Low-Energy, Adaptive, and Distributed MAC Protocol for Wireless Sensor-Actuator Networks

Muhammad Farukh Munir and Fethi Filali
Institut Eurécom, Department of Mobile Communications, Sophia Antipolis, France
Email: {munir,filali}@eurecom.fr

***Abstract*— Sensor-actuator networks are often limited in battery capacity and processing power. Therefore, it is exigent to develop solutions that are both energy and delay efficient. In this paper, we propose a low-energy and delay-sensitive TDMA based MAC for wireless sensor-actuator networks (WSANs). These networks are organized into *clusters* and each cluster is managed by a single actuator. To avoid inter-cluster downlink *interference*, we employ an *easy* use of CDMA codes. The actuators schedule sensors to reduce the total network energy consumption per *successful* transmission. We identify the advantages of our proposal over existing TDMA-MAC schemes for cluster-based sensor networks. Our simulation results demonstrate that the proposed MAC greatly improves the WSAN lifetime and achieves a good trade-off between the packet delay and sensor energy consumption.**
Keywords: WSANs, TDMA, CDMA, wakeup protocols, energy efficiency, delay guarantee.

I. Introduction and related Literature

SANETs[1] (Sensor-Actuator Networks) shown in Fig. 1, are appealing to researchers due to their wide spread of application potential, ranging from densely deployed habitat-monitoring setup to the real-time security applications. Sensor nodes are small, cheap devices with reduced processing and communication capabilities. They usually gather information about the physical world. Whereas, the actuator[2] nodes are expensive, resource rich, and higher communication capability devices.

The multi-actuator architecture raises many interesting issues such as cluster formation, cluster-based sensor organization, network management and task allocation among the actuators. In this paper, we only focus on the issues of network management within the clusters, particularly energy-aware medium access control (MAC) layer protocol and inter-cluster interference issues. The energy efficiency at the MAC layer has recently received attention, especially with the increasing interest in the applications of unattended sensor networks. The S-MAC [3] enables low-duty-cycle operation in a multi-hop sensor network. Nodes form virtual clusters based on *fixed common sleep schedules* to reduce control overhead and enable traffic-adaptive wake-up. T-MAC [5] extends S-MAC by adjusting the length of time sensors stay awake between sleep intervals based on the communication of neighboring sensors. To achieve low power operation, B-MAC [6] employs an *adaptive preamble sampling scheme* to reduce duty cycle and minimize idle listening which is a basic source of energy drain. Whereas, the Z-MAC proposal [7] combines the strengths of TDMA and CSMA while offsetting their weaknesses by switching the MAC to CSMA and TDMA at low and high contention periods, respectively. The performance of Z-MAC falls even below B-MAC in the case of low contention, so it is a more suited protocol for medium to high data rate applications.

In [8], the authors presented two scheduling schemes (breadth-first and depth-first assignment) for a *cluster based* sensor network. The proposed TDMA-MAC is shown to perform well in terms of energy-efficiency and end-to-end delay depending on the choice of scheduling scheme. The gateway nodes transmit the schedule in their cluster using larger transmission power. This introduces a new problem of schedule *interference* among neighboring clusters and is not discussed in the paper. PEDAMACS [4] proposal for sensor networks has utilized the presence of a powerful node called *access-point* (AP) among sensors which takes the transmission load from the constrained sensors and is further responsible for the reliable delivery of sensor data toward the sinks. In case of multiple APs, the neighboring APs should not transmit their coordination packets at the same time to avoid *inter-cluster interference.* The APs should take into account the sensors that are outside their largest range while generating the schedule. The power levels of the APs are adjusted so that the schedule can reach all the sensors in the cluster. If all the sensors cannot be reached by the schedule, they can still be scheduled at the cost of an increased synchronization overhead apart from the increased delay and energy consumption.

The introduction of actuators [2] in the network pose a *hard delay constraint* to timely actuate the required actions. Hence, as the base of the communication stack, the MAC layer should support real-time guarantees or QoS features. We propose LEAD-MAC (low-energy, adaptive, and distributed-MAC) protocol for SANETs, which ensures to minimize the end-to-end delay, improves throughput, and conserves sensor energy by using an adaptive wakeup scheme. The actuators compute a hybrid (involves both depth-first and breadth-first scheduling) collision-free TDMA-schedule for the sensors in their local-cluster and avoid interference using CDMA codes.

In Section II, we detail the assumptions for the design of LEAD-MAC. Section III describes the network model for SANETs. The detailed design of LEAD-MAC is presented in Section IV. A delay-energy analysis of the Wakeup protocol is

[1]The terms SANET and WSAN can be interchangeably used if SANETs are wireless.

[2]In design, actuators/access-points/gateways can all be thought of similar devices with higher processing and communication capabilities. Whereas, in practice, the actuators need to have some extra actuation capabilities.

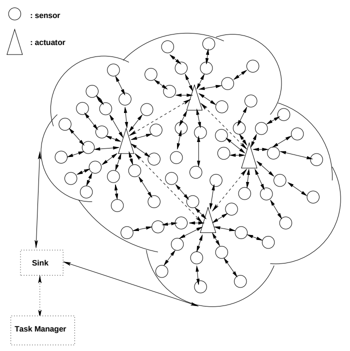

Fig. 1. Architecture of SANETs

given in Section V. In Section VI, we present the simulation results. Section VII concludes the paper and outlines the future directions.

II. Assumptions

The assumptions under which we have designed the LEAD-MAC protocol are as follows:

1) We consider a wireless ad hoc network that consists of a large number of sensor nodes along with a few uniformly distributed actuator nodes. Sensors do the application dependent sensing and transmit their sensed data towards their optimal[3] actuators.
2) Both sensors and actuators are static, capable of adjusting their transmission power, and a link between any two nodes is bidirectional.
3) The actuators can reach all the sensors in their local cluster in one-hop using maximum transmit power.

III. Network Model

Consider a static wireless sensor network with n sensor nodes and m actuator[4] nodes as shown in Fig. 1. Given is an $(n+m)\times(n+m)$ neighborhood relation matrix R that indicates the node pairs for which direct communication is possible. We will assume that R is a symmetric matrix, i.e., if node i can transmit to node j, then j can also transmit to node i. For such node pairs, the $(i,j)^{th}$entry of the matrix R is unity, i.e., $R_{ij}=1$ if node i and j can communicate with each other; we will set $R_{ij}=0$ if nodes i and j can not communicate. For any node i, we define $N_i=\{j : R_{i,j}=1\}$. Which is the set of neighboring nodes of node i. Similarly, a set of interference nodes (cannot be reached by one-hop) for node i (from where the transmissions can (possibly, not necessarily) be heard at node i), and is defined as

$S_i=\{K\notin N_i\cup\{i\} : R_{k,j}=1\ forsome\ j\in N_i\}$

Note that S_i does not include any of the first-hop neighbors of node i.

The topology of the network is represented by a graph $G=(V,E)$, in which V is the set of nodes (both sensors and actuators). The edges $E\in V\times V$ are such that $(i,j)\in E$ if nodes i and j can transmit to each other.

[3]Optimal refers to the outcome of a cost-function, e.g., min-hop or min-delay routing.

[4]Conceptually, we can assume that this actuator is also a sensor node, which has 0 sampling rate.

Power Consumption Model: For a sensor node, the energy consumption due to wireless communication (mostly idle times) is considered the dominant source in power consumption. If power consumed to receive a single multi-hop packet (for design, we assume all packets to be of same length) is given by P_{rx} (in j/pkt), then the power consumed P_r^i (in j/s) by a sensor node i for receiving is

$$P_r^i=P_{rx}\sum_{j\in N_i}\alpha_{j,i} \tag{1}$$

where $\alpha_{j,i}$ is the rate (pkt/s) at which node j is transmitting packets toward node i.

If the power consumed to sense and sample a packet is P_{sense}(in j/pkt), then the power consumed P_s^i (in j/s) by a sensor node i in sampling packets is

$$P_s^i=P_{sense}\lambda_i \tag{2}$$

where λ_i is the rate (pkt/s) at which node i performs environmental sensing.

If the power consumed to send a packet is given by P_{tx}, then the power consumed P_t^i (in j/s) by a sensor node i in transmitting its data (both locally originated and forwarded packets) is

$$P_t^i=P_{tx}\sum_{j\in N_i}\alpha_{i,j} \tag{3}$$

When the packets arrive from S_i due to interference, the power loss P_{in}^i (in j/s) at node i is

$$P_{in}^i=P_{rx}\sum_{j\in S_i}\alpha_j \tag{4}$$

where α_j in (pkt/s) is the total rate at which node j is transmitting: $\alpha_j=\sum_{k\in N_j}\alpha_{j,k}$

If node i is neither serving its forwarding queue nor sampling a new packet, it is in idle state. If the power consumed in idle state is given by P_{idle}, then the power consumption P_{id}^i (in j/s) by a sensor node i is given by

$$P_{id}^i=P_{idle}\left(1-\sum_{j\in N_i}\frac{\alpha_{i,j}}{C}-\sum_{j\in N_i}\frac{\alpha_{j,i}}{C}\right) \tag{5}$$

where C is the transmission capacity in pkt/s.

In this paper, the network Lifetime $T_{lifetime}^{network}$ is defined as the time spanned by the network before first *node death* as a result of energy outage.

$$T_{lifetime}^{network}=\min_i T_{life}^i \tag{6}$$

where the lifetime T_{life}^i of a sensor node i having battery capacity E_i is given by (7).

The lifetime T_{life}^i can also be maximized by controlling the flow coming into a node and its service rate using an adaptive routing protocol. But in this paper, we do not discuss any routing layer details. For the considered model, we try to optimize the system performance using a TDMA-MAC protocol by minimizing the *awake periods* and power loss due to *interference*.

$$T^i_{life} = \frac{E_i}{\left(P_{rx}\left(\sum_{j\in N_i}\alpha_{j,i}+\sum_{k\in S_i}\alpha_k\right)+P_{tx}\sum_{j\in N_i}\alpha_{i,j}+P_{sense}\lambda_i+P_{idle}\left(1-\sum_{j\in N_i}\frac{\alpha_{i,j}}{C}-\sum_{j\in N_i}\frac{\alpha_{j,i}}{C}\right)\right)} \tag{7}$$

IV. Design of LEAD-MAC

LEAD-MAC has three operational phases: (i) network learning phase, (ii) scheduling phase, and (iii) adjustment phase. The following discussion covers the different protocol phases in detail.

A. Network Learning Phase

A sensor node finds an optimal actuator using the proposed ADP (Actuator Discovery Protocol, a controlled flooding mechanism [1]), during the initial deployment phase. The sensors start the learning phase by transmitting a one hop broadcast *actuator-search_request*, using their lowest transmit power. When a broadcast reaches an actuator, it is replied with the actuator identity. A random access scheme is used in the topology learning phase, because the sensors do not yet have a transmission schedule. The scheme is designed so that, at the end of this phase, almost all nodes are attached (based on the outcome of an objective function) to an actuator and correctly determine their neighbors and interferers with high probability. We adopt a carrier sense multiple access (CSMA) mechanism similar to 802.11 [9]. The sensors listen for a random time before transmitting, and transmit if the channel is idle. A random delay is added before carrier sensing to further reduce collisions. However, because a collision will lead to incomplete cluster information at the actuators, the CSMA scheme itself cannot guarantee that an actuator will receive the full cluster information. Therefore, we proposed to include an implicit acknowledgment from the actuator, which occurs when a sensor transmits a packet to join a particular cluster.

B. Scheduling Phase

The actuator explicitly schedules all the sensors, based on its knowledge of the cluster. An actuator schedules the sensors in the depth-first order for end-to-end routes, and in a breadth-first order for any given parent node i, to capture forwarded data from all of its downlink sensors. We don't provide the scheduling algorithm owing to the lack of space. The scheduling frame duration T is divided into slots (a single slot-duration depends on the packet size, available transmission rate, and is typically application dependent). A slot extends the packet duration by a guard interval to compensate for synchronization errors. At the beginning of this phase, an actuator broadcasts the *scheduling packet* using maximum transmit power. Since the actuator reaches all the sensors at the same time, the error in synchronization from the delay between time-stamping and sending the packet at the transmitter is eliminated. Since the range of an actuator is on the order of hundreds of meters, the propagation delay is also negligible (few μsec). Based on the assumption that all the nodes run the same software, all of them will time-stamp the packet at the same time. Therefore, the only error of synchronization in this application comes from clock skew, the difference in the clock tick rates of the nodes. Typical clock drifts of a sensor node in 1sec is $10\mu sec$ [6]. If the packet generation period of each node is around 30sec, the maximum clock drift will be 0.3msec compared to approx. 20msec (the duration of the packet transmission of a packet of 50 byte at 50kbps). The total time-frame duration of this schedule is given by T, which depends on the number of sensors in the cluster. Therefore, the frame duration T is different for every cluster in the network. The minimum duration for a sensor to stay awake T_a is $T_a = T_{rx} + T_{tx} + T_g$, where T_{rx} is the time required to receive a packet, T_{tx} is the time required to transmit one packet to the parent node, and T_g is the guard interval for synchronization errors. The interval T_g is assumed to be a small percentage of the total slot duration. The maximum duration for a sensor to stay awake depends on its sub-tree and can be calculated as a multiple of T_a depending on the application, e.g., if the application allows for *data aggregation*: a sensor can receive forwarded data from its sub-tree, aggregate its own packet and transmit the resultant packet requiring only one *time slot*.

The first transmitted packet to contain the CDMA code is the *collision-free* TDMA schedule by each actuator in the network, so that the sensors receive the schedule from their attached actuator only once. The schedule packet contains a *current-time* field in order for all the sensors in one cluster to synchronize to a common clock before starting the transmissions and a *next-time* field, where all the sensors wakeup once in order to resynchronize to the common clock. There can be multiple data transmissions by the *non-radio interfering* sensors in the cluster at the same time.

C. Adjustment Phase

If a new node is added to the network or a *link level* failure is detected in the network, a sensor will try to attach itself by transmitting a *one-hop* broadcast request in its neighborhood. All the sensors in the network wakeup at *next-time* to resynchronize to the network. At this time, a sensor which receives the *actuator-search* broadcast replies to the sensor with its actuator *id* and *cost* to reach the actuator. Upon receiving the reply to its broadcast (there can be multiple replies), a sensor decides its optimal actuator and transmits an *attachment* request to the actuator. The new sensor is added to the transmission schedule and also acquires the same CDMA code as its *cluster*.

V. LEAD-Wakeup Protocol

The main idea of LEAD-Wakeup protocol is to extend the scheduling for event-driven sensing applications, where the slots assigned to the nodes do not have to be used. According to the opted scheduling scheme, all the nodes on one routing path remain active only for a small duration T_a to check the possibility of arrival of forwarding data.

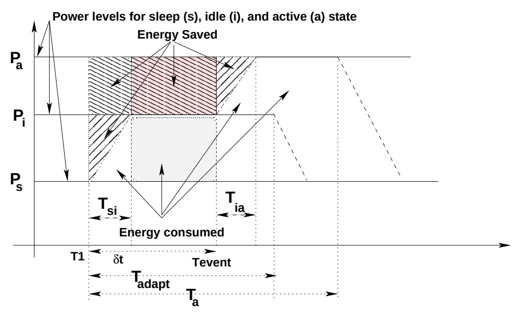

Fig. 2. Energy Savings through adaptive duty cycle

A. Adaptivity to Network Conditions

A sensor wakes up at the scheduled time to see if it has any new sensed data in its transmit queue. If it has no data to transmit and also, no data arrives from its children sensors during a defined interval (which is equivalent to the reception time for one packet and a guard interval), it immediately goes back to the sleep mode and saves considerable amount of energy (*adaptive duty cycle*). The duration of this adaptive validation period is atleast equivalent to $T_{adapt} = T_{rx} + T_g$.

B. Analysis of LEAD Wakeup protocol

In [8], the authors have shown that the depth-first scheduling works better than the breadth-first scheme for end-to-end delay, throughput and forwarding queue size at the sensors, but fails to perform well in energy consumption compared to the breadth-first scheduling. In this work, we will show that the hybrid scheduling scheme with an adaptive duty cycle achieves a good trade-off between the sensor energy consumption and end-to-end delay for sensor-actuator applications.

Energy Consumption: For a wireless sensor, typical states are "active", "idle", and "sleep". The energy saved for a sensor by not sending the sensors directly to *active* state is shown in Fig. 2. The only form of overhead seen in this power management is the time spent in settling from one state to another and is given by

$$E_{overhead} = T_{si}.\left(\frac{P_i - P_s}{2}\right) + T_{ia}.\left(\frac{P_a - P_i}{2}\right) \quad (8)$$

where T_{si} and T_{ia} is the time required to change the state from sleep-to-idle and idle-to-awake, respectively. The gain in energy using such an adaptive power management can be seen as

$$\begin{array}{c} E_{saved} = (\delta t - T_{si}).(P_a - P_i) + \\ T_{si}.\left(P_a - \left(\frac{P_i+P_s}{2}\right)\right) + T_{ia}\left(\frac{P_a-P_i}{2}\right) \end{array} \quad (9)$$

where $\delta t = T_{event} - T_1$, T_1= start of the adaptive awake period T_{adapt} and T_{event} = time of arrival of an event (transmission or reception).

If we see the case with Intel strong ARM (Table I), it is sensing the environment in S_1. At scheduled time, it will change its sleep state from S_1 to S_2. The sensor node will stay in this state until the arrival of event for T_{adapt}, if it do not receive a data packet, and itself has no packet to transmit then it will go back to sleep. Otherwise, it will jump to S_4 to transmit a packet. The minimum energy consumed by a sensor during one time frame T is given by

$$MinE_i = E_i^{s_1}.(T - T_{adapt}) + E_i^{s_2}.T_{adapt} \quad (10)$$

Similarly, maximum energy consumed by a sensor during T is given by

$$MaxE_i = E_i^{s_1}.(T - T_a) + E_i^{s_2}.\delta t + E_i^{s_4}.(T_a - \delta t) \quad (11)$$

The sensors only wakeup when a transmission or reception is expected, therefore, we save the *expected* energy drain due to interference from two-hop neighbors. Due to an adaptive sleep schedule, a sensor saves energy by configuring its transmitter state to sleep. Therefore (7) becomes

$$T^i_{life} = \frac{E_i}{\left(P_{rx}\sum_{j\in N_i}\alpha_{j,i} + P_{tx}\sum_{j\in N_i}\alpha_{i,j} + P_{sense}\lambda_i\right)} \quad (12)$$

Observed Latency: The average latency seen by a packet from $node_i$ is

$$delay_i = \sum_K (\delta_{s_2-s_4} + T_{data}) \quad (13)$$

where T_{data} is the time required to actually transmit a packet and K is the number of hops toward the actuator of $node_i$. And the worst case latency seen by a packet from $node_i$

$$delay_i = \sum_K (\delta_{s_2-s_4} + T_{data}) + T \quad (14)$$

which can happen if an event arrival in the current awake duration T_a do not reach the actuator due to long paths.

VI. Simulation Results

The metrics which are often used to compare sensor/actuator network MAC-protocols are energy efficiency, delay analysis and network lifetime. LEAD-MAC aims at performing well in terms of all these metrics. The purpose of this section is to demonstrate the effectiveness of LEAD-MAC. A delay and energy consumption analysis is presented for increasing network size. The simulations environment is ns2 [10], a discrete event simulator. Shortest path routing is used in the simulations. The average depth of the resulting routing trees is 4.4, 5.2, and 7 for 20, 30, and 60 sensors per cluster, respectively; correspondingly the average number of neighbors is 4.6, 5.0, and 5.5. The data packet length is 37 bytes. Sensors generate one packet in $30sec$, with a sampling rate of $1Hz$ and transmission rate is $50kbps$. The remaining simulation parameters are listed in Table II and Table III. The sensors in the network are always connected to the actuators. A comparison with the analytical model of PEDAMACS is presented for lifetime analysis. We compare LEAD-MAC only with PEDAMACS, because this work has already been shown to perform better compared to other listed MAC proposals for sensor networks. The evaluated lifetime for LEAD-MAC is given in Fig. 3. The lifetime decreases by increasing the

TABLE I

USEFUL STATES FOR THE SENSOR NODE WITH ASSOCIATED POWER CONSUMPTION AND DELAY (TIME TO REACH S_4 FROM ANY GIVEN STATE)

Operating State	Strong ARM	Memory	ADC	Radio	Power Consumption	Delay (ms)	Notation Used
S_0	Sleep	sleep	Off	Off	50 (μW)	50	E^{s0}_{node}
S_1	Sleep	Sleep	On	Off	5 (mW)	20	E^{s1}_{node}
S_2	Sleep	Sleep	On	Rx	10 (mW)	15	E^{s2}_{node}
S_3	Idle	Sleep	On	Rx	100 (mW)	5	E^{s3}_{node}
S_4	Active	Active	On	Tx, Rx	400 (mW)	NA	E^{s4}_{node}

TABLE II

SIMULATION PARAMETERS. THE SIMULATION AREA IS SET SUCH THAT ATLEAT TWO SENSORS ARE IN EACH OTHERS TRANSMISSION RANGE.

Sensors	Area (m^2)	Actuators/Base Stations
100	500 * 500	2
150	600 *600	3
⋮	⋮	⋮
400	970 * 970	8

TABLE III

POWER CONSUMPTION IN DISCRETE OPERATION STATES FOR *Mica* MOTES

Operation	Power Consumption
To transmit one packet	0.92 mj
To receive one packet	0.69 mj
Listening to channel	29.71 mj/sec
Operating radio in sleep mode	15 $\mu j/sec$
To sample a packet	1.5 $\mu j/sample$

number of sensors in a cluster due to longer data paths and hence more load on the sensors closer to the actuator. The maximum delay observed by a network can be seen in Fig. 4. The end-to-end delay is less due to the depth-first scheduling policy of end-to-end routes in the hybrid-schedule. Finally, we present a comparison for the energy consumption in Fig. 5, where sensors consumes less energy due to an adaptive duty cycle and longer sleep periods.

VII. CONCLUSIONS AND FUTURE WORK

The LEAD-MAC protocol is shown to perform better in determining network performance in terms of sensor energy consumption per successful transmission, end-to-end delay observed by the sensor transmissions, and network lifetime as a result of *distributed learning* during initial deployment and a delay-energy aware *hybrid* scheduling policy. The use of CDMA codes delimits the interference among neighboring clusters and the need of a separate scheduling among neighboring actuators to transmit the *schedule*.

Our future work will consider the development of LEAD-MAC in a *TinyOS [11]* based simulator known as *TOSSIM*. We will develop a routing layer protocol that should operate on top of LEAD-MAC to further optimize network *lifetime* by considering *delay-energy* issues at *routing layer.*

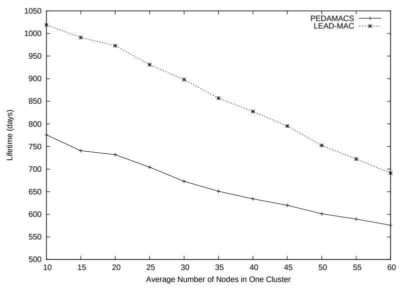

Fig. 3. Average lifetime of a cluster → increasing # of nodes

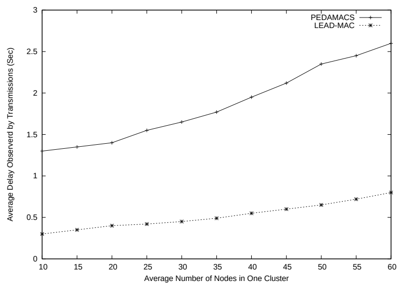

Fig. 4. Average delay in a cluster → increasing # of nodes

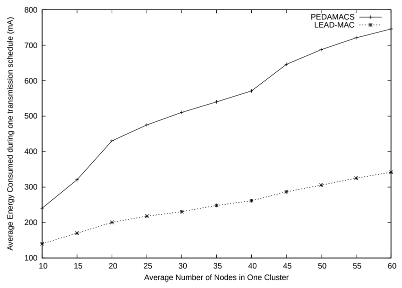

Fig. 5. Average energy consumption in a cluster → increasing # of nodes

REFERENCES

[1] M. Farukh Munir and F. Filali, An energy aware actuator discovery protocol for SANETs, Eurécom Research Report, RR-06-158, http://www.eurecom.fr/~munir/research.htm.

[2] I.F. Akyildiz and I. H. Kasimoglu, Wireless sensor and actor networks: research challenges, Elsevier jour. Ad-hoc nets., Vol. 2, No. 4, pp 351-367, October 2004.

[3] W. Ye, J. Heidemann and D. Estrin, Medium access control with coordinated adaptive sleeping for WSNs, IEEE/ACM Trans. Netw., Vol. 12, No. 3, pp. 493-506, 2004.

[4] S. Coleri and P. Varaiya, PEDAMACS: Power effi cient and delay aware medium access protocol for sensor networks, IEEE Trans. Mob. Comp., Vol. 5, No. 7, pp. 920-930, 2006.

[5] T. V. Dam, and K. Langendoen, "An adaptive energy effi cient MAC protocol for WSNs", in Proc. of ACM Sensys, November 2003.

[6] J. Polastre, J. Hill and D. Culler, Versatile low power media access for wireless sensor networks, in Proc. of ACM Sensys, November 2004.

[7] I Rhee, A. Warrier, M. Aia and J. Min, Z-MAC: a hybrid MAC for wireless sensor networks, in Proc. of ACM Sensys, November 2005.

[8] M. Younis, M. Youssef and K. Arisha, Energy-aware management for cluster-based sensor networks, Comp. Nets., Vol. 43, No. 5, pp. 649-668, 2003.

[9] LAN-MAN Standards committee of the IEEE Computer Society, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifi cation," IEEE Std 802.11-1997, 1997.

[10] The Network Simulator - ns (version 2.27). http://www.isi.edu/nsnam/ns/.)

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