时间敏感网络交换处理逻辑（TSS）

设计方案

（版本1.0）

OpenTSN开源项目组

2021年4月

**版本历史**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **修订时间** | **修订内容** | **文件标识** |
| **1.0** | **2021.4** | **初版编制** | **OpenTSN3.0** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

目录

[1、概述 4](#_Toc68946514)

[2、总体设计 4](#_Toc68946515)

[2.1. 总体架构 4](#_Toc68946516)

[2.1.1网络输入处理逻辑 5](#_Toc68946517)

[2.1.2网络输出处理逻辑 7](#_Toc68946518)

[2.1.3内部处理逻辑 8](#_Toc68946519)

[2. 帧的处理流程 10](#_Toc68946520)

[附录1：数据格式定义 13](#_Toc68946521)

[附录2：内部表项格式定义 15](#_Toc68946522)

[附录3：内部寄存器定义 16](#_Toc68946523)

[附录4：NMAC报文格式 22](#_Toc68946524)

[附录5：command/command\_ack命令格式 25](#_Toc68946525)

**1、概述**

OpenTSN2.0开源逻辑既可作为TSN网卡又可作为TSN交换机使用，为了简化其作为TSN交换机使用时的逻辑复杂度以及增强其作为TSN网卡使用时的功能可扩展性，并且考虑到逻辑模块的复用，计划将TSN2.0开源逻辑拆分为时间敏感网络硬件控制逻辑（HCP）、时间敏感网络端处理逻辑(TSE)、时间敏感网络交换处理逻辑(TSS)三个模块进行拆分设计。本文主要详细介绍时间敏感网络交换处理逻辑的设计。

**2、总体设计**

## 2.1. 总体架构

时间敏感网络交换处理逻辑模块的总体架构框图如图2-1，其中GMII4接口为内部连接HCP模块的专用接口。



图2-1 总体架构框图

总体架构图中的信号格式定义如下表2-1。

表2-1 TSS总体架构顶层信号定义

| 信号 | 位宽 | 含义 |
| --- | --- | --- |
| pkt\_134（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\_9（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 |
| descriptor | 46 | 报文描述符数据，用于标识报文数据 |
| pkt\_bufid | 9 | 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 |
| 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 |
| command | 204 | 读、写命令 |
| command\_ack | 204 | 读命令响应 |

下面将整个架构划分为三大逻辑模块进行介绍，分别对应顶层中的网络输入处理逻辑、网络输出处理逻辑以及内部处理逻辑。

### 2.1.1网络输入处理逻辑

网络输入处理逻辑的内部组成框图如图2-2。



图2-2网络输入处理(NIP)模块内部组成框图

NRX(Network RX)网络接收模块：主要功能是接收网络接口发送的报文，完成报文从外部时钟域到架构内部时钟域的切换，以及完成架构接收时间同步报文的时间信息记录，并在TSNTag中进行记录。模块内部维护一个寄存器，根据此寄存器的值判断是否接收并处理数据。

FPA(Frame PArse)帧解析模块：主要功能是提取帧的DMAC（TSNTag）中的信息，并构造报文描述符。还需根据报文的以太网类型字段判断此报文是否经过分类映射，若未经过分类映射则在构造描述符时默认送往主机。在往输入缓存接口模块发送数据之前，需要将一拍8bit的数据转换成一拍128bit的数据。

IBI(Input Bufm\_memory Interface)输入缓存接口模块：主要功能是将报文数据发送给报文集中缓存模块进行缓存。本模块接收帧解析模块的128bit数据使用两个寄存器进行缓存，其中任何一个寄存器有数据则往报文集中缓存门控发送一个写信号，同时将进入本模块的数据写入另一个寄存器。

### 2.1.2网络输出处理逻辑



图2-3网络输出处理(NOP)模块内部组成框图

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、队列门控模块发送的门控信息进行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：主要功能是控制当前输出端口队列打开或关闭。本模块根据架构的全局时钟进行门控列表的读取，并将门控列表(门控列表的格式如下表2-2)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2-2 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：主要功能是根据调度后的队列信息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发送的门控信息进行计算，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得到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、完成报文从内部处理时钟域到外部PHY架构的时钟域的切换、将数据报文构造帧前导符和帧开始符后由网络接口传输。读取报文时，需要先将pkt\_bufid映射成地址，并根据此地址往报文集中缓存模块进行报文数据的提取，同时需要将此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。

### 2.1.3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进行查找转发表（转发表的数据格式如下表2-3）、根据查表得到的输出端口进行pkt\_buf id和pkt\_type的转发。当描述符中查表使能信号（lookup\_en）为低（代表本描述符不需要查表）时，直接提取出描述符中的输出端口（outport）字段作为输出端口号进行转发。当得到的输出端口多位高，意味处理的描述符代表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的缓存。

表2-3 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 Bufm\_memory)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（报文缓存区的格式如下表2-4）、为每个缓存的报文分配一个pkt\_bufid进行标识、对所有空闲的pkt\_bufid进行缓存（空闲地址缓存区的格式如下表2-5）。为所有pkt\_bufid增加寄存器对输出端口数量进行计数，当pkt\_bufid使用完进行释放的时候需要检测该pkt\_bufid对应的计数值，只有当计数值为0才能进行释放，当不为0意味着该pkt\_bufid存着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64KB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长度最多为128B的报文。

表2-4 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2-5 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”中空闲的报文缓存块ID号。 | 使用RAM进行实现，深度为512。 |

CPA(Command PArse)命令解析模块：负责将接收到HCP的command命令进行解析，（命令数据格式见表2-6）来实现对本地寄存器、转发表、门控列表的配置。周期性的将各个模块的实时状态封装成command\_ack命令上报给HCP模块。

表2-6 command/command\_ack命令格式

| 位置 | 位宽 | 名称 | 说明 |
| --- | --- | --- | --- |
| [203:180] | 8 | node\_id | 该字段用来标识对哪个节点进行读写。每个TSE或TSS都有一个唯一的节点ID。该字段在TSN网卡+TSN交换机模式下使用到。 |
| [179:172] | 8 | dest\_module\_id | 该字段用来标识对一个节点内的哪个模块进行控制。TSE或TSS内部每个子模块都有一个唯一的模块ID |
| [171:168] | 4 | type | 4’b0001:寄存器或表项的写命令;  4’b0010:寄存器或表项的读命令；  4’b0110:寄存器或表项的读响应。 |
| [167:152] | 16 | addr | 寄存器或表项的读/写地址 |
| [151:0] | 152 | data | 寄存器或表项的读/写数据；其中五元组映射表的表项位宽最大，为152bit |

## 2. 帧的处理流程

网口配置的端口类型保持一致，所有从网口进入的报文的处理流程都是通过查表去确认输出端口，详细流程如下：

1. 数据传输报文由p0-p3网口（gmii\_rx）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、构造metadata与报文数据并行往帧解析模块发送；
2. 帧解析模块收到报文数据后，对报文进行解析，提取报文的特征信息（pkt\_type、Flow id、submit addr、Lookup\_en、Outport）与报文集中缓存模块分配的pkt\_bufid构造成描述符发送到转发查表模块；同时帧解析模块将8bit的数据写入到134bit的寄存器中，当一个寄存器写满了134bit便将寄存器中的数据传输到输入缓存接口模块，完成报文的位宽转换；
3. 输入缓存接口模块接收134bit的分组数据，因报文集中缓存区是采用分时复用的方式进行数据的写入，因此输入缓存接口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，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；
4. 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；
5. 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。
6. 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。
7. 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p0-p3接口（gmii\_tx）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。

**附录1：数据格式定义**

* TSNTag格式

在流量发送端的网卡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TSN网络的交换，直到接收端的网卡内部进行DMAC还原。被替换的DMAC字段被定义成TSNTag。

表附1-1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 |
| 16 | Sport | 报文源端口 |
| 16 | Dport | 报文目的端口 |

因同步报文的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。

表附1-2 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。 |

表附1-3 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 |

* 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。具体如图附1-1所示：



图附1-1 pkt\_data数据格式1

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。具体如图附1-2所示：



图附1-2 pkt\_data数据格式2

**附录2：内部表项格式定义**

* 转发表

转发表的格式如表附2-1所示。

表附2-1 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。

* 门控表

门控表的格式如表2-2所示：

表附2-2 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3：内部寄存器定义**

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。

表附3-1 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：、

表附3-2 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 |

* CSM模块

地址范围为Addr 0x0-0xffff。

表附3-3 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 | | | |

表附3-3 寄存器的具体含义

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| name | bit | R/W | description | default |
| offset\_l | 31:17 | R/W | 代表时间偏移的高位值的低15位，表示毫秒 | 0 |
| 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 | 31:1 | R/W | 保留 | 0 |
| 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报文，0代表合作类型，处理TSN报文 | 0 |
| qbv\_or\_ach | 31:2 | R/W | 保留 | 0 |
| 1:0 | 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/W | 保留 | 0 |
| 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 |

* TIS模块

地址范围为Addr 0x100000-0x1fffff。

表附3-4 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 | | | |

表附3-5 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 |  |

* TSS模块

地址范围为Addr 0x200000-0x2fffff。

表附3-6 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 | | | |

表附3-7 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“send 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“send addr” | 0 |

* QGC模块

地址范围为Addr 0x300000-0xafffff，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。

表附3-8 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 | | | |

表附3-9 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 |
| 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 |

* FLT模块

地址范围为Addr 0xc00000-0xcfffff。

表附3-10 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，…16384，  forward\_table\_0表示第0个转发表 | | | |
| 0xc04000-0xcfffff | 保留 | | | |

表附3-11 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\_16384 | 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 |  |

**附录4：NMAC报文格式**

* 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附4-1所示。



图附4-1 NMAC配置报文格式

NMAC命令的格式如图附4-2所示。



图附4-2 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。

* 上报报文格式

上报报文格式：其中报文类型为NMAC报文（0x1662）。



图附4-3 NMAC上报报文格式

表附4-1 上报类型格式

|  |  |  |
| --- | --- | --- |
| 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。

**附录5：command/command\_ack命令格式**

表附5-1 command/command\_ack命令格式

| 位置 | 位宽 | 名称 | 说明 |
| --- | --- | --- | --- |
| [203:180] | 8 | node\_id | 该字段用来标识对哪个节点进行读写。每个TSE或TSS都有一个唯一的节点ID。该字段在TSN网卡+TSN交换机模式下使用到。 |
| [179:172] | 8 | dest\_module\_id | 该字段用来标识对一个节点内的哪个模块进行控制。TSE或TSS内部每个子模块都有一个唯一的模块ID |
| [171:168] | 4 | type | 4’b0001:寄存器或表项的写命令;  4’b0010:寄存器或表项的读命令；  4’b0110:寄存器或表项的读响应。 |
| [167:152] | 16 | addr | 寄存器或表项的读/写地址 |
| [151:0] | 152 | data | 寄存器或表项的读/写数据；其中五元组映射表的表项位宽最大，为152bit |