**简介**

**定义**

ERPS（Ethernet Ring Protection Switching），即以太网多环保护技术，是ITU-T定义的一种二层破环协议标准，标准号为ITU-T G.8032/Y1344，因此又称为G.8032。它定义了RAPS(Ring Auto Protection Switching)协议报文和保护倒换机制。

ERPS目前支持v1和v2两个版本，v1是ITU-T组织在2008年6月发布的版本，v2是ITU-T在2010年8月发布的版本。v2版本完全兼容v1版本，并在v1版本的基础上进行了如下功能扩展：

* 相交环等多环组网方式
* 子环传输RAPS报文采用虚通道或非虚通道
* 人工进行阻塞点的切换，包括强制切换和手工切换
* ERPS环的回切模式可配置

**目的**

以太网交换网络中为了进行链路备份，提高网络可靠性，通常会使用冗余链路（例如：环形网络）。但是使用冗余链路会在网络上产生环路，可能会引起广播风暴以及MAC地址表不稳定等现象，从而影响用户通信质量，甚至导致通信中断。为了解决环路问题，定义了STP/RSTP/MSTP、ERPS等破环标准协议。

目前对于二层以太网中的故障倒换时间要求越来越高。生成树协议无法满足链路的收敛性能要求。ERPS吸取了STP等环网保护技术的优点，优化了检测机制，收敛速度快，可以满足电信级的要求；同时具备良好的兼容性，如果环网内制造商的设备都支持该协议，则可以实现互通。

**基本概念**

ERPS是一个用于以太网链路层破除环路的协议。它以ERPS环为基本单位，包含若干个节点，通过阻塞RPL Owner端口，并控制其他普通端口，使得端口的状态在Forwarding和Discarding之间切换，达到消除环路的目的。同时我们利用控制VLAN、数据VLAN和保护实例等机制，以更好地实现ERPS的功能。

如图1所示，为了提高链路可靠性，SwitchA～SwitchD组成双归链路，接入上级网络。这样的接入方式在整个网络中将引入新的环路。为了消除网络中的冗余环路，有效地保证链路连通性，需要启动破除环路机制。

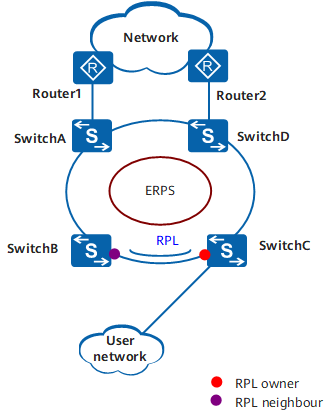


图1 ERPS单环示意图

**ERPS环**

ERPS环，就是由一组配置了相同的控制VLAN且互连的二层交换设备构成，是ERPS协议的基本单位。

ERPS环分为主环和子环。缺省情况下，ERPS环都是主环。主环是封闭的环，子环是非封闭的环，需要通过命令进行配置。如图2所示，SwitchA～SwitchD组成的ERPS环是主环，SwitchC～SwitchF组成的环是子环。

子环的配置只有v2版本支持，v1版本不支持。

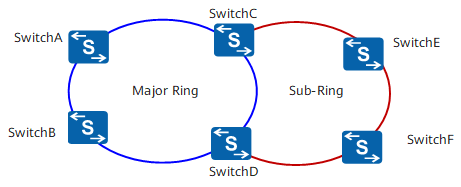


图2 ERPS主环和子环示意图

主环和子环的配置主要在相交节点上，子环在相交节点上只配置一个端口，主环在相交节点上按正常配置，指定east和west两个端口。

比如主环所有节点都会配置两个端口，east和west：

ethernet ring-protection *ring-number* **east** *interface-type* *interface-number* **west** *interface-type interface-number*

子环在除相交节点外其他普通转发节点上也和主环一样要配置两个端口，但是在相交节点上只配置单端口：

配置SwitchC端口east：

ethernet ring-protection *ring-number* **east** *interface-type interface-number*

配置SwitchD端口west：

ethernet ring-protection *ring-number* **west** *interface-type interface-number*

**节点**

加入ERPS环的二层交换设备称之为节点。每个节点不能多于两个端口加入同一个ERPS环。图2中的SwitchA～SwitchD便为ERPS主环的节点。

**端口角色**

ERPS协议中规定的端口角色主要有RPL owner端口、RPL neighbour端口和普通端口三种类型。其中RPL neighbour端口类型只有v2版本支持，v1版本不支持。

* **RPL owner端口**

一个ERPS环只有一个RPL owner端口，由用户配置决定，通过阻塞RPL owner端口转发用户流量来防止ERPS环中产生环路。

当RPL owner端口所在设备收到故障报文得知ERPS环上其他节点或链路故障时，会自动放开RPL owner端口，此端口恢复流量的接收和发送，保证流量不会中断。

RPL owner端口所在的链路即为环保护链路RPL（Ring Protection Link）。

* **RPL neighbour端口**

RPL neighbour端口指的是与RPL owner端口直接相连的端口。

正常情况下，RPL owner端口和RPL neighbour端口都会被阻塞，以防止环路产生。

当ERPS环出现故障时，RPL owner端口和RPL neighbour端口都会被放开。

引入RPL neighbour端口角色可以减少RPL neighbour端口所在设备刷新FDB表项的次数。

* **普通端口**

在ERPS环中，除RPL owner端口和RPL neighbour端口以外的端口都是普通端口。

普通端口负责监测自己直连的ERPS协议的链路状态，并把链路状态的变化消息及时通知其他端口。

**端口状态**

在ERPS环中，启动ERPS协议的端口状态分为两种：

* **Forwarding**：在Forwarding状态下，端口既转发用户流量又接收/发送ERPS协议报文。
* **Discarding**：在Discarding状态下，端口仅能发送和接收ERPS协议报文，一般也叫blocking状态。

**环状态**

**Idle**：表示当前阻塞点在RPL Owner端口，说明环路链路正常；

**Protection**：表示当前链路故障或设备故障，已经放开RPL链路，处于保护链路状态；

**Pending**：ERPS环协商的过渡状态，比如链路故障清除后处于WTR回切阻塞RPL计时中；

**ForcedSwitch**：表示通过强制切换，对ERPS环上的节点端口状态进行阻塞，放开原本阻塞端口；

**MannualSwitch**：表示通过人工切换，对ERPS环上的节点端口状态进行阻塞，放开原本阻塞端口；

**控制VLAN和数据VLAN**

在ERPS环中，控制VLAN用来传递ERPS协议报文。

每个ERPS环必须配置控制VLAN。当端口加入已经配置控制VLAN的ERPS环后，端口将自动加入控制VLAN。

不同ERPS环不能使用相同ID的控制VLAN。

与控制VLAN相对，数据VLAN用来传递正常的业务报文。

**保护实例**

对于运行ERPS协议的二层设备，传递ERPS协议报文和数据报文的VLAN必须映射到保护实例中，这样ERPS协议才会按照其阻塞原则对这些报文进行转发或阻塞。否则，VLAN报文可能会在成环的网络中产生广播风暴导致网络不可用。

简单理解，一个实例就是一个ERPS环，环上所有节点都在这个实例里面，这个实例运行的ERPS协议只对实例里的节点有破环作用，不在环上的节点需要另外创建一个实例来运行ERPS协议，或者修改已存在的实例，将新的节点加入已存在的实例。

**定时器**

ERPS协议中使用的定时器主要有Guard定时器、WTR（Wait to Restore）定时器、Holdoff定时器和WTB（Wait to Block）定时器。其中WTB定时器只有v2版本支持，v1版本不支持。

* **Guard Timer**

链路故障或节点故障所涉及到的设备在故障恢复或执行清除操作后，向其他设备发送NR RAPS报文，并同时启动Guard Timer定时器，在该定时器超时前不处理RAPS报文，目的是防止收到过期的RAPS报文，导致节点端口状态设置错误，可能形成环路或者链路恢复后收到NR RAPS报文准备重新block RPL端口时收到过期的SF RAPS，导致RPL继续保持放开。如果定时器超时后还能收到其他端口发送的NR RAPS报文，则本端口的转发状态变为Forwarding状态。

协议规定的可配置范围是10ms到2s，配置步长为10ms，我们现在的平台默认是2s。

* **WTR Timer**

RPL owner端口由于其他设备或链路故障而被放开后，当故障恢复时，有的端口可能还未由Down状态变为Up状态。为了防止立即阻塞RPL owner端口而引起网络震荡，当RPL owner端口收到某端口的NR RAPS报文后，会启动WTR定时器，环网状态迁移到Pending状态。如果在定时器未超时前收到其他端口的SF RAPS报文，则关闭WTR Timer定时器。如果在WTR定时器超时前始终没有收到其他端口的SF RAPS报文，则当WTR定时器超时后，阻塞RPL owner端口，发送NRRB RAPS报文。其他端口在收到该报文后，再将自己端口的转发状态设置为Forwarding状态。

协议规定的可配置范围是1分钟到12分钟，配置步长为1分钟，我们现在的平台默认是5分钟。

* **Holdoff Timer**

对于运行ERPS的二层网络，保护倒换的顺序可能会有不同的要求，例如：多层业务的应用中，服务器出现故障后，用户可能会希望能有一段时间恢复服务器的故障，而客户端感知不到，即不会立即进行保护倒换。可设置合适的Holdoff定时器，当发生故障时，故障并不会立即上报ERPS，进行保护倒换，而只有当Holdoff定时器超时后，如果故障仍未能恢复才会上报。

Holdoff配置后会影响50ms倒换性能，所以一般不设置这个定时器，只在特殊需求时使用。

标准建议的可配置范围是0到10秒，配置步长为100ms，默认配置0，关闭定时器。

* **WTB Timer**

当清除端口的手工切换状态（强制切换或手工切换）时，启用WTB定时器，因为ERPS环内可能存在多个手工切换阻塞节点，只有当定时器超时后，清除操作才起作用，这样可以防止立即阻塞RPL owner端口而引起阻塞点震荡。

WTB定时器不支持配置，该定时器的值为Guard定时器的值加5s。

**回切/非回切模式**

当ERPS链路恢复正常后，可以通过设置ERPS的回切/非回切模式来决定是否重新阻塞RPL owner端口。

在回切模式下，如果故障链路恢复，等待WTR时间后，会重新阻塞RPL owner端口。阻塞链路会重新切回到RPL上。

在非回切模式下，如果故障链路恢复，不启动WTR Timer定时器，而且阻塞链路还保持在原来的故障链路上，不会重新切回到RPL上。

缺省情况下，ERPS环处于回切模式。

v1版本只支持回切模式，v2版本两种模式都支持。

**阻塞点切换方式**

由于RPL owner端口所在链路可能拥有更高的带宽，此时可以考虑将带宽低的链路进行阻塞，让用户流量回到RPL上进行传输。ERPS支持通过人为的配置来干预端口的阻塞，包括强制切换（Force Switch）和手工切换（Manual Switch）两种倒换方式。

* **强制切换**：配置了强制切换的端口会马上被阻塞，不管环上其他链路是否存在故障等情况。
* **手工切换**：如果环的状态为Idle或Pending时，配置手工切换的端口就会阻塞，否则不阻塞。

除了强制切换和手工切换，ERPS还支持清除操作，该功能主要用于如下三种情况：

* 清除本地配置的手工切换和强制切换功能。
* 当ERPS环处于回切模式时，在WTB定时器或WTR定时器超时之前，手工触发回切动作。
* 当ERPS环处于非回切模式时，手工触发回切动作。

阻塞点手工切换是v2版本的功能，在v1版本不支持。

**子环RAPS报文传输方式**

v2版本除了支持单环组网，还支持相交环等多环组网方式。在相交环组网中，子环RAPS报文传输方式分为虚通道VC（Virtual-Channel）和非虚通道NVC（Non-Virtual-Channel）两种方式。

* **虚通道方式**：子环的RAPS协议报文会通过相交节点在主环内运行。即相交节点不终结子环的协议报文。子环的阻塞端口会同时阻塞子环的RAPS协议报文和数据流量。
* **非虚通道方式**：子环的RAPS协议报文会在相交节点上终结，子环的阻塞端口仅阻塞数据流量，不阻塞子环的RAPS协议报文。

如图3所示，一个主环分别和两个子环相交，其中左边子环的RAPS报文传输方式为虚通道方式，右边子环的RAPS报文传输方式为非虚通道方式。

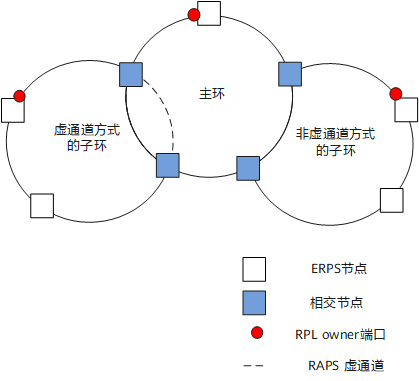


图3 虚通道和非虚通道相交环示意图

缺省情况下，子环RAPS报文传输方式为非虚通道方式，除了如图4所示的特殊组网场景下（**子环的链路是不连续的多个部分**）必须使用虚通道方式外，其他组网建议采用缺省的非虚通道方式即可。

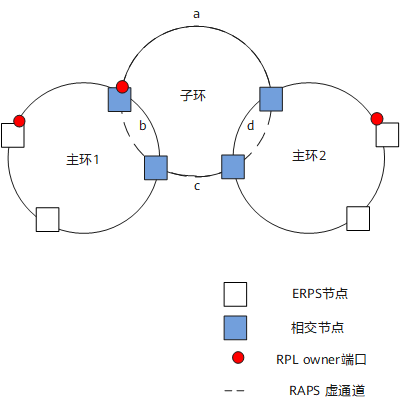
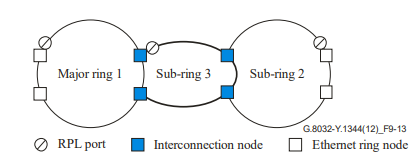


图4 虚通道特殊应用组网图

如图4所示，链路b和链路d分别属于主环1和主环2，只有链路a和链路c属于子环，链路a和链路c是独立的两条链路，无法感知到对方的链路变化，所以此时需要采用虚通道来传输RAPS报文。这样的话，两个主环各自发生保护倒换时不会互相影响，只有在子环发生保护倒换时才可能会导致两个主环刷新mac表。

而且图4这种两个主环通过一个子环相连的网络拓扑，如果使用非虚通道方式，那么在网络发生故障，拓扑发生变化时，其中一个主环可能会被重新定义为子环：



这样的话，右边两个子环的保护倒换可能会影响左边主环，导致mac表刷新。

另外，虚通道方式需要额外的带宽资源，而且如果RAPS消息在虚通道上传输的距离过长，可能会影响倒换效率，达不到电信级的50ms倒换要求。所以一般子环采用非虚通道方式配置。

虚通道和非虚通道两种子环RAPS报文传输方式的优缺点比较如下表所示：

|  |  |  |
| --- | --- | --- |
| 子环RAPS报文传输方式 | 优点 | 缺点 |
| 虚通道 | 可应用于如图4所示的特殊组网 | 子网的RAPS通道受相连的网络拓扑影响，需要在RAPS通道所在网络为虚通道预留资源、分配控制VLAN ID等 |
| 非虚通道 | 不需要相邻网络预留资源、分配控制VLAN ID等 | 不能应用于如图4所示的特殊组网 |

**协议报文**

ERPS协议的报文只有一种，即RAPS PDU报文，RAPS PDU报文包含ERPS环信息，在ERPS环上传递以实现各设备端口信息的互通。

R-APS报文具体包含在R-APS PDU（Protocol Data Unit）即协议数据单元中，属于以太网OAM消息的一种。

协议建议R-APS报文使用目的MAC地址：01-19-A7-00-00-[Ring ID]。MAC地址的最后一个字节指定为环ID，即命令行配置时指定的环ID。接口在处理收到的R-APS PDU报文时会检查目的MAC地址最后一个字节的环ID，看是否与本接口配置的环ID一致，不一致则丢弃不处理该协议报文。

RAPS PDU报文基本格式如图5所示：

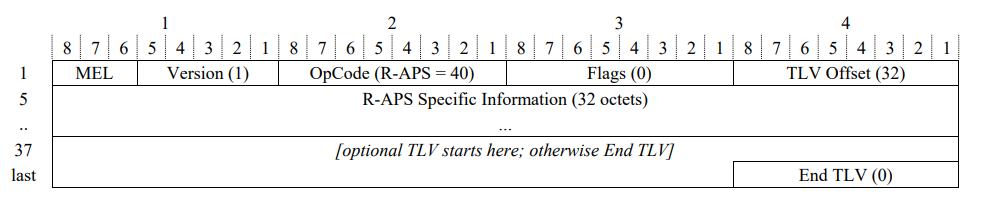


图5 RAPS PDU报文基本格式

RAPS PDU报文字段含义：

**MEL** (3bit)：

标识维护实例等级。ERPS一般会与CFM/Y1731配合使用。

**Version** (5bit)：

ERPS协议版本，0x00表示v1版本，0x01表示v2版本。

**OpCode** (8bit)：

固定值0x28，标识该PDU是RAPS PDU。

**Flags** (8bit)：

固定值0x00，该字段在接收的过程中会被忽略。

**TLV(type-length-value) Offset** (8bit)：

固定值0x20，表示报文中的TLV从该字段之后偏移32个字节后开始。

**R-APS Specific Information** (32 x 8bit)：

该字段携带RAPS环信息，是RAPS PDU的核心字段。对于该字段，v1版本和v2版本在某些子字段的定义上存在一定的差异。图6描述v1版本该字段具体包含的各子字段，图7描述v2版本该字段具体包含的各子字段。

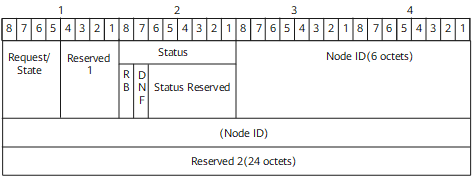


图6 v1版本RAPS Specific Information格式

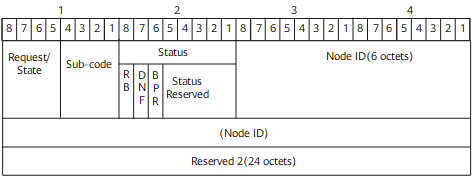


图7 v2版本RAPS Specific Information格式

R-APS Specific Information各子字段含义：

**Request/State** (4bit)：

标识该信息是请求信息或当前状态信息：

* 1101：FS（Forced Switch） RAPS
* 1110：Event报文
* 1011：SF（Signal Failed） RAPS
* 0111：MS（Manual Switch） RAPS
* 0000：NR（No Request） RAPS
* 其他：保留字段

**Reserved 1/Sub-code** (4bit)：

对于v1，该字段是“Reserved 1”，表示保留字段，留作以后报文应答或是保护类型标识。

对于v2，该字段是“Sub-code”：

* 当“Request/State”字段的取值为1110时，该字段为0000表示FDB表项刷新请求。
* 当“Request/State”字段取其他值时，该字段的取值为全0，为保留字段，且在接收过程中会被忽略。

**Status** (8bit)：

标识状态信息：

* RB（RPL Blocked，1位）：RB=1标识RPL链路被阻塞；RB=0标识RPL链路解除阻塞。非RPL Owner设备在发送RPL PDU时将该字段置为0。
* DNF（Do Not Flush，1位）：DNF=1标识收到当前信息不刷新FDB缓冲区；DNF=0标识收到当前信息可刷新FDB缓冲区。
* BPR（Blocked Port Reference，1位）：阻塞端口标志位，该字段为0表示阻塞第一个端口，该字段为1表示阻塞第二个端口。

只有v2版本支持该BPR字段。

* Status Reserved：保留字段。在发送过程中，此字段全置为0，且在接收的过程中会被忽略。该字段在v1版本有6位，在v2版本有5位。

**Node ID** (6 x 8bit)：

标识RAPS环节点设备的MAC地址，该字段属于提示信息，不影响RAPS环的保护切换操作。

**Reserved 2** (24 x 8bit)：

保留字段，在发送过程中，此字段全置为0，且在接收的过程中会被忽略。

**基本原理**

**单环基本原理**

**链路正常**

如图8所示，由SwitchA～SwitchE组成的环路上各设备通信正常。

为防止环路产生，ERPS首先会阻塞RPL owner端口，如果配置了RPL neighbour端口，该端口同样会被阻塞，其他端口可以正常转发业务流量。

ERPS环上的RPL owner端口以5s的时间间隔为周期向环中其他节点发送NRRB RAPS报文，表示ERPS环当前链路一切正常。

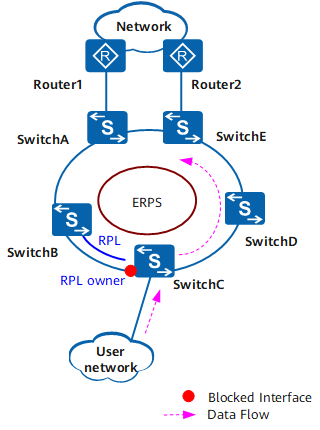


图8 部署ERPS的单环组网图（链路正常）

SwitchC：

ethernet ring-protection 1 east gigaethernet 1/1/2 west gigaethernet 1/1/3 node-type rpl-owner rpl east block-vlanlist 1-10

SwitchB：

ethernet ring-protection 1 east gigaethernet 1/2/2 west gigaethernet 1/2/3 node-type rpl-neighbour rpl west block-vlanlist 1-10

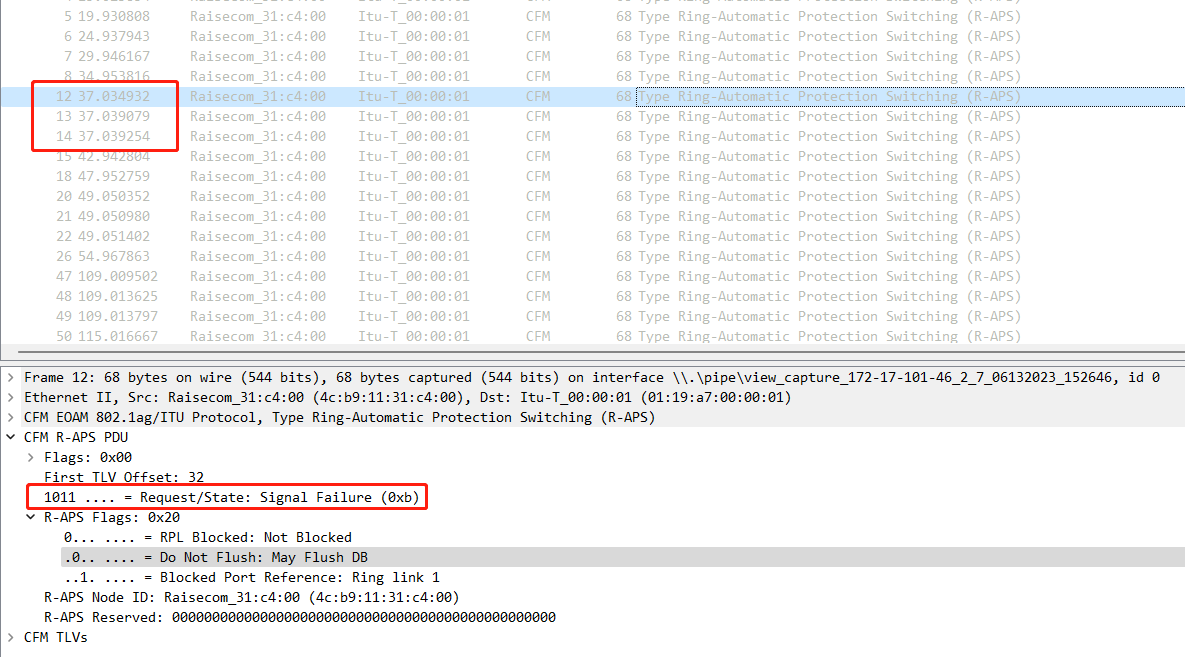
SwitchA：

ethernet ring-protection 1 east gigaethernet 1/1/2 west gigaethernet 1/1/3 block-vlanlist 1-10

**链路故障**

如图9所示，当SwitchD和SwitchE之间的链路发生故障时，ERPS协议启动保护倒换机制，将故障链路的两端端口阻塞，然后放开RPL owner端口和RPL neighbour端口，这两个端口重新恢复用户流量的接收和发送，从而保证了流量不中断。具体处理过程如下：

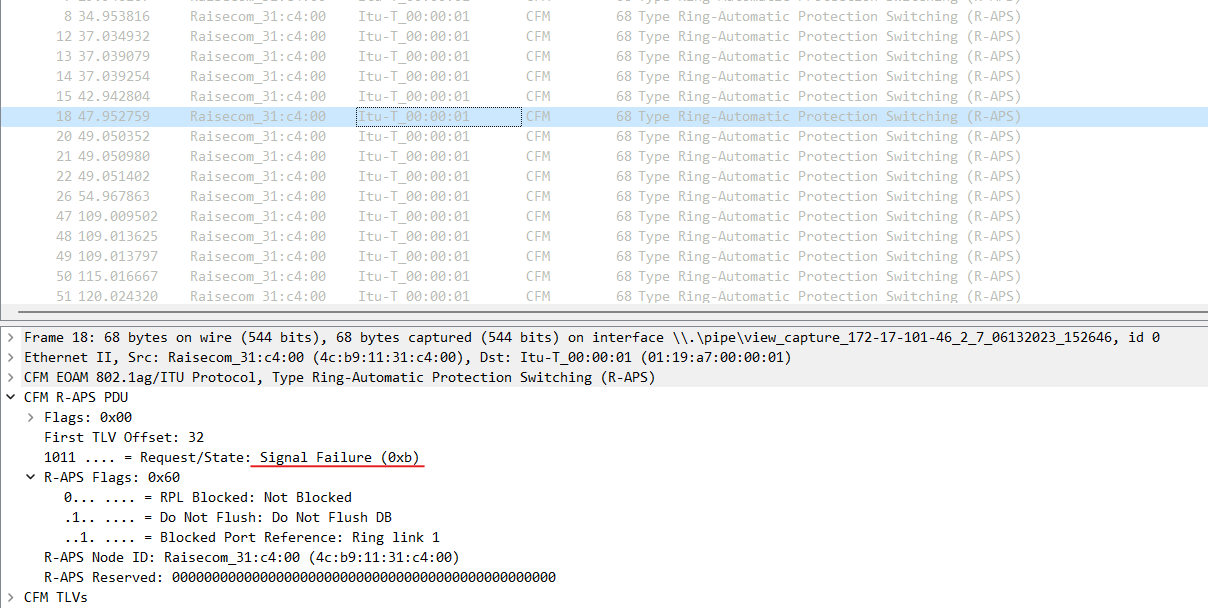
1. SwitchD和SwitchE检测到链路故障，将故障链路上的端口阻塞，并刷新本设备的FDB表项。
2. 然后，SwitchD和SwitchE向外发送携带本地端口链路故障消息的SF RAPS报文，即一旦感知到链路故障，SwitchD和SwitchE会连续发送3个相同的RAPS报文，然后以5s的间隔持续稳定发送。设备在环上的两个端口都往外发送RAPS报文。下图是SwitchC上另一个非RPL Owner端口上的收发包：



从收发包可以看到以毫秒为单位内连续收到三个SF报文，并且RAPS报文的Status标志位表示会清空FDB表项，将非RPL的端口block。

1. 其他设备收到SwitchD和SwitchE发送的SF RAPS报文后，都刷新本设备的FDB表项。当SwitchC设备（RPL owner端口所在设备）收到该RAPS报文后，放开RPL owner端口，并刷新自己的FDB表项。同样，当SwitchB设备（RPL neighbour端口所在设备）收到RAPS报文后，放开RPL neighbour端口，并刷新自己的FDB表项。

后面5秒间隔发送的SF RAPS报文：



开始的连续3个SF RAPS报文刷新mac表后，环网进入保护状态，已经稳定，这个时候5秒间隔发送的报文DNF标志位置位，不再刷新mac表。RPL Blocked标志位清空，表示RPL端口放开。

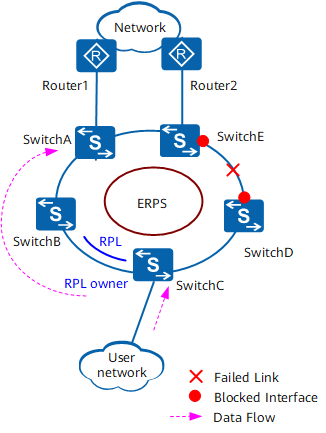


图9 部署ERPS的单环组网图（链路故障）

**链路恢复**

链路恢复正常后，如果ERPS环配置的是回切模式，RPL owner端口所在设备会重新阻塞RPL链路上的流量，故障链路重新被用来完成用户流量的传送。如果ERPS环配置的是非回切模式，阻塞链路还保持在原来的故障链路上，不会重新切回到RPL上。以回切模式为例，具体恢复过程如下：

1. 当SwitchD和SwitchE之间的链路恢复后，SwitchD和SwitchE为了防止收到过期的RAPS协议报文，分别启动Guard Timer定时器，在该定时器超时前不接收其他RAPS协议报文。同时SwitchD和SwitchE会向外发送NR RAPS报文。
2. 当RPL owner端口所在设备（SwitchC）收到NR RAPS报文后，启动WTR Timer定时器。当该定时器超时后，RPL owner端口被阻塞，同时向外发送NRRB RAPS报文。
3. 当SwitchD和SwitchE收到SwitchC发送的NRRB RAPS协议报文后，将自己设备上原来阻塞的端口放开，停止发送NR RAPS协议报文并且完成FDB表项的刷新。其他设备收到SwitchC发送的NRRB RAPS协议报文后，也完成FDB表项的刷新。

**多环基本原理**

v1只支持单环组网，v2不仅支持基本的单环组网，还支持相交环等多环组网方式。

在ERPS相交环组网中，既有主环也有子环。子环RAPS报文传输方式根据子环的RAPS报文是否会进入主环分为虚通道和非虚通道两种方式。

本节主要以相交环组网，子环RAPS报文传输方式为非虚通道方式为例，按照链路正常->链路故障->链路恢复的过程，介绍多环组网下ERPS的实现原理。

**链路正常**

如图10所示，由SwitchA～SwitchE组成的ERPS环为主环，SwitchB、SwitchC和SwitchF组成的ERPS环为子环1，SwitchC、SwitchD和SwitchG组成的ERPS环为子环2，各个环路都通信正常。

1. 为防止环路产生，ERPS的三个环分别阻塞自己的RPL owner端口。
2. 主环的RPL owner端口以5s的时间间隔为周期向主环其他节点发送NRRB RAPS报文。同样，子环1和子环2的RPL owner端口也以5s的时间间隔为周期向自己环中其他节点发送NR RAPS报文。主环的协议报文只在主环上传输，两个子环的报文会在相交节点终结，也不会进入主环。

PC1和上层网络之间的流量走向为PC1<->SwitchF<->SwitchB<->SwitchA<->Router1，PC2和上层网络之间的流量走向为PC2<->SwitchG<->SwitchD<->SwitchE<->Router2。

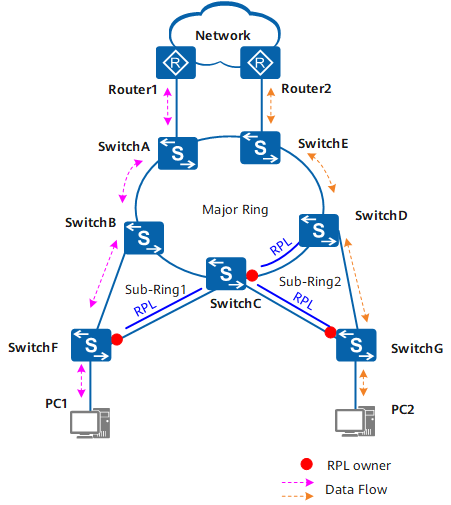


图10 部署ERPS的多环组网图（链路正常）

**链路故障**

如图11所示，当SwitchD和SwitchG之间的链路发生故障时，ERPS协议启动保护倒换机制，将故障链路的两端端口阻塞，子环2会将RPL owner端口放开，重新恢复用户流量的接收和发送，PC1的用户流量不受影响，为了保证PC2的下行流量不中断，需要在相交节点SwitchC和SwitchD将子环2的拓扑变化信息通告到主环。最终PC2和上层网络之间的流量走向为PC2<->SwitchG<->SwitchC<->SwitchB<->SwitchA<->SwitchE<->Router2。具体处理过程如下：

1. SwitchD和SwitchG检测到链路故障，将故障链路上的端口阻塞，并刷新本设备的FDB表项。
2. 然后，SwitchG在子环2内部发送携带本地端口链路故障消息的SF RAPS报文，即一旦感知到链路故障，SwitchG会连续发送3个相同的SF RAPS报文，然后以5s的间隔持续稳定发送SF RAPS报文。
3. 然后，SwitchG设备（RPL owner端口所在设备）放开RPL owner端口，并刷新自己的FDB表项。
4. 和主环的相交节点SwitchC收到该RAPS报文后，会刷新自己的FDB表项。而SwitchC和SwitchD在感知到这个网络拓扑变化后，会在主环内发送Event报文，通告子环2的网络拓扑发生变化。
5. 其他主环节点收到Event报文后，会刷新自己的FDB表项。

最终保证PC2的用户流量完成快速切换。

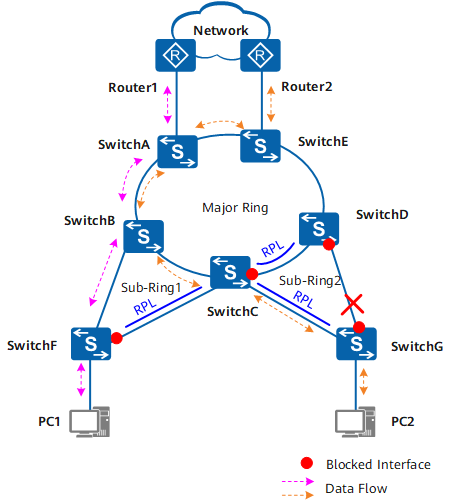


图11 部署ERPS的多环组网图（链路故障）

**链路恢复**

链路恢复后，如果各个ERPS环配置的是回切模式，RPL owner端口所在设备会重新阻塞RPL链路上的流量，故障链路重新被用来完成报文的传送。如果ERPS环配置的是非回切模式，阻塞链路还保持在原来的故障链路上，不会重新切回到RPL上。以回切模式为例，具体恢复过程如下：

1. 当SwitchD和SwitchG之间的链路恢复后，SwitchD和SwitchG为了防止收到过期的RAPS协议报文，分别启动Guard Timer，在该定时器超时前不接收其他RAPS协议报文。同时SwitchD和SwitchG会在子环2内部发送NR RAPS报文。
2. SwitchG（RPL owner端口所在设备）会启动WTR Timer。当WTR Timer超时后，SwitchG会阻塞RPL owner端口，放开故障恢复链路的端口，同时向外发送NRRB RAPS报文。
3. 当SwitchD收到SwitchG发送的NRRB协议报文后，将自己设备上原来阻塞的端口放开，停止发送NR RAPS报文并且完成FDB表项的刷新。SwitchC收到SwitchG发送的NRRB RAPS报文后，也完成FDB表项的刷新。
4. 相交节点SwitchC和SwitchD完成自身的FDB表项刷新以后，会在主环内发送Event报文，通告子环2的拓扑变化情况。
5. 其他主环节点收到Event报文后，会刷新自己的FDB表项。

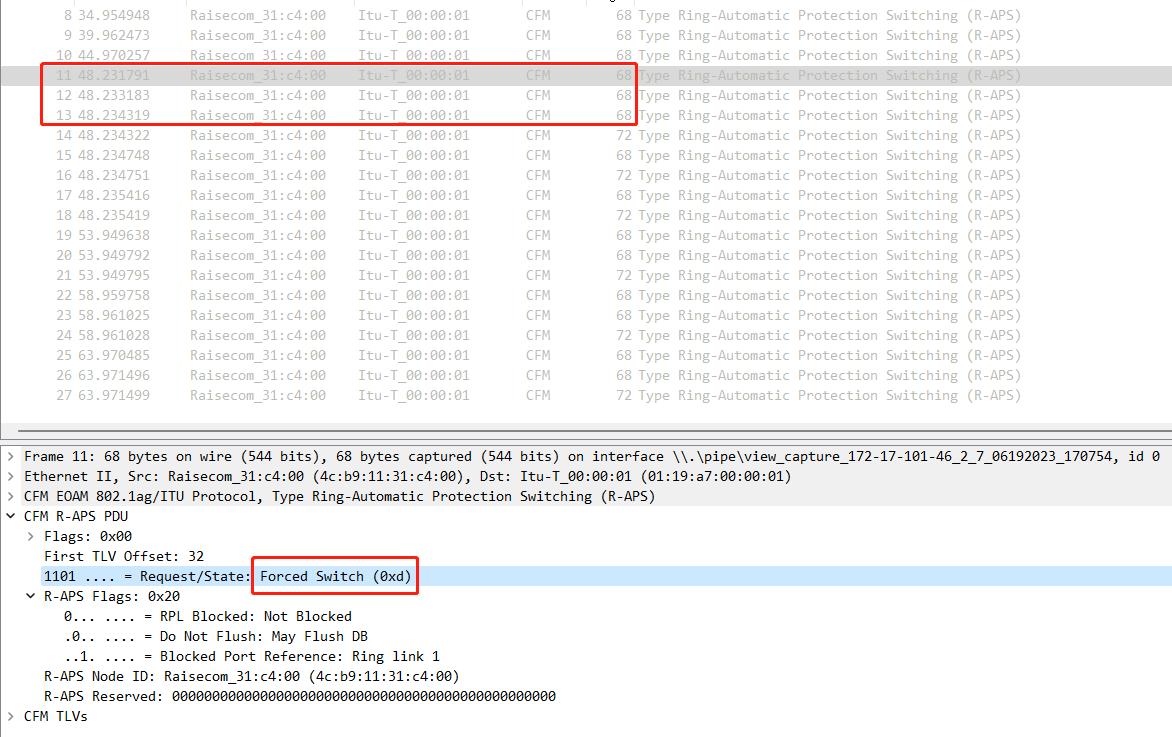
最终PC2的用户流量又会切换到如图10所示的走向。

**保护倒换**

**强制切换**

如图12所示，当由SwitchA～SwitchE组成的环路上各设备通信正常时，在SwitchE设备与SwitchD相连的端口上执行强制切换将端口阻塞，RPL owner端口和RPL neighbour端口同样会被放开，这两个端口重新恢复用户流量的接收和发送，从而保证了流量不中断。具体处理过程如下：

1. SwitchE与SwitchD相连的端口被强制阻塞后，刷新本设备的FDB表项。
2. SwitchE向外发送FS RAPS报文，即一旦端口阻塞，SwitchE会连续发送3个相同的FS RAPS报文，然后以5s的间隔持续稳定发送FS RAPS报文。



1. 其他设备收到SwitchE发送的FS RAPS报文后，都刷新本设备的FDB表项。当SwitchC设备（RPL owner端口所在设备）收到该RAPS报文后，放开RPL owner端口，并刷新自己的FDB表项。同样，当SwitchB设备（RPL neighbour端口所在设备）收到RAPS报文后，放开RPL neighbour端口，并刷新自己的FDB表项。

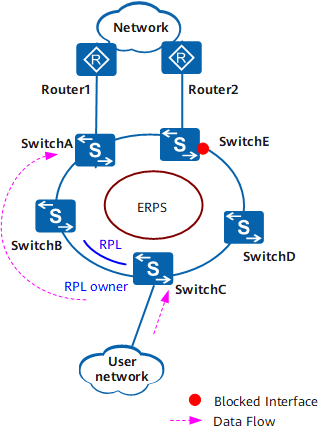


图12 部署ERPS的二层环网图（链路故障）

**清除**

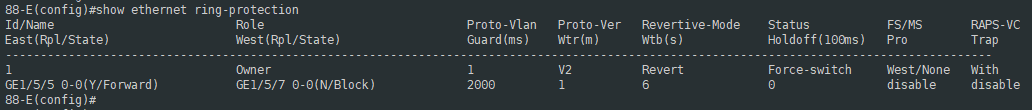
当在SwitchE设备上执行清除操作后，被强制阻塞的端口会发送NR RAPS报文给环上其他端口。

如果ERPS环配置的是回切模式，RPL owner端口在收到NR RAPS报文后会启动WTB Timer，在WTB定时器超时后，强制切换的操作会被清除。此时，RPL owner端口会被重新阻塞，被强制阻塞的端口会被放开。若在WTB Timer超时前，在SwitchC（RPL owner端口所在设备）上执行清除操作，RPL owner端口会马上被阻塞，被强制阻塞的端口被放开。

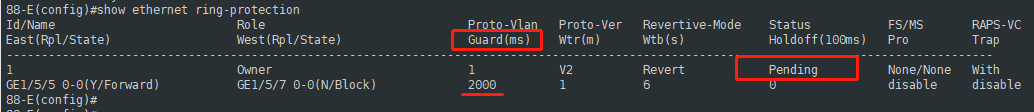
如果ERPS环配置的是非回切模式而又希望RPL owner端口被重新阻塞，可以在SwitchC（RPL owner端口所在设备）上执行清除操作。

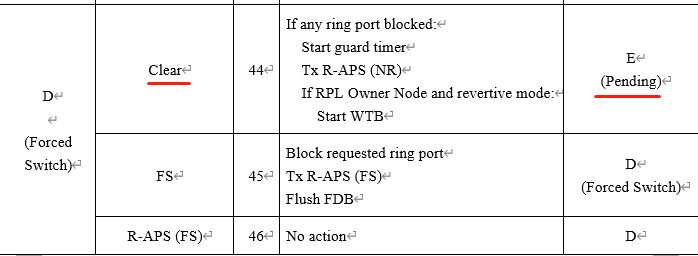
**示例：**

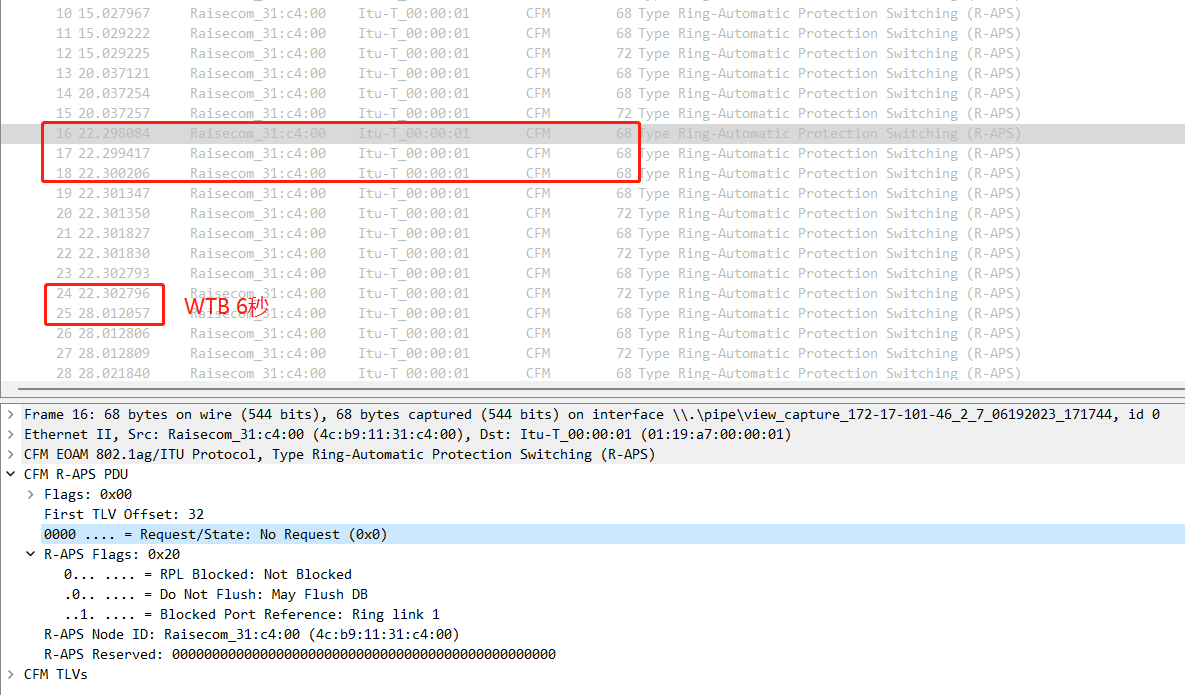
在RPL Owner节点设备上配置强制切换，使得RPL Owner接口放开，阻塞另一个接口，环状态迁移到Force-switch：



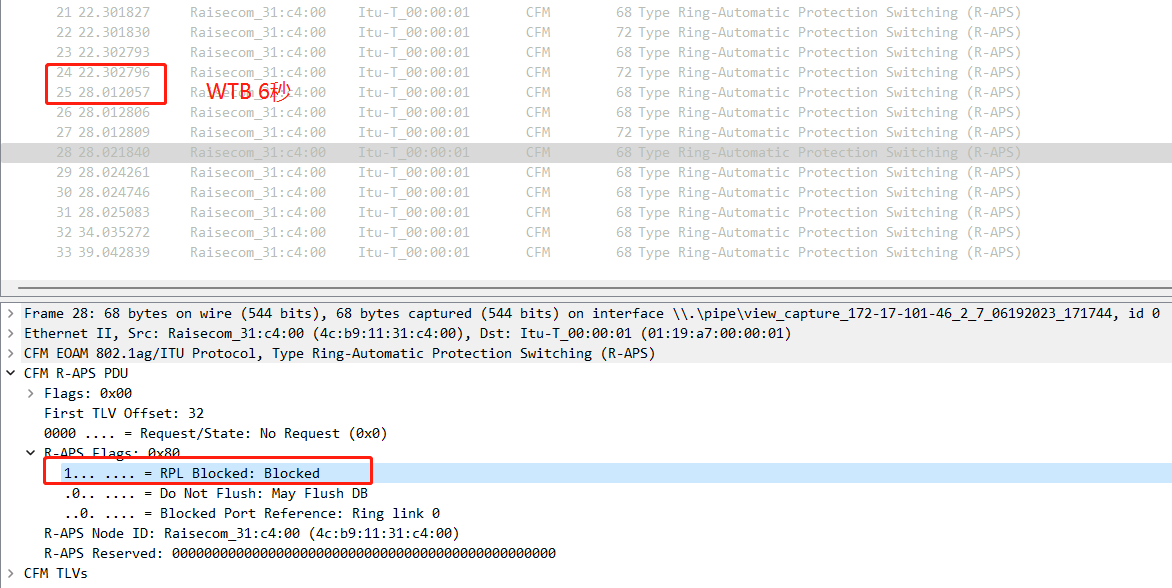
配置清除命令后，根据协议状态机规定，强制倒换状态下收到清除命令时会启动Guard Timer（配置默认2秒），Guard Timer超时后向外连续发送NR RAPS报文，同时状态迁移到Pending：

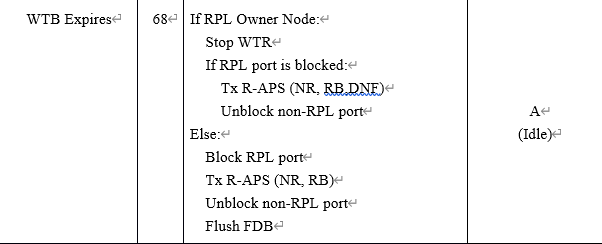




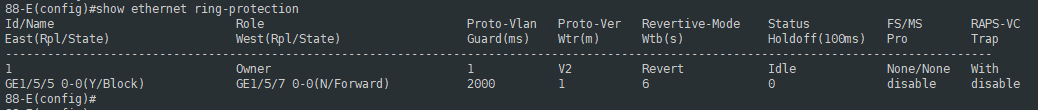


状态迁移到Pending后，因为是在RPL Owner节点上进行配置，所以同时启动WTB定时器，设备默认6秒。Pending状态下的WTB定时器超时后，RPL Owner设备接口往外发送NR RB RAPS报文：





再次查看环网状态，状态迁移到Idle：



注意，上述示例是在环网处于Idle状态，RPL正常block时，强制放开RPL，阻塞另一个普通端口时，该普通端口的收发包情况。如果命令行强制倒换时指定的端口本来就是block状态，比如上例中指定的本来是RPL Owner节点上的普通端口，换成指定本来就是block状态的RPL Owner端口时，普通端口的收发包与上述示例抓包结果是不同的，因为状态机迁移过程不同，详细参考状态机迁移表。

**手工切换**

对ERPS环上端口执行手工切换阻塞操作的流程和强制切换类似，区别在于如果环的状态不是Idle或者Pending时，手工切换操作将不发挥作用。

**多实例**

在普通的配置了ERPS的组网中，一个物理环上只能配置一个ERPS环，也只能指定一个阻塞点。当ERPS环处于正常状态时，阻塞端口会阻止所有的业务数据通过。这样，所有业务数据在ERPS环上只能通过一条路径传输，阻塞端口另一侧的链路空闲，造成带宽浪费。

如图13所示，当只配置了ERPS Ring1时，端口Interface1被阻塞，数据流量按照Data Flow1所示的路线进行转发，而此时SwitchC-SwitchD-SwitchE所在的链路处于空闲状态。

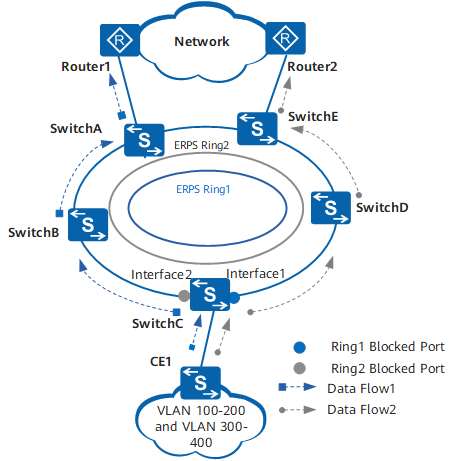
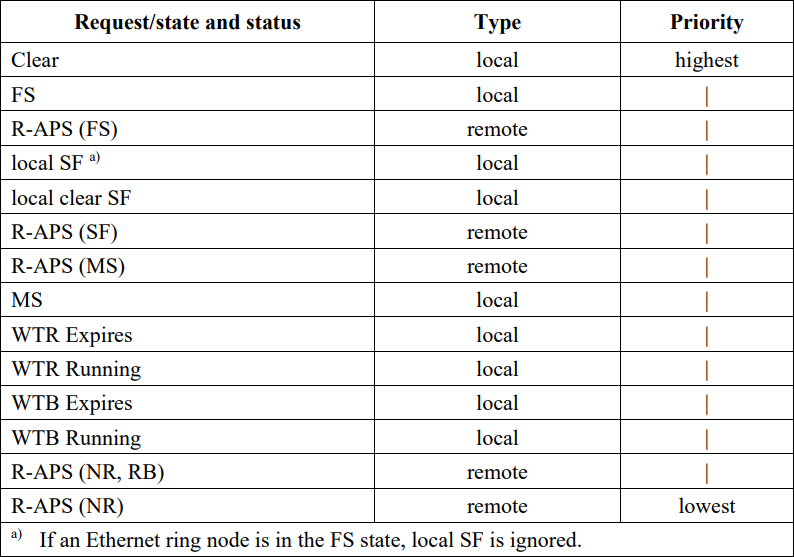


图13 ERPS多实例组网图

为了提高链路的利用率，ERPS多实例支持在一个物理环路上配置两个逻辑ERPS环路。每一个ERPS环上所有设备、端口角色、控制VLAN等都遵循基本的ERPS原则。相应的，一个物理环路上有两个阻塞端口，各个阻塞端口分别检测物理环路的完整性，并相应的阻塞或放开，彼此互不影响。每个ERPS环上均需要配置一个保护实例，每个保护实例代表一个VLAN范围。不同的ERPS环计算出的拓扑仅对该ERPS环有效，不影响其他ERPS环。通过给每个ERPS配置不同的保护实例，各阻塞端口只对本ERPS环所保护的VLAN有效。属于不同VLAN的数据流量就可以通过不同的路径传输，从而实现流量的负载分担和链路备份，最大化利用了链路资源。

如图13所示，我们可以在SwitchA、SwitchB、SwitchC、SwitchD和SwitchE组成的物理环路上配置ERPS Ring1和ERPS Ring2。ERPS Ring1的阻塞端口为Interface1，其保护实例对应的VLAN范围是VLAN100～VLAN200。ERPS Ring2的阻塞端口为Interface2，其保护实例对应的VLAN范围是VLAN300～VLAN400。完成相关配置后，数据VLAN100～VLAN200会按照Data Flow1所示的路线进行转发，数据VLAN300～VLAN400会按照Data Flow2所示的路线进行转发，从而实现了负载分担，提高了链路的利用率。

**RAPS消息优先级：**



1、对于从多个来源发来的消息，只有当前最高优先级的消息会通过该模块。

2、每次处理完当前消息后，节点的最高优先级都会重新评估。

3、对SF、FS、MS等请求，会一直在priority logic 中保持，直到有更高优先级的请求到来。

**状态机迁移表：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Inputs** |  | **Outputs** |  |
| **Node state** | **Top priority request** | Row | **Actions** | **Next node state** |
| **-** | State machine initialization | 1 | Stop guard timer  Stop WTR timer  Stop WTB timer  If RPL Owner Node:  Block RPL port  Unblock non-RPL port  Tx R-APS (NR)  If revertive:  Start WTR timer  Else if RPL Neighbour Node:  Block RPL Port  Unblock non-RPL port  Tx R-APS (NR)  Else:  Block one ring port  Unblock other ring port  Tx R-APS (NR) | E  (Pending) |
| A  (Idle) | Clear | 2 | No action | A |
| FS | 3 | If requested ring port is already blocked:  Tx R-APS (FS,DNF)  Unblock non-requested ring port  Else:  Block requested ring port  Tx R-APS (FS)  Unblock non-requested ring port  Flush FDB | D  (Forced Switch) |
| R-APS (FS) | 4 | Unblock ring ports  Stop Tx R-APS | D  (Forced Switch) |
| local SF | 5 | If failed ring port is already blocked:  Tx R-APS (SF,DNF)  Unblock non-failed ring port  Else:  Block failed ring port  Tx R-APS (SF)  Unblock non-failed ring port  Flush FDB | B  (Protection) |
| local clear SF | 6 | No action | A |
| R-APS (SF) | 7 | Unblock non-failed ring port  Stop Tx R-APS | B  (Protection) |
| R-APS (MS) | 8 | Unblock non-failed ring port  Stop Tx R-APS | C  (Manual Switch) |
| MS | 9 | If requested ring port is already blocked:  Tx R-APS (MS,DNF)  Unblock non-requested ring port  Else:  Block requested ring port  Tx R-APS (MS)  Unblock non-requested ring port  Flush FDB | C  (Manual Switch) |
| WTR Expires | 10 | No action | A |
| WTR Running | 11 | No action | A |
| WTB Expires | 12 | No action | A |
| WTB Running | 13 | No action | A |
| R-APS (NR, RB) | 14 | Unblock non-RPL port  If Not RPL Owner Node:  Stop Tx R-APS | A |
| R-APS (NR) | 15 | If neither RPL Owner Node nor RPL Neighbour Node, and remote Node ID is higher than own Node ID:  Unblock non-failed ring port  Stop Tx R-APS | A |
| B  (Protection) | Clear | 16 | No action | B  (Protection) |
| FS | 17 | If requested ring port is already blocked:  Tx R-APS (FS,DNF)  Unblock non-requested ring port  Else:  Block requested ring port  Tx R-APS (FS)  Unblock non-requested ring port  Flush FDB | D  (Forced Switch) |
| R-APS (FS) | 18 | Unblock ring ports  Stop Tx R-APS | D  (Forced Switch) |
| local SF | 19 | If failed ring port is already blocked:  Tx R-APS (SF,DNF)  Unblock non-failed ring port  Else:  Block failed ring port  Tx R-APS (SF)  Unblock non-failed ring port  Flush FDB | B  (Protection) |
| local clear SF | 20 | Start guard timer  Tx R-APS (NR)  If RPL Owner Node and revertive mode:  Start WTR | E  (Pending) |
| R-APS (SF) | 21 | No action | B  (Protection) |
| R-APS (MS) | 22 | No action | B |
| MS | 23 | No action | B |
| WTR Expires | 24 | No action | B |
| WTR Running | 25 | No action | B |
| WTB Expires | 26 | No action | B |
| WTB Running | 27 | No action | B |
| R-APS (NR, RB) | 28 | No action | E |
| R-APS (NR) | 29 | If RPL Owner Node and revertive mode:  Start WTR | E  (Pending) |
| C  (Manual Switch) | Clear | 30 | If any ring port blocked:  Start guard timer  Tx R-APS (NR)  If RPL Owner Node and revertive mode:  Start WTB | E  (Pending) |
| FS | 31 | If requested ring port is already blocked:  Tx R-APS (FS,DNF)  Unblock non-requested ring port  Else:  Block requested ring port  Tx R-APS (FS)  Unblock non-requested ring port  Flush FDB | D  (Forced Switch) |
| R-APS (FS) | 32 | Unblock ring ports  Stop Tx R-APS | D  (Forced Switch) |
| local SF | 33 | If failed ring port is already blocked:  Tx R-APS (SF,DNF)  Unblock non-failed ring port  Else:  Block failed ring port  Tx R-APS (SF)  Unblock non-failed ring port  Flush FDB | B  (Protection) |
| local clear SF | 34 | No action | C  (Manual Switch) |
| R-APS (SF) | 35 | Unblock non-failed ring port  Stop Tx R-APS | B  (Protection) |
| R-APS (MS) | 36 | If any ring port blocked:  Start guard timer  Tx R-APS (NR)  If RPL Owner Node and revertive mode:  Start WTB | E  (Pending) |
| MS | 37 | No action | C  (Manual Switch) |
| WTR Expires | 38 | No action | C |
| WTR Running | 39 | No action | C |
| WTB Expires | 40 | No action | C |
| WTB Running | 41 | No action | C |
| R-APS (NR, RB) | 42 | No action | E |
| R-APS (NR) | 43 | If RPL Owner Node and revertive mode:  Start WTB | E  (Pending) |
| D  (Forced Switch) | Clear | 44 | If any ring port blocked:  Start guard timer  Tx R-APS (NR)  If RPL Owner Node and revertive mode:  Start WTB | E  (Pending) |
| FS | 45 | Block requested ring port  Tx R-APS (FS)  Flush FDB | D  (Forced Switch) |
| R-APS (FS) | 46 | No action | D |
| local SF | 47 | No action | D |
| local clear SF | 48 | No action | D |
| R-APS (SF) | 49 | No action | D |
| R-APS (MS) | 50 | No action | D |
| MS | 51 | No action | D |
| WTR Expires | 52 | No action | D |
| WTR Running | 53 | No action | D |
| WTB Expires | 54 | No action | D |
| WTB Running | 55 | No action | D |
| R-APS (NR, RB) | 56 | No action | E |
| R-APS (NR) | 57 | If RPL Owner Node and revertive mode:  Start WTB | E  (Pending) |
| E  (Pending) | Clear | 58 | If RPL Owner Node:  Stop WTR  Stop WTB  If RPL port is blocked:  Tx R-APS (NR, RB,DNF)  Unblock non-RPL port  Else:  Block RPL port  Tx R-APS (NR, RB)  Unblock non-RPL port  Flush FDB | A  (Idle) |
| FS | 59 | If requested ring port is already blocked:  Tx R-APS (FS,DNF)  Unblock non-requested ring port  Else:  Block requested ring port  Tx R-APS (FS)  Unblock non-requested ring port  Flush FDB  If RPL Owner Node:  Stop WTR  Stop WTB | D  (Forced Switch) |
| R-APS (FS) | 60 | Unblock ring ports  Stop Tx R-APS  If RPL Owner Node:  Stop WTR  Stop WTB | D  (Forced Switch) |
| local SF | 61 | If failed ring port is already blocked:  Tx R-APS (SF,DNF)  Unblock non-failed ring port  Else:  Block failed ring port  Tx R-APS (SF)  Unblock non-failed ring port  Flush FDB  If RPL Owner Node:  Stop WTR  Stop WTB | B  (Protection) |
| local clear SF | 62 | No action | E  (Pending) |
| R-APS (SF) | 63 | Unblock non-failed ring port  Stop Tx R-APS  If RPL Owner Node:  Stop WTR  Stop WTB | B  (Protection) |
| R-APS (MS) | 64 | Unblock non-failed ring port  Stop Tx R-APS  If RPL Owner Node:  Stop WTR  Stop WTB | C  (Manual Switch) |
| MS | 65 | If RPL Owner Node:  Stop WTR  Stop WTB  If requested ring port is already blocked:  Tx R-APS (MS,DNF)  Unblock non-requested ring port  Else:  Block requested ring port  Tx R-APS (MS)  Unblock non-requested ring port  Flush FDB | C  (Manual Switch) |
| WTR Expires | 66 | If RPL Owner Node:  Stop WTB  If RPL port is blocked:  Tx R-APS (NR, RB,DNF)  Unblock non-RPL port  Else :  Block RPL port  Tx R-APS (NR, RB)  Unblock non-RPL port  Flush FDB | A  (Idle) |
| WTR Running | 67 | No action | E  (Pending) |
| WTB Expires | 68 | If RPL Owner Node:  Stop WTR  If RPL port is blocked:  Tx R-APS (NR, RB,DNF)  Unblock non-RPL port  Else:  Block RPL port  Tx R-APS (NR, RB)  Unblock non-RPL port  Flush FDB | A  (Idle) |
| WTB Running | 69 | No action | E |
| R-APS (NR, RB) | 70 | If RPL Owner Node:  Stop WTR  Stop WTB  If neither RPL Owner Node nor RPL Neighbour Node:  Unblock ring ports  Stop Tx R-APS  If RPL Neighbour Node:  Block RPL port  Unblock non-RPL port  Stop Tx R-APS | A  (Idle) |
| R-APS (NR) | 71 | If remote Node ID is higher than own Node ID:  Unblock non-failed ring port  Stop Tx R-APS | E  (Pending) |

表中各个命令解释：

Block requested ring port ：使被请求的端口变为 blocking状态。

Unblock non-requested ring port ：使未被请求的环路端口变为 forwarding状态。

Block failed port ：使报告SF事件的端口变为 blocking状态。

Unblock non-failed port ：使所有在环上且不在 SF状态的端口均变为forwarding状态。

Block RPL port ：使RPL端口变为 blocking状态。

Unblock non-RPL port ：使非RPL的环路端口变为 forwarding状态。

Block one ring port ：使其中一个环路端口变为 blocking状态。

Unblock other ring port ：使另一个环路端口变为 forwarding状态。

Unblock both ports ：使在环路上的端口均为 forwarding状态。

Start WTR ：如果WTR未开启，开启 WTR。

Stop WTR ：如果WTR已开启，关闭 WTR。

Start WTB ：如果WTB未开启，开启 WTB。

Stop WTB ：如果WTB已开启，关闭 WTB。

Start guard timer ：开启guard timer，在此期间内，任何 R-APS报文均不能进入priority logic。

Stop guard timer ：如果guard timer开启，关闭 guard timer。

Stop Tx R-APS ：停止发送任何 R-APS报文。

Tx R-APS(msgtype, status\_bits) ：发送相应的 R-APS报文。

Flush FDB ：清空FDB表。

R-APS 消息的传输：

1、如果待传送的R-APS 命令被修改，R-APS 最开始的三帧消息会以 burst 方式尽快传送。时间间隔不大于3.33ms，之后以5s 的间隔发送。

2、R-APS 消息并不是连续发送的，可以和其他消息并行发送。

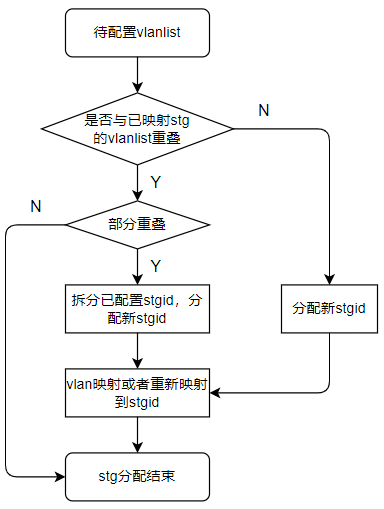
**驱动相关**

**stg分配逻辑**

stg的分配逻辑是每条命令行配置下发的vlanlist需要映射到不同的stg上，如果前后两条命令行配置的vlanlist有重叠部分，那么对应的需要将这两条命令行的vlanlist映射的stg进行拆分处理。

比如vlan 1-100命令行配置映射到stg 1，再下发vlan 50-150的配置时，因为有vlan 50-100的重叠，所以要根据vlan 1-49，50-100，101-150分段，分别映射到不同的stg。删除其中一条配置时将多余的stg释放，这时候又需要将stg资源进行合并。驱动层代码的功能就是实现stg的这个分配逻辑。

检查vlanlist，分配stgid的逻辑框图如下：

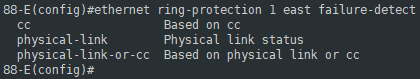


分配好硬件stgid，并将vlanlist映射到各自的stgid后，再设置接口在各个stgid中的状态。

之后保护倒换时就只需要根据端口设置端口在各个stg中的状态即可。比如，端口1/1/2在stg 1和2中设置成block，但是在stg 3-127中仍然保持forwarding状态。

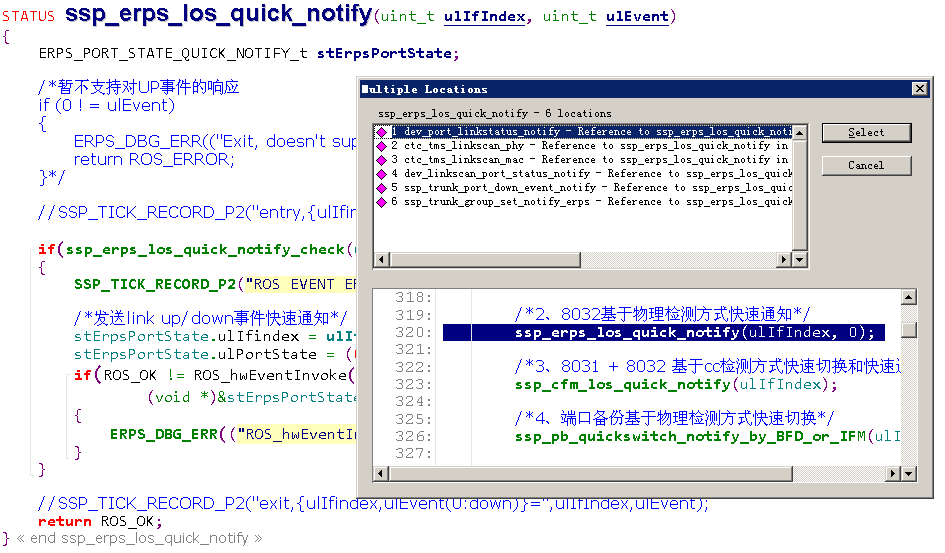
**故障检测方式**

ERPS支持CC链路检测和物理链路检测，也支持两种检测方式同时配置，只要其中任何一种方式检测到故障都上报。



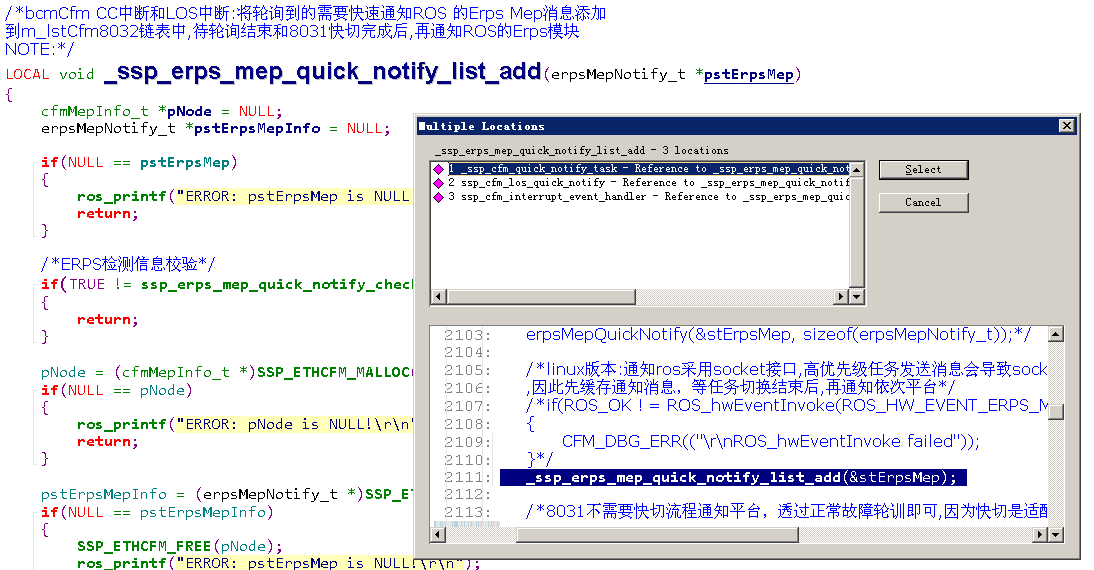
**物理链路检测**

物理口或者trunk口的link变化都是通过下面的函数上报平台，然后由平台根据状态机处理，下发保护倒换动作的：



**CC链路检测**

CC检测在驱动层是作为一个长期任务轮询的，当扫到有链路故障后，记录到软件的全局变量m\_lstCfm8032，等到本次轮询结束，elps完成快切后将链路故障上报平台的erps模块进行处理。



从调用逻辑来看，CC中断和LOS中断都会触发erps的保护倒换，在CFM模块这边是先通过上面的函数保存到全局链表m\_lstCfm8032，然后通过send函数上报平台的erps模块：



再由平台的erps模块根据状态机下发倒换处理动作到ssp的erps模块。

**trunk组的倒换处理**

平台和芯片支持trunk组的erps保护倒换，驱动需要检测到trunk组的link状态变化，来触发保护倒换。

与物理口不同，trunk口作为一个逻辑接口无法直接获取trunk口的link状态，只能根据trunk组的成员口是否正常link来判断trunk口的link状态，如果所有成员口都是down的，那么trunk口判断为down状态，只要有一个成员口link up，那么trunk口就判断为link状态，后续继续增加新的成员口不影响trunk口的link状态。

**linkdown处理**

当trunk的物理成员口down掉后，会通知trunk模块，从trunk组里删掉该物理口。如果down掉的这个端口是trunk的最后一个成员口，那么在调用sdk接口删掉这个成员口之前，将trunk口的down消息通知出去。在驱动里面是通过下面的函数处理trunk口down消息的：



函数会判断上报down消息的物理口是否是trunk的最后一个成员口：

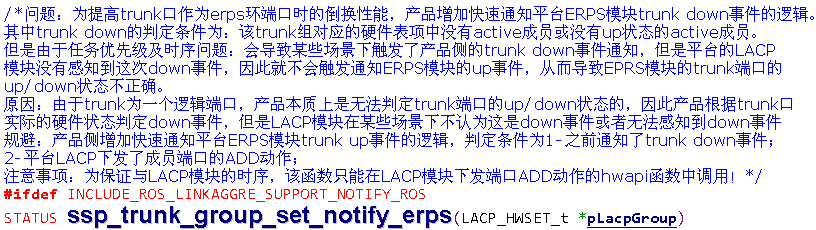


如果不是，则直接调用sdk函数从trunk里面删除这个物理口，如果是则需要通知对trunk口down消息感兴趣的模块：



**linkup处理**

trunk口的up状态判断除了需要成员口是up的，还需要根据trunk口up前的状态是否是down来判断，这样link状态发生变化了才需要通知对trunk口link状态变化有兴趣的模块：



也就是说，如果trunk组已经有成员口up时，trunk口状态此时是linkup的，再添加新的成员口时不会触发trunk的link状态变化。

函数中判断link状态是从down变化到up之后，才会通知erps模块处理：





linkdown或者linkup最后都通过函数：



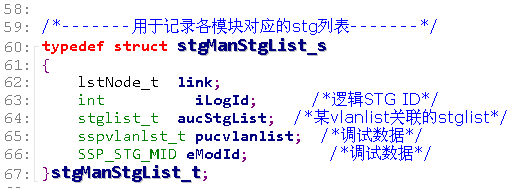
来通知erps处理。但是，这个函数不是直接在驱动层进行倒换，还是要通知到平台的erps模块，由平台处理状态机，然后下发倒换动作。

**主要数据结构**

**m\_StgManStg**



链表类型，保存的数据结构类型：



主要用来保存每一个配置命令使用到的逻辑stgid，实际分配的硬件stgid，涉及到的vlan，以及该该配置是哪个模块下发的。

各成员变量：

**iLogId**

用来保存软件配置的逻辑stgid，与实际下发芯片使用的硬件stgid不同，主要是记录配置使用的逻辑id，比如命令行配置两条erps环命令，使用的是两个不同的逻辑id，但是可能会因为有重复的vlan（第一条命令是vlan 1-100，第二条是vlan 50-150），在实际下发到芯片时会做拆分处理，出现芯片实际使用了三个stg id，vlan 1-49使用一个stg，vlan 50-100使用一个stg,vlan 101-150使用一个stg。

**aucStgList**

数组类型，使用bit位保存stgid，数组长度64，总共512个bit：



配置使用到的所有硬件stg id，比如第一条命令的vlanlist是1-100，在全局变量m\_StgManStg中新增一个链表节点，分配的硬件stgid为1，那么这个变量aucStgList的bit1置位；第二条命令使用到的vlanlist是50-150，与第一条命令的vlanlist有重叠，那么经过拆分，分配到两个新的硬件stgid为2和3，那么在全局变量m\_StgManStg中新增一个链表节点，变量aucStgList的bit2和bit3置位。

**pucvlanlist**

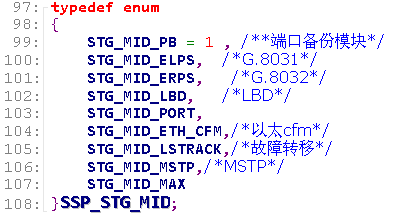
数组类型，使用bit位保存涉及到的vlan：



比如创建一条erps环配置时，需要block vlan 1-100，传入的vlanlist参数也是通过数组的bit位标识，然后处理保存到本变量。

**eModId**

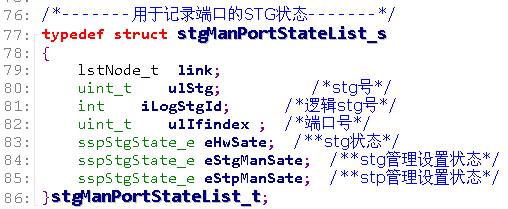
使用本stg的模块，是erps环网配置，还是stp协议的配置等：



**m\_StgManPortState**



链表类型，保存的数据结构：



主要用来保存接口对应的stgid，以及在该stg中的状态是转发还是丢弃。

**ulStg**

硬件stgid，实际下发到芯片的。

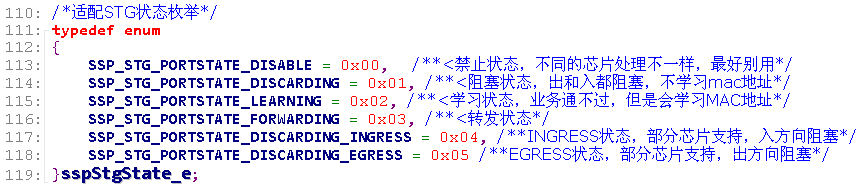
**iLogStgId**

逻辑stgid，用来保存软件配置，与实际下发芯片使用的硬件stgid不同，主要是记录配置使用的逻辑id，参考m\_StgManStg中的iLogId说明。

**eHwSate**

下发的端口硬件状态。

**eStgManSate/eStpManSate**



软件保存的端口状态，在设置端口状态时会用来做一些检查，一般在函数\_stg\_port\_state\_set中用到。

**m\_StgManPortVlanListMode**



链表类型，保存的数据结构：



主要用来保存同一个端口下，相同vlanlist使用到的stgid，避免不同配置产生冲突，互相影响。例：

同一个端口下，vlan 1-100分配了硬件stgid 1，状态为forwarding，那么该端口下需要再配置一条vlan 50-150时，会通过这个变量进行检查，vlan有重叠时返回错误。

如果不检查该条件，那么vlan 50-100的vlan可能会分配新的硬件stgid，并且设置端口在这个新的stgid上是discarding状态，导致第一条命令的vlan 50-100出现异常，原本转发的报文会突然被丢弃。

一般都是在\*\*\*\_take和\*\*\*\_give函数中使用。

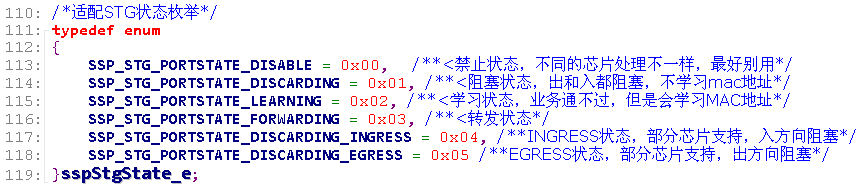
**pucvlanlist**

数组类型，使用bit位保存涉及到的vlan：



比如创建一条erps环配置时，需要block vlan 1-100，传入的vlanlist参数也是通过数组的bit位标识，然后处理保存到本变量。

**eStpManSate**

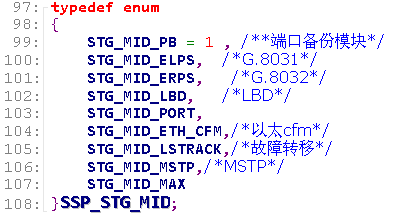


**ulStgTakeCnt/ulStpTakeCnt**

同一个模块使用同一个端口的次数，比如stp协议多次使用到同一个端口。注意如果是stg类型，一般计数只有一次，因为不能有vlan重叠，而同一个端口下不重叠的vlan会在本全局变量m\_StgManPortVlanListMode链表中新增一个节点。

**eModId/eStpModId**

使用本全局变量的模块，是erps环网配置，还是stp协议的配置等。如果是stg类型，那么eStpModId设置为0，如果是stp类型，那么eModId设置为0。



**m\_aucStgManStgUsed**



无符号字符类型，双指针全局变量，一层指针用来区分对应的芯片id，ctc设备一般最大支持两个芯片，第二层指针用来表示对应的硬件stgid（注意不是软件记录的stgid），初始化为512个stgid空间。

通过函数\_stg\_stg\_hw\_create申请分配硬件stgid，申请成功后保存到该全局变量。

通过函数\_stg\_stg\_hw\_del释放硬件stgid，该全局变量对应的数组下标置零。

**m\_aucStgManVlanToStg**



整型类型，双指针全局变量，一层指针用来区分对应的芯片id，ctc设备一般最大支持两个芯片，第二层指针用来表示vlan所在的硬件stgid（注意不是软件记录的stgid），初始化为4094个vlan空间。

初始化vlan对应的stgid为0，后续创建vlan时会设置vlan所在stgid为1，所以平台下发配置通过\_stg\_stg\_vlan\_hw\_set分配的硬件stgid保存到本全局变量，在删除配置时会将本全局变量置为1。

**m\_ulStgManStgUsed**



每次通过函数\_stg\_stg\_hw\_create分配硬件stgid后该全局变量加1，通过函数\_stg\_stg\_hw\_del释放硬件stgid时该全局变量减1。

主要作用是在分配硬件stgid时检查硬件资源是否够用。

**m\_aucStgManStgRef**



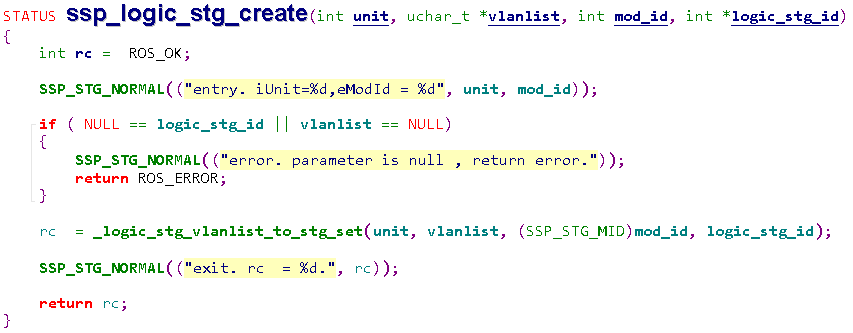
无符号字符类型，双指针全局变量，一层指针用来区分对应的芯片id，ctc设备一般最大支持两个芯片，第二层指针用来表示每个stg被复用的次数。

在删除命令释放一个stgid时，将其他与本stgid有重叠的配置合并到其他stgid。比如，两条配置，vlan 1-100配置时分配了硬件stgid 1，再配置vlan 50-150时，会将vlan 1-49拆分到stgid 1，vlan 50-100拆分到stgid 2，vlan 101-150拆分到stgid 3，然后删除第二条vlan 50-150的配置时，会删除stgid 2和3，将拆分到stgid 2的vlan 50-100合并到原先的stgid 1，这样恢复到只配第一条命令vlan 1-100时分配的stgid 1。

合并的操作主要在\_logic\_stg\_vlanlist\_to\_stg\_del函数中处理。相关全局变量m\_StgManStg。

**主要函数**

**分配stgid**



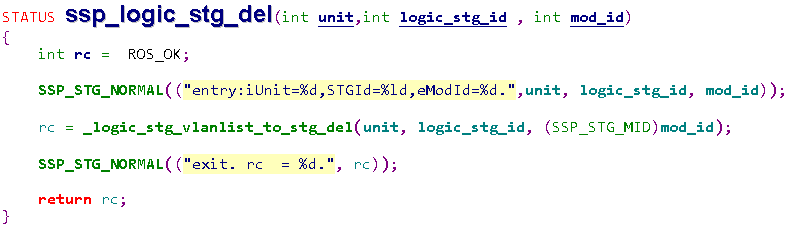
所有需要使用stg功能的模块都通过这个对外接口获取stgid。可以看到stg的分配只与vlan相关，mod\_id对应的是使用stg的模块id；logic\_stg\_id则是软件的stg索引，与实际下发芯片的硬件stgid不同，软件索引用来标识对应的命令行配置。比如创建两条ERPS环网配置，软件索引一定是不同的，但是硬件stgid则有可能相同。

其中调用的函数



会根据全局变量m\_aucStgManVlanToStg和需要的新stgid个数来判断是否需要进行拆分。

**删除stgid**

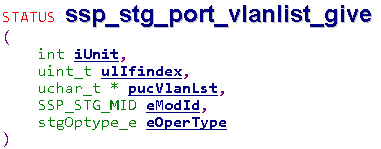


**port+vlan占用标识**



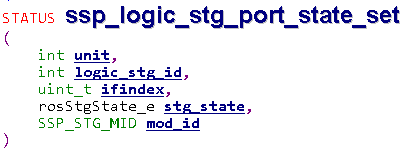
得到新的stgid后，要通过本函数设置软件变量，来标识指定端口和vlan已经被某模块使用，无法再进行配置。如果检查到已被占用，则需要通过ssp\_logic\_stg\_del删除已经创建的stgid。

**删除port+vlan占用标识**



命令行删除配置，释放stgid时清除软件的端口加vlan占用标识，下次配置时可以重新使用。

**设置端口状态**



设置端口在stg中的状态，是forwarding或者discarding。

通过软件的stg索引logic\_stg\_id在保存的全局变量m\_StgManStg中找到保存的硬件stgid，然后设置端口在这些stgid中的状态。

**debug调试命令**

在debug视图下使用的命令，一般是软件用来调试定位使用的。

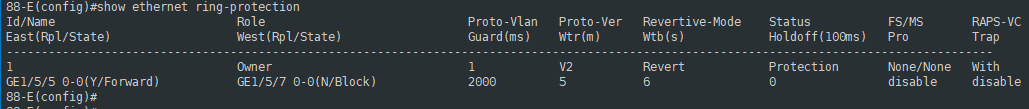
**查看驱动erps配置**

**debug ssp\_erps show**

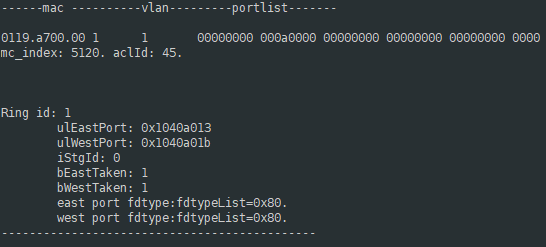
配置一个erps实例：

ethernet ring-protection 1 east gigaethernet 1/5/5 west gigaethernet 1/5/7 node-type rpl-owner rpl east block-vlanlist 1-10

设备show命令查看环配置：



定位问题时可以通过debug ssp\_erps show命令看驱动保存的这个实例配置，与上面的show命令对照，确认驱动的配置是否正确：



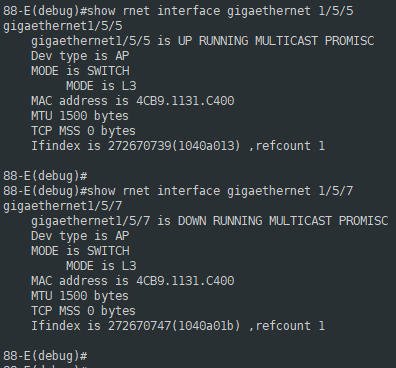
从打印信息可以看到协议的组播mac地址最后一个字节对应命令行指定的环id，使用的协议vlan是默认的vlan 1，对应的portlist是按bit位保存相关的snmp\_port值，上面使用到的两个端口snmp\_port分别是45和47，正好对应portlist显示的值。

显示的十六进制端口号可以通过命令show rnet interface gigaethernet 1/5/5查看。

**查看接口的ifindex**

**show rnet interface *interface-type interface-name***

命令在debug视图下使用：

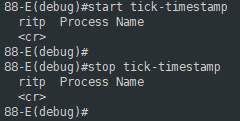


**打开/关闭时间戳记录功能**

**start tick-timestamp**

**stop tick-timestamp**

如果需要查看代码里面添加的时间戳信息，需要先打开这个开关。



**查看时间戳信息**

**show tick-timestamp [ritp]**

不带参数ritp时是查看平台和驱动所有的时间戳信息，ritp表示只查看驱动添加的时间戳信息。

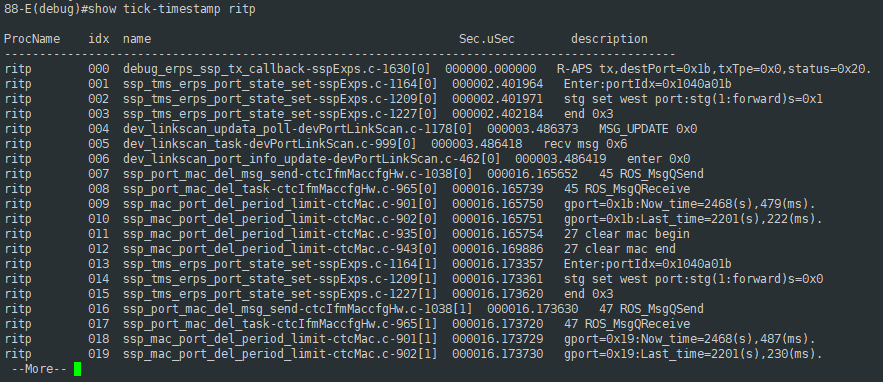
代码里面需要看哪块逻辑耗时情况时，在代码里加上时间戳信息：



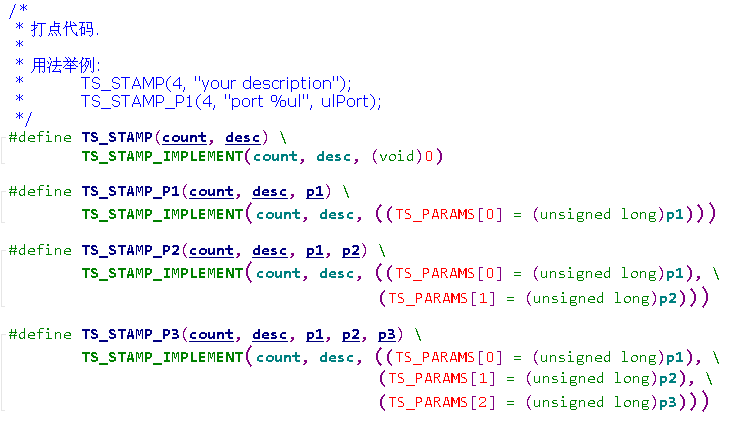
上图是erps设置端口状态加的时间戳信息，配置erps实例或者保护倒换时记录时间信息，先使用命令：

start tick-timestamp

打开时间戳记录开关，然后再使用show命令查看这两个打印：



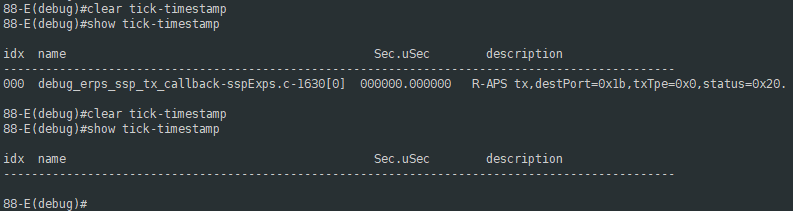
**注：**代码提供多个时间戳打点接口，上图中使用的是其中一种，其他接口如下，需要记录不同内容时选择相应的接口：



**清除时间戳信息**

**clear tick-timestamp [ritp]**

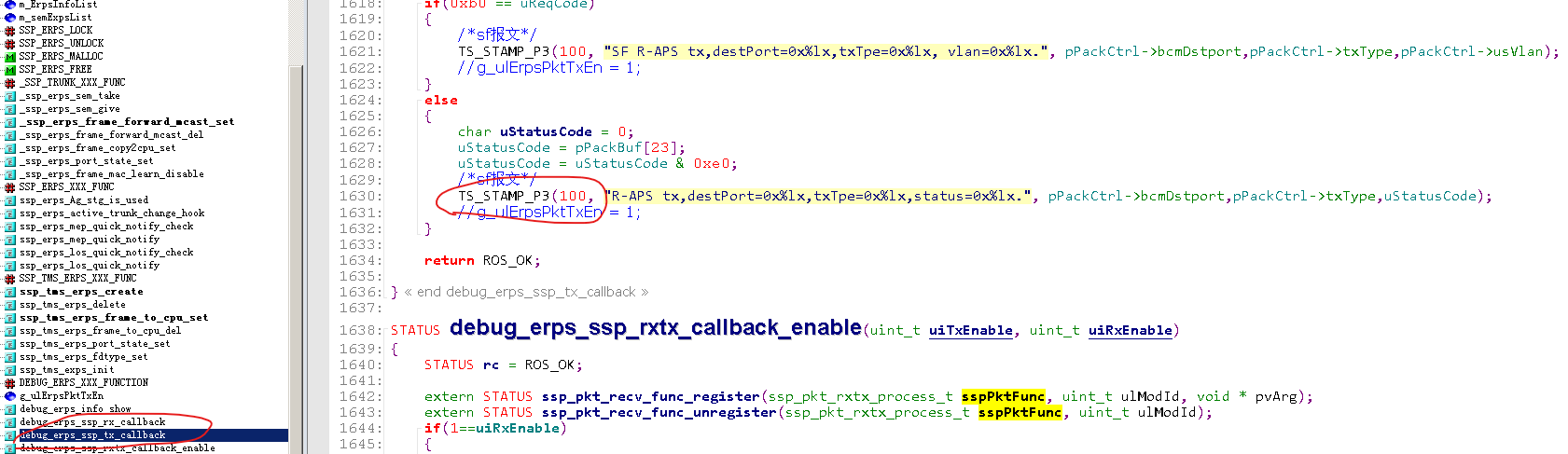
清除所有时间戳记录，或者只清除ritp记录的时间戳信息。一般在show命令显示太多，不方便查看需要的时间戳记录时先清除，等下一个时间戳记录。



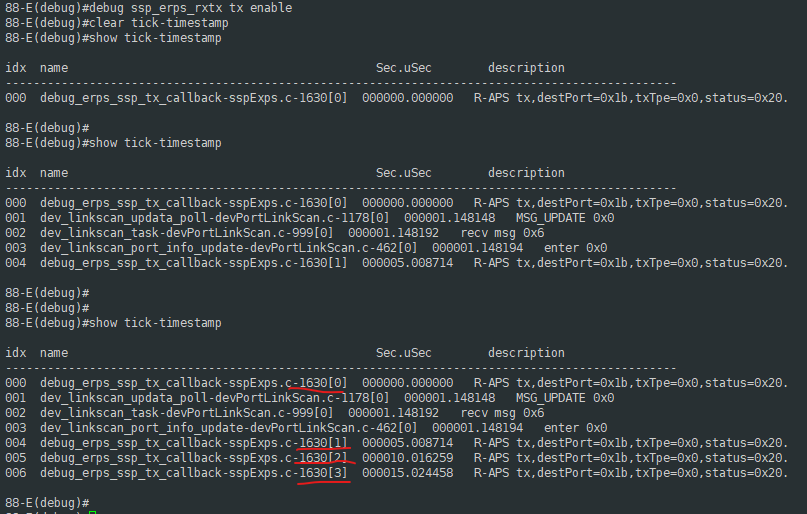
**协议报文收发打时间戳**

**debug ssp\_erps\_rxtx (rx | tx | both) (enable | disable)**

设置后在协议包收发处理时会打上处理该报文的时间戳，方便查看协议报文的收发是否正常。比如使能发包的时间戳记录：



查看时间戳记录，会看到在函数里添加的时间戳信息，协议规定的发包间隔5秒也都正确：



**ssp层时间戳命令**

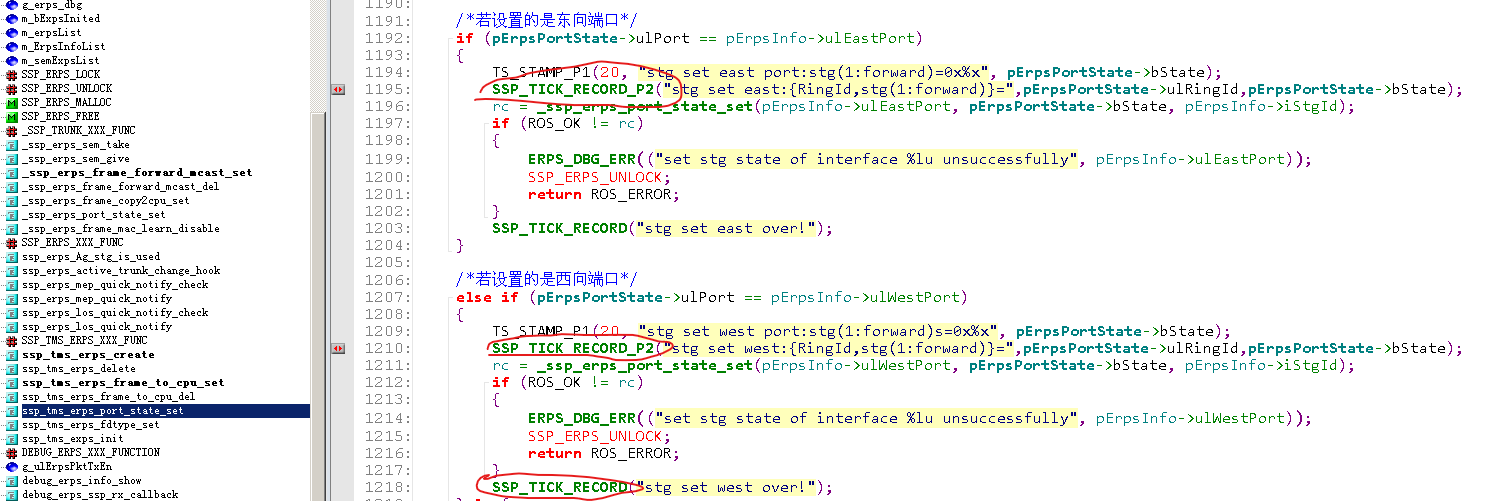
上面介绍的命令是平台提供的时间戳接口，ssp层也有相应的时间戳开关、记录和查看命令。

**debug ssp\_time\_tick (clear | begin | end | show [latest <1-1000>])**

同样的，要想给代码运行过程打上时间戳，需要先打开记录开关：

**debug ssp\_time\_tick begin**

比如，同样在erps设置端口状态加时间戳信息，配置erps实例或者保护倒换时记录时间信息：



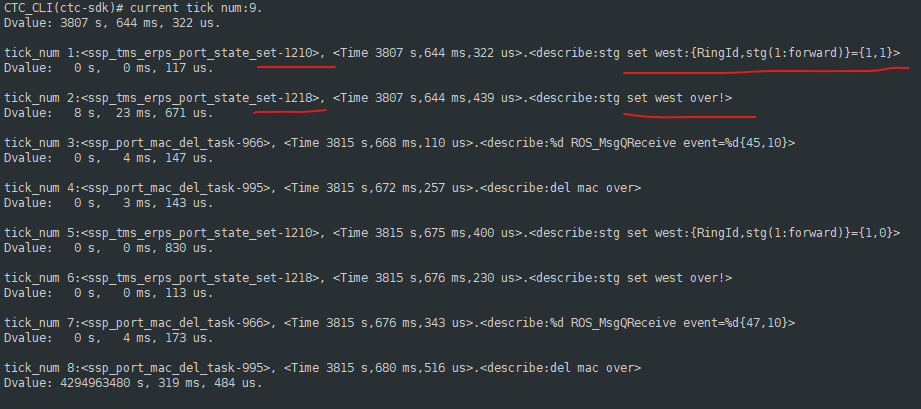
然后配置一个新的环实例

ethernet ring-protection 1 east gigaethernet 1/5/5 west gigaethernet 1/5/7 node-type rpl-owner rpl east block-vlanlist 1-10

再使用命令

**debug ssp\_time\_tick show**

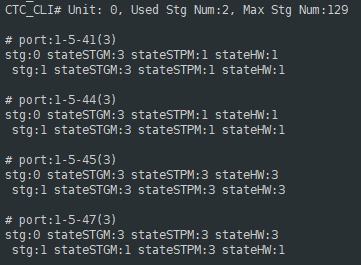
查看添加的时间戳信息：



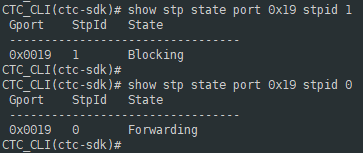
相应的函数名，所在行数，以及记录的参数内容都能看到。

**查看端口在stg中的状态**

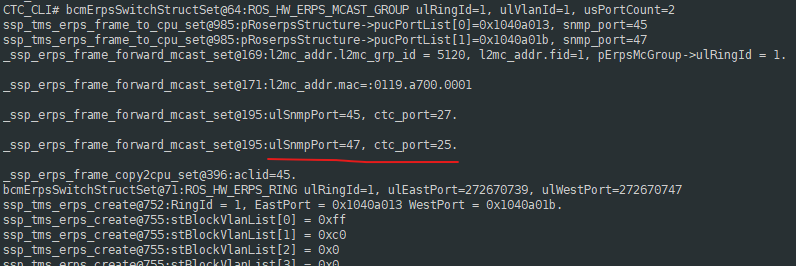
**debug ssp stg port status show**



显示的端口格式是unit-slot-snmp\_port，可以看到snmp\_port47在stg1中的状态是1，也就是block的，用sdk命令确认：



设备接口和sdk使用的接口号怎么对应的暂时没找到命令行，这里是通过配置环实例时的debug打印（debug ssp erps enable）看的：



**逻辑stgid与硬件stgid的使用关系**

**debug ssp stg mod info show**

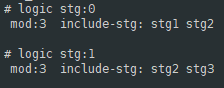


示例中逻辑stg是0，实际分配使用的硬件stg是1，最前面的mod表示这个stg的使用模块，3表示是erps模块在使用这个stg，如果是8，就是mstp模块。

再配置一个环实例：

ethernet ring-protection 2 east gigaethernet 1/5/6 west gigaethernet 1/5/8 node-type rpl-owner rpl east block-vlanlist 5-15

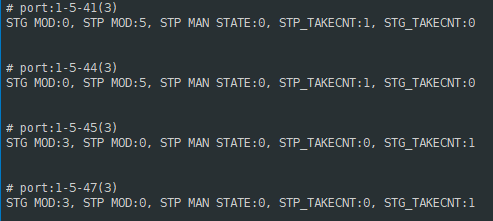
然后查看stgid的对应关系：



因为vlanlist有重叠，所以拆分了stgid。

**端口的stg使用情况**

**debug ssp stg mod port info show**



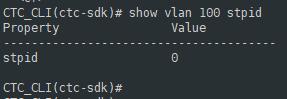
mod表示端口正在被这个模块使用，3表示erps模块，而且是以stg方式使用的。最后一列表示该端口被使用了一次。

**sdk相关**

**sdk相关命令**

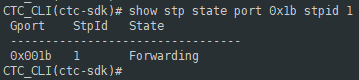
**查看vlan所在stp实例**

show vlan VLAN\_ID stpid



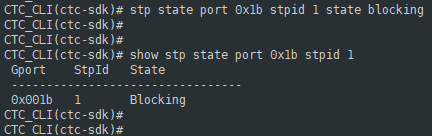
**查看接口stp状态**

show stp state port GPHYPORT\_ID stpid STP\_ID



**设置接口stp状态**

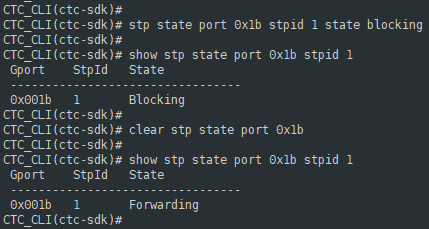
stp state port GPHYPORT\_ID stpid STP\_ID state (forwarding | blocking | learning)



设置接口在指定STP\_ID中的状态是forwarding，blocking或者learning。

**清除接口stp状态**

clear stp state port GPHYPORT\_ID



命令设置接口为默认forwarding状态。