframe cycle, these son nodes will transmit their packets within relative allocated TDMA slots. It is noted that every son node can only transmit one packet during the polling period; this further reduces the competition in the polling period.

Suppose there are \( n \) nodes with none zero queue length, so the total number of requested slots \( T_n \) is given as below:

\[
T_n = \sum_{i=1}^{n} N_i,
\]

where \( N_i \) denotes the requested slot number of node \( i \). Since the superframe has fixed length, there is an upper bound of the possible allocated slots, which is given by the following equation:

\[
UP_n = \frac{T_S - T_B - T_{poll} - T_R}{T_{slot}},
\]

where \( T_S \) denotes the duration of the superframe structure, \( T_B \) denotes the beacon interval, and \( T_{poll} \) and \( T_R \) are the largest interval the cluster head needs for adaptive channel polling and data relay, respectively. Thus, \( UP_n \) is the maximum number of slots that could be allocated. Let \( A_n \) denote the actual number of allocated time slots; then, we have

\[
A_n = \min\left( T_n, UP_n \right).
\]

In case \( T_n \) is larger than \( UP_n \), cluster head will allocate \( UP_n \) slots to \( n \) nodes proportionally according to their queue indicator values.

When events passed, the system returns into steady state, traffic load will decrease sharply, and the variable TDMA period will shrink down. In the case that the network traffic is extremely low, there will be no variable TDMA period, and only the synchronization based variable preamble sampling exists.

To better understand pQueue-MAC, its complete operation procedure is shown in the following with the help of Figure 6.

In the first round, though the event occurs at the end of the superframe, there is no packet buffering before cluster head’s channel polling. Due to finding no packet transmission during the first channel polling period, the cluster head turns into sleep mode, considering that there is no data coming. Because of no packet transmission in this period, there will be no TDMA slot allocation in the next superframe.

In the second round, due to the event occurring, there are a few data needed to be sent by son nodes, and they turn on their radio to transmit preamble packets in the polling period. Then the cluster head will receive packets (preamble and data packets) in the polling period of this round; it adaptively adds another polling period until it finds there is no transmission anymore. Remember that, during the data transmission, the buffered packet number is also transmitted to cluster head, which is used for TDMA slot allocation in the next superframe.

In the third round, by extracting and analyzing the piggybacked queue length information from received packets in the last superframe, the cluster head allocates requested TDMA slots to its son nodes for data transmission. When the variable TDMA period is over, there is still a regular channel
polling time. If a new son node wants to join this cluster, it waits for the beacon, learns the channel polling time by analyzing the received beacon, and then sends out its message in the polling period.

In the fourth round, though events passed, there are still some buffered packets in son nodes. When the variable TDMA period is over, there is no node transmitting in the polling period; cluster head turns off its radio to conserve energy.

In the fifth round, after the event is over, the network returns to low traffic situation and there is no buffered packet in son nodes. So when the cluster head finds no transmission in the first channel polling period, it turns into sleep mode.

3.3. Energy Consumption Model under Low Traffic Load Condition. Low traffic load, which is scattered packets used for system maintenances, occupies most of the time in event-driven application, so modeling this stage is very essential for system parameter configuration. In this section, we find the optimal operating parameters for our protocol to achieve its best performance. Referring to B-MAC [9] and SCP-MAC [11], a simplified model of energy consumption is explored and its lower bound is derived. Considering burst traffic’s uncertainties and rarity, its energy consumption modeling will be our future work.

Our analysis is based on a cluster which contains a cluster head and several son nodes. Son nodes nonperiodically generate data packets and transmit them in the polling period using synchronization based preamble sampling technique. We focus on the energy consumption of the radio, and we do not take other components consumption such as the CPU and sensors into consideration.

As we mentioned earlier, the radio module has four states: transmitting state, receiving state, listening state, and sleeping state. We use $P_{tx}$, $P_{rx}$, $P_{listen}$, and $P_{sleep}$ to present the energy cost of these four states per unit time, respectively. The energy consumption of the radio is determined by how much time it spends on transmitting, receiving, listening, polling, and sleeping, denoted by $t_{tx}$, $t_{tx}$, $t_{tx}$, $t_{poll}$, and $t_{sleep}$, respectively. In our analysis, all these time values are normalized to one second. They represent the fractions of time the node spends in one second in different states.

Channel polling period contains the time radio transitions from sleep to listen and a short time for channel detection. We denote polling duration by $t_{poll}$ and its average power consumption by $P_{poll}$. As preamble packets are sent out using CSMA technique, carrier sensing is needed. We denote the average time in carrier sensing by $t_{cs1}$. After carrier sensing and finding out that the channel is vacant, son nodes send a preamble to wake up the cluster head. We include it into the packet length $L_{data}$.

Table 1 summarizes the main variables used in the paper and gives their typical values for an IEEE802.15.4 radio (STM32W108 [21]). The expected energy consumption per node is the sum of energies consumed in the above five states as follows:

$$E = E_{cs} + E_{tx} + E_{rx} + E_{poll} + E_{sleep}$$

$$= P_{listen}t_{cs} + P_{tx}t_{tx} + P_{rx}t_{rx} + P_{poll}t_{poll} + P_{sleep}t_{sleep}. \quad (11)$$

3.3.1. Son Nodes’ Energy Consumption Modeling. Son nodes only conduct carrier sensing before transmitting data packet; the expected time in carrier sensing is

$$t_{cs,son} = t_{cs1}r_{data}, \quad (12)$$

where $r_{data}$ is the transmitting rate of each node. After carrier sensing, a node sends a wake-up tone to wake up the receiver and then sends data packets. The expected time in transmitting state for son nodes is

$$t_{tx,son} = (t_{tone} + L_{data}t_{B})r_{data}, \quad (13)$$
where \( t_{pl} \) is the time needed to transmit or receive a byte. Son nodes only receive beacon frames from the cluster head, so the expected time for receiving a beacon is

\[
\tau_{rx,son} = (t_{tone} + L_{beacon}t_{pl})r_{sync}.
\]

Son nodes need to poll the channel every superframe period for beacon frame, so

\[
t_{poll,son} = \tau_{pl}r_{sync}\left(\frac{T_{superframe}}{T_{reporting,period}}\right),
\]

where \( t_{pl} \) is the average time to poll the channel and \( t_{sleep,son} \) is expressed as follows:

\[
t_{sleep,son} = 1 - t_{cs,son} - \tau_{rx,son} - \tau_{tx,son}.
\]

Substituting (12)–(16) into (11), we have the energy consumption for son nodes in our protocol as

\[
E_{son} = E_{cs,son} + E_{tx,son} + E_{rx,son} + E_{poll,son} + E_{sleep,son}
\]

\[
= P_{\text{listen}}t_{cs1} \cdot r_{data} + P_{\text{tx}}(t_{tone} + L_{data}t_{pl})r_{data}
\]

\[
+ P_{\text{rx}}(t_{tone} + L_{beacon}t_{pl})r_{sync} + P_{\text{poll}} \cdot r_{sync}
\]

\[
+ P_{\text{sleep}}(1 - \tau_{rx,son} - \tau_{tx,son} - t_{cs,son}).
\]

\[\text{3.3.2. Cluster Head's Energy Consumption Modeling.}\]

Cluster head requires carrier sensing before transmitting beacon frame; the expected time in carrier sense is

\[
t_{cs,cluster} = t_{cs1}r_{sync}.
\]

After carrier sensing, cluster head sends a wake-up tone to wake up son nodes in its cluster and then sends the beacon frame. The expected time in transmitting state for cluster head is

\[
t_{tx,cluster} = (t_{tone} + L_{beacon}t_{pl})r_{sync}.
\]

Cluster head receives data packet generated by son nodes in its cluster, so the expected time in receiving is

\[
t_{rx,cluster} = n(t_{tone} + L_{data}t_{pl})r_{data}.
\]

where \( n \) is the son nodes number in one cluster. The expected polling time is

\[
t_{poll} = \frac{t_{pl}}{T_{p}}.
\]

where \( t_{pl} \) is the average time to poll the channel and \( T_{p} \) is the channel polling period. Then

\[
t_{sleep,cluster} = 1 - t_{rx,cluster} - t_{tx,cluster} - t_{poll} - t_{cs,cluster}.
\]

Substituting (18)–(22) into (11), we have the energy consumption for cluster head in our MAC as

\[
E_{cluster} = E_{cs,cluster} + E_{tx,cluster} + E_{rx,cluster} + E_{sleep,cluster} + E_{poll}
\]

\[
= P_{\text{listen}}t_{cs1} \cdot r_{sync} + P_{\text{tx}}(t_{tone} + L_{beacon}t_{pl})r_{sync}
\]

\[
+ P_{\text{rx}}(t_{tone} + L_{data}t_{pl})r_{data} + P_{\text{poll}} \cdot \left(\frac{t_{pl}}{T_{p}}\right).
\]

From the above discussions, we derive the energy consumption in one cluster as follows:

\[
E_{\text{sum}} = nE_{son} + E_{cluster}.
\]

Substituting (17) and (23) into (24), we conclude that energy consumption in a cluster \( E_{\text{sum}} \) is related to the number of son nodes \( n \), beacon frame length \( L_{beacon} \) and packet length \( L_{data} \), \( r_{data} \), \( T_{sync} \), and \( T_{p} \). While \( n \) is constant once the topology is stable, \( r_{data} \) is constant due to the almost the same average data generating period in this stage; \( L_{data} \) and \( L_{beacon} \) can be normalized to a constant length. Ideally, cluster head should only poll the channel when there is a transmission from son nodes. Thus the optimal polling period for this stage is

\[
T_{p}^* = \frac{1}{n r_{data}}.
\]

Then the problem of how to choose the optimal synchronization period \( T_{sync} \) to minimize \( E_{\text{sum}} \) can be solved by the following equation:

\[
\frac{dE_{\text{sum}}}{dT_{sync}} = 0.
\]

\[\text{4. Performance Evaluation}\]

To evaluate the realistic performance of pQueue-MAC, we implement pQueue-MAC on the STM32W108 chips. It is one of the commonly used chips that offer IEEE 802.15.4 standard communication. Moreover, it provides a Simple MAC library offering a set of APIs so that we can access the chips’ lower MAC functionality. This will greatly simplify the application development.

We then set up a test bed to compare pQueue-MAC with Queue-MAC and the modified SCP-MAC in which we
Table 2: Fundamental parameters setting for MAC protocols evaluation.

| Parameter                        | Setting       |
|----------------------------------|---------------|
| Superframe length                | 1000 ms       |
| Relay time                       | 500 ms        |
| Operation channel                | Sub (11, 14, 17), main (20) |
| Packet length                    | 64 bytes      |
| Max retry                        | 3             |
| Fixed CSMA period                | 20 ms         |
| Slot size                        | 5 ms          |
| Channel polling interval         | 5 ms          |
| Channel polling time             | 2 ms          |

The network has a clustering topology, which is shown in Figure 7.

The network is composed of one sink node and three clusters. Each cluster contains five son nodes and one cluster head. Every son node communicates with its cluster head, and cluster heads relay the packets received to sink node. To avoid interference from other clusters, we adopt FDMA (frequency division multiple access) to let these three clusters operate in different channels. Data exchange in one cluster is in their own channel. When relaying packets to the sink node, cluster heads will switch to the main channel to communicate with the sink node.

We set up two experimental scenarios to fully explore pQueue-MAC's performance. One is an event-driven application scenario in which packets are rarely generated when no event happens but has burst traffic when an event is assumed to be detected. The other one is a general one with varying periodic data rate. Fundamental parameters, such as superframe length, relay time, packet length, and operation channel, in these three MAC protocols are totally identical, as shown in Table 2.

4.1. Event-Driven Scenario Experiment. In this scenario, all son nodes generate packets conforming to uniform distribution with an average data generating period being 100 s to simulate a monitoring mode. And we set two bursts at 300 s and 520 s separately to simulate an event detection. The first burst packet is generated at a rate of one packet per 100 ms which lasts for 20 s, and the second burst packet is generated at the rate of one packet per 200 ms which lasts for 30 s. We measure the two key parameters, energy efficiency and packet delay, among which the packet delay is obtained by measuring the difference between the time the packet was generated and the time it was received by the sink node.

Figure 8 shows senders’ queue length situation along with the system time. Smaller buffered packet number means good schedule to transmit data packet and of course low latency. When burst occurs, SCP-MAC suffers from sharply data packet collision because of its contention scheme; this definitely leads to large amounts of data buffered in their own nodes. On the other hand, both Queue-MAC and pQueueMAC maintain short queue length by providing sufficient bandwidth for data transmission.

Figure 9 shows the packet delay of all three MAC protocols. When burst occurs, packet delay is measured every 2 s. Because of randomness of contention, the packet delay is mainly dependent on the traffic condition the moment it transmits. And as the load increases, its delay usually rises as well. SCP-MAC gains the largest delay due to its severe contention. While good schedule makes packet transmit in a faster way, pQueue-MAC and Queue-MAC handle well with burst as above chart shows. Previous two comparisons show the low latency character of our pQueue-MAC.

Figure 10 shows the duty cycle comparison of all protocols in event-driven scenario. SCP-MAC adopts contention based transmission scheme which has packet collision and retransmission issue; its performance when applied in burst or heavy traffic is poor. Duty cycle performance of Queue-MAC is quite satisfactory except it has some problems in transmitting packet when the traffic is low. On one hand, when the number of senders increases, fixed CSMA scheme cannot provide sufficient bandwidth, leading to packet loss. On the other hand, if there is no data packet coming to cluster heads, it keeps cluster heads active in case of possible
coming data packet which leads to idle listening. Duty cycle of receiver nodes in Queue-MAC is especially high. pQueue-MAC takes the merits of both Queue-MAC and SCP-MAC, which makes pQueue-MAC achieve high energy efficiency and low latency.

4.2. General Experiment. After verification of pQueue-MAC’s performance in event-driven scenario, we then conducted a new experiment to show its performance under general periodic data generation scenario, which can also show pQueue-MAC’s energy efficiency improvement under low traffic more clearly compared with Queue-MAC. We vary the data rate from less than 100 ms per packet to 50000 ms per packet to test energy efficiency and packet transmission delay of pQueue-MAC, Queue-MAC, and SCP-MAC.

We take the mean value of all successfully received packets, as shown in Figure 12. When data rate increases, SCP-MAC gains larger packet delay. This results from the reason that SCP-MAC is a contention based MAC protocol; more data packet transmission leads to more data collision and then larger packet delay. On the other hand, Queue-MAC arranges most of the data transmission in TDMA period which sharply reduces packet collisions. Like Queue-MAC, pQueue-MAC also schedules TDMA slots to son nodes for massive packet transmission which has less data collision. Less data collision makes fluent data transmission without additional packet retransmission and carrier sensing; thus Queue-MAC and pQueue-MAC achieve low latency. When data rate is quite low, both SCP-MAC and pQueue-MAC adopt adaptive transmission scheme, succeeding in sending all packets in one superframe. Since Queue-MAC adopts the fixed CSMA scheme to transmit low traffic load, the fixed bandwidth restricts the data transmission. Packets that failed to contend to transmit in this superframe need to contend to send in the next superframe, leading to a bit larger delay as shown in Figure 12.

Figures 12 and 13 show the comparison of energy consumption of receiver and sender node separately in all three MAC protocols. For all protocols, we calculate nodes’ energy consumption of receiver and sender node separately in all three MAC protocols.
consumption through energy they used divided by successfully received/transmitted packets' number to total packet number ratio as the following formulation shows:

\[ E_{\text{total}} = \frac{E}{N_{\text{success}}/N_{\text{total}}}, \]  

(27)

where \( E_{\text{total}} \) is the total energy consumption of nodes, \( E \) is nodes' actual energy consumption, \( N_{\text{success}} \) denotes the number of successfully received/transmitted packets, and \( N_{\text{total}} \) is the number of the total generated packets. Based on the findings in [22] and the characters of the chip [21] we used, we assume that power consumption under receiving/idle state and transmitting state is the same, as 27 mA.

When data rate is high, data collision in SCP-MAC leads to packet retransmission, additional channel listening, and channel carrier sensing; these cost lots of energy. Due to the queue indicator, pQueue-MAC and Queue-MAC can allocate slots to son nodes accordingly, making it possible to transmit data packets in a quick manner. When data rate decreases, data collision in SCP-MAC becomes less severe, and its energy efficiency improves. But due to the fixed CSMA period existing in Queue-MAC, it cannot deal well with the dynamic traffic when data rate is below one packet per superframe period. Fixed CSMA period only guarantees part of son nodes' packet transmission. Thus the energy efficiency of Queue-MAC in low traffic condition is not high. On the other hand, while dealing with low traffic condition, pQueue-MAC adopts transmission mechanism similar to SCP-MAC, so pQueue-MAC achieves higher energy efficiency compared to Queue-MAC.

Figure 14 shows the packet reception ratio (PRR) comparison in all three MAC protocols. When the network traffic is heavy, the data rate is more than 20 packets/s. Due to the contention scheme it adopts, PRR in SCP-MAC is lower than Queue-MAC and pQueue-MAC which transmit most of the packets through TDMA slots. But thanks to the two-phase contention which is the combination of preamble contention and packet contention, reception ratio of SCP-MAC is still high. When the traffic is low, especially when the data rate is below one packet per transmission period, the value of queue indicator is always zero and fixed CSMA period of Queue-MAC cannot deal with the dynamic traffic load. All five son nodes contend to transmit, but fixed CSMA period cannot provide sufficient bandwidth for all son nodes to successfully transmit packets, which leads to packet loss. SCP-MAC and pQueue-MAC can achieve high reception ratio performance in low traffic.

5. Conclusion

This paper proposed pQueue-MAC, a new traffic adaptive duty cycle MAC protocol designed for event-driven sensor networks. pQueue-MAC uses the synchronization based preamble sampling to deal with low traffic load in a long period and reacts to burst events by dynamically allocating the variable TDMA slots. SVP sampling has high energy efficiency with short channel listening period and dynamically allocated TDMA slots could achieve low transmission delay by making full use of the communication bandwidth according to the traffic load.

We have successfully implemented pQueue-MAC on STM32W108 chips, and for comparison we also implemented SCP-MAC and Queue-MAC to test their energy efficiency, network delay, and nodes' queue length in numerous experiments. The results clearly show that pQueue-MAC achieves better energy efficiency when no event happens, and once the event is detected, we can transmit this burst packet with low latency.

In the future, we are planning to extend the proposed pQueue-MAC for supporting multihop cluster-tree networks. Furthermore, we will develop a model to analyze the latency performance of pQueue-MAC.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.
Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (no. NSFC61273079), the Strategic Priority Research Program of the Chinese Academy of Sciences (nos. XDA06020200 and XDA06020201), the Open Research Project of the State Key Lab. of Industrial Control Tech., Zhejiang University (nos. ICTI423, ICTI424, ICTI425, and ICTI430), and the Key Lab. of Wireless Sensor Network & Communication of Chinese Academy of Sciences (no. WSNC2011001).

References

[1] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in Proceeding of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM ’02), vol. 3, pp. 1567–1576, June 2002.

[2] T. van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys ’03), November 2003.

[3] P. Hurni and T. Braun, "MaxMAC: a maximally traffic-adaptive MAC protocol for wireless sensor networks," in Proceedings of the 7th European Conference on Wireless Sensor Networks (EWSN ’10), Coimbra, Portugal, February 2010.

[4] I. Rhee, A. Warrier, and M. Aia, “Z-MAC: a hybrid MAC for wireless sensor networks,” IEEE/ACM Transactions on Networking, vol. 16, no. 3, pp. 511–524, 2008.

[5] S. Zhuo, Y. Song, and Z. Wang, "Queue-MAC: a queue-length aware hybrid CSMA/TDMA MAC protocol for providing dynamic adaptation to traffic and duty-cycle variation in wireless sensor networks," in Proceedings of the 9th IEEE International Workshop on Factory Communication Systems (WFCS ’12), pp. 105–114, May 2012.

[6] S. Zhuo, Z. Wang, and Y. Q. Song, "iQueue-MAC: a traffic adaptive duty-cycled MAC protocol with dynamic slot allocation," in Proceedings of the 10th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON ’13), 2013.

[7] Q. Zheng, X. Fei, T. Tang et al., "A hybrid, dynamic traffic-adaptive MAC protocol for Wireless Sensor Networks," in Proceedings of the 15th International Conference on Advanced Communication Technology: Smart Services with Internet of Things! (ICACT ’13), pp. 574–578, January 2013.

[8] D. Yang, M. Gidlund, W. Shen, Y. Xu, T. Zhang, and H. Zhang, "CCA-Embedded TDMA enabling acyclic traffic in industrial wireless sensor networks," Ad Hoc Networks, vol. 11, no. 3, pp. 1105–1121, 2013.

[9] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys ’04), pp. 95–107, November 2004.

[10] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks," in Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys ’06), pp. 307–320, November 2006.

[11] W. Ye, F. Silva, and J. Heidemann, "Ultra-low duty cycle MAC with scheduled channel polling," in Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys ’06), pp. 321–334, November 2006.

[12] Y. Sun, O. Gurewitz, and D. B. Johnson, "RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks," in Proceedings of the 6th ACM Conference on Embedded Networked Sensor Systems (SenSys ’08), pp. 1–14, November 2008.

[13] P. Park, C. Fischione, A. Bonivento, K. H. Johansson, and A. Sangiovanni-Vincenti, "Breath: an adaptive protocol for industrial control applications using wireless sensor networks," IEEE Transactions on Mobile Computing, vol. 10, no. 6, pp. 821–828, 2011.

[14] M. Zimmerling, F. Ferrari, L. Mottola, T. Voigt, and L. Thiele, "pTunes: runtime parameter adaptation for low-power MAC protocols," in Proceedings of the 11th ACM/IEEE Conference on Information Processing in Sensing Networks (IPSN ’12), pp. 173–184, Beijing, China, April 2012.

[15] F. Österlind, L. Mottola, T. Voigt, N. Tsiftes, and A. Dunkels, "Strawman: resolving collisions in bursty low-power wireless networks," in Proceedings of the 11th ACM/IEEE Conference on Information Processing in Sensing Networks (IPSN ’12), pp. 161–171, April 2012.

[16] S. H. Lee and L. Choi, "A+MAC: a streamlined variable duty-cycle MAC protocol for wireless sensor networks," International Journal of Distributed Sensor Networks, vol. 2013, Article ID 784509, 8 pages, 2013.

[17] IEEE 802.15.4 Standard, “Part 15.4, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs),” 2006.

[18] B. Namboodiri and A. Keshavarzian, “Alert: an adaptive low-latency event-driven MAC protocol for wireless sensor networks,” in Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN ’08), pp. 159–170, April 2008.

[19] B. Neﬁz and Y. Q. Song, “QoS for wireless sensor networks: enabling service differentiation at the MAC sub-layer using CoSenS,” Ad Hoc Networks, vol. 10, no. 4, pp. 680–695, 2012.

[20] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi, “The evolution of MAC protocols in wireless sensor networks: a survey,” IEEE Communications Surveys and Tutorials, vol. 15, no. 1, pp. 101–120, 2013.

[21] STM32W108 datasheet, http://www.datasheetarchive.com/ST-M32W108datasheet.html.

[22] A. Prayati, C. Antonopoulos, T. Stoyanova, C. Koulamas, and G. Papadopoulos, “A modeling approach on the TelosB WSN platform power consumption,” Journal of Systems and Software, vol. 83, no. 8, pp. 1355–1363, 2010.
Submit your manuscripts at
http://www.hindawi.com