ADCA: An Asynchronous Duty Cycle Adjustment MAC Protocol for Wireless Sensor Networks

Yu-Chia Chang[1], Jehn-Ruey Jiang[1], Jang-Ping Sheu[2] and Hsin-Yi Shih[1]
[1]Department of Computer Science and Information Engineering, National Central University, Jhongli, 32054, Taiwan
[2]Department of Computer Science, National Tsing Hua University, Hsinchu 30013, Taiwan
E-mail: jimchang@axp1.csie.ncu.edu.tw, jrjiang@csie.ncu.edu.tw and sheujp@cs.nthu.edu.tw

***Abstract*--** In this paper, we propose an asynchronous duty cycle adjustment MAC protocol, called ADCA, for the wireless sensor network (WSN). ADCA is a sleep/wakeup schedule-based protocol to reduce power consumption without lowering network throughput or lengthening transmission delay. It is asynchronous; it allows each node in the WSN to set its own sleep/wakeup schedule independently. The media access is thus staggered and collisions are reduced. According to the statuses of previous transmissions, ADCA adjusts the duty cycle length for shortening transmission delay and increasing throughput. Simulation results show that ADCA outperforms related ones in terms of energy saving, network throughput and transmission delay.

***Keywords*--**Wireless sensor networks, sleep/wakeup schedule, duty cycle, energy efficiency, transmission latency

I. INTRODUCTION

The rapid progress of wireless communications and micro-electro-mechanical system (MEMS) technology has made *wireless sensor networks (WSNs)* a hot research topic recently. A WSN consists of many spatially distributed, resource-constrained sensor nodes equipped with microcontrollers, short-range wireless radios, and analog/digital sensors. Sensor nodes sense environmental conditions, such as temperature, light, sound, or vibration, etc., and transmit the sensed data to the sink node through multi-hop communication links. There are many applications of WSNs, such as target tracking, environment monitoring, and home security, etc [1] [8] [12].

Energy conservation is one of the most important issues in WSNs, since sensor nodes are usually powered by batteries. The radio transceiver is the most power consuming component in a sensor node. A typical radio transceiver consists of four possible modes with different power consumption: *transmitting*, *receiving*, *listening*, and *sleeping*. The first three modes are also called *active* or *wakeup* modes, in which more energy is consumed. For example, the relative power consumptions of the four modes of MICAz mote [21] are 17.4, 19.7, 19.7 and 0.426, respectively. Observing *idle listening*, the status that a sensor node turns on the radio to monitor wireless medium but do not receive any packets, wastes a lot of energy, some researchers propose *medium access control* (MAC) protocols [2,5-7,9,13-15,17,18] to turn the radio into sleeping mode as long as possible to save energy for prolonging the network lifetime. However, the radio should be scheduled to be in wakeup mode periodically to monitor, send or receive data packets. Those protocols that make the radio alternate between sleeping and wakeup modes are called the *sleep/wakeup schedule protocols*. As shown in [10], when the *duty cycle* (i.e., active period) of the radio is reduced to 1 percent, the power consumption of the sensor node can be reduced by a factor of 50.

Some sleep/wakeup schedule protocols suggest that all sensor nodes synchronize their timers and maintain a global sleep/wakeup schedule. However, timer synchronization causes a large overhead. Furthermore, the global schedule may lead to low duty cycle utilization. For example, when two neighboring nodes are exchanging data, all two-hop neighbors of the two nodes are prohibited from doing so. Those prohibited nodes stay at receiving mode until the data exchanging finished, which causes an energy-wasting status called *overhearing*. The RTS/CTS scheme [17] can be used to avoid overhearing as well as to solve the hidden terminal problem. But, the scheme's overhead is relatively large for WSNs since packets in WSNs are usually very small [3] [21].

In this paper, we propose an *Asynchronous Duty Cycle Adjustment (ADCA)* MAC protocol to provide low energy consumption, low transmission latency and high throughput in WSNs. ADCA is an asynchronous sleep/wakeup schedule protocol which needs not synchronize nodes' timers, so the timer synchronization overhead is avoided. It also allows each node to set its own sleep/wakeup schedule independently. Since nodes' schedules are staggered, the channel utilization is increased and the occurrences of overhearing are reduced. Furthermore, ADCA tries to increase the throughput and to decrease the transmission delay by adjusting two time periods: the *extended period* and the *next contention period*. We will perform simulation experiments for ADCA and compare the simulation results with those of related protocols to show the advantages of ADCA.

The rest of the paper is organized as follows. Section 2 introduces some related sleep/wakeup schedule MAC protocols. The proposed ADCA protocol is then described in Section 3. The simulation results and comparisons of protocol performance are shown in Section 4. And at last, Section 5 concludes this paper.

II. RELATED WORK

Over the past few years, several sleep/wakeup schedule protocols have been developed for WSNs. The goals of those protocols are to decrease the energy consumption, to improve the network throughput and/or to shorten the transmission delay. The protocols can be classified into two categories: synchronous [7, 13-15, 17] and asynchronous [2, 5, 6, 9, 18]. Below, we review some synchronous and asynchronous sleep/wakeup schedule protocols for WSNs.

A. Synchronous sleep/wakeup schedule protocols

S-MAC [17] and T-MAC [14] are two well-known sleep/wakeup schedule MAC protocols in WSNs. S-MAC tries to eliminate the four sources of energy waste: collision, overhearing, control packet overhead, and idle listening. It divides time axis into fixed length *cycle period*, which is further divided into *SYN*, *contention* and *sleep periods* (see Fig. 1). Nodes try to synchronize their schedules in the SYN period,

and contend for sending packets in the contention period. It thus periodically puts sensor nodes into sleeping mode at a low and fixed duty cycle ratio. T-MAC improves S-MAC by using adaptive duty cycles (see Fig. 1).

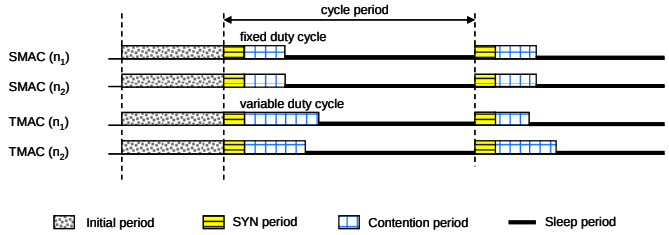

Fig. 1: the sleep/wakeup schedules of S-MAC and T-MAC protocols.

Sensor nodes turn the radio into sleeping mode when there is no activity during a time period $T_A = 1.5\ (C + R + T)$, where C is the length of the contention period, R is the time period of *RTS* packet transmission, and T is a short time between the end of the *RTS* and the begin of the *CTS* packet transmission. Because of the duty cycle adjustment, T-MAC provides a better throughput than S-MAC under varied traffic. The synchronous scheduling of S-MAC and T-MAC makes sensor nodes suffer high contention; the throughput and channel utilization are thus low.

U-MAC [13] is similar to S-MAC. It provides three modifications to improve S-MAC: (1) to assign different duty cycles to nodes of different traffic loads, (2) to calculate suitable duty cycles based on utilization and (3) to make nodes go to sleeping mode selectively after transmission. The calculation of utilization takes transmitting time, receiving time and idle listening time into consideration. If the current utilization is larger (resp., less) than the high (resp., low) traffic load threshold, the duty cycle will be increased (resp., decreased). U-MAC can save more energy than S-MAC. However, the problems of low channel utilization and long transmission delay are still not solved efficiently.

P-MAC [15] is a time-slotted scheduling protocol. Each sensor node determines its sleep/wakeup schedule based on the traffic patterns of itself and its neighbors. In P-MAC, time is divided into Super Time Frames (STFs), each of which has two sub-frames: PETF (Pattern Exchange Time Frame) and PRTF (Pattern Repeat Time Frame). In PETF, nodes collect and exchange with neighbors their traffic patterns which repeat in PRTE. The purpose of the pattern exchange is to ensure that the schedules of the sensor nodes can adapt to the current traffic load. The drawback of P-MAC protocol is that the time-slotted structure needs very accurate time synchronization which leads to a high maintenance overhead. P-MAC also has the problems of long transmission delay and low network throughput.

H-MAC [7] combines the schemes used in the contention-based and TDMA-based MAC protocols [16, 19] for WSNs. It uses a slotted frame structure and a wakeup technique to achieve high energy efficiency. The slots for data transmission in H-MAC are used on an on-demand basis. Each node randomly selects its own wakeup slot and notifies the slot number to all its neighbors. A node needs to collect one-hop neighbor information constantly. A sender can wake up the receiver by a wakeup message sent at the receiver's wakeup slot. Then, the receiver will wake up at the specified data slot to receive the data packet. H-MAC needs very accurate time synchronization which causes a high overhead.

B. Asynchronous sleep/wakeup schedule protocols

D-MAC [6] is a non-delay forwarding scheme for reporting data to the sink node along a data gathering tree. A node skews its wakeup schedule with *dt* time ahead of that of the sink (*d* is the depth of the tree and *t* is the time of data packet transmission). D-MAC staggers the sleep/wakeup schedule of nodes in the data gathering tree to decrease the transmission delay. D-MAC is not flexible, though. Communication between arbitrary nodes is not allowed. Furthermore, if the network topology is changing, all the nodes need to re-construct their sleep/wakeup schedules.

B-MAC [9] uses preamble signaling for a sender to wake up the receiver. A node periodically wakes up for a short period at every cycle period to check preamble signals. If no signal is present, the node turns the radio off after a time-out. It keeps the radio on if a preamble is sensed. It is obvious that the preamble signal should be long enough so that the periodically awaking receiver can detect it. The data packet is then sent after the preamble. Consequently, both the sender and the receiver waste much energy during the communication, and the transmission delay may be long.

In **Wise-MAC [2],** each node maintains the sleep/wakeup schedules of its neighbors. A node periodically wakes up for a short interval. If there is no wake up preamble (WUP) send from other nodes, the node goes into sleeping mode to conserve energy. When a sensor node has packets to send, the node will hold the data packets until the receiver is active. According to the receiver's schedule, the node sends a short duration of the WUP to wake up the receiver. Then it transmits data to the receiver and waits an ACK packet from the receiver. Wise-MAC is a simple protocol for saving energy. However, it also has the long transmission delay problem.

SyncWUF [18] combines both the simple signaling and the wakeup frame (WUF) technique together. The idea of SyncWUF is that the sender records all nodes' schedules and adjusts the wakeup signal (WUS) accordingly. To transmit a data packet, a sender checks the receiver's schedule first. If the schedule is up-to-date, a short simple wakeup preamble (WUP) is used as in the Wise-MAC protocol. If the schedule is out-of-date, multiple short wakeup frames (SWUFs) are used to reduce the unnecessary waiting time. SyncWUF is an energy efficient protocol. However, if a sender misses the receiver's active period, it must await until next cycle. The transmission delay of SyncWUF is thus long.

III. ADCA PROTOCOL

A. Overview

ADCA (asynchronous duty cycle adjustment) protocol is an asynchronous sleep/wakeup schedule protocol which needs not synchronize nodes' timers and allows each node to set its own sleep/wakeup schedule independently. When a node starts up, it first decides its own sleep/wakeup schedule, broadcasts the schedule and collects all neighbors' schedules within an initial period of arbitrary length (see Fig. 2). It then starts executing its sleep/wakeup schedule individually. The sleep/wakeup schedule of a node is composed of repeated and

fix-lengthened cycle periods, each of which in turn consists of a *contention period*, a *SYN period*, an *extended period* and a *sleep period* as shown in Fig. 2.

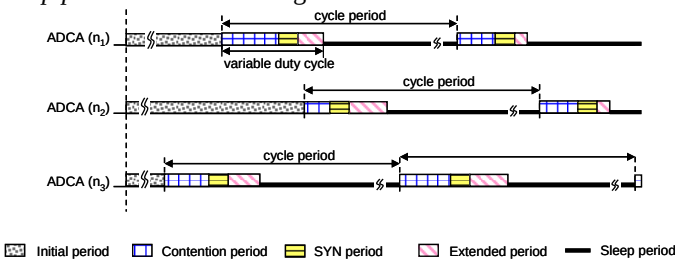

Fig. 2: the sleep/wakeup schedule of ADCA protocol.

In ADCA, a node listens to the channel for possible incoming packets at the contention period and broadcasts a SYN packet to the intended senders at the SYN period. An extended period immediately follows the SYN period to prolong the active time. A node turns its radio into sleeping mode to save energy in the sleeping period. When a node has a packet to send, it checks its neighbor-schedule table and contends to send the data packet in the receiver's contention period. The node then goes into sleeping mode after the transmission. If a sender fails to send the data packet in the receiver's contention period, it switches the radio into the receiving mode to wait for the receiver's SYN packet and tries to retransmit the data packet in the receiver's extended period. If the transmission still fails during the receiver's extended period, the sender waits for the contention period in the receiver's next cycle period.

Since nodes maintain their schedules asynchronously, the schedules are staggered and the successful transmission rate and channel utilization are thus increased. Furthermore, ADCA allows nodes to dynamically adjust the contention period and the extended period based on current transmission statuses and/or traffic loads. In this way, the throughput is increased and the transmission delay is decreased without scanting energy efficiency. Below, we show how to adjust the two periods in the next subsection.

B. Duty cycle adjustment

In ADCA, a node adjusts its active period based on transmission statuses and traffic loads. Each node records the time of channel idle (T_i), the time of channel busy (T_b) and the number of overheard packets (N_{oh}) during the *observed active periods*, i.e., the last extended period and the current contention period. It then calculates, at the end of the contention period (the *adjustment point*), the length of the extended period (A_1) and the length of the next contention period (A_2) accordingly (see Fig. 3). The node then broadcasts its new schedule with the two newly calculated periods in a SYN packet.

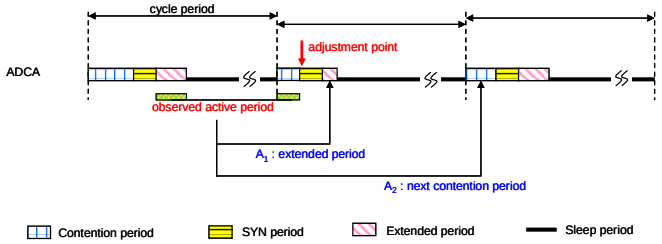

Fig. 3: the duty cycle adjustment of ADCA

The length of extended period (EP) is adjusted according to Eq. 1. T_{bad} in Eq. 1 represents the time of collision and channel unstable (interference), and N_{oh} stands for the number of overheard packets. As shown in Eq. 2, T_{data} is defined to be the average transmission time of a data packet including the time for transmitting data (*packet size/data rate*) and a random back-off within a contention window of fixed *cw* slots, each of which is of length T_{slot}.

$$EP = \left(\left\lfloor \frac{T_{bad}}{T_{data}} \right\rfloor + N_{oh}\right) \times T_{data} \quad (1)$$

$$T_{data} = \frac{cw}{2} \times T_{slot} + \frac{packet\ size}{data\ rate} \quad (2)$$

In Fig. 4, we show some bad receiving situations such as collision, overhearing and interference, which will increase the transmission delay and decrease the channel utilization. So, a node should extend the extended period to compensate for the bad receiving situations. If a receiver detects more collisions or overhearing events during the contention period, it knows that the sender has smaller probability to complete data packet transmission successfully. Therefore, it adds multiple times of T_{data} to the extended period to increase the sender's successful transmission probability and to decreases the transmission delay.

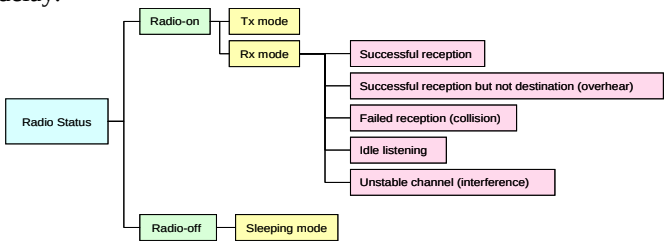

Fig. 4: the radio statuses of a sensor node

The next contention period adjustment is for the purpose of adapting to the traffic of the observed active period. Therefore, the length of the next contention period is proportional to traffic loads. The length of the next contention period (CP) is adjusted according to Eq. 3, where T_{rx} is the total time that a node is in the receiving mode during pervious cycle period, *CCP* means the length of the current contention period, and the sum of channel idle time T_i and channel busy time T_b is equal to T_{rx}. Eq. 3 takes channel idle time and channel busy time into consideration. α and β are weight parameters related to the two time spans. α should be negative to reflect the channel idle time and β should be positive to reflect the channel busy time. Their values are usually determined by the applications, and we suggest setting α=-1 and β=1 in this paper. Therefore, if $T_i > T_b$ then *CP* gets smaller; otherwise, *CP* gets larger. Certainly, *CP* should be larger than a pre-specified minimum value and should only make the contention period last until the end of the cycle period.

$$CP = CCP \times \left(1 + \alpha \frac{T_i}{T_{rx}} + \beta \frac{T_b}{T_{rx}}\right) \quad (3)$$

IV. SIMULATION RESULTS

In this section, we simulate ADCA for the sake of comparison. Because all existent asynchronous protocols do not address the problem of long transmission delay, we do not compare

ADCA with them. Also because adaptive protocols, such as T-MAC, have better performance than others, we only compare ADCA with T-MAC below.

A. The Simulation Environment

We use ns-2 [20] to simulate ADCA and T-MAC for two scenarios. One scenario is all-to-one environment where all nodes periodically report data to a sink node. The other is n-to-n environment where n/2 connections are randomly established among all n nodes for exchanging data. The RTS/CTS scheme is disabled in both ADCA and T-MAC. AODV [4] is adopted as the routing protocol and a CSMA-like scheme [11] is used to avoid collision. The transmission rate is 250kbps and the transmission range is 25m. The data, SYN and ACK packets are 50, 10 and 10 bytes, respectively. Each experiment lasts 1000s and each outcome is obtained by averaging 20 experiments' results. Two parameters are tunable in the simulation: network density and traffic load. We assume 20, 30 or 40 nodes are uniformly deployed in a 100m x 100m square area; i.e., the average degrees of nodes are 4, 6 or 8. The traffic loads are constant bit rates (CBR) which are set to 1, 10, 20, 30, 40, 50 or 60 packets per second. We measure the following three metrics: (1) the energy consumption, (2) the goodput and (3) the transmission delay. Below, we compare ADCA and T-MAC in terms of the three metrics. Note that we use ADCA(i) and T-MAC(i), i=2, 4, 6 to stands for the case of the two protocols with the node degree 2, 4 or 6.

B. Energy consumption

In this subsection, we observe the average energy consumption among the sensor nodes. We record the time in transmitting (tx), receiving (rx), idle listening (idle) and sleeping (slp) modes for each node during the entire simulation. The energy cost function of a sensor node is shown in Eq. 4 and the ratio of energy consumption of the four modes is 17.4: 19.7: 19.7: 0.426.

$$E = E_{tx} + E_{rx} + E_{idle} + E_{slp} \quad (4)$$

Figs. 5 and 6 show the simulation results of ADCA and T-MAC in terms of the energy consumption in the all-to-one and n-to-n scenarios, respectively. As shown in Fig. 5, the energy consumption of ADCA is lower than that of T-MAC. ADCA can be 33.65% better than T-MAC (when traffic=30 packets/sec and node degree=8). This is because ADCA quickly adjusts the duty cycle to adapt to the current traffic and its asynchronous schedule scheme decreases the number of contenders to reduce collisions. As shown in Fig. 6, the energy consumption of ADCA is lower than that of T-MAC in the n-to-n environment. ADCA can conserve up to 36.95% of energy (when traffic=30 packets/sec and node degree=8). Furthermore, the results of ADCA increase smoothly but T-MAC's grow quickly while traffic and network density increase. Therefore, the duty cycle adjustments and collisions will directly affect the transmission delay.

C. Transmission delay

In sleep/wakeup schedule schemes, transmission delay consists of a waiting time and a processing time. The waiting time is the duration that a sender is ready to send the data and waits until the receiver tunes its radio into the receiving mode. The length of the waiting time is dependent on both the cycle period length and the active/sleep ratio. Because we assume all nodes have the same cycle length, the active/sleep ratio becomes the major factor affecting the length of the waiting time. The processing time is the duration that a sender successfully transmits a data packet to the destination. It consists of the back-off time, packet propagating time and ACK waiting time. Figs. 7 and 8 show the results of the average one-hop delay for the two scenarios. The delay time of ADCA and T-MAC grows up as traffic is heavier.

In the all-to-one scenario (Fig. 7), ADCA has shorter delay than T-MAC when traffic is heavy. This is because T-MAC maintains a global synchronous schedule and thus sensor nodes contend the channel in the same period but ADCA maintains asynchronous schedules and the number of contenders is decreased. However, in light traffic cases (e.g., 1 packet/sec), senders in ADCA need to wait for the receivers' active periods to transmit data packets, but senders and receivers in T-MAC wake up simultaneously to handle the traffic. Thus, T-MAC's waiting time is smaller than ADCA's. In the n-to-n scenario (Fig. 8), T-MAC has shorter average hop delays than ADCA when traffic is light. However, as traffic is heavy, ADCA outperforms T-MAC.

D. Goodput

Goodput is defined to be the successful transmission rate from the source to the destination, i.e., the ratio of the number of received packets to the number of generated packets. Collision is the major factor affecting goodput. Figs. 9 and 10 present goodputs for the two scenarios. As traffic or network density grows, goodputs of ADCA and T-MAC both decrease. But, the active periods of the nodes in ADCA is stagger, so the collision probability of ADCA is less than T-MAC and goodput of ADCA is higher than T-MAC's. The goodput of ADCA can be 25.62% higher than T-MAC's in the all-to-one case (see Fig. 9), and 9.7 %, in the n-to-n case (see Fig. 10).

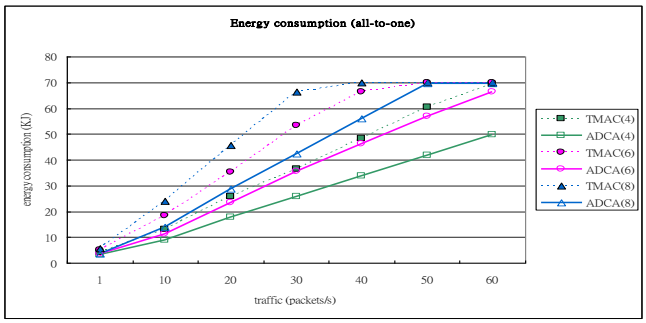

Fig. 5: The average energy consumption in the all-to-one scenario

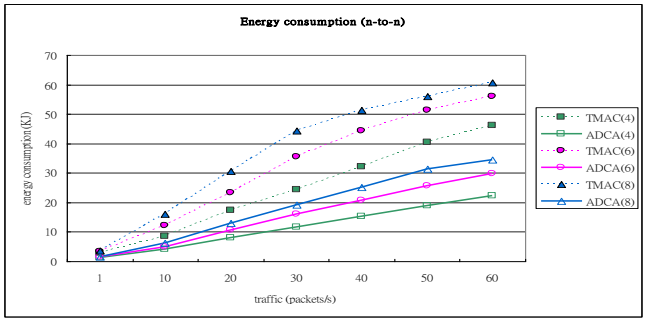

Fig. 6: The average energy consumption in the n-to-n scenario

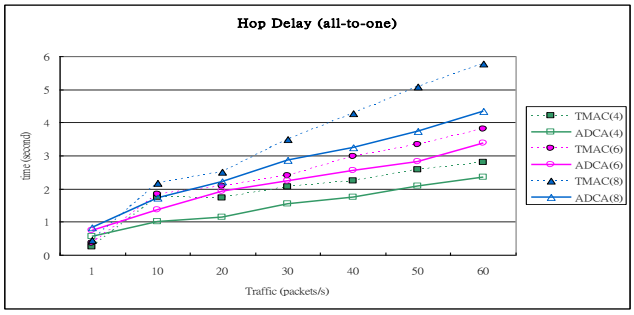

Fig. 7: The average transmission delay in the all-to-one scenario

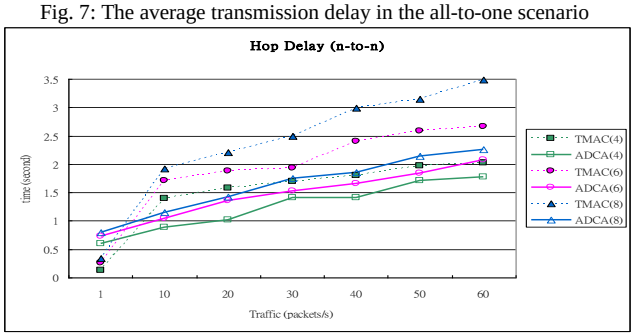

Fig. 8: The average transmission delay in the n-to-n scenario

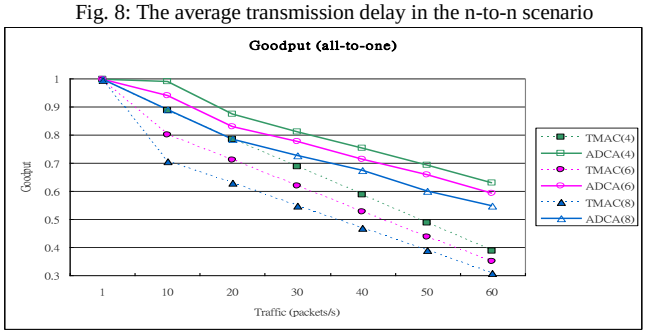

Fig. 9: The average goodputs in the all-to-one scenario

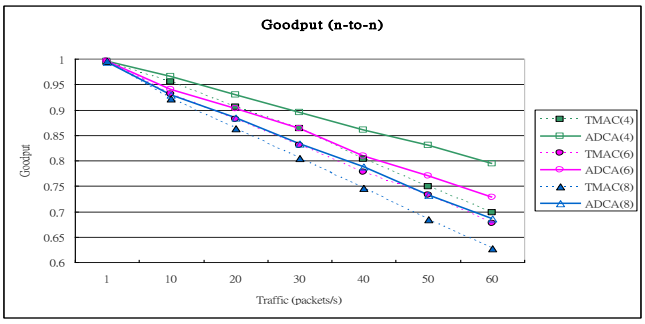

Fig. 10: The average goodputs in the n-to-n scenario

V. CONCLUSION

This paper presents an asynchronous duty-cycle adjustment MAC protocol, called ADCA, for wireless sensor networks. ADCA is an asynchronous sleep/wakeup schedule-based protocol. It needs not synchronize nodes' timers and allows nodes to keep schedules asynchronously. Therefore, the schedules are staggered and collision and overhearing are reduced. A node tunes the radio into sleeping mode as long as possible to save energy for prolonging the network lifetime. However, it adjusts the length of the active period to improve the duty cycle utilization and to reduce the transmission delay. Consequently, ADCA can save a lot of energy without sacrificing goodput and transmission delay. By the simulation results, we can observe that ADCA outperforms T-MAC in terms of energy consumption, transmission delay and goodput.

REFERENCES

[1] A. A. Ahmed, H. Shi, and Y. Shang, "A Survey on Network Protocols for Wireless Sensor Networks," *in Proceedings of International Conference on Information Technology: Research and Education*, pp. 301 – 305, August 2003.

[2] A. El-Hoiydi and J. D. Decotignie, "WiseMAC: An Ultra Low Power MAC Protocol for The Downlink of Infrastructure Wireless Sensor Networks," *in Proceedings of the 9th International Symposium on Computers and Communications*, Vol. 1, pp. 244-251, July 2004.

[3] Chipcon AS, "SmartRF CC2420 PRELIMINARY Datasheet," rev. 1.2, February 2004.

[4] C.E. Perkins, E.M. Belding-Royer, and S. Das, "Ad Hoc On-Demand Distance Vector (AODV) Routing," *IETF Internet draft*, draft-ietf-manetaodv -11.txt, July 2002.

[5] C. Schurgers, V. Tsiatsis and M. B. Srivastava, "STEM: Topology Management for Energy Efficient Sensor Networks," *in Proceedings of the Aerospace Conference*, Vol. 3, pp. 1099-1108, March 2002.

[6] G. Lu, B. Krishnamachari and C. S. Raghavendra, "An Adaptive Energy Efficient and Low-Latency MAC for Data Gathering in Wireless Sensor Networks," *in Proceedings of the 18th IEEE International Parallel and Distributed Processing Symposium*, pp. 224, April 2004.

[7] Heping Wang, Xiaobo Zhang and Ashfag Khokhar, "An Energy-Efficient Low-Latency MAC Protocol for Wireless Sensor Networks," *IEEE Global Telecommunications Conference (GLOBECOM '06)*, pp. 1-5, Nov. 2006.

[8] I. Demirkol, C. Ersoy and F. Alagoz, "MAC Protocols for Wireless Sensor Networks: A Survey," *IEEE Communications Magazine*, Vol. 44, pp. 115 – 121, April 2006.

[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*, pp. 95-107, November 2004.

[10] J.M. Rabaey, M.J. Ammer, J.L. da Silva, D. Patel, and S. Roundry,"PicoRadio Supports Ad Hoc Ultra-Low Power Wireless Networking," Computer, vol. 33, pp.42-48, July 2000.

[11] LAN MAN Standards Committee of the IEEE Computer Society, editor. *IEEE Std 802.11-1997, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*. IEEE, Nov. 1997.

[12] M. A. M. Vieira, C. N. Coelho Jr., D. C. da Silva Jr., and J.M. da Mata, "Survey on Wireless Sensor Network devices," *in Proceedings of IEEE International Conference on Emerging Technologies and Factory Automation*, Vol. 1, pp. 537 – 544, September 2003.

[13] Shih-Hsien Yang, Hung-Wei Tseng, Wu E.H.-K., Gen-Huey Chen, "Utilization based duty cycle tuning MAC protocol for wireless sensor networks," *IEEE Global Telecommunications Conference (GLOBECOM '05)*, Volume 6, pp. 5, 28 Nov.-2 Dec. 2005.

[14] T. V. 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* ,pp. 171-180 , November 2003.

[15] T. Zheng, S. Radhakrishnan and V. Sarangan, "PMAC: An Adaptive Energy Efficient MAC Protocol for Wireless Sensor Networks," *in Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium*, pp. 8, April 2005.

[16] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves, "Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks," *In the Journal of Wireless Networks*, Vol. 12, pp. 63-78, Feb. 2006.

[17] W. Ye, J. Heidemann and D. Estrin, "Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks," *IEEE/ACM Transactions on Networking*, Vol. 12, pp. 493-506, June 2004.

[18] Xiaolei Shi; Stromberg, G., "SyncWUF: An Ultra Low-Power MAC Protocol for Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, Volume 6, pp. 115-125, Jan. 2007.

[19] Zhihui Chen and Ashfaq Khokhar, "Self Organization and Energy Efficient TDMA MAC Protocol by Wake Up For Wireless Sensor Networks," *in The Proceedings of the First IEEE Communication Society Conference on Sensor and Ad Hoc Communications and Networks (SECON'04)*, pp. 335-341, Oct. 2004.

[20] http://www.isi.edu/nsnam/ns/

[21] http://www.tinyos.net/