Machine Translated by Google

Design scheme of TSN switch (TSNSwitch3.2)

(version 1.2)

OpenTSN

OpenTSN Open Source Project Team

November 2021

Machine Translated by Google

Version history

Version	revision time	Revision content 2021.10.14	file identification
1.0		In order to enable the TSN switch in the OpenTSN3.2 project to Better support for forwarding of standard Ethernet packets, in TSN Add MAC self-learning and standard Ethernet packets to the switch Check the MAC table forwarding function.	OpenTSN3.2
1.1	2021.11.10	1. The MSL module optimizes the aging logic of MAC forwarding table entries Edit function; 2. The document structure has been adjusted, mainly Chapter 2	
1.2	2021.11.18	1. Organize document format	

OpenTSN

Machine Translated by Google

content

OpenTSN

Machine Translated by Google

OpenTSN

Machine Translated by Google

1. Project Overview

This document is an introduction to the TSN switch in the OpenTSN (version 3.2) open source project

The design is mainly divided into two parts: project overview and overall design.

1.1. Design Goals

TSN has the advantages of time synchronization, delay determinism, reliable transmission and management of traditional Ethernet

Control and other aspects have been enhanced, and its application scenarios have been expanded from the initial industrial Internet to

Business network, vehicle network and aerospace vehicle network, etc. With the development of network applications in different fields

Enriched and expanded, the application scenarios have shown the characteristics of diversification and differentiation. In order to meet the above

To meet the diverse and differentiated application requirements of scenarios, the OpenTSN (version 3.2) open source project is designed

TSN switches are counted. TSN switches operate by extracting appropriate subsets of the TSN standard

Design, aiming to design a set that can meet the diversification and differences of TSN networks in different fields

The architecture of the requirements.

1.2. Design Metrics

ÿ Support IEEE 802.1AS, 802.1Qch, 802.1Qbv, 802.1Qcc standards

ÿ Supports forwarding and switching of three types of traffic: time-sensitive, bandwidth reservation and best-effort forwarding

ÿ Support MAC self-learning and standard Ethernet packet forwarding

ÿ Supports 8 Gigabit Ethernet network interfaces

ÿ The hardware scheduling time slot setting range is [4us, 512us]

ÿ Time-sensitive traffic supports a maximum delay of 1024 hardware scheduling time slots

ÿSwitching capacity 16Gbps

ÿExchange delay is less than 30us

ÿ Support 16K stream forwarding configuration

OpenTSN

Machine Translated by Google

ÿ Centralized message exchange buffer management

2. Overall Design

The overall design of the TSN switch (TSNSwitch3.2) is shown in Figure 2-1.

TSNSwitch3.2 includes two parts of logic, time-sensitive switching TSS and hardware control point HCP.

Decouple data processing logic and control logic.

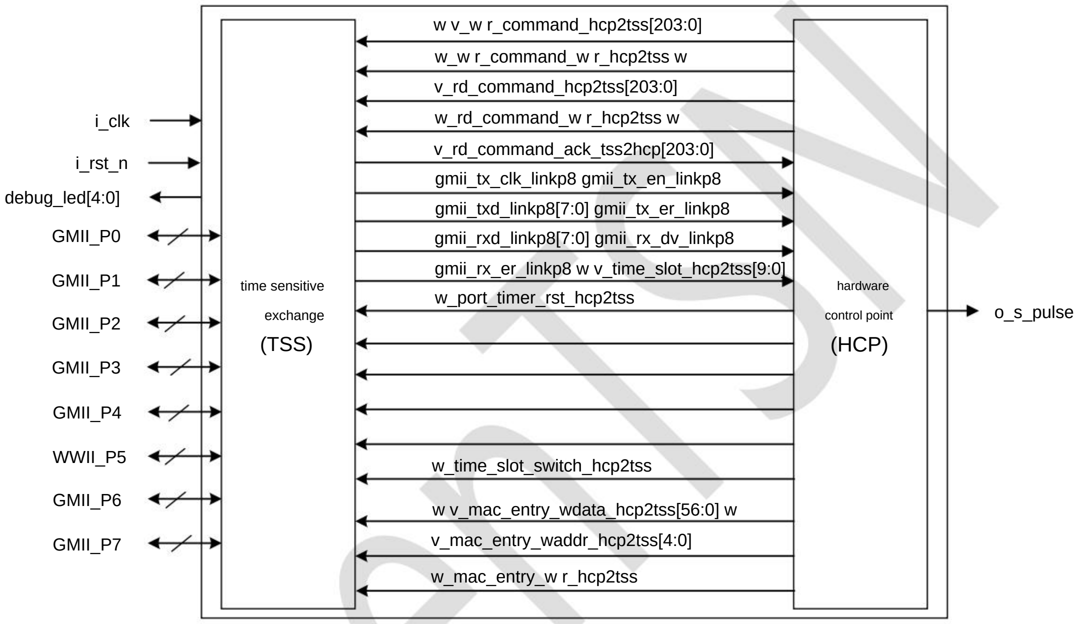

w v_w r_command_hcp2tss[203:0]
w_w r_command_w r_hcp2tss w
v_rd_command_hcp2tss[203:0]
w_rd_command_w r_hcp2tss w
v_rd_command_ack_tss2hcp[203:0]
gmii_tx_clk_linkp8 gmii_tx_en_linkp8
gmii_txd_linkp8[7:0] gmii_tx_er_linkp8
gmii_rxd_linkp8[7:0] gmii_rx_dv_linkp8
gmii_rx_er_linkp8 w v_time_slot_hcp2tss[9:0]
w_port_timer_rst_hcp2tss
w_time_slot_switch_hcp2tss
w v_mac_entry_wdata_hcp2tss[56:0] w
v_mac_entry_waddr_hcp2tss[4:0]
w_mac_entry_w r_hcp2tss
i_clk
i_rst_n
debug_led[4:0]
GMII_P0
GMII_P1
GMII_P2
GMII_P3
GMII_P4
WWII_P5
GMII_P6
GMII_P7
time sensitive
exchange
(TSS)
hardware
control point
(HCP)
o_s_pulse

Figure 2-1 Overall architecture of TSNSwitch3.2

Table 2-1 shows the meaning of the signals in the overall architecture diagram of TSNSwitch3.2.

Table 2-1 Signal meanings in the overall architecture of TSNSwitch3.2

Signal	Bit width	meaning
i_clk	1	clock signal, the clock frequency is 125MHz
i_rst_n	1	Reset signal, active low output to 5
debug_led	5	LED lights on the FPGA board, the 5 signals can be used to display the FPGA programming logic (TSN switch, TSN network card and ST SN tester, etc.) and hardware status
GMII_PN	.	N network exchange GMII interface, N is 0, 1ÿ…ÿ7
wv_wr_command_hcp2tss	204	Write command, see Appendix D for the format
w_wr_command_wr_hcp2tss	1	Write command enable signal
wv_rd_command_hcp2tss	204	Read command, see Appendix D for the format

OpenTSN

Machine Translated by Google

Signal	Bit width	meaning
w_rd_command_wr_hcp2tss	1	Read command enable signal
wv_rd_command_ack_tss2hcp	204	Read command response signal, see Appendix D for the format
gmii_tx_clk_linkp4	1	GMII clock signal
gmii_tx_en_linkp4	1	GMII data valid signal
gmii_txd_linkp4	8	GMII number
gmii_tx_er_linkp4	1	GMII data error signal
gmii_rxd_linkp4	8	GMII number
gmii_rx_dv_linkp4	1	GMII data valid signal
gmii_rx_er_linkp4	1	GMII data error signal
wv_time_slot_hcp2tss	10	within one cycle of the current global clock time slot ID
w_timer_rst_hcp2tss	1	19bit clock reset signal
w_time_slot_switch_hcp2tss	1	Time slot switching signal input
o_s_pulse	1	second pulse or millisecond pulse (configurable)
wv_mac_entry_wdata_hcp2tss	57	Cache RAM write data of MAC forwarding table
wv_mac_entry_waddr_hcp2tss	5	Cache the RAM write address of the MAC forwarding table
w_mac_entry_wr_hcp2tss	1	RAM write enable for cache MAC forwarding table, Highly effective

TSS (Timing Sensitive Switch, Time Sensitive Switch) module: The main function is Exchange TSN packets and standard Ethernet packets; shape traffic based on the IEEE 802.1Qch/IEEE 802.1 Qbv scheduling model; and calculate the transmission of PTP packets in TSS The input time is the transparent clock, and the transparent clock is accumulated into the transparent clock domain of the PTP packet. middle.

HCP (Hardware Control Point, hardware control point) module: the main function is Encapsulate PTP packets/status packets, and perform configuration encapsulation packets/PTP encapsulation packets. Decapsulation; parses configuration packets and generates configuration control registers, forwarding tables, gates Write/read commands of control table and mapping table, collect TSN switch status and report it periodically; The timestamp is recorded in the PTP packet, and the time when the PTP packet is transmitted in the HCP is calculated, that is, Transparent clock, and accumulate the transparent clock into the transparent clock domain of the PTP packet; and benchmarking The MAC address of the quasi-Ethernet message is self-learning.

OpenTSN

Machine Translated by Google

2.1. Overall design of TSS

2.1.1. Internal function division of TSS

The internal function division of time-sensitive switching TSS is shown in Figure 2-2, and the modules are described below.

function is described.

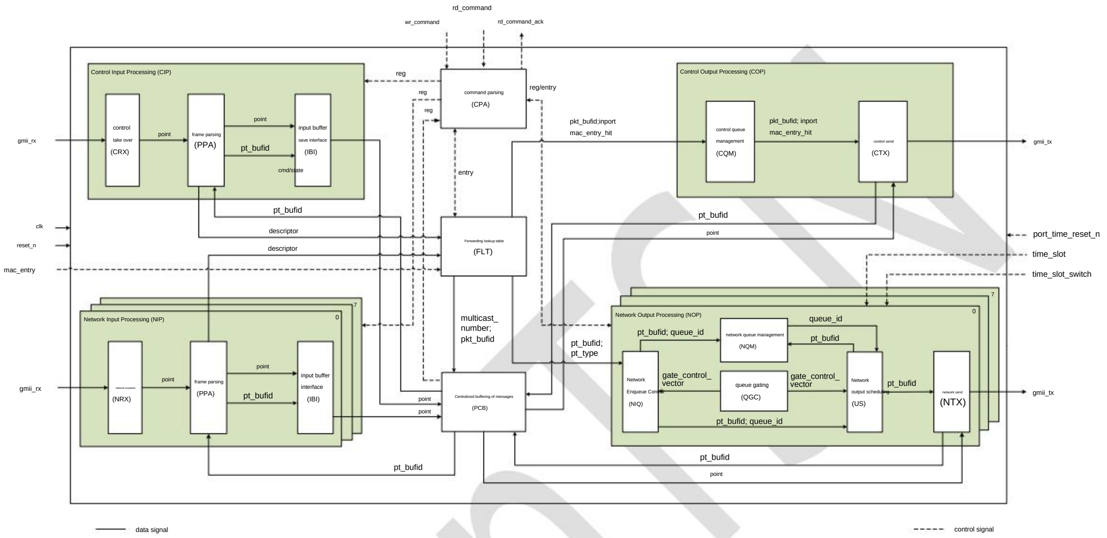


Figure 2-2 Internal function division of switching logical TSS

Table 2-2 shows the meanings of the signals in the time-sensitive switching TSS internal function division diagram.

Table 2-2 Signal meanings in the overall architecture of TSS

Signal	Bit width	meaning
pkt_134 (message centralized buffer receiving or sending signal sent)	134	Message body data, please refer to Appendix A for the specific format
pkt_9 (signal of non-message centralized cache module ÿÿ	9	Message body data, please refer to Appendix A for the specific format
descriptor	72	Message descriptor data, used to identify message data
pt_bufid	9	ID of the message data cached in the message buffer No
pt_type	3	Message Type Data
flow_id	14	The flow ID of the packet data, used to identify the flow
lookup_en	1	Lookup table enable
outport	9	Output port number (bitmap)
import	4	Enter the port number
queue_id	3	queue_id number, used to identify the ID of the queue cache

Machine Translated by Google

Signal	Bit width	meaning
		No
multicast_ count	4	Number of output ports for multicast packets
gate_ctrl_vector	8	gate signal
forward table	14	16K forwarding entries
mac_entry_hit	1	Check whether the MAC forwarding table hits the signal
mac_entry	-	MAC forwarding table
port_time_rst_n	1	port clock reset signal
time_slot	11	Time slot signal
time_slot_switch	1	Time slot switching signal
wr_command	204	Write command
rd_command	204	Read command
rd_command_ack	204	Read Command Response

OpenTSN

The entire architecture is now divided into five major parts of logic: network input processing logic, network input Output processing logic, control input processing, control output processing, and internal processing logic, the following The five parts of logic are introduced one by one.

2.1.1.1. Network Input Processing Logic

The internal block diagram of the network input processing logic is shown in Figure 2-3.

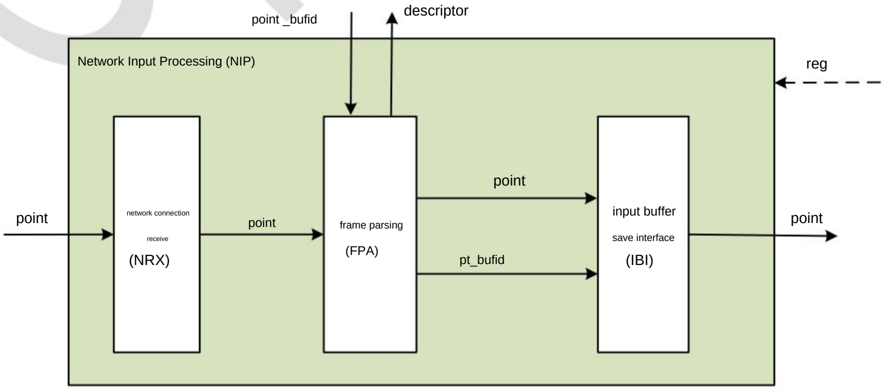


Figure 2-3 Internal function division of NIP module

Machine Translated by Google

NRX (Network RX, Network Receive) module: The main function is to receive the network interface

For incoming messages, switch the message transmission clock domain from the receiving clock domain of the GMII interface to the TSS

Internal clock domain; record the time when the module receives the time synchronization message and store it in the

In TSNTag; identify standard Ethernet packets and TSN packets and

segment to decide whether to transmit the message: if the TSS is in the initialization phase, discard all the messages;

If the TSS is in the configuration stage, it transmits standard Ethernet packets and configuration packets in TSN packets.

If the TSS is in the clock synchronization stage, it transmits standard Ethernet packets and TSN packets.

non-time-sensitive messages; if the TSS is in the normal working phase, it transmits standard Ethernet messages.

messages and TSN messages.

FPA (Frame Parse, frame parsing) module: The main function is to centrally cache received messages

The pkt_bufid allocated by the module converts the data bit width of each shot of the message from 9bit to 134bit, and

Output the bit-width converted message and pkt_bufid to the input buffer interface module; according to the connection

The received message constructs a message descriptor and outputs it to the forwarding table lookup module for table lookup;

Identify packet types and police RC traffic, BE traffic, and standard Ethernet traffic.

IBI (Input Buffer Interface, input buffer interface) module: the main function is to

The message data is sent to the message centralized caching module for caching. This module receives frame parsing module

The incoming 134bit data is buffered using two registers, any one of which registers

If the device has data, it sends a write request to the message centralized cache module, and caches the received message.

After the response from the module, the writing of one shot of data is completed;

Data is written to another register.

OpenTSN

Machine Translated by Google

2.1.1.2. Network output processing logic

The internal block diagram of the network output processing logic is shown in Figure 2-4.

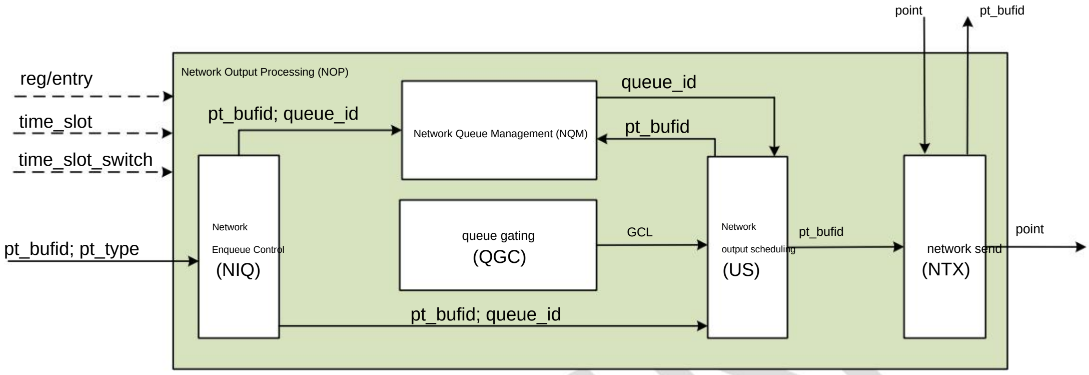


Figure 2-4 Internal function division of NOP module

OpenTSN

NIQ (Network Input Queue, network queue control) module: the main function is to

pkt_bufid is written to the network queue management module for caching. This module needs to receive

The received message type information and the gate control information from the queue gate control module are processed by queue_id.

Map, and send pkt_bufid and queue_id to the network queue management module for caching.

Send pkt_bufid and queue_id to the network output scheduling module at the same time, so that the queue

The first address is managed. This module also needs to be scheduled according to the information written to the queue and the network output

The signal of the scheduling queue transmitted by the module is used to manage the state of the queue, mainly for all

The number of bufids currently written in the queue is managed.

NQM (Network Queue Manage, Network Queue Management) module: main function

Can be centralized buffer management of queues for network output ports

QGC (Queue Gate Control, Queue Gate Control) module: the main function is to

The configured global clock reads the gated list, and converts the gated list (the format of the gated list

The gate switch information of the 8 queues in Table 2-3) is sent to the network queuing control module

Machine Translated by Google

and network output scheduling module.

Table 2-3 Gating List Data Format

name	meaning	Remark
gate_ctrl_vector[7:0]	Gating vector, corresponding to 8 Gating information.	Use RAM for real Now, the depth is 1024

NOS (Network Output Schedule, Network Output Schedule) module: main function

It is to extract pkt_bufid from the network queue management module according to the scheduled queue information. this model

The block needs to be calculated according to the current queue information and the gate control information from the queue gate control module,

Get a queue with the highest priority scheduling, and get it from the corresponding queue of the network queue management module

to pkt_bufid.

NTX (Network TX, network transmission) module: The main function is to retrieve the message from the message buffer

Read the message and release pkt_bufid, calculate the transparent clock of the time synchronization message and accumulate it to the transparent clock

Clear the clock domain, switch the message transmission clock domain from the internal processing clock domain of the architecture to the GMII transmission clock domain.

Send the clock domain, construct the frame preamble and frame start character, and output the message from the network interface. read

When sending a message, you need to first map pkt_bufid to the base where the message is cached in the centralized buffer area

address, and read the message to the message centralized buffer module according to this address, and the bit width of the message will be read out.

Convert from 134bit to 8bit, when the last beat of data is received, pass this pkt_bufid

It is released to the message centralized cache module for subsequent use by the messages entering the architecture.

2.1.1.3. Control input processing logic

The internal block diagram of the control input processing logic is shown in Figure 2-5.

OpenTSN

Machine Translated by Google

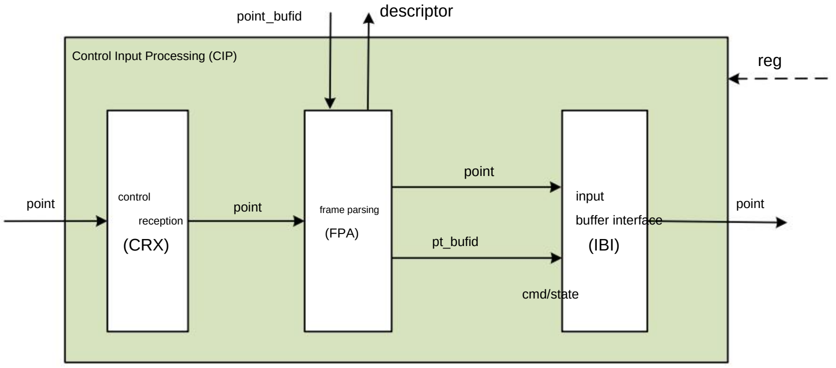


Figure 2-5 Internal function division of NIP module

CRX (Control RX, control receiving) module: the main function is to receive network interface transmission

The incoming message, the message transmission clock domain is switched from the receiving clock domain of the GMII interface to the TSS

external clock domain; record the time when the module receives the time synchronization message, and store it in the

In TSNTag; identify standard Ethernet packets and TSN packets and

segment to decide whether to transmit the message: if the TSS is in the initialization phase, discard all the messages;

If the TSS is in the configuration stage, it transmits standard Ethernet packets and configuration packets in TSN packets.

If the TSS is in the clock synchronization stage, it transmits standard Ethernet packets and TSN packets.

non-time-sensitive messages; if the TSS is in the normal working phase, it transmits standard Ethernet messages.

messages and TSN messages.

FPA (Frame Parse, frame parsing) module: The main function is to centrally cache received messages

The pkt_bufid allocated by the module converts the data bit width of each shot of the message from 9bit to 134bit, and

Output the bit-width converted message and pkt_bufid to the input buffer interface module; according to the connection

The received message constructs a message descriptor and outputs it to the forwarding table lookup module for table lookup;

Identify packet types and police RC traffic, BE traffic, and standard Ethernet traffic.

IBI (Input Buffer Interface, input buffer interface) module: the main function is to

The message data is sent to the message centralized caching module for caching. This module receives frame parsing module

OpenTSN

Machine Translated by Google

The incoming 134bit data is buffered using two registers, any one of which registers

If the device has data, it sends a write request to the message centralized cache module, and caches the received message.

After the response from the module, the writing of one shot of data is completed;

Data is written to another register.

2.1.1.4. Control output processing logic

The internal block diagram of the control output processing logic is shown in Figure 2-6.

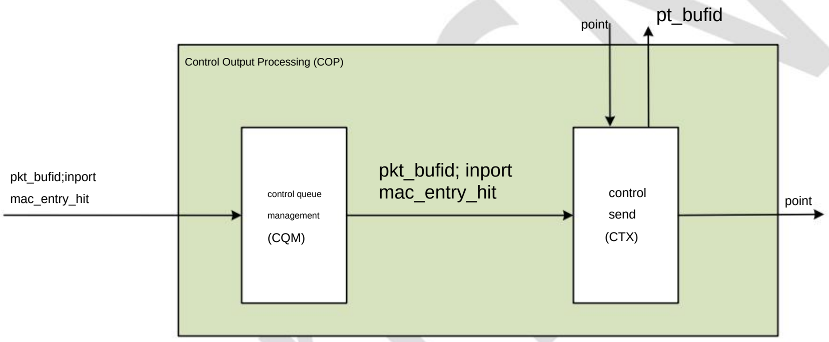


Figure 2-6 Internal function division of NIP module

CQM (Control Queue Management, Control Queue Management) module: main

The function is to write the message descriptor into the queue for buffering. When the control interface is idle, the

Descriptors in the queue are read out and transferred to the control sending module.

CTX (Control TX, control transmission) module: the main function is based on the received

pkt_bufid, reads the message from the centralized buffer, and converts the bit width of the message from 134bit to

8bit and add frame preamble and frame start character, and release pkt_bufid at the same time; identify standard Ethernet

network message and TSN message, and store mac_entry_hit and inport in standard Ethernet

DMAC field of the message; calculates the transparent clock of the PTP message and accumulates it in the PTP message

The message transparent clock domain is used to switch the message transmission clock domain from the system clock domain to the GMII transmission clock domain.

clock domain.

Machine Translated by Google

2.1.1.5. Internal processing logic

FLT (Forward Lookup Table) module: The main function is to

The descriptors sent from each interface are processed in a delayed manner to prevent the descriptors from being transmitted to the network too quickly. module/control sending module (but the message has not been written to the message set buffer) The network sending module/control sending module reads the empty problem from the buffer area of the message set; based on the time Demultiplexing technology receives descriptors from 9 interfaces and transmits them serially; identification standard Ethernet message descriptor and TSN message descriptor, use the standard Ethernet message descriptor with DMAC looks up the MAC forwarding table, and uses the FlowID to look up the TSN packet descriptor FlowID forwarding table, and output the message descriptor to the corresponding interface according to the table lookup result; root According to the results of the descriptor query and forwarding table, calculate the number of output interfaces of each descriptor, and put the number of The amount and its corresponding pkt_bufid are output to the packet centralized cache module, and the packet centralized cache module The block uses this information as a pkt_bufid release condition.

Table 2-4 Forwarding table data format

name		
outport[8:0]	Meaning Output port number (bitmap), a total of 8 network interfaces and 1 host interface.	Remarks are implemented using RAM with a depth of 16K

PCB (Pkt Centralize Buffer, message centralized buffer) module: the main function is to

The architecture needs to centrally cache all the messages that need to be forwarded, and the free address buffer pkt_bufid Cache management is performed. Each pkt_bufid needs to design a counter to control the output port The number is counted, each time pkt_bufid is released, the counter is decremented by one; when pkt_bufid is released When releasing after use, it is necessary to detect the value of the counter corresponding to the pkt_bufid, only When the counter value is 0, the pkt_bufid can be written into the free address buffer.

0 means that the pkt_bufid has a multicast packet and the packet has not been sent from all required The port output to output. The message buffer divides the 1024KB space into 512 messages

OpenTSN

Machine Translated by Google

Cache block, each message cache block can cache a 2KB message.

Table 2-5 Data format of the message buffer area

Address[8:0]	Content [133:0]
0-127	The first message buffer block
128-255	2nd message buffer block
…	..
65408-65535	The 512th message buffer block

Table 2-6 Free address buffer data format

name		Remark
pts_bufid [8: 0]	Indicates the idle packets in the current "packet buffer". File cache block ID number.	Implemented using RAM, depth 5 12ÿ

CPA (Command Parse, command parsing) module: responsible for receiving the HCP

The command command is parsed to implement local registers, forwarding tables, and gated lists.

configuration; according to the received read command, encapsulate the read data in the response and output it to the HCP

module.

Table 2-7 wr_command/rd_command/rd_command_ack command format

Location	Bit width name		illustrate
[203:196]	8	node_id	This field is used to identify which node to read Write. Each TSE or TSS has a unique the node ID. This field is in TSN network card + TS N is used in switch mode.
[195:188]	8	dest_module_id	This field is used to identify which modules are controlled. Each TSE or TSS internal Each submodule has a unique module ID
[187:184]	4	type	4'b0001: write command of register or table entry; 4'b0010: read command of register or table entry; 4'b0110: Read response of register or table entry.
[183:152]	32	addr	read/write address of a register or entry
[151:0]	152	data	Read/write data of registers or table entries

OpenTSN

Machine Translated by Google

2.1.2. TSS processing flow

2.1.2.1. The processing flow of network port input and network port output

The port type configured on each network port is the cooperative type, and the packets entering the network port have TSN messages and standard Ethernet messages, and the output port is confirmed by looking up the table.

The detailed processing flow is as follows:

The message enters the network receiving module through the network port (GMII_RX) to complete the GMII clock domain The transition to the clock domain is processed inside the fabric, and the message receipt timestamp is recorded; the message is identified Whether it is a standard Ethernet or TSN message, and then according to the stage of the switching processing logic Decide whether to transmit the message: if it is a standard Ethernet message, as long as the switching processing logic is completed After initialization, all standard Ethernet messages can be transmitted. If it is a TSN message, If the switching processing logic is in the initialization stage, all packets are discarded. In the configuration stage, only configuration packets are transmitted; if the switching processing logic is in the clock synchronization stage, Transmit non-time-sensitive messages; if the switching processing logic is in normal operation, transmit all message. Send the message to the frame parsing module;

After the frame parsing module receives the message, it parses the message and first recognizes that the message is a standard Ethernet packets are also TSN packets, and the feature information of the packets is extracted and the packets are centrally cached. The pkt_bufid allocated by the module is constructed into a descriptor, and then according to the remaining number of bufid and the message The size relationship of the threshold value is used to supervise the message: if the remaining number of bufid is less than the RC message When the threshold is exceeded, both RC packets and BE packets are discarded. If the remaining number of bufid is less than the BE packet When the packet threshold is exceeded, the BE packets are discarded. Send the message descriptor to the forwarding table lookup module; the same as The time frame analysis module writes 8bit data into 134bit registers, when a register When it is full of 134 bits, the data in the register is transferred to the input buffer interface module, and the report is completed. Bit width conversion of text;

OpenTSN

Machine Translated by Google

The input buffer interface module receives 134bit packet data, because the buffer area of the packet is

The data is written in a time-division multiplexing method, so the input cache interface module needs to wait

The next data can only be processed after the buffer module in the message set returns an acknowledgement signal.

write. When one of the registers is full, and the message returned by the message set cache module has not been received

When the signal is confirmed, the subsequent data is written to another register, and each register writes

When it is full, it is sent to the centralized message buffering module to cache the message, so as to poll until the message

All writing is completed;

The forwarding table lookup module will identify standard Ethernet message descriptors and TSN message descriptors,

Use the standard Ethernet message descriptor to find the MAC forwarding table with the DMAC, and use the TSN message

The descriptor uses FlowID to find the FlowID forwarding table, and describes the packet according to the table lookup result.

The descriptor is output to the corresponding network interface;

The network enqueue control block in the output port logic receives pkt_bufid and

After pkt_type, cache queue according to the gate control information of the queue gate control module and pkt_type

and send the selected queue_id and pkt_bufid to the network queue manager

The management module is cached.

The network output scheduling module is based on the gate control information of the queue gate control module and each queue

Calculate the queue_id with the highest priority scheduling from the status information, and put the first queue_id in the

The address of each pkt_bufid cache is sent to the network queue management module, waiting for the network queue

management module to read the data in the corresponding queue, and the address of the first pkt_bufid is sent to the network

sending module as pkt_bufid, and according to the read data update this

The buffered address of the first pkt_bufid in queue_id.

After receiving the pkt_bufid, the network output scheduling module sends it to the network sending module.

The network sending module extracts the message from the buffering module in the message set according to pkt_bufid.

The network sending module maintains two registers inside, and converts 134bit data into 8bit in turn

The data is sent later, processed across the clock domain, and then output to the interface (GMII_TX); when a

OpenTSN

Machine Translated by Google

After the register is read empty, read the next 134bit data to the message set buffer module.

fetch until all the message data is read.

2.1.2.2. The processing flow of the network port import and control export

The processing flow of the network port in and out of the control port and the processing flow of the network port in and out of the network port

The only difference is that in the forwarding table lookup module, the message lookup table does not hit, if it is a standard

Ethernet packet, the packet will be broadcast to all ports except the incoming port of the packet for output.

If it is a TSN packet, if the table lookup fails, it will be output directly from the control port. For look-up tables

If the packet hits, the packet will be output to the control port only when the packet is a TSN packet.

2.1.2.3. Processing flow of control port in and out of network port

The processing flow of the control port in and out of the network port and the processing flow of the network port in and out of the network port

Consistent.

OpenTSN

2.2. HCP Overall Design

2.2.1. Internal Function Division of HCP

The internal function division of the hardware control point HCP is shown in Figure 2-7.

function is described.

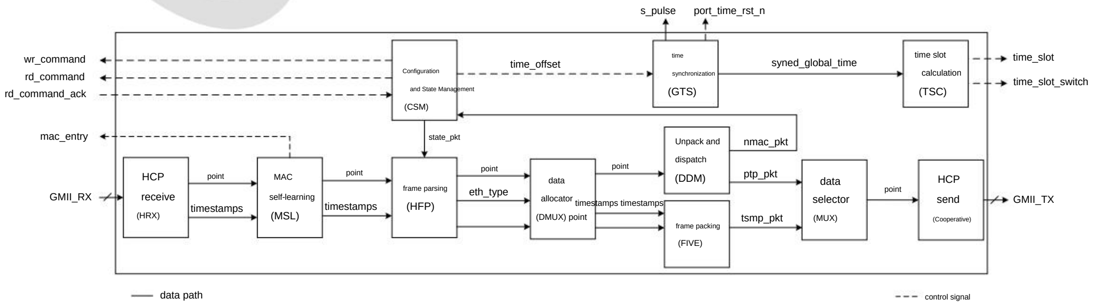


Figure 2-7 Internal function division of hardware control point HCP

Table 2-8 shows the meaning of the signal format in the HCP overall architecture diagram.

Machine Translated by Google

Table 2-8 Signal meanings in the overall architecture of HCP

Signal	Bit width	meaning
pkt_9 (signal of non-message set buffer module No)	9	Message body data, please refer to Appendix A for the specific format
timestamps	19	Receive time stamp signal
mac_entry	-	MAC forwarding table
eth_type	16	Message Type
nmac_pkt	9	Configuration message
ptp_pkt	9	Synchronization message
state_pkt	8	Status message
tsmp_pkt	9	Encapsulated message
s_pulse	1	Sync pulse signal
port_time_rst_n	1	port reset signal
time_offset	49	Clock offset value
syned_global_time	48	Global clock signal
time_slot	11	Time slot signal
time_slot_switch	1	Time slot switching signal
GMII_RX	8	GMII interface receive signal
GMII_TX	8	GMII interface sends signals
command	204	Read and write commands
command_ack	204	Read Command Response

HRX (HCP **RX, hcp** receive) module: The main function is to receive the control interface transmission

The incoming message, when the message transmission clock domain is switched from the GMII receive clock domain to the internal architecture

Clock field, and record the time when the control interface receives the time synchronization message, and store it in TSNTag

middle. Whether to transmit the message is determined according to the stage of the logic of the hardware control point: if the hardware control

The control point logic is in the initialization stage, and all packets are discarded; if the hardware control point logic is in the configuration

In the setup stage, only the configuration message is transmitted; if the hardware control point logic is in the clock synchronization stage, the transmission

Send non-time-sensitive messages; if the hardware control point logic is in the normal working stage, receive all

message.

Machine Translated by Google

OpenTSN

MSL (Mac-address Self Learning, MAC self-learning) module: main functions It is to maintain the contents of the entries in the MAC forwarding table; Match the hit packets, extract the SMAC and input from the miss packets in the MAC forwarding table Interface number inport, when the entry {SMAC,inport} does not exist in the MAC forwarding table, it will be The entry {SMAC,inport} is directly written into the MAC forwarding table.

HFP (Hcp Frame Parse, frame parsing) module: the main function is to extract from the message Take the Ethernet type, and combine the Ethernet type and the received timestamp to form a message descriptor; The packets sent from the interface and the status report packets are selected, and the packets sent from the control interface and the status report packets are selected. Its descriptors are first cached in the queue, and when there is no reporting request from the hardware control point, the The message and its descriptor are read out and output to the next-level module, if the hardware control point has a report request And when the message from the control interface is not transmitted, the status report message is output to the next level. module.

DMUX (DeMUltipleXer, data distributor) module: The main function is based on The Ethernet type dispatches the message and decides whether to Packets are discarded. This module discards all packets in the initialization phase, and in the configuration phase/clock TSMP packets (switch configuration frames, PTP encapsulated frames, ARP encapsulated frames) and their descriptors are dispatched to the decapsulation and dispatch module, when non-TSMP messages (standard Ethernet non-TSMP IP packets, PTP packets, and NMAC status report packets) are dispatched to the frame encapsulation module for processing. package processing.

DDM (Decapsulation_Dispatch_Module, decapsulation and dispatch) module: Decapsulate the received switch configuration frames, PTP encapsulated frames, and ARP encapsulated frames into NMAC configuration frames, PTP frames, ARP response frames, and dispatch NMAC configuration frames to

Machine Translated by Google

The configuration and state management module records the time stamp sent by the PTP frame and compares it with the ARP response frame.

Dispatched to the data selector module.

FEM (Frame_Encapsulation_Module, frame encapsulation) module: main function

It can be to encapsulate the received ARP request frame, PTP frame, and NMAC status report frame into the

In the TSMP message, for the PTP frame, the received timestamp is recorded, the transparent clock is calculated, and the

Accumulate to transparent clock domain.

MUX (MUltipleXer, data selector) module: the main function is to

The frame and the decapsulated frame are selected and output to the control sending module.

CSM (Configuration and State Manage) model

Block: The main function is to parse the received NMAC message and generate a write command;

Counting of received message receive/send/drop pulses; and periodic reporting of status.

GTS (Global_Time_Sync, global clock synchronization) module: the main function is to maintain

Protect a 48bit global clock and receive the offset value between the local clock and the main clock

offset corrects the global clock; maintains a local clock and passes the local clock to it

It is a module used to record the receiving/sending clock of the time synchronization message at the hardware control point.

And calculate the transparent clock that the time synchronization message is transmitted in the hardware control point; maintain a report

The cycle counter outputs a report pulse signal every time a report cycle passes.

TSC (Time_Slot_Calculation, time slot calculation) module: according to the global time

time, time slot length and period, and calculate which time slot in a period the current moment is in.

HTX (HCP **TX, hcp** send) module: The main function is to calculate the time synchronization report

The transparent clock of the message is updated and the transparent clock domain is updated, and the clock domain of the message transmission is processed from the internal clock.

The domain is switched to the clock domain sent by GMII, after adding the frame preamble and frame start, the message is sent from

Control interface output.

OpenTSN

Machine Translated by Google

2.2.2. HCP processing flow

The processing flow of hardware control point logic includes packet encapsulation and decapsulation, configuration packet parsing, reporting of status packets, and MAC self-learning.

2.2.2.1. Configuring Packet Parsing

HCP parses switch configuration packets to

And the configuration of registers and tables in TSS.

After the control receiving module receives the switch configuration message, it stores the bit width of the data in each shot of the message.

After converting from 8bit to 9bit, cross the clock domain for the message (from the GMII receiving clock domain to the

HCP internal logic working clock domain) conversion, and according to the stage of the HCP to the message

To drop or transmit: If the HCP is in the initialization phase/configuration phase, the switch configuration is discarded.

If the HCP is in the clock synchronization phase/normal working phase, it transmits TSMP packets.

The frame parsing module extracts the Ethernet type from the switch configuration message and converts the Ethernet type

and the receiving time to form the message descriptor, now write the message and its descriptor into the fifo, in

When HCP has no report request, read the message and descriptor in fifo and output it to data allocation

device.

The data distributor determines that the packet is TSMP according to the Ethernet type of the packet as 0xff01

message (switch configuration message is a type of TSMP message), in the HCP

During the initialization phase, the switch configuration packets are discarded, and when the HCP is in the configuration phase/clock

In the synchronization phase/normal working phase, the switch configuration packet is output to the decapsulation and dispatch mode.

yuan.

The switch configuration message is decapsulated and dispatched in the decapsulation and dispatch module. will be handed over

Switch configuration frames are decapsulated into NMAC configuration frames and dispatched to the configuration and state management module

OpenTSN

Machine Translated by Google

to parse.

The configuration and state management module parses the NMAC message and extracts the message from the NMAC message

The number of configuration data carried in the text, and each configuration data is encapsulated according to the configured base address.

The device is encapsulated into a write command and output to the end processing logic TSE or the exchange processing logic TSS.

2.2.2.2. Status message reporting

HCP peer-to-peer processing logical TSE and hardware control point logical HCP status, or exchange

Periodically report the status of the processing logic TSS and the hardware control point logic HCP.

Every time the local clock passes through a reporting period, the clock synchronization module will output a reporting pulse.

It is sent to the configuration and status management module. After the configuration and status management module detects the reporting pulse,

Send a report request to the frame parsing module, and after receiving the response signal from the frame parsing module,

According to the report type, a read command will be sent to the end processing logic TSE or the exchange processing logic TSS

command, the read command contains the address of the data to be read; when the receiving end processes the logic TSE or

After exchanging the read response sent by the processing logic TSS, read data from the read response

It is output to the frame parsing module as the data in the status message.

After the frame parsing module detects the report request from the configuration and status management module, when the

When there is no message output from this module, it sends a response to the configuration and status management module, and then starts to

The status message from the configuration and status management module is output to the data distributor.

The data distributor determines that the message is status based on the Ethernet type of the message being 0x1662

Report message, report the status when the HCP is in the clock synchronization phase/normal working phase

The text is output to the frame encapsulation module.

The frame encapsulation module encapsulates the status report message into the TSMP message and outputs it to the data

OpenTSN

Machine Translated by Google

Selector.

The data selector selects the decapsulated packet and the encapsulated packet.

When there is a transmission request for the following message, the message is output to the control sending module.

Control the sending module to switch the message transmission clock domain from the internal processing clock domain to the GMII

Send the clock domain, add frame preamble and frame start, and output the message from the control interface.

2.2.2.3. Decapsulation of PTP-encapsulated packets

HCP decapsulates PTP-encapsulated packets.

After the HCP receiving module receives the PTP-encapsulated message, it converts the bit width of each data shot of the message into

After converting from 8bit to 9bit, cross the clock domain for the message (from the GMII receiving clock domain to the

HCP internal logic working clock domain) conversion, and record the time when the HCP interface receives the message,

Frames are dropped or transmitted depending on the stage the HCP is in: if the HCP is in the initialization stage

In the segment/configuration phase, the PTP encapsulated packets are discarded; if the HCP is in the clock synchronization phase/normal

In the working phase, PTP encapsulated packets are transmitted.

The frame parsing module extracts the Ethernet type from the PTP encapsulated message, and converts the Ethernet type

and the receiving time to form the message descriptor, first write the message and its descriptor into the fifo, and then

When HCP has no report request, read the message and descriptor in fifo and output it to data allocation

device.

The data distributor determines that the packet is TSMP according to the Ethernet type of the packet as 0xff01

message (PTP encapsulated message belongs to TSMP message), when HCP is in the initialization stage,

Discard the PTP-encapsulated packets, when the HCP is in the configuration phase/clock synchronization phase/normal operation

In the operation stage, the PTP encapsulated packets are output to the decapsulation and dispatch module.

OpenTSN

Machine Translated by Google

The decapsulation and dispatch module decapsulates and dispatches the PTP encapsulated packets. the PTP

The encapsulated packet is decapsulated into a PTP packet, and dispatched to the data selector, while recording the PTP

Timestamp when the message is sent.

The data selector selects the decapsulated packet and the encapsulated packet.

When the loaded message has a transmission request, the message is output to the control sending module.

The HCP sending module switches the message transmission clock domain from the internal processing clock domain to the GMII

After sending the clock domain, adding the frame preamble and the frame start, the message is output from the control interface,

For PTP packets, the transparent clock of the PTP packets is calculated and the transparent clock domain is updated.

2.2.2.4. PTP Packet Packet Encapsulation

HCP encapsulates PTP packets. The following describes the PTP packets received by the control interface.

The encapsulation process flow is introduced.

After the HCP receiving module receives the PTP message, it changes the bit width of each data shot of the message from 8 bits.

After converting to 9bit, cross the clock domain for the message (from the GMII receiving clock domain to the HCP

Internal logic working clock domain) conversion, and record the time when the control interface receives the message, according to

The phase in which the HCP is in to drop or transmit the frame: if the HCP is in the initialization phase/

During the configuration phase, PTP packets are discarded; if the HCP is in the clock synchronization phase/normal working phase,

Transmit PTP packets.

The frame parsing module extracts the Ethernet type from the PTP message, and combines the Ethernet type with the connection.

The receiving time forms the message descriptor, first write the message and its descriptor into the fifo, and then write the message and its descriptor to the fifo.

When there is no report request, the message and descriptor in the fifo are read out and output to the data distributor.

The data distributor determines that the packet is PTP according to the Ethernet type of the packet as 0x98f7

OpenTSN

Machine Translated by Google

When the HCP is in the initialization phase/configuration phase, the PTP packet is discarded, and the

HCP is in the clock synchronization phase/normal working phase, and outputs PTP packets to the frame encapsulation mode.

yuan.

The frame encapsulation module encapsulates the status report message into the TSMP message and outputs it to the data

Selector, in which for PTP packets, the transparency of the transmission of PTP packets in HCP will be calculated.

clock and update the transparent clock domain.

The data selector selects the decapsulated packet and the encapsulated packet.

When there is a transmission request for the following message, the message is output to the control sending module.

Control the sending module to switch the message transmission clock domain from the internal processing clock domain to the GMII

Send the clock domain, add frame preamble and frame start, and output the message from the control interface.

2.2.2.5. MAC Self-Learning

After the control receiving module receives the standard Ethernet message, it will change the bit width of the data per shot of the message.

After converting from 8bit to 9bit, cross the clock domain for the message (from the GMII receiving clock domain to the

HCP internal logic working clock domain) conversion, according to the stage of the HCP to the frame

Discard or transmit: If the HCP is in the initialization phase, discard the standard Ethernet packets; if the HCP is in the initialization phase

In the configuration phase/clock synchronization phase/normal working phase, standard Ethernet packets are transmitted.

The MAC self-learning module identifies standard Ethernet packets according to the Ethernet type 0x0800.

text, extract the input interface number inport and source MAC from the DMAC of the standard Ethernet message;

Discard the message that does not hit the MAC forwarding table; it has been recorded in the MAC forwarding table.

Table entry content and its address, and check received (table miss message) SMAC and input

Whether the interface number inport already exists in the MAC forwarding table, if so, it does not need to be SMAC

OpenTSN

Machine Translated by Google

and inport are written to the MAC forwarding table; if it does not exist, {SMAC, inport} and its local

address is written to the MAC forwarding table.

Appendix A. Data Format Definition

A.1. TSN Label Format

Table A-1 TSNtags of time synchronization packets

field	Bit width	bit position	description
point type	3	[47:45]	Message type, where 3'b100 means time synchronization Message (see Table A-2 for other message types)
flow id/local i d	14	[44:31]	Flow ID or internal MAC address, where static The flow uses flowID, and each static flow has a unique A flowID, the dynamic flow uses the local id
reserve	12	[30:19]	Reserved
rx_timestamps 19		[18:0]	When the interface receives the local time synchronization packet time, used to calculate the transparency of PTP packets in the device Ming Clock

Table A-2 TSNTag of non-time synchronization packets

field	Bit width	bit position	description
point type	3	[47:45]	Message type, where 3'b000 represents ST message, 3'b001 means ST message, 3'b010 means ST Message 3'b011 represents RC message, 3'b101 represents shows NMAC packet, 3'b110 means BE packet , 3'b111: the packet to be reorganized
flow id/ local id	14	[44:31]	Flow ID or local ID, where static traffic uses Use flowid to identify, each static flow has a unique A flowid, dynamic flow is marked with localid knowledge
sequence id 16		[30:15]	The sequence number used to identify the packets in each flow
frag flag	1	[14]	Fragment identification bit, this field is in the case of packet fragmentation Will be used to identify the last part of the message Fragment, where 1'b0 indicates that the message is not the last fragment Fragment, 1'b1 represents the last fragment of the packet
frag id	4	[13:10]	Fragment number, this field will be used in the case of packet fragmentation is used, for each fragment of the message Numbering
inject addr 5		[9:5]	The cache address of the ST stream at the sender of the TSN network card

OpenTSN

Machine Translated by Google

submit addr 5		[4:0]	The cache address of the ST stream at the receiving end of the TSN network card

A.2. Message Descriptor Format

The format of the packet descriptor in TSNSwitch3.2 is shown in Table A-3 below.

Table A-3 Message Descriptor Format

field	Bit width bit position description		
dmac	48	[71:24]	Standard Ethernet message/TSN message DMAC field 4 8bit data
standeth_tsn_flag 1		[23]	Standard Ethernet message and TSN message identification bits, which 1'b1 means standard Ethernet message, 1'b0 means TSN message
import	4	[22:19] Packet input interface number of TSN exchange IP	
lookup_en	1	[18]	Check the forwarding table enable signal, where 1'b1 indicates the message The forwarding table needs to be searched, and 1'b0 means that the packet does not need to be searched. find forwarding table
outport	9	[17:9]	The output interface number of the packet without checking the forwarding table. This field is only used when lookup_en is 1'b0 use
pt_bufid	9	[8:0] The buffer space of the message in the centralized buffer area	

OpenTSN

Appendix B. Configuration Instructions

B.1. Hardware Address

The configuration content of the TSN switch includes control registers and tables. Each control register and Each entry has a unique hardware address, and the controller can Configure different control registers or table entries in the N network card. The hardware address is 27bit, its format For details, please refer to Table B-1, in which the upper 7 bits are used to identify the module where the control register or entry is located. Each module in the TSN network card has a unique number (referred to as "module ID"), The lower 20 bits are used to identify the address of the control register or table entry in the module (referred to as "module ground" for short). address"), the address of each control register or table entry in the module is unique.

Machine Translated by Google

Table B-1 Hardware Address Format for Control Registers and Tables

/	Location description
hardware address	[26:20] The number of the module where the control register or entry is located, that is, the module ID; The allocation of block IDs is detailed in Table B-2
	[19:0] The address of the control register or table entry in the module, that is, the module address; The module address range is 0x0_0000~0xf_ffff

Table B-2 Assignment of module IDs in TSN network cards

Module CSM TIS TSS				QGC _P0	QGC _P1	QGC _P2	QGC _P3	QGC _P4
Module ID 0x0		0x1	0x2	0x3	0x4	0x5	0x6	0x7
module	QGC _P5	QGC _P6	QGC _P7	GTS FLT PMD FIM				FMA _P0
Module ID 0x8		0x9	0xa	0xb	0xc	0xd	0xe	0xf
module	FMA _P1	FMA _P2	FMA _P3	FMA _P4	FMA _P5	FMA _P6	FMA _P7	
Module ID 0x10 0x11 0x12				0x13 0x14 0x15 0x16 -				

B.2. Configuration Content

The configuration contents of the TSN switch include control registers and tables, including the mapping table,

Forwarding table and gating table.

B.2.1. Control Register

Control registers include TSN hardware status register, time synchronization register, message

Scheduling register, interface type register, message supervision register and status reporting register,

The description of the control registers is shown in Table B-3.

Table B-3 Control Register Description

Classification	hardware address		name	bit width	describe
	Module ID	module address			
TSN hardware state	7'h0	20'h3	hardware_state 2		The state the TSN hardware is in, where 2'd0 indicates that the TSN hardware is in the initial stage Initialization phase, does not receive any processing message; 2'd1 indicates the TSN hardware

OpenTSN

Machine Translated by Google

					In the configuration phase, only receive and process configuration packets; 2'd2 means that the TSN hardware is in the time synchronization phase, receiving and processing non-ST packets; 2'd3 means that the TSN hardware is in the normal working
time synchronization		20'h1	time_offset_h 17		phase, receiving and processing all packets time_offset_h[16] When it is 1'b0, it means that the offset value between the slave clock and the master clock is negative. When time_offset_h[16] is 1', it means that the offset value between the slave clock and the master clock is positive. The upper 16 bits of the clock offset value, {time_offse t_h[15:0], time_offset_l[31: 7]}, the unit is the lower 32
		20'h0	time_offset_l	32	bits of the offset value between the slave clock and the master clock, {time_offset_h[15:0], time_offset_l[31 :7]} The unit is us, and the unit of {time_offset_l[6:0]} is the compensation period between the slave clock and the
		20'hb	time_offset_ period	24	master clock offset value in 8ns; the TSN hardware will compensate the local clock every time a compensation period passes, and the unit is 8ns The length of the time slot,
message scheduling		20'h2	time_slot_ length	10	in us; the value range is [4us, 512us] and the value must be 2n 802.1Qbv or 802.1Qch scheduling model selection signal, where
		20'h5	qbv_or_qch	1	1'b0 represents the 802.1Qbv scheduling model, and 1' b1 represents the 802.1Qch scheduling model
		20'h8	inject_slot_ period	11 Injection period, unit is time slot	
		20'h9	submit_slot_ period	11 Commit cycle, unit is time slot	
Interface Type		20'h4	port_type	8	Interface type, where 1'b0 indicates that the interface is a cooperative type, and 1'b1 indicates that the interface is a non-cooperative type. rc packet
Packet supervision		20'hc	rc_regulation_ value	9	supervision threshold. When the number of idle pkt_bufid in the TSN hardware is less than the rc packet supervision threshold, rc packets and rc packets are discarded. be packet monitoring threshold
		20'hd	be_regulation_ value	9	for be packets. When the number of idle pkt_bufid in TSN hardware is less than the monitoring threshold for be packets, discard the

OpenTSN

Machine Translated by Google

					Abandon the be message
		20’he	unmap_ regulation_ value	9	Unmapped packet policing threshold, when When the number of idle pkt_bufid in the TSN hardware is less than the unmapped packet supervision threshold, the unmapped
Status report		20'h6	report_type	packets will be discarded. TSN hardware only supports two	
		20'h7	report_en	1	configuration values, namely 12'd 1 and 12'd1000, where 12'd1 means the reporting period is 1024us, and 12'd1000 means the
		20 ha	report_period	12	reporting period is 1024 *1024us

B.2.2. TSN message forwarding table

The TSN packet forwarding table is used to forward TSN packets, and its format is shown in Table B-5 shown. The TSN packet forwarding table is designed in the Forwarding Lookup Table (FLT) module, and the FLT module will The flowid is extracted from the received message descriptor, and the flowid is used as the address of the lookup table to find TSN packet forwarding table, the result of the table lookup is the output port number, if a bit in the output port number is high, then Indicates that the message needs to be output from the corresponding port. This module converts the content of the descriptor according to the table lookup result. sent to the corresponding output port. The depth of the TSN packet forwarding table is 16K, that is, the TSN exchange The entire network composed of the machines can support the forwarding of 16K streams at most.

Table B-4 TSN packet forwarding table format

name		
outport[8:0]	Meaning Output port number (bitmap), a total of 8 network switching interfaces and 1 control interface.	Remarks are implemented using RAM with a depth of 16K

OpenTSN

Machine Translated by Google

B.2.3. Gated List

In order to implement the 802.1Qbv scheduling model, the TSN switch switches the output on each network

Each interface is designed with a gated list, which supports up to 1024 entries.

Table entry descriptions are shown in Table B-5.

Table B-5 Gating List Description

hardware address		name	Location description	
module ID	modular ground site			
7'h3~7'h (a No. 0 connection mouth~7 interface team column gated)	20'h00 0000 . 20'h00 03FF	gate_table_ entry_N	[7]	Gating status of the 7th queue, where 0 means The gate is closed, 1 means the gate is open
			[6]	Gating status of the 6th queue, where 0 means The gate is closed, 1 means the gate is open
			[5]	Gating status of the 5th queue, where 0 means The gate is closed, 1 means the gate is open
			[4]	Gating status of the 4th queue, where 0 means The gate is closed, 1 means the gate is open
			[3]	Gating status of the 3rd queue, where 0 means The gate is closed, 1 means the gate is open
			[2]	Gating state of the 2nd queue, where 0 means The gate is closed, 1 means the gate is open
			[1]	Gating state of the 1st queue, where 0 means The gate is closed, 1 means the gate is open
			[0]	The gate state of the 0th queue, where 0 means The gate is closed, 1 means the gate is open

The queue gate control module uses RAM to cache the gate control list. The RAM bit width is 8 bits and the depth is 8 bits.

The degree is 1024, and the RAM format is shown in Table B-6.

Table B-6 RAM Format of Cache Gating List

index address	Bit width (bit) Depth		Occupied resources (Kbit)
Time Slot	8	1024	8

Appendix C. Configuration/Report Message Format

C.1. Configuration message format

The configuration contents of the control registers and tables of the TSN switch are carried by Ethernet packets.

With, the user specifies the base address of the configuration, the number of configurations, and the configuration in the configuration message.

OpenTSN

Machine Translated by Google

content, to realize a register/entry or multiple registers/entry with consecutive addresses.

line configuration.

The format of the configuration message is shown in Figure C-1. Configuration message TSMP Ethernet header and NMA

The destination MAC and source MAC of the C Ethernet header are TSNtag semantics (the source MAC is also set

The reason why it is counted as TSNtag semantics is that the hardware is reporting NMAC packets and PTP packets, etc.

When encapsulated into the TSMP protocol, the SM in the TSMP Ethernet header of the configuration packet can be directly

The AC is used as the DMAC in the TSMP Ethernet header of the message encapsulation, and is constructed in the NMA

When C reports a packet, it can directly use the SMAC in the NMAC Ethernet header of the configuration packet as the

NMAC reports the DMAC in the Ethernet header of the packet, and the TSN network can recognize and process TSMP

message and NMAC report message), configure the Ethernet type of the message as 0xff01, subclass

The type is 0x2, the configuration packet also contains NMAC protocol, and its Ethernet type is 0x16

62. The 30B in the configuration packet indicates the number of configuration data carried in the packet, and the 32B~th

35B indicates the base address of all configuration data of the message, and 32 bits are reserved for each configuration data

storage bit width. When the number of configuration data is N (decimal), the first configuration register

The address of the register/entry is the base address + 0, and the address of the second configuration register/entry is the base address

+1, the address of the Nth configuration register/entry is the base address + (N-1).

OpenTSN

Machine Translated by Google

	bit			
	31 24	23 16	15 8	7 0
0 3	Destination TSN tags [47:16]			
7	Destination TSN Tag [15:0]		Source TSN tags [47:32]	
11	Source TSN Tag[31:0]			
15	length/type (0xff01)		Subtype (0x2)	interface number
19	Destination TSN tags [47:16]			
23	Destination TSN Tag [15:0]		Source TSNtag [47:32]	
27	Source TSNtag[31:0]			
31	length/type (0x1662)		Register/table entry configuration number n	reserve
35	Register/entry configuration base address			
39	First register/entry configuration data			
67~123	…			
71~127	nth register/entry configuration data			
79~131	CRC			

Figure C-1 Configuration message format

OpenTSN

C.2. Report message format

The report message constructed by the configuration and status management module of HCP is the NMAC protocol message. message (the message format is shown in Figure C-3 below), the message will be sent to the frame encapsulation module of HCP. It is encapsulated into the TSMP protocol (removing the first 16B of the NMAC protocol during encapsulation, and at the end of the message 2B is added at the end for placing the report type).

Machine Translated by Google

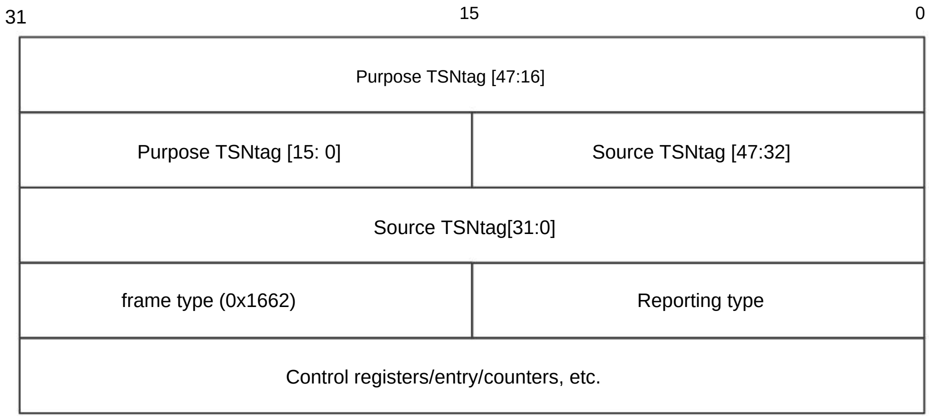


Figure C-3 NMAC report packet format

Table C-2 Report Type Format

Reporting type (16bit)		meaning
High 6bit	Low 10bit	
6'd0	10'd0	A single register (configured by the controller) that contains the configuration complete register device, port status register, time slot length register, time offset Shift register, report period register, report type register, application Use the period register.
6'd1	n (decimal)	32n to 32n+31 forwarding table entries, 0ÿnÿ511 (ten base), and n is an integer
6'd2	n (decimal)	Item 32n to 32n+31 injection schedule entry, 0ÿnÿ31 (decimal), and n is an integer. (not yet available)
6'd3	n (decimal)	Items 32n to 32n+31 submit timetables, 0ÿnÿ31 (decimal), and n is an integer. (not yet available)
6'd4	n (decimal)	32n to 32n+31 gating table entries of No. 0 output interface, 0ÿnÿ31 (decimal), and n is an integer
6'd5	n (decimal)	32n to 32n+31 gating table entries of No. 1 output interface, 0ÿnÿ31 (decimal), and n is an integer
6'd6	n (decimal)	32n to 32n+31 gating table entries of No. 2 output interface, 0ÿnÿ31 (decimal), and n is an integer
6'd7	n (decimal)	32n to 32n+31 gating table entries of No. 3 output interface, 0ÿnÿ31 (decimal), and n is an integer
6'd8	n (decimal)	32n to 32n+31 gating table entries of No. 4 output interface, 0ÿnÿ31 (decimal), and n is an integer
6'd9	n (decimal)	32n to 32n+31 gating table entries of No. 5 output interface, 0ÿnÿ31 (decimal), and n is an integer
6'd10	n (decimal)	32n to 32n+31 gating table entries of output interface No. 6, 0ÿnÿ31 (decimal), and n is an integer
6'd11	n (decimal)	32n to 32n+31 gating table entries of No. 7 output interface, 0ÿnÿ31 (decimal), and n is an integer
6'd12	10'd0	report status register

OpenTSN

Machine Translated by Google

Reporting type (16bit)		meaning
High 6bit	Low 10bit	
6'd13	n (decimal)	Items 2n and 2n+ in the message mapping and dispatching module in the TSN network card 1 mapping table entry, 0ÿnÿ15 (decimal), and n is an integer
6'd14	n (decimal)	Item 4n to Item 4n+3 inverse in the frame inverse mapping module in the TSN network card Mapping table entry, 0ÿnÿ63 (decimal), and n is an integer
6'd15	n (decimal)	Items 2n and 2n+1 in the frame mapping module of input interface 0 Mapping table entry, 0ÿnÿ15 (decimal), and n is an integer
6'd16	n (decimal)	Items 2n and 2n+1 in the frame mapping module of input interface 1 Mapping table entry, 0ÿnÿ15 (decimal), and n is an integer
6'd17	n (decimal)	Items 2n and 2n+1 in the frame mapping module of input interface 2 Mapping table entry, 0ÿnÿ15 (decimal), and n is an integer
6'd18	n (decimal)	Items 2n and 2n+1 in the frame mapping module of input interface 3 Mapping table entry, 0ÿnÿ15 (decimal), and n is an integer
6'd19	n (decimal)	Items 2n and 2n+1 in the frame mapping module of input interface 4 Mapping table entry, 0ÿnÿ15 (decimal), and n is an integer
6'd20	n (decimal)	Items 2n and 2n+1 in the frame mapping module of input interface 5 Mapping table entry, 0ÿnÿ15 (decimal), and n is an integer
6'd21	n (decimal)	Items 2n and 2n+1 in the frame mapping module of input interface 6 Mapping table entry, 0ÿnÿ15 (decimal), and n is an integer
6'd22	n (decimal)	Items 2n and 2n+1 in the frame mapping module of input interface 7 Mapping table entry, 0ÿnÿ15 (decimal), and n is an integer

OpenTSN

Machine Translated by Google

C.2.1. Single register reporting message format

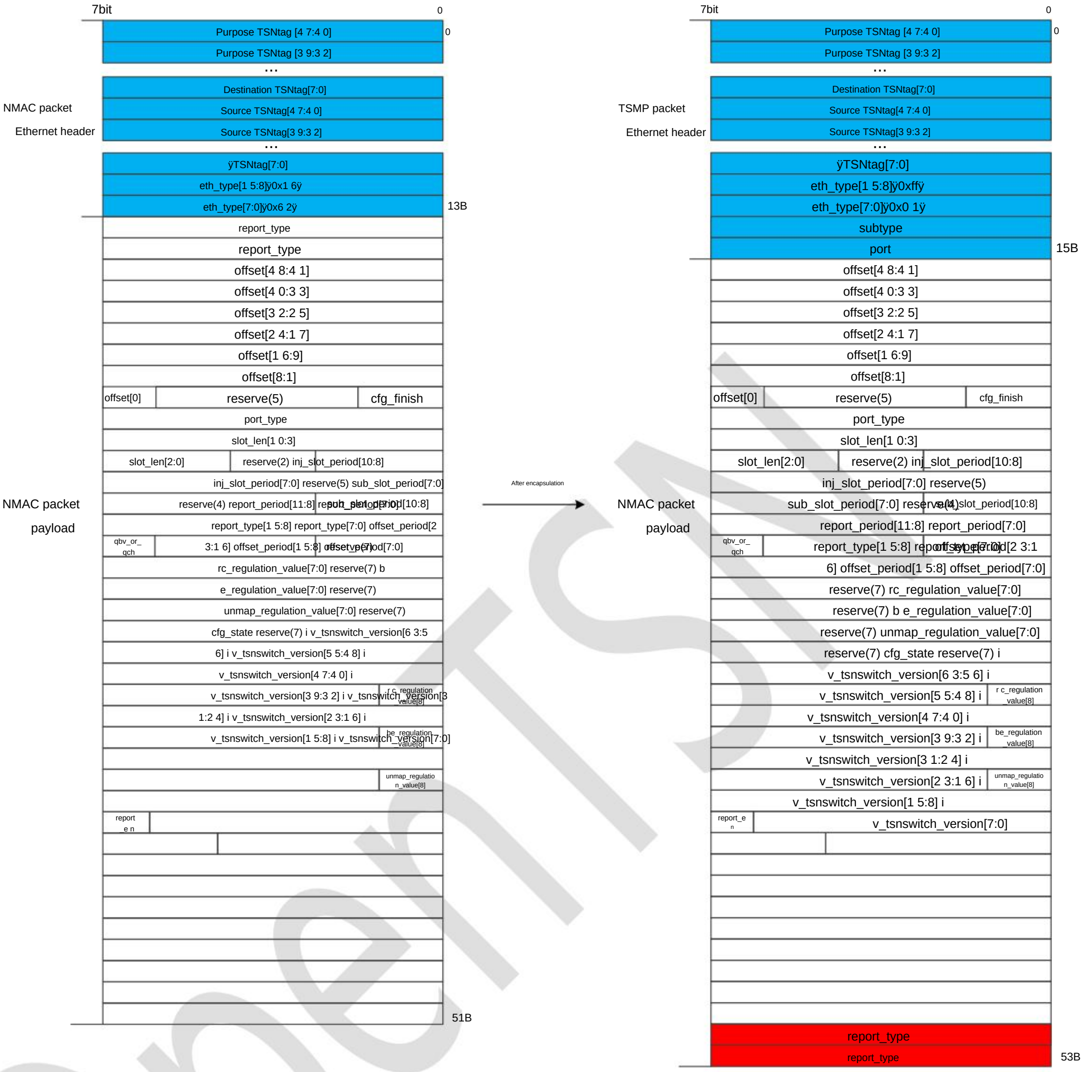
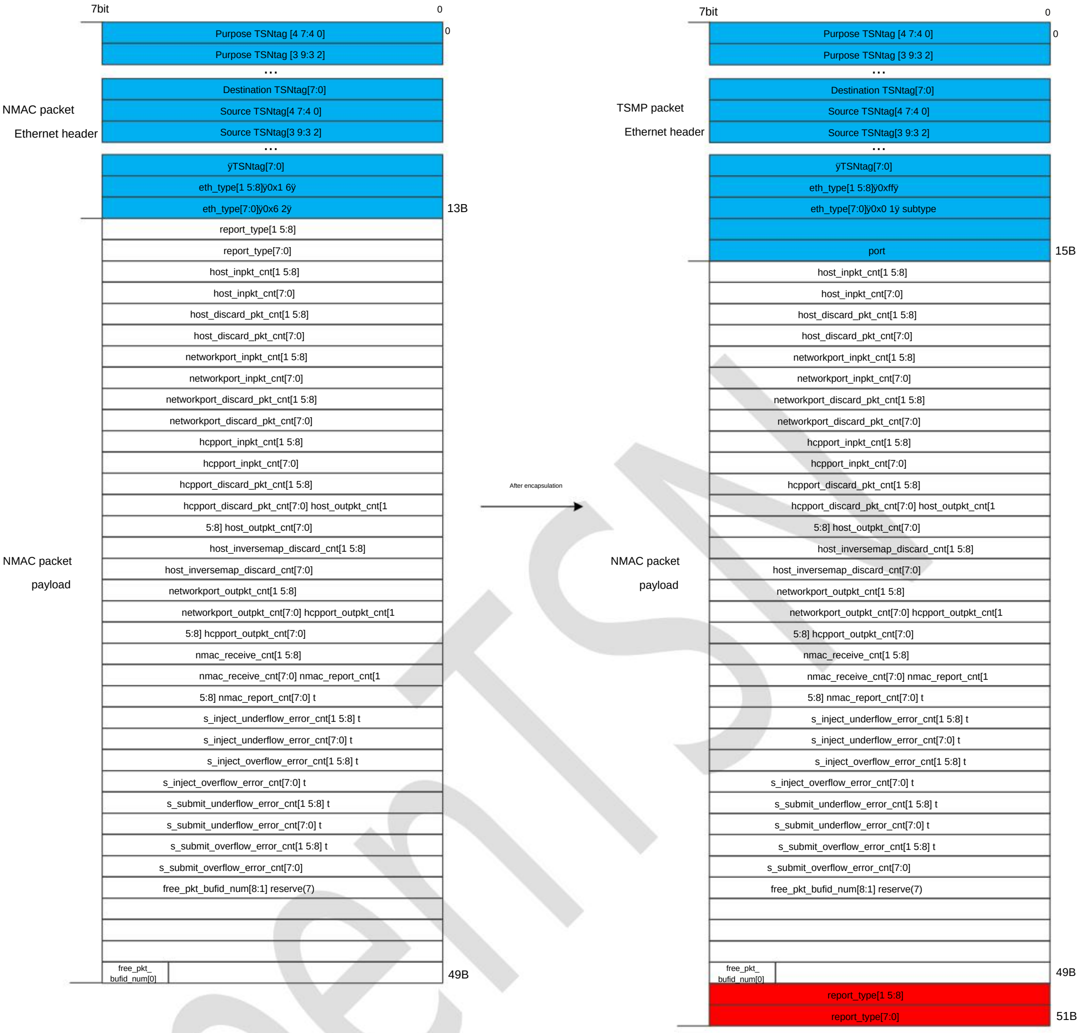

7bit
0
Purpose TSNtag [4 7:4 0]
Purpose TSNtag [3 9:3 2]
Destination TSNtag[7:0]
Source TSNtag[4 7:4 0]
Source TSNtag[3 9:3 2]
ýTSNtag[7:0]
eth_type[1 5:8]ý0x1 6ý
eth_type[7:0]ý0x6 2ý
13B
NMAC packet
Ethernet header
report_type
report_type
offset[4 8:4 1]
offset[4 0:3 3]
offset[3 2:2 5]
offset[2 4:1 7]
offset[1 6:9]
offset[8:1]
offset[0]
reserve(5)
cfg_finish
port_type
slot_len[1 0:3]
slot_len[2:0]
reserve(2) inj_slot_period[10:8]
inj_slot_period[7:0] reserve(5) sub_slot_period[7:0]
reserve(4) report_period[11:8] report_period[7:0]
report_type[1 5:8] report_type[7:0] offset_period[2
qbv_or_qch
3:1 6] offset_period[1 5:8] offset_period[7:0]
rc_regulation_value[7:0] reserve(7) b
e_regulation_value[7:0] reserve(7)
unmap_regulation_value[7:0] reserve(7)
cfg_state reserve(7) i v_tsnswitch_version[6 3:5
6] i v_tsnswitch_version[5 5:4 8] i
v_tsnswitch_version[4 7:4 0] i
v_tsnswitch_version[3 9:3 2] i v_tsnswitch_version[3
r c_regulation_value[8]
1:2 4] i v_tsnswitch_version[2 3:1 6] i
v_tsnswitch_version[1 5:8] i v_tsnswitch_version[7:0]
be_regulation_value[8]
unmap_regulation_value[8]
report_en
NMAC packet
payload
51B
After encapsulation
7bit
0
Purpose TSNtag [4 7:4 0]
Purpose TSNtag [3 9:3 2]
Destination TSNtag[7:0]
Source TSNtag[4 7:4 0]
Source TSNtag[3 9:3 2]
TSMP packet
Ethernet header
ýTSNtag[7:0]
eth_type[1 5:8]ý0xffý
eth_type[7:0]ý0x0 1ý
subtype
port
15B
offset[4 8:4 1]
offset[4 0:3 3]
offset[3 2:2 5]
offset[2 4:1 7]
offset[1 6:9]
offset[8:1]
offset[0]
reserve(5)
cfg_finish
port_type
slot_len[1 0:3]
slot_len[2:0]
reserve(2) inj_slot_period[10:8]
inj_slot_period[7:0] reserve(5)
sub_slot_period[7:0] reserve(4) slot_period[10:8]
report_period[11:8] report_period[7:0]
qbv_or_qch
report_type[1 5:8] report_type[7:0] offset_period[2 3:1
6] offset_period[1 5:8] offset_period[7:0]
reserve(7) rc_regulation_value[7:0]
reserve(7) b e_regulation_value[7:0]
reserve(7) unmap_regulation_value[7:0]
reserve(7) cfg_state reserve(7) i
v_tsnswitch_version[6 3:5 6] i
v_tsnswitch_version[5 5:4 8] i
r c_regulation_value[8]
v_tsnswitch_version[4 7:4 0] i
v_tsnswitch_version[3 9:3 2] i
be_regulation_value[8]
v_tsnswitch_version[3 1:2 4] i
v_tsnswitch_version[2 3:1 6] i
unmap_regulation_value[8]
v_tsnswitch_version[1 5:8] i
report_en
v_tsnswitch_version[7:0]
NMAC packet
payload
report_type
report_type
53B

Figure C-4 NMAC packet format for reporting a single register

OpenTSN

Machine Translated by Google

C.2.2. Format of hardware status report message


7bit
0
0
Purpose TSNtag [4 7:4 0]
Purpose TSNtag [3 9:3 2]
...
Destination TSNtag[7:0]
Source TSNtag[4 7:4 0]
Source TSNtag[3 9:3 2]
...
ÿTSNtag[7:0]
eth_type[1 5:8]ÿ0x1 6ÿ
eth_type[7:0]ÿ0x6 2ÿ
13B
NMAC packet
Ethernet header
report_type[1 5:8]
report_type[7:0]
host_inpkt_cnt[1 5:8]
host_inpkt_cnt[7:0]
host_discard_pkt_cnt[1 5:8]
host_discard_pkt_cnt[7:0]
networkport_inpkt_cnt[1 5:8]
networkport_inpkt_cnt[7:0]
networkport_discard_pkt_cnt[1 5:8]
networkport_discard_pkt_cnt[7:0]
hcpport_inpkt_cnt[1 5:8]
hcpport_inpkt_cnt[7:0]
hcpport_discard_pkt_cnt[1 5:8]
hcpport_discard_pkt_cnt[7:0] host_outpkt_cnt[1
5:8] host_outpkt_cnt[7:0]
host_inversemap_discard_cnt[1 5:8]
host_inversemap_discard_cnt[7:0]
networkport_outpkt_cnt[1 5:8]
networkport_outpkt_cnt[7:0] hcpport_outpkt_cnt[1
5:8] hcpport_outpkt_cnt[7:0]
nmac_receive_cnt[1 5:8]
nmac_receive_cnt[7:0] nmac_report_cnt[1
5:8] nmac_report_cnt[7:0] t
s_inject_underflow_error_cnt[1 5:8] t
s_inject_underflow_error_cnt[7:0] t
s_inject_overflow_error_cnt[1 5:8] t
s_inject_overflow_error_cnt[7:0] t
s_submit_underflow_error_cnt[1 5:8] t
s_submit_underflow_error_cnt[7:0] t
s_submit_overflow_error_cnt[1 5:8] t
s_submit_overflow_error_cnt[7:0]
free_pkt_bufid_num[8:1] reserve(7)
free_pkt_ bufid_num[0]
49B
NMAC packet
payload
After encapsulation
7bit
0
0
Purpose TSNtag [4 7:4 0]
Purpose TSNtag [3 9:3 2]
...
Destination TSNtag[7:0]
Source TSNtag[4 7:4 0]
Source TSNtag[3 9:3 2]
...
ÿTSNtag[7:0]
eth_type[1 5:8]ÿ0xffÿ
eth_type[7:0]ÿ0x0 1ÿ subtype
port
15B
TSMP packet
Ethernet header
host_inpkt_cnt[1 5:8]
host_inpkt_cnt[7:0]
host_discard_pkt_cnt[1 5:8]
host_discard_pkt_cnt[7:0]
networkport_inpkt_cnt[1 5:8]
networkport_inpkt_cnt[7:0]
networkport_discard_pkt_cnt[1 5:8]
networkport_discard_pkt_cnt[7:0]
hcpport_inpkt_cnt[1 5:8]
hcpport_inpkt_cnt[7:0]
hcpport_discard_pkt_cnt[1 5:8]
hcpport_discard_pkt_cnt[7:0] host_outpkt_cnt[1
5:8] host_outpkt_cnt[7:0]
host_inversemap_discard_cnt[1 5:8]
host_inversemap_discard_cnt[7:0]
networkport_outpkt_cnt[1 5:8]
networkport_outpkt_cnt[7:0] hcpport_outpkt_cnt[1
5:8] hcpport_outpkt_cnt[7:0]
nmac_receive_cnt[1 5:8]
nmac_receive_cnt[7:0] nmac_report_cnt[1
5:8] nmac_report_cnt[7:0] t
s_inject_underflow_error_cnt[1 5:8] t
s_inject_underflow_error_cnt[7:0] t
s_inject_overflow_error_cnt[1 5:8] t
s_inject_overflow_error_cnt[7:0] t
s_submit_underflow_error_cnt[1 5:8] t
s_submit_underflow_error_cnt[7:0] t
s_submit_overflow_error_cnt[1 5:8] t
s_submit_overflow_error_cnt[7:0]
free_pkt_bufid_num[8:1] reserve(7)
free_pkt_ bufid_num[0]
49B
report_type[1 5:8]
report_type[7:0]
51B
NMAC packet
payload

OpenTSN

Figure C-5 Format of the hardware status report message

Table C-3 Description of hardware reported counters

ÿÿ	Bit width meaning 16	
ctrlport_inpkt_cnt	Counter for the number of packets received by the control interface	
ctrlport_discard_pkt_cnt port0_inpkt_cnt	16 Counter for the number of packets discarded by traffic policing on the control interface	
port0_discard_pkt_cnt port1_inpkt_cnt	16	Counter for the number of packets received by network
port1_discard_pkt_cnt port2_inpkt_cnt	16	port 0 Counter for the number of packets discarded by traffic policing on
port2_discard_pkt_cnt port3_inpkt_cnt	16	network port 0 Counter for the number of packets received by network port 1
port3_discard_pkt_cnt	16	Number of received packets counter Number of packets discarded by traffic
	16	policing of network port 2 Counter of number of received packets of network
	16	port 3 Counter
	16	
	16	

Machine Translated by Google

port4_inpkt_cnt	16	Number of received packets counter for network port 4
port4_discard_pkt_cnt port5_inpkt_cnt	16	Counter for the number of packets discarded by traffic policing on network port 4
port5_discard_pkt_cnt port6_inpkt_cnt	16	Number of received packets counter on network port 5
port6_discard_pkt_cnt port7_inpkt_cnt	16	Counter for the number of packets discarded by traffic policing on network port 5
port7_discard_pkt_cnt	16	Number of received packets counter on network port 6
ctrlport_outpkt_cnt port0_outpkt_cnt	16	Counter for the number of packets discarded by traffic policing on network port 6
port1_outpkt_cnt port2_outpkt_cnt	16	Number of received packets counter on network port 7
port3_outpkt_cnt port4_outpkt_cnt	16	Counter for the number of packets discarded by traffic policing on network port 7
port5_outpkt_cnt port6_outpkt_cnt	16	Counter of the number of packets sent by the control interface
port7_outpkt_cnt nmac_receive_cnt	16	Number of packets sent by network port 0
nmac_report_cnt	16	Number of packets sent by network port 1
	16	Number of packets sent by network port 2
	16	Number of packets sent by network port 3
	16	Number of packets sent by network port 4
	16	Number of packets sent by network port 5
	16	Number of packets sent by network port 6
	16	Number of packets sent by port 7
	16	Counter of the number of NMAC messages received by the CSM module
	16	Counter for the number of NMAC packets reported by the CSM module
ts_inj_underflow_error_cnt	16	Underflow errors occur when time-sensitive packets are injected into scheduling Number (there is no time-sensitive packet injection control function)
ts_inj_overflow_error_cnt	16	Times of overflow error when time-sensitive message is injected into scheduling Number (there is no time-sensitive packet injection control function)
ts_sub_underflow_error_cnt	16	Underflow errors occur when time-sensitive packets are submitted for scheduling Number (there is no time-sensitive packet submission control function)
ts_sub_overflow_error_cnt	16	Overrun errors occur when time-sensitive packets are submitted for scheduling Number (there is no time-sensitive packet submission control function)
free_pkt_bufid_num total	16	Number of hardware free pkt_bufid
	/	34

OpenTSN

Machine Translated by Google

C.2.3. Gated list/forwarding table entry reporting message format

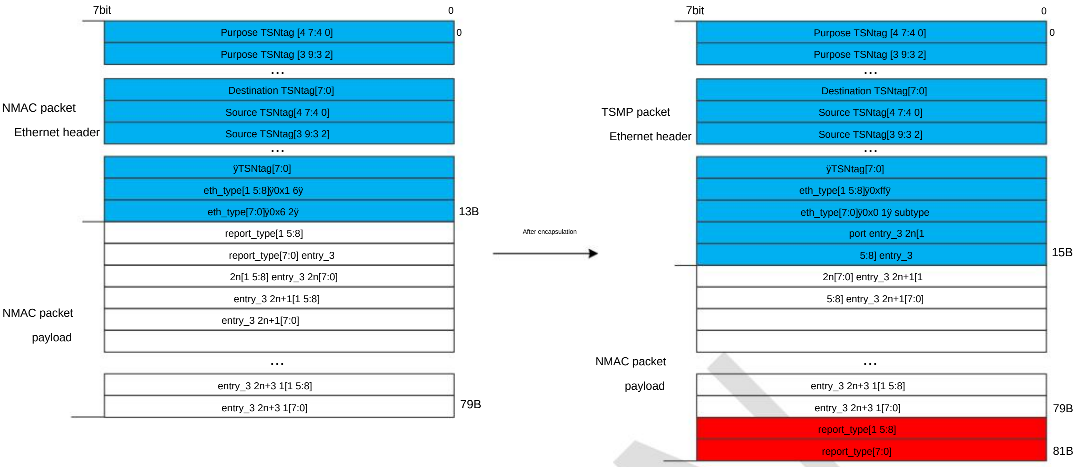


Figure C-6 Format of the report of the gating list entry

Appendix D. **wr_command/rd_command/rd_command_ack** commands

Format

Table D-1 wr_command/rd_command/rd_command_ack Command Format

Location	Bit width name		
[203:196]	8	node_id	Description This field is used to identify which node to read and write. Each TSE or TSS has a unique node ID. This field is used in TSN network card+TS N switch mode. This field is used to identify which module within a node is
[195:188]	8	dest_module_id	controlled. Each sub-module inside TSE or TSS has a unique module ID 4'b0001: write command of register or table entry; 4'b0010: read command of register or table entry;
[187:184]	4	type	4'b0110: read response of register or table entry. read/write address of a register or entry
[183:152]	32	addr	
[151:0]	152	data	Read/write data of registers or table entries

Appendix E. **PTP** Protocol Format

The format of the Sync, Delay_req, Delay_resq and test packets is shown in Figure E-1.

OpenTSN

Machine Translated by Google

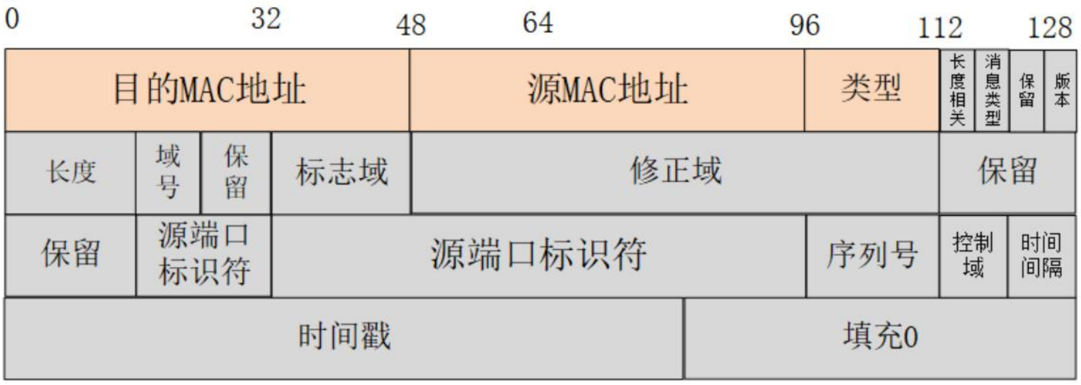


Figure E-1 PTP packet format

Type is: 16'h88F7;

Message type: sync is 4'd1, delay_req is 4'd3, delay_resq is 4'd4, delay_test is 4'd5;

The length is: 16'd64 bytes;

Correction field: transparent clock, at the beginning, this field is 0;

Timestamp: for timestamp (other PTP fields that don't need relationship are filled with 0)

Appendix F. TSMP Message Protocol Format

TSMP (Time Sensitive Message Protocol) is a TSN controller for network topology detection, The protocol that configures the TSN switch and HCP and encapsulates frames.

F.1. TSMP Frame Design Principles

(1) PTP frame is a subtype of TSMP frame;

(2) TSNtag is the result of frame mapping. In the TSN network, according to the TSNtag Frames are logically processed (including table lookup and forwarding, enqueueing, scheduling priority, on-time injection, on-time commit, output gating, etc.);

(3) Design relevant fields in the TSMP frame header to identify different types of TSMP frame.

OpenTSN

Machine Translated by Google

F.2. TSMP Frame Format

The format design of the TSMP frame is shown in Figure F-1 below.

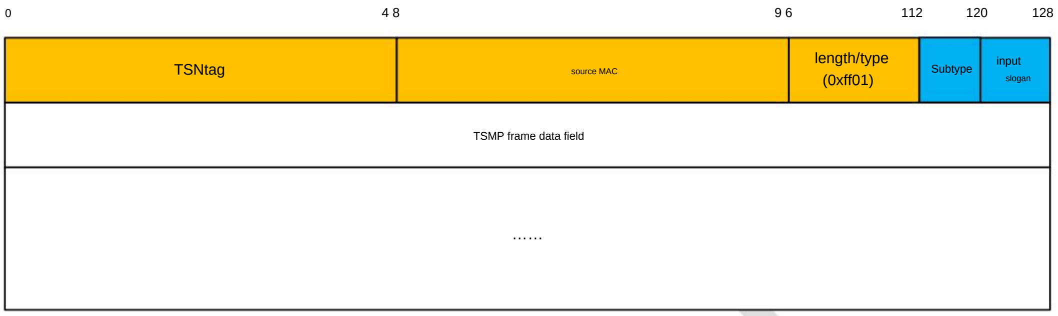


Figure F-1 Format of a TSMP frame

The yellow field in the figure is the Ethernet frame header, the blue field is the TSMP frame header, and the white field is

It is the data field of TSMP frame. The meaning of each field in TSMP frame Ethernet header and TSMP frame header

See Table F-1 and Table F-2 for details.

OpenTSN

Table F-1 Meaning of each field of TSMP frame header

field	Bit width	description
TSNtag	48	The result of mapping TSMP frames.
source mac	48	Not used yet
length/type	16	TSMP frame type is 0xff01 (custom).
Subtype	8	Used to identify different types of TSMP frames, currently including 6 types Type: ARP encapsulation frame, Beacon frame, switch configuration frame, H CP configuration frame, ICMP encapsulation frame, Probe frame.
Enter port number 8		The frame sent by the host to the TSN switch enters the end of the TSN switch. slogan

Table F-2 TSMP frame types

frame type	Subtype the value of	meaning
ARP encapsulated frame 8'h0		ARP frames are encapsulated into TSMP frames for transmission in the network, and A The RP frame is completely stored in the TSMP data field
Beacon frame	8'h1	The switch and network card report the status frame to the controller, and the switch, The status report frame of the network card is completely stored in the TSMP data field
switch configuration frame 8'h2		The controller configures the switch and network card frame, the controller will N MAC configuration frames are encapsulated into TSMP frames, where the NMAC configuration The frame is completely stored in the TSMP data field
HCP configuration frame 8'h3		The frame in which the controller configures the HCP; the configuration information is stored in the TS MP data field.
HCP status report frame 8'h4		The status information reported by HCP is stored in the TSMP data field

Machine Translated by Google

PTP encapsulated frame	8'h5	Encapsulate PTP frames (sync frames, delay_req frames, delay_resp frames) into TSMP frames, where PTP frames are completely stored in TSMP data field

OpenTSN