White Rabbit protocol (WRP) specification. Very preliminary.
2008/10/22

Table of contents:

1. Introduction

White Rabbit (WRP) protocol is a low-level timing and control transport protocol for White Rabbit network devices. The tasks of WRP are:

- providing reliable one-way transmission channels (without handshaking) for low-latency control and timing messages
- precise delay measurement and reporting, fine (sub-nanosecond scale) transparent time transmission based on PTPv2 protocol and synchronous Ethernet (frequency reference is embedded into Ethernet carrier).
- identification of WR-compliant devices in the network

WRP operates on 2nd (data link - MAC) network layer to make it simple and easy to implement in hardware. WRP is software-transparent protocol.

2. Network structure

White Rabbit network consists of following active components:

- **System timing master** – provides time and frequency reference for the whole network and sends all control and timing messages
- **Switches** capable of routing high priority (HP) WRP frames with integrated PTPv2 boundary clock functionality
- **Slave devices (nodes)** such as timing receivers, WorldFIP/Powerlink bridges, standard twisted-pair Ethernet bridges, etc.
- **Network cabling -** single-mode fiber (for backbone WR network or high precision timing receivers) or twisted-pair (for devices where ultrafine timing is not required)

3. Frame types

White Rabbit protocol uses two different frame types:

- **standard traffic (SP)** – standard Ethernet frames (Ethertype `0xa0a1`). For this kind of traffic, switches operate in store-then-transmit mode (like standard Ethernet gear). In case of collision, data is buffered. This type of traffic is non-deterministic. There are no restrictions on usage of SP frames. SP frames can be fragmented by WhiteRabbit switches when switch needs to route incoming HP frame. WR compatible devices shall be able to reconstruct fragmented SP frames. Compatibility with WR frame fragmentation and reconstruction can be achieved on purely software way, without hardware modification (e.g. using standard Ethernet NICs/switches)
- **high-priority traffic (HP)** – Ethernet frames with special unique value of EtherType field (`0xa0a0`). These frames have absolute priority over SP frames to maintain low and deterministic transmission delay. HP frames can be only sent in response to HP frame sent by master device.

Please note that other types of frames (not used by WRP protocol - with *ethertypes* different than `0xa0a0` or `0xa0a1`) are treated by switches like SP frames. They can be fragmented when HP frame arrives and continued by sending proper SP frame(s).

Frame structures are depicted on **fig. 1.**

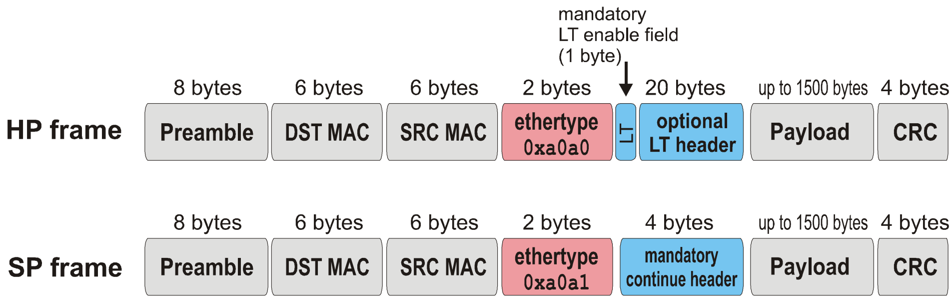

Fig. 1. White Rabbit ethernet frames structure

4. HP Frames

HP frames are frames for time-critical control data. They are routed with as low latency as possible, forcing fragmentation of SP/non-WR traffic if required. Structure of such frame is shown below:

```
__attribute__ ((packed)) struct WR_HP_frame {

      struct eth_header {              // standard Ethernet frame header
            uint8_t preamble[8];       // ethernet preamble (7 x 0xaa + 0xab)
            uint8_t dst_mac[6];        // destination MAC. Always ffffffffffff (bcast)
            uint8_t src_mac[6];        // source MAC
            uint16_t ethertype = 0xa0a0;     // EtherType
      };

      uint8_t lt_enabled;              // LT coded-frame (non-0 - yes, 0 - no)

      struct lt_header {               // optional LT-coding header
            uint32_t orig_frame_id;    // unique original frame id
            uint32_t eqn;              // equation coefficients
            uint16_t segment_size;     // size of LT segment in bytes
            uint16_t segment_num;      // number of current segment
            uint8_t md5sum[8];         // MD5 frame checksum (LT header + LT payload)
      };

      uint8_t payload[32..1500];       // frame payload
      uint32_t crc;                    // frame CRC32
};
```

To maintain compatibility with non-WR ethernet gear, HP frames are no different from Ethernet standard frames. WR switches recognize them only by unique value of `ethertype` field. HP frames may contain optional LT encoding header, when value of `lt_enabled` field is non-zero. If `lt_enabled` is 0, LT encoding header is not present (payload data begins right after `lt_enabled` field). LT coding is described in detail in **section 7**.

HP frames are physically broadcast through whole network (all switches are receiving them), however they reach only nodes to which they are assigned via destination MAC 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 SP/non-WR traffic.

5. SP frames

SP frames are used for non time-critical traffic:

- sending WRP messages
- continuation of transmission of fragmented SP frames or other Ethernet frames.

Below is shown C-like structure of SP frame.

```
__attribute__ ((packed)) struct WR_SP_frame {

	struct eth_header {		// standard Ethernet frame header
		uint8_t preamble[8];	// ethernet preamble (7 x 0xaa + 0xab)
		uint8_t dst_mac[6];	// destination MAC.
		uint8_t src_mac[6];	// source MAC
		uint16_t ethertype = 0xa0a1;	// EtherType
	};

	struct SP_mandatory_header {	// mandatory SP header
		uint16_t is_continue;
		uint16_t continue_offset;
	};

	uint8_t payload[42..1500];	// frame payload

	uint32_t original_crc;		// CRC of original frame (before fragmentation)
	uint32_t crc;			// frame CRC32
};
```

Each SP frame must contain special (WR-specific) SP header, containing:

- `is_continue` flag. Its non-zero value informs, that payload of this SP frame is continuation of previous SP or other frame which has been fragmented due to routing of HP frame.
- `contiunue_offset`. Contains payload offset at which previous frame was interrupted.

Frames can be fragmented multiple times. In this case multiple SP continuation frames are sent. Last segment contains value of original (received) CRC checksum of fragmented frame before its own CRC. This is done for two reasons:

- to make it easy to detect last segment of fragmented frame (if checksum of fragmented payload equals original CRC – this is the last segment of frame)
- to check if original frame (before fragmentation) was correct. If it wasn't, instead of being fragmented, frame is dropped.

6. HP frame handling and non-HP traffic fragmentation

As we mentioned previously, HP frames have absoulte priority over any other traffic in WR network. Therefore, when HP frame arrives, currently routed non-HP traffic shall be interrupted. Unfortunately, this may lead to excessive packet loss in non-HP traffic. To avoid this, WRP provides frame fragmentation and reconstruction mechanism, depicted on **fig. 2**.

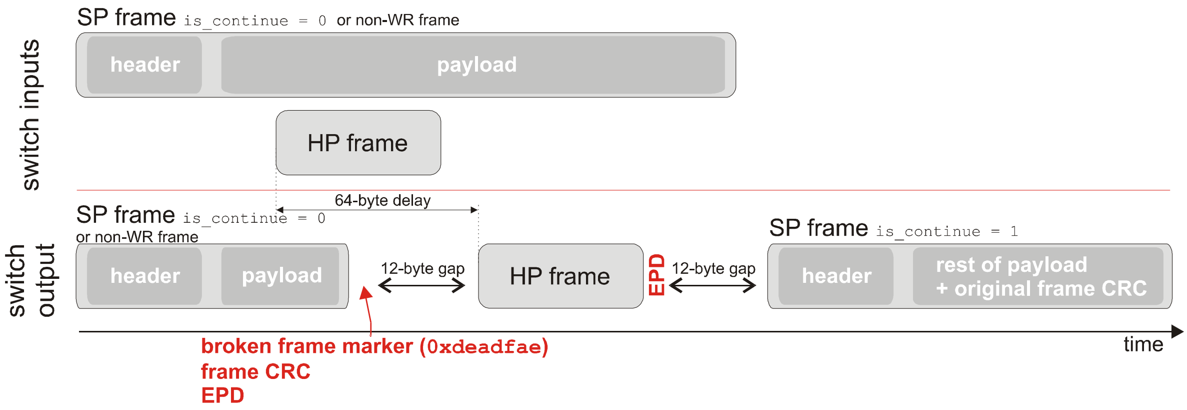

Fig. 2. Fragmentation of SP/non-WR frames

Below is description of situation shown on **fig. 2.**

1. At the beginning, WR switch is routing SP frame (or non-WR frame).
2. HP frame arrives. Switch immediately issues broken frame marker (to indicate that this frame is fragmented) followed by CRC of data sent so far. After then, switch issues end-of-packet delimeter (K29.7 8b10b control character) to terminate currently transmitted packet, then waits 12 byte clock cycles (required minimum frame-to-frame gap) and forwards the HP frame. In the meanwhile, data of incoming SP/non-WR frame is buffered.
3. After HP frame has been sent, switch sends SP frame with `is_continue` field set to non-zero value, containing rest of fragmented frame payload and its original (received) CRC.

Sometimes (althouth very rarely) nonfragmented frames may end broken frame markers. In such situation switch shall wait for reception of next frame and check if it is a continuation of previous frame.

Frames can be fragmented multiple times. Please note that HP frames are always delayed by switches by constant value of 64 byte clock cycles. This is to prevent fragmentation of headers of SP/non-WR frames. When at the same moment SP/non-WR header data is being sent while HP frame arrives, we have enough time to finish transmission of SP/non-WR header before start of transmission of HP frame. This approach should make it possible to implement frame fragmentation feature in software on standard (non-WR) Ethernet gear.

In order to prevent overflows in switch FIFOs, all nodes should immediately pause transmission of current SP/non-WR data when HP is being received.

7. LT Encoding

To maintain high reliability for non-handshaked HP frames containing 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 64 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 subsequent, separate HP frames with nonzero value of `lt_enabled` field and proper `lt_header` containing:
   - 32 bit unique identifier of original frame (`orig_frame_id`) allowing to identify to which frame each block belongs)
   - 32-bit integer with equation coefficients (`eqn`). For example, value of *100101* means that *Xi = A1 xor A4 xor A6.*
   - Segment size in bytes (`segment_size`)
   - Index of segment (`segment_num`)

   MD5 checksum of payload and LT header (`md5sum`)

4. Assuming that at least N blocks reached the receiver without errors, it 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 (less than 1 broken packet per second) this scheme will work perfectly well.

8. WRP messages

All WRP messages are sent using SP frames. HP frames provide only transport layer for timing-sensitivie traffic, and they shall never be used for WRP messages. All WRP packets have 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 1 shows available `field_types`. All multibyte binary data is encoded in big-endian format.

Table 1. WRP message field types

Type ID (`field_type`)	Flags	C-like name	Base size (bytes)	Description
`0x00`	none	`msgid_t`	2	16-bit unsigned integer, containing unique type identifier of the message. Can be used only once in

Table 1. WRP message field types				
Type ID (`field_type`)	**Flags**	**C-like name**	**Base size (bytes)**	**Description**
				each packet
`0x01, 0x41`	`FIELD_SIGNED` `FIELD_ARRAY`	`uint8_t` `sint8_t`	1	8-bit unsigned/signed integer in two's complement format
`0x02, 0x42`	`FIELD_SIGNED` `FIELD_ARRAY`	`uint16_t` `sint16_t`	2	16-bit unsigned/signed integer in two's complement format
`0x03, 0x43`	`FIELD_SIGNED` `FIELD_ARRAY`	`uint32_t` `sint32_t`	4	32-bit unsigned/signed integer in two's complement format
`0x04, 0x44`	`FIELD_SIGNED` `FIELD_ARRAY`	`uint64_t` `sint64_t`	8	64-bit unsigned/signed integer in two's complement format
`0x05`	`FIELD_ARRAY`	`float`	4	IEEE754 32-bit (single precision) floating point
`0x06`	`FIELD_ARRAY`	`double`	8	IEEE754 64-bit (double precision) floating point
`0x07`	none	`char *`	variable	Null-terminated string. UTF8 encoded.

Below is an example of WRP message containing 4 different NTLV fields:

- message ID 'MSID' = 0
- 32-bit signed integer 'INT1' = -100000
- array of 4 16-bit unsigned integers 'ARR1' = {10, 2000, 7777, 54321}
- ASCII string 'STR1' = 'Hello, world'

```
+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
```

8.1. WRP_INVITE (0x01)

Direction: switch/master → node
Sent in response to: detection of new node in network
Expected response: WRP_INVITE_RESPONSE
Source address: sender address
Destination address: slow protocol default (`01-80-c2-00-00-02`)

WRP_INVITE is sent by switch after detection that new node has been connected. Its purpose is to inform new node about network/timing configuration and 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`). The message is received by freshly connected node and acknowledged by sending WRP_INVITE_RESPONSE packet. Invite packet contains all addresses that node needs to know and important timing information.

Table 2. WRP_INVITE message fields

Field name	Type	4-char name	Purpose
msg_id	msgid_t	MSID	Unique message identifier for WRP_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 and maximal period of delay compensations performed by switch (in milliseconds)
max_delay_sync_period	uint32_t	MADS	
min_node_report_period	uint32_t	MIRP	Minimal and maximal period of reporting node status to switch (in milliseconds)
max_node_report_period	uint32_t	MARP	
capability_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` twisted-pair 100 Mbps WRP switch

8.2. WRP_INVITE_RESPONSE (0x02)

Direction: node → switch/master
Sent in response to: WRP_INVITE
Expected response: WRP_ACK
Source address: sender address
Destination address: switch address from WRP_INVITE

WRP_INVITE_RESPONSE packet contains capabilities of freshly connected device, allowing for switches/master to check if its WR-compliant. WRP_INVITE_RESPONSE is sent to closest switch, and then forwarded by switch to System Timing Master.

Table 3. WRP_INVITE_RESPONSE message fields

Field name	Type	4-char name	Purpose
`msg_id`	`msgid_t`	`MSID`	Unique message identifier for WRP_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 firmware 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 uplink 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)
`capability_sync_mode`	`uint8_t`	`CPSY`	Nonzero value means that node is capable of synchronous operation
`capability_phase_measurement`	`uint8_t`	`CPPM`	Nonzero value means that node can perform fine phase shift measurement for delay compensation
`capability_ptp`	`uint8_t`	`CPPT`	Nonzero value means that node supports delay compensation via PTPv2
`capability_hp_recv`	`uint8_t`	`CPHR`	Nonzero value means that node is capable of receiving HP frames
`capability_hp_send`	`uint8_t`	`CPHS`	Nonzero value means that node is capable of sending HP frames
`capability_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`	
`max_hp_delay_uncertainity`	`uint32_t`	`MADU`	*(switch/gateway only)* Maximal value of nondeterministic (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 path asymmetry in picoseconds.

8.3. WRP_ACK (0x03)

Direction: any
Sent in response to: WRP_INVITE_RESP, WRP_REPORT_NODE, WRP_REPORT_DELAY
Expected response: none
Source address: sender address
Destination address: any

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

Table 3. WRP_ACK message fields			
Field name	**Type**	**4-char name**	**Purpose**
`msg_id`	`msgid_t`	`MSID`	Unique message identifier for WRP_ACK. Equals to 0x03.
`ack_value`	`uint16_t`	`ACKV`	Zero means that packet previous request was acknowledged. Non-zero value means that error occurred and the message was rejected. `ack_value` field may be used to send value of error code.

8.4. WRP_REPORT_NODE (0x06)

Direction: node → switch, switch → STM
Sent in response to: none (periodic event)
Expected response: WRP_ACK
Source address: sender address
Destination address: switch or master

WRP_REPORT_NODE packet is used to report results of delay measurement and current node state to:

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

WRP_REPORT_NODE is sent periodically by each node sent to closest upper-layer switch. Switches shall forward these messages to System Timing Master. In case node stopped responding, the switch should issue appropriate WRP_REPORT_NODE itself.

Table 4. WRP_REPORT_NODE message fields			
Field name	**Type**	**4-char name**	**Purpose**
`msg_id`	`msgid_t`	`MSID`	Unique message identifier for WRP_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.
`state_changed`	`uint8_t`	`STCH`	Nonzero value means that state of node has changed since last report.

Table 4. WRP_REPORT_NODE message fields			
Field name	**Type**	**4-char name**	**Purpose**
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	uint64_t	RXTO	Total number of received packets (including HP)
rx_hp	uint64_t	RXHP	Total number of received HP packets
rx_errors	uint64_t	RXER	Total number of invalid packets received (including HP)
rx_errors_hp	uint64_t	RXEH	Total number of invalid HP packets received
node_state	uint16_t	NOST	Node state. Can be: 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 presnet – didn't report itself in expected time) 0x06 – DISCONNECTED (carrier lost, node is disconnected)
node_wr_compatible	uint8_t	NOTY	Nonzero value means that node is WR-compatible.
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.

8.5. WRP_REPORT_DELAY (0x07)

Direction: switch → node
Sent in response to: after delay measurement (periodic event)
Expected response: WRP_ACK
Source address: switch/master address
Destination address: measured node

WRP_REPORT_DELAY packet is sent to node after the switch had finished link delay measurement. Using information in this packet, slave can perform delay compensation.

Table 5. WRP_REPORT_DELAY message fields			
Field name	**Type**	**4-char name**	**Purpose**
msg_id	msgid_t	MSID	Unique message identifier for WRP_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

Table 5. WRP_REPORT_DELAY message fields			
Field name	**Type**	**4-char name**	**Purpose**
			between switch and node (in picoseconds)
`delay_downlink_master_to_switch`	`uint64_t`	`DMSD`	Overall downlink delay between System Timing Master 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.

9. PTPv2 Protocol compatibility

We've chosen subset of PTP version 2 protocol as a method for delay measurement and compensation in WR network. Therefore, each WR-compatible device shall implement PTP-compatible packet timestamping and (at least) following set of PTPv2 messages:

- `Pdelay_req`
- `Pdelay_resp`
- `Pdelay_resp_followup`
- Signalling message including TLVs for phase shift measurement

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

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

Custom TLV field, with command for slave to set its phase shift is introduced to enable phase shift measurement (using method described in *Fiber delay compensation* paper). All delay calculations are performed by master side (e.g. switch or STM).

10. Use cases

10.1. New node connected to switch

Occurs when switch detects carrier (e.g. valid 8b10b symbols) in one of downlink ports, indicating that new node has been connected.

1. Switch waits for 1 second and sends WRP_INVITE packet to freshly connected node
2. Node responds with WRP_INVITE_RESPONSE packet containing its capabilities
3. If correct WRP_INVITE_RESPONSE was received, switch sends ACK packet to the node. Otherwise switch sends WRP_INVITE packet again. Retransmission can be performed up to 5 times.
4. If there is no response from node, or the response is malformed, switch assumes that device is nonsynchronous and non-WR capable. Switch issues WRP_REPORT_NODE message to System Timing Master, stating that node was JUST_CONNECTED and is not WR-compatible.
5. If the node replied with proper WRP_INVITE_RESPONSE:

- WRP_ACK packet is sent to node,
- WRP_INVITE_RESPONSE packet received from node is forwarded to System Timing Master. Master replies with WRP_ACK. If there is no response, switch retransmits the packet (max. 5 times)
- WRP_REPORT_NODE packet is issued to STM with proper node state. Master replies with WRP_ACK. If there is no response, switch sends WRP_REPORT_NODE message again (max. 5 times)

6. Switch starts performing delay measurements.

10.2. Delay measurement

Delay measurement is done using PTPv2 peer delay measurement mechanism:

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. Master commands slave to change the phase shift value using PTPv2 signalling message with SET_PHASE_SHIFT TLV
4. Points (1-4) are repeated with different values of phase shift to determine phase shift introduced by link
5. Master calculates link delay and asymmetry and reports it to node using WRP_REPORT_DELAY message.

10.3. Node reporting

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

1. Node periodically issues WRP_REPORT_NODE message to switch. Switch responds with WRP_ACK. If there is no reply, WRP_REPORT_NODE is sent again (up to 5 times).
2. If the switch received proper WRP_REPORT_NODE message, it just forwards it to System Timing Master.
3. If there is no response, but carrier is present, switch issues WRP_REPORT_NODE packet with NO_RESPONSE state.
4. If carrier is lost, switch issues WRP_REPORT_NODE packet with DISCONNECTED state.
5. STM acknowledges reception of WRP_REPORT_NODE by sending WRP_ACK. If there is no response from STM, reporting packet is sent again (up to 5 times)

10.4. Operation with non-WR compliant nodes

WR protocol was meant to be compatible with non-WR nodes/network segments. Incompatible node is detected when the new node hadn't respond for invitation message with proper WRP_INVITE_RESPONSE. If such situation occurs, the switch:

- disables deterministic HP frame routing for incompatible node. HP frames delivered for the node are treated like normal frames to prevent fragmentation.
- enables on-the-fly fragmented frame reconstruction on port to which incompatible node is connected.

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

10.5. Link idle

For timing transmission, WR network uses reference clock embedded into serial Ethernet data stream. Therefore, PLLs in timing receivers must be locked for the whole time and there is no such thing like link idle state. If there is no data to send, transmitters shall send either commas (K28.5) or link idle sequences (K28.5/D5.6).

11. Changelog

2008/10/22	– bugfixes (interframe gap, minimum payload) – explained addressing /routing of HP frames – full PTP compatibility for non-WR devices – described network behaviour for non-WR compatible nodes
2008/10/17	Added link failure and alternate uplink paths support
2008/10/15	Initial release