Time Synchronization in Wireless Networks

Author: Michael Roche tke961@gmail.com

Abstract:

Time Synchronization in wireless networks is extremely important for basic communication, but it also provides the ability to detect movement, location, and proximity. The synchronization problem consists of four parts: send time, access time, propagation time, and receive time. Three current synchronization protocol Reference Broadcast Synchronization, Timing-sync Protocol for Sensor Networks, and Flooding Time Synchronization Protocol are presented and how they attempt solve the synchronization problem is also discussed. Security concerns as well as an industry case are also presented.

See Also:

Table of Contents

1.0 Introduction

Time synchronization in all networks either wired or wireless is important. It allows for successful communication between nodes on the network. It is, however, particularly vital for wireless networks. Synchronization in wireless nodes allows for a TDMA algorithm to be utilized over a multi-hop wireless network. Wireless time synchronization is used for many different purposes including location, proximity, energy efficiency, and mobility to name a few.

In sensor networks when the nodes are deployed, their exact location is not known so time synchronization is used to determine their location. Also time stamped messages will be transmitted among the nodes in order to determine their relative proximity to one another. Time synchronization is used to save energy; it will allow the nodes to sleep for a given time and then awaken periodically to receive a beacon signal. Many wireless nodes are battery powered, so energy efficient protocols are necessary. Lastly, having common timing between nodes will allow for the determination of the speed of a moving node.

The need for synchronization is apparent. Besides its many uses like determining location, proximity, or speed, it is also needed because hardware clocks are not perfect. There are variations in oscillators, which the clocks may drift and durations of time intervals of events will not be observed the same between nodes. The concept of time and time synchronization is needed, especially in wireless networks.

1.1 Wired Network Synchronization

For a wired network, two methods of time synchronization are most common. Network Time Protocol and Global Positioning System (GPS) are both used for synchronization. Neither protocol is useful for wireless synchronization. Both require resources not available in wireless networks.

The Network Time Protocol requires an extremely accurate clock, usually a server with an atomic clock. The client computer wanting to synchronize with the server will send a UDP packet requesting the time information. The server will then return the timing information and as a result the computers would be synchronized. Because of many wireless devices are powered by batteries, a server with an atomic clock is impractical for a wireless network.

GPS requires the wireless device to communicate with satellites in order to synchronize. This requires a GPS receiver in each wireless device. Again because of power constraints, this is impractical for wireless networks. Also sensor networks consist of inexpensive wireless nodes. A GPS receiver on each wireless node would be expensive and therefore unfeasible. The time accuracy of GPS depends on how many satellites the receiver can communicate with at a given time. This will not always be the same, so the time accuracy will vary. Furthermore Global Positioning System devices depend on line of sight communication to the satellite, which may not always be available where wireless networks are deployed.

The constraints of wireless networks do not allow for traditional wired network time synchronization protocols. Wireless networks are limited to size, power, and complexity. Neither the Network Time Protocol nor GPS were designed for such constraints.

1.2 Wireless Network Synchronization

The definition of time synchronization does not necessarily mean that all clocks are perfectly matched across the network. This would be the strictest form of synchronization as well as the most difficult to implement. Precise clock synchronization is not always essential, so protocols from lenient to strict are available to meet one's needs.

There are three basic types of synchronization methods for wireless networks. The first is relative timing and is the simplest. It relies on the ordering of messages and events. The basic idea is to be able to determine if event 1 occurred before event 2. Comparing the local clocks to determine the order is all that is needed. Clock synchronization is not important.

The next method is relative timing in which the network clocks are independent of each other and the nodes keep track of drift and offset. Usually a node keeps information about its drift and offset in correspondence to neighboring nodes. The nodes have the ability to synchronize their local time with another nodes local time at

any instant. Most synchronization protocols use this method.

The last method is global synchronization where there is a constant global timescale throughout the network. This is obviously the most complex and the toughest to implement. Very few synchronizing algorithms use this method particularly because this type of synchronization usually is not necessary.
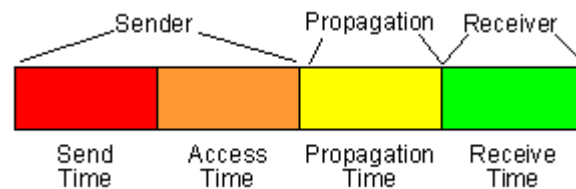

Figure 1 - Breakdown of packet delay components

As shown in figure 1, all the wireless synchronization schemes have four basic packet delay components: send time, access time, propagation time, and receive time. The send time is that of the sender constructing the time message to transmit on the network. The access time is that of the MAC layer delay in accessing the network. This could be waiting to transmit in a TDMA protocol. The time for the bits to by physically transmitted on the medium is considered the propagation time. Finally, the receive time is the receiving node processing the message and transferring it to the host. The major problem of time synchronization is not only that this packet delay exists, but also being able to predict the time spent on each can be difficult. Eliminating any of these will greatly increase the performance of the synchronization technique.

As illustrated there are many different variations of time synchronization or wireless networks. They range from very complex and difficult to implement to simpler and easy to implement. No matter the scheme used, all synchronization methods have the four basic components: send time, access time, propagation time, and receive time.

There are many synchronization protocols, many of which do not differ much from each other. As with any protocol, the basic idea is always there, but improving on the disadvantages is a constant evolution. Three protocols will be discussed at length: Reference Broadcast Synchronization (RBS), Timing-sync Protocol for Sensor Networks (TPSN), and Flooding Time Synchronization Protocol (FTSP). These three protocols are the major timing protocols currently in use for wireless networks. There are other synchronization protocols, but these three represent a good illustration of the different types of protocols. These three cover sender to receiver synchronization as well as receiver to receiver. Also, they cover single hop and multi hop synchronization schemes.

Back to Table of Contents

2.0 Reference Broadcast Synchronization

Many of the time synchronization protocols use a sender to receiver synchronization method where the sender will transmit the timestamp information and the receiver will synchronize. RBS is different because it uses receiver to receiver synchronization. The idea is that a third party will broadcast a beacon to all the receivers. The beacon does not contain any timing information; instead the receivers will compare their clocks to one another to calculate their relative phase offsets. The timing is based on when the node receives the reference beacon.

The simplest form of RBS is one broadcast beacon and two receivers. The timing packet will be broadcasted to the two receivers. The receivers will record when the packet was received according to their local clocks.

Then, the two receivers will exchange their timing information and be able to calculate the offset. This is enough information to retain a local timescale.

RBS can be expanded from the simplest form of one broadcast and two receivers to synchronization between n receivers, where n is greater than two. This may require more than one broadcast to be sent. Increasing the broadcasts will increase the precision of the synchronization.

RBS differs from the traditional sender to receiver synchronization by using receiver to receiver synchronization. The reference beacon is broadcasted across all nodes. Once it is received, the receivers note their local time and then exchange timing information with their neighboring nodes. The nodes will then be able to calculate their offset. [Elson2002]

2.1 Advantages of RBS

The main advantage of RBS is that it eliminates the uncertainty of the sender by removing the sender from the critical path. By removing the sender, the only uncertainty is the propagation and receive time. The propagation time is negligible in networks where the range is relatively small. It is claimed that the reference beacon will arrive at all the receiving nodes instantaneously. By removing the sender and propagation uncertainty the only room for error is the receiver uncertainty. Figure 2 illustrates this concept.
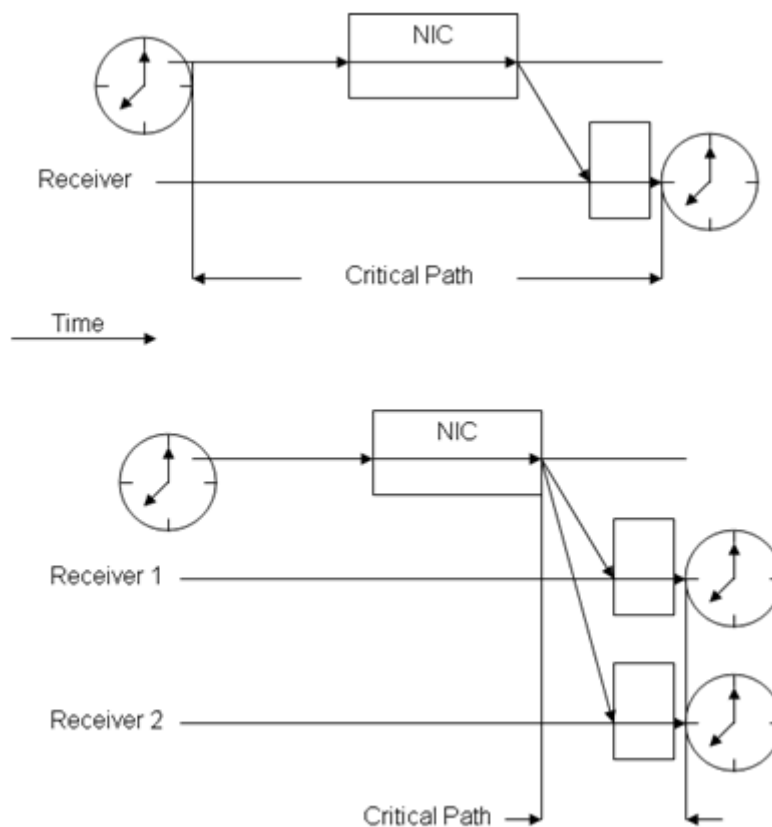
Figure 2 - Comparison of a traditional synchronization system with RBS

As seen here, the critical path in a traditional system, which is the top diagram, includes the sender. Since RBS is a receiver to receiver synchronization the sender is removed from the critical path. The critical path on contains the propagation and the receiver uncertainty. If, however, the transmission range is relatively small, then we can eliminate the propagation time and the critical path only contains the uncertainty of the receiver.

3.0 Timing-sync Protocol for Sensor Networks

TPSN is a traditional *sender-receiver* based synchronization that uses a tree to organize the network topology. The concept is broken up into two phases, the level discovery phase and the synchronization phase. The level discovery phase creates the hierarchical topology of the network in which each node is assigned a level. Only one node resides on level zero, the root node. In the synchronization phase all *i* level nodes will synchronize with *i-1* level nodes. This will synchronize all nodes with the root node. [Ganeriwal2003]

3.1 Level Discovery Phase

The level discovery phase is run on network deployment. First, the root node should be assigned. If one node was equipped with a GPS receiver, then that could be the root node and all nodes on the network would be synced to the world time. If not, then any node can be the root node and other nodes can periodically take over the functionality of the root node to share the responsibility.

Once the root node is determined, it will initiate the level discovery. The root, level zero, node will send out the *level_discovery* packet to its neighboring nodes. Included in the *level_discovery* packet is the identity and level of the sending node. The neighbors of the root node will then assign themselves as level one. They will in turn send out the *level_discovery* packet to their neighboring nodes. This process will continue until all nodes have received the *level_discovery* packet and are assign a level.

Once again all nodes are assigned a level to create a tree type topology. The root node is level zero continuing down the tree with level one and so on. All nodes of level *i* will broadcast the *level_discovery* with all nodes of level *i-1*. This is maintained until all nodes are assigned a level.

3.2 Synchronization Phase

The basic concept of the synchronization phase is two-way communications between two nodes. As mentioned before this is a sender to receiver communication. Similar to the level discovery phase, the synchronization phase begins at the root node and propagates through the network.
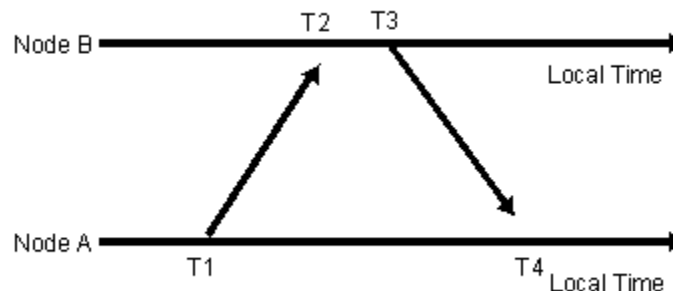

Figure 3 - Two-way communication between nodes

Figure 3 illustrates the two-way messaging between a pair of nodes. This messaging can synchronize a pair of nodes by following this method. The times T1, T2, T3, and T4 are all measured times. Node A will send the *synchronization_pulse* packet at time T1 to Node B. This packet will contain Node A's level and the time T1 when it was sent. Node B will receive the packet at time T2. Time T3 is when Node B sends the

acknowledgment_packet to Node A. That packet will contain the level number of Node B as well as times T1, T2, and T3. By knowing the drift, Node A can correct its clock and successfully synchronize to Node B. This is the basic communication for TPSN.

The synchronization process is again initiated by the root node. It broadcasts a *time_sync* packet to the level one nodes. These nodes will wait a random amount of time before initiating the two-way messaging. The root node will send the acknowledgment and the level one nodes will adjust their clocks to be synchronized with the root nodes.

The level two node will be able to hear the level one nodes communication since at least one level one node is a neighbor of a level two node. On hearing this communication the level two nodes will wait a random period of time before initiating the two-way messaging with the level one nodes. This process will continue until all nodes are synchronized to the root node.

Again the synchronization process executes much the same as the level discovery phase. All communication begins with the root node broadcasting information to the level 1 nodes. This communication propagates through the tree until all level *i-1* nodes are synchronized with the level *i* nodes. At this point all nodes will be synchronized with the root node.

3.3 Advantages of TPSN

Any synchronization packet has the four delays discussed earlier: send time, access time, propagation time, and receive time. Eliminating any of these would be a plus. Although TPSN does not eliminate the uncertainty of the sender it does, however, minimize it. Also, TPSN is designed to be a multi-hop protocol; so transmission range is not an issue.

Unlike RBS, TPSN has uncertainty in the sender. They attempt to reduce this non-determinism by time stamping packets in the MAC layer. It is claimed that the sender's uncertainty contributes very little to the total synchronization error. By reducing the uncertainty with low level time stamping, it is claimed that TPSN has a 2 to 1 better precision than RBS and that the sender to receiver synchronization is superior to the receiver to receiver synchronization. [Ganeriwal2003]

RBS also is limited by the transmission range. It was stated that RBS can ignore the propagation time if the range of transmission was relatively small. If it is a large multi-hop network, this is not the case. RBS would have to send more reference beacons for the node to synchronize. TPSN on the other hand was designed for multi-hop networks. Their protocol uses the tree based scheme so the timing information can accurately propagate through the network.

The sender to receiver synchronization method is claimed to be more precise than the receiver to receiver synchronization. Also TPSN is designed for multi-hop networks, where RBS works best on single hop networks. So, the transmission range is not a factor with TPSN.

Back to Table of Contents

4.0 Flooding Time Synchronization Protocol

Another form of sender to receiver synchronization is FTSP. This protocol is similar to TPSN, but it improves on the disadvantages to TPSN. It is similar in the fact that it has a structure with a root node and that all nodes are synchronized to the root.

The root node will transmit the time synchronization information with a single radio message to all participating receivers. The message contains the sender's time stamp of the global time at transmission. The receiver notes its local time when the message is received. Having both the sender's transmission time and the reception time, the receiver can estimate the clock offset. The message is MAC layer time stamped, as in TPSN, on both the sending and receiving side. To keep high precision compensation for clock drift is needed. FTSP uses linear regression for this.

FTSP was designed for large multi-hop networks. The root is elected dynamically and periodically reelected and is responsible for keeping the global time of the network. The receiving nodes will synchronize themselves to the root node and will organize in an ad hoc fashion to communicate the timing information amongst all nodes. The network structure is mesh type topology instead of a tree topology as in TPSN. [Maroti2004]

4.1 Advantages of FTSP

There are several advantages to FTSP, which it has improved on TPSN. Although TPSN did provide a protocol for a multi-hop network, it did not handle topology changes well. TPSN would have to reinitiate the level discovery phase if the root node changed or the topology changes. This would induce more network traffic and create additional overhead.

FTSP is robust in that is utilizes the flooding of synchronization messages to combat link and node failure. The flooding also provides the ability for dynamic topology changes. The protocol specifies the root node will be periodically reelected, so a dynamic topology is necessary. Like TPSN, FTSP also provides MAC layer time stamping which greatly increases the precision and reduces jitter. This will eliminate all but the propagation time error. It utilizes the multiple time stampings and linear regression to estimate clock drift and offset.
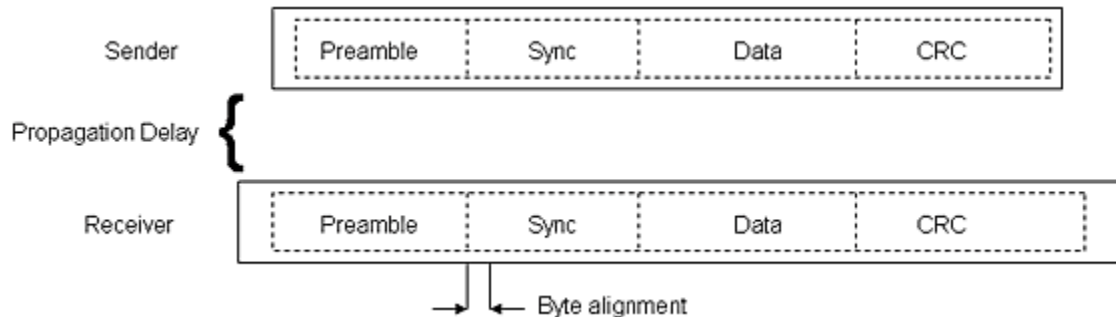

Figure 4 - Data packets transmitted with FTSP

The data packets transmitted with FTSP are constructed as shown in figure 4. There is a preamble then sync bytes followed by the data then finally the CRC. The dashed lines in the figure indicate the actual bytes in the packet and the solid line indicate the bytes in the buffer. When the sender is transmitting the preamble bytes, the receiver adjusts to the carrier frequency. Once the sync bits are received, the receiver can calculate the bit offset needed to accurately recreate the message. The time stamps are located at the boundaries of the sync bytes.

Allowing for dynamic topology changes, robustness for node and link failure, and MAC layer time stamping for precision are the major advantages of FTSP. It provides a low bandwidth flooding protocol to provide a network wide synchronization where all nodes are synchronized to the root node.

Back to Table of Contents

5.0 Attacks on Synchronization Protocols

There is a common problem among the three protocols presented. They were all developed to be energy efficient, precise, robust, and so on, but none of them were developed with security in mind. As in all computer protocols security is always an issue and attacks on protocols is inevitable.

In all synchronization attacks, the goal is to somehow convince nodes that their neighboring nodes are at a different time then they really are. Since global synchronization is the goal for some protocols and they rely on the neighboring nodes to pass the synchronization information on, compromising a node would disrupt the global synchronization.

5.1 Attacks on RBS, TPSN, and FTSP

For RBS, an attack on the synchronization can be executed easily. RBS works by receiver to receiver synchronization in which both nodes receive the reference beacon and then calculate their offset with one another. An attack would be as simple as compromising one of the nodes with an incorrect time. The non compromised node will then calculate an incorrect offset during the exchange period.

Remember TPSN is a sender to receiver tree based protocol with two phases, the level discovery phase and the synchronization phase. Both of the phases are initiated by the root node. In the synchronization phase the level number and the time are both sent through the tree. An attack would simply be to compromise a non-root node with the incorrect time. This will propagate through the tree and the closer the compromised node is to the root node, the more incorrect synchronization will occur.

Also a node could lie about its level. That would cause other nodes to request synchronization information in which it could give inaccurate information. That node also could refuse to participate in the level discovery phase, which could eliminate its children from the network.

The fundamental problem in FTSP is that it allows for any node to elect itself the root after a period of time of not receiving the synchronization information. A corrupt node could claim itself to be the root and now the other nodes will respond to its timing information instead of the correct information from the real root node. The will of course propagate through the network until all nodes have incorrectly calculated their skew and offset.

Since none of the protocols were designed with security in mind. Attacks on the synchronization are easily executed by following the rules of the protocol. In the sender to receiver synchronization, an attack will institute more damage because it will propagate through the network. [Manzo2005]

5.2 Countermeasures for Attacks

There are two major types of synchronization protocols. The single hop protocols, RBS, and the multi-hop protocols, TPSN and FTSP. In either case, the goal is to authenticate the synchronization messaging. Redundancy as well as nodes refusing to pass on bad information are other ways to combat synchronization attacks.

For single hop networks, the challenge for synchronization security is to make sure the sending node is not compromised to send out erroneous timing information. This can be accomplished by an authentication process. Either an authentication process or the use of a different private key between the sending node and each receiving node should be used for security.

In the multi-hop case an attack on a node close to the root could compromise a large portion of the network.

The use of private keys in this case could also be used, but there are a few other idea. For FTSP, redundancy could be introduced so that it does not calculate its timing from just one neighbor, but from several. It could then determine if there is a corrupt node. If a node was suspicious that it was receiving bad synchronization data, it could cease retransmission of the data. This would stop the desynchronization from propagating throughout the network.

Once again, none of the protocols discussed were designed with security in mind. Therefore it is easy to compromise a node's timing and have the erroneous timing propagate through the network, especially on multi-hop networks. Authentication, redundancy, and refusal to transmit corrupt synchronization information are ways to combat attacks. The tradeoff being that these countermeasures require overhead and will induce more network traffic, but it may be a small price to pay to keep synchronization attacks from compromising the network.

Back to Table of Contents

6.0 An Industry Case

This section represents an implementation example of a time synchronization protocol. An automation facility at a Swedish mining company is using a ZigBee, IEEE 802.15.4, wireless sensor network. The goal was to have a simple timing algorithm that was energy efficient because their sensor nodes were battery powered. Their idea was to put the nodes to sleep when not in use to conserve energy.
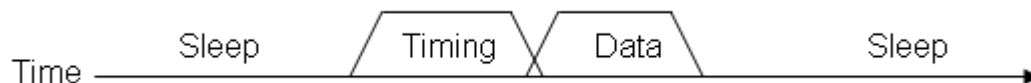

Figure 5 - Synchronization phases

The algorithm they developed was divided into three phases: timing phase, data phase, and sleep phase. The timing phase is when the node achieves or maintains synchronization. The data phase is when the data is transmitted over the network. Finally, the sleep phase is when the node enters a low power state where the radio is turned off. Figure 5 illustrates this.

The timing phase starts with the hub transmitting a single hop synchronization messages. In the message is *real_time* and *hop_count*. Initially they are both zero to indicate to the network the beginning of a frame. The nodes on the next hop will retransmit the message, after waiting a random period between 0 and 125 ms, to the next hop nodes until all nodes on the network have received the synchronization message. The *hop_count* is incremented on every hop and the *real_time* is the nodes elapsed time since the initialization. They are careful that all nodes do not retransmit if the network density is high to prevent flooding.

The company wants to further improve this algorithm. Currently every node is treated identically. Realistically, this is not the case since nodes on the edge of the network do not need the ability to route. Also they would like to trim the timing and data phases to as small as possible. [Aakvaag2005]

Again, they developed an synchronization algorithm that was a sender to receiver algorithm. They had a strict energy requirement. It was similar to TPSN in that the synchronization message started at the hub and propagated throughout the network one hop at a time. The nodes would also wait a random amount of time before retransmitting the synchronization message to avoid collisions much like TPSN.

Back to Table of Contents

Summary

In all wireless networks, the major problem for synchronization protocols is the variance in the send time, access time, propagation time, and the receive time. Elimination or the ability to accurately predict any of these greatly increases the effectiveness of the synchronization protocol. Discussion on RBS, TPSN, and FTSP was provided with each protocol's advantages. Finally a industrial case was presented and details of their protocol was given.

RBS is a broadcast protocol utilizing receiver to receiver synchronization. The designated root node would broadcast a beacon. Multiple nodes would receive the beacon simultaneously. The receivers would then note their local time upon reception and then compare with neighboring nodes. From this they would be able to calculate their phase offset. The main advantage is that this protocol removes the non-determinism from the sender. The major disadvantage is that it was not designed for large multi-hop networks. [Elson2002]

Next was TPSN which was a sender to receiver synchronization. This protocol has two major phases, the level discovery phase and the synchronization phase. In the level discovery phase each node is assigned a level with the root node being the only node at level zero. Next is the synchronization phase where the nodes at level one would initiate the two-way messaging with the root node. This process will propagate throughout the network. Both phases are initiated from the root node. The major advantage to this is that it provides a 2 to 1 improvement on precision. [Ganeriwal2003] The major disadvantage was that it did not allow for dynamic topology changes.

The final protocol discussed was FTSP, which was another sender to receiver synchronization protocol. FTSP broadcasts it timing information to all nodes that are able to receive the message. Those nodes in turn will calculate their offset from the global time. The receiving node will also calculate its clock skew using linear regression. The major advantage is that it is robust to compensate for node and link failures. It also allows for a dynamic topology. [Maroti2004].

These protocols were designed with performance in mind and did not take into account for security. It was shown that synchronization attacks on all these protocols were possible. Authentication, redundancy, and the refusal to pass on corrupt timing information were the countermeasure discussed. [Manzo2005].

Finally an industry case was presented where a Swedish automation facility is using a ZigBee sensor network and needed an energy efficient timing algorithm. They developed their own algorithm, but it performed much like TPSN. The hub node would start the timing and would transmit the message one hop. The receiving node(s) would then in turn retransmit the message. The nodes would enter a low power sleep mode to conserve energy.

Back to Table of Contents

References

These reference are ordered approximately in usefulness and relevance to this survey paper.

[Sivrikaya2004] Sivrikaya, F.; Yener, B. (2004). "Time synchronization in sensor networks: A Survey", *Network, IEEE Volume 18, Issue 4* Page(s):45 - 50, http://www.cs.rpi.edu/~sivrif/academic/papers/IEEEnetwork04.pdf

[Elson2002] Elson, J., Estrin, D. (2002). "Fine-Grained Network Time Synchronization using Reference

Broadcast.", *The Fifth Symposium on Operating Systems Design and Implementation (OSDI)*, p. 147-163

[Ganeriwal2003] Ganeriwal, S., Kumar, R., Srivastava, M. (2003). "Timing-Sync Protocol for Sensor Networks.", *The First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, p. 138-149

[Maroti2004] Maroti, M., Kusy, B., Simon, G., Ledeczi, A. "The Flooding Synchronization Protocol.", *Proc. Of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys)*. http://www.isis.vanderbilt.edu/publications/archive/Maroti_M_11_3_2004_The_Floodi.pdf

[Manzo2005] Manzo, M., Roosta, T., Sastry, S. "Time Synchronization Attacks in Sensor Networks." *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks SASN.*

[Aakvaag2005] Aakvaag, N., Mathiesen, M., Thonet, G. "Timing and Power Issues in Wireless Sensor Networks - An industrial Test Case.", *Parallel Processing. ICPP 2005 Workshops* Page(s):419 - 426.

Timing Synchronization in Sensor Networks, http://www.ee.ucla.edu/~saurabh/time_synchronization/

Characterizing Time Synchronization, http://www.circlemud.org/~jelson/writings/timesync/node2.html

[Tian2004] Tian, Z., Luo, X., Giannakis, G.B., "Cross-layer sensor network synchronization.", *Signals, Systems and Computers. Conference Record of the Thirty-Eighth Asilomar Conference,* Volume 1, 7-10, p. 1276 - 1280.

[Sheu2004] Sheu, J., Chao, C., Sun, C., "A Clock Synchronization Algorithm for Multi-hop Wireless Ad Hoc Networks.", *Distributed Computing Systems, Proceedings. 24th International Conference* p. 574 - 581

[Cox2005] Cox, D., Jovanov, E., Milenkovic, A., "Time Synchronization for ZigBee Networks.", *System Theory, SSST '05. Proceedings of the Thirty-Seventh Southwestern Symposium* p. 135 - 138

Back to Table of Contents

List of Acronyms

CRC	Cyclic Redundancy Check
FTSP	Flooding Time Synchronization Protocol
GPS	Global Positioning System
MAC	Medium Access Control
NIC	Network Interface Card
NTP	Network Time Protocol
RBS	Reference Broadcast Synchronization
TDMA	Time Division Multiple Access
TPSN	Timing-sync Protocol for Sensor Networks
UDP	User Datagram Protocol

Back to Table of Contents

Last Modified: April 23, 2006.

Note: This paper is available on-line at http://www1.cse .wustl.edu/~jain.