
Brajendra Kumar Singh  
Department of Electrical and Computer Engineering  
University of Windsor  
Windsor, Canada  
singh11g@uwindsor.ca

Kemal Ertugrul Tepe  
Department of Electrical and Computer Engineering  
University of Windsor  
Windsor, Canada  
ktepe@uwindsor.ca

Abstract—This paper presents a MAC layer protocol, called RT-MAC, for real-time data streaming in wireless sensor networks. RT-MAC eliminates the need for a wireless medium by potential transmitting nodes in a range by introducing feedback control packet, called Clear Channel (CC). RT-MAC maximizes spatial channel reuse by avoiding false blocking problem of RTS/CTS exchange based wireless MAC protocols. RT-MAC reduces contention duration for control packets to facilitate faster traveling of data packets; thus, it reduces end-to-end delay of data packet transmission. RT-MAC facilitates periodic delivery of data packets as well as fast reporting of an alarming event. This paper provides the upper bounds of end-to-end delay of data packet transmission with periodic sleep/listen schedule for RT-MAC protocol. In this paper, extensive simulation results are presented. RT-MAC supports single stream communication between a randomly selected source and sink node pair as well as multi-stream non-interfering communication among different source and sink node pairs.

Keywords—MAC protocol, real-time data communication, spatial channel reutilization, wireless sensor networks.

I. INTRODUCTION

Most research in wireless sensor networks (WSNs) is devoted to energy conservation since batteries are generally not replaceable once a sensor network is deployed. Timing constraints generally assume secondary importance. But with the advancement of sensor technology, sensors are increasingly used for time critical (real-time) applications. To fulfill real-time requirements, time critical aspects need to be addressed in WSNs. For a given sensor hardware, MAC layer protocol is crucial for ensuring real-time communication as it provides excellent platform for any time critical developments at higher layers of protocol stack in WSNs. That is why this paper will attempt to address real-time challenges at MAC layer.

Existing real-time MAC layer protocols for WSNs are designed either for some specific sensor network topologies, or have unrealistic assumptions such as hardware requirements and large delay guarantees. RT-MAC protocol for WSNs presented in this paper alleviates some of these unrealistic limitations. RT-MAC works with any network topology, requires no special hardware and has lower delay guarantees. This is achieved by introducing a novel feedback mechanism in the MAC protocol. RT-MAC is capable of handling soft real-time WSN applications. For example, due to its consistent and predictable packet transmission pattern, RT-MAC is useful for periodic monitoring applications such as the monitoring of fault in power lines. Because of reduced end-to-end delay and independent traveling of initial data packets, RT-MAC is also useful for event based monitoring applications such as wireless sensor based security setups, and natural disaster monitoring applications such as monitoring forest fire and flood situations.

Section II of this paper presents related research work. Problem analysis is provided in Section III. Section IV presents description of the proposed protocol. Information about the simulation is given in Section V. Section VI presents results and related discussions. Conclusion and future work are presented in section VII.

II. RELATED WORK

General purpose contention based MAC layer protocols such as S-MAC [1] and T-MAC [2] are not designed for real-time communication in WSNs due to their inconsistent data transmission patterns, which can cause unpredictable end-to-end delays. In the literature, there are seven major real-time MAC protocols for WSNs. These are VTS, I-EDF, Dual-mode real-time MAC protocol, TOMAC, PR-MAC, CR-SLF and RRMAC. Virtual TDMA for sensors (VTS) protocol [3,4] is for soft real-time WSN applications. VTS is based on S-MAC protocol. With regard to real-time communication, it is not possible in VTS to facilitate packet transfer to several hops in a given TDMA slot duration (which is a frame in S-MAC protocol); thus, it can not provide speed up to alarm messages by varying the duty cycle of a TDMA slot. In general, though VTS gives timeline guarantees, but these guarantees are too large due to poor spatial channel reutilization in TDMA mechanism. Implicit earliest deadline first (I-EDF) [5] is a hard real-time MAC layer protocol for WSN applications with periodic data transmission. It assumes cellular network structure and needs global synchronization. It also requires specialized hardware such as more capable sensor nodes called routers for cell to cell communication. Dual-mode real-time MAC protocol [6,7] is a hard real-time MAC protocol, and is based on I-EDF protocol. It uses reservation mechanism to avoid collisions. In general, reservation based real-time MAC approaches are more unpredictable in the sense that the control packet sent for reservation may not find the next few hop...
neighbors free to make this particular reservation attempt successful. In contrast to that, RT-MAC uses predictable feedback based approach to avoid collisions. RT-MAC starts sending the first data packet adventurously. The transmission of subsequent data packets depends upon the successful progression of their previous data packets towards the destination. Additionally, the dual-mode real-time MAC protocol relies heavily on global synchronization mechanism as well as nodes’ absolute position information. TOMAC protocol [8] provides hard real-time message ordering at the MAC layer using nondestructive bitwise arbitration for one hop mesh network. This protocol is hard to generalize for multi-hop networks and other communication topologies. Path oriented real-time MAC (PR-MAC) [9] is a soft real-time protocol for tree based WSNs. Being tree based protocol, PR-MAC needs global synchronization, and has no spatial channel reutilization. Channel reuse-based smallest latest-start-time first (CR-SLF) [10] algorithm schedules messages at the MAC layer to increase spatial channel reuse in soft real-time multi-hop WSNs. CR-SLF is developed for mobile WSNs and uses centralized scheduling algorithm, which creates scalability problems. RRMAC [11] is a TDMA based hard real-time MAC protocol for a multi-hop convergecast WSNs. RRMAC’s super frame is based on the IEEE 802.15.4 frame structure. Being a tree based protocol, RRMAC needs global synchronization and also has scalability issues due to the constraint of superframe length and the amount of data aggregation possible. RRMAC also assumes that the sensor nodes have two RF power levels.

III. PROBLEM ANALYSIS

We used the S-MAC as a basic protocol for the development of RT-MAC. As duty cycle doesn’t change during run time in S-MAC, it is possible to define timing deadline with some modification to this protocol. S-MAC itself is not suitable for real-time communication due to its unpredictable end-to-end delay for data packet transmission. In view of this, we made two major changes in S-MAC protocol to develop RT-MAC. First, we introduced a feedback mechanism with the help of an additional control signal, called CC. Feedback mechanism of RT-MAC removes uncertainty of medium access associated with S-MAC by clearly specifying as to who will gain medium access in a neighborhood. This enables RT-MAC to provide end-to-end delay guarantees. Second, large carrier sense duration (typically 3 to 5 times larger than the duration of a typical control packet) in S-MAC prior of RTS transmission is removed. This helps to reduce end-to-end delay significantly since carrier sense in S-MAC is needed prior to every data transmission irrespective of whether it is a successful transmission or not. Like S-MAC, RT-MAC uses RTS-CTS-DATA-ACK communication paradigm to avoid the hidden terminal problem of wireless communication. In addition to this, RTS-CTS exchange can falsely blocks a possible non-interfering transmission of consecutive data packets separated by four hops. Due to feedback based approach, RT-MAC solves this problem of false blocking [10,12,13] in the RTS-CTS exchange based MAC protocols for ad-hoc wireless network. Thus, RT-MAC ensures maximum possible spatial channel reutilization by guarantying simultaneous collision free data packet transmissions separated by four hops, which will be discussed further in Section IV. This is another reason of lower end-to-end delay in RT-MAC.

IV. PROTOCOL DESCRIPTION

RT-MAC is designed for single channel radio sensor node with similar transmission power. It supports source to sink communication pattern. Any node can be a source or sink node in the WSN, and it remains so throughout the lifetime of a communication stream.

The communication pattern of RT-MAC can be explained by considering 5 contiguous hops between nodes, say N₀ to N₅. N₀ node initiates a data packet (P₀ in this case) transmission with its one hop neighbor N₁ by sending an RTS control packet. Thus, there is an active exchange of RTS, CTS, DATA and ACK between N₀ and N₁. Therefore, it is called the active hop. Hop between N₁ and N₂ overhears communication between N₀ and N₁ passively, thus it is called the passive hop. Now, if we leave one hop between N₂ and N₃ as a buffer hop, then there is a possibility of another data packet (P₁ in this case) transmission between N₄ and N₅ node that will not interfere with P₀’s transmission in any way. It is because the hop between N₄ and N₅ will have an active communication, whereas the hop between N₃ and N₄ will be a passive hop due to overheard communication of P₁. This active-passive-buffer-passive-activate hop pattern is repeated throughout a packet stream. Hence, there could be several simultaneous data packet transmissions in a stream separated from each other by at least four hops. Thus, RT-MAC ensures maximum possible spatial channel reutilization.

RT-MAC follows the frame synchronization mechanism of S-MAC protocol. Thus, it keeps some duration reserved at the beginning of each frame for the synchronization purpose as shown in Fig. 1. Working of RT-MAC protocol is based on the CC control packet. CC is used to assign an appropriate value to the clear channel flag (CCF) of every sensor node. CCF is a Boolean variable. Central idea of this protocol is that the node that has CCF as 1 can transmit as well as receive data packets, whereas it can only receive if its CCF value is 0. However, a node can initiate a CC transmission even if its CCF is 0. Initially all nodes have CCF value as 1. CC control packet has a clear channel counter (CCC), which is an integer variable ranging from 0 to 3. The value of CCC is 3 at the originating node of CC and is decreased by one with one hop transmission of CC. CC is always transmitted from sink to source direction. If the value of CCC of a CC control packet is 2 or 3 in a node, then CCF of that node will remain 0. However, if the value of CCC of a CC is 0 or 1 in a node, then the CCF of that node will become 1, which enables it to initiate a data packet transmission.

In Fig. 1, N₀ is a source node and N₁₀ is a sink node. In the beginning, N₀ has several data packets to send to sink node. Now, N₀ sends RTS to N₁. N₁ responds with CTS. Then, N₀ sends data packet P₀ to N₁. After receiving P₀, N₁ sends ACK to N₀. This completes one data transfer cycle by one hop. The Duration of one data transfer cycle is denoted as Tₓ, and the duration of one control packet is denoted as Tc. After getting ACK, N₀ sets its CCF value to 0. Similarly, in the next three Tₓ
after successful transmission of ACK to N3, N4 waits for 2 immediate previous hop nodes, which are odd numbered nodes numbered nodes with respect to the source node. Thus, their observed that the HC of a data packet becomes 0 only at even durations, P0 is transmitted up to N4 and CCF of N1, N2 and N3 becomes 0. Each data packet has a hop counter (HC) integer variable whose value varies from 4 to 0 for the first four hops of a communication stream, and 2 to 0 for all subsequent two hops segments of the communication stream. At N0, the value of HC of P0 is 4 and it is decreased by one each time P0 is transmitted successfully by one hop. Once P0 reaches the N4, its HC becomes 0. Then, N4 sets HC of P0 to 2, which will become 0 again after the next 2 hops transmission. It is observed that the HC of a data packet becomes 0 only at even numbered nodes with respect to the source node. Thus, their immediate previous hop nodes, which are odd numbered nodes with respect to the source, will initiate CC transmission. Now, after successful transmission of ACK to N3, N4 waits for 2Tc duration prior to forwarding P0 to N5. This waiting period is needed to avoid collision between the CC originating from N1 node and the data packet transmission originating from N4 node. The same is true for all later nodes where HC of a data packet becomes 0. In the meantime, after receiving ACK from N4, N3 sends CC signal with its CCC as 3 to N2 in the first Tc duration. N2 decrements value of CCC by one upon successful reception of CC. In the second Tc duration, N2 forwards CC signal with its CCC as 2 to N1. This CC is also overhead by N1, which acts as implicit acknowledgement for it that N2 has received CC successfully. Thus, in general, the explicit acknowledgement of CC transmission is not needed. In the third Tc duration, N1 forwards CC signal with its CCC as 1 to N0 and sets its own CCF to 1. Hence, after getting CC from N1, N0 decrements the value of CCC of CC to 0, and sets its own CCF to one. Now, N0 can transmit new data packet P1 to N1 in the next Tx duration, and after that, N1 can also forward P1 to N2 in subsequent Tx duration. However, N2 node can not forward it further as its CCF value is 0. It has to wait for the next CC packet originating from N1 node. However, N2 will generate new CC only after N1 receives ACK from N0, which N0 will send to it after successful reception of P0. Thus, with the exception of initial 4 hops, the whole communication stream is divided into segments of two hops with respect to the source node. As last segment may have one or two hops, the node just before sink will initiate CC transmission.

A. Upper limit of end-to-end delay for \( T_{U} < 6T_{x} + 8T_{c} \)

\( T_{A} \) represents arrival interval between data packets at the MAC layer of a node for transmission. A data packet can travel one or more hops per frame depending upon the ON duration \( T_{U} \) of sensor nodes. Number of hops traveled by a data packet per frame is represented by \( \eta_{H} \). In RT-MAC, one complete data transfer cycle is possible if two neighboring nodes, say N4 and N5, are ON at least for one Tc duration. In this case, N4 sends RTS successfully to N5 in one Tc duration, and then both N4 and N5 will remain ON till the completion of one data transfer cycle. As explained earlier in this section, even numbered nodes wait for 2Tc duration prior to forwarding a data packet further. Now, if N4 and N5 are ON for just 2Tc duration, then there is no data transfer possible as both nodes will go in the sleep mode after 2Tc duration. But, if both N4 and N5 are ON for 3Tc duration, then it may happen that N4 could send CC to N5 in third Tc duration, and then instead of going to sleep mode, N5 waits for acknowledgement of CC from N4 in the next Tc duration. Then, N4 responds to CC by sending RTS for a data packet to N5; after that N4 responds with CTS and thus both N4 and N5 do not go into sleep mode until the end of ongoing data packet transfer. However, if we take 4Tc ON duration, then N5 will also be able to overhear CTS sent by N4 to N6. In that case, N5 will wake up as per adaptive listening mechanism of S-MAC [1] after the ongoing data transfer. Thus, in 4Tc duration, data packet transmission by 2 hops is possible. Therefore, we took the maximum ON duration limit as 3Tc that guarantees at the maximum one data transfer cycle per frame (\( \eta_{H} = 1 \)). Data packet transmission by more than one hop is possible in 3Tc duration, particularly if the starting node for a data transmission is an odd numbered since it need not to wait for 2Tc duration prior to initiating a transmission. Thus, limit derived in this manner is the upper limit of ON duration that guarantees a particular \( \eta_{H} \). Carrying out analysis similar to above, it is found that \( T_{U} = Tx+33 \) ensures \( \eta_{H} = 2 \) hops/frame, \( T_{U} = 2Tx+6Tc \) ensures \( \eta_{H} = 3 \) hops/frame, \( T_{U} = 3Tx+6Tc \) ensures \( \eta_{H} = 4 \) hops/frame, and so on. Thus, the following conditions for range of \( T_{U} \) can be derived that ensures a given number hops transmission per frame.

\[
\begin{align*}
3T_{c} \leq T_{U} & \leq T_{c}+3T_{c} \quad \text{for } \eta_{H} = 1 \\
2T_{c}+6T_{c} \leq T_{U} & \leq 3T_{c}+6T_{c} \quad \text{for } \eta_{H} = 3 \\
4T_{c}+9T_{c} \leq T_{U} & \leq 5T_{c}+9T_{c} \quad \text{for } \eta_{H} = 5 \\
(\eta_{H}-1)T_{c}+\{3(\eta_{H}-1)/2\}T_{c} & \leq T_{U} \leq \eta_{H}T_{c}+\{3(\eta_{H}-1)/2\}T_{c} \quad \text{for } \eta_{H} = 1, 3, 5, 7, \ldots \text{ up to } n
\end{align*}
\]  

Here, \( n \) represents the total number of hops between the source and sink node in a stream. Similarly,

\[
\begin{align*}
Tx+3T_{c} \leq T_{U} & \leq 2Tx+6T_{c} \quad \text{for } \eta_{H} = 2 \\
3T_{c}+6T_{c} \leq T_{U} & \leq 4T_{c}+9T_{c} \quad \text{for } \eta_{H} = 4 \\
5T_{c}+9T_{c} \leq T_{U} & \leq 6T_{c}+12T_{c} \quad \text{for } \eta_{H} = 6 \\
& \ldots
\end{align*}
\]
constraint for packet needs to travel by the second packet. Thus, it can be considered as if the second packet

In the case of the second packet, an offset delay of 4 hops is introduced due to the adaptive listening. If

This constraint represents the upper limit of end-to-end delay for the second packet. The remaining two hops will be traversed in the 9th frame. Thus, nine frames are needed for end-to-end data packet transmission. In this context, n \( \eta \eta \) represents number of hops (less than \( \eta \eta \)) just before the sink node that still need one additional frame duration. Here, \( \% \) represents reminder operation. In the following discussion, \([T_f \rightarrow \{\%\}]\) signifies that a data packet is transmitted by \( r \) number of hops in a particular frame duration \( T_f \). Thus, the upper limit of end-to-end delay (also referred to as packet transfer delay) for the first data packet over \( n \) hops is calculated as follows:

For \( n \% \eta = 0 \)

For \( n \% \eta \neq 0 \)

Referring to Fig. 1, minimum time interval between two consecutive packets is \( 4T_x + 5T_c \). It is called offset delay. Thus, in the case of the second packet, an offset delay of 4 hops transmission time is added to the packet transmission time of the second packet. Thus, it can be considered as if the second packet needs to travel by \( n+4 \) hops, instead of \( n \) hops. Thus, the upper limit of end-to-end delay for the second packet is as follows:

Similarly, the upper limit of end-to-end delay for the \( m^{th} \) data packet is as follows:

\[
T_D(m, n) = \left[ \frac{4(m-1)+n}{\eta} \right] T_f \\
\text{for } (4(m-1)+n) \% \eta = 0
\]  

B. Upper limit of end-to-end delay for \( T_d \geq 6T_x + 8T_c \)

Referring to Fig. 1, if \( T_d > 4T_x + 5T_c \), then offset delay will be equal to actual \( T_d \) instead of \( 4T_x + 5T_c \). In fact, if \( 4T_x + 5T_c < T_d < 6T_x + 8T_c \), then there is still some possibility of collision between the CC originating from \( N_i \) (and traveling up to \( N_j \)) with the data transmission between \( N_1 \) and \( N_2 \). However, if \( T_d \geq 6T_x + 8T_c \), then two consecutive packets will be separated by at least 6 hops that guarantees the non-interference of backward traveling CCs and forward traveling later data packets. Thus for \( T_d \geq 6T_x + 8T_c \), a newly arrived data packet at the MAC layer of a node will not be affected by the previous data packets because it need not to wait for CC from them since CCs had already reached the node prior to its arrival at the MAC layer in the node. Hence, once the transmission of a data packet starts from the source node, it will have the same packet transmission time as the first packet, which is given by (4) and (5). Thus, the only problem we have here is to find offset for the \( m^{th} \) packet. In this regard, Figs. 2 and 3 show two possible cases of a data packet arrival. In either case, we need to find out number of frames needed to transmit the previous \( m-1 \) packets. Thus, we consider a multiplier constant \( a \) that satisfies the following equation:

\[
(a-1)T_U \leq (m-1)T_d < aT_U
\]  

Equation (8) can be rewritten as follows.
Here, $T_{AL}$ and $T_U$ are known apriori. Now, we need to find out integer value of $a$ that satisfies equation (9). Let this integer value of $a$ be denoted by $||a||$, where $||a|| \geq 1$. Referring to Fig. 2, $||a||$ number of frames will definitely be used to transfer the previous $m-1$ data packets for the case of $T_U \leq 6Tc + 8Tc < T_{AL}$, whereas the $||a||$th frame could be the first frame for the case of $6Tc + 8Tc \leq T_{AL} - T_U$, as shown in Fig. 3. Additionally, there is a chance that some portion of the ON duration of the $||a||$th frame may also be involved in the transmission of the $m-1$th data packet. Therefore, the remaining ON duration of the $||a||$th frame, shown as $T_{RI}$ in above figures (which is $||a||T_U - (m-1)T_{UL}$), will be available for the transmission of the $m$th data packet. Thus, using (1), (2), (4), (5) and $T_{RI}$, the upper limit of end-to-end delay for the $m$th data packet over $n$ hops with $\eta_H$ hops per frame is given as below.

$$T_D(m,n) = |a| + (n / \eta_H) + 1 |T_f|$$

for $||a||T_U - (m-1)T_{AL} < ((n \% \eta_H)+1)Tc + (3(n \% \eta_H)+1)/2)Tc$

& (n \% \eta_H) is odd. \hfill (10)

$$T_D(m,n) = |a| + (n / \eta_H) |T_f|$$

for $||a||T_U - (m-1)T_{AL} > ((n \% \eta_H)+1)Tc + (3(n \% \eta_H)+1)/2)Tc$

& (n \% \eta_H) is odd. \hfill (11)

$$T_D(m,n) = |a| + (n / \eta_H) + 1 |T_f|$$

for $||a||T_U - (m-1)T_{AL} < ((n \% \eta_H)+1)Tc + (3(n \% \eta_H))/2)Tc$

& (n \% \eta_H) is even. \hfill (12)

$$T_D(m,n) = |a| + (n / \eta_H) |T_f|$$

for $||a||T_U - (m-1)T_{AL} > ((n \% \eta_H)+1)Tc + (3(n \% \eta_H))/2)Tc$

& (n \% \eta_H) is even. \hfill (13)

V. Simulation

We carried out simulation study using Omnet++ simulator. Performance of the RT-MAC is compared with VTS, S-MAC, and T-MAC protocols. Minimum hop routing protocol is used for all these protocols. Values of the major parameters are given in Table I. These values are based on reference [1], [2] and [3]. In Table I, one Tic of a crystal oscillator of a sensor node is taken as 30.518 µSec.

VI. Results and Discussions

Simulation results presented here are for the worst case load scenario in which all data packets (25 data packets in this study) are available at the source node in the beginning of a simulation session. We compared performance of protocols at lower as well as higher duty cycles. Fig. 4 shows packet transfer delay for the first packet as a function of number of nodes for low and high duty cycle. It is evident from this figure that packet transfer delay for the first packet is the lowest in RT-MAC at both low and high duty cycle. It is due to the fact that the first packet travels independently as it need not to wait for CC control signal. Additionally, reduced contention duration also contributes to the reducing delay in RT-MAC. VTS takes the maximum time to deliver the first data packet to the sink at lower duty cycles because for a given TDMA slot, data packet travels only by one hop; thus, the remaining ON duration goes waste in the idle listening.

The simulation session duration is 25 seconds for RT-MAC, S-MAC and T-MAC, whereas in the case of VTS, it is taken until the end of all packets transfer as VTS could not complete data transfer up to 100 nodes in 25 second session limit. As shown in Fig. 5, at low duty cycle, all 25 packets can...
be transferred up to 40 nodes in S-MAC and up to 60 nodes in RT-MAC within 25 second session limit. However, T-MAC is able to transfer all data packets up to 100 nodes even at lower duty cycles as T-MAC has adaptive duty cycle mechanism that prevent nodes from going into the sleep mode when there are data packets in network for transmission. However, T-MAC itself is not a good candidate for real-time communication due to its inconsistent data transmission pattern because all data packets keeps on traveling together like a bunch toward the sink. At higher duty cycles, RT-MAC provides the lowest packet transfer delay due to combined effect of the maximum spatial channel reutilization and reduced carrier sense duration. In general, delay performance of RT-MAC improves with increase in duty cycle as the same data packet transmission pattern keeps on continuing for long time; whereas, in the case of lower duty cycles, the data packet transmission pattern can not continue long due to nodes going to sleep mode frequently.

Additionally, the simulation study shows that, for the simulation parameters as given in Table I, normalized energy consumption is in the range of 49 to 55 milli-joules per node at low duty cycle and 348 to 353 milli-joules per node at high duty cycle for both RT-MAC and S-MAC. Thus, it is observed that the energy consumption behavior of RT-MAC and S-MAC are more of less the same. For VTS, normalized energy consumption is in the range of 43 to 136 milli-joules per node at low duty cycle and 322 to 867 milli-joules per node at high duty cycle.

VII. CONCLUSION AND FUTURE WORK

Feedback mechanism enables RT-MAC to provide end-to-end delay guarantees as well as lower end-to-end delay without any unusual increase in energy consumption. Simulation study shows that the delay and energy behavior of RT-MAC and S-MAC are comparable. It is also observed that being contention based real-time MAC protocol, RT-MAC is able to provide lower end-to-end delay and lesser energy consumption as compared to TDMA based VTS soft real-time MAC protocol. As a future work, interfering multi-stream communication support in RT-MAC needs to be included, and this can be done by providing ability to change the duty cycle during run time. We will also try to implement RT-MAC in real sensor nodes.

REFERENCES


