CMAC – A Multi-Channel Energy Efficient MAC for Wireless Sensor Networks

Kaushik R. Chowdhury, Nagesh Nandiraju, Dave Cavalcanti, and Dharma P. Agrawal
OBR Center for Distributed and Mobile Computing
Department of ECECS, University of Cincinnati – Cincinnati, OH 45221-0030
(kaushir, nandirns, cavalcdt, dpa)@ececs.uc.edu

Abstract–**This paper presents CMAC, a fully desynchronized MAC protocol that is designed to exploit the existing multi-channel support in sensor nodes. The hardware requirements of our protocol are minimal, requiring a single half-duplex transceiver and a low-power wake-up radio. CMAC takes into account the fundamental energy constraint in sensor nodes by placing them in a default sleep mode and waking them up only when necessary. As a contrast to other dual radio wake-up schemes, our protocol focuses on how communication and its preceding control message exchange mechanism can be undertaken in a multi-channel scenario without assuming a separate control channel. CMAC enables spatial channel re-use, nearly collision free communication, and addresses the deafness problem without incurring a tradeoff in fairness or latency. When compared with a recent MAC protocol SMAC, results show that CMAC obtains nearly 200% reduction in energy consumption, significantly improved throughput, and end-to-end delay values that are 50-150% better than SMAC for our simulated topologies.**

Keywords: Energy efficiency, MAC, multi-channel, wireless sensor networks.

I. INTRODUCTION

Wireless sensor nodes are low power, battery operated devices with limited computation and transmission ability. With improvements in hardware, wireless communication between these nodes need not be limited to a single common channel: Berkeley's third generation Mica2 Mote has an 868/916 MHz multi-channel transceiver [1]. In Rockwell's WINS nodes, the radio operates on one of 40 channels in the ISM frequency band, selectable by the controller [2].

The main concern in wireless sensor networks is energy loss due to collisions and re-transmissions, overhearing, control packet overhead, and idle listening. CMAC overcomes these issues by (1) supporting a default sleep mode in which sensors switch off their radio during idle times and (2) enabling multi-channel communications with minimal hardware requirements of a low power wake-up radio (LR), in the lines of the Berkeley pico-radio [6], and a main half duplex transceiver (MR). The LR radio used for wakeup can only emit a short train of pulses and thus cannot be used as a second interface to send data. It is always used to monitor a node's default channel while the MR is placed in the sleep mode thus conserving energy. The LR plays two roles: (1) when a node wishes to transmit, the receiver is woken up through a series of pulses and (2) channel negotiation is undertaken before the MR is switched on. Data communication is carried out by the MR only. In the absence of complex signal processing hardware [14], we allow the LR to be capable of merely discerning the presence or absence of 6-8 pulses during the control message exchange phase.

Before CMAC can be set in operation, each node must be assigned a channel that does not overlap in its 2-hop range, which we shall henceforth refer to as a node's *default* channel. This problem is well known in the literature [4 and the references therein] and we call this the 2-hop coloring problem. The adoption of a multichannel scenario enables communication between more than one node to occur simultaneously and in a collision free manner. Through our suggested handshaking mechanism, a node can issue a time duration for which it will be busy even when a reception is in progress. The control pulses are devoid of node addresses and senders are identified by channels alone further reducing the control overhead.

Rest of this paper is organized as follows. Section II discusses related work and the design of the CMAC protocol is presented in Section III. Section IV gives a detailed discussion on our protocol. In Section V we provide simulation results and performance evaluation. Finally Section V concludes with directions for future research.

II. RELATED WORK

Early MAC protocols for wireless sensor networks were TDMA based, enabled collision free communication and were simple to set up and maintain. Local synchronization was required for alternating between '*listen*' and '*sleep*' cycles. TDMA-W [5] and the work presented in [8] require non-repetition of the time slots over a 2-hop range, though the latter allows nodes to leave and join the network. S-MAC [10] and protocols inspired by it [11, 12], follow the listen and sleep cycles as described earlier, but contend for the channel during the listen cycles. Pair wise synchronization is usually sufficient and they trade fairness and latency for energy savings. While the above work assume a single radio, the idea of using an additional wake-up radio for sensor networks is not novel and is discussed in [14, 15, 16, 17]. According to the authors in [14], this low power radio may only use 1 μW of energy compared to the high dissipation of the MR (order of mW). In this work, however, the authors focus more on the

1-4244-0270-0/06/$20.00 (c)2006 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2006 proceedings.

channel allocation problem and the MAC protocol is left for future research. STEM [16,17] and RATE-EST [15] are similar in principle in which the authors assume the presence of two channels: primary channel for sending data and a wakeup channel used for the wakeup signals. Both these approaches use a '*busy tone*' instead of encoded data for the wakeup signal. Energy savings are obtained by allowing the radio for data communication to enter the low power sleep mode. Our work differs from all these approaches in the respect that we assume a multichannel scenario and in-band signaling. Furthermore, our protocol allows for listening and replying to control messages even when a reception of a data packet is in progress at a sensor node, thus taking a step towards solving the deafness problem in wireless networks.

III. CMAC SCHEME

CMAC has been designed to ensure maximum possible energy conservation without any compromise in fairness or latency. Recognizing that idle listening consists of 90% of power consumption [3], CMAC puts nodes in the sleep mode whenever the MR is idle. Thus, the sensor enjoys optimally minimum power usage when it is powered up only during transmission or reception. We outline the assumptions of our protocol below:

1. A single half-duplex transceiver (MR), capable of being dynamically tuned to any of the pre-decided set of channels, is present. The MR transmits at a constant power level in the selected band but can be switched off during sleep time. All data transmission/reception is handled by the MR.
2. A low-power radio, (LR) is present which can only emit and receive a short train of wake-up pulses. We do not assume time synchronization amongst the nodes of the network.
3. Nodes have been allocated channels that overlap at 3 hops or more through the execution of a suitable channel assignment algorithm.

CMAC relies on three types of control messages for its operation: Request (REQ), Confirm (CON) and Wait (WAIT). All these three messages are essentially short pulse trains and are sent and received through the LR. The MR is in the default sleep state to conserve energy, delegating the task of channel monitoring and negotiation to the LR. Once this negotiation is completed and the receiver is ready, the MR is used to transmit the actual data packet. Our protocol adopts a novel, dual mode communication architecture. During the negotiation phase, the sender tunes its LR to the channel of the receiver. Once the receiver is ready to accept the packet, CMAC follows a transmitter-oriented communication model in which the receiver tunes its MR to the channel of the sender for data packet exchange. The details of the protocol are given below with reference made to the Figure 1 for clarity of representation:

A. Control Message types & Channel Negotiation

The control messages are a sequence of $k+1$ pulses, where 2^k is the total number of assigned channels. The LR is always tuned to a node's default channel, unless the MAC receives a data packet for transmission from the higher layer.

REQ: When a packet has to be transmitted, the LR of the sender is tuned to the receiver's channel, (R). It monitors R for DIFS time period to avoid collision with ongoing transmissions. In case the channel of the intended receiver is sensed busy, the sender waits till the channel is available in a manner similar to the operation of 802.11. If R is found unused at DIFS timer expiry, the sender transmits an REQ in R, after a random delay decided by the backoff interval. This REQ comprises of a set of pulses that identifies the sender's default channel (S). The identification is achieved through a unique mapping of pulse patterns and the associated channel. As an example, the channel S may be represented by a pulse pattern *11001*. As allocated channels are unique within 2-hop range, the sender is identified through the choice of S alone. The response by the receiver is dependent on whether it is idle or busy receiving. As an example, consider Figure 1, in which nodes 4 and 5 are assigned channels *red* and *blue,* respectively. Node 4 first tunes its LR to *blue* and sends the REQ coded for identifying its default channel, i.e., *red*.

CON: If the receiver is idle, it responds to the REQ with a CON. This is sent by the receiver's LR in R after SIFS duration and consists of a sequence of pulses with the encoded representation of S. We reason as follows: a receiver can get multiple REQs from senders, each of which is tuned to R and awaiting a reply. The encoding of S helps to distinguish the node that has won the contention. From Figure 1, node 5 accepts the request by node 4 and sends a CON, coded with *red*, node 4's default channel.

WAIT: The presence of a single pulse in the $k+1^{st}$ position differentiates this message from a CON. In case a node currently receiving a data packet on its MR hears a REQ on its LR, it informs the interested sender of the time at which it will complete its current transaction. We accomplish this through WAIT. This message contains k pulses and when multiplied with a constant ψ, gives the approximate time for the next possible reception. In the case of multiple REQs from different senders, each of which resulted in a WAIT, it is necessary to identify the node which received the first WAIT. This node has transmission priority when the current data packet reception is completed. The provision of an additional pulse indicates whether the node receiving WAIT was the first and accordingly adopts its WAIT policy described in Section IV -E. Thus we see that the LR of the sender is tuned to the channel of the receiver at the start of the DIFS interval and remains so till it gets a CON/WAIT or in their absence, suffers a timeout. After this, the LR is reset to the default channel, free to monitor incoming requests. From Figure 1, when node 8 transmits a REQ to node 5, which is in turn, engaged in data reception, it gets a WAIT in reply. This gives the time for which node 5 will be busy.

B. Data Transmission/Reception

If the channel negotiation described earlier results in receipt of a CON by the sender, its MR is switched *on* and data transmission can begin. After the transmission of the CON, the receiver immediately switches its MR in the active state and tunes it to the channel described in the REQ (S).

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2006 proceedings.

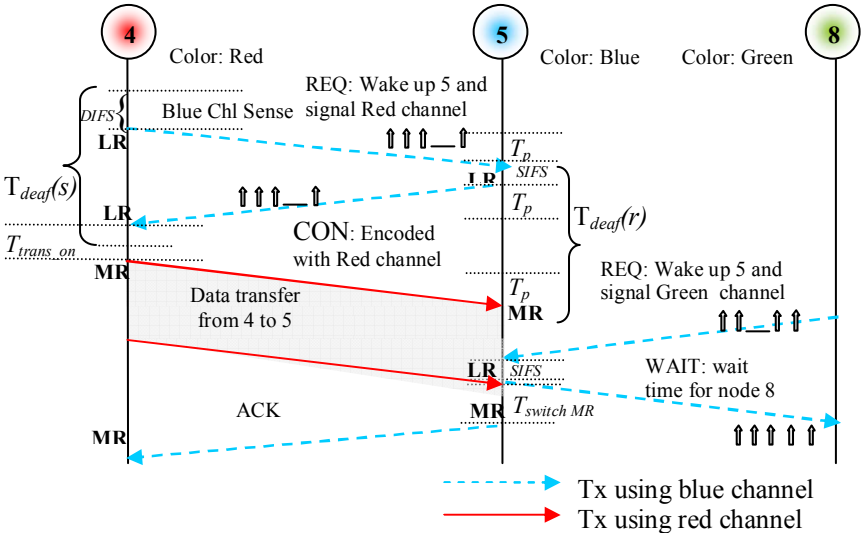
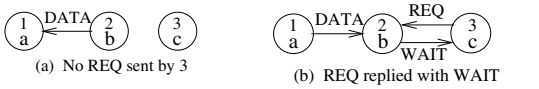


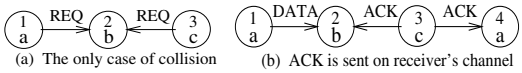
Figure 1. CMAC message exchanges.

Table-1. Specifications of the MR and LR		
	MR (mW)	LR (mW)
$P_{transmission}$	36	1
$P_{reception}$	14.4	0.450
P_{idle}	14.4	0.05
P_{sleep}	0.015	does not sleep



(a) No REQ sent by 3 (b) REQ replied with WAIT

Figure 2. Overcoming the deafness problem in CMAC



(a) The only case of collision (b) ACK is sent on receiver's channel

Figure 3. Collision Prevention

Sensor data frames are assumed to be large [10] and an early knowledge of the size helps the receiver to calculate the anticipated time for complete reception. All outgoing WAIT messages are encoded with the remaining time for reception. This scheme also allows the use of variable size data packets and a node need not pad a frame with additional bits. After reception of the data packet, the receiver replies with an ACK in its own channel. Thus, after the transmission of the data packet, the sender switches its MR to the channel of the receiver. We do not continue with the ACK in the same channel as that used for sending the data packet owing to a possibility of collision outlined in the next section. The MR reverts back to sleep mode after the ACK is sent (by the intended receiver) and received (by the sender). In Figure 1, after sending the CON, 5 switches *on* its MR, and tunes it to *red*, 4's default channel. It replies with an ACK in *blue*, 5's default color after the packet is received.

IV. PERFORMANCE PARAMETERS

In this section we analyze the working of CMAC from the perspective of energy consumption. We identify various sources of energy loss in sensor networks and explain how CMAC is designed to minimize, if not solve them completely. Table 1 gives the power dissipation parameters for the two radios which are in accordance with Berkeley MICA2 mote specifications [1] and the pico-radio project [6] respectively. In Figures 2 and 3, numbers (1, 2, 3, 4) identify the nodes and the letters (a, b, c) stand for their allotted channels.

A. Idle time reduction

CMAC attains energy conservation chiefly by placing nodes in the sleep mode and waking them up only if communication is necessary. Thus, energy spent in idle listening is significantly reduced. This idle energy cannot be neglected and is comparable to that used by the radio during reception (about 30 mW). Recognizing that idle listening consists of 90% of power consumption [3], our emphasis on the default sleep mode results in large savings when the network is in operation for prolonged time intervals. Most periodic self wakeup based MAC schemes [10-12] have a *listening* time in every cycle and all data exchange is undertaken only in this period. The on-demand wakeup makes CMAC preferable in time-critical applications like radiation monitoring in power plants, intruder tracking and sniper location, amongst others.

B. Deafness & Overhearing

Deafness is caused when node 3 (Figure 2(a)) attempts to contact node 2, which is currently engaged in communication with a third node (node 1). In our scheme, if node 2 is transmitting data, it will do so in its default channel, *b*. Recall that node 3 monitors *b* through its LR before transmitting the REQ and finding the channel occupied, it continues monitoring till the channel becomes free. Similarly if node 2 is receiving, its MR is tuned to the sender's default channel but its LR is still monitoring *b*. In this case, 3 finds *b* unused and transmits a REQ in channel *b*. This REQ will be received by 2 and replied with a WAIT (Figure 2(b)). We conclude that there are only two possible deaf periods: The first is the time between sending the REQ and switching back to the default channel after the reception of the CON/WAIT (or timeout), which we define as $T_{deaf}(s)$. In this case, a sender's LR is tuned to the intended receiver's default channel and it cannot hear the control messages sent by another node for $T_{deaf}(s)$ interval (Figure 1). We assume an inter-node separation of 40 meters, a train of 8 pulses for REQ and CON. Similar to SMAC[10], DIFS = 10 × slot time and SIFS = 5 × slot time. Channel switching time T_{switch} is 100µs. Here. Considering the channel capacity as 20 kbps and neglecting propagation time, the deaf period is then given by:

$$T_{deaf}(s) = DIFS + T_{REQ} + SIFS + T_{con} + T_{switch}$$
$$= (10 + 0.04 + 5 + 0.04 + 0.1)ms = 15.18ms$$

At the receiver end, the worst case deaf period, $T_{deaf}(r)$, begins at the time of hearing the first REQ and extends till the header

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2006 proceedings.

information in the data frame is read (Figure 1). Recall that the header contains the size of the packet and WAIT packets can be sent only after the duration of the data reception is known thus causing this delay. We take into account the time required (T_{trans_on}) for the MR to be switched *on* after reception of the CON on the LR. For the ATmega128 microcontroller on the MICA2 mote [1], T_{trans_on} =180μs and 6μs for the TI MSP430 used in Telos [9]. In the following calculation for $T_{deaf}(r)$ we assume that 20 bytes of header suffice for conveying the wait time:

$$T_{deaf}(r) = SIFS + T_{CON} + MRT_{trans\_on} + T_H + T_{switch}$$
$$= (5 + 0.04 + 0.18 + 20 \times 0.04 + 0.1)ms = 6.12ms$$

Similar to 802.11, after a successful transmission, there is a *post-backoff* which prevents channel capture by a node and allows this deaf period to be spaced in time, even if there are multiple packets queued for transmission. Through WAIT, we eliminate re-transmissions of the REQ in a node's effort to contact a particular neighbor. Through non-repeating channel assignment and careful channel transitions, CMAC ensures that neither control messages nor data is heard by any node other than the intended destination.

C. Collision Prevention

CMAC is collision free once data transmission begins. We identify only two cases of collisions during the phase of REQ/CON/WAIT control message exchanges. The REQ sent by a node to an intended receiver can collide with a CON or WAIT sent by the latter. We assume that the pulses, being of a very short duration are not identified by the LR during the channel sensing phase. Again, the REQ is sent on the default channel of the receiver. Hence, two or more REQ pulse trains may collide when different nodes attempt to wake up the same receiver (Figure 3(a)). In this case, analogous to 802.11, both nodes *backoff* and retry for the channel with an increase in their contention window. As all control message exchanges occur in the receiver's default channel, the collision domain is split to include only those nodes who are contending for the same destination node in their neighborhood. Recall that the control pulses are of a short duration, ranging in the order of half a millisecond for a 20kbps channel and the resulting probability of collision is significantly reduced.

D. ACK POLICY

In the earlier section, we have described the operation of the MR during transfer of data packets. Data transmission has been carried out in the channel of the sender and the receiver tuned its own MR to this channel. However, on completion of the transmission, the sender switches its MR to the channel of the receiver for the ACK. We now provide a simple explanation for the ACK being sent on the receiver's default channel, and not the one used for data reception. In Figure 3(b), there are two sets of simultaneous data transmissions: from nodes 4 → 3 and 1→ 2. On the reception of the data packet by node 3, an ACK in channel *a* would interfere with the ongoing communication between 1 and 2, also in the same channel. In our scheme, the ACK is sent in a node's default channel, here *c*, thus eliminating possibility of a collision.

E. WAIT POLICY

When a node currently engaged in data reception with a transmitter receives a REQ from yet another node, it generates a WAIT packet. In this packet, it places the binary representation of the index k which is a measure of the time duration left to complete the ongoing data reception. The actual time T_{left} (in millisecond units) is calculated as $T_{left} = 2^k + c$, where c is a constant.

In Figure 2 (b), 3 receives the WAIT from 2 who, in turn, is receiving a data packet from 1. As the period T_{left} indicates the minimum duration before which a subsequent RTS should be sent to 2, 3 should try and schedule packets to other nodes, if any, in that time. We now outline a simple scheduling policy, in which we leverage the knowledge of a node's busy period. The advantage of this scheme is that the control overhead for repeatedly probing a busy or bottleneck node is avoided.

In Figure 2(b), on receiving the WAIT packet w_1, 2 first checks whether it was the first node that received a WAIT from node 3. We identify this condition as "*receiving the first WAIT*". As mentioned earlier, this information is included in w_1 by the provision of an additional pulse. If node 2 is indeed the first node, it then suspends all transmissions up to time T_{left}. After this time, the data packet reception by node 3 is completed and 2 now re-transmits without any further channel sensing. Though *DIFS* period is no longer utilized in this case, we still allow for the REQ/CON handshaking procedure to alert the receiver (here node 3) of node 2's next incoming transmission. However, we allow for data reception by node 2 in this T_{left} interval in which case it suspends its wait. If a new data packet reception is initiated by 2, the earlier packet that incurred the WAIT is now treated as any other in its queue and must undergo the process of channel sensing and the arbitration procedure again.

In case a node was not the first to receive the WAIT, it places the data packet in a temporary queue Q_t, with a maximum capacity of 2 data packets. It then draws the next packet from the MAC queue and begins the arbitration procedure for its transmission. Now if this packet too incurs a WAIT, it is added to Q_t. As the maximum queue length for Q_t is set at 2, the second WAIT packet fills Q_t completely and the node now cannot draw a fresh packet from the MAC queue for transmission. There are now two conditions for the second wait packet (w_2) received:

1. The node received the first WAIT for the packet w_2 in which case it follows a procedure similar to the one described earlier. Even if the packet receiving w_1 is present in Q_t for longer time duration, the second packet is guaranteed transmission and is sent out first after the duration T_{left} for w_2.

2. If the node did not receive the first wait for both w_1 and w_2, a node waits for the minimum of T_{left} for w_2 and the time remaining for w_1 before sensing the channel. Both packets are considered as ordinary packets and are scheduled as such after appropriate channel sensing and handshaking procedure. As soon as one of the two packets queued Q_t is transmitted, the MAC can draw the next packet sent by the higher layer and resume its usual operation.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2006 proceedings.

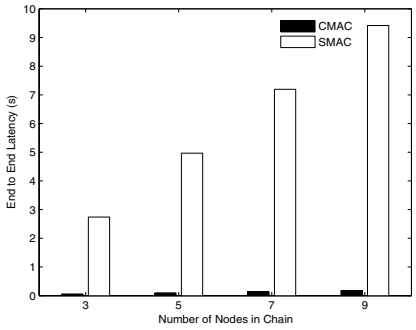


Figure 6(a). Comparison of analytical values of end to end latency

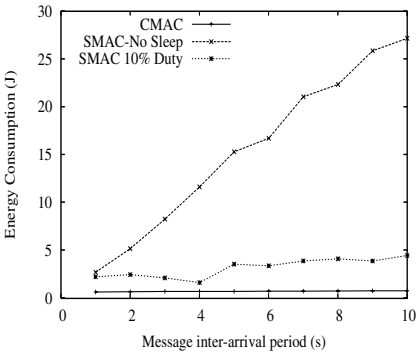


Figure 7(a). Aggregate Energy Consumption

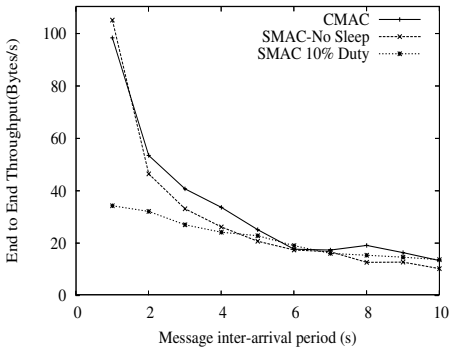


Figure 7(b).Throughput

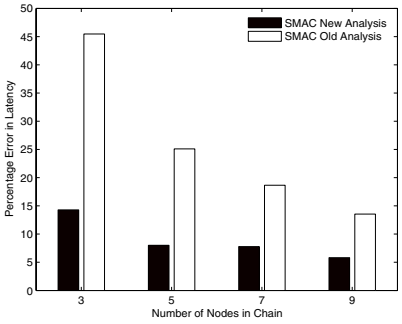


Figure 6(b).Percentage error in the proposed and earlier analytical formula for latency in SMAC

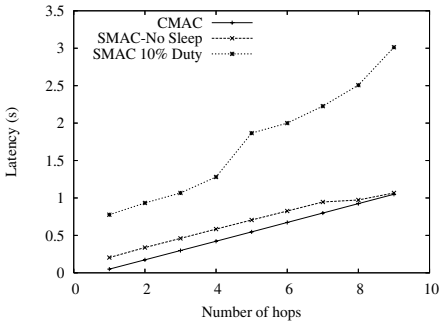


Figure 7(c). Latency in low load

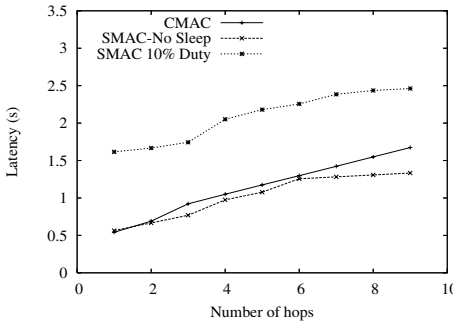


Figure 7(d). Latency in high load

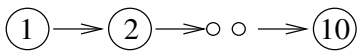


Figure 5. Chain topology used for simulation study

V. PERFORMANCE EVALUATION

We use *ns*-2 simulator (version 2.26) [7] for simulating CMAC. We have done a comparative analysis of the throughput, latency, and energy consumption in CMAC, and two modes of SMAC with No-sleep, SMAC(N) and SMAC with 10% duty cycle without adaptive listen, SMAC(10%) for a similar chain topology as in [10] for ease of comparison (Figure 5). All results presented here are averaged over 10 simulation runs with different seed values. Nodes are separated by 40 meters such that only adjacent nodes are in communication range. Distinct channels are assigned to nodes, satisfying the constraints of 2-hop coloring described earlier. For the chain topology, the source node (Node 1) generates 20 data messages each 100 bytes long and are sent to the destination node (Node 10). Table 1 summarizes the energy consumption of the two radios.

Results for chain topology:

A. Energy Consumption

Figure 7(a) shows the comparative aggregate energy consumption of nodes. Energy consumption of a node is obtained by calculating the total transmit, receive, idle and sleep time for both the radios in CMAC and the single radio in S-MAC. Results show that our protocol out-performs both the modes of SMAC with nearly 200% reduction in the total energy consumed. As can be observed from the figure, CMAC has very low and near-constant energy consumption. Recall from section III and IV, in CMAC, the MR wakes up only when it has to transmit or receive data and spends rest of the time in sleep mode. To avoid synchronization/deafness problems, the LR is switched on all the time. As the LR has negligible power requirements, it barely contributes to the total energy consumption and hence the energy consumed by the control messages is negligible. Only the MR's transmit/receive periods contribute significantly to the total energy consumed as the idle period is very minimal. Moreover, as nodes in the neighborhood are assigned different channels, overhearing is also avoided. As the total number of DATA messages to be transmitted is constant, the energy consumption is nearly constant. However, in S-MAC(N), the radio has to be active all the times even though it has no data to transmit, thus expending a lot of energy in idle listening and overhearing. As the inter-arrival period increases, the nodes spend more time in idle listening, adding to the energy cost. This is reflected in the linear increase in the total energy consumed. This idle listening is avoided by the SMAC(10%) by keeping the radio active only during listen times and sync periods [10]. Although the SMAC(10%) has fair savings when compared to the SMAC(N),

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2006 proceedings.

it has an adverse effect on throughput as we discover in the next result.

B. Aggregate Throughput

We measure the end to end throughput by varying the inter-arrival times of the messages as shown in Figure 7(b). Both CMAC and SMAC(N), transmit data whenever the next hop is ready to receive, resulting in high and almost comparable throughput. SMAC(10%) is allowed to transmit/receive for only one-tenth of the frame time and must wait for the next cycle to forward a received packet. This results in a severe reduction in throughput. However, as the inter-arrival period increases, messages arrive with greater separation in time and thus decreasing the throughput in both CMAC and SMAC(N). SMAC(10%) exhibits the expected decline in performance as the packet arrival delay adds to that induced by periodic sleep.

C. Per hop latency

In this section we discuss and analyze the per-hop latency at low and high traffic loads (Figures 7(c) and 7(d) respectively). An inter-arrival time of 10 seconds represents low-load conditions while high load means all messages are generated and ready for transmission at the source node at the same time. All the schemes display the expected increase in per-hop latency with SMAC(N) and CMAC having comparable values. SMAC(10%) performs poorly, ranging from 150% to 50% higher at the last hop for low and high loads respectively.

Due to periodic sleeping, a node can only transmit a received packet to next hop node in the listen period in the successive frame, thus adding additional delay. However in our protocol, nodes forward the packet immediately when the next hop node is ready for reception. At high load, the latency increases for all the protocols due to higher queuing delay. We can also observe slightly higher latency of our protocol at high load when compared with no-sleep node of SMAC. This delay is attributed to the channel negotiation period and a small deafness period when a node communicates with the next-hop node.

VI. CONCLUSION

This paper presents CMAC, a de-synchronized multi-channel MAC protocol for sensor networks that addresses the critical issue of energy conservation without trade-offs in latency and throughput. Through a wake-up radio in addition to the main transceiver, multi-channel communication is accomplished resulting in a virtually collision-free protocol. CMAC enables maximum possible sleep-time, prevents overhearing and has minimal control overhead. In addition to the above, our protocol supports high data rates making it application independent. CMAC however needs an adequate number of available channels to satisfy the 2-hop coloring constraint. We did a comprehensive study of CMAC through computer simulation and results indicate significant performance improvements. Simulation results reveal that our protocol achieves a dramatic 200% improvement in energy savings and 50% - 150% decrease in latency when compared to SMAC. As part of our future work, we plan to incorporate node mobility and implement other scheduling scheme to best utilize wait periods. By choosing the WAIT periods as a function of traffic class, service differentiation and a Quality of Service metric can be incorporated. An analytical model of a multichannel MAC scheme is another new research direction and work is underway to define such a framework for CMAC.

REFERENCES:

[1] Website: XBOW MICA2 Mote Specifications: http://www.xbow.com/

[2] J. R. Agre, L. P. Clare, G. J. Pottie and N. P. Romanov "Development platform for self-organizing wireless sensor networks," *SPIE-Aerosense, Unattended Ground Sensor Technologies and Applications*, Vol. 3713, pp. 257-268.

[3] J. Reason and J. Rabaey, "A study of energy consumption and reliability in a multi-hop sensor network," *ACM SIGMOBILE Mobile Computing and Communications Review*, Vol. 8, pp. 84-97, January 2004.

[4] T. Herman, S. Tixeuil, "A Distributed TDMA Slot Assignment Algorithm for Wireless Sensor Networks," *Proceedings of the First Workshop on Algorithmic Aspects of Wireless Sensor Networks*, Lecture Notes in Computer Science, Springer-Verlag, pp. 45-58, July 2004.

[5] Z. Chen, A. Khokar, "Self Organization and Energy Efficient TDMA MAC Protocol by Wake Up For Wireless Sensor Networks," IEEE International Conference on Sensors and Adhoc Communication and Networks (Secon), October 2004.

[6] B. Otis, Y. H. Chee, J. Rabaey, "A 400μW-Rx, 1.6mW-Tx Super Regenerative Transceiver for Wireless Sensor Networks," *IEEE Intl. Solid-State Circuits Conference*, *ISSCC* 2005.

[7] UCB/LBNL/VINT Network Simulator (NS-2), Available at http://www.isi.edu/nsam/ns/index.html.

[8] C. Busch, M. Ismail, F. Sinrikaya, and B. Yener, "Contention-Free MAC Protocols for Wireless Sensor Networks," *Proc. of the 18th Annual Conference on Distributed Computing (DISC 2004), LNCS 3704*, pp. 245-259, Netherlands, October 2004.

[9] TI MSP430 datasheet available at the website: http://www.moteiv.com/products/docs/telos-revb-datasheet.pdf

[10] W. Ye, J. Heidemann and D. Estrin, "Medium Access Control with Coordinated, Adaptive Sleeping forWireless Sensor Networks", *IEEE/ACM Transaction on Networking*, Vol. 12, No. 3, pp. 493-506, June 2004.

[11] T. V. Dam, and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," In Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys), Los Angeles, CA, November 2003.

[12] Jing Ai, Jingfei Kong, Damla Turgut, "An Adaptive Coordinated Medium Access Control for Wireless Sensor Networks," *Proc. of the Nineth IEEE Symposium on Computers and communications (ISCC 2004)*, Alexandria, Egypt, June, 2004.

[13] J.S.Pathmasutharam, A.Das, A.K.Gupta, "Primary channel assignment based MAC (PCAM) - A multi-channel MAC protocol for multi-hop wireless networks", *Proc. IEEE WCNC*, Atlanta, GA, 2004/03.

[14] C. Guo, L.C. Zhong, and J.M. Rabaey, "Low power distributed MAC for ad hoc sensor radio networks," *Global Telecommunications Conference (GLOBECOM '01). IEEE,* Vol. 5, November 2001.

[15] M. J. Miller and N. H. Vaidya. "A MAC Protocol to Reduce Sensor Network Energy Consumption Using a Wakeup Radio," *IEEE Transactions on Mobile Computing*, Vol. 4, No. 3, pp. 228-242, May/June 2005.

[16] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, "Optimizing Sensor Networks in the Energy-Latency-Density Design Space," *IEEE Transactions on Mobile Computing*, vol. 1, no. 1, pp. 7080, January-March 2002.

[17] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, "Topology Management for Sensor Networks: Exploiting Latency and Density," in *ACM MobiHoc 2002*, June 2002.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the WCNC 2006 proceedings.