Table of Contents

Table of Figures

1. Introduction

1.1. Purpose

This document describes the implementation of the IEEE 1588-2008 (PTP V2) protocol which is a revision of the Precision Time Protocol (PTP V1) as defined by the IEEE 1588-2002 standard. This implementation will utilize an existing open source module called *"Ptpd*". A specific profile of 1588 V2 will be used, according to the 802.1AS requirements., Note that this document do not substitute for reading the IEEE 1588-2008 standard because it dictates most operations of this implementation.

1.2. Abbreviations

Use "Ctrl + Click" to go to definition when letter bookmark are present (ex. [1]).

- **AVB**: Audio Video Bridging
- **BMCA**: Best Master Clock Algorithm
- **E2E TC**: End to End Transparent Clock
- **P2P TC**: Peer to Peer Transparent Clock
- **PTP**: Precision Time Protocol
- **TLV**: Type, Length, Value (according to 802.1AVB)

2. Precision Time Protocol

The “Precision Time Protocol” (PTP), defines a method to precisely synchronize the system clock of distributed nodes in a system using standardized packet based networks like Ethernet. The Precision Time Protocol allows all nodes to synchronize system-wide in the sub-microsecond range.

2.1. Main differences between V1 & V2

Version 2 of the IEEE 1588 standard is a consistent improvement of version 1 to enhance the usability and precision for large networks. To achieve these goals, V2 defines shorter synchronization frames to save network bandwidth, transparent clocks to avoid exponential error propagation in cascaded networks and many other new features, like message extensions (using TLV).

Version 2 is much more flexible regarding the configuration of the protocol by means of configuration sets used by specific devices. These sets, known as “profiles”, simplify the configuration of the nodes and guarantee the proper behavior and performance for specific systems. The implementation presented in this document will mostly use a specific profile define in the IEEE 802.1AS standard.

The state machine of the protocol engine is described on the figure below. It is quite similar to the 1588 V1 one.

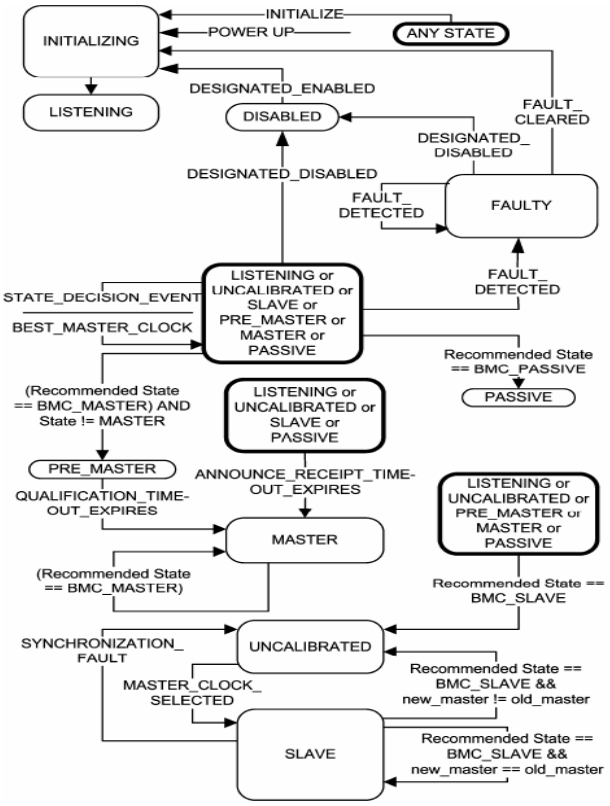


Figure 1 : State machine for a full implementation

2.1.1. Messages

There are five kinds of messages defined for PTPv1, and ten kinds in PTPv2.

	PTP V1	**PTP V2**
Event messages	-Sync -Delay_Req -Follow_Up -Delay_Resp -Management	-Sync -Delay_Req -Pdelay_Req -Announce -Follow_Up -Delay_Resp
General messages		-PDelay_Follow_Up -PDelay_Resp -Signaling -Management messages

The Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages are used to measure the link delay between two clock ports implementing the “peer delay mechanism”. It will be detailed in $ 2.1.2. PTP V2 split “timing information”, and “master-slave hierarchy” determination. In fact in V1, best master selection information was sent in a Sync message, which length was 124 bytes. On the contrary in V2, this is sent separately in an Announce message. It enables a much smaller Sync message (44 bytes), with higher Sync message rates using less bandwidth. Sync and Announce messages are sent periodically by the Master but Sync messages with a much more frequent rate. To give an idea, in the profile defined by 802.1 AS, the interval between two Announce messages is 1s, and the one between two Sync messages is nearly 8ms.

2.1.1.1. Management messages

The management messages are used to query and update the PTP data sets maintained by clocks. These messages are also used to customize a PTP system, to generate certain events and for initialization and fault management. The format of this kind of message is given below.

Management 1588 V2									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								34	0
targetPortIdentity								10	34
startingBoundaryHops								1	44
boundaryHops								1	45
reserved				actionField				1	46
reserved								1	47
managementTLV								M	48

2.1.1.2. Signaling messages

A signaling message is used to transport a sequence of one or more TLV entities (TLV mechanism will be described in $2.1.4). It is issued when it's required by a TLV on a received signaling message, or required by an optional or mandatory feature of the standard. It could also be required by implementation specific considerations outside the scope of the standard.

2.1.1.3. Messages format

2.1.1.3.1.Header

Header 1588 V1									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
versionPTP								2	0
versionNetwork								2	2
subdomain								16	4
messageType								1	20
sourceCommunicationTechnology								1	21
sourceUUID								6	22
sourcePortId								2	28
sequenceId								2	30
control								1	32
reserved								1	33
flags								2	34

Header 1588 V2									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
transportSpecific				messageType				1	0
reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
reserved								1	5
flagField								2	6
correctionField								8	8
reserved								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField								1	32
logMessageInterval								1	33

2.1.1.3.2.Sync & Delay_req

Sync & Delay_req 1588 V1									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								36	0
reserved								4	36
originTimestamp (seconds)								4	40
originTimestamp (nanoseconds)								4	44
epochNumber								2	48
currentUTCoffset								2	50
grandMasterCommunicationTechnology								2	52
grandMasterClockUuid								6	54
grandmasterPortId								2	60
grandmasterSequenceId								2	62
grandMasterClockStratum								4	64
grandMasterClockIdentifier								4	68
grandMasterClockVariance								4	72
grandMasterPreferred								2	76
grandMasterIsBoundaryClock								2	78
syncInterval								4	80
localClockVariance								4	84
localStepsRemoved								4	88
localClockStratum								4	92
localClockIdentifier								4	96
parentCommunicationTechnology								2	100
parenttUuid								6	102
parentPortField								4	108
estimatedMasterVariance								4	112
estimatedMasterDrift								4	116
utcReasonnable								4	120

Sync & Delay_req 1588 V2									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								34	0
originTimestamp								10	34

2.1.1.3.3.Follow_up (two-step clocks)

Follow_up 1588 V1										
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								36	0
reserved								4	36
associatedSequenceId								4	40
preciseOriginTimestamp (seconds)								4	44
preciseOriginTimestamp (nanoseconds)								4	48

Follow_up 1588 V2									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								34	0
originTimestamp								10	34

2.1.1.3.4.Delay_resp

Delay_resp 1588 V1									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								36	0
reserved								4	36
associatedSequenceId								4	40
delayReceiptTimestamp (seconds)								4	44
delayReceiptTimestamp (nanoseconds)								4	48
requestingSourceCommunicationTechnology								2	52
requestingSourceUuid								6	54
requestingSourcePortId								2	60
requestingSourceSequenceId								2	62

Delay_resp 1588 V2									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								34	0
receiveTimestamp								10	34
requestingPortIdentity								10	44

2.1.1.3.5.Pdelay messages (exclusively 1588 V2)

Pdelay_Req 1588 V2									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								34	0
originTimestamp								10	34
reserved								10	44

Pdelay_Resp 1588 V2									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								34	0
requestReceiptTimestamp								10	34
requestingPortIdentity								10	44

- **Two-step clock**

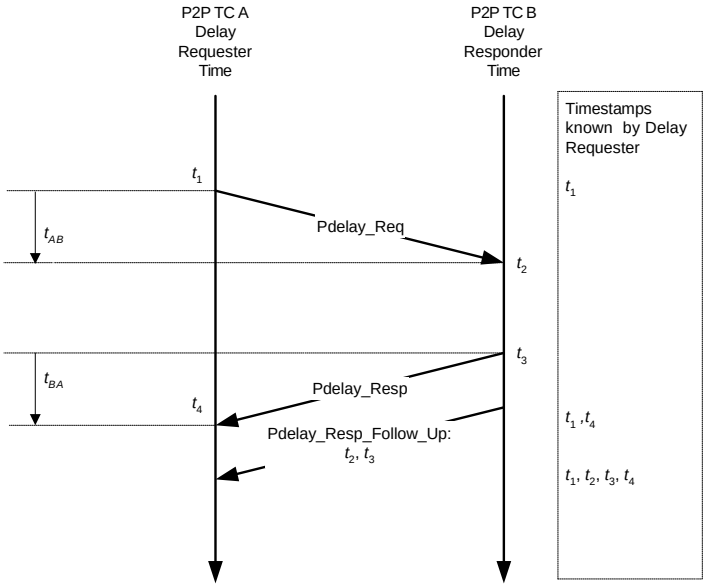
Pdelay_Resp_Follow_Up									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								34	0
responseOriginTimestamp								10	34
requestingPortIdentity								10	44

2.1.1.3.6.Announce messages (exclusively 1588 V2)

Announce 1588 V2									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
Header								34	0
originTimestamp								10	34
currentUTCoffset								2	44
reserved								1	46
grandmasterPriority1								1	47
grandmasterClockQuality								4	48
grandmasterPriority2								1	52
grandmasterIdentity								8	53
stepsRemoved								2	61
timeSource								1	63

2.1.2. Delay computing

In addition to the “Delay request-response mechanism” present in V1, PTP V2 introduces the “Peer delay mechanism”. The peer delay mechanism measures the port-to-port propagation time, i.e., the link delay, between two communicating ports supporting the peer delay mechanism. It uses the messages Pdelay_Req, Pdelay_Resp, and possibly Pdelay_Resp_Follow_Up, as shown in the timing diagram of Figure below. The Pdelay_Resp_Follow_Up is only sent if the time-aware system is a “two-step” clock, i.e. a Follow_Up message corresponding to each Sync message and a Pdelay_Resp_Follow_Up message corresponding to each Pdelay_Resp message are sent.



$$t_{AB} = t_2 - t_1$$

$$t_{BA} = t_4 - t_3$$

$$t_{mean-prop} = \frac{t_{AB} + t_{BA}}{2}$$

$t_{mean-prop}$ is the propagation time making the assumption that the propagation time in the two directions are the same.

Figure 2 : The peer delay mechanism

The peer delay mechanism is limited to point to-point links between two ordinary clocks, boundary clocks, and/or peer-to-peer transparent clocks and so the peer delay messages are not forwarded, contrary to the “Delay request-response” ones.

2.1.3. Clock types

In PTP V1, the only clock types were “ordinary clock” and “boundary clock”. Boundary Clocks were defined to support IEEE 1588 synchronization within networks containing several subnets. Boundary clocks typically have more than two ports, with one port serving as a PTP slave to an upstream clock master, and the other ports serving as PTP clock masters to downstream PTP slaves.

Due to the cumulative effect of multiple servo loops, cascading multiple Boundary Clocks within a network can significantly degrade clock accuracy of the system. In order to avoid this, a new clock type called “Transparent Clocks” (TC) appears in PTP V2. A transparent clock is a device that measures the time taken for a PTP event message to transit the device and provides this information to clocks receiving this PTP event message. Thus, each transparent clock appears to be a "wire" which does not skew the time calculation for packets passing through them. We can note the P2P TCs are not part of the Master-Slave hierarchy.

2.1.3.1. End To End Transparent Clock

An end-to-end transparent clock (E2E TC) only measures the time taken for a PTP event message to transit the device and provides this information to clocks receiving this PTP event message; i.e., it does not provide corrections for the propagation delay of the link connected to the port receiving the PTP event message. It does not use the peer delay measurement mechanism but, instead, supports use of the delay request-response mechanism. [1]

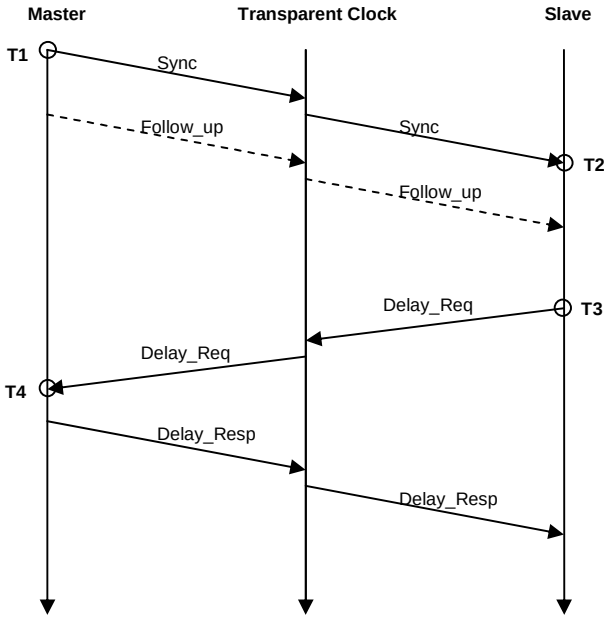


Figure 4 : Delay request-response path length measurement

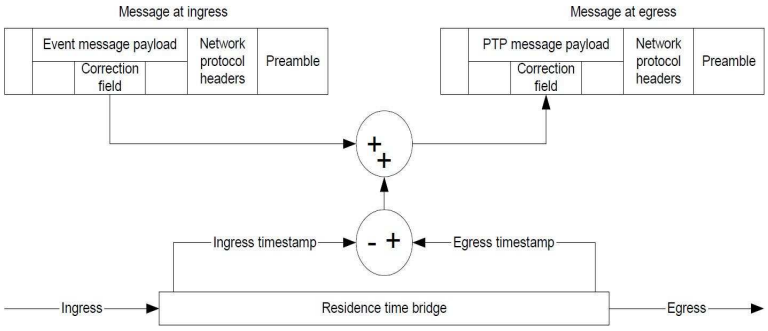


Figure 3 : End to End residence time correction model

2.1.3.2. Peer To Peer Transparent Clock

A peer-to-peer transparent clock (P2P TC) is a transparent clock that, in addition to providing PTP event transit time information, also provides corrections for the propagation delay of the link connected to the port receiving the PTP event message. In the presence of peer-to-peer transparent clocks, delay measurements between slave clocks and the master clock are performed using the peer delay measurement mechanism. [1]

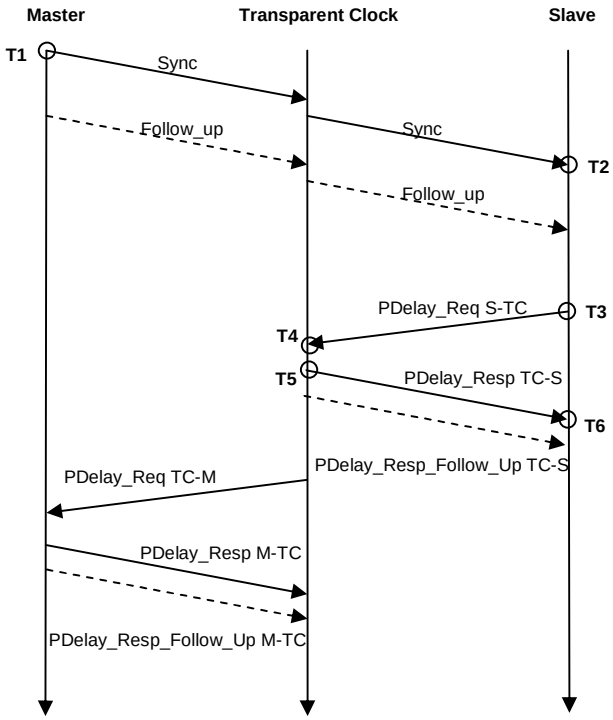


Figure 6 : Peer delay link measurement

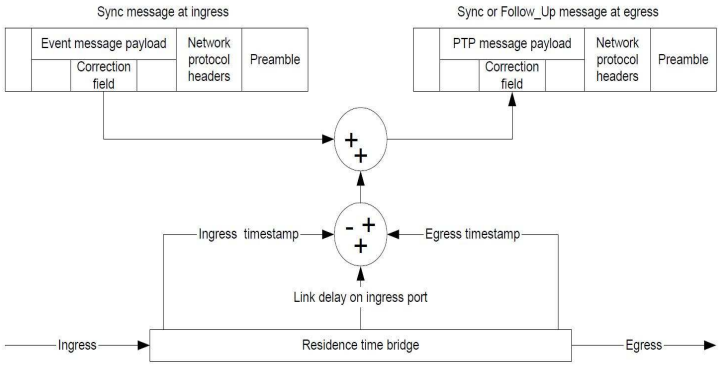


Figure 5 : Peer to Peer residence time and link delay correction model

2.1.4. Extension mechanism (TLV)

TLVs (Type, Length, Value) are used like extensions to the standard, in order to transmit additional datas. Their type is the structure below:

```
Struct TLV {
Enumeration16 tlvType;
Uinteger16 lengthField;
Octet[lengthfield] valueField
}
```

For example a management message can contain a TLV, to update the current value of data identified by the managementiD. The new value is included in the dataField and in this case tlvType is equal to “MANAGEMENT”

Management TLV fields									
Bits								Octets	Offset
7	6	5	4	3	2	1	0		
tlvType								2	0
lengthField								2	2
managementId								2	4
dataField								N	6

2.2. PTP V2 & 801.AS

802.1AS is one of three 802.1 AVB draft standards. It includes an IEEE 802-specific layer 2 profile of IEEE Std 1588TM–2008 (PTP V2), plus additional requirements needed to ensure performance for both full-duplex 802.3 (Ethernet) and 802.11 (WiFi) networks. It specifies:

- The propagation delays between neighboring bridges and/or end stations are measured using the peer delay mechanism.
- The clock types used are only Ordinary Clock (OC) and Peer to Peer Transparent Clock (P2P TC)
- All time-aware systems are two-step clocks.
- An 802.1AS network consists of a single PTP domain, with domain number 0.
- The timescale is the PTP timescale.
- While syntonization[1] of the P2P TC to the GM is not mandatory in IEEE 1588, it will be mandatory for AVB networks.
- Alternate Best Master Clock Algorithm (BMCA), though very similar to default which is simplified to provide faster convergence.

[1] **Syntonized clocks** : Two clocks are syntonized if the duration of the second is the same on both which means the time as measured by each advances at the same rate. They may or may not share the same epoch.

3. Implementation

The implementation of the IEEE 1588-2008 (PTP V2) protocol will largely utilize an existing software-only implementation “Ptpd” initially designed for PTP V1. [2]

3.1. PTPd v1

The PTP daemon (PTPd) implements the Precision Time protocol (PTP) as defined by the IEEE 1588 - 2002 standard. PTPd's source is grouped into a few components. The component delineations are based on the functionality defined by the spec. The following is a block diagram of PTPd's major components, in which occlusion indicates interfaces between components. [2]

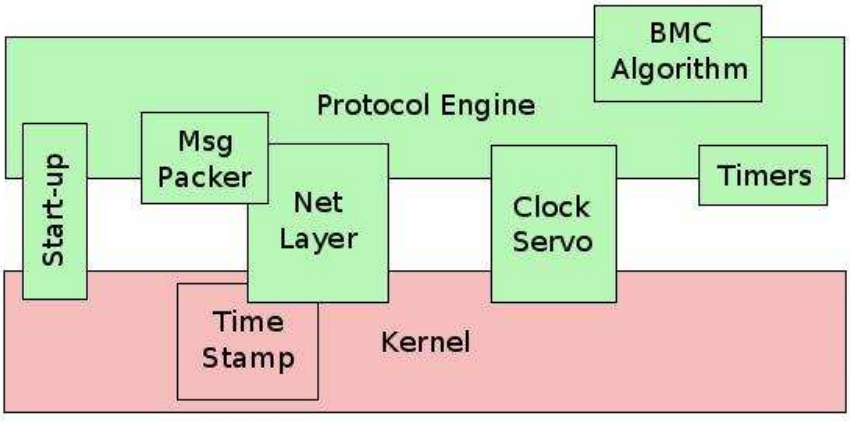


Figure 7 : Ptpd’s source code organization

3.2. PTPd v2

The PTPd v2 will implement the Precision Time protocol (PTP V2) as defined by the profile 802.1AS (P2P TC), and only for an ordinary clock. The whole structure and some existing blocks of Ptpd v1 will be reused, as the *clock servo*, *timers*, *start-up* and *timestamp* blocks.

Main changes appear in:

- Protocol Engine with use of P2P TC and peer delay mechanism.
- MsgPacker, taking into account new kinds of messages.
- Net layer to add a Raw Ethernet mapping, to the existing UDP/IP (V4) one.
- BMC algorithm will be the one defined in the IEEE 1588-2008 standard.

Moreover, some primitive PTP data types, added in PTP V2 like UInteger48, or Integer64 will be implemented as structures. The implementation will redefine arithmetic operations on this kind of data type.

- **Uinteger48**

```
typedef struct {

      unsigned int LSB;
      unsigned short MSB;

}Uinteger48;
```

- **Integer64**

```
typedef struct {

      int LSB;
      int MSB;

}Integer64;
```

4. Appendices

4.1. References

[1] IEEE 1588TM/2008, Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems

[2] PTPd Source code documentation

[3] IEEE P802.1AS/D5.0 Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks

[4] Description of use of IEEE 1588 Follow up P2P TC in AVB, Geoffroy M. Garner, March 03 2006

[5] IEEE 1588 Version 2 Summary, Geoffroy M. Garner, September 24 2008

[6] IEEE 802.1AS Tutorial, Kevin B.Stantton, November 13 2008

[7] IEEE 1588 Version 2 Tutorial, John Eidson, October 02 2006

[8] Using IEEE 1588 for synchronization of network-connected devices, Alexandra Dopplinger and Jim Innis , March 29 2007