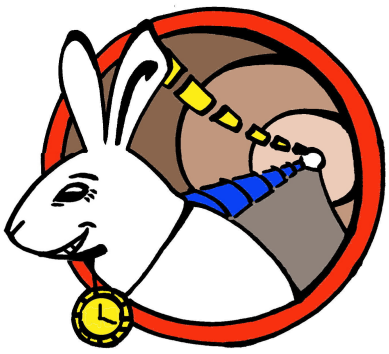
European Organization for Nuclear Research
Organisation Europeenne pour la Recherche Nucleaire



White Rabbit
Synchronization and Timing
Over Ethernet

Geneve, May 8, 2009

Contents

List of Figures

List of Tables

Chapter 1

Revision History

Version	Date	Changes	Author
0.1	02-Mar-2009	First draft for review	P. Moreira
0.2	17-Apr-2009		P. Moreira

This document is an unapproved draft. As such, this document is subject to change. USE AT YOUR OWN RISK!

Chapter 2

Terms, definitions and notation

2.1 Terms and definitions

Clock An equipment that provides a timing signal. Generally means, when used for synchornization networks, the generator of the frequencies which will be used to synchronize the network.

Accuracy: The mean of the time or frequency error between the clock under test and a perfect reference clock, over an ensemble of measurements.

Precision: a measure of the deviation of the time or frequency error between the clock under test and a perfect reference clock from the mean.

Stability: a measure of how the mean of the time or frequency error between the clock under test and a perfect reference clock varies with respect to variables such as time, temperature, etc.

Jitter: is an unwanted variation of one or more characteristics of a periodic signal in electronics and telecommunications

Wander: are long-term random variations of the significant instants of a digital signal from their ideal positions

Latency: is a time delay between the moment something is initiated, and the moment one of its effects begins or becomes detectable.

Frame: is a digital data transmission unit on the Layer 2 of the OSI model. It is used for data exchange between two or more nodes of a network.

2.2 Abbreviations and acronyms

WRP White Rabbit Protocol

WRCMP White Rabbit Control Message Protocol

CRC Cyclic Redundancy Check

STM System Timing Master

LT Luby Transform

MAC Media Access Control

PLL Phase-Lock Loop

PRC Primary Reference Clock

PTPv2 IEEE 1588 Precision Time Protocol Version 2

SyncE Synchronous Ethernet

TAI Temps Atomique International (International Atomic Time)

UTC Coordinated Universal Time

WDM Wavelength-division multiplexing

Chapter 3

Introduction

3.1 Statement of the Problem

Large control systems have distributed nodes that need to be synchronized. That is, at a given instant of time all the nodes in a system must agree with a notion of the current time. When this occurs we are in the presence of a synchronized system. Synchronization is necessary in a system to establish a global ordering of events and tasks. Synchronization may be accomplished by having all nodes using the same external time source. For example, the Coordinated Universal Time (UTC) can be known via telephone, radio, GPS etc. each one giving different levels of accuracy. However, due to the localization of some nodes the receptivity from global time reference can't be possible.

Alternatively, the system may have a single central clock and communications equipment to exchange messages between the nodes and the central clock. However in the case of the central clock system fails all nodes are left without time reference.

It is often preferable to allow each node to use its own clock, and to limit the differences between them with a synchronization algorithm. The synchronization algorithm must then ensure that the difference between the clocks of any nodes is never more than a given value, this value will define the performance of the synchronization algorithm. Synchronization algorithms typically have three phases: distribution of clock information, estimation of clock delay, and adjustment of clock values.

Synchronization algorithms may be classified according to the methods used by nodes to inform one another of their current clock values. Hardware algorithms use a dedicated network to broadcast each clock signal. They use

the principle of phase-locked loops in order to achieve a tight synchronization between the clocks with almost no time overhead. This is for instance the type of algorithm being used for the General Machine Timing (GMT) at CERN.

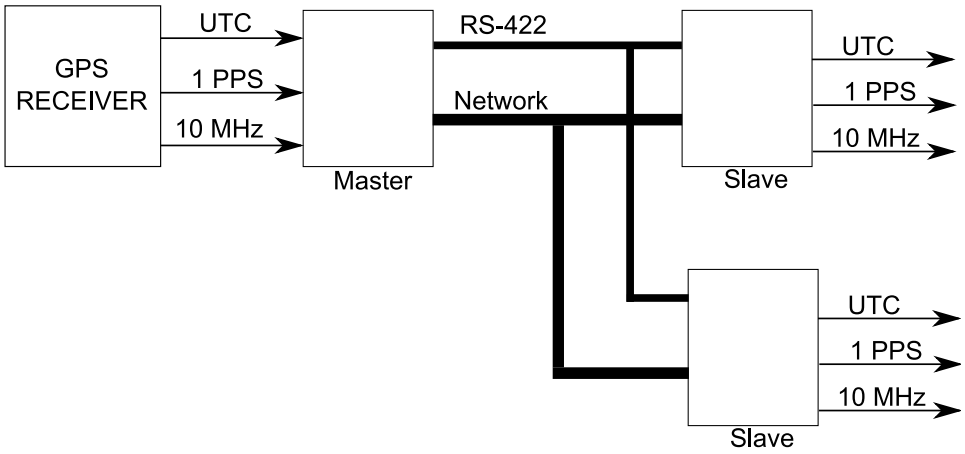


Figure 3.1: CERN's General Machine Timing System

The GMT uses a dedicated network, namely RS-422, to transmit a clock reference, send UTC timestamps and real-time control data. A GPS receiver provides two clock frequency references, videlicet a pulse per second (PPS) and a 10 MHz reference. The GPS also provides an UTC time stamp that refers to the last PPS tick. The Master Timing Station will multiply the 10 MHz reference, via a Phase-Locked Loop (PLL), to produce a 40 MHz clock, which will be used to produce a Manchester-encoded stream at 500 Kb/s. The master timing node encodes and transmits the UTC timestamps. At the receiver side, the slave PLL recovers the very stable 40 MHz reference from the 500 KB/s messages. The recovered clock shows a jitter of less than 1 ns. This 40 MHz signal is locked to UTC and can be used to generate pulses or to timestamp external events with high precision.

The slave node receives a UTC time message every second from the master. It then starts a 40 MHz count to keep an internal UTC time register with a granularity of 25 ns. This register is used for tagging all the events produced in the node.

On the other hand, network algorithms send messages across the existing communications network. These algorithms treat the clock values as

data values and exchange them periodically in order to ensure proper synchronization. Two examples of a network algorithm are the Network Time Protocol (NTP) and the Precise Time Protocol (PTP) [PTP08]. PTP timing message exchange with hardware timestaming is capable to reach network synchronization up to 10 ns.

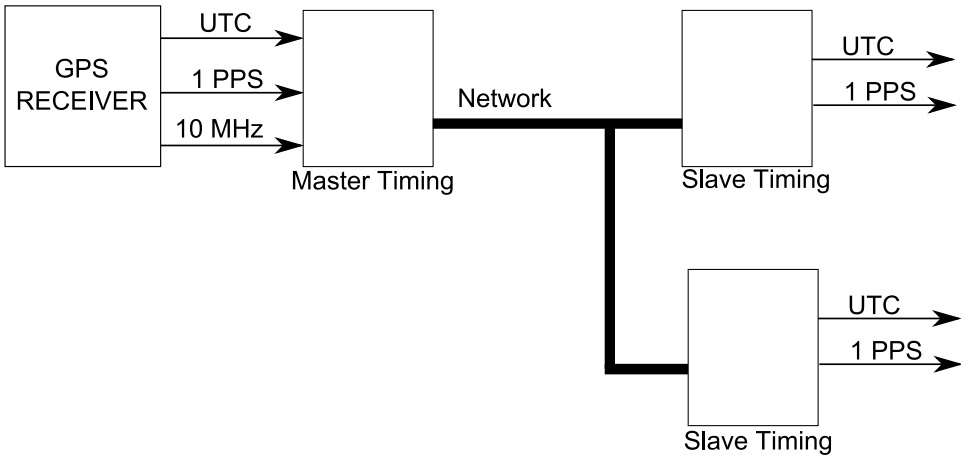


Figure 3.2: Syncronization network algorithm

Nowadays, the trend goes in the direction of using an existing general purpose network for synchronization and diagnostic data exchange in order to achieve the same or even to increase the accuracy reached by the hardware algorithms. For this purpose, the master frequency clock reference is embedded into the data link, which will be recovered by the slave with very low jitter. In addition, low level message exchange can be used to compensate long term phase deviations and for calculation of link delays.

The protocol that allows both timing and data transmission through an Ethernet network, while maintaining subnanosecond accuracy network synchronization, is White Rabbit. White Rabbit runs on general purpose high bandwidth networks as Ethernet. White Rabbit defines Ethernet as a fully deterministic switched network, capable of synchronizing nodes with very high accuracy. It brings the worlds of data and timing exchange together with minimal trade backs between them.

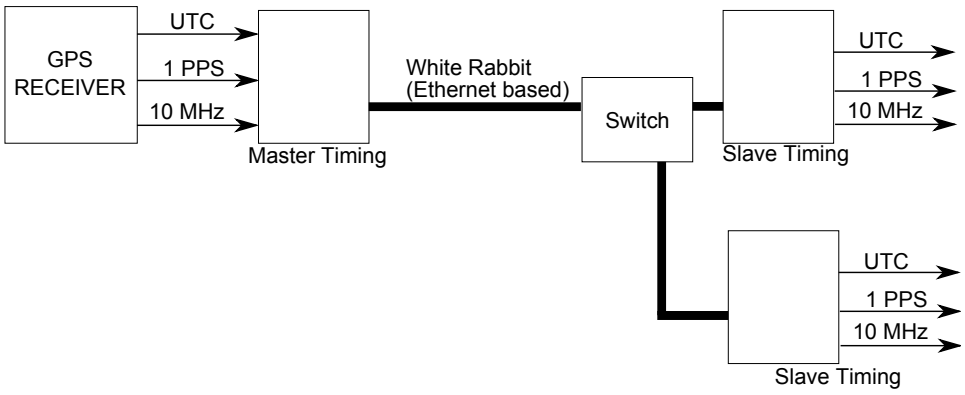


Figure 3.3: Hardware and Network Synchronization

3.2 Synchronization

Timing networks that require very accurate timing, i.e. below 1 ns, as it is the case for particle accelerators complexes, force the control networks to have parallel timing and diagnostic paths, i.e. a dedicated network for timing data and another network for diagnostics and large configuration data exchanges. Further, the timing networks present a data flow that is in most of the cases unidirectional, where timing information is transmitted downstream from the master timing system to the slaves.

Due to the unidirectional nature of the timing network, link delays can't be automatically compensated, which brings the need for costly and error-prone measuring campaigns.

Current dedicated distributed control networks, as is the case of FIP, Profibus, CAN among others lack bandwidth when compared with today's data networks. For example WorldFIP defines 2.5 Mb/s for copper wire physical layer over a distance of 500 m while the Ethernet 1000 BASE-LX physical layer standard allows 1Gb/s trough 5 km links.

Let's describe the current concept of distributed timing networks. From the point of view of frequency transmission, shown in Fig.3.4, the Master Timing Node receives a clock frequency that is traceable to a GPS clock, or another precise clock source. This clock reference will be transmitted to the Slave Timing Node, that will recover it. Nevertheless the recovered clock contains phase noise introduced by both master and slave electronics and the transmission link.

The phase delays can be compensated up to a certain degree through

out manual calibrations directly in a lab environment or in an industrial complex.

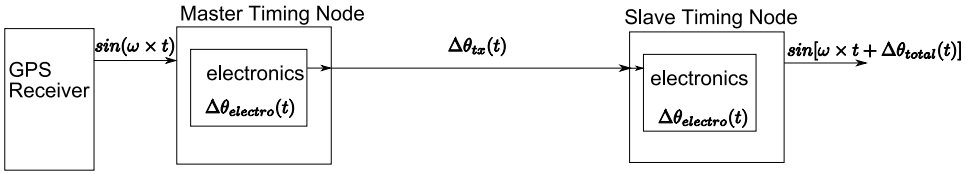


Figure 3.4: Phase Noise Sources

In addition to clock distribution, the Master Timing Node receives a UTC reference, synchronized to a PPS signal, and sends it to its Slave timing node. Obviously the slave has to compensate for all delays driven by the master/slave electronics and also due to the transmission link. This delay compensation can be done manually through expensive calibrations campaigns. The dynamic delays due to the mechanical stress, and temperature drifts are disregarded in this case.

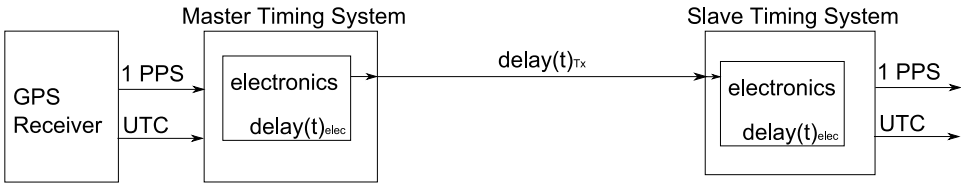


Figure 3.5: Time transmission

Both systems shown in Fig. 3.4 and in Fig. 3.5 are physically merged together, that is frequency and time reference are transmitted over the same network link. They are separated in the figures shown to get some simplicity in the description of the timing system.

More over, most of the timing systems have a separate link for timing and diagnostic/configuration data. The reason for it is the following: most if not

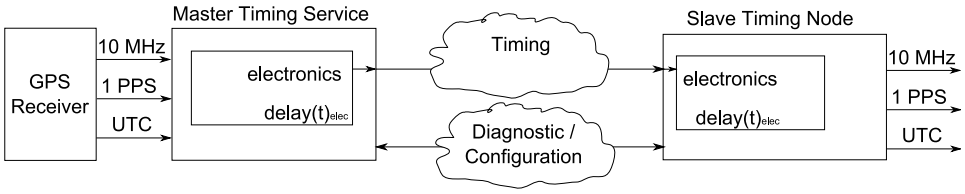


Figure 3.6: Timing and Data Network

all timing networks have a relatively low bandwidth links, as it is the case of RS-422. The exchange of large sets of data, that can be diagnostic requests or configuration data, will obviously take a big amount of time. This large time to transmit a message will obviously remove the determinism of critical messaging events, required to be transmitted within a certain time interval. In addition, in timing networks that allow high bandwidth the transmission of large amount of data will make the clock recovery on the timing to be less accurate. For systems that require accurate timing requirements this loss of accuracy is prohibitive.

The problems here enumerated are acknowledged but up to now they are being neglected, for various reasons. However new timing requirements are introduced with the upgrade of particle accelerators and in order to cope with their requirements a new distributed timing and control system needs to be researched and developed.

The new timing and control network needs to reduce the cost of maintenance and in addition it needs to implement automatic delay compensation.

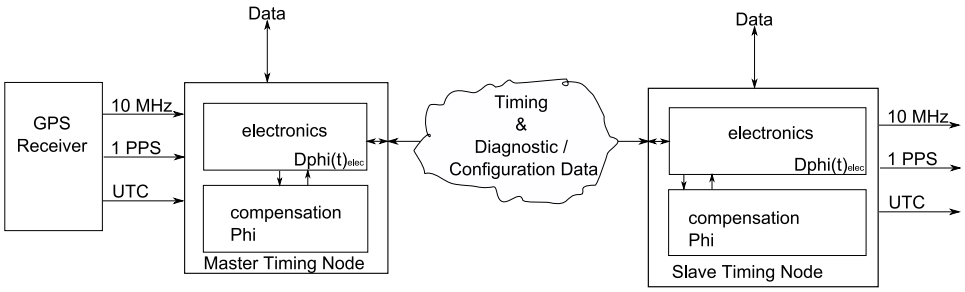


Figure 3.7: Timing and Control System

The overall timing and control system should allow bidirectional trans-

mission of data, and so that it can have feedback information about transmission delay and automatically compensate for it on the slave timing side.

In addition both diagnostic/configuration data will be transmitted using the same links as the timing data, as illustrated in Fig. 3.7. This system at its lowest level should be as abstract as possible to allow higher level solutions to perfectly work on it.

Some questions that arise are the following:

- How can we merge together both timing and data communications without sacrificing the timing accuracy, i.e. achieving sub-nanosecond accuracy?
- How can we cope with the asymmetry existent in the communication links?
- How can we achieve compensation of phases deviation?
- How can we ensure determinism in data exchange?
- How can be solve the problem of cascaded clocks, and phase noise accumulation?

This new timing and control distributed system is denominated as White Rabbit. The reason for this name can be found by looking back to our childhood and to Alice in Wonderland. From this point on we will refer to this innovative distributed timing and control system as White Rabbit Protocol, so lets follow the White Rabbit.

3.3 Technical Approach

White Rabbit Protocol (WRP) is a protocol that allows the delivery of timing and control data over a local area network, while reaching subnanosecond time accuracy.

Clearly, one may say that so this adds another protocol to the already big stack of protocols based on Ethernet that allows real-time message exchange and timing distribution, and as an example of some we can mention Ethernet PowerLink, SERCOS III and EtherCAT among others.

This document will not make a description of the existing protocols described above, or others, but by the end of the document the reader will be able to understand the main differences between existing Ethernet based protocols and White Rabbit Protocol.

To reduce its design complexity WRP is organized as a stack of layers and it is specified from the Physical Layer, Data-Link Layer and Network Layer, as described by the OSI model. The purpose of each layer is to offer certain services to the higher layers, shielding those layers from details of how the offered services are actually implemented.

Following this design methodology requires that each layer performs a specific and well defined collection of well-understood functions. During the elaboration of WRP it allowed us just focus on the real purpose of each layer in the WRP context. This document is structured to follow each abstract representation layer of the OSI Model, in a bottom-up context, to allow a sequential knowledge of how WR is structured. Following this criterion, in Chapter 4 we will introduce the use of Synchronous Ethernet in WRP networks. Chapter 5 deals with the description of the White Rabbit medium access control, which is part of the data link layer. Here we will specify how WRP frames access the transmission medium and which are its mains differences with respect to standard Ethernet traffic. In the following chapter 6 we will describe White Rabbit Control Message Protocol (WRCMP) that defines a protocol for low level network management. This message protocol belongs to the Network Layer, Layer 3 of the OSI model. The upper layers will be described in another document and they are outside the scope of this document. The linkage between the OSI Model, the WRP and also its connection to how this document is structured is illustrated in Fig.3.8.

As it is shown in Fig. 3.8, each layer has a well known set of services provided to the upper layers in the WRP context. A brief description of each layer will be provided to clear readers minds about this layering approach. From bottom-up we first come to the Physical Layer; from an abstract point of view this layer is concerned with transmitting raw bits over a communication channel. Its design issues are concerned with making sure that when one side sends a “1” bit, it is received by the other side as a “1” bit, not as a “0” bit.

The following layer is the Data Link Layer. Its main task is to transform a raw transmission facility into a line that appears free of undetected transmission errors. Broadcast networks have an additional issue in the data link layer which is how to start transmission in a shared medium. A special sublayer of the data link layer, the medium access layer is responsible to deal with this problem.

Up next, the OSI model defines the Network Layer which controls who determines how packets are routed from source client to destination client.

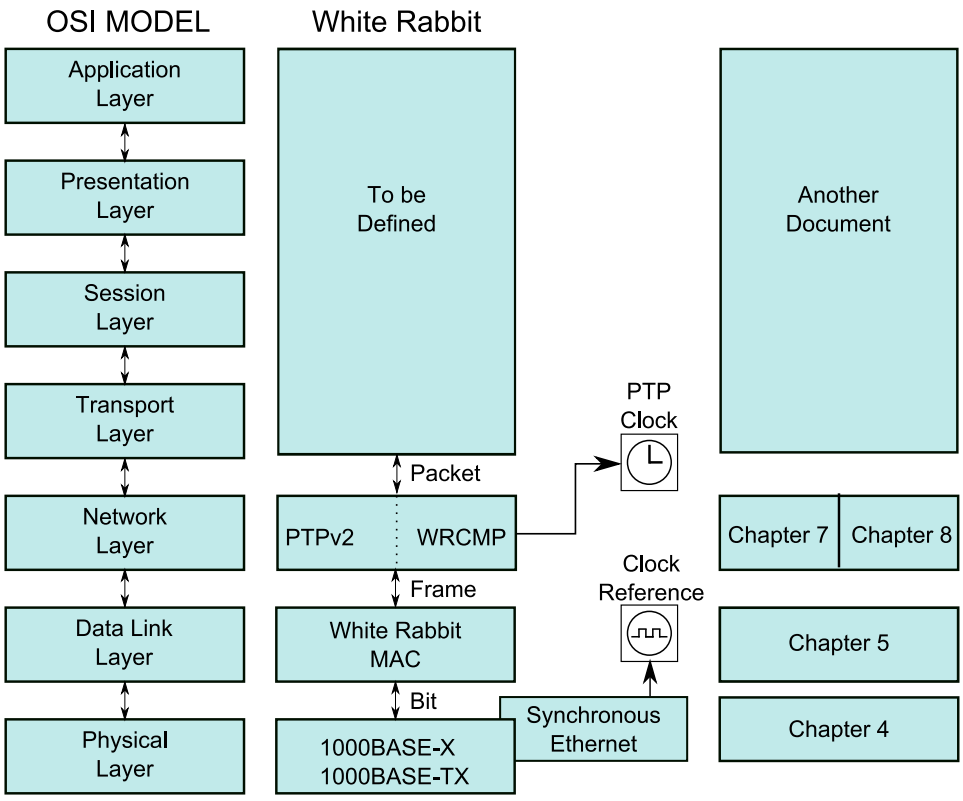


Figure 3.8: WR Layer Overview

Routes can be based on static tables that are "wired into" the network and rarely changed, or they can be highly dynamic, being determined anew for each packet, to reflect the current network load. In broadcast networks, the routing problem is simple which reflects on a thin or nonexistent network layer.

The Transport Layer defines how the data received by the upper layer will be segmented into packets and passed to the network layer. This layer level will also be responsible to set the rules for reconstructing them correctly on the destination client side. The transport layer is a true end-to-end layer, all the way from the source to the destination. In other words, a program on the source client carries on a conversation in a similar program on the destination client using the message header and control messages.

Next we have the Session Layer, it allows users on different machines to establish sessions between them. Sessions offer various services, including dialog control, token management, and synchronization.

Succeeding the Session Layer we have the Presentation Layer, but unlike the lower layers, which are mostly concerned with moving bits around, the presentation layer is concerned with the syntax and semantics of the information transmitted. In order to make it possible for computers with different data representations to communicate, the data structures to be exchanged can be defined in an abstract way along with a standard encoding to be used "on the wire". The presentation layer manages these abstracts data structures and allows higher-level data structures, to be defined and exchanged.

Finally we have in the upper most layer the Application Layer, which contains a variety of protocols that are commonly needed by users.

Although we have given a brief description of all layer presented in the OSI model, this document will only focus on the 3 lower layers defined by it. In addition, each of the described layer can be further divided into sub-layers in other to reduce the complexity of its implementation. The OSI model as described in this documents should be used as guidance for understanding at which layer level we are specifying WRP.

3.4 Scope

This document specifies the protocol and procedures used to ensure that timing and control requirements are met for time sensitive applications, such as accelerators complexes, across IEEE 802.3 full duplex networks where the transmission delay is fixed and symmetrical.

It includes the use of Synchronous Ethernet and PTP [PTP08] specifications. The synchronization to an externally provided timing signal will be also detailed. To this end it,

- Specifies a set of Ethernet Type fields which contains a prioritization scheme.
- Specifies the rules for fragmentation the non-high priority frames.
- Establishes the requirements for generation of the WR topology.
- Specifies the requirements to be satisfied by equipment claiming conformance to this specification.
- Specifies a master clock synchronization algorithm.
- Defines the rules for phase noise compensation.

3.5 Benefits

White Rabbit protocol (WRP) is a low-level timing and control transport protocol for White Rabbit network devices. WRP aims to offer the following benefits:

- Be able to synchronize up to 2000 stations with sub-nanosecond accuracy.
- Bi-directional exchange of messages between timing stations.
- Maintain compatibility between existing non-WR stations.
- Precise delay measurement and reporting, fine (sub-nanosecond scale) transparent time transmission based on PTP [PTP08] protocol and Synchronous Ethernet.
- Identification of WR-compliant devices in the network.

3.6 Network Topology

Different approaches to organize timing distribution to all the nodes of the network can be chosen, depending on the specific requeriment of the applications.

White Rabbit strategy for timing distribution is based on the distribution of the timing reference from one clock, the master clock, to all the other clocks of the network, slaves clock, directly or indirectly according to a Star or Tree topology.

In Star synchronization networks a common master synchronizes all the slaves clocks throught direct links. In Tree synchronization networks timing is distributed along routes that may incoporate several slave clocks, organized in hierarchical levels.

While the master clock should be an high-precision expensive oscillator, the slave clocks can be much cheaper. The slave clock will be locked to the reference timing signal coming to the master by means of a PLL. The loop time constant of the PLL controls its filtering properties against the phase fluctuations on the input reference and on the internal oscillator. An important step in the design of WR networks is therefore the design of the loop time constants of the slave clocks. Further details on PLL properties will be given on this document.

To resume, in normal operation all the network clocks are traceable to the master clock. Synchronization is transferred from one clock to the other according to synchronization techniques described in the following of this document.

Questions may arise on what happens should the master fail ?

(In case the master fails PTP takes care of finding the best available clock ensuring the stabilitys of the network clocks, modeling the behaviour of this dynamic and insuring the stability of the network can be quite dificulty, specially because the transient bahaviour is very hard to predict. White Rabbit design in such a way can be quite complex in term of managment and control but the results will be extremely reliable. This is called mutual synchronization.)

The master network clock, runs autonomously and most have the highest frequency accuracy 1×10^{-11} and stability; it complies with ITU-T Rec. G.811.

The switch is a slave clock which distributes its timing to all other clocks in the lower nodes of the timing hierarchy. Its main duty in then to filter effectively timing impariments on its reference and to suppy a highly stable

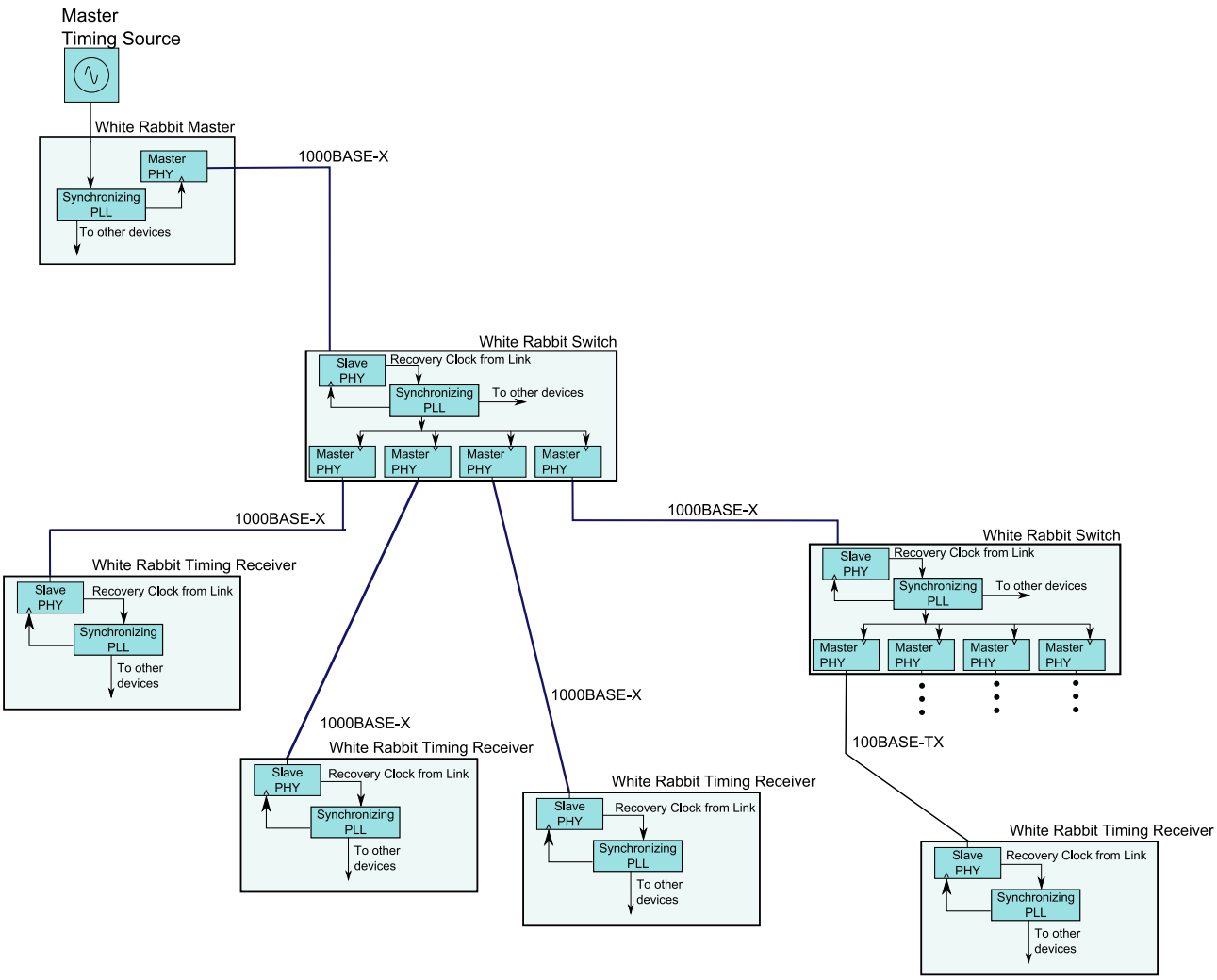


Figure 3.9: Tree synchronization Network

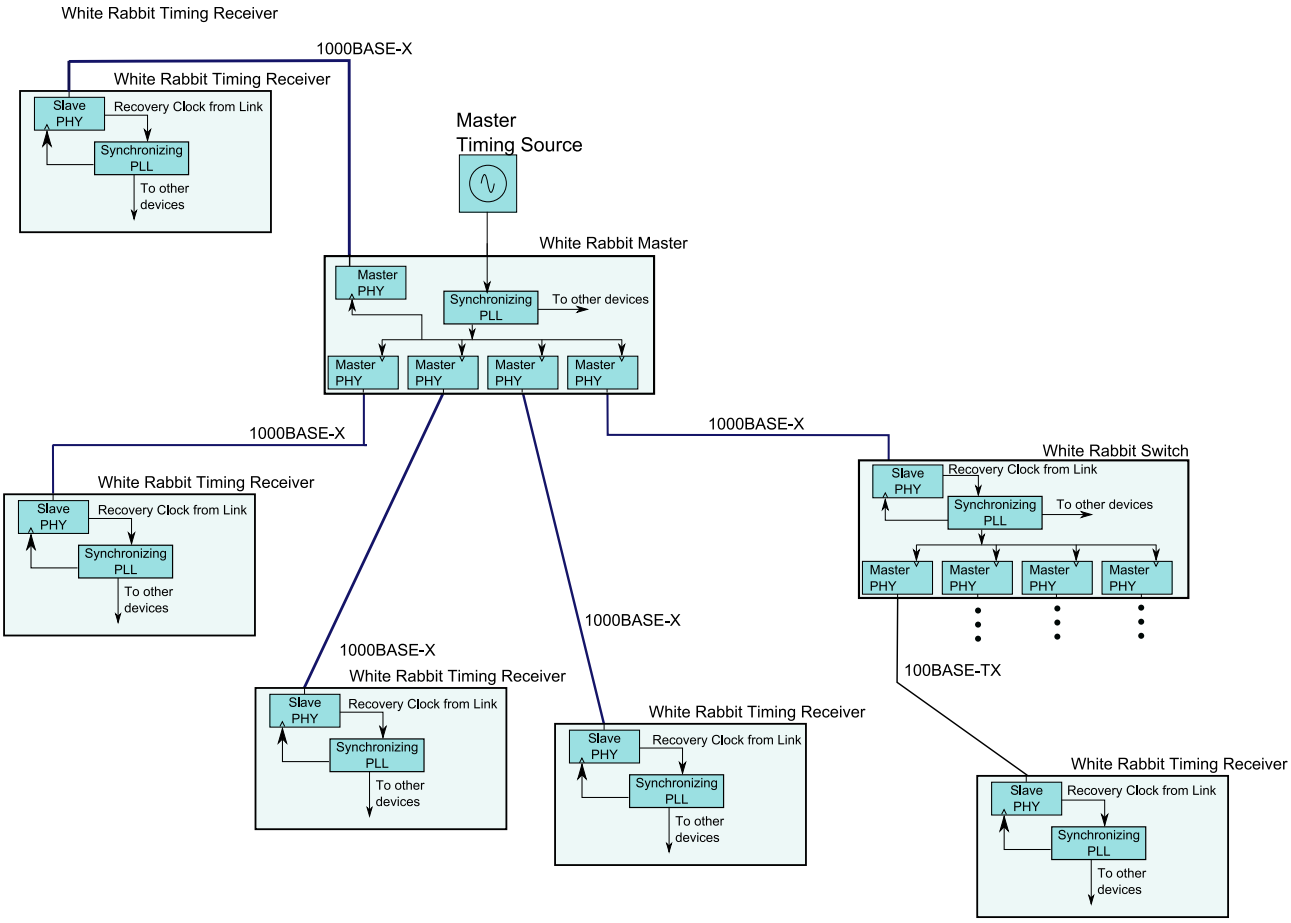


Figure 3.10: Star synchronization Network

timing should all references fail; its lock time constant is therefore very long to filter out as much phase noise as possible; it complies with ITU-T Rec. G.812. The switch should be the node with the most robust clock, the clock with the norrowest loop bandwidth, ie. with the highest input phase noise filtering capability and the best performing in hold-over among all other elements in the White Rabbit network.

The timing receiver is a slave clock with poor long term stability requirements and with short loop time constant it; complies with ITU-T Rec G.813.

For high precision timing receiver's single mode optical fiber is used with WDM so only one fiber is needed for transmission and reception, therefore ensuring better symmetry in transmission and reception delays. In the stations where high timing accuracy is not required a twisted-pair copper cable or multi mode optical fiber can be used.

Chapter 4

Synchronous Ethernet

Over the past two decades Ethernet has become the dominant technology for data transmission, in particular with telecom and wireless providers, due to its simplicity and low cost. However, the asynchronous nature of Ethernet provides certain transmission challenges.

While there are several ways to achieve synchronization over Ethernet, one gaining momentum is Synchronous Ethernet (SyncE). SyncE uses the physical layer interface to pass timing from node to node in the same way timing is passed in SONET/SDH. This gives telecom and wireless providers confidence that networks based on SyncE will be not only cost-effective, but also as highly reliable as SONET/SDH based networks.

WRP is, as mentioned, an Ethernet based protocol. It does not prescribe a particular set of rules about the physical medium. Instead WRP is designed to be independent of the transmission medium, nevertheless it recommends the use of shielded twisted pair wire for sub-microsecond accuracy or optical fiber to achieve the sub-nanosecond accuracy. Typical physical standard for gigabit Ethernet using optical fibers is 1000Base-LX and a typical one that uses twisted pair cable is 1000Base-TX. This WR specification allows the use of all physical standards defined by the IEEE802.3z [IEE04]. This specification adds in addition to the IEEE802.3z a set of rules that will allow optimal clock recovery hardware and a way to minimize low frequency phase noise.

SyncE provides a mechanism to transfer frequency over the Ethernet physical layer, which can be traceable to an external source such as a network clock. Its primary idea is that the input reference clock is not free-running with an accuracy of $\pm$ 100 ppm, but rather locked and traceable to a primary reference clock as defined in ITU G811(achieving long-term accuracy of $\pm$

10 ppt).

As such, the Ethernet link may be used and considered part of the synchronization network. SyncE gives a frequency synchronization path formed on a link by link basis. Because it works on the physical layer, SyncE has been shown to be immune to traffic load and packet delay variation.

Synchronization does exist in traditional Ethernet on each hop between two adjacent nodes, but it is not passed from hop to hop. Passing synchronization is relatively simple – take the recovered clock from the node receiving synchronization, and with this clock, feed all nodes that are transmitting synchronization. A typical SyncE line card includes the Ethernet MAC and PHY transceiver. Frequency conversion adapts back-plane frequencies to frequencies that are required as input reference clocks to the transceiver (25 or 125 MHz). The output of the timing device serves as the TX clock reference into the transceiver, replacing the free-running crystals with ± 100 ppm accuracy in the conventional line card. SyncE clocks are within +/- 4.6 ppm, which is within the frequency range of Ethernet interfaces. The reference is then used to synchronize the physical line coding (e.g., 4B/5B for fast Ethernet, 8B/10B for gigabit Ethernet) of the interface toward the slave, as shown in Fig. 4.1. At the slave node, the clock is recovered within the transceiver clock data recovery in the Ethernet PHY block of the Synchronous Ethernet line card (current operation of transceivers). From this point on, the slave station behaves like the master for the next downstream station, and frequency synchronization transported on a node-to-node basis, where each node participates in recovery and distribution. It is by this process that synchronization traceable to a primary reference source can be recovered and distributed within each station. It is also important to note that Synchronous Ethernet does not disrupt the IEEE 802.3 architecture.

The deployment of synchronization networks using the Physical Layer provides a useful tool to distribute frequency that is not subject to packet delay variation.

ITU-T Recommendation G.8261 [1], released in 2006, was the first document to introduce the Synchronous Ethernet concept as part of the studies related to the synchronization aspects in packet networks. This technology has been standardized by the ITU-T as a direct result of the experience gained with the standardization of timing distribution over SONET/SDH networks. Since the very beginning, G.8261 was recognized as the main reference for this technology, although the first version of the standard presented

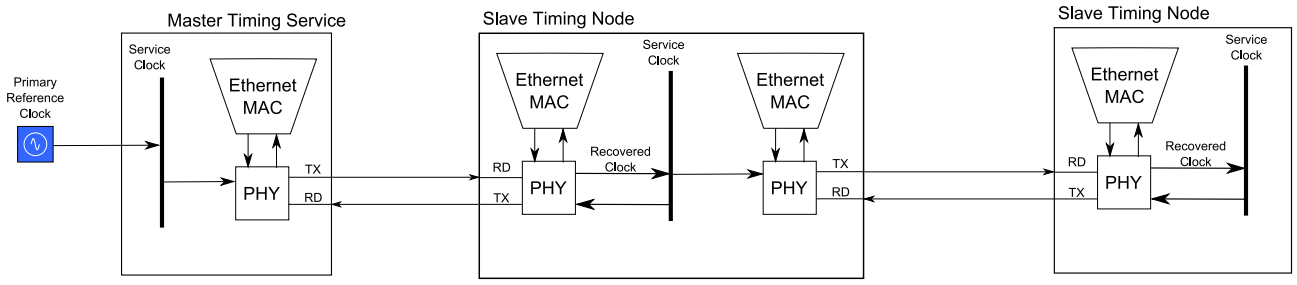


Figure 4.1: Synchronous Ethernet

several aspects not fully defined, it aims at describing different methods to fulfill the synchronization-related requirements.

4.1 Clock recovery

Since WR defines a synchronous network, it specifies that data will be emitted whether or not there are any frames to send to maintain slave clock locked. This issue is solved in by both Fast and Gigabit Ethernet data, their physical specification defines a 4b/5b and 8b/10b coding scheme, respectively, that allows that if there aren't frames to be sent, transmitters will send control characters defined by their physical link coding scheme and they can be either commas (K28.5) or link idle sequences (K28.5/D5.6).

In fact, the old 10Mbps (10Base-T) Ethernet is not even capable of synchronization signal transmission over the physical layer interface because a 10Base-T transmitter stops sending pulses during idle periods.

A 10Base-T transmitter simply sends a single pulse ("I am alive" pulse) every 16 ms to notify its presence to the receiving end. Of course, such infrequent pulses are not sufficient for clock recovery at the receiver.

Idle periods in faster Ethernet, higher than 100 Mbps inclusive, are continuously filed with pulse transitions, allowing continuous high-quality clock recovery at the receiver.

Clock recovery will be done at the slave side according with the specification of the physical link standard , which can be, as mentioned, 1000BASE-LX or 100BASE-TX. These physical standards define a second order PLL model used for clock recovery and a maximum jitter allowed.

As part of the architecture, a distinction should be made between the network clock and the service network as described below.

Network clock The network clock is used to discipline the synchronization function within the Ethernet Layer and therefor the bit rate leaving the Ethernet station.

Service clock Whithin existing Ethernet techologies the service is effectively asynchronous. In synchronous Ethernet, existing Ethernet services will continue to be mapped into and out of the Ethernet physical layer at the appropriate rates.

4.2 Synchronous Ethernet in White Rabbit

Network Synchronization in WRP is based on clock hierarchy, as mentioned before , with the highest accuracy clock at the top.

Slave clocks are PLL-based, locked to an external reference that is being transmitted by the data link . Slave clocks can be modeled as a low pass filter acting on phase fluctuations present on the input timing signal. Such-low pass filters can be described with a simple two-pole transfer function, whose most important parameter is the cut-off frequency f_c. For $f < f_c$ the slave clock is suppose to track input noise phase fluctuations. They are due to some thermal effects or some clock misbehaviour. On the other hand for $f > f_c$ the slave clock cuts off input phase fluctuations. In this case, they are usually due to external interferences.

This PLL will clean the recovery clock, ie. it will remove jitter generated from the clock recovery circuity before being fed to the transmitting device. However the PLL in WRP must provide additional functions beyond jitter cleaning.

The PLL used in SyncE must be able to detect failure of the recovered clock and switch the PLL to either another good reference in the system or into holdover mode. Requirements for SyncE are outlined in the timing characteristics of synchronous Ethernet equipment clock (ITU G.8262/Y1362) specifications. These specifications are based on ITU-T G.813 specification for SDH clocks.

The synchronization method used in White Rabbit is very similar to SyncE, in the sense that the network clock, that is traceable to a primary clock reference, is used to lock the service clock. In addition, White Rabbit defines the tool to compensate for long term phase deviations, where long-term implies that these variations are of frequecy greater that 10 Hz. This long term phase variations can't be attenuated by means of the PLL, but nevertheless they need to be reduced when requirements for the timing

network is to reach subnanosecond accuracy.

Long term phase deviations are due to thermal variation on both the oscillators, electronics and the data link.

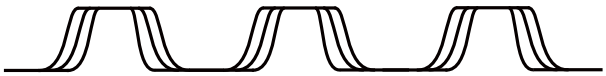


Figure 4.2: Phase drifts without WR

White Rabbit specifies the means to compensate for the phase deviations by allowing the slave timing node to feedback the recovery clock from the RX data link back its master. In SyncE specification only downstream frequency is required. The master recovers the clock from the RX data line and measures its phase difference to the clock which is traceable to the primary reference clock.

Figure 4.3: Phase drifts with WR

The measured phase difference will be transmitted to the slave timing station using WRCMP or PTP [PTP08]. The slave will use this value to compensate its phase.

The Phase Measurement block, as depicted in Fig. 4.4, should be able to measure phase with an accuracy of (To be defined).

The general arquitecture for delivering the physical layer clock from the Master to the Slave node in a peer-to-peer configuration is shown in Fig. 4.4.

Depicted in Fig. 4.5 is a White Rabbit Switch. All downlinks are synchronized to the recovered clock from the uplink, i.e. the master timing service. In addition, the long term phase deviations will be further cleaned by means of the phase compensation block. The result is a service clock synchronized to a clean recovered clock that is traceable to the PRC.

Further figures showing the frequency accuracy and stability of the clean clock related with the PRC will be shown later on.

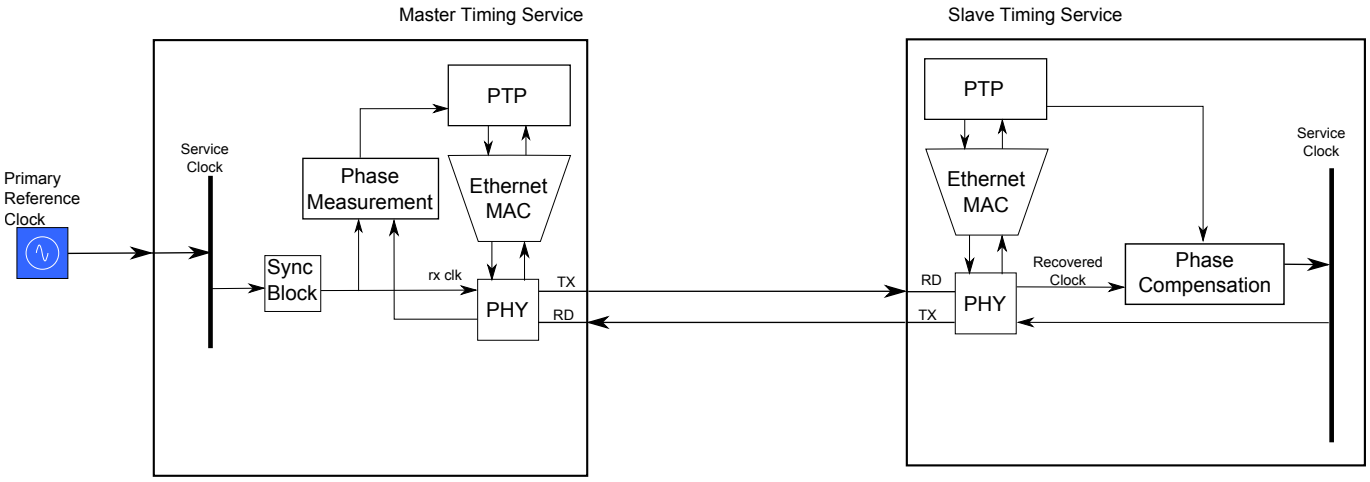


Figure 4.4: Synchronous Ethernet White Rabbit

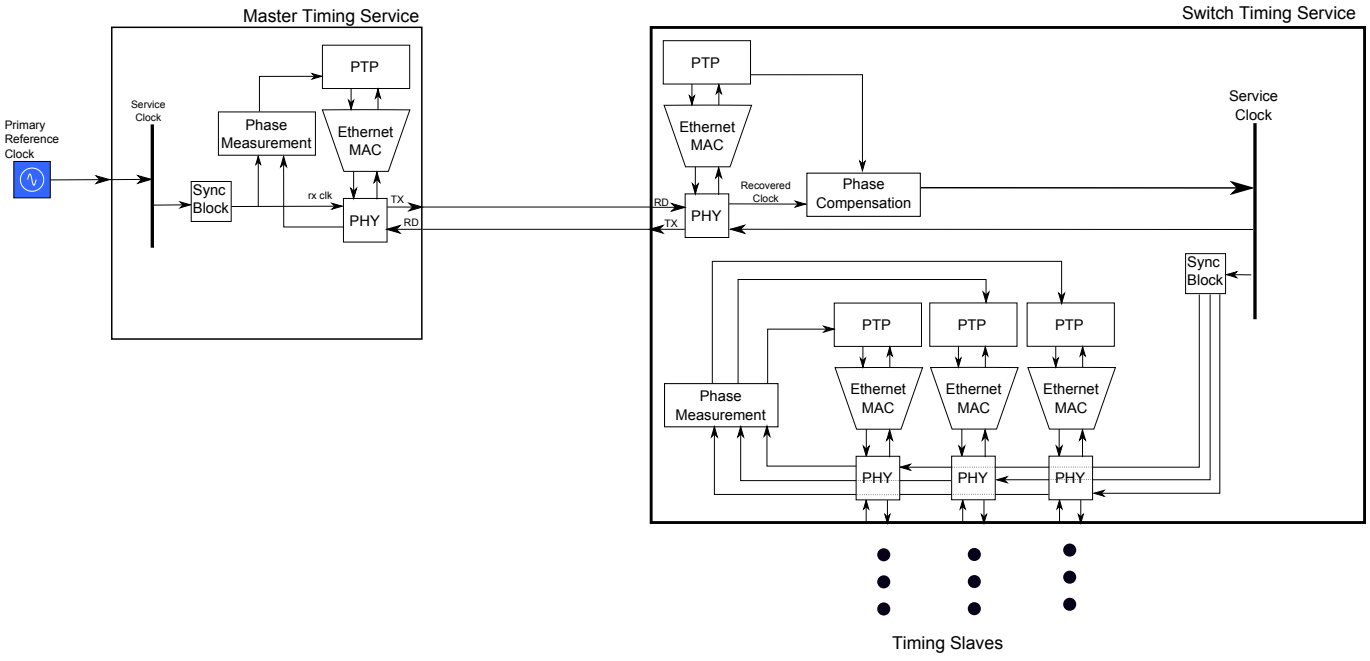


Figure 4.5: White Rabbit Switch

Chapter 5

Media access control frame

The description of White Rabbit, within this chapter, is to emphasize the order in which the bits are being transmitted to the destination machine so that they can be reconstructed and sent to the upper layers, in other words it defines the fields that form the structure of a White Rabbit frame. It also describes how the White Rabbit frames are being sent to the physical layer. An important remark to take into account in WRP networks is that it it is fully compliant with standard Ethernet traffic. This chapter is organized as follows: We first introduce the Ethernet elements of a normal frames and its Q extension, their description and functionality. In section 5.2, we describe the algorithm used for the frames to access the transmission medium. Starting in section 5.3 we introduce the White Rabbit frames types and their access to the transmission medium. Within this section, a description will be presented of how White Rabbit ensures that a frame will be transmitted and received by its destination node in a given time, that is prove that we can and how we can achieve determinism in critical data communications.

5.1 Elements of the MAC frame

The Ethernet Frame has seven fields:

- The Preamble
- Start Frame Delimiter (SFD)
- The Destination Address
- The Source Address,

- Length or Type field to indicate the length or protocol type,
- The Data which may include some padding if required,
- The frame sequence field containing a cycle redundancy check value to detect errors in a received frame.

Of these seven fields, all are of fixed size except for the data, pad, which may contain an integer number of bytes between the minimum and maximum values that are determined by the specific implementation. The ordering of the fields contained in an Ethernet frame can be seen in the figure below.

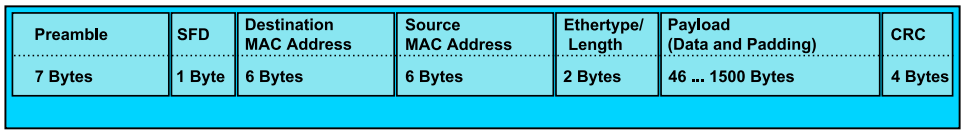


Figure 5.1: Ethernet Frame

The minimum frame size to an Ethernet frame, from the preamble, is set to be 64 bytes long, which justifies for the minimal size of the Payload field to be 46 bytes. The justification for this minimal frame size is to prevent a station from completing the transmission of a short frame before the first bit has even reached the far end of the cable, where it may collide with another frame. For a 10 Mbps LAN with a maximum length of 2500 meters, the round-trip has been determined to be around 500us. In a 10 Mbps LAN a bit takes 100 ns to be transmitted so the frame must have at least 500 bits to make this round trip. This number was rounded up to 512 bits or 64 bytes.

As the network speed is increased, the minimal frame length should obviously go up, or the cable length must come down proportionally. For instant, within a 2.5 Km LAN operating at 1Gbps network the minimal frame length would have to be 6400 bytes.

Each byte of the MAC frame is transmitted from the least-significant bit. As mentioned, the minimum size for data payload is 46 byte and the maximum size is of 1500 byte. Further, to send an Ethernet frame we will send, at the minimum 84 byte (with the preamble included) and at the maximum 1538 byte which in terms of time means a minimum of 0.672us and a maximum of 12.30us of slot time per frame in a GbE network or a

minimum of 6.72us to 123.04us in an 100Mb/s network. To conclude the minimum frame size is specified for historical reasons and also to maintain compability to the early standards, it could be reduced if required.

5.1.1 Elements of the Tagged MAC Frame

This format is an extension of the MAC Frame specified in Fig.5.1. The extensions for tagging are as follows:

- 4 bytes QTag Prefix is inserted between the end of the Source Address and the Length/Type field of the MAC frame. The QTag Prefix comprises two fields:
  - 2 byte constant Length/Type field value consistent with the Type interpretation and equal to the value of the 802.1Q Tag Protocol Type.
  - 2 byte field containing Tag Control Information.
- Following the QTag Prefix is the MAC Client Length/Type field, MAC Client Data, Pad (if necessary), CRC, fields of the basic MAC frame.
- The length of the frame is extended by 4 byte by the QTag Prefix.

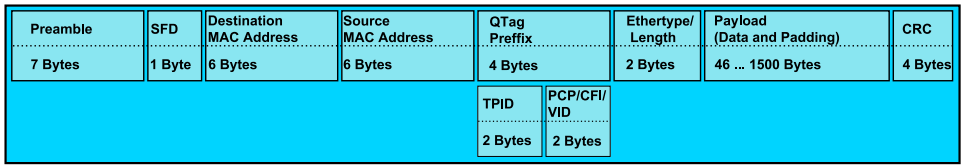


Figure 5.2: IEEE 802.3 Tag Mac Frame

The structure and semantic within the Tag Control Information field is defined in IEEE P802.1Q [IEE06].

White Rabbit Frame will be the Q-Tagged MAC Frame.

Preamble The preamble is a 7 byte field that is used to allow the circuit clock to reach its steady-state synchronization within the received frame's timing.

Start Frame Delimiter The SFD field is the sequence 10101011. It immediately follows the preamble pattern and indicates the start of a frame.

Destination MAC Address The Destination Address field specifies the station(s) for which the frame is intended. It may be an individual or multicast (including broadcast) address.

Source MAC Address The Source Address field specifies the station sending the frame. The Source Address field is not interpreted by the CSMA/CD MAC sublayer.

EtherType/Lenght This two bytes field takes one of two meanings, depending on its numeric value. For numerical evaluation, the first byte is the most significant octet of this field.

- If the value of this field is less than or equal to the value of maxValidFrame, then the Length/Type field indicates the number of MAC client data bytes contained in the subsequent data field of the frame (Length interpretation).
- If the value of this field is greater than or equal to 1536 decimal (equal to 0600 hexadecimal), then the Length/Type field indicates the nature of the MAC client protocol (Type interpretation).

The Length and Type interpretations of this field are mutually exclusive. When used as a Type field, it is the responsibility of the MAC client to ensure that it operates properly when the MAC sublayer pads the supplied data. Regardless of the interpretation of the Length/Type field, if the length of the data field is less than the minimum required for proper operation of the protocol, a PAD field (a sequence of bytes) will be added at the end of the data field but prior to the CRC field, specified below. The procedure that determines the size of the PAD field

Payload This payload field contains the data to be sent by the MAC client.

The shortest valid transmission in full duplex mode must be at least 48 bits in length. While collisions do not occur in full duplex mode MACs, a full duplex MAC nevertheless discards received frames containing less than 48 bits. The discarding of such a frame by a MAC is not reported as an error[IEE04].

CRC A cyclic redundancy check (CRC) is used by the transmit and receive algorithms to generate a CRC value for the CRC field. The frame check sequence field contains a 4 byte cyclic redundancy check value. This value is computed as a function of the contents of the source address, destination address, length, data and pad (that is, all fields except the preamble, SFD, FCS, and extension)[IEE04].

White Rabbit will use of QTag frame extension frame structure defined in 5.1.1. This will allow the use of QoS features. Nevertheless Virtual LANs is not forseen for WRP.

5.2 Media access control

The communication operation mode of WRP will be full duplex as defined in IEEE 802.3 standard. Since there is no contention for use of a shared medium, the multiple access (i.e., CSMA/CD) algorithms are unnecessary.

The configuration envisioned for operation consists of a switch with a dedicated link connecting each switch port to a single node.

5.2.1 Frame Transmission

When a MAC client requests the transmission of a frame, the MAC constructs the frame from the client-supplied data. It prepends a preamble and a Start Frame Delimiter (SFD) to the beginning of the frame. Using information provided by the client, it also appends a PAD at the end of the MAC information field of sufficient length to ensure that the transmitted frame length satisfies a minimum frame-size requirement. It also preempts destination and source addresses, the length/type field, and appends a CRC to provide for error detection.

In full duplex mode, there is no need to avoid contention with other traffic on the medium. Frame transmission may be initiated after the interframe delay, regardless of the presence of received activity.

When transmission has completed without contention, the MAC informs the upper layers and awaits the next request for frame transmission.

5.2.2 Frame Reception

The receiving clock of both Fast and Gigabit Ethernet is always synchronized to their upper station link. Even if there aren't frames to send the station will send control characters to the transmission medium to maintain the clocks synchronized and locked. As the encoded bits arrive from the medium,

they are decoded and translated back into binary data. The physical layer passes subsequent bits up to the MAC sublayer, where the leading bits are discarded, up to and including the end of the preamble and Start Frame Delimiter.

The MAC checks the frame's Destination Address field to decide whether the frame should be received by the station. If so, it passes the destination address , the source address, the length/type, the data and (optionally) the CRC fields to the MAC client, along with an appropriate status code. It also checks for invalid MAC frames by inspecting the CRC to detect any damage to the frame en route, and by checking for proper byte boundary alignment of the end of the frame. Frames with a valid CRC may also be checked for proper byte boundary alignment.

CRC validation is essentially identical to CRC generation. If the bits of the incoming frame (exclusive of the CRC field itself) do not generate a CRC value identical to the one received, an error has occurred and the frame is identified as invalid or as fragmented frame, we will discuss this fragmented frames in the next sections.

5.3 White Rabbit Frames

To distinguish between White Rabbit and other Ethernet traffic two different fields of EtherTypes are defined. All the others frames types are treated as being non-WRP traffic and the White Rabbit station will handle them as if it was an "normal" Ethernet station.

White Rabbit uses the same frame structure as it is defined by the IEEE 802.3Q standard. Its main difference is the access to the transmission medium.

As described before, in full duplex transmission link the access of an Ethernet frames to the transmission medium starts as soon as the MAC receive the signal that the last frame transmission is concluded, that is that the transmission link is free for transmission. In half duplex transmission links the MAC starts its transmission according with the CSMA/CD algorithm. Half duplex is not envisioned to be used in White Rabbit networks so this algorithm will not be described in this document.

5.3.1 High Priority Frames

The WRP specifies the use of an High Priority Frame to allow real-time communications. To achieve this end the MAC node will provide a packet with an unique EtherType Field equal to 0xa0a0.

These frames will be used for time-critical control data where data transmission is deterministic.

Besides the Ethernet type field the fields of an HP follows a structure that contains four mandatory fields, the hp_flag field, the header_crc field, the lt_header followed by its data payload.

The structure of the frame is shown below:

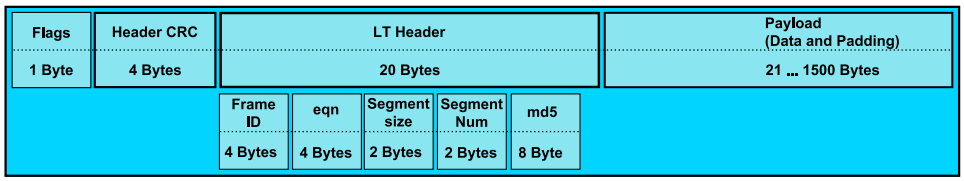


Figure 5.3: High Priority Frame

hp_flags field

The hp_flags is a one byte field containing frame flags. Table below shows this field description.

header_crc field

Contains a 32-bit CRC of the received data so far, it will allow an early detection of an corrupted header. In the case of a corrupted frame detection the full frame will be dropped

lt_header field

This field is only mandatory in case of LT_ENABLE flag is set.

(Detailled Description is needed)

5.3.2 Standard Priority

The White Rabbit protocol also specifies the use of a Standard Priority frame (SP). This frame type is specified in the WRP network to transport messages with a lower priority relative with the HP frames. The Standard

Flag	Flag Name	Description
0x01	LT_ENABLE	Frame contains LT encoded payload and LT encoding header is present
0x02	BURST_NEXT	The current frame is part of burst of HP frames and switch/node should expect reception of another HP frame immediately after current HP frame (after respecting the inter frame gap). If BURST_NEXT flag is set, the switch/node shouldn't try to send buffered SP/non-WR frames after transmission of current HP frame.
0x04	BURST_LAST	This flags informs about the last frame of the HP burst.
0x08	DROP_ON_COLLISION	Collision of two HP frames, this frame should be dropped.
0x10	DELAYED	This frame was delayed due to a collision.

Table 5.1: HP Frame fields

frame type provides a package with an unique EtherType Field equal to 0xa0a1.

If during the transmission of a SP package there is a the request for the transmission of an HP frame, the SP frame transmission will be interrupted and its remaining data will be stored until the transmission of the HP frame is completed.

After sending the HP Frame, the WRP station will continue to transmit the remaining SP frame data. As can be easily assumed, due to the priority scheme we can't ensure the determinism of a SP frame as we can for the the HP.

The SP frames are used for WRP non time critical data exchange. Its structure is composed by two fields, the sp_header field and the payload field.

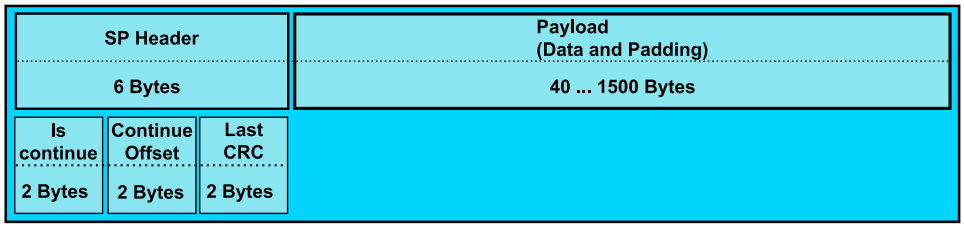


Figure 5.4: High Priority Frame

sp_header field

The sp_header field structure contains three variables that are described in the table 5.2.

5.3.3 HP Frames transmission

HP frames are transmitted through the whole network (all stations are receiving them, ie. broadcast), however they are processed only by the nodes to which they are assigned to via MAC destination address (either a single node or multicast group). This is to prevent overflowing FIFOs in switches. While HP frame is being received by switch, it should pause routing of whole non-HP traffic.

Element Name	Type	Description
is_continue	Non-zero	The value of this element informs, that payload of current SP frame contains continuation of previous non-HP frame which has been fragmented due to routing of HP frame.
continue_offset	uint16_t	Contains payload offset at which previous frame was interrupted
prev_segment_crc	uint32_t	Value of CRC32 of previous segment of fragmValue of CRC32 of previous segment of fragmented frame, allowing for detection of proper segment order during reassembly the frame, allowing for detection of proper segment order during reassembly

Table 5.2: SP Frame fields

Comments:the pause command should be used wisely...,how come by broadcasting the HP frames we will prevent overflowing the switches. When the WRP client requests the transmission of an HP frame, all current frames transmission will be set stored to be sent a soon as the HP transmission completes.

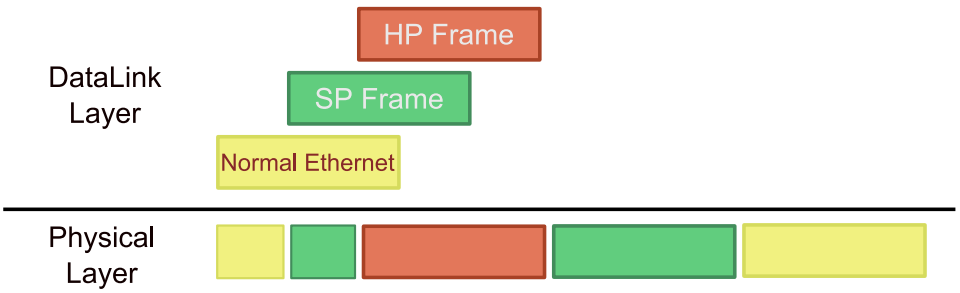


Figure 5.5: WRP Frames Transmission

Traffic transmission according to the diferent priority levels defined by WRP is shown in Fig. 5.5. Independent if a frame is being transmitted or not an HP frame will be transmitted to the as soon as there is a request to

start it at the Data Link Layer.

5.3.4 SP Frames transmission

White Rabbit SP Frames have the same priority level as normal Ethernet traffic, so they will transport nonsensitive timing messages. As mentioned before, in the request of an HP frame transmission the current (being transmitted) SP frame will be interrupted, and the transmission of the HP frame will start immediately. As soon as the HP frame is transmitted, the WRP will flag the interrupted SP frame, and will start to transmit the remaining data.

TO DO ...

5.3.5 Fragmentation

TO BE FUTHER DISCUSSED ...

To decrease the latency from the request of an HP frame transmission to its arrival at the destination station, a new feature is added to the data link layer that suspends the non-HP frame that is being transmitted to start the transmission of the HP frame. The suspended non-HP frame will be stored for later transmission. They will be retransmitted as soon as there is no HP frames requested for transmission. It can be easily conclude that this prioritization scheme will decrease the bandwidth of non-HP frames. To turn around this lack of efficacy a feature is added to WRP that allows fragmentation of non-HP frames by only retransmitted the not transmitted part of the non-HP frame.

This fragmentation process needs to be implemented on the client node so that SP or non-WRP frames can be reconstructed. The fragmentation process can occur multiples time on a non-HP frame. The MAC detects the reception of an fragmented frames by verifying that it received a frame with an invalid CRC. On the reception of an invalid CRC the MAC will store the received frame, the client should then expect the reception of an HP Frame, in the case that this does not occur then the MAC can delete the previous frame because in fact it isn't part of a fragmented frame but instead it is a c corrupt frame.

In case the station receives an HP frame after the invalid CRC, the MAC should assume that the previous frame is fragmented and so it should wait until the HP frame reception is terminated to reassemble the fragmented frame. After the reception of the HP frame the client should expect that

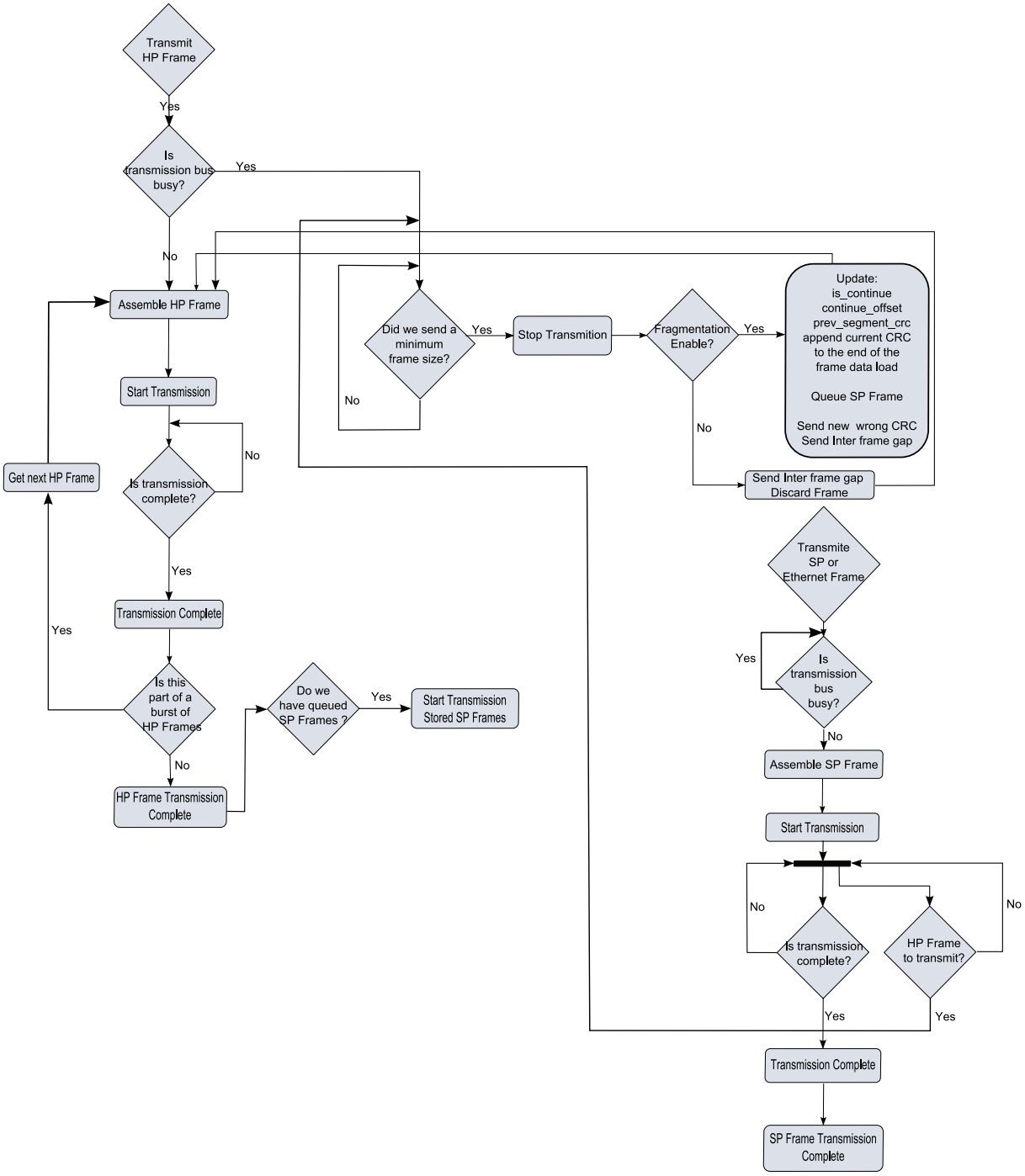


Figure 5.6: WRP Transmission

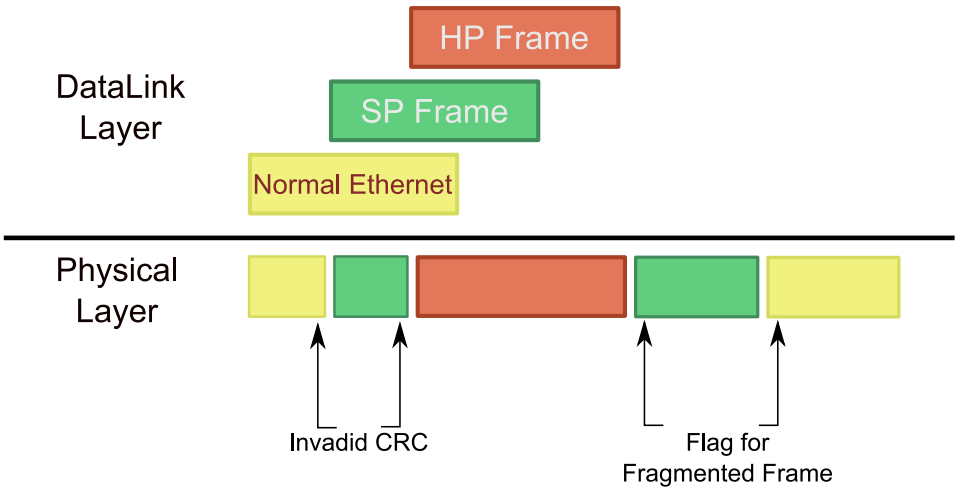


Figure 5.7: White Rabbit Fragmentation

next frame will contain the remaining data of the previous invalid frame, so upon receiving the complete frame with a valid CRC the client should reassemble the frame and verify if the full frame CRC is correct if this occurs it should send the complete frame to the MAC client.

TO DO ....

5.3.6 Congestion

TO BE DISCUSSED ...

As the HP frames are routed deterministically, with highest priority, collisions may occur when multiple devices try to send HP frames independently. To avoid such situations, higher layer protocols should schedule HP frame transmissions. Nevertheless, collisions may still happen, caused by stray HP frames sent by faulty or non-WRP nodes. WRP protocol provides simple collision management for HP frames by adding DROP_ON_COLLISION, DELAY_ON_COLLISION and DELAYED flags in each HP frame header. The switch may behave in multiple ways (see figure 5), depending on state of these flags:

1. Both packets have DROP_ON_COLLISION = 1 - both packets are dropped.

2. 1st packet: DROP_ON_COLLISION = 1, 2nd packet: DROP_ON_COLLISION = 0: 1st packet is immediately terminated and 2nd packet is routed

deterministically.

3. 1st packet: DROP_ON_COLLISION = 0, 2nd packet: DROP_ON_COLLISION = 1: 1st packet is routed deterministically, 2nd packet is dropped.

4. Both packet have DROP_ON_COLLISION = 0: 1st packet is routed deterministically while 2nd packet is buffered. After transmission of 1st packet, 2nd packet is routed non deterministically with DELAYED flag set.

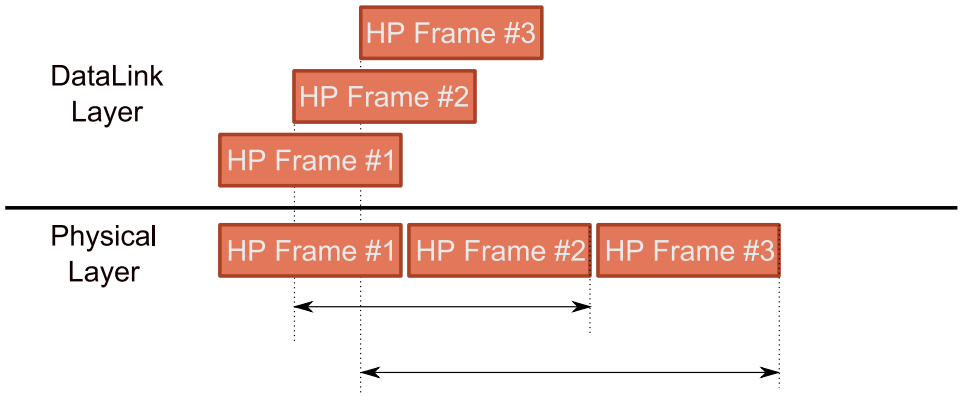


Figure 5.8: WRP Collisions

The DELAY_ON_COLLISION flag allows the switch to delay colliding frames when the its value is set to 1. Switches should interpret DELAY_ON_COLLISION in similar way as described above, except no packets should be dropped. If both colliding frames have DELAY_ON_COLLISION= 0 and DROP_ON_COLLISION = 0, the second frame is delayed without respect to flags and appropriate DELAYED bit is set.

White Rabbit assumes that the network eliminates cyclic forwarding of frames within each communication path (e.g. by using a spanning tree protocol)

Chapter 6

The WRCMP Protocol

In this section, the White Rabbit Control Message Protocol (WRCMP) format is described along with some use cases. WRCMP along with PTPv2 is used by WR-compatible network devices for detection, low-level maintenance and network synchronization.

6.1 WRCMP message encapsulation

All WRCMP messages are sent using SP frames. HP frames provide only transport layer for timing-sensitive traffic, and they shall never be used for WRCMP messages. Every WRCMP packet has following NTLV (name-type-length-value) structure:

define FIELD_SIGNED 0x40 // field type modifier flag , stating that field value is signed

define FIELD_ARRAY 0x80 // field type modifier flag , stating that field value is array of field_type

__attribute__ ((packed)) struct wrp_message {
uint16_t num_fields ; // Number of NTLV fields in packet
struct field_table [] { // Table of NTLV fields (num_fields entries)
char field_name [4]; // 4- character unique name of field
uint8_t field_type ; // field type (see table 1)
uint16_t field_length ; // total field data size (in bytes) };
char field_values [...];

};

Table 6.2 shows available field_types. All multibyte binary data is encoded in big-endian format. An example of WRCMP message containing 4 different NTLV fields is provided in table 6.1:

+00:	00 04	- num_fields (4)
+02:	4D 53 49 44	- field 0 name ('MSID')
+06:	00	- field 0 type (msgid_t)
+07:	00 02	- field 0 length (2 bytes)
+09:	49 4E 54 31	- field 1 name ('INT1')
+0D:	43	- field 1 type (sint32_t)
+0E:	00 04	-field 1 length (4 bytes)
+10:	41 52 52 31	- field 2 name ('ARR1')
+14:	82	- field 2 type (array of uint16_t)
+15:	00 08	- field 2 length (8 bytes = 4 uint16_t's)
+17:	53 54 52 31	- field 3 name ('STR1')
+1B:	07	- field 3 type (UTF8 string)
+1C:	00 0D	- field 3 length
+1E:	04 D2	- field 0 value (MSID = 1234 decimal)
+20:	FF FE 79 80	- field 1 value (INT1 = -100000)
+24:	00 0A 07 D0	
+28:	1E 61 D4 31	- field 2 value (ARR1 = 10, 2000, 7777, 54321)
+2C:	48 65 6C 6C	- field 3 value (STR1 = 'Hello, world')
+30:	6F 2C 20 77	
+34:	6F 72 6C 64 00	

Table 6.1: WRCMP Message

6.1.1 WRCMP INVITE

Direction: switch/master $\longrightarrow$ node

Sent in response to: detection of new node in network

Expected response: WRCMP_INVITE_RESPONSE

Type ID	Flags	C type	Base size	Description
0x00	none	msgid_t	2	16-bit unsigned integer, containing unique type identifier of the message. Can be used only once in each packet
0x01 0x41	FIELD_SIGNED FIELD_ARRAY	sint8_t uint8_t	1	8-bit unsigned/signed integer in two's complement format
0x02 0x42	FIELD_SIGNED FIELD_ARRAY	sint16_t uint16_t	2	16-bit unsigned/signed integer in two's complement format
0x03 0x43	FIELD_SIGNED FIELD_ARRAY	sint32_t uint32_t	4	32-bit unsigned/signed integer in two's complement format
0x04 0x44	FIELD_SIGNED FIELD_ARRAY	sint64_t	8	64-bit unsigned/signed integer in two's complement format
0x05	FIELD_ARRAY	uint64_t	4	IEEE754 32-bit (single precision) floating point
0x06	FIELD_ARRAY	float	8	IEEE754 64-bit (double precision) floating point
0x07	none	char *	variable	ASCII null-terminated string, UTF8 encoding

Table 6.2: WRCMP message field types

Source address: sender address

Destination address: slow protocol default (01-80-c2-00-00-02)

Format: see table 6.3

WRCMP_INVITE is sent by the switch after the detection that a new node has been connected. Its purpose is to inform the new node about network/timing configuration and make a request for its capabilities. To make sure that the message will not be routed further, destination address is the slow protocol default address (01-80-c2-00-00-02). This message will be sent to new connected nodes and it will be acknowledged by a WRCMP_INVITE_RESPONSE message. Invite message contains all addresses that node needs to know and important timing information.

6.1.2 WRCMP INVITE RESPONSE

Direction: node $\longrightarrow$ switch/master

Sent in response to: WRCMP_INVITE

Expected response: WRCMP_ACK

Source address: sender address

Destination address: switch address from

WRCMP_INVITE Format: see table 6.4

WRCMP_INVITE_RESPONSE packet contains the distinguishing configuration of a new connected device, allowing the switch/master to check if it is WR-compliant.

6.1.3 WRCMP ACK

Direction: any

Sent in response to: WRCMP_INVITE_RESP, WRCMP_REPORT_NODE, WRCMP_REPORT_DELAY, WRCMP_HEARTBEAT

Expected response: none

Source address: sender address

Destination address: any Format: see table 6.5

WRCMP_ACK packet is used to acknowledge reception of packets listed above or to indicate error.

6.1.4 WRCMP REPORT NODE

Direction: node $\longrightarrow$ switch, switch $\longrightarrow$ STM

Sent in response to: node/switch event or state change

Expected response: WRCMP_ACK

Source address: sender address

Field	Type	Code	Description
msg_id	msgid_t	MSID	Unique message identifier for WRCMP_INVITE. Equals to 0x01.
switch_mac	uint8_t[6]	SWMA	MAC address of switch to which node is directly connected.
master_mac	uint8_t[6]	STMM	MAC address of System Timing Master
switch_port_id	uint16_t	SPRT	Number of switch port to which node is connected
switch_layer_id	uint16_t	SLAY	Layer of network to which device is connected: 0 - directly to master, 1 - 1 hop to master, 2 - 2 hops to master, etc...
min_delay_sync_period	uint32_t	MIDS	Minimal period of delay compensations performed by switch (in milliseconds)
max_delay_sync_period	uint32_t	MADS	Maximum period of delay compensations performed by switch (in milliseconds)
min_heartbeat_period	uint32_t	MIRP	Minimal period of sending WRCMP_HEARTBEAT messages to closest switch
max_heartbeat_period	uint32_t	MARP	Maximum period of sending WRCMP_HEARTBEAT messages to closest switch
cap_wr_switch_type	uint8_t	CPST	Type of switch to which node is connected: • 0x01 - SWITCH_TYPE_BB - backbone (fiber) WRP switch • 0x02 - SWITCH_TYPE_TP_100M - twisted-pair 100 Mbps WRP switch • 0x03 - SWITCH_TYPE_TP_1G - twisted-pair 1 Gbps WRP switch

Table 6.3: WRCMP INVITE message format

Field	Type	Code	Description
msg_id	msgid_t	MSID	Unique message identifier for WRCMP_INVITE_RESPONSE. Equals to 0x02.
node_mac	uint8_t[6]	NOMA	Node's own MAC address.
node_version	uint32_t	VRSN	32-bit identifier of node protocol version
node_type	uint16_t	NOTY	Type of node: • 0x01 - NODE_BB_SWITCH (backbone WR fiber switch), • 0x02 - NODE_BB_SWITCH_ALTERNATE (backbone WR fiber switch, alternate up-link port), • 0x03 - NODE_BB_SLAVE (generic WR slave device), • 0x04 - NODE_TP_GATEWAY (WR twisted pair gateway), • 0x05 - NODE_TP_SWITCH (twisted-pair WR switch)
cap_sync_mode	uint8_t	CPSY	Nonzero value means that node is capable of synchronous operation
cap_phase_measurement	uint8_t	CPPM	Nonzero value means that node can perform fine phase shift measurement for delay compensation
cap_ptp	uint8_t	CPPT	Nonzero value means that node supports delay compensation via PTPv2
cap_hp_recv	uint8_t	CPHR	Nonzero value means that node is capable of receiving HP frames
cap_hp_send	uint8_t	CPHS	Nonzero value means that node is capable of sending HP frames
cap_fragmentation	uint8_t	CPFR	Nonzero value means that node is capable of sending and receiving fragmented SP/non-WR frames
min_hp_routing_delay	uint32_t	MIHR	(switch/gateway only) Minimum and maximum values of HP packet routing delays (in 8ns clock cycles). If they are equal, routing time is assumed to be constant.
max_hp_routing_delay	uint32_t	MAHR	See above
max_hp_delay_uncert	uint32_t	MADU	(switch/gateway only) Maximal value of non-deterministic (caused by clock misalignment) delay fluctuations for HP routing in units of picoseconds.
max_slave_asymmetry	uint32_t	MASS	Maximal value of slave node RX/TX

Field	Type	Code	Description
msg_id	msgid_t	MSID	Unique message identifier for WRCMP_ACK. Equals to 0x03.
ack_value	uint16_t	ACKV	Zero means that packet previous requested has been acknowledged. Non-zero value means that error occurred and the message was rejected. Acknowledge value field may be used to send value of error code.

Table 6.5: WRCMP ACK message format

Destination address: switch or master
Format: see tables 6.6 & 6.7
WRCMP_REPORT_NODE packet is used to report delay measurements results and the current node state to:

- System Timing Master (used for calculating delays between HP frames to prevent collisions and reporting state of nodes).
- closest switch

WRCMP_REPORT_NODE is sent by each node to closest switch when node state changes. It can be issued when:

- node have performed initial delay compensation,
- node has changed its uplink port,
- amount of corrupted packets received by node exceeded certain limit.
- when switch/master requests immediate report by sending WRCMP_REQUEST_REPORT message.

The switch can also independently issue WRCMP_REPORT_NODE message to STM in response to state change of one of connected nodes:

- when new node has just been physically connected/disconnected to any of switch ports,
- when node stopped sending WRCMP_HEARTBEAT messages and seems dead,
- if amount of corrupted packets received from node exceeds certain limit,

- when link delay between switch and any of nodes changed significantly.

Switches shall forward all ingress WRCMP_REPORT_NODE messages to System Timing Master.

Field	Type	Code	Description
msg_id	msgid_t	MSID	Unique message identifier for WRCMP_REPORT_NODE. Equals to 0x06.
node_mac	uint8_t[6]	NOMA	MAC address of reported node
switch_mac	uint8_t[6]	SWMA	MAC address of switch to which reported node is directly connected.
current_uplink_port	uint8_t	CUUP	(switch/gateway only) Current uplink port used. 0 = primary, 1 = alternate
uptime	uint64_t	UPTM	Node uptime in seconds.
rx_total	ARRAY of uint64_t	RXTO	Total number of received packets (including HP)
rx_hp	ARRAY of uint64_t	RXHP	Total number of received HP packets
rx_errors	ARRAY of uint64_t	RXER	Total number of invalid packets received (including HP)
rx_errors_hp	ARRAY of uint64_t	RXEH	Total number of invalid HP packets received
event	uint16_t	EVNT	Reported event. Detailed description in table ??
delay_twoway	uint64_t	DETW	Two-way delay (STM - node - STM). In picoseconds.
delay_twoway_alternate	uint64_t	DETA	(switch/gateway only) Two-way delay on alternate uplink port (STM - node - STM). In picoseconds.
max_asymmetry	uint32_t	MAAS	Maximal uplink path asymmetry in picoseconds.

Table 6.6: WRCMP REPORT NODE message format

6.1.5 WRCMP REPORT DELAY

Direction: switch ⟶ node

Sent in response to: after delay measurement (periodic event)

Expected response: WRCMP_ACK

Source address: switch/master address

Value	Name	Description
0x01	JUST_CONNECTED	Node have just been connected to switch
0x02	UNSYNCHRONIZED	Node is connected but it haven't yet performed delay compensation
0x03	READY	Node is ready
0x04	UPLINK_PORT_CHANGE	(switch/gateway only) uplink port has been changed due to primary link failure or when primary link is working again
0x05	NODE_NOT_RESPONDING	Issued by switch when node which seems to be connected - carrier is present - didn't report itself in expected time
0x06	DISCONNECTED	Carrier lost, node is disconnected
0x07	ERROR_LIMIT_EXCEEDED	Amount of erroneously received packets received

Table 6.7: WRCMP REPORT NODE event field values

Destination address: measured node

Format: see table see table 6.8

The WRCMP_REPORT_DELAY packet is sent to node after the switch finished the link delay measurement. Using this message information slaves can perform delay compensation. These packets are never forwarded by switches, as defined in the WR specification.

6.1.6 WRCMP HEARTBEAT

Direction: node $\longrightarrow$ switch

Sent in response to: none (periodic event)

Expected response: WRCMP_ACK

Source address: node address

Destination address: switch/master address

Format: see table see table 6.9

WRCMP_HEARTBEAT packet is sent periodically by the node to closest switch to indicate that it is alive. The frequency boundaries for sending heartbeat messages are defined by the WRCMP_INVITE message. If the node hasn't sent WRCMP_HEARTBEAT messages for several periods, switch will assume that it is dead (hanged up or faulty) and issue proper WRCMP_REPORT_NODE message for STM. To avoid making excessive traffic in network, heartbeat packets are never forwarded by switches.

Field	Type	Code	Description
msg_id	msgid_t	MSID	Unique message identifier for WRCMP_REPORT_DELAY. Equals to 0x07.
utc_timestamp	uint64_t	UTCT	UTC send timestamp of this packet.
ts_value	uint32_t	TSVA	PTP counter send timestamp of this packet (in 8ns clock cycles)
delay_twoway	uint64_t	DSN2	Measurement result - two-way overall link delay between switch and node (in picoseconds)
delay_downlink master_to_switch	uint64_t	DMSD	Overall downlink delay between STM and switch to which node is directly connected (in picoseconds)
delay_uplink switch_to_master	uint64_t	DSMU	Overall uplink delay between switch and System Timing Master to which node is directly connected (in picoseconds)
master_asymmetry	uint32_t	MASY	RX/TX path asymmetry of switch to which node is directly connected (in picoseconds)
link_asymmetry	uint32_t	LASY	Link asymmetry in picoseconds

Table 6.8: WRCMP REPORT DELAY message format

Field	Type	Code	Description
msg_id	msgid_t	MSID	Unique message identifier for WRCMP_HEARTBEAT. Equals to 0x08.

Table 6.9: CMP HEARTBEAT message format

6.1.7 WRCMP REQUEST REPORT

Direction: switch ⟶ node

Sent in response to: none

Expected response: WRCMP_REPORT_NODE

Source address: switch/master address

Destination address: node address

Format: see table see table 6.10

WRP_REQUEST_REPORT packet is issued by STM/switch to force node to send unconditional WRP_REPORT_NODE message (e.g. not in response to state change or event).

Field	Type	Size	Description
msg_id	msgid_t	MSID	Unique message identifier for WRCMP_REQUEST_REPORT. Equals to 0x09.

Table 6.10: WRCMP REQUEST REPORT message format

6.2 WRCMP use cases

6.2.1 New node connected to switch

Detection of new connected nodes is done by looking for presence carrier (e.g. valid 8b10b symbols) on switch's downlink ports. If carrier is present, the following procedure is executed:

1. Switch waits for 1 second and sends WRCMP_INVITE packet to the new connected node
2. Node responds with WRCMP_INVITE_RESPONSE packet containing its capabilities
3. If correct WRCMP_INVITE_RESPONSE was received, switch sends ACK packet to the node. Otherwise switch sends WRCMP_INVITE packet again. Retransmission can be performed many times, depending on switch settings.
4. If there is no response from node, or the response is malformed, switch assumes that device is non-synchronous and non-WR capable. Switch issues WRCMP_REPORT_NODE message to System Timing Master,

stating that node has been JUST_CONNECTED and is not WR-compatible.

5. If the node replied with proper WRCMP_INVITE_RESPONSE:

   (a) WRCMP_ACK packet is sent to node,

   (b) WRCMP_INVITE_RESPONSE packet received from node is forwarded to STM. STM replies with WRCMP_ACK. If there is no response, retransmission is performed.

   (c) WRCMP_REPORT_NODE packet is issued to STM with proper node state. STM replies with WRCMP_ACK. If there is no response, switch retransmits WRCMP_REPORT_NODE message.

6. Switch starts performing delay measurements if the slave is WR-compliant.

6.2.2 Delay measurement

Delay/phase measurement is done using PTPv2 peer to peer delay message exchange as described:

1. Master (switch or STM) sends Pdelay_req packet to slave

2. Slave (node) responds with Pdelay_resp packet containing value of Pdelay_req reception timestamp (t2) and Pdelay_resp_followup containing value of Pdelay_resp send timestamp (t3).

3. Points (1-3) can be repeated several times, so the master could determine the average phase shift introduced by link.

4. Master calculates link delay and asymmetry and reports it to node using WRCMP_REPORT_DELAY message, or plain PTP message.

6.2.3 Node reporting and maintenance

For network maintenance and timing model optimization, nodes are obliged to report their state. Reporting is done following way:

1. If state of the node has changed, node must immediately issue proper WRCMP_REPORT_NODE message to switch. Switch responds with WRCMP_ACK. If there is no reply, WRCMP_REPORT_NODE is sent again. These packets are exchanged only between node and the closest switch.

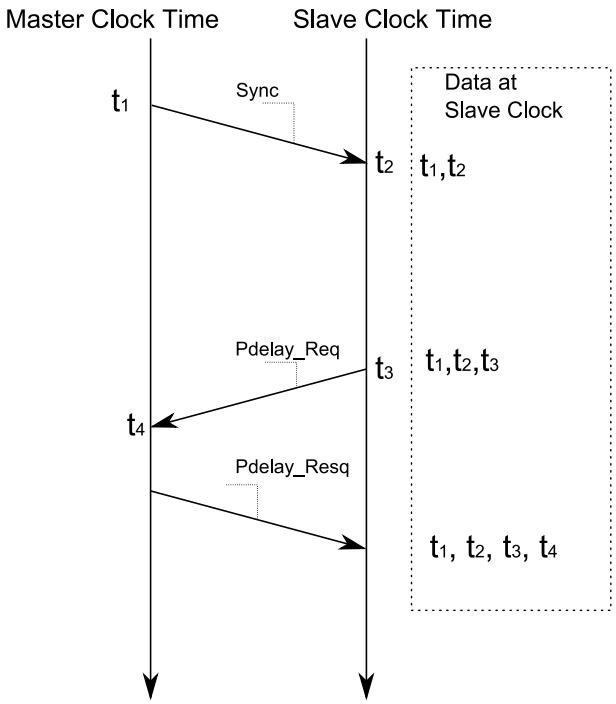


Figure 6.1: PTP Clock Synchronization

2. If the switch received proper WRCMP_REPORT_NODE message and everything seems OK it just forwards it to System Timing Master.

3. If node is not sending WRCMP_HEARTBEAT packets, but valid carrier is present, switch issues WRCMP_REPORT_NODE packet with NO_RESPONSE state and forwards it to System Timing Master. If carrier has been lost, switch issues WRCMP_REPORT_NODE packet with DISCONNECTED state to System Timing Master. STM acknowledges reception of WRCMP_REPORT_NODE by sending WRCMP_ACK. If there is no response from STM, reporting packet is sent again.

Switch/STM can also force nodes to send WRCMP_REPORT_NODE messages immediately by sending WRCMP_REQUEST_REPORT messages.

6.2.4 Interoperability with non-WR compliant nodes

WR protocol is specified to be compatible with non-WR nodes/network segments, i.e. normal Ethernet traffic. If a new node doesn't respond to invitation message with proper WRP_INVITE_RESPONSE, it will be assumed that it is incompatible with WR protocol. In this situations, the switch:

- disables deterministic HP frame routing for incompatible node. HP frames delivered for the node are treated like normal frames to prevent fragmentation.

- (optional) enables on-the-fly fragmented frame reconstruction on port to which incompatible node is connected or drops fragmented frames (depending on configuration)

Non-WR nodes can still use synchronous mode and PTP delay measurements with precision of single Ethernet clock cycle (8 ns).

Chapter 7

IEEE 1588 - Precise Time Protocol

Precise Time Protocol (PTP) [PTP08] also named IEEE1588 enables precise synchronization of clocks in measurement and control systems implemented with technologies such as network communication, local computing and distributed objects.It includes formal mechanisms for message extensions, higher sampling rates, correction for asymmetry, a clock type to reduce error accumulation in large topologies, and specifications on how to incorporate the resulting additional data into the synchronization protocol. The standard permits synchronization accuracies better than 1 nanosecond.

PTPv2 was choosen as the protocol for delay measurement and compensation in WR network. Therefore, every WR device shall implement PTPv2 packet timestamping and at least following set of PTPv2 messages. White Rabbit will be complient with both one-step clock and two-steps clock as defined by its standard.

- Sync
- Follow_up
- Delay_req
- Delay_resp
- Pdelay_resp
- Pdelay_resp_follow_up

These messages are necessary to synchronize two WR devices. However, to keep interoperability with non-WR PTP slaves, all switches must implement fully featured PTPv2 daemon.

PTPv2 messages used by WR-compatible devices are encapsulated into Ethernet frames according to IEEE1588.D2.2 annex F. Such frames are distinguished by unique value of EtherType (0x88f7), and by the non-forwardable destination address 01-80-C2-00-00-0E.

All delay calculations and phase shift measurements must be performed by master side (e.g. switch or STM). Slaves (both WR-compatible and incompatible with WR) should be agnostic of delay/phase measurement method used. Phase measurement algorithm must not not rely on slave capabilities (e.g. slave-side phase shifting) except for operation in synchronous mode.

Boundary Clocks as defined by the PTPv2 standard will not be used in WR networks.

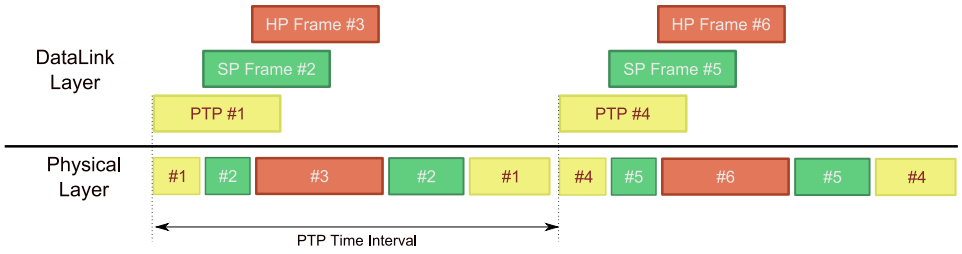


Figure 7.1: PTP Traffic

asdas

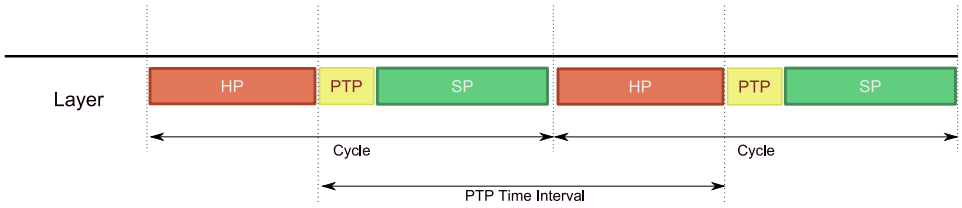


Figure 7.2: WR Traffic

White Rabbit will only use one-step clock between compliante ports. One-step clock, as define by the PTP standard, its a clock that that provides time information using a single event message.

7.1 Generation of ingress/egress event message time stamps

A timestamp event is generated at the time of transmission and reception of any event message. The timestamp event occurs when the message's timestamp point crosses the boundary between the node and the network, as indicated in Fig. 7.3.

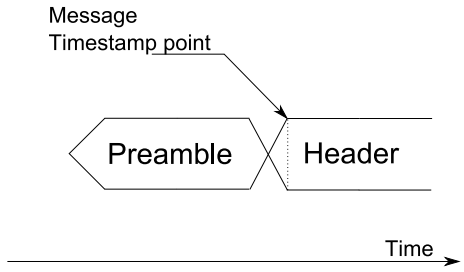


Figure 7.3: PTP message Timestamp generation

7.2 PTP Clock Variance

Variance estimates characterize boundary clock in a PTP system.

Each PTP clock shall maintain an estimate, the offsetScaledLogVariace, of its inherent precision. This is the precision of the timestamps included in messages issued by the clock when it is not synchronized to another clock using the protocol.

If such a clock, clock_A, is synchronized to another using the PTP protocol, it may maintain an estimate, the observedParentOffsetScaledLogVariance, of the precision of the clock to which it is synchornized as observed by clock_A.

7.2.1 Variance Algorithm

The PTP variance, from which offsetScaledLogVariance and observedParentOffsetScaledLogVariance are derived, is based on the theory of Allan deviation as follows.

The Allan deviation $\sigma_y(\tau)$ is estimated as follows:

$$\sigma_y(\tau) = \left[\frac{1}{2(N-2)\tau^2} \times \sum_{k=1}^{N-2} (x_{k+2} - 2x_{k+1} + x_k)^2 \right]^{\frac{1}{2}} \tag{7.1}$$

where x_k, x_{k+1} and x_{k+2} are time residual measurements made at times t_k , $t_k + \tau$, and $t_k + 2 \times \tau$. τ is the sample period between measurements and N is the number of data samples. The term residual implies that any consistent systematic effects have been removed from the data.

The Allan deviation as stated is a second-order statistic on the variation of the frequency of the oscillator used as the basis of the time base.

The PTP variance is defined by $\sigma_y^2 = \tau^2 \times \frac{1}{3}\sigma_y^2$. An unbiased estimate of the PTP variance shall be computed as follows:

$$\sigma_{PTP}^2 = \frac{1}{3}\left[\frac{1}{2(N-2)} \times \sum_{k=1}^{N-2}(x_{k+2} - 2x_{k+1} + x_k)^2\right] \tag{7.2}$$

where x_k , x_{k+1} , and x_{k+2} are time residual measurements, made at times t_k, $t_k + \tau$, and $t_k + 2\tau$, between the time provided by the measured clock and a local reference clock, and N is the number of data samples. For a PTP variance the quantityτ , the sample period, shall be the value defined in the applicable PTP profile. τ should be a multiple of the syncInterval.

Implementations may compute a conservative estimate of the PTP variance rather than computing the exact value given here. Note this may be necessary in implementations with limited computational or memory resources.

The dependence of the Allan deviation on the sample period provides information on the type of the underlying noise processes. The Allan deviation is not sensitive to constant offsets in time or in frequency, even though those offsets may be important in some applications of this standard. In addition, the Allan deviation does not provide a useful diagnostic when the noise spectrum contains "bright lines" — power-line induced variations at 60 Hz, for example. Finally, the Allan deviation is computed as an average over the ensemble of observations, and it is most useful when the data are statistically stationary. The deviation does not provide a good measure of the frequency or amplitude of occasional glitches, even though such events be important in some applications of this standard.

7.2.2 Variance Representation

PTP variances shall be represented as follows:

1. An estimate of the variance,$\sigma_P^2 TP$, is computed in units of seconds squared.

2. The logarithm to the base 2 of this estimate is computed. The computation of the logarithm need not be more precise than the precision of the estimate of the variance.

3. The logarithm is multiplied by 2^8 to produce a scaled value.

4. This scaled value is modified per the hysteresis specification of this subclause to produce the reported value.

5. The reported value is represented as a 2's complement Integer16, The value $8000_16$is added to the reported value represented in this form and any overflow is ignored. The result, i.e., the offset scaled report value, is cast as a UInteger16.

6. This offset scaled reported value, represented as Uinteger16, shall be the value if the log variances specified in 7.6.3.1.

For example, suppose the PTP variance value is $1.414 \times 2 - 73 = 1.497 \times 10 - 22s^2$. Therefore, $log_2(1.414 \times 2^-73) = 73 + 0.5 = -72.5$. If this were expressed as an Integer16, it would truncate to -72. To retain some precision the value is scaled by2^8 to yield a scaledLogVariance of -18560,$D780_16$, which retains 8 bits more precision. To this is added 8000_16 to yield the offset scaled reported value 5780_16.

The smallest variance that can be represented is $2^-128or\ 3 \times 10^-39s^2$, which results in an offsetScaledLogVariance of 0000_16. The maximum variance that can be represented is $2^+127.99609$, which results in an 39 offsetScaledLogVariance of $FFFF_16$.

This representation ensures that the ordering of variances algorithm of 7.6.3.4 produces identical results in all implementations. This cannot be guaranteed with a floating-point representation.

The largest possible positive number, $FFFF_16$, for the offsetScaledLogVariance attribute shall indicate that the variance is either too large to be represented, or has not been computed.

Since variance values are used in the selection of the best master clock, implementations in which variance values are computed during operation shall include hysteresis in the estimation of the variances to preclude thrashing in the process of selecting the master clock. The magnitude of this hysteresis applied to the 2^8 scaled values of the log_2 of the reported estimates shall be PTP_SCALED_LOG_VARIANCE_HYSTERESIS. This hysteresis shall be applied to the scaled values of the logarithm of the estimate used to generate the offset scaled values of the logarithm of the estimate

that is actually reported and used in the computations of the best master clock algorithm. Sufficient local state needs to be be maintained to allow correct implementations of the hysteresis properties for both increasing and decreasing trends in the actual variance estimate. The value of PTP_SCALED_LOG_VARIANCE_HYSTERESIS is , 2^7.

The computation of the value of defaultDS.offsetScaledLogVariance shall be based on the characteristics of the local clock as measured by a perfect clock. The value of defaultDS.offsetScaledLogVariance shall:

Be a static constant determined by the manufacturer, or

Be computed based on measured or modeled behavior of the components of the local clock and its environment.

The value of defaultDS.offsetScaledLogVariance shall be computed and represented as described above by the variance algorithm.

The value of defaultDS.offsetScaledLogVariance shall be an estimate of the variations of the local clock from a linear timescale when it is not synchronized to another clock using the protocol. The reference clock when not synchronized to another clock may be an atomic clock, a GPS receiver, a stable local oscillator, a suite of clocks synchronized via NTP, etc. These sources may contribute to the variance estimate.

The value of defaultDS.offsetScaledLogVariance shall be the variance applicable to an ensemble of measurements that include error contributions from:

Synchronization to its reference clock,

- Variation in clock phase change rate, and noise characteristics of the local oscillator
- Sampling quantization errors, fluctuations in inbound and outbound latency, and other fluctuations of the local clock.

7.3 PTPv2 Message Format

All PTP messages have a header, body and suffix. The suffix may have zero length.

7.3.1 Header

Header specifications

TransportSpecific The transportSpecific field may be used by a lower layer transport protocol and is defined by the mapping specification of that

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
transportSpecific				MessageType				1	1
reserved				versionPTP				1	2
messageLenght								2	
domainNumber								1	
reserved								1	
flagField								2	
correctionField								8	
reserved								4	
SourcePortIdentity								10	
sequenceId								2	
controlField								1	
logMessageInterval								1	33

Table 7.1: PTP Message Header

protocol in an annex of this standard

messageType The value messageType shall indicate the type of the message as defined in Table7.2.

The most significant bit of the messageType field divides this field in half between event and general messages.

versionPTP The value of the versionPTP field shall be the value of the portDS.versionNumber member of the data set of the originating node.

messageLength The value of the messageLength shall be the total number of bytes that form the PTP message.

domainNumber For ordinary or boundary clocks, the value of domainNumber shall be the value of the defaultDS.domainNumber member of the data set of the originating ordinary or boundary clock.

For peer delay mechanism messages originating from a peer-to-peer transparent clock, the value shall be the value defined in [XYS].

For management messages, the value shall be the value defined in [XYS].

Message Type	Message Class	Value
Sync	Event	0
Delay_Req	Event	1
Pdelay_Req	Event	2
Pdelay_Resp	Event	3
Reserved	-	4-7
Follow_up	General	8
Delay_Resp	General	9
Pdelay_Resp_Follow_Up	General	10
Announce	General	11
Signaling	General	12
Management	General	13
Reserved	-	14-15

Table 7.2: messageType field

flagField The value of the bits of the flagField array shall be as defined in Table 7.3. For message types where the bit 3 is not defined in Table 20, the values shall be FALSE.

correctionField The corretionField is the value of the correction measured in nanoseconds and multiplied by 2^16. For example, 2.5ns is represented as 0x0000 0000 0002 8000. The minimal correction to that can be represented in this field is so 152.587890625000 fs.

The value 0x8fff ffff ffff ffff indicates that the correction is too big to be represented.

The value of the correctionField depends on the message type as described in the next Table.

sourcePortIdentity The value of the sourcePortIdentity field shall be the value of the portDS.portIdentity member of the data set of the port that originated this message.

sequenceId The value of the sequenceId field shall be assigned by the originator of the message in conformance with 7.3.7 except in the case of Follow_Up, Delay_Resp, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages and management messages that are a response to another management message. The sequenceId field values for these exceptions are defined in the references listed in Table below

Byte	Bit	Message types	Name	Description
0	0	Announce, Sync, Follow_Up, Delay_Resp	alternateMasterFlag	FALSE if the port of the originator is in the MASTER state. Conditions to set the flag to TRUE are specified in subclauses 17.3 and 17.4.
0	1	Sync, Pdelay_Resp	twoStepFlag	For a one-step clock, the value of twoStepFlag shall be FALSE. For a two-step clock, the value of twoStepFlag shall be TRUE.
0	2	ALL	unicastFlag	TRUE, if the transport layer protocol address to which this message was sent is a unicast address. FALSE, if the transport layer protocol address to which this message was sent is a multicast address.
0	5	ALL	PTP profile Specific 1	As defined by an alternate PTP profile; otherwise FALSE
0	6	ALL	PTP profile Specific 2	As defined by an alternate PTP profile; otherwise FALSE
0	7	ALL	Reserved	
1	0	Announce	leap61	The value of timePropertiesDS.leap61
1	1	Announce	leap59	The value of of timePropertiesDS.leap59
1	2	Announce	currentUtcOffsetValid	The value of timePropertiesDS.currentUtcOffsetValid
1	3	Announce	ptpTimescale	The value of timePropertiesDS.ptpTimescale
1	4	Announce	timeTraceable	The value of timePropertiesDS.timeTraceable
1	5	Announce	frequencyTraceable	The value of timePropertiesDS.frequencyTraceable

Table 7.3: Values of flagField

Message Type	correctionField Description
Sync	Correction for fractional nanoseconds
Delay_Req	Correction for fractional nanoseconds
Pdelay_Req	Correction for fractional nanoseconds
Pdelay_Resp	Correction for fractional nanoseconds
Follow_up	Correction for fractional nanoseconds
Delay_Resp	Correction for fractional nanoseconds
Pdelay_Resp_Follow_Up	Correction for fractional nanoseconds
Announce	Zero
Signaling	Zero
Management	Zero

Table 7.4: correctionField semantics

ControlField The value of controlField depends on the message type defined in the messageType field, see 13.3.2.2, and shall have the value specified in Table 23. The use of this field by the receiver is deprecated.

logMessageInterval The value of the logMessageInterval field is determined by the type of the message and shall be as defined in Table 7.5.

7.3.2 Announce message

The field of Announce messages shall be as specified in Table 7.6.

originTimestamp The value of originTimestamp shall be 0 or an estimate no worse than +- 1 second of the local time if the originating clock when Announce message was transmitted.

currentUtcOffset The value of currentUtcOffset shall be the value of the timePropertiesDS.currentUtcOffset member of the 7 data set.

grandmasterPriority1 The value of grandmasterPriority1 shall be the value of the parentDS.grandmasterPriority1 member of the data set.

grandmasterClockQuality The value of grandmasterClockQuality shall be the value of the parentDS.grandmasterClockQuality member of the data set.

Message type	Value of logMessageInterval
Announce	The value of the portDS.logAnnounceInterval member of the data set
Sync, Follow_Up	The value of the portDS.logSyncInterval member of the data set in a multicast message, and 7F in a unicast message
Delay_Resp	The value of the portDS.logMinDelayReqInterval member of the data set in a multicast message, and 7F16 in a unicast message
Delay_Req	0x7F
Signaling	0x7F
Management	0x7F
Pdelay_Req	0x7F
Pdelay_Resq	0x7F
Pdelay_Resp_Follow_Up	0x7F

Table 7.5: logMessageInterval field

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
header (as described in 8.1.1)								34	0
originTimestamp								10	
currentUtcOffset								2	
reserved								1	
grandmasterPriority1								1	
grandmasterClockQuality								4	
grandmasterPriority2								1	
grandmasterIdentity								8	
stepsRemoved								2	
timeSource								1	63

Table 7.6: Announce message fields

grandmasterPriority2 The value of grandmasterPriority2 shall be the value of the parentDS.grandmasterPriority2 member of the data set.

grandmasterIdentity The value of grandmasterIdentity shall be the value of the parentDS.grandmasterIdentity member of the data set.

stepsRemoved The value of stepsRemoved shall be the value of currentDS.stepsRemoved of the data set of the clock issuing this message.

timeSource The value of timeSource shall be the value of the timePropertiesDS.timeSource member of the data set.

7.3.3 Sync and Delay_Req messages

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
header (as described in 8.1.1)								34	0
originTimestamp								10	34

Table 7.7: Sync and Delay_Req message fields

originTimestamp The value of originTimestamp shall be 0 as specified by ptp specification 9.5.9 and 11.3.

7.3.4 Follow_Up messages

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
header (as described in 8.1.1)								34	0
preciseoriginTimestamp								10	34

Table 7.8: Follow_Up message fields

preciseoriginTimestamp The value of preciseoriginTimestamp shall be 0 as specified by ptp specification 9.5.10 and 11.3.

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
header (as described in 8.1.1)								34	0
receiveTimestamp								10	34
requestingPortIdentity								10	44

Table 7.9: Delay_Resp message fields

7.3.5 Delay_Resp messages

7.3.6 Pdelay_Req messages

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
header (as described in 8.1.1)								34	0
originTimestamp								10	34
reserved								10	44

Table 7.10: Pdelay_Req message fields

7.3.7 Pdelay_Resp messages

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
header (as described in 8.1.1)								34	0
requestReceiptTimestamp								10	34
requestingPortIdentity								10	44

Table 7.11: Pdelay_Resp message fields

7.3.8 Pdelay_Resp_Follow_Up messages

7.3.9 Signaling message

Based on the indicated fields of a received signaling message, the message shall be accepted and applied as indicated by the PTP standard.

The common fields of a signaling message shall be as specified in Table 7.13.

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
header (as described in 8.1.1)								34	0
responseOriginTimestamp								10	34
requestingPortIdentity								10	44

Table 7.12: Pdelay_Resp_Follow_Up message fields

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
header (as described in 8.1.1)								34	0
targetPortIdentity								10	34
One or more TLV								M	44

Table 7.13: Signaling message fields

7.3.10 Management message

Management are used to access attributes and to generate certain events defined in this standard.

Common fields

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
header (as described in 8.1.1)								34	0
targetPortIdentity								10	34
startingBoundaryHops								1	
boundaryHops								1	
reserved				actionField				1	
reserved								1	
managementTLV								M	48

Table 7.14: Management message fields

targetPortIdentity(PortIdentity) The targetPortIdentity field shall be the portIdentity of the port or node on which the management message acts.

NOTE—The port identified by targetPortIdentity is not necessarily the port on which the management message was received.

In the case of a management message transmitted by a clock to a manager, the targetPortIdentity field shall be set to the sourcePortIdentity of the management message to which it is a response

startingBoundaryHops The value of the startingBoundaryHops field is implementation-dependent for messages that are not issued in response to a request from another management message. For management messages that are issued in response to a request from another management message, the value of startingBoundaryHops shall be the value computed from the startingBoundaryHops and boundaryHops fields of the requesting message as (startingBoundaryHops minus boundaryHops).

boundaryHops The value of the boundaryHops field indicates the remaining number of successive retransmissions of the management message by boundary clocks receiving the message per 15.3.3. The value of boundaryHops shall be identical to the value of the field startingBoundaryHops when first transmitted by the issuing clock.

actionField The value of the actionField shall indicate the action to be taken on receipt of the message as defined in the table below.

managementTLV Management messages shall be suffixed with zero or one TLV.

Management TLVs

Management TLV field format

tlvType The tlvType shall be MANAGEMENT.

lengthField The value of the lengthField is 2+N where N is an even number.

managementId The values of the managementId field are defined in PTPv2 Table 40 standard. TLV semantics for each managementId value shall be as defined in the following subclauses and in optional clauses of this standard.

The entries in the allowed actions column of PTPv2 Table 40 standard indicate the permissible values of the actionField field in the management

Action	Action taken	value (hex)
GET	The management message shall carry a single management TLV. The managementId field of the TLV indicates the specific information that needs to be retrieved. The current values of the data identified by the managementId shall be returned in a management TLV with the actionField value set to RESPONSE. If an error occurs, a management error status TLV shall be returned with the actionField value set to RESPONSE.	0
SET	The management message shall carry a single management TLV. The data in the TLV shall be used to update the current value of the data identified by the managementId field. Attempts to set a static or non-configurable value shall return a management error status TLV, see 15.5.4. If the update is successful, a management message with the actionField value set to RESPONSE shall be returned. If an error occurs, a management error status TLV shall be returned with the actionField value set to RESPONSE. If the data identified by the managementId consists of several fields, the update shall be considered as an atomic actionField and the failure to update any item shall be considered an error in the execution of the SET. TLVs with data definitions that mix configurable and non-configurable data are not permitted.	1
RESPONSE	The data in the TLV shall be the current values of the data identified by the managementId field of the management message with the GET or SET actionField. The value of the managementId shall be identical to that in the requesting message. If the actionField required by the GET or SET actionFields could not be fully executed, the response shall be a management error status TLV, see 15.5.4.	2
COMMAND	The event indicated by the managementId field shall be initiated. The results of this command shall be acknowledged by a management message with actionField set to ACKNOWLEDGE.	3
ACKNOWLEDGE	An acknowledge management message is a response to a command management message. The value of the managementId shall be identical to that in the command message. If the command could not be executed the acknowledge message shall be a management error status TLV.	4
reserved		5-F

Bits								Byte	Offset
7	6	5	4	3	2	1	0		
tlvType								2	0
lengthField								2	2
managementId								2	
dataField								N	6

Table 7.16: Management TLV fields

message common fields, see PTPv2 15.4.1.6. The receipt of a management with a disallowed actionField value shall:

Cause the contents of the management TLV to be disregarded

Return a management error status TLV, see 15.5.4, NOT_SUPPORTED.

5

dataField

Appendix A

LT Encoding

To maintain high reliability for non-handshake HP protocols for sensitive timing or control data, special encoding scheme, called LT coding is provided. Its idea is described below:

1. Sender splits original frame into n segments of size segment_size, varying from 32 to 384 bytes. These segments are named A1...An
2. Sender generates m¿n blocks X1...X2n by XORing randomly chosen segments from set A1...An. All blocks must be different. For example: X1 = A1 xor A3 xor A7 xor A8 X2 = A2 xor A4 xor A6 xor A8 Xm = ...
3. Sender transmits X1..Xm in burst of separate HP frames with LT_ENABLED flag set and proper LT coding header containing:
   (a) A 32 bit unique identifier of original frame (orig_frame_id) allowing to identify to which frame each block belongs)
   (b) 32-bit integer with equation coefficients (eqn). For example, value of 100101 means that Xi = A1 xor A4 xor A6.
   (c) Segment size in bytes (segment_size)
   (d) Index of segment (segment_num)
   (e) MD5 checksum of payload and LT header (md5sum)
4. Assuming that at least N blocks reached the receiver without errors, receiver can solve the equation system from p. 2. and reconstruct the original frame.

Reliability of this encoding system depends on:

- effectiveness of used checksum algorithm (MD5 should be more than sufficient)
- bit error rate of link. Assuming that BER will be the same as observed in test GbE link (very faulty { about 1 broken packet per second) this scheme will work perfectly well.

Appendix B

Time-scales and epochs

B.1 TAI and UTC

B.2 NTP and GPS

Two standard time sources of particular interest in implementing PTP systems: NTP and GPS. Both NTP and GPS systems are expected to provide time references for calibration of the grand-master supplied PTP time.

NTP represents seconds as a 32 bit unsigned integer that rolls-over every 232 seconds $\approx$ 136 years, with the first such rollover occurring in the year 2036. The precision of NTP systems is usually in the millisecond range.

NTP is a widely used protocol for synchronizing computer systems. NTP is based on sets of servers, to which NTP clients synchronize. These servers themselves are synchronized to time servers that are traceable to international standards.

NTP provides the current time. In NTP version 4, the current leapSeconds value and warning flags marking indicating when a leapSecond will be inserted at the end of the current UTC day. The NTP clock effectively stops for one second when the leap second is inserted.

GPS time comes from a global positioning satellite system, GPS, maintained by the U.S. Department of Defense. The precision of GPS system is usually in the 10-100 ns range. GPS system transmissions represent the time as {weeks, secondsInWeek}, the number of weeks since the GPS epoch and the number of seconds since the beginning of the current week.

GPS also provides the current leapSeconds value, and warning flags marking the introduction of a leap second correction. UTC and TAI times can be computed solely based the information contained in the GPS trans-

missions.

GPS timing receivers generally manage the epoch transitions (1024-week rollovers), providing the correct time (YYYY-MM-DD hh:mm:ss) in TAI and/or UTC time scales, and often also local time; in addition to providing the raw GPS week, second of week, and leap second information.

Appendix C

Time-of-day format considerations

C.1 IEEE 1588 timer format

IEEE Std 1588-2002 timer format consists of seconds and nanoseconds fields components, as illustrated in figure below. The nanoseconds field must be less than 10e9; a distinct sign bit indicates whether the time represents before or after the epoch duration. Figure 4 . IEEE 1588 timer format

```
struct Timestamp {
    UInteger48 seconds;
    UInteger32 nanoseconds;
    };
Extended Timestamp
struct ExtendedTimestamp {
    UInteger48 seconds;
    UInteger48 fractionalNanoseconds;
    };
Time of day
```

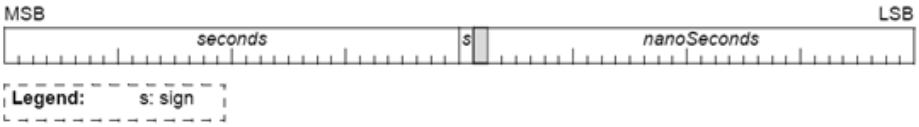


Figure C.1: IEEE 1588 timer format

Appendix D

Simulation results

Simulations
    Congestion modelling

Appendix E

C-code Illustrations

Bibliography

[IEE04] Ieee standard for local and metropolitan area networks media access control (mac) bridges. *IEEE Std 802.1D-2004 (Revision of IEEE Std 802.1D-1998)*, pages 1–269, 2004.

[IEE06] Ieee standard for local and metropolitan area networks virtual bridged local area networks. *IEEE Std 802.1Q-2005 (Incorporates IEEE Std 802.1Q1998, IEEE Std 802.1u-2001, IEEE Std 802.1v-2001, and IEEE Std 802.1s-2002)*, pages 1–285, 2006.

[PTP08] Ieee standard for a precision clock synchronization protocol for networked measurement and control systems. *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, pages c1–269, 24 2008.