August 2014
Master's Degree Thesis

# Improving IEEE 1588v2 Time Synchronization Performance with Phase Locked Loop

Graduate School of Chosun University

Department of Computer Engineering

Rico Hartono Jahja

# Improving IEEE 1588v2 Time Synchronization Performance with Phase Locked Loop

PLL 을 이용한 IEEE1588v2 의 시간 동기 성능 개선

August 23, 2014

Graduate School of Chosun University

Department of Computer Engineering

Rico Hartono Jahja

# Improving IEEE 1588v2 Time Synchronization Performance with Phase Locked Loop

Advisor: Prof. Seokjoo Shin, PhD

A thesis submitted in partial fulfillment of the requirements for a Master's degree

April 2014

Graduate School of Chosun University

Department of Computer Engineering

Rico Hartono Jahja

# 자흐자 리코하토노의
# 석사학위논문을 인준함

| | | |
|---|---|---|
| 위원장 | 조선대학교 교수 | 모상만 (인) |
| 위 원 | 조선대학교 교수 | 정일용 (인) |
| 위 원 | 조선대학교 교수 | 신석주 (인) |

2014 년 5 월

조선대학교 대학원

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# ABSTRACT

## Improving IEEE 1588v2 Time Synchronization Performance with Phase Locked Loop

Rico Hartono Jahja
Advisor: Prof. Seokjoo Shin, Ph.D.
Department of Computer Engineering
Graduate School of Chosun University

Time synchronization plays a very important role in communication. Therefore, there are several approaches that have been proposed in order to reduce the PDV (Packet Delay Variation). PDV consists of several possible sources such as queuing delay, processing delay, and transmission delay. However, the last two sources can be ignored since the only contribution to the offset accuracy is just a few microseconds. On the contrary, the queuing delay turns out to become a big problem for the offset accuracy, due to the reason of its unpredictability and the possibility of incurring a very big value. Moreover, there is also another factor that should be taken into consideration, which is the difference of the clock frequency in every device.

In this thesis, we are going to analyze the IEEE 1588 and the PLL. Then, the next step is to propose a method that is capable to mitigate the errors that arise from the queuing delay and the clock drift. In conclusion, our simulation results show that the proposed method can actually reduce the PDV to the network requirement. However, this type of method wills only works for several types of clock. Therefore, the usage of the PLL for all type of clock will still remains a challenge.

# 한글요약

## PLL을 이용한 IEEE1588v2의 시간 동기 성능 개선

자흐자 리코하토노
지도 교수: 신석주
컴퓨터공학과
대학원, 조선대학교

시간 동기화는 통신에서 매우 중요한 문제이다. 몇몇 방법들이 PDV(Packet Delay Variation)을 줄이기 위해 제안되었으며, 이는 PDV가 큐 지연(Queuing Delay), 처리 지연(Processing Delay), 전송 지연(Transmission Delay) 등으로 구성되어 있다는 것을 이미 많은 사람이 알고 있음을 의미한다. PDV에서 처리 지연과 전송 지연은 수 마이크로초의 낮은 값을 갖고 있기 때문에 무시할 수 있지만, 큐 지연은 값을 예측할 수 없고 지연 시간도 매우 길기 때문에 큰 문제가 된다. 게다가 고려해야 할 또 다른 문제는 모든 디바이스의 시간 주파수(Clock Frequency)가 각각 다르다는 것이다.

이 연구에서, 우리는 IEEE1588v2와 PLL(Phase Locked Loop)를 중점적으로 분석했으며, 큐 지연의 오차와 Clock Drift를 완화시키는 방법을 제안한다. 제안한 방법이 네트워크 요구사항 내에서 PDV를 줄일 수 있다는 것은 시뮬레이션 결과에서 확인할 수 있다. 그러나, 제안된 방법은 단지 몇몇 디바이스의 Clock에서만 실행될 수 있다. 모든 디바이스 Clock에서 작동할 수 있는 방법은 연구 중에 있다.

# I. INTRODUCTION

## A. Synchronization Algorithm

In the world of telecommunication, there are several parameters that can be classified as important issues. For instance, one of the most important issues in telecommunication applications that include mobile communications, such as CDMA, GSM, and UMTS[1][2], is the time synchronization. A proper synchronization is very important in order to fulfill the requirements for both the time and the frequency synchronization and also in order to ensure that the communication is well performed. Moreover, due to the advance improvement in the communication applications that requires better synchronization technique, a proper synchronization will be able to better support the performance of the applications.

There are several important terms in the context of the synchronization scope, synchronization, and syntonization. The Synchronization means that there are two or more signals that run in the same frequency and without any phase difference. Meanwhile, syntonization indicates signals that run in the same frequency, but in a different phase with each other.

Nowadays, there are two types of synchronization technique. The first one is over the air synchronization technique, and the last one is over the packet synchronization technique. The first technique requires some help from the satellite; hence a GPS assisted node is required. With this kind of node, the synchronization happened through the process of sending the information to the satellite and by adjusting the device clock with the received information from the satellite. This type of synchronization will produce high accuracy synchronization. However, it

will incur a high cost and is not suitable for indoor environment, due to the limitation of the GPS signal that cannot go through the indoor environment.

On the other hand, the second technique synchronizes each device through exchanging timestamp messages between them. There are several examples of the over the packet technique. Precise Time Protocol (PTP)[3][4] and Network Time Protocol[5] are just a couple of examples. Nevertheless, in this second technique there are several obstacles that might reduce the accuracy of the synchronization process. The obstacles come from the characteristic of its node and the network environment. Recently, the issue of the time synchronization has also been mainly taken care of through the utilization of the PTP. Moreover, the PTP has also been published into the IEEE standard. IEEE 1588 is the name of the standards that managed the synchronization algorithm for wired network.

### B. Introduction to IEEE 1588

IEEE 1588 is a standard protocol that works on the application layer of OSI system. The main purpose of IEEE 1588 or PTP is to fulfill the communication requirement in term of the time synchronization between the devices. As mentioned before, PTP is one of the over the packet synchronization technique, therefore PTP has its own special message transmission that will be deployed when the synchronization process is going to be started.

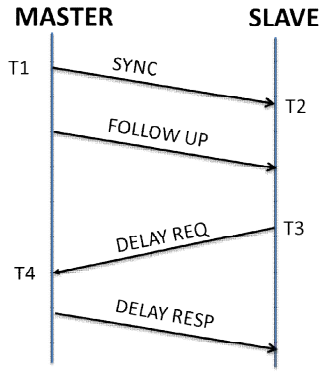


**Figure I-1 IEEE 1588 message transmission**

The main idea of this protocol is to apply the master slave message transmission, where the slave device adjusts their clock to be synchronized with the master device. Typical PTP message transmission can be seen in Figure 1-1. In that figure, it is shown that the PTP message transmission consists of 4 main messages, which are SYNC, FOLLOW UP, DELAY_REQUEST, and DELAY_RESPONSE messages. Each message will pair up with their respective message, for example SYNC with FOLLOW UP and DELAY_REQUEST with DELAY_RESPONSE, in order to get the information about the transmission delay between the master and the slave device and vice versa.

Due to the synchronization requirements, IEEE 1588 Precise Time Protocol (PTP) emerges as one of the solutions for the time synchronization techniques. At the moment, IEEE 1588 had already come with two versions. The first version that was released in 2002 came as the advance improvement from the previous technique, which was NTP. The first version was also managed to achieve microsecond accuracy in term of the clock offset between the master and the slave clock. Additionally, the second version, which is called the IEEE1588 PTPv2, was introduced in 2008. One of the main significant differences between the first and

the second version is the existence of the Transparent Clock (TC). With this TC, the accuracy between the devices can be improved up to a sub-microseconds.

## C. Problem Statement

| Network | Frequency Accuracy | Phase Accuracy |
|---|---|---|
| CDMA 2000 | 50 ppb | Range 3 μs to 10 μs |
| GSM | 50 ppb | |
| WCDMA | 50 ppb | |
| TD-SCDMA | 50 ppb | 3 μs |
| LTE | 50 ppb | 1 μs |
| WiMAX | 50 ppb | 1 – 1.5 μs |

**Table I-1. Synchronization requirements**

Speaking of synchronization accuracy, table I-1 shows the requirement of the network time synchronization. It is clear enough to state that the accuracy of the synchronization depends on the network technology. Usually, a higher network technology requires more accurate synchronization technique. Therefore, in accordance with this table, the goal of this thesis should be the microsecond phase accuracy.

Previously, it has been claimed that the IEEE 1588 is one of the possible solution that able to synchronize one device to another in the wired network. Although theoretically the IEEE 1588 version two can achieve sub-microseconds phase accuracy, in practice it requires a lot of effort to achieve that level of accuracy. There are several factors that can cause the synchronization accuracy to become higher than one microsecond. For instance, the unpredicted queuing delay in the network and the different internal clock frequency in every node play a role in determining the accuracy of the synchronization. Moreover, the asymmetric link between the devices also contributes some major parts in determining the

synchronization accuracy. This unpredicted factors has caused the instability of the phase different between the devices. Therefore, from this point on, this term is going to be called as the Packet Delivery Variation (PDV). A good synchronization algorithm should have a very little PDV. The reason is because a small PDV can ensure that the synchronization is robust enough to acquire the desired phase of accuracy between the devices.

### D. Research Contribution

Our main objective in this thesis is to get a proper time synchronization algorithm that fulfills the network synchronization requirement, which is one microsecond. In order to meet this goal, we need to test the performance of the IEEE 1588 first. Eventually, the simulation results show that the IEEE 1588v1 is not able to meet the requirement of one microsecond phase accuracy, and so does the IEEE 1588v2, despite the phase different in IEEE 1588v2 that has been decreasing tremendously compare with the previous version. Consequently, it can be assumed that there is something else that is needed to be conduct with this issue. Another method is required in this wired network in order to achieve one microsecond phase accuracy. Therefore, another method that has already been used in the NTP, which is called Phase Locked Loop (PLL), is added later on. A modification of this PLL is proposed in order to get a stable and matched phase difference between master and slave device.

### E. Thesis Layout

The other four chapters in this thesis will be organized as follows. In chapter II, more detail explanation about the IEEE 1588 and PLL will be presented and will be follows by the related works. Our modified algorithm will be briefly explained in chapter III. Furthermore, in chapter IV, the performance of our modified

algorithm is going to be presented, and will be analyzed in detail. Lastly, the conclusion of this thesis will be given in chapter V.

# II. IEEE 1588 And Phase Locked Loop

## A. IEEE 1588 Mechanism

IEEE 1588 Precise Time Protocol (PTP) is a protocol that has the ability to synchronize between the master and the slave clock. The synchronization here refers to both the time and the frequency synchronization. Accordingly, IEEE 1588 PTP protocol is also consists of several messages exchanges as shown in figure 1.

Figure 1 completely describes how the PTP works. The synchronization process is started with the transmission of SYNC message at T1 from the Master, while the Slave receives the time stamp information at T2. Then a FOLLOW UP message is transmitted by the master, which contains the actual time stamp of the SYNC message in the physical layer that is mentioned as T1. After receiving the FOLLOW UP message, the slave will send a DELAY REQ message at T3 to the master, and the master will reply with a DELAY RESP message containing the time stamp of the arrival time of DELAY REQ message (T4) at the master. Finally, the offset time between the master and the slave clock can be figured out as an equation, $Offset = \frac{(T2-T1)-(T4-T3)}{2}$[6]. However, the equation is created with the assumption that the link delay between the master and the slave clock is the same in both the direction of the links. This assumption is the key point of the implementation of the IEEE 1588 PTP so far. On the other hand, in many cases in the real world, it has been very difficult to reach an equal link delay between the two devices in both of the transmission links.

As mentioned before, this protocol came with two versions, which is version one and two. In the version one, the concept of Boundary Clock (BC)[5] is introduced, meanwhile, the second version introduced another type of clock, which is

Transparent Clock. Due to that reason, a proper explanation about these two types of clock will be described in the next part.

## 1. Boundary Clock

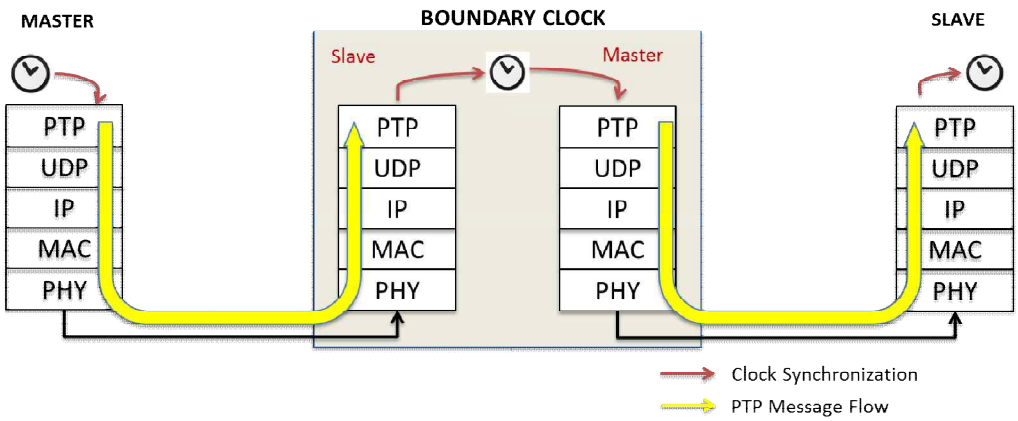


**Figure II-1. Boundary clock mechanism**

The figure below explains about the principal concept of the boundary clock. In short, this boundary clock wills allows the slave clock to be synchronized with the master clock through the synchronization process that happened in every hop between the master and the slave. When the master clock send the PTP message to the network, the intermediate node will perform a synchronization process at the time it receives the message. And the synchronized message will be passed on to the next node, until it reaches the slave. In order to use the boundary clock, the entire nodes in the network must be IEEE 1588 supported. Consequently, from this point, if every node in the synchronization chain supports PTP or IEEE 1588, it will be called as an on-pass supported network. On the contrary, if the network does not support PTP, it will be called as a no-pass supported network.

## 2. Transparent Clock

Transparent Clock (TC) is first published in IEEE 1588 version two in 2008[7]. It came as the improvement from the previous version, which disclosed in 2002[8]. The second version with the Transparent Clock (TC) is claimed as a protocol that can achieve the sub-microseconds synchronization accuracy. Literally, the transparent clock will allow the PTP message to get through the entire network without having any delay in the network, since the intermediate nodes are "transparent". The reason why the intermediate nodes can be assumed as "transparent" is because every intermediate node has the ability to measure how long does the packet stay in their node.

| Criteria | PTPv1 | PTPv2 |
|---|---|---|
| Clock Types | Ordinary Clock (OC)<br>Boundary Clock (BC) | Ordinary Clock (OC)<br>Boundary Clock (BC)<br>E2E TC<br>P2P TC<br>Management Node |
| Time Representation | Epoc number (16 bit)<br>Seconds (32 bit)<br>Nanoseconds (32 bit) | Seconds (48 bit)<br>Nanoseconds (32 bit) |
| Time Interval Resolution | 1 ns | $2^{-16}$ ns (15.26 fs) |
| Message Types | Sync/Follow Up | Announce/Sync/Follow Up |
| | Delay Req/Resp | Delay Req/Resp |
| | Management | Management |
| | | Pdelay Req/Resp<br>Pdelay Resp Follow Up |
| | | Signaling |
| Message Rate | Multicast | Multicast/Unicast |
| Mappings | UDP/IPv4 over IEEE 802.3 | UDP/IPv4 over IEEE 802.3<br>UDP/IPv6 over IEEE 802.3<br>Directly over IEEE 802.3<br>PROFINET<br>DeviceNet/ControlNet |
| Extensions | None | By Type/Length/Value (TLV) |
| Redundancy | BMC<br>No | BMC, Alternate Master, and Master Cluster<br>Yes |
| Multiple Domains | By 4 multicast addresses | By domain number (8 bit) |

**Tabel II-1. Comparison between IEEE 1588v1 and IEEE 1588v2**

This table above explains regarding the improvement that is being introduced in the IEEE 1588v2. Additionally, there is also another thing that is noted as the most improving factor, which is the message type. In the second version there is another message transmission that is being offered. This message transmission is introduced as peer-to-peer message transmission. This type of message transmission is used in the peer-to-peer transparent clock (P2P TC), on the contrary, the normal transmission is used in the end-to-end transparent clock (E2E TC),

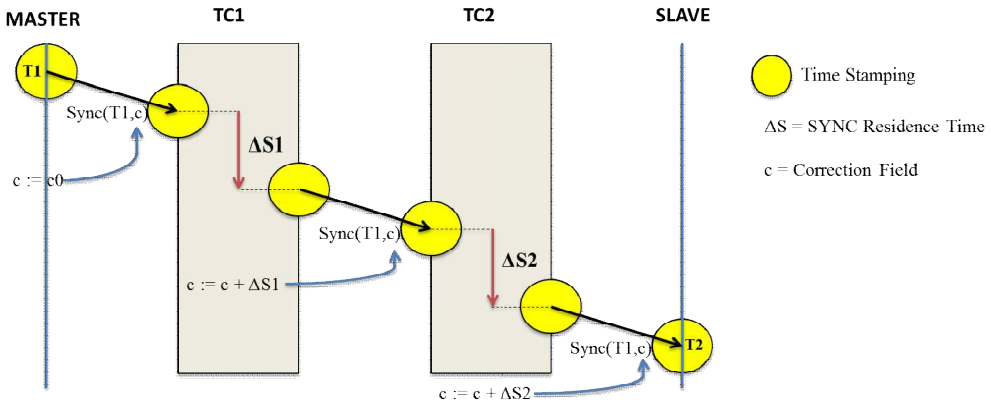


**Figure II-2. End-to-end transparent clock mechanism**

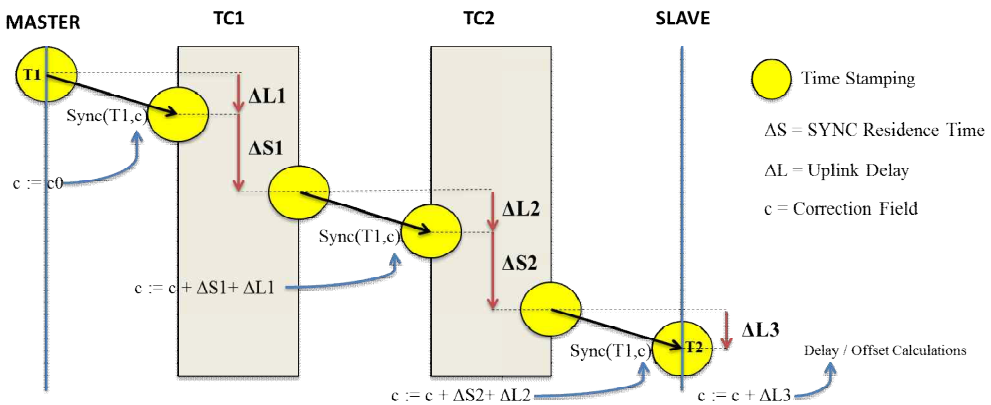


**Figure II-3. Peer-to-peer transparent clock mechanism**

These two figures above showed the difference between the end-to-end transparent clock and peer-to-peer transparent clock. In the end-to-end transparent clock, the intermediate nodes only measure the residence time or how long does the packet stay in the nodes. This residence time is registered as the correction field and this correction field is always updated whenever the message is going to leave the intermediate node. In the end, after the message has arrived in the slave, the arriving timestamp will be adjusted in accordance with the correction field.

Moreover, the peer-to-peer transparent clock also has the same purpose. However, there is a slight difference in the correction field. In the peer-to-peer transparent clock, the residence time is not the only correction value, but there is also an uplink delay value. This uplink delay value is the advantage of the peer-to-peer transparent clock. Additionally, the only way to measure the uplink delay is through the utilization of the peer-to-peer message transmission in every intermediate node. For instance, the peer delay message transmission happens in every constant period from the intermediate nodes to their neighbors.

However, this kind of mechanism can only be done if the network is full on-pass supported network. Hence, despite their high accuracy that has been proven, it is obvious that this method does required network support.

**B. Phase Locked Loop**

Phase locked loop is a control system, which have an output signal phase that is related to its input signal phase. This loop is designed to get a stable output phase. Therefore, like a typical control loop, this control loop usually consists of the proportional, integration, and derivation. The major advantage of deploying this loop is not only to synchronized the phase between the input and the output, but also the frequency between them that can also be synchronized as well.

Normally, PLL [9][10][11] is being used to discipline one source that has a variable that always changes every time. PLL will adjust that source to have a stable one, and close or exactly same as the desired value.

PLL is consisted of two big parts. The first one is the phase detector. The main function of this part is to detect the phase different between the input and the output. Furthermore, the measured error between the input and the output will become the input for the second part, which is the control loop. This control loop is normally uses the PI controller to adjust and maintain the output value. This figure below shows the basic block diagram of the PLL

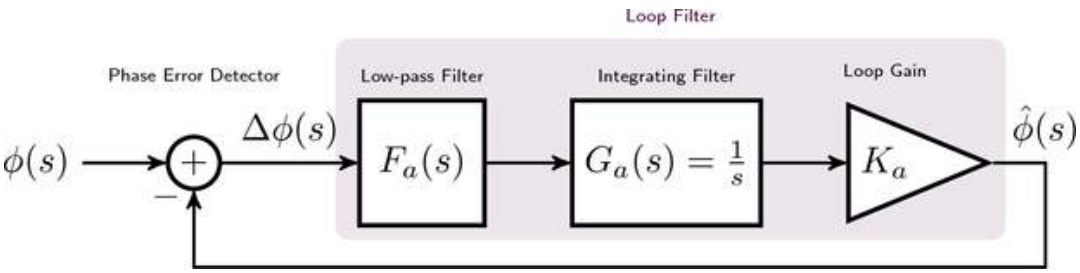


**Figure II-5. Example of PLL block diagram**

## C. Related Works

During the implementations of the IEEE 1588 PTP, there are some challenges that must be handled by the developers. However, since there are two versions of the PTP, some of the challenges faced in the first version might not appear in the second version and vice versa or might also appeared in both version. In [12], the simulation work for the IEEE 1588v1 has already developed and this simulator seemed to achieve a microsecond offset delay only in the background traffic free network. Consequently, due to the different time of the publications and taking into consideration that the second version is the improvement from the first version, some of the challenges that occurred in the first version can already be solved by

the second version. However, the second version still has some extra challenges left in the synchronization itself.

### 1. Asymmetric Link between Master and Slave

The PTPv1 always considers the assumption that the link between the master and the slave is symmetric. However, in the reality, this assumption is actually difficult to achieve because the link is highly dependent on the communication activity in the network. In [13], the authors came with some new methods to mitigate the asymmetric link problem, especially in a highly cascaded network. Furthermore, in [14], the authors proposed a novel technique called queuing estimation method to improve the synchronization accuracy. Previously, in [15], the authors mentioned the delay components in every offset measurements, which includes propagation delay, queuing delay, and quantization error that indicates the difference between the actual value and the digitized value. In [2][15], the authors clearly defined that the most influencing factor in this asymmetric link is the queuing delay in the network, which is proof to be hard to be solved with the IEEE 1588 PTPv1.

On the other hand, the IEEE 1588 PTPv2 introduces the TC technique in order to solve this problem. In TC the residence time value, which is the estimated queuing delay that happened in each intermediate node, is included in the SYNC message packet and also continuously updated every time the SYNC message comes and leaves the intermediate node. Consequently, PTPv2 can solve the link symmetry problem partially but should not be taken as the optimal solution.

Moreover, in [16] and [20], the authors proposed a similar technique in order to improve the performance of IEEE 1588v2, which states that the rate of timing message have an influence to improve the performance of clock synchronization. The difference between the two methods is that one of the methods uses a fix

number of timing message, while the other one use a flexible number of timing messages that can be controlled in accordance with the network congestion.

### 2. Mismatch Rate between Local and Master Clock

The mismatch rate problem will occurs when using TC, especially in P2P TC and E2E TC. On the other hand, the problem will not appeared in the BC in PTPv1 because of its nature. Moreover, when implementing the P2P TC, the mismatch problem becomes very crucial because the residence time estimation error happens in every intermediate node. Furthermore, the source of the mismatch error depends highly on the characteristic of the local clock in the intermediate node. In Figure 3 and 4, the comparison between E2E TC and P2P TC has been clearly described. If there is a significant mismatch clock rate in the networks, the desired offset value cannot be archived. In this regards, a method to reduce the error caused by the mismatch rate is a potential issue to be worked on. There have been several works that addressed the issue such as [17], which avoid the local clock in order to measure the residence time using its own clock rate, but let the local clock to calculate the clock rate based on the master clock. Nevertheless, it stills become one of the promising issues that can be investigated more deeply.

### 3. Multiple Layers Processing

Figure II-6 shows how the IEEE 1588 works in the sense of End-to-End protocol stacks. IEEE 1588 PTP is an application layer protocol. Therefore, since the PTP event messages need to travel from the physical layer to the application layer and vice versa, there will be time differences in the application layer and the actual leaving or coming timestamp. Consequently, there are still some issues regarding the multiple layers processing that required further investigation about the multiple

layered processes, which need further studied in order to get more precise synchronization value in each end.

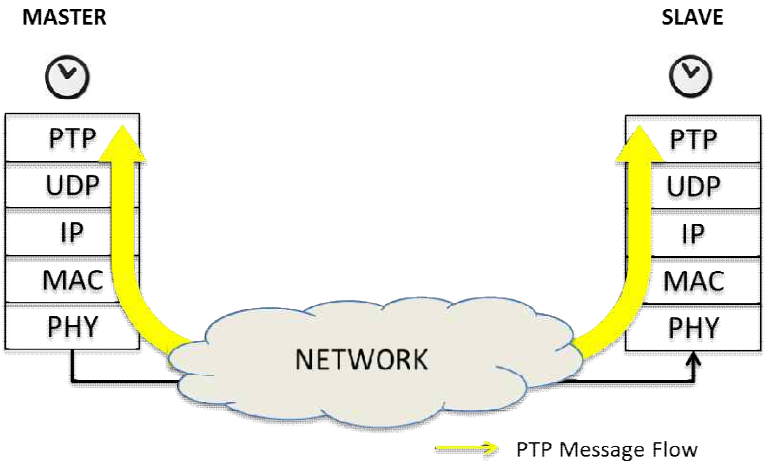


**Figure II-6. Multiple layers illustration**

## 4. Indiscipline Clock

Indiscipline clock or uncertainty in time stamping can degrades the accuracy of time synchronization. In addition some of the hardware noises in the clock servo also create another uncertainty in the synchronization protocol. In [18] and [19] in order to provide a more precise synchronization protocol, filtering method can be used as one of the possible ways through the application of Kalman filter. The characteristic of Kalman filter, which is low mean error in tracking a Gaussian distributed signal, lower computational overhead when distribution is stationary, and improved robustness in the presence of the lost signal, are the main reason behind choosing Kalman filter. Additionally, in order to get a better result, another kind of filter can also be taking into consideration, which opens another chance to exploit.

### D. OMNeT++

According to [21], OMNeT++ is an extensible, modular, component-based C++ simulation library and framework that is primarily utilized for building network simulators. Moreover, OMNeT++ also able to simulate the network in wired and wireless communication network, queuing network, and so on. Besides, OMNeT++ also offers a graphical runtime environment, which make the network interpretation easier.

# III. Modification of Phase Locked Loop With IEEE 1588v2

## A. Network Topology

In this part, the proposed idea will be explained. The objective of this proposed idea is to get a small and stable PDV between the master and the slave that fulfill the requirement of 1 microsecond phase different between the master and the slave through the combination of both techniques, which are IEEE 1588v2 and PLL.

The network topology that is going to be tested in this scheme is a single hop network between the master and the slave that has an intermediate node in between them. In this network, the data traffic is not contributed by the master or the slave device, but there is a traffic generator and a sink node that are connected in the other end of the intermediate node. This figure below will show the network topology of the scheme

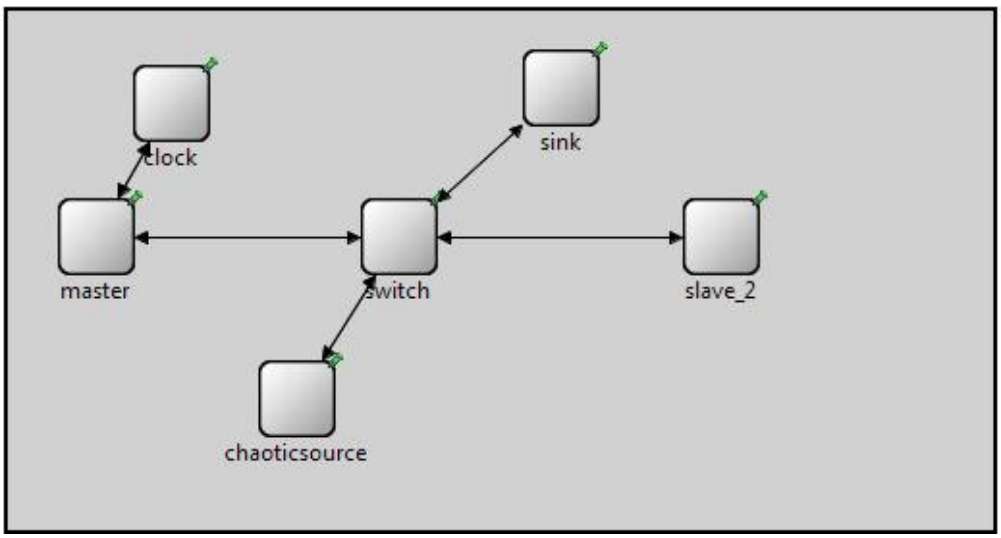


**Figure III-1. Network topology simulation**

Meanwhile, the slave and the switch node is a compound node, which indicates that the nodes can be broken into these two figures below.

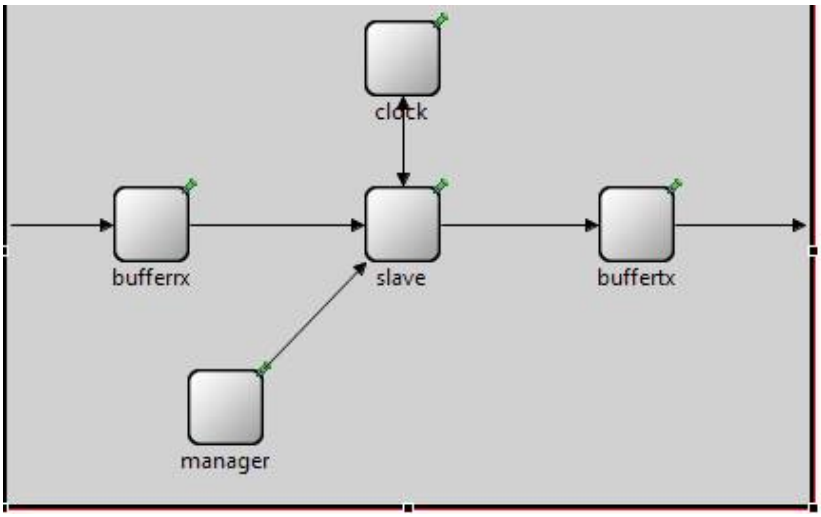


**Figure III-2. Slave node organization**

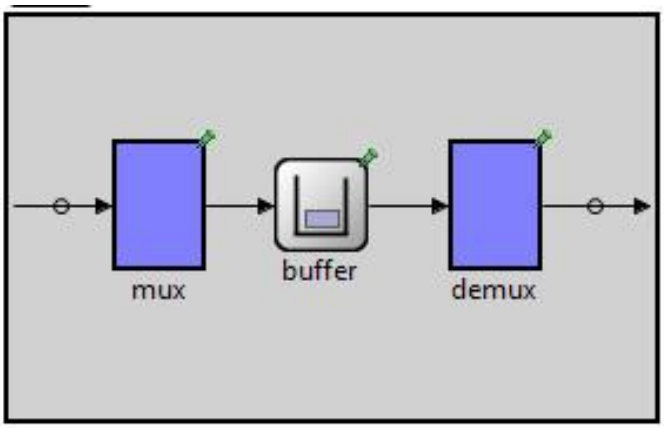


**Figure III-3. Switch node organization**

## B. PLL Modification

From figure III-1 and III-2, it is clear that both the master the slave device have their own clock, and these are the two clocks that are going to be synchronized. In

[11], a similar work has already been done. The main difference between this work with the other one is in the PLL design. They used frequency as the input and the output of the PLL. However in this proposed method, the input and the output of the PLL will be changed. It will use frequency of both the clock as the input, and the output is the direct offset or the phase difference between the master and the slave.

In this work, we also did some modifications in the time stamping event of IEEE 1588v2. We differentiate the timestamp into two kinds. The first one is called the approximation timestamp, which is symbolized as Td and Ta. This timestamp is always being generated without taking into consideration the processing time in every layer. Furthermore, there is another type of timestamp, which is T1, T2, T3, and T4. This timestamps are called the actual timestamp due to the reason that this timestamp is considering the processing time from application layer to the physical layer and the fluctuation of the frequency clock.

From the figure above, it is clear enough that T1 is possessed by SYNC departure timestamp, and the arrival timestamp is in T2. Meanwhile T3 and T4 are respectively possessed by DELAY_REQUEST departure timestamp and arrival timestamp. In this case, even though T1 and Td stand for the same event, it does not guarantee that both of them have the same value. However, some observations indicate that the actual and the approximation timestamp always have a different value.

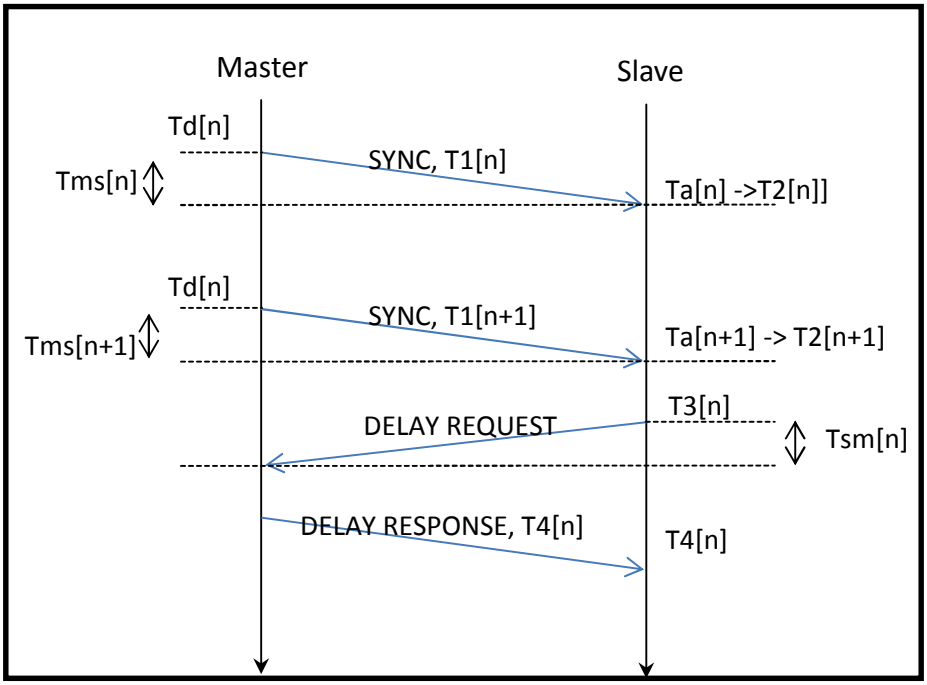


**Figure III-4. PTP message transmission**

In order to generate the actual timestamp from the clock frequency, we need several equations. First, if $f_m$ stand for master frequency and $f_s$ stand for slave frequency and both of them have their normal frequency they will be $f_{m0}$ and $f_{s0}$.

$$T_1[n] = T_1[0] + \frac{1}{f_{m0}} \sum_{i=1}^{n} \overline{f_m[i]}\, \Delta T_d[i] \qquad (1)$$

And

$$T_2[n] = T_2[0] + \frac{1}{f_{s0}} \sum_{i=1}^{n} \overline{f_s[i]}\, (\Delta T_d[i] + \Delta T_{m\,s}[i]) \qquad (2)$$

The initial timestamp for $T_1[0]$ and $T_2[0]$ are assumed to be the same as the $T_a[0]$ and $T_d[0]$. Moreover, $\overline{fm}$ and $\overline{fs}$ stand for the mean frequency between the $i^{th}$ packet and $(i-1)^{th}$ packet. Additionally, the inter-departure time of Td between the $i^{th}$ and $(i-1)^{th}$ packet is symbolizes as $\Delta Td[i]$ and the delay transmission from master to slave is $\Delta Tms$. From both of this equations, it can be concluded that if the clock is perfectly synchronized and there is no queuing delay, the time difference between T1[n] and T2[n] should be zero. Nevertheless, this kind of situation can be labeled as a very ideal situation that is hardly to achieve. Therefore, the PLL is needed in this step. The time difference will become the input to the PLL. Meanwhile, the block diagram of the complete PLL can be seen in figure III-5.

On the contrary, the loop filter block diagram can be seen in the figure (SEE TOP). From that figure, we can obtain the analog transfer function of the loop filter block, which is:

$$\hat{\phi}(s) = \Delta\phi(s)F_a(s)G_a(s)K_a \qquad (4)$$

Taking into consideration that $\Delta\phi = \hat{\phi} - \phi$, the transfer function Ha(s) for the loop will be:

$$H_a(s) \triangleq \frac{\hat{\phi}(s)}{\phi(s)} = \frac{F_a(s)G_a(s)K_a}{1 + F_a(s)G_a(s)K_a} \qquad (5)$$

Since the loop filter model is consisted of a low pass filter, then we should consider that the standard form of the first order low pass filter is through converting the denominator of Ha(s) into $s^2 + 2\zeta\omega_n + \omega_n^2$. Furthermore, if we assumed that the Fa(s) form is$\frac{1+\tau_2 s}{\tau_1 s}$, we can get,

- $H_a(s) = \frac{K_a/\tau_1(1+s\tau_2)}{s^2+s\frac{K_a\tau_2}{\tau_1}+\frac{K_a}{\tau_1}}$ (6) , where $\tau_1 = \frac{K_a}{\omega_n^2}, \tau_2 = 2\zeta/\omega_n$

Moreover, another important thing to remember is that we are going to work in digital domain. Therefore, it is essential to convert the equation above into the digital domain since it is still in the analog domain. In order to convert it into digital domain, we need this equation, $s = \frac{1}{2}\frac{1-z^{-1}}{1+z^{-1}}$, hence the digital transfer function will be

$$H_d(z) = \frac{4K_a}{\tau 1}\left[\frac{(1+\tau 2/_2) + 2z^{-1} + (1+\tau 2/_2)z^{-2}}{1-2z^{-1}+z^{-2}}\right] \qquad (7)$$

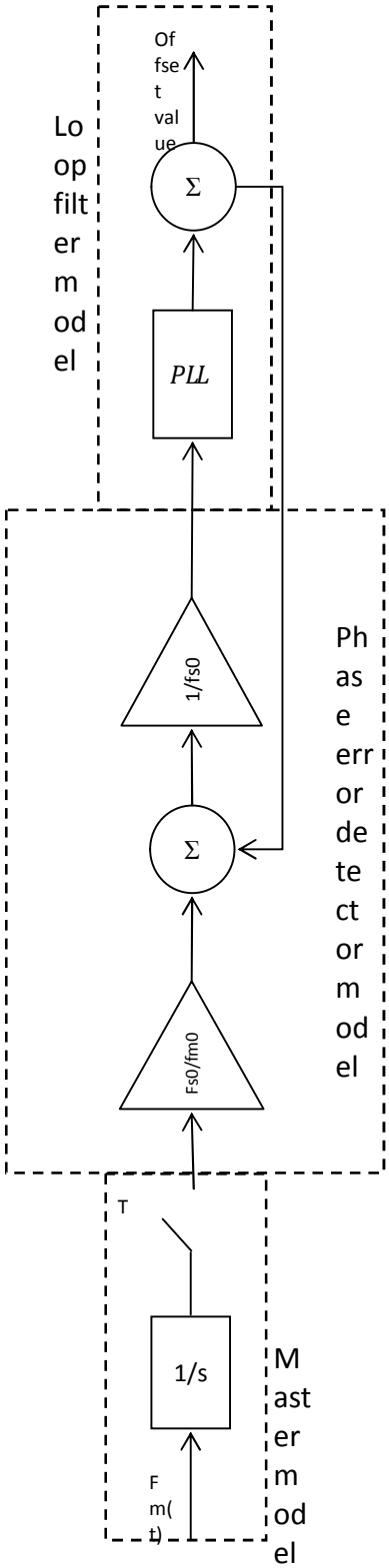


**Figure III-5. Complete block diagram**

### C. Combination of IEEE 1588v2 and PLL

There are a lot of things that need to be taken into consideration in combining the IEEE 1588v2 and the PLL. For examples are the when the PLL is being called upon, in what kind of circumstances does the PLL will work, the offset calculation between the master and the slave, and how we decide the value for each parameter in both the PLL and the IEEE 1588v2.

From the figure III-4, we can see that after the slave receives the SYNC message, we are able to get the information about the actual timestamp of ingress and egress in the slave and the master. This is the information that is going to be used as the input for the PLL block. However, this raises another question, which is how about the other message, such as DELAY_REQUEST message. Consequently, in this case, we will not use the PLL for the reversed transmission, because we assumed that the master would always get the right clock. Therefore, eliminating the task of correcting the master clock. Hence after the PLL process is finished, the arrival timestamp for SYNC message (T2) will be adjusted with the output value of the PLL. Furthermore, through the utilization of the IEEE 1588v2 concept, the transmission delay from the master to the slave is achieved.

If we refer to the IEEE 1588v2 standard, the DELAY REQUEST message is not a reply for the SYNC message, instead, they has their own period in generating the message. Nevertheless, the DELAY RESPONSE is a reply for the DELAY REQUEST message. These two messages are used to measure the delay from the slave to the master. Hence, the offset calculation can be obtained when the slave has already received the DELAY RESPONSE message. In this work, we randomly distribute the DELAY REQUEST period in between the SYNC periods and up to 14 times of the SYNC period, in order to ensure the correct offset calculation because the delay measurement has already been calculated. Furthermore, for the

first time calculation, the initial offset value between the master and the slave is considered as one of the input for PLL block, or in other word, only happened in the 1st packet of SYNC. However, if the PLL cannot approximate the offset value into zero, the rest of the packet wills only taking into consideration the offset of the clock that is resulted from the PLL.

Figure III-5 shows the complete block diagram of the proposed PLL. In this diagram, at the left side of the PLL block, there is one block that contains the period of the slave clock. This block is intended for normalizing the offset, in order to ensure that it have a proper unit (normalization process).

# IV. Performance Evaluation

## A. Step Response of The Control Loop Model

Section III-B has already explained about the modification of the PLL in order to ensure the consistency between the output and our goals. However, there is also another point that is needed to be consider, which is the value of the loop parameters that consists of the proportional gain, natural frequency, and damping factor. Furthermore, through the utilization of the matlab simulator, it is easy to check the performance of the closed loop. Hence, the step response of the Hd(z) is shown in figure X.

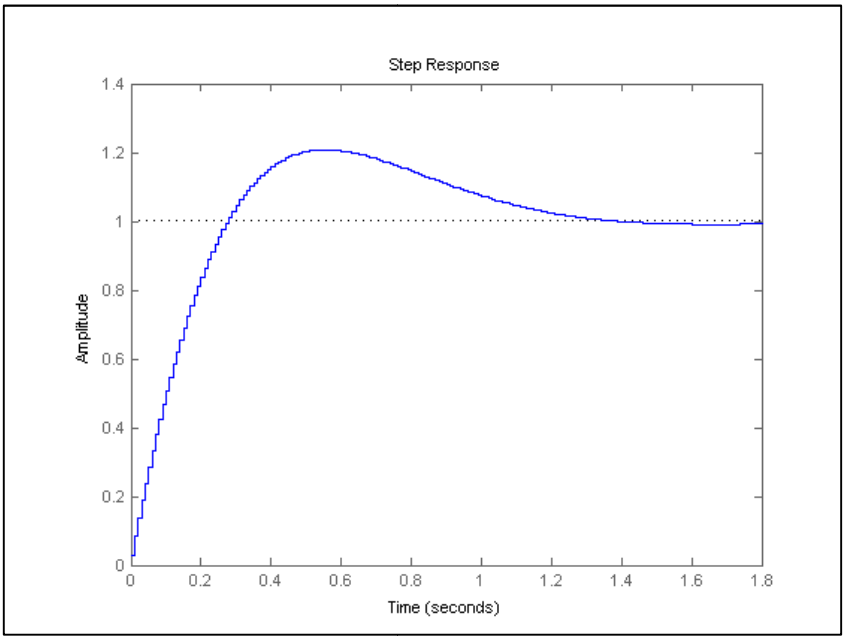


**Figure IV-1. Step response of control loop**

From figure IV-1, it is clear that the step response of the closed loop has already met our expectation. First, we want to get the control loop that achieves a stable

output in less than 2 seconds. Since the SYNC period is going to be set at 2 seconds constant, it is quite logical that we want the system to generate a proper output when the other SYNC message arrives. Therefore, in this scheme, we set the value of the proportional gain to 1000, damping factor to 32, and natural frequency (ωn) as 2 x 10-3.

### B. Offset measurement in Different Circumstances

| Parameter | Parameter values |
|---|---|
| **Master Clock** | |
| **SYNC Period** | 2 second |
| **Slave Clock** | |
| **Initial offset** | 5 microseconds |
| **Initial drift** | 10 ppm |
| **Traffic Generator** | |
| **Period** | 2 milliseconds |
| **PLL Parameters** | |
| **Damping factor (ζ)** | 32 |
| **Natural frequency ($\omega_n$)** | 2x10-3 |
| **Proportional gain (Ka)** | 1000 |

**Table IV-1. Default values simulation parameters**

There is only one way to clarify our method, which is through the observation of the offset between the master and the slave in the network. In our simulation design, we design one network with one master and one slave, with an intermediate node between them as seen in figure III-1. At first we are going to observe the offset measurement that include the background traffic in order to generate the traffic packet with the period 2 x 10-3 seconds. The result can be seen in figure Y.

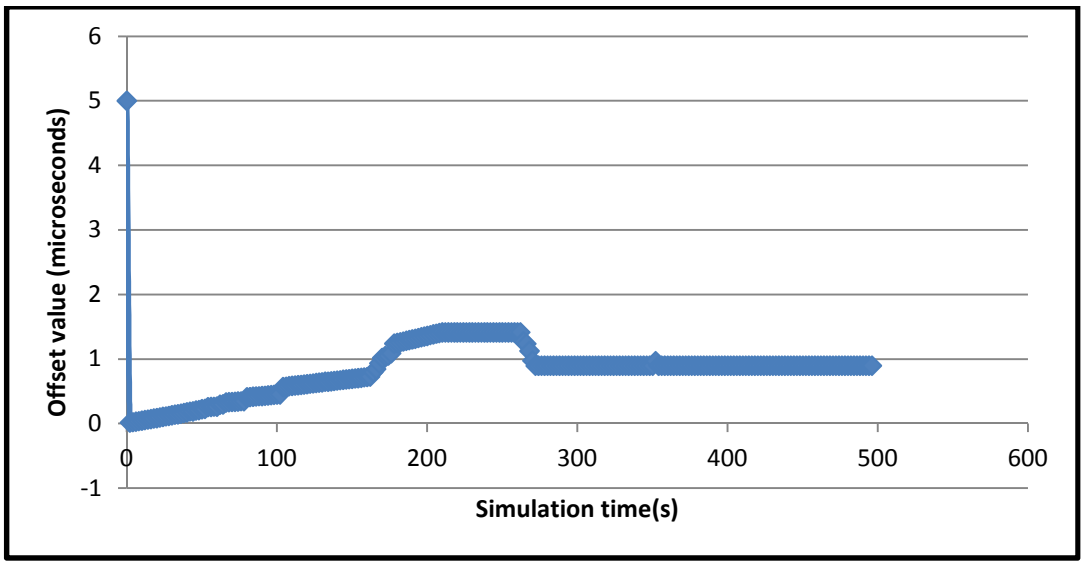


**Figure IV-2. Simulation result with default value**

Figure IV-2 shows the performance of our proposed method, which uses the parameter from table IV-1 that is set to be the default condition for our method. Figure IV-2 also indicates that our method is able to achieve less than one microsecond offset accuracy. However, there is some drawback, which is a long acquisition time that is required in order to achieve the stable output. In this scheme, we need to wait for around 280 seconds before we were able to get the desired output, which is around 0.8 microseconds.

### 1. Effect of The Background Traffic Toward Offset Accuracy

The main purpose behind this performance measurement is to get an idea about the robustness of our method toward the different value of the queuing delay. Or in other word, we also want to verify the capability of the Transparent Clock in mitigating the effect of the unpredictable queuing delay. In this simulation, we are going to simulate the network with several value of the background traffic period.

We have prepared four conditions, which are without any background traffic, 1.2 Mbps, 6 Mbps, and 12 Mbps.

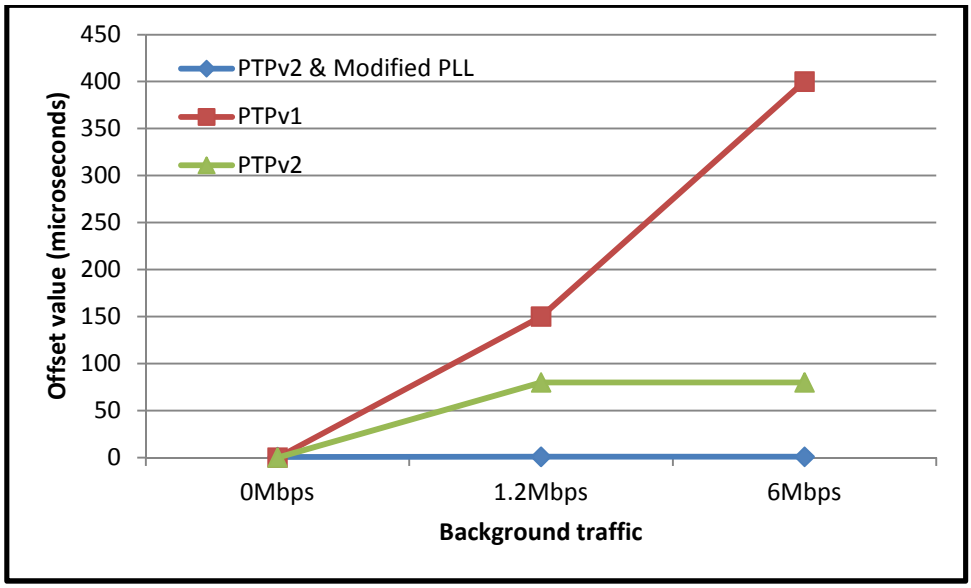


**Figure IV-3. Simulation result with default value**

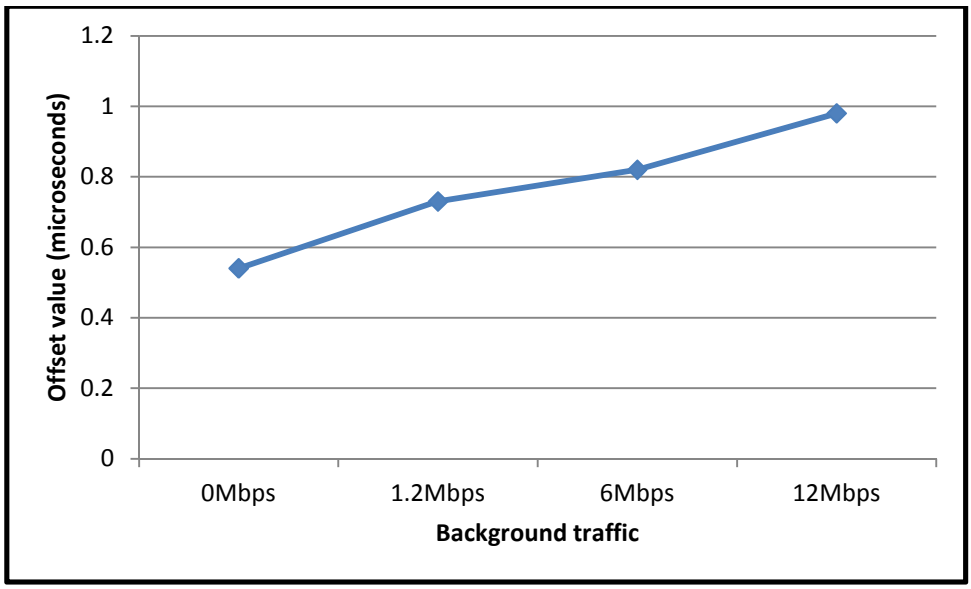


**Figure IV-4. Simulation result in different background traffic**

As we can see in figure IV-4, our result is quite robust in a 12 Mbps network. Since our results always need some period of acquisition time, we choose the offset value from the maximum value of the stable output. Hopefully, this value will be able to serve as the worst-case assumption in this scheme. In figure IV-3, we can see that the offset value is always increasing somehow. However, when the background traffic is up to 12 mbps, the offset measurement shows the offset accuracy that is less than one microsecond. Nevertheless, according to figure IV-3, the offset measurement that we achieve is far better than the standard one. Therefore, we are safe to assume that in our simulation, the transparent clock works perfectly in mitigating the queuing delay that is going to reduce the accuracy of the synchronization process.

**2. Effect of Different Quality of Clock Toward Offset Accuracy**

Our goal in this work is to achieve a better PDV measurement in clock synchronization process by taking into consideration all the factors, such as unpredictable queuing delay and unmatched clock frequency in every device. Therefore, the PLL is being utilized in order to mitigate the delay that happened because of the unmatched clock frequency in every device. In a perfect world, PLL will be able to synchronize both the offset and the frequency into the same frequency and generates zero offsets between them. However, in the real world it proofs to be difficult to achieve. One of the reasons is due to the imperfection of the filter design. Moreover, in this design, we converted the analog filter to the digital filter that caused even more imperfection for the filter.

In our simulation, we also investigated how important is the effect of the clock quality toward the synchronization accuracy. In this scheme, we chose several numbers of the clock drift (10 – 100 ppm), and run the simulation in every 10 value of the drift for a hundred time iteration. The result can be seen in the figure IV-5.

This value of this graph is also taken from the maximum stable offset value, in order to give some worst-case point of view.

It can be clearly noticed that this method is far from perfection. This method does not work well for every clock quality. This method will only compensated the clock with less than 40 ppm drift value. Therefore, this result concludes that in the previous observation, the remaining offset value between the master and the slave is caused by the imperfection of the PLL in compensating the mismatch clock frequency. However, this method can take into account the clock drift calculation in determining the offset value rather than the standard one, which is shown in figure IV-6. Both the standards give the same output in every value of the clock drift, despite that the proposed method offset difference can also barely be seen in this figure.

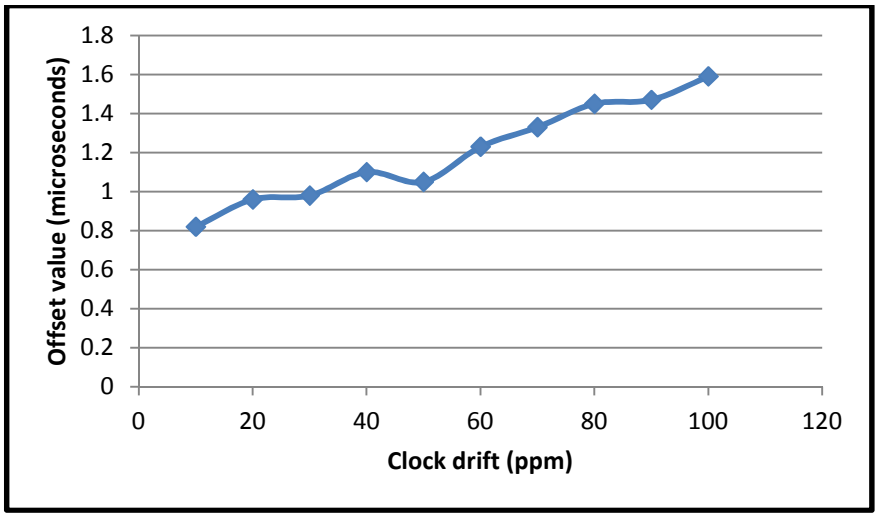


**Figure IV-5. Simulation result in different clock drifts value**

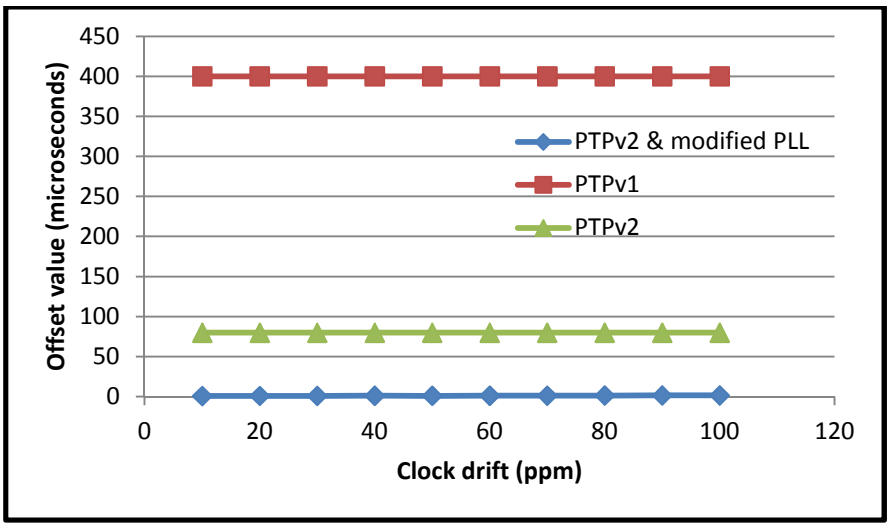


**Figure IV-6. Performance comparison with existing method**

## C. Multiple Intermediate Nodes Network Topology Performance Evaluation

Since the communication can be performed everywhere, it is possible to locate both the master and the slave clock in a very far distance. Therefore, the intermediate nodes between the master and the slave could also be more than one node. Figure IV-7 shows the network topology that uses two identic intermediate nodes. Furthermore, in order to make the simulation more reasonable, cross background traffic is applied. In this case, the chaoticsource 1 node will generate traffic packet to sink 1 node, while the chaoticsource node will sends packet to sink node. Through the utilization of the same network traffic, some comparison between one intermediate node and two intermediate nodes can be performed.

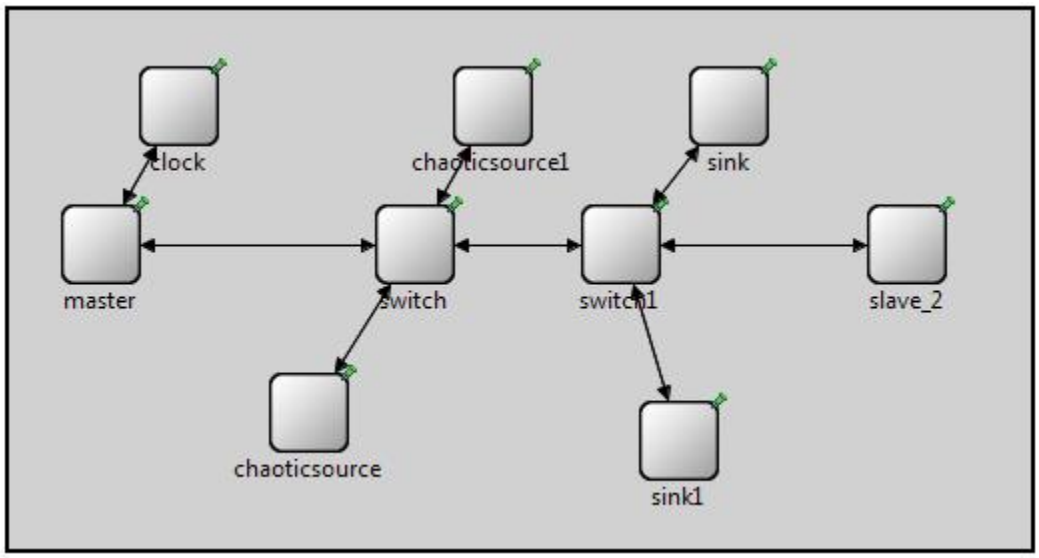


**Figure IV-7. Multiple intermediate nodes network topology**

## 1. Offset Measurement in Different Traffic Network

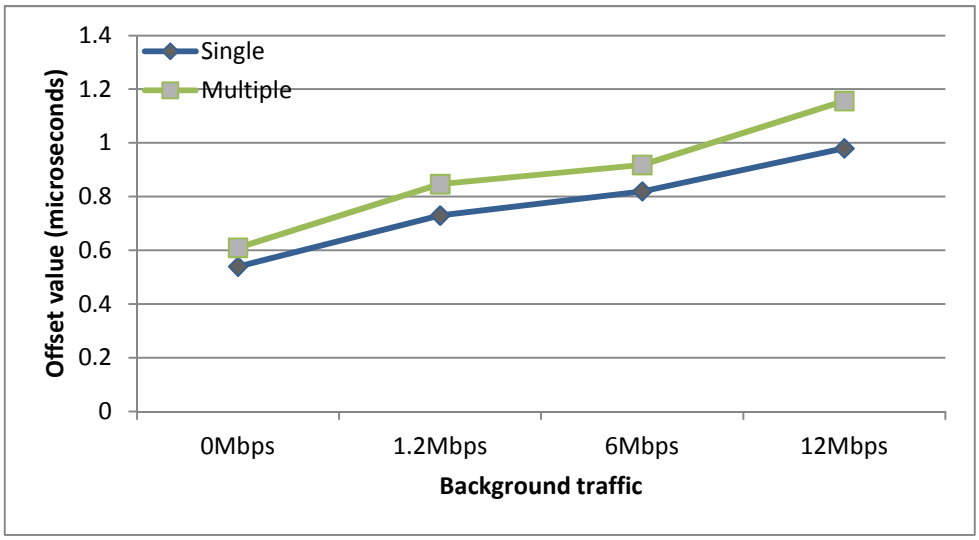


**Figure IV-8. Offset measurement in multiple intermediate nodes**

Figure IV-8 shows the simulation comparison between the single intermediate node and the multiple intermediate nodes. Despite the increasing background traffic, the

offset measurement in the multiple intermediate nodes between the master and the slave still shows an acceptable value, which is below one microsecond. However, there is some probability, when the background traffic has exceeds 12Mbps, the offset measurement of the master and the slave can be more than one microsecond.

## 2. Offset Measurement in Different Quality of Clocks

Since the internal clock in every device will be different, a robust synchronization algorithm must consider the effect of the different clock drift. Figure IV-9 shows that the modified method?? still needs some more improvement in order to achieve desired offset measurement. The reason is because in a bigger clock drift value the results in the multiple intermediate nodes are worse than in the single intermediate node. If another improvement of the PLL is impossible to achieve a better value, the number of the intermediate node needs to be considered in the implementation process due to the reason that fewer intermediate nodes provide a better synchronization process.

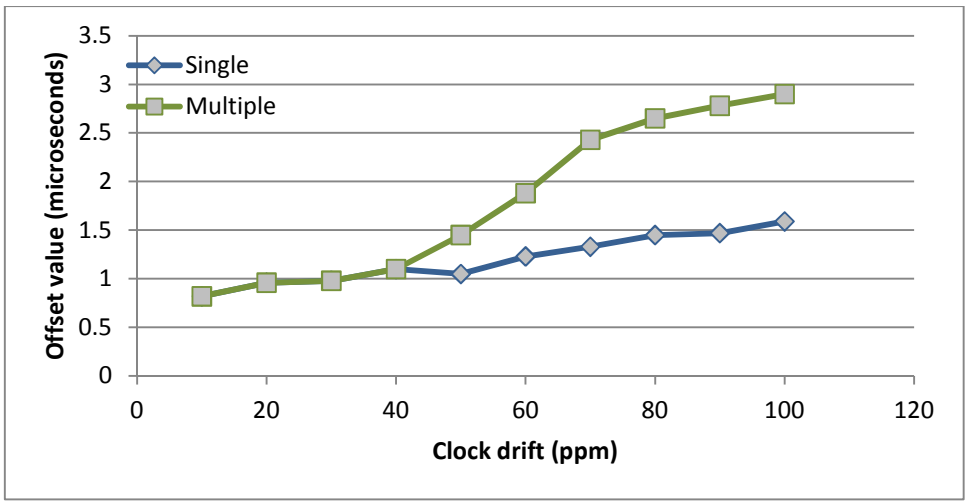


**Figure IV-9. Clock drift effect in multiple intermediate nodes network**

### D. Comparison between Proposed Method versus Existing One

This method is proposed in order to get a better approach in mitigating the PDV compare to the existing one. Table IV-1 is shows some comparison point between our method and IEEE 1588v1 and version 2.

From table IV-1, it is clear enough that our method is capable to mitigate the offset value up to 0.8 microseconds. This offset or phase different has already meets the requirement for the LTE network. However, there is a major drawback in this scheme due to the reason that this scheme requires a lot of time for the acquisition time. In this simulation, we record 280 seconds for the acquisition time. It is almost more than half of the simulation time. On the contrary, the positive effect of this method is that this method considers the effect of the clock drift and able to achieve a proper synchronization process, despite applied only to the good quality clock.

| Criteria | IEEE 1588v1 | IEEE 1588v2 | Our Method |
|---|---|---|---|
| Max Offset Value | 1.5 seconds | 80 milliseconds | 0.8 microseconds |
| Acquisition time | Almost zero | Almost zero | 280 seconds |
| Queuing delay effect | Very big | Very small | Very small |
| Clock frequency effect on offset | Not measurable | Not measurable | < 1 microsecond for clock drift < 40ppm |

**Table IV-2. Comparison of our method and existing methods**

# V. Conclusions

Since the major improvement of the mobile network to the LTE network, another issue is rising up. The issue will be a new clock synchronization algorithm that is required in order to fulfill the requirement of 1 microsecond offset accuracy for the LTE network. In this thesis, we propose an approach to reduce the PDV through the utilization of IEEE 1588v2 and PLL. Both of IEEE 1588v2 and PLL has already widely used as the possible solution for the synchronization algorithm, but the imperfection of the PLL still open some possibilities for improvement. The simulation result of our method shows that our method meets the requirement in regards to the offset accuracy of the LTE networks. Furthermore, our method also considers the effect of the different clock frequency in every clock devices. In short, we conclude that, although we may be able to meet the LTE network requirement, another improvement for the PLL is still required.

In order for this method to be able to be widely used in every kind of clock, a better filter is required or through the utilization of other approaches, which will be more suitable if we can find another conversion method to convert the analog to the digital instead using a bipolar z domain. Moreover, using another simulator is suggested because of the limitation in time segmentation in OMNet++.

# BIBLIOGRAPHY


[1] P. Donegan. "IEEE 1588 Timing for Mobile Backhaul: The Road to Plug & Play," *Heavy Reading*, June 2011.

[2] J. Han, and D.-K. Jeong. "Practical Considerations in the Design and Implementation of Time Synchronization Systems Using IEEE 1588," *IEEE Communications Magazine*, November 2009.

[3] IEEE Std 1588-2002: "IEEE Standard for A Precision Clock Synchronization Protocol for Networked Measurement and Control Systems".

[4] IEEE Std 1588-2008: "IEEE Standard for A Precision Clock Synchronization Protocol for Networked Measurement and Control Systems".

[5] H. Weibel. "Technology Update on IEEE 1588: The Second Edition of the High Precision Clock Synchronization Protocol," *Zurich University of Applied Sciences*, 2009.

[6] L. Shuai, L. Yueming, and J. Yufeng. "An Enhanced IEEE 1588 Time Synchronization for Asymmetric Communication Link in Packet Transport Network," *IEEE Communications Letters*, Vol. 14, No. 8, August 2010.

[7] S. Lee, S. Lee, and C. Hong. "An Accuracy Enhanced IEEE 1588 Synchronization Protocol for Dynamically Changing and Asymmetric Wireless Link," *IEEE Communications Letters*, Vol. 16, No. 2, February 2012.

[8] J. Jasperneite, K. Shehab, and K. Weber. "Enhancements to the Time Synchronization Standard IEEE 1588 for a System of Cascaded Bridges," *IEEE International Workshop on Factory Communication Systems*, 2004.

[9] I. Hadzic, D. R. Morgan, and Z. Sayeed, "A Synchronization Algorithm for Packet MANs", *IEEE Transaction on Communication*, Vol. 59, No. 4, April, 2011.

[10] W. C. Lindsey, and C. M. Chie, “A Survey of Digital Phase-Locked Loop”, *Proceedings of The IEEE*, Vol. 69, No. 4, April 1981.

[11] G. Zhou, S. Shetty, G. Simms, and M. Song, “PLL Based Time Synchronization in Wireless Sensor Network”, *15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, 2009

[12] G. Giorgi and C. Narduzzi. “Modeling and Simulation Analysis of PTP Clock Servo,” *International IEEE Symposium on Precision Clock Synchronization (ISPCS),* October 2007.

[13] D. Mohl, and M. Renz. “Improved Synchronization Behavior in Highly Cascaded Networks,” *International IEEE Symposium on Precision Clock Synchronization (ICPCS)*, October 2007.

[14] T. Murakami, and Y. Horiuchi. “Improvement of Synchronization Accuracy in IEEE 1588 Using a Queuing Estimation Method,” *International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, October 2009.

[15] J. Han, and D.-K. Jeong. “A Practical Implementation of IEEE 1588-2008 Transparent Clock for Distributed Measurement and Control Systems,” *IEEE Transactions on Instrumentation and Measurement*, Vol. 59, No. 2, February 2010.

[16] D.T. Bui, A. Dupas, M. Le Pallec. “Packet Delay Variation Management for a better IEEE 1588v2 Performance,” *International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, October 2009.

[17] Z. Du, Y. Lu, and Y. Ji. “An Enhanced End-to-End Transparent Clock Mechanism with a Fixed Delay Ratio,” *IEEE Communications Letters*, Vol. 15, No. 8, August 2011.

[18] H. Abubakari and S. Sastry. “IEEE 1588 Style Synchronization over Wireless Link,” *International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, September 2008.

[19] G. Giorgi and C. Narduzzi. “Performance Analysis of Kalman-Filter-Based Clock Synchronization in IEEE 1588 Networks,” *IEEE Transactions on Instrumentation and Measurement*, Vol. 60, No. 8, August 2011.

[20] S. Lee. “An Enhanced IEEE 1588 Time Synchronization Algorithm for Asymmetric Communication Link using Block Burst Transmission,” *IEEE Communications Letters*, Vol. 12, No. 9, September 2008.

[21] "OMNeT++ Network Simulation Framework." . N.p., 2001. Web. 10 Apr 2014. <*http://www.omnetpp.org/*>.