开源枫林 TSN 架构设计
（V4.0）

OpenTSN

OpenTSN 开源项目组

2020 年 11 月

版本历史

版本	修订时间	修订内容	修订人	文件标识
1.0	2020.10.20			OpenTSN2.0
1.1	2020.10.23	附录 C、附录 D，主要是配置与上报相关寄存器的说明		
2.0	2020.10.24	增加总体设计中所有表项的抽象与介绍		
3.0	2020.10.27	修改标题，以及后续文档架构		
4.0	2020.11.25	1. 针对各个模块的信号定义、功能、内部结构等设计，与代码比对后进行统一更改； 2. Port_type 配置类型的修改：0 代表合作类型，处理 TSN 报文，1 代表非合作类型，处理标准以太网类型； 3. 配置报文类型“0662”更正为“1662”		

OpenTSN

目录

OpenTSN

OpenTSN

1. 项目概述

本文档是介绍 OpenTSN 开源项目中的开源枫林 TSN（时间敏感网络）架构设计，主要分为项目概述、总体设计以及详细设计三大部分。

1.1 设计目标

TSN 对传统以太网在时间同步、延迟确定性、可靠性传输和管理控制等方面进行增强，其应用场景已经由最初的工业互联网扩充至运营商网络、车载网络和航空航天器网络等。随着不同领域网络应用的丰富和扩展，应用场景已呈现多样化和差异化的特点。为了满足上述场景多样化与差异化的应用需求，OpenTSN 开源项目推出了开源枫林 TSN 架构。开源枫林 TSN 架构通过提取 TSN 标准中合适的子集进行设计，旨在设计一套能够满足不同领域对 TSN 网络的多样化及差异化需求的架构。

1.2 设计指标

■支持 IEEE 802.1AS、802.1Qch、802.1Qbv、802.1Qcc 标准

■同时支持时间敏感、带宽预约和尽力转发三种流量的转发交换

■根据注入时刻表对时间敏感报文进行缓存管理以及延迟注入

■根据提交时刻表对时间敏感报文进行缓存管理以及延迟提交

■支持 9 个千兆以太网接口（8 个网络接口，1 个主机接口）

■硬件调度时间槽设置范围为[4us，512us]

■每个架构支持 32 条时间敏感流

OpenTSN

■ST 流量最大支持 1024 个硬件调度时间槽的延迟

■交换容量 16Gbps

■交换延迟小于 30us

■支持 16K 条流的转发配置

■集中式报文交换缓存管理

2. 总体设计

2.1 总体架构

开源枫林 TSN 架构总体架构如图 2-1 所示。

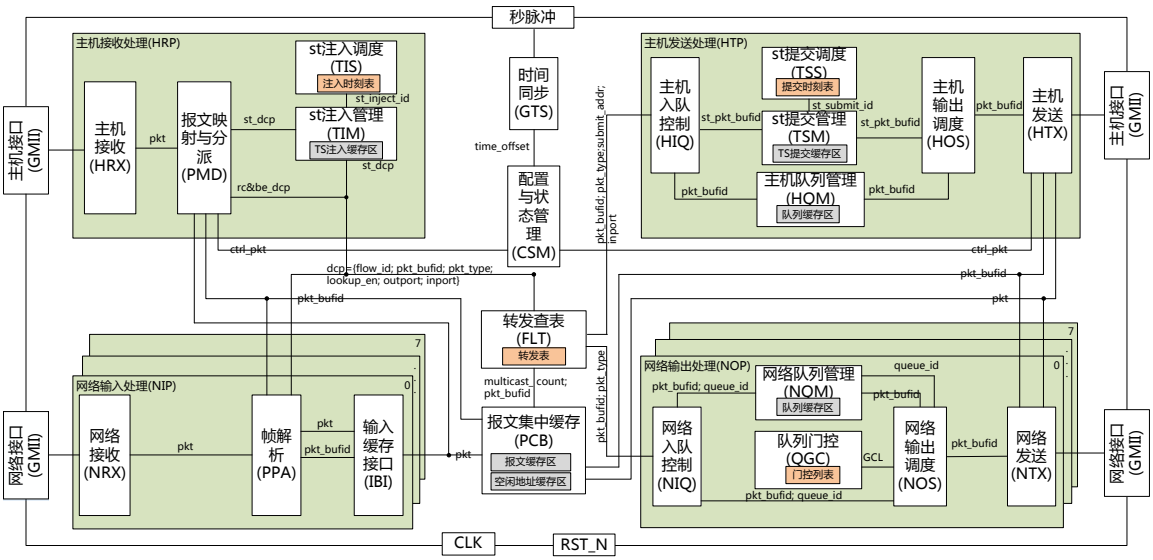


图2-1　开源枫林 TSN 架构顶层架构图

顶层架构图中的信号格式定义如下表 2-1。

表 2-1 开源枫林 TSN 架构顶层信号定义

信号	位宽	含义
pkt（报文集中缓存接收或发送的信号）	134	报文体数据，具体格式参考附录 A

pkt（非报文集中缓存模块的信号）	9	报文体数据，具体格式参考附录 A
ctrl_pkt	9	控制报文体数据
dcp(descriptor)	46	报文描述符数据，用于标识报文数据
st_dcp	36	主机下发的 ST 报文描述符数据
rc&be_dcp	46	主机下发的 RC 和 BE 报文描述符数据
pkt_bufid	9	报文数据在报文缓存区中缓存的 ID 号
st_pkt_bufid	9	ST 报文数据在报文缓存区中缓存的 ID 号
pkt_type	3	报文类型数据
flow_id	14	报文数据的流 ID，用于标识流
lookup_en	1	查表使能
outport	9	输出端口号（bitmap）
inport	4	输入端口号
queue_id	3	queue_id 号，用于标识队列缓存的 ID 号
multicast_ count	4	组播报文的输出端口数量
st_inject_id	5	st 描述符在注入调度时存放的 ID 号
st_submit_id	5	ST 的 pkt_bufid 在提交调度时存放的 ID 号
GCL(gate_ctrl_vector)	8	门控信号
time_offset	49	1588 同步从时钟架构需要补偿的值

现将整个架构划分为五部分：主机接收处理逻辑、主机发送处理逻辑、网络输入处理逻辑、网络输出处理逻辑、以及内部处理逻辑。下面对此五部分逐一进行介绍。

OpenTSN

2.1.1 主机接收处理逻辑

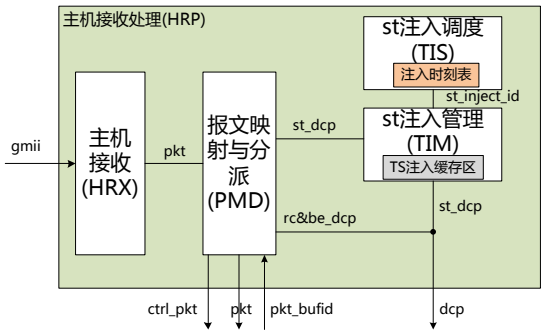


图2-2　主机接收处理(HRP)内部逻辑图

HRX(Host RX，主机接收)模块：主要功能是接收主机接口传来的报文，将报文传输时钟域由从 GMII 接收时钟域切换到架构内部时钟域，以及记录时间同步报文的时间戳；然后根据 TSN 架构所处的阶段来判断是否接收并处理报文；接下来根据接收到的报文前 8B 中的低 45bit 是否为全 0 来判断该报文是否为杂包，将杂包进行丢弃。主机接口接收的映射报文都携带了 metadata（具体格式参考附录 A.2），metadata 由主机中的软件进行添加。

PMD（Pkt Map and Dispatch，报文映射与分派）模块：主要功能是对 BE 流和 RC 流进行流量监管，将报文转换位宽后写入集中缓存区中，同时构造报文描述符（报文描述符的格式见下表 2），以及根据报文类型将报文描述符分派到不同的模块。流量监管，是根据 bufid 剩余数量与报文阈值大小关系来判断是否传输该报文：当 bufid 剩余的数量少于 RC 报文阈值时，将 RC 报文和 BE 报文进行丢弃；当 bufid 剩余的数量少于 BE 报文阈值时，将 BE 报文进行丢弃。报文写入集中缓存区，是用两个 ping-pong 寄存器来依次缓存报文，将

报文位宽由 8bit 转换为 128bit，每个寄存器写满后便将数据写入集中缓存区中。报文描述符的构造，是根据报文集中缓存模块分配给本接口的 pkt_bufid 以及报文映射后的信息(TSNTag，参考附录 A)构造一个能够标识报文的描述符数据。报文分派，需要先区分报文是 ST 报文、RC 报文和 RC 报文还是 NMAC 报文（NMAC 报文格式定义参考附录 D），再将不同类型的报文描述符分派到不同的目的模块。另外，本模块还需将主机下发报文的 metadata 丢弃。

表2-2 描述符格式

内容	位宽	位置	含义
inject_addr/ submit_addr	5	[45:41]	ST 流注入/提交时缓存的地址。
reserve	1	[40]	保留
inport	4	[39:36]	报文的输入端口，用于最终构造 metadata。
pkt_type	3	[35:33]	报文类型，用于入队控制时区分报文类型，选择队列。
flow ID/IMAC	14	[32:19]	流 ID，用于 FLT 模块查表时的地址索引。
lookup_en	1	[18]	查表使能，用于 FLT 判断该报文是否需要进行查表操作。
outport	9	[17:9]	输出端口号，用于 FLT 模块区分输出端口。
pkt_bufid	9	[8:0]	报文在缓存区中缓存的 ID 号，用于标识每个报文。

TIM（TS Inject manage，TS 注入管理）模块：主要功能是将接收到的 ST 报文描述符写入到 ST 描述符缓存区（缓存区的数据格式见下表 3）中进行缓存，其中缓存地址为注入地址（报文映射后得到，

携带在 TSNTag 中），并等待 ST 注入调度模块的调度信号进行 ST 报文描述符的读取和发送。

表2-3 ST 注入缓存区数据格式

名称	含义	备注
descriptor[40:0]	用于标识 ST 流的描述符，所包含的内容如上“Descriptor 数据格式”的低 41 位。	使用 RAM 进行实现，深度为 32。

TIS（TS Inject Schedule，TS 注入调度）模块：主要功能是用当前时间槽去查找注入时刻表（数据格式见下表 4），判断当前时间槽是否有 ST 报文描述符需要调度，并将需要调度的 ST 报文描述符传给 ST 注入管理模块。

表2-4 注入时刻表格式

名称	位置	含义	备注
Value[15:0]	[15]	表项有效位	使用 RAM 进行实现，深度为 1024。
	[14:5]	发送时间槽	
	[4:0]	TSNTag 中注入地址“inject addr”	

2.1.2 主机发送处理逻辑

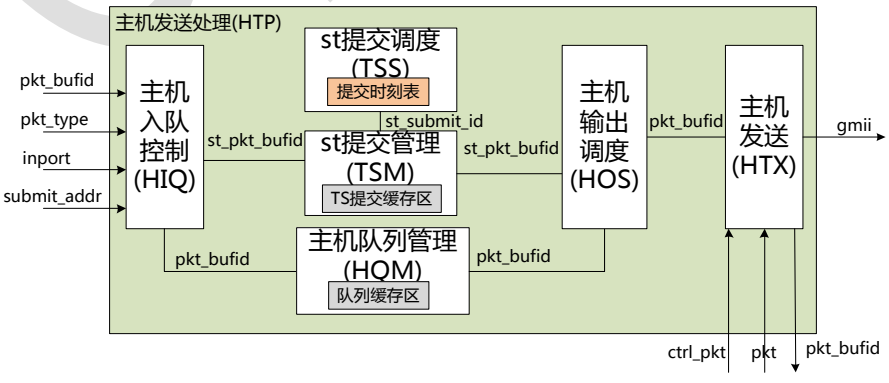


图3-1 主机发送处理(HTP)内部逻辑图

HIQ（Host Input Queue，主机入队控制）模块：主要功能是根据 pkt_type 来识别报文类型并将报文描述符传给对应的目的模块。将 ST 报文描述符发送给 ST 提交管理模块等待调度；将非 ST 报文描述符发送给主机队列管理模块等待调度。

TSS（TS Submit Manage，TS 提交管理）模块：主要功能是将接收到的 ST 报文描述符写入到 ST 报文描述符缓存区（缓存区的数据格式见下表 5）中，其中缓存地址为提交地址（报文映射后得到，携带在 TSNTag 中），并等待 ST 提交调度模块的调度信号进行 ST 报文描述符的读取和发送。

表2-5 ST 提交缓存区数据格式

名称	含义	备注
descriptor[12:0]	用于标识 ST 流的 pkt_bufid 以及输入端口号。	使用 RAM 进行实现，深度为 32。

TSM（TS Submit Schedule，TS 提交调度）模块：主要功能是用当前时间槽去查找提交时刻表（内容如下表 6），判断当前时间槽是否有 ST 报文描述符需要调度，并将需要调度的 ST 报文描述符传给 ST 提交管理模块。

表2-6 提交时刻表格式

名称	位置	含义	备注
Value[15:0]	[15]	表项有效位	使用 RAM 进行实现，深度为 1024。
	[14:5]	提交时间槽	
	[4:0]	TSNTag 中的提交地址“submit addr”	

HQM （Host Queue Manage，主机队列管理）模块：主要功能

是将从主机接口输出的非 ST 报文描述符写入队列（队列数据格式见下表 7）中进行缓存管理，等待主机输出调度模块的调度信号进行 pkt_bufid 的调度输出。

表2-7 非 ST 报文描述符缓存队列的数据格式

名称	含义	备注
pkt_bufid[8:0]	用于标识非 ST 流的 pkt_bufid。	使用 FIFO 进行实现，深度为 256。

HOS（Host Output Schedule，主机输出调度）模块：主要功能是按照优先级调度策略对 ST 报文描述符和非 ST 报文描述符进行调度。当 ST 提交管理模块有 ST 报文描述符传输请求时，先将此描述符调度传输给主机发送模块，当 ST 提交管理模块没有 ST 报文描述符传输请求且主机队列管理模块中队列非空时，才能调度传输主机队列管理模块中的非 ST 报文描述符。

HTX(Host TX)主机发送模块：主要功能是从报文集中缓存区中读取报文并释放 pkt_bufid，将报文从主机接口输出。读取报文时，需要先将 pkt_bufid 映射成报文读取的基地址，并根据此基地址往报文集中缓存模块读取报文，将报文每拍数据位宽由 128bit 转换为 8bit，并添加帧前导符、帧开始符以及 8B 的 metadata；在接收到报文最后一拍数据时，将此 pkt_bufid 释放给报文集中缓存模块以便后续进入架构的报文使用。当 CSM 模块有状态上报请求时，在构造帧前导符、帧开始符以及 8B 的 metadata 后直接将 CSM 模块传来的上报报文从主机接口输出。

2.1.3 网络输入处理逻辑

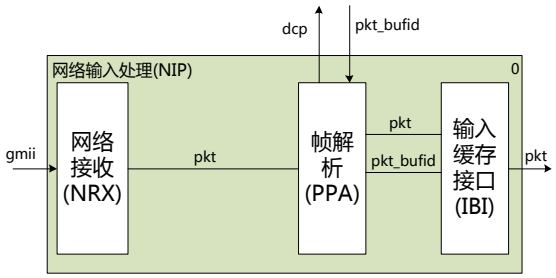


图2-3　网络输入处理(NIP)内部逻辑图

NRX（Network RX，网络接收）模块：主要功能是接收网络接口传来的报文，将报文传输时钟域从 GMII 接收时钟域切换到架构内部时钟域，以及记录架构接收时间同步报文的时间信息，并存放在 TSNTag 中，并根据架构所处的阶段来判断是否接收并处理报文。

FPA(Frame PArse)帧解析模块：主要功能是从 TSNtag 中提取信息用来构造报文描述符，并根据接口类型来判断此报文是否经过映射，若报文未经映射，则在其报文描述符中指定其输出接口为主机接口；将每拍数据位宽由 8bit 转换成 128bit 后输出给输入缓存接口模块。

IBI（Input Bufm_memory Interface，输入缓存接口）模块：主要功能是将报文发送给报文集中缓存模块进行缓存。本模块用两个 ping-pong 寄存器来依次缓存数据，每个寄存器写满后就将其发送给报文集中缓存模块。

OpenTSN

2.1.4 网络输出处理逻辑

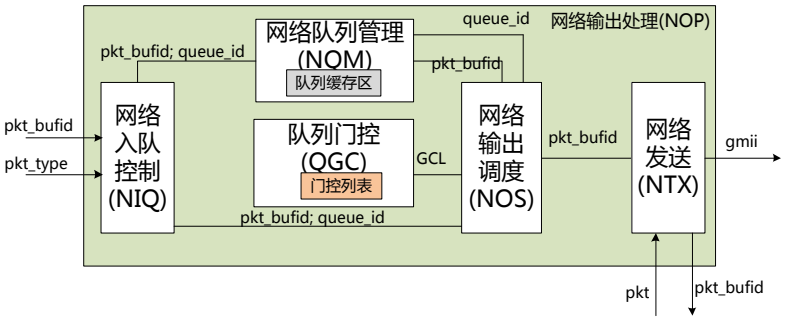


图2-4　网络输出处理(NOP)内部逻辑图

NIQ（Network Input Queue，网络入队控制）模块：主要功能是将 pkt_bufid 写入到网络队列管理模块中进行缓存。本模块需要根据接收到的报文类型信息、队列门控模块传来的门控信息进行 queue_id 的映射，并将 pkt_bufid 与 queue_id 发送给网络队列管理模块进行缓存，同时将 pkt_bufid 与 queue_id 发送给网络输出调度模块，以便对队列首地址进行管理；本模块还需要根据写入队列的信息与网络输出调度模块传输的调度队列的信号来对队列的状态进行管理，主要是对所有队列中目前所写入的 bufid 数量进行管理。

QGC（Queue Gate Control，队列门控）模块：主要功能是根据架构的全局时钟进行门控列表的读取，并将门控列表（门控列表的数据格式如下表 8）中 8 个队列的门控开关信息发送给网络入队控制模块和网络输出调度模块。

表2-8　门控列表数据格式

名称	含义	备注
gate_ctrl_vector[7:0]	门控向量，对应 8 个门控信息。	使用 RAM 进行实现，深度为 1024

NOS（Network Output Schedule，网络输出调度）模块：主要功能是根据队列优先级和门控状态对 8 个队列进行调度并根据调度结果从网络队列管理模块提取出 pkt_bufid。本模块需要根据当前队列信息与队列门控模块传来的门控信息进行逻辑处理，得出一个最优先调度的队列，并从网络队列管理模块的对应队列中读取 pkt_bufid。

NTX（Network TX，网络发送）模块：主要功能是从报文缓存区中读取报文并释放 pkt_bufid，计算时间同步报文的透明时钟并更新透明时钟域，将报文传输时钟域从架构内部处理时钟域切换到 GMII 发送时钟域，将报文添加帧前导符和帧开始符后从网络接口输出。读取报文时，需要先将 pkt_bufid 映射成报文在集中缓存区中的基地址，并根据此基地址往报文集中缓存模块读出报文，在接收到报文最后一拍数据时，将此 pkt_bufid 归还给报文集中缓存模块以便后续进入架构的报文使用。

2.1.5 内部处理逻辑

FLT（Forward Lookup Table，转发查表）模块：主要功能是根据接收到的报文描述符提取出 flow_id，根据 flow_id 查找转发表（转发表的格式见下表 9），根据查表得到的输出端口进行 pkt_bufid 和 pkt_type 的转发。当描述符中 lookup_en 为低（代表本描述符不需要查表）时，直接将描述符中的 outport 作为描述符输出端口号；当查表结果中输出端口多位为高时，意味着描述符对应的报文是个多播报文，需要将 pkt_bufid 和 pkt_type 往多个端口进行转发，同时将

pkt_bufid 与输出端口的数量发送给报文集中缓存模块进行组播计数。

表2-9 转发表数据格式

名称	含义	备注
outport[8:0]	输出端口号(bitmap)，总共 8 个网络接口与 1 个主机接口。	使用 RAM 进行实现，深度为 16K

PCB（Pkt Centralize Buffer，报文集中缓存）模块：主要功能是对架构需要转发的所有报文进行集中缓存（报文缓存区的数据格式见下表 10），对集中缓存区的空闲地址进行缓存管理（空闲地址缓存区的数据格式见下表 11）。对所有 pkt_bufid 对输出端口数量进行计数，pkt_bufid 每释放一次便将该计数值减 1，当 pkt_bufid 使用完进行释放的时候需要检测该 pkt_bufid 的计数值，只有当计数值为 0 才能将 pkt_bufid 写入到空闲地址缓存区中，当不为 0 时意味着该 pkt_bufid 对应一个组播报文并且该报文还未从所有需要输出的端口输出。

报文缓存区将 2KB 的空间划分成 512 个报文缓存块，每个报文缓存块能缓存一个 2KB 的报文。

表2-10 报文缓存区数据格式

地址[8:0]	内容[133:0]
0-127	第 1 个报文缓存块
128-255	第 2 个报文缓存块
…	..
65408-65535	第 512 个报文缓存块

表2-11 空闲地址缓存区数据格式

名称	含义	备注
pkt_bufid[8:0]	当前“报文缓存区”中空闲的	使用 RAM 进行实现，深度

	报文缓存块 ID 号。	为 512。

GTS（Global Time Synchromzation，时间同步）模块：主要功能是维护一个本地时钟，并根据控制器计算的 offset 值进行时钟的修正。

CSM（Configuration and State Manage，配置与状态管理）模块：主要功能是接收 NMAC 配置报文，解析 NMAC 配置报文并构造命令来对各个模块的可写寄存器和表进行配置；并且会周期性构造 NMAC 报文上报本地状态。

2.2 主要处理流程

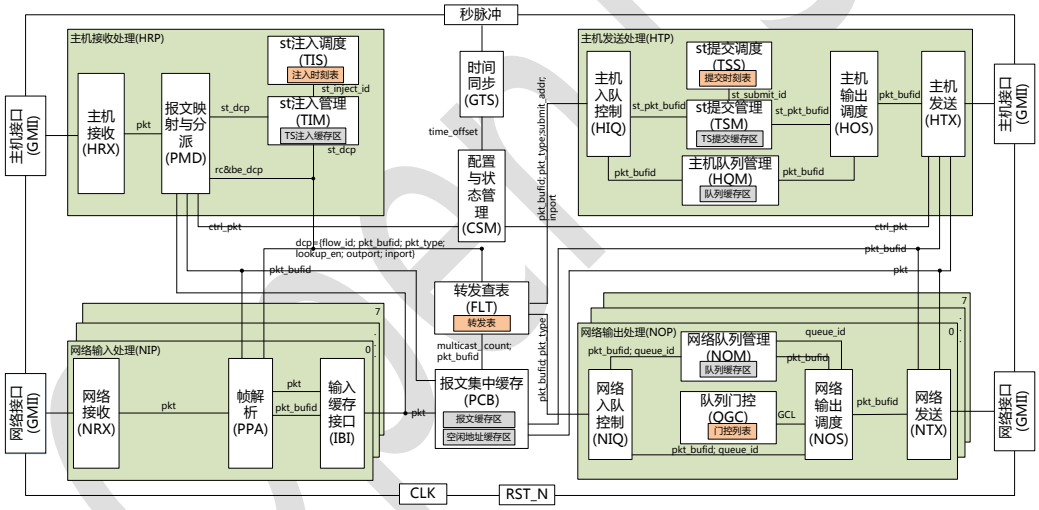


图2-5　开源枫林 TSN 架构顶层架构图

架构内部对报文的处理可分为两部分，一部分是将报文进行集中缓存；另一部分是构造报文描述符，根据描述符进行查表、转发、802.1Qbv/802.1Qch 调度、st 的注入调度、st 的提交调度等一系列的操作。

■报文缓存部分处理流程：

（1）主机接收/网络输入处理模块从主机/网络接口接收到报文，经过各自输入处理之后将报文位宽转换成每拍 128bit 数据，并发送给报文集中缓存模块进行集中缓存；

（2）报文缓存区将整个缓存区划分成 512 个缓存块，每个缓存块能储存 2K 字节数据，并为每个缓存块分配一个 ID 号进行标识。同时，使用另外一块缓存区对所有缓存块的 ID 号进行统一管理，在初始化的时候，需要将 512 个缓存块的 ID 号写入到空闲地址缓存区，并预先依次为每个输入端口取出 ID 号供后续进入的报文使用；

（3）主机发送/网络输出处理模块根据各自的调度策略，将报文从报文集中缓存模块读取出来，并由主机/网络接口发送到主机/网络。

■报文描述符部分处理流程：

（1）主机接收/网络输入处理模块根据从主机/网络接口接收到报文信息，以及报文集中缓存模块给的缓存块 ID 号（pkt_bufid），构造报文描述符并发送给查表转发模块；

（2）查表转发模块根据报文描述符中的 flow_id 进行查表，得到输出端口号，并将报文描述符中的 pkt_type 和 pkt_bufid 信息发送给对应的输出端口处理逻辑。当得到的输出端口多位为高时，意味处理的描述符对应报文是个多播报文，需要将 pkt_bufid 和 pkt_type 往多个端口进行转发，同时将 pkt_bufid 做为写地址，将输出端口的数量作为写数据写入到报文集中

缓存模块进行组播端口数的缓存；

（3）主机发送/网络输出处理模块接收到 pkt_type 后对报文进行识别，将不同类型的 pkt_bufid 写入到对应的队列进行缓存。各端口的输出处理模块会经过调度，将对应的 pkt_bufid 读取出来并根据此 pkt_bufid 从报文集中缓存模块进行报文数据的读取。

下面对网络接口接收与网络接口发送、主机接口接收与网络接口发送、网络接口接收与主机接口发送这三种传输路径的数据处理流程进行详细介绍。

2.2.1 网络接口接收、网络接口发送

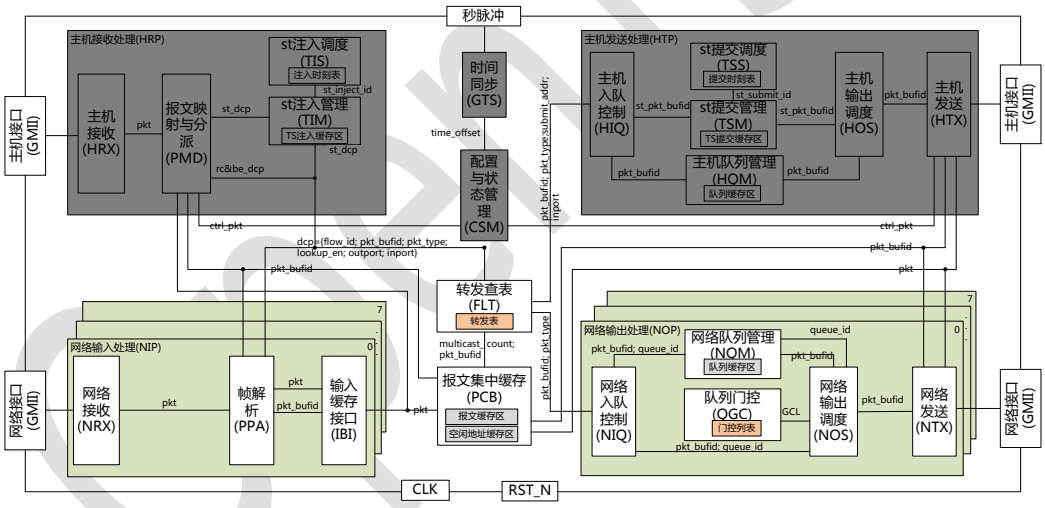


图2-6 网络口接收、网络口发送流程示意图

网络口根据配置的端口类型不一致，可将网络口接收、网络口发送路径上传输的报文分为数据、同步与控制三种报文，下面对这三种报文的处理流程进行介绍。

■数据报文的处理

（1）数据报文由 p0-p7 接口进入架构，经过**网络接收模块**完成

gmii 时钟域到架构内部处理时钟域的转换，并记录报文接收时间戳、将报文往帧解析模块发送；

（2）再由**帧解析模块**对帧进行解析，并提取出 flow_id、pkt_type、lookup_en、outport、submit_addr 信息与**报文集中缓存模块**分配的 pkt_bufid 信息一起构造成报文描述符，将描述符发送给**转发查表模块**进行查表；

（3）报文需要与 pkt_bufid 一起发送给**输入缓存接口模块**，在输入缓存接口模块中将 8bit 的数据写入到 134bit 的寄存器中，，完成报文位宽转换，当一个寄存器写满了 134bit 便将寄存器中的数据写入到报文缓存区；

（4）由于报文缓存区是用分时复用技术进行数据的写入，因此本模块需要等到报文集中缓存模块返回一个确认信号之后才能进行下一次的数据写入。当其中一个寄存器写满，且还未收到报文集中缓存模块返回的确认信号时，后续进行处理的报文就写入到另一个寄存器（双寄存器轮询写入），写满再发送给**报文集中缓存模块**进行报文的缓存，以此轮询，直到报文全部写入完成。

（5）报文描述符数据由**帧解析模块**构造后发送给**转发查表模块**处理，**转发查表模块**会根据报文描述符的 lookup_en 信号判断该报文是否需要查表转发，若不需要查表，则直接将报文描述符中的 outport 信息作为转发的输出端口，将 pkt_bufid 与 pkt_type 信息发送给对应输出端口逻辑；若需要查表，则

从报文描述符中提取出 flow_id 进行查找转发表，得到输出端口号，并将 pkt_bufid 与 pkt_type 信息发送给对应输出端口逻辑；

（6）输出端口逻辑中的**网络入队控制模块**接收到 pkt_bufid 与 pkt_type 后，根据**队列门控模块**的门控信息与 pkt_type 进行缓存队列的选择，并将选择好的 queue_id 与 pkt_bufid 一起发送给**网络队列管理模块**进行缓存。

（7）**网络输出调度模块**根据**队列门控模块**的门控信息以及每个队列的状态信息计算出最优先调度的 queue_id，并将此 queue_id 中的第一个 pkt_bufid 缓存的地址发送给**网络队列管理模块**，等待**网络队列管理模块**将对应的队列中的数据读取出来，将第一个 pkt_bufid 的地址作为 pkt_bufid 发送给**网络发送模块**，并根据读取出来的数据更新此 queue_id 中第一个 pkt_bufid 所缓存的地址。**网络输出调度模块**接收到 pkt_bufid 之后发送给**网络发送模块**，**网络发送模块**根据 pkt_bufid 往**报文集中缓存模块**进行报文的提取。**网络发送模块**内部维持两个寄存器，依次将 134bit 的数据转换成 8bit 的数据往后发送，最后往 gmii 接口输出；当一个寄存器读空之后再往**报文集中缓存模块**进行下一个 134bit 数据的读取，直到报文数据全部读取完成。

■同步报文的处理

同步报文会在网络发送模块进行识别，完成透明时钟的更新，其

他的处理与数据报文的处理一样，因此不再过多赘述。

■控制报文的处理

控制报文与数据报文处理流程一致。

2.2.2 主机接口接收、网络接口发送

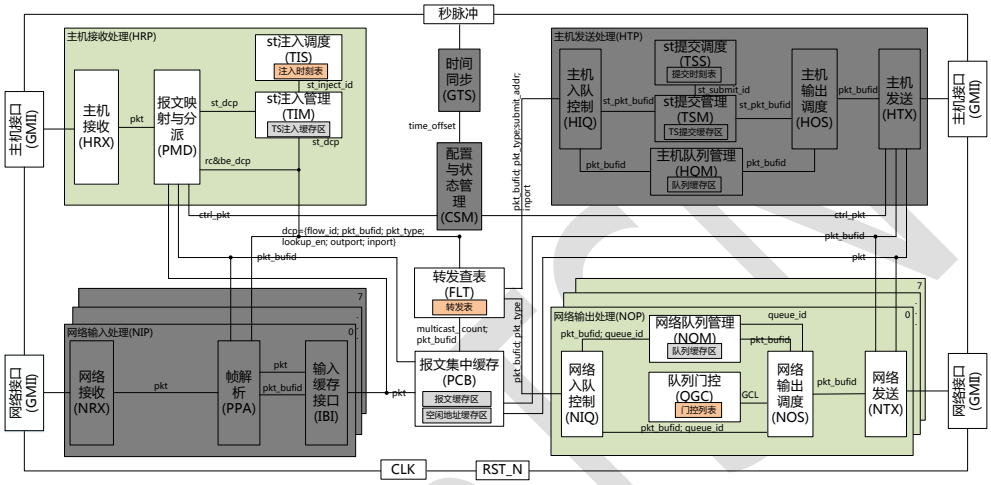


图2-7　主机口接收、网络口发送流程示意图

■数据报文的处理

主机下发的数据报文可分为 ST、BE、RC 三种，由于 BE 与 RC 报文的处理一样，因此可分成 ST 报文的处理、非 ST 报文的处理两种情况进行介绍。

（1）ST 报文的处理

主机下发的 ST 报文，经过**主机接收模块**实现跨时钟域传输后发送给**报文映射与分派模块**；在**报文映射与分派模块**将报文数据与 2.2.1 中的数据处理一样，从报文集中缓存区得到 pkt_bufid，并将 pkt_bufid 转换为基地址，把报文数据存放到报文缓存区。将 pkt_bufid 信息与报文 flow_id、pkt_type、lookup_en、outport、submit_addr 信

息组合构造成报文描述符，发送给 **TS 注入管理模块**。**TS 注入管理模块**将报文描述符进行缓存，并等待 **TS 注入调度模块**经过查找注入时刻表来判断当前时间是否有 ST 流需要被注入。若有，**TS 注入调度模块**则将需要调度信息发送给 **TS 注入管理模块**进行描述符的调度；若没有则不做任何处理。处理完成后发送给**转发查表模块**。之后的处理与 2.2.1 中数据报文的处理一样，不再过多赘述。

（2）非 ST 报文的处理

主机下发的非 ST 报文处理，与 ST 报文的处理大致相同，从报文缓存区得到 pkt_bufid，并将 pkt_bufid 转换为基地址，把报文数据存放到报文缓存区，构造报文描述符。不同的地方是将报文描述符直接**转发查表模块**进行接下来的处理。

■同步报文的处理

主机下发的同步报文，在**主机接收模块**记录接收时间戳，并存放到同步报文 payload 中，之后的处理流程与非 ST 流的处理流程一样，因此不再赘述。

■控制报文的处理

主机下发的控制报文，在**报文映射与分派模块**进行识别并分派给**配置与状态管理模块**；在**配置与状态管理模块**内进行控制报文的解析，并进行寄存器和表的配置。

OpenTSN

2.2.3 网络接口接收、主机接口发送

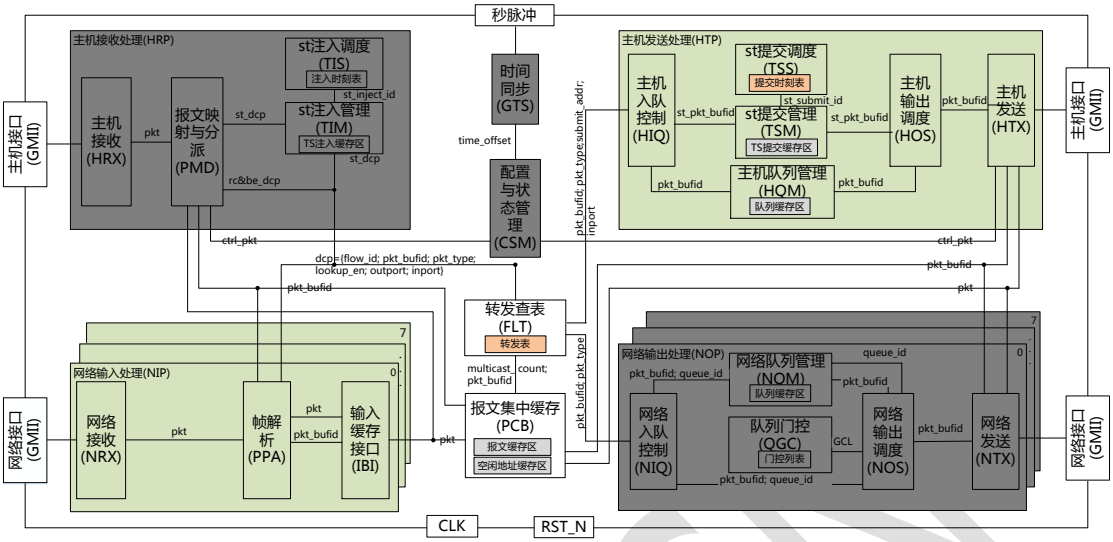


图2-8　网络口接收、主机口发送流程示意图

■数据报文的处理

需要往主机发送的数据报文可分为 ST、BE、RC 三种，由于 BE 与 RC 报文的处理一样，因此可分成 ST 报文的处理、非 ST 报文的处理两种情况进行介绍。

（1）ST 报文的处理

网络接口接收到 ST 报文之后，与 2.2.1 中的数据报文处理流程一样，从报文缓存区得到 pkt_bufid，并将 pkt_bufid 转换为基地址，把报文数据存放到报文缓存区，构造报文描述符并进行转发查表。不同的是在查表之后，得到的输出端口是往主机发送的，因此将 pkt_bufid 与 pkt_type 信息发送给对应的**主机输出端口逻辑**；**主机输出端口逻辑**的**主机入队控制模块**会将 ST 的 pkt_bufid 发送给 **TS 提交管理模块**，**TS 提交管理模块**将 pkt_bufid 进行缓存，并等待 **TS 提交**

调度模块经过查找提交时刻表来判断当前时间是否有 ST 流需要被提交，若有，**TS 提交调度模块**则将需要调度信息发送给 **TS 提交管理模块**进行 pkt_bufid 的调度。**主机输出调度模块**收到调度后的 st 报文的 pkt_bufid 则直接发送给**主机发送模块**；**主机发送模块**根据 pkt_bufid 往**报文集中缓存模块**进行报文的提取。**主机发送模块**内部维持两个寄存器，依次将 134bit 的数据转换成 8bit 的数据往后发送，最后往 gmii 接口输出；当一个寄存器读空之后再往**报文集中缓存模块**进行下一个 134bit 数据的读取，直到报文数据全部读取完成。另外，从报文缓存区读出的报文 metadata 也一样从 gmii 接口发送给主机。

（2）非 ST 报文的处理

需要往主机发送的非 ST 报文的处理，与 ST 报文的处理大致相同。从报文缓存区得到 pkt_bufid，并将 pkt_bufid 作为基地址，把报文数据存放到报文缓存区，构造报文描述符并进行转发查表，查表后将 pkt_bufid 与 pkt_type 信息发送给对应的**主机输出端口逻辑**。不同的是，**主机输出端口逻辑**的**主机入队控制模块**会将 st 的 pkt_bufid 发送给**主机队列管理模块**进行缓存；**主机输出调度模块**需要在没有 ST 报文的提交调度时才能去调度非 ST 报文的 pkt_bufid，从**主机队列管理模块**得到 pkt_bufid 后发送给**主机发送模块**进行处理。

■同步报文的处理

网络接口接收到同步报文后与本节 2.2.3 中的“非 ST 报文的处理”基本一样，唯一的不同是在**主机发送模块**需要将发送时间戳信息写入

到 metadata 中，计算时间同步报文在架构中传输的透明时钟并更新透明时钟域。

■控制报文的处理

往主机接口发送的控制报文都是由**配置与状态管理模块**构造的 NMAC 报文，**配置与状态管理模块**构造的 NMAC 报文直接发送给**主机输出处理模块**，由 gmii 接口发送给主机。

附录 A：数据格式定义

A.1 TSNTag 格式

在发送端的主机内部需要根据报文七元组（目的 mac、type、IP 五元组）对时间敏感、带宽预约、尽力转发流量进行分类映射。将分类映射的结果与原报文的 DMAC 字段进行替换，以此进行网络的交换，直到接收端系统内部进行 DMAC 还原。被替换的 DMAC 字段被定义成 TSNTag。

A.1.1 分类映射关键字 Key

表 A-1 分类映射关键字 Key

位宽	名称	描述
48	DMAC	报文目的 MAC
16	ETHTYPE	报文以太网类型
8	protocol	报文协议类型
32	Sip	报文源 ip
32	Dip	报文目的 ip

OpenTSN

16	Sport	报文源端口
16	Dport	报文目的端口

A.1.2 分类映射结果 ACTION(TSNTag)

由于架构内部不再携带 metadata，因此时间同步报文的架构接收时间戳需要通过其他方式在架构内部传输，目前的方案是在 TSNTag 中携带，将 TSNTag 的格式调整如下。

由于时间同步报文的 TSNTag 中“seq_id”、“frag_id”、“inject_addr”、“submit_addr”信息是无用的，因此可以将时间同步报文的这些字段用来存放架构的接收时间戳信息。而其他非时间同步报文的架构接收时间戳信息是无用的，因此可以延用这些字段的信息。

表 A-2 时间同步报文的 TSNtag

位宽	名称	位置	描述
3	Flow type	[47:45]	流类型。100：同步报文 （其他报文的格式如下表）
14	Flow id/IMAC	[44:31]	静态流量使用 flowID，每条静态流分配一个唯一 flowID，动态流使用 imac 地址，imac 地址相同的则在交换架构命中同一条表项。
12	Reserve	[30:19]	保留
19	Rx_timestamps	[18:0]	架构接收到时间同步报文的本地时间信息，用于架构发送报文时计算透明时钟。

表 A-3 非时间同步报文的 TSNTag

位宽	名称	位置	描述
3	Flow type	[47:45]	流类型。000:ST 分组 001:ST 分组 010: ST 分

			组 011：RC 分组　　101：NMAC 分组　　110：BE 分组 111：BE 分组
14	Flow id/IMAC	[44:31]	静态流量使用 flowID，每条静态流分配一个唯一 flowID，动态流使用 imac 地址，imac 地址相同的则在交换架构命中同一条表项。
16	Seq id	[30:15]	用于标识每条流中报文的序列号
1	Frag flag	[14]	用于标识分片后的尾。0：分片后的中间报文 1：尾拍
4	Frag ID	[13:10]	用于表示当前分片报文在原报文中的分片序列号
5	inject addr	[9:5]	ST 流在源端等待发送调度时缓存地址
5	submit addr	[4:0]	ST 流在终端等待接收调度时缓存地址

A.2 metadata 格式

架构内部处理的报文数据都不携带 metadata，只在主机与架构之间携带 metadata。主机发送一个报文时，会在报文头携带一个 64B 的 metadata。主机接收处理逻辑在主机接口接收到报文后，将 metadata 中的有效数据提出并将 metadata 丢弃掉。而主机发送处理逻辑在主机接口发送报文前，将 64B 的 metadata 添加到报文头。

表 A-4 主机下发报文时，携带的 metadata 格式

位置	位宽	名称	含义	备注
[63:61]	3	pkttype	报文的类型。	000:ST 分组　　001:ST 分组 010: ST 分组　　011：RC 分组　　100：PTP 分组　　101：NMAC 分组　　110：BE 分组 111：BE 分组

[60:56]	5	inject_addr	ST 注入缓存地址	/
[55:47]	9	outport	报文的输出端口号	/
[46]	1	lookup_en	查表使能信号	/
[45:0]	46	reserve	保留	/

表 A-5 架构上传报文时，携带的 metadata 格式

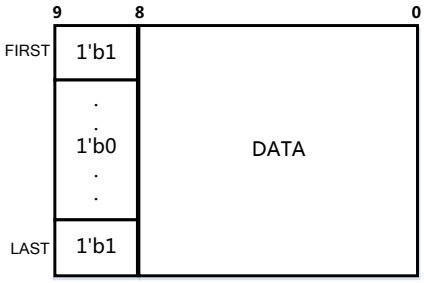
位置	位宽	名称	含义	备注
[63:16]	48	ts	同步报文全局时间信息	用于记录时间同步报文在主时钟设备接收到 req 报文的全局时间信息，以及从时钟设备接收到 sync 报文的全局时间信息
[15:12]	4	inport	报文的输入端口号	/
[11:0]	12	reserve	保留	/

A.3 架构内部传输的 pkt 数据格式

在模块之间传输的有两种格式，一种位宽为 9bit；另一种位宽为 134bit。具体区别如下：

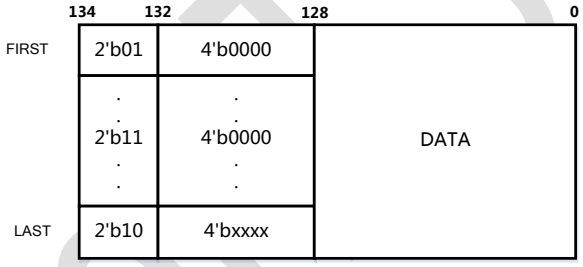
pkt_data 位宽为 9 位，包含 1bit 头尾标志、8bit 报文数据。头尾标志： 1 表示报文头/尾数据；0 标识报文体中间数据。具体如下图所示：

OpenTSN



Pkt_data 格式定义

pkt_data 位宽为 134 位，包含 2bit 头尾标志、4bit 无效字节、128bit 报文数据。头尾标志： 01 表示报文头；11 表示报文体中间数据；10 表示报文尾。无效字节用于标识报文尾中无效的字节数。具体如下图所示：



Pkt_data 格式定义

附录 B：架构内部表项定义

本章节主要对架构内部的几个关键表项进行详细介绍。

B.1 注入时刻表的设计

注入时刻表的格式如下。

表 B-1 注入时刻表格式

名称	位置	含义	备注

Value[15:0]	[15]	表项有效位；	使用 RAM 进行实现，深度为 1024。
	[14:5]	发送 Slot ；	
	[4:0]	TSNTag 中的“inject addr”	

注入时刻表设计在端接收处理逻辑中的 TS 注入调度(TIS)模块，注入时刻表需要与 TS 注入缓存管理组合工作，TS 注入缓存区的格式如下：

表 B-2TS 注入缓存区数据格式

名称	含义	备注
descriptor[40:0]	用于标识 TS 流的描述符，所包含的内容如上“Descriptor 数据格式”的低 41 位。	使用 RAM 进行实现，深度为 32。

TS 注入缓存区设计在端接收处理逻辑中的 TS 注入管理(TIM)模块，该模块接收到描述符之后，根据描述符中的注入地址将该描述符写入 TS 注入缓存区等待调度。TS 注入调度模块从头开始查找 TS 注入时刻表，当查表结果的最高 1bit 为高时，代表该查表结果有效；当该 bit 为低时，代表该查表结果无效，该模块不会再继续往下继续查表。查表结果的[4:0]为下一个需要从 TS 注入缓存区调度的描述符所储存的地址，而查表结果的[14:5]为该描述符调度的时间槽，等待对应的时间槽到达之后对描述符调度传输并读取 TS 注入时刻表的下一个地址中的数据。注入时刻表的深度为 32，代表单个 TSNSwitch 架构上的主机最多能够支持注入 32 条 ST 流。

B.2 提交时刻表的设计

提交时刻表的格式如下。

表 B-3 提交时刻表格式

名称	位置	含义	备注
Value[15:0]	[15]	表项有效位；	使用 RAM 进行实现，深度为 1024。
	[14:5]	发送 Slot ；	
	[4:0]	TSNTag 中的“submit addr”	

提交时刻表设计在端接收处理逻辑中的 TS 提交调度(TSS)模块，注入时刻表需要与 TS 提交缓存管理组合工作，TS 提交缓存区的格式如下：

表 B-4 TS 提交缓存区数据格式

名称	含义	备注
descriptor[12:0]	用于标识 TS 流的描述符，所包含的内容：pkt_bufid 以及 inport。	使用 RAM 进行实现，深度为 32。

TS 提交缓存区设计在端接收处理逻辑中的 TS 提交管理(TSM)模块，该模块接收到描述符之后，根据描述符中的提交地址将该描述符写入 TS 提交缓存区等待调度。TS 提交调度模块从头开始查找 TS 提交时刻表，当查表结果的最高 1bit 为高时，代表该查表结果有效；当该 bit 为低时，代表该查表结果无效，该模块不会再继续往下继续查表。查表结果的[4:0]为下一个需要从 TS 提交缓存区调度的描述符所储存的地址，而查表结果的[14:5]为该描述符调度的时间槽，等待对应的时间槽到达之后对描述符调度传输并读取 TS 提交时刻表的下一

个地址中的数据。提交时刻表的深度为 32，代表单个 TSNSwitch 架构上的主机最多能够支持接收 32 条 ST 流。

B.3 转发表的设计

转发表的格式如下。

表 B-5 转发表数据格式

名称	含义	备注
outport[8:0]	输出端口号(bitmap)，总共 8 个网络接口与 1 个主机接口。	使用 RAM 进行实现，深度为 16K

转发表设计在转发查表(FLT)模块，该模块会根据接收到的描述符提取出 flow_id，再将 flow_id 作为查表地址进行查找转发表。查表结果为输出端口号，其中某位为高则代表往某个对应的输出端口进行转发，本模块根据查表结果将描述符内容转发给对应的输出端口。转发表的深度为 16K，代表由 TSNSwicth 组成的整个网络最多能够支持 16K 条流的转发。

B.4 门控表的设计

门控表的格式如下：

表 B-6 门控表数据格式

名称	含义	备注
gate_ctrl_vector[7:0]	门控向量，对应 8 个门控信息。	使用 RAM 进行实现，深度为 1024

门控表设计在交换发送处理逻辑中的队列门控(QGC)模块，该模块根据接收到的时间槽信息以及时间槽切换信号来进行查表。本模块

每接收到一个时间槽切换信号，便将时间槽的 ID 作为查表地址进行查找门控表，查表结果为门控向量，对应 8 个门控信号，当某位为高则代表打开对应的某个队列的门控，网络输出调度(NOS)模块根据该门控进行对队列进行调度。门控表的深度为 1024，代表 TSNSwitch 最多能够支持 1024 个时间槽做为一个周期。

附录 C：寄存器说明

开源枫林 TSN 架构内部可配置地址空间主要有两部分，包括：MDID 模块号和真实地址空间，其中 MDID 模块号主要用来区分不同模块，而后 20 位为各个模块使用的地址空间。地址的第 19bit 位用于区别地址类型，控制/表项寄存器可读可写，调试和版本寄存器只读，每个模块的地址空间为 1024k,其中可读可写和只读寄存器各有 512k。具体地址含义如下：

表 C-1 地址格式

ADDR[26:0]		
MDID[26:20]	ADDR[19]	ADDR[18:0]
MDID ： 0-127	0	该模块的控制寄存器,表项等,可读可写
	1	只读

每个处理模块的 MDID 号分配如下：

表 C-2 模块中的 MDID 和地址

处理模块	CSM	TIS	TSS	QGC	GTS	FLT
MDID	0x0	0x1	0x2	0x3-0xa	0xb	0xc
地址	0x0-0xfffff	0x100000-0x1fffff	0x200000-0x2fffff	0x300000-0xafffff	0xb00000-0xbfffff	0xc00000-0xcfffff

C.1 CSM 模块

地址范围为 Addr 0x0-0xffff

表 C-3 CSM 模块寄存器

Addr	Data			
	[31:24]	[23:16]	[15:8]	[7:0]
0x0	offset_l			
0x1	offset_h			
0x2	time_slot			
0x3	cfg_finish			
0x4	port_type			
0x5	qbv_or_ach			
0x6	report_type			
0x7	report_en			
0x8	inject_slot_period			
0x9	submit_slot_period			
0xa	report_period			
0xb	offset_period			
0xc	rc_regulation_value			
0xd	be_regulation_value			
0xe	unmap_regulation_value			
0xf ~ 0xfffff	reserve			

表 C-4 寄存器的具体含义

name	bit	R/W	description	default
offset_l	31:17	R/	代表时间偏移的高位值的低 15	0

OpenTSN

		W	位，表示毫秒	
	16:0	R/W	时间偏移的低位，表示拍数	0
offset_h	31:17	R/W	保留位	0
	16	R/W	代表时间偏移的正负值，1 代表正值，如果为 0，则代表负值	0
	15:0	R/W	代表时间偏移的高位值的高 16 位，表示毫秒	0
time_slot	31:11	R/W	保留	
	10:0	R/W	时间槽大小	0
cfg_finish	31:2	R/W	保留	0
	1:0	R/W	配置完成寄存器， 0 代表架构正在初始化，不接收任何报文， 1 代表初始化完成，可以接收 NMAC 配置报文 2 代表配置完成，可以接收除 ST 报文的任何报文 3 代表可以接收任何报文	0
port_type	31:8	R/W	保留	0
	7:0	R/W	网络端口类型寄存器，架构共有 8 个网络端口，寄存器的 0-7 位分别代表 0-7 端口的类型，1 代表非合作类型，处理标准以太网类型的	0

OpenTSN

			报文，0 代表合作类型，处理 TSN 报文	
qbv_or_ach	31:1	R/W	保留	0
	0	R/W	调度模式选择信号，网络输出逻辑中的调度机制是 QBV 模式还是 QCH 模式 0 代表 QBV 模式；1 代表 QCH 模式	0
report_type	31:16	R/W	保留	0
	15:0	R/W	上报类型，具体参考附录 D	0
report_en	31:1	R/W	保留	0
	0	R/W	上报使能信号，配置与状态管理模块是否进行周期性上报 0 代表不上报；1 代表上报	0
inject_slot_period	31:12	R/W	保留	0
	10:0	R/W	注入时间槽周期，架构内部时间槽切换的周期值 配置的值范围：1-1024 个	0
submit_slot_period	31:12	R/W	保留	0
	10:0	R/W	提交时间槽周期，架构内部时间槽切换的周期值 配置的值范围：1-1024 个	0
report_period	31:12	R/	保留	0

OpenTSN

		W		
	11:0	R/W	上报周期，配置与状态管理模块上报的周期值 配置的值范围：1（ms）或 1000（ms）	0
offset_period	31:24	R/W	保留	0
	23:0	R/W	offset 补偿的配置周期	
rc_regulation_value	31:9	R/W	保留	0
	8:0	R/W	RC 流的监管阈值，当 BUFID 的剩余个数小于该值，开始丢弃 RC 报文	
be_regulation_value	31:9	R/W	保留	0
	8:0	R/W	BE 流的监管阈值，当 BUFID 的剩余个数小于该值，开始丢弃 BE 报文和 RC 报文	
unmap_regulation_value	31:9	R/W	保留	0
	8:0	R/W	非映射流的监管阈值，当 BUFID 的剩余个数小于该值，开始丢弃非映射报文	
reserve	31:9	R/W	保留	0

C.2 TIS 模块

地址范围为 Addr 0x100000-0x1fffff

表 C-5 地址格式

Addr	Data			
	[31:24]	[23:16]	[15:8]	[7:0]
send_table_N 0x100000-0x1003ff	ST 报文发送时刻表每项内容，N=0、1、…、1023 send_table_0 表示第 0 个发送表			
0x100400-0x1fffff	保留			

寄存器的具体含义

name	bit	R/W	description	default
send_table_0	16	R/W	保留	0
	15	R/W	表项有效位，0 代表无效，1 代表有效	0
	14:5	R/W	ST 流在一个应用周期内的注入时间槽	0
	4:0	R/W	TSNTag 中的“send addr”	0
……				
send_table_1023	16	R/W	保留	0
	15	R/W	表项有效位，0 代表无效，1 代表有效	0
	14:5	R/W	ST 流在一个应用周期内的注入时间槽	0
	4:0	R/W	TSNTag 中的“send addr”	0
0x100400-0x1fffff			保留	

C.3 TSS 模块

地址范围为 Addr 0x200000-0x2fffff。

表 C-6 地址格式

Addr	Data			
	[31:24]	[23:16]	[15:8]	[7:0]
submit_table_N 0x200000-0x2003ff	ST 报文提交时刻表每项内容，N=0、1、…、1023 submit_table_0 表示第 0 个提交表			
0x200400-0x2fffff	保留			

表 C-7 寄存器的具体含义

name	bit	R/W	description	default
submit_table_0	31:16	R/W	保留	0
	15	R/W	表项有效位，0 代表无效，1 代表有效	0
	14:5	R/W	ST 流的提交时间槽	0
	4:0	R/W	TSNTag 中的“submit addr”	0
……				
submit _table_1023	31:16	R/W	保留	0
	15	R/W	表项有效位，0 代表无效，1 代表有效	0
	14:5	R/W	当前 Slot	0
	4:0	R/W	TSNTag 中的“submit addr”	0

C.4 QGC 模块

地址范围为 Addr 0x300000-0xaffffff，其中 0x300000-0x3fffff 表示第一个端口的门控表，以此类推，共有 8 个端口门控。

表 C-8 地址格式

Addr	Data			
	[31:24]	[23:16]	[15:8]	[7:0]
port0_gate_table_N 0x300000-0x3003ff	0 号端口的门控表，N=0、1、…、1023，输出门控 port0_gate_table_0 表示 0 号端口的第一个时刻的门控状态			
port1_gate_table_N 0x400000-0x4003ff	1 号端口的门控表，N=0、1、…、1023			
port2_gate_table_N 0x500000-0x5003ff	2 号端口的门控表，N=0、1、…、1023			
port3_gate_table_N 0x600000-0x6003ff	3 号端口的门控表，N=0、1、…、1023			
port4_gate_table_N 0x700000-0x7003ff	4 号端口的门控表，N=0、1、…、1023			
port5_gate_table_N 0x800000-0x8003ff	5 号端口的门控表，N=0、1、…、1023			
port6_gate_table_N 0x900000-0x9003ff	6 号端口的门控表，N=0、1、…、1023			
port7_gate_table_N 0xa00000-0xa003ff	7 号端口的门控表，N=0、1、…、1023			

表 C-9 寄存器的具体含义

name	bit	R/W	description	default
port0_gate_table_0	31:8	R/W	保留	0
	7:0	R/W	0-7 位分别代表 0-7 共 8 个队列的门控状态，0 代表该队列的门控关闭，1 代表开启	0
……				
port7_gate_table_1023	31:8	R/W	保留	0

OpenTSN

	7:0	R/W	0-7 位分别代表 0-7 共 8 个队列的门控状态，0 代表该队列的门控关闭，1 代表开启	0

OpenTSN

C.5 FLT 模块

地址范围为 Addr 0xc00000-0xcfffff

表 C-10 地址格式

Addr	Data			
	[31:24]	[23:16]	[15:8]	[7:0]
0xc00000-0xc03fff	forward_table_N，表示转发表，N=0,1,2，…16383，forward_table_0 表示第 0 个转发表			
0xc04000-0xcfffff	保留			

表 C-11 寄存器的具体含义

name	bit	R/W	description	default
forward_table_0	31:16	R/W	保留	0
	8:0	R/W	转发表的内容，使用 bitmap 的形式，0-8 位分别代表向 0-8 号端口，每位的值 0 代表不向该端口转发，1 代表向该端口转发	0
……				
forward_table_16383	31:16	R/W	保留	0
	8:0	R/W	转发表的内容，使用 bitmap 的形式，0-8 位分别代表向 0-8 号端口，每位的值为 0 代表不向该端口转发，1 代表向该端口转发	0
0xc04000-0xcfffff			保留	

OpenTSN

附录 D：NMAC 报文格式

以太网帧头部中，类型字段为 0x1662，表示封装为 NMAC 报文。NMAC 分为配置报文和上报报文，配置报文是从主机发送给架构，上报报文是从架构发送给主机；因此将主机发送的默认为配置报文，架构发送的默认为上报报文，不在报文内部进行标识。

D.1 配置报文格式

配置报文格式：在报文中用 count 字段（8bit）表示报文中包含的配置条目数，报文最小为 64 字节，最后不够 64 字节的报文需要补零。NMAC 命令在以太网报文中的封装如图所示：

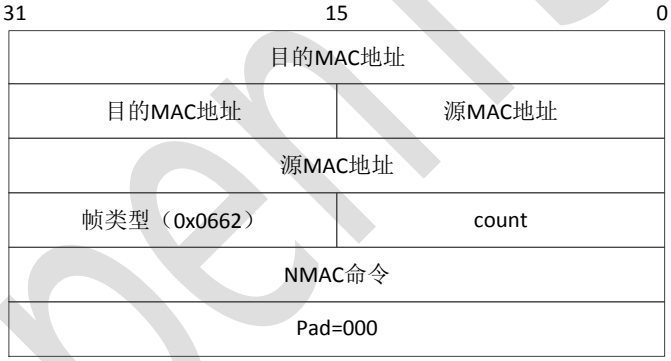


图 D-1 NMAC 配置报文格式

NMAC 命令的格式如下：

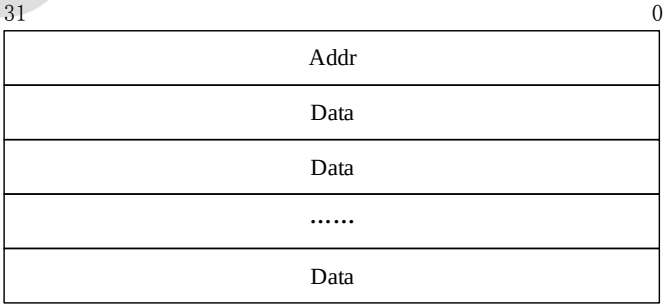


图 D-2 NMAC 命令格式

当配置的寄存器数量为 1，NMAC 命令就包括 32bit 的 ADDR 和 32bit 的 DATA；当配置的寄存器数量为 N（N>1），NMAC 命令就包括 32bitADDR 和 N*32bit 的 DATA，第一个 DATA 以 ADDR 作为 RAM 写地址，第二个以及后续 DATA 以 ADDR 循环加 1 作为 RAM 写地址。

D.2 上报报文格式

上报报文格式：其中报文类型为 NMAC 报文（0x1662）

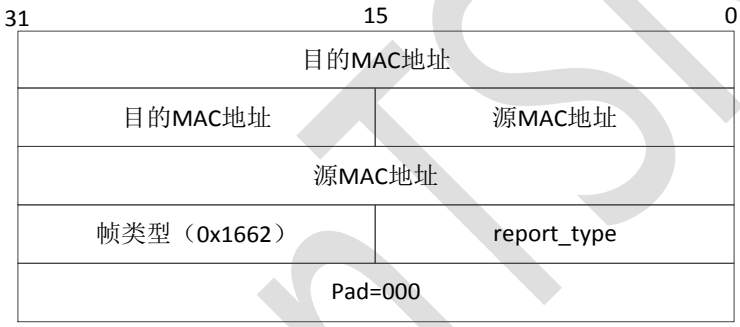


图 D-3 NMAC 上报报文格式

表 D-1 上报类型格式如下：

上报类型（16bit）		含义
高 6bit	低 10bit	
000000 单个寄存器	0	配置的单个寄存器，包含配置完成寄存器、端口状态寄存器、 时间槽大小寄存器、时间偏移寄存器、上报周期寄存器、 上报类型寄存器、应用周期寄存器
000001 转发表	0	第 0-63 条转发表
	1	第 64-127 条转发表
	2-255	第 128-16383 条转发表
000010 注入时刻表	0	第 0-63 条注入时刻表
	1	第 64-127 条注入时刻表

	2-15	第 128-1023 条注入时刻表
000011 提交时刻表	0	第 0-63 条注入时刻表
	1	第 64-127 条注入时刻表
	2-15	第 128-1023 条注入时刻表
000100-001011 P0-P7 输出门控表	0	第 0-63 条注入时刻表
	1	第 64-127 条注入时刻表
	2-15	第 128-1023 条注入时刻表
001100 xx_state	0	具体参考附录 D

单个寄存器，report_type 高 6bit 为 0000，低 10bit 为 0。

转发表上报报文格式，report_type 高 6bit 为 000001，低 10bit 为上报的第几块，转发表一共有 16K 条，每条转发表占用 2 字节（9bit），因此每个报文可以携带 64 条，总共需要 256 个报文。

注入时刻表上报报文格式，report_type 高 6bit 为 000010，低 10bit 为上报的第几块，注入时刻表一共有 1024 条，每条转发表占用 2 字节（9bit），因此每个报文可以携带 64 条，总共需要 16 个报文。

提交时刻表上报报文格式，report_type 高 6bit 为 000011，低 10bit 为上报的第几块，注入时刻表一共有 1024 条，每条转发表占用 2 字节（9bit），因此每个报文可以携带 64 条，总共需要 16 个报文。

门控表按照端口划分，每个门控占用一块 RAM，共有 8 个端口。每个端口两块 RAM，总共需要 16 块 RAM，report_type 的具体格式如上表所示。

状态上报内容，由各个模块提供需要上报的内容，详细参考附录D。

报文内部具体含义如下：

1）单个寄存器上报报文：

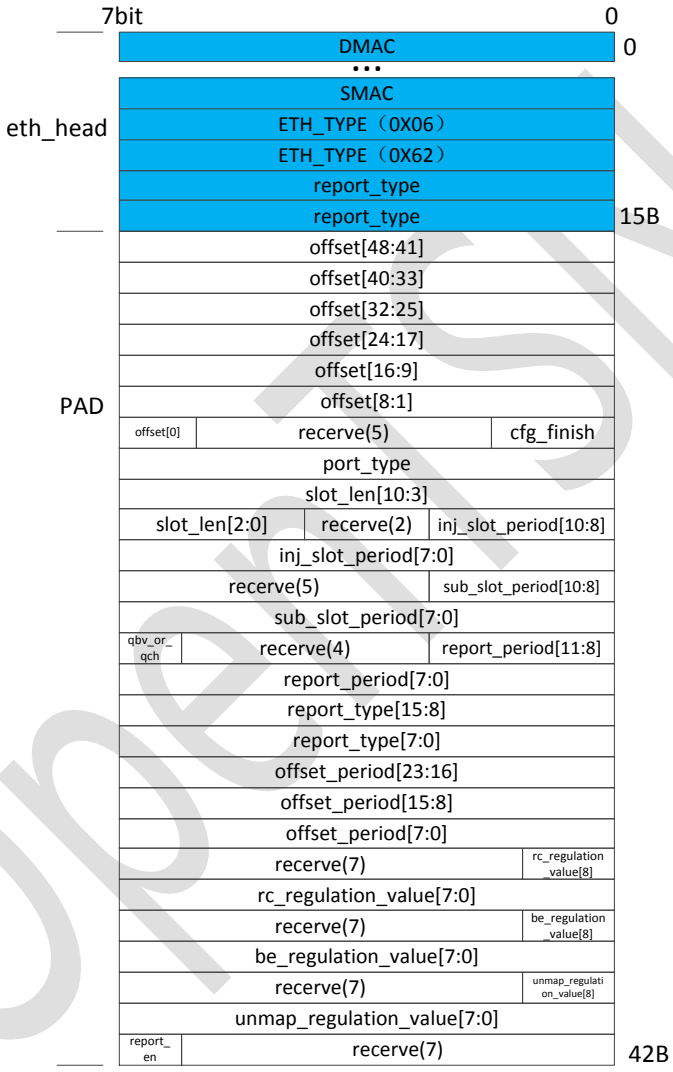


图 D-4 单个寄存器 NMAC 上报报文格式

2）各模块状态上报报文：

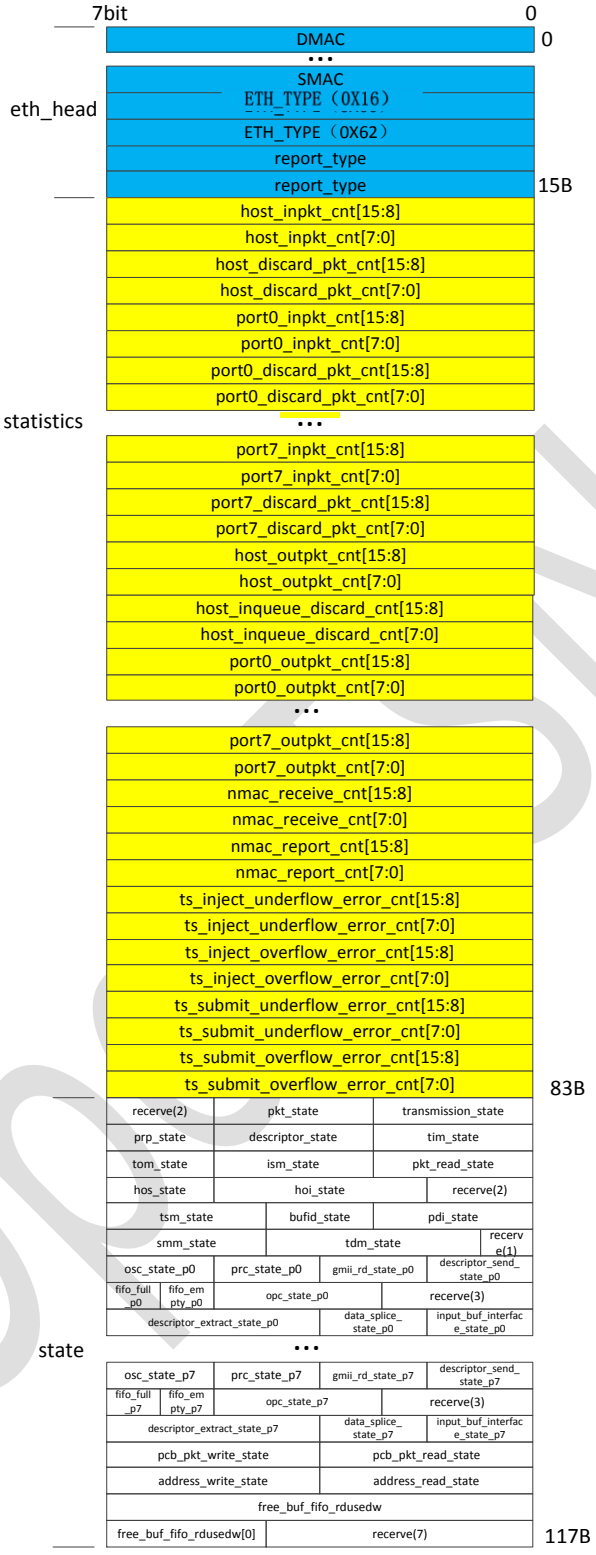


图 D-5 各模块状态 NMAC 上报报文格式

3）表项上报报文：

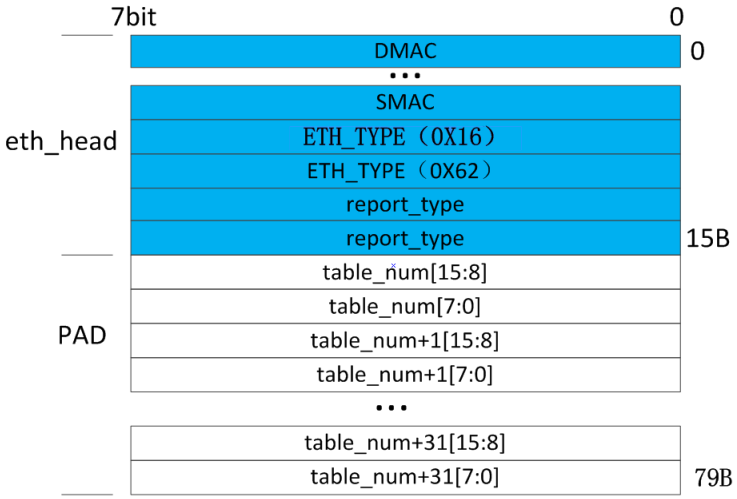


图 D-6 表项 NMAC 上报报文格式

各模块通过 NMAC 上报的状态信息如下表，NMAC 上报的状态主要包括各个端口的收发报文计数器和各个模块的状态信息。

表 D-2 各模块上报的计数器统计

名称	位宽	含义
host_inpkt_cnt	16	主机口接收报文个数计数器
host_discard_pkt_cnt	16	主机口接收后丢弃的报文个数计数器
port0_inpkt_cnt	16	0 号网络口接收报文个数计数器
port0_discard_pkt_cnt	16	0 号网络口接收后丢弃的报文个数计数器
port1_inpkt_cnt	16	1 号网络口接收报文个数计数器
port1_discard_pkt_cnt	16	1 号网络口接收后丢弃的报文个数计数器
port2_inpkt_cnt	16	2 号网络口接收报文个数计数器
port2_discard_pkt_cnt	16	2 号网络口接收后丢弃的报文个数计数器
port3_inpkt_cnt	16	3 号网络口接收报文个数计数器

port3_discard_pkt_cnt	16	3 号网络口接收后丢弃的报文个数计数器
port4_inpkt_cnt	16	4 号网络口接收报文个数计数器
port4_discard_pkt_cnt	16	4 号网络口接收后丢弃的报文个数计数器
port5_inpkt_cnt	16	5 号网络口接收报文个数计数器
port5_discard_pkt_cnt	16	5 号网络口接收后丢弃的报文个数计数器
port6_inpkt_cnt	16	6 号网络口接收报文个数计数器
port6_discard_pkt_cnt	16	6 号网络口接收后丢弃的报文个数计数器
port7_inpkt_cnt	16	7 号网络口接收报文个数计数器
port7_discard_pkt_cnt	16	7 号网络口接收后丢弃的报文个数计数器
host_outpkt_cnt	16	主机口发送报文个数计数器
host_inqueue_discard_cnt	16	主机发送模块入队时丢弃报文个数计数器
port0_outpkt_cnt	16	0 号网络口发送报文个数计数器
port1_outpkt_cnt	16	1 号网络口发送报文个数计数器
port2_outpkt_cnt	16	2 号网络口发送报文个数计数器
port3_outpkt_cnt	16	3 号网络口发送报文个数计数器
port4_outpkt_cnt	16	4 号网络口发送报文个数计数器
port5_outpkt_cnt	16	5 号网络口发送报文个数计数器
port6_outpkt_cnt	16	6 号网络口发送报文个数计数器
port7_outpkt_cnt	16	7 号网络口发送报文个数计数器
nmac_receive_cnt	16	CSM 模块接收到的 NMAC 报文个数计数器
nmac_report_cnt	16	CSM 模块上报的 NMAC 报文个数计数器

OpenTSN

ts_inj_underflow_error_cnt	16	ST 报文注入下溢错误计数器
ts_inj_overflow_error_cnt	16	ST 报文注入上溢错误计数器
ts_sub_underflow_error_cnt	16	ST 报文提交下溢错误计数器
ts_sub_overflow_error_cnt	16	ST 报文提交上溢错误计数器
总计	/	34 个，544bit，68B

表 D-3 各模块上报的状态统计

名称	所属模块	位宽	含义
pkt_state	PMD>TRW	3	TRW 状态机
transmission_state	PMD>TRR	3	TRR 状态机
prp_state	HRX>PRP	2	PRP 状态机
descriptor_state	PMD>PDG	3	PDG 状态机
tim_state	TIM	3	TIM 状态机
tom_state	PMD>TOM	2	TOM 状态机
iv_ism_state	TIS>ISM	3	ISM 注入调度状态机
pkt_read_state	HTX>HRC	3	HRC 状态机
hos_state	HOS	2	HOS 状态机
hoi_state	HTX>HOI	4	HOI 状态机
tsm_state	TSM	3	TSM 状态机
bufid_state	TSM	2	BUFID 状态机
pdi_state	HRX	3	PDI 状态机
smm_state	TSS>SS	3	SSM 提交调度状态机

OpenTSN

	M		
tdm_state	FLT>TDM	4	TDM 的状态机
osc_state_p*N	NOS>OSC	2	OSC 的状态机（N=0-7）
prc_state_p*N	NOS>PRC	2	PRC 的状态机（N=0-7）
gmii_read_state*N	NRX>GR	2	GW 的状态机（N=0-7）
descriptor_send_state*N	FPM>DSM	2	DSM 的状态机（N=0-7）
gmii_fifo_full*N	NRX	1	输入时钟域切换 fifo 满信号（N=0-7）
gmii_fifo_empty*N	NRX	1	输入时钟域切换 fifo 空信号（N=0-7）
opc_state_p*N	NOS>OPC	3	OPC 的状态机（N=0-7）
descriptor_extract_state*N	FPM>DEM	4	DEM 的状态机（N=0-7）
data_splice_state*N	FPM>DSP	2	DSP 的状态机（N=0-7）
input_buf_interface_state*N	IBI	2	IBI 的状态机（N=0-7）
pkt_write_state	PCB>PW	4	PW 的状态机
pkt_read_state	PCB>PR	4	PR 的状态机
address_write_state	PCB>AW	4	AW 的状态机
address_read_state	PCB>AR	4	AR 的状态机
free_buf_fifo_rdusedw	PCB	9	空闲地址管理 fifo 的个数

OpenTSN