2008 International Conference on Information Processing in Sensor Networks

On the Design of MAC protocols for Low-Latency Hard Real-Time Discrete Control Applications Over 802.15.4 Hardware

Krishna Kant Chintalapudi
Robert Bosch Research and Technology Center
4001 Miranda Ave,Palo Alto, CA, USA

Lakshmi Venkatraman
Robert Bosch Research and Technology Center
4001 Miranda Ave,Palo Alto, CA, USA

Abstract

Discrete event control loops in modern-day machines often comprise a large number of sensors (50-200) reporting to a controller. Many discrete control applications must cater to hard real-time *requirements* i.e., *sensors must communicate the occurrence of critical events to the controller within a real-time deadline (usually 5-50ms) specified by the control system's design requirements. Messages reached after this deadline are considered lost. In the event of traffic bursts where several sensors may attempt to communicate with the PLC at the same time, messages from* all *the sensors must reach within the specified deadline. The metric for performance in such systems is then the probability that a message from all the sensors succeeds in being received at the controller within this deadline. Further, for such solutions to be viable, the sensors must last for several years without requiring change of batteries. In this paper we examine the potential for using 802.15.4 based radios for wireless sensing in low-latency hard real-time discrete event control applications.*

1. Introduction

The operation of several modern-day control systems such as computer numerically controlled (CNC) machines, vehicles, manufacturing robot arrays *etc.* are based on discrete event control. In a discrete event control system, sensors convey the occurrence of critical events (rather than sampled values of continuous physical phenomena) to a controller. The controller then, based on these inputs from the sensors, induces the necessary actuation to control the system.

For example, proximity sensors at a welding unit may detect and notify the arrival of a new work-piece to the controller. The controller may then induce a robotic arm to pick the piece up and place it on the welding platform. In another example, some sensors may detect a possible oil/gas leak and upon notification, the controller may require shutting down some sections of the system.

The typical modern-day discrete event control-based CNC machine or a vehicle spans 3-15 mts along its largest dimension (controller to sensors) and houses 50-200 sensors. For a large number of discrete event control-based systems the control loop must cater to *hard real-time* requirements *i.e.,* the sensing (detection of the event), communication (sensor to controller and controller to actuator) and actuation must occur within a pre-specified deadline. Given fixed sensing and actuation delays, such deadlines can usually be translated into communication delay deadlines. A message that does not reach its destination before this deadline may cause the machine to go into an error condition that requires its temporary halting or resetting. In machines today, sensors and actuators communicate to the controller via cables and cater to hard real-time communication latencies ranging from 5-50ms depending on the specifics of the machine.

Inherent to most discrete event control systems is the unpredictable bursty nature of the traffic *i.e.,* it is hard to predict when, how many, or which sensors will be triggered to communicate at the same time to the controller. This is because the communication is primarily event-driven and it is often impossible to predict the times and nature of occurrence of external events. In general, traffic bursts are common in most discrete event systems because, i) a single event may lead to several sensors triggering at the same time and ii) more than one event may occur at the same time (or "close enough times"). In the rest of the paper we shall refer to such event driven bursts as *sensor bursts*. In the event of a sensor burst, then, messages from *all* the sensors must reach the controller within the specified deadline since the controller can take appropriate action only upon receiving all the inputs. Failure of receipt of a message from even one sensor may lead to unpredictable failures in the system forcing it into an error recovery state.

Consider a production machine of 100 sensors that produces a finished product every 10sec. Suppose that the manufacturing of each product triggers an average of about 50 sensor burst events. A communication failure probability of 1ppm (1 in 10^6) translates to an expected time

978-0-7695-3157-1/08 $25.00 © 2008 IEEE
DOI 10.1109/IPSN.2008.39

between failures of about 2 days ($\frac{10^6}{86400\frac{50}{10}}$). A failure will lead to a decreased production throughput and hence significant financial losses. Probability of failure (or in some cases expected time between failures) is thus, perhaps the most important performance measure in most production systems. Failures in machines such as vehicles may lead to graver consequences including loss of life and property. Not surprisingly, in modern-day machines, sensor-actuator-controller communication is conducted via communication cables (buses or wires carrying analogue signals).

The design of present day machines requires careful deliberation for routing the cables from various locations within the machine to the controller. Eliminating the cables can not only provide the potential of enabling compact and simple mechanical designs by avoiding cumbersome cabling, but also provide additional benefits in terms of ease of installation and maintenance. Furthermore, cables are often subject to wear and tear, especially when they are drawn from moving parts within the machine and require frequent maintenance. Each maintenance cycle translates to decreased usage and increased maintenance costs. Elimination of cable wear and tear events translates into less maintenance expenditure and fewer down periods. Further, the possibility of wireless sensors encourages the design of systems with a larger number of sensing points for more efficient control.

For most machines, actuators have very high power requirements. The actuators are usually expected to induce mechanical operations such as lifting a part or turning a high speed drill. Actuators thus, cannot be un-tethered and require power-cables to be routed to them. Making the controller-actuator communication wireless does not offer significant advantages since controller-actuator communication cables can be "bundled" up along with the power-lines without significant overhead. Sensors, on the other hand, have modest power requirements, and can be made "completely wireless", operating only on batteries. Replacing batteries in hundreds of sensors in a machine, however, can be a time-consuming, labor-intensive job. Frequent battery changes resulting in maintenance downtime can offset the gains offered by a wireless sensing system. Considering that the lifetime of typical manufacturing machines or vehicles is 10 years, it will be expected that wireless sensors must operate for at least a few years on batteries before requiring battery replacement.

There are thus, two metrics that completely capture the performance of a communication protocol for low-latency hard real-time discrete event systems:

Probability of communication error - probability that at least one sensor (among all sensors attempting to communicate in the event of a sensor burst) does not succeed in transmitting its message to the controller within the deadline and,

Longevity of the system - the average life expectancy of wireless sensor nodes.

Communication cable-based solutions used in modern-day machines typically provide error rates of 1ppm (1 in 10^6) or less. A wireless system that replaces an existing system only to become a performance bottleneck will not be accepted as a viable solution. *In this paper we ask the question, "Can 802.15.4 based radios be used for wireless sensing in low-latency hard real-time discrete control systems?"* Rather than taking a theoretical approach, in this paper, our approach is to consider a commonly used off-the-shelf radio - CC2420, (considering the acceptance it has received in the community) and attempt to design MAC protocols that can provide 1ppm or less error probabilities within deadlines ranging in 5-50ms. Our hope is that this exercise will lead to both qualitative and quantitative generalizations that can be used across other low-power radios based on the 802.15.4 standard as well. Considering that in the typical scenarios all sensors are within 3-15 mts of the controller, in this paper, we restrict ourselves to single-hop MAC solutions. Given that MAC protocols are broadly classified into contention-free and contention-based protocols, in this paper we take a strawman approach to designing and analyzing them systematically.

We start in Section 3 by carefully examining the sources of latency involved in transmitting a packet over the CC2420 radio. We believe that some of the bottlenecks found in CC2420 are common to most low-power radio platforms available today. Based on the analysis in Section 3, in Section 4 we propose a novel technique called *transmission pipelining* that can be used in any time-slotted MAC protocol to increase channel utilization during a sensor burst. In Section 5 we examine the performance of contention-free MAC schemes such as TDMA. One novel possibility we consider in this paper is the use of multiple transceivers at the controller (this can be done since the controller is not power constrained). This will allow packets from several sensors to be received at the same time over different channels and thus help relieve the bandwidth bottleneck at the controller during a sensor burst.

While the sensor traffic is unpredictable, it is often possible to assume a maximum possible burst size for a given system. For example, in a 200 sensor system, typical number of sensors being triggered at the same time will lie in the range of 2-20 sensors. In Section 6 we ask the question, "If we knew the maximum possible burst size, can we design contention-based MAC protocols that can perform better than contention-free protocols?"

We believe that the paper provides a good understanding of the practical limitations and the performance we can expect out of 802.15.4 based radios for hard real-time discrete control applications.

2. Related Work

The body of work on low-power MAC protocols for wireless sensor networks is rather intimidating. Consequently, we only provide a very brief survey of existing MAC mechanisms that we believe to be representative of existing literature. S-MAC [1] relies on locally (topologically) synchronizing the sleep and wake up schedules of sensor nodes to increase network longevity. TMAC [2] enhances S-MAC's performance under variable load traffic. DSMAC [3] allows for an adaptive duty-cycling window to cater to delay-sensitive applications. WiseMAC [4] uses spatial TDMA and np-CSMA, where nodes sample the channel periodically to sniff the channel based on wake up schedules that are offset to avoid collisions. TRAMA [5] also relies on spa-

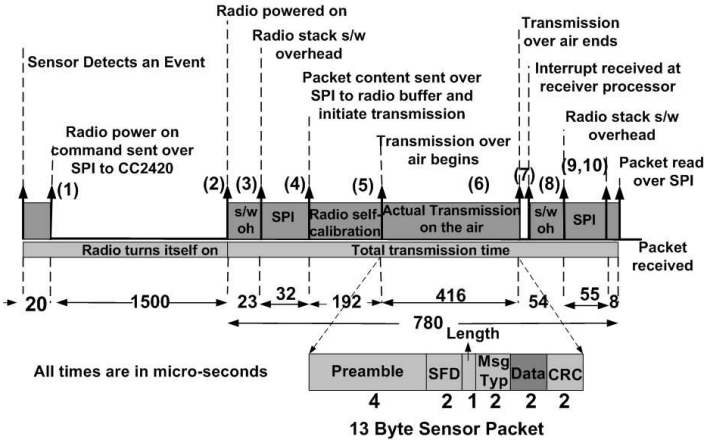

Figure 1. An outline of events that occur between a sensory event at the wireless sensor until the packet reception at the controller over a CC2420 radio.

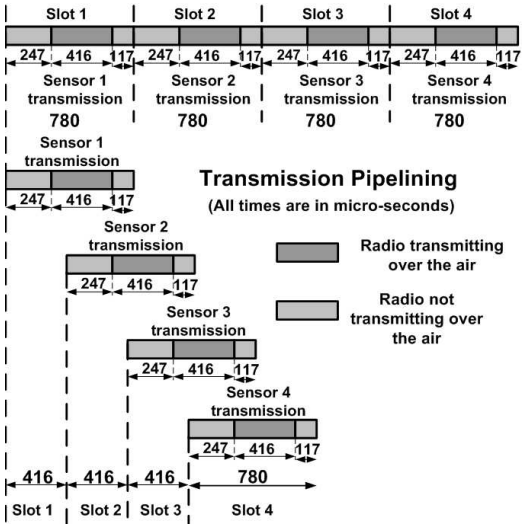

Figure 2. An example illustrating transmission pipelining.

tial TDMA but proposes an additional mechanism to avoid energy wastage due to the problem of hidden terminals not addressed in WiseMAC. DMAC [6] is a MAC designed specifically for networks where sensors form a tree topology to transmit data to a base station. DMAC provides both low-latencies and longevity by ensuring skewed schedules among various levels in the trees which enable children nodes to transmit exactly when parent nodes are ready to listen. SIFT [7] is a contention based MAC that uses a non-uniform probability distribution to pick transmission slots and exponentially adapts the transmission probabilities on noticing idle slots. PTDMA [8] attempts to seamlessly transition between TDMA and CSMA by assigning probabilistic ownerships to slots that are adjusted based on the number of transmitters. Finally, ZMAC [9] proposes a scheme that transitions between TDMA and CSMA seamlessly but is suitable for multi-hop networks unlike PTDMA that was designed with single hop networks in mind.

3. Anatomy of Packet Transmission

Perhaps the most fundamental step towards building a low-latency MAC is to minimize the latency of transmission of a single packet. Consequently, in this section, we carefully quantify the latency involved in the transmission of a single packet while striving to minimize it. Our entire discussion in this section is based on actual careful measurements performed on the CC2420 radio using an oscilloscope. We believe that while the exact numbers may depend on the radio and the specifics of the software being used, the nature of the analysis itself and the lessons learnt will be useful for most current day low-power radios.

3.1 Contents of a sensor packet

Smaller the packet, the smaller the time to transmit. Further, a smaller packet also has a greater chance of succeeding in the channel. For example, at a channel bit-error-rate of 10^{-3}, the packet success rate (PSR) for an 11 byte packet is 91.5% ($(1-BER)^{8\times 11}$) whereas that for a 20 byte packet is only 85.2%. *In this section we start by asking the question, "How small can a sensor packet possibly be?"*

The radio first starts by transmitting a 4 byte *preamble* over the air that is used by the receiver for synchronization. The preamble is followed by the 2-byte *Start of Frame Delimiter* (SFD) field which is required to ensure that the receiver is indeed receiving a valid packet. The CC2420 radio also requires a *length field*, so that the radio hardware can know how many bytes to receive.

A *network identifier* is often necessary so that packet transmissions from neighboring machines can be rejected. One way to avoid the use of network ID is to use the SFD bytes as the network identifier. This will help in not only reducing the packet size, but also speed up the network stack since packets will be rejected based on the SFD in the radio hardware itself. CC2420 allows the programmer to configure the SFD. The *destination* and *source* identifiers are usually the next fields used in typical packets. While the source identifier is necessary to identify which sensor transmitted the packet, the destination for a sensor packet is always the controller and thus not required.

A two-byte *data field* usually is sufficient for most sensors used in discrete event control systems. Finally a two byte *CRC* is computed by the CC2420 radio hardware to ensure the integrity of the content of the packet. Thus, we are left with a 13 byte sensor packet (Figure 1). In general however, 9 bytes is the absolute minimum that a packet must have including the preamble, SFD, length and CRC. In the rest of the paper we consider all other fields as payload.

3.2 Latency of packet transmission

In general the radio software stack can be implemented in several different ways depending on the generality and flexibility desired. The more layers and function calls in the software the more the execution time. We found that every few lines of code removed can result in reduction of several tens μsec of latency. Our implementation of the radio stack

on MSP430 was carefully optimized to a bare minimum specifically for our application in order to minimize latency at the cost of possible loss of generality.

The communication between the CC2420 radio and the MSP430 is via an SPI interface (this is common in many off-the-shelf radios available today). The CC2420 acts as a slave and expects commands to be sent from MSP430 over the SPI interface for operations such as turning the radio on, switching the channel, initiation of transmission, and reading the CC2420 radio buffer *etc.* By performing several measurements on our stack we found that the latency involved in writing/reading n bytes over the SPI to the radio is roughly given by $(17+3n)\mu$sec[1]. Sending a command such as "power on" over the SPI, thus, takes about 20μsec. Setting a register on the CC2420 for configuration such as changing power level, requires about 23μsec since two bytes are transmitted - the address of the register and the value. For a 13 byte packet, while transmitting, we write 5 bytes over the SPI - one length and four data bytes of payload (CRC, SFD and preamble are all generated by CC2420 radio hardware); thus, this takes 32μsec. While reading, first, the length field is read to determine the size of the packet (this takes 20μsec) and then the rest of the packet, 6 bytes (four data bytes and 2 byte CRC), is read. This requires 35μsec. The total time is thus, 55μsec to read a 13 byte packet.

Figure 1 depicts the sequence of various significant events that occur during the transmission and reception of a 13 byte packet with approximate processing times. The sequence of events below lists in detail the delays seen on our sensor node at every stage of packet transmission for a data payload of d (total $d+9$ bytes). 1) Micro-controller sends a command over the SPI interface to turn on the radio - 20μsec, 2) The radio wakes up - variable delay up to 1.5msec, 3) A packet is created and the necessary software state in the communication stack is updated - 23μsec, 4) Packet is sent to radio transmit buffer over SPI - 17+3(d+1) μsec, 5) CC2420 calibrates itself for transmission - 192 μsec, 6) Radio transmits packet over the air - $32(d+9)\mu$sec, 7) Receiver gets a packet interrupt and is notified of the presence of a packet in the receive buffer - 8 μsec, 8) Communication stack overhead at receiver - 54 μsec, 9) Receiver reads the length byte of received packet over SPI - 20μsec, 10) Receiver reads packet contents over SPI - 17 + 3(d+2) (2 bytes extra for reading CRC), 11) Receiver updates software states in the communication stack and hands over packet to application - 8μsec. In general, thus, the application-to-application transmission time excluding variable wakeup delay can be calculated in μsec as[2],

$$\tau_s = 628 + 38d. \quad (1)$$

4. Transmission pipelining

In this section we describe a novel technique - transmission pipelining, that can be used while designing MAC protocols to enhance the channel throughput in the event of a traffic burst.

To explain the essence of transmission pipelining we take the help of a simple example. In Figure 2 four nodes have data to transmit 13 byte packets at the same time. Suppose that we have an arbitrary slotted - MAC protocol, where time is divided into slots (*e.g.,* slotted ALOHA, TDMA *etc.*) and each node is allowed to transmit only at the beginning of a slot. If each of the sensor nodes were scheduled to transmit exactly one after the other, each slot would be 780μsec long and they would require 3.1 ms to transmit the data.

An examination of Figure 1 however, reveals that while transmitting the 13 byte packet takes 780 μsec, the time when bits are actually being transmitted over the channel is only 416 μsec. In other words almost 50% of the total application-to-application transmission time is essentially processing overheads at the sensor nodes. Transmission pipelining is based on the idea that another node could potentially use these idle processing times to transmit data. Thus, as shown in Figure 2, the slot width can actually be reduced to 416μsec. Node 2 initiates its transmission 416μsec after node 1 instead of after 780μsec. The reason transmission from node 2 will succeed is because by the time node 2 starts actual transmission over the air, node 1 will have finished transmitting over the air.

In transmission pipelining, transmissions from various sensor nodes are overlapped in a manner so as to ensure that their periods of transmission over the air are placed immediately one after the other. Thus, while one sensor node is transmitting bits over the air, another node is processing for getting ready to transmit over the air - hence the choice of the name. Transmission pipelining can significantly improve channel utilization in the wake of a sensor traffic burst. This technique is not specific to CC2420, but can be used in almost any radio that has significant software/hardware overheads in the packet transmission process.

4.1 Implementation issues

In a practical implementation, however, two factors limited us from achieving 100% channel utilization - i) time synchronization errors among nodes and ii) receiver's inability to receive packets immediately one after the other.

In any slotted scheme, nodes must be time-synchronized to have a common notion of time and the slot boundaries. Any practical scheme must account for time-synchronization errors and provide guard time around each slot. For our implementation we used the 32Khz real-time clock in MSP430 for time-keeping and ensured that time-synchronization error between any two sensor nodes was less than 2 ticks - *i.e.,* 64μsec. This widened each slot by 64μsec.

As depicted in Figure 2, the radio transmission of one node starts as soon as the radio transmission of another ends. This means that the transceiver at the controller starts to receive a new packet as soon as it has finished receiving the previous packet. In practice we found that after receiving one packet CC2420 was unable to receive another packet immediately (CC2420 would not detect the packet at all). Several trial and error measurements indi-

[1]We used MSP430's 5Mhz clock for the SPI

[2]Additional operations such as setting transmission power-level or setting transmission will incur an additional 23μsec each.

cated that it took the radio about 96 μsec before it could reliably receive another radio transmission. We are not entirely sure as to the reason behind this phenomenon, however, this required us to allow an extra 96μsec in addition to the time-synchronization error.

Thus a total guard-time of 96+64=160μsec was added to every transmission pipelined slot and 64μsec to every non-transmission pipelined slot.

5. Contention Free Protocols

Perhaps the most popular choice for MAC protocols when guaranteed performance is desired and traffic bursts are possible, has been to preclude the possibility of collisions *i.e.,* use contention-free protocols (*e.g.,* TDMA). The entire class of contention-free MACs can be divided into two categories, *reservation-based* and *round-robin*.

In a reservation-based MAC, nodes transmit reservation requests to obtain exclusive rights for transmitting data. In a round-robin mechanism, nodes simply take turns to transmit in a pre-determined sequence (*e.g.,* TDMA). A reservation-based MAC is suitable when the amount of data to be transmitted is large enough to justify the overhead of the reservation request mechanism. In our application where a sensor typically transmits two bytes of data, using an explicit transmission reservation request will prove wasteful. Consequently, we limit ourselves to exploring round-robin MAC mechanisms where sensors take turns to transmit. In this section we shall take a strawman approach to designing the most efficient round-robin MAC protocol.

5.1 The Basic TDMA (T1)

We start our design with perhaps the most basic round-robin mechanism, namely, the TDMA MAC. As depicted in Figure 3, time is divided into frames, each comprising n time slots where n is the number sensors in the system. Each time-slot is intended for one transmission from a sensor and then transmission of an acknowledgement packet from the controller that notifies the sensor of the packet's receipt. Each of the n sensors is assigned a unique time-slot for transmission in which it is allowed to transmit. The acknowledgement packet from the controller requires neither the the source nor the data fields since no other node in the system will transmit during this time. The acknowledgement packet, as depicted in Figure 3 is thus only 9 bytes long. As seen in Figure 3 the duration of each time-slot is about 1472μsec including the 64μsec intended to cope with time-synchronization errors. For a 200 sensor node system thus, the duration of each frame is 280ms. This means that a sensor may have to wait a worst case of 280ms to for transmitting even once. T1 is thus clearly not suitable for our target applications that desire latencies in the range of 5-50ms.

5.2 Transmission Pipelined TDMA (T2)

Following the discussion in section 4 it is clear that the performance of T1 can be significantly improved by using transmission pipelining. Figure 3 depicts a single frame of TDMA MAC - T2. Each frame of the T2 MAC comprises of n consecutive transmission pipelined times-slots followed by a single acknowledgement packet from the controller that acknowledges all packets successfully received in the frame. The acknowledgement packet comprises an n-bit map in its payload. 1 in the k^{th} position indicates that a packet was successfully received in the k^{th} time slot in the frame while a 0 indicates that a packet was not successfully received in the k^{th} slot. For example, in an 8 sensor system an acknowledgement of 11000101 means that the controller received packet successfully in slots 1,2,6 and 8.

Each frame in T2 has $n-1$ transmission pipelined slots each 576 μsec long. The n^{th} slot cannot be transmission pipelined since the acknowledgement packet can only be transmitted after the last packet has been received. Thus, the n^{th} slot is $780 + 64$ μsec long (64μsec to account for time-synchronization errors). The acknowledgement packet does not require a source field (as in case of MAC T1) and its data field is $\lceil \frac{n}{8} \rceil$ bytes long. The duration of the acknowledgement slot can be calculated using equation 1 as $628 + 38\lceil \frac{n}{8} \rceil$. The duration of a frame in T2 thus becomes, $1472 + (n-1)576 + 38\lceil \frac{n}{8} \rceil$ μsec. For a 200 sensor nodes this evaluates to about 118 μsec. While this is almost 40% of the frame duration of scheme T1, it is still much larger than the latencies desired for the target applications. This indicates that even the most efficient TDMA implementation cannot be used over CC2420 for our target applications.

5.3 Transmission Pipelined FTDMA

The primary bottleneck in both TDMA schemes T1 and T2 is the bandwidth at the controller. One way to relieve this bottleneck is to equip the controller with several transceivers so that multiple packets from various sensors can be received at the same time over different channels. For example, if we equip the controller with m transceivers, m different sensors could potentially transmit packets to the controller at the same time over different channels. We can now extend our design of the time division multiple access MAC to a frequency-time division multiple access (FTDMA) MAC where each sensor is assigned a unique time-frequency slot within a frame (see Figure 3) to transmit. The number of time slots in the FTDMA frame is thus given by $s = \lceil \frac{n}{m} \rceil$.

At the end of the sensor transmissions, the controller transmits m acknowledgements simultaneously, one over each of its transceivers (see Figure 3). Each acknowledgement consists of a s-bit map (similar to the n-bit map in MAC T2), where a 1 in the k^{th} bit indicates that a packet was successfully received in the k^{th} time-slot over that channel and 0 otherwise. The number of data bytes in the acknowledgement packet is thus given by $\lceil \frac{s}{8} \rceil$ and its duration is given by $628 + 38\lceil \frac{s}{8} \rceil$ μsec. The frame duration of the transmission pipelined FTDMA MAC can now be calculated as, $1472 + (s-1)576 + 38\lceil \frac{s}{8} \rceil$ μsec. Table 1 depicts the frame durations for various number values of m and n. Table 1 provides a clear idea of how many transceivers at will be required to meet 5-50ms deadline for various number of sensor nodes in the system if the channel had no packet losses.

In real environments however, packet losses are quite common. These may occur due to temporary fading in the channel (given that the system was initially configured

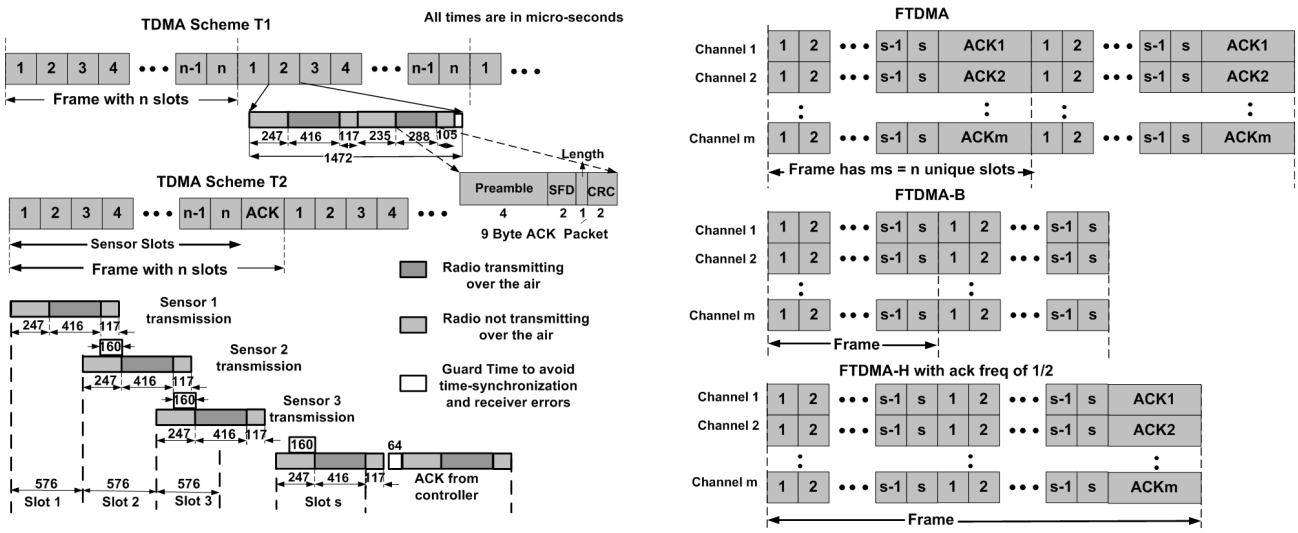

Figure 3. Various contention-free MACs, TDMA, Transmission pipelined TDMA, FTDMA, FTDMA-B and FTDMA-H

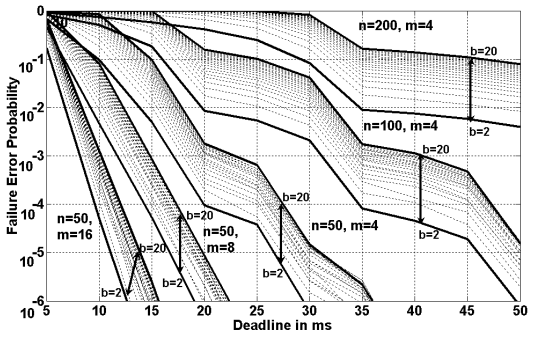

Figure 4. Failure Probability of FTDMA for $p = 0.99$

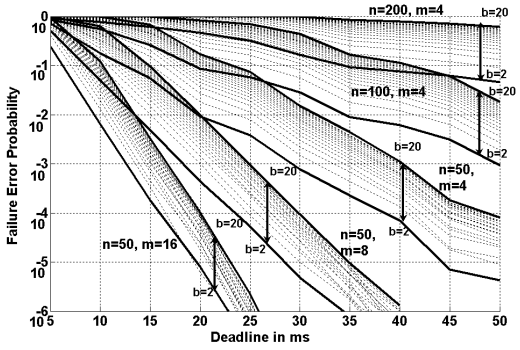

Figure 5. Failure Probability of FTDMA for $p = 0.9$

Table 1. Frame Durations for Transmission Pipelined FTDMA

n	$m = 2$	$m = 4$	$m = 8$	$m = 16$
50	16ms	8.5ms	5ms	3.3ms
100	30.5ms	16ms	8.5ms	5.0ms
200	59.6ms	30.5ms	16ms	8.5ms

such that all sensor-controller links had a "sufficient SNR margin") or due to collisions from transmissions from other co-existing systems. In the event of a packet loss, the sensor must re-transmit the packet in its allocated slot in the next frame. In a real system thus, several frames may be required to successfully transmit a packet to the controller.

Table 2. Number of retransmission required for 1ppm error

b, p	No of transmissions (frames) needed
p=0.999, $b \leq 1000$	3
p=0.99, $b \leq 100$	4
p=0.9, $2 \leq b \leq 10$	7
p=0.9, $10 < b \leq 100$	8

In the event of a sensor burst, as discussed in section 1, transmissions from *all* the sensors must reach the controller within the deadline. The error probability is defined as the probability that at least one sensor fails in transmitting its packet successfully within the given deadline. Given that b sensors attempt to transmit at the same time and that the channel's instantaneous packet success rate is p (including the possibility of collisions and fading[3] *etc.*), the probability that all b sensors will succeed within f frames is given by $1-(1-(1-p)^f)^b$. Thus, the minimum number of frames that need to be accommodated within the deadline to guarantee a error probability of ϵ is given by, $\lceil \frac{\log\left(1-e^{\frac{1}{b}\log(1-\epsilon)}\right)}{\log(1-p)} \rceil$.

Table 2 provides the number of frames required to attain a 1ppm error rate for various values of b and p. As seen from Table 2, while for good links only 3 frames are

[3]To circumvent the effects of temporary fading, it is crucial that frequency hopping be used in re-transmissions. Simple deterministic schemes such as all sensors increasing their channel number by a fixed number may be sufficient. Since sensor nodes are placed within 3-15 mts of the controller, in most situations we expect most links to be operating near the $p = 0.99$ regime. Even if temporarily a link fades, a change of channel in the next frame will probably provide a good link for the packet to succeed.

required to guarantee an error probability of 1ppm, as the link degenerates, the number of frames required rises very rapidly. In fact, it is quite clear from Tables 1 and 2 that for channel conditions with $p = 90\%$ even with $m = 16$ transceivers (the maximum possible in 802.15.4) we cannot achieve 1ppm error for a 200 sensor node system

5.4 Performance of FTDMA MAC

To evaluate the performance of FTDMA we used a home-grown simulator. Figures 4 and 5 depict the probability of error of FTDMA for various values of $m = \{4, 8, 16\}$, $b = \{2, \cdots, 20\}$, $n = \{50, 100, 200\}$ and $p = 0.99, 0.9$ for deadlines in the range of 5-50ms. Each data point in Figures 4 and 5 was obtained over 10^7 sensor burst events. This rather large number of simulations was required to capture events that are as rare as 1ppm. Further, we deducted 1.5ms from the deadline to account for the time required to wake the radio up.

One fact that we observed is that the performance of the MAC depends only on the ratio $\frac{n}{m}$. In other words the performance of a 100 node system with 4 transceivers at the controller will be almost the same as that of a 200 nodes system with 8 transceivers at the controller. This should not come as a surprise since the duration of a frame depends significantly only on the ratio $\frac{n}{m}$. While there is a dependence of the failure probability on the sensor burst size (b) (for example the probability that all 4 sensors succeed will be less than the probability that 2 sensors succeed), the dependence is not "significant".

What Figure 4 indicates is that a system with 200 sensors will require at least more than 8 transceivers at the controller to provide a 1ppm error guarantee in a fairly typical channel ($p = 99\%$) under 50ms. Further, even small machines with 50 sensors cannot be engineered with 10ms or less using 802.15.4 even if all the 16 channel available were used in parallel to ensure 1ppm error probability. As expected in Figure 5 for $p = 90\%$, it is significantly harder to achieve higher than 1ppm level of error probabilities. In fact, even with all 16 transceivers at the controller one can only cater to systems with 25ms deadlines.

5.5 Longevity Assessment

To ensure that minimum power is consumed, the node radios must be powered off during times of inactivity. There are two major factors that determine the power consumption in a time-slotted MAC.

Event Frequency : the frequency of occurrence of events that trigger the sensors to transmit packets, and,

Time Synchronization : the periodic receipt of time synchronization beacons from the controller to maintain time slots.

We shall consider each of these sources of power consumption systematically.

Consumption due to events When an event occurs, a node may re-transmit several times. For each transmission, the sensor will transmit once and receive once. In between re-transmissions, the CC2420 radio can either be put in idle mode or powered down. Each time the radio is powered up, it draws a startup current of about 15mA for about 500μsec or about 7.5μmAsec. If however, it is set to the idle mode between re-transmissions, it will continuously draw 426μA during the entire period it is on. Based on these calculations we found that the best strategy is to put the radio in idle mode between re-transmissions but power it off completely between two events *i.e.,* after receiving an acknowledgement from the controller.

In general *the average number of re-transmissions determines this part of the power consumption.* The average number of transmissions required by sensor node to succeed in a channel with success rate of p is given by $\frac{1}{p}$. Given that acknowledgements may be lost in the channel with a probability p[4], the average number of times a sensor node will re-transmit is given by $\frac{1}{p^2}$. One optimization that can be done to avoid the effect of lost acknowledgements is that after receiving a packet from a sensor, the controller continues to acknowledge it for a few successive frames irrespective of whether or not it received a packet in the current frame. Under this scheme it is trivial to show that the expected number of times a sensor will transmit is given by $\frac{2}{p}$.

The time a radio should be in transmit mode for transmitting is given by 780μsec, while the time it must be in receive mode to receive the acknowledgement packet is determined by the duration of the acknowledgement slot. Knowing that the CC2420 draws 17.4mA in transmit mode and 19.7mA in receive mode we can compute the energy drawn per frame. Considering that the average number of transmissions is $\frac{2}{p}$ which is about 2 for $p \in (0.99, 0.9)$, we can expect a node to wait a little over 2 frames on an average. While the exact power consumption depends on the frame duration, most of the contribution is due to the transmissions of packets and the receptions of the acknowledgements. Considering this, the energy consumption per event can be computed to be around 60μAsec. If the *event frequency* is κ events per second per sensor, then the average current drawn will be 60$\kappa\mu$Asec.

Consumption due to time synchronization Periodic time synchronization beacons will be required to maintain the time synchronization among nodes. For MSP430, a beacon transmission frequency of roughly once a second is sufficient to ensure an error under 64μsec [5]. Since, time synchronization beacons can also be lost in the channel, the frequency of these beacons must be at least 2 to 3 times higher than the minimum required rate (assuming a good channel). Time of these beacons can be pre-decided, so that the nodes wake up exactly at these times for receiving the beacon. For each beacon the node must wake up in the receive mode once. Time-synchronization packets are 11 bytes long (with 2 bytes of time-synch information in the payload)[6]. The duration of the time-synchronization slot is 704μsec. This entails the node drawing 12.5μAsec of energy per beacon or at a constant rate of about 40μA assuming three time synchronization beacons every second. The total

[4]We ignore the fact that the PSR of the acknowledgement packet will be slightly less than a sensor packet since a few bytes larger

[5]Run-time configuration commands (these are usually not latency critical) can be piggy-backed with the periodic beacons from the controller.

[6]Piggy-backing time-synchronization packets with acknowledgements can lead to further power savings however in this analysis we assume that piggy-backing is not used.

power consumption can thus be written as $40 + 60\kappa\mu$A.

Longevity of FTDMA The longevity of a node in years depends on the capacity of the batteries being used. NiCd AA battery capacities usually range between 650 mAHr - 1000 mAHr, while NiMH batteries can provide between 1400 mAHr - 2900 mAHr. The frequency of occurrence of events in manufacturing machine typically depends on the throughput of the machine. For example if a machine manufactures at a rate of one product every 10 seconds, we can expect an event roughly every 10 seconds per sensor ($\kappa = 0.1$). Typical machine cycles in manufacturing environments can range between as low as a few seconds to as high as a few minutes. For an event frequency of 1 event in 10 seconds, the average current drawn will be about 46μA (40+6). For a 1400 mAHr battery this gives life time of about 3.5 years and about 2.5 years for a high end NiCd battery (1000mAHr)[7]. *One interesting observation here is the fact that the energy expended for time-synchronization may be up to an order of magnitude greater than that consumed due to sensor traffic.*

5.6 Other Variations of FTDMA

Considering that failure error is determined by the number of re-transmissions that can be accommodated within the deadline, why waste precious time by accommodating acknowledgements from the controller? The nodes can simply re-transmit as many times as possible within the deadline and stop after the deadline has passed; the controller does not transmit any acknowledgements. We shall refer to this variation of FTDMA as FTDMA-B. The frame of FTDMA-B is depicted in Figure 3. The savings in frame duration obtained by using FTDMA-B instead of FTDMA, however are not significant. For example, consider that in a 12.5ms frame, the acknowledgement slot will typically be around 0.8ms - a savings of only 6%. FTDMA-B based MAC will however, provide a much lower longevity than an FTDMA MAC since nodes in FTDMA-B keep retransmitting even though their packet may have been successfully received at the controller.

Given that FTDMA-B suffers significantly in terms of energy consumption, a hybrid approach to FTDMA and FTDMA-B can be implemented where acknowledgements from the controller are transmitted once every few frames rather than every single frame in case of FTDMA. We call this protocol FTDMA-H (depicted in Figure 3). FTDMA-H thus, has a tunable parameter - the acknowledgement frequency μ_{ack} that determines how often the acknowledgements are transmitted from the controller. A $\mu_{ack} = 0.5$ signifies that the controller will transmit its acknowledgement once after every two sensor transmissions. FTDMA and FTDMA-B thus form the ends of a continuum in FTDMA-H with $\mu_{ack} = 1$ and $\mu_{ack} = 0$ respectively. While the variations FTDMA-B and FTDMA-H maybe useful mechanisms to ensure certain combinations of longevity and error probability guarantees, we do not provide results for these schemes due to lack of space.

[7]In this paper, however, we concern ourselves only with the feasibility of 802.15.4 and not the effects of sensors in our analysis. In a practical system sensors may consume much higher energy than the radio itself and may have a dramatic effect on the longevity of the system.

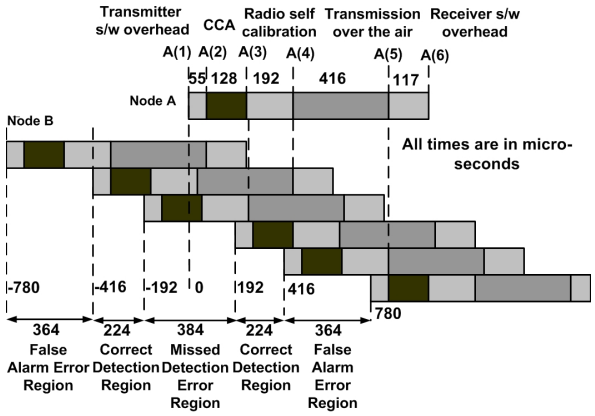

Figure 7. Clear channel assessment is not very efficient for small packets.

6. Performance of Contention-Based MACs

While there may be hundreds of sensors in a system, it is unlikely that all the sensors will be triggered at the same time because of an event or a combination of events. In most machines, burst sizes (number of sensors in a burst) will typically be in the order of 2-20 sensors. The potential advantage of using a contention-based scheme then, lies in the fact that its performance will depend only on the sensor burst size rather the actual number of sensors in the system. A contention-based scheme may thus perform better than FTDMA for systems with a large number of sensors but small burst sizes.

The performance of any contention-based scheme depends on the underlying traffic characteristics. In theory, it may be possible to design a MAC that is specifically tuned to a given underlying traffic model. In practice however, the underlying sensor traffic characteristics of a machine depends on its construction, its interaction with neighboring machines, its operating conditions and its environment. In fact, traffic characteristics may themselves change with time as the operating conditions (*e.g.,* high load, low load in case of a manufacturing system) change. In other words constructing a reasonably accurate traffic model may be in itself quite a challenging problem.

Rather than attempting to estimate the underlying traffic characteristics, in this section we take slightly different approach. We ask the question, "Given a maximum burst size, what is the error guarantee (in terms of probability that all sensors will succeed in transmitting before the deadline) that one can provide?"; the rationale being that any burst that is smaller than the maximum burst size will only perform better in terms of the probability of success. In our strawman approach towards designing the contention-based MAC, thus, we aim to maximize the probability that all the sensors will succeed within the given deadline, for a given maximum burst size that may occur in the system.

Contention-based MACs can be slotted (where time is divided into time slots) as well as unslotted. In general,

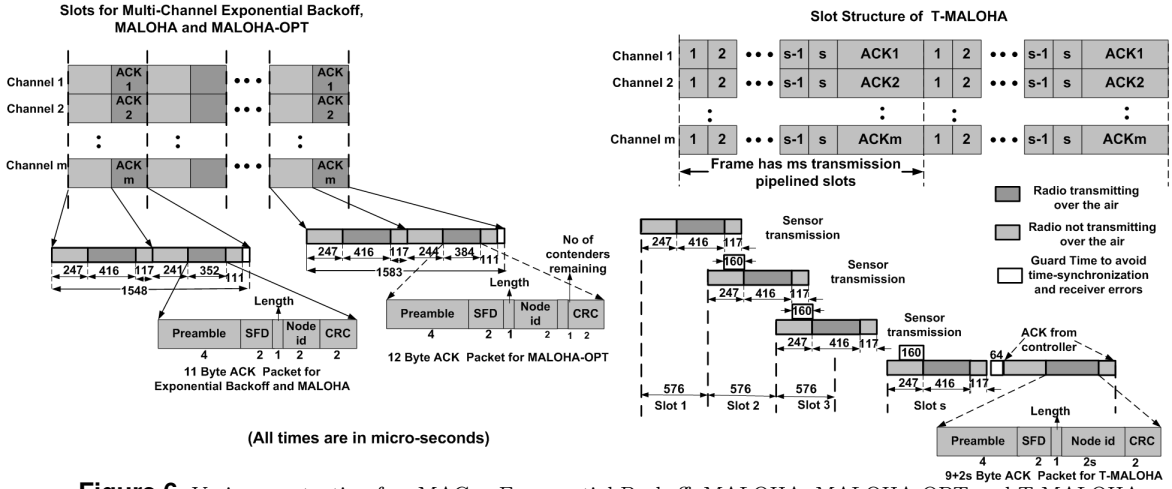

Figure 6. Various contention-free MACs - Exponential Backoff, MALOHA, MALOHA-OPT and T-MALOHA

slotted MACs perform better in terms of throughout compared to unslotted MACs (for example slotted ALOHA versus its unslotted version). Consequently, in our strawman approach to designing a contention-based MAC, we shall limit ourselves to slotted MAC protocols.

6.1 Ineffectiveness of CCA

Clear channel assessment (CCA) forms an important mechanism to avoid wasteful transmissions due to packet collisions in many contention-based MACs. In CCA each node first listens to the wireless channel for potential ongoing transmissions and transmits only if the channel is free to transmit. CCA can potentially help in avoiding wasteful packet collisions. A node performs CCA by listening to the wireless channel to measure the received signal strength (RSS). A high RSS indicates ongoing transmissions and typically if the measured RSS is above a certain threshold the channel is deemed busy; otherwise not. RSS being lower than a threshold however, does not guarantee the absence of ongoing transmissions because of the possibility of a transmitting hidden node (a node within the radio range of the CCU but not within that of the transmitting node). *In this section we start by asking the question "How effective is CCA in avoiding potential collisions for our target applications where the sensor packets are very small (13 bytes)?*

In CC2420 (and other radios) the most efficient strategy to perform CCA is to pre-configure the radio with an RSS threshold in the hardware. Prior to any transmission, the radio samples the channel and transmits only if the RSS is lower than the threshold. A CCA is performed with the transceiver in the receive mode. Thus, in order to transmit after performing the CCA, the radio has to change from receive to transmit mode. To change modes, the transceiver has to self-calibrate which takes 192μsec for CC2420. To obtain one sample of RSS, CC2420 averages over a period of 128μsec. Based on these facts, Figure 7 depicts the chronology of critical events in such a mechanism for a 13 byte sensor packet. (1)Node begins transmission, (2) Micro-controller sends transmit command to CC2420 over SPI - 55μsec, (3) Radio performs CCA - 128μsec, (4)Radio self-calibrates to transmit mode - 192μsec, (5) Radio finishes transmission over the air - 416μsec, (6) Packet is received at the receiver - 117μsec.

We now analyze the efficiency of the CCA mechanism when two nodes A and B transmit simultaneously. In figure 7 the starting of transmission of Node A is considered as the origin and different starting points for the transmission from Node B are considered. In the figure, $A(i)$ denotes the occurrence of the i^{th} event for node A, for example, B(3) means that Node B has finished CCA. We assume that Node A will detect an ongoing transmission from Node B if it is performing CCA (the period between A(2) and A(3)) while Node B is transmitting over the air (the period between $B(4)$ and $B(5)$).

False Alarm Errors : Consider transmission B's starting times in the interval (-780, -416)μsec. For all these transmissions, Node A's CCA interval overlaps with over the air transmission of Node B, thus, Node A will infer that there is an ongoing transmission and backoff. However, this would not be a prudent judgment since, if Node A had gone ahead with its transmission ignoring the result of the CCA, there would have been no collision. This is because Node B would have finished its over the air transmission by the time Node A started *its* over the air transmission. In other words, Node A inferred a collision when there was none to occur. This period of false alarm error lasts 364μsec long. A similar situation occurs in the interval (416,780)μsec, when Node B detects Node A's ongoing transmission and backs off unnecessarily. Performing CCA during these time intervals will thus lead to false alarms.

True detection : If node B transmits in the interval (-416,-192)μsec, Node A will correctly infer an ongoing transmission and avoid a potential collision. The same is true in the region (192,416)μsec when Node B will detect Node A's transmission and avoid a collision. We call this these regions of true detection.

Missed Detection Error : If Node B transmits in the interval (-192,192) μsec, Node A and Node B will completely miss detecting each others' ongoing transmissions leading to a collision.

The combined period of false detections and false alarms

is 1112 μsec (2×364 + 384) while the duration of correct detections is only 448μsec (2× 224). In other words, *the best CCA can correctly detect and avoid collisions is only about 29%.* It will miss detecting 24% of the time and raise a false alarm about 47% of the time. This is a rather disappointing result since it means that the decision taken based on CCA will be incorrect 71% of the time!!! [8]

6.2 Multi-Channel Exponential Backoff

The exponential back off based MACs are probably the most ubiquitous contention-based MACs found today. While it will not come as a surprise that the exponential back off mechanism is not best suited for the design goals desired in this paper (*i.e.,* maximizing the probability of success of all sensors within a given deadline), we shall evaluate it for the sake of comparison; the motivation being to answer the question, "How much better will our designed MAC perform compared to the exponential backoff based MACs?" In our analysis we assume that the controller is equipped m transceivers and thus it could potentially receive m packets within the same time-slot over different channels.

As depicted in Figure 6, time is divided into slots. Each slot comprises a sensor transmission and an acknowledgement transmission from the controller. The acknowledgement packet must contain the ID of the node being acknowledged since sometimes is is possible that if two or more nodes transmit in the slot one of them might be successfully received. The acknowledgement packet is thus 11 bytes long (see Figure 6).

Each node maintains a window of time-slots in which it uniformly randomly selects a time-slot for transmitting. While transmitting within a time-slot, the node uniformly randomly chooses one of the m channels [9]. In case of a collision the node backs off and the contention window is simply increased by a factor of two.

6.3 Multi-Channel ALOHA (MALOHA)

In our strawman approach to building a contention-based MAC we start with ALOHA[10]. We shall start by trivially extending ALOHA to accommodate multiple transceivers at the controller. Each time slot of the multi-channel ALOHA (MALOHA) is similar to the slot in the exponential back off scheme and comprises a sensor transmission and an acknowledgement (see Figure 6).

In MALOHA (similar to ALOHA), every sensor first decides whether or not to transmit in the current time slot with a probability α. Once having decided to transmit in a given time slot, the optimal strategy is to choose one among the m channels uniformly randomly (with a probability of $\frac{1}{m}$). It can be shown that the value of α that maximizes the number of successful sensor transmissions in a given time slot when b sensors are contending and the controller has m transceivers is given by,

$$\alpha = \begin{array}{ll} \frac{m}{b} & for \quad m < b \\ 1.0 & for \quad m \geq b. \end{array} \tag{2}$$

In practice, it may not be possible to estimate the number of contending sensors. Thus, MALOHA uses a conservative strategy which is to use b_{max} for b. In MALOHA thus, every sensor transmits in a time-slot with a probability $\frac{m}{b_{max}}$ and chooses one of the m channels at random for transmission.

6.4 Adapting α in MALOHA

If initially there were b_{max} sensors in the system, with more and more sensors succeeding the number of contending nodes decreases. What if somehow the controller could accurately estimate the number of contending sensors and continually convey this information in its acknowledgements to all the sensors?

If the sensor bursts are "sufficiently separated" (100ms or so), the controller can estimate the maximum possible remaining sensors in the system simply by subtracting the number of sensors that have succeeded from b_{max} [11] The controller can notify this value in its acknowledgement packets so that the sensors can now adapt their rates based on this knowledge. The sensors could then adapt their rates accordingly and ensure that the throughput is continuously maximized. We call this scheme MALOHA-OPT.

In MALOHA-OPT, the controller transmits an additional byte indicating the number of remaining contending sensors and thus the packet is 12 bytes long (see Figure 6) in the acknowledgement packet. The nodes use this value for b while computing the value of α for each time slot based on equation 2. Acknowledgement packets can however, be lost in the channel. In the event of the loss of an acknowledgement, the nodes continue with the value of α used in the previous time slot.

6.5 Transmission Pipelined MALOHA

In all previous contention-based MAC protocols we have not taken advantage of transmission pipelining. One can extend MALOHA such that time is divided into frames, each consisting of a set of s consecutive transmission pipelined time slots followed by an acknowledgement slot. We call this scheme transmission pipelined MALOHA (T-MALOHA). The communication pattern of T-MALOHA is depicted in Figure 6. Each frame in T-MALOHA can be thought of as equivalent to an extended time slot of MALOHA. A node decides whether or not to transmit a packet in a frame with a probability α (as in MALOHA). Once having decided to transmit within a frame, it does so by choosing uniformly from a set of ms time-frequency channels. α and s are not adapted or altered within the period of a single sensor burst. The controller transmits m acknowledgements in the acknowledgement slot as a list of node IDs and hence has a payload of $2s$ bytes.

[8] For radios without the facility for configuring CCA in hardware, the effectiveness of CCA will be further reduced due software and SPI communication overheads.

[9] it is trivial to show that choosing uniformly randomly among the m available channels is the optimal strategy to maximize the number of successes within the time slot.

[10] ALOHA promises the maximum throughput when the number of contenders is exactly known.

[11] In practice however, for some machines it may be hard to guarantee that bursts will always be well separated and it may not be practical to use MALOHA-OPT. We consider MALOHA-OPT primarily to evaluate a practical upper bound in performance on that could be achieved.

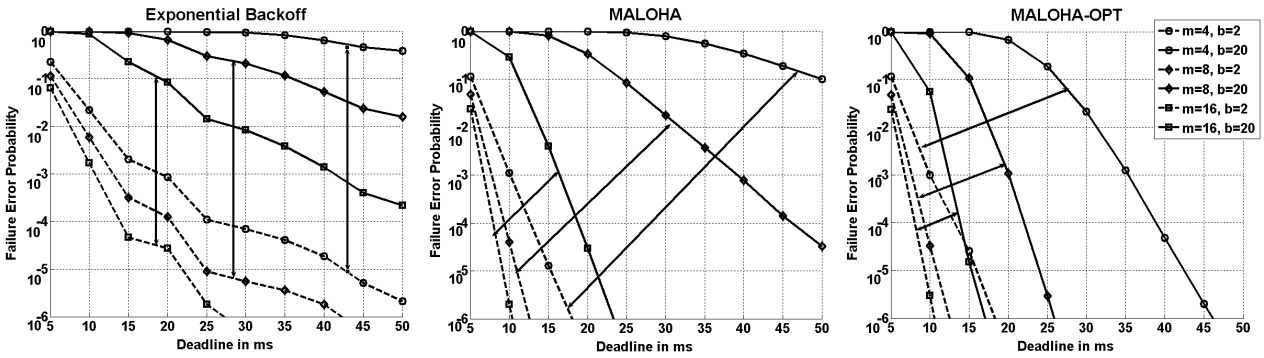

Figure 8. Performance of Exponential Backoff, MALOHA and MALOHA-OPT for $p = 0.99$

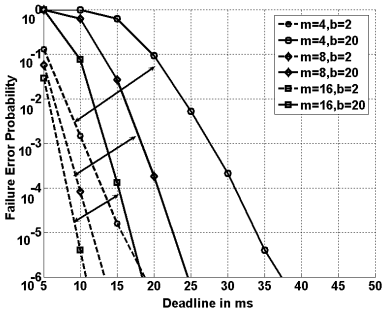

Figure 9. Performance of T-MALOHA for $p = 0.99$

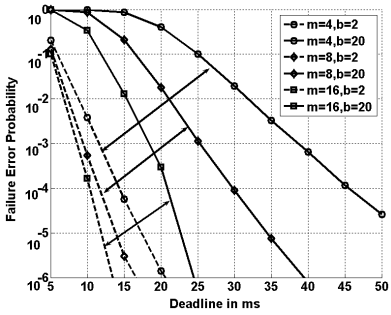

Figure 10. Performance of T-MALOHA for $p = 0.9$

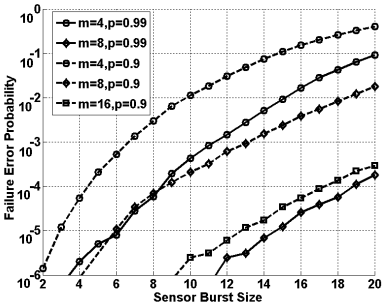

Figure 11. Performance of T-MALOHA for deadline of 20ms

Choosing an appropriate value of s and α are thus crucial to the performance of T-MALOHA. In the absence of an analytical solution to determine and optimal choice for m and s, we resorted to exhaustive search based on simulations. Thus, for each combination of $< m, b, p, \tau >$ we exhaustively searched for the value of $< s, \alpha >$ that maximizes the probability that all sensors are successfully received by the end of the deadline. Based on the list of obtained optimal values we found the following rules,
i) The optimal value of $< s, \alpha >$ does not depend on the channel quality p (we tried in the range $0.999 < p < 0.7$), ii) The optimal value of s always lies in the range $(\lceil \frac{b}{m} \rceil, \lceil \frac{b}{2m} \rceil)$, iii) The optimal value of α is always equal to 1.0 *i.e.,* fixing $\alpha = 1.0$ and searching for the value of s is sufficient, iv) The optimal value of s is closer to $\lceil \frac{b}{m} \rceil$ when the deadline is small and increases towards $\lceil \frac{b}{2m} \rceil$) as the deadline increases, and finally, v) For the range of values in the applications considered in this paper, we found that the approximation $s = \max\left(\lfloor \frac{b}{m} \rfloor, 1\right)$ to work very well in almost all cases except for 5 and 10ms deadlines.

Similar to MALOHA-OPT one could, in principle, consider adapting the values of s and α. The controller in principle could transmit its estimate of the number of contenders in the channel and the sensors would then choose the appropriate values of s and α in each frame by a precomputed lookup table. Adapting s has serious practical limitations since the acknowledgements from the controller may be lost in the channel. All sensor nodes must have exactly the same notion of s for the protocol to work. The value of α can however be adapted and we tried several schemes for adapting alpha given a fixed value of s. However, we found the improvement in performance to be only incremental and not "worth" the additional complexity of the protocol. We do not describe the adaptive schemes in this paper for lack of space.

6.6 Performance of various MACs

We evaluated the performance of the MACs described in this section on a homegrown simulator. Each point is the result from 10^7 events in the simulator. As in case of FTDMA, 1.5 ms was subtracted from each deadline to account for the delay in waking the radio up. We assumed that in a sensor burst all sensors are triggered at exactly the same time. This constitutes a harsher scenario than reality where nodes may be triggered at different times within a small interval. Figures 8 9 and 10 depict the dependence of probability of failure on value of the deadline, number of transceivers used at the receiver and the burst size. Since the dependence of failure probability on b has a large variation, rather than depict every single curve for each value of b (as in FTDMA), we only show the curves for $b = 2$ and $b = 20$. The arrows indicate the area that would be covered by the family of performance curves between burst sizes of 2 and 20 for a given value of m and p.

As expected, in Figure 8, the performance of multi-channel exponential back off MAC is unsuitable for our target application given that even with 16 transceivers (the

maximum possible) and a channel success rate (p) of 99%, at the end of 50ms, the failure rate is above 1ppm for most burst sizes between 2 and 20[12].

The performance of MALOHA depicted in Figure 8 for a channel with packet success rates of 99% for various values of b_{max} and m is significantly higher than the Exponential Back off based scheme. For example with 16 transceivers at the controller 1ppm error rate can be provided under 20ms for a burst size of 20!! Further improvement is seen in Figure 8 which depicts the performance of MALOHA-OPT for a channel with $p = 99\%$, especially for larger burst sizes.

The performance of T-MALOHA is depicted in Figures 9, 10 and 11. As seen from Figure 9, T-MALOHA clearly performs better than MALOHA-OPT for all values of b and m. In fact using T-MALOHA with 4 transceivers will be better than using FTDMA based systems for 50 nodes or higher and maximum burst sizes under 20 and is thus a clear choice for our target applications over FTDMA. Even for channels with a packet success rate of 90%, by using greater than 4 transceivers at the controller T-MALOHA can provide 1ppm error. Figure 11 depicts the dependence of performance on burst size for a deadline of 20ms. The curve corresponding to $m = 16$ and $p = 0.99$ is not present in the figure since T-MALOHA provides under 1ppm for all burst sizes between 2 and 20 for this deadline. Similarly with $m = 8$ and $p = 0.99$ systems with burst size under 11 can be guaranteed a 1ppm error.

6.7 Longevity of contention-based MACs

The energy consumption of contention-based MACs will be higher than that of FTDMA since the expected number of retransmissions will be higher and more energy will be spent in retransmitting the same packet. In our simulations we found that the expected number of retransmissions varies between 3-5 for depending on channel conditions, burst size and the number of transceivers at the controller. Considering that the expected number of transmissions for FTDMA systems is around 2, the component of power consumption due to events will be 1.5 to 3 times higher. The power consumption due to time synchronization however, will remain the same. For systems with low event-frequency (κ around 0.1) the effect on longevity will not be "significant"; however, systems with higher event frequencies maybe severely impacted.

7. Conclusions

In this paper we set out to determine whether or not the stringent reliability and longevity requirements posed by hard real-time discrete event control systems can be met using off-the-shelf 802.15.4 radios. We carefully analyzed the practical bottlenecks involved in designing a low-latency MAC and analyzed both contention-based and contention-free protocols. In the process, we proposed several novel techniques and MAC schemes suitable for our target applications.

[12]Further, we found that for channels with packet success rate of 90%, even with 16 transceivers, and $b = 2$, exponential backoff achieves a failure rate of only 10^{-5}.

Our results indicate that FTDMA MACs that use 4 or more transceivers at the controller may be a suitable option for small systems (50-100 sensors), however, FTDMA does not scale well for larger systems. We have designed a contention based MAC, T-MALOHA that promises to perform as well or better than FTDMA systems in terms of error probability for sensor burst sizes under 20. The drawback of T-MALOHA however, lies in the fact that its longevity may be much lower than that of FTDMA depending on the frequency of occurrence of events. The choice between using FTDMA and T-MALOHA is based on the tradeoff between i) size of the system in terms of number of sensors, ii) the burst sizes that may occur and iii) frequency of occurrence of sensory events since this may impact T-MALOHA very adversely. When the sensory event frequency is "low" (one event per 10 seconds or less) and burst sizes are below 20, T-MALOHA provides a clear benefit over FTDMA.

$m = 4$ transceiver T-MALOHA and FTDMA systems have been implemented on a home grown sensor node platform at Robert Bosch and deployed in model machines. The platform uses 4 CC2420s for transceivers and 4 MSP430 micro-controllers (one for each radio). A 5th micro-controller is used to aggregate the information from each of the 4 radios. Due to lack of space, we avoided a detailed description of the platform.

8. References

[1] W. Ye, J. Heidemann, and D. Estrin, "Medium Access Control With Coordinated Adaptive Sleeping for Wireless Sensor Networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, pp. 493–506, June 2004.

[2] T. V. Dam and K. Langendoen, "An Adaptive Energy Efficient MAC Protocol for Wireless Sensor Networks," in *The First ACM Conference on Embedded Networked Sensor Systems (Sensys'03)*, Los Angeles, CA, USA, Novemeber 2003.

[3] P. Lin, C. Qiao, and X. Wang, "Medium Access Control with a Dynamic Duty Cycle for Sensor Networks," in *IEEE Wireless COmmunication and Networking Conference*, vol. 3, March 2004, pp. 1534–1539.

[4] C. C. Enz, A. El-Hoiydi, J. D. Decotignie, and V. Peiris, "WiseNET: An Ultralow-Power Wireless Network Solution," *IEEE Computer*, vol. 37, no. 8, November 2004.

[5] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-Efficient, Collision-Free Medium Access Contorl for Wireless Sensor Networks," in *The First ACM Conference on Embedded Networked Sensor Systems (Sensys'03)*, Los Angeles, CA, USA, Novemeber 2003.

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

[7] J. Jamieson, H. Balakrishnan, and Y. C. T. Straser, "Sift: A MAC Protocol for Event-Driven Wireless Sensor Networks," MIT Laboratory for Computer Science, Tech. Rep. 893, May 2003.

[8] A. Ephremides and O. A. Mowafi, "Analysis of a Hybrid Access Scheme for Buffered Users - Probabilistic Time Division," *IEEE Transactions on Software Engineering*, vol. SE-8, no. 1, pp. 52–61, Jan 1982.

[9] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-MAC: a Hybrid MAC for Wireless Sensor Networks," in *The Third ACM Conference on Embedded Networked Sensor Systems (Sensys'05)*, San Diego, CA, USA, Novemeber 2005.