TSN Tester 3.0

Design

(version 1.0)

OpenTSN Open Source Project Team

May 2021

Version history

Version revision time revision	content revisioner 1.0		file identification
2021.5 First edition preparation			TSN Tester 3.0

content

1 Overview

TSN tester is a data adaptation node applied between TSN networks.

It is to inject application data into TSN according to traffic transmission requirements and TSN network encapsulation format

Data transmission in the network, and encapsulation and return of data received from the TSN network

application and perform related performance statistics. Specifically including software configuration capabilities, IEEE 1588

Clock synchronization capability, generating and sending time-sensitive streams at deterministic points in time, time-aware shaping

(TAS), time-sensitive flow/non-time-sensitive flow concurrency, network traffic capture and analysis, etc.

2. Overall design

2.1. Overall Architecture

The overall block diagram of the hardware logic of the TSN tester is shown in Figure 2-1. According to the TSN test

The functional requirements of the tester are mapped to the interface application (respectively for control, data output, data

According to the input, flow sampling), see Table 2-1 for details.

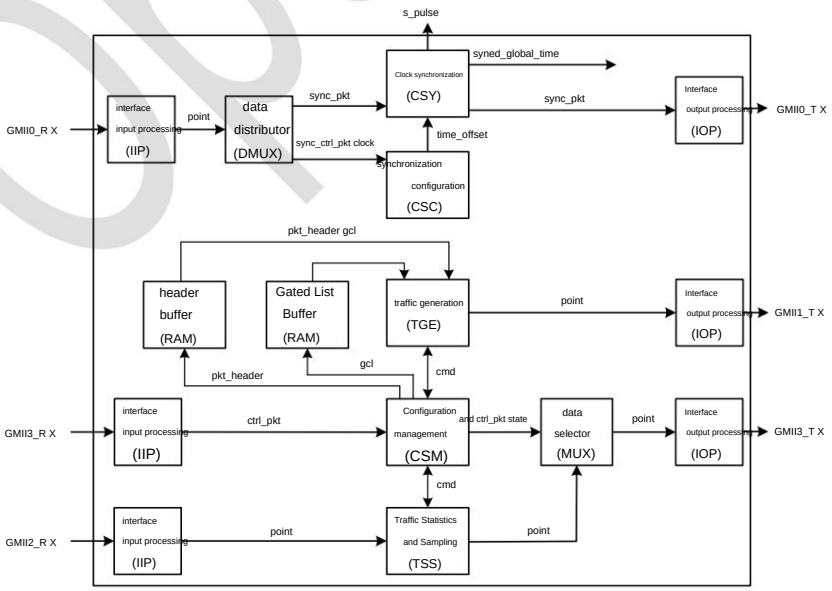

Figure 2-1 Overall architecture block diagram

Table 2-1 GMII interface description

GMII junction		
GMII0	RX	Description Responsible for receiving PTP packets, PTP encapsulation packets, and configuration packets (configure time synchronization related parameters).
	TX is responsible for sending PTP packets and PTP encapsulation packets	
GMII1	RX signal floating	
	TX is responsible for sending the test message generated by the tester	
GMII2	RX is responsible for receiving test packets returned from the network	
	TX signal output is fixed value 0	
GMII3	RX is responsible for receiving configuration packets (feature parameters of configuration test packets).	
	TX is responsible for sending sampling messages and status reporting messages	

IIP (Interface Input Process) module: responsible for discarding

Receive the preamble, frame start, and CRC of the message, and change the message transmission clock domain from GMII

The receiving clock domain is switched to the internal clock domain of the architecture to record the receiving time of the time synchronization message.

Convert the message bit width.

DMUX (Demultiplexer, data distributor) module: responsible for the Ethernet class

Distributes packets by type, and assigns time synchronization packets and time synchronization encapsulated packets to time

Synchronization module, the configuration message (configuration time synchronization related parameters) is dispatched to the clock synchronization configuration

module.

CSY (Clock Synchronization, time synchronization) module: responsible for recording time synchronization

Receive timestamp of step message, calculate transparent clock and update transparent clock domain, synchronize time

packets are encapsulated into TSMP (Time Sensitive Management Protocol) frames, and

Decapsulate time synchronization encapsulated packets into time synchronization packets, and record the transmission of time synchronization packets.

send time stamp, and correct the local time according to the received deviation value between the local clock and the master clock

bell.

CSC (Clock Synchronization Configurate, clock synchronization configuration) mode

Block: responsible for parsing configuration packets (configuring related parameters of clock synchronization), completing the synchronization of the clock

Configuration of step related parameters.

CSM (Configuration and State Management)

Module: responsible for parsing configuration packets (configuring the characteristic parameters of test packets), completing the local

Configuration of registers and tables; and periodic reporting of local status information.

TGE (Traffic Generate, traffic generation) module: based on token bucket mechanism, time

Perceptual shaping (TAS) to achieve the generation and transmission of multiple streams, in which the token bucket mechanism is used to

Control the traffic sending rate, and use time-aware shaping to control the traffic sending time.

TSS (Traffic Statistic and Sample, traffic statistics and sampling) module: responsible for

Count the number of each traffic received by the TSN tester based on quintuple matching (with mask)

and the total number of received messages, extract the quintuple of received messages and store them in metadata, and then press

The received packets are encapsulated and sampled at a certain sampling frequency.

MUX (Multiplexer, data distributor) module: responsible for reporting the NMAC status

The message and the sampled message are selected and output.

IOP (Interface Output Process, interface output processing) module: responsible for real

Now the message bit width is converted, the frame preamble, frame start and CRC are added, and the time synchronization is calculated.

The transparent clock of the message transmitted in the tester and the transparent clock domain are updated, and the time when the message is transmitted is updated.

The clock domain is switched from the internal processing clock domain to the clock domain of the external PHY.

2.2 Frame processing flow

2.2.1 Analysis and Processing Flow of Clock Synchronization Configuration Frame

The clock synchronization configuration frame is input through the GMII0 interface, and the processing mode is input through the GMII3 interface.

The block first crosses the clock domain for the frame (from the GMII receive clock domain to the HCP internal logic work

Clock domain) conversion, then convert the bit width of each frame data from 8bit to 134bit, and

Add two shots of metadata, and the first shot of metadata carries the frame length and the number of frames received by the interface

timestamp and then output the frame to the data distributor module.

The data distributor module is 0xff01 and its subtypes are 0x02 according to the frame ether type

Determine that the frame is a clock synchronization configuration frame (configure the clock synchronization parameters), and then synchronize the clock

The configuration frame is assigned to the clock synchronization configuration block.

The clock synchronization configuration module parses the received clock synchronization configuration frame, according to the clock

Synchronize the number of configuration data carried in the configuration frame and the configuration base address, and write the configuration data into

corresponding register.

2.2.2 Test parameter configuration frame parsing processing flow

The test parameter configuration frame is input by the GMII3 interface, and the processing mode is input in the GMII3 interface.

The block first crosses the clock domain for the frame (from the GMII receive clock domain to the HCP internal logic work

Clock domain) conversion, then convert the bit width of each frame data from 8bit to 134bit, and

Add two shots of metadata, and the first shot of metadata carries the frame length and the number of frames received by the interface

timestamp and then output the frame to the data distributor module.

The data distributor module is 0xff01 according to the frame ether type and its subtype is 0x10

Or 0x20 to determine that the frame is a test parameter configuration frame (configure the characteristic parameters of the test packet), and then

Then assign the test parameter configuration frame to the configuration and state management module.

The configuration and status management module parses the received test parameter configuration frame.

The Ethernet type is 0xff01 and its subtype is 0x10, then the test parameter configuration frame is used to configure

Set the packet headers of 8 types of traffic, and write the packet headers of the 8 types of traffic into the buffer area

If the frame ether type is 0xff01 and its subtype is 0x20, the test

The test parameter configuration frame is used to configure the gate control list, the length and transmission rate of the test packet, and the required system

The metered flow quintuple information, etc. After the configuration of the test parameter configuration frame is completed, the configuration and status

The management module gives a configuration completion signal.

2.2.3 Processing Flow of Status Report Frame

The configuration and status management module will output a report request every time it receives a report pulse.

Ask the data selector, and start transmitting the report message to the data selector after receiving the response.

Data selector, reporting local status (see Table D-1 for details), including configurable registers, status

status register.

When the data selector detects a report request, it will pass it to the configuration and

The state management module responds, and then starts to receive and transmit the configuration and state management module

The incoming message is sent to the interface output processing module.

The interface output processing module converts the bit width of the test message from 134bit to 8bit, and

Switch the message transmission across the clock domain from the tester's internal logic working clock domain to GMII for transmission

clock domain, and then data is output from the GMII3 interface.

2.2.4 PTP frame encapsulation processing flow

The PTP frame is input by the GMII0 interface, and the input processing module of the interface first crosses the frame.
Clock domain (from GMII receiving clock domain to master clock internal logic working clock domain) conversion,
Next, after converting the bit width of each shot of the frame from 8bit to 134bit, add two shots
metadata, which carries the frame length and the timestamp of the frame received by the interface in the first metadata.
The frame is then output to the data distributor module.

The data distributor module determines that the frame is a PTP frame according to the frame Ethernet type of 0x98f7.
Then assign to the clock synchronization module. The clock synchronization module sends the PTP message to the module's
Subtract the timestamp recorded by PTP on the interface to get the PTP packet transmitted in the tester
The transparent clock is added to the transparent clock domain, and the PTP packet is added to the
The time stamp of arriving this module is recorded in the payload of the PTP message; the PTP message is encapsulated
In the TSMP frame, it is transmitted to the interface output processing module.

The interface output processing module converts the bit width of the status report message from 134bit to 8bit.
And switch the message transmission across the clock domain from the internal logic working clock domain of the tester to the GMII
The clock domain is sent, and then the data is output from the GMII0 interface.

2.2.5 PTP encapsulation frame decapsulation process flow

The PTP encapsulated frame is input by the GMII0 interface, and the input processing module of the interface first processes the frame.
Row across clock domains (from the GMII receive clock domain to the master clock's internal logic operating clock domain)
Next, after converting the bit width of the data per shot of the frame from 8bit to 134bit, add two shots
metadata, which carries the frame length and the timestamp of the frame received by the interface in the first metadata.
The frame is then output to the data distributor module.

The data distributor module determines that the frame ether type is 0xff01 and the subtype is 0x5

The frame is a PTP encapsulated frame and then distributed to the clock synchronization module. Clock synchronization module removed

The TSMP Ethernet header of the PTP-encapsulated frame, decapsulates the PTP-encapsulated frame into a PTP frame, and

And record the timestamp of the PTP frame output of this module and store it in the payload.

The interface output processing module subtracts the time stamp of the arrival of the PTP frame from the PTP frame from the

The time stamp output by the clock synchronization module is the transparent time when the PTP message is transmitted in the tester.

clock, and accumulate the transparent clock into the transparent clock domain, and then change the bit width of the PTP packet by

134bit is converted to 8bit, and the transmission of the message across the clock domain is performed by the internal logic of the tester

The clock domain is switched to the GMII transmit clock domain, and then the data is output from the GMII0 interface.

2.2.6 Test Packet Sending Processing Flow

When the configuration and status management module completes the configuration of the test message and the tester clock has

After synchronizing with the network clock, the traffic generation module controls the flow rate of each traffic based on the token bucket mechanism.

Sending rate, based on gating list to control the sending time of traffic; scheduling policy according to priority

omitted, to schedule 8 flows, and report them in the report according to the configured packet length of each flow.

Fill the corresponding data 0 at the end of the header, and then output the message to the interface output processing module. exist

During the flow scheduling and sending process, the tester controller can update the packet headers of 8 flows,

After the traffic generation module detects the message header update completion signal, it will schedule and send the updated message.

header.

The interface output processing module converts the bit width of the test message from 134bit to 8bit, and

Switch the message transmission across the clock domain from the tester's internal logic working clock domain to GMII for transmission

clock domain, and then data is output from the GMII1 interface.

2.2.7 Test message reception processing flow

The test message returned to the tester from the network is input through the GMII2 interface, and the GMII2

The interface receiving and processing module first crosses the clock domain for the message (from the GMII receiving clock domain to the measurement

The internal logic working clock domain of the tester is converted, and then the bit width of the data per shot of the frame is changed from 8bit

After converting to 134bit, add two shots of metadata, and carry the frame length in the first shot of metadata

and the timestamp of the frame received by the interface, and then output the frame to the traffic statistics and sampling module.

The traffic statistics module receives the test packet, extracts the quintuple information of the packet, and compares it with the test packet.

The quintuple configured by the tester controller is matched (with mask), and each traffic received is counted

The number of packets received; at the same time, the total number of packets received from the GMII2 interface is counted, and

When encapsulating the packet, it will detect whether the encapsulated packet exceeds the

1514B (excluding the CRC of 4B), and discard the data beyond 1514B to ensure

The length of the encapsulated message does not exceed 1514B; then the received message is processed according to the configured sampling frequency.

The sampled message is sampled, and the sampled message is output to the data selector.

The data selector first writes the sampled packets into the fifo for caching, and then checks the configuration and status management

When the management module has no report request and no message transmission, the message in the fifo will be read out and output to the receiver.

output processing module.

The interface output processing module converts the bit width of the test message from 134bit to 8bit, and

Switch the message transmission across the clock domain from the tester's internal logic working clock domain to GMII for transmission

clock domain, and then data is output from the GMII3 interface.

Appendix **A** Packet Data Structure Definition

The grouping in Tester 3.0 includes two parts: Metadata header and valid data grouping.

As shown in Figure A-1, Metadata is carried in the first 32 bytes of the packet, and each packet is

The first 16 bytes in the internal logic of the tester is Metadata0, and the second data is

Metadata1ÿ

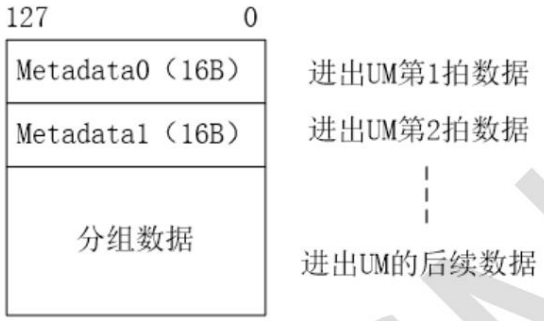

Figure A-1 Packet Data Transmission Format

The first two beats of the packet are the 32-byte metadata added by the interface input processing module.

The data after two beats is valid packet data. 134-bit data is identified by 2-bit header and tail, 4

The number of invalid bytes, consisting of 128 bits of valid data.

Among them, the [133:132] bits are the head and tail identifiers of the message data, 01 represents the header of the message, and 11

Represents the middle data of the message, 10 represents the tail of the message; bits [131:128] are 4-bit invalid words

The number of sections, where 0000 means all 16 bytes are valid, and 0001 means the lowest byte

Invalid, the highest 15 bytes are valid, and so on, 1111 means the lowest 15 bytes are invalid,

The highest byte is valid. The format is shown in Figure A-2.

Figure A-2 Packet Transmission Format

Appendix **B TSMP** message format design

TSMP (Time Sensitive Management Protocol) is a TSN network centralized controller for network topology

Protocol for flutter detection, node configuration in the network, and packet encapsulation.

ÿTSMP frame format

The format design of TSMP frame is shown in Figure B-1.

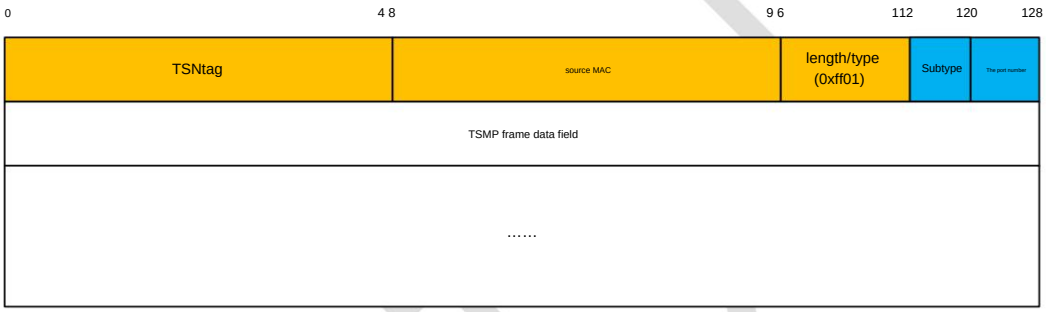

Figure B-1 TSMP frame format

The yellow and blue fields in the figure are the TSMP packet headers, and the yellow field is

TSMP packet Ethernet header; the white field is the TSMP packet data field. TSMP header

Refer to Table B-1 for the meaning of each field in Table B-1, and Table B-2 for the TSMP packet types.

Table B-1 Meaning of each field in the TSMP header

field	Bit width description	
TSNtag	48	The result of mapping the TSMP message.
source mac	48 Not used yet	
length/type	16 TSMP message type is 0xff01 (custom)	
Subtype	8	Used to identify different types of TSMP packets, currently including 6 types Type: ARP encapsulated message, chip reported Beacon message, HCP Report Beacon message, tester report Beacon message, chip Configuration message, HCP configuration message, Tester configuration message, PTP encapsulated message
The port number	8	The input port number of the first hop of the packet (this field is filled by HCP) Or the output port number of the packet at the last hop (controlled by the centralized to fill this field)

Table B-2 TSMP packet types

value of the subtype of the packet type		Meaning
ARP encapsulated packet	8'h0	ARP packets are encapsulated into TSMP packets for transmission in the network, and the ARP packets are completely stored in the TSMP data domain
Chip report Beacon message	8'h1	switches and network cards to report status packets to the controller, and the status packets of switches and network cards are completely reported. The packets that are stored in the TSMP data domain controller to configure switches
Chip configuration message	8'h2	and network cards, the controller encapsulates the NMAC configuration packets into TSMP packets, and the NMAC configuration packets are completely stored in the TSMP data domain controller to configure the HCP. Message; configuration
HCP configuration message	8'h3	information is stored in the TSMP data field.
HCP escalation Beacon message	8'h4	The status information reported by HCP is stored in the TSMP data field
PTP encapsulated packet	8'h5	Encapsulate PTP messages (sync message, delay_req message, delay_resp message) into TSMP message, in which the PTP message is completely stored in the TSMP data domain. In the configured packet, the packet header information
Tester configuration message	8:10 a.m.	of the 8 flows is stored in the TSMP data field The
	8'h20	tester controller configures the tester registers and gate control list. The registers and gate control list are stored in the TSMP data field.
Tester report Beacon message	The status information reported by the 8'h30 tester is stored in the TSMP data field	

Appendix **C** Tester Configuration Message Format Design

This part introduces the tester configuration message (specifically refers to the characteristic parameters of the configuration test message) grid style detailed design.

In order to facilitate the tester controller to dynamically update 8 types of messages during the test header, which further subdivides the tester configuration message into configurations for updating 8 types of message headers messages and configuration messages for updating gate lists, configurable registers, both testers Configuration packets are distinguished by subtypes of TSMP packets, see Table B-2 for details.

ÿ Configure the packet format of 8 types of packet headers

8 types of headers received in the configuration and status management module

The formula is shown in Figure C-1, and the TSMP data field division of the configuration packet is shown in Table C-1.

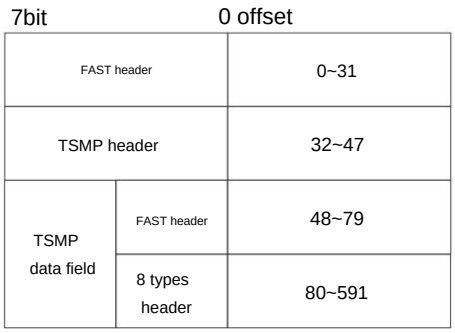

Figure C-1 TSMP packet format for configuring eight types of packet headers

Table C-1 Configuration of TSMP packet data fields for eight types of packet headers

Module Position (beat) Field		Signal Name		meaning
/	4~5	[127:0]		FAST head
header buffer	6~9	[127:0]		Type 1 header
	10~13	[127:0]		Type 2 header
	14~17	[127:0]		Type 3 header
	18~21	[127:0]		Type 4 header
	22~25	[127:0]		Type 5 header
	26~29	[127:0]		Type 6 header
	30~33	[127:0]		Type 7 header
	34~37	[127:0]		Type 8 header

ÿ The message format of the configuration gate list and configurable registers

The configuration gate list and configurable registers received in the configuration and state management module

The message format of the configuration message is shown in Figure C-2, and the TSMP data field division of the configuration message is shown in Table C-2

shown.

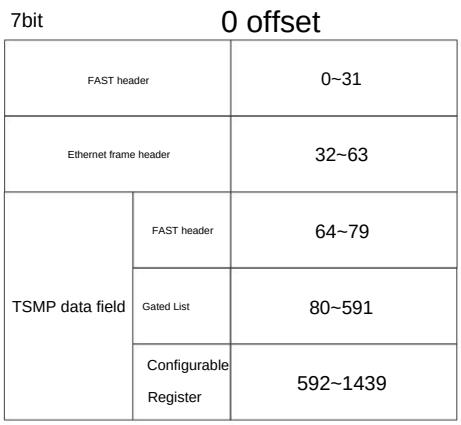

Figure C-2 Message Format for Configuration Gating List and Configurable Registers

Table C-2 TSMP message data field division of configuration gate list and configurable registers

module	Location (shoot)	field	Signal name	meaning
/	4~5 [127:0]			FAST head
gating list buffer	6~37 [127:0]			A list of gated cycles containing 16 slots for 8 types of messages gated state
traffic generation into modules	38	[32]	test_stop	test end signal
		[31:0]	One time slot size for gcl_time_slot_cycle gate list	8
	39	[95:80]	tb3_size	The token corresponding to the third type of message barrel depth
		[79:64]	tb3_rate	Report to the third type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[63:48]	tb2_size	The token corresponding to the second type of message barrel depth
		[47:32]	tb2_rate	Report to the second type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[31:16]	tb1_size	The token corresponding to the first type of message barrel depth
		[15:0]	tb1_rate	Report to the first type every other slot The command added to the token bucket corresponding to the text Number of cards
	40	[95:80]	tb6_size	Token corresponding to the sixth type of message barrel depth
		[79:64]	tb6_rate	Report to the 6th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[63:48]	tb5_size	The token corresponding to the fifth type of message barrel depth
		[47:32]	tb5_rate	Report to the 5th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[31:16]	tb4_size	The token corresponding to the fourth type of message barrel depth
		[15:0]	tb4_rate	Report to the 4th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
	41	[63:48]	tb8_size	Token corresponding to the eighth type of message barrel depth

module	Location (shoot)	field	Signal name	meaning
/	4~5 [127:0]			FAST head
		[47:32]	tb8_rate	Report to the 8th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[31:16]	tb7_size	The token corresponding to the seventh type of message barrel depth
		[15:0]	tb7_rate	Report to the 7th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
	42	[127:112]	pkt_8_len	The number of bytes of the 8th type of message (not including CRC of 4B)
		[111: 96]	pkt_7_len	The number of bytes of the 7th type of message (not including CRC of 4B)
		[95:80]	pkt_6_len	The number of bytes of the 6th type of message (not including CRC of 4B)
		[79:64]	pkt_5_len	The number of bytes of the 5th type of message (not including CRC of 4B)
		[63:48]	pkt_4_len	The number of bytes of the 4th type of message (not including CRC of 4B)
		[47:32]	pkt_3_len	Number of bytes of type 3 message (not including CRC of 4B)
		[31:16]	pkt_2_len	Number of bytes of type 2 message (not including CRC of 4B)
		[15:0]	pkt_1_len	Number of bytes of type 1 message (not including CRC of 4B)
	43	reserve		
Traffic statistics planning and mining sample module	44 [103:0]		{src_ip_1,dst_ip_1, protocol_1,src_port_1 dst_port_1}	quintuple of type 1 message
	45	[103:0]	mask_1	The quintuple mask of the first type of message code
	46 [103:0]		{src_ip_2,dst_ip_2, protocol_2,src_port_2 dst_port_2}	quintuple of type 2 message
	47 [103:0]	mask_2		Quintuple mask for type 2 message code
	48	[103:0]	{src_ip_3,dst_ip_3, protocol_3,src_port_3 ,dst_port_3}	The quintuple of the third type of message

module	Location (shoot)	field	Signal name	meaning
/	4~5 [127:0]		.	FAST head
	49 [103:0]	mask_3		The quintuple mask of the third type of message code
	50	[103:0]	{src_ip_4,dst_ip_4, protocol_4,src_port_4 . dst_port_4}	quintuple of type 4 message
	51	[103:0]	mask_4	Five-tuple mask for type 4 message code
	52	[103:0]	{src_ip_5,dst_ip_5, protocol_5,src_port_5 . dst_port_5}	quintuple of type 5 message
	53	[103:0]	mask_5	The quintuple mask of the fifth type of message code
	54	[103:0]	{src_ip_6,dst_ip_6, protocol_6,src_port_6 . dst_port_6}	Quintuple of type 6 message
	55	[103:0]	mask_6	The quintuple mask of the sixth type of message code
	56	[103:0]	{src_ip_7,dst_ip_7, protocol_7,src_port_7 . dst_port_7}	quintuple of type 7 message
	57	[103:0]	mask_7	The quintuple mask of the seventh type of message code
	58	[103:0]	{src_ip_8,dst_ip_8, protocol_8,src_port_8 ,dst_port_8}	The quintuple of the eighth type of message
	59	[103:0]	mask_8	The quintuple mask of the eighth type of message code
	60	reserve		
	61	[15:0]	samp_freq Note:	Used to control the frequency of reading messages

8 types of message lengths (number of bytes) do not include the length of 2 beats of metadata.

Appendix **D** Design of the format of the **Beacon** message reported by the tester

The tester will periodically report local status information to the tester controller.

The format of the Beacon message reported by the tester generated by the state management module is shown in Figure D-1.

Table D-1 shows the division of the TSMP data field of the Beacon message.

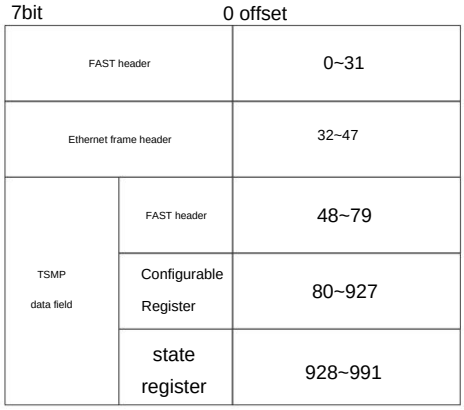

Figure D-1 The format of the Beacon packet reported by the tester

Table D-1 TSMP data field division of the Beacon packet reported by the tester

module	Location (shoot)	field	Signal name	meaning
PGM (THAT)	6	[32]	test_stop	Test end signal. 0: the test starts; 1: the test ends
		[31:0]	gcl_time_slot_cycle	One slot size of the gated list
	7	[95:80]	tb3_size	The token corresponding to the third type of message barrel depth
		[79:64]	tb3_rate	Report to the third type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[63:48]	tb2_size	The token corresponding to the second type of message barrel depth
		[47:32]	tb2_rate	Report to the second type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[31:16]	tb1_size	The token corresponding to the first type of message barrel depth
		[15:0]	tb1_rate	Report to the first type every other slot The command added to the token bucket corresponding to the text Number of cards
	8	[95:80]	tb6_size	Token corresponding to the sixth type of message barrel depth
		[79:64]	tb6_rate	Report to the 6th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[63:48]	tb5_size	The token corresponding to the fifth type of message barrel depth

module	Location (shoot)	field	Signal name	meaning
		[47:32]	tb5_rate	Report to the 5th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[31:16]	tb4_size	The token corresponding to the fourth type of message barrel depth
		[15:0]	tb4_rate	Report to the 4th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
	9	[63:48]	tb8_size	Token corresponding to the eighth type of message barrel depth
		[47:32]	tb8_rate	Report to the 8th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
		[31:16]	tb7_size	The token corresponding to the seventh type of message barrel depth
		[15:0]	tb7_rate	Report to the 7th type every 1 slot The command added to the token bucket corresponding to the text Number of cards
	10	[127:112]	pkt_8_len	The number of bytes of the 8th type of message (packet CRC including 4B)
		[111: 96]	pkt_7_len	The number of bytes of the 7th type of message (packet CRC including 4B)
		[95:80]	pkt_6_len	The number of bytes of the 6th type of message (packet CRC including 4B)
		[79:64]	pkt_5_len	The number of bytes of the 5th type of message (packet CRC including 4B)
		[63:48]	pkt_4_len	The number of bytes of the 4th type of message (packet CRC including 4B)
		[47:32]	pkt_3_len	The number of bytes of the third type of message (packet CRC including 4B)
		[31:16]	pkt_2_len	Number of bytes of type 2 message (packet CRC including 4B)
		[15:0]	pkt_1_len	The number of bytes of the first type of message (packet CRC including 4B)
	11	reserve		
FSM (THAT)	12	[103:0]	{src_ip_1,dst_ip_ 1, protocol_1, src_port_1, dst_port_1}	quintuple of type 1 message
	13	[103:0] mask_1		The quintuple mask of the first type of message

module	Location (shoot)	field	Signal name	meaning
				code
	14	[103:0]	{src_ip_2,dst_ip_2, protocol_2, src_port_2, dst_port_2}	quintuple of type 2 message
	15	[103:0] mask_2		Quintuple mask for type 2 message code
	16	[103:0]	{src_ip_3,dst_ip_3, protocol_3, src_port_3, dst_port_3}	The quintuple of the third type of message
	17	[103:0] mask_3		The quintuple mask of the third type of message code
	18	[103:0]	{src_ip_4,dst_ip_4, protocol_4, src_port_4, dst_port_4}	quintuple of type 4 message
	19	[103:0] mask_4		Five-tuple mask for type 4 message code
	20 [103:0]		{src_ip_5,dst_ip_5, protocol_5, src_port_5, dst_port_5}	quintuple of type 5 message
	21	[103:0] mask_5		The quintuple mask of the fifth type of message code
	22 [103:0]		{src_ip_6,dst_ip_6, protocol_6, src_port_6, dst_port_6}	Quintuple of type 6 message
	23 [103:0] mask_6			The quintuple mask of the sixth type of message code
	24 [103:0]		{src_ip_7,dst_ip_7, protocol_7, src_port_7, dst_port_7}	quintuple of type 7 message
	25 [103:0] mask_7			The quintuple mask of the seventh type of message code
	26 [103:0]		{src_ip_8,dst_ip_	quintuple of the 8th type of message

module	Location (shoot)	field	Signal name	meaning
			8, protocol_8, src_port_8, dst_port_8}	
	27 [103:0]	mask_8		The quintuple mask of the eighth type of message code
	28	reserve		
SSM (THAT)	29 [15:0]		samp_freq	Used to control the frequency of reading messages
	30	reserve		
PGM (SA)	31	[127:96]	pgm_pkt_4_cnt	The number of the 4th type of message sent
		[95:64]	pgm_pkt_3_cnt	The number of the third type of message sent
		[63:32]	pgm_pkt_2_cnt	The number of packets of type 2 sent
		[31:0]	pgm_pkt_1_cnt	The number of sent packets of the first type
	32	[127:96]	pgm_pkt_8_cnt	The number of the 8th type of message sent
		[95:64]	pgm_pkt_7_cnt	The number of the 7th type of message sent
		[63:32]	pgm_pkt_6_cnt	The number of the 6th type of message sent
		[31:0]	pgm_pkt_5_cnt	The number of the 5th type of packets sent
	33	reserve		
FSM (SA)	34	[127:96]	ssm_pkt_4_cnt	Receives the number of type 4 packets
		[95:64]	ssm_pkt_3_cnt	The number of the third type of packets received
		[63:32]	ssm_pkt_2_cnt	The number of the second type of packets received
		[31:0]	ssm_pkt_1_cnt	The number of the first type of packets received
	35	[127:96]	ssm_pkt_8_cnt	Receive the number of the 8th type of message
		[95:64]	ssm_pkt_7_cnt	The number of the 7th type of messages received
		[63:32]	ssm_pkt_6_cnt	The number of the 6th type of packets received
		[31:0]	ssm_pkt_5_cnt	The number of the fifth type of packets received
	36	reserve		
SSM (SA)	37 [31:0]		pt_cnt	Number of received messages

Note:

The PGM module stamps the sending timestamp on the fourth beat [47:0] of the packet (excluding the two beats of metadata).

The length field metadata0[107:96] of the encapsulated FAST header does not contain the 2 added by the FPGA OS Take the FAST header length and the Ethernet header length.

Priority: Type 1 message > Type 2 message >…> Type 8 message.

Each shot of gated list data [127:120], ..., [7:0] corresponds to the 16th slot, ..., and the 1st slot, respectively. Maximum number of sent/received messages per unit time

$$= \frac{1\text{Gbit}}{\text{message length 64B+ Ethernet minimum frame spacing 12B+ Ethernet frame preamble 7B+ frame start character 1B)ÿ8}} = 1.488 \text{ ÿ } 106; \text{ (minimum}$$

The maximum number of 1h messages sent/received = 1.488 ÿ 106 ÿ 3600 = 5.357 ÿ 109.

2 of them ^{16} = 65536ÿ2 ^{32} = 4.3 ÿ 109ÿ

The slot value configured by the software is ~~the size of the~~ $\frac{}{8}$ time slot ; the maximum time slot is 200 ÿs.

The relationship between the rate of adding tokens to the token bucket and the time slot: add one to the token bucket every other time slot Secondary token, one token represents packet 1B, the rate required to add the number of tokens to the token

$$\text{tb\_rate=rate to be limited (bps)} \div \frac{\text{Time slot size (ns)}}{\frac{109\text{ns}}{1\text{s}}} = \frac{\text{bucket per unit time slot (bps)ÿtime slot size}}{\text{(ns) 8ÿ109}}$$

The bit width of tb_rate: the maximum time slot is 200ÿs. To support the sending of messages at the full rate of 1Gbps, the number of tokens added per unit time slot tb_rate=10243 ÿ200_000 = 26844

$$\frac{}{8\text{ÿ}109}$$

while 2^{16} = 65536, so tb_rate selects a bit width of 16 bits.

Appendix E **FAST** Packet Transmission Format Over Ethernet Connections

The FAST APP in FAST 2.0 runs locally and communicates with the FPGA through the PCIe bus

OS connected while the TSN tester is a pure FPGA design running on a remote Linux host

Run FAST APP, so you need to redesign FAST lib, and define the transmission through Ethernet

The format of the output FAST packet. As shown in Figure E-1 below.

1) The programming interface API provided by FAST lib to APP remains unchanged;

2) When the FAST packet is transmitted through Ethernet, it needs to be encapsulated into a new Ethernet frame;

3) When the FPGA OS receives the FAST packet, in addition to removing the CRC, it also sends

Before the P3-rx interface, add a FAST header;

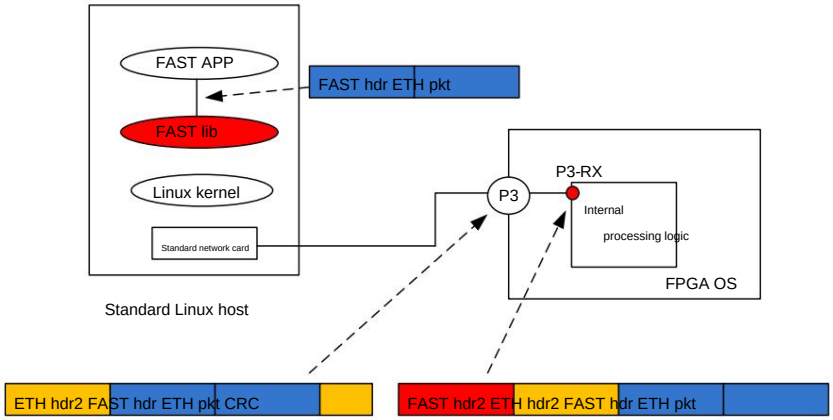

Figure E-1 FAST packet transmission format over an Ethernet connection

ETH hdr2 is defined as:

ÿÿ MACÿ0xffffffffffff

Source MAC: Don't Care

Length type field: 0xff01

The internal processing logic of the TSN tester sends a message to the external host FAST APP through P3-TX.

When sending packets, the above format is also followed.

The P3 port of the TSN tester is the interface connected to the external tester controller by default.

The internal processing logic of the TSN tester needs to seal the FAST packets entering and leaving the P3 port.

Packing and unpacking processing.

Appendix **F TSN** Tester Test Connection Diagram

The TSN tester test connection diagram is shown in Figure F-1. TSN tester controller sends

Beacon configuration packets enter the TSN tester from interface 3;

The Beacon reports packets are output from interface 3 to the TSN tester Insight; the TSN tester

Generate test packets and output them from No. 1 interface to the network under test, and then send them from No. 2 after passing through the network under test.

The port returns to the TSN tester; after the TSN tester encapsulates and samples the returned test packets,

Output from No. 3 interface to TSN tester Insight; TSN tester's No. 0 interface receives

Clock synchronization configuration packets, PTP packets, and PTP encapsulated packets transmitted from the network under test,

And output PTP packets and PTP encapsulated packets from interface 0 to the network under test.

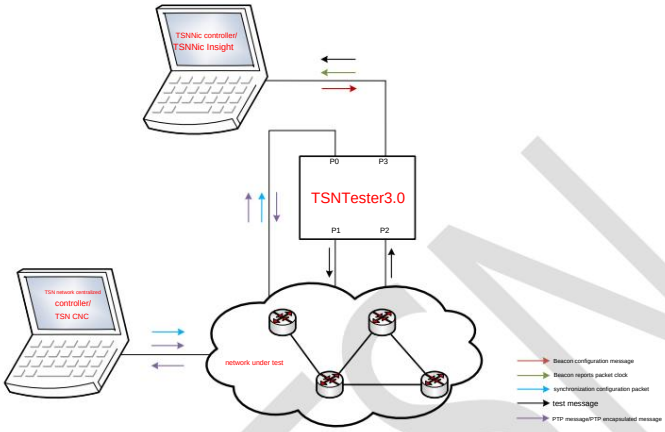

Figure F-1 TSN Tester Test Connection Diagram

Appendix **G TSN** Tester Performance

The TSN tester transmit bandwidth was tested with a commercial tester Ixia.

Test scenario: use TSN tester to send packets, the packet length is 512B (including 4B CRC),

TSN Tester with Ixia

the actual sending bandwidth. The rate value displayed by Ixia fluctuates very little, taking three values for each test

The group data is averaged to obtain the actual transmission bandwidth. The measured data are shown in Table G-1.

Table G-1 TSN Tester Bandwidth Test Data

TSN Tester Theoretical Transmission Bandwidth (bps)	Actual transmit bandwidth measured with Ixia (bps)	Difference (bps)
200,000,000	200,000,648	648
400,000,000	400,009,503	9,503
600,000,000	600,000,072	72
800,000,000	800,008,197	8,197

Data analysis: from the difference list can be obtained: TSN tester theoretical transmission bandwidth and Ixia

Measured actual send

The difference in bandwidth is less than 10,000bps.

Use the commercial tester Ixia to test the maximum transmit bandwidth and throughput of the TSN tester, and

Compare with Ixia.

testing scenarios:

The TSN tester sends packets, and Ixia is used to test that the TSN tester is sending different packet lengths.

Maximum bandwidth and throughput at 4B (including 4B CRC); values displayed by Ixia fluctuate widely

The maximum bandwidth and throughput are obtained by taking the average of three sets of data in each test;

The commercial tester Ixia sends packets by itself, and loopback tests the maximum sending bandwidth and throughput of Ixia

The measured data are shown in Table G-2 and Table G-3.

Table G-2 TSN Tester and Ixia Maximum Transmit Bandwidth Comparison

Maximum sending bandwidth (bps)	TSN Tester	Ixia	Difference (bps)
64B	761,912,625	761,905,402	7,223
128B	864,873,619	864,864,028	9,591
256B	927,545,901	927,537,389	8,512
512B	962,415,725	962,406,107	9,618
1518B	987,010,570	987,007,541	3,029

Table G-3 TSN Tester and Ixia Throughput Comparison

Throughput (pps)	TSN Tester	Ixia	Difference (pps)
64B	1,488,111	1,488,095	16
128B	844,603	844,594	9
256B	452,903	452,899	4
512B	234,965	234,964	1
1518B	81,275	81,273	2

Data Analysis: From the Difference at a Glance: Maximum Transmit Bands of TSN Tester and Ixia

The width difference is less than 10,000bps; the throughput difference between the TSN tester and Ixia is less than 20pps.