School of Engineering

Degree Program in Information Technology

Telecommunication Engineering

BACHELOR'S THESIS

Time Synchronization in Wireless Sensor Networks

Wenli Liu

Accepted ____.____._____ _____

SAVONIA UNIVERSITY OF APPLIED SCIENCES, ENGINEERING KUOPIO
Degree program
Information Technology

Author
Wenli Liu

Title of Project
Time Synchronization in Wireless Sensor Networks

Type of Project	Date	Pages
Final Project	2 May 2011	51

Academic Supervisor
Mr. Arto Toppinen , Principal Lecturer

Company
Savonia University of Applied Sciences, Engineering Kuopio

Abstract

The aim of this thesis was to study the time synchronization in wireless sensor networks which are based on standards NTP and IEEE1588. Time synchronization is timekeeping which requires the coordination of events to operate a system in unison. This can be compared to the conductor of an orchestra keeping the orchestra in time. Another purpose was then to compare the differences of three synchronization protocols.

First, three types of synchronization protocols and standards were studied. The main idea was to compare the differences between RBS, FTSP and IEEE1588. Previous experiments and their results were collected and compared.

As a result of this thesis, the three tests on these protocols show the differences in many ways, like in drift, error and offset. They are helpful in understanding time synchronization in depth.

Keywords
NTP, IEEE1588,PTP, RBS, FTSP

SAVONIA TEKNIIKKA KUOPIO
Koulutusohjelma
Information Technology

Kirjoittaja
Wenli Liu

Työn nimi
Aikatahdistus langattomissa sensoriverkoissa

Projekti	Päivämäärä	sivuja
Päättötyö	14 May 2011	51

Työn ohjaaja
Yliopettaja Arto Toppinen

Yritys
Savonia tekniikka Kuopio

Lyhennelmä

Työn tavoitteena oli tutkia kirjallisuuden pohjalta FTPS, RBS ja IEEE 1588 standardeihin perustuvia aikatahdistusmenetelmiä ja niiden eroja langattomissa sensoriverkoissa.

Aikatahdistus tarkoittaa ajanhallintaa, joka vaatii tapahtumien tarkkaa koordinointia, jotta järjestelmä toimii yhtenä kokonaisuutena. Yleinen vertaus orkesterin johtajaan pitämässä orkesteria oikea-aikaisena sopii hyvin senroriverkkoihin.

Tässä työssä esitellään ja tutkitaan kolmea protokollaa: RBS, FTBS ja IEEE 1588 stadardeja.
Työssä on etsitty kirjallisuudesta tutkimustuloksia ja esitetty Savonian Lange projektin tuloksia

Työn tuloksena esitellään em protokollien kirjallisuudesta poimittuja testituloksia. Tuloksissa nähdään eri standardien välillä eroja monessa suhteessa kuten kellojen liukuman ja offsetin suhteen. Erojen tutkiminen auttaa ymmärtämään aikasynkronoinnin toimintaa syvällisesti ja protokollien konfigurointia.

Avainsanat
NTP, IEEE1588,PTP, RBS, FTSP

Julkisuus
Julkinen

Acknowledgements

This thesis was carried out in Finland during my last year at Savonia University of Applied Sciences. Here I would like to thank Mr. Arto Toppinen, my supervisor. I appreciate that he gave me the opportunity to do this thesis. He is very obliging and gave me very many pieces of advices. Without him I could not have completed my thesis and graduated on time.

Moreover, I am thankful to all the teachers who taught me during the past five years.

At last, I want to thank my parents in particular for their support. And I will love you forever.

Wenli Liu

2 May 2011
Kuopio, Finland

Abbreviations

TDMA	Time Division Multiple Access
NTP	Network Time Protocol
PTP	Precision Time Protocol
WSN	Wireless Sensor Network
WLAN	Wireless Local Area Network
MAC	Media Access Control
RBS	References Broadcast Synchronization
TPSN	Timing-synchronization Protocol Sensor Network
FTSP	Flooding Time Synchronization Protocol
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
NIC	Network Interface Card
HRTS	Hierarchy Referencing Time Synchronization

Contents

1. INTRODUCTION

In the modern world, communication networks have entered into every aspect of people's lives. Especially today, wireless networks have changed dramatically and become more and more important in local networks. Time synchronization plays a significant role in wireless networks, even in wire line networks. In wireless networks, time synchronization is needed for nodes to communicate with each other on the networks. Synchronization in wireless nodes allows for a Time Division Multiple Access (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.

Time synchronization is a critical piece of substructure in any distributed system, but wireless sensor networks make particularly extensive use of synchronized time. Synchronize physical time for reasoning about events is required by almost all forms of sensor data fusion of coordinated actuation in the physical world. However, while the clock accuracy and precision requirements are often stricter in sensor networks than in traditional distributed systems, energy and channel constraints limit the resources available to meet these goals.

Obviously, synchronization is necessary. 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, particularly in wireless networks.

For more in depth understanding of the time synchronization, three types of synchronization protocols will be discussed. They are Reference Broadcast Synchronization (RBS), Timing-synchronization Protocol Sensor Network (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. In this thesis, it is necessary to compare the differences in results which comes from the experiment with each protocol

2. STANDARDS

In this thesis, all the situations are based on standards NTP, IEEE1588 and PTP.

2.1 NTP

The Network Time Protocol (NTP) is a protocol for synchronizing the clocks of computer systems over packet-switched, variable-latency data networks. It is designed particularly to resist the effects of variable latency by using a jitter buffer. [1]

2.2 IEEE1588

The aim of IEEE 1588 is to fill a niche not well served by either of the two dominant protocols, NTP and GPS. IEEE 1588 is designed for local systems requiring accuracies beyond those achievable using NTP. It is also designed for applications that cannot stand the cost of a GPS receiver at each node, or for which GPS signals are unapproachable. [2]

The technology behind the IEEE 1588 standard was used for distributed measuring and control tasks. The challenge was to synchronize networked measuring devices with each other in terms of time so that they are able to record measured values and provide them with an exact system time stamp. Based on this time stamp, the measured values can then be correlated with each other.

2.3 PTP

The Precision Time Protocol (PTP) is a high-precision time protocol for synchronization used in measurement and control systems residing on a local area network. Precision in the sub-microsecond range may be achieved with low-cost implementations.

PTP was originally defined in the IEEE 1588 (2002 version) standard, officially entitled "Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems". In 2008 a revised standard, IEEE 1588 (2008 version) was released. This new version, also known as PTP Version 2, improves accuracy, precision and robustness but is not backwards compatible with the original 2002 version. [3]

3. BACKGROUND

In this thesis, the main idea is to study the time synchronization in wireless sensor networks. Therefore, the background must be introduced at first.

3.1 Wireless Sensor Network

3.1.1 Introduction

A wireless sensor network (WSN) consists of spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, and to cooperatively pass their data through the network to a main location. [4]

Sensor networks are the key to gathering the information needed by smart environments, whether in buildings, utilities, industrial, home, shipboard, transportation systems automation, or elsewhere. In such applications, running wires or cabling is usually impractical. A sensor network is required that is fast and easy to install and maintain.

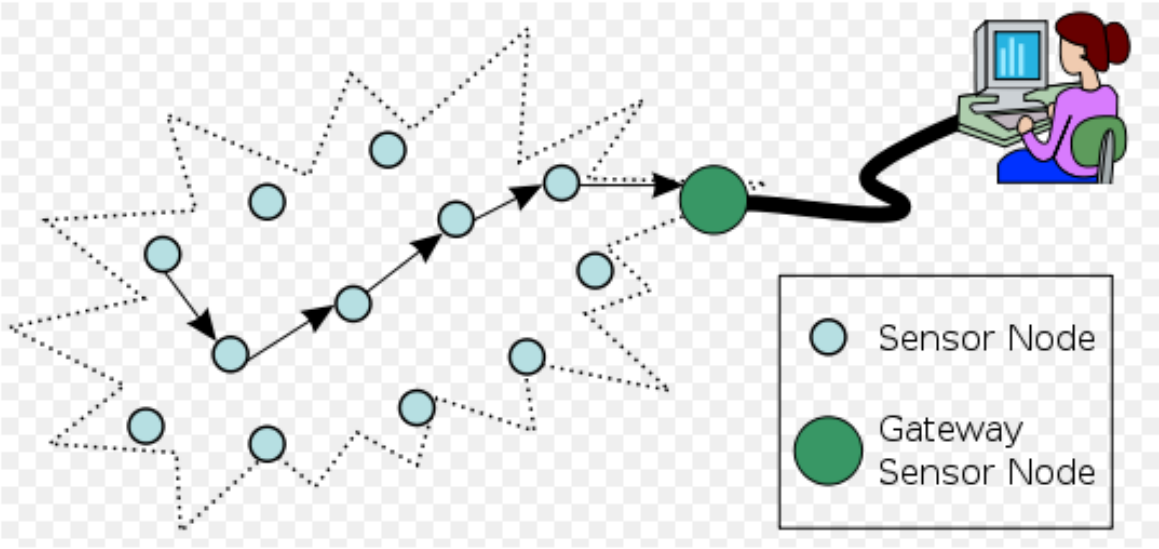

Figure 1 Typical multi-hop wireless sensor network architecture.

3.1.2 Network Topology

A communication network is composed of nodes, each of which has computing power. It can transmit and receive messages over communication links, wireless or wired. The basic network topologies are shown in the Figure 2 and include fully connected, mesh, star, ring, tree, bus. A single network may consist of several interconnected subnets of different topologies. Networks are further classified as Local Area Networks (LAN), e.g. inside one building, or Wide Area Networks (WAN), e.g. between buildings. [5]

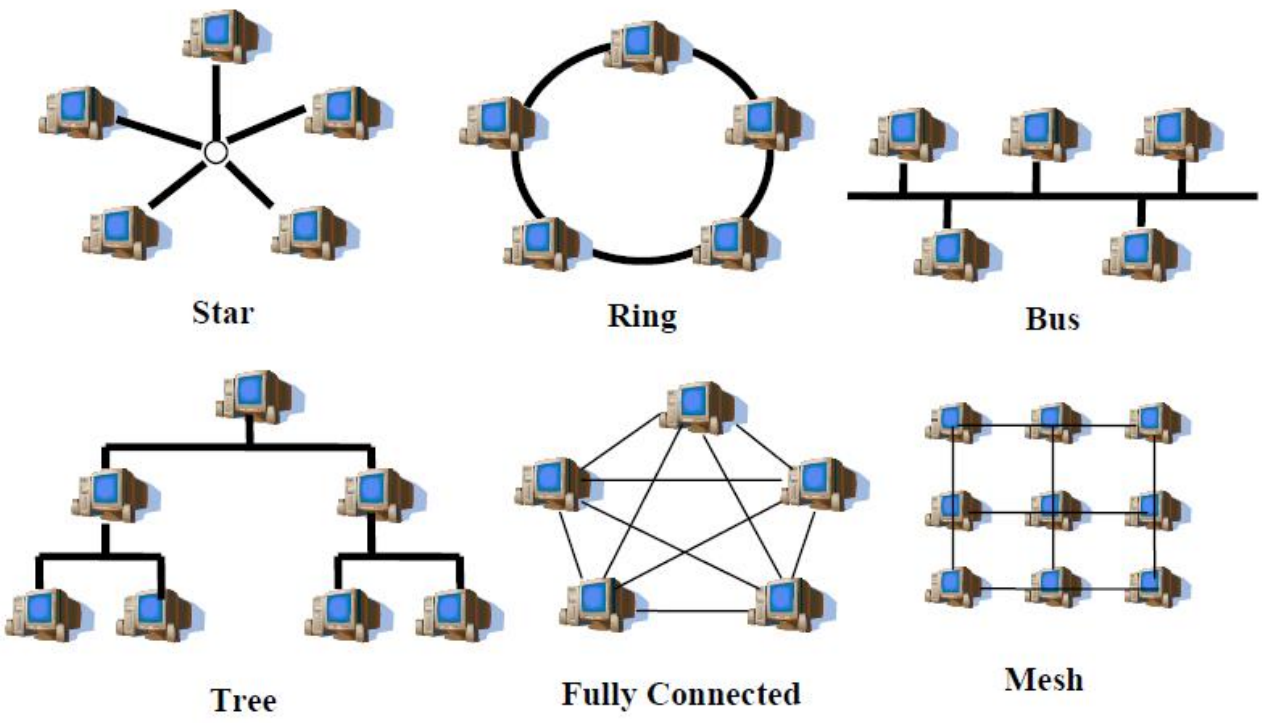

Figure 2 Basic network topologies.

3.2 Wireless Local Area Networks

3.2.1 Introduction

A wireless local area network (WLAN) links two or more devices using some wireless distribution method (typically spread-spectrum or Orthogonal Frequency-division Multiplexing radio), and usually providing a connection through an access point to the wider internet. This gives users the mobility to move around within a local coverage area and still be connected to the network. [6]

Figure 3 Wireless local area network.

3.2.2 Types of WLAN

An ad-hoc network is a network where stations communicate only peer to peer (P2P). There is no base and no one gives permission to talk. This is accomplished using the Independent Basic Service Set (IBSS).

A peer-to-peer (P2P) network allows wireless devices to directly communicate with each other. Wireless devices within range of each other can discover and communicate directly without involving central access points. This method is typically used by two computers so that they can connect to each other to form a network. [7]

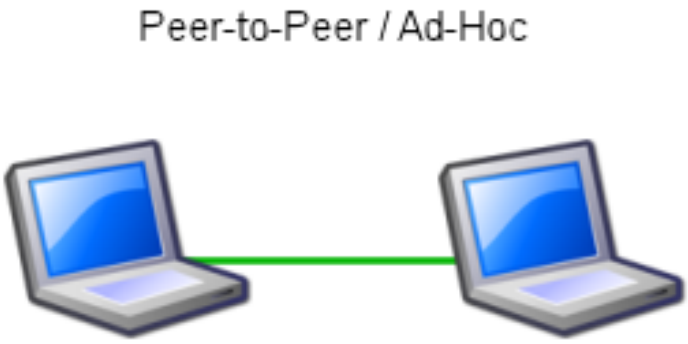

Figure4 Peer-to-peer or ad-hoc wireless LAN

4. TIME SYNCHRONIZATION IN WIRELESS SENSOR NETWORK

4.1 Introduction to Time Synchronization

Synchronization is timekeeping which requires the coordination of events to operate a system in unison. The familiar conductor of an orchestra serves to keep the orchestra in time. Systems operating with all their parts in synchrony are said to be synchronous or in sync. [8]

Time synchronization is important in all networks either wired or wireless. It allows for successful communication between nodes on the network. However, it is 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.

When the nodes are deployed in the sensor network, time synchronization is used to determine the exact 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. Energy efficient protocols are necessary because many wireless nodes are battery powered. Lastly, having common timing between nodes will allow for the determination of the speed of a moving node. [10]

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. Time synchronization and the concept of time are needed, especially in wireless networks.

4.2 Wireless Network Synchronization

4.2.1 Introduction

Time synchronization is an important issue in multi-hop, ad-hoc wireless networks such as sensor networks. Many applications of sensor networks need local clocks of sensor nodes to be synchronized, requiring various degrees of precision. Some intrinsic properties of sensor networks such as limited resources of energy, storage, computation, and bandwidth, combined with potentially high density of nodes make traditional synchronization methods unsuitable for these networks. Hence there has been an increasing research focus on designing synchronization algorithms especially for sensor networks.

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. It is not always necessary for precise clock synchronization, so protocols from lenient to strict are available to meet one's needs.

4.2.2 Synchronization Methods for Wireless Networks

There are three basic types of synchronization methods for wireless networks. The first method is the simplest. It is relative timing. 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. It all needed is comparing the local clocks to determine the order. 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. At any instant the nodes have the ability to synchronize their local time with another node local time. This method is used in most synchronization protocols.

The last method is global synchronization where there is a constant global timescale throughout the network. Obviously, this is 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.

4.2.3 Synchronization Schemes

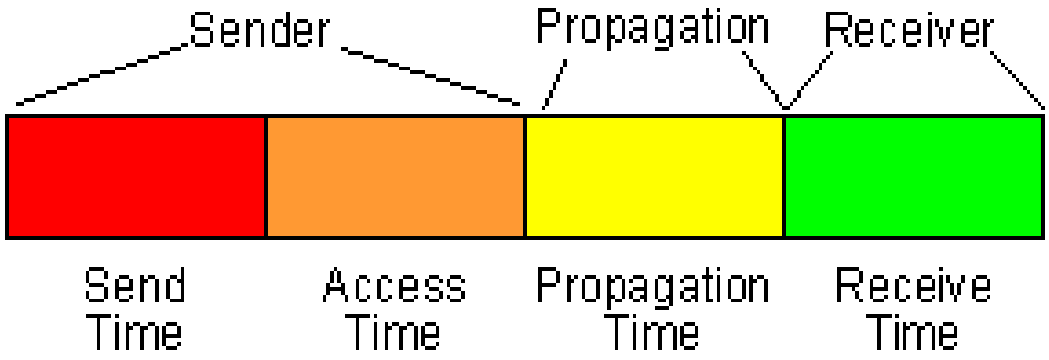

Figure 5 Breakdown of packet delay components. [11]

As shown in Figure 5, all the wireless synchronization schemes have four basic packet delay components: send time, access time, propagation time, and receive time.

Here, the four components will be expounded separately. [12]

- Send Time—the time spent at the sender to construct the message. This includes kernel protocol processing and variable delays introduced by the operating system, e.g. context switches and system call overhead incurred by the synchronization application. Send time also accounts for the time required to transfer the message from the host to its network interface.

- Access Time—delay incurred waiting for access to the transmit channel. This is specific to the MAC protocol in use. Contention-based MACs must wait for the channel to be clear before transmitting, and retransmit in the case of a collision. Wireless RTS/CTS schemes such as those in 802.11 networks require an exchange of control packets before data can be transmitted. TDMA channels require the sender to wait for its slot before transmitting.

- Propagation Time—the time needed for the message to transit from sender to receivers once it has left the sender. When the sender and receiver share access to the same physical media (e.g., neighbors in an ad-hoc wireless network, or on a LAN), this time is very small as it is simply the physical propagation time of the message through the media. In contrast, Propagation Time dominates the delay in wide-area networks, where it includes the queuing and switching delay at each router as the message transits through the network.

- Receive Time—processing required for the receiver's network interface to receive the message from the channel and notify the host of its arrival. This is typically the time required for the network interface to generate a message reception signal. If the arrival time is times tamped at a low enough level in the host's operating system kernel (e.g., inside of the network driver's interrupt handler), the Receive Time does not include the overhead of system calls, context switches, or even the transfer of the message from the network interface to the host.

As showed in Figure 3 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 what the scheme used, all synchronization methods have the four basic components: send time, access time, propagation time, and receive time.

5. SYNCHRONIZATION PROTOCOLS

There are many synchronization protocols, many of which do not differ much from each other. As with any protocols, the basic idea is always there, but improving the disadvantages is constant evolution.

There are three protocols: Reference Broadcast Synchronization (RBS), Timing-sync Protocol for Sensor Network (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 the sender to receiver synchronization as well as the receiver to receiver. And they also cover single hop and multi hop synchronization schemes.

5.1 RBS

5.1.1 Introduction

Reference Broadcast Synchronization (RBS) is a method in which the receiver uses the physical layer broadcasts for comparing the clocks. This is slightly different form traditional methods which synchronize the senders with the receivers. 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. The receivers will compare their clocks to one another to calculate their relative phase offsets. The timing is found when the node receives the reference beacon.

RBS has one broadcast beacon and two receivers. It is the simplest form. It broadcast the timing packet to the two receivers. Then the receivers will record when the packet was received according to their local clocks. And then the two receivers will swap their timing information and be able to calculate the offset. This is enough information to keep a local timescale.

RBS can be expanded from the simplest form of one broadcast and two receivers to synchronization between n receivers (n is great than two). Maybe more than one broadcast needs to be sent. It will increase the precision of the synchronization by increasing the broadcast. [13]

5.1.2 Advantages of RBS

The main advantage of RBS is that it removes the uncertainty of the sender by removing the sender from the critical path. The propagation and receive time is the only uncertainty by removing the sender. The propagation time is insignificant in networks where the range is relatively small. It is claimed that the reference beacon will arrive at all the receiving nodes instantly. By removing the sender and propagation uncertainty the only space for error is the receiver uncertainty.

Figure 3 illustrates this concept.

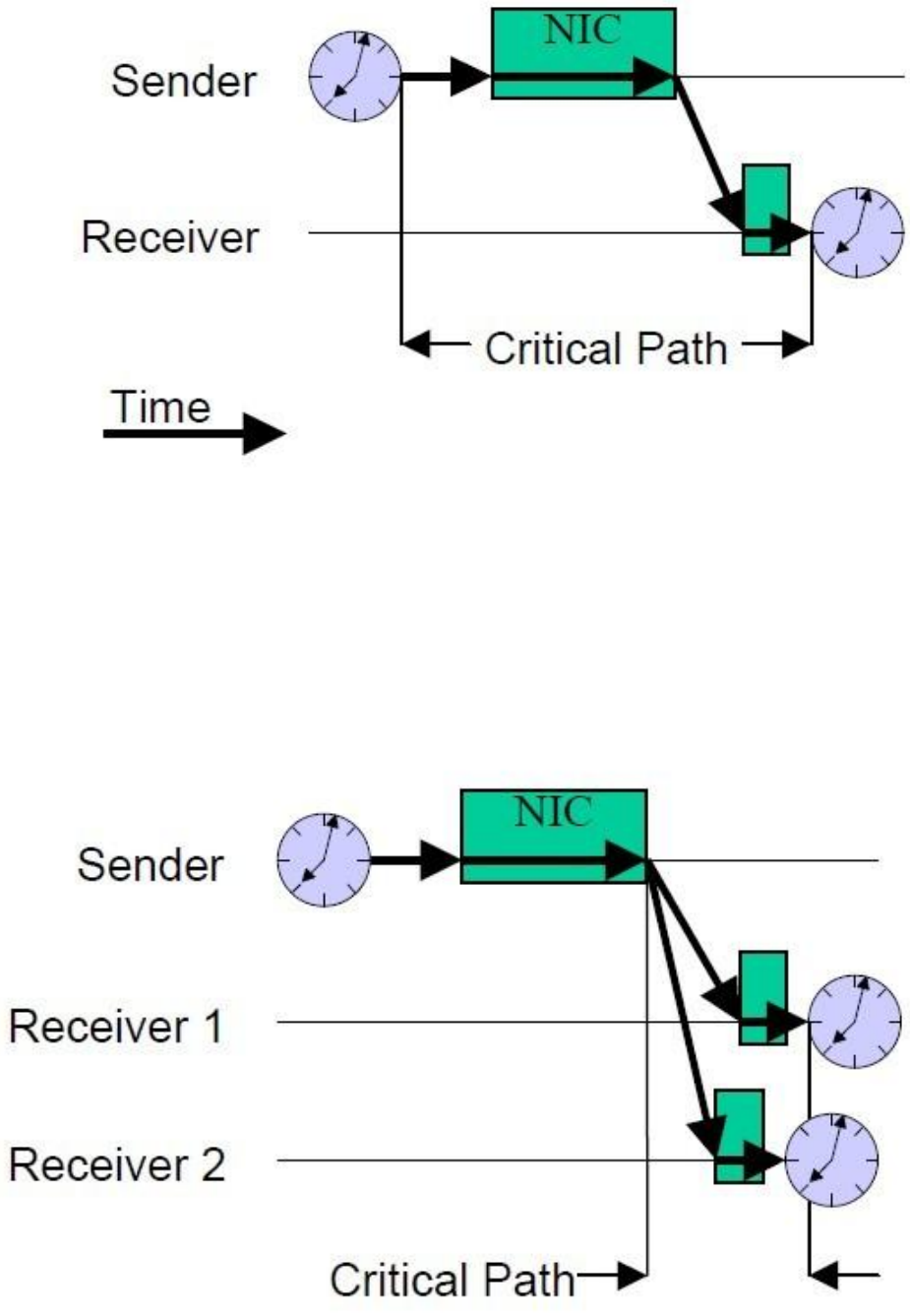

Figure 6 Comparison of a traditional synchronization system with RBS [14]

Figure 6 shows a critical path analysis for traditional time synchronization protocols (top) and RBS (bottom). For traditional protocols working on a LAN, the largest contributions to

nondeterministic latency are the Send Time (from the sender's clock read to delivery of the packet to its NIC, including protocol processing) and Access Time (the delay in the NIC until the channel becomes free). The Receive Time tends to be much smaller than the Sent Time. The clock can be read at interrupt time, before protocol processing. In RBS, the critical path length is shortened to include only the time from the instillation of the packet into the channel to the last clock read.

5.2 TPSN

5.2.1 Introduction

Timing-synchronization Protocol Sensor Network (TPSN) is a traditional sender-receiver based synchronization that uses a tree to organize the network topology. The concept is separated 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. This will synchronize all nodes with the root node. [15]

The basic concept of the synchronization phase is a two-way communication 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 the network.

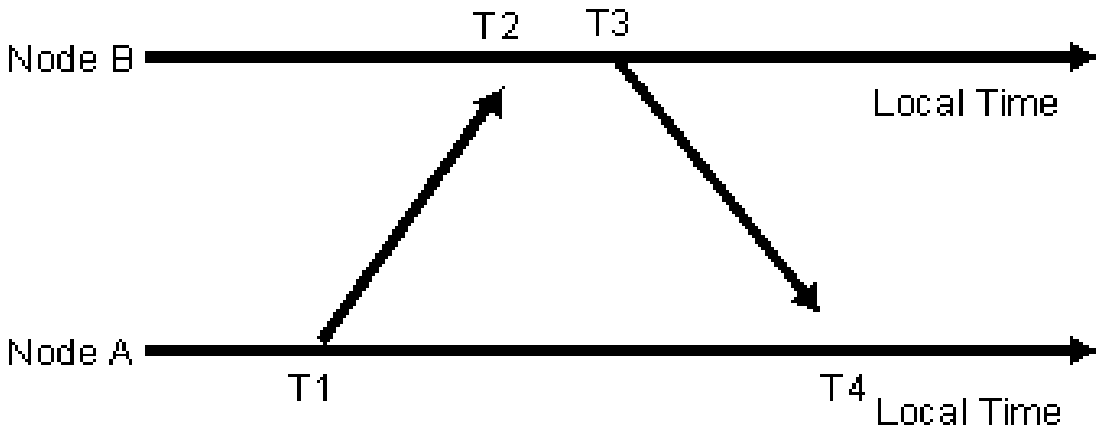

Figure 7 Two-way communications between nodes. [15]

Figure 7 illustrates the two-way messaging between a pair of nodes. By following this method, this messaging can synchronize a pair of nodes. 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 learning the drift, Node A can correct its clock and successfully synchronize to Node B. This is the basic communication for TPSN.

5.2.2 Advantages of TPSN

Any synchronization packet has the four delays that were discussed before: send time, access time, propagation time, and receive time. It would be a priority by eliminating any of these. Although TPSN does not eliminate the uncertainty of the sender, but minimize it. Also, TPSN is designed to be a multi-hop protocol; the transmission range is not an issue.

TPSN has uncertainty in the sender. It is different form RBS. They try to reduce this non-determinism by time stamping packets in the MAC layer. It is said that the sender's uncertainty contributes very little to the total synchronization error. By reducing the uncertainty with low level time stamping, it is said that TPSN has a 2 to 1 better precision than RBS. The sender to receiver synchronization is superior to the receiver to receiver synchronization.

TPSN was designed for multi-hop networks. Their protocol uses the tree based scheme then the timing information can accurately propagate through the network. The sender to receiver synchronization method is more precise than the receiver to receiver synchronization. [15]

5.3 FTSP

5.3.1 Introduction

Flooding Time Synchronization Protocol (FTSP) is a sender to receiver synchronization. It is an extendable, robust and steady protocol, which has high synchronization precision in wireless sensor networks. This protocol is similar to TPSN, but its disadvantages have been improved. It is similar in the fact that it has a structure with a root node. And all nodes are synchronized to the root. [16]

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 notices its local time when the message is received. The receiver can estimate the clock offset when having both the sender's transmission time and the reception time. The message is MAC layer time stamped, like in TPSN, on both sides of sending and receiving. [16]

FTSP is 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. It is mesh type topology instead of a tree topology as in TPSN of the network structure. Table 1 summarizes the magnitudes and distribution of the various delays in message transmissions.

Table 1 The sources of delays in message transmissions. [16]

Time	Magnitude	Distribution
Send and Receive	0 – 100 ms	nondeterministic, depends on the processor load
Access	10 – 500 ms	nondeterministic, depends on the channel contention
Transmission / Reception	10 – 20 ms	deterministic, depends on message length
Propagation	< 1μs for distances up to 300 meters	deterministic, depends on the distance between sender and receiver
Interrupt Handling	< 5μs in most cases, but can be as high as 30μs	nondeterministic, depends on interrupts being disabled
Encoding plus Decoding	100 – 200μs, < 2μs variance	deterministic, depends on radio chipset and settings
Byte Alignment	0 – 400μs	deterministic, can be calculated

5.3.2 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 could not handle topology changes well. TPSN would have to restart the level discovery phase if the root node changed or the topology changes. This would induce more network traffics and create additional overhead.

FTSP is utilizes the flooding of synchronization messages to combat link and node failure. That is robust in FTSP. The flooding also provides the ability for dynamic topology changes. It is need a dynamic topology, because the protocol specifies the root node will be periodically reelected. 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. [16]

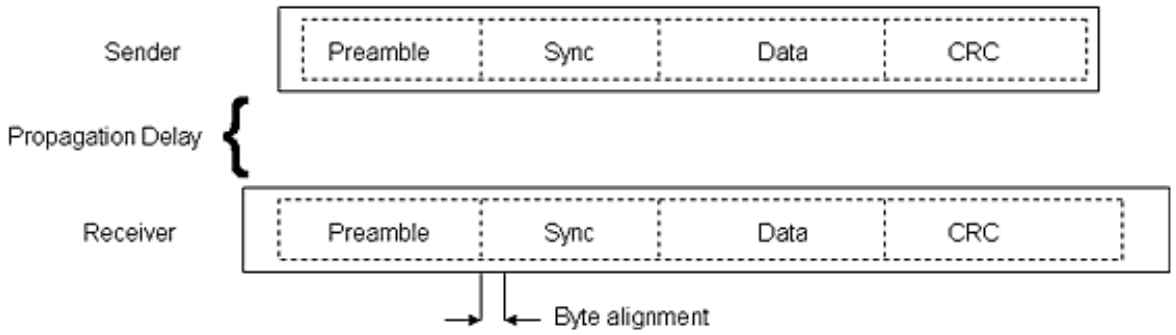

Figure 8 Data packets transmitted with FTSP

The data packets transmitted with FTSP are constructed as shown in Figure 8. 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. The receiver adjusts to the carrier frequency when the sender is transmitting the preamble bytes. 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.

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

5.4 IEEE1588

In Ethernet systems, unpredictable collisions due to the Carrier Sense Multiple Access with collision Detection (CSMA/CD) procedure may lead to time packages being delayed or disappearing completely. For this reason, IEEE1588 defines a special "clock synchronization" procedure.

Figure 9 shows the details of synchronization principle.

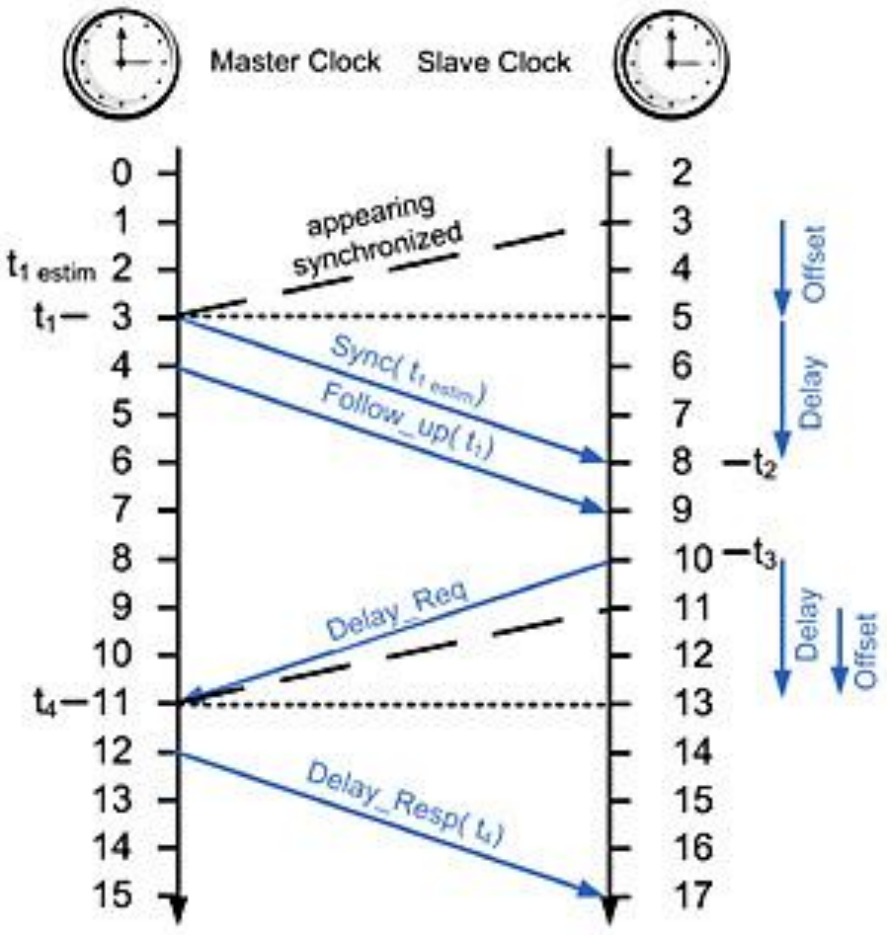

Figure 9 synchronization principle. [17]

First, one node (IEEE1588 master clock) transmits a sync telegram which contains the estimated transmission time. A clock gets the exact transmission time and then transmits it as a second follow up message. Based on these two telegrams and by means of its own clock, the receiver can now calculate the time difference between its clock and the master clock. To get the best results, the IEEE 1588 time stamps should be generated in hardware or as close as possible to the hardware. [17]

The telegram propagation time is determined cyclically in a second transmission process between the slave and the master (delay telegrams). The slave clock then corrects its clock and adapts it to the current bus propagation time.

5.5 HRTS

The centralized version of the Tsync-protocol is called for Hierarchical Referencing Time Synchronization protocol (HRTS) and the decentralized version for Individual Time Request (ITR) protocol. The key idea in HRTS is that the notion of hierarchical synchronization is combined with receiver-to-receiver synchronization and the performance is further improved by using dedicated MAC-layer channel for synchronization. In HRTS the synchronization is initiated by a designated root node wile in ITR any node can initiate a resynchronization.

5.5.1 Single Reference Nodes

In Figure 10, HRTS consists of three simple steps that are repeated at each level in the hierarchy. First, a base station, namely the reference node, broadcasts a beacon on the control channel (Figure 3(a)). One child node specified by the reference node will jump to the specified clock channel, and will send a reply on the clock channel (Figure 3(b)). The base station will then calculate the clock offset and broadcast it to all child nodes, synchronizing the first ripple of child nodes around the base station (Figure 3(c)). This process can be repeated at subsequent levels in the hierarchy further from the base station (Figure 3(d)). [18]

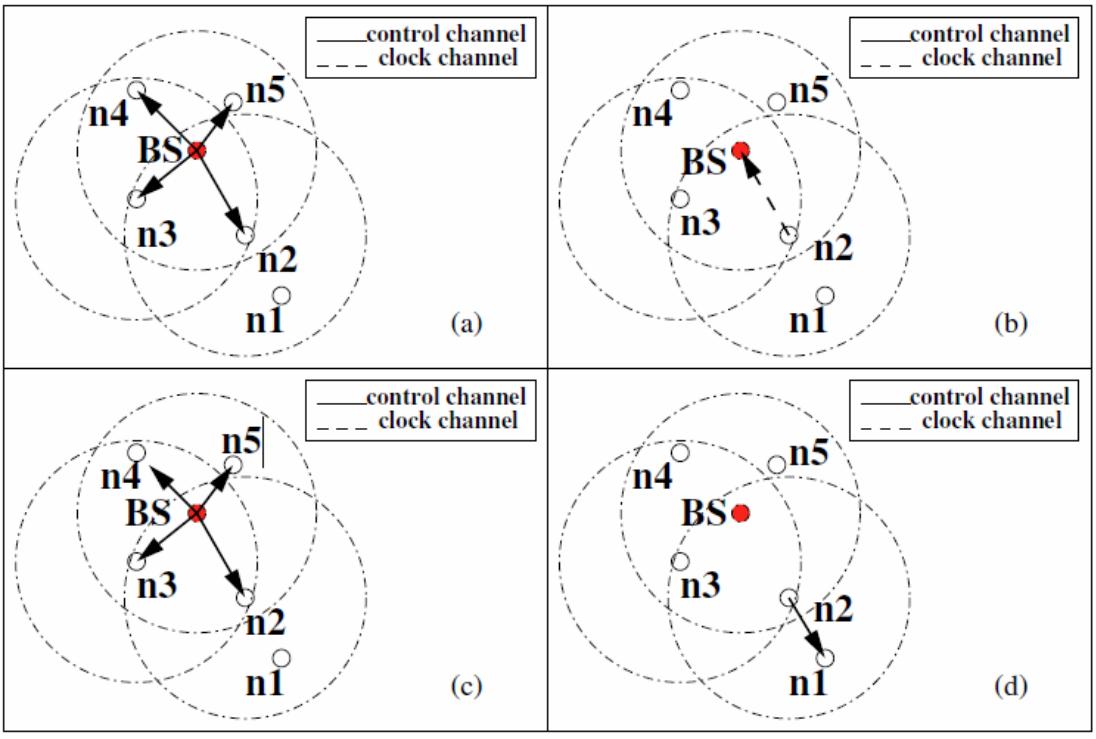

Figure 10 Push-based time synchronization: (a) Reference node broadcasts (b) A neighbor replies (c) All neighbors are synchronized (d) Repeat at lower layers [18]

5.5.2 Multiple Reference Nodes

By parameterizing each synchronization request, HRTS permits the existence of multiple reference nodes in the sensor network to provide accurate clock readings.

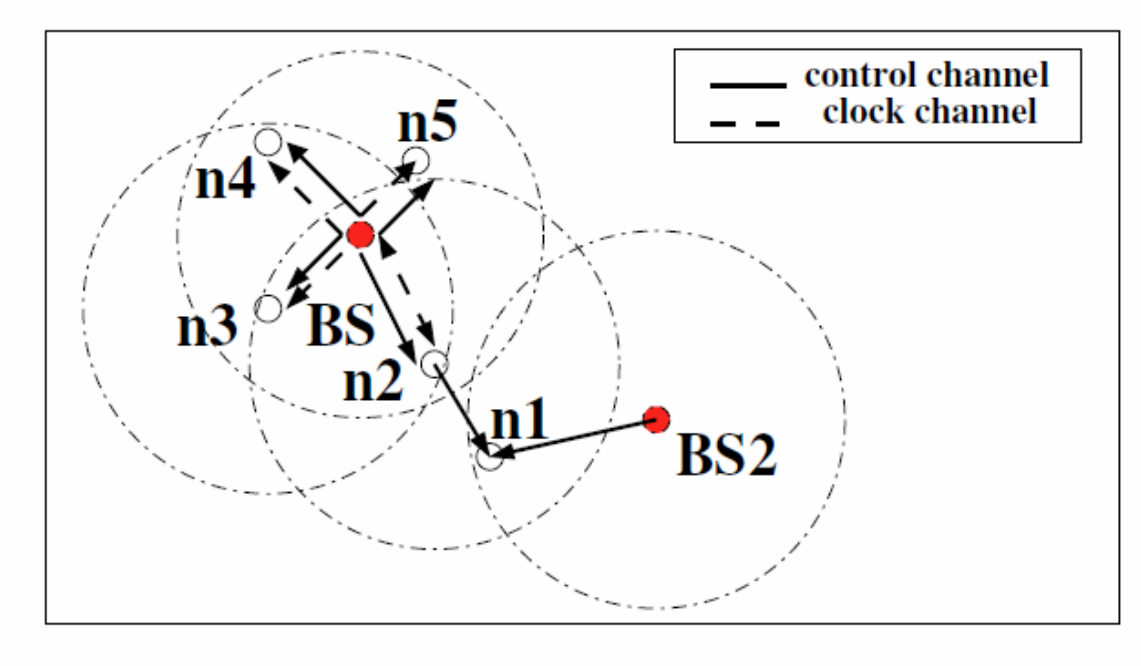

Figure 11 Synchronized by Multiple Reference Points [18]

In Figure 11, two reference nodes BS and BS2 exist in the sensor network. Node n1 is in the broadcasting domain of BS2 and 2 hops away from the BS. If the node n1 is updated by n2 before receiving a synchronization message from BS2, it will synchronize its clock again in response to a sync begin packet from BS2, because the ,level in the packet from BS2 is 0, which is smaller than that from n2. On the contrary, n1 will ignore n2's updating request if it is updated first by BS2. When there are several reference nodes existing in the network together, a shortest path tree is formed around each reference node. [18]

6. EXPERIMENTS ON PROTOCOLS

6.1 IEEE1588 Test

6.1.1 Introduction

This brief experiment describes some results obtained when connecting two Luminary boards and synchronizing them using the IEEE 1588 precision time protocol. Figure 10 shows the connection scheme:

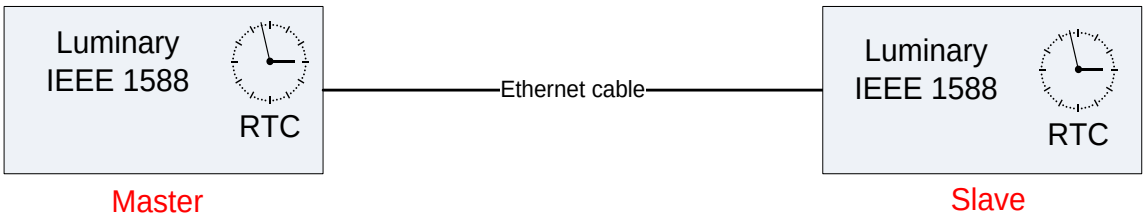

Figure 12 connection of two Luminary boards.

Experiment conditions were as follow:

- UART connection on the slave board in order to get the offset computation results at each resynchronization computation.
- An oscilloscope is connected to the PPS (Pulse Per Second) output of both boards.

6.1.2 Results

The results are presented on the next pages with the help of graphics showing the evolution of the computed offset value. 1561 offset values have been computed (in ns) following to the resynchronization rate which seems to be 1 per second (to be verified). The test has thus been performed during approximately 25 minutes.

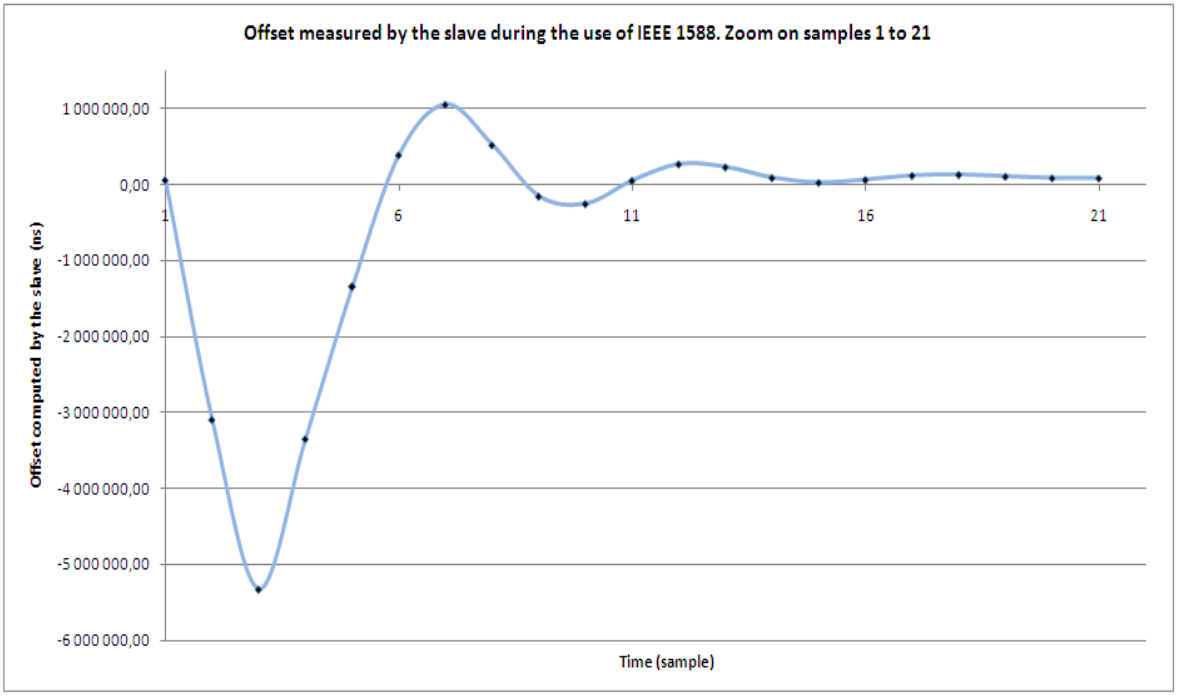

Figure 13 Offset measurement 1.

Figure 13 shows the effect of a filter implemented in the code (PI controller). The offset is high since this situation is only the first seconds after the start of the protocol.

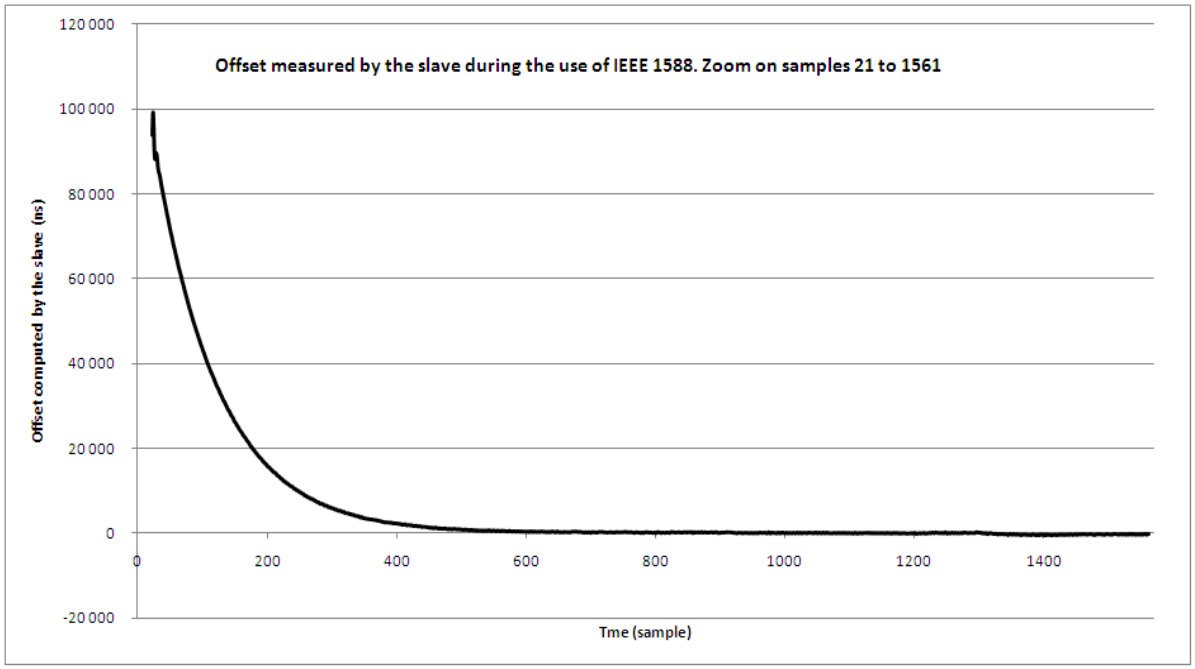

Figure 14 Offset measurement 2.

Obviously, in Figure 14 the offset tends to zero, which is the purpose.

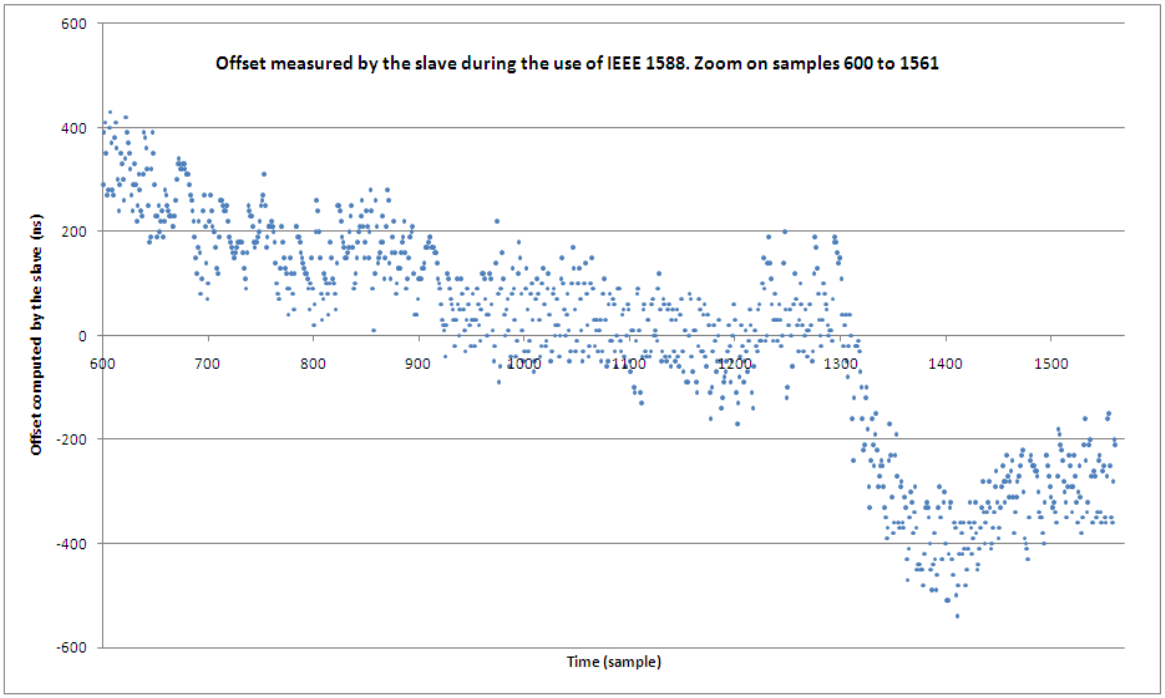

Figure15 Zoom on the samples after 600 offset computations.

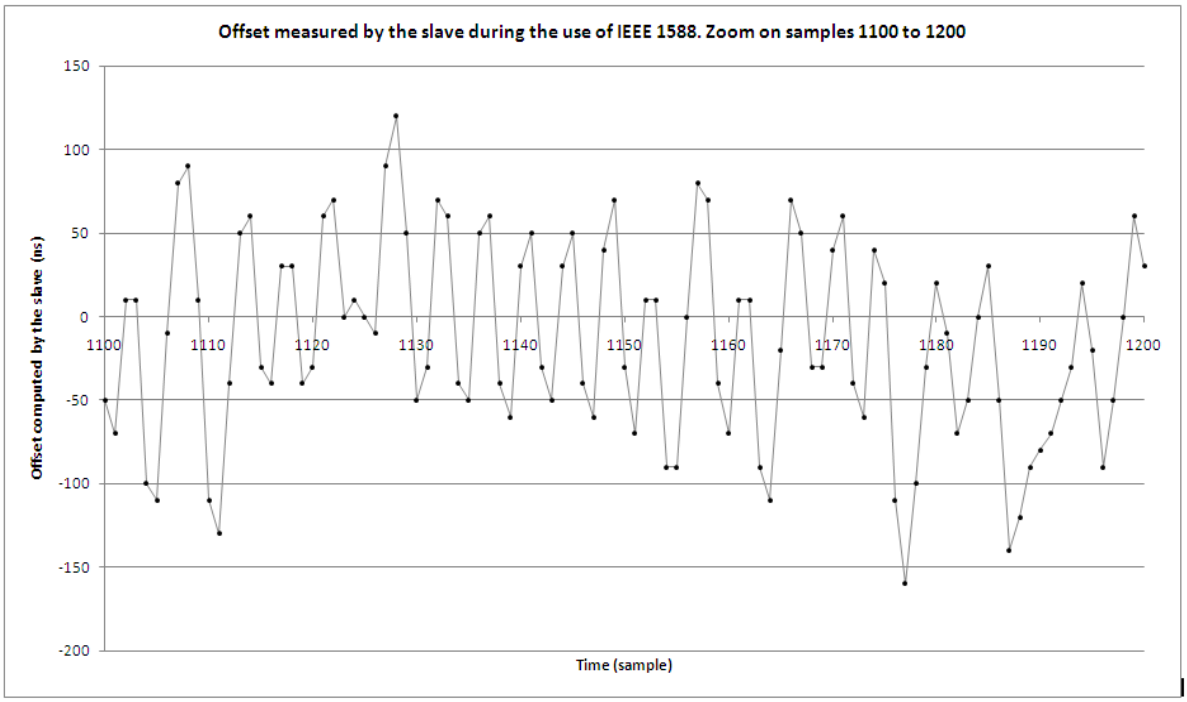

Figure 16 Zoom on the region 1100 à 1200 samples where the offset turns around zero.

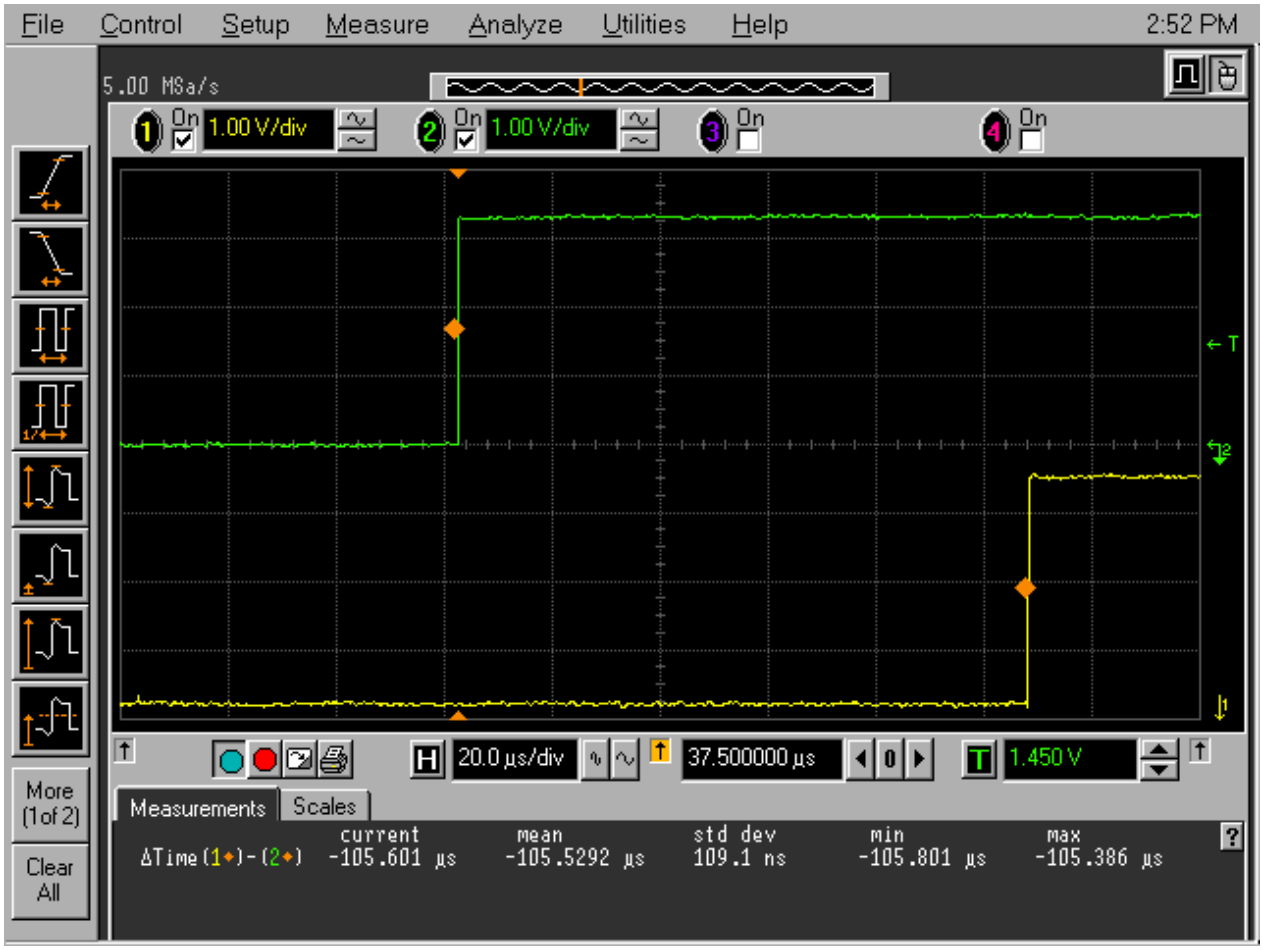

Figure 17 Results from the oscilloscope.

When the slave thinks that its offset between it and the master is around zero (capture taken between samples 1100 and 1200, see the Figure 15 and Figure 16), in Figure 17the oscilloscope shows a 105µs delay between the PPS of the master (green) and the slave (yellow). It is also interesting to notice that when the offset computed was around 10,000 for example, the observed delay was around 95µs, only 10µs away from the final value.

7.1.3 Conclusion

It should be noticed that when the offset reaches zero, the observed differences between the PPS are around 105µs with a very small standard deviation around 100ns on the capture. The origin of the offset is not yet known. Two hypotheses are now considered:

- Hardware difference between the two boards influencing the creation of the pulses.

- Due to their different mode, the master and the slave program do not perform the same actions. It is possible that the creation of the PPS for the slave is delayed by other operations with a higher priority.

Some new experiments are planned to test these two possibilities. Respectively:

- Testing another luminary board for the slave and verifying if the delay observed is still the same.
- Testing the synchronization with one master and two slaves. The use of a switch is required engendering a higher standard deviation of the delay. However, comparing the PPS from both slave nodes can help us to verify the second hypothesis. [19]

6.2 IEEE1588 Test from the Internet

6.2.1 Introduction

This brief experiment shows an automatic evaluation system for IEEE1588 synchronization clock unit. Figure 16 shows the system structure.

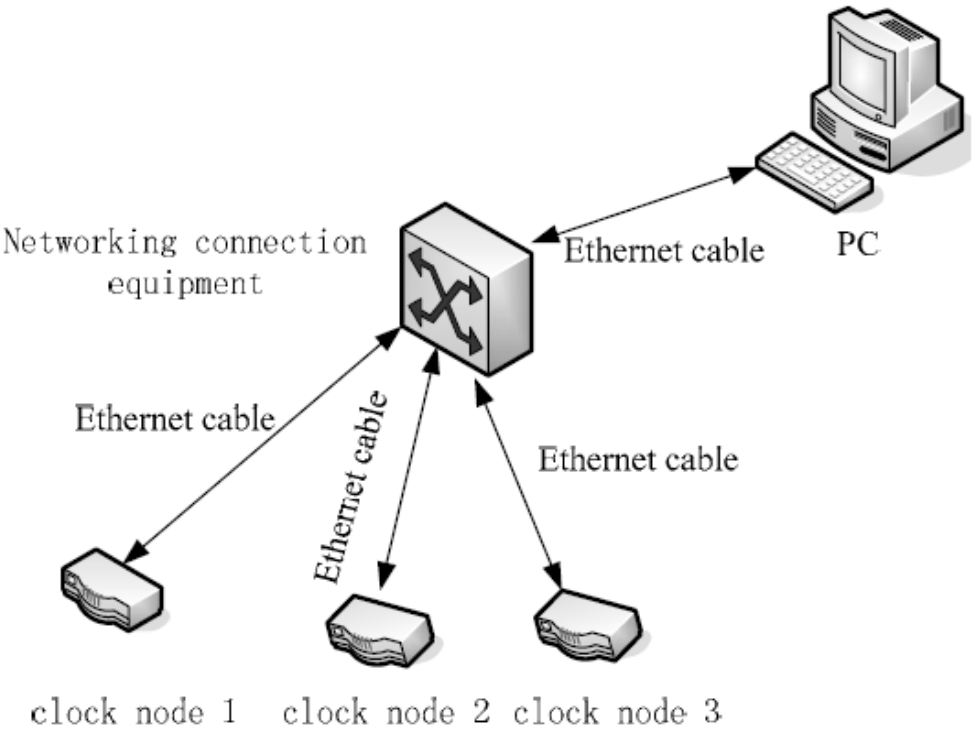

Figure 18 System structure.

6.2.2 Experiment Conditions

This platform consists of a computer, a master clock and a two slave clocks module connected via different networking connection equipments as shown in Figure 18.

6.2.3 Results

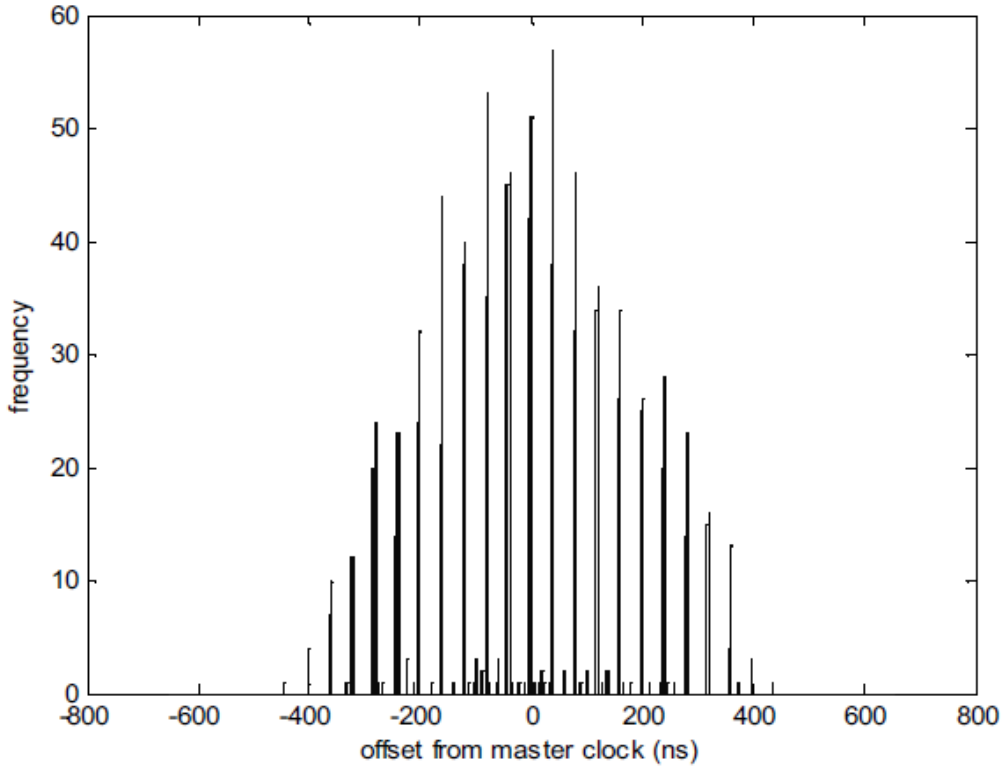

Figure 19 Synchronization accuracy of the first slave clock after start-u

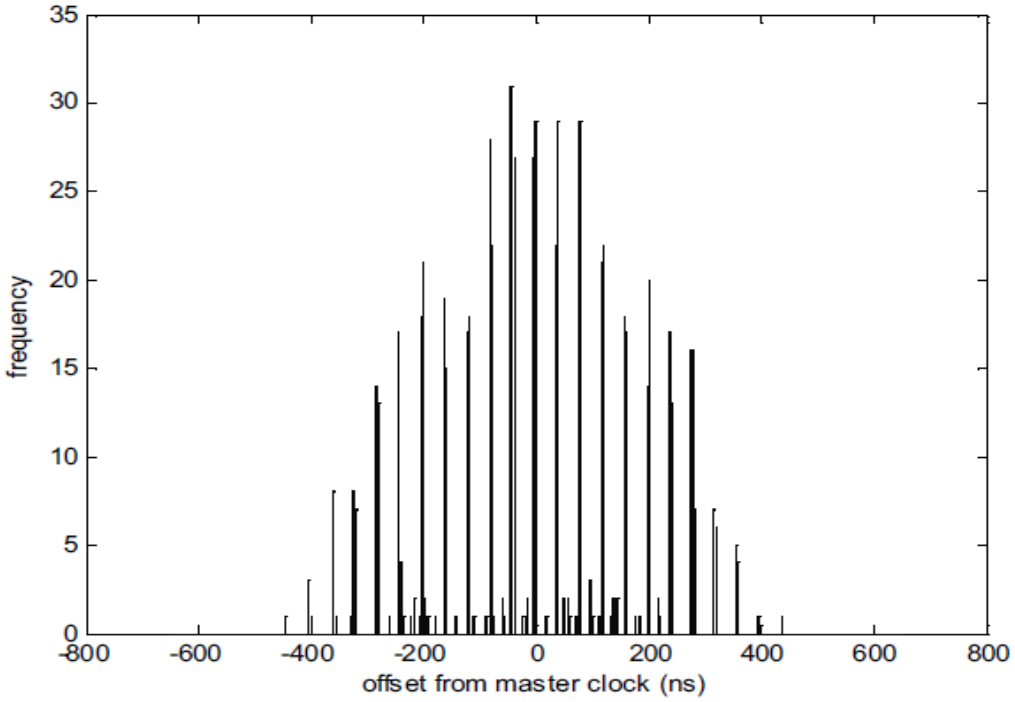

Figure 20 Synchronization accuracy of the second slave clock after start-up.

The histogram in Figure 19 shows the offset between the first slave clock and master clock under stable conditions. The significant variation lies within ±400ns. The fluctuation can be explained as the combination of the switch jitter and propagation jitter. The histogram in Figure 18 shows the offset between the second slave clock and master clock under stable conditions. The evaluation of the first and the second slave clock are done in parallel. It can be seen from the two histograms that their synchronization performance is almost the same.

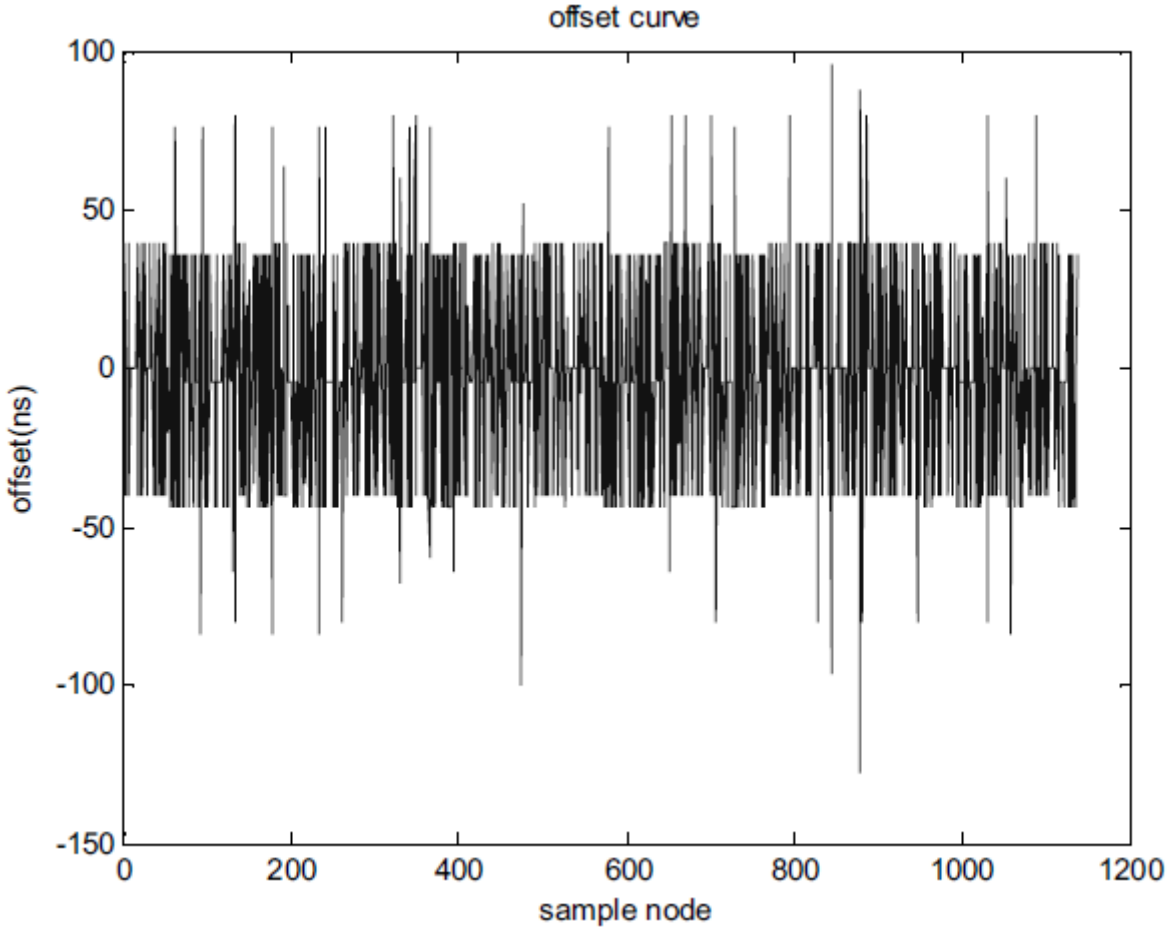

Figure 21 Offset curve.

Figure 21 shows the most of offset is within 80ns. The histogram in Figure 22 shows the offset between the first slave clock and master clock under stable conditions. The signification variation is within ±40ns. The synchronization performance of two slave clocks

is almost same.

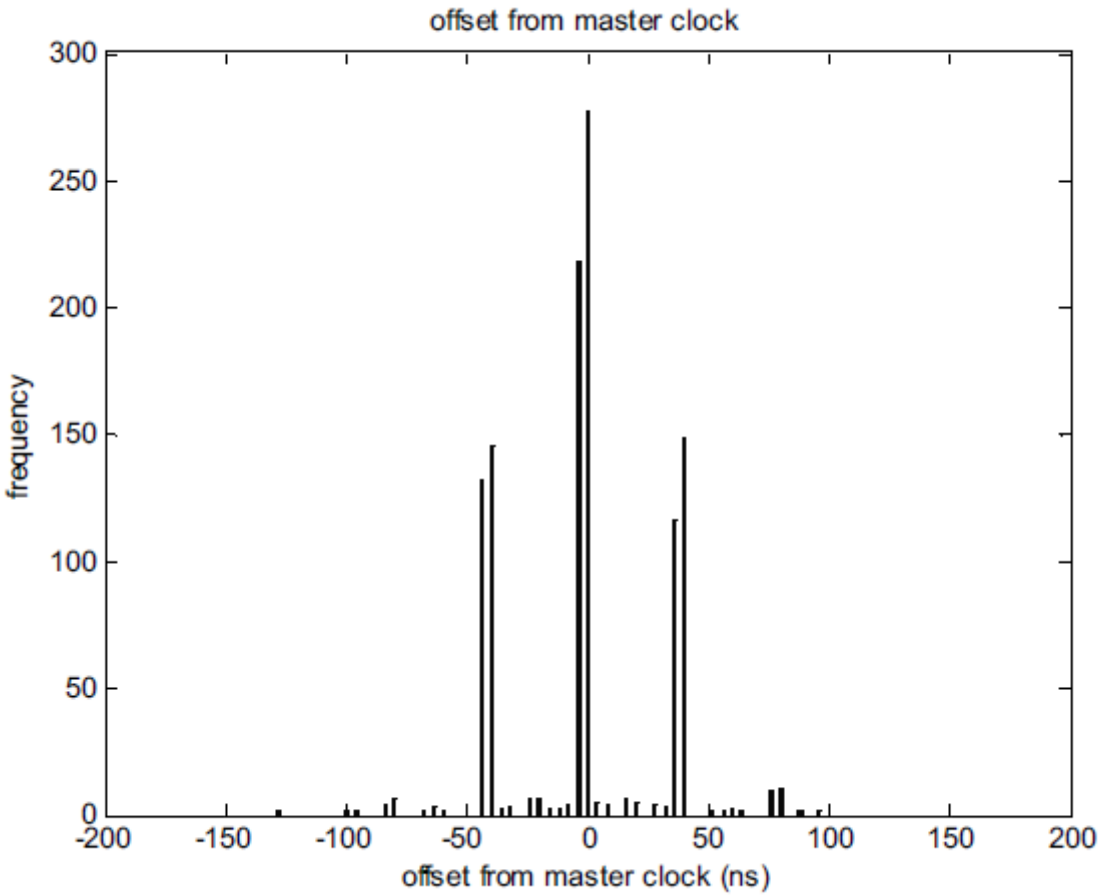

Figure 22 Synchronization accuracy after start-up.

Table 1 and Table 3 show that the performance of synchronization system is different when using different network connecting equipment under different Sync intervals. The number of sample points is more than 1,000.

Table 2 Performance of system with AFS-1008 switch.

Interval	**1s**	**2s**	**4s**	**8s**	**18s**
mean(ns)	-0.07	-1.58	-2.37	-1.92	-0.79
standard deviation(ns)	176.77	173.74	178.02	174.20	179.83

Table 3 Performance of the system with a H3C S1024R switch.

Interval	1s	2s	4s	8s	16s
mean(ns)	-3.54	-1.77	-2.24	-1.79	-1.24
standard deviation(ns)	37.67	32.22	34.57	33.94	37.76

6.2.4 Conclusion

The result of this experiment shows that it can configure the operating parameter. It can be configured very convenient on every clock node. The real-time offset information and its statistical parameters of every slave clock can be observer intuitively while the performance of synchronization system is not affected. [20]

6.3 RBS Test

6.3.1 Introduction

This brief experiment shows how the clock drift between Luminary boards can be evaluated following the RBS beacon principle. Figure 21 shows the system structure.

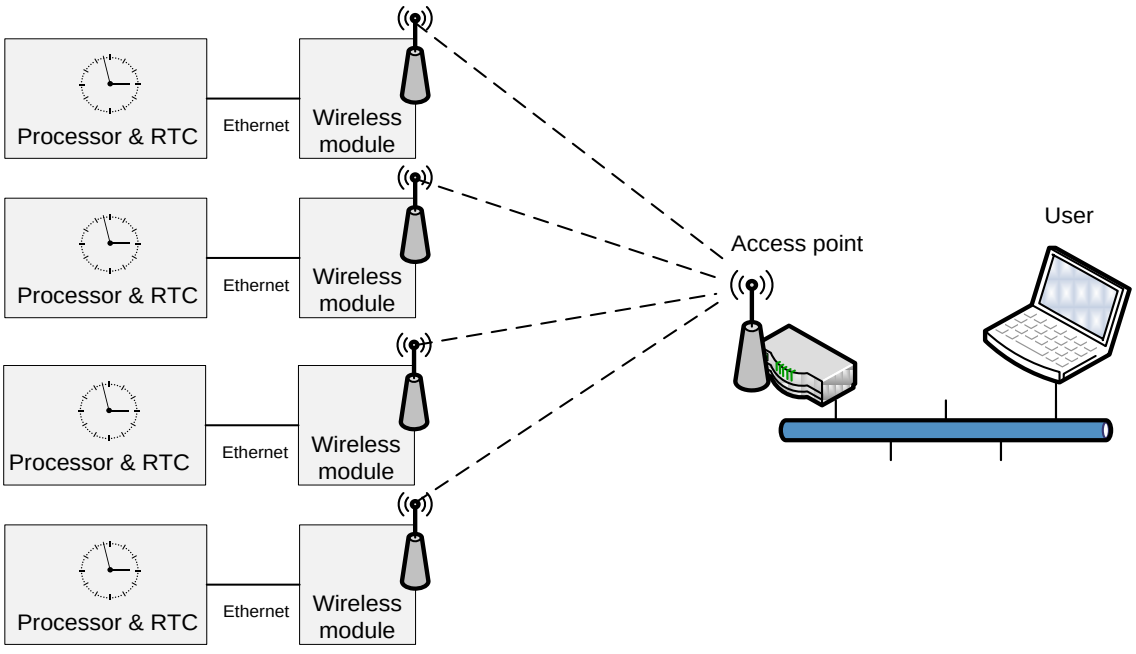

Figure 23 system structure.

Experiment Conditions:

- The USB connection to each board allows us to get the timestamp generated when the beacon is received by the four slaves.

- The user's computer implements a modified IEEE 1588 protocol in a master mode. The SYNC message acts as the RBS beacon since it is broadcasted to all slaves. The latter implements also a modified IEEE 1588 protocol which does not computed the offset and does not correct the clock but only receive the SYNC message and sends the timestamp to the UART output. The others messages are dropped or not sent.

6.3.2 Results

The results are presented in Figure 22 with the help of graphics showing the evolution of the received timestamps. 4000 values have been received. The SYNC message is sent every 2 seconds. Duration of the experiment: 2h10. To name the boards, it can use the name of the COM port used for getting the data. There is: COM19, COM15, COM17 and COM21.

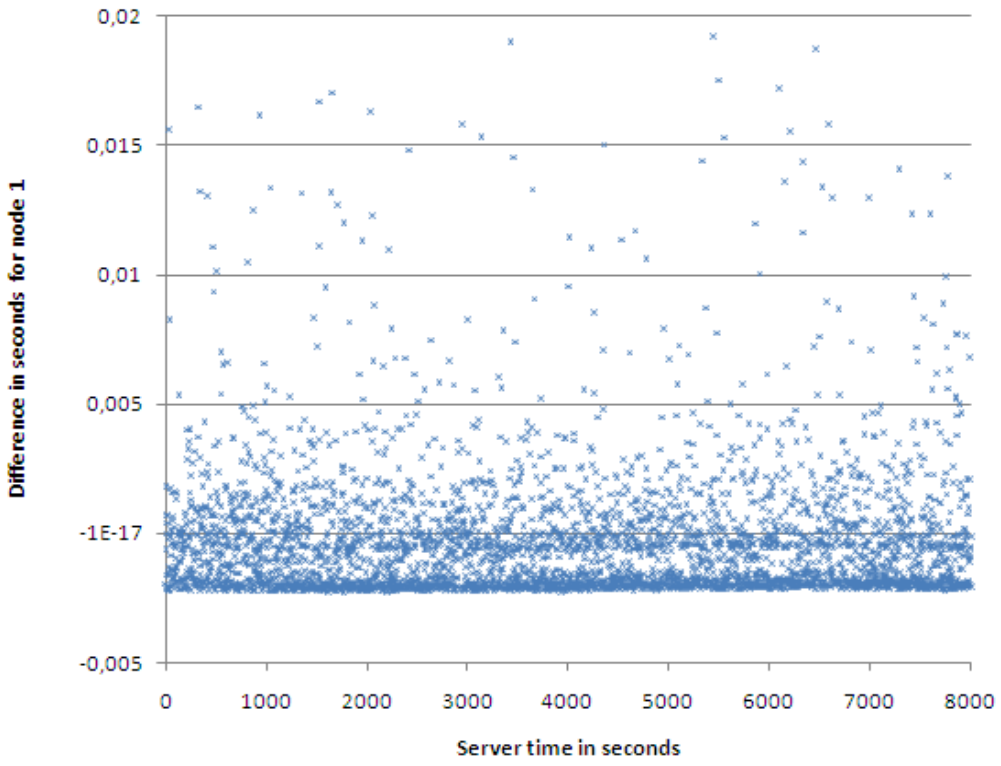

Figure 24 Difference between the regression line and the real timestamp values for node 1

The server sends a message every two seconds. The reception of it by a slave is time-stamped, allowing drawing the graph of the timestamp value in function of the server time. The slope of this line enables to know the drift between server and slave thanks to a regression line. The graph above represents the difference between the regression line and the real timestamp values for node 1. It is obvious the SYNC message transmission time is subject to a lot of variations, as already noticed with the PTP. There is also a minimal value due to an immediate transmission. Indeed, the variations of the transmission observed are probably due to the Carrier Sense, Multiple Access and Collision Avoidance (CSMA/CA) principle of the Wi-Fi protocol which introduces random delays.

Table 4 Slope of the regression line Timestamp values in function of server time

	Node 1	**Node 2**	**Node 3**	**Node 4**
Slope	0,999920708	0,999924523	0,999921456	0,999923072

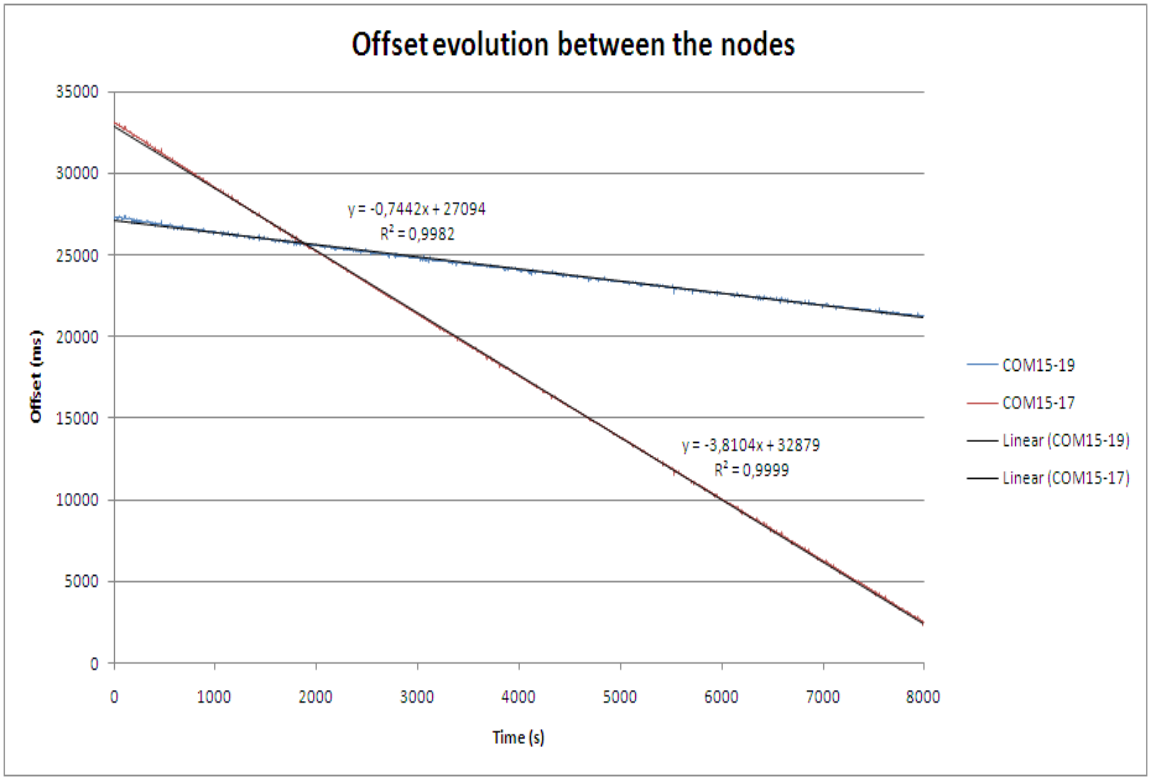

Figure 25 Offset evolutions between the nodes.

As shown in Figure 25, each node timestamps the receiving time of the SYNC messages. The results above shows the difference between the timestamps received. Obviously there is a drift defined by the slope of the regression lines. These values are logical compared to the experiments previously done with an oscilloscope (see folder 110407, file drift.xlsx). The objective of RBS is to know the slope in order to be able to convert the timescale of one node to another.

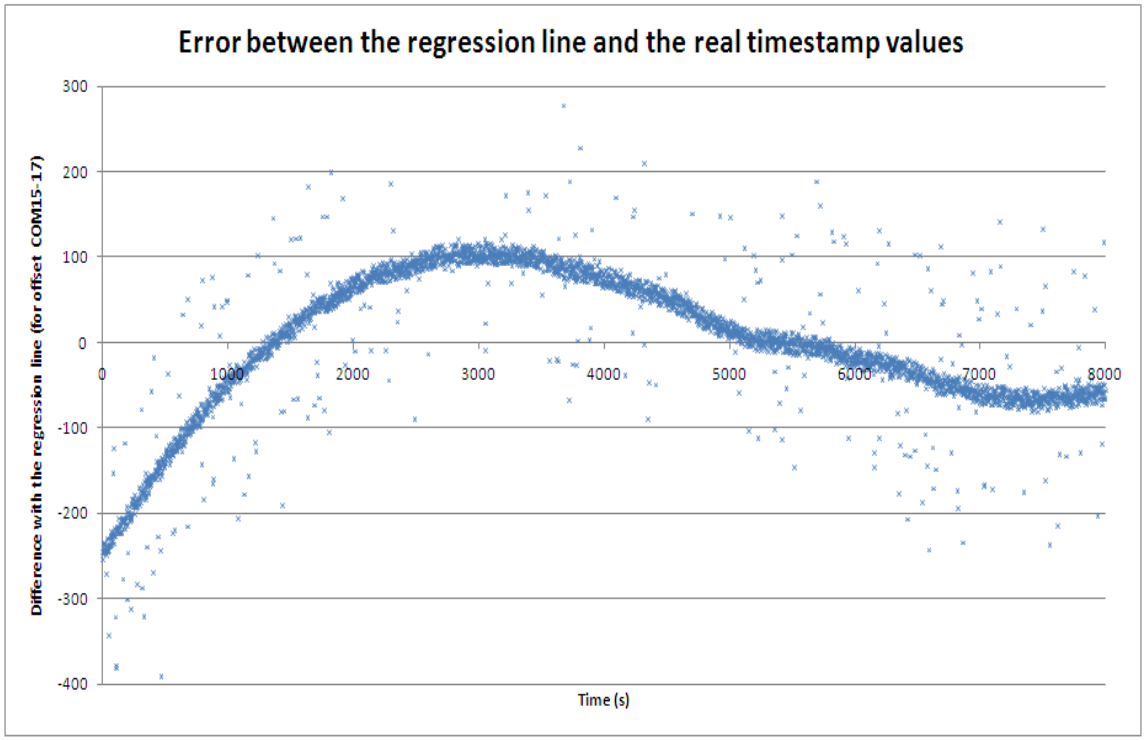

Figure 26 Error between the regression line and the real timestamp values.

In Figure 26, the curve shows the difference between the regression straight line and the real values of the offset curve COM15-17. It reveals that the drift varies in time. Then, approximating the drift using all values according to a regression line is not the best solution.

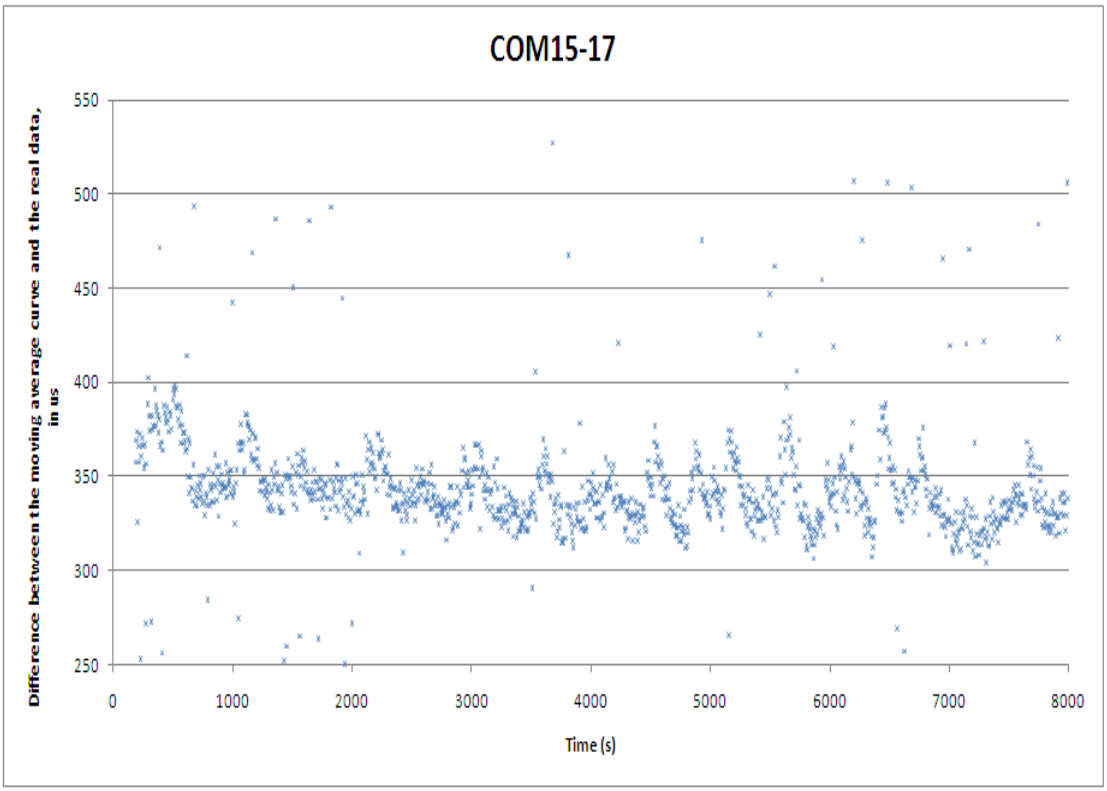

Figure 27 Difference between moving average curve and real data.

In reality, the current emission rate for RBS beacon is about 30 beacons in 3 minutes or 1 per 6 seconds. A regression straight line is computed using the last 30 timestamps received. The Figure 26 and Figure 27 show the difference between the moving average over 30 points and the real data of the offset between nodes COM15 and 17. This time, the drift variation is not visible since the moving average takes care of that. The samples considered here are taken every 6 seconds instead of 2 as before. The "distance" between the two curves has an average of around 350us. It is normal since it is the delay of computation of the moving average over 30 points. The standard deviation is around 35us, which is quite good.

6.3.3 Conclusions

These experiments show that the estimation of the drift using the RBS principle seems more promising than the IEEE 1588 protocol. However the results shown are not completely fitting the specifications of RBS. Indeed, the last graph uses the moving average of over 30 points instead of regression lines over 30 points. If simulate the use of regression lines to timestamp the samples from the sensors, steps are observed during the transition from one line to a newer one after the reception of a new beacon. Figure 26 show an example of these transitions:

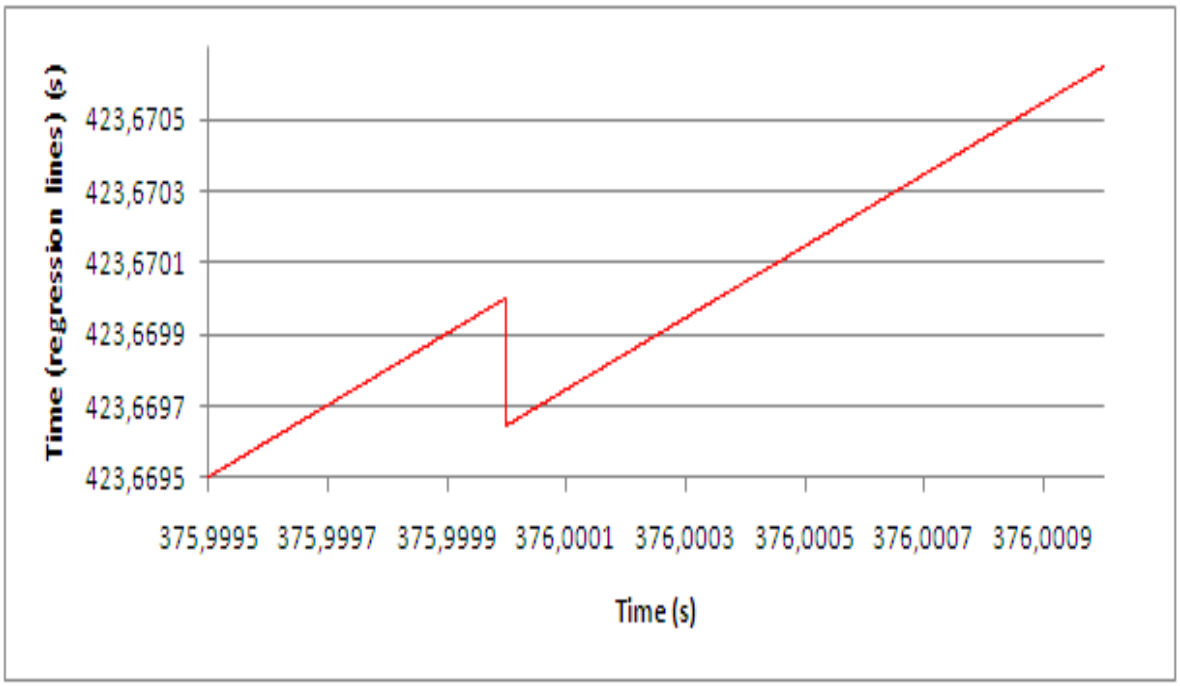

Figure 28 Time regression lines.

The first line is the extrapolation of the regression line computed using 30 timestamps (COM15). The second line is the newer one computed using the new timestamp received and the 29 previous ones. The conclusion of this is that, just before and after this transition, the sensor's data is going to be time-stamped with the same timescale resulting in possible permutations of some samples. Sometimes the step is positive and not negative. Therefore, there are no inversions but a small jump in time.

During the experiments, there is no traffic on the network. Probably, it is adding traffic will lead to less accurate results. That is why using a channel independent of the traffic can lead to very precise results. [21]

6.4 RBS Test from the Internet

6.4.1 Introduction

The brief document shows how RBS can be used to estimate the relative phase offset among a group of receivers. And also describe a slightly more complex scheme that corrects for clock skew.

6.4.2 Results

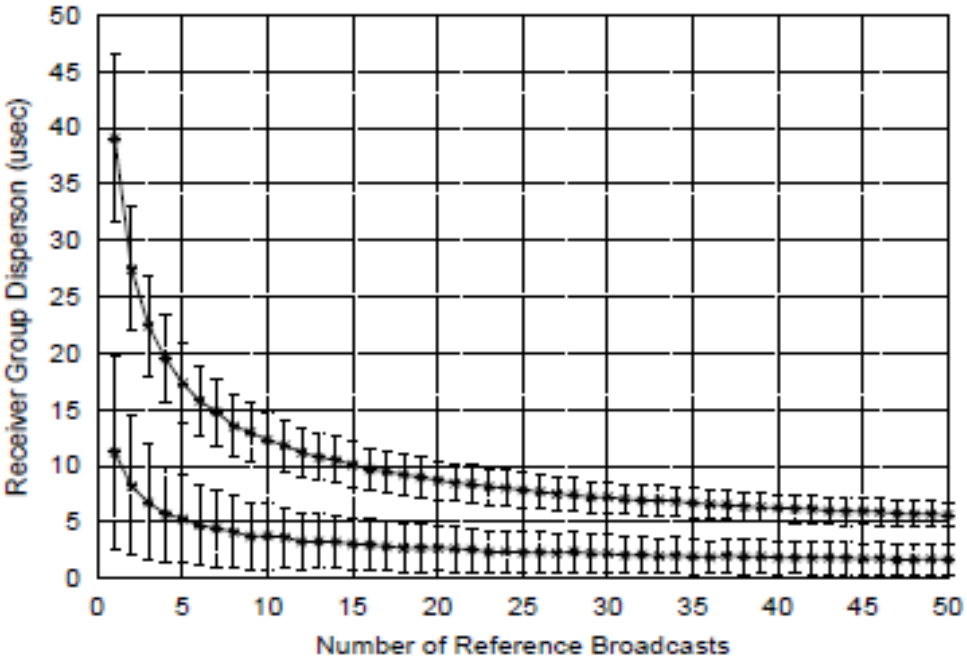

Figure 29 2-D view. Analysis of RBS algorithm for single broadcast domain (no clock skew).

In Figure 29, mean group dispersion from the average of 1000 simulated trials for 20-receiver group (top) and 2-receiver group (bottom).

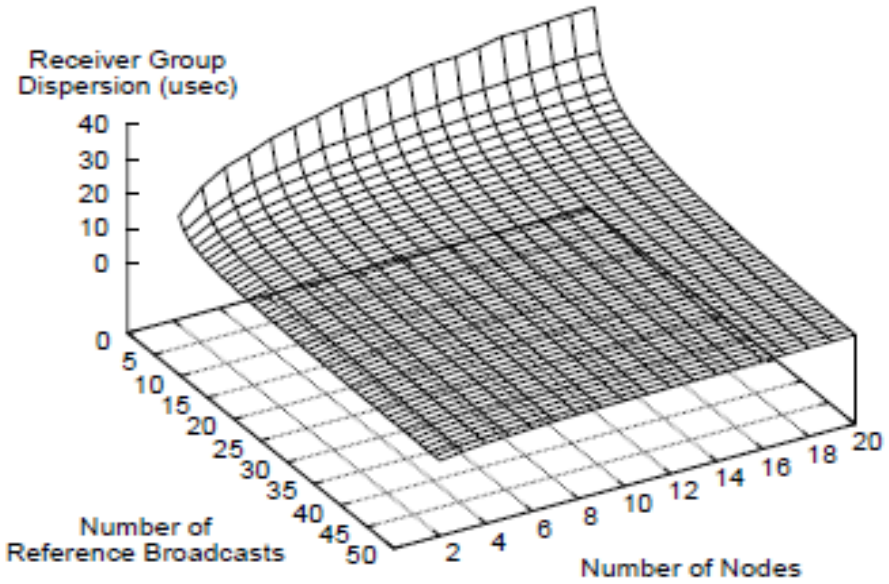

Figure 30 3-D view. Analysis of RBS algorithm for single broadcast domain (no clock skew).

In Figure 30, mean group dispersion from the average of 1000 simulated trials for the same data set, from 2 to 20 receivers (inclusive)

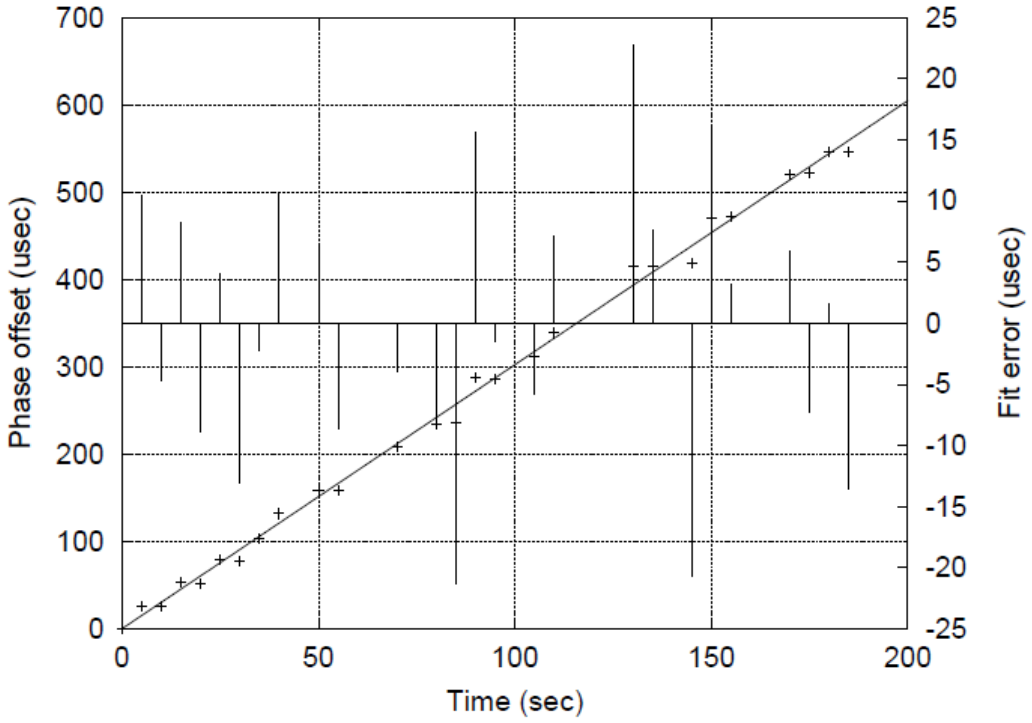

Figure 31 Synchronization of the Mote's internal clock

Figure 31 shows an analysis of the clock skew's effect on RBS. Each point represents the phase offset between two nodes as implied by the value of their clocks after receiving a reference broadcast. A node can compute a least-squared-error fit to these observations (diagonal lines), and thus convert time values between its own clock and that of its peer.

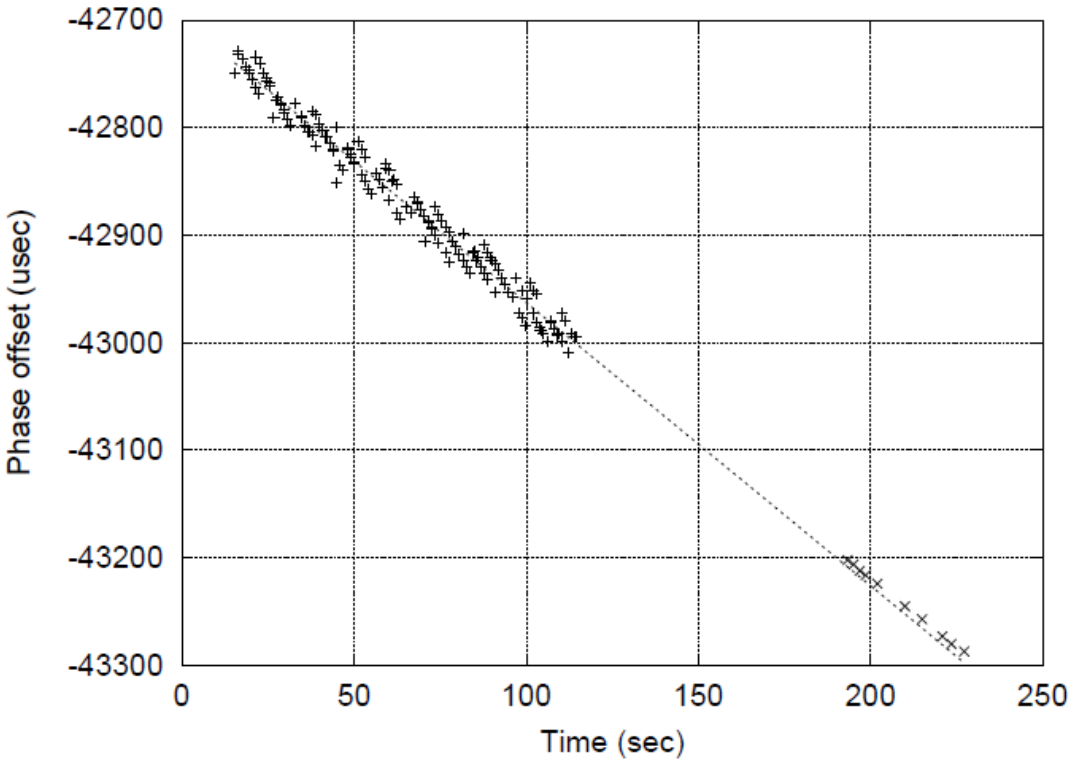

Figure 32 analysis of RBS algorithm for single broadcast domain (with clock skew).

Figure 32 shows synchronization of clocks on PC104-compatible single board computers using Mote as NIC.

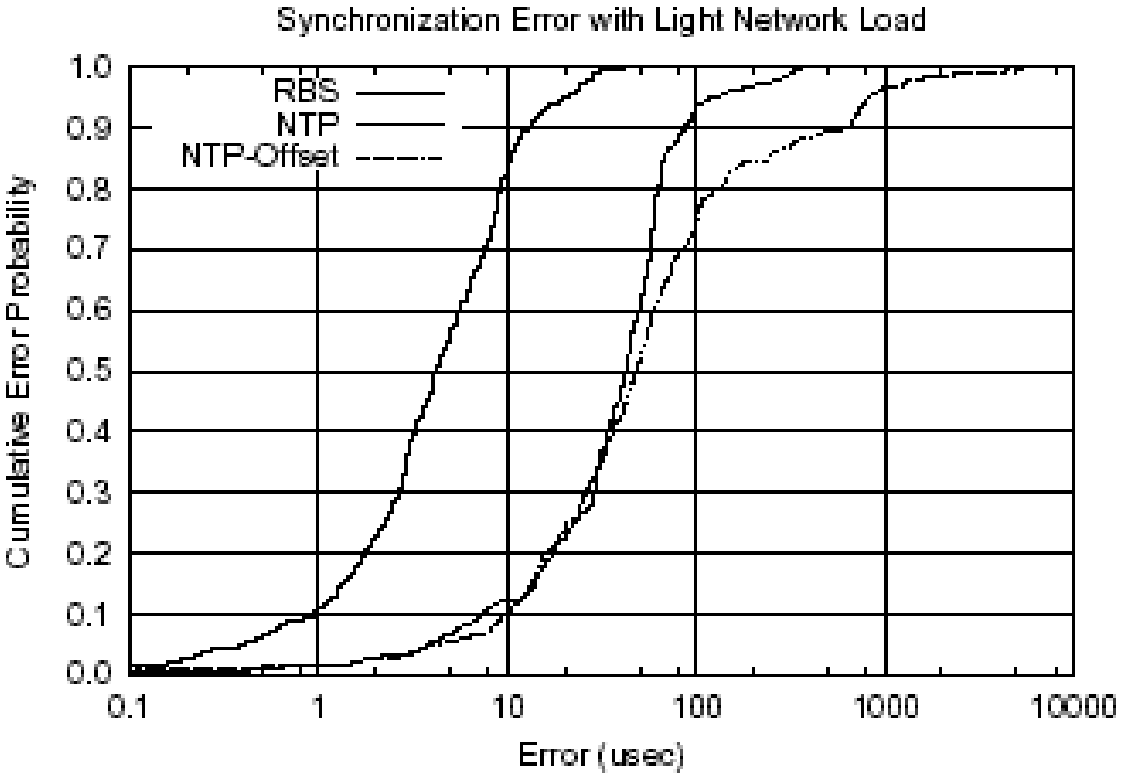

Figure 33

Table 5

Methods	Mean Error	Std Dev	50%	95%	99%
RBS	6.29	6.45	4.22	20.53	29.61
NTP	51.58	53.30	42.52	131.20	313.64
NTP-Offset	204.17	599.44	48.15	827.42	4334.50

Figure 33 and Table 5are both for light traffic. Clock resolution was 1 µsec. All units are µsec. "NTP-Offset"uses an NTP-disciplined clock with a correction based on NTP's instantaneous estimate of its phase error; unexpectedly, this correction led to poorer synchronization. RBS performed more than 8 times better than NTP on a lightly loaded network.

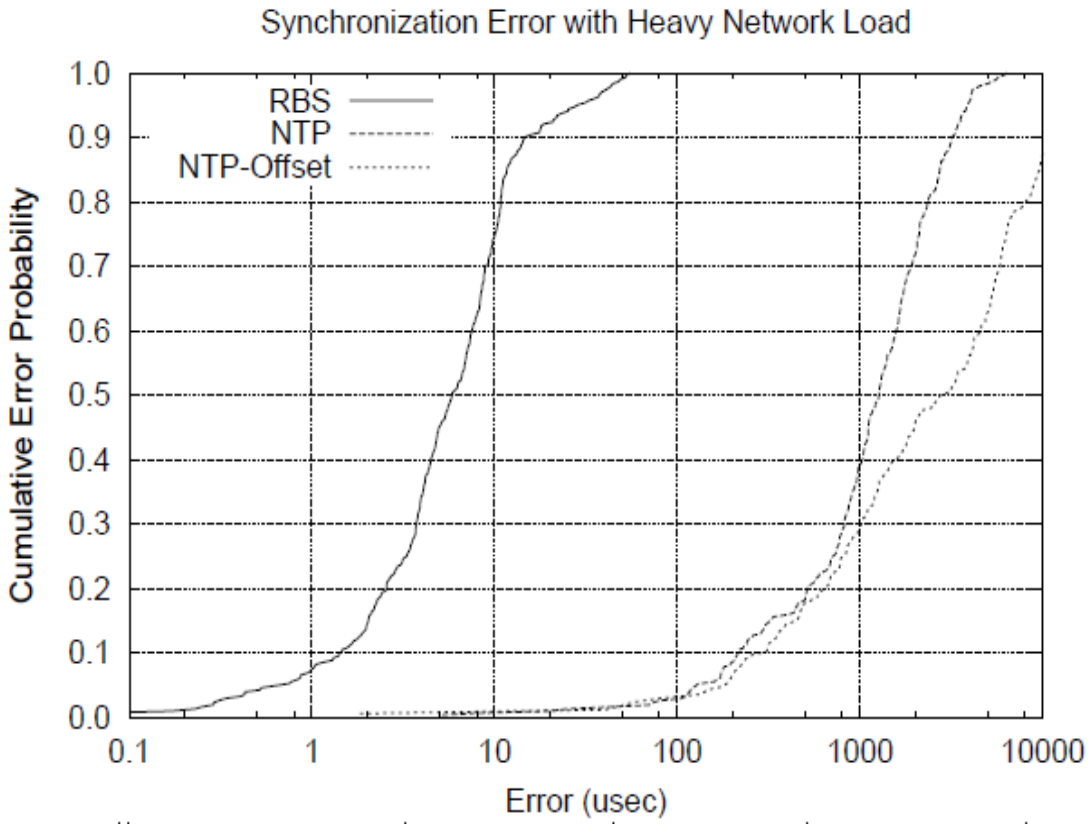

Figure 34

Table 6

RBS	8.44	9.37	5.86	28.53	48.61
NTP	1542.27	1192.53	1271.38	3888.79	5577.82
NTP-Offset	5139.08	6994.58	3163.11	15156.44	38897.36

Both Figure 34 and Table 6 show the heavy traffic. It is similar to that in Figure 31 and Table 5, but in the presence of cross-traffic with an average aggregate offered load of approximately 6.5Mbit/sec. On this heavily loaded network, NTP further degraded by more than a factor of 30, while RBS was virtually unaffected.

6.4.3 Conclusion

In this test, there explored two different forms of post-facto synchronization. The first, single-pulse synchronization requires frequency calibration at the time the network is deployed (and perhaps periodically afterward). With this information, high-precision retrospective timescales can be

quickly constructed using a single packet. Second, it has shown that Reference-Broadcast Synchronization can be used for post-facto synchronization. In our test, the phase error of a retrospective RBS timescale was still less than the bit-detection jitter when 60 seconds elapsed between the last synchronization packet and first time-stamped event. [22]

6.5 FTSP Test

6.5.1 Introductions

This brief experiment described the drift management, and gives the percentage of synchronized nodes in FTSP protocol. Figure 35 show the system structure.

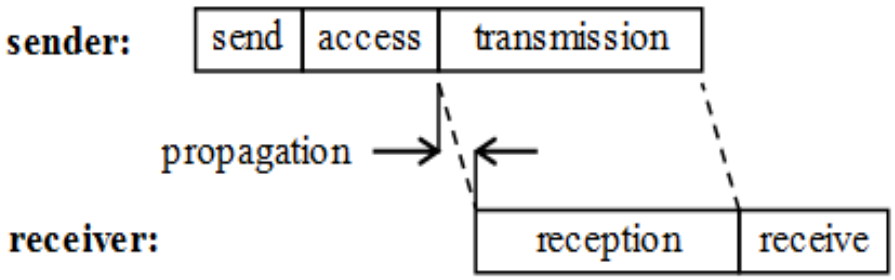

Figure 35 system structure.

6.5.2 Results

The implementation of FTSP on the Mica and Mica2 platforms was used to carry out the experiments described in this section is available on internet. It tested the protocol focusing on the most problematic scenarios, such as switching off the root of the network, removing a substantial part of the nodes form the network, so that the remaining nodes still formed a connected network, and switching on a substantial number of the new nodes in the network.

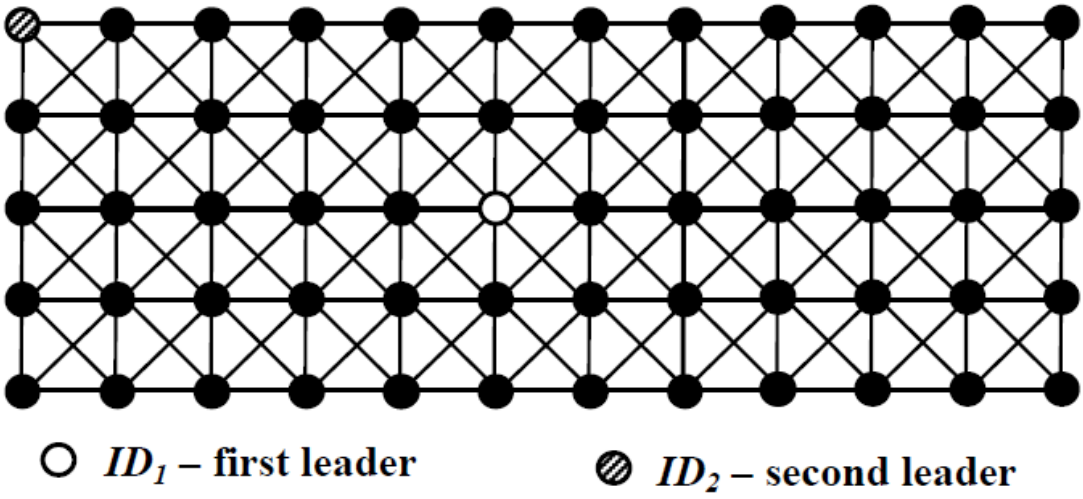

Figure 36 The layout and links of the experimental setup.

Figure 36 show that each node can only communication with its (at most 8) neighbors.

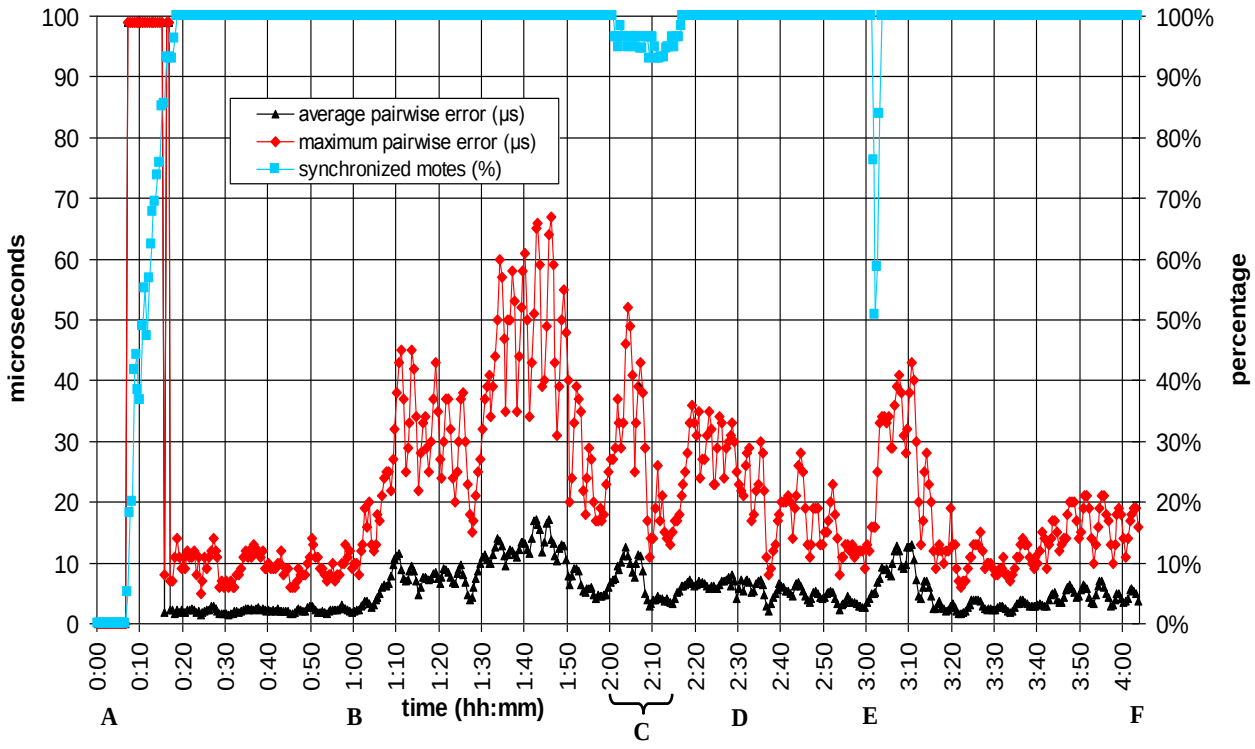

Figure 37 FTSP experimental evaluations.

In Figure 37, a 5x12 grid experiment shows the percentage of synchronized nodes, the maximum and average error (the maximum and average of the pair wise differences of the reported global times). At time A, the node was switched, the root ID1 was switched off at B, randomly selected motes were reset during C, half of the motes were switched off at D, and the same motes were

switched back on at E, and the experiment ended at F.[23]

6.5.3 Conclusions

The FTSP was tested and its performance was verified in a real world application. This is important because the service have to operate not in isolation, but as part of a complex application where resource constraints as well as intended and unintended interactions between components can and usually do cause undesirable effects. [24]

7. COMPARISON OF RBS , FTSP AND IEEE1588

The differences between RBS and FTSP can be summarized in Table 7.

Table 7 Differences between RBS and FTSP. [25]

	RBS	FTSP
Uncompensated delays	propagation, decoding, byte alignment, interrupt handing and receive times	propagation time
Network overhead	1.5 msgs per synchronization period	1 msg per synchronization period
Hierarchy	clustered with timescale transformation	flooding

Table 4 show an analysis of the differences of these three protocols. I, N represents the number of exchanged packets in a synchronization cycle and L the number of network nodes.

Table 8 Classification on synchronization and energy issues. [26]

Synchronization / Issue / Protocols	Scheme used	Number of message	Time clock precision	Implementation level	Power consumption
IEEE1588	Single hop	4*N*L	200 ns	App/ Physical	High
RBS	Receiver to Receiver	N*L	29.1 μs	App	High
FTSP	Sender to Receiver	N*L	1.7μs	MAC	Low

8. CONCLUSION

As in many other distributed systems, time synchronization is an important service in wireless sensor networks. Particularly, wireless sensor networks make extensive use of synchronized time in many contexts (e.g. for data fusion, TDMA schedules, synchronized sleep periods, etc.). Most wireless sensor network applications are targeted at retrieving information from surrounding environments. In many situations, the temporal property of a physical event is critical to wireless sensor network applications. Existing time synchronization methods were not designed with wireless sensors in mind. It needs to be extended or redesigned. The solution centers on the development of a deterministic time synchronization method relevant to wireless sensor networks. Highly related for sensor networks, it also provides tight, deterministic bounds on both sides of the offsets and clock drifts

In this thesis, some common uses of synchronized time in sensor networks were described. It was also described how synchronized time is a critical service in sensor networks. It is a basic requirement for virtually all algorithms that seek to reason about time-varying observations made by distributed sensors.

After studying time synchronization for wireless sensor network, the standards of IEEE1588, NTP and PTP were studied. Time synchronization protocols that were mentioned in the thesis are all based on these standards. Time Synchronization in wireless networks is extremely important for basic communication, but it also provides the ability to detect movement, location, and proximity.

There are many synchronization protocols and they 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. In this thesis, three protocols which are the major timing protocols currently in use for wireless networks were studied. They are IEEE1588, Reference Broadcast Synchronization (RBS) and Flooding Time Synchronization Protocol (FTSP).

Through the experiments, the main idea was to compare the differences of these three protocols. Depending on different situations, the users can choose any of these protocols to measure the exact time stamp in wireless sensor network.

In this thesis, I learned how to synchronize time in wireless sensor networks and the three protocols. I collected results from the experiments by using the different protocols. All the graphs and tables showed the delays, offsets and drifts. As we known, there were many methods of synchronization, like sender to receiver synchronization and receiver to receiver synchronization. By comparing the differences of the results, we got the advantages of each protocol. Then for different situations, we can choose a different protocol.

REFERENCES

[1] Wikipedia, January 2011 [online]
http://en.wikipedia.org/wiki/Network_Time_Protocol
[2] Wikipedia, January 2011 [online]
http://en.wikipedia.org/wiki/Precision_Time_Protocol
[3] Wikipedia, January 2011 [online]
http://en.wikipedia.org/wiki/Precision_Time_Protocol
[4] Wikipedia, January 2011 [online]
http://en.wikipedia.org/wiki/Wireless_sensor_network
[5] ed. D.J. Cook and S.K. Das, John Wiley, January 2011 [online, PDF], *Wireless Sensor Networks*
http://arri.uta.edu/acs/networks/WirelessSensorNetChap04.pdf
[6] Wikipedia, January 2011 [online]
http://en.wikipedia.org/wiki/WLAN
[7]Wikipedia, January 2011 [online]
http://en.wikipedia.org/wiki/WLAN
[8] Wikipedia, January 2011 [online]
http://en.wikipedia.org/wiki/Time_synchronization
[9] Michael Roche, January 2011 [online], *Time Synchronization in Wireless Networks*
http://www1.cse.wustl.edu/~jain/cse574-06/ftp/time_sync/index.html#Section2.0
[10] Jeremy Elson, Lewis Girod and Deborah Estrin, January 2011 [online, PDF], *Fine-Grained Network Time Synchronization using Reference Broadcasts*
http://lecs.cs.ucla.edu/Publications/papers/broadcast-osdi.pdf
[11] Michael Roche, January 2011[online], *Time Synchronization in Wireless Networks*
http://www1.cse.wustl.edu/~jain/cse574-06/ftp/time_sync/index.html#Section2.0
[12] Jeremy Elson, Lewis Girod and Deborah Estrin, January 2011 [online, PDF], *Fine-Grained Network Time Synchronization using Reference Broadcasts*
http://lecs.cs.ucla.edu/Publications/papers/broadcast-osdi.pdf
[13] Michael Roche, January 2011[online], *Time Synchronization in Wireless Networks*
http://www1.cse.wustl.edu/~jain/cse574-06/ftp/time_sync/index.html#Section3.0
[14]Michael Roche, January 2011[online], *Time Synchronization in Wireless Networks*
http://www1.cse.wustl.edu/~jain/cse574-06/ftp/time_sync/index.html#Section4.0
[15] Miklós Maróti, Branislav Kusy, Gyula Simon, Ákos Lédeczi, January 2011 [online, PDF], *The Flooding Time Synchronization Protocol*
[16] Michael Roche, January 2011[online], *Time Synchronization in Wireless Networks*
http://www1.cse.wustl.edu/~jain/cse574-06/ftp/time_sync/index.html#Section4.0
[17]IXXAT, January 2011 [online], *IEEE1588 PTP Introduction*
http://www.ixxat.com/introduction_ieee_1588_en.html?gclid=COGRwoy94qcCFQY03wodn19n9w
[18] Hui Dai, Richard Han, May 2011, [online,PDF], *TSync : A Lightweight Bidirectional Time Synchronization Service for Wireless Sensor Networks*
[19] Savonia tests, May 2011, *IEEE1588 PTP test reports*
[20] Yu Peng, Qinghua Luo, Zhaoqing Liu, May 2011 [online], *Electronic Measurement &*

Instruments
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5274295

[21] Savonia tests, May 2011, *IEEE1588 PTP test reports*

[22] Jeremy Eric Elson, May 2011 [online, PDF], *Time Synchronization in Wireless Sensor Networks*
http://citeseerx.ist.psu.edu

[23] Miklós Maróti, Branislav Kusy, Gyula Simon, Ákos Lédeczi, January 2011 [online, PDF], *The Flooding Time Synchronization Protocol*
http://citeseerx.ist.psu.edu

[24] Miklós Maróti, Branislav Kusy, Gyula Simon, Ákos Lédeczi, January 2011 [online, PDF], *The Flooding Time Synchronization Protocol*
http://citeseerx.ist.psu.edu

[25] Miklos Maroti, Branislav Kusy, Gyula Simon and Akos Ledeczi [online, PDF], *The Flooding Time Synchronization Protocol*
http://citeseerx.ist.psu.edu

[26] Roxana Albu, Yann Labit, Thierry Gayraud, Pascal Berthou, May 2011 [online, PDF], *An Energy-efficient Clock Synchronization Protocol for Wireless Sensor Networks*
http://hal.archives-ouvertes.fr/docs/00/54/49/72/PDF/An_Energy-efficient_Clock_Synchronization_Protocol_for_Wireless_Sensor_Networks_2_.pdf