A Configurable TLM of Wireless Sensor Networks for Fast Exploration of System Communication Performance

Ines Viskic and Rainer Dömer

Center for Embedded Computer Systems
University of California, Irvine
`iviskic@uci.edu, doemer@uci.edu`,

Abstract. Transaction Level Modeling (TLM) is seen as an efficient Embedded System modeling technique to reduce the simulation time in large and complex designs. This is achieved by abstracting away pin- and cycle- accurate details from communication transactions, which reduces the number of events that need to be simulated.

In this paper, we apply TLM principles to communication modeling in Wireless Sensor Networks (WSN). Modeling and simulating wireless communication is critical in exploration and optimization of WSNs as it enables evaluation of system design choices early in the design process. Unlike on-chip bus modeling, wireless communication modeling is broadcast-based and unreliable, which requires distributed medium access arbitration and timeout/retransmission capabilities. We present two TLMs of TDMA and CSMA/CA protocols. Our models are scalable to large networks and flexible in parameters and protocol configuration. Our experiments demonstrate insights to how adjusting protocol parameters in various network configurations affects the overall WSN performance.

1 Introduction

Advances in hardware and wireless network technologies have enabled widespread and cost effective deployment of WSNs. WSNs are scalable and robust ad-hoc networks of embedded sensing and transmitting devices that represent a new paradigm in system communication. Their applications include surveillance, natural phenomena detection (e.g. wind, flood, fire), GPS and traffic monitoring, all of which have differing requirements and operating conditions. With growing application complexity and expanding design space, WSNs are becoming more difficult to design and optimize. With modeling and simulating the WSN, the designer is able to observe the effects of taken design choices early and can optimize the configuration prior to system implementation.

Network communication is paramount to the functioning of a WSN. The selection of an appropriate communication protocol greatly effects the performance of the entire system. Therefore, modeling and simulating WSN communication is very useful in predicting the overall system behavior.

2

This paper reports on the modeling of two popular broadcast protocols for wireless communication: *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA) and *Time Division Multiple Access* (TDMA). CSMA/CA is part of the 802.11 standard for wireless communication based on medium sensing and collision avoidance. TDMA is a protocol that divides the time-share of the medium among the users and allocates time slots to each one. Both protocols have properties that benefit certain types of traffic profiles, but are a limiting factor in other deployment conditions. We present a TLM [7], [11] configurable for both protocols and compare their performance on a set of performance indicators.

The rest of the paper is structured as follows: related work is addressed in Section 2. Section 3 and Section 4 describe the components of a wireless sensor network and the broadcast communication protocols, respectively. Our modeling of WSN and its broadcast channels is outlined in Section 5. Finally, we present the experimental setup and analysis of the performed simulations in Section 6 and end the report with concluding remarks in Section 7.

2   Related Work

The widespread application of wireless communication systems (e.g., PDAs, cellphones, WSN, etc.) has generated significant research for their efficient modeling. With respect to the modeling objective, several approaches can be found.

The authors of [6] use SystemC [9], [8] modules to simulate and analyze the performance of the Bluetooth standard at behavioral level. The goal of their simulations is to identify noise levels and creation of a piconet in presence of noise in the channel. Similarly, we use SystemC to model communication primitives of wireless broadcast protocols at transaction level (TL), but with the objective of protocol performance estimation and network configuration.

The methodologies presented in [3] and [10] model WSN to validate the configuration before synthesis. As such, in [3] only TDMA is supported and [10] focuses on automatic code generation for sensor nodes. In [14] and [12], modeling provides insight into power dissipation and computational bottlenecks, respectively. This paper addresses the issue of design exploration and qualitative analysis of WSN communication. The objective is to identify the trade-off of various protocols and their configurations.

NS-2 is a network simulator engine traditionally used to model TCP/IP networks but has recently included features for wireless communication modeling. [2] uses NS-2 together with SystemC to model and simulate a large set of heterogeneous networked embedded systems (both wireless and wired). With respect to wireless network modeling, different network configuration parameters of NS-2 can be set (node distance and speed, power dissipation per transmisson). However, only a general statistical parameters (such as packet loss rate) are available for modelling communication. Also, the application is abstracted with statistical data of message transmission rate over a period of time. In contrast, our model includes the actual functionality of each node as an executable specifica-

tion in SystemC and therefore will help in identifying the optimal configuration parameters of the actual application at hand.

3 WSN Architecture Components

A WSN consists of a set of sensor devices (nodes) communicating with the base station via a wireless communication protocol. Sensor devices sense the environmental phenomena and transmit the measured data to the base station, while the base station gathers and processes the obtained data. An example WSN is shown in Figure 1. All components contain embedded processors and radio-frequency (RF) transceivers for local data processing and broadcasting, respectively. The capacities of memory units in each component determine the amount of sensing history. Additionally, the sensor devices contain sensing HW and local power supplies (battery).

With regards to the mobility of sensor nodes, we classify WSN as static or dynamic [13]. Static WSN consists of stationary components and is initialized for operation with a set-up infrastructure communication phase where the nodes exchange their status information (node location, available memory size, battery matter, etc). In the operational phase, the sensors regularly transfer the sensed (and locally processed) data to the base station. On the other hand, a dynamic WSN is characterized with mobile sensor
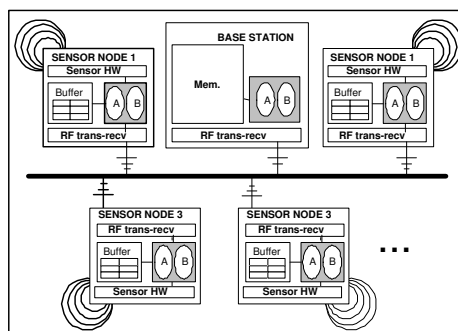

Fig. 1. Example of a Wireless Sensor Network.

nodes and/or base station, with multiple set-up communication phases for updating the nodes on network's status. This paper addresses static WSN, where the models contain a single infrastructure phase followed by an operational phase. Further, our current TLMs are limited to one-hop WSN, with a single base station and multiple sensor instances. A multi-hop topology, on the other hand, contains multiple base stations with broadcasting capabilities. However, our TLM can be extended to support multi-hop WSN by supplying the current base station implementation with (a) a broadcasting method and (b) support for multiple instantiations.

4 WSN Communication Protocol

Regardless of the specific functionality of a WSN, its performance greatly depends on the efficiency of the underlying communication mechanism. Therefore, one of the most important aspects in configuring a WSN is selection of an appropriate communication protocol.

In selecting WSN communication, we must take into account the unique features and application requirements of sensor networks, such as limited memory
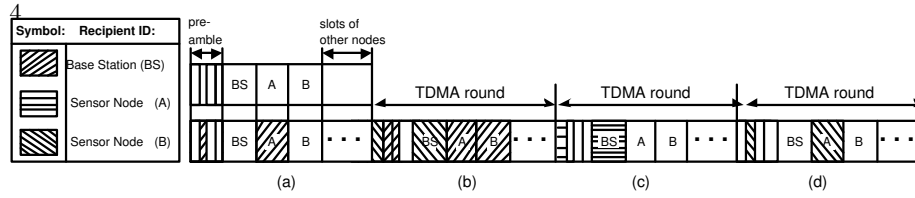
Fig. 2. TDMA protocol. (a) Sensor node A to Base Station transmission, (b) Sensor nodes A and B transmitting to Base Station and Base Station transmitting to Sensor node B, (c) Base Station transmitting to Sensor node A, (d) Sensor node A transmitting to Sensor node B.

and computational capacities, and proneness to node failure. Therefore, traditional end-to-end communication that requires single and double handshake mechanisms are generally not suitable for WSN. Furthermore, due to strict requirements for power conservation and failure recovery, the WSN should not implement protocols with demand-based QoS, large message overheads, and/or long link setup delays. Finally, WNS should tolerate unreliable and faulty transmissions, which makes broadcast communication with best-effort transport and timeout capabilities desirable.

At TL, the broadcast protocols are described within the Data Link Layer that covers the MAC (Media Access Control) and the Physical Layer, as specified in the ISO OSI reference model. With regards to the mode of medium access, we model representatives of protocols with both random access and fixed allocation channel access.

4.1 Time Division Multiple Access (TDMA)

TDMA is a communication protocol that allows multiple transmitters to access a single radio-frequency (RF) channel by dividing channel access into time slots and allocating each to a specific transmitter (fixed allocation medium access). This time separation ensures that multiple users will not experience interference from other simultaneous transmissions and substantially improves the efficiency and quality of wireless communication.

Figure 2 illustrates the TDMA protocol. The figure shows the RF channel timeline divided among N transmitters: we highlight only Sensor Nodes A, B, and a Base Station, for clarity. Each TDMA transmission (in figure: TDMA round) is separated into a single *preamble* slot (with N sub-slots) and N data slots. Each transmitter announces its intent to transmit by broadcasting the recipient's ID during its *preamble* sub-slot. All transmitters scan the *preamble* to identify whether they are someone's recipient. If so, the recipient expects to receive the data during the sender's data slot. On the other hand, if the transmitter does not read its ID in the *preamble*, it will sleep for the duration of N data slots to conserve energy.

For example, in the first TDMA round (Figure 2(a)), *Sensor node A* broadcasts the ID of the recipient (*Base Station*) in its *preamble* sub-slot, and broadcasts the corresponding data during its data slot. Round two (Figure 2(b)) shows multiple receives, as *Sensor node A* and *Sensor node B* send to *Base Station* while *Base Station* sends to *Sensor node B*. Figure 2(c) shows a transfer from

Base Station to *Sensor node A*. Finally, communication between two sensors is shown in Figure 2(d), where *Sensor node A* is sending to *Sensor node B*.

The main weakness of the TDMA protocol is in the amount of wasted bandwidth in a low traffic environment, since the time slot is allocated to a transmitter whether or not the transmission is scheduled. For example, only Figure 2 (b) shows the full utilization of the TDMA round. In contrast, Figure 2 (a), (c) and (d) show $N-1$ data slots left unused since only a single node transmits in each round.

4.2   Carrier Sense Multiple Access (CSMA/CA)

Carrier Sense Multiple Access (CSMA) protocol has a random medium access strategy that is best suited for applications with light traffic load and short delay data exchange, as it allows transmissions as soon as the sender senses a free medium. However, interference occurs if more than one transmitter sense the medium free and decide to transmit at once.

In order to circumvent interference of multiple simultaneous broadcasts, protocol 802.11. uses a *Collision Avoidance* mechanism together with a *Positive Acknowledge* scheme, i.e. CSMA/CA.

Collision Avoidance senses that the medium is free for a specified time (i.e. Distributed Inter Frame Space) before trying to transmit the data. In addition, a Positive Ack scheme is used to assure the transmitted packets are indeed received. If the sender does not receive an acknowledgement of the packet by the receiver, it will retransmit the packet. In case the receiver experienced node failure, the sender will recover by aborting further transmissions after a specified number of retransmission attempts.

Exponential Random Backoff Algorithm is a mechanism to resolve contention for the shared medium access between transmitters. The same algorithm runs on each transmitter independently of each other, generating a random period (M_i) from the *Backoff interval(i)*. M_i is the time transmitter i will wait before sensing the medium again. The waiting time needs to be at least long enough to allow the transmitter to determine whether the medium has already been accessed by another transmitter.

After each unsuccessful attempt to transmit, the transmitter i increases its *Backoff interval(i)* exponentially until it reaches the *Linear interval(i)* (a user defined parameter). After that, the increase continues linearly. As previously stated, each transmitter has only *n_tries* attempts before aborting further transmissions.

5   Transaction Level Modeling of WSN

We model WSN in SystemC [9], which provides module constructs *sc_module* with computational processes (*SC_THREAD*) sensitive to input changes and event occurences (*sc_event*). Using SystemC constructs, we implement sensor nodes as objects of class *SensorNode* while the base station is an object in class *BaseStation*. Every component contains one or more concurrent processes that
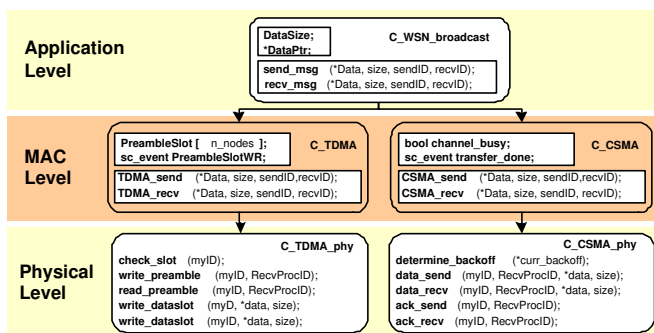
Fig. 3. Overview of our Broadcast Channel Model.

encapsulates a sequential C/C++ code with calls to wireless communication methods.

Our communication methods are modeled based on ISO OSI layering, as shown in Figure 3: at the application level, the processes invoke generic functions *send_msg()* and *recv_msg()* of channel *C_WSN_broadcast* to transfer messages of variable size. In contrast, the MAC layer provides services for fragmenting the message into fixed size packets and forwarding it to either the CSMA/CA (random access) or the TDMA (fixed access) broadcast protocol layers. Furthermore, the MAC layer implements the unreliability of WSN communication as follows: based on the probability of a successful broadcast, the sender's MAC layer randomly determines whether it will forward the packet to lower layers for broadcasting. As a consequence, the broadcasts of the packets discarded by the MAC layer are considered to have failed. The success/fail rate of the broadcast is a user defined parameter (in percentages).

In case of CSMA/CA broadcast, the processes use channel *C_CSMA*, invoking communication routines with built-in timeout and retransmission mechanisms. We use *sc_event transfer_done* to denote the start and end of transaction, and a *channel_busy* variable to detect packet collision. The physical layer primitives of the CSMA (encapsulated in channel *C_CSMA_phy*) model the exponential backoff algorithm and the data byte and acknowledge transfers.

TDMA broadcast, on the other hand, divides the radio frequency into time slots alotted to different transmitters which announce their intent to transmit during their *preamble* sub-slots. Therefore, the model of the TDMA channel at the MAC layer (channel *C_TDMA*) has an array *PreambleSlot*, which the transmitters read on every TDMA round and a *sc_event* that occurs on each write to the *PreambleSlot*. The physical layer of the same protocol (channel *C_TDMA_phy*) contains transfer announcement and data transfer routines.

5.1 TDMA modeling

This section describes the flow of executing a package broadcast with TDMA protocol (as seen in Figure 4(a)). At the sending side, the transmitter waits until its first available *PreambleSlot* sub-slot to announce the transfer (*write_preamble*), then writes into its data slot. The receiver reads the entire *PreambleSlot* array. The index of the array element that contains receiver's ID is identified as its sender. Note that more than one sender can be identified in the same TDMA round. If one or more senders exist, the receiver will read the corresponding data
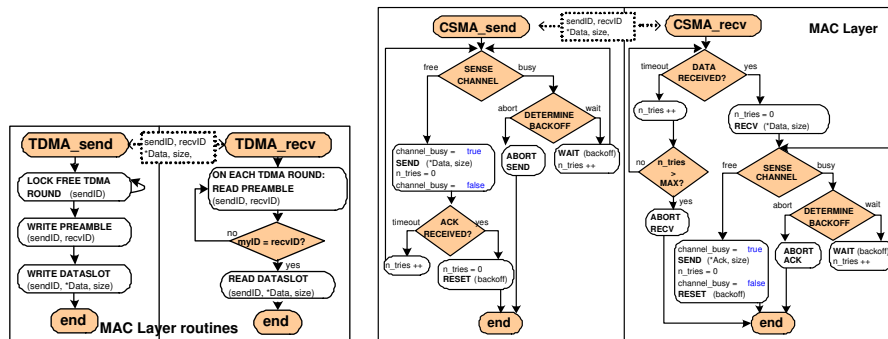
(a) TDMA functionality at the MAC Level.

(b) CSMA/CA func. at the MAC Level.

Fig. 4. TL modeling for broadcast protocols.

slot(s), otherwise the receiver will go to sleep. The sleeping process is modeled with a *wait* for event *PreambleSlotWR*. On the next *write_preamble*, the process returns to the normal operating mode.

5.2 CSMA/CA modeling

The send and receive routines for package transfer with the CSMA/CA protocol are shown in Figure 4(b)).

Here, the sender senses if the medium is busy before starting data transfer by testing the boolean variable *channel_busy*. On each failed attempt to send (i.e. *channel_busy* is *true*), the waiting time before the next medium sensing will increase according to the *Exp backoff* algorithm. The sender attempts transmission for *n_tries* times before aborting. After successful data broadcast, the sender expects an acknowledge (*ack*) packet from the receiver.

The receiver, on the other hand, will sense the channel for BROADCAST_DUR time before aborting transfer due to timeout. However, if the correct recipient ID has been sensed on the medium before timeout, the receiver will accept the broadcasted data and initiate sending of an (*ack*) packet.

Note that the unsuccessful acknowledge is considered a failed attempt of transmission: the receiver has limited number of attempts to transmit an *ack* (up to *n_tries* times) before aborting, and the sender will abort waiting for an *ack* after experiencing timeout at most *n_tries* times.
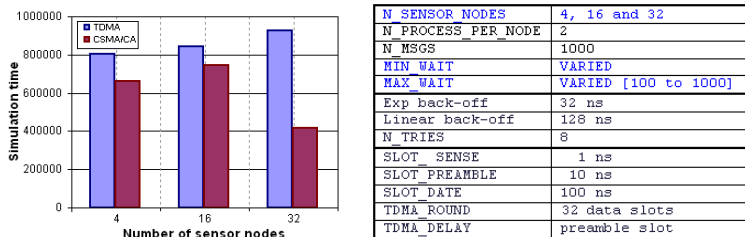
6 Experimental Setup and Analysis

We have modeled the described protocols according to their specification and implemented following a temperature sensoring application [1]. The application senses and logs the temperature value and, if the sensed temperature is beyond the threshold limits (specified by the user), an alarm signal toggles a LED. The platform consists of a base station and N_NODES sensors. The application flow is as follows:

1. Upon starting, the base station enters configuration mode, where the infrastructure is determined and the lower and upper bounds for the normal temperature values are set.

2. Once the threshold values are set, the base station enters operational mode. Each sensor node transmits an average of 16 sensed temperature values ($Avg\ Value$ message) to the base station.
3. Regardless of the sampling period, if the temperature crosses the threshold values, the sensor node transmits a $Critical\ Value$ message.
4. In addition, the base station can query the current average temperature aperiodically and get a reply in the form of a $Avg\ Value$ message.

Figure 1 outlines the WSN implementing the described application. Each network component contains two processes: processes marked with A on Figure 1 denote transfer of *'Avg Value'* messages and processes marked B encapsulate *'Critical Value'* transmissions. The following experiments simulate the wireless communication of this WSN with both CSMA/CA and TDMA protocols.

N_SENSOR_NODES	4, 16 and 32
N_PROCESS_PER_NODE	2
N_MSGS	1000
MIN_WAIT	VARIED
MAX_WAIT	VARIED [100 to 1000]
Exp back-off	32 ns
Linear back-off	128 ns
N_TRIES	8
SLOT_SENSE	1 ns
SLOT_PREAMBLE	10 ns
SLOT_DATE	100 ns
TDMA_ROUND	32 data slots
TDMA_DELAY	preamble slot

(a) Simulated exe. time   (b) WSN configuration param.

Fig. 5. Experiment 1: Comparison of TDMA and CSMA/CA for the same load.

6.1 Comparison of Broadcast Protocols

Unlike CSMA/CA protocol, TDMA does not require message acknowledgements, since the fixed schedule among the nodes prevents medium access collisions. However, by pre-allocating the medium access, a significant time of a node can be wasted on idle waiting for its slot to broadcast. Therefore, **Experiment 1** (Figure 5) compares the CSMA/CA and TDMA communication protocols with regards to (simulated) execution time for 1000 successful broadcast messages.

In Figure 5(a), the x-axis denotes the number of nodes in WSN (4, 16 and 32) and y-axis shows the simulated execution time of the application transmitting 1000 messages from each node. Figure 5(b) presents the full list of configuration parameters for the experiment. As shown in Figure 5(a), even though TDMA protocol needs no retransmissions, broadcasting 1000 messages with TDMA takes more than double the time of CSMA/CA broadcast for the WSN with 32 sensor nodes.

Since CSMA/CA is clearly better suited for the selected WSN model in our experiment set, we will focus on CSMA/CA for the remainder of this paper.
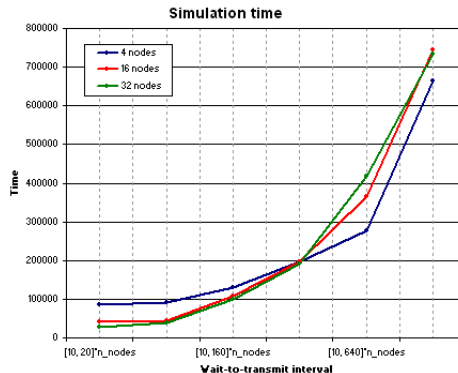
6.2 Scalability of CSMA/CA TLM

Experiment 2 (Figure 6) aims to gauge the scalability of our CSMA/CA communication models by measuring the execution time of sensor networks of different sizes: WSN with 4, 16 and 32 sensor nodes. Each simulation measures

N_SENSOR_NODES	4, 16 and 32
N_PROCESS_PER_NODE	1
N_MSGS	1000
MIN_WAIT	10
MAX_WAIT	VARIED [100 to 1000]
Exp back-off	32 ns
Linear back-off	128 ns
N_TRIES	8

(a) WSN config. parameters   (b) Scalability of CSMA/CA TLM

Fig. 6. Experiment 2: Dependence of simulated execution time of WSN (with CSMA/CA and decreasing contention) to the WSN size.

sensor nodes broadcasting a total of 1000 messages to the base station using CSMA/CA communication protocol. Further, for each of 3 sensor network configurations (shown in Figure 6(a)), we perform 6 simulations with varied waiting time between two consecutive message broadcasts in each node (in Figure 6(b): wait-time interval), based on:

```
wait-time(node_i) = rand_of [10, a* N_SENSOR_NODES], where:
    N_SENSOR_NODES = 4, 16, 32
                a = 20, 40, 160, 320, 640 and 1280
```

The function above de-facto configures the level of contention for the WSN communication medium. The longer the wait-time for each transmitter is, the less contention the WSN will experience. For example, the network consisting of 4 nodes simulates broadcast of 1000 messages with 250 messages sent from each node. Each node is programmed to wait between broadcasting two consecutive messages for a random number of [10, 80], [10, 640], [10, 1280], [10, 2560] and [10, 5120] time units (ns).
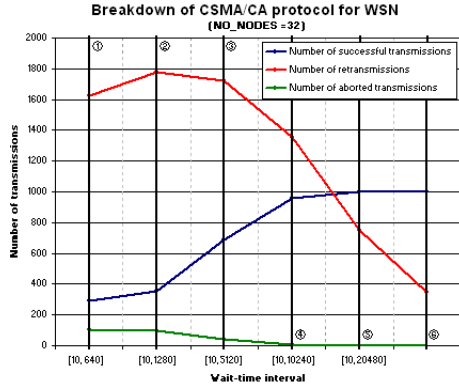
Figure 6 (a and b) demonstrates that our communication model performs equally well for small and large WSN. The graph shows the increase of (simulated) execution time as contention decreases. As expected, the simulated time of system execution is proportional to the waiting time between two consecutive broadcasts in each node. However, it is virtually independent of the number of nodes in the WSN. We conclude that our communication model scales well for large WSN models.

6.3 Analysis of TLM CSMA/CA

In order to validate the correctness of our modeling approach, **Experiment 3** simulates CSMA/CA protocol on a WSN with configuration listed in Figure 6(a), but with its size fixed to 32 nodes.

The simulated results, shown in Figure 7, confirm the expected behavior of the CSMA/CA protocol: with heavy traffic load and high contention, the majority of broadcast messages will experience collisions. This is evident by the number of retransmissions significantly exceeding the number of successful

broadcasts where the wait-to-transmit intervals are ≤ 5120 ns for each node (measure points 1 and 2 in Figure 7).

Fig. 7. Experiment 3: Performance of CSMA/CA under decreasing contention.

As the contention level decreases, less messages get aborted. However, the WSN still experiences large numbers of retransmissions of messages before each successful transmission. This is demonstrated with the retransmission curve peak at measure point 3 (wait-to-transmit interval = 5120 ns). For networks with sporadic message transfers (wait-to-transmit $\geq$ approx. 20000), CSMA/CA efficiently transmits $\geq 98\%$ of the total scheduled messages (measure points 4, 5 and 6). Moreover, after point 5 the number of retransmissions is less than the number of successful transfers, falling to less than a third of the successful transfer rate at point 6.
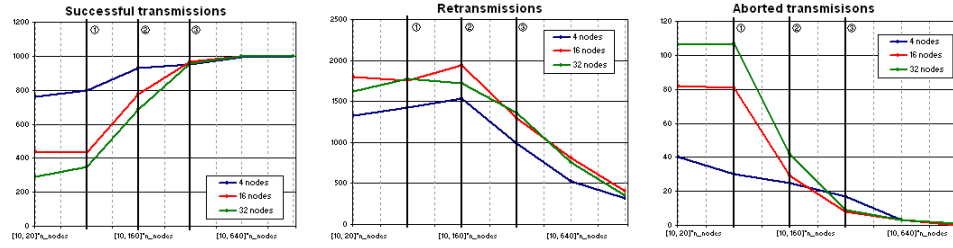
6.4 Configuring an efficient WSN supporting CSMA/CA protocol

The fourth set of experiments (Figure 8 and Figure 9) varies the size of the WSN, the wait-to-transmit interval for each node, and the number of retransmission attempts (n_tries). Experimenting with values of these parameters, the users can determine a WSN configuration that complies with their specified performance metrics, such as rate of successful, retransmitted or aborted transfers.
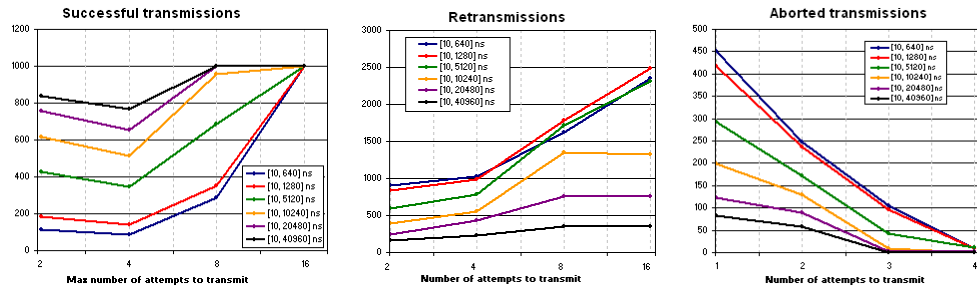
Experiment 4a Varyies the size and wait-to-transfer interval. As the size of WSN increases, the messages need to be transmitted less frequently to decrease contention and avoid collisions. By varying the level of the wait-to-transmit interval, it is possible to identify the threshold size of WSN in which message transfers are of desired/specified efficiency. More specifically, lines (1), (2) and (3) denote characteristic thresholds for our third experiment setup.

In Figure 8(a), (b) and (c), contention levels marked with lines (1) identify the maximal number of aborted transfers. At this contention level, the minimal number of messages is transferred successfully, as every attempt to transmit experiences collision, timeout, and eventually aborts. As contention decreases (from line (1) to (2) in Figure 8(b)), the number of retransmissions is rising until it peaks at line (2). This means that with more slack between consecutive broadcasts, more messages are delivered after repeated retransmission, rather than aborted entirely. Finally, lines (3) denote the acceptable rate of successful transfers and corresponding numbers for retransmissions and aborted messages. This rate will vary with individual WSN designs, according to their performance specifications. For example, a WSN with 80% successful message transfers and $\leq 10\%$ aborted transfers is accomplished with the wait-to-transmit interval is no less than $160*n\_sensor\_nodes$.

(a) (b) (c)

Fig. 8. Experiment 4a: The dependence of transmissions to contention and WSN size. (a) Successful transmisisons, (b) Retransmissions and (c) Aborted transmissions.

(a) (b) (c)

Fig. 9. Experiment 4b: The dependence of transmissions to retransmission attempts and WSN size. (a) Successful transmisisons, (b) Retransmissions and (c) Aborted transmissions.

Experiment 4b: In addition, by experimenting with the maximal number of retransmission attempts before aborting transmissions (n_tries), we can determine the size and/or wait-time interval for most efficient message transfers. The rates of successful transmissions, retransmissions and aborted transmissions, when varying values for n_tries, are shown in Figure 9 (a), (b), (c), respectively. This experiment demonstrates that the value of n_tries for our WSN configuration is at most 8 (for high contention traffic) and at least 4 (for low levels of contention). This is evident from Figure 9 (a), where the change of n_tries from 4 to 8 yields the sharpest rise in the number of successful transfers. During low contention and for $n\_tries \geq 8$, this parameter has little to no effect on the overall WSN performance, as transfers experience a negligible number of retransmissions.

N_SENSOR_NODES	4, 16 and 32
N_PROCESS_PER_NODE	2
N_MSGS	1000
MIN_WAIT	10
MAX_WAIT	VARIED [100 to 1000]
Exp back-off	32 ns
Linear back-off	128 ns
N_TRIES	2, 4, 8 and 16

Fig. 10. Configuration parameters for experimental setup 4 (a) and (b).

7 Conclusion

We have demonstrated that SystemC TLM can support WSN models. In particular, we describe the TLM of two popular broadcast communication protocols: TDMA and CSMA/CA. A detailed analysis of the CSMA/CA protocol confirms the expected performance of CSMA/CA under varied contention levels and, therefore, validates our modeling approach.

In addition, we demonstrate that applying TLM principles to WSNs is an efficient way to configure WSN systems. By varying the parameters of WSN TLMs, we can quickly explore and identify the WSN configuration that will yield optimal results with regards to the given environment and specified performance metrics.

References

1. Temperature sensor application using st lm135. *http://www.st.com/stonline/products/literature/an/11890.htm*, 2007.
2. E. Alessio, F. Fummi, D. Quaglia, and M. Turolla. Modeling and simulation alternatives for the design of networked embedded systems. *DATE, Nice*, 2007.
3. Alvise Bonivento, Luca Carloni, and Alberto Sangiovanni-Vincentelli. Platform based design of wireless sensor networks for industrial applications. *DATE, Munich*, March 2006.
4. Victor Cionca, Thomas Newe, and Vasile Dadrlat. TDMA protocol requirements for wireless sensor networks. *Proceedings of the 2nd International Conference on Sensor Technologies and Applications*, pages 30–35, 2008.
5. A. Clouard, K. Jain, F. Ghenassia, L. Maillet-Contoz, and J.P. Strassen. *Using Transactional Level Models in a SoC Design Flow.* Kluwer Academic Publishers, 2003.
6. M. Conti and D. Moretti. System level analysis of the bluetooth standard. *DATE*, pages 118–123, March 2005.
7. D. D. Gajski, S. Abdi, and I. Viskic. Model based synthesis of embedded software. *SEUS*, 2008.
8. Frank Ghenassia. *TL Modeling with Systemc: TLM Concepts and Applications for Embedded Systems.* Springer-Verlag New York, USA, 2006.
9. Thorsten Grötker, Stan Liao, Grant Martin, and Stuart Swan. *System Design with SystemC.* 2002.
10. M.M.R. Mozumdar, F. Gregoretti, L. Lavagno, L. Vanzago, and S. Olivieri. A framework for modeling, simulation and automatic code generation of sensor network application. *SECON, San Francisco*, 2008.
11. G. Schirner and R. Doemer. Quantitative analysis of the speed/accuracy trade-off in transaction level modeling. *In ACM Transactions on Embedded Computing Systems (TECS)*, 2008.
12. M. Singh and V. K. Prasanna. A hierarchical model for distributed collaborative computation in wireless sensor networks. *IPDPS*, 2003.
13. Sameer Tilak, Nael B. Abu-Ghazaleh, and Wendi Heinzelman. A taxonomy of wireless micro-sensor network models. *ACM SIGMOBILE Mobile Computing and Communications Review archive*, pages 28–36, 2002.
14. Nicholas H. Zamora, Jung-Chun Kao, and Radu Marculescu. Distributed power-management techniques for wireless network video systems. *DATE, Nice*, 2007.