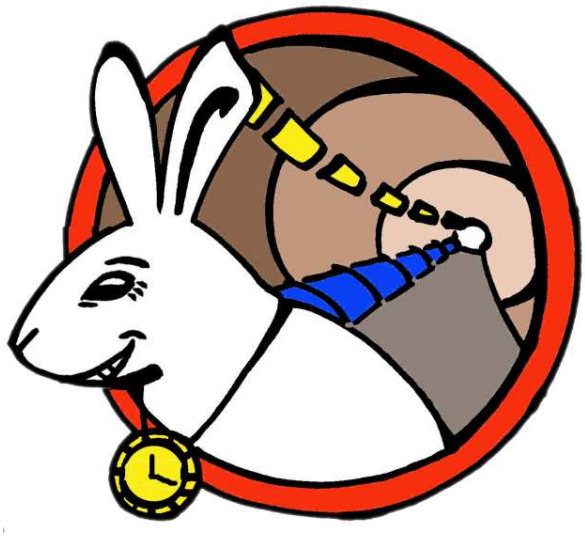
# White Rabbit Specification: Draft for Comments

Emilio G. Cota
Maciej Lipinski
Tomasz Wlostowski

September 2010



Revision History Table

| Version | Date | Authors | Description |
|---|---|---|---|
| 0.1 | 10/09/2010 | E.G., M.L. | First draft for comments. |
| 0.2 | 7/09/2010 | T.W., M.L. | Added introduction about PTP. |
| 0.3 | 8/09/2010 | J.S., M.L. | Language&Style-related corrections, reference to Peter's paper. |
| 0.4 | 14/09/2010 | E.V.D.B, M.L. | 1. Merged the FSM of the Slave and the FSM of the Master into single and linear FSM,<br>2. Changed WR managementIDs,<br>3. Added authors, this rev. table, explanatory figures and descriptions. |

# 1 Introduction

White Rabbit (WR) is a protocol developed to synchronize nodes in a packet-based network with sub-ns accuracy. The protocol results from the combination of IEEE1588-2008 (PTP)[1] with two further requirements: precise knowledge of the link delay and clock syntonization[1] over the physical layer.

A WR link is formed by a pair of nodes, master and slave (Figure 1). The master node uses a traceable clock to encode data over the physical layer, while the slave recovers this clock (syntonization) and bases its timekeeping on it. Absolute time synchronisation between master and slave is achieved by adjusting the clock phase and offset of the slave to that of the master. The *offset* refers to the clock (e.g. time defined in Coordinated Universal Time standard), while the *phase* refers to the clock signal (e.g. 125 MHz clock signal). The phase and offset adjustment is done through the two-way exchange of PTP sync messages, which are corrected to achieve sub-ns accuracy due to the precise knowledge of the link delay.

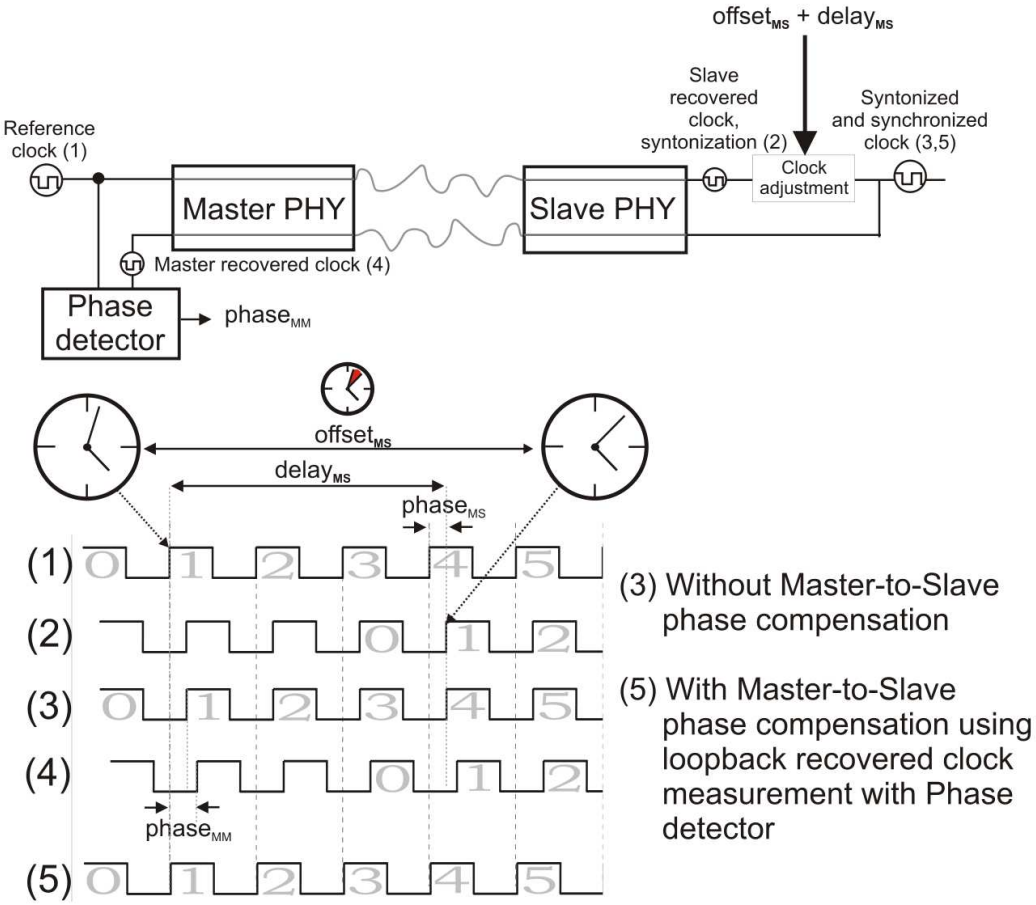



Figure 1: Synchronization and syntonization in a White Rabbit link.

[1]The adjustment of two electronic circuits or devices in term of frequency.

The precise knowledge of the link delay is obtained by accurate hardware timestamps and calculation of the delay asymmetry (see section 4).

The described single-link synchronization can be replicated. Multi-link WR networks are obtained by chaining WR links forming a hierarchical topology. This hierarchy is imposed by the fact that a frequency traceable to a common grandmaster must be distributed over the physical layer, resulting in a *cascade* of master and slave nodes. As a result of this topology, WR network consists of two kinds of WR network devices: *WR boundary clocks* (WR Switches) and *WR ordinary clocks* (WR Timing Receivers), see section 6.1. It should be noted that the problem of non-linear error accumulation of chained boundary clocks does not apply, or is at least greatly diminished by the clock recovery mechanism.

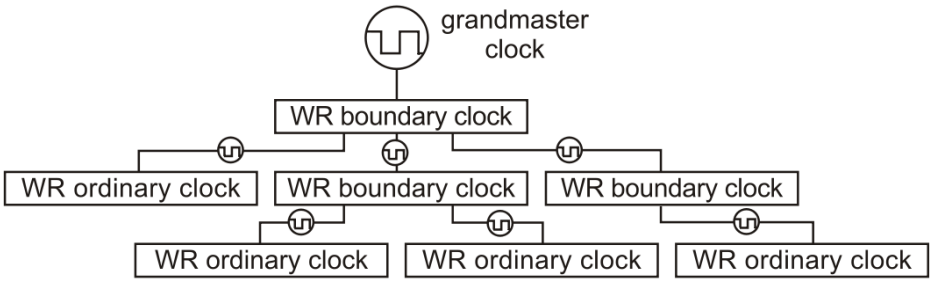



Figure 2: White Rabbit network; it forms a hierarchical topology.

Some applications need WR and IEEE1588-2008 nodes to coexist. Examples of this are existing IEEE1588 installations which are to be migrated progressively to White Rabbit and networks where the need for highly accurate time synchronisation is concentrated on a certain group of nodes. For this purpose the WR protocol enables WR nodes to defer to IEEE1588 behaviour when not connected to another WR node.

# 2 Precision Time Protocol

The IEEE1588-2008 standard, known as Precise Time Protocol (PTP), is repeatedly referenced in this document. Knowledge of basic PTP concepts is required to read this specification, therefore they are explained in this section.

PTP is a packet-based protocol designed to synchronize devices in distributed systems. The standard defines two kinds of messages which are exchanged between *PTP nodes*: *event messages* and *general messages*. Both, the time of transmission and the time of reception of event messages are *timestamped*. General messages are used by PTP nodes to identify other PTP nodes, establish clock hierarchy and exchange data, e.g. timestamps, settings or parameters. PTP defines several methods for node' synchronization. Figure 3 presents the messages used when the *delay request-response mechanism* (with *two-step clock*) is used, which is the case in White Rabbit. For simplicity reasons, a PTP node is considered an *ordinary clock* in the remainder of this section; such clocks have only one port.

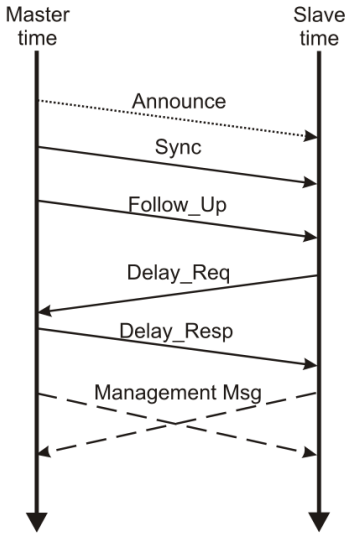



Figure 3: PTP messages used by WRPTP.

An *Announce Message* is periodically broadcast by the PTP node which is in the Master state. The message carries information about its originator and the originator's clock source quality. This enables other PTP nodes receiving the announce message to perform the Best Master Clock (BMC) Algorithm. The algorithm defines the role of each PTP node in PTP network hierarchy; the outcome of the algorithm is the recommended next state of the PTP node and the node's synchronization source (grandparent). In other words, a PTP node decides to which other PTP node it should synchronize based on the information provided in announce messages and using the BMC algorithm. A PTP node which is in the SLAVE state synchronizes to the clock of another PTP node. A PTP node which is in the MASTER state is regarded as a source of synchronization for the other PTP nodes. The full PTP state machine with state descriptions is included in Appendix B.

*Sync Messages* and *Delay_Req Messages* are timestamped ($t_1$, $t_2$, $t_3$, $t_4$) and these timestamps are used to calculate the offset and the delay between the nodes exchanging the messages. *Follow_UP Messages* and *Delay_Resp Messages* are used to send timestamps between

Master and Slave (in the case of a two-step clock).

*Management Messages* are used only for configuration and administrative purposes. They are not essential for the PTP synchronization.

The flow of events in the PTP delay request-respons (two-step clock) mechanism is the following (simplified overview):

1. The master sends periodically Announce messages.
2. The slave receives the Announce message, uses the BMC algorithm to establish its place in the network hierarchy.
3. The master sends periodically a Sync message (timestamped on transmission, $t_1$) followed by a Follow_UP message which carries $t_1$.
4. The slave receives the Sync message sent by the master (timestamped on reception, $t_2$).
5. The slave receives the Follow_Up message sent by the master.
6. The slave sends a Delay_Req message (timestamped on transmission, $t_3$).
7. The master receives the Delay_Req message sent by the master (timestamped on reception, $t_4$).
8. The master sends the Delay_Resp message which carries $t_4$.
9. The slave receives the Delay_Resp.
10. The slave adjusts its clock using the offset and the delay calculated with timestamps ($t_1$, $t_2$, $t_3$, $t_4$). It results in the Slave's synchronization with the Master clock.
11. Repeat 1-10.

# 3 Link Delay Model

The delay of a message travelling from master to slave (see Figure 4) can be expressed as the sum

$$\text{delay}_{ms} = \Delta_{tx_m} + \delta_{ms} + \Delta_{rx_s} \tag{1}$$

where $\Delta_{tx_m}$ is the fixed delay due to the master's transmission circuitry, $\delta_{ms}$ is the variable delay incurred in the transmission medium, and $\Delta_{rx_s}$ is the fixed delay due to the slave's reception circuitry. In a similar fashion, the delay of a message travelling from slave to master can be decomposed as

$$\text{delay}_{sm} = \Delta_{tx_s} + \delta_{sm} + \Delta_{rx_m} \tag{2}$$

The characterization of the link is completed with an equation to relate the two variable delays $\delta_{ms}$ and $\delta_{sm}$. From now on in this document we refer to this missing equation as the *physical medium correlation*. Describing a procedure to obtain this equation is out of the scope of this document. However, section 3.1 provides correlations obtained empirically for one scenario.

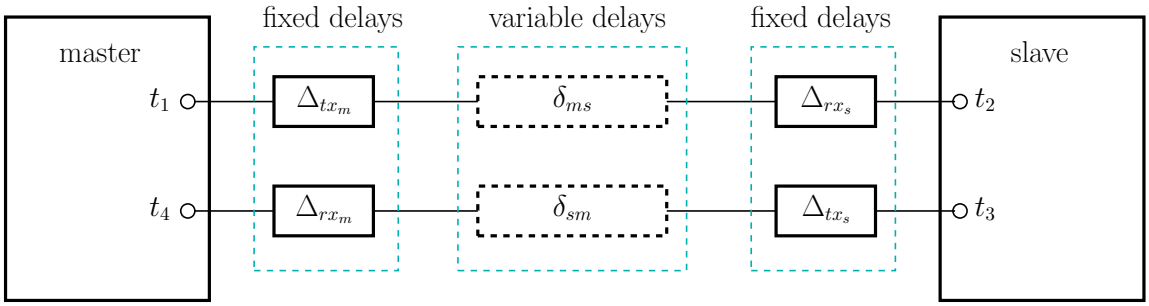



Figure 4: Delay model of a WR link. The timestamps are accurately corrected for link asymmetries by the usage of the four fixed delays $\Delta_{\{tx_m,rx_s,tx_s,rx_m\}}$ and the relationship between both variable delays $\delta_{\{ms,sm\}}$.

## 3.1 Physical Medium Correlation

An accurate correlation between both variable delays on the transmission line is essential for obtaining an acceptable estimate of the delay asymmetry on a WR link. The origin of this correlation is highly implementation-dependent. Thus this document just assumes that such correlation exists and is known. The correlation between $\delta_{ms}$ and $\delta_{sm}$ is represented in this document by the *medium correlation parameter* ($\alpha$).

### 3.1.1 Ethernet over a Single-mode Optical Fibre

When a single-mode fibre is used as bi-directional communication medium, it can be shown that both variable delays are related by an equation of the form [2]:

$$\delta_{ms} = (1+\alpha)\,\delta_{sm} \tag{3}$$

# 4 Delay Asymmetry Calculation

Let us start from the PTP sync timestamps, represented by the familiar set $t_1$, $t_2$, $t_3$ and $t_4$. The mean path delay is then defined as

$$\mu = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} \tag{4}$$

Note that the transmission delays $t_2 - t_1$ and $t_4 - t_3$ can be expressed in terms of WR's Delay Model:

$$t_2 - t_1 = \Delta_{tx_m} + \delta_{ms} + \Delta_{rx_s} + \text{offset}_{ms} \tag{5}$$
$$t_4 - t_3 = \Delta_{tx_s} + \delta_{sm} + \Delta_{rx_m} - \text{offset}_{ms} \tag{6}$$

where $\text{offset}_{ms}$ is the time offset between the slave's clock and the master's. Combining the three equations above we obtain

$$2\mu = \Delta + \delta_{sm} + \delta_{ms} \tag{7}$$

where $\Delta$ accounts for all fixed delays in the path, i.e.

$$\Delta = \Delta_{tx_m} + \Delta_{rx_s} + \Delta_{tx_s} + \Delta_{rx_m} \tag{8}$$

The delay asymmetry as specified in section 7.4.2 of IEEE1588-2008 is expressed in our own notation by using equations (1), (2) and (7) as follows:

$$\text{delay}_{ms} = \mu + \text{asymmetry} \tag{9}$$
$$\text{delay}_{sm} = \mu - \text{asymmetry} \tag{10}$$

The delay asymmetry cannot be calculated unless we use the physical medium correlation.

## 4.1 Solution for Ethernet over a Single-mode Optical Fiber

Combining equations (3) and (7) we obtain:

$$\delta_{ms} = \frac{1+\alpha}{2+\alpha}(2\mu - \Delta) \tag{11}$$
$$\delta_{sm} = \frac{2\mu - \Delta}{2+\alpha} \tag{12}$$

The delay asymmetry can then be derived from equations (1), (9), (11) and (12):

$$\text{asymmetry} = \Delta_{tx_m} + \Delta_{rx_s} - \frac{\Delta - \alpha\mu + \alpha\Delta}{2+\alpha} \tag{13}$$

It can be noted that if $\Delta << \mu$, the above equation can be simplified:

$$\text{asymmetry} = \Delta_{tx_m} + \Delta_{rx_s} - \frac{\Delta - \alpha\mu}{2+\alpha} \tag{14}$$

.

# 5 Fixed delays

The knowledge of fixed delays $\Delta_{\{tx_m,rx_s,tx_s,rx_m\}}$ is necessary to calculate delay asymmetry (13). Such delays may be constant for the lifetime of the hardware, its up-time or the duration of the link connection. Therefore, the method for obtaining fixed delays is medium-specific and implementation-dependent. The delays are measured (if necessary) and the information about its values is distributed across the link during the process of establishing the WR link, which is called *WR Link Setup* in this document. A WR node participates in the measurement of another WR node's reception fixed delay ($\Delta_{\{rx_m,rx_s\}}$) upon request, e.g. by sending a calibration pattern in Gigabit Ethernet. Measurement of fixed delays during WR Link Setup is optional. It is only required if a non-deterministic reception/transmission elements are used.

An example implementation of the method to obtain fixed delays for non-deterministic Gigabit Ethernet PHY is described in Appendix A.

# 6 White Rabbit PTP Extension

## 6.1 Overview

White Rabbit extends the IEEE1588-2008 (PTP) standard to achieve sub-ns accuracy while still benefiting from PTP's synchronization and management mechanisms. From now on in this document, the White Rabbit extension to the PTP standard will be referred to as *WRPTP*. WRPTP introduces *the White Rabbit Link Setup* (WR Link Setup), which is a process for establishing the WR link (Figure 5). It includes syntonization of the local clock over the physical layer, measurement of fixed delays (calibration) and distribution of the information about fixed delays over the link. The WR extension takes advantage of the Link Delay Model to obtain an accurate delay estimation, e.g. it uses the delay asymmetry equation (13) for Gigabit Ethernet over Fiber Optic. WRPTP extends the PTP messages and Data Sets (DS).

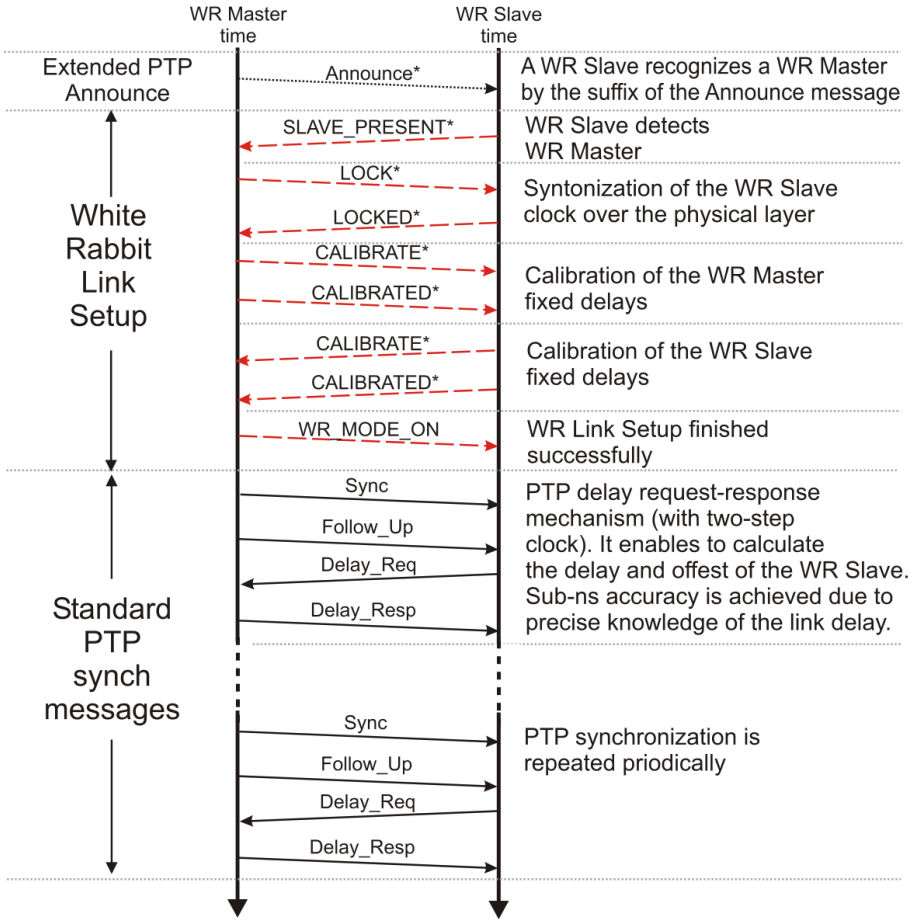



Figure 5: Simplified overview of a message flow in WRPTP.

The flow of events for standard PTP which is presented in section 2 is extended as depicted in Figure 5 and described below (simplified overview):

1. The WR Master sends periodically WR Announce message with the custom suffix.
2. The WR Slave receives an Announce message, recognizes it as the WR Announce message and uses the modified BMC algorithm to establish its place in the WR network hierarchy.
3. The WR Slave starts WR Link Setup by sending the SLAVE_PRESENT WR Management message.
4. The WR Master starts sends the LOCK WR Management message to request the WR Slave to start syntonization.
5. The WR Slave sends LOCKED WR Management message as soon as the syntonization process is finished (notification from the hardware).
6. The WR Master sends the CALIBRATE WR Management message to request calibration of its reception fixed delay.
7. The WR Master sends the CALIBRATED WR Management message as soon as the calibration is finished (notification from hardware).
8. The WR Slave sends the CALIBRATE WR Management message to request calibration of its reception fixed delay.
9. The WR Slave sends the CALIBRATED WR Management message as soon as the calibration is finished (notification from hardware).
10. The WR Master sends the WR_MODE_ON WR Management message to indicate completion of the WR Link Setup process.
11. The WR Master sends periodically a Sync message (timestamped on transmission, $t_1$) followed by a Follow_UP message which carries $t_1$.
12. The WR Slave receives the Sync message sent by the master (timestamped on reception, $t_2$).
13. The WR Slave receives the Follow_Up message sent by the master.
14. The WR Slave sends a Delay_Req message (timestamped on transmission, $t_3$).
15. The WR Master receives the Delay_Req message sent by the master (timestamped on reception, $t_4$).
16. The WR Master sends a Delay_Resp message which carries $t_4$.
17. The WR Slave receives the Delay_Resp.
18. The WR Slave adjusts its clock using the offset and the delay calculated with the timestamps ($t_1$, $t_2$, $t_3$, $t_4$) corrected due to the precise knowledge of the link delay. It results in the Slave's synchronization with the Master clock with sub-ns accuracy.
19. Repeat 1, 11-18.

Since the WR Link Setup is performed in the PTP UNCALIBRATED state, it is essential for a WR node to implement this state as specified in the PTP state machine (Appendix B): a transition state between the LISTENING or PRE_MASTER or MASTER or PASSIVE state and the SLAVE state.

In this document, the term *node* is used interchangeably with *port*. A WR Master node/port is an ordinary clock with predefined Master functionality, working as a Master on the link. A WR Slave node/port is an ordinary clock with predefined Slave functionality, working as a Slave on the link.

A *White Rabbit Switch* (WRSW) is not a PTP-compliant boundary clock. It is considered a set of ordinary clocks with predefined functionality (WR Master or WR Slave) rather than a clock with multiple PTP ports. As a consequence, WRPTP messages are never forwarded. A *White Rabbit Timing Receiver* is an ordinary clock with predefined WR Slave functionality (slave node). Proper performance of a WR network is ensured by connecting a WR Slave port to a WR Master port.

Figure 6 depicts the topology of a *hybrid* WR/IEEE1588 network where optimal synchronization with grandmaster is achieved by connecting a non-WR Slaves to WR Masters.

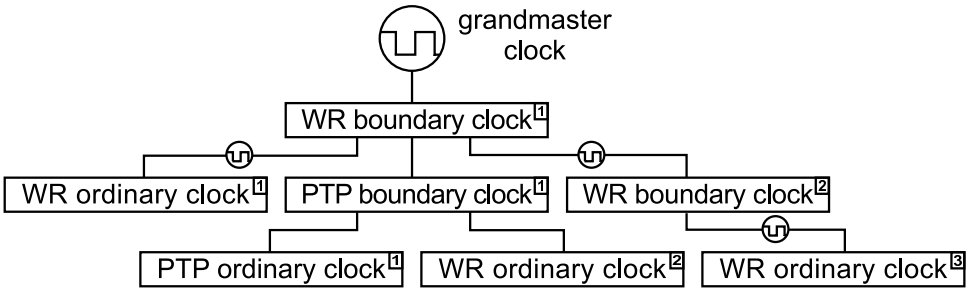



Figure 6: Hybrid WR/IEEE1588 network. White Rabbit nodes work transparently with PTP nodes. WR ordinary clock 3 is more accurately synchronised to the grandmaster than WR ordinary clock 2, which is below a PTP boundary clock.

## 6.2 WRPTP Data Sets Fields

The PTP standard defines data sets (DS) to store the static and dynamic variables needed for the operation of the protocol (section 8, PTP). WRPTP requires additional DS fields to store the WR-specific parameters. Table 1 defines and describes the additional required DS fields that are not a part of PTP standard.

Table 1: WRPTP Data Sets fields

| **DS member** | **DS name** | **Values** | **Description** |
|---|---|---|---|
| wrPortMode | portDS | NON_WR, WR_SLAVE, WR_MASTER | Determines predefined function of WR port (static). |
| calibrated | portDS | TRUE, FALSE | Indicates whether fixed delays of the given port are known. |
| deltaTx | portDS | 64 bit value | Port's $\Delta_{tx}$ measured in picoseconds and multiplied by $2^{16}$. |
| deltaRx | portDS | 64 bit value | Port's $\Delta_{rx}$ measured in picoseconds and multiplied by $2^{16}$. |
| calPeriod | portDS | 32 bit value | Calibration period in microseconds. |
| calPattern | portDS | 32 bit value | Medium specific calibration pattern. |
| calPatternLen | portDS | 16 bit value | Number of bits of calPattern to be repeated. |
| wrMode | portDS | TRUE, FALSE | If TRUE, the port is working in WR mode. |
| wrAlpha | portDS | 32 bit value | *Medium correlation parameter* as described in section 3.1.1. |
| grandmasterWrPortMode | parentDS | NON_WR, WR_SLAVE, WR_MASTER | Determines predefined function of the PTP grandmaster. |
| grandmasterDeltaTx | parentDS | 64 bit value | Grandmaster's $\Delta_{tx}$ measured in picoseconds and multiplied by $2^{16}$. |
| grandmasterDeltaRx | parentDS | 64 bit value | Grandmaster's $\Delta_{rx}$ measured in picoseconds and multiplied by $2^{16}$. |
| grandmasterWrMode | parentDS | TRUE, FALSE | If TRUE, the grandmaster is working in WR mode. |

## 6.3 Modified Best Master Clock Algorithm

The Best Master Clock algorithm is used in PTP to compare local clocks, to determine which clock is the *"best"* and to recommend the next state of the PTP state machine (section 9.3, PTP).

It is required from modified BMC that the comparison of a WR Master with a WR Slave or non-WR clock results in the WR Master being the *"best"* clock. To ensure a proper BMC outcome, the WR Master *clockClass* value shall be in the range of 1 through 127 (recommended 6) and the WR Slave *clockClass* shall be in the range of 128 through 255 (recommended 255).

## 6.4 WRPTP Messages

### 6.4.1 Overview

White Rabbit benefits from PTP's messaging facilities. It uses a two-step clock delay request-response mechanism and customizes Announce and Management messages (Figure 5). In particular, it adds a suffix to the Announce message, defines a *WR Type-Length-Valu*e (WR TLV) type, WR management action and management IDs.

A White Rabbit Master node announces its presence by adding a WR TLV suffix to the Announce message. The suffix is defined by PTP standard as a set of TLV entities (section 13.4, PTP), unrecognized TLVs are ignored by standard PTP nodes (section 14.1, PTP), but read and interpreted by White Rabbit nodes. The information provided in the WRPTP Announce message is sufficient for a WR Slave to decide whether the WR link can be established and maintained. The WR link is established through the WR Link Setup process in the PTP UNCALIBRATED state. If a WR Link Setup is required, the WR Slave starts the process and requests the WR Master to do the same. During the WR Link Setup, communication between the WR Master and the WR Slave is performed using the PTP Management mechanism extended for White Rabbit requirements. Once the WR link has been established, the WR nodes use a PTP delay request-response mechanism (section 11.3, PTP).

### 6.4.2 WR Type-Length-Value Type

All PTP messages can be extended by means of a standard *type, length, value* (TLV) extension mechanism. White Rabbit defines the value of TLV type out of the range reserved for Experimental TLVs (Table 34, PTP) as depicted in Table 2. This value is used to recognize the WR TLV entity in all WR custom messages.

Table 2: White Rabbit Type-Length-Value (WR TLV) type

| **tlvType values** | **Value (hex)** | **Defined in clause** |
|---|---|---|
| White Rabbit TLV (WR TLV type) | 0x2004 | — |

### 6.4.3 WRPTP Announce Message

The standard PTP Announce Message is suffixed by one entity of the data type TLV with the tlvType of WR TLV. The WRPTP Announce message has the structure defined in Table 3. The *dataField* of the suffix TLV stores the *wrFlags* which are defined in Table 4.

Table 3: White Rabbit Announce Message

| Bits | | | | | | | | Octets | TLV Offset | Content |
|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | |
| header | | | | | | | | 34 | 0 | section 13.3, PTP. |
| body | | | | | | | | 30 | 34 | section 13.5, PTP. |
| tlvType | | | | | | | | 2 | 64 | 0x2004, see 6.4.2. |
| lengthField | | | | | | | | 2 | 66 | 0x2, section 14.1.2, PTP. |
| wrFlags | | | | | | | | 2 | 68 | see Table 4. |

Table 4: White Rabbit flags (unused flags are reserved and shall be written to 0, ignored when read)

| Octet | Bit | Message type | Name | Description |
|---|---|---|---|---|
| 0 | 0 | Announce | wrMaster | TRUE if the port of the originator is predefined WR Master. |
| 0 | 1 | Announce | wrSlave | TRUE if the port of the originator is predefined WR Slave. |
| 0 | 2 | Announce | calibrated | TRUE if the port of the originator is calibrated. |
| 0 | 3 | Announce | wrModeOn | TRUE if the port of the originator is in WR mode. |

### 6.4.4 WRPTP Management Messages

White Rabbit extends the default PTP management mechanism described in section 15.2 of PTP. The extension conforms to the PTP management message format presented in Table 5. It uses the reserved range of *actionField* values (Table 38, PTP) to define a *White Rabbit Command* (WRC) as described in Table 6. WRC management messages trigger transitions in WR state machines. They are recognized by the *managementId* field of *managementTLV* (Table 8, 9 & 10). WR *management IDs* are defined in Table 12. WRPTP management messages are exchanged only within one link connection (no forwarding), therefore the *startingBoundaryHops* and *boundaryHops* fields are unused and set to 0x0. The rest of this subsection describes the WR management messages in detail.

Table 5: PTP Management Message (Table 37, PTP)

| Bits | | | | | | | | Octets | TLV 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 |

Table 6: White Rabbit value of the actionField

| Action | Action taken | Value (hex) |
|---|---|---|
| WR_CMD | The management message shall carry a single TLV. The *managementId* field of the TLV indicates the specific event which triggers transition in WR FSMs. | 0x5 |

Table 7: White Rabbit managementId values

| managementId name | managementId value (hex) | Allowed actions | Applies to |
|---|---|---|---|
| SLAVE_PRESENT | 0x6000 | WR_CMD | port |
| LOCK | 0x6001 | WR_CMD | port |
| LOCKED | 0x6002 | WR_CMD | port |
| CALIBRATE | 0x6003 | WR_CMD | port |
| CALIBRATED | 0x6004 | WR_CMD | port |
| WR_MODE_ON | 0x6005 | WR_CMD | port |

#### 6.4.4.1 SLAVE_PRESENT

Message sent by the WR Slave to the WR Master. It initiates the WR Link Setup process in the WR Master. The message shall have the form specified in Table 8.

#### 6.4.4.2 LOCK

Message sent by the WR Master to the WR Slave to request the start of frequency locking. The message shall have the form specified in Table 8.

#### 6.4.4.3 LOCKED

Message sent by the WR Slave to the WR Master. It indicates successful completion of frequency locking. The message shall have the format specified in Table 8.

Table 8: WR Management TLV

| Bits | | | | | | | | Octets | TLV Offset | Content |
|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | |
| tlvType | | | | | | | | 2 | 0 | 0x2004, see 6.4.2. |
| lengthField | | | | | | | | 2 | 2 | 0x2, section 15.5.2.3, PTP. |
| managementId | | | | | | | | 2 | 4 | Defined in Table 12. |

#### 6.4.4.4 CALIBRATE

Messages sent by the WR node entering the REQ_CALIBRATION state (see section 6.5). It informs the other node whether sending calibration pattern (see section 5) is required (defined

by the value of *calibrationSendPattern* flag). If calibration is required, it carries a set of parameters describing the calibration pattern to be sent. The message format and parameters are described in Table 9.

Table 9: CALIBRATE WR Management TLV

| **Bits** | | | | | | | | **Octets** | **TLV Offset** | **Content** |
|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | |
| tlvType | | | | | | | | 2 | 0 | 0x2004, see 6.4.2. |
| lengthField | | | | | | | | 2 | 2 | 0xC, section 15.5.2.3 PTP. |
| managementId | | | | | | | | 2 | 4 | CALIBRATE. |
| calibrationSendPattern | | | | | | | | 2 | 6 | The value determines whether calibration pattern should be sent. If the value is 0x1, the calibration pattern is sent. If the value is 0x0, the calibration pattern is not sent. |
| calibrationPeriod | | | | | | | | 4 | 8 | The value defines the time (in microseconds) for which the calibration pattern should be sent by receiving node. |
| calibrationPattern | | | | | | | | 4 | 12 | The value defines the calibration pattern which should be sent by the receiving node. |
| calibrationPatternLen | | | | | | | | 2 | 14 | The value defines the number of bits of *calibrationPattern* field which should be used as repeated pattern (starting with the LSB). |

#### 6.4.4.5 CALIBRATED

Message sent by the WR node entering *CALIBRATED* state. If preceded by the *CALIBRATE* message with *calibrationSendPattern* set to TRUE, it indicates successful completion of the calibration. The message provides the other node with the values of its fixed delays ($\Delta_{tx}$ and $\Delta_{rx}$). The messages shall have the format specified in Table 10.

Table 10: CALIBRATED WR Management TLV

| **Bits** | | | | | | | | **Octets** | **TLV Offset** | **Content** |
|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | |
| tlvType | | | | | | | | 2 | 0 | 0x2004, see 6.4.2. |
| lengthField | | | | | | | | 2 | 2 | 0x24, section 15.5.2.3 IEEE1588 [1]. |
| managementId | | | | | | | | 2 | 4 | MASTER_CALIBRATED. |
| deltaTx | | | | | | | | 16 | 6 | The value of $\Delta_{tx_m}$ measured in picoseconds and multiplied by $2^{16}$. |
| deltaRx | | | | | | | | 16 | 22 | The value of $\Delta_{rx_m}$ measured in picoseconds and multiplied by $2^{16}$. |

#### 6.4.4.6 WR_MODE_ON

Message sent by WR Master to WR Slave. It indicates successful completion of the WR Link Setup process and requests the WR Slave to enter WR mode. The message shall have the format specified in Table 8.

## 6.5 White Rabbit State Machine

The White Rabbit finite state machine (WR FSM) controls the process of establishing a White Rabbit link between a WR Master and a WR Slave (WR Link Setup). It involves recognition of two compatible WR nodes, syntonization over the physical layer, measurement of fixed delays and exchange of their values across the link. The procedure differs between WR Master and WR Slave, therefore three states are Slave-only (entered only if the node is in WR Slave mode) and one state is Master-only (entered only if the node is in WR Master mode). The WR FSM shall be executed in the PTP UNCALIBRATED state, it is depicted in Figure 7 and described in the rest of this section.

A simplified flow of WR Management Message exchange between a WR Slave and a WR Master during WR Link Setup is presented in Appedix 10.

### 6.5.1 Condition to start WR FSM by WR Slave

The WR FSM in WR Slave exits the IDLE state and starts execution by entering the *PRESENT* state, only when the PTP state machine (Appendix B) is in the PTP UNCALIBRATED state and the following conditions are met:

- the node is WR Slave:
  ($portDS.wrPortMode = WR\_SLAVE$) **AND**
- the parent node is WR Master:
  ($parentDS.grandmasterWrPortMode = WR\_MASTER$) **AND**
- the node or parent node or both nodes are not in WR Mode:
  (*portDS.wrMode = FALSE OR parentDS.grandmasterWrMode = FALSE*).

### 6.5.2 Condition to start WR FSM by WR Master

The WR Master node shall enter the PTP UNCALBRATED state, and start execution of the WR FSM by entering *LOCK* state, when it receives a SLAVE_PRESENT WR Management message (Table 12).

#### 6.5.2.1 State Description

Table 11 specifies the WR states notation used in Figure 7.

Table 11: WR state definition

| PTP portState | Description |
|---|---|
| IDLE | WR FSM shall be in the IDLE state if the PTP FSM is in a state other than UNCALIBRATED. |
| PRESENT | Slave-only state. The WR Slave sends SLAVE_PRESENT message to the WR Master and waits for the *LOCK* message. |
| M_LOCK | Master-only state. The WR master waits for the WR Slave to finish successfully the locking process. |
| S_LOCK | Slave-only state. The WR Slave locks its logic to the frequency distributed over physical layer by the WR Master. |
| LOCKED | Slave-only state. The WR Slave is syntonized, it sends *LOCKED* message to the WR Master and waits for the *CALIBRATE* message. |
| REQ_CALIBRATION | In this state, optional calibration of the node's reception fixed delay can be performed. The node sends *CALIBRATE* message to the other node. If the calibration is needed, (*calibrated* is set to false), the *calibrationSendPattern* flag in the *CALIBRATE* message is sent to TRUE (0x1). If the calibration is not needed, the *calibrationSendPattern* flag is set to FALSE (0x0). If calibration is not needed, next state is entered directly, otherwise an indication from the hardware that the calibration has been finished successfully is awaited. |
| CALIBRATED | The node sends CALIBRATED message with information about its fixed delays. |
| RESP_CALIB_REQ | The node's action in this state depends on the value of the *calibrationSendPattern* flag received in the *CALIBRATE* message. TRUE value of the flag indicates that calibration pattern shall be enabled. The pattern shall be disabled after *calibrationPeriod* or on reception of the *CALIBRATED* message. If the value of the *calibrationSendPattern* flag is FALSE, the *CALIBRATED* message is awaited for a default timeout. On reception of the *CALIBRATED* message, the next state is entered. |
| WR_LINK_ON | The value of *wrMode* is set to TRUE and the *IDLE* state is entered. |

#### 6.5.2.2 WR FSM Transition Events and Conditions

**POWERUP** Turning on power to the device or reseting.

**WR LINK SETUP REQUIRED DECISION** (abrv. D_WR_SETUP_REQ) Event indicating that WR Link Setup is required and WR FSM should be executed starting with *PRESENT* state, see section 6.5.1.

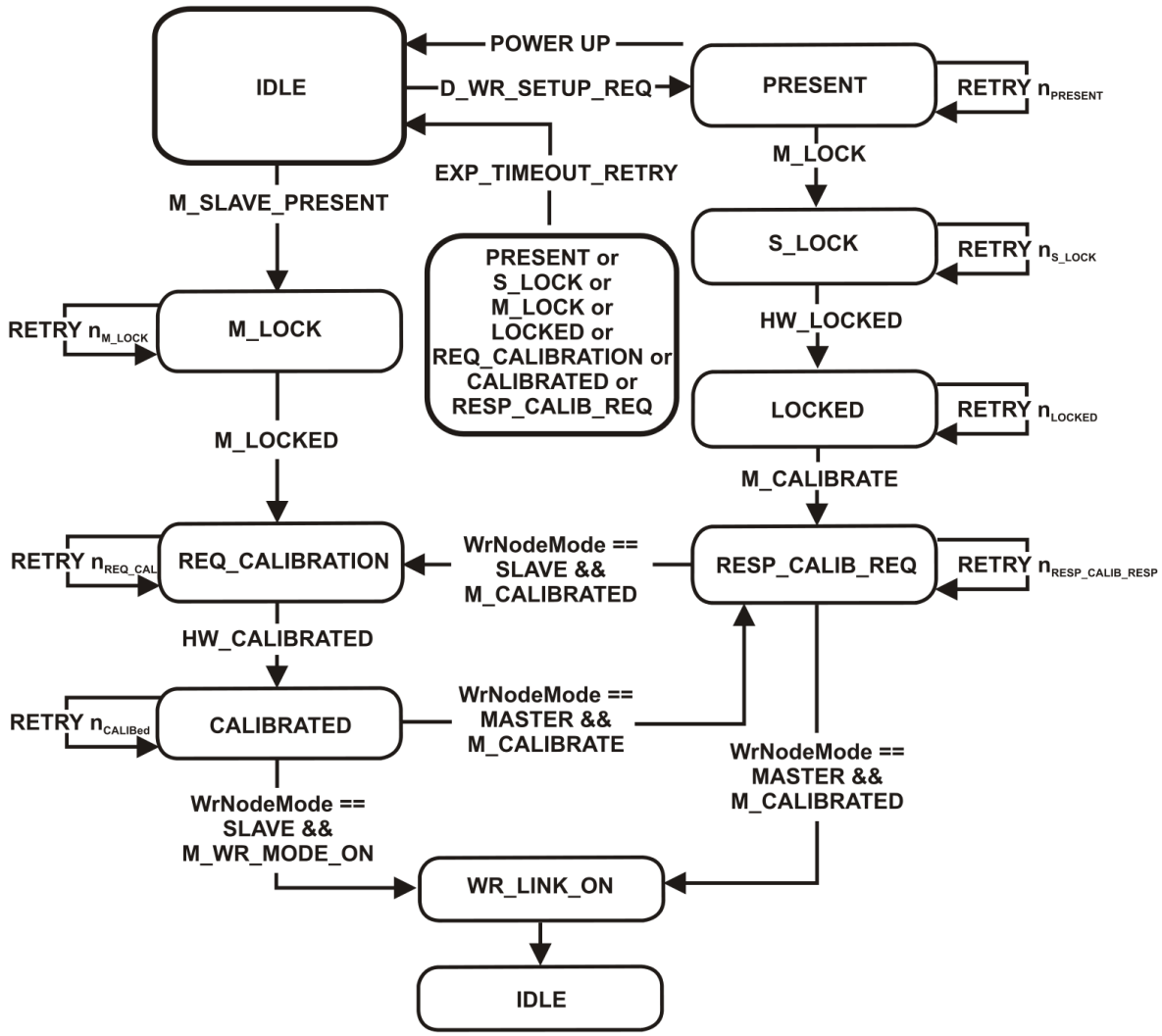



Figure 7: White Rabbit state machine.

**LOCK MESSAGE** (abrv. M_LOCK) WR *LOCK* Management message which triggers frequency locking over the physical layer.

**LOCKED HARDWARE EVENT** (abrv. HW_LOCKED) Indication from the hardware that frequency locking has been completed successfully.

**LOCKED MESSAGE** (abrv. M_LOCKED) WR *LOCKED* Management message which notifies the WR Master that frequency locking has been completed successfully by the WR Slave.

**CALIBRATE MESSAGE** (abrv. M_CALIBRATE) WR *CALIBRATE* Management message. It request the recipient to enter *RESP_CALIB_REQ* state, indicates whether sending of the calibration pattern is required and provides calibration parameters.

**CALIBRATED HARDWARE EVENT** (abrv. HW_CALIBRATE) Notification from the hardware indicating that calibration has been completed successfully.

**CALIBRATED MESSAGE** (abrv. M_CALIBRATED) WR CALIBRATED Management message. It indicates that the node is calibrated. If the *CALIBRATED* message is received when the calibration pattern is being sent by the recipient, sending of the pattern shall be disabled. The message carries information about fixed delays of the sending node.

**WR MODE ON** (abrv. M_WR_MODE_ON) WR *WR_MODE_ON* Management message. It indicates that the WR Master finished successfully WR Link Setup and set the wrMode flag to TRUE. It requests the WR Slave to set the wrMode flag to TRUE.

**RETRY** $n_{name}$ White Rabbit state machine waits in a given state for a transition event only for a limited time (*TIMEOUT*) The states to which this rule applies are the following: *M_LOCK*, *REQ_CALIBRATEION*, *CALIBRATED*, *PRESENT*, *S_LOCK*, *LOCKED*, *RESP_CALIB_REQ*. After the *TIMEOUT* expires, the state is re-entered for $n_{\{M\_LOCK,REQ\_CAL,CALIBed,PRESENT,S\_LOCK,LOCKED,RESP\_CALIB\_RESP\}}$ number of times.

**EXCEED TIMEOUT RETRIES** (abrv. EXC_TIMEOUT_RETRY) Indicates that the state has been re-entered for a set number of times ($n_{\{M\_LOCK,REQ\_CAL,CALIBed,PRESENT,S\_LOCK,LOCKED,RESP\_CAL}$ and the *TIMEOUT* has expired for the $n+1$ time.

# Appendix

## A Measurement of fixed delays for Gigabit Ethernet over Optic Fiber

The variation of $\Delta_{\{tx_m,rx_s,tx_s,rx_m\}}$ delays is often caused by the PHY's serializer / deserializer (SerDes), phase locked loop (PLL) or clock and data recovery circuitry (CDR). The delay on the PHY can be measured by detecting the phase shift between SerDes I/O and Tx/Rx clock. This can be done, in example, by sending a repeated pattern of five ”0” and five ”1” (0000011111) over Gigabit Ethernet. Such signal creates a 125 MHz clock on the SerDes I/O. Since the Tx/Rx clock frequency is 125 MHz, the phase shift between the SerDes I/O and the Tx/Rx clocks is equal to the fixed delay of the PHY (see Figure 8). The repeated pattern of five ”0” and five ”1” is an example of *calibration pattern* which is defined by the node requesting calibration.

The calibration pattern used to describe the methods is not compliant with 8b/10b encoding standard. For the real-life implementation, a 8b/10b compliant pattern is preferable.

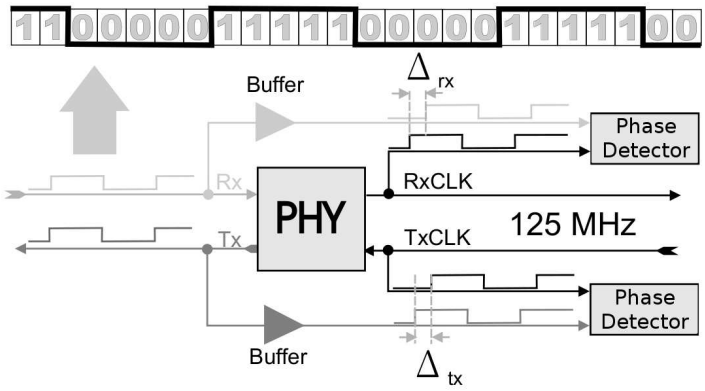



Figure 8: Measurement of fixed delays $\Delta_{\{tx,rx\}}$ in Gigabit Ethernet-based WR node with not full-deterministic PHY.

Measurement of fixed delays for Gigabit Ethernet over optic fiber, and any other medium, is optional. It is not needed if deterministic PHYs or internal FPGA transceivers which can be internally characterized [2] are used. In such case, the information about the fixed delays is distributed across the link without preceding measurement.

# B PTP State Machine

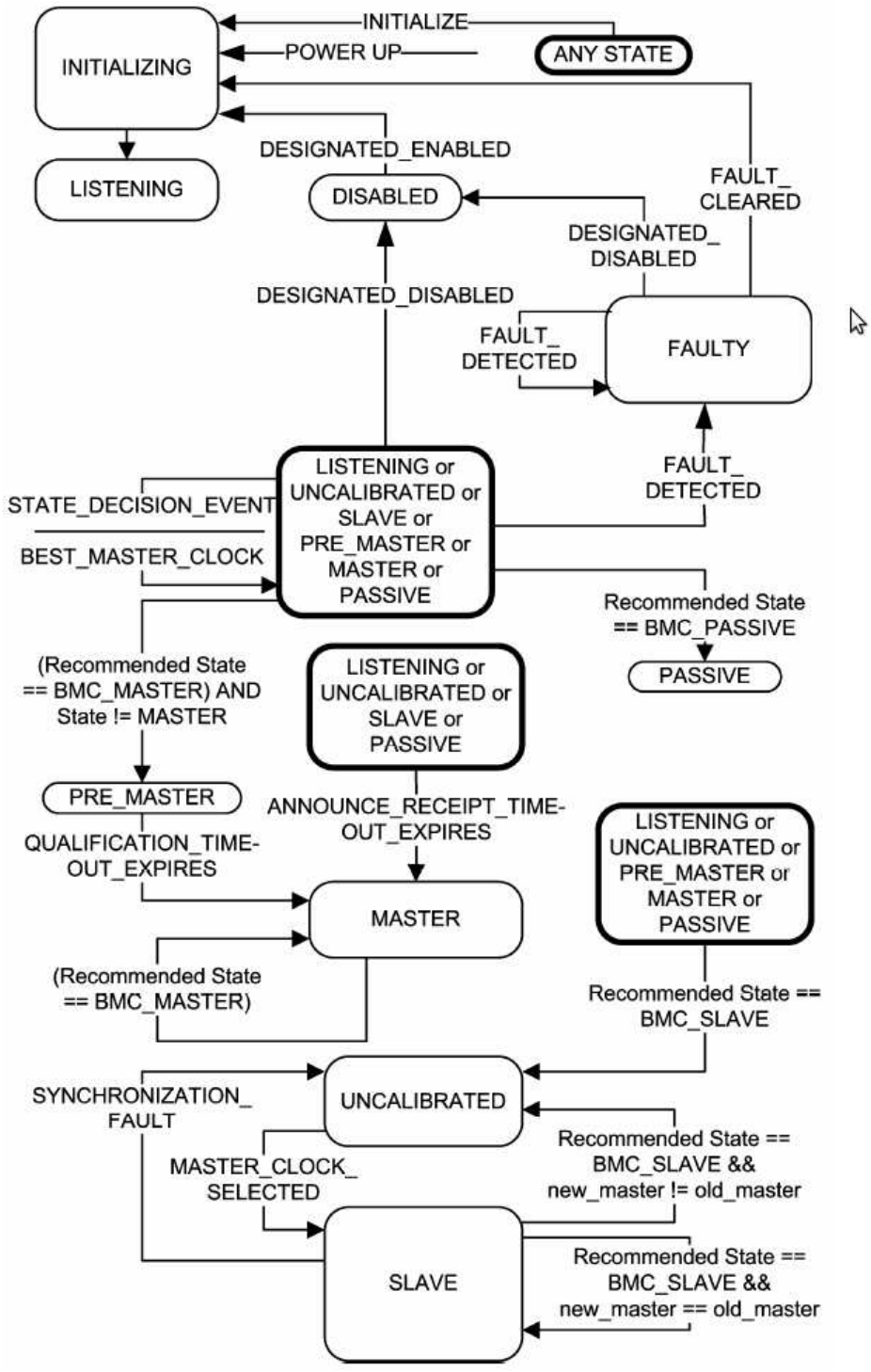



Figure 9: State machine for a full implementation of PTP (Figure 23, IEEE1588).

Table 12: PTP portState definition (Table 10, IEEE1588).

| **PTP portState** | **Description** |
|---|---|
| INITIALIZING | While a port is in the INITIALIZING state, the port initializes its data sets, hardware, and communication facilities. No port of the clock shall place any PTP messages on its communication path. If one port of a boundary clock is in the INITIALIZING state, then all ports shall be in the INITIALIZING state. |
| FAULTY | The fault state of the protocol. A port in this state shall not place any PTP messages except for management messages that are a required response to another management message on its communication path. In a boundary clock, no activity on a faulty port shall affect the other ports of the device. If fault activity on a port in this state cannot be confined to the faulty port, then all ports shall be in the FAULTY state. |
| DISABLED | The port shall not place any messages on its communication path. In a boundary clock, no activity at the port shall be allowed to affect the activity at any other port of the boundary clock. A port in this state shall discard all PTP received messages except for management messages. |
| LISTENING | The port is waiting for the announceReceiptTimeout to expire or to receive an Announce message from a master. The purpose of this state is to allow orderly addition of clocks to a domain. A port in this state shall not place any PTP messages on its communication path except for Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up, or signaling messages, or management messages that are a required response to another management message. |
| PRE_MASTER | The port shall behave in all respects as though it were in the MASTER state except that it shall not place any messages on its communication path except for Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up, signaling, or management messages. |
| MASTER | The port is behaving as a master port. |
| PASSIVE | The port shall not place any messages on its communication path except for Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up, or signaling messages, or management messages that are a required response to another management message. |
| UNCALIBRATED | One or more master ports have been detected in the domain. The appropriate master port has been selected, and the local port is preparing to synchronize to the selected master port. This is a transient state to allow initialization of synchronization servos, updating of data sets when a new master port has been selected, and other implementation-specific activity. |
| SLAVE | The port is synchronizing to the selected master port. |

# C Flow of WR Management message exchange between a WR Master and a WR Slave (no exceptions) during WR Link Setup

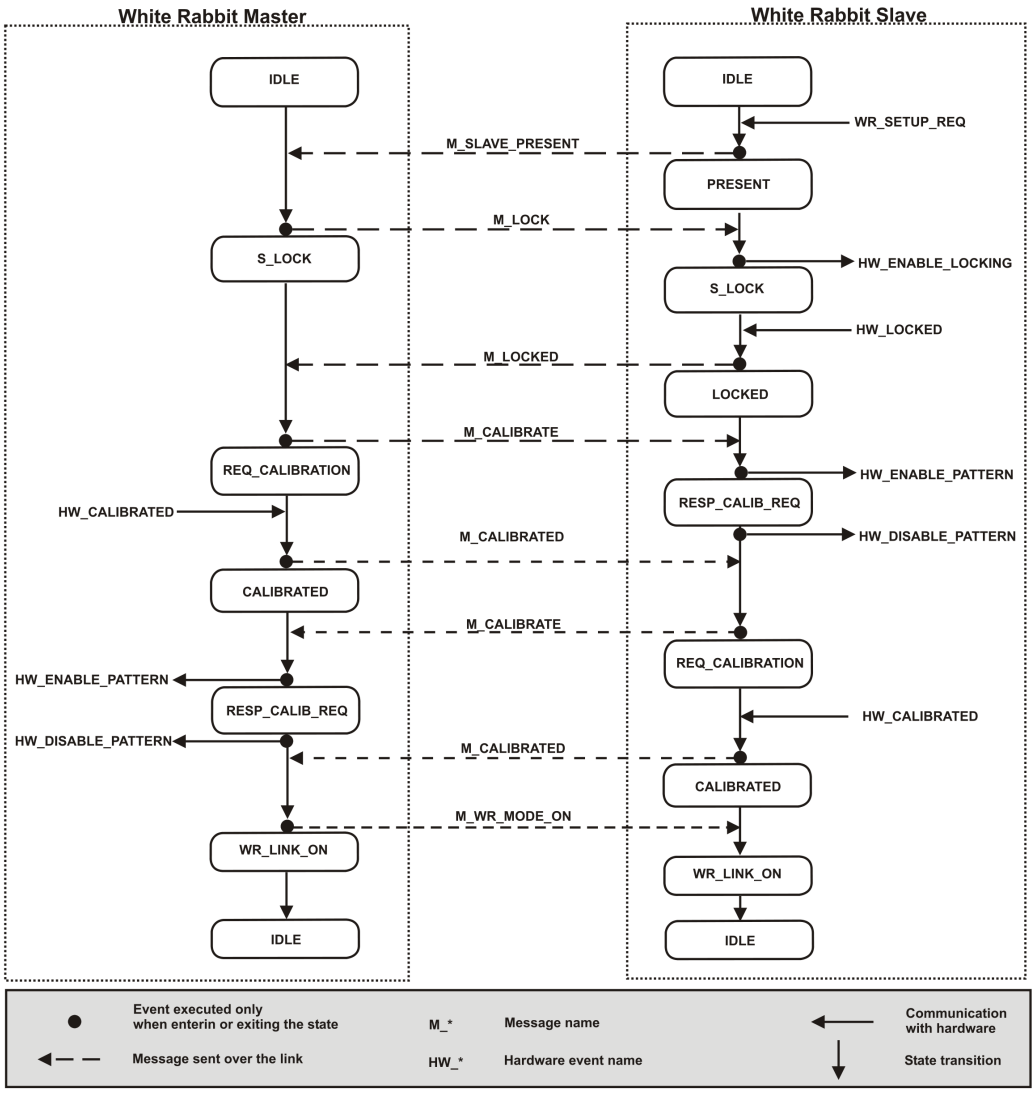



Figure 10: Flow of events (no exceptions) during WR Link Setup.

# References

[1] IEEE Std 1588-2008 *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*. IEEE Instrumentation and Measurement Society, New York, 2008, http://ieee1588.nist.gov/.

[2] P.P.M. Jansweijer, H.Z. Peek, *Measuring propagation delay over a 1.25 Gbps bidirectional data link*. National Institute for Subatomic Physics, Amsterdam, 2010, http://www.nikhef.nl/pub/services/biblio/technicalreports/ETR2010-01.pdf.