# TSN network card (TSNNic3.2) design scheme

# (version 1.0)



OpenTSN Open Source Project Team

October 2021

## Version history

| Version revision time Revision | content | file identification |
|---|---|---|
| 1.0 2021.10.7 Organize the logic of HCP and TSE, integrate all Changed design to keep documentation logic code consistent | | OpenTSN3.2 |

content

# 1. Project Overview

This document is an introduction to the TSN network card design in the OpenTSN (version 3.2) open source project.

The project 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 card is counted. TSN network cards are designed by extracting a suitable subset of the TSN standard, It aims to design a set of TSN networks that can meet the diverse and differentiated needs of different fields 's architecture.

## 1.2. Design specifications

- ÿ Support IEEE 802.1AS, 802.1Qch, 802.1Qbv, 802.1Qcc standards
- ÿ Support message mapping and inverse mapping
- ÿ Support standard Ethernet message transmission
- ÿ Supports 4 Gigabit Ethernet interfaces (3 network interfaces, 1 host interface)
- ÿ The setting range of hardware scheduling time slot is [4us, 512us]
- ÿ Time-sensitive traffic supports a maximum delay of 1024 hardware scheduling time slots
- ÿ Support 32 mapping table entries
- ÿ Support 256 reverse mapping entries

## 2. Overall architecture

The overall design of TSNNic3.2 is shown in Figure 2-1. TSNNic3.2 includes time-sensitive Terminal TSE and hardware control point HCP two parts of logic, data processing logic and control logic decoupling.

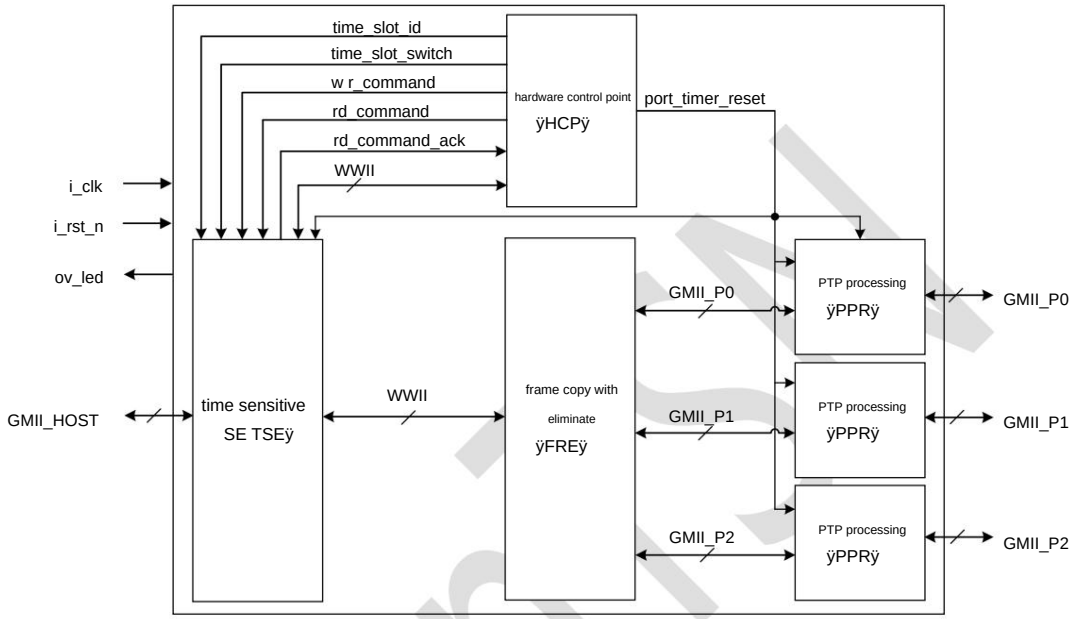


Figure 2-1 Overall architecture of TSNNic3.2

The meanings of the signals in the overall architecture diagram of TSNNic3.2 are shown in Table 2-1.

Table 2-1 Signal meanings in the overall architecture of TSE

| Signal | | Bit width bit | meaning |
|---|---|---|---|
| i_clk | | 1 | clock signal, the clock frequency is 125MHz |
| i_rst_n | | 1 | reset signal, active low |
| ov_led | | 5 | Signals connected to the LED lights on the daughter board |
| GMII_HOST/ WWII / GMII_P0 / GMII_P1 / GMII_P2 | gmii_rx_clk | 1 | GMII receive clock |
| | gmii_rx_data | 8 | GMII receive data |
| | gmii_rx_dv | 1 | GMII receives data valid signal |
| | gmii_rx_er | 1 | GMII receive data error signal |
| | gmii_tx_clk | 1 | GMII transmit clock |
| | gmii_tx_data | 8 | GMII send data |
| | gmii_tx_en | 1 | GMII sends data valid signal |
| | gmii_tx_er | 1 | GMII signals data error |

| Signal | | Bit width bit meaning | |
|---|---|---|---|
| o_s_pulse | | 1 second pulse or millisecond pulse (configurable) | |
| wr_command | wr_command | 204 Write command | |
| | wr_command_wr 1 Write command valid signal | | |
| rd_command | rd_command | 204 Read command | |
| | rd_command_wr 1 Read command valid signal | | |
| rd_command_ack | | 204 read response | |
| port_timer_reset | | 1 Interface timer reset signal, active high | |
| time_slot | | 10 | the current time in a cycle groove |
| time_slot_switch | | 1 Time slot switching signal, active high | |

**TSE (Timing Sensitive** End, Time Sensitive End) module: The main function is to Quasi-Ethernet packets are mapped to TSN packets, and TSN packets are inversely mapped to standard Ethernet packets. Text; Time-sensitive traffic shaping based on 802.1Qbv/Qch.

**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. Decapsulate; parse the configuration message, generate a write command and transmit it to the TSE, and send it to the TSE Send read commands to collect TSE status and report periodically; and in PTP messages Record the time stamp, calculate the time when the PTP message is transmitted in the HCP, that is, the transparent clock, and Accumulate the transparent clock into the transparent clock domain of the PTP packet.

**FRE (Frame Replication and Elimination) mode** Block: The main function is to reserve the received data packets (including time-sensitive packets, bandwidth message, etc.) are copied in three copies and output from three interfaces, and the frame removal function is under development.

**PPR (Precision_time_protocol Process, PTP** processing) module: main The function is to record the timestamp of the PTP packet received by the interface and the time stamp of the PTP packet sent by the interface. Timestamp, calculates and accumulates the transparent clock of PTP packets transmitted in TSE and FRE into the transparent clock domain.

## 2.1. TSE overall design

### 2.1.1. Internal function division of TSE

Figure 2-2 shows the internal function division of the time-sensitive end TSE.

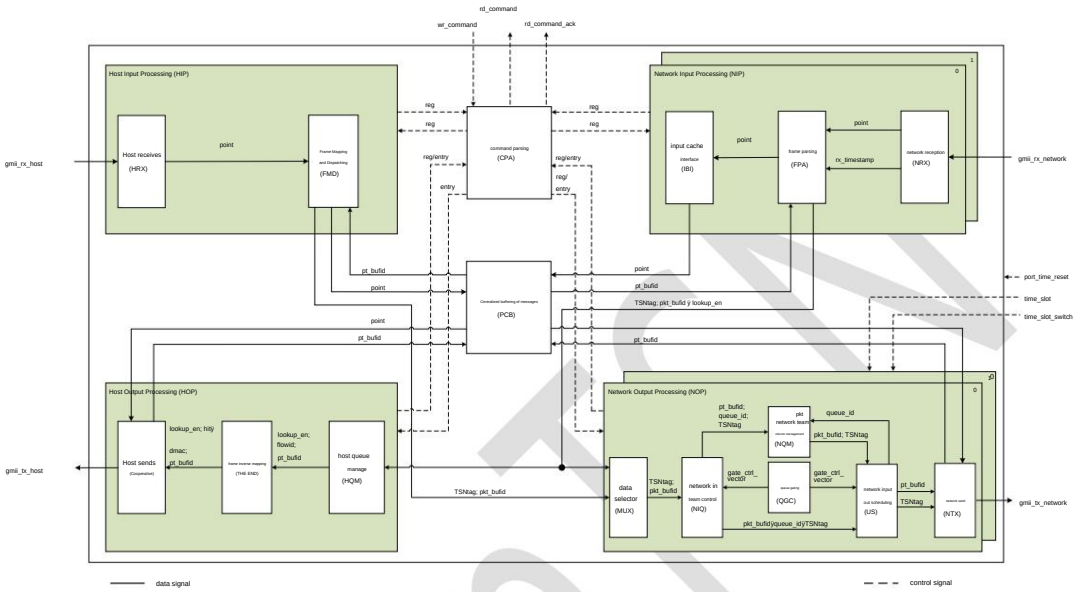


Figure 2-2 Internal function division of TSE

Table 2-2 shows the signal description in the internal function division diagram of the time-sensitive end TSE.

Table 2-2 Signal description in TSE internal function division diagram

| Signal | Bit width description | |
|---|---|---|
| pkt (message centralized buffer reception or signal sent) pkt (non- | 134-bit data with a width of 134 bits, please refer to Appendix A for the specific format | |
| message set cache modulo block received or sent signal) pkt_bufid | 9-bit data with a width of 9 bits, please refer to Appendix A for the specific format | |
| | 9 Message buffer space in the centralized buffer area | |
| TSNtag | 48 The result of the packet lookup table mapping | |
| flowid | 14 | Flow ID of the TSN packet |
| lookup_en | 1 | Message descriptor check inverse mapping table enable signal, of which 1'b1 The message descriptor looks up the inverse mapping table, 1'b0 means the message descriptor Do not look up the inverse mapping table |
| hit | 1 | The message descriptor checks whether the inverse mapping table hits the signal, where 1'b1 The message descriptor hits the inverse mapping table, 1'b0 means the message The inverse mapping table of the text descriptor is not hit |

| Signal | Bit width description |
|---|---|
| queue_id | 3 Queue of TSNtag and pkt_bufid cache of message |
| gcl_ctrl_vector | 8 Gated state of 8 queues |
| rx_timestamp | 19 The network switch interface/control interface receives the PTP packet 19bit timestamp, used to calculate the transparent clock of PTP packets. |
| dmac | 48 The destination mac address of the packet |
| reg | xx configure/report registers |
| entry | xx Items configured/reported |
| time_slot | 10 The time slot in which the current (synchronized) time is located |
| time_slot_switch | 1 Time slot switching signal |
| port_time_reset | 1 Interface time reset signal, active high |
| wr_command | 204 Write command, see Appendix D for the format |
| rd_command | 204 Read command, see Appendix D for the format |
| rd_command_ack | 204 Read command response, see Appendix D for the format |

**HRX (Host** RX, host receiving) module: the main function is to receive the host interface message, switch the message transmission clock domain from the GMII receiving clock domain to the TSE internal clock domain, and discard all packets when the end processing logic is initialized; after the initialization is completed, Only transmit all messages.

**FMD (Frame Mapping and** Dispatching, message mapping and dispatch) module: The main function is to realize packet bit width conversion, packet supervision, packet caching, and table lookup mapping. newspaper The data bit width conversion is to convert the data bit width of each shot of the message from 9bit to 134bit; the message supervision is Indicates that the number of pkt_bufid remaining in the cache module of the packet set is less than the BE packet supervision threshold When the BE message is discarded, the number of pkt_bufid remaining in the message set cache module is small When the RC packet monitoring threshold is exceeded, RC packets and BE packets are discarded; packet buffering refers to Convert the bufid allocated by the cache module in the message set to the base address of the message cache, and write the message into the centralized buffer area; look-up mapping means that TCP/UDP packets use quintuple to look up the mapping The injection table, the pkt_bufid and the table lookup result TSNtag are transmitted to the network switching interface; TCP/UDP packets do not look up the mapping table, but directly use the values of the pkt_bufid and DMAC fields (ie TSNtag semantics) to the network switch interface.

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

All packets that need to be forwarded by the architecture are centrally cached (the format of the packet buffer area is shown in Table 2-3).

shown), perform cache management on the free address buffer pkt_bufid (free address buffer

The format is shown in Table 2-4). Each pkt_bufid needs to design a counter to

The number of output ports is counted, and each time pkt_bufid is released, the counter is decremented by one;

When the pkt_bufid is released after use, it is necessary to check the value of the counter corresponding to the pkt_bufid.

Only when the counter value is 0 can the pkt_bufid be written into the free address buffer area. When it is not 0, it means

that the pkt_bufid has a multicast packet and The message has not

Output from all ports that require output.

The message buffer area divides the 1024KB space into 512 message buffer blocks.

The message cache block can cache a 2KB message.

Table 2-3 Data format of the message buffer area

| Address[8:0] | Content[133:0] |
|---|---|
| 0-127 | 1st message buffer block |
| 128-255 | 2nd message buffer block |
| … | .. |
| 65408-65535 | The 512th message buffer block |

Table 2-4 Free address buffer data format

| name | Meaning Remarks The idle messages in the current "message buffer area" | |
|---|---|---|
| pts_bufid [8: 0] | are implemented using RAM, and the ID number of the block is delayffered. is 512. | |

**HQM (Host Queue** Manage, host queue management) module: the main function is to

The message descriptor transmitted from the network interface is cached, and the frame inverse mapping module is received after receiving the frame.

After the ready signal is received, the message descriptor in the queue is read out.

**FIM (Frame Inverse Mapping** frame inverse mapping) module: The main function is to

To find the message descriptor of the inverse mapping table to look up the table, check whether the pkt_bufid and the table lookup are hit.

The output of the table lookup result when the flag bit and the table lookup hit; for the message that does not need to look up the inverse mapping table

Descriptors are output directly.

**HTX (Host** TX, host sending) module: the main function is to collect the buffer from the message

Read the message and release the pkt_bufid, and modify the TSN message hit by the inverse mapping table to

Standard Ethernet packets, discard TSN packets that are not hit by the inverse mapping table, and will

The standard Ethernet message in the mapping table is directly output. When outputting the message, the bit width of the message is changed from

134bit is converted to 8bit, and the preamble and frame start character are added in front of the Ethernet header of the message and controlled

Control the message output interval, and convert the message transmission clock domain from the TSNNic internal clock domain

Transmit clock domain for GMII. When reading the message, map pkt_bufid to the message buffer

The base address of the buffer in the memory area, and read the message from the buffer area of the message set according to this address,

When receiving the first data of the packet, it will check the TSN packet hit by the inverse mapping table.

TSNtag is restored to DMAC and the Ethernet type of the message is changed from 0x1800 to 0x0800,

If the message does not check the inverse mapping table, the content of the message does not need to be modified; after the last message received

When shooting data, pass pkt_bufid to the packet centralized cache module for release, so as to facilitate subsequent

Used for packets entering the TSE.

**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 TSE

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 TSE is in the initialization phase, discard all the messages;

If the TSE is in the configuration phase, it transmits standard Ethernet packets and configuration packets in TSN packets.

If the TSE is in the clock synchronization phase, it transmits standard Ethernet packets and TSN packets.

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

messages and TSN messages.

**FPA (Frame** Parse, Frame Parse) module: The main function is to receive the pkt_bufid allocated by the central buffer module of the

message, convert 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 message types and supervise RC messages and BE messages.

**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.

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

The message descriptor is written to the network queue management module for buffering. This module receives

The message type in the message descriptor and the gated state from the queue-gating module describe the message.

The message descriptor and queue ID are sent to the network output caller at the same time.

degree module, so that the network output scheduling module manages the queue head address; and according to the write

The information of entering the queue and the scheduled queue ID transmitted from the network output scheduling module are used to match the queue.

The number of bufids currently written in is managed.

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

It can be a centralized buffer management for the queue of the network output port

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

read the corresponding gating state in the gating list by using the office clock, and convert the gating list (the

The gating status of the 8 queues in the format shown in Table 2-5) is sent to the network queue control module

and network output scheduling module.

Table 2-5 Gating List Data Format

| name | meaning | |
|---|---|---|
| gate_ctrl_vector[7:0] Gate vector, corresponding to 8 gate information. | | Remarks are implemented using RAM with a depth of 1024 |

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

It is to extract the message descriptor from the network queue management module according to the scheduled queue information. Book

The module needs to transmit according to whether the current queue is empty, the queue priority and the queue gate control module

The gated state of the queue is obtained to obtain a queue with the highest priority scheduling, and the first address of the queue is used as the read ground.

address, read the message descriptor from the network queue management module, and send the message corresponding to the queue head address

The message descriptor is output to the network sending module, and the pkt_bufid in the read message descriptor is

as the new head address of the queue.

**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, change the message transmission clock domain from the TSNNic internal clock domain

Switch to GMII to send clock domain, construct frame preamble and frame starter, and control message output

interval. When reading a message, map pkt_bufid into a message and buffer it in the centralized buffer

store the base address, and read the message from the message set cache module according to this address, and according to the report

The information in the message descriptor is used to modify the DMAC and Ethernet types that are mapped into TSN messages.

Change; convert the bit width of the read message from 134bit to 8bit, after receiving the last beat of data

When the pkt_bufid is transmitted to the message set cache module for release, so as to facilitate subsequent processing

It is used for packets entering the TSE.

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

The write command (see Appendix D for the write command data format) is parsed to realize the local register,

Configuration of the mapping table and inverse mapping table; according to the received read command (see the read command data format

Appendix D) to read the corresponding register or table entry, and encapsulate the read data in the read response (read response

The data format should be output to the HCP module in Appendix D).

### 2.1.2. Processing flow of TSE

The overall architecture of TSE includes 1 host interface, 1 network switch interface and 1 controller control interface, the host interface is connected to an external device (which needs to be connected to the TSN network), and the network communication The switch interface is connected to the external TSNSwitch, and the control interface is connected to the HCP inside the TSNNic. catch. The frame processing flow is described in detail below.

#### 2.1.2.1. The frame processing flow of the host interface entering and exiting the network switching interface

The frame input by the host interface is processed across the clock domain in the host receiving module (HRX). And add 1 bit of head and tail flags to each beat data, and then according to the stage of TSE Decide whether to transmit the message, that is, when the TSE is in the initialization pkt_bufid stage, discard all message; TSE transmits all messages to the frame map only after the initialization of pkt_bufid is completed. Shooting and Dispatching Module (FMD).

After the frame mapping and dispatch module (FMD) receives the message, it will The width is converted from 9bit to 134bit, and the message is allocated by the central buffer module (PCB). pkt_bufid is converted into the base address of the message cache, and the message is written to the message set buffer area; At the same time, the message is mapped: for TCP/UDP message (Ethernet type is 0x0800, And the IP protocol is 0x6/0x11), this module first extracts the quintuple of the message, and uses the quintuple to check the Find the mapping table, if the table lookup hits, map the message to a TSN message, and obtain the table lookup result. If the TSNtag is not hit, it is determined that the packet is a standard Ethernet packet, and the The packet type is fixed to the BE packet type; for non-TCP/UDP packets, this packet does not need to be Look up the table mapping, determine that the packet is a standard Ethernet packet, and its packet type is fixed as BE packet. message type; then construct the message descriptor, and cache the rest of the module according to the message set The number of pkt_bufid monitors the RC and BE messages and caches them in the message set When the number of remaining pkt_bufid in the module is less than the BE packet supervision threshold, the BE packet is discarded, and when the remaining pkt_bufid in the message set cache module is less than the RC packet supervision threshold,

Discard RC messages and BE messages; after message policing, send message descriptors to the network

Data selector for switching output interface.

The data selector selects the descriptors from the host interface and the control interface, and outputs

Out to the network enqueue control module.

After the network enqueue control module receives the message descriptor, it transmits it according to the queue gate control module.

The gated state and the message type in the descriptor are queued, and the message descriptor is passed to the

Network queue management module for caching.

The network output scheduling module is based on the gate control state and queue priority transmitted from the queue gate control module.

level and queue status (whether it is empty or not) to get the queue ID of the most prioritized scheduling, and assign this queue

The first address in the column ID is sent to the network queue management module, waiting for the network queue management module to

The message descriptor in the corresponding queue is read out, and the current queue head address is output to the network.

Send the module, and use the pkt_bufid in the read message descriptor as the first address of the linked list.

The network sending module sends the cache module to the message set according to the pkt_bufid in the message descriptor

Extract the message. The network sending module maintains two 134bit ping-pong registers,

The messages read from the buffer area of the message set are buffered in these two registers in turn,

When the data of the first beat of the message is reached, compare the value of the DMAC field in the first beat of the data with the description of the message.

Whether the TSNtag/DMAC information in the descriptor is equal, if they are equal, the message is judged as standard

Ethernet packet, no need to modify the content of the packet; if it is not equal, it is determined that the packet is mapped

For the TSN message, store the TSNtag in the message descriptor in the first beat data of the message

DMAC field, and change the Ethernet type of the message from 0x0800 to 0x1800;

The data bit width in the register is converted from 134bit to 8bit, and the message is received from GMII after crossing the clock domain.

port output, when the last data of the message is received, the pkt_bufid is passed to the message set

The cache module is released.

#### 2.1.2.2. Frame processing flow of incoming and outgoing host interfaces on network switching interfaces

The network receiving module (NRX) of the network switching interface is in the data of each frame of the received message
Add 1bit head and tail flags and perform cross-clock domain processing, according to the Ethernet type of the message
Identify standard Ethernet messages and TSN messages (Ether type 0x1800, 0x98f7 and
0xff01) and decide whether to transmit the message according to the stage of the end processing logic: if the end
If the processing logic is in the initialization stage, all packets are discarded; if the processing logic is in the configuration stage,
It transmits standard Ethernet packets and configuration packets in TSN packets; if the end processing logic is in
In the clock synchronization stage, standard Ethernet packets and non-time-sensitive packets in TSN packets are transmitted;
If the end processing logic is in the normal working stage, it receives standard Ethernet packets and TSN packets.

After the frame parsing module (FPA) receives the message from the network receiving module (NRX),
Convert the message bit width from 9bit to 134bit, and output the 134bit data and its pkt_bufid
output to the input buffer interface module; at the same time, the DMAC field data of the message is extracted, and the
134bit data, the bufid allocated by the buffer module in the message set and the enable bit group of the inverse mapping table
form a message descriptor; then supervise the RC message and the BE message, if the message is concentrated in the buffer
When the number of remaining pkt_bufid in the memory module (PCB) is less than the RC message supervision threshold, the
Both the RC message descriptor and the BE message descriptor are discarded. If the message is centralized in the cache module (PCB)
When the number of remaining pkt_bufid in the BE packet is less than the BE packet supervision threshold, the BE packet descriptor will be discarded.
Discarded; then the message descriptor is dispatched according to the message Ethernet type, and the standard Ethernet
The message descriptor or the mapped IP message descriptor is transmitted to the host interface.

The input buffer interface module (IBI) converts the pkt_bufid to the message in the centralized buffer area
Central cache base address, write the message to the centralized cache area, each time a beat of 134bit data is received
When the message is received, the write request is output to the centralized cache module of the message.
After the response, the writing of one beat of data is completed.

The host queue management module (HQM) will receive the pkt_bufid, check the reverse mapping table
The enable bit and the stream ID in the TSNtag are written to the queue for buffering, and the frame is reverse mapped after receiving the frame.

After the ready signal from the module (FIM), the pkt_bufid, the enable bit of the inverse mapping table and the stream ID in the TSNtag are output to the frame inverse mapping module.

The frame inverse mapping module determines whether the message descriptor is based on the enable bit of the inverse mapping table. Need to look up the inverse mapping table, if the enable bit of the inverse mapping table is high, use the stream ID to find the inverse Mapping table, and pkt_bufid, whether the table lookup hits the flag, and the table lookup result when the table lookup hits DMAC output to the host transmission module (HTX); if the inverse mapping table enable bit is low, Then there is no need to look up the inverse mapping table, and output the pkt_bufid and the information about the unchecked inverse mapping table of the message. Send the module (HTX) to the host.

The host sending module (HTX) judges whether the message descriptor check inverse mapping table hits and Whether the DMAC needs to be replaced, if the packet descriptor lookup table fails, the pkt_bufid will be released Put it into the message set cache module, but do not need to read the message from the message set cache module; if If the message descriptor lookup table hits, the received pkt_bufid will be converted into the base of the message read. address, read the message from the buffer area of the buffer module (PCB) in the message set, and receive the message At the last shot of data, pass pkt_bufid to the message set cache module for release; When the first data of the message is read, the DMAC will replace the message with the result of the inverse mapping table TSNtag, change the Ethernet type from 0x1800 to 0x0800; if the message descriptor is not found Inverse mapping table, the received pkt_bufid is converted into the base address of the message read, from the message The message is read from the buffer area of the buffer module (PCB) in the corpus, and the last beat count is received. When the data is received, the pkt_bufid is passed to the packet centralized cache module for release. put from the centralized cache The data bit width of each shot of the message read from the area is converted from 134bit to 8bit, and the message is transmitted in the clock domain. The internal processing clock domain of the TSE is converted to the GMII sending clock domain, and the message is output.

#### 2.1.2.3. Frame processing flow of incoming and outgoing control interfaces of network switching interfaces

The network receiving module (NRX) of the network switching interface is in the data of each frame of the received message Add 1bit head and tail flags and perform cross-clock domain processing, according to the Ethernet type of the message

Identify standard Ethernet messages and TSN messages (Ether type 0x1800, 0x98f7 and
0xff01) and decide whether to transmit the message according to the stage of the end processing logic: if the end
If the processing logic is in the initialization stage, all packets are discarded; if the processing logic is in the configuration stage,
It transmits standard Ethernet packets and configuration packets in TSN packets; if the end processing logic is in
In the clock synchronization stage, standard Ethernet packets and non-time-sensitive packets in TSN packets are transmitted;
If the end processing logic is in the normal working stage, it receives standard Ethernet packets and TSN packets.
For PTP packets, you need to record the timestamp of the interface receiving the PTP packet, and store it in the
in TSNtag.

After the frame parsing module (FPA) receives the message from the network receiving module (NRX),
Convert the message bit width from 9bit to 134bit, and output the 134bit data and its pkt_bufid
output to the input buffer interface module; at the same time, the DMAC field data of the message is extracted, and the
134bit data, pkt_bufid allocated by the buffer module in the message set, and inverse mapping table check enable
The bits form the message descriptor; then the RC message and the BE message are supervised, if the message set
When the number of remaining pkt_bufid in the buffer module (PCB) is less than the RC packet supervision threshold,
Discard both the RC message descriptor and the BE message descriptor. If the message is centralized in the cache module
When the number of remaining pkt_bufid in (PCB) is less than the BE packet supervision threshold, the BE packet will be
The descriptor is discarded; the message descriptor is then dispatched according to the message Ethernet type, and future
The PTP message descriptor or TSMP message descriptor from the switch is transmitted to the control interface output.
out processing logic.

The input buffer interface module (IBI) converts the pkt_bufid to the message in the centralized buffer area
Central cache base address, write the message to the centralized cache area, each time a beat of 134bit data is received
When the message is received, the write request is output to the centralized cache module of the message.
After the response, the writing of one beat of data is completed.

The data selector outputs the descriptor from the network switch interface to the network enqueue control
module.

After the network enqueue control module receives the message descriptor, it transmits it according to the queue gate control module.
The gated state and the message type in the descriptor are queued, and the message descriptor is passed to the
Network queue management module for caching.

The network output scheduling module is based on the gate control state and queue priority transmitted from the queue gate control module.
level and queue status (whether it is empty or not) to get the queue ID of the most prioritized scheduling, and assign this queue
The first address in the column ID is sent to the network queue management module, waiting for the network queue management module to
The message descriptor in the corresponding queue is read out, and the current queue head address is output to the network.
Send the module, and use the pkt_bufid in the read message descriptor as the first address of the linked list.

The network sending module sends the cache module to the message set according to the pkt_bufid in the message descriptor
Extract the message. The network sending module maintains two 134bit ping-pong registers,
The messages read from the buffer area of the message set are buffered in these two registers in turn,
When the data of the first beat of the message is reached, compare the value of the DMAC field in the first beat of the data with the description of the message.
Whether the TSNtag/DMAC information in the descriptor is equal, if they are equal, the message is judged as standard
Ethernet packet, no need to modify the content of the packet; if it is not equal, it is determined that the packet is mapped
For the TSN message, store the TSNtag in the message descriptor in the first beat data of the message
DMAC field, and change the Ethernet type of the message from 0x0800 to 0x1800;
The data bit width in the register is converted from 134bit to 8bit, and the message is received from GMII after crossing the clock domain.
port output, when the last data of the message is received, the pkt_bufid is passed to the message set
The cache module is released.

#### 2.1.2.4. The frame processing flow of the control interface entering and exiting the network switching interface

The network receiving module (NRX) of the control interface increases the data in each frame of the received message
1bit head and tail identification bits and cross-clock domain processing, identified according to the Ethernet type of the message
Standard Ethernet packets and TSN packets (Ethernet types are 0x1800, 0x98f7 and 0xff01)
And decide whether to transmit the message according to the stage of the end processing logic: if the end processing logic

In the initialization phase, all packets are discarded; if the end processing logic is in the configuration phase, the transmission Configuration messages in quasi-Ethernet messages and TSN messages; if the end processing logic is in clock synchronization Phase, transmits standard Ethernet packets and non-time-sensitive packets in TSN packets; The logic is in the normal working stage, receiving standard Ethernet packets and TSN packets.

After the frame parsing module (FPA) receives the message from the network receiving module (NRX), Convert the message bit width from 9bit to 134bit, and output the 134bit data and its pkt_bufid output to the input buffer interface module; at the same time, the DMAC field data of the message is extracted, and the 134bit data, the bufid allocated by the buffer module in the message set and the enable bit group of the inverse mapping table form a message descriptor; then supervise the RC message and the BE message, if the message is concentrated in the buffer When the number of remaining pkt_bufid in the memory module (PCB) is less than the RC message supervision threshold, the Both the RC message descriptor and the BE message descriptor are discarded. If the message is centralized in the cache module (PCB) When the number of remaining pkt_bufid in the BE packet is less than the BE packet supervision threshold, the BE packet descriptor will be discarded. Discarded; then the message descriptor is dispatched according to the message Ethernet type, and will be sent from the HCP The PTP message descriptor or TSMP message descriptor is transmitted to the output of the network switch interface Logic.

The input buffer interface module (IBI) converts the pkt_bufid to the message in the centralized buffer area Central cache base address, write the message to the centralized cache area, each time a beat of 134bit data is received When the message is received, the write request is output to the centralized cache module of the message. After the response, the writing of one beat of data is completed.

The data selector outputs the descriptor from the control interface to the network enqueue control module.

After the network enqueue control module receives the message descriptor, it transmits it according to the queue gate control module. The gated state and the message type in the descriptor are queued, and the message descriptor is passed to the Network queue management module for caching.

The network output scheduling module is based on the gate control state and queue priority transmitted from the queue gate control module. level and queue status (whether it is empty or not) to get the queue ID of the most prioritized scheduling, and assign this queue

The first address in the column ID is sent to the network queue management module, waiting for the network queue management module to

The message descriptor in the corresponding queue is read out, and the current queue head address is output to the network.

Send the module, and use the pkt_bufid in the read message descriptor as the first address of the linked list.

The network sending module sends the cache module to the message set according to the pkt_bufid in the message descriptor

Extract the message. The network sending module maintains two 134bit ping-pong registers,

The messages read from the buffer area of the message set are buffered in these two registers in turn,

When the data of the first beat of the message is reached, compare the value of the DMAC field in the first beat of the data with the description of the message.

Whether the TSNtag/DMAC information in the descriptor is equal, if they are equal, the message is judged as standard

Ethernet packet, no need to modify the content of the packet; if it is not equal, it is determined that the packet is mapped

For the TSN message, store the TSNtag in the message descriptor in the first beat data of the message

DMAC field, and change the Ethernet type of the message from 0x0800 to 0x1800;

The data bit width in the register is converted from 134bit to 8bit, and the message is received from GMII after crossing the clock domain.

port output, when the last data of the message is received, the pkt_bufid is passed to the message set

The cache module is released.

## 2.2. HCP Overall Design

### 2.2.1. HCP internal function division

The internal function division of the hardware control point HCP is shown in Figure 2-3.

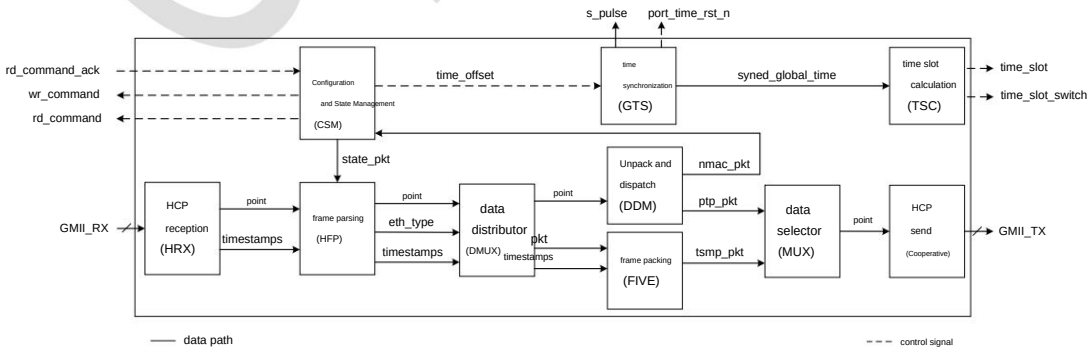


Figure 2-3 Internal function division of HCP

The meanings of the signals in the HCP internal function division diagram are shown in Table 2-6.

Table 2-6 Signal definitions in HCP internal function division

| Signal | Bit width meaning |
|---|---|
| pt / ptp_pkt / tsmp_pkt / nmac_pkt | 9 Message, please refer to Appendix A for the specific format |
| state_pkt | 8 Status report message |
| wr_command | 204 Write command, see Appendix B for the format |
| rd_command | 204 Read command, see Appendix B for the format |
| rd_command_ack | 204 Read command response, format see Appendix B |
| timestamps | 19 The time when the HCP receiving module receives the PTP message stamp |
| eth_type | 16 Telegram Ethernet Type |
| time_offset | 49 The clock compensation value of the local clock and the master clock, Among them, the high 1bit indicates the positive and negative of the clock compensation value |
| syned_global_time | 48 Synchronized clock |
| time_slot | 10 The time at which the current (synchronized) time is located groove |
| time_slot_switch | 1 Time slot switching signal |
| s_pulse | 1 Clock synchronization pulse signal, can be configured as second pulse pulse or millisecond pulse |
| port_time_reset | 1 Interface time reset signal, active high |

The following will introduce the functions of the internal modules in HCP one by one.

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

The incoming message, switch the message transmission clock domain from the GMII receive clock domain to the HCP internal clock domain

Clock domain, and records the time when the control interface receives time synchronization packets. Controlled by hardware

The stage of the point logic to decide whether to transmit the message: if the hardware control point logic is in the initial

In the configuration stage, all packets are discarded; if the hardware control point logic is in the configuration stage, only the configuration

Set the message; if the hardware control point logic is in the clock synchronization stage, transmit the non-time-sensitive message;

If the logic of the hardware control point is in the normal working stage, all messages are received.

**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 (chip configuration frame, PTP

encapsulation frames) and their descriptors are dispatched to the decapsulation and dispatch module during the clock synchronization phase/positive Non-TSMP packets (standard Ethernet non-IP packets, PTP

packets, NMAC status report packets) are dispatched to the frame encapsulation module for encapsulation processing.

**DDM (Decapsulation_Dispatch_Module, decapsulation and dispatch) module:**

Decapsulate the received TSNNic configuration frame and PTP encapsulated frame into NMAC configuration respectively frames, PTP frames, and dispatch NMAC configuration frames to the configuration and state management module, record The timestamp of the transmission of the PTP frame, which dispatches the PTP frame to the data selector module.

**FEM (Frame_Encapsulation_Module, frame encapsulation) module: main function**

It can be to encapsulate the received PTP frame and NMAC status report frame into the TSMP message,

For PTP frames, the received timestamp is recorded, the transparent clock is calculated and accumulated to the transparent time clock domain.

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

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

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.

**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.

### 2.2.2. Process flow of HCP

The processing flow of hardware control point logic includes packet encapsulation and decapsulation, configuration packet

parsing and reporting of status packets.

#### 2.2.2.1. Configure Packet Parsing

HCP will parse the chip configuration message, realize the register,

table configuration.

After the HCP receiving module receives the chip configuration message, it converts the bit width of each data shot of the message to

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

Discard or transmit: If the HCP is in the initialization phase/configuration phase, discard the chip configuration

message; if the HCP is in the clock synchronization stage/normal working stage, it transmits TSMP messages.

The frame parsing module extracts the Ethernet type from the chip configuration message, and combines the Ethernet type with the The received time constitutes the message descriptor, and now the message and its descriptor are written into the fifo, in the HCP 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 TSMP according to the Ethernet type of the packet as 0xff01 message (chip configuration message is a type of TSMP message), when the HCP is in the During the initialization phase, the chip configuration packet is discarded, and when the HCP is in the configuration phase/the clock is the same In the step stage/normal working stage, the chip configuration message is output to the decapsulation and dispatch module.

The chip configuration message is decapsulated and dispatched in the decapsulation and dispatch module. chip The configuration frame is decapsulated into an NMAC configuration frame and dispatched to the configuration and state management module for 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 terminal processing logic TSE.

#### 2.2.2.2. Status message reporting

The HCP will cycle through the status of the end processing logic TSE and the hardware control point logic HCP. Periodic reporting.

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. The address of the read data; when the read response from the end processing logic TSE is received, the read The data is read at the parsing section in the response and output to the frame parsing module as the data in the status message. yuan.

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

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 HCP sending module.

The HCP sending module switches 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 control 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.

The decapsulation and dispatch module decapsulates and dispatches the PTP encapsulated packets. the PTP Decapsulate the encapsulated packets into PTP packets, record the sending timestamp of the PTP packets, and convert the PTP packets into PTP packets. The message is dispatched to the data selector.

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 HCP 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

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 HCP sending module.

The HCP sending module switches 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.

# **Appendix A.** Data Format Definition

## A.1. Message Format

There are two data bit widths per beat of the message transmitted within TSNNic, one is

9bit; another bit width is 134bit. The message formats of different bit widths are described below.

The format of pkt with a data bit width of 9 bits per shot is shown in Table A-1 below.

Contains 1bit head and tail flags and 8bit message valid data, of which 1'b1 in the head and tail flags

Indicates the data of the first or last beat of the message, and 1'b0 represents the data of the middle beat of the message.

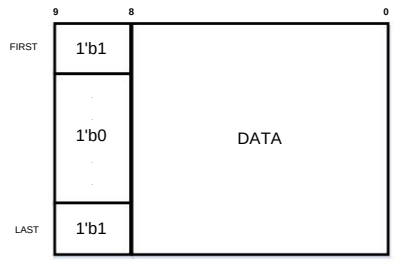


Figure A-1 pkt format with a bit width of 9 bits

The format of pkt with a data bit width of 134 bits per shot is shown in Table A-2 below.

The data includes 2bit head and tail identifier, 4bit invalid byte number and 128bit message valid data, among which

2'b01 in the header and tail marks indicates the first beat of the packet, and 2'b11 indicates the middle beat of the packet

data, 2'b10 represents the last beat of the message; the number of invalid bytes is used to identify the number of beats of the message

Invalid number of bytes in data.

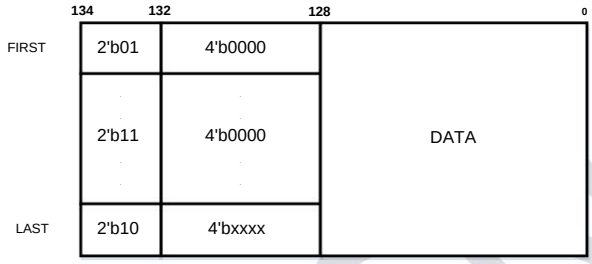


Figure A-2 pkt format with a bit width of 134bit

## A.2. TSN Label Format

Table A-3 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-4 for other message types) |
| flow id/local id 14 | | [44:31] | Stream ID or local ID, where time-sensitive Use of static traffic such as traffic and bandwidth reservation traffic flow id, management control packets, time synchronization packets etc. use 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-4 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 means RC message, 3'b101 Indicates NMAC packet, 3'b110 indicates BE score group, 3'b111: shard to reorganize |
| flow id/ | 14 | [44:31] Stream ID or local ID, where time-sensitive streams | |

| | | | |
|---|---|---|---|
| local id | | | Static traffic such as traffic volume and bandwidth reservation traffic use flow id, and management control packets, time synchronization packets, etc. with local id |
| 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] | Buffering of time-sensitive streams at the sender end of the TSN network card address |
| submit addr 5 | | [4:0] | Cache of time-sensitive streams at the receiving end of the TSN network card address |

## A.3. Descriptor format

Table 4-5 Descriptor data format

| field | Bit width bit position description | | |
|---|---|---|---|
| TSNtag | 48 | [56:9] TSN tag of TSN message | |
| pkt_bufid | 9 | [8:0] The buffer space of the message in the centralized buffer area | |

**Appendix B.** Configuration Instructions

## B.1. Hardware Address

The TSN network card needs to be configured including control registers and tables, each control register and each Each entry has a unique hardware address, and the controller can Configure different control registers or table entries in the network card. The hardware address is 27bit, its format See Table B-1 for details, where the upper 7 bits are used to identify the control register or the module where the entry is located. number, 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.

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

| Modules CSM TIS TSS QGC GTS FLT PMD FIM | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Module ID 0x0 0xe | | 0x1 | 0x2 | 0x3 | 0xb | 0xc | 0xd | |

## B.2. Configuration Content

The configuration contents of the TSN network card include control registers and tables, including the mapping table, Inverse mapping table, injection schedule, commit schedule 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 descriptions of the control registers are detailed in Table B-3.

Table B-3 Control Register Description

| Classification | hardware address | | | bit width bit description | |
|---|---|---|---|---|---|
| | Module ID Module Name 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 During the configuration phase, only receive and process configuration Set message; 2'd2 means TSN hardware In the time synchronization phase, the receiving Handle non-ST packets; 2'd3 means TSN The hardware is in normal working stage, connect Receive and process all messages |

| | | | | | |
|---|---|---|---|---|---|
| time synchronization | | 20'h1 | time_offset_h | 17 | When time_offset_h[16] 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'b1, it means that the offset value between the slave clock and the master clock is positive. The high 16bit of the offset value between the clock and the main clock, {time_offset_h[15:0], time_offset_l[31:7]} The unit is us |
| | | 20'h0 | time_offset_l | 32 The lower 32 bits of the offset value of the slave clock and the master clock, {time_offset_h[15:0], time_offset_l[31:7]} unit is us, {time_offset_l[6:0]} unit is 8ns 24 The slave clock and master clock | |
| | | 20'hb | time_offset_ period | offset Compensation period of the shift value; TSN hardware will compensate the local clock once every compensation period, the unit is 8ns 10 Time slot length, the unit is us; | |
| message scheduling | | 20'h2 | time_slot_ length | the value range is [4us, 512us] and the value must be 2n 802.1Qbv or 802.1Qch scheduling model selection signal, where 1'b0 | |
| | | 20'h5 | qbv_or_qch | 1 | represents the 802.1Qbv scheduling model and 1'b1 represents the 802.1Qch scheduling model |
| | | 20'h8 | inject_slot_ period | 11 | Injection period, in time slots |
| | | 20'h9 | submit_slot_ period | 11 | Commit cycle, in 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 TSN hardware is less than the rc packet supervision threshold, the rc packet is discarded and 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 be packet monitoring threshold, be packets are discarded. 9 Unmapped packet monitoring |
| | | 20'he | unmap_ regulation_ value | threshold, when The number of free pkt_bufid in TSN hardware is less than unmapped packet supervision | |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | When the threshold is exceeded, the unmapped packets are discarded |
| Status report | | 20'h6 | report_type | 16 Reporting | type reporting |
| | | 20'h7 | report_en | 1 | enable, where 1'b0 means the reporting function is enabled, 1'b1 means the reporting function is disabled 12 Reporting |
| | | 20 ha | report_period | period, TSN | hardware only supports two configuration values, 12'd1 and 12'd1000, of which 12'd1 It means the reporting period is 1024us, 12'd1000 means the reporting period is 1024*1024us |

B.2.2. Mapping Table

The mapping table is used to map the standard Ethernet packets received by the host interface.

Table descriptions are detailed in Table B-4.

Table B-4 Mapping table description

| Hardware Address | | name | Bit width (bit) | describe |
|---|---|---|---|---|
| Module Address ID | Module | | | |
| 7'hd (Frame Mapping and Dispatching Module) | 20'h00 0000 ~ 20'h00 001F | map_table _entry_N | [151:144] | IP protocol in quintuple [143:112] |
| | | | Source IP in quintuple [111:80] | Destination |
| | | | IP in quintuple [79:64] | Source port in |
| | | | quintuple [63:48] | Five The destination port |
| | | | in the tuple | |
| | | | [47:45] | Packet type in TSNtag pkt_type Flow number |
| | | | [44:31] | in TSNtag flow_id/internal mac address imac Packet sequence number in TSNtag sequence_id; This field is |
| | | | [30:15] | not stored in hardware ram. |
| | | | [14] | Fragment flag bit frag_flag in TSNtag; this field is not stored in hardware ram. |
| | | | [13:10] | Fragment number frag_id in TSNtag; this field is not stored in hardware ram. |
| | | | [9:5] | ST message injection address in TSNtag injection_addr ST message submission address |
| | | | [4:0] | in TSNtag submit_addr |

Since the content of the controller configuration mapping table entry contains the entire 48bit TSNtag information, But currently TSNNic only pays attention to the packet type pkt_type, flow ID flowid/ Internal mac address imac, ST message injection address injection_addr and ST message submission address submit_addr, so the RAM of the cache mapping table in the frame mapping and dispatch module only buffers Store quintuple and TSNtag information that TSNNic pays attention to, cache the RAM format of the mapping table As shown in Table B-5, the RAM is 131 bits wide and 32 deep.

Table B-5 RAM Format of Cache Mapping Table

| index address | entry content | location field | | meaning |
|---|---|---|---|---|
| from address 5'd0 to 5'd31 by times to find, until read in the entry Rongzhong five yuan Group with Frame Five tuple hit or table entry none effect | 5tuple | [130:123] | ip_protocol | IP protocol in quintuple |
| | | [122:91] | src_ip | Source IP in quintuple |
| | | [90:59] | dst_ip | The destination IP in the quintuple |
| | | [58:43] | src_port | source port in quintuple |
| | | [42:27] | dst_port | The destination port in the quintuple |
| | TSNtag follow field | [26:24] | pt_type | message type, where 3'h0, 3'h1 and 3'h2 means ST grouping, 3'h3 means RC packet, 3'h4 means PTP packet, 3'h5 means NMAC packet, 3'h6 table Indicates BE grouping, 3'h7 indicates the need for recombination Group |
| | | [23:10] | flow_id/imac | flow number/internal mac address |
| | | [9:5] | injection_addr | Time-sensitive packets are waiting to be sent at the source Cache address when scheduling |
| | | [4:0] | submit_addr | Time-sensitive packets are waiting to be received at the terminal Cache address when scheduling |

B.2.3. Inverse mapping table

The inverse mapping table is used to inversely map the TSN packets that need to be sent from the host interface to the standard Ethernet packet, the reverse mapping table supports up to 256 entries, and the content of the entry is the 14-bit flowid, 48bit dmac and 9bit outport, see Table B-6 for the description of the inverse mapping table.

Table B-6 Inverse mapping table description

| Hardware Address Name | Location Description | | |
|---|---|---|---|

| module ID | Module address | | | |
|---|---|---|---|---|
| 7'df (frame inverse mapping module) | 20'h00 0000 . 20'h00 00FF | regroup_ table_entry _N | [77:64] Flow number [63:16] | |
| | | | The real destination MAC address of the message [15:9] | |
| | | | Reserved [8:0] The message output interface number, | |
| | | | using the bitmap frame reverse mapping module to use RAM to | |

cache the reverse mapping table, RAM The bit width is 71bit,

With a depth of 256, the RAM format is shown in Table B-7.

Table B-7 RAM format of cache reverse mapping table

| index address | Field location description | | |
|---|---|---|---|
| Read the contents of the table entries in sequence from address 8'd0 to 8'd255. When the flowid in the readout table entry is the same as the flowid in the message, it means that the table lookup is hit and the table lookup is over. | flowid | [70:57] The actual destination | |
| | dmac | [56:9] | MAC address of the flow number packet is the output interface number |
| | outport | [8:0] | of the packet, using bitmap |

B.2.4. Gated List

In order to implement the 802.1Qbv scheduling model, TSNNic designs a network output interface Gate control list, the gate control list supports up to 1024 entries, and the description of each entry is as follows shown in B-8.

Table B-8 Gating List Description

| Hardware Address | | name location description | | |
|---|---|---|---|---|
| Module Module ID | Address | | | |
| 7'd3 (queue gating module) | 20'h00 0000 . 20'h00 03FF | gate_table_ entry_N | [7] | The gated state of the 7th queue, where 0 means the gate is closed, 1 means the gate is open The gated state of the |
| | | | [6] | sixth queue, where 0 means the gate is closed, and 1 means the gate is turned on. The gated state of the fifth |
| | | | [5] | queue , where 0 means that the gate is closed, 1 means that the gate is on the gated state of the fourth queue, 0 |
| | | | [4] | means that the gate is closed, 1 means that the gate is open to the gated state of the third queue, and 0 means |
| | | | [3] | that the gate is closed, 1 means that the gate is open to the gated state of the second queue, where 0 means that the |
| | | | [2] | gate is closed, and 1 means that the gate is open |

| hardware address | | name location description | |
|---|---|---|---|
| Module ID | module address | | |
| | | [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-9.

Table B-9 RAM Format for 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

#### C.1.1. Configuration Control Register/Gate List Message Format

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

By specifying the base address of the configuration, the number of configurations and the content of the configuration in the configuration packet,

To realize the configuration of a register/entry or multiple registers/entry with consecutive addresses.

The format of the configuration packet is shown in Figure C-1. The configuration packet TSMP Ethernet header and NMAC

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

The reason for the semantics of TSNtag is that the hardware is encapsulating NMAC packets and PTP packets.

When installed in the TSMP protocol, the SMAC in the TSMP Ethernet header of the configuration message can be directly

As the DMAC in the TSMP Ethernet header encapsulated in the message, and on constructing the NMAC

When sending a packet, you 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 the NMAC protocol, and its Ethernet type is 0x1662.

30B in the configuration packet indicates the number of configuration data carried in the packet, 32B~35B

Indicates the base address of all configuration data of the message, and reserves 32 bits of memory for each configuration data.

Loosen up. When the number of configuration data is N (decimal), the first configuration register/

The address of the entry is the base address + 0, 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).

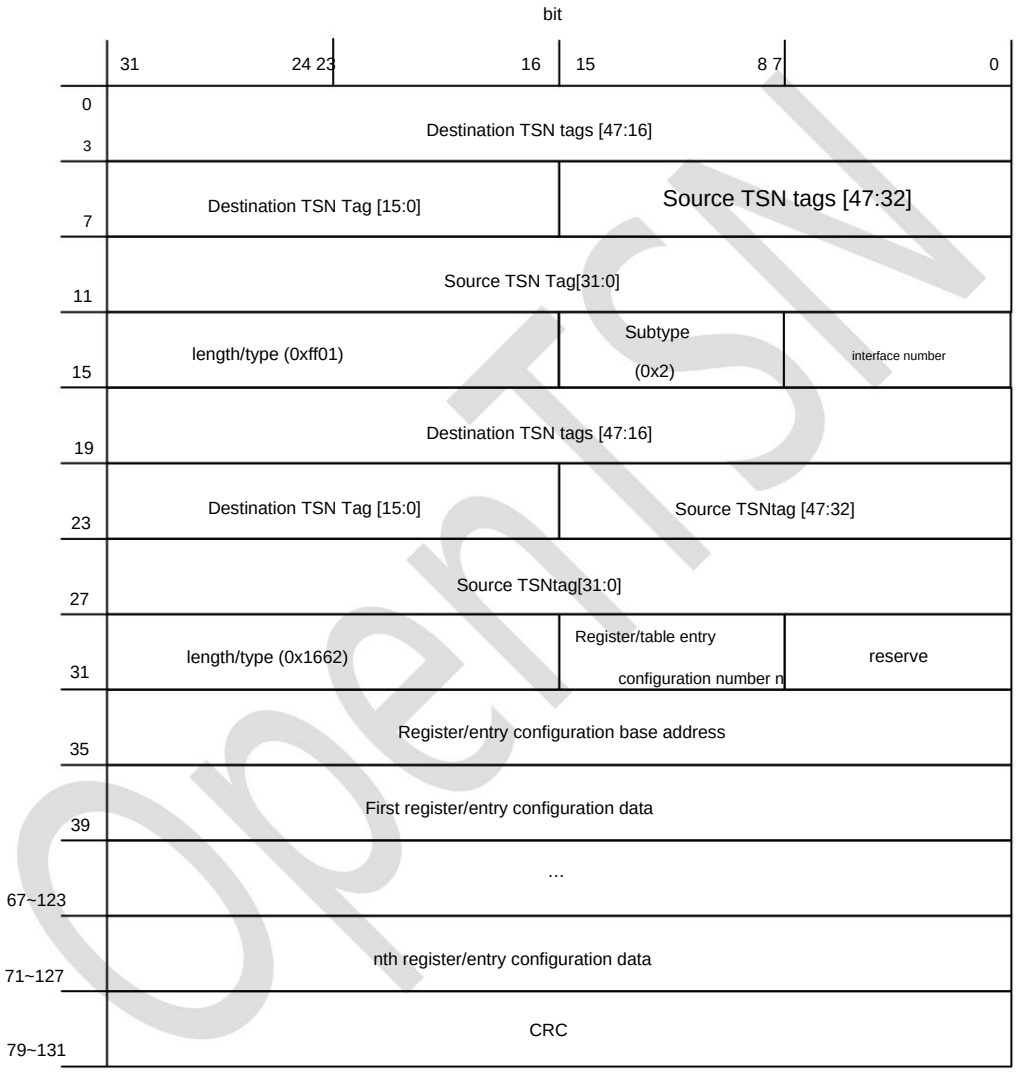


Figure C-1 Configuration message format

C.1.2. Configure the message format of the mapping table/inverse mapping table

The configuration contents of the mapping table and inverse mapping table are carried by standard Ethernet packets.

See Figure C-2 for the format of the packet for configuring the mapping table and the inverse mapping table.

The Ethernet type is 0xff01, the subtype is 0x2, and each entry in the mapping table and inverse mapping table is configured

Need to include 3 parts of information (see Table C-1 for details): configuration valid bits, configuration address, configuration table entry. Reserve a bit width of 224 bits for each mapping table entry, and reserve 224 bits for each inverse mapping table entry. Leave a bit width of 96bit.

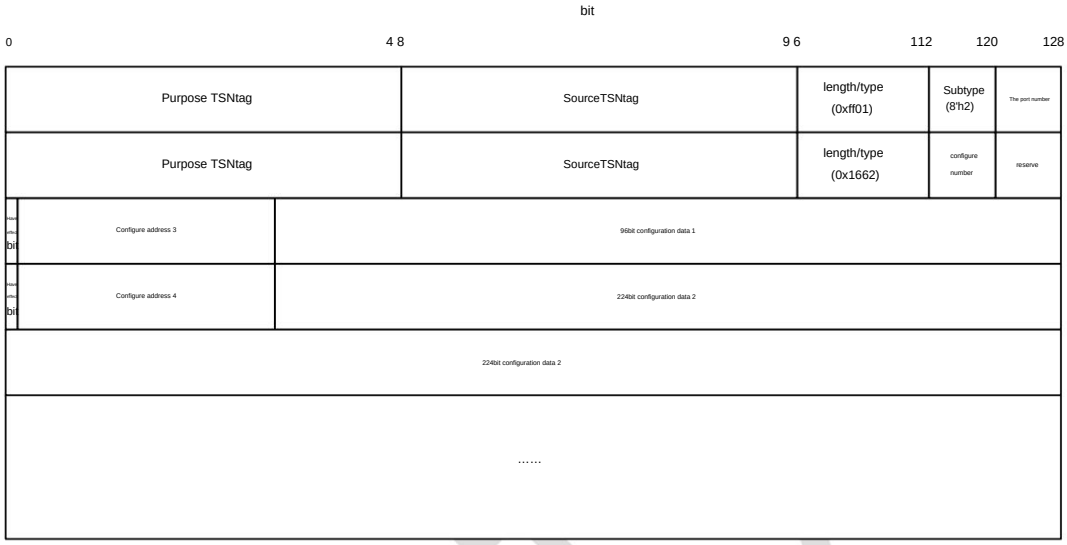


Figure C-2 Packet format of configuration mapping table and inverse mapping table

Table C-1 Description of the information contained in each entry in the configuration

| field | bit width (bit) | describe |
|---|---|---|
| Configured valid bits Valid | 1 | 1 means the register/entry configuration is valid; 0 means the register/entry configuration is invalid effect; |
| configured address addr | 31 | High 7bit (addr[30:24]) indicates the hardware module where the register/entry is located No. MID, supports up to 128 hardware modules; The lower 24bit (addr[23:0]) indicates that the register/entry is used in the hardware module assigned address. |
| configured entry/ | | For the mapping table (each item is 152bit), this field is 224bit; For the reverse mapping table (57 bits for each entry), this field is 96 bits. |

## 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).

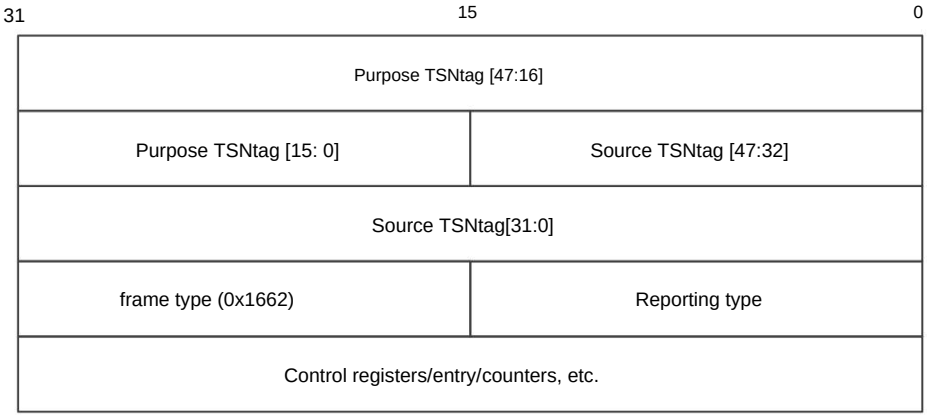


Figure C-3 NMAC report packet format

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'd2 | n (decimal) | 32n to 32n+31 injection schedule, 0ÿnÿ31 (decimal system), and n is an integer. (not yet available) |
| 6'd3 | n (decimal) | Submission timetables for Articles 32n to 32n+31, 0ÿnÿ31 (decimal system), and n is an integer. (not yet available) |
| 6'd4 | n (decimal) | Item 32n to Item 32n+31 gating table, 0ÿnÿ31 (decimal), and n is an integer |
| 6'd12 | 10'd0 report status register | |
| 6'd13 | n (decimal) | Item 2n and Item 2n+1 mapping table, 0ÿnÿ15 (decimal), and n is an integer |
| 6'd14 | n (decimal) | Item 4n to Item 4n+3 mapping table, 0ÿnÿ63 (decimal), and n is an integer |

### C.2.1. Single register reporting message format

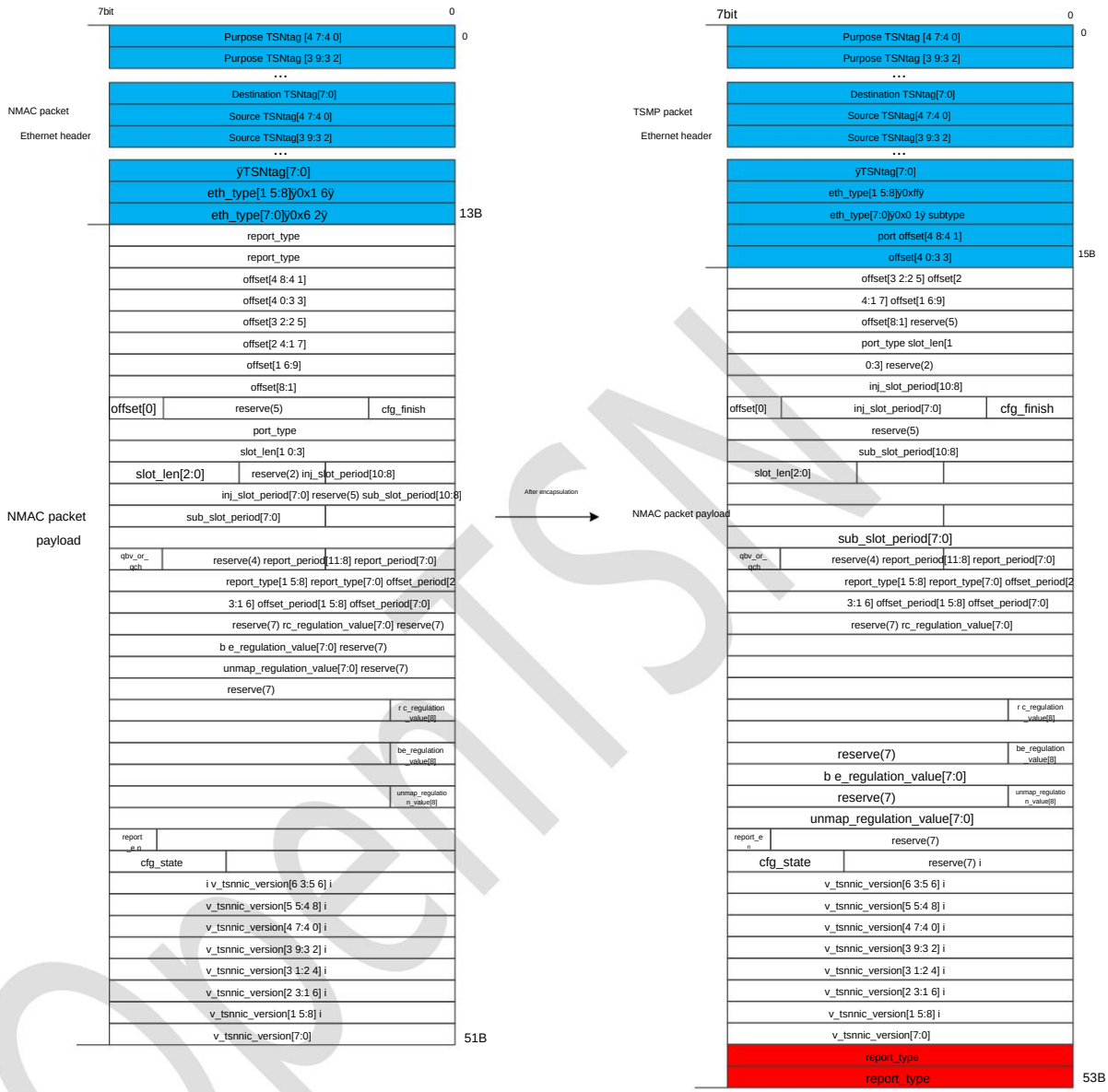


Figure C-4 NMAC packet format for reporting a single register

C.2.2. Format of hardware status report message

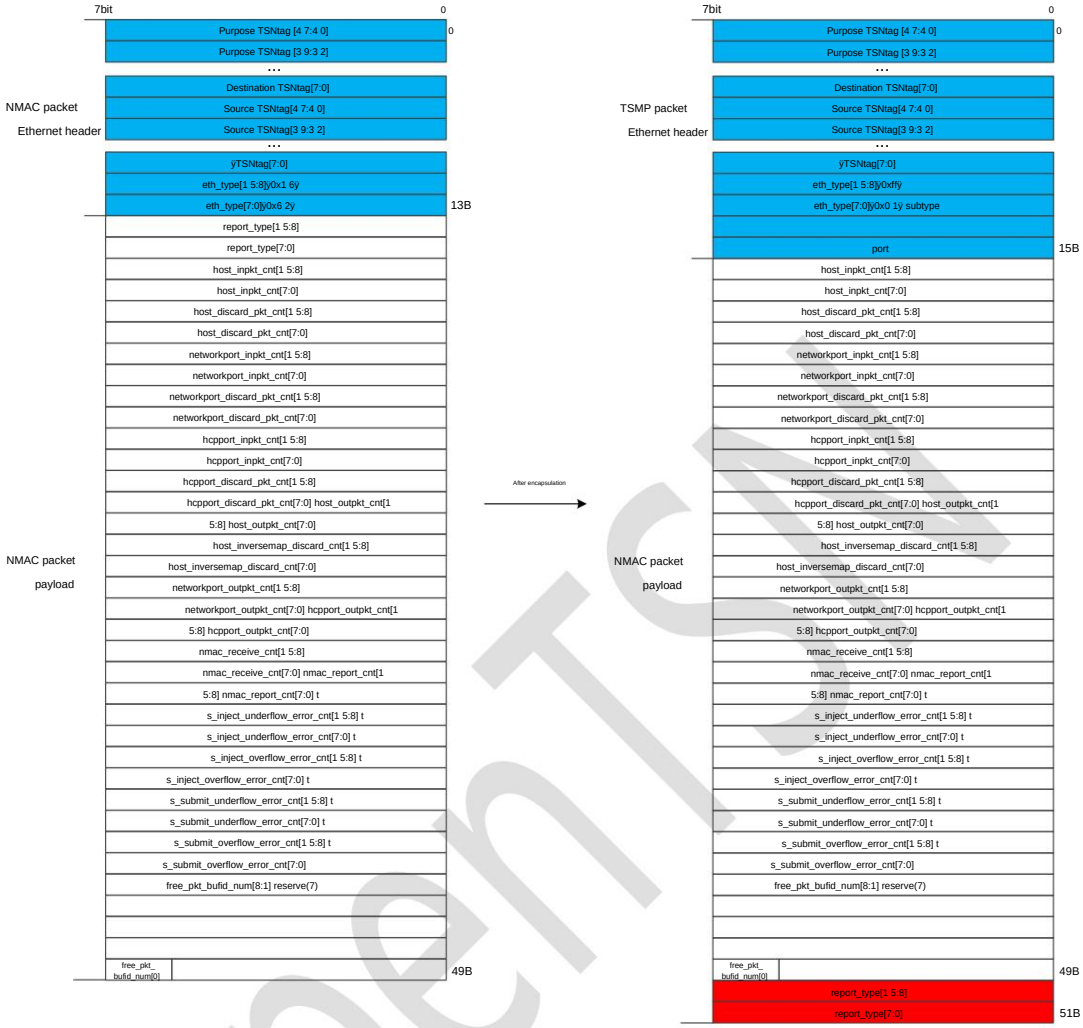


Figure C-5 Format of the hardware status report message

Table C-3 Description of hardware reported counters

| Name Bit Width Meaning 16 The number of packets | | |
|---|---|---|
| 16 The number of packets discarded by the host interface host_discard_pkt_cnt 16 | | received by the host interface host_inpkt_cnt |
| networkport_inpkt_cnt networkport_discard_pkt_cnt 16 The number of packets discarded by the | | The number of packets received by the network interface |
| received by the control interface hcpport_inpkt_cnt 16 Counter for the number of packets discarded | | network interface 16 Counter for the number of packets |
| by the control interface after receiving hcpport_discard_pkt_cnt 16 Counter for the number of | | |
| counter 16 Counter of the number of messages sent by the control interface | | packets sent by the host interface Number of messages |
| | | |
| | | |
| host_inversemap_discard_cnt 16 | | |
| networkport_outpkt_cnt | | |
| hcpport_outpkt_cnt | | |

| | | |
|---|---|---|
| nmac_receive_cnt | 16 | Counter of the number of NMAC messages received by the CSM module |
| nmac_report_cnt | 16 | Counter for the number of NMAC packets reported by the CSM module |
| ts_inj_underflow_error_cnt | 16 | ST message injection underflow error counter (no time-sensitive message injection control function) |
| ts_inj_overflow_error_cnt | 16 | ST message injection overflow error counter (no time-sensitive message injection control function) |
| ts_sub_underflow_error_cnt 16 | | ST message submission underflow error counter (there is no time-sensitive message submission control function) |
| ts_sub_overflow_error_cnt | 16 | ST message submission overflow error counter (there is no time-sensitive message submission control function) 35 |
| total | / | |

## C.2.3. Gated list entry report message format

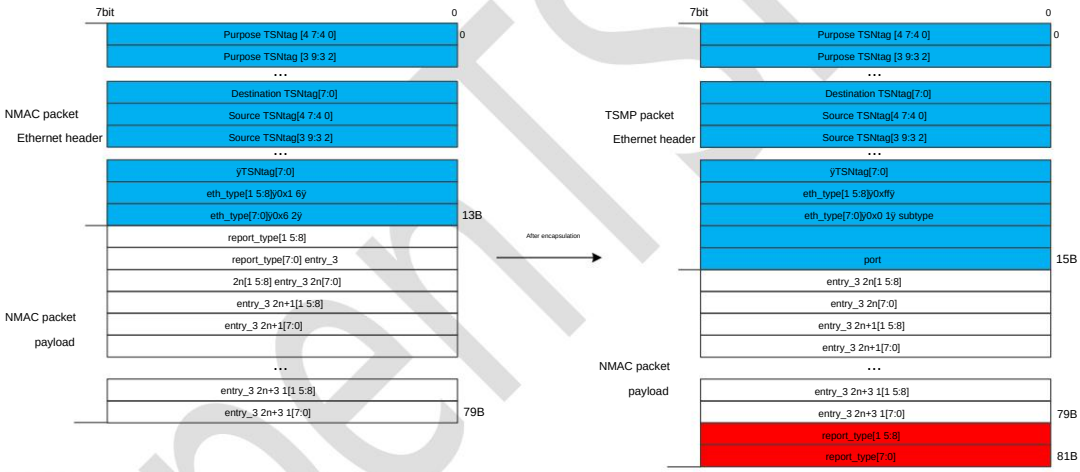


Figure C-6 Format of the report of the gating list entry

### C.2.4. Mapping table entry report message format

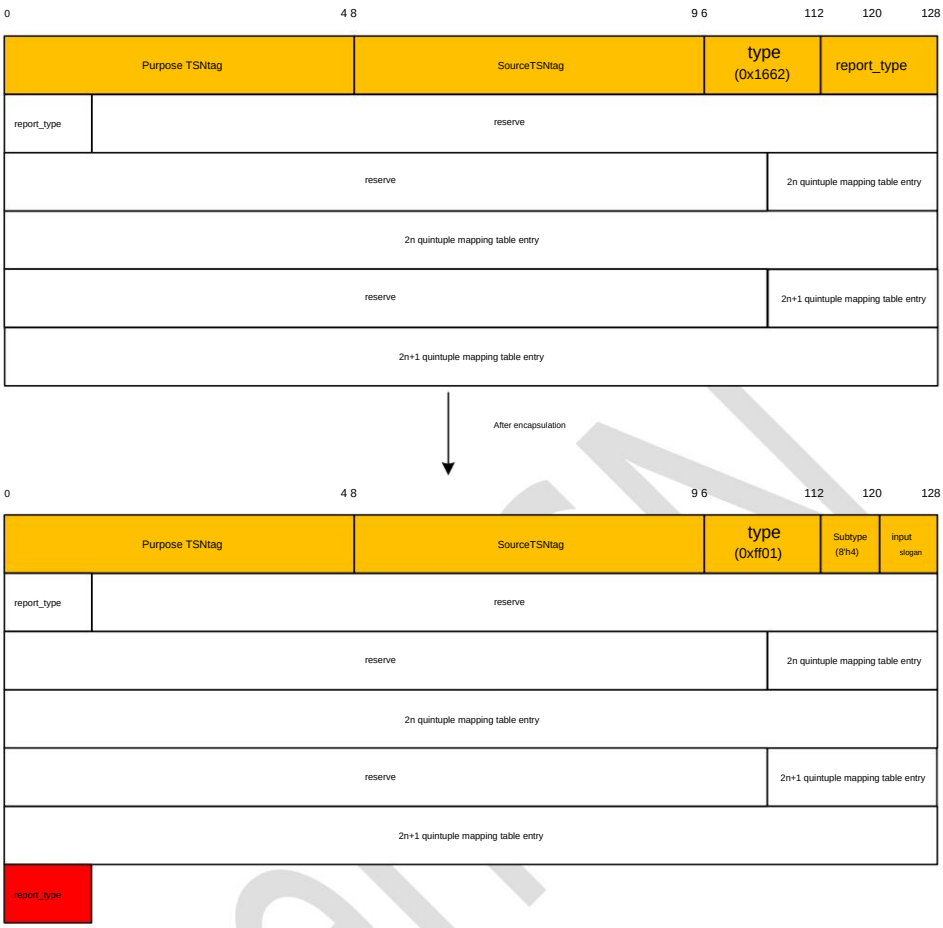


Figure C-7 Format of the message reported by the mapping table entry

C.2.5. Inverse mapping table entry report message format

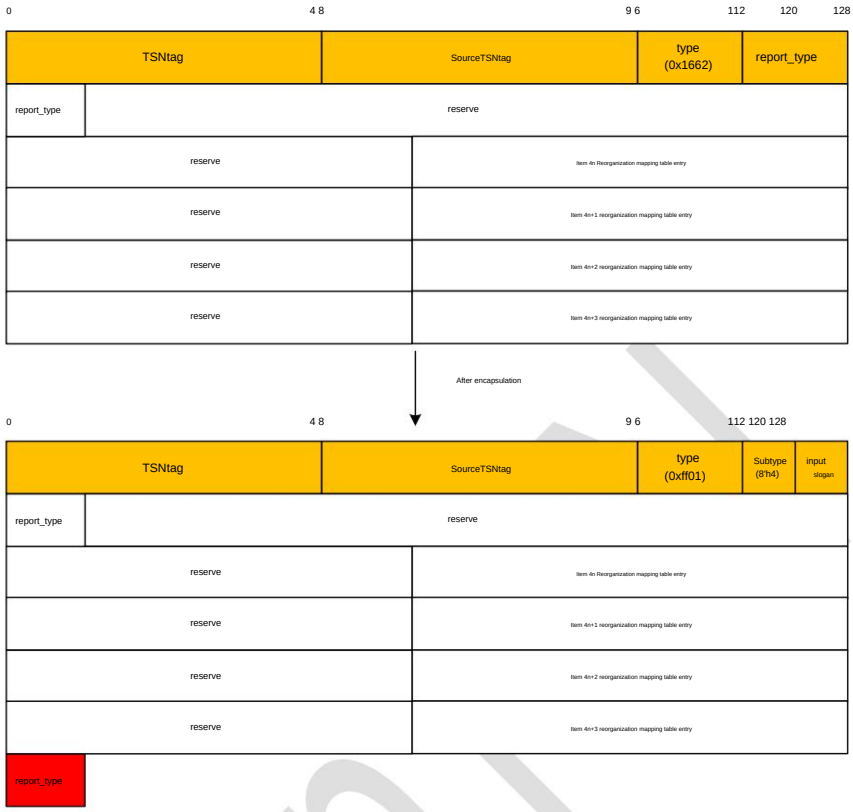


Figure C-8 Inverse mapping table entry report message format

## Appendix D. wr_command/rd_command/rd_command_ack commands

## Format

Table D-1 wr_command/rd_command/rd_command_ack Command Format

| 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+TSN 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;<br>4'b0010: read command of register or table entry; |

| bit width | | name | illustrate |
|---|---|---|---|
| | | | 4'b0110: Read response of register or table entry. |
| [183:152] | 32 | read/write address of addr register or table entry | |
| [151:0] | 152 | data The read/write data of the register or table entry | |

## **Appendix E. PTP** Protocol Format

The format of the Sync, Delay_req and Delay_resq messages is shown in the figure below.

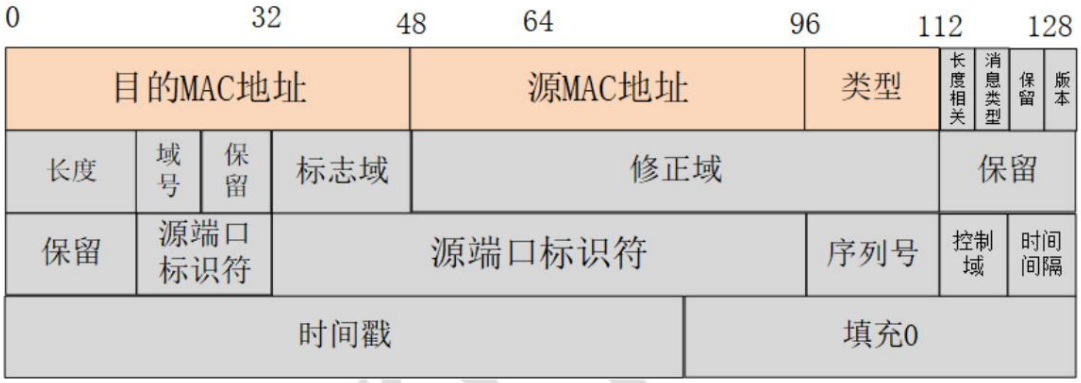


Figure E-1 Sync, Delay_req and Delay_resq message formats

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,

Protocol to configure the TSN chip and HCP and encapsulate 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, and timing of ST streams.

injection, commit on time, output gating, etc.);

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

frame.

## F.2. TSMP Frame Format

The format design of TSMP frame is shown in the following figure.

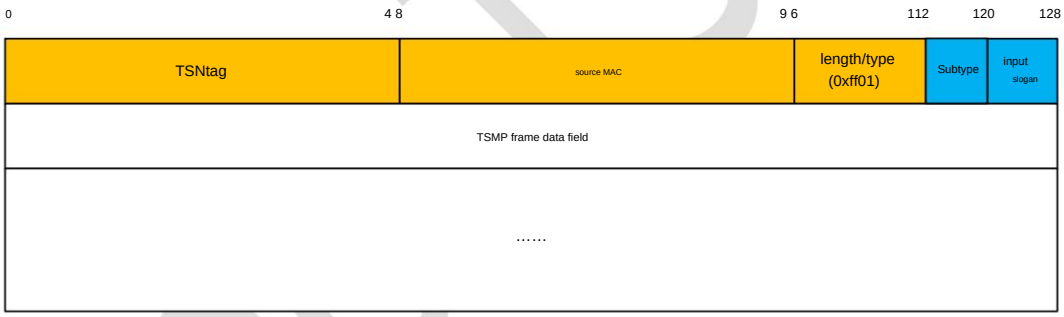


Figure F-1 Format of a TSMP frame

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 the table below for details.

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

| field | Bit width description | |
|---|---|---|
| TSNtag | The result of mapping 2 48 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, there are currently 6 types:<br>ARP Encapsulation Frames, Beacon Frames, Switch/NIC Configuration Frames, HCP |

| | | |
|---|---|---|
| | | Configuration frame, ICMP encapsulation frame, Probe frame. |
| Input port number 8 The frame | sent by the host to the TSN chip enters the port number of the TSN chip Table F-2 TSMP frame | |

type

| frame type | The value of the subtype means | |
|---|---|---|
| ARP encapsulated frame 8'h0 | | that ARP frames are encapsulated into TSMP frames for transmission in the network, and the ARP frames are completely stored in the TSMP data domain |
| Switch/NIC status reporting Beacon frame | 8'h1 | switch and the status frame reported by the network card to the controller, and the status report frame of the switch and network card is completely stored. In the frame that the TSMP data domain controller configures the switch and network |
| Switch/NIC configuration frame | 8'h2 | card, the controller encapsulates the NMAC configuration frame into the TSMP frame, and the NMAC configuration frame is completely stored in the frame where the TSMP data domain controller configures the HCP; the configuration information |
| HCP configuration frame 8'h3 | | is stored in the TSMP data field. |
| HCP status report Beacon frame | 8'h4 | The status information reported by HCP is stored in the TSMP data field |
| PTP encapsulated frame 8'h5 | | Encapsulate PTP frames (sync frames, delay_req frames, delay_resp frames) into TSMP frames, where the PTP frames are completely stored in the TSMP data field |