Project: IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs)

Submission Title: [Mediation Device Operation]

Date Submitted: [5 April, 2001]

Source: [Ed Callaway] **Company: [**Motorola]

Address: [8000 W. Sunrise Blvd., M/S 2141, Plantation, FL 33322]

Voice: [(954) 723-8341]**, FAX:** [(954) 723-3712]**,**

E-mail: [ed.callaway@motorola.com]

Re: [WPAN-802.15.4 Proposal support]

Abstract: [This document describes the operation of the Mediation Device, proposed by Motorola for the 802.15.4 standard, including its operation in a dedicated device and as a function distributed among network devices. Operation in small networks is also described.]

Purpose: [Supporting information to Motorola 802.15.4 proposal]

Mediation Device Operation

April 2001

Qicai Shi Ed Callaway

Motorola Labs

1. Introduction

IEEE 802.15.4 is a standard that has more emphasis on longer battery life and lower cost, less emphasis on message latency.

To lower the power consumption, the duty cycle has to be reduced. However, for an asynchronized system, a low duty cycle means a poor chance for the devices to synchronize with each other. For instance, device-A tries to contact with device-B , but device-B is likely to be sleeping while device-A is talking due to their low duty cycle.

To reduce the cost, a low cost crystal and/or an on-chip Micro Electro-Mechanical System (MEMS) resonator are considered as key frequency reference technologies. The issue with these technologies is the poor frequency stability, which will compound the device synchronization problem.

This paper introduces a new protocol. It enables low duty cycle devices to communicate with each other very well, and it doesn't require a high accuracy synchronization reference so that it overcomes the issue of poor frequency stability.

2. Low duty cycle frame structure

To low the power consumption, the duty cycle has to be reduced. Figure.1 gives an example of low duty cycle frame structures :

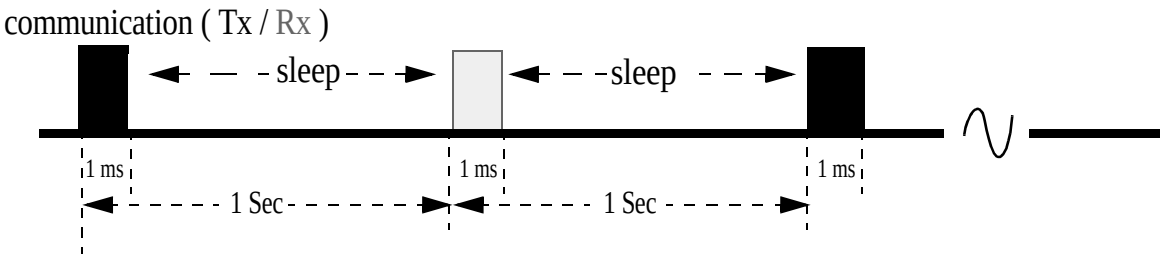

Fig.1

It takes 1 ms for the device to transmit and/or receive messages; then it goes back to deep sleep. After 1 second it wakes up again and repeats the cycle. The duty cycle is about 0.1%, therefore it is very energy efficient.

3. Issue for low duty cycle schema

Take a look at the example shown in figure 2 :

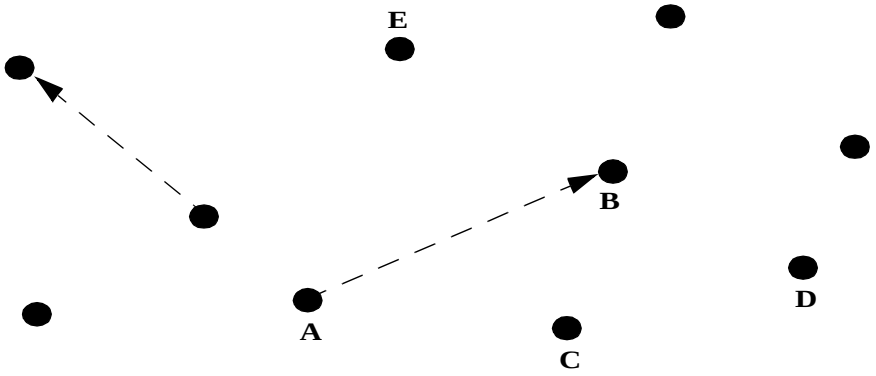

Fig.2

Device-A tries to talk to device B. Since both A and B wake up only 0.1% of the time, and A and B don't know each other's time schedule, the probability for A to hook-up with B is very low.

4. Mediation Device (MD)

In order to overcome this issue, a 'Mediation Device' (MD) is introduced here: A device that can record and replay a message, it functions as an "answering machine".

An MD may or may not have a relatively high duty cycle, and may have a frame structure as shown in figure 3 :

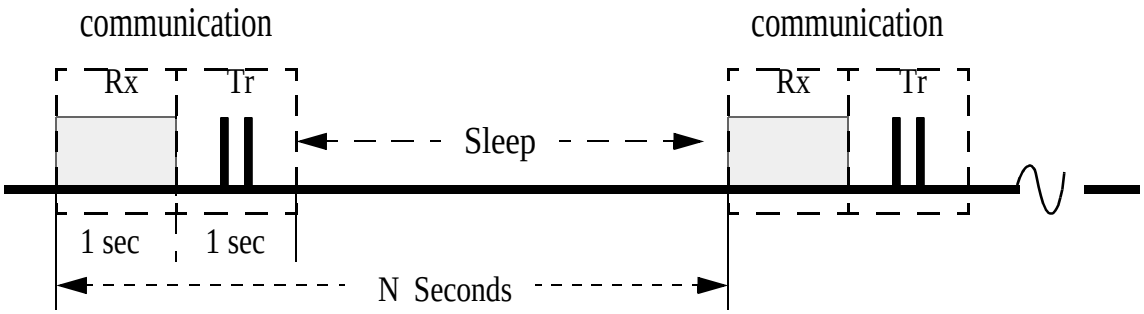

Fig.3

It takes 1 second for the MD to receive messages, then sends back some 1ms Ack messages during the next 1 second ***if necessary***, then it goes back to deep sleep mode. If the MD has a receive period that is long enough (> 1 sleep period + 2 communication periods of the regular devices),

then it is guaranteed that it can communicate with all the other devices as long as they are within the communication range of the MD.

The most important feature of a MD is to record and replay some simple control messages such as: “who is talking”, “to whom it wants to talk to”, and “ what time it will talk again” , etc.

5. Dedicated MD

In this implementation, the MD is a dedicated device. For star networks, in which all devices are within range of a single device, there is one MD in each network. The MD has larger power consumption than the regular devices.

Figure 4 shows us an example of how the system functions.

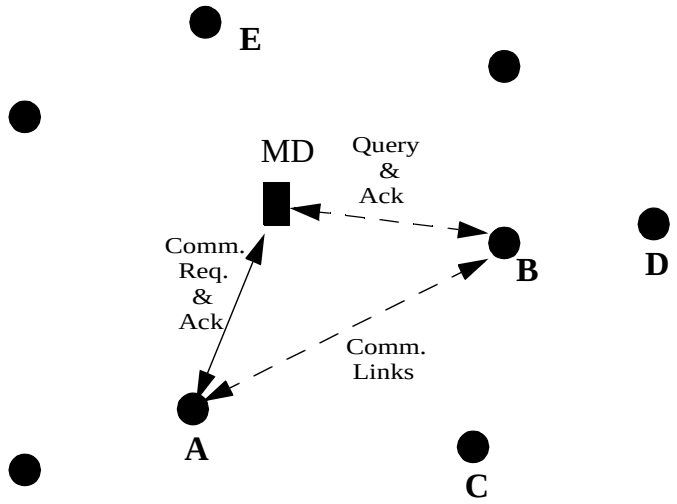

Fig.4

1). A wants to talk to B; it sends out a communication request message and waits for Ack.
2). B is sleeping at this moment, but fortunately the MD has a long communication period; it will intercept A’s message which includes A’s id, B’s id and A’s communication timing schedule information.
3). When B wakes up, B will send out a query message to check with MD.
4). Once the MD receives B’s query message, the MD will Ack back and forward A’s control message to B.
5). Once B receives the Ack from the MD, B will find out that A tries to contact with him and B also will find out when A will contact with him again.
6). Since B now knows A’s time schedule, B will sync on with A. From now on, they can start communication.
(An alternate way : Once B knows that A is trying to talk to him, B will wake up for a long period, and sync on with A)

Since the MD is a dedicated device, it can have a fixed location; its location can be used as a reference point for a relative location determination algorithm.

This protocol is good for an indoor environment where the MD can be plugged into the wall to get AC power.

5.1 Timing schedule for this protocol

Figure 5 shows us a detailed timing schedule for this protocol. In this example, the MD receives the ***Query message later than the Communication Request***.

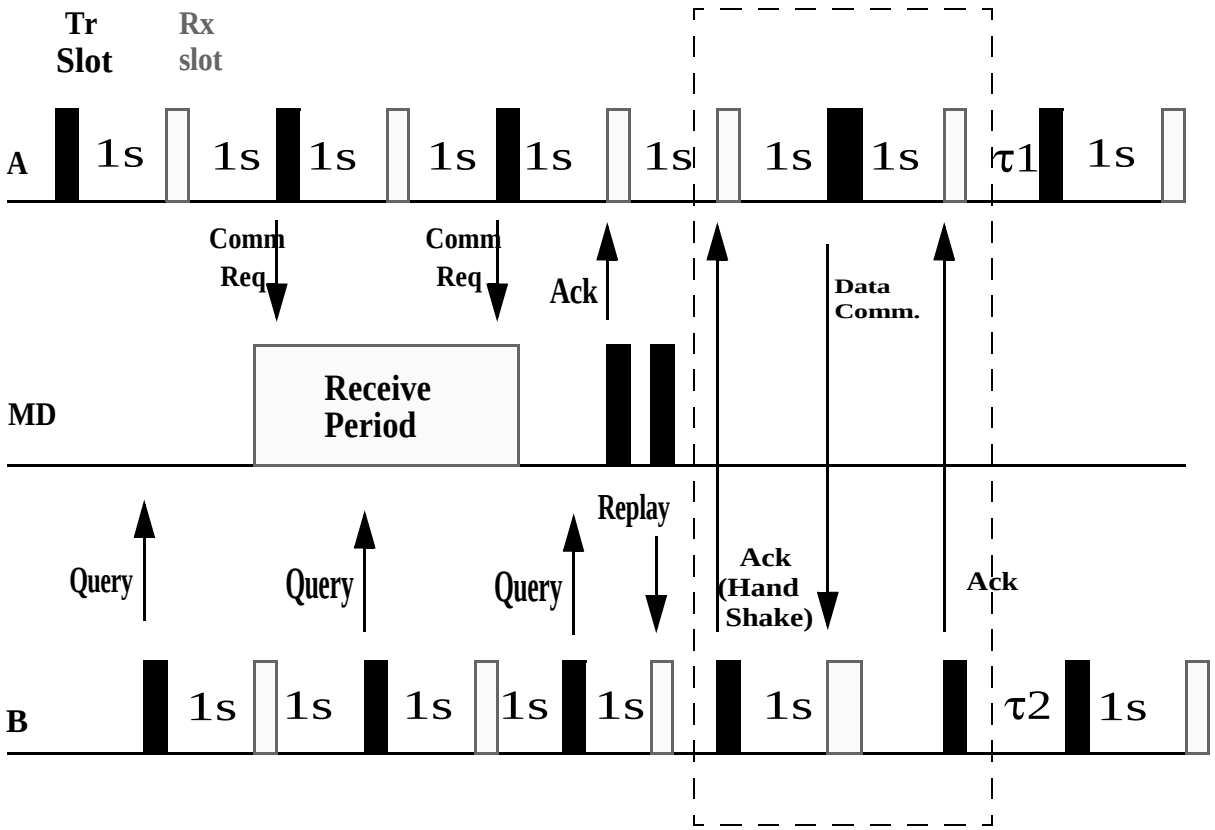

Fig.5

Timing schedule for device A :

In this example, A is the device that want to communicate with device B. A's transmit/ receive window is 1ms; A's sleep period is 1 second. A has a 0.1% duty cycle.

1). A sends out a communication request message, however, no device receives this message.
2). A sleeps for 1 second, then wakes up and listens. Since no device receive A's request message, A will not receive any Ack message.

3). A sleeps for 1 second, and sends out the communication request message again. At this time, the MD receives A's message.
4). A sleeps for 1 second, then wakes up and listens. However, the MD is still in the receive period at this moment, so that the MD will not send back an Ack to A. A will not receive anything.
5). A sleeps for 1 second, and sends out the communication request message again. At this time, the MD receives A's message again.
6). A sleeps for 1 second, then wakes up and listens. At this time A receives an Ack from the MD.
7). A knows that his message has been picked by the MD, so that A will not send out communication request message again. A sleeps for 1 second, then wakes up and listens. At this time, A receives B's Ack (hand-shake) message. A now knows that B is synchronized with him.
8). A sleeps for 1 second and transmits the data to B.
9). A sleeps for 1 second and receives an ack message from B. At this point, A sucessfully finishes the task of sending data to B.
10). Instead of sleeping the regular 1 second period, A sleeps a random time $\tau 1$ (0~2 Sec) to avoid future collision with B. Then A wakes up and sends out a query message.

.....

So on and so forth.

Timing schedule for the MD :

In this example, the MD is the device that mediates the communication between device A and device B. It has a long listen period (2002 ms), so that it is guaranteed that the MD can receive messages for all those regular devices that are within the MD's communication range.

1). Initially, the MD is in sleep mode. (The sleep period can be several seconds.)
2). After a while the MD wakes up and listens. During the listen period, the MD first receives a communication request message from A; it records this message. After a while, the MD receives a query message from B; it records this message. After a while, the MD receives the third message which is the same message as the first message from A; the MD refreshes this message.
3). Since the MD has A's timing slot information, it syncs with A and sends back an Ack message to A.
4). The MD also has B's timing information; it then syncs with B and replays A's contact information to B. The contact information includes A's timing information.
5). Then the MD goes back to sleep.

.....

So on and so forth.

Timing schedule for device B :

In this example, B is the device that A tries to talk to. B's transmit / receive window is 1 ms; B's sleep period is 1 second. B has a 0.1% duty cycle.

1). B sends out a query message to the MD to see if there any message for him; however,

the MD is sleeping at this time, so no device receives this message.

2). B sleeps for 1 second, then wakes up and listens. Since no device receives B's query message, B will not receive any Ack message.

3). B sleeps for 1 second, and sends out the query message again. At this time, the MD receives B's message.

4). B sleeps for 1 second, then wakes up and listens. However, the MD is still in the receive period at this moment, so that the MD will not send back an Ack to B. B will not receive anything.

5). B sleeps for 1 second, and sends out the query message again. At this time, the MD already passes the receive period, so that the MD cannot receive B's message. That is O.K. since the MD already has B's Query message.

6). B sleeps for 1 second, then wakes up and listens. At this time B receives the replay message from the MD. The replay message includes A's timing information.

7). B now knows A's timing information, so that it will adjust its timing and sync on with A. B then sends an Ack (hand-shake) message to A.

8). B sleeps for 1 second and listens. B successfully receives the data from A.

9). B sleeps for 1 second and sends an ack message to A.

10). Instead of sleeping the regular 1 second period, B sleeps a random time $\tau 2$ (0~ 2 Sec) to avoid future collision with A. Then wakes up and sends out a query message.

.....

So on and so forth.

What if the ***Query is earlier than the Communication Request*** ?

It is O.K since the MD will store all the messages and process them after the ***Receive Period***. At the end of the receive period, the MD already has A's timing information, so that the MD can pass this message to B, and then sends an Ack to A. Based on A's timing information, B will delay a little bit timing and sync with A.

6. Distributed MD

In this implementation, the MD isn't a dedicated device, it is a feature of any regular device. Any device within the group has the responsibility to function as a MD (just as in a bicycle race, each player has to be the leader once a while).

A device becomes an MD randomly. Once it is an MD, it functions exactly as a dedicated MD as described above. After one MD period, this device goes back to the normal model. So on and so forth.

A timing schedule example for this protocol is shown in figure 6:

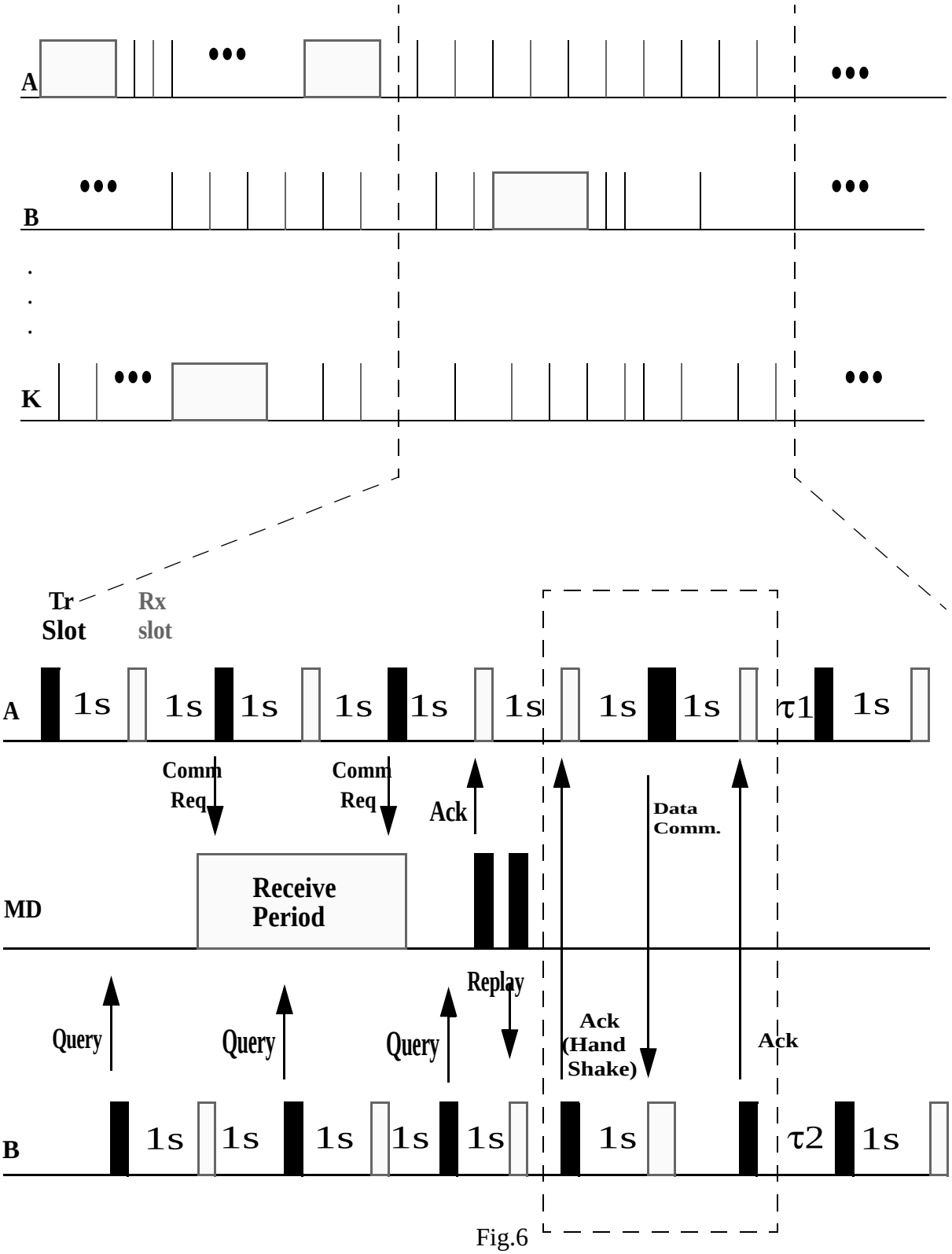
A
B
K
Tr Slot
Rx slot
1s
τ1
τ2
Comm Req
Comm Req
Ack
Data Comm.
MD
Receive Period
Replay
Query
Query
Query
Ack (Hand Shake)
Ack

Fig.6

The top part of figure 6 shows a big picture of the timing schedule of the system, meanwhile the bottom part is a zoom-in picture of the selected time window.

From the big picture, we can see that all the devices have the same function except that they have a different time schedule. They function as a MD once a while, then go back to regular mode.

From the detailed picture, we can see that once the device becomes a MD, it works exactly the same way as a dedicated MD.

Take a closer look at one specific device, as shown in Figure 7.

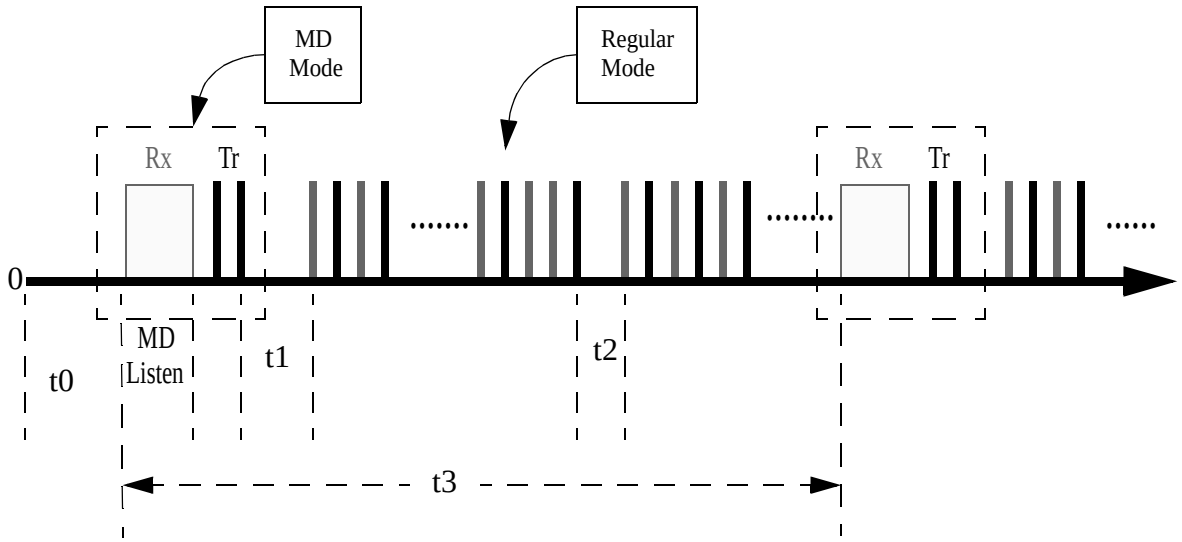

Fig.7

In Figure 7

1). In MD mode, the listen period is 2002 ms, the transmit periods are 1 ms, the delay (sleep) periods are depend on the timing schedules of the devices that the MD tries to talk to.

2). In regular mode, the transmit and receive periods are 1 ms, the sleep period is 1 second.

3). During the communication period, the sleep time need to adjust to sync the devices is as discussed in Section 5, Dedicated MD.

4). t0 is a random time, (0 < t0 < 2 seconds). If two devices turn on at the same time, t0 will randomize the timing schedules of these two devices.

5). t1 is a random delay time, (0 < t1 < 2 seconds); it is a delay between MD mode and regular mode.

6). t2 is a random delay time, (0 < t2 < 2 seconds); it is a delay after a successful communication. We try to randomize the time schedules of the two devices that have been synchronized.

7). t3 is a random time, (1500 seconds < t3 < 2500 seconds). We try to randomly pick

devices to function as MD, while keeping the duty cycle of individual devices low.

6.1 How to avoid collision ?

For distributed MD, collision is an issue: *What if more than one devices function as a MD at the same time?* It is O.K if more than 1 devices listen at the same time, but if they talk at the same time, they may interfere with each other.

To avoid collision, the device need to send out an Alarm message at the end of the MD receive period, as shown in Figure 8. If any other device is also in MD mode at this moment and receives the Alarm message, it should turn off its MD mode immediately and delay a random time, then goes back to regular mode.

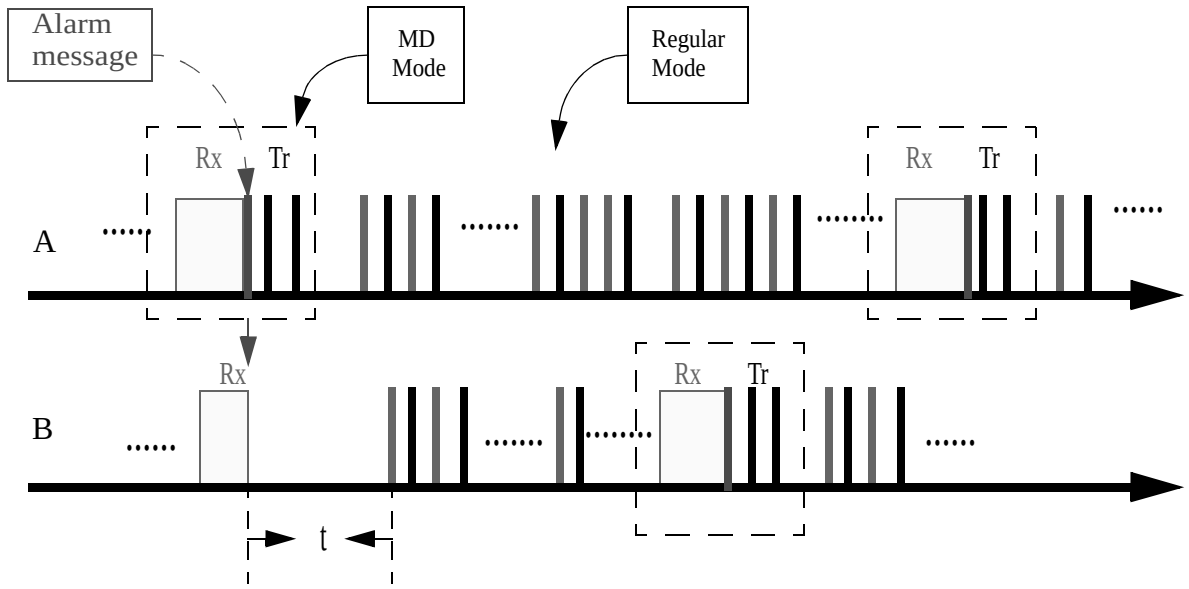

Fig.8

1). Device A is the device that functions as MD first.

2). After a while (less than 2s), B functions as MD.

3). At the end of A's MD receiving period, A sends out an ALARM message to ask other MD devices to turn off their MD mode.

4). During B's MD receiving period, B receives the ALARM from A. Once B receives this Alarm, it will turn off its MD mode, sleep a random time t to avoid future collision, and start function as a regular device.

7.0 MD for small network

The distributed MD protocol works well for large network (more than 10 nodes). However, for a small network (less than 5 nodes), one may suffer a long latency. For instance, for the protocol described in 6.0, if the number of nodes is 5, then the average latency is about 2000 Seconds / 5 = 400 Seconds.

How to reduce the latency for a small network?

Since it is a small network, we can regulate the communication window to a small range. For instance, we can regulate all the nodes to communicate within a 100 ms window, as illustrated in figure 9.

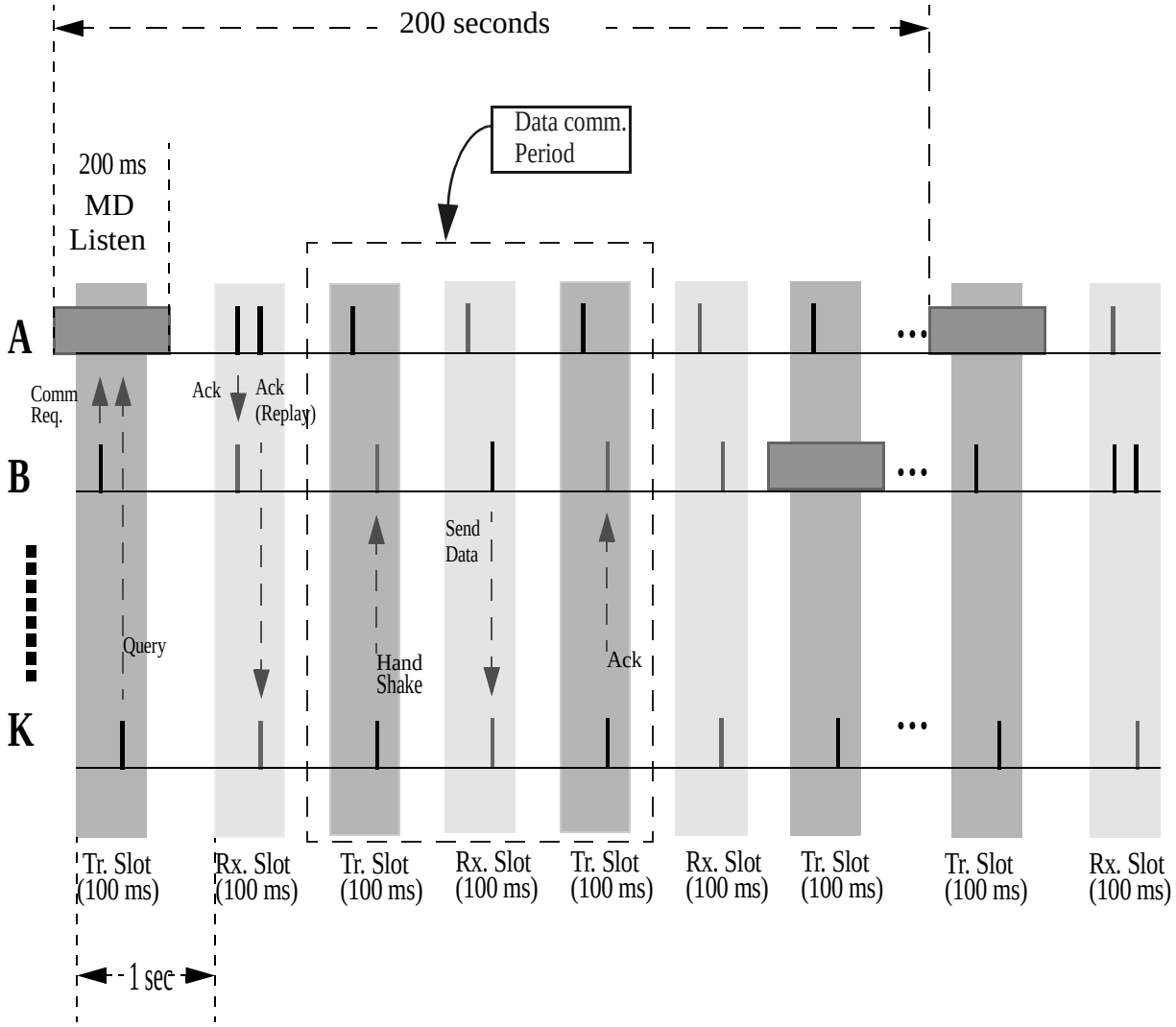

Fig.9

1). As shown above, all the communication occurs within the special time slots. The size of all the time slots is 100 ms. The slots are defined as Tr Slot (100ms) and Rx Slot (100 ms) alternately. The gap between the slots is 900ms.

2). The MD listening can ***only*** occur during the Tr Slot as shown above, the MD listening period is 200 ms to guarantee the overlap with the ***whole*** Tr Slot. The repeat period of the MD mode is about 200 seconds.

3). The routine Query and Communication Request can only occur during the Tr Slot. They have a period of 1 ms.

4). The Ack and Replay from the MD to the devices occur during the next Rx Slot.

5). Usually the data communication period includes a Handshake, a Send data link, and a return ack. The Ack from the destination device to the source device occurs during the next Tr Slot after the Handshake. The Send Data Link from the source to the destination device occurs during the next Rx slot after the Handshake.

6). After the data communication, all the devices go back to the routine Tr-Rx-Tr-Rx ... modes.

This is a new multiple access scheme. In a synchronized scheme, we know the exactly timing information of all the nodes. On the other hand, in an asynchronized scheme, we don't know anything about the timing. For the time schedule described above, we don't know the exact timing, but we know the timing within a 100 ms range.

The issues with this scheme are:

1). Nodes get lost (nodes drift away from the communication window).
2). New nodes join this network.

To overcome these issues, we need a long MD window (2002 ms listening period) once a while. For instance, we have a 2002 ms long MD mode once every 2000 seconds. This long MD mode is used to find any "lost" nodes and pull them back to the 100 ms window. It is also used to regulate new members to within the 100 ms communication windows.

An overall time schedule for any specific device of a small network is illustrated in figure 10:

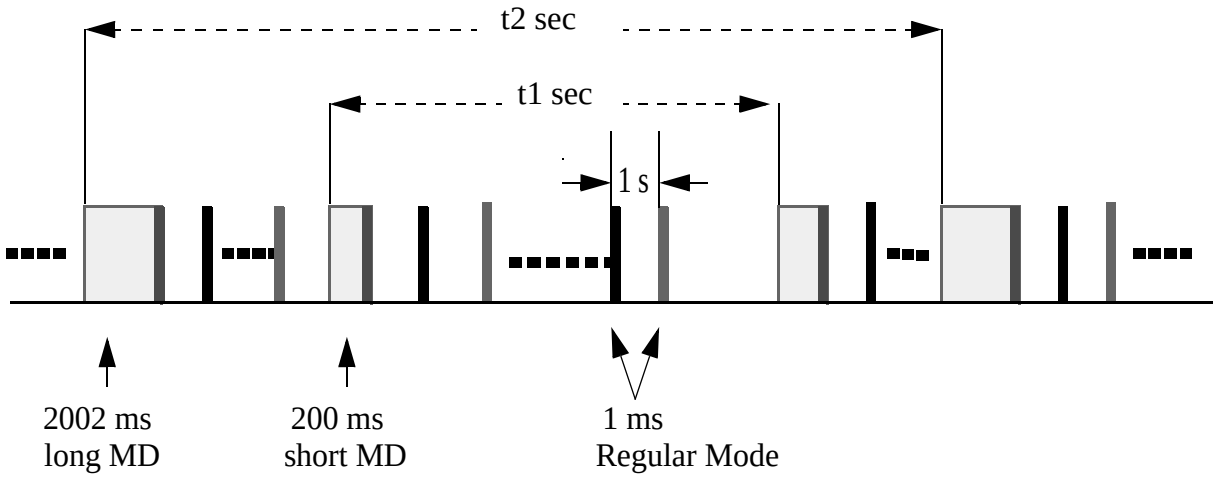

Fig.10

1). A 1 ms communication (Tr or Rx) every second.
2). A 200 ms short MD mode every t1 seconds (t1 is a random time, 150 < t1 < 250).
3). A 2002 ms long MD mode every t2 seconds (t2 is a random time, 1500 < t2 < 2500).

The overall duty cycle is about 0.25 %.
The average latency is about 200 / K seconds, where K is the number of the devices within the network. For example, for a network of 5 nodes, the latency is about 40 seconds.